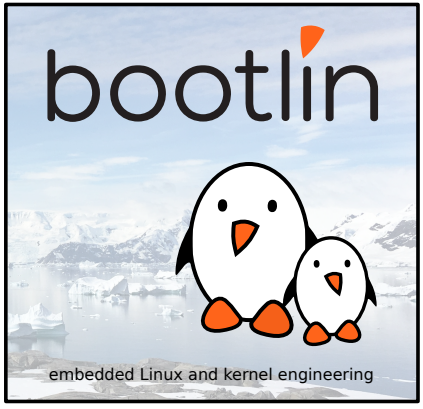
Representing the front-facing port of Ethernet interfaces

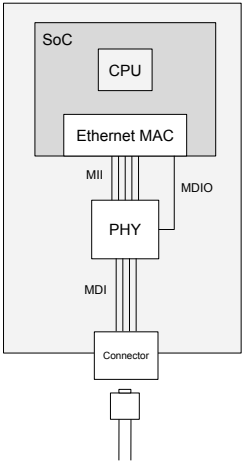
Maxime Chevallier
maxime.chevallier@bootlin.com



Maxime Chevallier

- Embedded Linux engineer at Bootlin
  - Embedded Linux **expertise**
  - **Development**, consulting and training
  - Strong open-source focus
- Open-source contributor
- Living near **Toulouse**, France

Typical embedded hardware design

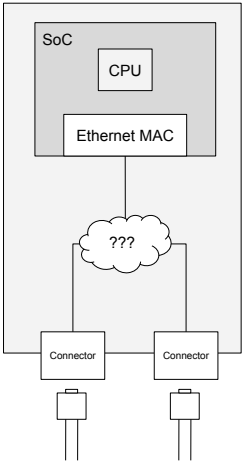


- `MII`: Media Independent Interface
  - SGMII, RGMII, RXAUI, etc.
- `MDI`: Media Dependent Interface
  - 1000BaseT4, 10BaseT1S, 10GBaseSR, etc.

Variants

- The PHY can be integrated in the SoC or MAC
- The PHY might not exist at all
- The PHY can be handled by a firmware
- The Port isn't always BaseT4
- The Port can be internal (backplane ethernet)

Mutli-port designs



- One interface, multiple front-facing ports
- SFP + Copper combo ports (MCBin for example)
- Also used for redundancy

What do we know about the port ?

PHY information

- `int phydev.port` : `PORT_FIBRE`, `PORT_TP`, etc. Active port.
- `phydev.supported`, `phydev.advertising` : Supported ports and linkmodes
- `ethtool_ksettings` : whole NIC (MAC + PHY) supported modes
- Works well, but only makes sense for a single port.

SFP/SFF

- *upstream* MAC or PHY knows precisely the SFP cage protocols
- When the module is inserted, we detect the real link modes

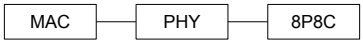
New representation

- `struct phy_port` (or `struct phy_mdi` ? `struct mdi` ?)
- Set of ops to get the status and support using `ethtool_ksettings`
  - `phy_port_ethtool_ksettings_[g|s]et`
- Set of ops to configure the port :
  - `enabled`
  - `preferred` ?
- Port "provider" implement these ops : PHY driver, phylink, MAC, SFP
- Helpers in `phylib` to make it as transparent as possible

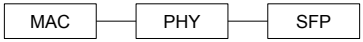
MDI information

- Populated by the PHY driver or NIC driver, based on what it knows.
  - Hardcoded, if only one possible mode
  - Reported by HW straps, eeproms or FW (sometimes incorrect)
  - Extracted from devicetree
- Device Tree current status : Not ideal
  - `ti,fiber-mode`, `micrel,fiber-mode`
  - `ti,op-mode` : Correlates `MII` and MDI
  - `max-speed` : (ab)used for 100M limitation (2 lanes wired instead of 4)
- We lack information about the presence or absence of the MDI

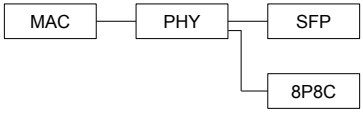
Combo SFP + Copper




```
ethernet-phy@0 {
    reg = <0>;
};
```




```
ethernet-phy@0 {
    reg = <0>;
    sfp = <&sfp0>;
};
```




```
ethernet-phy@0 {
    reg = <0>;
    sfp = <&sfp0>;
    // No indication about the 8P8C presence
};
```

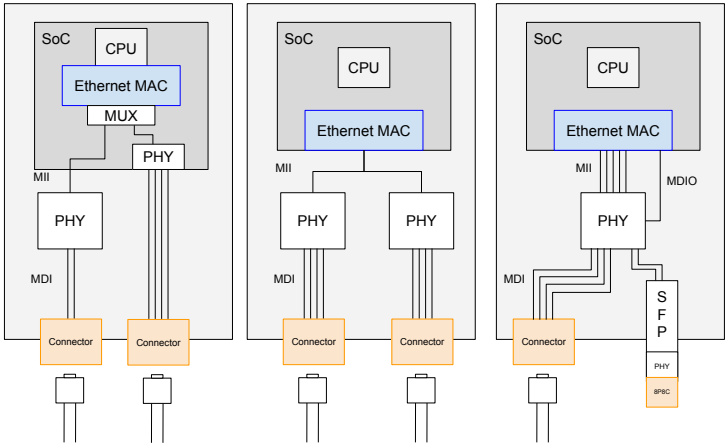
Devicetree example - WIP

example.dts

```
ethernet-phy@0 {
 ...

  mdi {
    port@0 {
      media = "10baseT", "100baseT", "1000baseT";
      lanes = <4>;

      /* Attach PSE ports to the actual connector */
      pses = <&pse_pi0>;
    };

    port@1 {
      sfp = <&sfp>;
      lanes = <1>;
    };
  };
};
```

- Only relevant when the current representation isn't enough

Port switching



User cares about which front-facing-ports are on a given interface

Link detection and commutation

- The port's parent must report how link detection happens.
- Case 1 : We can always independently detect and negociate the link
  - Dual-PHY with MII multiplexer
  - Support for preferred port
  - Support for highest-speed link
- Case 2 : While one port is actively used, we can't detect link on other ports
  - Fiber SFP + Copper Combo-PHY
  - Limited support for preferred port
- Case 3 : We can only detect link one port at a time, even if no port has link
  - Dual-PHY, no multiplexer, PHY has broken isolation
- Send a gratuitous ARP upon switching port
- Similarities with `bonding`, but with only one netdev

uAPI

- Prototype code using a new set of ethtool netlink messages
  - `ETHTOOL_A_PORT_GET`
  - `ETHTOOL_A_PORT_SET`
- Relies on setting port attributes :
  - `enabled` : Power the port link detection on/off
  - `forced` : Force this port to be used.
  - `preferred` : Prefer this port, if possible

Conclusion

Open questions :

- Naming : `mdi` ? `port` ? something else ?
- Device tree binding
- uAPI
- Integration with bonding ?

Ongoing work :

- Support for multiple PHYs started :`phy_link_topology`
- Support for PHY isolation submitted
- Next : Port representation and multiplexing
- PSE integration, with Köry Maincent

Questions? Suggestions? Comments?

Maxime Chevallier
maxime.chevallier@bootlin.com

https://bootlin.com/pub/conferences/2024/lpc/multi-port.pdf