Ethernet Switch Framework

Fully utilize your Wifi router

Stefan Bethke

BSDCan 2012

The Power to Serve...

- ... in a $30 box
- with Ethernet and Wifi
- with USB

Why?

- Built-in firmware is limited
  - Configuration management
  - Remote access
  - Special applications

Why FreeBSD?

- OpenWrt, DD-WRT, etc.
- Great projects, but not BSD

So what’s missing?

- Adrian Chadd did the heavy lifting for QualcommAtheros HW
- Drivers for Ethernet switch, some wireless HW
- Shrinking FreeBSD to 8MB or even 4MB image
- Configuration mechanisms
- Flash file system

Architecture & Design

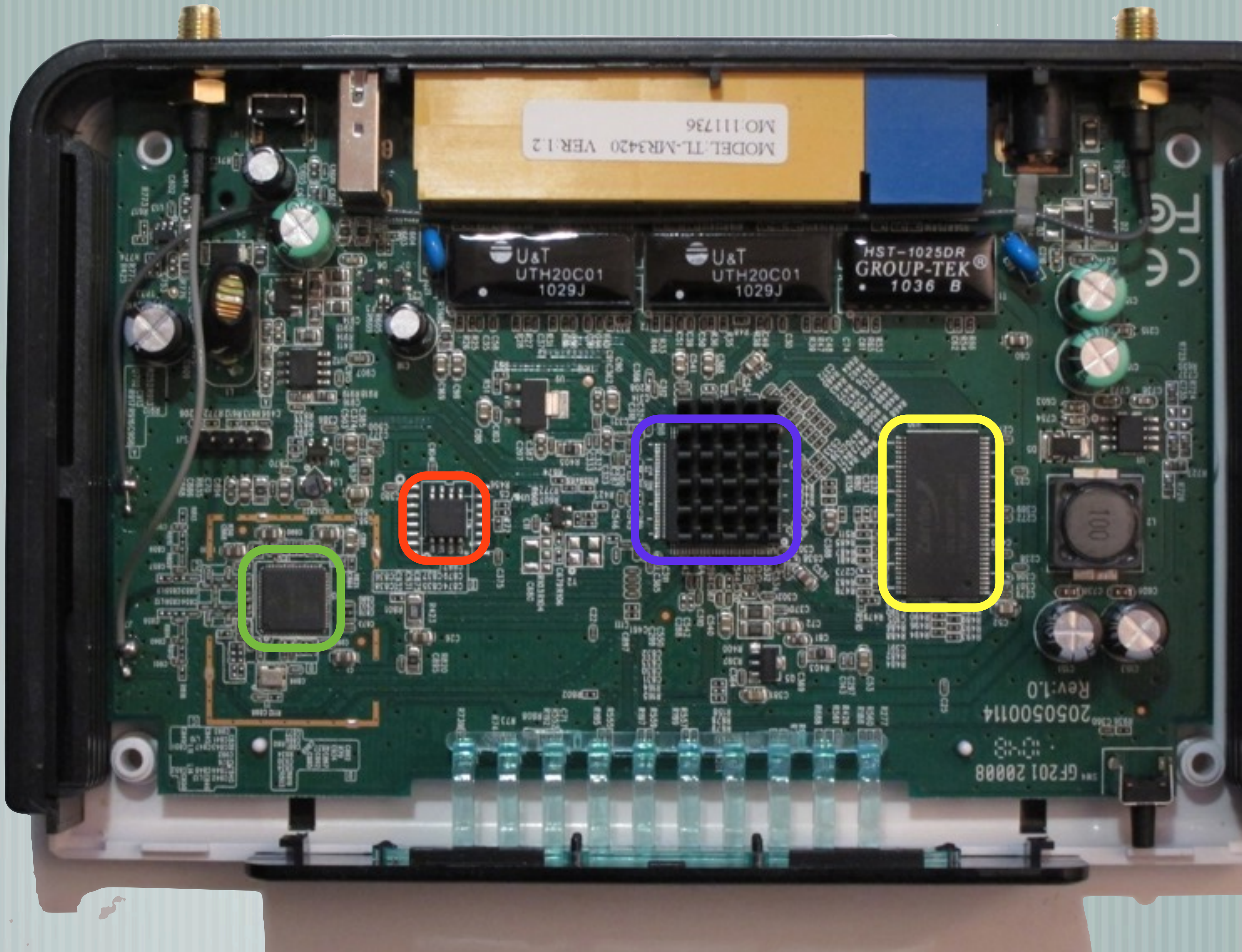
- Wifi Router Hardware
- Framework Architecture
- Configuration Interface
- Further Work

Hardware

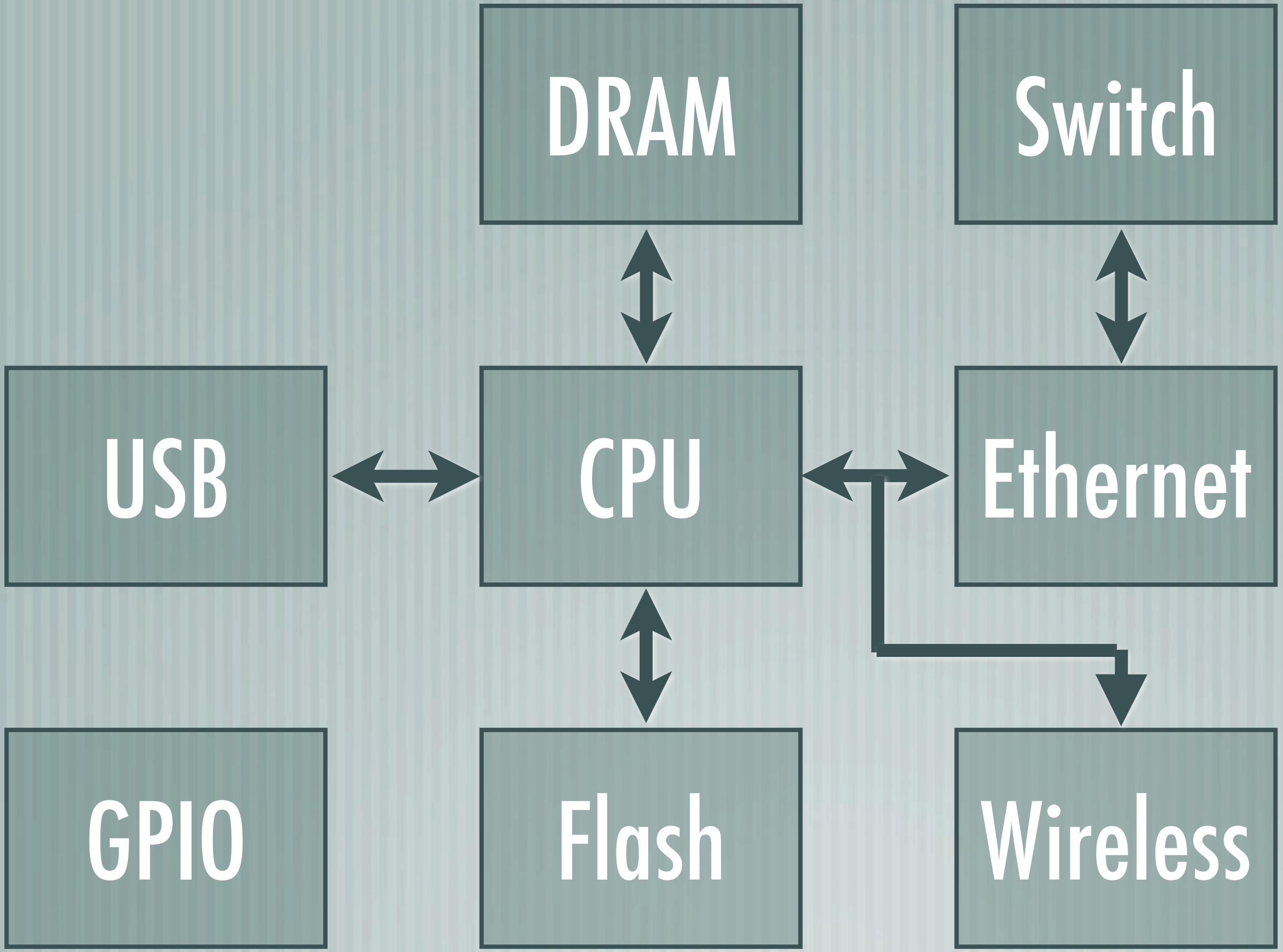
- What‘s in a box?
- System Components

What's in a box?

- ● SoC
- ● SPI ROM
- ● RAM
- ● Radio



System Components



Typical Busses

SPI: Flash

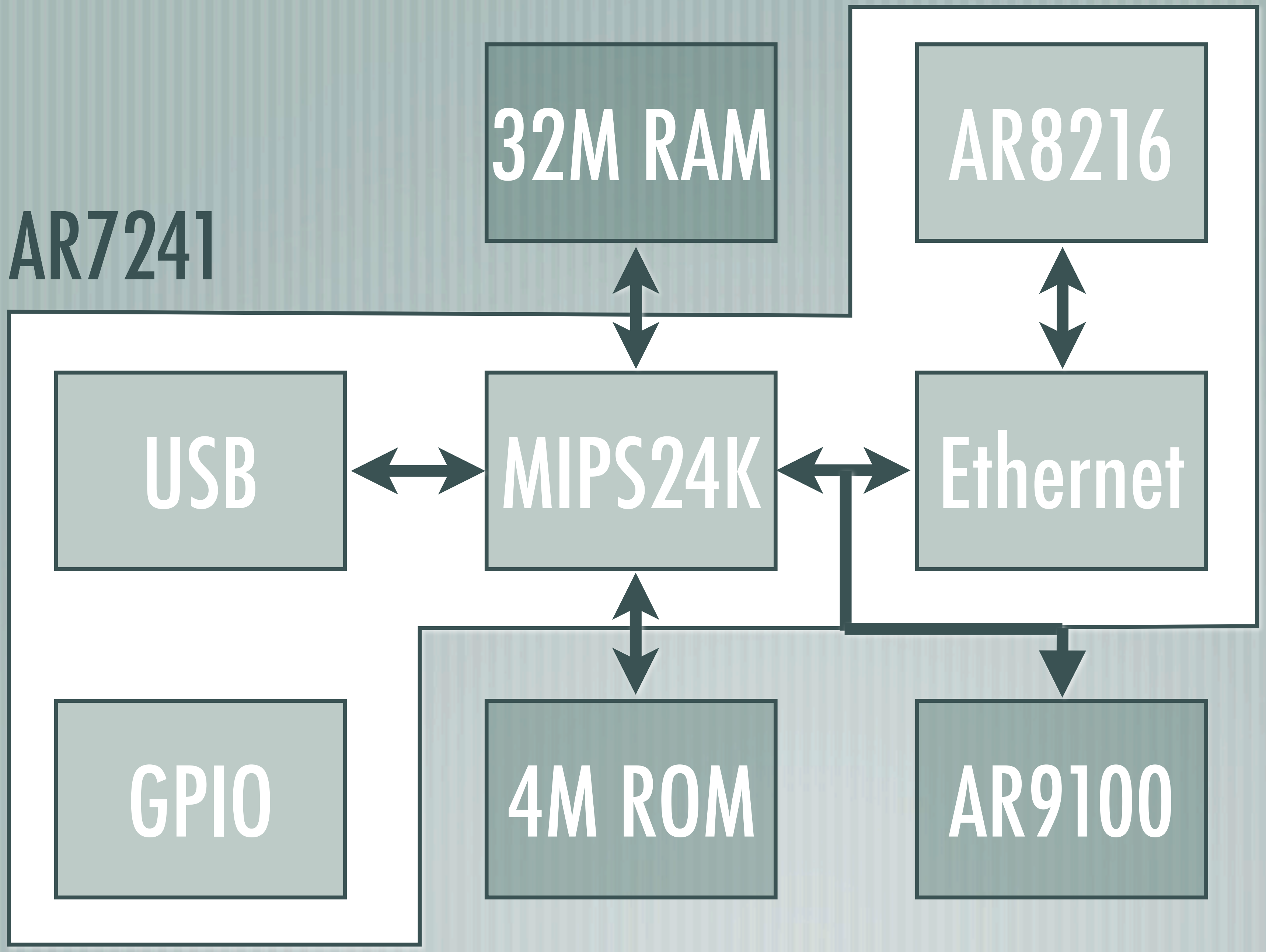
PCI: Wireless

MII: Switch, PHY

I^{2}C: Switch

Various platform-specific ones

TL-MR3420

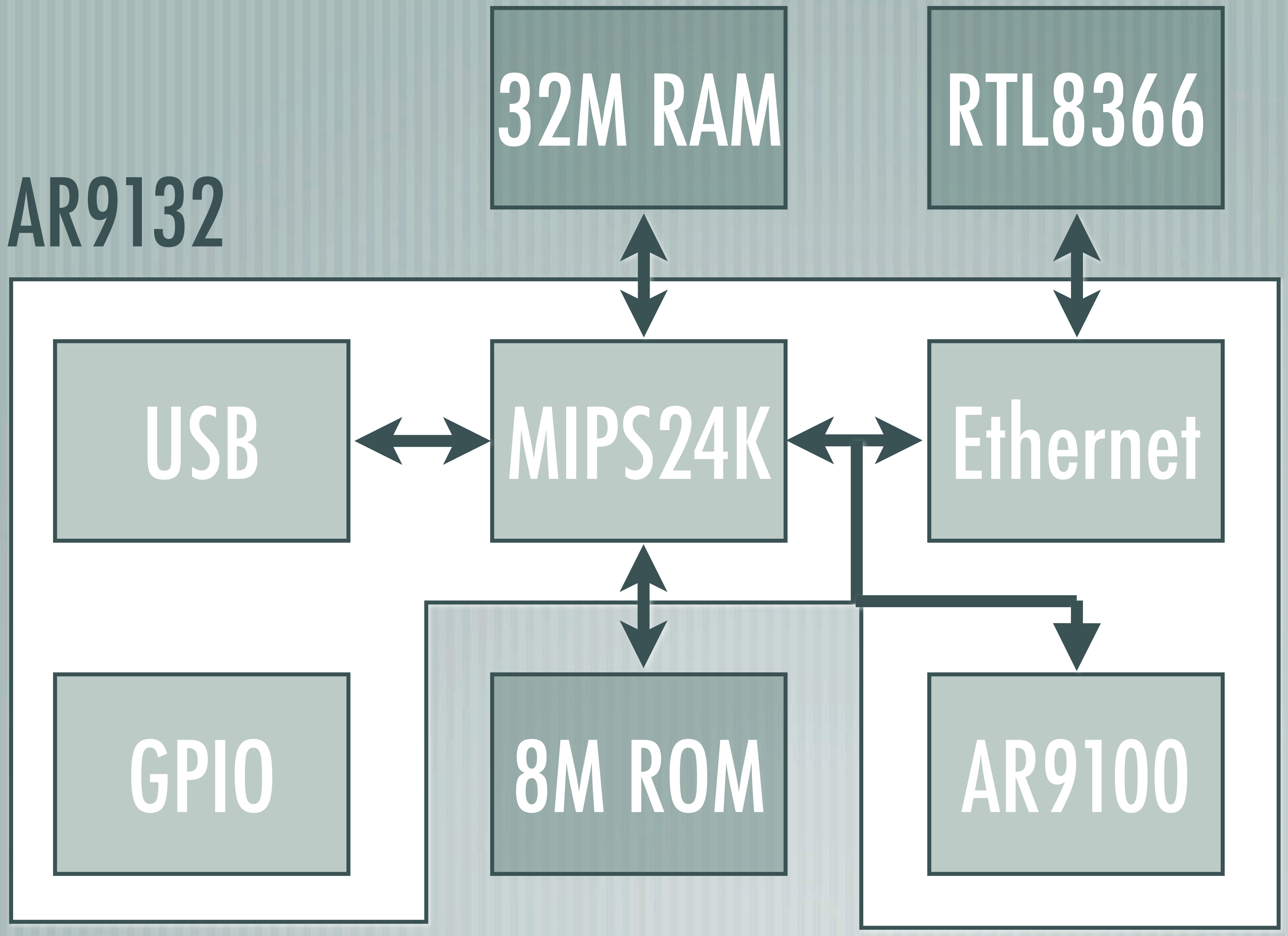


AR7241
32M RAM
AR8216
USB
MIPS24K
Ethernet
GPIO
4M ROM
AR9100

TL-MR3420

- 5 100-BaseT ports, 802.11n Wifi, USB 2.0
- Integrated Switch Controller
  - Controlled via MDIO interface
  - .1q VLAN tagging, priority
- 1 WAN Ethernet, 2nd Ethernet connected to switch (4 ports)

TL-WR1043ND


32M RAM
RTL8366
AR9132
USB
MIPS24K
Ethernet
GPIO
8M ROM
AR9100

TL-WR1043ND

- 5 1000-BaseT ports, 802.11n Wifi, USB 2.0
- Realtek RTL8366RB Gigabit Switch Chip
    - Controlled via I^{2}C-like interface, connected to CPU GPIO
    - .1q VLAN tagging, priority
- Only one Ethernet interface, needs VLAN configuration for LAN/WAN split

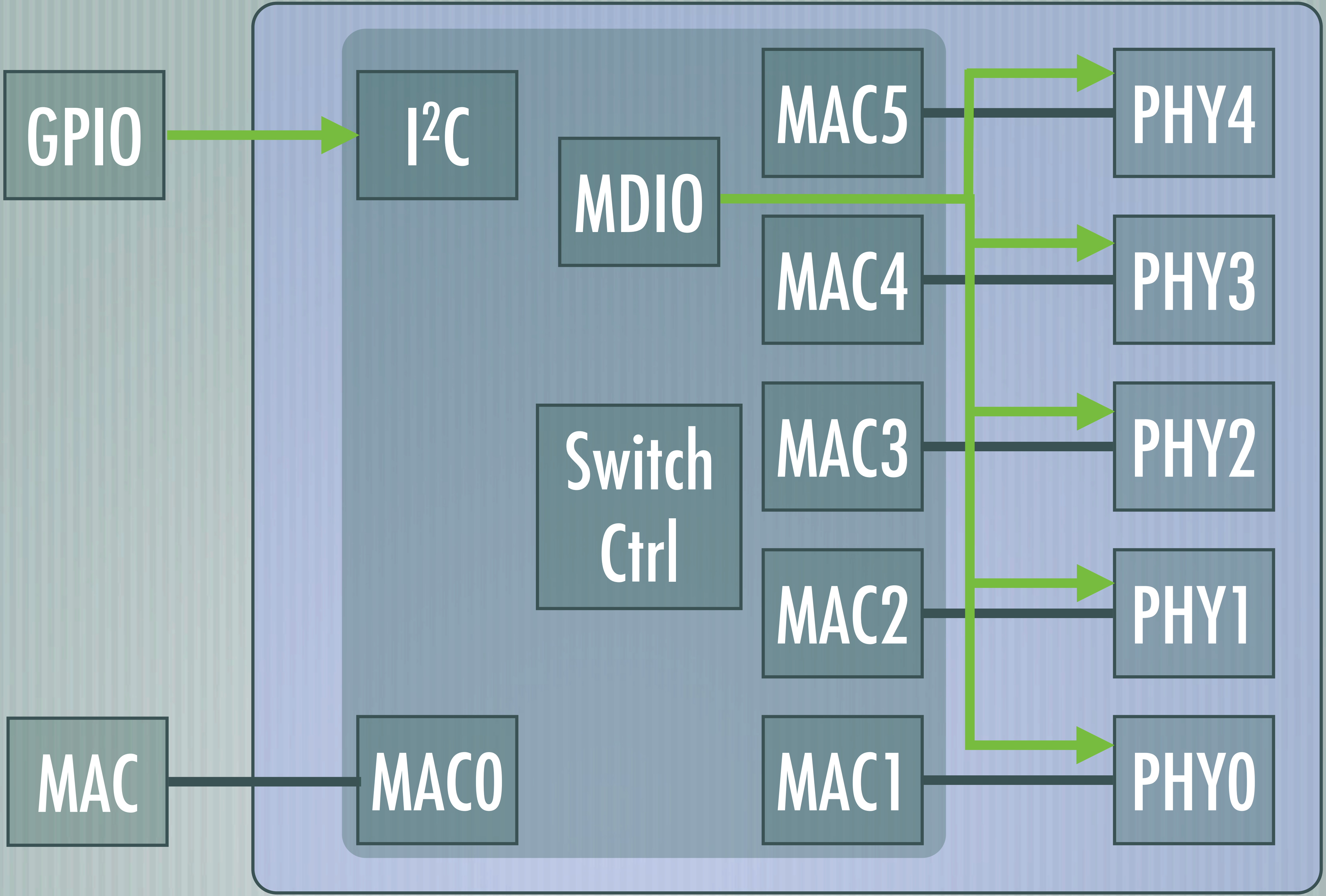
Architecture & Design

- Wifi Router Hardware
- Framework Architecture
- Configuration Interface
- Further Work

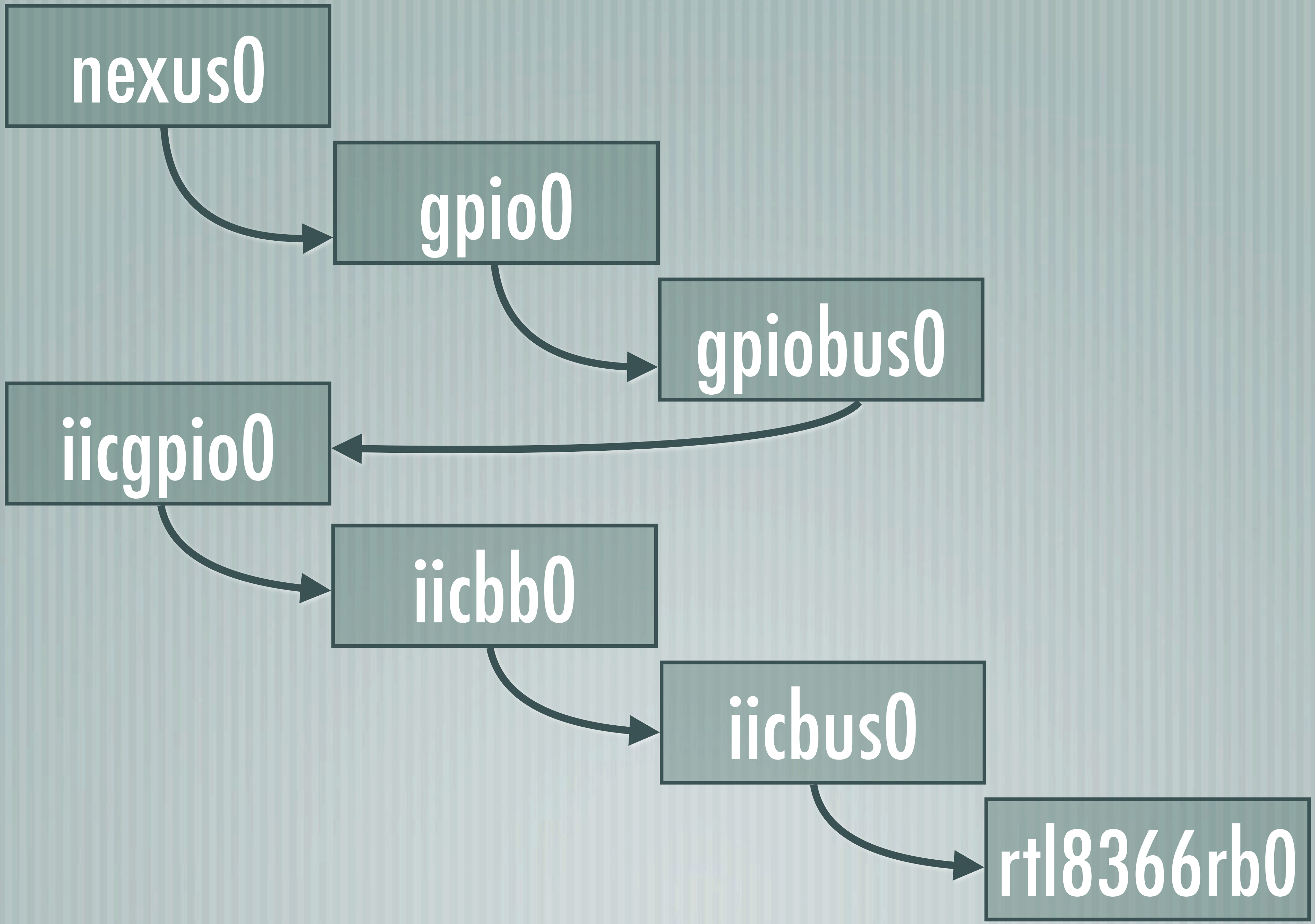
Framework Architecture

- Hardware-specific drivers for each chip (family)
- Generic kernel API for configuration, management
- IOCTL interface for userland via generic driver
- PHY management via miibus(4)

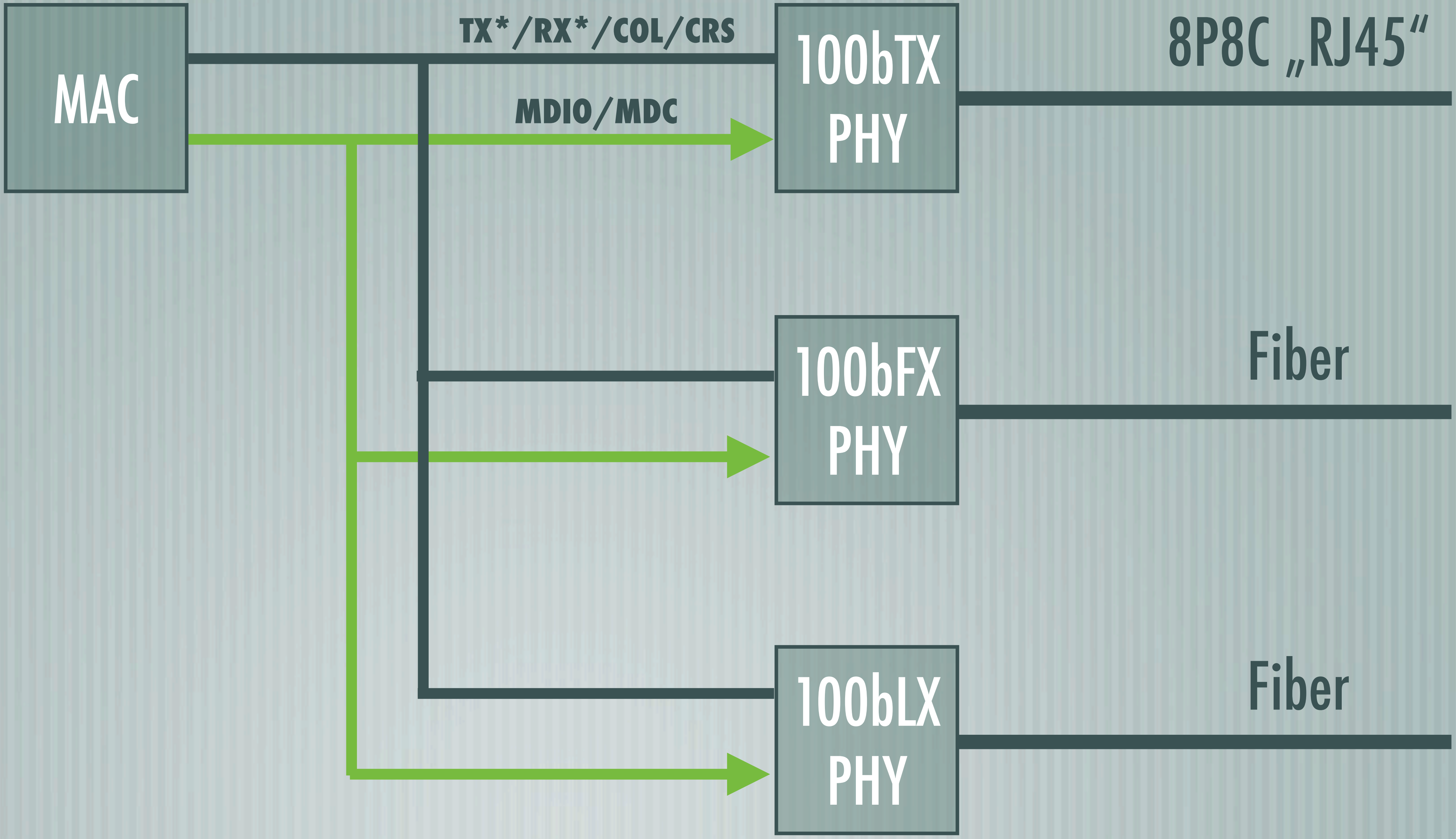
RTL8366RB in TL-WR1043



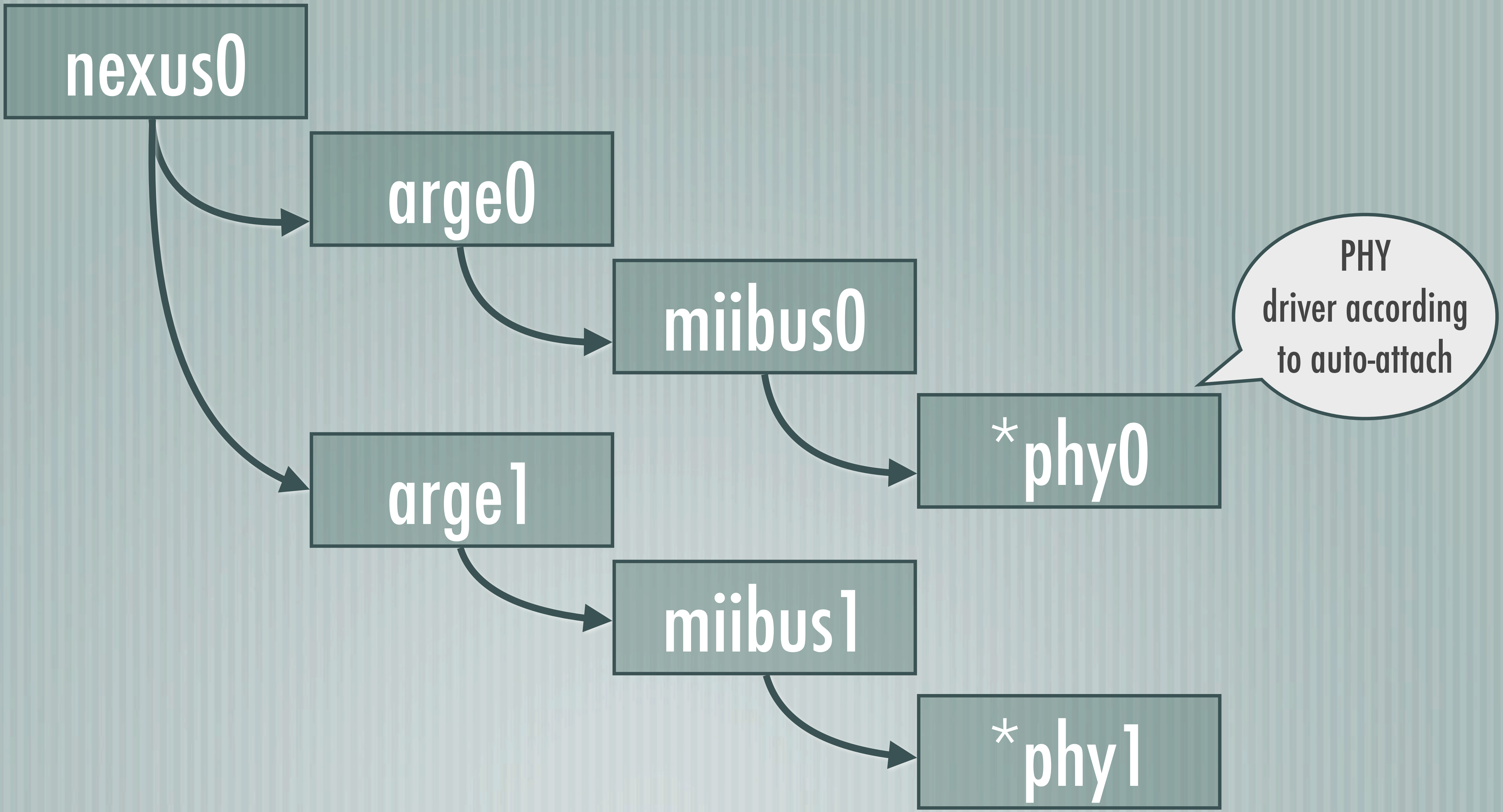
Device Tree – TL-WR1043



802.3 MII Model



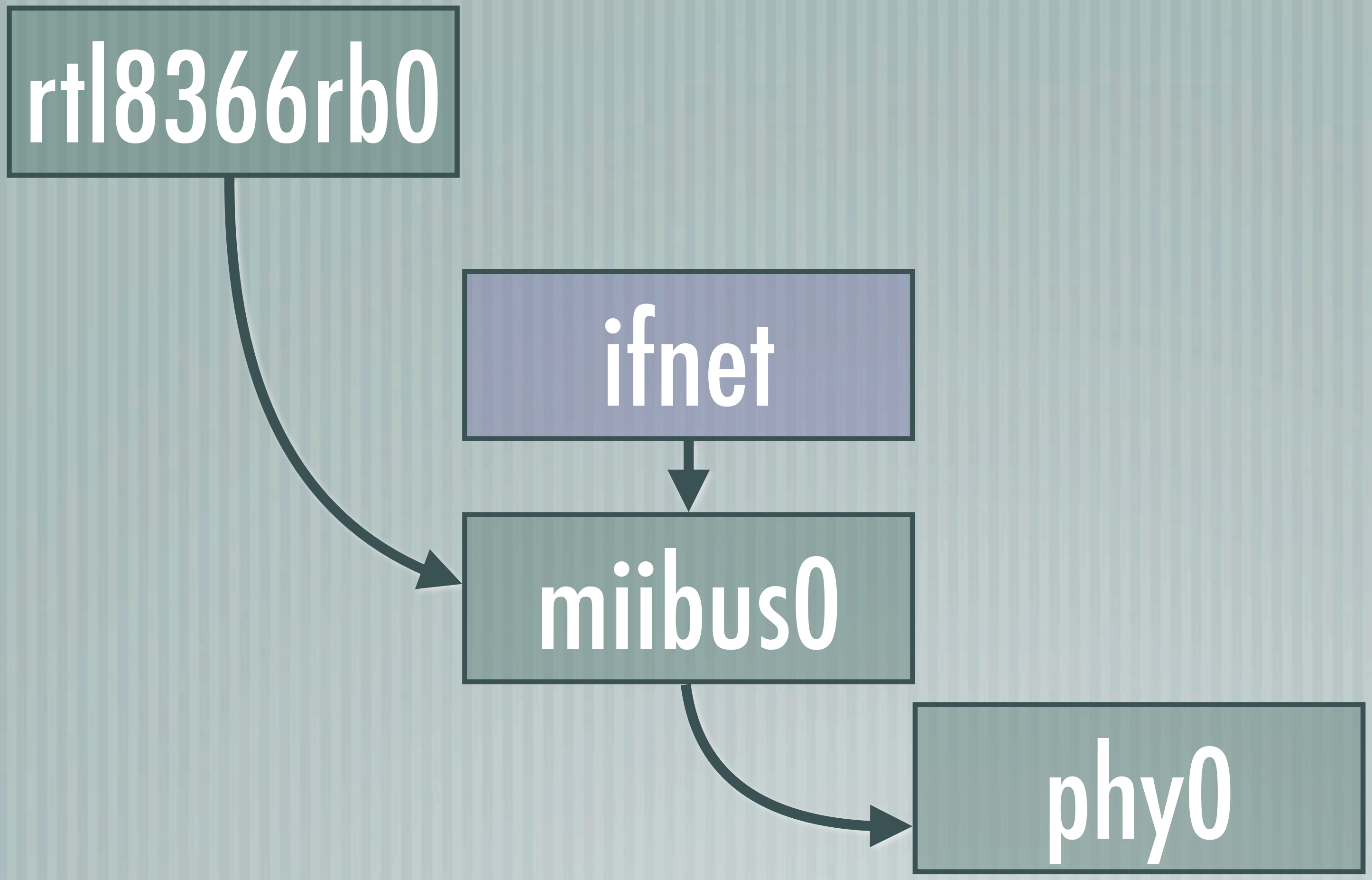
Device Tree—2 Ethernets



miibus(4) API

- miibus_if.m methods
  - MDIO access: readreg, writereg
  - MAC configuration: linkchg, statchg, mediainit
- if_media.h callbacks
  - MAC configuration: change, status
  - mii_attach uses both device_t and ifnet

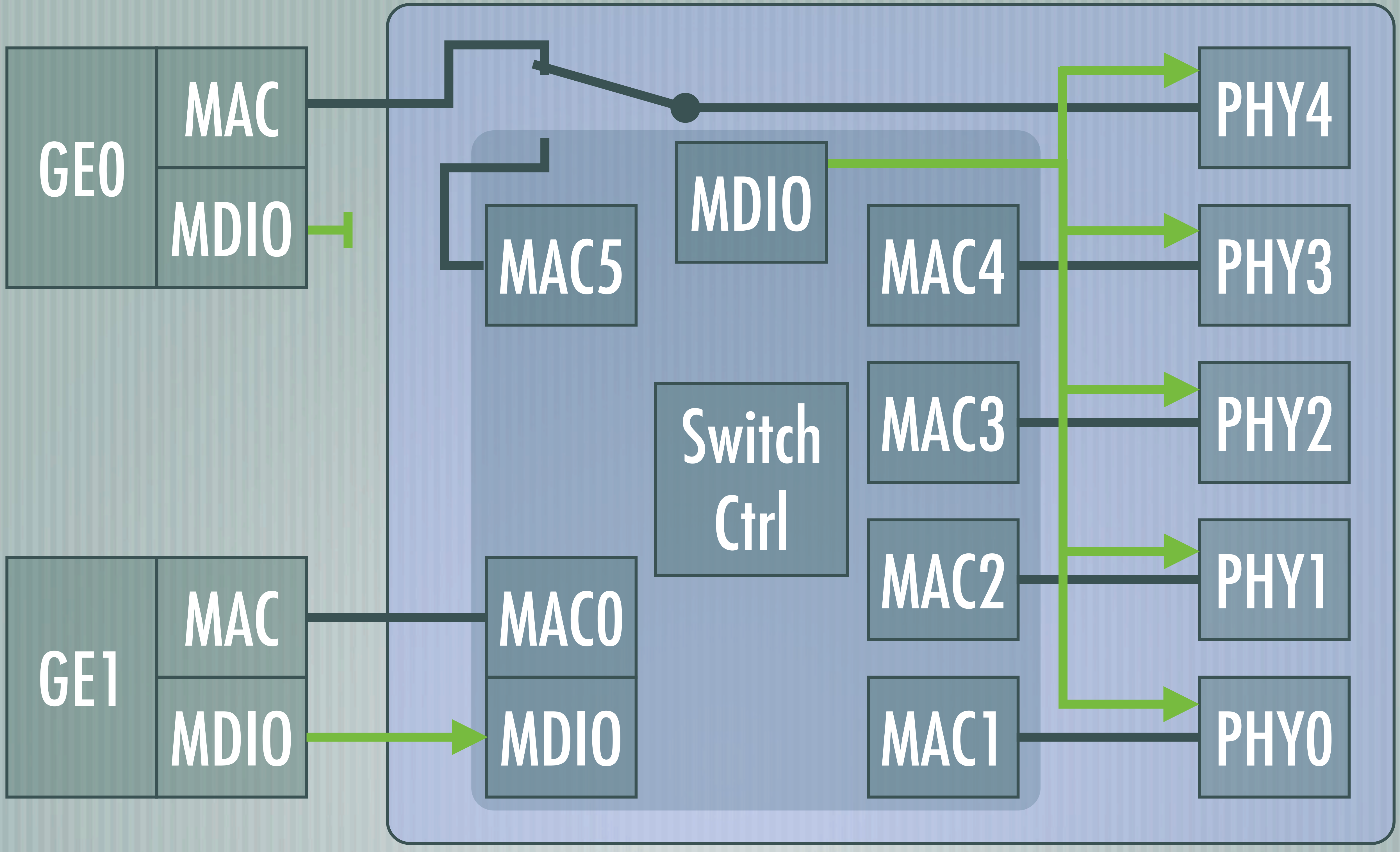
Port PHYs – TL-WR1043



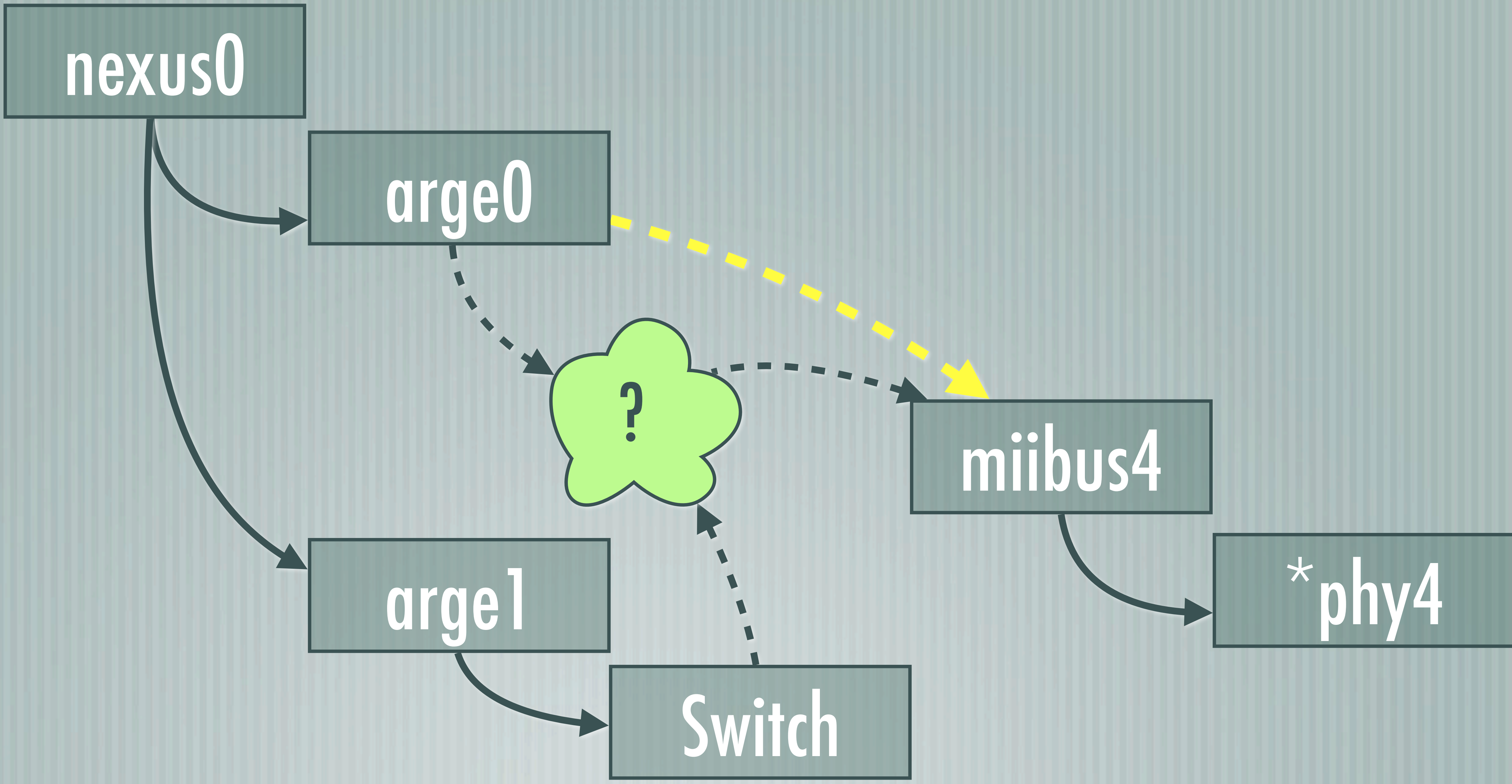
Switch Controllers on MDIO

- Connected to the CPU via MDC/MDIO lines
- Some look like PHYs with additional registers
- Some have completely different register model
- miibus(4) not really prepared to deal with this

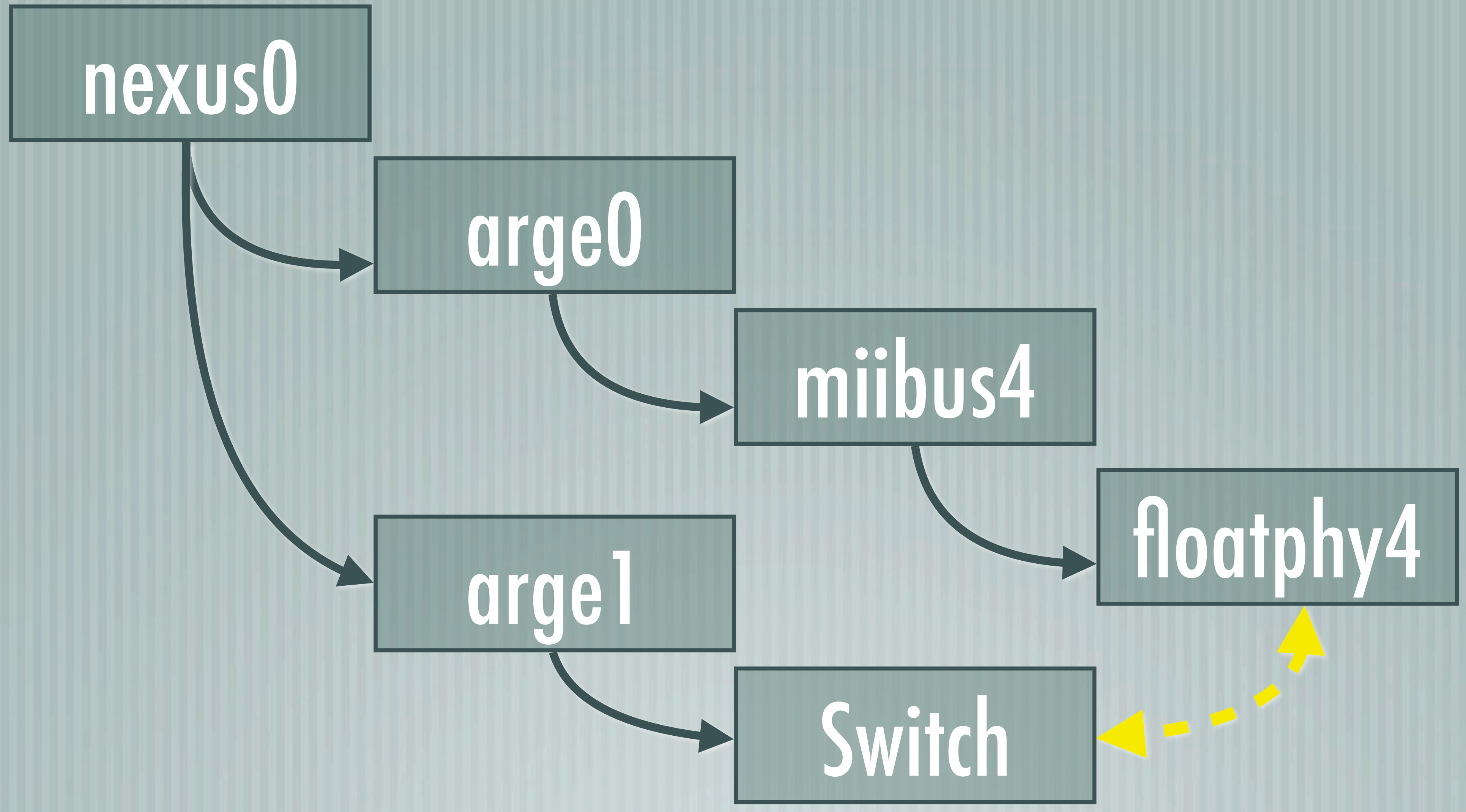
AR7241 Switch



Device Tree—Switch/PHY



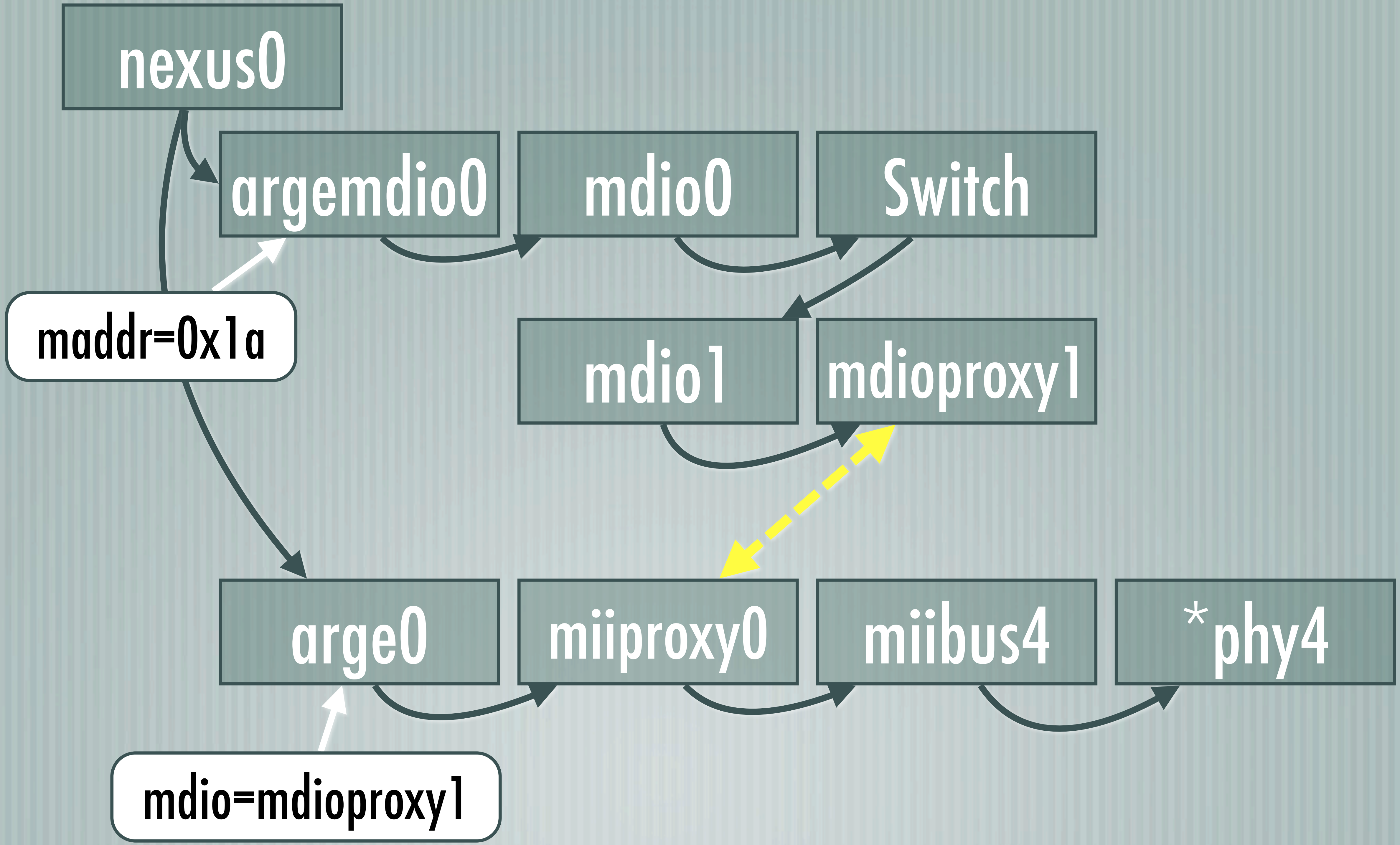
Device Tree—floatphy



floatphy

- Presents as a PHY driver attached via hint
- Funnels MDIO access through hidden channel to switch driver
- Replaces existing PHY drivers

Device Tree—miiproxy



MDIO/MII Proxy

- Separates MDIO access from MAC configuration
- Provides attachements to both MDIO and Ethernet driver
- Fully transparent to miibus(4) and PHY drivers

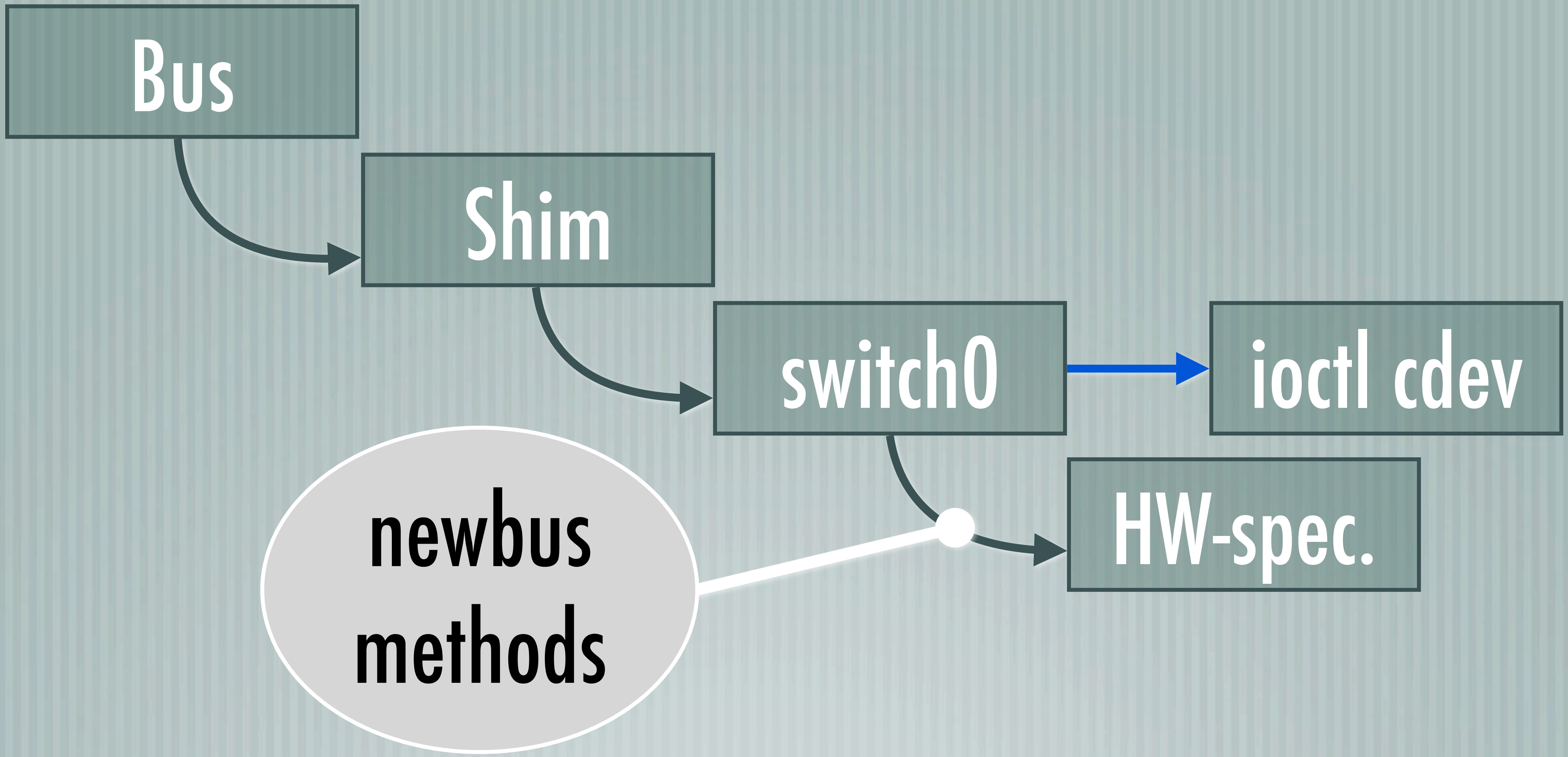
Switch Driver Attachment

- Generic “switch bus” abstraction
- Standard newbus APIs

Switch Driver Attachment

- Generic “switch bus” abstraction
  - Bus-specific driver shim attaches to bus
  - Generic code provides external API & register abstraction
  - Switch driver attaches to generic driver

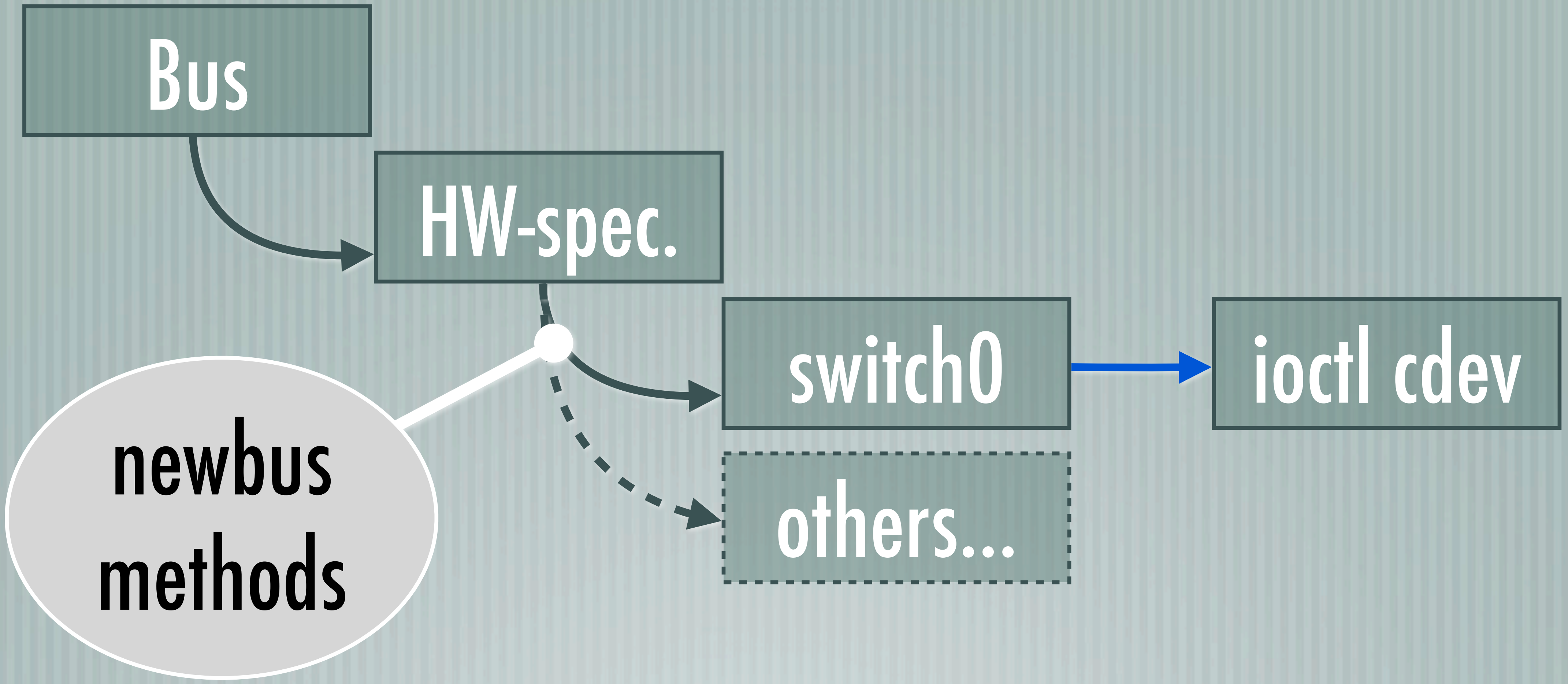
Device Tree—Switch Bus



Switch Driver Attachment

- Standard newbus, bus_space APIs
  - Hardware-specific switch driver attaches to bus
  - Provides generic API through newbus methods
  - IOCTL driver attaches to HW-specific driver
  - Additional drivers can attach to in-kernel API
  - Auto-attaching

Device Tree—Std. Newbus



Architecture & Design

- Wifi Router Hardware
- Framework Architecture
- Configuration Interface
- Further Work

Abstract Switch

- Switches vary considerably, esp. in advanced features
- Base feature set comparable
  - PHYs on ports
  - 16 VLAN entries
  - MAC table management

Must-have Features

- Initialization
- Register Peek and Poke
- Capability API
- Port-based and .1q VLANs

VLAN Managment

- Port-based and .1q VLANs are mutually exclusive
- Ports are either trunked (.1q tagged) or untagged
- Ports have a default VLAN ID
- VLAN entries have a VLAN ID, member port list

Architecture & Design

- Wifi Router Hardware
- Framework Architecture
- Configuration Interface
- Further Work

To-Do

- Finalize open questions (attachment, miibus, API) ✓
- Commit base version ✓
- Update existing drivers
- Add additional drivers for common hardware

The Future

- Advanced switch features
  - .1Q Priority Queues
  - Forwarding table management, Packet Filtering
  - NAT
- Spanning Tree
- .1X Port Security

People & Links

- Adrian Chadd adrian@freebsd.org
- Aleksandr Rybalko ray@freebsd.org
- Stefan Bethke stb@freebsd.org
- wiki.freebsd.org/StefanBethke/EtherSwitch
- zrouter.org