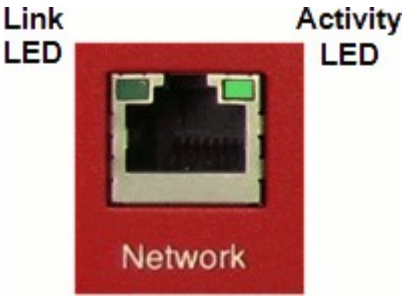
Blinking Lights[1]: Getting it wrong again, again and again



Andrew Lunn
<andrew@lunn.ch>

[1] These slide contain animated gifs. Use the .odp file, not .pdf

Problem: Configure LED controller

0000 = On - Link, Off - No Link
0001 = On - Link, Blink - Activity, Off - No Link
0010 = On- Full Duplex, Blink- Collision, Off- Half Duplex
0011 = On - Activity, Off - No Activity
0100 = Blink - Activity, Off - No Activity
0101 = On - Transmit, Off - No Transmit
….

0	Link/Activity
1	Link1000/Activity
2	Link100/Activity
3	Link10/Activity
4	Link100/1000/Activity
5	Link10/1000/Activity
6	Link10/100/Activity

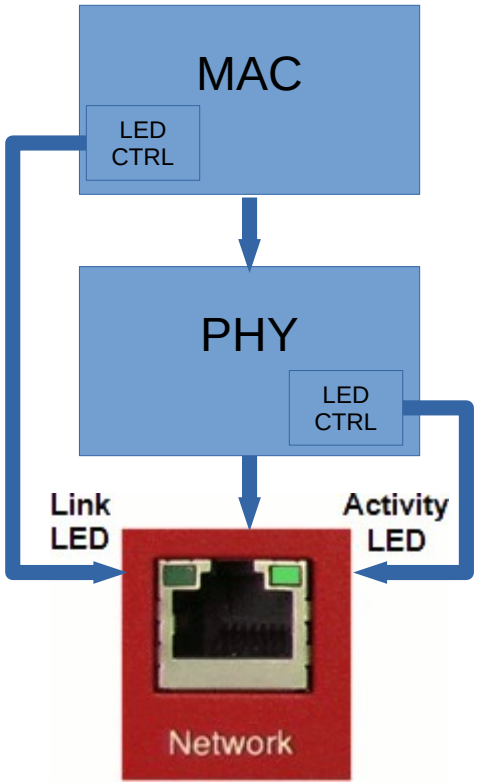
….

00 Link up

01 Frame reception

10 Symbol Error

11 CRS signal



Bad existing solutions

```
Fri Nov 19 12:13:18 2010 +0000
            marvell,reg-init =
                    /* irq, blink-activity, blink-link */
                    <3 0x10 0 0x0242>; /* Reg 3,16 <- 0x0242 */
```

The Marvell PHYs have a page select register at register 22 (0x16), we can specify any register by its page and register number. These are the first and second word. The third word contains a mask to be ANDed with the existing register value, and the fourth word is ORed with the result to yield the new register value.

Bad existing solutions...

```
2018-09-03 10:48:53 +0200

- vsc8531,led-[N]-mode  : LED mode. Specify how the LED[N] should behave.
                          N depends on the number of LEDs supported by a PHY.
                          Allowed values are defined in
                          "include/dt-bindings/net/mscc-phy-vsc8531.h".
                          Default values are VSC8531_LINK_1000_ACTIVITY (1),
                          VSC8531_LINK_100_ACTIVITY (2),
                          VSC8531_LINK_ACTIVITY (0) and
                          VSC8531_DUPLEX_COLLISION (8).
```

Why is it wrong?

- No consistency across PHYs. But it is a common problem across PHYs and MACs
- Device tree describes hardware, not configuration
- The user wants to decide, not the DT writer

A Better Way To Do This?

```
Wed Jul 03 2019: Florian Fainelli
> + A 0..3 element vector, with each element configuring the operating
> + mode of an LED. Omitted LEDs are turned off. Allowed values are
> + defined in "include/dt-bindings/net/realtek.h".

This should probably be made more general and we should define LED modes
that makes sense regardless of the PHY device, introduce a set of
generic functions for validating and then add new function pointer for
setting the LED configuration to the PHY driver. This would allow to be
more future proof where each PHY driver could expose standard LEDs class
devices to user-space, and it would also allow facilities like: ethtool
-p to plug into that.
```

Again, again, and again

2022-11-18: add dt configuration for dp83867 led modes

- Sorry, but NACK.

2022-09-22: net: phy: mxl-gpy: Add mode for 2 leds

- We have NACKed patches like this for a few years now

2020-08-22: net: phy: dp83867: apply ti,led-function and ti, led-ctrl to registers

- Sorry, but NACK.

2021-10-01: net: phy: mscc: Add possibilty to disable combined LED mode

- Sorry, but no DT property.

2021-08-09: net:phy:dp83867:implement the binding for status led

- Private properties for status LEDs are no longer permitted for new code.

Learning #1

- Search to see if somebody else has already been NACKed for the same idea!
- “Those who do not learn history are doomed to repeat it.”
  - Most likely Philosopher George Santayana

Learning #2

- Solve common problems for every driver, not your driver
- There is too much silo thinking in netdev
  - Patches, reviews, reading the netdev list
- You can learn a lot from other drivers, if you make the effort
- Page Pool is a good example.

Getting is wrong again, again, and again...

- Everybody gets Pause wrong.
- Everybody gets Energy Efficient Ethernet wrong.
- Why?
  - API is poorly designed and documented
  - No core support. Drivers do everything
- Phylink core now does most of Pause
- EEE being rewritten moving ~80% code into core

Learning #3

- Make core do as much as possible
- Driver code KISS, just configure the hardware
- Rusty Russell ‘How Do I Make This API Hard to Misuse?’ metric
  - 10. It's impossible to get wrong.
  - 9. The compiler/linker won't let you get it wrong.
  - 8. The compiler will warn if you get it wrong.
  - 7. The obvious use is (probably) the correct one.
  - ...
  - 2. Read the implementation and you'll get it right.
  - 1. Read the correct mailing list thread and you'll get it right.

https://ozlabs.org/~rusty/index.cgi/tech/2008-03-30.html

Blinking Lights: Getting it Right

Christian Marangi did most of the implementation work.

Blinking Lights: Getting it Right

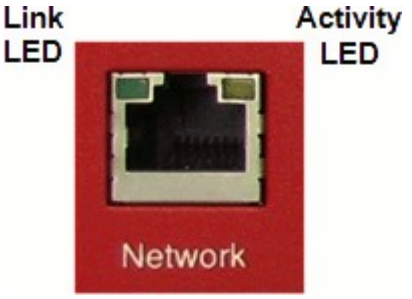
- Offload to hardware what Linux can already do in Software

What Can Linux do with LEDs ?

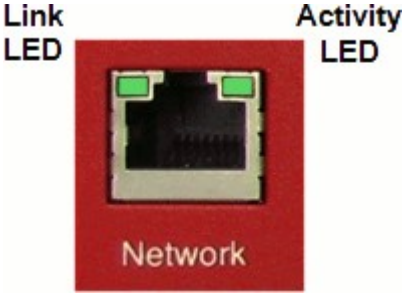
```
$ ls /sys/class/leds/

input0::capslock      input0::scrolllock    platform::mute
tpacpi::lid_logo_dot  tpacpi::standby       input0::numlock
platform::micmute     tpacpi::kbd_backlight tpacpi::power
tpacpi::thinkvantage

$ ls /sys/class/leds/input0\:\:numlock

brightness  device  max_brightness  power  subsystem  trigger  uevent
```

Brightness

```
echo 0 > brightness
```




```
echo 1 > brightness
```



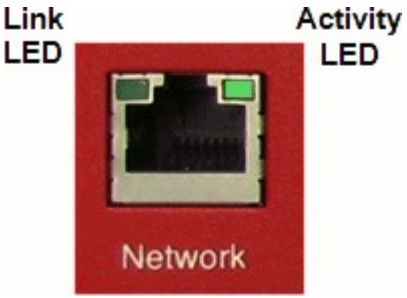

```
$ cat /sys/class/leds/tpacpi\:\:kbd_backlight/max_brightness
2
```

Triggers – kernel controlling the LED

```
cat /sys/class/leds/tpacpi\:\:kbd_backlight/trigger

[none] kbd-scrolllock kbd-numlock kbd-capslock kbd-
kanalock kbd-shiftlock kbd-altgrlock kbd-ctrllock kbd-
altlock kbd-shiftllock kbd-shiftrlock kbd-ctrlllock kbd-
ctrlrlock disk-activity disk-read disk-write mtd nand-
disk cpu cpu0 cpu1 cpu2 cpu3 cpu4 cpu5 cpu6 cpu7 panic
BAT0-charging-or-full BAT0-charging BAT0-full BAT0-
charging-blink-full-solid usb-gadget usb-host rc-
feedback AC-online rfkill-any rfkill-none audio-mute
audio-micmute rfkill0 bluetooth-power hci0-power rfkill1
phy0rx phy0tx phy0assoc phy0radio rfkill2 heartbeat
```

Heartbeat trigger

```
echo heartbeat > \
/sys/class/leds/input0\:\:numlock/trigger
```

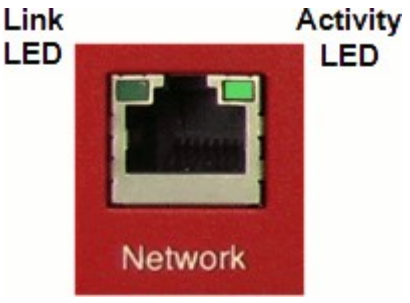


- How fast it blinks depends on CPU load!

Netdev trigger

```
echo netdev > /sys/class/leds/input0\:\:numlock/trigger
ls /sys/class/leds/input0\:\:numlock/
brightness   half_duplex link_100       power    tx
device       interval    link_1000      rx       uevent
device_name  link        max_brightness subsystem
full_duplex  link_10     offloaded      trigger

echo enp2s0 > device_name
echo 1 > rx
echo 1 > tx
```

Polls netdev stats every 50ms to decide it to blink LED.



Offloading to Hardware

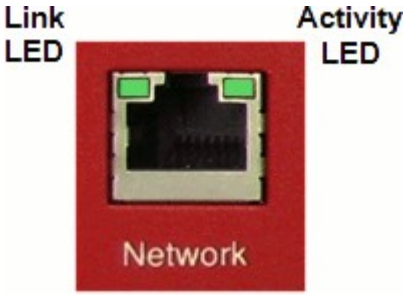
```
int (*brightness_set_blocking)(struct led_classdev *led_cdev,
                               enum led_brightness brightness);
/*
 * Check if the LED driver supports the requested mode provided by the
 * defined supported trigger to setup the LED to hw control mode.
 */
int (*hw_control_is_supported)(struct led_classdev *led_cdev,
                               unsigned long flags);
/*
 * Activate hardware control, LED driver will use the provided flags
 * from the supported trigger and setup the LED to be driven by hardware
 * following the requested mode from the trigger flags.
int (*hw_control_set)(struct led_classdev *led_cdev,
                      unsigned long flags);
/*
 * Get the device this LED blinks in response to.
 * e.g. for a PHY LED, it is the network device. If the LED is
 * not yet associated to a device, return NULL.
 */
struct device *(*hw_control_get_device)(struct led_classdev *led_cdev);
```

`flags` bitmap

```
enum led_trigger_netdev_modes {
        TRIGGER_NETDEV_LINK = 0,
        TRIGGER_NETDEV_LINK_10,
        TRIGGER_NETDEV_LINK_100,
        TRIGGER_NETDEV_LINK_1000,
        TRIGGER_NETDEV_HALF_DUPLEX,
        TRIGGER_NETDEV_FULL_DUPLEX,
        TRIGGER_NETDEV_TX,
        TRIGGER_NETDEV_RX,
```

- One to one mapping to files in sysfs

```
full_duplex half_duplex link
link_10     link_100    link_1000
rx          tx
```




```
echo 1 > left_led/link
echo 1 > right_led/rx
echo 1 > right_led/tx
```

Go blink your lights the right way!

Questions?