Improving IEEE 802.15.4 MAC management support in the Linux kernel

Miquel Raynal
miquel.raynal@bootlin.com
Embedded Linux engineer at Bootlin
- Embedded Linux expertise
- Development, consulting and training
- Strong open-source focus

Open-source contributor
- Maintainer of the NAND subsystem
- Co-maintainer of the MTD subsystem
- Kernel support for various ARM SoCs
- Active contributor to the WPAN subsystem with Qorvo support

Living in Toulouse, France
The IEEE 802.15.4 specification in a nutshell
Functional description

- Defines the PHY layer and the MAC sublayer
  - Introduced to build Wireless Personal Area Networks (WPAN)
  - Low power, low range (10m), low rate (up to 250kib/s)
  - Easy connection between sensors and actuators
  - A base for Zigbee and 6LowPan

- Focus on the MAC sublayer:
  - MAC data services
  - MAC management services through the MAC subLayer Management Entity (MLME)
Personal Area Networks (PAN)

Devices connect together to form PANs

➤ One PAN coordinator which takes a PAN ID
  • Advertises the PAN
  • Allows devices associations
  • May serve as a bridge with the Internet

➤ Coordinators
  • Advertise the PAN
  • Allow devices associations
  • Follow the PAN coordinator realignments

➤ Leaf nodes
  • Follow their coordinator realignments
  • Send data
PAN advertisement and discovery

- Coordinators and PAN coordinators shall advertise their PAN by sending beacons
  - Either upon reception of a BEACON REQUEST
  - Or “passively” at a given rate, in beacon-enabled PANs
- Beacons are short frames with information about the emitting device and its PAN

- Devices can scan the various channels they support
  - Passive scans to detect surrounding beacon enabled PANs with their LQI
  - Active scans to detect surrounding beacon and non-beacon enabled PANs with their LQI
MLME operations

- MAC management commands
  - Discovering surrounding devices (scanning, beaconsing)
  - Enlarging/shrinking the network (associating, dis-associating)
  - Keeping all devices synchronized (beaconing, acknowledgments, etc)
  - Handling faulty situations (loss of contact, conflicts, etc)
Improving IEEE 802.15.4 MAC management support

The Linux kernel IEEE 802.15.4 stack
New MLME interfaces: scanning/beaconing

- Netlink user requests with commands related payloads
  - Type of scan
  - Beacon interval
  - Channels to use

- Once processed, the request is forwarded to the MAC layer:
  - Changes on the ongoing Tx traffic
  - Hardware address filters update
  - Start of a background thread

- Background jobs can be aborted at any moment (netlink command) or may end naturally
  - Upon completion/abort userspace gets notified
  - The interface is set back in its original state
New MLME interfaces: associating/dis-associating

- No background job involved
- MLME commands must be ACKed
- Processing received requests often involves sending a response
- Association requests should be forwarded to userspace for validation (not implemented)
Demo time

Hardware setup:

- One ATUSB device acting like a PAN coordinator (wpan0/coord0)
- One ATUSB device acting like a leaf node (wpan1/coord1)
- One ATUSB device monitoring (wpan2/mon2)
- One Arduino Nano 33 BLE running Zephyr being a leaf node
Upstream proposals, discussions ongoing

Kernel patches:
- v2 https://lore.kernel.org/all/20220826144049.256134-1-miquel.raynal@bootlin.com/
- v3/only filtering https://lore.kernel.org/all/20220905203412.1322947-1-miquel.raynal@bootlin.com/
- Latest version https://github.com/miquelraynal/linux/tree/wpan-next/scan

wpan-tools patches:
- Last patches https://lore.kernel.org/all/20220701143434.1267864-1-miquel.raynal@bootlin.com/

Zephyr changes https://github.com/zephyrproject-rtos/zephyr/pull/49947

No support for orphan notifications/coordinator realignments yet
Improving IEEE 802.15.4 MAC management support

Major issues, past and present
Description of the problem:

- The wpan core expects transmit callbacks to be asynchronous
  - Only packet offloading to the transceiver is synchronous
    - Errors during this step can be returned
  - The actual packet transmission may happen later
    - Can only happen when the medium is available
    - Perhaps there won’t be any timeslot available immediately
    - The transceiver needs to wait for an ACK
    - Possibility to repeat the packet up to 7 times (usually 3)
  - MLME transmissions must be ACKed, we need a status
  - We want MLME operations to be over before resuming normal operations

- Asynchronous transmissions cannot work alone, we need an alternative
Solution considered:

- A hot path is considered for data transmission
- Creation of a second, slow and synchronous Tx Path for MLME transmissions
  - Device drivers already call helpers upon completion to:
    - Handle Inter Frame Spacing (IFS), a short inactive period after each frame
    - Consume the skb
  - We could use them to:
    - Follow the count of ongoing transmissions
    - Possibly return an error code, if any?

```
#include/net/mac802154.h

void ieee802154_xmit_complete(struct ieee802154_hw *hw, struct sk_buff *skb, bool ifs_handling);
void ieee802154_xmit_error(struct ieee802154_hw *hw, struct sk_buff *skb, int reason);
```
Sub-problem: forwarding errors

- It is important to know if the MLME frame was successfully received
- Example of status register: TRAC
  - Not in the spec
  - Hopefully wide spread

TRAC values

```
TRAC_SUCCESS
TRAC_SUCCESS_DATA_PENDING
TRAC_SUCCESS_WAIT_FOR_ACK
TRAC_CHANNEL_ACCESS_FAILURE
TRAC_NO_ACK
TRAC_INVALID
```
Down side:
- Using ATUSB devices: no TRAC register support in the firmware

Bright side:
- The firmware is open source
- Another driver (`at86rf230.c`) is very close and has TRAC support
- Alexander Aring (WPAN co-maintainer) does not need sleep and knows very well the firmware
- Patches now available
  
  https://lore.kernel.org/all/20220906082104.1338694-3-miquel.raynal@bootlin.com/

In general, all devices and drivers should have access to these information, otherwise they are badly designed
Filtering constraints

- Intermediate filtering levels not described
  - Currently supported: “no filtering” or “full filtering”
  - Specific filtering modes shall be entered during scans
  - Must be supported by software if not available in hardware

- Enabling promiscuous mode (the one which disables all checks) also disables ACKs for sniffing devices
  - As opposed to filtering levels, ACK generation cannot be handled by software
  - Running interfaces in promiscuous mode and non promiscuous mode on the same PHY shall be prevented

- The promiscuous mode alone does not make sense
  - Alexander has proposed a first patch to move towards a more unified filtering API and give to the `drv_start()` hook a filtering level to work with
Lockdep hates me 1/2

There is still an issue with the scan:

- Scanning involves changing channels in the background job
  - Channel changes are protected by the `rtnl`
  - `mac802154_hwsim.c` for instance requires the `rtnl` when changing some of its PIB attributes like the channel/page.
    - Lock dependency: `scan_lock → rtnl`
  - The `rtnl` is always acquired before changing scan parameters
    - There is an `ASSERT_RTNL()` in the `nl802154_pre_doit()` as soon as we need a `netdev`
      - Lock dependency: `rtnl → scan_lock`
  - Circular dependency!
  - Mandatory to take the `rtnl` before the `scan_lock` in the background job

So the in the end the `scan_lock` is useless, besides showing what needs to be protected if we ever drop the `rtnl` there
Situation:

- Scanning/beaconing background jobs use the device’s workqueue
  - Workqueue completion lock acquired when the job is running
  - The job acquires the `rtnl`
    - Dependency lock: `workqueue lock → rtnl`
  - At `stop()` time the `rtnl` must be acquired
  - Stopping a device involves flushing the workqueue, with its lock acquired
    - Dependency lock: `rtnl → workqueue lock`
  - Circular dependency!

Workaround: using another queue for the MLME background jobs, which must be stopped before removal anyway and does not need to be flushed explicitly with the `rtnl` acquired
Questions? Suggestions? Comments?

Miquel Raynal
miquel.raynal@bootlin.com

Slides under CC-BY-SA 3.0
https://bootlin.com/pub/conferences/