rtl8xxxu

- true love for cheap USB WiFi dongles

Jes Sorensen <Jes.Sorensen@gmail.com>

How this project came about

- Purchased laptop with rlt8723au device
  - No driver in upstream kernel.
  - No specs
  - Out of tree vendor provided driver
    - I can clean this up in a couple of months and get it merged... riiiiiiight!
- 6 months hacking relentlessly on vendor code, finally got rtl8723au included into staging

Linux 802.11 stack

- FullMAC vs SoftMAC
  - MLME (Media Access Control (MAC) Sublayer Management Entity)
- SoftMAC use mac80211
- FullMAC use cfg80211

Realtek hardware and software

- Only 802.11N+ devices: 1T1R 2T2R 1WiFi+1BT
- Simple device, limited FW assist
- SW or firmware rate control
- Multiple TX+RX packet descriptor formats
- No direct method reporting TX speed. Gen 2 parts have some
- Internal 'paths' configurable to external antenna. Bits set which antenna is enabled for TX+RX
- USB/SDIO/PCIe version of each
- USB DMA packet aggregation
- Could use different RF modules

Chip	MIMO/BT	Gen	Support
8188su	1T1R	0	No
8192su	2T2R	0	No
8723au	1T1R+BT	1	Yes
8188cu	1T1R	1	Yes
8192cu	2T2R	1	Yes
8188ru	1T1R hi-pa	1	Yes
8192du	2T2R abgn	1	No
8188eu	1T1R	1.5	Almost
8192eu	2T2R	2	Yes
8723bu	1T1R+BT	2	Yes
881xau	4T4R ac	3	Not yet

• Realtek development process

1) Respin hardware
2) cp -a driver-<oldchip> driver-<newchip>
3) Hack driver-<newchip>
4) Release driver-<newchip>
5) goto 1

- Endless revisions of drivers
- No multi-device support – no multi bus support

Realtek vendor drivers

- Cross platform #ifdefmeharder: Windows XP, Windows CE, OSX, FreeBSD, Android, ARM embedded Linux (routers & TVs), Linux
- Emulates fullmac driver – comes with own 802.11 stack
- Multiple teams maintaining different driver modules using different styles: hal, ODM, core, OS:
  - Multiple defines for the same registers
  - Different APIs for accessing the same registers:

```
PHY_SetBBReg(PADAPTER Adapter, u32 RegAddr, u32 BitMask, u32 Data)
ODM_Write1Byte(PDM_ODM_T pDM_Odm, u4Byte RegAddr, u1Byte Data)
ODM_SetBBReg(PDM_ODM_T pDM_Odm, u4Byte RegAddr, u4Byte BitMask, u4Byte Data)
```

- Command/event architecture to match hardware
  - Except hardware doesn't have command/event
  - Let's emulate in software!

rtl8723au mac80211 (softmac) driver

- How hard can it be?
- X days later .... receive data .... connect to AP .... crypto
- Documentation? what documentation?
  - Read vendor driver over and over and over and over to understand what it is doing and why.
  - Trace register read+writes compare to vendor flow
  - Document registers based on vendor code + comments
- Lots of initialization via register files (reg value + data)
- Register files for power state change
- Relies on firmware rate control
- BT control via register reads/writes – no work on BT

8723au and 8188cu/8192cu

- Very similar devices – adding support was a couple of days of work
- Same TX+RX descriptor format
- Same firmware API (48 bits for H2C commands/C2H events)
- Device feature detection
- Retrieve init register files (reg+data) from vendor driver
- Handle 2T2R setup and channel config
- Special handling for 8188ru due to high power amplifier (special version of 8188cu)
- DMA packet aggregation

Moving on to gen2 - 8723bu

- More! more! ordered every cheap dongle I found
- Had to get 8723bu from online Chinese retailer. Now started to show up in mini desktops + tablets
- New RX and TX descriptor formats
- New firmware API (64 bits for H2C commands + C2H events)
- Init flow in vendor driver reordered
- Firmware commands for selecting antennas and BT assignment
- New S0S1 internal path switch – haven't figured out how this work yet. Presumably related to BT vs WiFi
- No work on BT

8192eu

- Standard 2T2R similar to 8192cu – no BT
- Same TX/RX descriptor format as 8723bu
- Same firmware H2C/C2H API as 8723bu
- Reordered init sequence

8188eu

- Oddball inbetween device – one of the most common 150N devices on the market
- Odd IOL firmware assist API
  - Used for device setup – can be ignored
- Uses gen2 firmware H2C/C2H API
- Uses gen1 channel configuration API
- No firmware rate control!
- Currently works for non MCS (N) rates (B+G)

Status

- 8723au/8188cu/8188ru/8192cu/8192eu/8723bu upstream
- 8188eu work in progress
- Station and monitor mode supported
- DMA aggregation added
- Patch to remove drivers/staging/rtl8723au submitted

TODO – Help wanted!

- Finish up 8188eu
- Host rate control
- Beacon support – AP and Ad-Hoc mode
- SDIO support
- AMPDU support
- Automatic antenna detection
- PCIe support
- 802.11ac devices
- 8192du
- Bluetooth support for 8723au/8723bu

Lessons learned

- WiFi is just a „little“ more complex than Ethernet
- Getting something into staging does not mean the job is done
  - Magnet for cosmetic fixups, few real fixes
- Register access traces are magic
- „How hard can it be“ really means: Walk away now!

Acknowledgements

- Johannes Berg: Answering endless questions
- Larry Finger: Help with vendor drivers and vendor communication
- Andrea Merello & Taehee Yoo: 8188eu
- Bruno Randolf: Monitor mode
- Jakub Sitnicki: Early 8192eu work