

Coordinator Rev1

Notes on the first revision coordinator hardware and software

- [Hardware Errata](#)
- [Firmware Notes](#)
- [Host Firmware Notes](#)

Hardware Errata

Collection of some notes on the rev1 coordinator hardware

Assembly

- Combine 100nF capacitors on BoM
 - C804 (100V) and others (logic level) can be served by the same item
- 0402 size for bottom indicator LEDs is ass
 - They ought to be larger, 0603 or 0805
 - Alternatively, use reverse-mount LEDs that are easier to work with (and can be reflowed)
- Ethernet PHY is not recognized
 - Appears to just be a soldering issue with the pads on the SoM not making proper connection
 - Next rev should have larger SoM pads with more cream?
- Fix footprint for C802, C803
 - They are actually 3225, not 3216

Operation

- Wifi seems to be busted
 - This likely is a software issue; the module initializes correctly and is recognized on boot (with the appropriate drivers being loaded)

```
[ 19.308901] mwifiex_sdio mmc0:0001:1: info: FW download over, size 616840
bytes
[ 19.834461] mwifiex_sdio mmc0:0001:1: WLAN FW is active
[ 20.254606] mwifiex_sdio mmc0:0001:1: info: MWIFIEX VERSION: mwifiex 1.0
(15.68.7.p189)
[ 20.261290] mwifiex_sdio mmc0:0001:1: driver_version = mwifiex 1.0
(15.68.7.p189)
[ 32.322732] fixed-3v3: disabling
[ 32.324575] vdd_sd: disabling
[ 317.632606] mwifiex_sdio mmc0:0001:1: Firmware wakeup failed
[ 317.637160] mwifiex_sdio mmc0:0001:1: PREP_CMD: FW in reset state
[ 317.653566] mwifiex_sdio mmc0:0001:1: info: shutdown mwifiex...
[ 317.672972] mwifiex_sdio mmc0:0001:1: PREP_CMD: card is removed
[ 317.775670] mwifiex_sdio mmc0:0001:1: WLAN FW already running! Skip FW dnld
[ 317.781226] mwifiex_sdio mmc0:0001:1: WLAN FW is active
[ 327.952520] mwifiex_sdio mmc0:0001:1: mwifiex_cmd_timeout_func: Timeout cmd
id = 0xa9, act = 0x0
[ 327.959898] mwifiex_sdio mmc0:0001:1: num_data_h2c_failure = 0
```

```

[ 327.965752] mwifiex_sdio mmc0:0001:1: num_cmd_h2c_failure = 0
[ 327.971442] mwifiex_sdio mmc0:0001:1: is_cmd_timedout = 1
[ 327.976846] mwifiex_sdio mmc0:0001:1: num_tx_timeout = 0
[ 327.982131] mwifiex_sdio mmc0:0001:1: last_cmd_index = 0
[ 327.987447] mwifiex_sdio mmc0:0001:1: last_cmd_id: a9 00 28 00 16 00 cd 00 1e
00
[ 327.994853] mwifiex_sdio mmc0:0001:1: last_cmd_act: 00 00 13 00 01 00 01 00
00 00
[ 328.002295] mwifiex_sdio mmc0:0001:1: last_cmd_resp_index = 4
[ 328.008046] mwifiex_sdio mmc0:0001:1: last_cmd_resp_id: df 80 28 80 16 80 cd
80 1e 80
[ 328.015874] mwifiex_sdio mmc0:0001:1: last_event_index = 1
[ 328.021326] mwifiex_sdio mmc0:0001:1: last_event: 00 00 0b 00 00 00 00 00 00
00
[ 328.028643] mwifiex_sdio mmc0:0001:1: data_sent=1 cmd_sent=1
[ 328.034285] mwifiex_sdio mmc0:0001:1: ps_mode=0 ps_state=0
[ 328.043571] mwifiex_sdio mmc0:0001:1: info: _mwifiex_fw_dpc: unregister
device

```

- No pull-up on /RF_RESET line
 - This means that every time the SoC resets, the RF part also resets. A hardware pull-up prevents this when the GPIOs go tristate during reset
- Ethernet port LEDs are ~ bright ~
 - Should be increased from 330Ω, they're a little on the bright side. 1k is probably fine, as they needn't be super bright

Firmware Notes

This page collects some notes about the [host-driven RF firmware](#), running on the EFR32FG23 chip on the board.

- RAIL initialization fails with custom build system
 - This is because a LDMA transfer is set up, but something gets wonky with the initialization and the destination address mode is set to *decrement*.
 - Most likely caused due to an ABI mismatch between the compiled RAIL library, and our code (or some headers?) compiled for the drivers, specifically the LDMA driver; current workaround is to monkeypatch the LDMA driver to never set the direction bits (thus completely ignoring the "broken" transfer struct)
- Temperature sensor driver has cast to double
 - In `TEMPDRV_GetTemp()`, the 0.5 constant needs to have a `f` suffix to make it float, rather than double to compile with the enhanced warnings about upcasts
- Host irq system is busted
 - There should be a separate "irq acknowledge" register, instead of making the interrupt levels be dependent on doing some action
 - Currently there's a possible race between an irq handler and a packet receive, which wedges irq's

Host Firmware Notes

Some notes about the Linux host firmware (see this [GitHub repo](#)).

- UARTs don't work in DMA mode
 - All UARTs (including the high speed RF TTY/serial spew) can only operate in interrupt driven mode. This is pretty inefficient
- GPIO IRQs don't work
 - Kernel drivers (such as buttons) fail to get external interrupts
 - Buttons need to be operated in polling mode