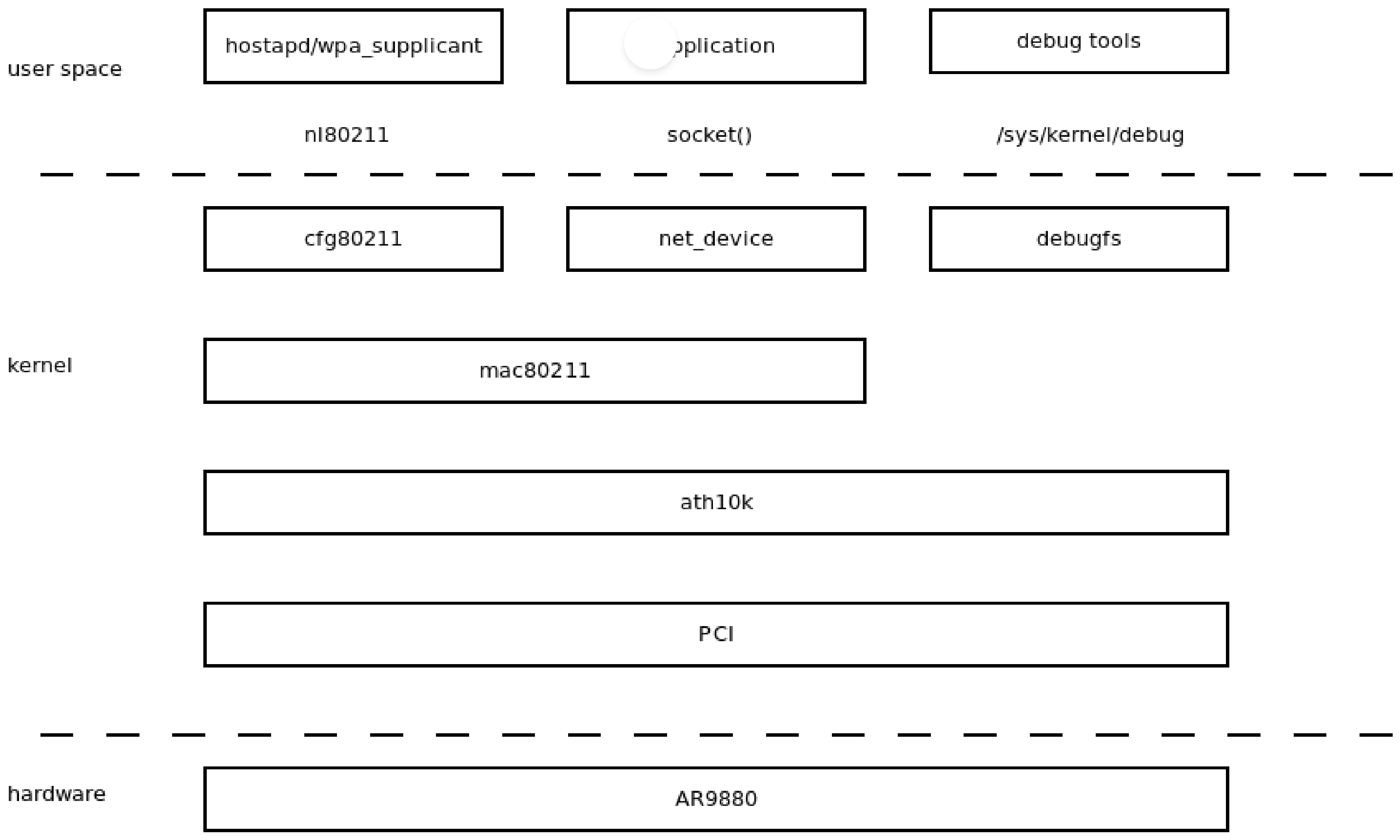
Go back –> ath10k

ath10k architecture

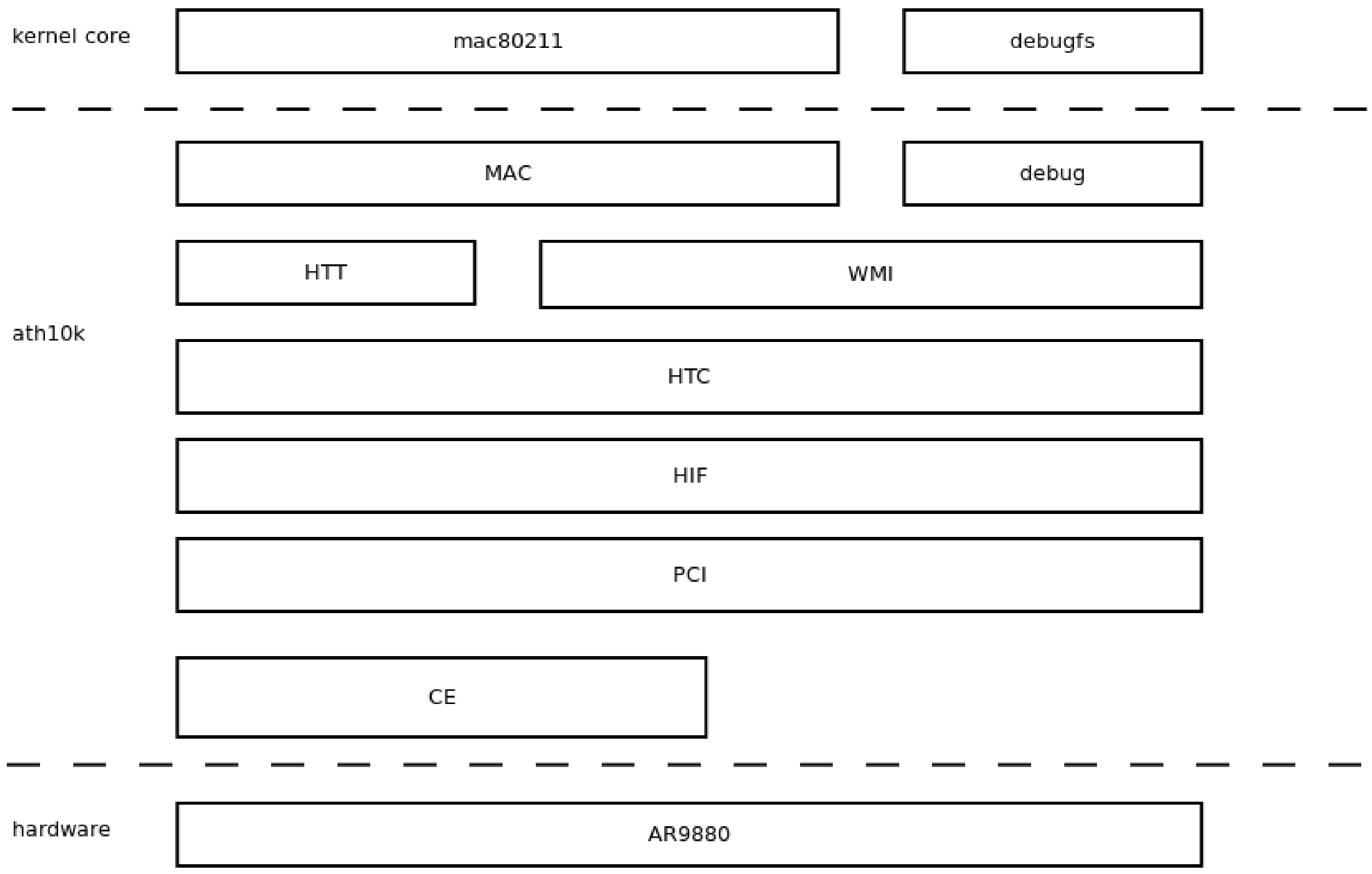
ath10k is a mac80211 driver, the architecture is depicted in the diagram below.



The driver is located in directory drivers/net/wireless/ath/ath10k/. The source code is available for browsing from this location:

https://github.com/kvalo/ath/tree/master/drivers/net/wireless/ath/ath10k [https://github.com/kvalo/ath/tree/master/drivers/net/wireless/ath/ath10k]

ath10k components



MAC

- Files: mac.h mac.c This is the glue layer between mac80211 and ath10k lower levels. The interface to mac80211 is implemented through ath10k_ops. Data and management frames are sent to HTT, configuration commands to WMI.

Host-Target Transport (HTT)

- Files: htt.c htt.h htt_rx.c htt_tx.c The data path for ath10k. Sends frame descriptors to the firmware using HTC.

Wireless Module Interface (WMI)

- Files: wmi.h wmi.c The control path for ath10k. Sends all sorts configuration commands to the firmware and receives configuration events from the firmware.

Host-Target Communication (HTC)

- Files: htc.h htc.c Multiplexes the bus for different services. The services are defined in enum ath10k_htc_svc_gid.

Host interconnect Framework (HIF)

- Files: hif.h Abstracts the access to different bus types. Currently only supports PCI, but it's easy to add different bus types.

Debug

- Files: debug.h debug.c Component for various debug related to code. Currently only log messages and debugfs.

Tracing

- Files: trace.h trace.c Provides tracing data (HTT/WMI packets) etc to userspace using Linux tracepoints. trace-cmd is the recommend tool to access the tracepoints.

PCI

- Files: pci.h pci.c ce.h ce.c
- Module: ath10k_pci.ko All PCI related code. Interface to HIF happens through ath10k_pci_hif_ops.

Copy Engine (CE)

The firmware/ hardware has 8 rings for communication with host, defined in host_ce_config_wlan:

```
static const struct ce_attr host_ce_config_wlan[] = {
        /* CE0: host->target HTC control and raw streams */
        {
                .flags = CE_ATTR_FLAGS,
                .src_nentries = 16,
                .src_sz_max = 256,
                .dest_nentries = 0,
        },

        /* CE1: target->host HTT + HTC control */
        {
                .flags = CE_ATTR_FLAGS,
                .src_nentries = 0,
                .src_sz_max = 512,
                .dest_nentries = 512,
        },

        /* CE2: target->host WMI */
        {
                .flags = CE_ATTR_FLAGS,
                .src_nentries = 0,
                .src_sz_max = 2048,
                .dest_nentries = 32,
        },

        /* CE3: host->target WMI */
        {
                .flags = CE_ATTR_FLAGS,
                .src_nentries = 32,
                .src_sz_max = 2048,
                .dest_nentries = 0,
        },

        /* CE4: host->target HTT */
        {
                .flags = CE_ATTR_FLAGS | CE_ATTR_DIS_INTR,
                .src_nentries = CE_HTT_H2T_MSG_SRC_NENTRIES,
                .src_sz_max = 256,
                .dest_nentries = 0,
        },

        /* CE5: unused */
        {
                .flags = CE_ATTR_FLAGS,
                .src_nentries = 0,
                .src_sz_max = 0,
                .dest_nentries = 0,
        },

        /* CE6: target autonomous hif_memcpy */
        {
                .flags = CE_ATTR_FLAGS,
                .src_nentries = 0,
                .src_sz_max = 0,
                .dest_nentries = 0,
        },

        /* CE7: ce_diag, the Diagnostic Window */
        {
                .flags = CE_ATTR_FLAGS,
                .src_nentries = 2,
                .src_sz_max = DIAG_TRANSFER_LIMIT,
                .dest_nentries = 2,
        },
};
```

Copy Engine provides abstraction for these ring buffers and calls each ring a pipe.

Bootloader Messaging Interface (BMI)

- Files: bmi.h bmi.c Firmware upload and everything else which happens before firmware is booted.

Core

- Files: core.h core.c Driver initialisation and firmware booting. Manages all ath10k components.