Felix Chen, Per Stromhaug
PA4 Design Doc

Device Driver:
We have several functions in our network.c file:

- void network_init(); - finds the network device, initializes the ring buffer, turns the device on, sets the rx_base and rx_capacity values, and allocates space for the dma_ring_slots.

- void network_start_receive(); - and lets it start receiving packets by writing the NET_SET_RECEIVE constants to dev_net->cmd and writing 1 to dev_net->data

- void network_set_interrupts(int opt); - turns on interrupts in a similar manner, by writing to a register

- void network_poll(); - loops forever to see if ring buffer is non-empty.

- void network_trap(); - will get called whenever an interrupt happens in order to handle a packet.

Polling vs. Interrupts

Polling is more efficient than interrupts, as interrupts involve passing control to the OS, and all packets that are delivered while an interrupt from a previous packet is ongoing will be dropped. Interrupts may be a simpler solution, but we will have a dedicated core for polling that will continuously scan for new packets and add them to the ring buffer. With this method, there will be less potentially dropped packets.

Plans for cores:

Our plans for cores currently aren't incredibly ambitious. We will first focus heavily on getting a working design, with the hopes of improving performance with concurrency if there is time at the end. Here's the current plan:

- One core handling polling and updating the ring buffer with new packets
- Several other cores taking packets from the ring buffer and immediately storing them in memory. The faster we get packets out of the ring buffer and into memory, the lower the chances are that the ring buffer fills up and we drop packets.
- The few remaining cores taking packets from memory, and computing and updating the statistics

This plan is a pipelined approach because there can be at least two separate stages: one stage to retrieve packets from the ring buffer store into memory and one stage to retrieve from memory and compute and update statistics. We can possible even break the process down into more stages to increase performance.