```
            let n = receive_packet_over_spi(buf).await;
            rx_chan.rx_done(n);
        }
        Either::Second(buf) => {
            // a packet is ready to be sent!
            send_packet_over_spi(buf).await;
            tx_chan.tx_done();
        }
    }
}
```

However, this code has a latent deadlock bug. The symptom is it can hang at `rx_chan.rx_buf().await` under load.

The reason is that, under load, both the TX and RX queues can get full at the same time. When this happens, the `embassy-net` task stalls trying to send because the TX queue is full, therefore it stops processing packets in the RX queue. Your driver task also stalls because the RX queue is full, therefore it stops processing packets in the TX queue.

The fix is to make sure to always service the TX queue while you're waiting for space to become available in the RX queue. For example, select on either "tx_chan.tx_buf() available" or "INT is low AND rx_chan.rx_buf() available":

```
loop {
```