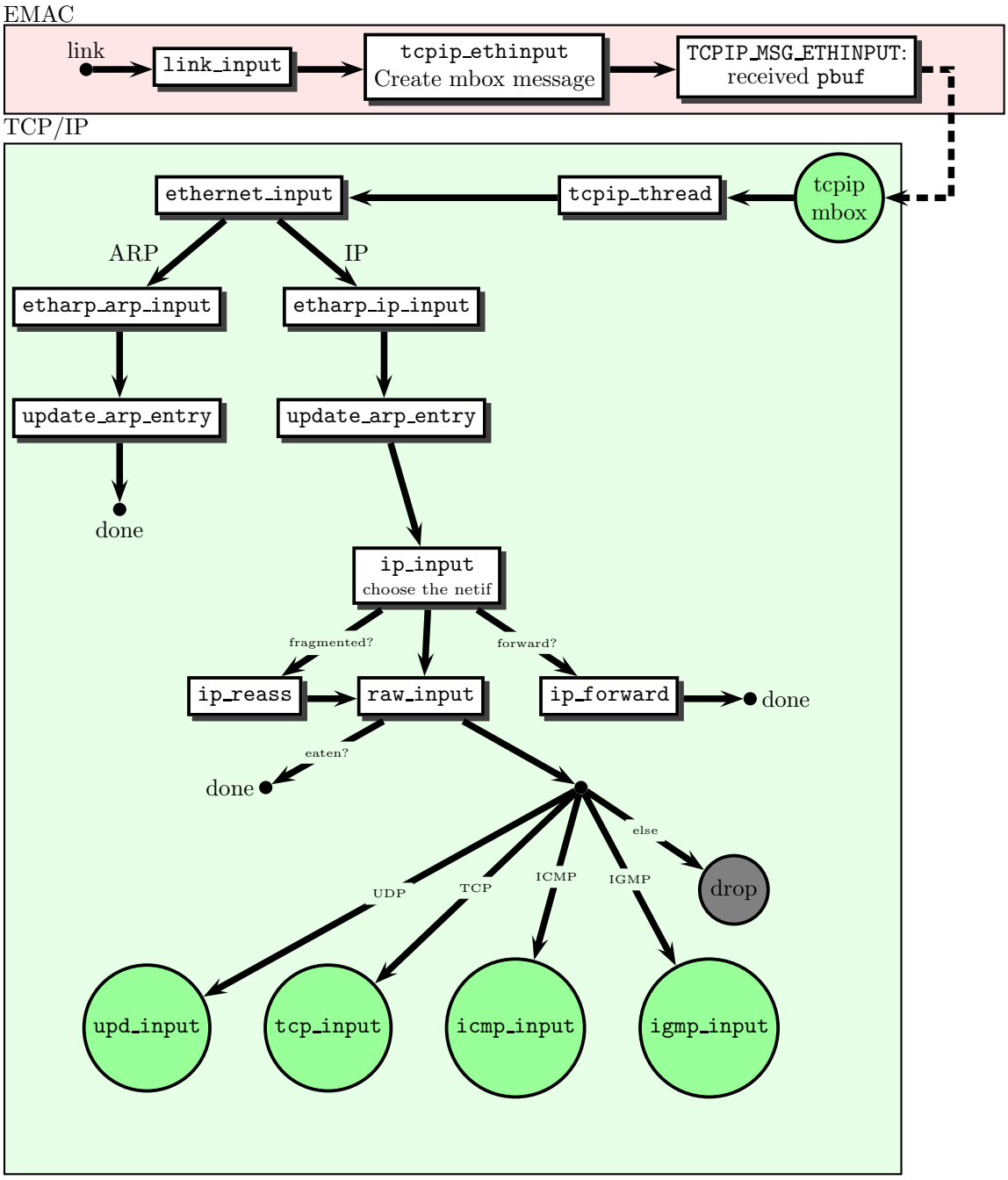
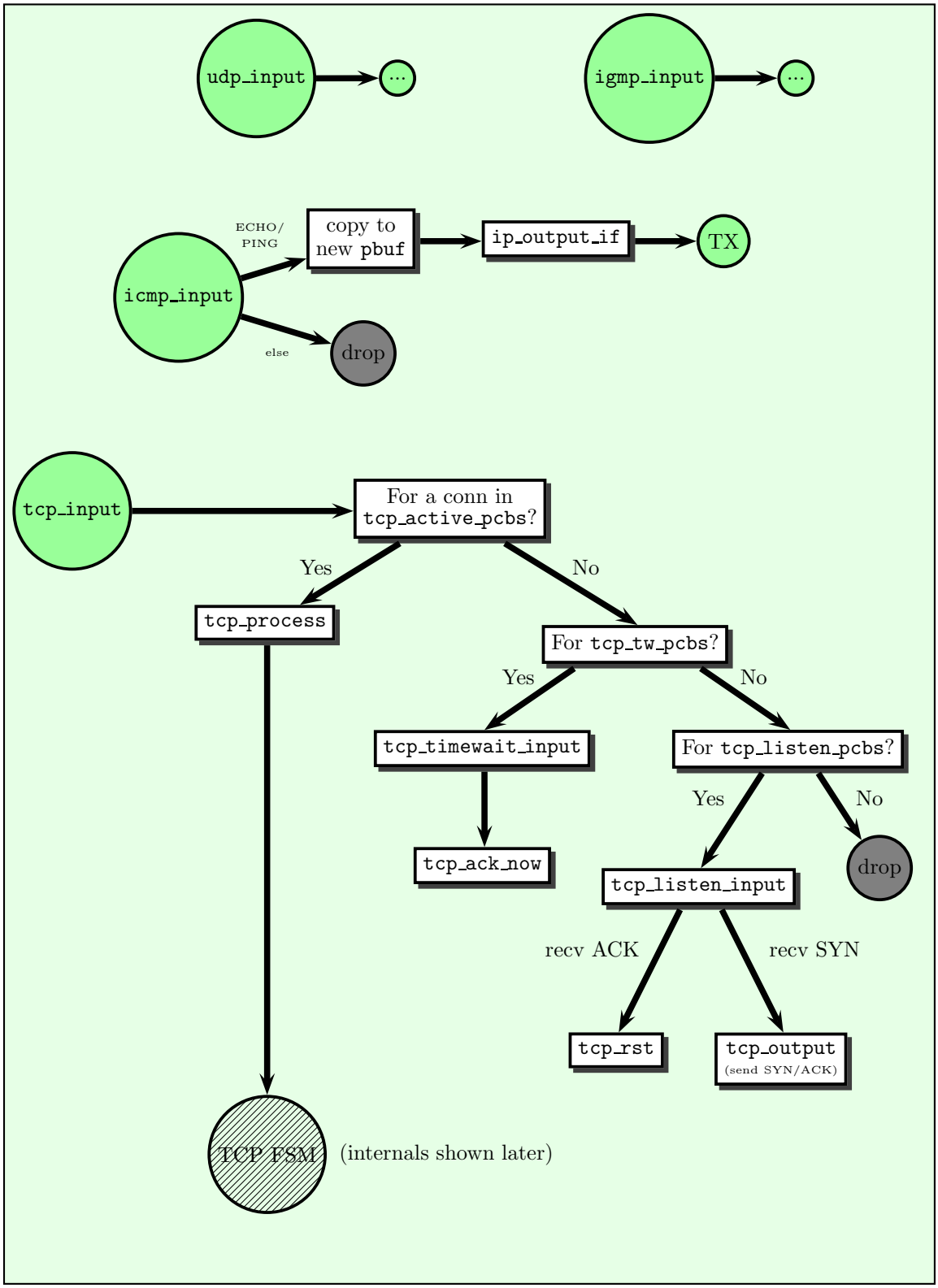
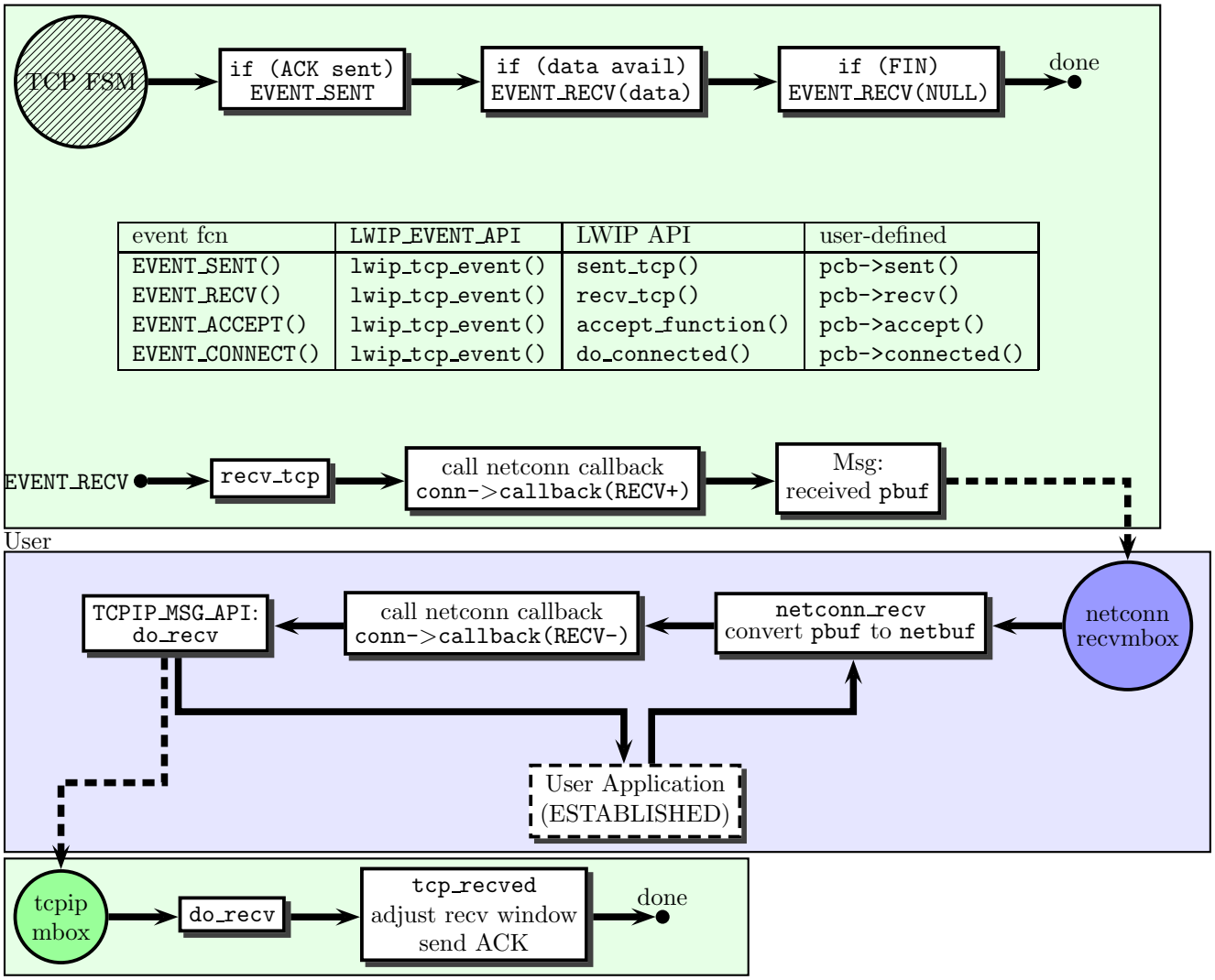
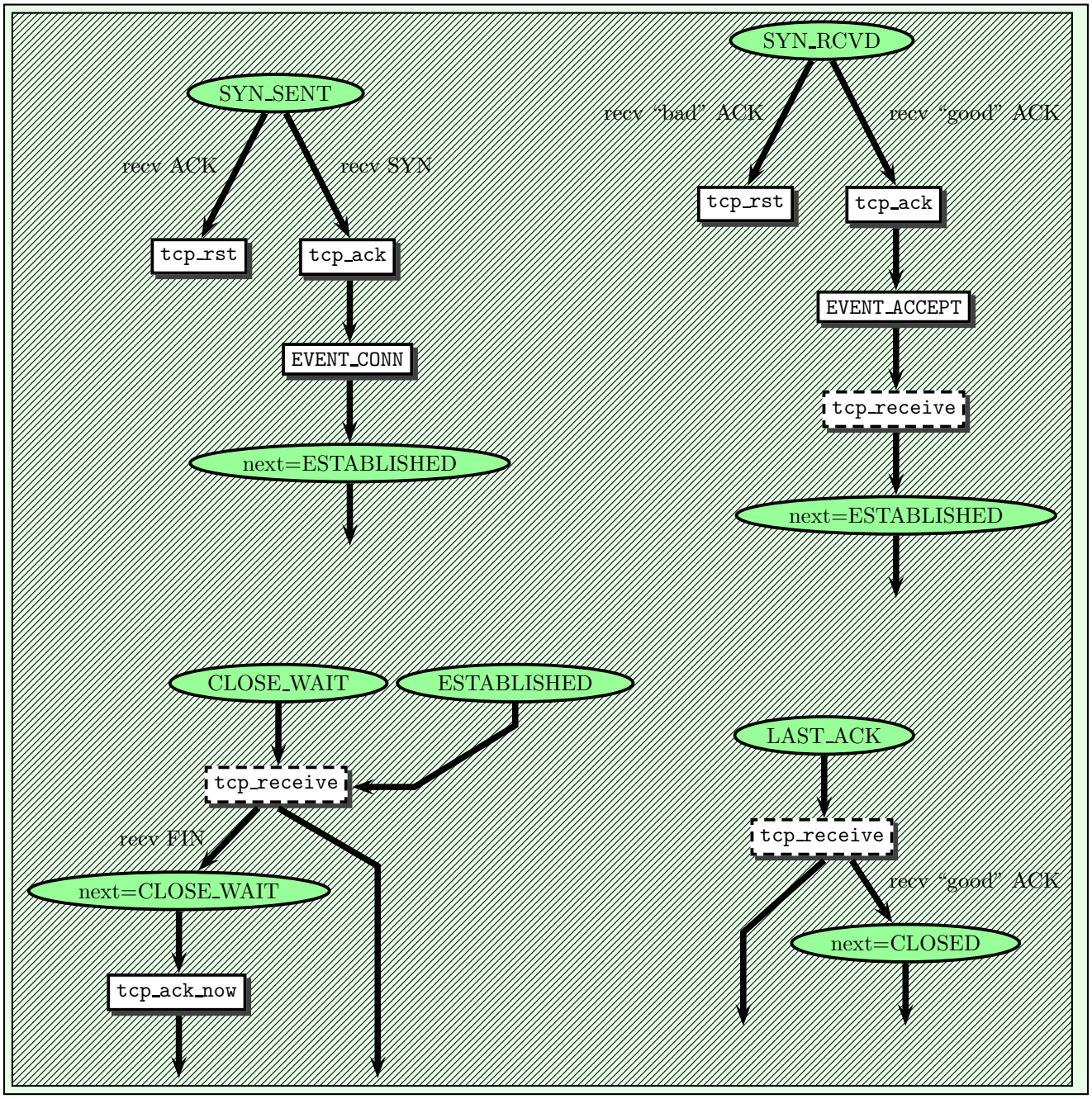
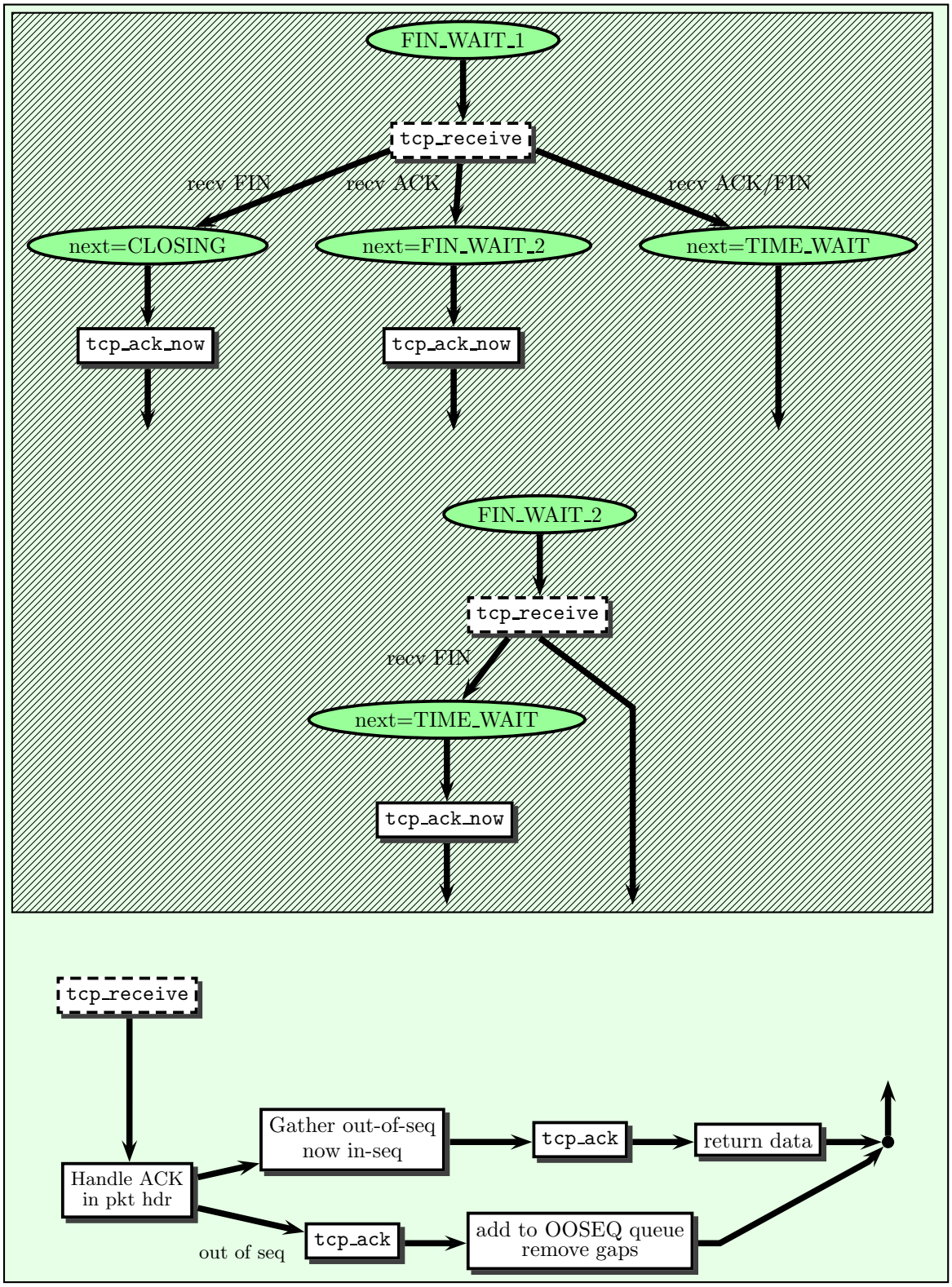

EMAC
link
link_input
tcpip_ethinput
Create mbox message
TCPIP_MSG_ETHINPUT:
received pbuf
TCP/IP
ethernet_input
tcpip_thread
tcpip
mbox
ARP
IP
etharp_arp_input
etharp_ip_input
update_arp_entry
update_arp_entry
done
ip_input
choose the netif
fragmented?
forward?
ip_reass
raw_input
ip_forward
done
eaten?
done
else
drop
UDP
TCP
ICMP
IGMP
upd_input
tcp_input
icmp_input
igmp_input


udp_input
...
igmp_input
...
ECHO/
PING
copy to
new pbuf
ip_output_if
TX
icmp_input
else
drop
tcp_input
For a conn in
tcp_active_pcbs?
Yes
No
tcp_process
For tcp_tw_pcbs?
Yes
No
tcp_timewait_input
For tcp_listen_pcbs?
Yes
No
tcp_ack_now
drop
tcp_listen_input
recv ACK
recv SYN
tcp_rst
tcp_output
(send SYN/ACK)
TCP FSM
(internals shown later)



event fcn	LWIP_EVENT_API	LWIP API	user-defined
EVENT_SENT()	lwip_tcp_event()	sent_tcp()	pcb->sent()
EVENT_RECV()	lwip_tcp_event()	recv_tcp()	pcb->recv()
EVENT_ACCEPT()	lwip_tcp_event()	accept_function()	pcb->accept()
EVENT_CONNECT()	lwip_tcp_event()	do_connected()	pcb->connected()


SYN_SENT
recv ACK
recv SYN
tcp_rst
tcp_ack
EVENT_CONN
next=ESTABLISHED
SYN_RCVD
recv "bad" ACK
recv "good" ACK
tcp_rst
tcp_ack
EVENT_ACCEPT
tcp_receive
next=ESTABLISHED
CLOSE_WAIT
ESTABLISHED
tcp_receive
recv FIN
next=CLOSE_WAIT
tcp_ack_now
LAST_ACK
tcp_receive
recv "good" ACK
next=CLOSED


FIN_WAIT_1
tcp_receive
recv FIN
recv ACK
recv ACK/FIN
next=CLOSING
next=FIN_WAIT_2
next=TIME_WAIT
tcp_ack_now
tcp_ack_now
FIN_WAIT_2
tcp_receive
recv FIN
next=TIME_WAIT
tcp_ack_now
tcp_receive
Handle ACK in pkt hdr
Gather out-of-seq now in-seq
tcp_ack
return data
out of seq
tcp_ack
add to OOSEQ queue remove gaps