```
Void send_frame (void)
{
  s.info = buffer;
 s.seq = next_frame_to_send:
 to_physical_layer(&s);
}
#define MAX_SEQ 1
typedef enum {frame_arrival, cksum_err, timeout} event_type;
include “protocol.h”
void sender_NAK (void)
{
  seqnr next_frame_to_send;
  frame s,r;
  packet buffer
  event_type event;
  next_frame_to_send = 0;
  from_network_layer(&buffer);
  while (true)
  {
     send frame ();
     start_timer (s.seq);
     wait_for_event (&event);
      if (event == frame_arrival)
       { from_physical_layer(&r);
         If (r.kind == ACK  and r.ack == next_frame_to_send)
         {  stop_timer(s.ack);
            from_network_layer(&buffer);
            inc(next_frame_to_send} }
       }
   } /* Note – cksum_err, timeout and ACK-mismatch and NAK all come here!
}
```

```
void receiver_NAK (void)
{
 seq_nr  frame_expected;
 frame r, s;
 event_type event;

 frame_expected = 0;

 while (true)
 {  wait_for_event (&event);
    if (event == frame_arrival)
    {  from_physical_layer (&r);
       if (r.seq == frame_expected)
       {  to_network_layer(&r.info)
          inc (frame_expected);
          frame_expected = frame_expected MOD MAX_SEQ;
        }
          s.kind = ACK
          s.ack =  1 – frame_expected;
          to_physical_layer (&s);
    }
     If (event == chksum_err)     /* cksum_err comes here */
      {
          s.kind = NAK
          s.ack =  ?
          to_physical_layer (&s);
      }
 }
}
```