Common.h

• You can add your own data structures here.

• struct Sender_t
  • SWS – Sliding window size
  • LAR (Last Acknowledgement Received) - Sequence number of last acknowledgement received, defines lower bound of the sender window
  • LFS (Last Frame Sent)- Sequence number of the last frame sent, defines upper bound of the window
  • Window is from [LAR+1, LFS], that is all frames that have been sent but not yet Acked.
Frame Sequence Number in Sender

**CASE 1: Usual Case**
LAR <= LFS

LAR <= LFS && seqNo > LAR && seqNo <= LFS

**CASE 2: Sequence Number Wrap Around**
LAR > LFS

LAR > LFS && (seqNo > LAR || seqNo <= LFS)

In this case, we are not using the full window of 4.

Sender with SWS = 4, sequence number in [0,7]
Receiver_t

• RWS - Max receiver window size
• NFE - Next Frame Expected
• LFR - Sequence number of largest consecutive frame received
• LAF - Sequence number of largest acceptable frame
• LFR = NFE - 1
• LAF = NFE + RWS - 1
Frame Sequence Number in Receiver

CASE 1: Usual Case
\[ NFE + RWS - 1 \geq NFE \]
\[ NFE + RWS - 1 \geq NFE \land \text{seqNo} \geq NFE \land \text{seqNo} \leq NFE + RWS - 1 \]

Remember NFE is just LFR + 1 and LAF is just NFE + RWS - 1.

Green sequence numbers are in window and grey are outside.

CASE 2: Sequence Number Wrap Around
\[ NFE + RWS - 1 < NFE \]
\[ NFE + RWS - 1 < NFE \land (\text{seqNo} \geq NFE \lor \text{seqNo} \leq NFE + RWS - 1) \]

Receiver with RWS = 4, sequence number in [0,7]
Circular Sender/Receiver Window

• Implement send and receive queue as circular array or list
• Index in to sender’s send queue using sequence number % SWS
• Index in to receiver’s receive queue using sequence number % RWS
• Use P&D as reference
Sequence Number Wrap Around

• You should NOT use more than 8 bits (unsigned char) for seq/ack numbers.
• You need to handle sequence number wrap around once the value reaches 255. Your seq/ack number should wrap back to 0.
• How to do this?
• Answer: % modulus