Sender_TopLevel

[Entry]
ctx.Crcflg=true; ctx.prep1stBlk(); ctx.errCnt=0;
ctx.firstCrcBlk=true;
[Exit]

START
[Entry]
[Exit]

ACKNAK
[Entry]
[Exit]

CAN
[Entry]
[Exit]

EOT1
[Entry]
/*ctx.sendByte(
EOT)*/
[Exit]

EOTEOT
[Entry]
[Exit]

onEvent(SER)
[(c==ACK) & ctx.bytesRd]
/ctx.sendBlkPrepNext();
ctx.errCnt=0;
ctx.firstCrcBlk=false;

onEvent(SER)
[(c==NAK || c=='C') & ctx.bytesRd]
/if (c==NAK) {ctx.Crcflg=false;
    ctx.cs1stBlk();
    ctx.firstCrcBlk=false;}
ctx.sendBlkPrepNext();

onEvent(SER)
[c == CAN]

onEvent(SER)
[(c==NAK ||
(c=='C' & ctx.firstCrcBlk)) &
(ctx.errCnt < errB)]
/ctx.resendBlk();
ctx.errCnt++;

onEvent(SER)
[((c == NAK) || (c == 'C')) &
!ctx.bytesRd]
/if (c==NAK) {ctx.firstCrcBlk=false;}
ctx.sendByte(EOT);

onEvent(SER)
[(c==ACK) & !ctx.bytesRd]
/ctx.sendByte(EOT);ctx.errCnt=0;
ctx.firstCrcBlk=false;

onEvent(SER)
[c == CAN]
/ctx.result=
    "RcvCancelled";
/*ctx.clearCan()*/

onEvent(SER)
[c==NAK]
/ctx.sendByte(EOT);

onEvent(SER)
[c==NAK]
/ctx.sendByte(EOT);

onEvent(SER)
[c == ACK]
/ctx.result="1st EOT ACK'd";

onEvent(SER)
[c==ACK]
/ctx.result="Done";

onEvent(SER)
/cerr << "Sender received totally unexpected char #" << c << ": " <<
(char) c << endl;
exit(EXIT_FAILURE);

onEvent(SER)
[c==NAK & (ctx.errCnt >= errB)]
/ctx.can8();
ctx.result="ExcessiveNAKs";

Simplified Sender Statechart
No dropped characters, no timeouts,
    no cancellation via keyboard.
Copyright 2017 Craig Scratchley
wcs AT sfu.ca