```
1  ---------- MODULE AJupiterExtended ----------
   AJupiter extended with JupiterCtx. This is used to show that AJupiter implements XJupiter.
5  EXTENDS JupiterCtx  TODO: To extend AJupiter
6  ----------
7  VARIABLES cbuf, crec, sbuf, srec, cincomingXJ, sincomingXJ

9  commXJ ≜ INSTANCE CSComm WITH Msg ← Seq(Cop),
10                 cincoming ← cincomingXJ, sincoming ← sincomingXJ

12 varsEx ≜ ⟨intVars, ctxVars, cbuf, crec, sbuf, srec, cincomingXJ, sincomingXJ⟩

14 AJMsgEx ≜ [ack : Nat, cop : Cop, oid : Oid]
15 ----------
16 TypeOKEx ≜
17     ∧ TypeOKInt
18     ∧ TypeOKCtx
19     ∧ commXJ!TypeOK
20     ∧ crec ∈ [Client → Nat]
21     ∧ srec ∈ [Client → Nat]
22     ∧ cbuf ∈ [Client → Seq(Cop)]
23     ∧ sbuf ∈ [Client → Seq(Cop)]
24 ----------
25 InitEx ≜
26     ∧ InitInt
27     ∧ InitCtx
28     ∧ commXJ!Init
29     ∧ crec = [c ∈ Client ↦ 0]
30     ∧ srec = [c ∈ Client ↦ 0]
31     ∧ cbuf = [c ∈ Client ↦ ⟨⟩]
32     ∧ sbuf = [c ∈ Client ↦ ⟨⟩]
33 ----------
34 DoOpEx(c, op) ≜
35     LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq[c]], ctx ↦ ds[c]]
36     IN  ∧ crec′ = [crec EXCEPT ![c] = 0]
37         ∧ cbuf′ = [cbuf EXCEPT ![c] = Append(@, cop)]
38         ∧ SetNewAop(c, op)
39         ∧ Comm!CSend([ack ↦ crec[c], cop ↦ cop, oid ↦ cop.oid])
40         ∧ commXJ!CSend(cop)

42 ClientPerformEx(c, m) ≜
43     LET cBuf ≜ cbuf[c]
44         cShiftedBuf ≜ SubSeq(cBuf, m.ack + 1, Len(cBuf))
45         xcop ≜ XformOpOps(COT, m.cop, cShiftedBuf)
46         xcBuf ≜ XformOpsOp(COT, cShiftedBuf, m.cop)
47     IN   ∧ cbuf′ = [cbuf EXCEPT ![c] = xcBuf]
48          ∧ crec′ = [crec EXCEPT ![c] = @ + 1]
```

```
49              ∧ SetNewAop(c, xcop.op)

51  ServerPerformEx(m) ≜
52      LET c ≜ ClientOf(m.cop)
53          cBuf ≜ sbuf[c]
54          cShiftedBuf ≜ SubSeq(cBuf, m.ack + 1, Len(cBuf))
55          xcop ≜ XformOpOps(COT, m.cop, cShiftedBuf)
56           xcBuf ≜ XformOpsOp(COT, cShiftedBuf, m.cop)
57      IN   ∧ srec' = [cl ∈ Client ↦
58                          IF cl = c THEN srec[cl] + 1 ELSE 0]
59           ∧ sbuf' = [cl ∈ Client ↦
60                          IF cl = c THEN xcBuf ELSE Append(sbuf[cl], xcop)]
61           ∧ SetNewAop(Server, xcop.op)
62           ∧ Comm!SSend(c, [cl ∈ Client ↦ [ack ↦ srec[cl], cop ↦ xcop, oid ↦ xcop.oid]])
63           ∧ commXJ!SSendSame(c, xcop)
64  ──────────────────────────────────────────────────────────────────────────
65  DoEx(c) ≜
66        ∧ DoInt(DoOpEx, c)
67        ∧ DoCtx(c)
68        ∧ UNCHANGED ⟨sbuf, srec⟩

70  RevEx(c) ≜
71      ∧ RevInt(ClientPerformEx, c)
72      ∧ RevCtx(c)
73      ∧ commXJ!CRev(c)
74      ∧ UNCHANGED ⟨sbuf, srec⟩

76  SRevEx ≜
77      ∧   SRevInt(ServerPerformEx)
78      ∧   SRevCtx
79      ∧   commXJ!SRev
80      ∧   UNCHANGED ⟨cbuf, crec⟩
81  ──────────────────────────────────────────────────────────────────────────
82  NextEx ≜
83      ∨  ∃ c ∈ Client : DoEx(c) ∨ RevEx(c)
84      ∨  SRevEx

86  FairnessEx ≜
87      WF_varsEx(SRevEx ∨ ∃ c ∈ Client : RevEx(c))

89  SpecEx ≜ InitEx ∧ □[NextEx]_varsEx  ∧ FairnessEx
90  ──────────────────────────────────────────────────────────────────────────
91  QC ≜  Quiescent Consistency
92       Comm!EmptyChannel ⇒ Cardinality(Range(state)) = 1

94  THEOREM SpecEx ⇒ □QC
95  ──────────────────────────────────────────────────────────────────────────
```

\ * Modification History
\ * Last modified *Wed Jan* 02 21:50:37 *CST* 2019 by *hengxin*
\ * Created *Thu Dec* 27 21:15:09 *CST* 2018 by *hengxin*