```
──────────────────── MODULE AJupiterExtended ────────────────────
AJupiter extended with JupiterCtx. This is used to show that AJupiter implements XJupiter.
EXTENDS JupiterCtx, BufferStateSpace   TODO: To extend AJupiter
─────────────────────────────────────────────────────────────────
VARIABLES cbuf, crec, sbuf, srec, cincomingXJ, sincomingXJ
varsEx ≜ ⟨intVars, ctxVars, cbuf, crec, sbuf, srec, cincomingXJ, sincomingXJ⟩

AJMsgEx ≜ [ack : Nat, cop : Cop, oid : Oid]
commXJ ≜ INSTANCE CSComm WITH Msg ← Seq(Cop),
                    cincoming ← cincomingXJ, sincoming ← sincomingXJ
─────────────────────────────────────────────────────────────────
TypeOKEx ≜
    ∧ TypeOKInt
    ∧ TypeOKCtx
    ∧ commXJ!TypeOK
    ∧ crec ∈ [Client → Nat]
    ∧ srec ∈ [Client → Nat]
    ∧ cbuf ∈ [Client → Seq(Cop)]
    ∧ sbuf ∈ [Client → Seq(Cop)]
─────────────────────────────────────────────────────────────────
InitEx ≜
    ∧ InitInt
    ∧ InitCtx
    ∧ commXJ!Init
    ∧ crec = [c ∈ Client ↦ 0]
    ∧ srec = [c ∈ Client ↦ 0]
    ∧ cbuf = [c ∈ Client ↦ ⟨⟩]
    ∧ sbuf = [c ∈ Client ↦ ⟨⟩]
─────────────────────────────────────────────────────────────────
DoOpEx(c, op) ≜
    LET cop ≜ [op ↦ op, oid ↦ [c ↦ c, seq ↦ cseq[c]], ctx ↦ ds[c]]
    IN  ∧ crec′ = [crec EXCEPT ![c] = 0]
        ∧ cbuf′ = [cbuf EXCEPT ![c] = Append(@, cop)]
        ∧ SetNewAop(c, op)
        ∧ Comm!CSend([ack ↦ crec[c], cop ↦ cop, oid ↦ cop.oid])
        ∧ commXJ!CSend(cop)

ClientPerformEx(c, m) ≜
    LET xform ≜ xFormShift(COT, m.cop, cbuf[c], m.ack)
    IN  ∧ cbuf′ = [cbuf EXCEPT ![c] = xform.xops]
        ∧ crec′ = [crec EXCEPT ![c] = @ + 1]
        ∧ SetNewAop(c, xform.xop.op)

ServerPerformEx(m) ≜
    LET    c    ≜ ClientOf(m.cop)
```

```
          xform ≜ xFormShift(COT, m.cop, sbuf[c], m.ack)
           xcop ≜ xform.xop
    IN    ∧ srec′ = [cl ∈ Client ↦ IF cl = c THEN srec[cl] + 1 ELSE 0]
          ∧ sbuf′ = [cl ∈ Client ↦ IF cl = c THEN xform.xops
                                             ELSE Append(sbuf[cl], xcop)]
          ∧ SetNewAop(Server, xcop.op)
          ∧ Comm!SSend(c, [cl ∈ Client ↦
                              [ack ↦ srec[cl], cop ↦ xcop, oid ↦ xcop.oid]])
          ∧ commXJ!SSendSame(c, xcop)

DoEx(c) ≜
      ∧ DoInt(DoOpEx, c)
      ∧ DoCtx(c)
      ∧ UNCHANGED ⟨sbuf, srec⟩

RevEx(c) ≜
    ∧ RevInt(ClientPerformEx, c)
    ∧ RevCtx(c)
    ∧ commXJ!CRev(c)
    ∧ UNCHANGED ⟨sbuf, srec⟩

SRevEx ≜
    ∧  SRevInt(ServerPerformEx)
    ∧  SRevCtx
    ∧  commXJ!SRev
    ∧  UNCHANGED ⟨cbuf, crec⟩

NextEx ≜
    ∨  ∃ c ∈ Client : DoEx(c) ∨ RevEx(c)
    ∨  SRevEx

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

SpecEx ≜ InitEx ∧ □[NextEx]_varsEx  ∧ FairnessEx

QC ≜  Quiescent Consistency
     Comm!EmptyChannel ⇒ Cardinality(Range(state)) = 1

THEOREM SpecEx ⇒ □QC

\* Modification History
\* Last modified Thu Jan 17 10:38:50 CST 2019 by anonymous
\* Created Thu Dec 27 21:15:09 CST 2018 by anonymous
```