VERIFICATION OF THE STENNING PROTOCOL

Benedetto L. DiVito

Technical Report 26 August 1982

Institute for Computing Science
The University of Texas at Austin
Austin, Texas 78712

This report contains the transcript of a mechanical verification of the Stenning protocol [3]. A description of this protocol, as well as complete documentation on the methods used, can be found in a separate report [2]. Reference to this report is necessary, since the following material is not self-contained. The transcripts themselves were produced by the Boyer-Moore theorem prover [1].

```
VC Proof Log                                          27-Jun-82 09:54:20


++++++++++++++++++++++++++++++++++++++++

Proof of VC  'TRANSPORT#1'
     (IMPLIES (AND (SENDER.EXT SOURCE ACK.IN PKT.OUT)
                   (RECEIVER.EXT PKT.IN SINK ACK.OUT)
                   (FOLLOWS PKT.IN PKT.OUT)
                   (FOLLOWS ACK.IN ACK.OUT))
              (INITIAL SINK SOURCE))

This formula can be simplified, using the abbreviations SENDER.EXT,
AND, and IMPLIES, to:

      (IMPLIES
        (AND (CONSISTENT (QUOTE SEQNO)
                         (QUOTE EQUAL)
                         PKT.OUT)
             (EQUAL SOURCE
                    (FAPPLY (QUOTE MSSG)
                            (RANGE (LATEST (QUOTE SEQNO) PKT.OUT))))
             (RECEIVER.EXT PKT.IN SINK ACK.OUT)
             (FOLLOWS PKT.IN PKT.OUT)
             (FOLLOWS ACK.IN ACK.OUT))
        (INITIAL SINK SOURCE)),

which we simplify, applying CONSISTENT.FOLLOWS, INITIAL.FAPPLY,
INITIAL.CONSEC.FOLLOWS, NUMBERP.SEQNO, FOLLOWS.LATEST.CONSISTENT,
PMAPP.LATEST, INITIAL.RANGE, and INITIAL.TRANS, and opening up
RECEIVER.EXT and INITIAL, to:

      T.

Q.E.D.

13069 conses
9.963 seconds
0.0 seconds, garbage collection time


----------------------------------------



VC Proof Log                                          27-Jun-82 09:55:11


++++++++++++++++++++++++++++++++++++++++


Proof of VC  'SENDER#1'

     (SENDER.INT (NULL)
                 (NULL)
                 (NULL)
                 (QUOTE IDLE)
```

```
                0 0
                (NULL)
                0)
```

This formula can be simplified, using the abbreviation SENDER.INT,
to the following six new conjectures:

Case 6. (PMAPP (QUOTE (1QUOTE NULL))),

  which simplifies, opening up PMAPP, to:

        T.

Case 5. (NUMBERP 0).

  This simplifies, clearly, to:

        T.

Case 4. (FOLLOWS (RANGE (QUOTE (1QUOTE NULL)))
                 (QUOTE (1QUOTE NULL))),

  which simplifies, opening up the definitions of RANGE and FOLLOWS,
  to:

        T.

Case 3. (CONSISTENT (QUOTE SEQNO)
                    (QUOTE EQUAL)
                    (QUOTE (1QUOTE NULL))),

  which we simplify, opening up SEQP and CONSISTENT, to:

        T.

Case 2. (EQUAL (QUOTE (1QUOTE NULL))
               (FAPPLY (QUOTE MSSG)
                       (RANGE (LATEST (QUOTE SEQNO)
                                      (QUOTE (1QUOTE NULL)))))).

  This simplifies, opening up the definitions of SEQP, LATEST,
  RANGE, FAPPLY, and EQUAL, to:

        T.

Case 1. (IF
         (EQUAL 0 0)
         (IF (EQUAL (QUOTE (1QUOTE NULL))
                    (QUOTE (1QUOTE NULL)))
             (EQUAL (QUOTE (1QUOTE NULL))
                    (QUOTE (1QUOTE NULL)))
             F)
         (IF (SEQP (QUOTE (1QUOTE NULL)))
             (IF (EQUAL (DOM (LST (LATEST (QUOTE SEQNO)
                                          (QUOTE (1QUOTE NULL)))))
                        (SUB1 0))
                 (IF (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO)
```
```

```
                                          (QUOTE (1QUOTE NULL))))
                          (SUB1 0))
                    (IF (SEQP (QUOTE (1QUOTE NULL)))
                        (EQUAL (DOM (LST (QUOTE (1QUOTE NULL))))
                               (SUB1 0))
                        T)
                    F)
                F)
            F)).
```

This simplifies, opening up the function EQUAL, to:

```
       T.
```

Q.E.D.

1052 conses
2.57 seconds
0.0 seconds, garbage collection time

++

Proof of VC 'SENDER#2'

```
     (IMPLIES (SENDER.INT SOURCE ACK.IN PKT.OUT STATE UNACK NEXT
                          QUEUE TOT)
              (SENDER.EXT SOURCE ACK.IN PKT.OUT))
```

This conjecture can be simplified, using the abbreviations SENDER.INT and IMPLIES, to:

```
$     (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (FOLLOWS (RANGE QUEUE) PKT.OUT)
        (CONSISTENT (QUOTE SEQNO)
                    (QUOTE EQUAL)
                    PKT.OUT)
        (EQUAL SOURCE
               (FAPPLY (QUOTE MSSG)
                       (RANGE (LATEST (QUOTE SEQNO) PKT.OUT))))
        (IF
         (EQUAL NEXT 0)
         (IF (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
             (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
             F)
         (IF (SEQP PKT.OUT)
             (IF (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                        (SUB1 NEXT))
```

```
                (IF (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                           (SUB1 NEXT))
                    (IF (SEQP QUEUE)
                        (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                        T)
                    F)
                F)
            F)))
       (SENDER.EXT SOURCE ACK.IN PKT.OUT)),
```

which simplifies, opening up SENDER.EXT, LATEST, RANGE, SEQP, FAPPLY, EQUAL, and CONSISTENT, to:

```
      T.
```

Q.E.D.

15990 conses
13.576 seconds
3.261 seconds, garbage collection time

++

Proof of VC 'SENDER#3'

```
     (IMPLIES (AND (SENDER.INT SOURCE ACK.IN PKT.OUT STATE UNACK
                               NEXT QUEUE TOT)
                   (NUMBERP ACK)
                   (LESSP UNACK ACK))
              (SENDER.INT SOURCE
                          (APR ACK.IN ACK)
                          PKT.OUT STATE UNACK NEXT
                          (UPPER QUEUE ACK)
                          TOT))
```

This conjecture can be simplified, using the abbreviations SENDER.INT, AND, and IMPLIES, to:

```
      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (FOLLOWS (RANGE QUEUE) PKT.OUT)
        (CONSISTENT (QUOTE SEQNO)
                    (QUOTE EQUAL)
                    PKT.OUT)
        (EQUAL SOURCE
               (FAPPLY (QUOTE MSSG)
                       (RANGE (LATEST (QUOTE SEQNO) PKT.OUT))))
        (IF
```

```
          (EQUAL NEXT 0)
          (IF (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
              (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
              F)
          (IF (SEQP PKT.OUT)
              (IF (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                         (SUB1 NEXT))
                  (IF (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                             (SUB1 NEXT))
                      (IF (SEQP QUEUE)
                          (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                          T)
                      F)
                  F)
              F))
           (NUMBERP ACK)
           (LESSP UNACK ACK))
         (SENDER.INT SOURCE
                     (APR ACK.IN ACK)
                     PKT.OUT STATE UNACK NEXT
                     (UPPER QUEUE ACK)
                     TOT)),
```

which simplifies, rewriting with the lemmas LST.UPPER, FOLLOWS.RANGE.UPPER, and PMAPP.UPPER, and expanding the functions UPPER, FOLLOWS, SEQP, RANGE, PMAPP, SENDER.INT, LATEST, FAPPLY, EQUAL, and CONSISTENT, to:

```
      T.
```

Q.E.D.

```
24249 conses
20.704 seconds
0.0 seconds, garbage collection time
```

++

Proof of VC 'SENDER#4'

```
     (IMPLIES (AND (SENDER.INT SOURCE ACK.IN PKT.OUT STATE UNACK
                               NEXT QUEUE TOT)
                   (NUMBERP ACK)
                   (EQUAL ACK NEXT))
              (SENDER.INT SOURCE
                          (APR ACK.IN ACK)
                          PKT.OUT
                          (QUOTE IDLE)
                          ACK NEXT
                          (NULL)
```

```
                        TOT))
```

This conjecture can be simplified, using the abbreviations SENDER.INT, AND, and IMPLIES, to:$

```
      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (FOLLOWS (RANGE QUEUE) PKT.OUT)
        (CONSISTENT (QUOTE SEQNO)
                    (QUOTE EQUAL)
                    PKT.OUT)
        (EQUAL SOURCE
               (FAPPLY (QUOTE MSSG)
                       (RANGE (LATEST (QUOTE SEQNO) PKT.OUT))))
        (IF
         (EQUAL NEXT 0)
         (IF (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
             (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
             F)
         (IF (SEQP PKT.OUT)
             (IF (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                        (SUB1 NEXT))
                 (IF (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                            (SUB1 NEXT))
                     (IF (SEQP QUEUE)
                         (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                         T)
                     F)
                 F)
             F))
        (NUMBERP ACK)
        (EQUAL ACK NEXT))
       (SENDER.INT SOURCE
                   (APR ACK.IN ACK)
                   PKT.OUT
                   (QUOTE IDLE)
                   ACK NEXT
                   (QUOTE (1QUOTE NULL))
                   TOT)),
```

which simplifies, opening up FOLLOWS, SEQP, RANGE, PMAPP, SENDER.INT, LATEST, FAPPLY, EQUAL, and CONSISTENT, to:

```
      T.
```

Q.E.D.

```
21684 conses
18.532 seconds
3.342 seconds, garbage collection time
```

```
++++++++++++++++++++++++++++++++++++++++

Proof of VC  'SENDER#5'

     (IMPLIES (SENDER.INT SOURCE ACK.IN PKT.OUT STATE UNACK NEXT
                          QUEUE TOT)
              (SENDER.INT SOURCE ACK.IN
                          (JOIN PKT.OUT (RANGE QUEUE))
                          STATE UNACK NEXT QUEUE TOT))

This conjecture can be simplified, using the abbreviations
SENDER.INT and IMPLIES, to:$

      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (FOLLOWS (RANGE QUEUE) PKT.OUT)
        (CONSISTENT (QUOTE SEQNO)
                    (QUOTE EQUAL)
                    PKT.OUT)
        (EQUAL SOURCE
               (FAPPLY (QUOTE MSSG)
                       (RANGE (LATEST (QUOTE SEQNO) PKT.OUT))))
        (IF
         (EQUAL NEXT 0)
         (IF (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
             (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
             F)
         (IF (SEQP PKT.OUT)
             (IF (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                        (SUB1 NEXT))
                 (IF (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                            (SUB1 NEXT))
                     (IF (SEQP QUEUE)
                         (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                         T)
                     F)
                 F)
             F)))
       (SENDER.INT SOURCE ACK.IN
                   (JOIN PKT.OUT (RANGE QUEUE))
                   STATE UNACK NEXT QUEUE TOT)),

which simplifies, rewriting with the lemmas HIGHEST.JOIN.FOLLOWS,
PISEQP.FAPPLY, FOLLOWS.FAPPLY, FAPPLY.JOIN, LATEST.JOIN.FOLLOWS,
NUMBERP.SEQNO, CONSISTENT.JOIN.FOLLOWS, FOLLOWS.JOIN.2, and
PMAPP.NLST, and expanding the functions RANGE, SEQP, JOIN, FOLLOWS,
PMAPP, SENDER.INT, LATEST, FAPPLY, EQUAL, and CONSISTENT, to six
new conjectures:

Case 6. (IMPLIES
               (AND (PMAPP QUEUE)
                    (NUMBERP NEXT)
                    (FOLLOWS (RANGE QUEUE) PKT.OUT)
                    (CONSISTENT (QUOTE SEQNO)
                                (QUOTE EQUAL)
```

```
                        PKT.OUT)
                  (NOT (EQUAL NEXT 0))
                  (SEQP PKT.OUT)
                  (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                         (SUB1 NEXT))
                  (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                         (SUB1 NEXT))
                  (NOT (SEQP QUEUE)))
            (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which again simplifies, opening up the definition of PMAPP, to:

```
      T.
```

```
Case 5. (IMPLIES
            (AND (PMAPP QUEUE)
                 (NUMBERP NEXT)
                 (FOLLOWS (RANGE QUEUE) PKT.OUT)
                 (CONSISTENT (QUOTE SEQNO)
                             (QUOTE EQUAL)
                             PKT.OUT)
                 (NOT (EQUAL NEXT 0))
                 (SEQP PKT.OUT)
                 (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                        (SUB1 NEXT))
                 (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                        (SUB1 NEXT))
                 (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                 (NOT (SEQP QUEUE)))
            (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which we again simplify, opening up PMAPP, to:

```
      T.
```

```
Case 4. (IMPLIES
            (AND (PMAPP QUEUE)
                 (NUMBERP NEXT)
                 (FOLLOWS (RANGE QUEUE) PKT.OUT)
                 (CONSISTENT (QUOTE SEQNO)
                             (QUOTE EQUAL)
                             PKT.OUT)
                 (NOT (EQUAL NEXT 0))
                 (SEQP PKT.OUT)
                 (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                        (SUB1 NEXT))
                 (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                        (SUB1 NEXT))
                 (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                 (SEQP QUEUE)
                 (SEQP (NLST QUEUE)))
            (MPAIRP (LST (NLST QUEUE)))).
```

However this simplifies again, expanding PMAPP, to:

```
      T.
```

```
Case 3. (IMPLIES
```

```
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                   (SEQP QUEUE)
                   (SEQP (NLST QUEUE)))
              (LESSP (DOM (LST (NLST QUEUE)))
                     (DOM (LST QUEUE)))).
```

But this simplifies again, unfolding the definition of PMAPP, to:

```
       T.
```

```
Case 2. (IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                   (SEQP QUEUE)
                   (NOT (SEQP (NLST QUEUE))))
              (EQUAL (NLST QUEUE)
                     (QUOTE (1QUOTE NULL)))).
```

This simplifies again, unfolding the function PMAPP, to:

```
       T.
```

```
Case 1. (IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
```

```
                        (SUB1 NEXT))
                    (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                    (SEQP QUEUE))
                (MPAIRP (LST QUEUE))).
```

However this simplifies again, unfolding the function PMAPP, to:

```
       T.
```

Q.E.D.

```
26435 conses
24.43 seconds
3.375 seconds, garbage collection time
```

++

Proof of VC 'SENDER#6'

```
     (IMPLIES (SENDER.INT SOURCE ACK.IN PKT.OUT STATE UNACK NEXT
                          QUEUE TOT)
              (SENDER.INT (APR SOURCE MESS)
                          ACK.IN
                          (APR PKT.OUT (PACKET MESS NEXT))
                          (QUOTE BUSY)
                          UNACK
                          (ADD1 NEXT)
                          (WITHE QUEUE NEXT (PACKET MESS NEXT))
                          (PLUS TOT DELTA)))
```

This conjecture can be simplified, using the abbreviations SENDER.INT and IMPLIES, to:$

```
      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (FOLLOWS (RANGE QUEUE) PKT.OUT)
        (CONSISTENT (QUOTE SEQNO)
                    (QUOTE EQUAL)
                    PKT.OUT)
        (EQUAL SOURCE
               (FAPPLY (QUOTE MSSG)
                       (RANGE (LATEST (QUOTE SEQNO) PKT.OUT))))
        (IF
         (EQUAL NEXT 0)
         (IF (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
             (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
             F)
         (IF (SEQP PKT.OUT)
```

```
            (IF (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                       (SUB1 NEXT))
                (IF (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                           (SUB1 NEXT))
                    (IF (SEQP QUEUE)
                        (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                        T)
                    F)
                F)
            F)))
      (SENDER.INT (APR SOURCE MESS)
                  ACK.IN
                  (APR PKT.OUT (PACKET MESS NEXT))
                  (QUOTE BUSY)
                  UNACK
                  (ADD1 NEXT)
                  (WITHE QUEUE NEXT (PACKET MESS NEXT))
                  (PLUS TOT DELTA))).
```

This simplifies, applying the lemmas LST.NSEQP, WITHE.LESSP.DOM.LST, DOM.MPAIR, SUB1.ADD1, LST.WITHE, PMAPP.LATEST, APPLY2.EQUAL, SEQNO.PACKET, APPLY1.SEQNO, FOLLOWS.APR.IN, FOLLOWS.APR, FOLLOWS.SAME, FOLLOWS.TRANS, RNG.MPAIR, LST.APR, NLST.APR, PMAPP.WITHE, WITHE.EQUAL.DOM.LST, NLST.NSEQP, and CONSISTENT.APR.NOT.IN, and opening up LESSP, EQUAL, DOM, HIGHEST, MAX, FAPPLY, LATEST, CONSISTENT, CONSISTENT2, IN, RANGE, PMAPP, SENDER.INT, SEQP, and FOLLOWS, to the following 23 new goals:

```
Case 23.(IMPLIES
             (AND (PMAPP QUEUE)
                  (NUMBERP NEXT)
                  (FOLLOWS (RANGE QUEUE) PKT.OUT)
                  (CONSISTENT (QUOTE SEQNO)
                              (QUOTE EQUAL)
                              PKT.OUT)
                  (NOT (EQUAL NEXT 0))
                  (SEQP PKT.OUT)
                  (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                         (SUB1 NEXT))
                  (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                         (SUB1 NEXT))
                  (NOT (SEQP QUEUE)))
             (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which we again simplify, expanding the definition of PMAPP, to:

```
       T.
```

```
Case 22.(IMPLIES
             (AND (PMAPP QUEUE)
                  (NUMBERP NEXT)
                  (FOLLOWS (RANGE QUEUE) PKT.OUT)
                  (CONSISTENT (QUOTE SEQNO)
                              (QUOTE EQUAL)
                              PKT.OUT)
                  (NOT (EQUAL NEXT 0))
                  (SEQP PKT.OUT)
```

```
                (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                       (SUB1 NEXT))
                (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                       (SUB1 NEXT))
                (NOT (SEQP QUEUE)))
           (CONSISTENT2 (QUOTE SEQNO)
                        (QUOTE EQUAL)
                        PKT.OUT
                        (PACKET MESS NEXT))),
```

which again simplifies, using linear arithmetic, applying SEQNO.PACKET, APPLY1.SEQNO, NUMBERP.SEQNO, and CONSISTENT2.LESSP.HIGHEST, and opening up the functions PMAPP, RANGE, SEQP, and FOLLOWS, to:

T.$

```
Case 21.(IMPLIES
         (AND (PMAPP QUEUE)
              (NUMBERP NEXT)
              (FOLLOWS (RANGE QUEUE) PKT.OUT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.OUT)
              (NOT (EQUAL NEXT 0))
              (SEQP PKT.OUT)
              (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                     (SUB1 NEXT))
              (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                     (SUB1 NEXT))
              (NOT (SEQP QUEUE)))
         (EQUAL (APR (FAPPLY (QUOTE MSSG)
                             (RANGE (LATEST (QUOTE SEQNO) PKT.OUT)))
                     MESS)
                (FAPPLY (QUOTE MSSG)
                        (RANGE (WITHE (LATEST (QUOTE SEQNO) PKT.OUT)
                                      NEXT
                                      (PACKET MESS NEXT)))))).
```

This again simplifies, using linear arithmetic, applying the lemmas PMAPP.LATEST, WITHE.LESSP.DOM.LST, RNG.MPAIR, LST.APR, NLST.APR, MSSG.PACKET, and APPLY1.MSSG, and unfolding PMAPP, RANGE, SEQP, FOLLOWS, and FAPPLY, to:

T.

```
Case 20.(IMPLIES
         (AND
           (PMAPP QUEUE)
           (NUMBERP NEXT)
           (FOLLOWS (RANGE QUEUE) PKT.OUT)
           (CONSISTENT (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.OUT)
           (NOT (EQUAL NEXT 0))
           (SEQP PKT.OUT)
           (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                  (SUB1 NEXT))
```

```
           (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                  (SUB1 NEXT))
           (NOT (SEQP QUEUE))
           (NOT (LESSP NEXT
                       (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT))))))
         (EQUAL (DOM (MPAIR NEXT (PACKET MESS NEXT)))
                NEXT)),
```

which we again simplify, rewriting with DOM.MPAIR, and expanding the definitions of PMAPP, RANGE, SEQP, and FOLLOWS, to:

```
       T.

Case 19.(IMPLIES
            (AND (PMAPP QUEUE)
                 (NUMBERP NEXT)
                 (FOLLOWS (RANGE QUEUE) PKT.OUT)
                 (CONSISTENT (QUOTE SEQNO)
                             (QUOTE EQUAL)
                             PKT.OUT)
                 (NOT (EQUAL NEXT 0))
                 (SEQP PKT.OUT)
                 (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                        (SUB1 NEXT))
                 (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                        (SUB1 NEXT))
                 (NOT (SEQP QUEUE))
                 (LESSP NEXT
                        (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))))
            (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                   NEXT)).
```

But this again simplifies, using linear arithmetic, to:

```
       T.

Case 18.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (NOT (SEQP QUEUE))
                   (LESSP (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          NEXT))
              (EQUAL NEXT NEXT)),
```

which again simplifies, using linear arithmetic, to:

```
       T.
```

```
Case 17.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (NOT (SEQP QUEUE))
                   (NOT (SEQP (FAPPLY (QUOTE SEQNO) PKT.OUT))))
              (EQUAL NEXT NEXT)).
```

This again simplifies, using linear arithmetic, to:

```
       T.
```

```
Case 16.(IMPLIES
           (AND (PMAPP QUEUE)
                (NUMBERP NEXT)
                (FOLLOWS (RANGE QUEUE) PKT.OUT)
                (CONSISTENT (QUOTE SEQNO)
                            (QUOTE EQUAL)
                            PKT.OUT)
                (NOT (EQUAL NEXT 0))
                (SEQP PKT.OUT)
                (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                       (SUB1 NEXT))
                (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                       (SUB1 NEXT))
                (NOT (SEQP QUEUE))
                (SEQP (FAPPLY (QUOTE SEQNO) PKT.OUT))
                (NOT (LESSP (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                            NEXT)))
           (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                  NEXT)),
```

which we again simplify, using linear arithmetic, to:

```
       T.
```

```
Case 15.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
```

```
                        (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT)))
              (FOLLOWS (RANGE (WITHE QUEUE NEXT (PACKET MESS NEXT)))
                       (APR PKT.OUT (PACKET MESS NEXT)))),
```

which again simplifies, using linear arithmetic, applying WITHE.LESSP.DOM.LST, RNG.MPAIR, LST.APR, NLST.APR, FOLLOWS.TRANS, FOLLOWS.SAME, FOLLOWS.APR, and FOLLOWS.APR.IN, and expanding RANGE and IN, to:

```
      T.

Case 14.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT)))
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.OUT
                           (PACKET MESS NEXT))).
```

This again simplifies, using linear arithmetic and applying SEQNO.PACKET, APPLY1.SEQNO, NUMBERP.SEQNO, and CONSISTENT2.LESSP.HIGHEST, to:

```
      T.$

Case 13.(IMPLIES
         (AND (PMAPP QUEUE)
              (NUMBERP NEXT)
              (FOLLOWS (RANGE QUEUE) PKT.OUT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.OUT)
              (NOT (EQUAL NEXT 0))
              (SEQP PKT.OUT)
              (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                     (SUB1 NEXT))
              (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                     (SUB1 NEXT))
              (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT)))
         (EQUAL (APR (FAPPLY (QUOTE MSSG)
                             (RANGE (LATEST (QUOTE SEQNO) PKT.OUT)))
                     MESS)
                (FAPPLY (QUOTE MSSG)
                        (RANGE (WITHE (LATEST (QUOTE SEQNO) PKT.OUT)
                                      NEXT
```

```
                                    (PACKET MESS NEXT))))))).
```

However this simplifies again, using linear arithmetic, applying PMAPP.LATEST, WITHE.LESSP.DOM.LST, RNG.MPAIR, LST.APR, NLST.APR, MSSG.PACKET, and APPLY1.MSSG, and expanding the functions RANGE and FAPPLY, to:

```
       T.

Case 12.(IMPLIES
         (AND
           (PMAPP QUEUE)
           (NUMBERP NEXT)
           (FOLLOWS (RANGE QUEUE) PKT.OUT)
           (CONSISTENT (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.OUT)
           (NOT (EQUAL NEXT 0))
           (SEQP PKT.OUT)
           (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                  (SUB1 NEXT))
           (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                  (SUB1 NEXT))
           (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
           (NOT (LESSP NEXT
                       (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT))))))
         (EQUAL (DOM (MPAIR NEXT (PACKET MESS NEXT)))
                NEXT)).
```

This simplifies again, rewriting with DOM.MPAIR, to:

```
       T.

Case 11.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                   (NOT (LESSP NEXT (DOM (LST QUEUE)))))
              (EQUAL (DOM (MPAIR NEXT (PACKET MESS NEXT)))
                     NEXT)).
```

This simplifies again, applying DOM.MPAIR, to:

```
       T.

Case 10.(IMPLIES
            (AND (PMAPP QUEUE)
```

```
                (NUMBERP NEXT)
                (FOLLOWS (RANGE QUEUE) PKT.OUT)
                (CONSISTENT (QUOTE SEQNO)
                            (QUOTE EQUAL)
                            PKT.OUT)
                (NOT (EQUAL NEXT 0))
                (SEQP PKT.OUT)
                (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                       (SUB1 NEXT))
                (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                       (SUB1 NEXT))
                (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                (LESSP NEXT
                       (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))))
           (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                  NEXT)),
```

which again simplifies, using linear arithmetic, to:

```
       T.

Case 9. (IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                   (LESSP (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          NEXT))
              (EQUAL NEXT NEXT)),
```

which again simplifies, using linear arithmetic, to:

```
       T.

Case 8. (IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                   (NOT (SEQP (FAPPLY (QUOTE SEQNO) PKT.OUT))))
```

```
              (EQUAL NEXT NEXT)).
```

This again simplifies, using linear arithmetic, to:

```
       T.

Case 7. (IMPLIES
           (AND (PMAPP QUEUE)
                (NUMBERP NEXT)
                (FOLLOWS (RANGE QUEUE) PKT.OUT)
                (CONSISTENT (QUOTE SEQNO)
                            (QUOTE EQUAL)
                            PKT.OUT)
                (NOT (EQUAL NEXT 0))
                (SEQP PKT.OUT)
                (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                       (SUB1 NEXT))
                (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                       (SUB1 NEXT))
                (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                (SEQP (FAPPLY (QUOTE SEQNO) PKT.OUT))
                (NOT (LESSP (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                            NEXT)))
           (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                  NEXT)),
```

which again simplifies, using linear arithmetic, to:

```
       T.

Case 6. (IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (NOT (EQUAL NEXT 0))
                   (SEQP PKT.OUT)
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          (SUB1 NEXT))
                   (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                          (SUB1 NEXT))
                   (EQUAL (DOM (LST QUEUE)) (SUB1 NEXT))
                   (LESSP NEXT (DOM (LST QUEUE))))
              (EQUAL (DOM (LST QUEUE)) NEXT)),
```

which again simplifies, using linear arithmetic, to:

```
       T.

Case 5. (IMPLIES
         (AND (PMAPP QUEUE)
              (NUMBERP NEXT)
              (FOLLOWS (RANGE QUEUE) PKT.OUT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
```

```
                    PKT.OUT)
              (EQUAL NEXT 0)
              (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
              (EQUAL QUEUE (QUOTE (1QUOTE NULL))))
         (EQUAL (APR (QUOTE (1QUOTE NULL)) MESS)
                (FAPPLY (QUOTE MSSG)
                        (RANGE (WITHE (LATEST (QUOTE SEQNO) PKT.OUT)
                                      NEXT
                                      (PACKET MESS NEXT))))))).
```

But this again simplifies, rewriting with WITHE.ZERO.2, RNG.MPAIR, LST.APR, NLST.APR, MSSG.PACKET, APPLY1.MSSG, and APR.EQUAL, and opening up the definitions of PMAPP, NUMBERP, RANGE, FOLLOWS, SEQP, CONSISTENT, LATEST, DOMAIN, IN, JOIN, and FAPPLY, to:

```
       (EQUAL (QUOTE (1QUOTE NULL))
              (FAPPLY (QUOTE MSSG)
                      (QUOTE (1QUOTE NULL)))).
```

This again simplifies, opening up the definitions of SEQP, FAPPLY, and EQUAL, to:

```
       T.
```

```
Case 4. (IMPLIES
         (AND (PMAPP QUEUE)
              (NUMBERP NEXT)
              (FOLLOWS (RANGE QUEUE) PKT.OUT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.OUT)
              (EQUAL NEXT 0)
              (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
              (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
              (NOT (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          0)))
         (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                NEXT)).
```

But this again simplifies, expanding the definitions of PMAPP, NUMBERP, RANGE, FOLLOWS, SEQP, CONSISTENT, LATEST, LST, DOM, and EQUAL, to:

```
       T.
```

```
Case 3. (IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS (RANGE QUEUE) PKT.OUT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.OUT)
                   (EQUAL NEXT 0)
                   (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
                   (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
                   (EQUAL (DOM (LST (LATEST (QUOTE SEQNO) PKT.OUT)))
                          0))
```

```
              (EQUAL (DOM (MPAIR NEXT (PACKET MESS NEXT)))
                     NEXT)).
```

But this simplifies again, applying DOM.MPAIR, and expanding the functions PMAPP, NUMBERP, RANGE, FOLLOWS, SEQP, CONSISTENT, LATEST, LST, DOM, and EQUAL, to:

```
       T.

Case 2. (IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (FOLLOWS (RANGE QUEUE) PKT.OUT)
                      (CONSISTENT (QUOTE SEQNO)
                                  (QUOTE EQUAL)
                                  PKT.OUT)
                      (EQUAL NEXT 0)
                      (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
                      (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
                      (NOT (SEQP (FAPPLY (QUOTE SEQNO) PKT.OUT))))
                 (EQUAL NEXT NEXT)),
```

which again simplifies, using linear arithmetic, to:

```
       T.

Case 1. (IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (FOLLOWS (RANGE QUEUE) PKT.OUT)
                      (CONSISTENT (QUOTE SEQNO)
                                  (QUOTE EQUAL)
                                  PKT.OUT)
                      (EQUAL NEXT 0)
                      (EQUAL PKT.OUT (QUOTE (1QUOTE NULL)))
                      (EQUAL QUEUE (QUOTE (1QUOTE NULL)))
                      (SEQP (FAPPLY (QUOTE SEQNO) PKT.OUT)))
                 (EQUAL (HIGHEST (FAPPLY (QUOTE SEQNO) PKT.OUT))
                        NEXT)),
```

which again simplifies, expanding PMAPP, NUMBERP, RANGE, FOLLOWS, SEQP, CONSISTENT, and FAPPLY, to:

```
       T.

Q.E.D.

101452 conses
94.17 seconds
10.246 seconds, garbage collection time


---------------------------------------

VC Proof Log                                        27-Jun-82 10:06:29


++++++++++++++++++++++++++++++++++++++++

Proof of VC  'RECEIVER#1'
```

```
     (RECEIVER.INT (NULL)
                   (NULL)
                   (NULL)
                   0
                   (NULL))
```

This conjecture can be simplified, using the abbreviation RECEIVER.INT, to three new conjectures:

Case 3. (PMAPP (QUOTE (1QUOTE NULL))),

which simplifies, expanding the definition of PMAPP, to:

T.

Case 2. (NUMBERP 0),

which we simplify, clearly, to:

T.

Case 1.
```
(IMPLIES
 (CONSISTENT (QUOTE SEQNO)
             (QUOTE EQUAL)
             (QUOTE (1QUOTE NULL)))
 (IF
  (FOLLOWS (QUOTE (1QUOTE NULL))
           (UPPER (LATEST (QUOTE SEQNO)
                          (QUOTE (1QUOTE NULL)))
                  1))
  (IF
   (LESSP (REACH (LATEST (QUOTE SEQNO)
                         (QUOTE (1QUOTE NULL))))
          0)
   F
   (IF
    (IN 0
        (DOMAIN (LATEST (QUOTE SEQNO)
                        (QUOTE (1QUOTE NULL)))))
    (IF
     (LESSP 0 0)
     (EQUAL
        (QUOTE (1QUOTE NULL))
        (FAPPLY (QUOTE MSSG)
                (RANGE (LOWER (LATEST (QUOTE SEQNO)
                                      (QUOTE (1QUOTE NULL)))
                              (SUB1 0)))))
     F)
    (IF (EQUAL 0 0)
        (EQUAL (QUOTE (1QUOTE NULL))
               (QUOTE (1QUOTE NULL)))
        F)))
   F)),
```

which simplifies, expanding the functions SEQP, CONSISTENT, LATEST, UPPER, FOLLOWS, REACH, LESSP, DOMAIN, IN, and EQUAL, to:

T.

```
Q.E.D.

1444 conses
4.3 seconds
0.0 seconds, garbage collection time


---------------------------------------




++++++++++++++++++++++++++++++++++++++++

Proof of VC  'RECEIVER#2'

     (IMPLIES (RECEIVER.INT PKT.IN SINK ACK.OUT NEXT QUEUE)
              (RECEIVER.EXT PKT.IN SINK ACK.OUT))

This formula can be simplified, using the abbreviations
RECEIVER.EXT, RECEIVER.INT, and IMPLIES, to:$

      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (IF
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (IF
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (IF
           (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                  NEXT)
           F
           (IF
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (IF
             (LESSP 0 NEXT)
             (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
             F)
            (IF (EQUAL NEXT 0)
                (EQUAL SINK (QUOTE (1QUOTE NULL)))
                F)))
           F)
          T)
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN))
```

```
       (IF
        (IN 0
            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
        (INITIAL SINK
            (FAPPLY (QUOTE MSSG)
                    (RANGE (CONSEC (LATEST (QUOTE SEQNO) PKT.IN)))))
        (EQUAL SINK (QUOTE (1QUOTE NULL)))),
```

which simplifies, expanding the functions EQUAL and LESSP, to:$

```
      (IMPLIES
       (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN))
       (INITIAL SINK
          (FAPPLY (QUOTE MSSG)
                  (RANGE (CONSEC (LATEST (QUOTE SEQNO) PKT.IN))))),
```

which again simplifies, using linear arithmetic and applying
INITIAL.RANGE, PMAPP.LATEST, INITIAL.LOWER.CONSEC, and
INITIAL.FAPPLY, to:

```
      T.

Q.E.D.

42883 conses
35.007 seconds
6.695 seconds, garbage collection time


----------------------------------------




++++++++++++++++++++++++++++++++++++++++

Proof of VC  'RECEIVER#3'

     (IMPLIES (AND (RECEIVER.INT PKT.IN SINK ACK.OUT NEXT QUEUE)
```

```
                (LESSP NEXT (SEQNO PKT))
                (PKTP PKT))
           (RECEIVER.INT (APR PKT.IN PKT)
                         SINK ACK.OUT NEXT QUEUE))
```

This conjecture can be simplified, using the abbreviations RECEIVER.INT, AND, and IMPLIES, to the conjecture:$$

```
      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (IF
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (IF
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (IF
           (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                  NEXT)
           F
           (IF
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (IF
             (LESSP 0 NEXT)
             (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
             F)
            (IF (EQUAL NEXT 0)
                (EQUAL SINK (QUOTE (1QUOTE NULL)))
                F)))
           F)
          T)
         (LESSP NEXT (SEQNO PKT))
         (PKTP PKT))
        (RECEIVER.INT (APR PKT.IN PKT)
                      SINK ACK.OUT NEXT QUEUE)),
```

which we simplify, applying LST.APR, NLST.APR, APPLY2.EQUAL, APPLY1.SEQNO, PMAPP.NLST, IN.DOMAIN.WITHE.IF, SUB1.ADD1, UPPER.WITHE.IF, PMAPP.LATEST, and LOWER.WITHE.IF, and expanding the definitions of EQUAL, LESSP, CONSISTENT, CONSISTENT2, PMAPP, RECEIVER.INT, and LATEST, to 21 new conjectures:

```
Case 21.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP NEXT (SEQNO PKT))
                      (PKTP PKT)
```

```
                    (NOT (SEQP QUEUE)))
               (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which we again simplify, opening up PMAPP, to:

```
       T.

Case 20.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP NEXT (SEQNO PKT))
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (SEQP (NLST QUEUE)))
                 (MPAIRP (LST (NLST QUEUE)))).
```

However this simplifies again, expanding PMAPP, to:

```
       T.
Case 19.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP NEXT (SEQNO PKT))
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (SEQP (NLST QUEUE)))
                 (LESSP (DOM (LST (NLST QUEUE)))
                        (DOM (LST QUEUE)))).
```

But this simplifies again, unfolding the definition of PMAPP, to:

```
       T.

Case 18.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP NEXT (SEQNO PKT))
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (NOT (SEQP (NLST QUEUE))))
                 (EQUAL (NLST QUEUE)
                        (QUOTE (1QUOTE NULL)))).
```

This simplifies again, unfolding the function PMAPP, to:

```
       T.

Case 17.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
```

```
                                    PKT.IN))
                        (LESSP NEXT (SEQNO PKT))
                        (PKTP PKT)
                        (SEQP QUEUE))
                    (MPAIRP (LST QUEUE))).
```

However this simplifies again, unfolding the function PMAPP, to:

```
       T.

Case 16.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS QUEUE
                            (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                   (ADD1 NEXT)))
                   (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                               NEXT))
                   (NOT (IN 0
                            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                   (EQUAL NEXT 0)
                   (EQUAL SINK (QUOTE (1QUOTE NULL)))
                   (NOT (EQUAL (SEQNO PKT) 0))
                   (PKTP PKT)
                   (NOT (SEQP QUEUE)))
              (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which we again simplify, expanding the definition of PMAPP, to:

```
       T.

Case 15.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS QUEUE
                            (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                   (ADD1 NEXT)))
                   (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                               NEXT))
                   (NOT (IN 0
                            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                   (EQUAL NEXT 0)
                   (EQUAL SINK (QUOTE (1QUOTE NULL)))
                   (NOT (EQUAL (SEQNO PKT) 0))
                   (PKTP PKT)
                   (SEQP QUEUE)
                   (SEQP (NLST QUEUE)))
              (MPAIRP (LST (NLST QUEUE)))),
```

which again simplifies, unfolding the definition of PMAPP, to:

```
       T.

Case 14.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS QUEUE
```

```
                        (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                               (ADD1 NEXT)))
                  (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                              NEXT))
                  (NOT (IN 0
                           (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                  (EQUAL NEXT 0)
                  (EQUAL SINK (QUOTE (1QUOTE NULL)))
                  (NOT (EQUAL (SEQNO PKT) 0))
                  (PKTP PKT)
                  (SEQP QUEUE)
                  (SEQP (NLST QUEUE)))
            (LESSP (DOM (LST (NLST QUEUE)))
                   (DOM (LST QUEUE)))).
```

This again simplifies, unfolding the function PMAPP, to:

```
      T.

Case 13.(IMPLIES
            (AND (PMAPP QUEUE)
                 (NUMBERP NEXT)
                 (FOLLOWS QUEUE
                          (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                 (ADD1 NEXT)))
                 (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                             NEXT))
                 (NOT (IN 0
                          (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                 (EQUAL NEXT 0)
                 (EQUAL SINK (QUOTE (1QUOTE NULL)))
                 (NOT (EQUAL (SEQNO PKT) 0))
                 (PKTP PKT)
                 (SEQP QUEUE)
                 (NOT (SEQP (NLST QUEUE))))
            (EQUAL (NLST QUEUE)
                   (QUOTE (1QUOTE NULL)))),
```

which again simplifies, opening up the definition of PMAPP, to:

```
      T.

Case 12.(IMPLIES
            (AND (PMAPP QUEUE)
                 (NUMBERP NEXT)
                 (FOLLOWS QUEUE
                          (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                 (ADD1 NEXT)))
                 (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                             NEXT))
                 (NOT (IN 0
                          (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                 (EQUAL NEXT 0)
                 (EQUAL SINK (QUOTE (1QUOTE NULL)))
                 (NOT (EQUAL (SEQNO PKT) 0))
                 (PKTP PKT)
                 (SEQP QUEUE))
```

```
              (MPAIRP (LST QUEUE))).
```

This again simplifies, expanding the definition of PMAPP, to:

```
       T.$$
Case 11.(IMPLIES
              (AND (PMAPP QUEUE)
                   (NUMBERP NEXT)
                   (FOLLOWS QUEUE
                            (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                   (ADD1 NEXT)))
                   (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                               NEXT))
                   (NOT (IN 0
                            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                   (EQUAL NEXT 0)
                   (EQUAL SINK (QUOTE (1QUOTE NULL)))
                   (NOT (EQUAL (SEQNO PKT) 0))
                   (PKTP PKT)
                   (CONSISTENT2 (QUOTE SEQNO)
                                (QUOTE EQUAL)
                                PKT.IN PKT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.IN)
                   (NOT (LESSP (SUB1 (SEQNO PKT)) NEXT)))
              (FOLLOWS QUEUE
                       (WITHE (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                     (ADD1 NEXT))
                              (SEQNO PKT)
                              PKT))),
```

which again simplifies, applying the lemmas PMAPP.UPPER, PMAPP.LATEST, FOLLOWS.WITHE.IF, IN.DOMAIN.UPPER, IN.UPPER, and DOM.MPAIR, and expanding the definitions of NUMBERP, EQUAL, and LESSP, to:

```
       (IMPLIES
              (AND (PMAPP QUEUE)
                   (FOLLOWS QUEUE
                            (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                   1))
                   (NOT (IN 0
                            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                   (NOT (EQUAL (SEQNO PKT) 0))
                   (PKTP PKT)
                   (CONSISTENT2 (QUOTE SEQNO)
                                (QUOTE EQUAL)
                                PKT.IN PKT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.IN)
                   (NOT (LESSP (SEQNO PKT) 1))
                   (IN (SEQNO PKT)
                       (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
                   (IN (SEQNO PKT) (DOMAIN QUEUE)))
              (IN (MPAIR (SEQNO PKT) PKT)
```

```
                (LATEST (QUOTE SEQNO) PKT.IN))).
```

This simplifies again, applying NUMBERP.SEQNO, APPLY1.SEQNO, and IN.MPAIR.LATEST.IN, to:

```
       T.

Case 10.(IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (NOT (SEQP QUEUE)))
         (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which again simplifies, expanding the function PMAPP, to:

```
       T.

Case 9. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (SEQP QUEUE)
          (SEQP (NLST QUEUE)))
         (MPAIRP (LST (NLST QUEUE)))).
```

However this simplifies again, expanding the definition of PMAPP, to:

```
       T.
```

```
Case 8. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (SEQP QUEUE)
          (SEQP (NLST QUEUE)))
         (LESSP (DOM (LST (NLST QUEUE)))
                (DOM (LST QUEUE)))).
```

This again simplifies, opening up the definition of PMAPP, to:

```
T.$
```

```
Case 7. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (SEQP QUEUE)
          (NOT (SEQP (NLST QUEUE))))
         (EQUAL (NLST QUEUE)
                (QUOTE (1QUOTE NULL)))).
```

But this again simplifies, expanding the definition of PMAPP, to:

```
T.
```

```
Case 6. (IMPLIES
         (AND
          (PMAPP QUEUE)
```

```
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (SEQP QUEUE))
         (MPAIRP (LST QUEUE))).
```

But this simplifies again, expanding the definition of PMAPP, to:

T.

```
Case 5. (IMPLIES
          (AND
           (PMAPP QUEUE)
           (NUMBERP NEXT)
           (FOLLOWS QUEUE
                    (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                           (ADD1 NEXT)))
           (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                       NEXT))
           (IN 0
               (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
           (NOT (EQUAL NEXT 0))
           (EQUAL SINK
                  (FAPPLY (QUOTE MSSG)
                          (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                        (SUB1 NEXT)))))
           (LESSP NEXT (SEQNO PKT))
           (PKTP PKT)
           (CONSISTENT2 (QUOTE SEQNO)
                        (QUOTE EQUAL)
                        PKT.IN PKT)
           (CONSISTENT (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN)
           (EQUAL (SEQNO PKT) (SUB1 NEXT))
           (NOT (IN (MPAIR (SEQNO PKT) PKT)
                    (LATEST (QUOTE SEQNO) PKT.IN)))
           (EQUAL (SUB1 NEXT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (APR (QUOTE (1QUOTE NULL))
                                     (MPAIR 0 PKT))))))),
```

which we again simplify, using linear arithmetic, to:

T.

```
Case 4. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT) (SUB1 NEXT))
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SUB1 NEXT) 0)))
         (EQUAL SINK
            (FAPPLY (QUOTE MSSG)
                    (RANGE (APR (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SUB1 NEXT)))
                                (MPAIR (SEQNO PKT) PKT))))))).
```

But this again simplifies, using linear arithmetic, to:

```
       T.

Case 3. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
```

```
                    (QUOTE EQUAL)
                    PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (NOT (EQUAL (SEQNO PKT) (SUB1 NEXT)))
          (NOT (LESSP (SUB1 NEXT) (SEQNO PKT))))
         (EQUAL SINK
          (FAPPLY (QUOTE MSSG)
                  (RANGE (WITHE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT))
                                (SEQNO PKT)
                                PKT))))).
```

However this simplifies again, using linear arithmetic, to:

T.

```
Case 2. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN))
         (NOT (LESSP (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                   (SEQNO PKT)
                                   PKT))
                     NEXT))),
```

which we again simplify, using linear arithmetic and applying LESSP.REACH.WITHE.2 and PMAPP.LATEST, to:

T.$

```
Case 1. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
```

```
                (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                       (ADD1 NEXT)))
         (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                     NEXT))
         (IN 0
             (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
         (NOT (EQUAL NEXT 0))
         (EQUAL SINK
                (FAPPLY (QUOTE MSSG)
                        (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                      (SUB1 NEXT)))))
         (LESSP NEXT (SEQNO PKT))
         (PKTP PKT)
         (CONSISTENT2 (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN PKT)
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (NOT (EQUAL (SEQNO PKT) 0))
         (NOT (LESSP (SUB1 (SEQNO PKT)) NEXT)))
        (FOLLOWS QUEUE
                 (WITHE (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                               (ADD1 NEXT))
                        (SEQNO PKT)
                        PKT))).
```

But this simplifies again, applying PMAPP.UPPER, PMAPP.LATEST, FOLLOWS.WITHE.IF, IN.DOMAIN.UPPER, SUB1.ADD1, IN.UPPER, DOM.MPAIR, NUMBERP.SEQNO, APPLY1.SEQNO, and IN.MPAIR.LATEST.IN, and expanding the definition of LESSP, to:

```
      T.

Q.E.D.

179171 conses
164.418 seconds
20.28 seconds, garbage collection time


----------------------------------------




++++++++++++++++++++++++++++++++++++++++

Proof of VC  'RECEIVER#4'

     (IMPLIES (AND (RECEIVER.INT PKT.IN SINK ACK.OUT NEXT QUEUE)
                   (LESSP (SEQNO PKT) NEXT)
                   (PKTP PKT))
              (RECEIVER.INT (APR PKT.IN PKT)
                            SINK
                            (APR ACK.OUT NEXT)
```

```
                          NEXT QUEUE))
```

This formula can be simplified, using the abbreviations RECEIVER.INT, AND, and IMPLIES, to:$$

```
      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (IF
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (IF
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (IF
           (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                  NEXT)
           F
           (IF
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (IF
             (LESSP 0 NEXT)
             (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
             F)
            (IF (EQUAL NEXT 0)
                (EQUAL SINK (QUOTE (1QUOTE NULL)))
                F)))
          F)
         T)
        (LESSP (SEQNO PKT) NEXT)
        (PKTP PKT))
       (RECEIVER.INT (APR PKT.IN PKT)
                     SINK
                     (APR ACK.OUT NEXT)
                     NEXT QUEUE)).
```

This simplifies, applying LST.APR, NLST.APR, APPLY2.EQUAL, APPLY1.SEQNO, PMAPP.NLST, LOWER.WITHE.IF, IN.DOMAIN.WITHE.IF, SUB1.ADD1, UPPER.WITHE.IF, and PMAPP.LATEST, and expanding the functions EQUAL, LESSP, CONSISTENT, CONSISTENT2, PMAPP, RECEIVER.INT, and LATEST, to the following 15 new goals:

```
Case 15.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP (SEQNO PKT) NEXT)
                      (PKTP PKT)
                      (NOT (SEQP QUEUE)))
```

```
                  (EQUAL QUEUE (QUOTE (1QUOTE NULL)))).
```

This simplifies again, opening up the definition of PMAPP, to:

```
        T.

Case 14.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP (SEQNO PKT) NEXT)
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (SEQP (NLST QUEUE)))
                 (MPAIRP (LST (NLST QUEUE)))),
```

which again simplifies, opening up the definition of PMAPP, to:

```
        T.

Case 13.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP (SEQNO PKT) NEXT)
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (SEQP (NLST QUEUE)))
                 (LESSP (DOM (LST (NLST QUEUE)))
                        (DOM (LST QUEUE)))),
```

which we again simplify, opening up PMAPP, to:

```
        T.

Case 12.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (LESSP (SEQNO PKT) NEXT)
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (NOT (SEQP (NLST QUEUE))))
                 (EQUAL (NLST QUEUE)
                        (QUOTE (1QUOTE NULL)))).
```

However this simplifies again, expanding PMAPP, to:

```
        T.

Case 11.(IMPLIES (AND (PMAPP QUEUE)
                      (NUMBERP NEXT)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
```

```
                                  PKT.IN))
                    (LESSP (SEQNO PKT) NEXT)
                    (PKTP PKT)
                    (SEQP QUEUE))
                (MPAIRP (LST QUEUE))).
```

But this simplifies again, unfolding the definition of PMAPP, to:

```
      T.
```

```
Case 10.(IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (NOT (SEQP QUEUE)))
         (EQUAL QUEUE (QUOTE (1QUOTE NULL)))).
```

This simplifies again, unfolding the function PMAPP, to:

```
       T.
```

```
Case 9. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (SEQP QUEUE)
          (SEQP (NLST QUEUE)))
         (MPAIRP (LST (NLST QUEUE)))).
```

However this simplifies again, unfolding the function PMAPP, to:

```
        T.

Case 8. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (SEQP QUEUE)
          (SEQP (NLST QUEUE)))
         (LESSP (DOM (LST (NLST QUEUE)))
                (DOM (LST QUEUE)))),
```

which we again simplify, expanding the definition of PMAPP, to:

```
        T.

Case 7. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (SEQP QUEUE)
          (NOT (SEQP (NLST QUEUE))))
         (EQUAL (NLST QUEUE)
                (QUOTE (1QUOTE NULL)))),
```

which again simplifies, unfolding the definition of PMAPP, to:

```
        T.

Case 6. (IMPLIES
```

```
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (SEQP QUEUE))
         (MPAIRP (LST QUEUE))).
```

This again simplifies, unfolding the function PMAPP, to:

```
       T.$

Case 5. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT) (SUB1 NEXT))
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (EQUAL (SUB1 NEXT) 0))
         (EQUAL SINK
                (FAPPLY (QUOTE MSSG)
                        (RANGE (APR (QUOTE (1QUOTE NULL))
                                    (MPAIR 0 PKT))))))),
```

which again simplifies, using linear arithmetic, applying the

```
  lemmas NUMBERP.SEQNO, APPLY1.SEQNO, and IN.MPAIR.LATEST.IN, and
  expanding the definitions of EQUAL and LESSP, to:

        T.

Case 4. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT) (SUB1 NEXT))
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SUB1 NEXT) 0)))
         (EQUAL SINK
            (FAPPLY (QUOTE MSSG)
                    (RANGE (APR (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SUB1 NEXT)))
                                (MPAIR (SEQNO PKT) PKT)))))).

  This simplifies again, using linear arithmetic and applying
  NUMBERP.SEQNO, IN.DOMAIN.REACH, PMAPP.LATEST, APPLY1.SEQNO, and
  IN.MPAIR.LATEST.IN, to:

        T.$

Case 3. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
```

```
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (NOT (EQUAL (SEQNO PKT) (SUB1 NEXT)))
          (NOT (LESSP (SUB1 NEXT) (SEQNO PKT))))
         (EQUAL SINK
          (FAPPLY (QUOTE MSSG)
                  (RANGE (WITHE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT))
                                (SEQNO PKT)
                                PKT)))))).
```

But this again simplifies, using linear arithmetic and applying IN.MPAIR.LATEST.IN, APPLY1.SEQNO, IN.DOMAIN.REACH, NUMBERP.SEQNO, DOM.MPAIR, IN.LOWER, PMAPP.LOWER, PMAPP.LATEST, and WITHE.IN.MPAIR, to:

```
       T.
```

```
Case 2. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN))
         (NOT (LESSP (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                   (SEQNO PKT)
                                   PKT))
                     NEXT))).
```

However this again simplifies, using linear arithmetic and

```
  applying LESSP.REACH.WITHE.2 and PMAPP.LATEST, to:

        T.

Case 1. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP (SEQNO PKT) NEXT)
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (NOT (EQUAL (SEQNO PKT) 0))
          (NOT (LESSP (SUB1 (SEQNO PKT)) NEXT)))
         (FOLLOWS QUEUE
                  (WITHE (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                (ADD1 NEXT))
                         (SEQNO PKT)
                         PKT))),

  which again simplifies, using linear arithmetic, to:

        T.

Q.E.D.

126660 conses
114.513 seconds
13.528 seconds, garbage collection time


---------------------------------------




++++++++++++++++++++++++++++++++++++++++

Proof of VC  'RECEIVER#5'

     (IMPLIES (AND (RECEIVER.INT PKT.IN SINK ACK.OUT NEXT QUEUE)
```

```
                (LESSP NEXT (SEQNO PKT))
                (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
                (PKTP PKT))
            (RECEIVER.INT (APR PKT.IN PKT)
                          SINK ACK.OUT NEXT
                          (WITHE QUEUE (SEQNO PKT) PKT)))
```

This formula can be simplified, using the abbreviations NOT, RECEIVER.INT, AND, and IMPLIES, to:$$$

```
      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (IF
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (IF
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (IF
           (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                  NEXT)
           F
           (IF
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (IF
             (LESSP 0 NEXT)
             (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
             F)
            (IF (EQUAL NEXT 0)
                (EQUAL SINK (QUOTE (1QUOTE NULL)))
                F)))
           F)
          T)
         (LESSP NEXT (SEQNO PKT))
         (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
         (PKTP PKT))
        (RECEIVER.INT (APR PKT.IN PKT)
                      SINK ACK.OUT NEXT
                      (WITHE QUEUE (SEQNO PKT) PKT))),
```

which we simplify, applying LST.APR, NLST.APR, APPLY2.EQUAL, APPLY1.SEQNO, PMAPP.WITHE, IN.DOMAIN.WITHE.IF, SUB1.ADD1, UPPER.WITHE.IF, PMAPP.LATEST, and LOWER.WITHE.IF, and opening up EQUAL, LESSP, CONSISTENT, CONSISTENT2, RECEIVER.INT, and LATEST, to the following nine new goals:

```
Case 9. (IMPLIES
             (AND (PMAPP QUEUE)
                  (NUMBERP NEXT)
```

```
                (FOLLOWS QUEUE
                         (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                (ADD1 NEXT)))
                (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                            NEXT))
                (NOT (IN 0
                         (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                (EQUAL NEXT 0)
                (EQUAL SINK (QUOTE (1QUOTE NULL)))
                (NOT (EQUAL (SEQNO PKT) 0))
                (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
                (PKTP PKT)
                (CONSISTENT2 (QUOTE SEQNO)
                             (QUOTE EQUAL)
                             PKT.IN PKT)
                (CONSISTENT (QUOTE SEQNO)
                            (QUOTE EQUAL)
                            PKT.IN)
                (LESSP (SUB1 (SEQNO PKT)) NEXT))
           (FOLLOWS (WITHE QUEUE (SEQNO PKT) PKT)
                    (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                           (ADD1 NEXT)))),
```

which we again simplify, using linear arithmetic, to:

```
      T.$$
```

```
Case 8. (IMPLIES
           (AND (PMAPP QUEUE)
                (NUMBERP NEXT)
                (FOLLOWS QUEUE
                         (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                (ADD1 NEXT)))
                (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                            NEXT))
                (NOT (IN 0
                         (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                (EQUAL NEXT 0)
                (EQUAL SINK (QUOTE (1QUOTE NULL)))
                (NOT (EQUAL (SEQNO PKT) 0))
                (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
                (PKTP PKT)
                (CONSISTENT2 (QUOTE SEQNO)
                             (QUOTE EQUAL)
                             PKT.IN PKT)
                (CONSISTENT (QUOTE SEQNO)
                            (QUOTE EQUAL)
                            PKT.IN)
                (NOT (LESSP (SUB1 (SEQNO PKT)) NEXT)))
           (FOLLOWS (WITHE QUEUE (SEQNO PKT) PKT)
                    (WITHE (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                  (ADD1 NEXT))
                           (SEQNO PKT)
                           PKT))),
```

which again simplifies, applying the lemmas RNG.MPAIR, DOM.MPAIR, IN.WITHE.MPAIRP, IN.DOMAIN.UPPER, FOLLOWS.WITHE.IF, PMAPP.WITHE,

```
  PMAPP.LATEST, PMAPP.UPPER, and FOLLOWS.WITHE.IN.MPAIR, and
  expanding the functions NUMBERP, EQUAL, and LESSP, to:

        T.

Case 7. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT) (SUB1 NEXT))
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (EQUAL (SUB1 NEXT) 0))
         (EQUAL SINK
                (FAPPLY (QUOTE MSSG)
                        (RANGE (APR (QUOTE (1QUOTE NULL))
                                    (MPAIR 0 PKT))))))),

  which again simplifies, using linear arithmetic, to:

        T.

Case 6. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
```

```
                    (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                  (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT) (SUB1 NEXT))
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SUB1 NEXT) 0)))
         (EQUAL SINK
            (FAPPLY (QUOTE MSSG)
                    (RANGE (APR (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SUB1 NEXT)))
                                (MPAIR (SEQNO PKT) PKT)))))).
```

However this simplifies again, using linear arithmetic, to:

T.

```
Case 5. (IMPLIES
          (AND
           (PMAPP QUEUE)
           (NUMBERP NEXT)
           (FOLLOWS QUEUE
                    (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                           (ADD1 NEXT)))
           (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                       NEXT))
           (IN 0
               (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
           (NOT (EQUAL NEXT 0))
           (EQUAL SINK
                  (FAPPLY (QUOTE MSSG)
                          (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                        (SUB1 NEXT)))))
           (LESSP NEXT (SEQNO PKT))
           (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
           (PKTP PKT)
           (CONSISTENT2 (QUOTE SEQNO)
                        (QUOTE EQUAL)
                        PKT.IN PKT)
           (CONSISTENT (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN)
           (NOT (EQUAL (SEQNO PKT) (SUB1 NEXT)))
           (NOT (LESSP (SUB1 NEXT) (SEQNO PKT))))
          (EQUAL SINK
           (FAPPLY (QUOTE MSSG)
                   (RANGE (WITHE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                        (SUB1 NEXT))
                                 (SEQNO PKT)
```

```
                                PKT))))),
```

which again simplifies, using linear arithmetic, to:

```
       T.

Case 4. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN))
         (NOT (LESSP (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                   (SEQNO PKT)
                                   PKT))
                     NEXT))).
```

But this simplifies again, using linear arithmetic and rewriting with LESSP.REACH.WITHE.2 and PMAPP.LATEST, to:

```
       T.

Case 3. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
```

```
          (LESSP NEXT (SEQNO PKT))
          (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT) 0))
         (FOLLOWS (WITHE QUEUE (SEQNO PKT) PKT)
                  (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                         (ADD1 NEXT)))).
```

This again simplifies, using linear arithmetic, to:

```
       T.

Case 2. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (LESSP (SUB1 (SEQNO PKT)) NEXT))
         (FOLLOWS (WITHE QUEUE (SEQNO PKT) PKT)
                  (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                         (ADD1 NEXT)))),
```

which again simplifies, using linear arithmetic, to:

```
       T.$$
Case 1. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (NUMBERP NEXT)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
```

```
                    (ADD1 NEXT)))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      NEXT))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL NEXT 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
          (LESSP NEXT (SEQNO PKT))
          (NOT (IN (SEQNO PKT) (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (NOT (EQUAL (SEQNO PKT) 0))
          (NOT (LESSP (SUB1 (SEQNO PKT)) NEXT)))
         (FOLLOWS (WITHE QUEUE (SEQNO PKT) PKT)
                  (WITHE (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                (ADD1 NEXT))
                         (SEQNO PKT)
                         PKT))),
```

which again simplifies, rewriting with RNG.MPAIR, DOM.MPAIR, IN.WITHE.MPAIRP, SUB1.ADD1, IN.DOMAIN.UPPER, FOLLOWS.WITHE.IF, PMAPP.WITHE, PMAPP.LATEST, PMAPP.UPPER, and FOLLOWS.WITHE.IN.MPAIR, and expanding LESSP, to:

```
      T.
```

Q.E.D.

```
176501 conses
161.821 seconds
23.681 seconds, garbage collection time


----------------------------------------

++++++++++++++++++++++++++++++++++++++++

Proof of VC  'RECEIVER#6'

     (IMPLIES (AND (RECEIVER.INT PKT.IN SINK ACK.OUT NEXT QUEUE)
                   (NOT (IN (ADD1 NEXT) (DOMAIN QUEUE)))
                   (EQUAL (SEQNO PKT) NEXT)
                   (PKTP PKT))
              (RECEIVER.INT (APR PKT.IN PKT)
                            (APR SINK (MSSG PKT))
                            (APR ACK.OUT (ADD1 NEXT))
                            (ADD1 NEXT)
                            QUEUE))
```

This formula can be simplified, using the abbreviations NOT,

```
RECEIVER.INT, AND, and IMPLIES, to:$$

      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (IF
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (IF
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (IF
           (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                  NEXT)
           F
           (IF
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (IF
             (LESSP 0 NEXT)
             (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
             F)
            (IF (EQUAL NEXT 0)
                (EQUAL SINK (QUOTE (1QUOTE NULL)))
                F)))
          F)
         T)
        (NOT (IN (ADD1 NEXT) (DOMAIN QUEUE)))
        (EQUAL (SEQNO PKT) NEXT)
        (PKTP PKT))
       (RECEIVER.INT (APR PKT.IN PKT)
                     (APR SINK (MSSG PKT))
                     (APR ACK.OUT (ADD1 NEXT))
                     (ADD1 NEXT)
                     QUEUE)),

which simplifies, applying LST.APR, NLST.APR, APPLY2.EQUAL,
APPLY1.SEQNO, PMAPP.NLST, LOWER.WITHE.IF, IN.DOMAIN.WITHE.IF,
EQUAL.REACH.ZERO, SUB1.ADD1, FOLLOWS.UPPER.ADD1, UPPER.WITHE.IF,
and PMAPP.LATEST, and expanding the definitions of EQUAL, LESSP,
CONSISTENT, CONSISTENT2, PMAPP, RECEIVER.INT, and LATEST, to 23 new
conjectures:

Case 23.(IMPLIES (AND (PMAPP QUEUE)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (NOT (IN (ADD1 (SEQNO PKT))
                               (DOMAIN QUEUE)))
                      (PKTP PKT)
                      (NOT (SEQP QUEUE)))
```

```
                (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which again simplifies, expanding the function PMAPP, to:

```
       T.

Case 22.(IMPLIES (AND (PMAPP QUEUE)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (NOT (IN (ADD1 (SEQNO PKT))
                               (DOMAIN QUEUE)))
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (SEQP (NLST QUEUE)))
                 (MPAIRP (LST (NLST QUEUE)))),
```

which we again simplify, expanding the definition of PMAPP, to:

```
       T.

Case 21.(IMPLIES (AND (PMAPP QUEUE)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (NOT (IN (ADD1 (SEQNO PKT))
                               (DOMAIN QUEUE)))
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (SEQP (NLST QUEUE)))
                 (LESSP (DOM (LST (NLST QUEUE)))
                        (DOM (LST QUEUE)))),
```

which again simplifies, expanding the function PMAPP, to:

```
       T.

Case 20.(IMPLIES (AND (PMAPP QUEUE)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
                      (NOT (IN (ADD1 (SEQNO PKT))
                               (DOMAIN QUEUE)))
                      (PKTP PKT)
                      (SEQP QUEUE)
                      (NOT (SEQP (NLST QUEUE))))
                 (EQUAL (NLST QUEUE)
                        (QUOTE (1QUOTE NULL)))),
```

which again simplifies, expanding PMAPP, to:

```
       T.

Case 19.(IMPLIES (AND (PMAPP QUEUE)
                      (NOT (CONSISTENT (QUOTE SEQNO)
                                       (QUOTE EQUAL)
                                       PKT.IN))
```

```
                    (NOT (IN (ADD1 (SEQNO PKT))
                             (DOMAIN QUEUE)))
                    (PKTP PKT)
                    (SEQP QUEUE))
                (MPAIRP (LST QUEUE))),
```

which we again simplify, unfolding PMAPP, to:

```
       T.

Case 18.(IMPLIES
             (AND (PMAPP QUEUE)
                  (FOLLOWS QUEUE
                           (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                  (ADD1 (SEQNO PKT))))
                  (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                              (SEQNO PKT)))
                  (NOT (IN 0
                           (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                  (EQUAL (SEQNO PKT) 0)
                  (EQUAL SINK (QUOTE (1QUOTE NULL)))
                  (NOT (IN (ADD1 (SEQNO PKT))
                           (DOMAIN QUEUE)))
                  (PKTP PKT)
                  (NOT (SEQP QUEUE)))
             (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which we again simplify, unfolding the definition of PMAPP, to:

```
       T.

Case 17.(IMPLIES
             (AND (PMAPP QUEUE)
                  (FOLLOWS QUEUE
                           (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                  (ADD1 (SEQNO PKT))))
                  (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                              (SEQNO PKT)))
                  (NOT (IN 0
                           (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                  (EQUAL (SEQNO PKT) 0)
                  (EQUAL SINK (QUOTE (1QUOTE NULL)))
                  (NOT (IN (ADD1 (SEQNO PKT))
                           (DOMAIN QUEUE)))
                  (PKTP PKT)
                  (SEQP QUEUE)
                  (SEQP (NLST QUEUE)))
             (MPAIRP (LST (NLST QUEUE)))).
```

This simplifies again, unfolding the function PMAPP, to:

```
       T.

Case 16.(IMPLIES
             (AND (PMAPP QUEUE)
                  (FOLLOWS QUEUE
                           (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
```

```
                              (ADD1 (SEQNO PKT))))
                    (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                                (SEQNO PKT)))
                    (NOT (IN 0
                             (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                    (EQUAL (SEQNO PKT) 0)
                    (EQUAL SINK (QUOTE (1QUOTE NULL)))
                    (NOT (IN (ADD1 (SEQNO PKT))
                             (DOMAIN QUEUE)))
                    (PKTP PKT)
                    (SEQP QUEUE)
                    (SEQP (NLST QUEUE)))
               (LESSP (DOM (LST (NLST QUEUE)))
                      (DOM (LST QUEUE)))).
```

However this simplifies again, unfolding the function PMAPP, to:

```
       T.

Case 15.(IMPLIES
               (AND (PMAPP QUEUE)
                    (FOLLOWS QUEUE
                             (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                    (ADD1 (SEQNO PKT))))
                    (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                                (SEQNO PKT)))
                    (NOT (IN 0
                             (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                    (EQUAL (SEQNO PKT) 0)
                    (EQUAL SINK (QUOTE (1QUOTE NULL)))
                    (NOT (IN (ADD1 (SEQNO PKT))
                             (DOMAIN QUEUE)))
                    (PKTP PKT)
                    (SEQP QUEUE)
                    (NOT (SEQP (NLST QUEUE))))
               (EQUAL (NLST QUEUE)
                      (QUOTE (1QUOTE NULL)))),
```

which we again simplify, expanding the definition of PMAPP, to:

```
       T.

Case 14.(IMPLIES
               (AND (PMAPP QUEUE)
                    (FOLLOWS QUEUE
                             (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                    (ADD1 (SEQNO PKT))))
                    (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                                (SEQNO PKT)))
                    (NOT (IN 0
                             (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                    (EQUAL (SEQNO PKT) 0)
                    (EQUAL SINK (QUOTE (1QUOTE NULL)))
                    (NOT (IN (ADD1 (SEQNO PKT))
                             (DOMAIN QUEUE)))
                    (PKTP PKT)
                    (SEQP QUEUE))
```

```
              (MPAIRP (LST QUEUE))),
```

which again simplifies, unfolding the definition of PMAPP, to:

```
       T.$

Case 13.(IMPLIES
          (AND (PMAPP QUEUE)
               (FOLLOWS QUEUE
                        (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                               (ADD1 (SEQNO PKT))))
               (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                           (SEQNO PKT)))
               (NOT (IN 0
                        (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
               (EQUAL (SEQNO PKT) 0)
               (EQUAL SINK (QUOTE (1QUOTE NULL)))
               (NOT (IN (ADD1 (SEQNO PKT))
                        (DOMAIN QUEUE)))
               (PKTP PKT)
               (CONSISTENT2 (QUOTE SEQNO)
                            (QUOTE EQUAL)
                            PKT.IN PKT)
               (CONSISTENT (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN)
               (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (EQUAL (APR SINK (MSSG PKT))
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT)))))).
```

This again simplifies, rewriting with PMAPP.LATEST and IN.DOMAIN.MPAIR, and opening up the definitions of EQUAL and LESSP, to:

```
       T.$

Case 12.(IMPLIES
              (AND (PMAPP QUEUE)
                   (FOLLOWS QUEUE
                            (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                   (ADD1 (SEQNO PKT))))
                   (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                               (SEQNO PKT)))
                   (NOT (IN 0
                            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                   (EQUAL (SEQNO PKT) 0)
                   (EQUAL SINK (QUOTE (1QUOTE NULL)))
                   (NOT (IN (ADD1 (SEQNO PKT))
                            (DOMAIN QUEUE)))
                   (PKTP PKT)
                   (CONSISTENT2 (QUOTE SEQNO)
                                (QUOTE EQUAL)
                                PKT.IN PKT)
                   (CONSISTENT (QUOTE SEQNO)
```

```
                              (QUOTE EQUAL)
                              PKT.IN)
                    (NOT (IN (MPAIR (SEQNO PKT) PKT)
                             (LATEST (QUOTE SEQNO) PKT.IN))))
               (EQUAL (APR SINK (MSSG PKT))
                      (FAPPLY (QUOTE MSSG)
                              (RANGE (APR (QUOTE (1QUOTE NULL))
                                          (MPAIR 0 PKT))))))).
```

But this again simplifies, applying PMAPP.LATEST, IN.MPAIR.DOMAIN, RNG.MPAIR, LST.APR, NLST.APR, APPLY1.MSSG, and APR.EQUAL, and expanding EQUAL, LESSP, RANGE, and FAPPLY, to:

```
      (IMPLIES
            (AND (PMAPP QUEUE)
                 (FOLLOWS QUEUE
                          (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                 1))
                 (NOT (IN 0
                          (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                 (EQUAL (SEQNO PKT) 0)
                 (NOT (IN 1 (DOMAIN QUEUE)))
                 (PKTP PKT)
                 (CONSISTENT2 (QUOTE SEQNO)
                              (QUOTE EQUAL)
                              PKT.IN PKT)
                 (CONSISTENT (QUOTE SEQNO)
                             (QUOTE EQUAL)
                             PKT.IN))
               (EQUAL (QUOTE (1QUOTE NULL))
                      (FAPPLY (QUOTE MSSG)
                              (QUOTE (1QUOTE NULL))))),
```

which we again simplify, expanding the functions SEQP, FAPPLY, and EQUAL, to:

```
      T.$

Case 11.(IMPLIES
         (AND (PMAPP QUEUE)
              (FOLLOWS QUEUE
                       (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                              (ADD1 (SEQNO PKT))))
              (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                          (SEQNO PKT)))
              (NOT (IN 0
                       (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
              (EQUAL (SEQNO PKT) 0)
              (EQUAL SINK (QUOTE (1QUOTE NULL)))
              (NOT (IN (ADD1 (SEQNO PKT))
                       (DOMAIN QUEUE)))
              (PKTP PKT)
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN PKT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
```

```
                    PKT.IN))
         (NOT
            (LESSP (SUB1 (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT)
                                       PKT)))
                   (SEQNO PKT)))).
```

This again simplifies, rewriting with PMAPP.LATEST and WITHE.ZERO.2, and expanding the definitions of EQUAL and LESSP, to:

```
       T.

Case 10.(IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (NOT (SEQP QUEUE)))
         (EQUAL QUEUE (QUOTE (1QUOTE NULL)))),
```

which again simplifies, expanding the definition of PMAPP, to:

```
       T.

Case 9. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (SEQP QUEUE)
```

```
          (SEQP (NLST QUEUE)))
         (MPAIRP (LST (NLST QUEUE)))).
```

But this again simplifies, unfolding the function PMAPP, to:

```
       T.$
```

```
Case 8. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (SEQP QUEUE)
          (SEQP (NLST QUEUE)))
         (LESSP (DOM (LST (NLST QUEUE)))
                (DOM (LST QUEUE)))).
```

This again simplifies, expanding the definition of PMAPP, to:

```
       T.
```

```
Case 7. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (SEQP QUEUE)
          (NOT (SEQP (NLST QUEUE))))
         (EQUAL (NLST QUEUE)
                (QUOTE (1QUOTE NULL)))),
```

which we again simplify, unfolding the function PMAPP, to:

```
        T.

Case 6. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (SEQP QUEUE))
         (MPAIRP (LST QUEUE))).
```

This simplifies again, unfolding the function PMAPP, to:

```
        T.$

Case 5. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (IN (MPAIR (SEQNO PKT) PKT)
              (LATEST (QUOTE SEQNO) PKT.IN)))
         (EQUAL (APR SINK (MSSG PKT))
                (FAPPLY (QUOTE MSSG)
                        (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                      (SEQNO PKT))))),
```

which again simplifies, clearly, to the new formula:$

```
      (IMPLIES
       (AND (PMAPP QUEUE)
            (FOLLOWS QUEUE
                     (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                            (ADD1 (SEQNO PKT))))
            (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                        (SEQNO PKT)))
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (NOT (EQUAL (SEQNO PKT) 0))
            (NOT (IN (ADD1 (SEQNO PKT))
                     (DOMAIN QUEUE)))
            (PKTP PKT)
            (CONSISTENT2 (QUOTE SEQNO)
                         (QUOTE EQUAL)
                         PKT.IN PKT)
            (CONSISTENT (QUOTE SEQNO)
                        (QUOTE EQUAL)
                        PKT.IN)
            (IN (MPAIR (SEQNO PKT) PKT)
                (LATEST (QUOTE SEQNO) PKT.IN)))
       (EQUAL
          (APR (FAPPLY (QUOTE MSSG)
                       (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                     (SUB1 (SEQNO PKT)))))
               (MSSG PKT))
          (FAPPLY (QUOTE MSSG)
                  (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                (SEQNO PKT)))))).
```

Applying the lemmas MSSG/SEQNO.ELIM and SUB1.ELIM, replace PKT by (PACKET Z X) to eliminate (SEQNO PKT) and (MSSG PKT) and X by (ADD1 V) to eliminate (SUB1 X). We rely upon the type restriction lemma noted when SEQNO was introduced and the type restriction lemma noted when SUB1 was introduced to restrict the new variables. The result is:$

```
      (IMPLIES
       (AND (NUMBERP V)
            (PMAPP QUEUE)
            (FOLLOWS QUEUE
                     (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                            (ADD1 (ADD1 V))))
            (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                        (ADD1 V)))
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (NOT (EQUAL (ADD1 V) 0))
            (NOT (IN (ADD1 (ADD1 V)) (DOMAIN QUEUE)))
            (CONSISTENT2 (QUOTE SEQNO)
                         (QUOTE EQUAL)
                         PKT.IN
                         (PACKET Z (ADD1 V)))
            (CONSISTENT (QUOTE SEQNO)
                        (QUOTE EQUAL)
```

```
                      PKT.IN)
            (IN (MPAIR (ADD1 V) (PACKET Z (ADD1 V)))
                (LATEST (QUOTE SEQNO) PKT.IN)))
         (EQUAL
            (APR (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       V)))
                 Z)
            (FAPPLY (QUOTE MSSG)
                    (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                  (ADD1 V))))))),
```

which further simplifies, rewriting with SUB1.ADD1, EQUAL.REACH.ZERO, PMAPP.LATEST, LOWER.ADD1, RNG.MPAIR, LST.APR, NLST.APR, MSSG.PACKET, and APPLY1.MSSG, and opening up the functions LESSP, RANGE, and FAPPLY, to:

T.$

```
Case 4. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN))))
         (EQUAL
            (APR SINK (MSSG PKT))
            (FAPPLY (QUOTE MSSG)
                    (RANGE (APR (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT)))
                                (MPAIR (SEQNO PKT) PKT))))))),
```

which again simplifies, applying RNG.MPAIR, LST.APR, NLST.APR, and APPLY1.MSSG, and opening up the definitions of RANGE and FAPPLY, to:

T.$

```
Case 3. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN))
         (NOT
            (LESSP (SUB1 (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT)
                                       PKT)))
                   (SEQNO PKT)))).
```

This again simplifies, using linear arithmetic and applying LESSP.REACH.WITHE.2 and PMAPP.LATEST, to:

```
        (IMPLIES
         (AND (EQUAL (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                     (SEQNO PKT))
              (PMAPP QUEUE)
              (FOLLOWS QUEUE
                       (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                              (ADD1 (SEQNO PKT))))
              (NOT (LESSP (SEQNO PKT) (SEQNO PKT)))
              (IN 0
                  (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
              (NOT (EQUAL (SEQNO PKT) 0))
              (NOT (IN (ADD1 (SEQNO PKT))
                       (DOMAIN QUEUE)))
              (PKTP PKT)
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN PKT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.IN))
         (NOT
            (LESSP (SUB1 (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT)
                                       PKT)))
```

```
                (SEQNO PKT)))).
```

This simplifies again, using linear arithmetic and applying LESSP.SUB1.REACH.WITHE and PMAPP.LATEST, to:$

```
       (IMPLIES
        (AND (EQUAL (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                  (SEQNO PKT)
                                  PKT))
                    0)
             (EQUAL (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                    (SEQNO PKT))
             (PMAPP QUEUE)
             (FOLLOWS QUEUE
                      (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                             (ADD1 (SEQNO PKT))))
             (NOT (LESSP (SEQNO PKT) (SEQNO PKT)))
             (IN 0
                 (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
             (NOT (EQUAL (SEQNO PKT) 0))
             (NOT (IN (ADD1 (SEQNO PKT))
                      (DOMAIN QUEUE)))
             (PKTP PKT)
             (CONSISTENT2 (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.IN PKT)
             (CONSISTENT (QUOTE SEQNO)
                         (QUOTE EQUAL)
                         PKT.IN))
        (NOT
           (LESSP (SUB1 (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                      (SEQNO PKT)
                                      PKT)))
                  (SEQNO PKT)))),
```

which we again simplify, using linear arithmetic and applying LESSP.REACH.WITHE.2 and PMAPP.LATEST, to:

```
       T.$

Case 2. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
```

```
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (LESSP (SUB1 (SEQNO PKT))
                 (ADD1 (SEQNO PKT))))
         (FOLLOWS QUEUE
                  (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                         (ADD1 (ADD1 (SEQNO PKT))))))),
```

which we again simplify, applying SUB1.ADD1, PMAPP.LATEST, and FOLLOWS.UPPER.ADD1, and expanding the function LESSP, to:

```
       T.
```

```
Case 1. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (NOT (IN (ADD1 (SEQNO PKT))
                   (DOMAIN QUEUE)))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (NOT (LESSP (SUB1 (SEQNO PKT))
                      (ADD1 (SEQNO PKT)))))
         (FOLLOWS QUEUE
                  (WITHE (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                (ADD1 (ADD1 (SEQNO PKT))))
                         (SEQNO PKT)
                         PKT))),
```

which we again simplify, using linear arithmetic, to:

```
       T.
```

Q.E.D.

390914 conses

```
360.125 seconds
44.228 seconds, garbage collection time


--------------------------------------




++++++++++++++++++++++++++++++++++++++++

Proof of VC  'RECEIVER#7'
     (IMPLIES
          (AND (RECEIVER.INT PKT.IN SINK ACK.OUT NEXT QUEUE)
               (IN (ADD1 NEXT) (DOMAIN QUEUE))
               (EQUAL (SEQNO PKT) NEXT)
               (PKTP PKT))
          (RECEIVER.INT (APR PKT.IN PKT)
                        (JOIN SINK
                              (APL (MSSG PKT)
                                   (FAPPLY (QUOTE MSSG)
                                           (RANGE (CONSEC QUEUE)))))
                        (APR ACK.OUT (REACH QUEUE))
                        (REACH QUEUE)
                        (UPPER QUEUE (REACH QUEUE))))

This conjecture can be simplified, using the abbreviations
RECEIVER.INT, AND, IMPLIES, and JOIN.APL.APR, to:$$$$

      (IMPLIES
       (AND
        (PMAPP QUEUE)
        (NUMBERP NEXT)
        (IF
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (IF
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 NEXT)))
          (IF
           (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                  NEXT)
           F
           (IF
            (IN 0
                (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
            (IF
             (LESSP 0 NEXT)
             (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 NEXT)))))
             F)
            (IF (EQUAL NEXT 0)
```

```
              (EQUAL SINK (QUOTE (1QUOTE NULL)))
              F)))
         F)
        T)
       (IN (ADD1 NEXT) (DOMAIN QUEUE))
       (EQUAL (SEQNO PKT) NEXT)
       (PKTP PKT))
      (RECEIVER.INT (APR PKT.IN PKT)
                    (JOIN (APR SINK (MSSG PKT))
                          (FAPPLY (QUOTE MSSG)
                                  (RANGE (CONSEC QUEUE))))
                    (APR ACK.OUT (REACH QUEUE))
                    (REACH QUEUE)
                    (UPPER QUEUE (REACH QUEUE)))),
```

which simplifies, using linear arithmetic, rewriting with the lemmas LST.APR, NLST.APR, APPLY2.EQUAL, APPLY1.SEQNO, PMAPP.UPPER, SUB1.ADD1, FIRST.DOMAIN.FOLLOWS.UPPER, LOWER.SUB1.REACH.3, IN.DOMAIN.FOLLOWS.UPPER, LOWER.WITHE.IF, LESSP.ZERO.REACH, IN.DOMAIN.WITHE.IF, FOLLOWS.UPPER.ADD1, IN.DOMAIN.UPPER, IN.REACH.DOMAIN, FOLLOWS.TRANS, FOLLOWS.SAME, FOLLOWS.UPPER, FOLLOWS.UPPER.UPPER, UPPER.WITHE.IF, PMAPP.LATEST, WITHE.JOIN.LOWER, FIRST.DOMAIN.CONSEC, and PMAPP.CONSEC, and expanding the functions EQUAL, LESSP, CONSISTENT, CONSISTENT2, RECEIVER.INT, and LATEST, to 15 new conjectures:

```
Case 15.(IMPLIES
             (AND (PMAPP QUEUE)
                  (FOLLOWS QUEUE
                           (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                  (ADD1 (SEQNO PKT))))
                  (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                              (SEQNO PKT)))
                  (NOT (IN 0
                           (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                  (EQUAL (SEQNO PKT) 0)
                  (EQUAL SINK (QUOTE (1QUOTE NULL)))
                  (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
                  (PKTP PKT)
                  (CONSISTENT2 (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.IN PKT)
                  (CONSISTENT (QUOTE SEQNO)
                              (QUOTE EQUAL)
                              PKT.IN)
                  (EQUAL (SEQNO PKT)
                         (SUB1 (REACH QUEUE)))
                  (NOT (IN (MPAIR (SEQNO PKT) PKT)
                           (LATEST (QUOTE SEQNO) PKT.IN)))
                  (EQUAL (SUB1 (REACH QUEUE)) 0))
             (EQUAL (JOIN (APR SINK (MSSG PKT))
                          (FAPPLY (QUOTE MSSG)
                                  (RANGE (CONSEC QUEUE))))
                    (FAPPLY (QUOTE MSSG)
                            (RANGE (APR (QUOTE (1QUOTE NULL))
                                        (MPAIR 0 PKT)))))),
```

which again simplifies, applying the lemma IN.ADD1.DOMAIN, and

```
  expanding the functions EQUAL and LESSP, to:
        T.

Case 14.(IMPLIES
         (AND (PMAPP QUEUE)
              (FOLLOWS QUEUE
                       (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                              (ADD1 (SEQNO PKT))))
              (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                          (SEQNO PKT)))
              (NOT (IN 0
                       (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
              (EQUAL (SEQNO PKT) 0)
              (EQUAL SINK (QUOTE (1QUOTE NULL)))
              (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
              (PKTP PKT)
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN PKT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.IN)
              (EQUAL (SEQNO PKT)
                     (SUB1 (REACH QUEUE)))
              (NOT (IN (MPAIR (SEQNO PKT) PKT)
                       (LATEST (QUOTE SEQNO) PKT.IN)))
              (NOT (EQUAL (SUB1 (REACH QUEUE)) 0)))
         (EQUAL
            (JOIN (APR SINK (MSSG PKT))
                  (FAPPLY (QUOTE MSSG)
                          (RANGE (CONSEC QUEUE))))
            (FAPPLY (QUOTE MSSG)
                    (RANGE (APR (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SUB1 (REACH QUEUE))))
                                (MPAIR (SEQNO PKT) PKT)))))),

  which again simplifies, using linear arithmetic, to:

        T.$

Case 13.(IMPLIES
         (AND (PMAPP QUEUE)
              (FOLLOWS QUEUE
                       (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                              (ADD1 (SEQNO PKT))))
              (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                          (SEQNO PKT)))
              (NOT (IN 0
                       (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
              (EQUAL (SEQNO PKT) 0)
              (EQUAL SINK (QUOTE (1QUOTE NULL)))
              (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
              (PKTP PKT)
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN PKT)
              (CONSISTENT (QUOTE SEQNO)
```

```
                        (QUOTE EQUAL)
                        PKT.IN)
              (EQUAL (SEQNO PKT)
                     (SUB1 (REACH QUEUE)))
              (IN (MPAIR (SEQNO PKT) PKT)
                  (LATEST (QUOTE SEQNO) PKT.IN)))
         (EQUAL
           (JOIN (APR SINK (MSSG PKT))
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (CONSEC QUEUE))))
           (FAPPLY (QUOTE MSSG)
                   (RANGE (JOIN (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT))
                                (CONSEC QUEUE))))))).
```

However this simplifies again, using linear arithmetic, applying LESSP.REACH.REACH.1, IN.DOMAIN.MPAIR, PMAPP.LATEST, IN.DOMAIN.REACH, and IN.ADD1.DOMAIN, and unfolding EQUAL and LESSP, to:

```
       T.$

Case 12.(IMPLIES
         (AND (PMAPP QUEUE)
              (FOLLOWS QUEUE
                       (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                              (ADD1 (SEQNO PKT))))
              (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                          (SEQNO PKT)))
              (NOT (IN 0
                       (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
              (EQUAL (SEQNO PKT) 0)
              (EQUAL SINK (QUOTE (1QUOTE NULL)))
              (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
              (PKTP PKT)
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN PKT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.IN)
              (NOT (EQUAL (SEQNO PKT)
                          (SUB1 (REACH QUEUE)))))
         (EQUAL
          (JOIN (APR SINK (MSSG PKT))
                (FAPPLY (QUOTE MSSG)
                        (RANGE (CONSEC QUEUE))))
          (FAPPLY
            (QUOTE MSSG)
            (RANGE (WITHE (JOIN (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT))
                                (CONSEC QUEUE))
                          (SEQNO PKT)
                          PKT))))),
```

which we again simplify, rewriting with PMAPP.LATEST, LOWER.ZERO.2, PMAPP.CONSEC, JOIN.NULL.PMAPP, WITHE.ZERO.2,

```
  IN.DOMAIN.FOLLOWS.UPPER, IN.DOMAIN.CONSEC.NOT, RANGE.JOIN,
  RNG.MPAIR, LST.APR, NLST.APR, FAPPLY.JOIN, and APPLY1.MSSG, and
  expanding the definitions of EQUAL, LESSP, RANGE, and FAPPLY, to:

$       (IMPLIES
              (AND (PMAPP QUEUE)
                   (FOLLOWS QUEUE
                            (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                   1))
                   (NOT (IN 0
                            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                   (EQUAL (SEQNO PKT) 0)
                   (IN 1 (DOMAIN QUEUE))
                   (PKTP PKT)
                   (CONSISTENT2 (QUOTE SEQNO)
                                (QUOTE EQUAL)
                                PKT.IN PKT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.IN)
                   (NOT (EQUAL 0 (SUB1 (REACH QUEUE)))))
              (EQUAL (JOIN (APR (QUOTE (1QUOTE NULL)) (MSSG PKT))
                           (FAPPLY (QUOTE MSSG)
                                   (RANGE (CONSEC QUEUE))))
                     (JOIN (APR (FAPPLY (QUOTE MSSG)
                                        (QUOTE (1QUOTE NULL)))
                                (MSSG PKT))
                           (FAPPLY (QUOTE MSSG)
                                   (RANGE (CONSEC QUEUE))))))),

  which again simplifies, opening up SEQP and FAPPLY, to:

       T.

Case 11.(IMPLIES
              (AND (PMAPP QUEUE)
                   (FOLLOWS QUEUE
                            (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                   (ADD1 (SEQNO PKT))))
                   (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                               (SEQNO PKT)))
                   (NOT (IN 0
                            (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                   (EQUAL (SEQNO PKT) 0)
                   (EQUAL SINK (QUOTE (1QUOTE NULL)))
                   (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
                   (PKTP PKT)
                   (CONSISTENT2 (QUOTE SEQNO)
                                (QUOTE EQUAL)
                                PKT.IN PKT)
                   (CONSISTENT (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.IN))
              (SEQP QUEUE)).

  This again simplifies, applying the lemmas UPPER.ZERO,
  PMAPP.LATEST, and FOLLOWS.UPPER.ADD1, and opening up the
```

definitions of PMAPP, FOLLOWS, IN, DOMAIN, EQUAL, and LESSP, to:

```
      T.$

Case 10.(IMPLIES
             (AND (PMAPP QUEUE)
                  (FOLLOWS QUEUE
                           (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                  (ADD1 (SEQNO PKT))))
                  (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                              (SEQNO PKT)))
                  (NOT (IN 0
                           (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                  (EQUAL (SEQNO PKT) 0)
                  (EQUAL SINK (QUOTE (1QUOTE NULL)))
                  (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
                  (PKTP PKT)
                  (CONSISTENT2 (QUOTE SEQNO)
                               (QUOTE EQUAL)
                               PKT.IN PKT)
                  (CONSISTENT (QUOTE SEQNO)
                              (QUOTE EQUAL)
                              PKT.IN))
             (NOT (LESSP (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT)
                                       PKT))
                         (REACH QUEUE)))).
```

This again simplifies, rewriting with PMAPP.LATEST, WITHE.ZERO.2, and REACH.JOIN.APR.NULL.2, and expanding EQUAL and LESSP, to:

```
      (IMPLIES
            (AND (PMAPP QUEUE)
                 (FOLLOWS QUEUE
                          (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                 1))
                 (NOT (IN 0
                          (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN))))
                 (EQUAL (SEQNO PKT) 0)
                 (IN 1 (DOMAIN QUEUE))
                 (PKTP PKT)
                 (CONSISTENT2 (QUOTE SEQNO)
                              (QUOTE EQUAL)
                              PKT.IN PKT)
                 (CONSISTENT (QUOTE SEQNO)
                             (QUOTE EQUAL)
                             PKT.IN))
             (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                         (REACH QUEUE)))).
```

However this simplifies again, using linear arithmetic and applying LESSP.REACH.REACH.1 and PMAPP.LATEST, to:

```
      T.

Case 9. (IMPLIES
         (AND
```

```
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT)
                 (SUB1 (REACH QUEUE)))
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (EQUAL (SUB1 (REACH QUEUE)) 0))
         (EQUAL (JOIN (APR SINK (MSSG PKT))
                      (FAPPLY (QUOTE MSSG)
                              (RANGE (CONSEC QUEUE))))
                (FAPPLY (QUOTE MSSG)
                        (RANGE (APR (QUOTE (1QUOTE NULL))
                                    (MPAIR 0 PKT)))))).
```

But this again simplifies, using linear arithmetic, to:

```
        T.$

Case 8. (IMPLIES
          (AND
           (PMAPP QUEUE)
           (FOLLOWS QUEUE
                    (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                           (ADD1 (SEQNO PKT))))
           (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                       (SEQNO PKT)))
           (IN 0
               (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
           (NOT (EQUAL (SEQNO PKT) 0))
           (EQUAL SINK
                  (FAPPLY (QUOTE MSSG)
                          (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                        (SUB1 (SEQNO PKT))))))
           (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
           (PKTP PKT)
           (CONSISTENT2 (QUOTE SEQNO)
                        (QUOTE EQUAL)
                        PKT.IN PKT)
           (CONSISTENT (QUOTE SEQNO)
```

```
                    (QUOTE EQUAL)
                    PKT.IN)
          (EQUAL (SEQNO PKT)
                 (SUB1 (REACH QUEUE)))
          (NOT (IN (MPAIR (SEQNO PKT) PKT)
                   (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SUB1 (REACH QUEUE)) 0)))
         (EQUAL
           (JOIN (APR SINK (MSSG PKT))
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (CONSEC QUEUE))))
           (FAPPLY (QUOTE MSSG)
                   (RANGE (APR (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                      (SUB1 (SUB1 (REACH QUEUE))))
                               (MPAIR (SEQNO PKT) PKT))))))).
```

This again simplifies, applying IN.ADD1.DOMAIN, to:

```
       T.
```

```
Case 7. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (EQUAL (SEQNO PKT)
                 (SUB1 (REACH QUEUE)))
          (IN (MPAIR (SEQNO PKT) PKT)
              (LATEST (QUOTE SEQNO) PKT.IN)))
         (EQUAL
           (JOIN (APR SINK (MSSG PKT))
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (CONSEC QUEUE))))
           (FAPPLY (QUOTE MSSG)
                   (RANGE (JOIN (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT))
                                (CONSEC QUEUE))))),
```

which we again simplify, applying IN.ADD1.DOMAIN, to:

```
        T.$

Case 6. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (NOT (EQUAL (SEQNO PKT)
                      (SUB1 (REACH QUEUE))))
          (LESSP (SUB1 (REACH QUEUE))
                 (SEQNO PKT)))
         (EQUAL
           (JOIN (APR SINK (MSSG PKT))
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (CONSEC QUEUE))))
           (FAPPLY (QUOTE MSSG)
                   (RANGE (JOIN (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SEQNO PKT))
                                (CONSEC QUEUE))))))),

  which again simplifies, applying RANGE.JOIN and FAPPLY.JOIN, to:$

        (IMPLIES
         (AND (PMAPP QUEUE)
              (FOLLOWS QUEUE
                       (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                              (ADD1 (SEQNO PKT))))
              (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                          (SEQNO PKT)))
              (IN 0
                  (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
              (NOT (EQUAL (SEQNO PKT) 0))
              (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
              (PKTP PKT)
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN PKT)
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
```

```
                    PKT.IN)
          (NOT (EQUAL (SEQNO PKT)
                      (SUB1 (REACH QUEUE))))
          (LESSP (SUB1 (REACH QUEUE))
                 (SEQNO PKT)))
         (EQUAL
          (JOIN
            (APR (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT)))))
                 (MSSG PKT))
            (FAPPLY (QUOTE MSSG)
                    (RANGE (CONSEC QUEUE))))
          (JOIN (FAPPLY (QUOTE MSSG)
                        (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                      (SEQNO PKT))))
                (FAPPLY (QUOTE MSSG)
                        (RANGE (CONSEC QUEUE))))))).
```

Appealing to the lemmas MSSG/SEQNO.ELIM and SUB1.ELIM, replace PKT by (PACKET Z X) to eliminate (SEQNO PKT) and (MSSG PKT) and X by (ADD1 V) to eliminate (SUB1 X). We rely upon the type restriction lemma noted when SEQNO was introduced and the type restriction lemma noted when SUB1 was introduced to constrain the new variables. This generates:

```
        (IMPLIES
         (AND (NUMBERP V)
              (PMAPP QUEUE)
              (FOLLOWS QUEUE
                       (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                              (ADD1 (ADD1 V))))
              (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                          (ADD1 V)))
              (IN 0
                  (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
              (NOT (EQUAL (ADD1 V) 0))
              (IN (ADD1 (ADD1 V)) (DOMAIN QUEUE))
              (CONSISTENT2 (QUOTE SEQNO)
                           (QUOTE EQUAL)
                           PKT.IN
                           (PACKET Z (ADD1 V)))
              (CONSISTENT (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.IN)
              (NOT (EQUAL (ADD1 V) (SUB1 (REACH QUEUE))))
              (LESSP (SUB1 (REACH QUEUE)) (ADD1 V)))
         (EQUAL
          (JOIN
            (APR (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       V)))
                 Z)
            (FAPPLY (QUOTE MSSG)
                    (RANGE (CONSEC QUEUE))))
          (JOIN (FAPPLY (QUOTE MSSG)
                        (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
```

```
                                    (ADD1 V))))
                (FAPPLY (QUOTE MSSG)
                        (RANGE (CONSEC QUEUE))))))).
```

This simplifies further, using linear arithmetic and rewriting with LESSP.SUB1.REACH.FOLLOWS and PMAPP.LATEST, to:

```
       (IMPLIES
        (AND (EQUAL (REACH QUEUE) 0)
             (NUMBERP V)
             (PMAPP QUEUE)
             (FOLLOWS QUEUE
                      (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                             (ADD1 (ADD1 V))))
             (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                         (ADD1 V)))
             (IN 0
                 (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
             (NOT (EQUAL (ADD1 V) 0))
             (IN (ADD1 (ADD1 V)) (DOMAIN QUEUE))
             (CONSISTENT2 (QUOTE SEQNO)
                          (QUOTE EQUAL)
                          PKT.IN
                          (PACKET Z (ADD1 V)))
             (CONSISTENT (QUOTE SEQNO)
                         (QUOTE EQUAL)
                         PKT.IN)
             (NOT (EQUAL (ADD1 V) (SUB1 (REACH QUEUE))))
             (LESSP (SUB1 (REACH QUEUE)) (ADD1 V)))
        (EQUAL
         (JOIN
           (APR (FAPPLY (QUOTE MSSG)
                        (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                      V)))
                Z)
           (FAPPLY (QUOTE MSSG)
                   (RANGE (CONSEC QUEUE))))
         (JOIN (FAPPLY (QUOTE MSSG)
                       (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                     (ADD1 V))))
               (FAPPLY (QUOTE MSSG)
                       (RANGE (CONSEC QUEUE)))))),
```

which again simplifies, applying EQUAL.REACH.ZERO, FOLLOWS.UPPER.ADD1, PMAPP.LATEST, and SUB1.ADD1, and expanding the definitions of PMAPP, FOLLOWS, IN, SEQP, DOMAIN, and LESSP, to:

```
       T.$

Case 5. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
```

```
                (SEQNO PKT)))
         (IN 0
             (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
         (NOT (EQUAL (SEQNO PKT) 0))
         (EQUAL SINK
                (FAPPLY (QUOTE MSSG)
                        (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                      (SUB1 (SEQNO PKT))))))
         (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
         (PKTP PKT)
         (CONSISTENT2 (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN PKT)
         (CONSISTENT (QUOTE SEQNO)
                     (QUOTE EQUAL)
                     PKT.IN)
         (NOT (EQUAL (SEQNO PKT)
                     (SUB1 (REACH QUEUE))))
         (NOT (LESSP (SUB1 (REACH QUEUE))
                     (SEQNO PKT))))
        (EQUAL
         (JOIN (APR SINK (MSSG PKT))
               (FAPPLY (QUOTE MSSG)
                       (RANGE (CONSEC QUEUE))))
         (FAPPLY
             (QUOTE MSSG)
             (RANGE (JOIN (APR (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                      (SUB1 (SEQNO PKT)))
                               (MPAIR (SEQNO PKT) PKT))
                          (CONSEC QUEUE)))))).
```

But this simplifies again, rewriting with RANGE.JOIN, RNG.MPAIR, LST.APR, NLST.APR, FAPPLY.JOIN, and APPLY1.MSSG, and opening up RANGE and FAPPLY, to:

```
       T.$

Case 4. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
```

```
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN))
         (SEQP QUEUE)),
```

which we again simplify, rewriting with PMAPP.LATEST and FOLLOWS.UPPER.ADD1, and opening up the definitions of PMAPP, FOLLOWS, IN, SEQP, and DOMAIN, to:

```
       T.
```

```
Case 3. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN))
         (NOT (LESSP (REACH (WITHE (LATEST (QUOTE SEQNO) PKT.IN)
                                   (SEQNO PKT)
                                   PKT))
                     (REACH QUEUE)))).
```

This simplifies again, using linear arithmetic and applying LESSP.REACH.WITHE.REACH, PMAPP.LATEST, FOLLOWS.TRANS, FOLLOWS.LATEST.CONSISTENT, FOLLOWS.SAME, NUMBERP.SEQNO, FOLLOWS.UPPER, FOLLOWS.WITHE.IF, and IN.DOMAIN.FOLLOWS.UPPER, to:

```
       T.$
```

```
Case 2. (IMPLIES
         (AND
          (PMAPP QUEUE)
          (FOLLOWS QUEUE
                   (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                          (ADD1 (SEQNO PKT))))
          (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                      (SEQNO PKT)))
          (IN 0
              (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
          (NOT (EQUAL (SEQNO PKT) 0))
```

```
          (EQUAL SINK
                 (FAPPLY (QUOTE MSSG)
                         (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                       (SUB1 (SEQNO PKT))))))
          (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
          (PKTP PKT)
          (CONSISTENT2 (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN PKT)
          (CONSISTENT (QUOTE SEQNO)
                      (QUOTE EQUAL)
                      PKT.IN)
          (LESSP (SUB1 (SEQNO PKT))
                 (REACH QUEUE)))
         (FOLLOWS (UPPER QUEUE (REACH QUEUE))
                  (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                         (ADD1 (REACH QUEUE))))).
```

But this again simplifies, applying PMAPP.UPPER, FOLLOWS.UPPER.UPPER, FOLLOWS.UPPER, NUMBERP.SEQNO, FOLLOWS.SAME, FOLLOWS.LATEST.CONSISTENT, FOLLOWS.TRANS, PMAPP.LATEST, IN.REACH.DOMAIN, IN.DOMAIN.UPPER, and FOLLOWS.UPPER.ADD1, to:

T.$$

```
Case 1. (IMPLIES
          (AND
           (PMAPP QUEUE)
           (FOLLOWS QUEUE
                    (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                           (ADD1 (SEQNO PKT))))
           (NOT (LESSP (REACH (LATEST (QUOTE SEQNO) PKT.IN))
                       (SEQNO PKT)))
           (IN 0
               (DOMAIN (LATEST (QUOTE SEQNO) PKT.IN)))
           (NOT (EQUAL (SEQNO PKT) 0))
           (EQUAL SINK
                  (FAPPLY (QUOTE MSSG)
                          (RANGE (LOWER (LATEST (QUOTE SEQNO) PKT.IN)
                                        (SUB1 (SEQNO PKT))))))
           (IN (ADD1 (SEQNO PKT)) (DOMAIN QUEUE))
           (PKTP PKT)
           (CONSISTENT2 (QUOTE SEQNO)
                        (QUOTE EQUAL)
                        PKT.IN PKT)
           (CONSISTENT (QUOTE SEQNO)
                       (QUOTE EQUAL)
                       PKT.IN)
           (NOT (LESSP (SUB1 (SEQNO PKT))
                       (REACH QUEUE))))
          (FOLLOWS (UPPER QUEUE (REACH QUEUE))
                   (WITHE (UPPER (LATEST (QUOTE SEQNO) PKT.IN)
                                 (ADD1 (REACH QUEUE)))
                          (SEQNO PKT)
                          PKT))).
```

This simplifies again, using linear arithmetic, applying

```
  FOLLOWS.UPPER.ADD1, IN.DOMAIN.UPPER, PMAPP.LATEST,
  IN.DOMAIN.FOLLOWS.UPPER, FOLLOWS.TRANS, FOLLOWS.LATEST.CONSISTENT,
  FOLLOWS.SAME, NUMBERP.SEQNO, FOLLOWS.UPPER, FOLLOWS.UPPER.UPPER,
  PMAPP.UPPER, FOLLOWS.WITHE.IF, and SUB1.ADD1, and expanding LESSP,
  to:

        T.

Q.E.D.

434606 conses
393.51 seconds
54.115 seconds, garbage collection time


----------------------------------------
```

References

[1] R. S. Boyer and J S. Moore.
A Computational Logic.
Academic Press, New York, 1979.

[2] B. L. DiVito.
Verification of Communications Protocols and Abstract Process Models.
PhD thesis, University of Texas at Austin, 1982.
Technical Report 25, Institute for Computing Science.

[3] N. V. Stenning.
A Data Transfer Protocol.
Computer Networks 1(2), September, 1976.