TCAS II CHANGE PROPOSAL (CP)

DATE: 28 / Feb / 2007 **No.:** **CP112E**

TCAS II Version: **DO-185A (v7)** X **Other (Specify)** _____

MOPS Function Area: **Surveillance** ____ **Display Req'ts** ____ **CRS** X

CAS Pseudocode X **Test Suites** X **Other** ____

Priority: **URGENT** X **Necessary** ____ **Optional** ____

CP Type: **ERROR** ____ **Enhancement** X **Evaluation Request** ____

Editorial (Logic) ____ **Editorial (Text)** ____

Description of Problem/Issue:

When compared with the previous TCAS II version 6.04a, one significant change included in TCAS II version 7.0 is the sense reversals that are now permitted in TCAS-TCAS encounters. This change was introduced to cope with changing situations where the original sense has clearly become the wrong thing to do, in particular when one of the pilots decides not to follow RAs.

Within the EUROCONTROL EMOTION7 Project, two areas of improvements have been identified for this change on the CAS logic area. These areas of improvement were referenced as issue SA01a and issue SA01b.

Issue SA01a
An area of improvement for the reversal logic of TCAS II logic version 7.0 has been identified in early 2000. It deals with the late issuance of reversal RAs in a geometry in which it is necessary to have them triggered well before.

The issue has been identified for an encounter set, which is representative of operationally realistic scenarios: two aircraft are flying at the same FL and are converging in range with a very late ATC instruction inducing an intruder manoeuvre that thwarts the initial RAs.

The EMOTION7 Project has provided evidences that the SA01a scenario was indeed happening in operational use (e.g., Japanese event January 2001, Belgian event July 2001, French event November 2001, German event February 2002, French Event March 2002, Überlingen collision July 2002)

Issue SA01b
The potential for failure to initiate the reversal logic has also been identified either in single equipage encounters or similarly in double equipage encounters but with one TCAS II either in stand-by mode or in TA-only mode.
The geometry involved is comparable to the one already described for issue SA01a.

Proposed Resolution:

CP112E is introducing changes to TCAS II logic version 7.0 in order to address issues SA01a and SA01b.

CP112E aims at detecting that the ongoing encounter is corresponding to the SA01a or SA01b geometry (i.e., two aircraft vertically close, climbing and descending towards the same point), and then possibly triggers a reversal RA if required. The detection is made using 2 means:

- By detecting that own is not following its RAs, and is manoeuvring opposite to it. In this case, the modelling performed in the reversal logic takes into account the fact that own is not following RAs;
- By using the Vertical Miss distance at CPA. This parameter is used to circumvent the mode S priority rule, which prevents the triggering of reversal RAs after the detection that own is going opposite to RAs, when own is the slave aircraft.

In addition, the conditions to reverse in case the SA01a or SA01b geometry is detected are weakened when compared with TCAS II logic version 7.0.

References:

1. Stéphan Chabert, “SIR Final Report”, EUROCONTROL, SIR/WP3/20/D, Version 1.2, 16 July 2004.
2. RTCA/DO-298, “Safety Analysis of Proposed Change to TCAS RA Reversal Logic,” 8 November, 2005.
3. RTCA Special Committee 147 Requirements Working Group, “Update to RTCA/DO-298 Safety Analysis of Proposed Change to TCAS RA Reversal Logic,” 15 November, 2007.
4. Men, H., “Translating Change Proposals 112E, 115, and 116 Pseudo-Code into Statechart Representation,” Johns Hopkins University Applied Physics Laboratory, 30 November, 2007.
5. Men, H., “Updating the Test Suite for DO185B,” Johns Hopkins University Applied Physics Laboratory, 30 November, 2007.

Requester: SIRE+ Project

Organization: EUROCONTROL

DISPOSITION OF CHANGE PROPOSAL (Per RWG):

DATE OF DISPOSITION 02 / 28 / 07

Rejected _____ **Deferred** _____ **[Review Date:** ____ / ____ / ____ **]**

Accepted X **Modified** _____ **Withdrawn** _____

DISPOSITION OF CHANGE:

On Hold _____ **Designing** _____ **Testing** _____ **Done** x **[Date:** 12 / 11 / 07 **]**

Final Approval of Changes:

Signature: Andy Zeitlin, RWG Chair
Date: 12 / 11 / 2007

<THIS PAGE INTENTIONALLY LEFT BLANK>

1. Pseudocode changes

High level Pseudocode BEFORE

PROCESS New_threat_file_entry;

<This process initializes existing or creates new threat file entry>

IF (intruder is Mode S-equipped)
THEN Search for threat file entry with same discrete address;
IF (matching entry not found)
THEN create new threat file entry;
Save threat's Mode S ID, if any, in TF entry;
Clear RA and intent indices;
Set threat intent refresh timer to initial negative value;
<indicates no intent received>
Clear advisory bit string;
<cleared bit 4 indicates no advisory present>
Save TF back pointer to ITF;
Indicate new threat for display;
Initialize the tiebreaker reversal flag;
Initialize the reversal modeling validity counter;
Initialize the geometric reversal flag;
Set own RA change timer to current time;
Save TF pointer in ITF;

END New_threat_file_entry;

High level Pseudocode AFTER

PROCESS New_threat_file_entry;

. <This process initializes existing or creates new threat file entry>

. IF (intruder is Mode S-equipped)
. . THEN Search for threat file entry with same discrete address;
. IF (matching entry not found)
. . THEN Create new threat file entry;
. . . . Save threat's Mode S ID, if any, in TF entry;
. . . . Clear RA and intent indices;
. . . . Set threat intent refresh timer to initial negative value;
. . . . . <indicates no intent received>
. . . . Clear advisory bit string;
. . . . . <cleared bit 4 indicates no advisory present>
. Save TF back pointer to ITF;
. Indicate new threat for display;
. Initialize the tiebreaker reversal flag;
. Initialize the reversal modeling validity counter;
. Initialize SA01 reversal flag;
. Initialize the geometric reversal flag;
. Set own RA change timer to current time;
. Save TF pointer in ITF;

END New_threat_file_entry;

Low level Pseudocode BEFORE

PROCESS New_threat_file_entry;

```
CLEAR SUCCESS;
IF (ITF.EQP NE $ATCRBS)
     THEN   REPEAT WHILE (more entries in TF AND SUCCESS EQ $FALSE);
                 IF (ITF.IDINT EQ TF.ID)
                      THEN SET SUCCESS;
                      ELSE select next TF entry;
            ENDREPEAT;
IF (SUCCESS EQ $FALSE)
     THEN   create new TF entry;
            TF.ID = ITF.IDINT;
            TF.POOWRAR, TF.POTHRAR(1), TF.POTHRAR(2) = 0;
            TF.TTHLRCM = P.TINIT;
            CLEAR All bits in TF.PERMTENT;
TF.IPTR = ITF.IROW;
SET TF.NEW;
CLEAR ITF.TIEBREAKER_REVERSAL;
ITF.VALREVS = 0;
CLEAR ITF.REV_GEOM;
ITF.TCMD = G.TCUR;
ITF.TPTR = address of TF entry;


END New_threat_file_entry;
```

Low level Pseudocode AFTER

PROCESS New_threat_file_entry;

```
.   CLEAR SUCCESS;
.   IF (ITF.EQP NE $ATCRBS)
.   .   THEN   REPEAT WHILE (more entries in TF AND SUCCESS EQ $FALSE);
.   .   .   .   .   IF (ITF.IDINT EQ TF.ID)
.   .   .   .   .   .   THEN   SET SUCCESS;
.   .   .   .   .   .   ELSE   select next TF entry;
.   .   .   .   ENDREPEAT;
.   IF (SUCCESS EQ $FALSE)
.   .   THEN   create new TF entry;
.   .   .   .   TF.ID = ITF.IDINT;
.   .   .   .   TF.POOWRAR, TF.POTHRAR(1), TF.POTHRAR(2) = 0;
.   .   .   .   TF.TTHLRCM = P.TINIT;
.   .   .   .   CLEAR All bits in TF.PERMTENT;
.   TF.IPTR = ITF.IROW;
.   SET TF.NEW;
.   CLEAR ITF.TIEBREAKER_REVERSAL;
.   ITF.VALREVS = 0;
.   ITF.CPT_REV = 0;
.   CLEAR ITF.REV_GEOM;
.   ITF.TCMD = G.TCUR;
.   ITF.TPTR = address of TF entry;

END New_threat_file_entry;
```

High level Pseudocode BEFORE

PROCESS Process_new_or_continuing_threat;

```
IF (status is 'continuing' AND threat is established)
    THEN PERFORM Reversal_check;
PERFORM Select_advisory;
IF (multiple threats this cycle)
    THEN   IF (own's advisory this cycle against current threat is not same as previous cycle)
               THEN save own's advisory this cycle against current threat;
    ELSE   PERFORM Update_threat_file_own;
           CALL RESOLUTION_UPDATE
               IN (advisory to delete, advisory to add);
IF (status is 'continuing' AND threat is established)
    THEN PERFORM Increase_check;


END Process_new_or_continuing_threat;
```

High level Pseudocode AFTER

PROCESS Process_new_or_continuing_threat;

```
    IF (less than 5 seconds have elapsed since initial RA against the threat)
.   .   THEN   set the computed time to follow initial RA to 0;
.   .   ELSE   IF (the time to follow initial RA is 0)
.   .   .   .   .   THEN compute time to follow initial RA;

.   IF (status is 'continuing' AND threat is established)
.   .   THEN   PERFORM Reversal_check;
.   PERFORM Select_advisory;
.   IF (multiple threats this cycle)
.   .   THEN   IF (own's advisory this cycle against current threat is not same as previous cycle)
.   .   .   .   .   THEN save own's advisory this cycle against current threat;
.   .   ELSE   PERFORM Update_threat_file_own;
.   .   .   .   CALL RESOLUTION_UPDATE
.   .   .   .   .   IN (advisory to delete, advisory to add);
.   IF (status is 'continuing' AND threat is established)
.   .   THEN   PERFORM Increase_check;

END Process_new_or_continuing_threat;
```

Low level Pseudocode BEFORE

```
PROCESS Process_new_or_continuing_threat;

    IF (WL.STATUS EQ $CONT AND ITF.KHIT EQ 3)
        THEN PERFORM Reversal_check;
    PERFORM Select_advisory;
    IF (G.MACFLG EQ $TRUE)
        THEN   IF (OWNTENT NE ITF.TPTR->TF.PERMTENT)
                    THEN ITF.TPTR->TF.PERMTENT = OWNTENT;
        ELSE   PERFORM Update_threat_file_own;
               Save current TF pointer;
               CALL RESOLUTION_UPDATE
                    IN (OLDPOI, OPTR);
               Restore current TF pointer;
    IF (WL.STATUS EQ $CONT AND ITF.KHIT EQ 3)
        THEN PERFORM Increase_check;

END Process_new_or_continuing_threat;
```

Low level Pseudocode AFTER

PROCESS Process_new_or_continuing_threat;

```
.   IF (G.TCUR LT G.TLASTNEWRA + P.TV1)
.   .   THEN   G.TTOFOLLOW = 0;
.   .   ELSE   IF (G.TTOFOLLOW EQ 0)
.   .   .   .   .   THEN G.TTOFOLLOW = ABS(G.ZDOWN - G.ZDMODEL)/P.VACCEL + P.TV1;

.   IF (WL.STATUS EQ $CONT AND ITF.KHIT EQ 3)
.   .   THEN   PERFORM Reversal_check;

.   PERFORM Select_advisory;
.   IF (G.MACFLG EQ $TRUE)
.   .   THEN   IF (OWNTENT NE ITF.TPTR->TF.PERMTENT)
.   .   .   .   .   THEN ITF.TPTR->TF.PERMTENT = OWNTENT;
.   .   ELSE   PERFORM Update_threat_file_own;
.   .   .   .   Save current TF pointer;
.   .   .   .   CALL RESOLUTION_UPDATE
.   .   .   .   .   IN (OLDPOI, OPTR);
.   .   .   .   Restore current TF pointer;
.   IF (WL.STATUS EQ $CONT AND ITF.KHIT EQ 3)
.   .   THEN   PERFORM Increase_check;

END Process_new_or_continuing_threat;
```

High level Pseudocode BEFORE

PROCESS Reversal_check;

```
    CLEAR flag to consider an increase rate RA;

    IF (no reversal has been issued)
        THEN IF (current RA is crossing)
                THEN IF (P.MIN_RI_TIME sec. or more remain AND range TAU did not
                        start rising when the threat was more than P.NAFRANGE
                        miles away)
                        THEN calculate int's proj. alt. at CPA using ITF.ZDINT;
                            PERFORM Reversal_proj_check;
                            IF (reversal not selected AND time to CPA is not sufficient
                                for reversal against threat which may be close in
                                 altitude AND intruder is not TCAS-equipped)
                                THEN SET flag to consider increase rate RA;
                ELSE PERFORM Cross_through_check;

    IF (past validity sequence is '100', '101', '110', or '111')
        THEN remove the leading '1';  <Subtract '100'>
            <No need to remove leading '0' if past validity sequence is '000', '001', '010', or '011'>
    Left shift sequence 1 bit;  <Multiply by 2>
    <New sequence is '000', '010', '100', or '110'>
    IF (reversal flag set this cycle)
        THEN add '1' to new sequence;
                <Sequence becomes '001', '011', '101', or '111'>
    <If reversal flag not set this cycle, sequence remains '000', '010', '100', or '110'>
                IF (intruder is TCAS-equipped AND ((own Mode S ID is higher) OR ((past
                    validity sequence is not '011', '101', or '111') AND (current RA is
                    crossing))))
                    THEN   CLEAR reversal flag;
                    ELSE   IF (intruder is TCAS-equipped OR only one threat has been declared)
                                THEN SET geometric reversal flag;
                           Reset validity counter to zero;
                           Select new sense;
                           PERFORM Set_up_for_advisory;
                           CLEAR increase flag in ITF;
                           Initialize increase rate RA counter;

    IF (intruder is TCAS-equipped)
        THEN IF (own Mode S ID is higher)
                THEN IF (threat has selected same sense as own)
                        THEN PERFORM Form_complement; <Reverse own sense>
                            Indicate that previous intent must be cancelled;
                            <Using reversal indication flag>
                            PERFORM Set_up_for_advisory;

    IF (a reversal has been selected)
        THEN indicate that a reversal RA is currently in effect;
                CLEAR indication of a forced level-off against the threat;

END Reversal_check;
```

High level Pseudocode AFTER

PROCESS Reversal_check;

```
.
.   SET flag that indicates that own is following his RA;
.   CLEAR flag that indicates a reversal has been considered because of vertical chase, low VMD geometry
.   CLEAR flag to consider an increase rate RA;

.   IF (no reversal has been issued)
.   .   THEN   IF (current RA is crossing)
.   .   .   .   .   THEN   IF (P.MIN_RI_TIME sec. or more remain AND range TAU did not start
.   .   .   .   .   .   .   .   rising when the threat was more than P.NAFRANGE miles away)
.   .   .   .   .   .   .   .   THEN   calculate int's proj. alt. at CPA using ITF.ZDINT;
.   .   .   .   .   .   .   .   .   .   PERFORM Reversal_proj_check;
.   .   .   .   .   .   .   .   .   .   IF (reversal not selected AND time to CPA is not sufficient for
.   .   .   .   .   .   .   .   .   .   .   reversal against threat which may be close in altitude AND
.   .   .   .   .   .   .   .   .   .   .   intruder is not TCAS-equipped)
.   .   .   .   .   .   .   .   .   .   .   THEN SET flag to consider increase rate RA;
.   .   .   .   .   ELSE   PERFORM Cross_through_check;
.   .   .   .   IF (past validity sequence for SA01 validity counter is '100', '101', '110', or '111')
.   .   .   .   .   THEN   remove the leading '1'; <Subtract '100'>;
.   .   .   .   Left shift sequence 1 bit;  <Multiply by 2>;
.   .   .   .   IF (a reversal was not selected AND only 1 threat has been declared AND
.   .   .   .   .   P.MIN_RI_TIME sec. or more remain AND
.   .   .   .   .   initial RA against this threat is more than 10 seconds old)
.   .   .   .   .   THEN   PERFORM RA_monitoring;

.   IF (past validity sequence is '100', '101', '110', or '111')
.   .   THEN   remove the leading '1';  <Subtract '100'>
.   .   .   <No need to remove leading '0' if past validity sequence is '000', '001', '010', or '011'>
.   Left shift sequence 1 bit;  <Multiply by 2>
.   <New sequence is '000', '010', '100', or '110'>
.   IF (reversal flag set this cycle)
.   .   THEN   add '1' to new sequence;
.   .   .   .   <Sequence becomes '001', '011', '101', or '111'>
.   .   .   .   <If reversal flag not set this cycle, sequence remains '000', '010', '100', or '110'>
.   .   .   .   IF (intruder is TCAS-equipped AND ((own Mode S ID is higher) OR
.   .   .   .   .   ((past validity sequence is not ‘011’, '101', or '111') AND
.   .   .   .   .   (past validity sequence for SA01 reversal flag is not ‘011’, '101', or '111') AND
.   .   .   .   .   (own is following his RA) AND (current RA is crossing))))
.   .   .   .   .   THEN   CLEAR reversal flag
.   .   .   .   .   ELSE   IF (intruder is TCAS-equipped OR only one threat has been declared)
.   .   .   .   .   .   .   .   THEN   SET geometric reversal flag;
.   .   .   .   .   .   .   Reset validity counter to zero;
.   .   .   .   .   .   .   Reset SA01 validity counter to zero;
.   .   .   .   .   .   .   Select new sense;
.   .   .   .   .   .   .   PERFORM Set_up_for_advisory;
.   .   .   .   .   .   .   CLEAR increase flag in ITF;
.   .   .   .   .   .   .   Initialize increase rate RA counter;

.   IF (intruder is TCAS-equipped)
.   .   THEN   IF (own Mode S ID is higher)
.   .   .   .   .   THEN   IF (threat has selected same sense as own)
.   .   .   .   .   .   .   .   THEN   PERFORM Form_complement; <Reverse own sense>
.   .   .   .   .   .   .   .   .   .   Indicate that previous intent must be cancelled;
.   .   .   .   .   .   .   .   .   .   <Using reversal indication flag>
.   .   .   .   .   .   .   .   .   .   PERFORM Set_up_for_advisory;
```

. IF (a reversal has been selected)
. . THEN indicate that a reversal RA is currently in effect;
. . . . CLEAR indication of a forced level-off against the threat;

END Reversal_check;

<THIS PAGE INTENTIONALLY LEFT BLANK>

Low level Pseudocode BEFORE

```
PROCESS Reversal_check;

    CLEAR CONSIDER_INCREASE;

    IF (ITF.REV_GEOM EQ $FALSE)
        THEN IF (ITF.INT_CROSS EQ $TRUE OR ITF.OWN_CROSS EQ $TRUE)
                THEN IF (ITF.TRTRU GT P.MIN_RI_TIME AND ITF.TAURISE LT
                            P.TAURISE_THR)
                        THEN  PROJ_ZINT = ITF.ZINT + (ITF.ZDINT *
                                MIN(P.TVPETBL(ITF.LEV), ITF.TRTRU));
                              PERFORM Reversal_proj_check;
                              IF (ITF.REVERSE EQ $FALSE AND ITF.TRTRU LE
                                    P.MINRVSTIME AND ITF.EQP NE $TCAS)
                                  THEN SET CONSIDER_INCREASE;
                ELSE PERFORM Cross_through_check;

    IF (ITF.VALREVS GT 3)
        THEN ITF.VALREVS = ITF.VALREVS – 4;
    ITF.VALREVS = 2 * ITF.VALREVS;
    IF (ITF.REVERSE EQ $TRUE)
        THEN ITF.VALREVS = ITF.VALREVS + 1;
             IF (ITF.EQP EQ $TCAS AND ((G.IDOWN GT ITF.IDINT) OR
                 ((ITF.VALREVS NE 3, 5, or 7) AND (ITF.INT_CROSS EQ $TRUE
                 OR ITF.OWN_CROSS EQ $TRUE))))
                THEN  CLEAR ITF.REVERSE;
                ELSE  IF (ITF.EQP EQ $TCAS OR G.MACFLG EQ $FALSE)
                          THEN SET ITF.REV_GEOM;
                      ITF.VALREVS = 0;
                      OWNTENT(7) = NEW_SENSE;
                      PERFORM Set_up_for_advisory;
                      CLEAR ITF.INCREASE;
                      ITF.INCTEST = 0;

    IF (ITF.EQP EQ $TCAS)
        THEN IF (G.IDOWN GT ITF.IDINT)
                THEN IF ((TF.POTHRAR(1) EQ 1 AND OWNTENT(7) EQ $TRUE) OR
                            (TF.POTHRAR(1) EQ 2 AND OWNTENT(7) EQ $FALSE))
                        THEN  PERFORM Form_complement;
                              SET ITF.TIEBREAKER_REVERSAL;
                              SET ITF.REVERSE;
                              PERFORM Set_up_for_advisory;

    IF (ITF.REVERSE EQ $TRUE)
        THEN  SET ITF.REV_RA;
              CLEAR TF.TTLO;
END Reversal_check;
```

Low level Pseudocode AFTER

PROCESS Reversal_check;

```
.   SET G.OWN_FOLLOW;
.   CLEAR G.REV_CONSDRD;
.   CLEAR CONSIDER_INCREASE;

.   IF (ITF.REV_GEOM EQ $FALSE)
.   .   THEN   IF (ITF.INT_CROSS EQ $TRUE OR ITF.OWN_CROSS EQ $TRUE)
.   .   .   .   .   THEN   IF (ITF.TRTRU GT P.MIN_RI_TIME AND
.   .   .   .   .   .   .   .   .   ITF.TAURISE LT P.TAURISE_THR)
.   .   .   .   .   .   .   .   .   THEN   PROJ_ZINT = ITF.ZINT + (ITF.ZDINT *
.   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   MIN(P.TVPETBL(ITF.LEV), ITF.TRTRU));
.   .   .   .   .   .   .   .   .   .   PERFORM Reversal_proj_check;
.   .   .   .   .   .   .   .   .   .   IF (ITF.REVERSE EQ $FALSE AND
.   .   .   .   .   .   .   .   .   .   .   ITF.TRTRU LE P.MINRVSTIME AND ITF.EQP NE $TCAS)
.   .   .   .   .   .   .   .   .   .   .   THEN   SET CONSIDER_INCREASE;
.   .   .   .   .   ELSE   PERFORM Cross_through_check;
.   .   .   .   IF (ITF.CPT_REV GT 3)
.   .   .   .   .   THEN   ITF.CPT_REV = ITF.CPT_REV – 4;
.   .   .   .   ITF.CPT_REV = 2 * ITF.CPT_REV;
.   .   .   .   IF (ITF.REVERSE EQ $FALSE AND G.MACFLG EQ $FALSE AND
.   .   .   .   .   ITF.TRTRU GT P.MIN_RI_TIME AND (G.TCUR - G.TLASTNEWRA) GE 10)
.   .   .   .   .   THEN   PERFORM RA_monitoring;

.   IF (ITF.VALREVS GT 3)
.   .   THEN   ITF.VALREVS = ITF.VALREVS – 4;
.   ITF.VALREVS = 2 * ITF.VALREVS;
.   IF (ITF.REVERSE EQ $TRUE)
.   .   THEN   ITF.VALREVS = ITF.VALREVS + 1;
.   .   .   .   IF (ITF.EQP EQ $TCAS AND ((G.IDOWN GT ITF.IDINT) OR
.   .   .   .   .   ((ITF.VALREVS NE 3, 5, or 7) AND (ITF.CPT_REV NE 3, 5, or 7) AND
.   .   .   .   .   (G.OWN_FOLLOW EQ $TRUE) AND
.   .   .   .   .   (ITF.INT_CROSS EQ $TRUE OR ITF.OWN_CROSS EQ $TRUE))))
.   .   .   .   .   THEN   CLEAR ITF.REVERSE;
.   .   .   .   .   ELSE   IF (ITF.EQP EQ $TCAS OR G.MACFLG EQ $FALSE)
.   .   .   .   .   .   .   .   THEN   SET ITF.REV_GEOM;
.   .   .   .   .   .   .   ITF.VALREVS = 0;
.   .   .   .   .   .   .   ITF.CPT_REV = 0;
.   .   .   .   .   .   .   OWNTENT(7) = NEW_SENSE;
.   .   .   .   .   .   .   PERFORM Set_up_for_advisory;
.   .   .   .   .   .   .   CLEAR ITF.INCREASE;
.   .   .   .   .   .   .   ITF.INCTEST = 0;

.   IF (ITF.EQP EQ $TCAS)
.   .   THEN   IF (G.IDOWN GT ITF.IDINT)
.   .   .   .   .   THEN   IF ((TF.POTHRAR(1) EQ 1 AND OWNTENT(7) EQ $TRUE) OR
.   .   .   .   .   .   .   .   .   (TF.POTHRAR(1) EQ 2 AND OWNTENT(7) EQ $FALSE))
.   .   .   .   .   .   .   .   .   THEN   PERFORM Form_complement;
.   .   .   .   .   .   .   .   .   .   .   SET ITF.TIEBREAKER_REVERSAL;
.   .   .   .   .   .   .   .   .   .   .   SET ITF.REVERSE;
.   .   .   .   .   .   .   .   .   .   .   PERFORM Set_up_for_advisory;

.   IF (ITF.REVERSE EQ $TRUE)
.   .   THEN   SET ITF.REV_RA;
.   .   .   .   CLEAR TF.TTLO;
```

END Reversal_check;

<THIS PAGE INTENTIONALLY LEFT BLANK>

High level Pseudocode BEFORE

PROCESS Reversal_modeling;

Set own altitude and own rate to own tracked altitude and own tracked rate;
IF (current RA is positive)
THEN model response to current RA;
<model maximum displayable rate for climb if current rate exceeds maximum displayable rate or minimum displayable rate for descent if current rate is less than minimum displayable rate>

IF (tracked response lags modeled response in RA direction AND time since RA less than a parameter time AND own's rate has not changed by more than P.MODEL_ZD since the RA was first issued)
THEN set own altitude and own rate to modeled altitude and rate for use in reversal modeling;
Model separation achieved by continuing current RA;

Set delay time to greater of pilot delay time remaining for last advisory against a new threat, and the pilot quick reaction time;

IF (considering a reversal from a descend RA to a climb RA)
THEN set own goal rate to greater of own tracked rate (or maximum displayable rate, whichever is less) and nominal climb rate;
ELSE IF(own too close to ground to descend)
THEN set own goal rate to zero;
ELSE set own goal rate to lesser of own tracked rate (or minimum displayable rate, whichever is greater) and nominal descent rate;

IF (intruder causing crossing OR intruder level and own crossing from above OR intruder rate and own modeled rate are opposite in sign)
THEN use outer rate bound to model intruder;
ELSE use inner rate bound to model intruder;

CALL MODEL_SEP
IN (delay, goal rate, own altitude, own rate, acceleration response, sense after reversal, intruder altitude, modeled intruder rate, ITF entry)
OUT (predicted separation for sense reversal);

IF (predicted separation for sense reversal is not positive OR modeled separation achieved by continuing current RA GE G.ALIM)
THEN CLEAR reversal flag in ITF;

END Reversal_modeling;

High level Pseudocode AFTER

PROCESS Reversal_modeling;

```
. Default modeled separation for current RA is 0 if current RA is negative;
. Set own altitude and own rate to own tracked altitude and own tracked rate;
. IF (own does not follow his RAs)
. . THEN Model separation achieved assuming RA not followed;
. . . . IF (current RA is a climb RA)
. . . . . THEN CLEAR flag indicating the sense of the RA after a reversal;
. . . . . ELSE SET flag indicating the sense of the RA after a reversal;
. . . . IF (modeled separation achieved by continuing current RA greater than 1.2 * P.CROSSTHR)
. . . . . THEN CLEAR reversal flag in ITF
. . ELSE
. . <Begin own is assumed to follow its RA>
. . . . IF (current RA is positive)
. . . . . THEN model response to current RA;
. . . . . . . <model maximum displayable rate for climb if current rate exceeds maximum
. . . . . . . displayable rate or minimum displayable rate for descent if current rate is less
. . . . . . . than minimum displayable rate>

. . . . . . . IF (tracked response lags modeled response in RA direction AND
. . . . . . . . time since RA less than a parameter time AND
. . . . . . . . own's rate has not changed by more than P.MODEL_ZD since the RA was
. . . . . . . . first issued)
. . . . . . . . THEN set own altitude and own rate to modeled altitude and rate for use in
. . . . . . . . . . reversal modeling;
. . . . . . . Model separation achieved by continuing current RA;

. . . . Set delay time to greater of pilot delay time remaining for last advisory against a new threat,
. . . . and the pilot quick reaction time;

. . . . IF (considering a reversal from a descend RA to a climb RA)
. . . . . THEN set own goal rate to greater of own tracked rate (or maximum displayable rate,
. . . . . . . whichever is less) and nominal climb rate;
. . . . . ELSE IF (own too close to ground to descend)
. . . . . . . . THEN set own goal rate to zero;
. . . . . . . . ELSE set own goal rate to lesser of own tracked rate (or minimum
. . . . . . . . . . displayable rate, whichever is greater) and nominal descent rate;
. . . . IF (vertical chase, low VMD geometry was not the reason for considering reversal)
. . . . . THEN IF (intruder causing crossing OR (intruder level AND own crossing from above)
. . . . . . . . OR intruder rate and own modeled rate are opposite in sign)
. . . . . . . . THEN use outer rate bound to model intruder;
. . . . . . . . ELSE use inner rate bound to model intruder;
. . . . . ELSE use intruder's tracked vertical rate to model intruder;
. . . . CALL MODEL_SEP
. . . . . IN (delay, goal rate, own altitude, own rate, acceleration response, sense after
. . . . . . reversal, intruder altitude, modeled intruder rate, ITF entry)
. . . . . OUT (predicted separation for sense reversal);

. . . . IF (Predicted separation for sense reversal is not positive OR
. . . . . modeled separation achieved by continuing current RA GE G.ALIM)
. . . . . THEN CLEAR reversal flag in ITF;
. . <End own is assumed to follow its RA>
END Reversal_modeling;
```

Low level Pseudocode BEFORE

```
PROCESS Reversal_modeling;

NOMINAL_SEP = 0;
Z = G.ZOWN;
ZD = G.ZDOWN;
IF (OWNTENT(5, 6) EQ '00')
    THEN   DELAY = MAX(P.TV1 - (G.TCUR - G.TPOSRA), 0);
           IF (OWNTENT(7) EQ $FALSE)
               THEN   ZDGOAL = MAX(MIN(G.ZDOWN, P.MAXDRATE), P.CLMRT);
               ELSE   ZDGOAL = MIN(MAX(G.ZDOWN, P.MINDRATE), P.DESRT);
           CALL PROJECT_VERTICAL_GIVEN_ZDGOAL
               IN ((G.TCUR - G.TPOSRA), G.ZTV, G.ZDTV, ZDGOAL, P.TV1,
                   P.VACCEL)
               OUT (ZPROJ, ZDPROJ);
           IF (((OWNTENT(7) EQ $FALSE AND ZPROJ GT G.ZOWN
                   AND (G.ZDOWN GE G.ZDTV - P.MODEL_ZD)) OR
                   (OWNTENT(7) EQ $TRUE AND ZPROJ LT G.ZOWN
                   AND (G.ZDOWN LE G.ZDTV + P.MODEL_ZD)))
                   AND G.TCUR – G.TPOSRA LT P.MODEL_T)
               THEN   Z = ZPROJ;
                      ZD = ZDPROJ;
           CALL MODEL_SEP
               IN (DELAY, ZDGOAL, Z, ZD, P.VACCEL, OWNTENT(7), ITF.ZINT,
                   ITF.ZDINT, ITF entry)
               OUT (NOMINAL_SEP);
IF (OWNTENT(7) EQ $TRUE)
    THEN NEW_SENSE = $FALSE;
    ELSE NEW_SENSE = $TRUE;
DELAY = MAX(P.TV1 - (G.TCUR - G.TLASTNEWRA), P.QUIKREAC);

IF (NEW_SENSE EQ $FALSE)
    THEN   ZDGOAL = MAX(P.CLMRT, MIN(G.ZDOWN, P.MAXDRATE));
    ELSE   IF (G.NODESCENT EQ $TRUE)
               THEN ZDGOAL = 0;
               ELSE ZDGOAL = MIN(P.DESRT, MAX(G.ZDOWN, P.MINDRATE));

IF ((ITF.INT_CROSS EQ $TRUE) OR (ITF.ZDINT EQ 0 AND ITF.RZ GT 0) OR
           (ITF.ZDINT * G.ZDMODEL LT 0))
    THEN MZDINT= ITF.ZDOUTR;
    ELSE MZDINT= ITF.ZDINR;

CALL MODEL_SEP
    IN (DELAY, ZDGOAL, Z, ZD, P.RACCEL, NEW_SENSE, ITF.ZINT, MZDINT,
        ITF entry)
    OUT (ZMP);

IF (ZMP LE 0 OR NOMINAL_SEP GE G.ALIM)
    THEN CLEAR ITF.REVERSE;
END Reversal_modeling;
```

Low level Pseudocode AFTER

PROCESS Reversal_modeling;

```
.  NOMINAL_SEP = 0;
.  Z = G.ZOWN;
.  ZD = G.ZDOWN;
.  DELAY = 0;

.  IF (G.OWN_FOLLOW EQ FALSE)
.  .  THEN  CALL MODEL_SEP
.  .  .  .  .  IN (DELAY, ZD, Z, ZD, P.VACCEL, OWNTENT(7), ITF.ZINT, ITF.ZDINT, ITF entry)
.  .  .  .  .  OUT (NOMINAL_SEP);
.  .  .  .  IF (OWNTENT(7) EQ $TRUE)
.  .  .  .  .  THEN  NEW_SENSE = $FALSE;
.  .  .  .  .  ELSE  NEW_SENSE = $TRUE;
.  .  .  .  IF (NOMINAL_SEP GT 1.2 * P.CROSSTHR)
.  .  .  .  .  THEN  CLEAR ITF.REVERSE;
.  .  ELSE
.  .  <Begin own is assumed to follow its RA>
.  .  .  .  IF (OWNTENT(5.6) EQ '00')
.  .  .  .  .  THEN  DELAY = MAX(P.TV1 - (G.TCUR - G.TPOSRA), 0);
.  .  .  .  .  .  .  IF (OWNTENT(7) EQ $FALSE)
.  .  .  .  .  .  .  .  THEN  ZDGOAL = MAX(MIN(G.ZDOWN, P.MAXDRATE), P.CLMRT);
.  .  .  .  .  .  .  .  ELSE  ZDGOAL = MIN(MAX(G.ZDOWN, P.MINDRATE), P.DESRT);
.  .  .  .  .  .  .  CALL PROJECT_VERTICAL_GIVEN_ZDGOAL
.  .  .  .  .  .  .  .  IN ((G.TCUR - G.TPOSRA), G.ZTV, G.ZDTV, ZDGOAL, P.TV1,
.  .  .  .  .  .  .  .  .  P.VACCEL)
.  .  .  .  .  .  .  .  OUT (ZPROJ, ZDPROJ);
.  .  .  .  .  .  .  IF  (((OWNTENT(7) EQ $FALSE AND ZPROJ GT G.ZOWN AND
.  .  .  .  .  .  .  .  (G.ZDOWN GE G.ZDTV - P.MODEL_ZD)) OR
.  .  .  .  .  .  .  .  (OWNTENT(7) EQ $TRUE AND ZPROJ LT G.ZOWN AND
.  .  .  .  .  .  .  .  (G.ZDOWN LE G.ZDTV + P.MODEL_ZD))) AND
.  .  .  .  .  .  .  .  G.TCUR – G.TPOSRA LT P.MODEL_T)
.  .  .  .  .  .  .  .  THEN  Z = ZPROJ;
.  .  .  .  .  .  .  .  .  .  ZD = ZDPROJ;
.  .  .  .  .  .  .  CALL MODEL_SEP
.  .  .  .  .  .  .  .  IN (DELAY, ZDGOAL, Z, ZD, P.VACCEL, OWNTENT(7), ITF.ZINT,
.  .  .  .  .  .  .  .  .  ITF.ZDINT, ITF entry)
.  .  .  .  .  .  .  .  OUT (NOMINAL_SEP);

.  .  .  .  IF (OWNTENT(7) EQ $TRUE)
.  .  .  .  .  THEN  NEW_SENSE = $FALSE;
.  .  .  .  .  ELSE  NEW_SENSE = $TRUE;
.  .  .  .  DELAY = MAX(P.TV1 - (G.TCUR - G.TLASTNEWRA), P.QUIKREAC);

.  .  .  .  IF (NEW_SENSE EQ $FALSE)
.  .  .  .  .  THEN  ZDGOAL = MAX(P.CLMRT, MIN(G.ZDOWN, P.MAXDRATE));
.  .  .  .  .  ELSE  IF (G.NODESCENT EQ $TRUE)
.  .  .  .  .  .  .  .  THEN  ZDGOAL = 0;
.  .  .  .  .  .  .  .  ELSE  ZDGOAL = MIN(P.DESRT, MAX(G.ZDOWN, P.MINDRATE));
.  .  .  .  IF (G.REV_CONSDRD EQ FALSE)
.  .  .  .  .  THEN  IF ((ITF.INT_CROSS EQ $TRUE) OR (ITF.ZDINT EQ 0 AND ITF.RZ GT 0)
.  .  .  .  .  .  .  OR (ITF.ZDINT * G.ZDMODEL LT 0))
.  .  .  .  .  .  .  THEN  MZDINT = ITF.ZDOUTR;
.  .  .  .  .  .  .  ELSE  MZDINT = ITF.ZDINR;
.  .  .  .  .  ELSE  MZDINT = ITF.ZDINT;
```

```
. . . . CALL MODEL_SEP
. . . . . IN (DELAY, ZDGOAL, Z, ZD, P.RACCEL, NEW_SENSE, ITF.ZINT, MZDINT,
. . . . . . ITF entry)
. . . . . OUT (ZMP);

. . . . IF (ZMP LE 0 OR NOMINAL_SEP GE G.ALIM)
. . . . . THEN CLEAR ITF.REVERSE;
. . <End own is assumed to follow its RA>
END Reversal_modeling;
```

<THIS PAGE INTENTIONALLY LEFT BLANK>

High level Pseudocode NEW

PROCESS RA_monitoring;

. Set G.BOUNDED_TTF as the bounded value of G.TTOFOLLOW between 10 and 15 seconds;

. IF (intruder is TCAS-equipped)
. . THEN set T_RZ to the maximum of the times required for own aircraft and the intruder to level off;
. . ELSE set T_RZ to the time required for own aircraft to level off;
. Compute projected altitude of own at CPA using the current vertical rate;

. Compute maximum, minimum and tracked projected altitudes of intruder at CPA using the current vertical
. rate of intruder, and intruder's vertical rate limits;
. Compute the maximum, minimum and tracked projected vertical separations at CPA;

. Set values of FACT_MULT and DELTA_T_RZ according to intruder equipage, current vertical rate of own
. and intruder;

. IF (an increase RA was triggered in the past five seconds)
. . THEN use a reduced value for FACT_MULT;

. IF ((at least (T_RZ + DELTA_T_RZ) seconds remain) AND (convergence is not slow) AND
. . (at least 10 seconds remain))
. . THEN set THRES_RZ to -FACT_MULT * P.CROSSTHR;
. . ELSE IF ((at least (T_RZ + DELTA_T_RZ) seconds remain) AND (convergence is not slow))
. . . . . THEN set THRES_RZ to -FACT_MULT * P.CROSSTHR * 0.25;
. . . . . ELSE set THRES_RZ to 0;

. IF (firmness is not good AND CPA close)
. . THEN use increased values of THRES_RZ;
. PERFORM Take_decision;

END RA_monitoring;

Low level Pseudocode NEW

```
PROCESS RA_monitoring;

.   G.BOUNDED_TTF = MIN(MAX(10, G.TTOFOLLOW), 15);

.   IF (ITF.EQP EQ $TCAS)
.   .   THEN   T_RZ = MAX(ABS(G.ZDOWN)/P.RACCEL, ABS(ITF.ZDINT)/P.RACCEL) +
.   .   .   .   .   P.QUIKREAC;
.   .   ELSE   T_RZ = ABS(G.ZDOWN)/P.RACCEL + P.QUIKREAC;

.   Z = G.ZOWN + G.ZDOWN * MIN(P.TVPETBL(ITF.LEV), ITF.TRTRU);

.   ZI = ITF.ZINT + ITF.ZDINT * MIN(P.TVPETBL(ITF.LEV), ITF.TRTRU);
.   ZI_IN = ITF.ZINT + ITF.ZDINR * MIN(P.TVPETBL(ITF.LEV), ITF.TRTRU);
.   ZI_OUT = ITF.ZINT + ITF.ZDOUTR * MIN(P.TVPETBL(ITF.LEV), ITF.TRTRU);

.   DELTA_Z_CPA = Z - ZI;
.   DELTA_Z_MIN = MIN(Z - ZI_OUT, Z - ZI, Z - ZI_IN);
.   DELTA_Z_MAX = MAX(Z - ZI_OUT, Z - ZI, Z - ZI_IN);

.   IF (ITF.EQP EQ $TCAS)
.   .   THEN   IF (ABS(G.ZDOWN) GT 3 * P.ILEV AND ABS (ITF.ZDINT) GT 3 * P.ILEV)
.   .   .   .   .   THEN   FACT_MULT = 1.5;
.   .   .   .   .   .   .   DELTA_T_RZ = 3.0;
.   .   .   .   .   ELSE   IF (ABS(G.ZDOWN) GT 2 * P.ILEV AND ABS (ITF.ZDINT) GT 2 * P.ILEV)
.   .   .   .   .   .   .   .   THEN   FACT_MULT = 1.25;
.   .   .   .   .   .   .   .   .   .   DELTA_T_RZ = 3.0;
.   .   .   .   .   .   .   .   ELSE   IF (ABS(G.ZDOWN) GT 1.5 * P.ILEV AND
.   .   .   .   .   .   .   .   .   .   .   ABS(ITF.ZDINT) GT 1.5 * P.ILEV)
.   .   .   .   .   .   .   .   .   .   .   THEN   FACT_MULT = 1.1;
.   .   .   .   .   .   .   .   .   .   .   .   .   DELTA_T_RZ = 3.0;
.   .   .   .   .   .   .   .   .   .   .   ELSE   FACT_MULT = 1.0;
.   .   .   .   .   .   .   .   .   .   .   .   .   DELTA_T_RZ = 5.0;
.   .   ELSE   FACT_MULT = 1.0;
.   .   .   .   DELTA_T_RZ = 5.0;

.   IF ((G.TCUR - ITF.TCMD) LE P.TV1 AND G.ANYINCREASE EQ $TRUE)
.   .   THEN   FACT_MULT = FACT_MULT/4;

.   IF ((ITF.TRTRU GE (T_RZ + DELTA_T_RZ)) AND (ITF.RD LT -5 * P.RDTHR) AND
.   .   (ITF.TRTRU GT P.MINRVSTIME))
.   .   THEN   THRES_RZ = -FACT_MULT * P.CROSSTHR;
.   .   ELSE   IF ((ITF.TRTRU GE (T_RZ + DELTA_T_RZ)) AND (ITF.RD LT -5 * P.RDTHR))
.   .   .   .   .   THEN   THRES_RZ = -FACT_MULT * P.CROSSTHR * 0.25;
.   .   .   .   .   ELSE   THRES_RZ = 0;

.   IF (ITF.TRTRU LE P.MINRVSTIME AND ITF.IFIRM LT P.MINFIRM)
.   .   THEN   THRES_RZ = THRES_RZ + P.CROSSTHR;
.   .   ELSE   IF (ITF.TRTRU LE P.MINRVSTIME AND ITF.IFIRM EQ P.MINFIRM)
.   .   .   .   .   THEN   THRES_RZ = THRES_RZ + P.CROSSTHR/2;

.   PERFORM Take_decision;

END RA_monitoring;
```

High level Pseudocode NEW

```
PROCESS Take_decision;

.   IF (both own and intruder tracked vertical rates have the same sign
.   .   AND own is not level
.   .   AND intruder is not level
.   .   AND enough time has elapsed since the initial RA for the pilot to achieve the goal rate for the RA
.   .   AND (there is enough time to level off before CPA
.   .   .   OR no increase RA has been issued in the previous 5 processing cycles))
.   THEN   IF (worst-case projected separation at CPA is too small or in the wrong direction for current RA
.   .   .   .   AND the aircraft are separated by less than P.MAXALTDIFF (600 ft) in altitude
.   .   .   .   AND (intruder aircraft is TCAS-equipped, and so should reverse if own TCAS reverses
.   .   .   .   .   OR own aircraft appears to be complying with RAs, and so should reverse if TCAS
.   .   .   .   .   .   reverses)
.   .   .   .   AND ((current RA sense is climb AND intruder is at least THRES_RZ above own)
.   .   .   .   .   OR (current RA sense is descend AND intruder is at least THRES_RZ below own)))
.   .   .   THEN   increment ITF.CPT_REV by 1;
.   .   .   .   .   IF (no increase RA was triggered in the previous cycle
.   .   .   .   .   .   AND ITF.CPT_REV is equal to 3, 5, or 7)
.   .   .   .   .   THEN   SET reversal flag in ITF;
.   .   .   .   .   .   .   SET flag used to signal this reversal case to Reversal_modeling;
.   .   .   .   .   .   .   PERFORM Reversal_modeling;

.   .   .   IF (no increase RA was triggered in the previous cycle
.   .   .   .   AND reversal flag in ITF structure is not already set
.   .   .   .   AND projected vertical separation at CPA is less than ALIM/2 ft or in wrong sense for RA
.   .   .   .   AND intruder aircraft is TCAS-equipped, and so should reverse if own TCAS reverses
.   .   .   .   AND current RA is not a VSL RA
.   .   .   .   AND own aircraft vertical rate is at least 1.2 * P.ILEV (1200 fpm) in opposite sense from
.   .   .   .   .   current RA
.   .   .   .   AND ((current RA is positive AND (it was initial RA OR has been in effect at least P.TV1 (5)
.   .   .   .   .   .   sec)
.   .   .   .   .   OR (current RA is negative AND own vertical rate magnitude is at least 1.5 * P.ILEV
.   .   .   .   .   .   (1500 fpm))))
.   .   .   THEN   IF (there are more than P.MINRVSTIME (10) seconds until CPA
.   .   .   .   .   .   AND intruder's altitude tracking confidence is below the minimum confidence)
.   .   .   .   .   THEN increase THRES_RZ by P.CROSSTHR/2 (50) ft;
.   .   .   .   .   ELSE   IF (there are more than P.MINRVSTIME (10) seconds until CPA
.   .   .   .   .   .   .   .   AND intruder's altitude tracking confidence is equal to minimum)
.   .   .   .   .   .   .   THEN increase THRES_RZ by P.CROSSTHR/4 (25) ft;
.   .   .   .   .   IF ((current RA sense is climb AND intruder is at least THRES_RZ above own)
.   .   .   .   .   .   OR (current RA sense is descend AND intruder is at least THRES_RZ below own))
.   .   .   .   .   THEN   SET reversal flag in ITF;
.   .   .   .   .   .   .   CLEAR flag that indicates own is following its RA;
.   .   .   .   .   .   .   PERFORM Reversal_modeling;

.   Compute altitudes of own and intruder 2.5 seconds from now using current vertical rates;
.   IF (reversal flag in ITF is set
.   .   AND (the aircraft are converging vertically faster than P.OLEV (600) fpm
.   .   .   OR are separated by more than P.CROSSTHR/4 (25) ft)
.   .   AND there is less than 1.25 * P.MINRVSTIME (12.5) seconds until CPA
.   .   AND (current RA is a crossing RA and the aircraft are projected to cross in altitude
.   .   .   .   before the pilot is expected to be able to respond to the reversal)
.   .   .   THEN   CLEAR reversal flag in ITF;
END Take_decision;
```

Low level Pseudocode NEW

```
PROCESS Take_decision;

.   IF (G.ZDOWN * ITF.ZDINT GT 0
.   .   AND ABS(G.ZDOWN) GT P.ILEV
.   .   AND ABS(ITF.ZDINT) GT P.ILEV
.   .   AND (G.TCUR - G.TLASTNEWRA) GE G.BOUNDED_TTF
.   .   AND (ITF.TRTRU GT T_RZ OR (G.TCUR - ITF.TCMD) GT P.TV1 OR G.ANYINCREASE EQ
.   .   .   .   $FALSE))
.   THEN   IF (((OWNTENT(7) EQ $FALSE AND DELTA_Z_MIN LT 1.2 * P.CROSSTHR)
.   .   .   .   .   OR (OWNTENT(7) EQ $TRUE AND DELTA_Z_MAX GT -1.2 * P.CROSSTHR))
.   .   .   .   AND ABS(ITF.RZ) LT P.MAXALTDIFF
.   .   .   .   AND (ITF.EQP EQ $TCAS OR G.ZDOWN * G.ZDMODEL GT 0)
.   .   .   .   AND ((OWNTENT(7) EQ $FALSE AND ITF.RZ LE -THRES_RZ)
.   .   .   .   .   OR (OWNTENT(7) EQ $TRUE AND ITF.RZ GE THRES_RZ)))
.   .   .   THEN   ITF.CPT_REV = ITF.CPT_REV + 1;
.   .   .   .   .   IF ((G.ANYINCREASE EQ $FALSE OR (G.TCUR - ITF.TCMD) GT 1)
.   .   .   .   .   .   AND (ITF.CPT_REV EQ 3 OR ITF.CPT_REV EQ 5 OR ITF.CPT_REV EQ 7))
.   .   .   .   .   THEN   SET ITF.REVERSE;
.   .   .   .   .   .   .   SET G.REV_CONSDRD;
.   .   .   .   .   .   .   PERFORM Reversal_modeling;

.   .   .   IF ((G.ANYINCREASE EQ $FALSE OR (G.TCUR - ITF.TCMD) GT 1)
.   .   .   .   AND ITF.REVERSE EQ $FALSE
.   .   .   .   AND ((OWNTENT(7) EQ $FALSE AND DELTA_Z_CPA LT G.ALIM/2)
.   .   .   .   .   OR (OWNTENT(7) EQ $TRUE AND DELTA_Z_CPA GT –G.ALIM/2))
.   .   .   .   AND ITF.EQP EQ $TCAS
.   .   .   .   AND OWNTENT(6) EQ $FALSE
.   .   .   .   AND ((G.ZDOWN LT -1.2 * P.ILEV AND OWNTENT(7) EQ $FALSE)
.   .   .   .   .   OR (G.ZDOWN GT 1.2 * P.ILEV AND OWNTENT(7) EQ $TRUE))
.   .   .   .   AND ((OWNTENT(5) EQ $FALSE
.   .   .   .   .   .   AND (G.TPOSRA EQ G.TLASTNEWRA OR (G.TCUR - G.TPOSRA) GT P.TV1))
.   .   .   .   .   OR (OWNTENT(5) EQ $TRUE AND ABS(G.ZDOWN) GT 1.5 * P.ILEV)))
.   .   .   THEN   IF (ITF.TRTRU GT P.MINRVSTIME AND ITF.IFIRM LT P.MINFIRM)
.   .   .   .   .   THEN   THRES_RZ = THRES_RZ + P.CROSSTHR/2;
.   .   .   .   .   ELSE   IF (ITF.TRTRU GT P.MINRVSTIME AND ITF.IFIRM EQ P.MINFIRM)
.   .   .   .   .   .   .   THEN THRES_RZ = THRES_RZ + P.CROSSTHR/4;
.   .   .   .   .   IF ((OWNTENT(7) EQ $FALSE AND ITF.RZ LE -THRES_RZ)
.   .   .   .   .   .   OR (OWNTENT(7) EQ $TRUE AND ITF.RZ GE THRES_RZ))
.   .   .   .   .   THEN   SET ITF.REVERSE;
.   .   .   .   .   .   .   CLEAR G.OWN_FOLLOW;
.   .   .   .   .   .   .   PERFORM Reversal_modeling;

.   ZI25 = ITF.ZINT + 2.5 * ITF.ZDINT;
.   ZO25 = G.ZOWN + 2.5 * G.ZDOWN;
.   IF (ITF.REVERSE EQ $TRUE
.   .   AND (ITF.ADOT LT -P.OLEV OR ABS(ITF.RZ) GT P.CROSSTHR/4)
.   .   AND ITF.TRTRU LE 1.25 * P.MINRVSTIME
.   .   AND ((OWNTENT(7) EQ $FALSE AND ITF.RZ LT 0 AND ZO25 GT ZI25)
.   .   .   OR (OWNTENT(7) EQ $TRUE AND ITF.RZ GT 0 AND ZO25 LT ZI25)))
.   THEN   CLEAR ITF.REVERSE;

END Take_decision;
```

NEW VARIABLES

STRUCTURE RESVAR
<***RESOLUTION LOCAL VARIABLES***>

GROUP RA_monitoring	<Variables needed for RA_monitoring only>
FLT THRES_RZ	<Altitude threshold>
FLT T_RZ	<Time threshold for reversal RA leading to altitude crossing>
FLT DELTA_Z_CPA	<CPA>
FLT DELTA_Z_MIN	<Minimum CPA>
FLT DELTA_Z_MAX	<Maximum CPA>
FLT DELTA_T_RZ	<Delta time threshold to add to t_rz>
FLT FACT_MULT	<Multiplying factor>
FLT ZI_IN	<Intruder's altitude computed with ITF.ZDINR>
FLT ZI_OUT	<Intruder's altitude computed with ITF.ZDOUTR>
GROUP Take_decision	<Variables needed for Take_decision only>
FLT ZI25	<Projected altitude of intruder 2.5s after current time>
FLT ZO25	<Projected altitude of own 2.5s after current time>

<***GLOBAL VARIABLES***>
STRUCTURE G

GROUP reversal	
BIT OWN_FOLLOW	<Indication of own following RAs or not>
BIT REV_CONSDRD	<Indication of a vertical chase, low VMD geometry detected>
FLT TTOFOLLOW	<Time to follow RA>
FLT BOUNDED_TTF	<Bounded time to follow initial RA>

STRUCTURE ITF

GROUP reversal	
INT CPT_REV	<SA01 reversal flag>

Appendix A

The new variables introduced by CP112E should be added to the System Cross-Reference Table in Appendix A.

2. State Charts Changes

3.70 Macro: Reversal_Conditions

Definition:

				OR			
	Range_Test$_{s-207}$ **in state** Passed	T	T	T	T	T	T
	Other_Capability$_{v-162}$ = TA/RA	T	T	F	T	T	.
	Own_Mode_S_Address$_{v-48}$ > Other_Mode_S_Address$_{v-158}$	T	T	·	F	F	.
	Intent_Received$_{s-309}$ **in state** Yes	T	T	·	.	·	·
	Sense$_{s-219}$ **in state** Climb	T	·	·	.	·	·
AND	Sense$_{s-219}$ **in state** Descend	·	T	·	.	·	·
	Other_VRC$_{v-159}$ = Dont_Climb	T	·	·	.	·	·
	Other_VRC$_{v-159}$ = Dont_Descend	·	T	·	.	·	·
	Reversal_Geometry$_{m-420}$	·	·	T	T	T	F
	Reversal_Inhibit$_{s-209}$ **in state** No	·	·	T	T	T	T
	PREV(Consider_Reversal$_{s-228}$) **in one of** {One_Hit, Previous_Hit}	.	.	.	T	.	.
	Crossing$_{s-210}$ **in state** Non_Crossing	.	.	.	.	T	.
	CP112E_Reversal_Geometry$_{m-428}$	.	.	.	.	.	T

Notes: ***1. Description:*** *This macro tests conditions to determine whether reversal of the current resolution advisory sense will be effected.*

2. Pseudocode Reference: *Reversal_check, Reversal_proj_check.*

3.72 Macro: CP112E_Reversal_Geometry

Definition:

				OR	
	CP112E_COMMON_CONDITIONS_PASSED	T	T	T
	$Other\_Capability_{v\text{-}162}$ = TA/RA	F	T	T
	$Own\_Mode\_S\_Address_{v\text{-}48} > Other\_Mode\_S\_Address_{v\text{-}158}$	.	F	F
	$CP112E\_VMD\_Reversal_{m\text{-}422\text{-}10}$	T	T	.
	$CP112E\_VMD\_2\_Out\_3_{m\text{-}422\text{-}9}$	T	T	.
AND	CP112E_ENOUGH_TIME_AFTER_INCREASE_RA	T	T	T
	$CP112E\_Own\_Not\_Follow\_Reversal_{m\text{-}422\text{-}7}$	.	.	T
	CP112E_VMD_NOMINAL_SEPARATION < ALIM	T	T	.
	CP112E_VMD_REVERSAL_SEPARATION > 0	T	T	.
	$CP112E\_ONF\_NOMINAL\_SEPARATION \leq 1.2 \bullet 100\ ft_{(CROSSTHR)}$	.	.	T
	$CP112E\_Cancel\_Reversal_{m\text{-}422\text{-}6}$	F	F	F

Abbreviations:

CP112E_COMMON_CONDITIONS_PASSED

	$Multiple\_Threats_{m\text{-}400}$	F
	$True\_Tau\_Capped_{f\text{-}568} > 4.0\ s_{(MINRITIME)}$	T
	$(t - Time\_Last\_New\_RA_{f\text{-}563}) \geq 10.0\ s$	T
AND	$Own\_Tracked\_Alt\_Rate_{f\text{-}549} \bullet Other\_Tracked\_Alt\_Rate_{f\text{-}536} > 0$	T
	$\lvert Own\_Tracked\_Alt\_Rate_{f\text{-}549} \rvert > 1{,}000\ ft/min_{(ILEV)}$	T
	$\lvert Other\_Tracked\_Alt\_Rate_{f\text{-}536} \rvert > 1{,}000\ ft/min_{(ILEV)}$	T
	$(t - Time\_Last\_New\_RA_{f\text{-}563}) \geq$ CP112E_TIME_TO_FOLLOW_RA_BOUNDED	T
	CP112E_ENOUGH_TIME_TO_LEVEL_OFF	T

CP112E_TIME_TO_FOLLOW_RA_BOUNDED =

Min(**Max**(10, CP112E_TIME_TO_FOLLOW_INITIAL_RA), 15)

CP112E_TIME_TO_FOLLOW_INITIAL_RA =

0 s	**if** $t < Time\_Last\_New\_RA_{f\text{-}563} + 5\ s_{(TV1)}$
$\lvert Own\_Tracked\_Alt\_Rate_{f\text{-}549} - Displayed\_Model\_Goal_{f\text{-}461} \rvert / 8.2\ ft/s^2_{(VACCEL)} + 5\ s_{(TV1)}$	**if** $t \geq Time\_Last\_New\_RA_{f\text{-}563} + 5\ s_{(TV1)}$ **and** PREV(CP112E_TIME_TO_FOLLOW_INITIAL_RA) = 0 s
PREV(CP112E_TIME_TO_FOLLOW_INITIAL_RA)	**Otherwise**

CP112E_ENOUGH_TIME_TO_LEVEL_OFF

			OR	
	$True\_Tau\_Capped_{f\text{-}568} > CP112E\_TIME\_TO\_LEVEL\_OFF$	T	.	.
AND	$(t - Time\_Advisory\_Changed_{f\text{-}562}) > 5\ s_{(TV1)}$	.	T	.
	$Any\_Increase_{m\text{-}366}$	.	.	F

CP112E_TIME_TO_LEVEL_OFF =

Max($\lvert Own\_Tracked\_Alt\_Rate_{f\text{-}549} \rvert / 11.2\ ft/s^2_{(RACCEL)}$, $\lvert Other\_Tracked\_Alt\_Rate_{f\text{-}536} \rvert / 11.2\ ft/s^2_{(RACCEL)}$) + 2.5 $s_{(QUIKREAC)}$	**if** $Other\_Capability_{v\text{-}162}$ = TA/RA
$\lvert Own\_Tracked\_Alt\_Rate_{f\text{-}549} \rvert / 11.2\ ft/s^2_{(RACCEL)} + 2.5\ s_{(QUIKREAC)}$	**Otherwise**

CP112E_ENOUGH_TIME_AFTER_INCREASE_RA

		OR	
	$Any\_Increase_{m-366}$	F	.
AND	$t - Time\_Advisory\_Changed_{f-562} > 1$ s	.	T

CP112E_VMD_NOMINAL_SEPARATION =

$Separation\_At\_Current\_RA_{f-555}$	**if** $Strength_{s-230}$ **in state** Positive
0	**Otherwise**

ALIM =

$Positive\_RA\_Altitude\_Limit\_Threshold_{t-673}[Alt\_Layer\_Value_{f-451}]$

CP112E_VMD_REVERSAL_SEPARATION =

$Separation_{f-552}$(CP112E_REVERSAL_ZDGOAL, CP112E_REVERSAL_DELAY, 11.2 $ft/s^2{}_{(RACCEL)}$, CP112E_REVERSAL_DIRECTION, $Modified\_Own\_Tracked\_Altitude_{f-526}$, $Modified\_Own\_Tracked\_Altitude\_Rate_{f-527}$, $Other\_Tracked\_Alt_{f-535}$, $Other\_Tracked\_Alt\_Rate_{f-536}$)

CP112E_REVERSAL_ZDGOAL =

Max(**Min**($Own\_Tracked\_Alt\_Rate_{f-549}$, 4,400ft/$min_{(MAXDRATE)}$), 1,500 ft/$min_{(CLMRT)}$)	**if** CP112E_REVERSAL_DIRECTION = Up
Min(**Max**($Own\_Tracked\_Alt\_Rate_{f-549}$, – 4,400ft/$min_{(MINDRATE)}$), –1,500 ft/$min_{(DESRT)}$)	**if** CP112E_REVERSAL_DIRECTION = Down **and** $Descend\_Inhibit_{s-80}$ **in state** Not_Inhibited
0 ft/min	**Otherwise**

CP112E_REVERSAL_DELAY =

Max(5 $s_{(TV1)}$ – (t – PREV($Time\_Last\_New\_RA_{f-563}$)), 2.5 $s_{(QUIKREAC)}$)

CP112E_REVERSAL_DIRECTION =

Down	**if** $Sense_{s\text{-}219}$ **in state** Climb
Up	**Otherwise**

CP112E_ONF_NOMINAL_SEP =

$Separation_{f\text{-}552}$(Own_Tracked_Alt_Rate $_{f\text{-}549}$, 0, 8 $ft/s^2{}_{(VACCEL)}$, CP112E_CURRENT_DIRECTION, $Own\_Tracked\_Alt_{f\text{-}548}$, $Own\_Tracked\_Alt\_Rate_{f\text{-}549}$, $Other\_Tracked\_Alt_{f\text{-}535}$, $Other\_Tracked\_Alt\_Rate_{f\text{-}536}$)

CP112E_CURRENT_DIRECTION =

Up	**if** $Sense_{s\text{-}219}$ **in state** Climb
Down	**Otherwise**

Notes: 1. ***Description:*** *This macro describes the two new geometric tests and new reversal validation criteria introduced by Change Proposal 112E to correct some deficiencies in* $Reversal\_Geometry_{m\text{-}420}$*. The VMD and Own-Not-Follow tests are described in* $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$ *and* $CP112E\_Own\_Not\_Follow\_Reversal_{m\text{-}422\text{-}6}$*,respectively.*

Both tests must satisfy the pre-conditions that (1) there is only one threat, (2) CPA is not imminent, (3) enough time has elapsed to allow pilot to react the initial RA, (4) aircraft will be able to level off before CPA, and (5) they are in a high rate vertical chase, i.e., flying in the same vertical direction at a rate greater than 1000 fpm (see CP112E_COMMON_CONDITIONS_PASSED). In addition, the tests must ensure (1) reversal is not issued in the cycle immediately after an increase RA (see CP112E_ENOUGH_TIME_AFTER_INCREASE_RA)) and (2) the aircraft pair is not projected to cross in altitude when CPA is near (see $CP112E\_Cancel\_Reversal_{m\text{-}422\text{-}5}$*).*

VMD reversals are subject to a redundancy check, requiring such reversals be recommended twice in the most recent three processing cycles (see $CP112E\_VMD\_2\_Out\_3_{m\text{-}422\text{-}8}$*). Own-Not-Follow reversals are not subject to this check. They, however, do not apply if a threat aircraft is not equipped with TCAS. TCAS is ineffective if the only equipped aircraft is not obeying the TCAS instructions.*

The validation of a VMD reversal is similar to that in Reversal_Geometry$_{m\text{-}422}$ except a threat aircraft's tracked altitude rate instead of its inner or outer bound is used during the modeling of the projected separation at CPA if sense reversal were to take place. The validation of an Own-Not-Follow reversal will assume own aircraft will continue in its current vertical rate until CPA. An Own-Not-Follow reversal is validated if aircraft will be separated by no more than 120 feet in positive vertical separation at CPA.

2. ***Pseudocode Reference:*** *New_threat_file_entry, Process_new_or_continuing_threat, Reversal_check, RA_monitoring, Take_decision, Reversal_modeling.*

3.73 Macro: CP112E_Cancel_Reversal

Definition:

			OR
	ADOT < -600 ft/min$_{(OLEV)}$	T	.
AND	|Other_Tracked_Relative_Alt$_{f-541}$| > 0.25 •100 ft$_{(CROSSTHR)}$	.	T
	True_Tau_Capped$_{f-568}$ ≤ 1.25 •10 s$_{(MINRVSTIME)}$	T	T
	CP112E_ALT_CROSSING_IN_QUICKREAC	T	T

Abbreviations:

ADOT =

Other_Tracked_Relative_Alt_Rate$_{f-542}$. **Sign**(Other_Tracked_Relative_Alt$_{f-541}$)

CP112E_ALT_CROSSING_IN_QUICKREAC

			OR
	Sense$_{s-219}$ **in state** Climb	T	.
	Other_Tracked_Relative_Alt$_{f-541}$ < 0	T	.
AND	CP112E_ZO_25 > CP112E_ZI_25	T	.
	Sense$_{s-219}$ **in state** Descend	.	T
	Other_Tracked_Relative_Alt$_{f-541}$ > 0	.	T
	CP112E_ZO_25 < CP112E_ZI_25	.	T

CP112E_ZO_25 =

Own_Tracked_Alt$_{f-548}$ + Own_Tracked_Alt_Rate$_{f-549}$ • 2.5 s$_{(QUIKREAC)}$

CP112E_ZI_25 =

Other_Tracked_Alt$_{f-535}$ + Other_Tracked_Alt_Rate$_{f-536}$ • 2.5 s$_{(QUIKREAC)}$

Notes: *1.* ***Description:*** *This macro determines if a VMD reversal or Own-Not-Following reversal should be canceled due to projected altitude crossing near the closest point of approach.*

2. ***Pseudocode Reference:*** *Take_decision.*

3.74 Macro: CP112E_Own_Not_Follow_Reversal

Definition:

				OR			
	$Sense_{s\text{-}219}$ **in state** Climb	T	T	.	.
	CP112E_PROJECTED_SEPARATION_AT_CPA < 0.5 • ALIM	T	T	.	.
	$Own\_Tracked\_Alt\_Rate_{f\text{-}549} < -1.2 \bullet 1000$ ft/$min_{(ILEV)}$	T	T	.	.
	$Other\_Tracked\_Relative\_Alt_{f\text{-}541} \leq$ –CP112E_ONF_ALT_CROSSING_THRESHOLD	T	T	.	.
	$Sense_{s\text{-}219}$ **in state** Descend	.	.	T	T
	CP112E_PROJECTED_SEPARATION_AT_CPA > –0.5 • ALIM	.	.	T	T
AND	$Own\_Tracked\_Alt\_Rate_{f\text{-}549} > 1.2 \bullet 1000$ ft/$min_{(ILEV)}$	.	.	T	T
	$Other\_Tracked\_Relative\_Alt_{f\text{-}541} \geq$ CP112E_ONF_ALT_CROSSING_THRESHOLD	.	.	T	T
	$Strength_{s\text{-}230}$ **in state** Positive	T	.	T	.
	CP112E_ENOUGH_TIME_AFTER_POSITIVE_RA	T	.	T	.
	$Strength_{s\text{-}230}$ **in state** VSL_0fpm	.	T	.	T
	$\lvert Own\_Tracked\_Alt\_Rate_{f\text{-}549} \rvert > 1.5 \bullet 1000$ ft/$min_{(ILEV)}$	.	T	.	T
	CP112E_VSL_RA	F	F	F	F

Abbreviations:

CP112E_VSL_RA

			OR	
	$Strength_{s\text{-}230}$ **in state** VSL_500fpm	T	.	.
AND	$Strength_{s\text{-}230}$ **in state** VSL_1000fpm	.	T	.
	$Strength_{s\text{-}230}$ **in state** VSL_2000fpm	.	.	T

CP112E_PROJECTED_SEPARATION_AT_CPA =

CP112E_OWN_PROJ_ALT_AT_CPA – CP112E_THREAT_PROJ_ALT_AT_CPA

CP112E_ENOUGH_TIME_AFTER_POSITIVE_RA

		OR	
	$Time\_Positive\_RA_{f\text{-}564} = Time\_Last\_New\_RA_{f\text{-}563}$	T	.
AND	$t - Time\_Positive\_RA_{f\text{-}564} > 5\ s_{(TV1)}$	.	T

CP112E_ONF_ALT_CROSSING_THRESHOLD =

CP112E_VMD_ALT_CROSSING_ THRESHOLD + 0.5 • 100 $ft_{(CROSSTHR)}$	**if** $True\_Tau\_Capped_{f\text{-}568} > 10\ s_{(MINRVSTIME)}$ **and** $Other\_Track\_Firmness_{f\text{-}534} < 2_{(MINFIRM)}$
CP112E_VMD_ALT_CROSSING_ THRESHOLD + 0.25 • 100 $ft_{(CROSSTHR)}$	**if** $True\_Tau\_Capped_{f\text{-}568} > 10\ s_{(MINRVSTIME)}$ **and** $Other\_Track\_Firmness_{f\text{-}534} = 2_{(MINFIRM)}$
CP112E_VMD_ALT_CROSSING_ THRESHOLD	**Otherwise**

Notes: **1.** ***Description:*** *This macro contains Change Proposal(CP) 112E's Own-Not-Follow test and determines if own aircraft is complying with its initial RA. This test is similar to the VMD test (see* $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$*) but relaxes some of the test criteria and emphasizes the deviation between own aircraft's current altitude rate and its RA goal altitude rate. If the deviation exceeds a certain threshold, this test will declare own aircraft non-compliant. This test will fail if the initial RA is a Vertical Speed Limit (VSL) RA because such RAs do not have a well defined goal altitude rate. The test ensures that extra time has been given to the pilot to react to an RA that has recently been strengthed to a positive RA.*

2. ***Pseudocode Reference:*** *RA_monitoring, Take_decision.*

3.75 Macro: CP112E_VMD_2_Out_3

Definition:

			OR	
	CP112E_VMD_ COUNTER = 3	T	.	.
AND	CP112E_VMD_ COUNTER = 5	.	T	.
	CP112E_VMD_ COUNTER = 7	.	.	T

Abbreviations:

CP112E_VMD_ COUNTER =

1	**if** $Status_{s\text{-}256}$ **in state** New **and** CP112E_ COMMON_CONDITIONS_PASSED = True **and** $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$ = True
0	**if** $Status_{s\text{-}256}$ **in state** New **and** (CP112E_ COMMON_CONDITIONS_PASSED = False **or** $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$ = False)
2 • (PREV(CP112E_VMD_ COUNTER) – 4) + 1	**if** $Status_{s\text{-}256}$ **in state** Continuing **and** PREV(CP112E_VMD_COUNTER) > 3 **and** CP112E_ COMMON_CONDITIONS_PASSED = True **and** $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$ = True
2 • PREV(CP112E_VMD_COUNTER) + 1	**if** $Status_{s\text{-}256}$ **in state** Continuing **and** PREV(CP112E_VMD_COUNTER) ≤ 3 **and** CP112E_ COMMON_CONDITIONS_PASSED = True **and** $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$ = True
2 • (PREV(CP112E_VMD_COUNTER) – 4)	**if** $Status_{s\text{-}256}$ **in state** Continuing **and** PREV(CP112E_VMD_COUNTER) > 3 **and** (CP112E_ COMMON_CONDITIONS_PASSED = False **or** $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$ = False)
2 • PREV(CP112E_VMD_COUNTER)	**if** $Status_{s\text{-}256}$ **in state** Continuing **and** PREV(CP112E_VMD_COUNTER) ≤ 3 **and** (CP112E_ COMMON_CONDITIONS_PASSED = False **or** $CP112E\_VMD\_Reversal_{m\text{-}422\text{-}9}$ = False)

Notes: **1.** ***Description:*** *This macro determines if a VMD reversal test has been passed twice in the most recent three processing cycles.*

2. ***Pseudocode Reference:*** *Take_decision.*

3.76 Macro: CP112E_VMD_Reversal

Definition:

				OR	
	$Sense_{s\text{-}219}$ **in state** Climb	T	T	.	.
	CP112E_MIN_PROJECTED_SEPARATION_AT_CPA < 1.2 • 100 $ft_{(CROSSTHR)}$	T	T	.	.
	$Other\_Tracked\_Relative\_Alt_{f\text{-}541}$ ≤ –CP112E_VMD_ALT_CROSSING_THRESHOLD	T	T	.	.
	$Sense_{s\text{-}219}$ **in state** Descend	.	.	T	T
AND	CP112E_MAX_PROJECTED_SEPARATION_AT_CPA > –1.2 • 100 $ft_{(CROSSTHR)}$	.	.	T	T
	$Other\_Tracked\_Relative\_Alt_{f\text{-}541}$ ≥ CP112E_VMD_ALT_CROSSING_THRESHOLD	.	.	T	T
	$Other\_Capability_{v\text{-}162}$ = TA/RA	T	.	T	.
	$Own\_Tracked\_Alt\_Rate_{f\text{-}549}$ · $Displayed\_Model\_Goal_{f\text{-}461}$ > 0	.	T	.	T
	|$Other\_Tracked\_Relative\_Alt_{f\text{-}541}$| < 600 $ft_{(MAXALTDIFF)}$	T	T	T	T

Abbreviations:

CP112E_MIN_PROJECTED_SEPARATION_AT_CPA =

Min(
CP112E_OWN_PROJ_ALT_AT_CPA – CP112E_THREAT_PROJ_ALT_AT_CPA_OUTER,
CP112E_OWN_PROJ_ALT_AT_CPA – CP112E_THREAT_PROJ_ALT_AT_CPA,
CP112E_OWN_PROJ_ALT_AT_CPA – CP112E_THREAT_PROJ_ALT_AT_CPA_INNER)

CP112E_OWN_PROJECTED_ALT_AT_CPA =

$Own\_Tracked\_Alt_{f\text{-}548}$ + $Own\_Tracked\_Alt\_Rate_{f\text{-}549}$ • **Min**($XTVPETBLX_{t\text{-}680}$[$Conflict\_SL_{f\text{-}454}$], $True\_Tau\_Capped_{f\text{-}568}$)

CP112E_THREAT_PROJ_ALT_AT_CPA_OUTER =

$Other\_Tracked\_Alt_{f\text{-}535}$ + $Other\_Tracked\_Alt\_Rate\_Outer_{f\text{-}538}$ • **Min**($XTVPETBLX_{t\text{-}680}$[$Conflict\_SL_{f\text{-}454}$], $True\_Tau\_Capped_{f\text{-}568}$)

CP112E_THREAT_PROJ_ALT_AT_CPA =

$Other\_Tracked\_Alt_{f\text{-}535} + Other\_Tracked\_Alt\_Rate_{f\text{-}536} \bullet \textbf{Min}(XTVPETBLX_{t\text{-}680}[Conflict\_SL_{f\text{-}454}], True\_Tau\_Capped_{f\text{-}568})$

CP112E_THREAT_PROJ_ALT_AT_CPA_INNER =

$Other\_Tracked\_Alt_{f\text{-}535} + Other\_Tracked\_Alt\_Rate\_Inner_{f\text{-}537} \bullet \textbf{Min}(XTVPETBLX_{t\text{-}680}[Conflict\_SL_{f\text{-}454}], True\_Tau\_Capped_{f\text{-}568})$

CP112E_MAX_PROJECTED_SEPARATION_AT_CPA =

Max(
CP112E_OWN_PROJ_ALT_AT_CPA – CP112E_THREAT_PROJ_ALT_AT_CPA_OUTER,
CP112E_OWN_PROJ_ALT_AT_CPA – CP112E_THREAT_PROJ_ALT_AT_CPA,
CP112E_OWN_PROJ_ALT_AT_CPA – CP112E_THREAT_PROJ_ALT_AT_CPA_INNER)

CP112E_VMD_ALT_CROSSING_THRESHOLD =

$-$CP112E_FACT_MULT_REDUCED • $100\ ft_{(CROSSTHR)}$ + CP112E_VMD_THRESHOLD_REDUCTION	**if** $True\_Tau\_Capped_{f\text{-}568} \geq$ CP112E_TIME_TO_LEVEL_OFF + CP112E_DELTA_TIME_TO_LEVEL_OFF **and** $Other\_Tracked\_Range\_Rate_{f\text{-}540} < -5 \bullet 10\ ft/s_{(RDTHR)}$ **and** $True\_Tau\_Capped_{f\text{-}568} > 10\ s_{(MINRVSTIME)}$
$-$CP112E_FACT_MULT_REDUCED • $0.25 \bullet 100\ ft_{(CROSSTHR)}$ + CP112E_VMD_THRESHOLD_REDUCTION	**if** $True\_Tau\_Capped_{f\text{-}568} \geq$ CP112E_TIME_TO_LEVEL_OFF + CP112E_DELTA_TIME_TO_LEVEL_OFF **and** $Other\_Tracked\_Range\_Rate_{f\text{-}540} < -5 \bullet 10\ ft/s_{(RDTHR)}$ **and** $True\_Tau\_Capped_{f\text{-}568} \leq 10\ s_{(MINRVSTIME)}$
CP112E_VMD_THRESHOLD_REDUCTION	**Otherwise**

CP112E_FACT_MULT_REDUCED =

CP112E_FACT_MULT / 4	**if** $(t - Time\_Advisory\_Changed_{f\text{-}562}) \leq 5\ s_{(TV1)}$ **and** $Any\_Increase_{m\text{-}366}$
CP112E_FACT_MULT	**Otherwise**

CP112E_FACT_MULT =

1.5	**if** Other_Capability$_{v\text{-}162}$ = TA/RA **and** |Own_Tracked_Alt_Rate$_{f\text{-}549}$| > 3.0 • 1,000 ft/min$_{(ILEV)}$ **and** |Other_Tracked_Alt_Rate$_{f\text{-}536}$| > 3.0 • 1,000 ft/min$_{(ILEV)}$
1.25	**if** Other_Capability$_{v\text{-}162}$ = TA/RA **and** |Own_Tracked_Alt_Rate$_{f\text{-}549}$| ≤ 3.0 • 1,000 ft/min$_{(ILEV)}$ **and** |Own_Tracked_Alt_Rate$_{f\text{-}549}$| > 2.0 • 1,000 ft/min$_{(ILEV)}$ **and** |Other_Tracked_Alt_Rate$_{f\text{-}536}$| ≤ 3.0 • 1,000 ft/min$_{(ILEV)}$ **and** |Other_Tracked_Alt_Rate$_{f\text{-}536}$| > 2.0 • 1,000 ft/min$_{(ILEV)}$
1.1	**if** Other_Capability$_{v\text{-}162}$ = TA/RA **and** |Own_Tracked_Alt_Rate$_{f\text{-}549}$| ≤ 2.0 • 1,000 ft/min$_{(ILEV)}$ **and** |Own_Tracked_Alt_Rate$_{f\text{-}549}$| > 1.5 • 1,000 ft/min$_{(ILEV)}$ **and** |Other_Tracked_Alt_Rate$_{f\text{-}536}$| ≤ 2.0 • 1,000 ft/min$_{(ILEV)}$ **and** |Other_Tracked_Alt_Rate$_{f\text{-}536}$| > 1.5 • 1,000 ft/min$_{(ILEV)}$
1.0	**Otherwise**

CP112E_VMD_THRESHOLD_REDUCTION =

100 ft$_{(CROSSTHR)}$	**if** True_Tau_Capped$_{f\text{-}568}$ ≤ 10 s$_{(MINRVSTIME)}$ **and** Other_Track_Firmness$_{f\text{-}534}$ < 2$_{(MINFIRM)}$
0.5 • 100 ft$_{(CROSSTHR)}$	**if** True_Tau_Capped$_{f\text{-}568}$ ≤ 10 s$_{(MINRVSTIME)}$ **and** Other_Track_Firmness$_{f\text{-}534}$ = 2$_{(MINFIRM)}$
0 ft	**Otherwise**

CP112E_DELTA_TIME_TO_LEVEL_OFF =

3.0 s	**if** Other_Capability$_{v\text{-}162}$ = TA/RA **and** |Own_Tracked_Alt_Rate$_{f\text{-}549}$| > 1.5 • 1,000 ft/min$_{(ILEV)}$ **and** |Other_Tracked_Alt_Rate$_{f\text{-}536}$| > 1.5 • 1,000 ft/min$_{(ILEV)}$
5.0 s	**Otherwise**

Notes: ***1. Description:*** *This macro contains Change Proposal 112E's Vertical Miss Distance (VMD) test. This test is based on the high rate vertical chase test in Reversal_Geometry$_{m\text{-}420}$, but relaxes the reversing criteria by enlarging the co-altitude band, allowing two aircraft in the same altitude band to reverse, and permitting reversal to an altitude-crossing sense against an equipped intruder.*

2. Pseudocode Reference: *RA_monitoring, Take_decision.*

3. Test Suite Changes

The outputs of three DO-185A test cases are modified by CP112E:

1. EN03TS25.DAT
2. EN05TS15.DAT
3. EN05TS16.DAT

Nine new test cases are added to the test suite to exercise the new logic introduced by CP112E, which are listed below:

1. EN05TS31.DAT
2. EN05TS32.DAT
3. EN05TS33.DAT
4. EN05TS34.DAT
5. EN05TS35.DAT
6. EN05TS36.DAT
7. EN05TS37.DAT
8. EN05TS38.DAT
9. EN05TS39.DAT.

*** END OF DOCUMENT ***