DOT/FAA/NR-92/3

Project Report
ATC-187

Terminal Doppler Weather Radar/ Low-Level Wind Shear Alert System Integration Algorithm Specification Version 1.1

R. E. Cole

24 February 1992

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

Prepared for the Federal Aviation Administration,
Washington, D.C. 20591

This document is available to the public through
the National Technical Information Service,
Springfield, VA 22161

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
ATC-187	DOT/FAA/NR-92/3	

4. Title and Subtitle	5. Report Date
Terminal Doppler Weather Radar/Low-Level Wind Shear Alert System Integration Algorithm Specification Version 1.1	24 February 1992
	6. Performing Organization Code
7. Author(s) Rodney E. Cole	8. Performing Organization Report No. ATC-187
9. Performing Organization Name and Address Lincoln Laboratory, MIT P.O. Box 73 Lexington, MA 02173-9108	10. Work Unit No. (TRAIS)
	11. Contract or Grant No. DTFA-01-89-Z-02033
12. Sponsoring Agency Name and Address Department of Transportation Federal Aviation Administration Systems Research and Development Service Washington, DC 20591	13. Type of Report and Period Covered Project Report
	14. Sponsoring Agency Code

15. Supplementary Notes

This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Department of the Air Force under Contract F19628-90-C-0002.

16. Abstract

There will be a number of airports that receive both a Terminal Doppler Weather Radar (TDWR) windshear detection system and a phase III Low-Level Wind Shear Alert System (LLWAS). At those airports, the two systems will need to be combined into a single windshear detection system. This report specifies the algorithm to be used to integrate the two subsystems. The algorithm takes in the alphanumeric runway alert messages generated by each subsystem and joins them into integrated alert messages.

The design goals of this windshear detection system are 1) to maintain the probability of detection for hazardous events while reducing the number of false alerts and microburst overwarnings and 2) to increase the accuracy of the loss/gain estimates. The first design goal is accomplished by issuing an integrated alert for an operational runway whenever either subsystem issues a "strong" alert for that runway; by canceling a "weak" windshear alert on an operational runway if only one subsystem is making the declaration; and by reducing a "weak" microburst alert on an operational runway to a "strong" windshear alert if only one subsystem is making the declaration. The second design goal is accomplished by using the average of the two loss/gain values, when appropriate.

17. Key Words		18. Distribution Statement
TDWR LLWAS	windshear algorithm specification	This document is available to the public through the National Technical Information Service, Springfield, VA 22161.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	36	

FORM DOT F 1700.7 (8-72) Reproduction of completed page authorized

ABSTRACT

There will be a number of airports that receive both a Terminal Doppler Weather Radar (TDWR) windshear detection system and a phase III Low Level Wind Shear Alert System (LLWAS). At those airports, the two systems will need to be combined into a single windshear detection system. This report specifies the algorithm to be used to integrate the two subsystems. The algorithm takes in the alphanumeric runway alert messages generated by each subsystem and joins them into integrated alert messages.

The design goals of this windshear detection system are 1) to maintain the probability of detection for hazardous events while reducing the number of false alerts and microburst overwarnings and 2) to increase the accuracy of the loss/gain estimates. The first design goal is accomplished by issuing an integrated alert for an operational runway whenever either subsystem issues a "strong" alert for that runway; by canceling a "weak" windshear alert on an operational runway if only one subsystem is making the declaration; and by reducing a "weak" microburst alert on an operational runway to a "strong" windshear alert if only one subsystem is making the declaration. The second design goal is accomplished by using the average of the two loss/gain values, when appropriate.

TABLE OF CONTENTS

1. OVERVIEW OF THE TDWR/LLWAS INTEGRATION ALGORITHM SPECIFICATION

This algorithm integrates the runway alerts generated by the TDWR system and Network Expansion LLWAS to give a single runway alert for each operational runway.

A. DESIGN GOALS

1. The primary goal of Integration is to issue all correct alerts from either subsystem. If only one of the systems detects an event, say TDWR misses a dry microburst or LLWAS misses a very small microburst, the alert generated by Integration should agree with the (stand–alone) system making the detection. The only time this should not be true is when the Integration algorithm decides that the alert should be altered in the interest of the second design goal.

2. Integration should use the available information to reduce false alerts, over–warning, and nuisance alerts. This is accomplished by dropping weak windshear alerts if there is no additional evidence that they are correct, by reducing weak microbursts to wind shear alerts if there is no additional evidence that they are correct, and by using both systems to determine the loss or gain value when both detect an event.

B. INPUTS

1. LLWAS runway alerts for each operational runway (loss/gain value and location).
2. TDWR runway alerts for each operational runway (loss/gain value and location).
3. Various parameters from the Integration Parameter File (IPF).

C. OUTPUTS

1. Runway alerts for each operational runway (loss/gain value and location).

The output loss/gain values from the Integration algorithm are in meters per second, and there is no alert type, i.e., an alert is not classified as either a MBA or WSA. The loss/gain values are to be converted into multiples of 5 knots and alert types attached after Integration.

D. ALGORITHM LOGIC

Alert Screening

The subsystem alerts are first screened before being joined using the following logic:

1. Strong microburst alerts pass forward.
2. Weak microburst level alerts are passed forward unchanged if there is a loss above threshold from the other system; otherwise, the alert is reduced to the maximum allowed windshear alert.

3. Strong windshear with loss alerts are passed forward.
4. Weak windshear with loss alerts are passed forward unchanged if there is a loss above threshold from the other system; otherwise, the alert is dropped.
5. Weak windshear with gain alerts are passed forward unchanged if there is a gain above threshold from the other system; otherwise, the alert is dropped.
6. Strong windshear with gain alerts are passed forward.

The thresholds above are set separately for each system being screened and for each operational runway. This allows for setting the difficulty of the screening tests to depend on the subsystem performance for each operational runway. For example, when TDWR has a favorable viewing angle for a runway, the thresholds are set low so that all but the weakest TDWR alerts automatically pass the screening tests. But when TDWR does not have a good viewing angle, the thresholds are raised so that fewer TDWR alerts automatically pass the screening tests.

In general, there are six thresholds used to screen an alert:

1. Thresh1 defines weak WSA/gain: Any gain alert between 0 and Thresh1 requires confirmation from the other system.
2. Thresh2 is the threshold for confirmation of a weak gain. The weak gain from 1. is confirmed if the other system issues an alert above Thresh2.
3. Thresh3 defines weak WSA/loss: Any loss alert between 0 and Thresh3 requires confirmation from the other system.
4. Thresh4 is the threshold for confirmation of a weak loss. The weak loss from 3. is confirmed if the other system issues an alert below (i.e., stronger loss than) Thresh4.
5. Thresh5 defines weak MBA: Any loss alert between the minimum MBA and Thresh5 requires confirmation from the other system.
6. Thresh6 is the threshold for confirmation of a weak MBA. The weak MBA from 5. is confirmed if the other system issues an alert below (i.e., stronger loss than) Thresh6.

Joining Alerts

The screened runway alerts from the two subsystems are joined as follows:

1. If only one system is giving an alert, it is used as the Integration alert.
2. If both systems are giving a loss, the location used is the first encounter from either system, and the loss value is the minimum of { the average of the two losses, αLLWAS loss, βTDWR loss } where α and β are between 0.0 and 1.0.

3. If both systems are giving a gain, the location used is the first encounter from either system, and the gain is the maximum of { the average of the two gains, γLLWAS gain, δTDWR gain } where γ and δ are between 0.0 and 1.0.
4. If one system is giving a loss and the other is giving a gain, arbitrate to determine which alert to issue. The arbitration logic is equivalent to the arbitration logic contained in the Phase III LLWAS algorithm.

The loss given in 2 is just the average loss unless the average is much lower than the larger of the LLWAS loss and the TDWR loss. This allows for a more accurate loss estimate and at the same time protects against dropping the loss estimate too far. The gain estimate given in 3 is similar to the loss estimate.

The arbitration logic uses the following priorities:

1. Microburst level loss
2. Strong wind shear with loss
3. Strong wind shear with gain
4. Weak wind shear with loss
5. Weak wind shear with gain

The terms "strong" and "weak" used here are not the same as in the alert screening. The parameter (LOSS_GAIN_BUFFER) used to determine the difference between strong and weak wind shear alerts can be set so that all losses are "strong."

Appendices

Appendix A contains the data dictionary for the TDWR/LLWAS integration algorithm. Appendix B specifies the inputs to TDWR/LLWAS integration from the LLWAS system.

Notes

1. Losses are negative, gains are positive, and both are in meters per second.
2. Null alerts must have loss/gain values of precisely zero.
3. It is assumed that integration alerts will be issued each time LLWAS alerts are issued. Each time the LLWAS system provides alerts to the integration system the current TDWR alerts need to be provided to the integration system.

Programming Notes

These comments are intended as helpful information, not requirements.

1. There are a number of variables which have not been specified, but may be helpful to have available. These include, but are not restricted to, the following:

   - The number of operational runways.
   - The names of the operational runways.

2. The various loops and conditional structures employed in this document were chosen for clarity of exposition, not efficiency of code. Any construction which has been shown to be logically equivalent may be used.

3. In general, subscripts are avoided. Loops are written in the form: DO FOR EACH operational runway. The method of implementation is not specified.

4. There are several subroutines employed in the algorithm. A list of inputs is provided for each. This is not meant to imply that only this information can be passed in the call. For example, when only one element of an array or data structure is required, only that element is shown being passed into the subroutine. It may prove easier to pass in the entire array or data structure. IPF parameters are not shown being passed in, but need to be made available.

5. Certain variables are listed as YES/NO variables. They are listed in the data dictionary as "logical." The formal declaration need not be logical; any binary variable may be used.

6. The notation $|x|$ means the absolute value of x.

2. TDWR/LLWAS INTEGRATION ALGORITHM SPECIFICATION

INPUTS: (from LLWAS, for each operational runway)

LLWAS_ALERT_VALUE (loss or gain)

LLWAS_ALERT_LOCATION (location of first encounter)

INPUTS: (from TDWR, for each operational runway)

TDWR_ALERT_VALUE (loss or gain)

TDWR_ALERT_LOCATION (location of first encounter)

INPUTS: (from IPF)

LLWAS_COVERAGE (for each operational runway)

LLWAS_THRESHi (i = 1, ... , 6, for each operational runway)

TDWR_COVERAGE (for each operational runway)

TDWR_THRESHi (i = 1, ... , 6, for each operational runway)

OUTPUTS: (for each operational runway)

INTEGRATION_ALERT_VALUE (loss or gain)

INTEGRATION_ALERT_LOCATION (location of first encounter).

BEGIN INTEGRATION ALGORITHM

1. SCREEN LLWAS ALERTS

```
CALL SCREEN ALERT SUBROUTINE(LLWAS_ALERT_VALUE,
                             LLWAS_ALERT_LOCATION,
                             TDWR_COVERAGE,
                             TDWR_ALERT_VALUE,
                             LLWAS_THRESHi, i = 1, ... ,6
                             LLWAS_SCREENED_ALERT_VALUE )
```

2. SCREEN TDWR ALERTS

```
CALL SCREEN ALERT SUBROUTINE(TDWR_ALERT_VALUE,
                             TDWR_ALERT_LOCATION,
                             LLWAS_COVERAGE,
                             LLWAS_ALERT_VALUE,
                             TDWR_THRESHi, i = 1, ... ,6
                             TDWR_SCREENED_ALERT_VALUE )
```

3. JOIN LLWAS AND TDWR ALERTS

```
CALL JOIN ALERTS SUBROUTINE  (LLWAS_SCREENED_ALERT_VALUE,
                              LLWAS_ALERT_LOCATION
                              TDWR_SCREENED_ALERT_VALUE,
                              TDWR_ALERT_LOCATION,
                              INTEGRATION_ALERT_VALUE,
                              INTEGRATION_ALERT_LOCATION )
```

END INTEGRATION ALGORITHM

ALERT ARBITRATION SUBROUTINE(**LAST_GAIN,**
LAST_LOSS,
GAIN_ALERT_VALUE,
GAIN_ALERT_LOCATION,
LOSS_ALERT_VALUE,
LOSS_ALERT_LOCATION,
ALERT_VALUE,
ALERT_LOCATION)

This subroutine takes the loss and gain alerts for a single operational runway and arbitrates between the two to give a single alert for that runway.

INPUTS:

LAST_GAIN	(YES/NO, for this operational runway)
LAST_LOSS	(YES/NO, for this operational runway)
GAIN__ALERT_VALUE	(for this operational runway)
GAIN_ALERT_LOCATION	(for this operational runway)
LOSS_ALERT_VALUE	(for this operational runway)
LOSS_ALERT_LOCATION	(for this operational runway)

OTHER INPUTS: (from IPF)

LOSS_GAIN_BUFFER

LOSS_INCREMENT

MIN_MBA

OUTPUTS:

ALERT_VALUE	(for this operational runway)
ALERT_LOCATION	(for this operational runway)

```
BEGIN ARBITRATION SUBROUTINE
LOSS_ALERT = NO
comment: If the loss is above the microburst level the alert is a loss alert.
IF ( | LOSS_ALERT_VALUE | ≥ MIN_MBA )
        LOSS_ALERT = YES
comment: If not a microburst alert give the gain only if it is much stronger than the loss.
ELSE
        IF ( LAST_GAIN = YES )
                IF ( | LOSS_ALERT_VALUE | + LOSS_INCREMENT
                        - LOSS_GAIN_BUFFER > GAIN_ALERT_VALUE )
                        LOSS_ALERT = YES
                END IF
        ELSE IF ( LAST_LOSS = YES )
                IF ( | LOSS_ALERT_VALUE | + LOSS_INCREMENT
                        + LOSS_GAIN_BUFFER > GAIN_ALERT_VALUE )
                        LOSS_ALERT = YES
                END IF
        ELSE
                IF ( | LOSS_ALERT_VALUE | + LOSS_INCREMENT
                        > GAIN_ALERT_VALUE )
                        LOSS_ALERT = YES
                END IF
        END IF
END IF
```

comment: Now compute the alert to issue.

IF (LOSS_ALERT = YES)

ALERT_VALUE = LOSS_ALERT_VALUE

ALERT_LOCATION = LOSS_ALERT_LOCATION

ELSE

comment: If not a loss it must be a gain.

ALERT_VALUE = GAIN_ALERT_VALUE

ALERT_LOCATION = GAIN_ALERT_LOCATION

END IF

END ARBITRATION SUBROUTINE

JOIN ALERTS SUBROUTINE (LLWAS_ALERT_VALUE,
LLWAS_ALERT_LOCATION,
TDWR_ALERT_VALUE,
TDWR_ALERT_LOCATION,
INTEGRATION_ALERT_VALUE,
INTEGRATION_ALERT_LOCATION)

This subroutine joins the two alerts for the entire set of operational runways.

INPUTS:

LLWAS_ALERT_VALUE	(for each operational runway)
LLWAS_ALERT_LOCATION	(for each operational runway)
TDWR_ALERT_VALUE	(for each operational runway)
TDWR_ALERT_LOCATION	(for each operational runway)

INPUTS: (from IPF)

LLWAS_COVERAGE	(for each operational runway)
LLWAS_GF_FACTOR	(for each operational runway)
LLWAS_MB_FACTOR	(for each operational runway)
TDWR_COVERAGE	(for each operational runway)
TDWR_GF_FACTOR	(for each operational runway)
TDWR_MB_FACTOR	(for each operational runway)

OUTPUTS: (for each operational runway)

INTEGRATION_ALERT_VALUE

INTEGRATION_ALERT_LOCATION

OTHER VARIABLES:

LAST_GAIN	(for each operational runway, must be saved from one alert cycle to the next)
LAST_LOSS	(for each operational runway, must be saved from one alert cycle to the next)

BEGIN JOIN ALERTS SUBROUTINE:

comment: Ordering of alert locations: RWY < 1MF < 2MF < 3MF, and RWY < 1MD < 2MD

DO FOR EACH operational runway

comment: If no alert.

IF (LLWAS_ALERT_VALUE = 0 and TDWR_ALERT_VALUE = 0)

INTEGRATION_ALERT_VALUE = 0

INTEGRATION_ALERT_LOCATION = 'RWY'

comment: If only an LLWAS alert.

ELSE IF (|LLWAS_ALERT_VALUE| > 0 and TDWR_ALERT_VALUE = 0)

INTEGRATION_ALERT_VALUE = LLWAS_ALERT_VALUE

INTEGRATION_ALERT_LOCATION = LLWAS_ALERT_LOCATION

comment: If only a TDWR alert.

ELSE IF (LLWAS_ALERT_VALUE = 0 and |TDWR_ALERT_VALUE| > 0)

INTEGRATION_ALERT_VALUE = TDWR_ALERT_VALUE

INTEGRATION_ALERT_LOCATION = TDWR_ALERT_LOCATION *comment: If both LLWAS and TDWR are issuing a loss alert.*

```
ELSE IF ( LLWAS_ALERT_VALUE < 0 and TDWR_ALERT_VALUE < 0 )

        CALL JOIN LOSS ALERTS SUBROUTINE( LLWAS_ALERT_VALUE,
                                          LLWAS_ALERT_LOCATION,
                                          TDWR_ALERT_VALUE,
                                          TDWR_ALERT_LOCATION,
                                          INTEGRATION_ALERT_VALUE )

        IF  arrival runway

                INTEGRATION_ALERT_LOCATION = maximum
                        {LLWAS_ALERT_LOCATION, TDWR_ALERT_LOCATION}

        ELSE

                INTEGRATION_ALERT_LOCATION = minimum
                        {LLWAS_ALERT_LOCATION, TDWR_ALERT_LOCATION}

        END IF
```

comment: If both LLWAS and TDWR are issuing a gain alert.

```
ELSE IF ( LLWAS_ALERT_VALUE > 0 and TDWR_ALERT_VALUE > 0 )

    CALL JOIN GAIN ALERTS SUBROUTINE( LLWAS_ALERT_VALUE,
                                      LLWAS_ALERT_LOCATION,
                                      TDWR_ALERT_VALUE,
                                      TDWR_ALERT_LOCATION,
                                      INTEGRATION_ALERT_VALUE )

    IF  arrival runway

        INTEGRATION_ALERT_LOCATION = maximum
                {LLWAS_ALERT_LOCATION, TDWR_ALERT_LOCATION}

    ELSE

        INTEGRATION_ALERT_LOCATION = minimum
                {LLWAS_ALERT_LOCATION, TDWR_ALERT_LOCATION}

    END IF
```

comment: If both a loss and a gain arbitrate.

```
ELSE IF ( LLWAS_ALERT_VALUE < 0 and TDWR_ALERT_VALUE > 0 )

        CALL ALERT ARBITRATION SUBROUTINE
                        ( LAST_GAIN,
                          LAST_LOSS,
                          TDWR_ALERT_VALUE,
                          TDWR_ALERT_LOCATION,
                          LLWAS_ALERT_VALUE,
                          LLWAS_ALERT_LOCATION,
                          INTEGRATION_ALERT_VALUE,
                          INTEGRATION_ALERT_LOCATION )

ELSE IF ( LLWAS_ALERT_VALUE > 0 and TDWR_ALERT_VALUE < 0 )

        CALL ALERT ARBITRATION SUBROUTINE
                        ( LAST_GAIN,
                          LAST_LOSS,
                          LLWAS_ALERT_VALUE,
                          LLWAS_ALERT_LOCATION,
                          TDWR_ALERT_VALUE,
                          TDWR_ALERT_LOCATION,
                          INTEGRATION_ALERT_VALUE,
                          INTEGRATION_ALERT_LOCATION )

END IF
```

comment: Now compute last_loss and last_gain.

```
    IF      ( INTEGRATION_ALERT_VALUE < 0 )
            LAST_LOSS = YES
            LAST_GAIN = NO
    ELSE IF ( INTEGRATION_ALERT_VALUE > 0 )
            LAST_LOSS = NO
            LAST_GAIN = YES
    ELSE
            LAST_LOSS = NO
            LAST_GAIN = NO
    END IF
END DO
END JOIN ALERTS SUBROUTINE
```

JOIN GAIN ALERTS SUBROUTINE(LLWAS_ALERT_VALUE,
LLWAS_ALERT_LOCATION,
TDWR_ALERT_VALUE,
TDWR_ALERT_LOCATION,
INTEGRATION_ALERT_VALUE)

This subroutine computes the integration alert value for a single operational runway in the case that both TDWR and LLWAS are issuing a gain.

INPUTS:

LLWAS_ALERT_VALUE	(for this operational runway)
LLWAS_ALERT_LOCATION	(for this operational runway)
TDWR_ALERT_VALUE	(for this operational runway)
TDWR_ALERT_LOCATION	(for this operational runway)

INPUTS: (from IPF)

LLWAS_COVERAGE	(for this operational runway)
LLWAS_GF_FACTOR	(for this operational runway)
TDWR_COVERAGE	(for this operational runway)
TDWR_GF_FACTOR	(for this operational runway)

OUTPUTS:

INTEGRATION_ALERT_VALUE (for this operational runway)

BEGIN JOIN GAIN ALERTS SUBROUTINE

```
IF ( ( LLWAS_ALERT_LOCATION ≤ TDWR_COVERAGE ) and
     ( TDWR_ALERT_LOCATION ≤ LLWAS_COVERAGE ) )

     AVE = ( LLWAS_ALERT_VALUE + TDWR_ALERT_VALUE ) / 2

     INTEGRATION_ALERT_VALUE
           = maximum { AVE,
                       LLWAS_GF_FACTOR x LLWAS_ALERT_VALUE,
                       TDWR_GF_FACTOR x TDWR_ALERT_VALUE }

ELSE

     INTEGRATION_ALERT_VALUE = maximum{ LLWAS_ALERT_VALUE,
                                        TDWR_ALERT_VALUE }

END IF
```

END JOIN GAIN ALERTS SUBROUTINE

JOIN LOSS ALERTS SUBROUTINE(LLWAS_ALERT_VALUE,
LLWAS_ALERT_LOCATION,
TDWR_ALERT_VALUE,
TDWR_ALERT_LOCATION,
INTEGRATION_ALERT_VALUE)

This subroutine computes the integration alert value for a single operational runway in the case that both TDWR and LLWAS are issuing a loss.

LLWAS_ALERT_VALUE (for this operational runway)

LLWAS_ALERT_LOCATION (for this operational runway)

TDWR_ALERT_VALUE (for this operational runway)

TDWR_ALERT_LOCATION (for this operational runway)

INPUTS: (from IPF)

LLWAS_COVERAGE (for this operational runway)

LLWAS_MB_FACTOR (for this operational runway)

TDWR_COVERAGE (for this operational runway)

TDWR_MB_FACTOR (for this operational runway)

OUTPUTS:

INTEGRATION_ALERT_VALUE (for this operational runway)

```
BEGIN JOIN LOSS ALERTS SUBROUTINE

IF  (  ( LLWAS_ALERT_LOCATION ≤ TDWR_COVERAGE ) and
       ( TDWR_ALERT_LOCATION ≤ LLWAS_COVERAGE )  )

       AVE = ( LLWAS_ALERT_VALUE + TDWR_ALERT_VALUE ) / 2

       INTEGRATION_ALERT_VALUE =
                  minimum {  AVE,
                             LLWAS_MB_FACTOR x LLWAS_ALERT_VALUE,
                             TDWR_MB_FACTOR x TDWR_ALERT_VALUE }

ELSE

       INTEGRATION_ALERT_VALUE = minimum { LLWAS_ALERT_VALUE,
                                           TDWR_ALERT_VALUE }

END IF

END JOIN LOSS ALERT SUBROUTINE
```

SCREEN ALERT SUBROUTINE(ALERT_VALUE,
ALERT_LOCATION,
OTHER_ALERT_COVERAGE,
OTHER_ALERT_VALUE,
THRESHi, i = 1, ... , 6
SCREENED_ALERT_VALUE)

This subroutine takes all the runway alerts from one system and compares weak alerts to the runway alerts generated by the other system. Weak alerts which are not confirmed by the other system are dropped or reduced.

INPUTS:

ALERT_VALUE	(loss or gain, for each operational runway)
ALERT_LOCATION	(for each operational runway)
OTHER_ALERT_COVERAGE	(for each operational runway)
OTHER_ALERT_VALUE	(loss or gain, for each operational runway)
THRESHi	(i = 1, ... , 6, for each operational runway)

OTHER VARIABLES: (from IPF)

MAX_WSA

MIN_MBA

OUTPUTS:

SCREENED_ALERT_VALUE	(loss or gain, for each operational runway)

comment: Ordering of alert locations: RWY < 1MF < 2MF < 3MF, and RWY < 1MD < 2MD

```
BEGIN ALERT SCREENING SUBROUTINE

DO FOR EACH operational runway

comment: If the alert is in the coverage region of the other system and weak check for lack of
         confirmation.

    IF      ( ( ALERT_LOCATION ≤ OTHER_ALERT_COVERAGE ) and
            ( 0 ≤ ALERT_VALUE ≤ THRESH1 ) and
            ( OTHER_ALERT_VALUE ≤ THRESH2 ) )

            SCREENED_ALERT_VALUE = 0

    ELSE IF ( ( ALERT_LOCATION ≤ OTHER_ALERT_COVERAGE ) and
              ( 0 ≥ ALERT_VALUE ≥ THRESH3 ) and
              ( OTHER_ALERT_VALUE ≥ THRESH4 ) )

            SCREENED_ALERT_VALUE = 0

    ELSE IF ( ( ALERT_LOCATION ≤ OTHER_ALERT_COVERAGE ) and
              ( –MIN_MBA ≥ ALERT_VALUE ≥ THRESH5 ) and
              ( OTHER_ALERT_VALUE ≥ THRESH6 ) )

            SCREENED_ALERT_VALUE = MAX_WSA

    ELSE

            SCREENED_ALERT_VALUE = ALERT_VALUE

    END IF

END DO

END ALERT SCREENING SUBROUTINE
```

APPENDIX A

DATA DICTIONARY FOR THE TDWR/LLWAS INTEGRATION ALGORITHM SPECIFICATION

ALERT_LOCATION

description: Set to either LLWAS_ALERT_LOCATION or TDWR_ALERT_LOCATION.

type: character (1 x number of operational runways)

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF', one for each operational runway

ALERT_VALUE

description: Set to either LLWAS_ALERT_VALUE, or TDWR_ALERT_VALUE.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

AVE

description: The average of an LLWAS alert value and a TDWR alert value for a single operational runway.

type: real

values: meters per second

GAIN_ALERT_LOCATION

description: Set to either the LLWAS_ALERT_LOCATION or the TDWR_ALERT_LOCATION for a single operational runway.

type: character

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF'

GAIN_ALERT_VALUE

description: Set to either the LLWAS_ALERT_VALUE or the TDWR_ALERT_VALUE for a single operational runway.

type: real

values: meters per second

INTEGRATION_ALERT_LOCATION

description: The location of first encounter of the windshear for each operational runway to be issued by the integration algorithm.

type: character (1 x number of operational runways)

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF', one for each operational runway

INTEGRATION_ALERT_VALUE

description: The loss or gain value for each operational runway to be issued by the integration algorithm.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

LAST_GAIN

description: For each operational runway answers the question "was the last integration alert on this operational runway a gain?" These values must be stored from call to call.

type: logical (1 x number of operational runways)

values: YES/NO, one for each operational runway

LAST_LOSS

description: For each operational runway answers the question "was the last integration alert on this operational runway a loss?" These values must be stored from call to call.

type: logical (1 x number of operational runways)

values: YES/NO, one for each operational runway

LLWAS_ALERT_LOCATION

description: The "location of first encounter of windshear'' portion of each operational runway alert issued by the LLWAS system.

type: character (1 x number of operational runways)

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF', one for each operational runway

LLWAS_ALERT_VALUE

description: The loss or gain value portion of each operational runway alert issued by the LLWAS system.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

LLWAS_COVERAGE

description: The extent of LLWAS coverage for each operational runway. These are fixed parameters input from the IPF.

type: character (1 x number of operational runways)

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF', one for each operational runway

LLWAS_GF_FACTOR

description: The minimum allowed fraction of a screened LLWAS gain value that can be issued. These are fixed parameters input from the IPF.

type: real (1 x number of operational runways)

values: 0.0 – 1.0, one for each operational runway

LLWAS_MB_FACTOR

description: The minimum allowed fraction of a screened LLWAS loss value that can be issued. These are fixed parameters input from the IPF.

type: real (1 x number of operational runways)

values: 0.0 – 1.0, one for each operational runway

LLWAS_SCREENED_ALERT_VALUE

description: The loss or gain value portion of each LLWAS operational runway alert after passing through ALERT SCREENING.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

LLWAS_THRESHi (i = 1, ... , 6)

description: Thresholds used to screen weak LLWAS alerts. These thresholds are set individually for each operational runway. These are fixed parameters input from the IPF.

type: real (6 x number of operational runways)

values: meters per second, one set (i= 1, ..., 6) for each operational runway

LOSS_ALERT

description: Used internally to ALERT ARBITRATION to determine if the output alert will be the loss or gain alert.

type: logical

values: YES/NO

LOSS_ALERT_LOCATION

description: Set to either the LLWAS_ALERT_LOCATION or the TDWR_ALERT_LOCATION for a single operational runway.

type: character

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF'

LOSS_ALERT_VALUE

description: Set to either the LLWAS_ALERT_VALUE or the TDWR_ALERT_VALUE for a single operational runway.

type: real

values: meters per second

LOSS_GAIN_BUFFER

description: Used to lean towards giving a loss alert if the last integration alert was a loss, or to lean towards giving a gain alert if the last integration alert was a gain. This is a fixed parameter input from the IPF.

type: real

values: meters per second

LOSS_INCREMENT

description: Used to determine if a gain alert value is enough larger than a loss alert value to override the loss. This is a fixed parameter input from the IPF.

type: real

values: meters per second

MAX_WSA

description: The value that a weak unconfirmed MBA is reduced to. This is a fixed parameter input from the IPF.

type: real

values: meters per second

MIN_MBA

description: The magnitude of the weakest allowed microburst alert value. This is a fixed parameter input from the IPF.

type: real

values: meters per second

OTHER_ALERT_COVERAGE

description: Set to either LLWAS_COVERAGE or TDWR_COVERAGE.

type: character (1 x number of operational runways)

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF', one for each operational runway

OTHER_ALERT_VALUE

description: Set to either LLWAS_ALERT_VALUE or TDWR_ALERT_VALUE.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

SCREENED_ALERT_VALUE

description: The screened alert values from either LLWAS or TDWR.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

TDWR_ALERT_LOCATION

description: The "location of first encounter of windshear" portion of each operational runway alert issued by the TDWR system.

type: character (1 x number of operational runways)

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF', one for each operational runway

TDWR_ALERT_VALUE

description: The loss or gain value portion of each operational runway alert issued by the TDWR system.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

TDWR_COVERAGE

description: The extent of TDWR coverage for each operational runway. These are fixed parameters input from the IPF.

type: character (1 x number of operational runways)

values: 'RWY', '1MF', '1MD', '2MF', '2MD', or '3MF', one for each operational runway

TDWR_GF_FACTOR

description: The minimum allowed fraction of a screened TDWR gain value that can be issued. These are fixed parameters input from the IPF.

type: real (1 x number of operational runways)

values: 0.0 – 1.0, one for each operational runway

TDWR_MB_FACTOR

description: The minimum allowed fraction of a screened TDWR loss value that can be issued. These are fixed parameters input from the IPF.

type: real (1 x number of operational runways)

values: 0.0 – 1.0, one for each operational runway

TDWR_SCREENED_ALERT

description: The loss or gain value portion of each TDWR operational runway alert after passing through ALERT SCREENING.

type: real (1 x number of operational runways)

values: meters per second, one for each operational runway

TDWR_THRESHi (i = 1, ... , 6)

description: Thresholds used to screen weak TDWR alerts. The thresholds are set individually for each operational runway. These are fixed parameters input from the IPF.

type: real (6 x number of operational runways)

values: meters per second, one set (i= 1, ..., 6) for each operational runway

THRESHi (i = 1, ... , 6)

description: Set to either LLWAS_THRESHi or TDWR_THRESHi

type: real (6 x number of operational runways)

values: meters per second, one set (i= 1, ..., 6) for each operational runway

APPENDIX B

LLWAS DATA INPUTS

This appendix specifies the variables that are required to be passed from the LLWAS system to TDWR/LLWAS integration and refers to the Network Expansion LLWAS Algorithm Specification, Version 1990.01. These variables are computed in the LLWAS system for each data polling cycle and are required to be passed to TDWR/LLWAS integration for each polling cycle.

1. Alert information (LLWAS_ALERT_VALUE, LLWAS_ALERT_LOCATION)

   Message (rwy, 1) rwy = 1, NUM_RWY ; "1" signifies arrival runway
   Message (rwy, 2) rwy = 1, NUM_RWY ; "2" signifies departure runway

   Each "Message" consists of the following information:

   - Alert type (N.A. for Message Level Integration Algorithm)
   - Alert value (in meters/sec)
   - Alert location (where location = 0 [RWY], 1 [1MF], 2 [2MF], or 3 [3MF] for arrival runways, and 0 [RWY], 1 [1MD], 2 [2MD] for departure runways)

   These global (or "save") variables are set in subroutine "runway_alert_arbitration" (p. 62–65 of the LLWAS specification).

2. Station wind variables (U, V) *The wind data are not used in the TDWR/LLWAS integration algorithm. They are used for display only.*

   U (stat, mode) stat = 1, NUM_STAT ; mode = 3
   V (stat, mode) stat = 1, NUM_STAT ; mode = 3

   These global (or "save") wind variables are computed in subroutine "filter_wind_data" (p. 27–29 of the LLWAS specification) using filter mode 3 (MB_LOSS). These are calculated at each station wind update.