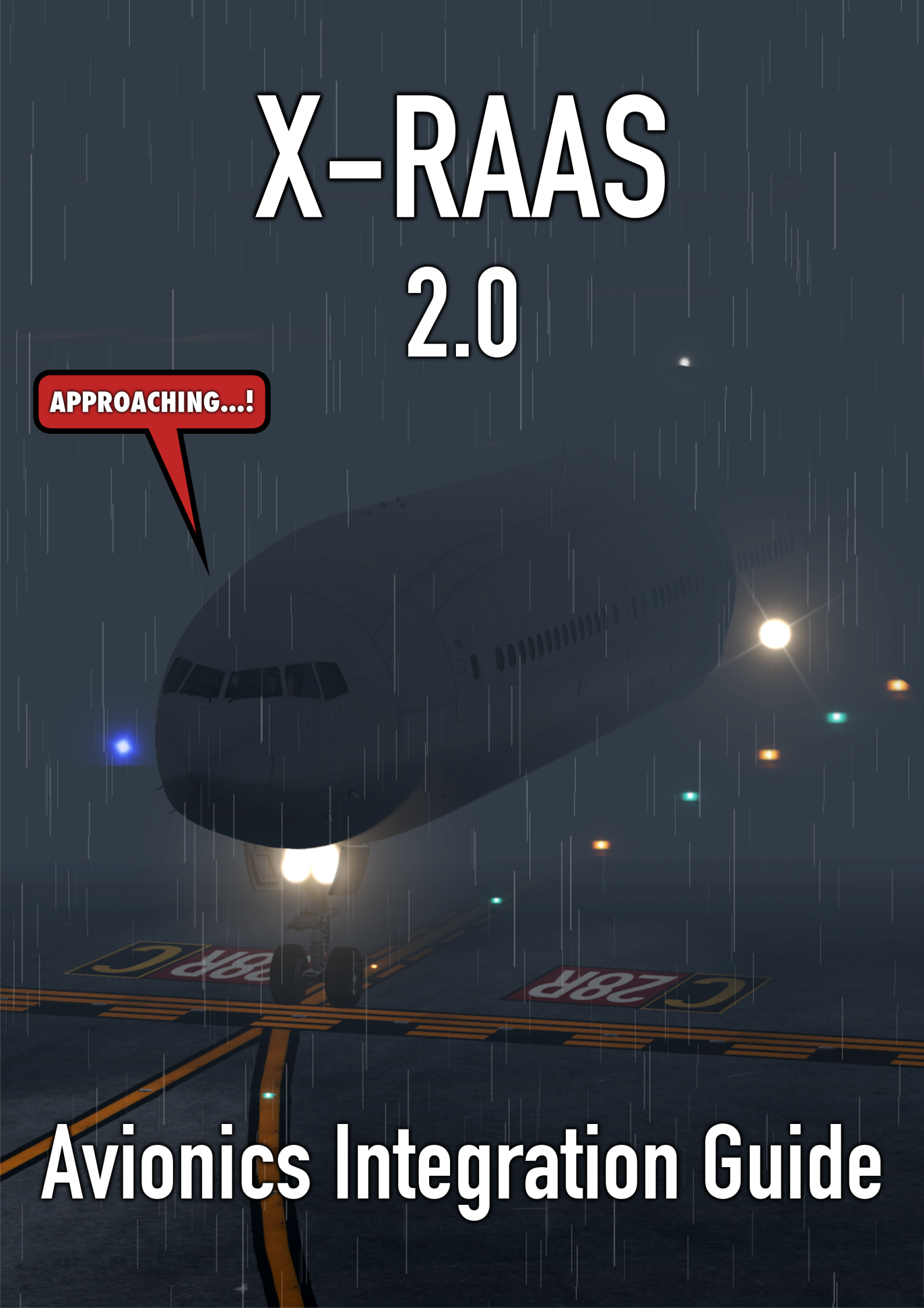
X-RAAS 2.0



Avionics Integration Guide

Table of Contents

1 Introduction

This guide details is meant for aircraft cockpit builders and developers who wish to integrate X-RAAS into their avionics package. Integration has many benefits, such as enabling the display of visual alerts on the aircraft's navigation display in the 3D cockpit and the ability to control the behavior of the stabilized approach monitor.

X-RAAS can be installed either globally in **`Resources/plugins`** or can be embedded directly in the aircraft model in the plugins subdirectory of that model. See section 6 “Embedding X-RAAS into your aircraft“ for more information on how properly embed X-RAAS. Regardless of the method of delivery, however, X-RAAS's interface doesn't change.

2 Receiving visual alert messages

In many situations, when the plugin generates an aural annunciation, it can also generate a visual advisory. This is normally meant to be displayed on the navigation display or a similar such display device in the cockpit. However, since X-RAAS can't directly paint onto the 3D cockpit, this needs to be provided by the aircraft's 3D cockpit logic itself, necessitating this integration. If such integration is not available (e.g. the aircraft model isn't aware of X-RAAS), X-RAAS implements a fallback mechanism where the visual alerts are displayed using an on-screen overlay.

If you choose to provide integration of X-RAAS's visual alerts into your virtual cockpit logic, you will want to do two things:

1. Disable the fallback overlay GUI.
2. Receive the alert text to be rendered onto the ND.

2.1 Disabling the fallback overlay GUI

For this purpose, all you need to do is write a '1' (or any non-zero value) to the “**`xraas/ND_alert_overlay_disabled`**” integer dataref. Doing so disables any new messages from being displayed. Please note that this dataref gets reset back to '0' after each aircraft unload (i.e. in response to **`XPLM_MSG_PLANE_UNLOADED`** when **`param == 0`**). This is to allow for switching between aircraft which do and do not want the overlay enabled. You should set it to '1' in your aircraft avionics loading completion handler. Afterwards, X-RAAS shouldn't touch it until your aircraft is unloaded again.

2.2 Receiving the alert text to be displayed

X-RAAS encodes the string it wants to display into the “**`xraas/ND_alert`**” integer dataref using a somewhat non-trivial format (detailed in appendix). However, to facilitate ease of integration, X-RAAS includes sample code in the “api” directory that helps you with this task. The code is licensed under the MIT license, giving you maximum freedom in its use, so you needn't worry about tainting your project with copyleft-licensed code. The “api” directory contains implementations in three languages: C, Lua and Python. If necessary, implementations in other languages can be provided on request.

The entire API simply consists of one function, **`XRAAS_ND_msg_decode`**. You pass it the value of the “**`xraas/ND_alert`**” dataref and it returns a string-based representation of the encoded message, as well as the color code that is to be applied to the message. For an example of how to use this function, see the included "test_sample" programs in the respective language subdirectories. Alternatively, you can also see how X-RAAS uses it to render its own fallback overlay in the **`render_alert_texture`** function in **`src/nd_alert.c`** in the X-RAAS source code.

When the ND alert is supposed to be removed from the display, X-RAAS sets “**`xraas/ND_alert`**” back to 0.

3 GPWS integration

In the real world, RAAS is a software function of the EGPWS computer. As such, the EGPWS can do things such as override or disable certain or all RAAS annunciations. For X-RAAS, this facility is available through the following set of datarefs (all datarefs in this table are integer datarefs):

Name	Description
xraas/override/GPWS_prio	Setting to 1 or 0 enables or disables the use of the **xraas/override/GPWS_prio_act** dataref.
xraas/override/GPWS_prio_act	Provided the above dataref is set to 1, X-RAAS monitors this dataref to determine if GPWS callouts currently have priority and consequently if X-RAAS annunciations should be silenced. You would typically want to set this dataref to 1 when sounding any GPWS callout such as 'PULL UP!' or 'CAUTION TERRAIN!'. As soon as this is set to '1', X-RAAS pauses any currently playing annunciation and restarts it after the dataref is reset back down to '0'.
xraas/override/GPWS_inop	Setting to 1 or 0 enables or disables the use of the **xraas/override/GPWS_inop_act** dataref.
xraas/override/GPWS_inop_act	Provided the above dataref is set to 1, X-RAAS monitors this dataref to determine if the GPWS computer is inoperative. This could happen e.g. during an electrical fault. While set to '1', X-RAAS is inhibited. When reset back to '0', X-RAAS continues normal operation (assuming sufficient electrical power). Missed or interrupted annunciations are not replayed, X-RAAS will behave as if it had been forcibly powered down and back up.
xraas/override/GPWS_flaps_ovrd	Setting to 1 or 0 enables or disables the use of the **xraas/override/GPWS_flaps_ovrd_act** dataref.
xraas/override/GPWS_flaps_ovrd_act	Provided the above dataref is set to 1, X-RAAS monitors this dataref to determine if the GPWS flaps override mode is active. When the GPWS flaps override mode is active, all X-RAAS flaps configuration checks are inhibited.
xraas/override/GPWS_terr_ovrd	Setting to 1 or 0 enables or disables the use of the **xraas/override/GPWS_terr_ovrd_act** dataref.
xraas/override/GPWS_terr_ovrd_act	Provided the above dataref is set to 1, X-RAAS monitors this dataref to determine if the GPWS terrain override mode is active. When the GPWS terrain override mode is active, the steep and excessive speed approach monitors as well as the off-runway landing monitor are inhibited.

All these datarefs are automatically reset by X-RAAS back to 0 after an X-RAAS reset (caused by either repositioning the aircraft or cycling the avionics electrical power), so be sure to set them periodically if the condition pertaining to a dataref persists.

4 Approach speed monitor

During approach, X-RAAS can monitor the aircraft's speed in relation to its position from the runway threshold and check if the aircraft is exceeding the pre-determined approach speed set in the FMS at stabilized approach check gates. To set this approach speed, use the following integer datarefs:

Name	Description
xraas/override/Vapp	Setting to 1 or 0 enables or disables the use of the **xraas/override/Vapp_act** dataref.
xraas/override/Vapp_act	Provided the above dataref is set to 1, X-RAAS reads the value of this dataref to determine the approach speed (V_{APP}). Refer to section 4.13 of the X-RAAS user manual for details on what an approach speed is.
xraas/override/Vref	Setting to 1 or 0 enables or disables the use of the **xraas/override/Vref_act** dataref.
xraas/override/Vref_act	Provided the above dataref is set to 1, X-RAAS reads the value of this dataref to determine the reference speed (V_{REF}). Refer to section 4.13 of the X-RAAS user manual for details on what a reference speed is.

If both **xraas/override/Vapp** and **xraas/override/Vref** are '1', **xraas/override/Vapp_act** has priority.

5 Exact flaps setting checks

X-RAAS checks flaps settings in two monitors:

- The on-runway lineup monitor: designed to check that the appropriate takeoff flaps setting has been made prior to lining up on the runway. This is to help avoiding triggering the takeoff configuration warning if the pilots forgot to set the appropriate takeoff flaps setting and initiate takeoff.
- The stabilized approach flaps monitor: designed to verify that the appropriate landing flaps setting is selected at specific altitude gates on approach. This is to help prevent unstabilized approaches.

The default behavior is to use an upper and lower flaps setting gate for the takeoff flaps check and only a lower gate for the landing flaps check. Naturally, these are rather imprecise so as to allow for meaningful defaults, assuming no tuning for a specific aircraft. However, if your aircraft simulates an FMS where the flaps setting can be made explicitly by the pilot[1], you can tell X-RAAS **exactly** what kind of flaps setting it should expect to see in either of these cases, using the datarefs in the take below. X-RAAS will then issue caution annunciations if it detects a flaps setting that deviates in any way from the value set in the datarefs.

Name	Description
xraas/override/takeoff_flaps	Setting to 1 or 0 enables or disables the use of the **xraas/override/takeoff_act** dataref.
xraas/override/takeoff_flaps_act	Provided the above dataref is set to 1, X-RAAS reads the value of this dataref to determine the exact required flaprqst value that must be set prior to lining up on the runway. If the actual value of the flaprqst dataref differs by more than 0.01 from this dataref's value, X-RAAS will issue a caution advisory on lining up on a runway for takeoff.
xraas/override/landing_flaps	Setting to 1 or 0 enables or disables the use of the **xraas/override/landing_flaps_act** dataref.

1 For example on some FMCs on Boeing aircraft on the TAKEOFF REF and APPROACH REF pages.

Name	Description
xraas/override/landing_flaps_act	Provided the above dataref is set to 1, X-RAAS reads the value of this dataref to determine the exact required flaprqst value that must be set during approach to a runway. If the actual value of the flaprqst dataref differs by more than 0.01 from this dataref's value, X-RAAS will issue appropriate advisories during approach.

6 Embedding X-RAAS into your aircraft

While X-RAAS can operate as a stand-alone plugin in the global X-Plane **Resources/plugin** directory, embedding X-RAAS into an aircraft model is a great way to make sure that the RAAS functionality is always available in a particular aircraft model, no matter the global environment. To build an embeddable version of X-RAAS, run the following command in the top level of the X-RAAS source tree:

```
$ ./build_release -e
```

The resulting “X-RAAS2” directory contains the embeddable version of the plugin. An embeddable build differs from a stand-alone build in the following ways:

1. The plugin signature, name and description are altered to prevent a conflict and confusion with the globally installed version in the plugin manager GUI.
2. The plugin automatically disables itself if it detects that a stand-alone installation of X-RAAS is already present in the global **Resources/plugins**. This is to prevent duplicate annunciations and also allows users to upgrade their X-RAAS installation without having to modify the aircraft model's internal directory layout.
3. An embeddable build doesn't load a global **X-RAAS.cfg**, since it cannot be invoked when other aircraft are loaded the global config file shouldn't even be present (and if it is, it might contain stanzas we don't understand if it's from a different X-RAAS version).
4. The config GUI only allows saving and resetting the configuration of the aircraft that the plugin is embedded in. Rather than having 4 buttons:
   - SAVE aircraft configuration
   - SAVE global configuration
   - RESET aircraft configuration
   - RESET global configuration

   the config GUI only shows two buttons (which only operate on the aircraft-specific configuration):
   - SAVE configuration
   - RESET configuration
5. The X-RAAS.cache for caching the airport layouts and navdata is relocated from the global **Output/caches/X-RAAS.cache** into the plugin installation directory itself to avoid polluting the global simulator environment.

In terms of interface code, nothing changes between the stand-alone and embedded version. You should still use the same interfaces and datarefs described in this guide.

7 About the X-RAAS project

7.1 Author

X-RAAS was written by Sašo Kiselkov. You can contact the author at:

skiselkov@gmail.com

7.2 License

X-RAAS is open-source software distributed under the terms of the **Common Distribution and Development License**. A copy of the license text is included in the software package in **`COPYING`** file. The quick'n'dirty of the terms of this license:

1. You can copy, modify, run and use X-RAAS in any way you want.
2. You can redistribute your copies (whether modified or not) and even sell X-RAAS. You can incorporate X-RAAS into your own projects (whether open-source or not).
3. If you modify X-RAAS and wish to distribute it in any way, you must share the source code for the modifications you have made to it. If you've made it part of a larger work, you don't have to share the source code for all of your work, only the bits of X-RAAS you've modified.

For the full list of terms, refer to the **`COPYING`** file.

An exception to this license are the files under the **`api`** folder. These are distributed under the terms of the MIT License. This pretty much allows you to do whatever you want with them. The full license text is embedded in the header of each file under **`api`**.

A Visual alert encoding rules

This section describes how X-RAAS encodes visual alert messages into the "**`xraas/ND_alert`**" dataref. It is mentioned here for completeness and you are not required to understand the encoding rules unless you plan on implementing your own decoder.

The message is encoded as a binary integer in native byte order. This is broken down into multiple bitfields:

Bits	Description
0 – 5	**Message type** Valid values: • 0: no message present • 1: 'FLAPS' message • 2: 'TOO HIGH' message • 3: 'TOO FAST' message • 4: 'UNSTABLE' message • 5: 'TAXIWAY' message • 6: 'SHORT RUNWAY' message • 7: 'ALTM SETTING' message • 8: 'APP' message • 9: 'ON' message • 10: 'LONG LANDING' message • 11: 'DEEP LANDING' message Message types 8 and 9 ('APP' and 'ON' messages) also fill the bitfields for the runway ID, runway ID suffix and distance available.
6 – 7	**Color** Valid values: • 0: message should display in a green color (normal message). • 1: message should display in an amber color (caution message). • 2-3: reserved
8 – 13	**Runway ID** Used by message types 8 and 9. Valid values: • 0: 'TAXIWAY'. This is used to signal "ON TAXIWAY", by passing message type 9 and a runway ID of 0. • 1 – 36: The runway ID for runway '01' through '36'. • 37: "RWYS" value. Used when multiple runways are being approached, e.g. "APP RWYS" (message type 8, runway ID 37).
14 – 15	**Runway ID suffix** Used by message types 8 and 9. Valid values: • 0: no suffix. ND should simply show the runway ID, e.g. "36". • 1: 'RIGHT'. ND should show runway ID with abbreviated suffix, e.g. "36R". • 2: 'LEFT'. ND should show runway ID with abbreviated suffix, e.g. "36L". • 3: 'CENTER'. ND should show runway ID with abbreviated suffix, e.g. "36C".
16 – 23	**Runway length available** Used by message types 8 and 9. Runway length rounded down to the nearest 100 feet or meters. A value of '0' in this field means '*no display*'. The ND should display the value (if non-zero) as a fixed two-digit number (printf-like format string "%02d").

A.1 Sample message encodings

Dataref value	Description	ND Display
0x00000041	Amber 'FLAPS' message	FLAPS
0x00000042	Amber 'TOO HIGH' message	TOO HIGH
0x00000043	Amber 'TOO FAST' message	TOO FAST
0x00000044	Amber 'UNSTABLE' message	UNSTABLE
0x00000045	Amber 'TAXIWAY' message	TAXIWAY
0x00000046	Amber 'SHORT RUNWAY' message	SHORT RUNWAY
0x00000047	Amber 'ALTM SETTING' message	ALTM SETTING
0x00002308	Green 'APP' message, runway ID **0x23** (35), runway suffix 0 (“”)	APP 35
0x00006308	Green 'APP' message, runway ID **0x23** (35), runway suffix 1 (“R”)	APP 35R
0x00002508	Green 'APP' message, runway ID **0x25** (37, 'RWYS')	APP RWYS
0x00142348	Amber 'APP' message, runway ID **0x23** (35), runway suffix 0 (“”), distance available **0x14** (2000 feet/meters)	APP 35 20
0x00086348	Amber 'APP' message, runway ID **0x23** (35), runway suffix 1 (“R”), distance available **0x08** (800 feet/meters)	APP 35R 08
0x00000049	Amber 'ON' message, runway ID **0x00** ('TAXIWAY')	ON TAXIWAY
0x00002309	Green 'ON' message, runway ID **0x23** (35), runway suffix 0 (“”)	ON 35
0x00006309	Green 'ON' message, runway ID **0x23** (35), runway suffix 1 (“R”)	ON 35R
0x00002509	Green 'ON' message, runway ID **0x25** (37, 'RWYS')	ON RWYS
0x0014E349	Amber 'ON' message, runway ID **0x23** (35), runway suffix 3 (“C”), distance available **0x14** (2000 feet/meters)	ON 35C 20
0x0008A349	Amber 'ON' message, runway ID **0x23** (35), runway suffix 2 (“L”), distance available **0x08** (800 feet/meters)	ON 35L 08
0x0000004A	Amber 'LONG LANDING' message	LONG LANDING
0x0000004B	Amber 'DEEP LANDING' message	DEEP LANDING