

BMS TECHNICAL MANUAL



Version: BMS 4.35

CHANGE 1.00

12. 2020



FOREWORD

PURPOSE AND SCOPE

This Falcon 4 BMS manual contains information on managing the more technical aspects of the simulation.

The following manuals supplement this manual to establish the complete Falcon BMS 4.35 series:

- TO-BMS1F-16CM-1 (aircraft, avionics, normal procedures and abnormal procedures).
- BMS-Training (documentation to accompany Falcon BMS training missions).
- TO-BMS1F-16CM-34-1-1 (weapon systems, support equipment and munitions).
- BMS-Comms & Navigation Manual (how to get the best out of ATC, Radio & Charts).
- BMS-Naval-Ops (Naval Operations from aircraft carrier in BMS).
- Checklists and Cockpit Diagrams (avionics, weapons,...)
- Airport Approach & Navigation Charts (including KTO AIP, Chart Tutorial).
- Key Files & Input (Key File, Key File Editor, Keystrokes, Callbacks, etc.).

These are all located in the `\Docs` folder of your Falcon BMS install, with other supporting documents.

COPYRIGHT STATEMENTS

Falcon BMS is a community mod developed and published by Benchmark Sims for use with licensed copies of Falcon 4.0. Unauthorized rental, sales, arcade use, charging for use, or any commercial use of this mod or part thereof is prohibited.

This mod is for non-commercial use only.

This mod was created by Benchmark Sims with the permission of Billion Soft (Hong Kong) Limited.

This mod and all included content are in no way affiliated with Billion Soft (Hong Kong) Limited or Retroism.

© 2003-2020 Benchmark Sims. All rights reserved.

Falcon is a registered trademark of Billion Soft (Hong Kong) Limited. Falcon Collection and Falcon 4.0 are published by Retroism.

Retroism, the Retroism logo and the Billion Soft logo are trademarks or registered trademarks.

© 2020 Billion Soft (Hong Kong) Limited. All rights reserved.

The manufacturers and intellectual property right owners of the vehicles, weapons, sensors and other systems represented in Falcon BMS in no way endorse, sponsor or are otherwise involved in the development of Falcon BMS.

The BMS Technical Manual is published by the BMS DOC team.

Unauthorized rental, sales, charging for use, or any commercial use of this manual or part thereof is prohibited.

This manual is for non-commercial use only.

No reproduction of this manual or part of this manual (except printing for your own personal use) is allowed without the written permission of the BMS DOC team.

© 2003-2020 Benchmark Sims. All rights reserved.





1. TABLE OF CONTENTS

PART 1 : For the User

2. The HOTAS issue	2-11
2.1 Keystroke vs DirectX.....	2-11
2.2 Use an existing configuration	2-13
2.2.1 What is included and where are the files?	2-13
2.2.2 Adapting the Falcon BMS.cfg to your stickfile.....	2-15
2.2.3 Axis declaration	2-16
2.2.4 Conclusion	2-16
2.3 Creating your own (Keystrokes)	2-17
2.4 Creating your own (DirectX)	2-21
2.5 Conclusion	2-22
3. Avionics Configurator	3-23
3.1 Examples of modifications.....	3-24
3.2 Options explanation	3-28
4. List of Keyboard Layouts.....	4-29
4.1 Default US (QWERTY) Keyboard layout.....	4-29
4.2 Pitbuilder US (QWERTY) Keyboard layout	4-30
4.3 Default French (AZERTY) Keyboard layout	4-31
4.4 Pitbuilder French (AZERTY) Keyboard file.....	4-32
4.5 Default German (QWERTZ) Keyboard Layout.....	4-33
4.6 Pitbuilder German (QWERTZ) Keyboard Layout.....	4-34
5. External Display Support	5-35
5.1 Falcon BMS Display Extraction	5-35
5.2 RTT Remote	5-35
5.2.1 RTTServer settings	5-35
5.2.2 RTT Client settings	5-36
5.2.3 Using RTTRemote	5-37
5.2.4 Pros & Cons.....	5-37
5.3 Third-Party extraction.....	5-37
5.3.1 MFDE	5-37
5.3.2 Y.A.M.E.....	5-37
6. The FPS Quest.....	6-38
7. Weather and How to create FMAPs.....	7-40





7.1	Weather changes in 4.35	7-41
7.2	Weather changes in 4.34	7-41
7.2.1	Wind change	7-41
7.2.2	Clouds change.....	7-41
7.3	Weather Commander	7-42
7.4	F4Weather (F4Wx).....	7-46
7.5	Maps Auto Update.....	7-48
7.5.1	Naming convention	7-49
8.	Views	8-50
8.1	Field of View	8-50
8.1.1	How to change default FOV settings	8-51
8.2	View panning with the mouse.....	8-52
8.3	Touchscreen Use for Cockpit Interaction	8-52
8.4	Custom Views	8-52
8.4.1	Capture View Position	8-53
8.4.2	Custom view code syntax	8-53
8.4.3	Edit 3dckpit.dat files	8-53
8.4.4	Custom Views in 3d	8-54
8.5	How to define TopGun Views.....	8-54
8.6	Toggle HUD Rendering (alt c : h).....	8-55
8.7	Displays.....	8-55
8.7.1	Infobar (CTL 1)	8-55
8.7.2	Radio Subtitles (CTL 2).....	8-56
8.7.3	Flap Display (CTL 3).....	8-56
8.7.4	Engine Display (CTL 4).....	8-56
8.7.5	Show Score (ALT c : d).....	8-57
8.7.6	Frame Rate (ALT c : f).....	8-57
8.7.7	Online Status (ALT c : o).....	8-57
8.8	Pilot kneeboards.....	8-59
8.8.1	Managing Kneeboards with the latest reslease of WDP	8-59
9.	Aircraft Radios	9-62
9.1	Introduction.....	9-62
9.2	UHF	9-62
9.3	VHF.....	9-62
9.4	What about the AI? How do they fit in?	9-63





9.5	IVC impact on the new Radio Code	9-64
9.6	Managing two radios	9-64
9.7	Setting the UI Radio Frequencies	9-65

PART 2 : For the Advanced User

10.	Key Files	10-67
10.1	Categories & Sections:.....	10-67
10.1.1	Category and Section code lines.....	10-67
10.1.2	Non category & section lines in the UI:	10-68
10.2	The Terms:.....	10-69
10.2.1	First Term:.....	10-70
10.2.2	Second Term:.....	10-70
10.2.3	Third Term:	10-70
10.2.4	Upper case vs. lower case:	10-72
10.2.5	Keys vs. Mouse (right / left click, scroll wheel):.....	10-72
10.3	Overview categories & sections:	10-73
10.4	Key files - general info:	10-75
10.4.1	Deprecated & outdated / Dev callbacks:.....	10-75
10.4.2	TrackIR, Fraps, Vac & TeamSpeak:.....	10-75
10.4.3	The key file profiles:.....	10-76
10.4.4	Notes on the new pitbuilder key file:	10-76
10.4.5	Backup	10-77
10.5	The inner working of key files.....	10-77
10.5.1	Keyboard code lines.....	10-77
10.5.2	DirectX button code lines	10-83
10.5.3	DirectX POV Hat code lines.....	10-84
10.6	How to edit key files:	10-86
10.6.1	Assignments via BMS Setup menu	10-86
10.6.2	Using an editor:	10-89
10.6.3	Editing key files with the Key File Editor:	10-92
10.6.4	DirectX shifting facility.....	10-92
10.6.5	DX device limitation:.....	10-98
10.7	The DX Device Order:	10-99
10.7.1	How to find out your DX device order and DX button IDs:	10-99
10.7.2	How to avoid DX device order and xaxis changes	10-100





10.7.3	How to set DX device order	10-100
10.7.4	Basic DX device setup flow	10-102
10.7.5	Technical details about the DX device sorting.....	10-102
10.8	DirectX device specifics:	10-104
10.8.1	How to cancel MRM/DF override Modes:.....	10-104
10.8.2	How to overcome DX button shortcomings?	10-104
10.8.3	How to use POV hats on two devices	10-106
10.9	Key file options & specifics:	10-108
10.9.1	Changing ICP-Numpad mapping (1=7 -> 7=1):	10-108
10.9.2	How to change the DX POV functions (Trim vs. View & other functions):.....	10-108
10.9.3	Assigning keys to Extra MFDs (3rd & 4th MFD):.....	10-110
10.9.4	Double entries:	10-110
10.10	Troubleshooting:	10-111
10.10.1	Why does BMS crash when loading a key file?	10-111
10.10.2	Stuck key:.....	10-111
10.10.3	How to enable EyeFly (Free Cam):.....	10-112
10.10.4	The mouse does not work anymore in pit:.....	10-112
10.10.5	Keyboard keys and combinations you have to be careful with:.....	10-112
10.10.6	(Pretty) Screenshot vs. PrtScr:.....	10-113
10.10.7	Why we do not hear cockpit sounds when pressing a key:.....	10-114
10.11	Notes on the Development Callbacks.....	10-115
10.11.1	Introduction:.....	10-115
10.11.2	Dev Functions:	10-116
11.	Notes on the 4.34 Comms system	11-124
11.1	Introduction.....	11-124
11.2	The Ground agencies	11-125
11.3	The Air agencies.....	11-125
11.4	Presets	11-126
11.5	DTC Storage	11-127
11.6	The VHF assignment system.....	11-127
11.7	How to avoid VHF conflicts.....	11-129
11.8	Fallback solution and AI.....	11-130
11.9	3rd party theater dev notes	11-130
11.10	Stock BMS VHF layout.....	11-130





- 12. User File Structure 12-131**
 - 12.1 Structure of the TEmissionname.ini 12-131
 - 12.2 Structure of the CALLSIGN.INI 12-133
 - 12.2.1 Example of a callsign ini file:..... 12-133
 - 12.3 Notes regarding callsign.ini and TEmissionname.ini 12-136
 - 12.4 Structure of the IFF file 12-137
 - 12.4.1 IFF MODES AND CODES FOR A GIVEN MISSION:..... 12-137
 - 12.4.2 MODE 4 KEYS 12-137
 - 12.4.3 MODE 1 CODES..... 12-137
 - 12.4.4 MODE 2 CODES..... 12-138
 - 12.4.5 MODE 3 CODES..... 12-138
 - 12.4.6 MODE ACTIVATION..... 12-139
 - 12.4.7 AI USE OF IFF..... 12-139
 - 12.4.8 IFF POLICY FILE “.IFF” FORMAT..... 12-140
 - 12.5 Structure of the ATC file 12-143
 - 12.5.1 Airbase general setting..... 12-143
 - 12.5.2 The runway information section: 12-145
 - 12.6 Structure of the STATIONS+ILS.DAT file 12-150
 - 12.7 Structure of the RADIOMAP.DAT file..... 12-151
- 13. List of the Config options 13-152**
 - 13.1 Avionics..... 13-152
 - 13.2 Campaigns (the options below also influence Tactical Engagement) 13-152
 - 13.3 General 13-152
 - 13.4 Hardware 13-153
 - 13.5 Shaders 13-154
 - 13.6 Track IR settings..... 13-155
 - 13.7 Multiplayer 13-155
 - 13.8 Other Options 13-156
 - 13.9 New 4.35 (& 4.34.1, 2 ,3 & 4) Options..... 13-166
 - 13.10 Options removed from 4.35 13-169
 - 13.11 Options removed from 4.34 13-171
 - 13.12 TrackIR Axis Customisation..... 13-172
- 14. Shared Memory Changes 14-173**
 - 14.1 New 4.35.0 changes..... 14-173
 - 14.2 Changes for 4.34.1, not previously documented 14-173





- 14.3 4.34.0 changes 14-174
- 14.4 From 4.32 to 4.33 14-174
- 15. New Keyboard Commands 15-176**
 - 15.1 Quick Reference..... 15-176
 - 15.2 Changes For 4.35.0 15-181
 - 15.3 Changes For 4.34.0 15-183
 - 15.4 Changes For 4.33 Update 1 15-189
 - 15.5 Conversion From 4.32 To 4.33..... 15-190
 - 15.6 Renamed Callbacks..... 15-199
 - 15.7 Deleted Callbacks..... 15-200
 - 15.8 Outdated Callbacks..... 15-201

PART 3: For the third party developer

- 16. Training Scripts..... 16-203**
 - 16.1 Script Files:..... 16-203
 - 16.1.1 Manual Scripts: 16-203
 - 16.1.2 Automatic Scripts: 16-204
 - 16.1.3 How to assign script files to a mission file..... 16-204
 - 16.2 Basic Concepts 16-205
 - 16.2.1 Variables: 16-205
 - 16.2.2 Script vs. Falcon Process:..... 16-205
 - 16.2.3 Boolean Functions: 16-205
 - 16.2.4 Code Sections: 16-205
 - 16.2.5 Code Lines:..... 16-205
 - 16.2.6 Mission Start with Scripts: 16-206
 - 16.3 Screen Coordinates / Cursor Position:..... 16-206
 - 16.4 Functions and Arguments:..... 16-208
 - 16.5 Function Reference..... 16-208
 - 16.5.1 Cursor Positioning Functions: 16-208
 - 16.5.2 Output Functions:..... 16-209
 - 16.5.3 Layout Functions..... 16-211
 - 16.5.4 User Interaction Functions: 16-216
 - 16.5.5 Section Functions:..... 16-218
 - 16.5.6 Sound Functions: 16-222





16.5.7	Cockpit Orientation Functions:.....	16-223
16.5.8	Command & Fault Functions:	16-225
16.5.9	Initial Mission Setup Functions:.....	16-227
16.5.10	Time Management Functions:.....	16-228
16.5.11	Other Script Functions:	16-229
16.5.12	Boolean Functions:	16-231
16.5.13	Removed Functions Reference:.....	16-232
16.6	Training Script Appendix.....	16-232
16.6.1	Color Code Examples	16-232
16.6.2	Fault Codes	16-233
17.	Third Party Theater DEV notes.....	17-238
17.1	Tilesets	17-238
17.2	Add-On Theaters - Tilesets	17-239
17.3	Add-On Theaters - File locations	17-239
17.4	HiRes Textures	17-240
17.5	Adding Carrier to campaign.....	17-240
17.5.1	Database: Creating Objectives associated with Airbases.....	17-240
17.5.2	Database: Carrier Objective Data	17-241
17.5.3	Campaign Files: Placing Objectives in the Campaign	17-241
18.	IVC.....	18-243
18.1	Purpose & Implementation Overview	18-243
18.2	IVC dos and don'ts.....	18-244
18.3	Using the Voice Server Program.....	18-244
18.4	Using the Voice Client.....	18-246
18.4.1	Command Line Switches and Options	18-246
18.4.2	INI File Setup.....	18-256
18.4.3	Client User Interface.....	18-259
18.4.4	Interactions between the Client and the Game	18-261
18.5	Radio Sound Effects.....	18-262
18.5.1	Digital Signal Processing Sound Effects	18-262
18.5.2	Sidetone.....	18-264



PART 1

FOR THE USER

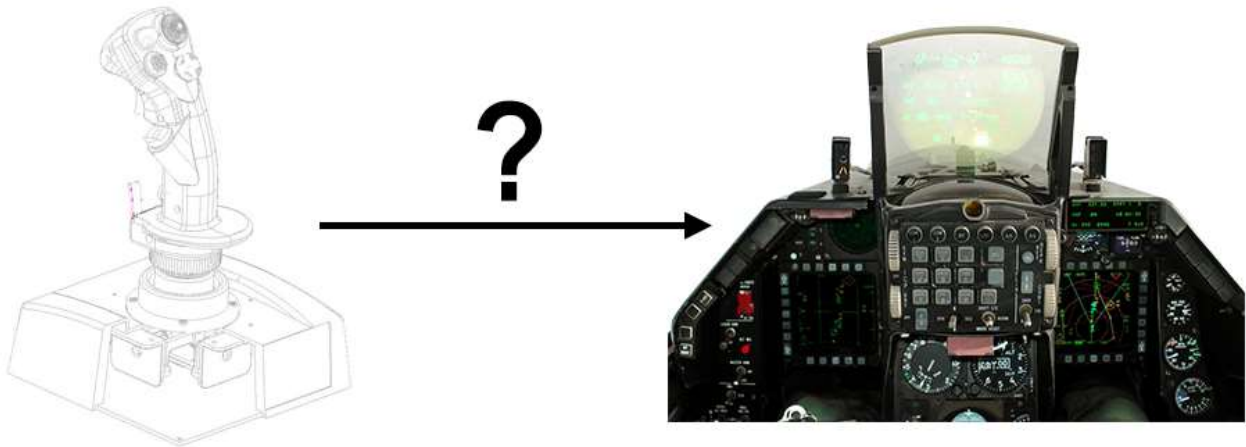


2. The HOTAS issue

One of the first challenges new users face is configuring their HOTAS system. Understanding the basic concept of linking HOTAS or Joystick buttons to cockpit functions in the BMS F-16 cockpit is essential. It is a challenging concept to explain because it depends on the type of Joystick you use and we obviously can't explain them all.

Nevertheless, we will try to clear possible confusion and provide general guidelines on how to proceed with the many options you have at your disposal.

On one side you have your HOTAS and on the other side you have the F-16 Cockpit in BMS. How can we link them both together so the action on the HOTAS has an influence in the F-16 cockpit?



This is done through Keystrokes or Direct X inputs.

There you go; your first challenge is to understand the different philosophy between keystrokes and DX inputs.

2.1 Keystroke vs DirectX

Keystrokes are Keyboard inputs. For instance, a combined press of SHIFT and CONTROL and the b key together yields a "SHF CTRL b" key input.

Note: Keyboards emit a keycode when a key is pressed and another keycode when the key is released.

DirectX are default joystick buttons. A 12-button joystick will have 12 DirectX entries that could be assigned to specific SIM functions depending on your joystick or software.

Direct X devices are limited to 32 entries (labelled 1 to 32 in Windows and 0 to 31 in BMS - adding another layer of confusion when you attempt to trace issues between Windows and BMS).

The maximum number of devices you can connect to Windows is 16; giving you a total of 512 possible entries.

Both Keystrokes and DirectX entries are explained in detail in Chapter 10 of this Technical Manual.





Keystrokes are not directly recognised by BMS. Each cockpit function, or more specifically for this chapter HOTAS function, has a callback associated with it. For your joystick to activate the desired function you have to invoke the specific callback for that function.

The callbacks are linked to the keyboard inputs (keystrokes) in the .key files.

BMS Function → Callback → Keystrokes → Joystick
or
Joystick → Keystrokes → Callback → BMS Function

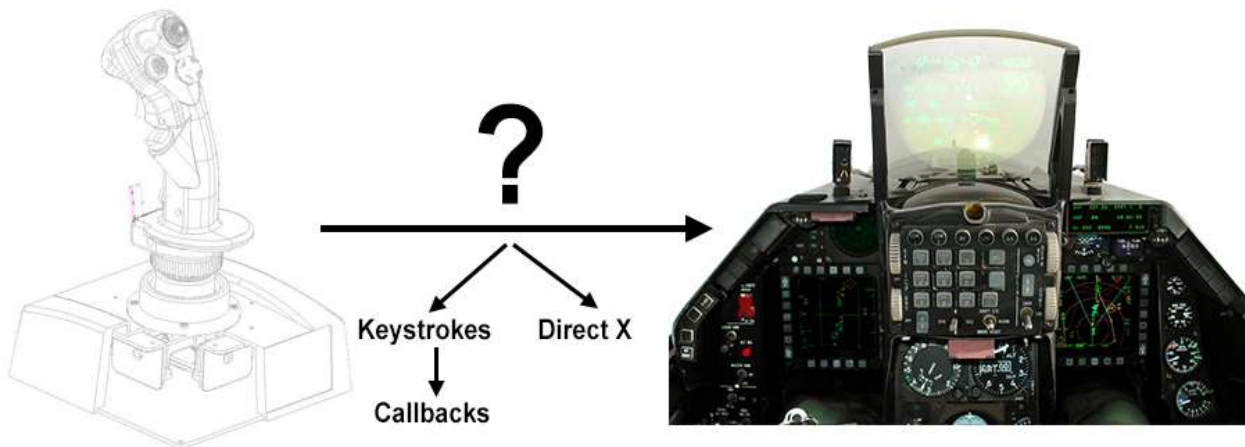
So first you need to know which callback activates the desired function. Next locate the associated keystroke for that callback in the key file and finally tell your joystick that when you press this button you want it to invoke the callback which emits this keystroke.

Still here?

Direct X inputs are easier to set up because you do not have that extra “callback” step. Assigning a DirectX function is usually just point and click: You select a function; hit the desired joystick button and it’s done.

On the other hand it is not quite so easy to manage advanced options such as double layer functions (like the S3/pinky button on the Cougar which allows the user to press a button and have one key emulated and press the same button with S3 held in and have a different key emulated).

It is possible, but more complicated and not recommended for most first-time users.



The choice of using one versus the other is yours to make. Both have their advantages and disadvantages.

One thing is critical though: some functions in BMS require a **long** press. A keystroke press may be interrupted by another keystroke and the long press might not be detected by BMS. A DirectX input is never interrupted; it remains active as long as the button is kept depressed. Therefore all long presses should be set up using DirectX to avoid this problem.

You are still here? Okay, the hardest part is done; let’s see if we can confuse you even more.





2.2 Use an existing configuration

The Docs folder (`\Docs\01 Input Devices\03 HOTAS Setup`) contains pre-made configurations for most popular HOTAS solutions on the market, providing everything you need without having to start programming a joystick. The files provided are all DirectX based and most invoke double layer, which is not always the simplest to setup or use. These files will get you flying, but can be confusing for users who want to start customising them.

HOTAS Cougar is unsurprisingly still the most used HOTAS configuration for BMS. In previous versions of BMS we provided configuration setup for the Cougar called *Dunc_DX*. With 4.34 these files have been updated and renamed *Cougar_DX*. We will use the *Cougar_DX* configuration files as an example.

Note: the files may be different, but the procedure is the same for the other joystick files. Each is explained in its own section in the BMS Device Setup Guide.pdf.

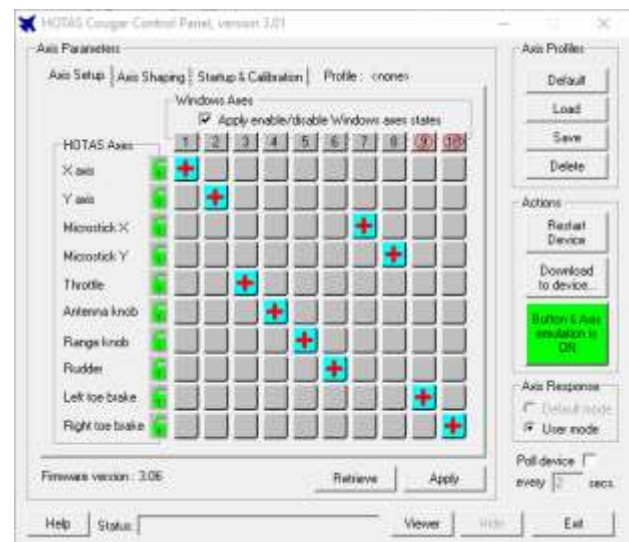
2.2.1 What is included and where are the files?

Let's have a look at the provided files in the `Docs\01 Input Devices\03 HOTAS Setup\TM Cougar` folder.

- *Cougar_DX.tmm* => Foxy macro file
- *Cougar_DX.tmj* => Foxy joystick file
- *BMS - Basic TM Cougar_DX.key* => BMS keyfile
- *DX Code Line.txt* => Specific DX programming for this device
- *DeviceSorting_Example.txt* => Example DX device order
- *HOTAS Print Layout.pdf* => Printable overview of assigned functions

Cougar_DX tmj and tmm files are the native format for Cougar programming. These files can be opened with Foxy HOTAS Cougar Edition but can also be loaded into your Cougar straight from the CCP (Cougar Control Panel) by clicking on the 'Download to device' button just above the green emulation ON button. You will be prompted for the location of the TMJ and TMM file. Navigate to the `\01 Input Devices\03 HOTAS Setup\TM Cougar` folder.

Your Cougar axis must be set in the same way as the CCP image to the right. Microsticks must be on axis 7 & 8 and toebrakes on 9 & 10. This ensures that the microstick can be declared as an analogue axis in the BMS UI.



The key file "BMS – Basic TM Cougar_DX.key" must be selected in BMS for this stick configuration to work. This is because all these stick configurations are DirectX and none of the DirectX bindings are included in the default keyfile located in the `\User\Config` folder because they are specific to each device.

In our example, the *Cougar_DX* key files do still correspond to the Basic (default) key file. The only difference is the addition of relevant Cougar-specific DX lines at the end of the key file.





The provided key file must be copied to the `\User\Config` folder and then selected in the BMS UI. First copy “BMS - Basic TM Cougar_DX.key” into your `\User\Config` folder. Next launch BMS and go into the `SETUP > CONTROLLERS`. Click the `LOAD` button; a window will open and prompt you for the file you want to load. Select “BMS - Basic TM Cougar_DX” and click `LOAD`.



The **DX Code Lines.txt** file is actually the difference between the basic key file and the Cougar DX basic key file (BMS - BasicDX TM Cougar.key or BMS - BasicDX TM Cougar_MFD.key if you also have MFDs).

The contents of this txt file are pasted at the end of the BMS_basic.key file to create the specific Cougar_DX basic key file. In other words, you can paste the contents of this file into any custom key file you may use to adapt your keyfile to the Cougar programming provided with BMS.

The **DeviceSorting_Example.txt** is an example related to using a Cougar and 2 Thrustmaster MFDs. Direct X can be a bit touchy when controllers are connected and disconnected often (or fail to be recognised by Windows). Each device has a dynamic priority number and if the first device disappears from the list, for whatever reason, the next device takes up its priority number and all following devices change their ID as well.

DX bindings (buttons 1 to 32 (0 to 31 in BMS)) correspond to the ID device number. ID 1 should be the Cougar, but if the Cougar fails and drops from the list the next device connected (the left MFD for example) will become ID 1 and all your DX bindings will now refer to the left MFD instead. What a mess!

Some say DX is easier but look at that “simple” explanation above and you will realise that it’s great when everything is working as expected, but if anything goes wrong it will drive you completely mad unless you understand how DirectX works.

To overcome this issue a DeviceSorting.txt file was created, which lists the order of the DX devices (connected or not) that BMS will use. A good DeviceSorting file will ensure that your button configuration is less volatile and less dependent on device unplugging. As a general rule, the order of the DeviceSorting file should always have the primary controls first: stick, throttle & rudder if connected separately (Cougar is one device for all 3), then the other controllers dedicated to avionics (MFDs etc).

For more information about the DeviceSorting file, please refer to chapter 10.7.2 later in this manual.





The **HOTAS Print Layout.pdf** is a graphical layout showing what the buttons do. Print this file and take it with you on your first flights. Keep it handy so you can refer to it as often as needed.

DX HOTAS Cougar (Stick & Throttle) - F4BMS Print Layout																																						
DX-button numbers = BMS DX (just add 1 to calculate the Win DX button numbers)	Shifting offset value: 256	Controller number: 1																																				
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="font-size: x-small;">T7 (Dogfight SRM) DX: 24 TQS: DOGFIGHT Switch - DF Override</td> <td style="font-size: x-small;">T8 (Dogfight MRM) DX: 25 TQS: DOGFIGHT Switch - MRM Override</td> </tr> <tr> <td colspan="2" style="font-size: x-small;">Middle Position (Dogfight Cancel) Check falcon bms.cfg and change value: set g_bHotasDgftSelfCancel 1</td> </tr> <tr> <td style="font-size: x-small;">T9 (Speedbrakes Out) DX: 26 TQS: SPD BRAKE Switch - Open VIEWGEN: Look Closer - Toggle</td> <td style="font-size: x-small;">T10 (Speedbrakes In) DX: 27 TQS: SPD BRAKE Switch - Close</td> </tr> <tr> <td colspan="2" style="font-size: x-small;">T1 (Cursor Enable) DX: 18 TQS: RDR CURSOR - Cursor Enable TQS: RDR CURSOR - Cursor Zero</td> </tr> </table>	T7 (Dogfight SRM) DX: 24 TQS: DOGFIGHT Switch - DF Override	T8 (Dogfight MRM) DX: 25 TQS: DOGFIGHT Switch - MRM Override	Middle Position (Dogfight Cancel) Check falcon bms.cfg and change value: set g_bHotasDgftSelfCancel 1		T9 (Speedbrakes Out) DX: 26 TQS: SPD BRAKE Switch - Open VIEWGEN: Look Closer - Toggle	T10 (Speedbrakes In) DX: 27 TQS: SPD BRAKE Switch - Close	T1 (Cursor Enable) DX: 18 TQS: RDR CURSOR - Cursor Enable TQS: RDR CURSOR - Cursor Zero		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="font-size: x-small;">T6 (Uncage) DX: 23 TQS: MAN RANGE Knob - UNCAGE TQS: CUTOFF RELEASE - Idle Detent - Toggle</td> <td style="font-size: x-small;">T2 (Radio Switch Up) DX: 19 TQS: COMMS Switch Down - VHF TQS: COMMS Switch Down - VHF</td> <td style="font-size: x-small;">T4 (Radio Switch Right) DX: 21 TQS: COMMS Switch Right - IFF IN TQS: COMMS Switch Right - IFF IN</td> <td style="font-size: x-small;">T3 (Radio Switch Down) DX: 20 TQS: COMMS Switch Up - UHF TQS: COMMS Switch Up - UHF</td> <td style="font-size: x-small;">T5 (Radio Switch Left) DX: 22 TQS: COMMS Switch Left - IFF OUT TQS: COMMS Switch Left - IFF OUT</td> </tr> </table>	T6 (Uncage) DX: 23 TQS: MAN RANGE Knob - UNCAGE TQS: CUTOFF RELEASE - Idle Detent - Toggle	T2 (Radio Switch Up) DX: 19 TQS: COMMS Switch Down - VHF TQS: COMMS Switch Down - VHF	T4 (Radio Switch Right) DX: 21 TQS: COMMS Switch Right - IFF IN TQS: COMMS Switch Right - IFF IN	T3 (Radio Switch Down) DX: 20 TQS: COMMS Switch Up - UHF TQS: COMMS Switch Up - UHF	T5 (Radio Switch Left) DX: 22 TQS: COMMS Switch Left - IFF OUT TQS: COMMS Switch Left - IFF OUT	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="font-size: x-small;">S2 (Pickle) DX: 1 STICK: WEAPON RELEASE (Pickle) SIM: TRACKIR Recenter (additional)</td> <td style="font-size: x-small;">H2U (TMS Up) DX: 6 STICK: TMS Up CKPIT: Trim-Reset (Change here)</td> <td style="font-size: x-small;">H2R (TMS Right) DX: 7 STICK: TMS Right ICP: A-A Button - Push</td> <td style="font-size: x-small;">H2D (TMS Down) DX: 8 STICK: TMS Down ICP: A-G Button - Push</td> <td style="font-size: x-small;">H2L (TMS Left) DX: 9 STICK: TMS Left ICP: NAV Mode (no such button in F16)</td> </tr> <tr> <td style="font-size: x-small;">TG1 (Trigger 1) DX: 0 STICK: FIRST TRIGGER DETENT STICK: FIRST TRIGGER DETENT</td> <td style="font-size: x-small;">TG2 (Trigger 2) DX: 5 STICK: SECOND TRIGGER DETENT SEAT: EJECT Handle - Hold For Eject</td> <td style="font-size: x-small;">S4 (Paddle) DX: 3 STICK: PADDLE SWITCH STICK: PADDLE SWITCH</td> <td colspan="2" style="font-size: x-small;">Own Remarks: Shifted Layer for TG1 (Trigger 1): Toggle 2D / 3D Cockpit Shifted Layer for S4 (Paddle): Wheelbrakes (Hold)</td> </tr> </table>	S2 (Pickle) DX: 1 STICK: WEAPON RELEASE (Pickle) SIM: TRACKIR Recenter (additional)	H2U (TMS Up) DX: 6 STICK: TMS Up CKPIT: Trim-Reset (Change here)	H2R (TMS Right) DX: 7 STICK: TMS Right ICP: A-A Button - Push	H2D (TMS Down) DX: 8 STICK: TMS Down ICP: A-G Button - Push	H2L (TMS Left) DX: 9 STICK: TMS Left ICP: NAV Mode (no such button in F16)	TG1 (Trigger 1) DX: 0 STICK: FIRST TRIGGER DETENT STICK: FIRST TRIGGER DETENT	TG2 (Trigger 2) DX: 5 STICK: SECOND TRIGGER DETENT SEAT: EJECT Handle - Hold For Eject	S4 (Paddle) DX: 3 STICK: PADDLE SWITCH STICK: PADDLE SWITCH	Own Remarks: Shifted Layer for TG1 (Trigger 1): Toggle 2D / 3D Cockpit Shifted Layer for S4 (Paddle): Wheelbrakes (Hold)		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="font-size: x-small;">H1U (Up) POV STICK: TRIM Up - Nose Down VIEWGEN: Rotate View Up</td> <td style="font-size: x-small;">H1R (Right) POV STICK: TRIM Right - Roll Right VIEWGEN: Rotate View Right</td> <td style="font-size: x-small;">H1D (Down) POV STICK: TRIM Down - Nose Up VIEWGEN: Rotate View Down</td> <td style="font-size: x-small;">H1L (Left) POV STICK: TRIM Left - Roll Left VIEWGEN: Rotate View Left</td> </tr> <tr> <td style="font-size: x-small;">S1 (MslStep) DX: 4 STICK: NVIS AIR DISC MSL STEP SWITCH FUEL AIR REFUEL Switch - Toggle</td> <td style="font-size: x-small;">H3U (DMS Up) DX: 10 STICK: DMS Up CKPIT: Night/visn - Toggle</td> <td style="font-size: x-small;">H3R (DMS Right) DX: 11 STICK: DMS Right AVTR: AVTR Switch - Toggle ON / OFF</td> <td style="font-size: x-small;">H3D (DMS Down) DX: 12 STICK: DMS Down CKPIT: Visor - Toggle</td> </tr> <tr> <td style="font-size: x-small;">H4U (CMS Up) DX: 14 STICK: CMS Up GEAR: LG Handle - Toggle</td> <td style="font-size: x-small;">H4R (CMS Right) DX: 15 STICK: CMS Right GEAR: PARKING BREAK Switch - Toggle</td> <td style="font-size: x-small;">H4D (CMS Down) DX: 16 STICK: CMS Down GEAR: STORES CONFIG Switch - Toggle</td> <td style="font-size: x-small;">H4L (CMS Left) DX: 17 STICK: CMS Left</td> </tr> </table>	H1U (Up) POV STICK: TRIM Up - Nose Down VIEWGEN: Rotate View Up	H1R (Right) POV STICK: TRIM Right - Roll Right VIEWGEN: Rotate View Right	H1D (Down) POV STICK: TRIM Down - Nose Up VIEWGEN: Rotate View Down	H1L (Left) POV STICK: TRIM Left - Roll Left VIEWGEN: Rotate View Left	S1 (MslStep) DX: 4 STICK: NVIS AIR DISC MSL STEP SWITCH FUEL AIR REFUEL Switch - Toggle	H3U (DMS Up) DX: 10 STICK: DMS Up CKPIT: Night/visn - Toggle	H3R (DMS Right) DX: 11 STICK: DMS Right AVTR: AVTR Switch - Toggle ON / OFF	H3D (DMS Down) DX: 12 STICK: DMS Down CKPIT: Visor - Toggle	H4U (CMS Up) DX: 14 STICK: CMS Up GEAR: LG Handle - Toggle	H4R (CMS Right) DX: 15 STICK: CMS Right GEAR: PARKING BREAK Switch - Toggle	H4D (CMS Down) DX: 16 STICK: CMS Down GEAR: STORES CONFIG Switch - Toggle	H4L (CMS Left) DX: 17 STICK: CMS Left
T7 (Dogfight SRM) DX: 24 TQS: DOGFIGHT Switch - DF Override	T8 (Dogfight MRM) DX: 25 TQS: DOGFIGHT Switch - MRM Override																																					
Middle Position (Dogfight Cancel) Check falcon bms.cfg and change value: set g_bHotasDgftSelfCancel 1																																						
T9 (Speedbrakes Out) DX: 26 TQS: SPD BRAKE Switch - Open VIEWGEN: Look Closer - Toggle	T10 (Speedbrakes In) DX: 27 TQS: SPD BRAKE Switch - Close																																					
T1 (Cursor Enable) DX: 18 TQS: RDR CURSOR - Cursor Enable TQS: RDR CURSOR - Cursor Zero																																						
T6 (Uncage) DX: 23 TQS: MAN RANGE Knob - UNCAGE TQS: CUTOFF RELEASE - Idle Detent - Toggle	T2 (Radio Switch Up) DX: 19 TQS: COMMS Switch Down - VHF TQS: COMMS Switch Down - VHF	T4 (Radio Switch Right) DX: 21 TQS: COMMS Switch Right - IFF IN TQS: COMMS Switch Right - IFF IN	T3 (Radio Switch Down) DX: 20 TQS: COMMS Switch Up - UHF TQS: COMMS Switch Up - UHF	T5 (Radio Switch Left) DX: 22 TQS: COMMS Switch Left - IFF OUT TQS: COMMS Switch Left - IFF OUT																																		
S2 (Pickle) DX: 1 STICK: WEAPON RELEASE (Pickle) SIM: TRACKIR Recenter (additional)	H2U (TMS Up) DX: 6 STICK: TMS Up CKPIT: Trim-Reset (Change here)	H2R (TMS Right) DX: 7 STICK: TMS Right ICP: A-A Button - Push	H2D (TMS Down) DX: 8 STICK: TMS Down ICP: A-G Button - Push	H2L (TMS Left) DX: 9 STICK: TMS Left ICP: NAV Mode (no such button in F16)																																		
TG1 (Trigger 1) DX: 0 STICK: FIRST TRIGGER DETENT STICK: FIRST TRIGGER DETENT	TG2 (Trigger 2) DX: 5 STICK: SECOND TRIGGER DETENT SEAT: EJECT Handle - Hold For Eject	S4 (Paddle) DX: 3 STICK: PADDLE SWITCH STICK: PADDLE SWITCH	Own Remarks: Shifted Layer for TG1 (Trigger 1): Toggle 2D / 3D Cockpit Shifted Layer for S4 (Paddle): Wheelbrakes (Hold)																																			
H1U (Up) POV STICK: TRIM Up - Nose Down VIEWGEN: Rotate View Up	H1R (Right) POV STICK: TRIM Right - Roll Right VIEWGEN: Rotate View Right	H1D (Down) POV STICK: TRIM Down - Nose Up VIEWGEN: Rotate View Down	H1L (Left) POV STICK: TRIM Left - Roll Left VIEWGEN: Rotate View Left																																			
S1 (MslStep) DX: 4 STICK: NVIS AIR DISC MSL STEP SWITCH FUEL AIR REFUEL Switch - Toggle	H3U (DMS Up) DX: 10 STICK: DMS Up CKPIT: Night/visn - Toggle	H3R (DMS Right) DX: 11 STICK: DMS Right AVTR: AVTR Switch - Toggle ON / OFF	H3D (DMS Down) DX: 12 STICK: DMS Down CKPIT: Visor - Toggle																																			
H4U (CMS Up) DX: 14 STICK: CMS Up GEAR: LG Handle - Toggle	H4R (CMS Right) DX: 15 STICK: CMS Right GEAR: PARKING BREAK Switch - Toggle	H4D (CMS Down) DX: 16 STICK: CMS Down GEAR: STORES CONFIG Switch - Toggle	H4L (CMS Left) DX: 17 STICK: CMS Left																																			

2.2.2 Adapting the Falcon BMS.cfg to your stickfile

If you have followed the instructions above you are *almost* ready to go.

But nothing is ever simple in BMS. The Cougar DX file features a shifted DX layer which allows you to benefit from a double layer of programming for each button. The primary layer corresponds to the real F-16 controls usually and the second layer is used for sim-specific functions.

This feature is impossible to check in the UI SETUP > CONTROLLER pages; you must be in 3D to check the effect of the shifted layer. You would not be the first person to spend time in the UI trying to fix something that is not broken. It is suggested you read chapter 10.6.4 of this manual to understand how this feature works.

Finally, for the Cougar_DX setup to work, some config lines in the Falcon BMS.cfg located located in your `User\Config` folder needs to be verified or changed:

Find `set g_bHotasDgftSelfCancel`. It is set to 0 by default, change it to 1.

Verify `set g_nHotasPinkyShiftMagnitude` is set to 256 (which is its default value).





2.2.3 Axis declaration

The final step required is to setup your analogue axis in the UI setup. Indeed, buttons are only a part of a HOTAS declaration; we still have to declare all the different axes of your Cougar.

This is done in the UI SETUP page and explained in detail in BMS Manual Chapter 4.4.

There is no need to document this again here as the procedure is the same as explained in the BMS Manual. Basically, you define your stick axis (X and Y), your throttle axis (Z) and the two detents (idle and afterburner) and eventually a rudder as primary flight controls and then you set up each secondary control axis, such as Antenna Elevation, Cursor X and Y and the Range knob.

All these settings are saved in an `axismapping.dat` file saved in your `User\Config` folder. It is advised to backup this file once you are satisfied with the axis configuration. If the configuration changes in the future for reasons that you may not fully control, a restore of your backup will be the fastest way to get flying again.

Alternatively, you can simply set this file to read only in Windows Explorer to prevent any unwanted future changes. Just remember to change it back if you want to modify the axis configuration.

2.2.4 Conclusion

Here are the steps required to use one of the ready-made configurations:

- Locate the folder for your joystick (`\Docs\01 Input Devices\03 HOTAS Setup`).
- Copy the key file needed for that joystick configuration to the `\User\Config` folder.
- Activate that key file in the UI (SETUP > CONTROLLERS).
- Use the provided joystick files and load them up with your joystick software.
- Follow any device-specific instructions in the Readme or the Device Setup Guide.
- Set up each analogue axis.
- Print the graphical layout.
- Go flying and test your HOTAS for expected responses.

This is a brief example, but a more complete document is available to explain how to setup every HOTAS file included with BMS.

Please refer to The BMS Device Setup Guide located in your `\Docs\01 Input Devices\03 HOTAS Setup` folder.





2.3 Creating your own (Keystrokes)

For this chapter we will work on the assumption that you own a Cougar and want to program it with the real HOTAS programming of the F-16. We will use the default BMS – Full.key file.

To do that your must:

- Locate the callbacks associated with the stick and throttle.
- Ensure you use the correct .key file that uses the above callbacks.
- Activate that key file in the UI > SETUP > CONTROLLERS.
- Program your joystick buttons accordingly with the provided software.
- Test in the UI and go flying.

The first step seems to be easier said than done, but the BMS documentation provides all you need to locate these callbacks. For instance, the BMS Dash 1 document (TO-BMS1F-16CM-1.pdf) in your Docs folder lists the sidestick and throttle configurations according to master modes. All the callbacks you need to program your joystick are listed there.

The graphics may look complicated, but their main advantage is that the callbacks are identified in blue. The complicated thing comes from the fact that a HOTAS button may have multiple functions depending on which master mode you are in. But as a user you should not worry about that while programming your joystick, because all these multiple functions are managed by the code and are relatively transparent to you.

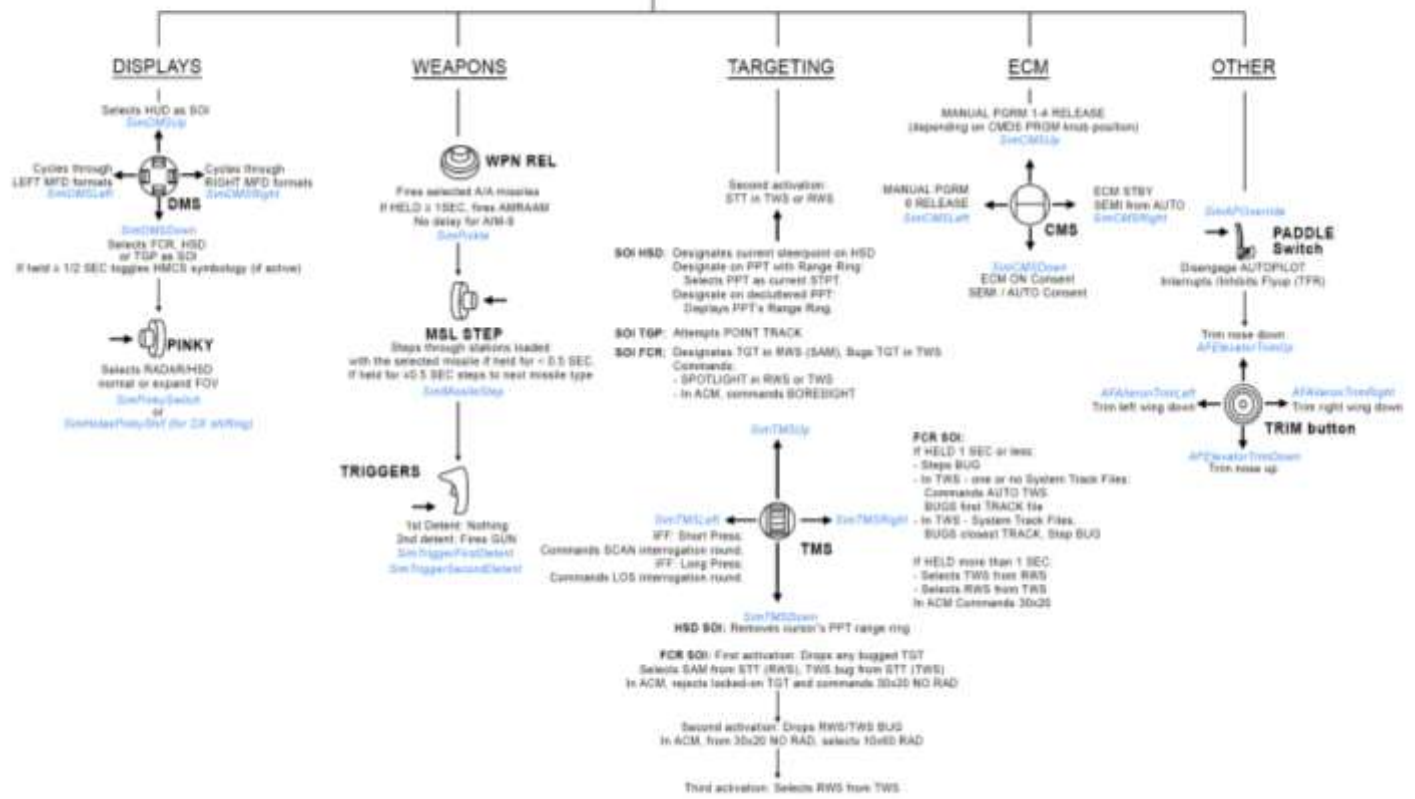
As an example, you will notice that the 4 TMS buttons on the next graphic are explained in detail, but basically there are only 4 relevant callbacks (up, right, down and left) and they are the same in both graphics (i.e. in all master modes); the first graphic dedicated to Air to Air and the second being dedicated to Air to Ground mode.

So basically, programming the TMS button for 4 directions comes down to programming only 4 single callbacks: TMS up, TMS right, TMS down and TMS left.

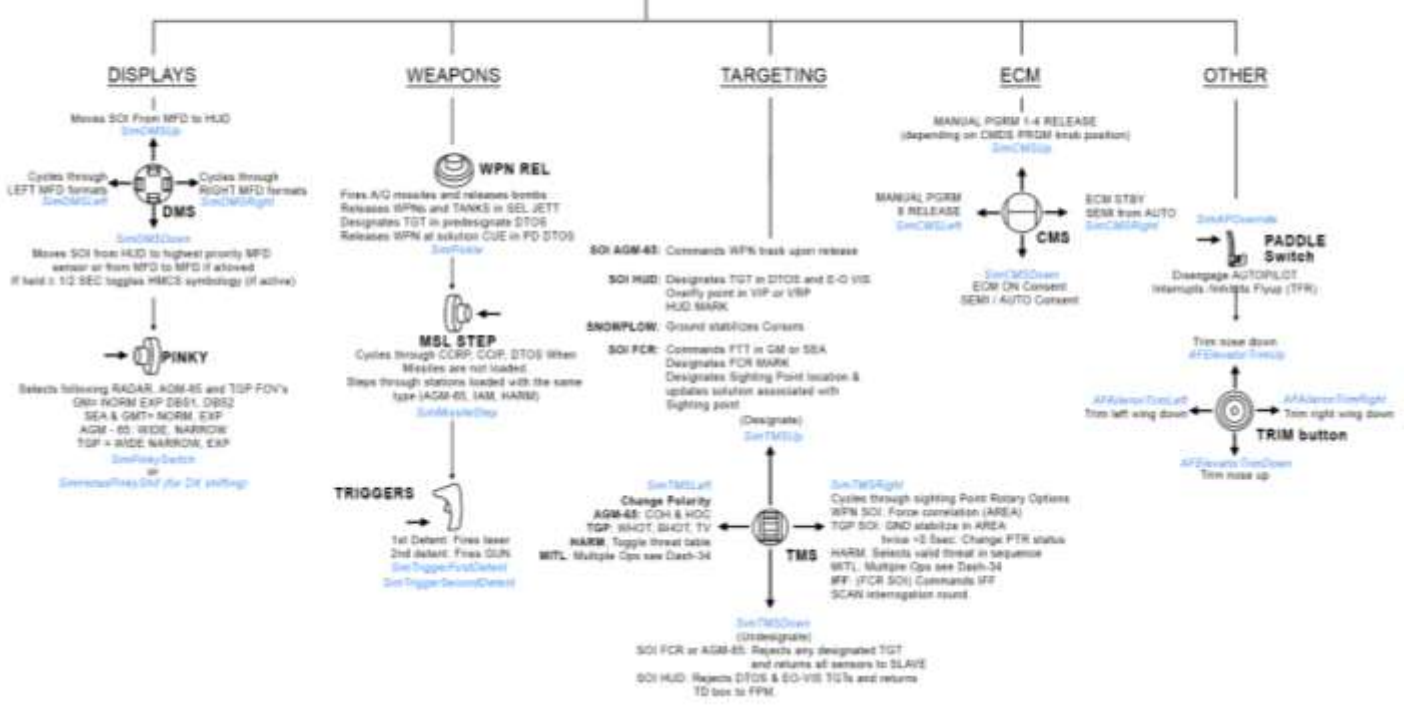




HANDS- ON CONTROLS AIR TO AIR MISSION SIDE STICK CONTROLLER A-A, MSL OVRD, DGFT



HANDS- ON CONTROLS AIR TO GROUND MISSION SIDE STICK CONTROLLER A-G (& NAV) MASTERMODE



From reading the above 2 graphics, stick programming basically requires the callbacks shown below.





And by looking at the BMS full.key file you can easily find the corresponding keystrokes.

SimPickle = *Space*

SimMissileStep = *SHF =*

AFElevatorTrimUp = *CTL Up Arrow*

AFElevatorTrimLeft = *CTL Left Arrow*

AFElevatorTrimRight = *CTL Right Arrow*

AFElevatorTrimDown = *CTL Down Arrow*

SimTMSUp = *SHF Home*

SimTMSLeft = *SHF Delete*

SimTMSRight = *SHF PageDown*

SimTMSDown = *SHF End*

SimDMSUp = *CTL Home*

SimDMSLeft = *CTL Delete*

SimDMSRight = *CTL PageDown*

SimDMSDown = *CTL End*

SimTriggerFirstDetent = *CTL =*

SimTriggerSecondDetent = *ALT =*

SimCMSUp = *ALT Home*

SimCMSLeft = *ALT Delete*

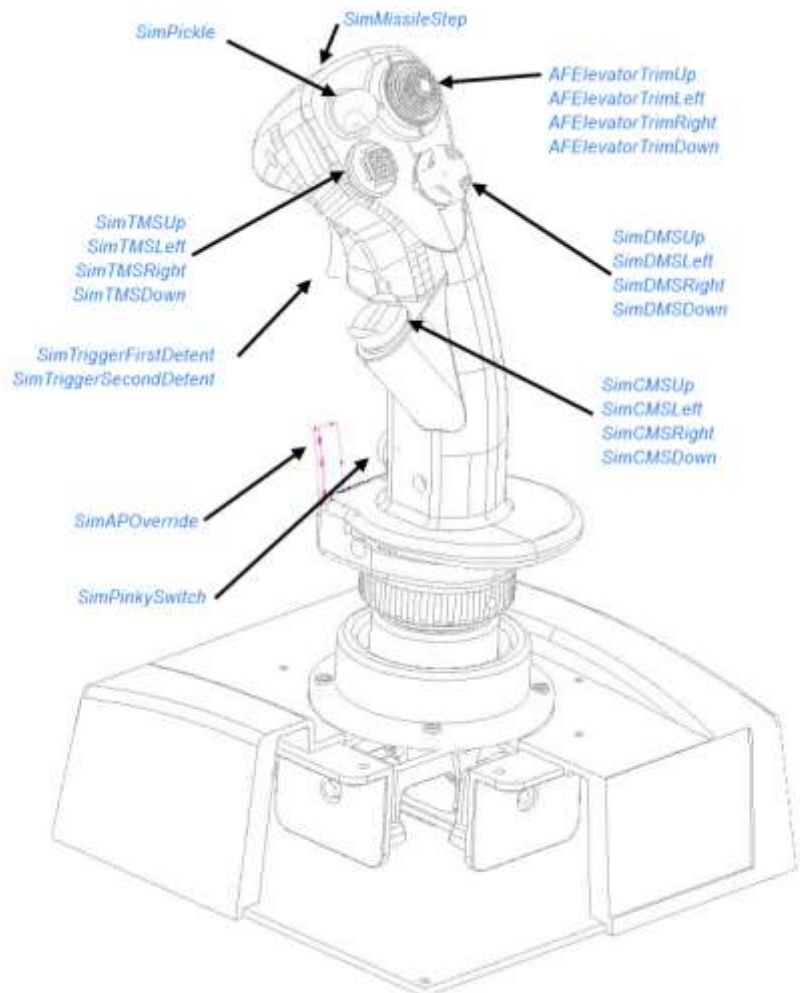
SimCMSRight = *ALT PageDown*

SimCMSDown = *ALT End*

SimPinkySwitch = *v*

SimAPOverride = *ALT q*

CURRENT KEYFILE: BMS - FULL	
KEY	MAPPING
===== 5.11 FLIGHT STICK =====	
Shift Home	STICK: TMS Up
Shift End	STICK: TMS Down
Shift Delete	STICK: TMS Left
Shift PgDn	STICK: TMS Right
Ctrl Home	STICK: DMS Up
Ctrl End	STICK: DMS Down
Ctrl Delete	STICK: DMS Left
Ctrl PgDn	STICK: DMS Right
Alt Home	STICK: CMS Up
Alt End	STICK: CMS Down
Alt Delete	STICK: CMS Left
Alt PgDn	STICK: CMS Right
Ctrl Up Arrow	STICK: TRIM Up - Nose Down
Ctrl Dn Arrow	STICK: TRIM Down - Nose Up
Ctrl Lt Arrow	STICK: TRIM Left - Roll Left
Ctrl Rt Arrow	STICK: TRIM Right - Roll Right
No Key Assigned	REM: Trim-Reset (change @ CKPIT)





Making the links as illustrated on the previous page is the hardest step when making your own joystick file. Once you have linked joystick buttons to callbacks with the relevant documentation and then linked callbacks to keystrokes through the active keyfile, all there is left to do is to open your joystick brand native software and program the keystrokes to the relevant buttons.

With a Cougar, that would be done with Foxy, creating a TMM and a TMJ file. The TMM will be the macro file defining the relevant keystroke and the TMJ will be the actual button definition file.

For the TMS example it would go like the following pictures (yellow from the TMJ and Blue from the TMM). The /H is internal Thrustmaster syntax to ensure the functions are Held

```
rem H2 is TMS (target management switch)
BTN H2U /H TMS_up
BTN H2D /H TMS_down
BTN H2L /H TMS_left
BTN H2R /H TMS_right
```

```
TMS_up = CTL UARROW
TMS_down = CTL DARROW
TMS_right = CTL RARROW
TMS_left = CTL LARROW
```

Obviously we can't document all joystick brand method for programming; you will have to refer to your HOTAS documentation.

The same method can be used to program the throttle. Using the blue callbacks identified in the Dash-1 graphics relevant to AA and AG mastermodes throttle functions and reading the BMS-full.key file you can associate callbacks to keystroke and where to program them.

The advantage of this method is that you can easily use the advanced programming features of your HOTAS to program more keystrokes to the same buttons. That is the shifted layer we referred to earlier. In TM syntax it is done by using /I /O layers. Each button programmed will have one function (/O) when used without the S3 button and another function (/I) when pressed simultaneously with the Cougar S3 button.

It allows the programming of SIM specific features on the S3in layer and realistic programming on the S3out layer. Sim specific functions have either no equivalent in real life (trackball, screenshots, pause) or are activated through controls we do not have in the sim (helmet visor, NVGs, etc) and since all the first layer buttons are taken up by the real-world programming, having that second layer is very useful.

```
rem H3 is DMS (display management switch)
BTN H3U //TIR_center
//O /H DMS_up
BTN H3D //^
//O /H DMS_down
BTN H3L //A IVCvsAI_down
//O /H DMS_left
BTN H3R //A IVCvsAI_up
//O /H DMS_right
```

The advantage of this method is that it is native to your HOTAS software and should be relatively easy to setup. The second advantage is that you can test the HOTAS response on the two layers (something you cannot do when you use the DX shifting layer) in the UI; you don't have to go into 3D.

The disadvantage of this method (as you are using only keystrokes) is that some joystick functions may get interrupted if you press other keystrokes simultaneously, especially when using SHF CTL and ALT key combinations.

Imagine if you press SHF CTL A and while you keep this depressed you need to press SHF b. When you release SHF b, you will actually release the SHF in the SHF CTL A combination as well. You will either induce stuck keys, or some callbacks might not get invoked.





2.4 Creating your own (DirectX)

It is even simpler but it's more limited!

This may be the ideal solution for newcomers and first-time BMS users.

With your stick not programmed at all, but declared in the BMS UI you open the UI Setup Controllers page and click on the key relevant to the function you are about to program on your Stick using DirectX:

The keystroke becomes Cyan (SHF Home) in this example as we are programming the TMS up function.



Double check that the keystroke is selected (Cyan) and hit the TMS up button on your joystick:



Nothing seems to have changed but look at the bottom of the window, the Input Button 7 is now assigned to STICK: TMS Up function.

You may also do that with your native Hotas software, but you have to ensure that the DX entries correspond (DX1 is indeed button 1 in this case) between the joystick file declaration and the UI button numbers.

BTN S2 // screenshot
/O /H DX_pickle

DX_pickle = DX1
DX_Firsttriggerdetent = DX2
DX_APOverride = DX3
DX_Transmit_UHF = DX4
DX_Transmit_VHF = DX5





The above example allows you to make a mix of DX and keystrokes in your own Joystick file programming.

The advantage of this method is obvious. It is very easy to do as it is simply point and click. The disadvantage is that you can not profit from double layer button programming; it is one button, one function only.

Programming more advanced DirectX functions is possible though and explained in chapter 10 of this manual.

2.5 Conclusion

If you are a new user, you can choose to not use any stick file at all. Simply declare your analogue axis in the UI SETUP > CONTROLLER > ADVANCED. Then declare single layered DirectX bindings for each button as explained in chapter 2.4 above. This is the simplest, hassle free, quickest way to get flying.

After a while you will probably want to create a more complete setup and chances are that you will consider double layered buttons. That is usually where all the fun starts.

You can simply use one of the provided setups for your HOTAS and follow the instructions in the BMS Device Setup Guide document located in your `\Docs\01 Input Devices\03 HOTAS Setup` folder. This document explains how to set up all the stick files provided with BMS as illustrated in chapter 2.2 above.

If you choose to build your own custom setup, or there is not a ready-made solution for your HOTAS the next big question is to go full DX, or use a callback file, or a combination of the two. Many try full DX first and realise it's too complicated, too touchy and too unstable to be easily manageable:

- The different numbering between Windows (1-32) and BMS (0-31) does not make it intuitive.
- The need for the DeviceSorting.txt because of the instability of DirectX devices numbering does not make it intuitive.
- The DirectX shifting capability is a great feature but far from intuitive. Many users end up with stuck keys. The fact that most of these problems are user errors proves the point that it is not intuitive.

Nevertheless, the manuals (see the BMS Device Setup Guide document and chapter 10 of this manual) try to address these issues but we know how hard it is for the users to find the right bit of information in such a huge manual suite.

That is where the next solution comes into play: creating your own stick file with a combination of DirectX and keystroke bindings. That is explained in chapter 2.3.

Usually once you reach that point you reached stability and you will never turn back.

Bottom line: Your mileage may vary and what is perfect for one might be a pain for another one.

Hopefully this chapter will help you decide what is best suited for you and will have given you the necessary details to chain all the links together, so you understand what is required to set up your controllers in BMS.





3. Avionics Configurator

Since 4.33 users can customise the avionics of each F-16 block present in BMS.

By default the models in BMS should be as realistic as possible at launch but reality and simulation do not evolve at the same time and real-life conflicts always accelerate avionic and aircraft capabilities.

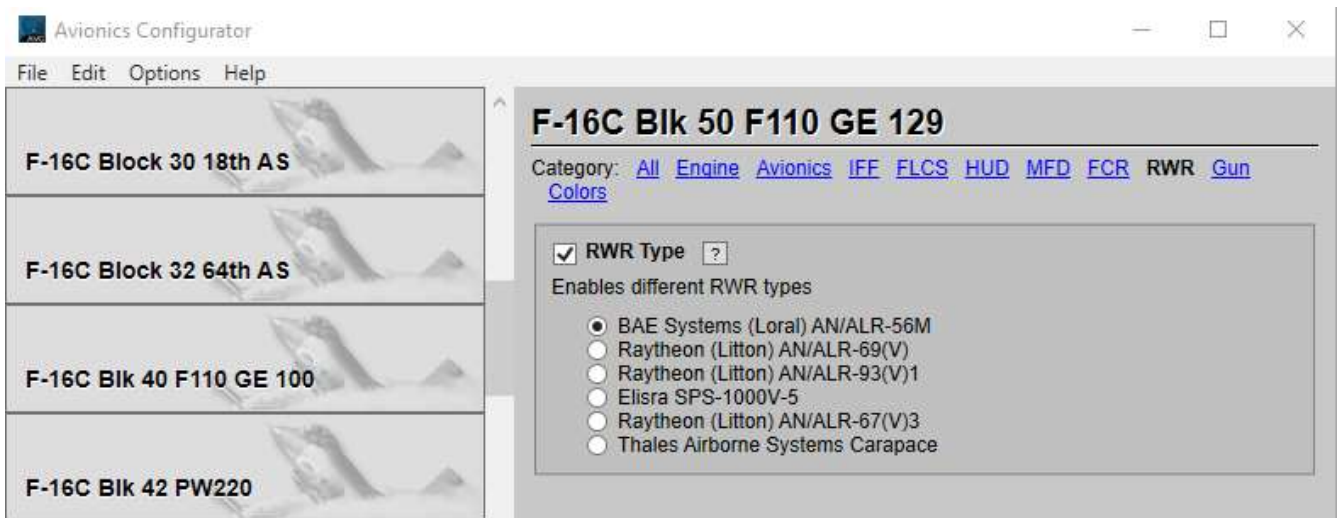
To partly address this issue, BMS Avionics Configurator allows the user to change some avionics of the F-16 blocks in the database.

Another reason for using this tool is that the same F-16 block can evolve differently according to which country is using it. Each country may pay to update their F-16 with slightly different avionics systems, so a user wanting to fly a country-specific aircraft can customise the relevant model to match the real capabilities of that aircraft.



Avionics Configurator is launched from the BMS Launcher as shown above.

The program will display a list of all available F-16 blocks in the BMS database on the left and a list of avionic options on the right. The options on the right can be filtered by category to shorten the list. The options are the same for all blocks, the difference being some options are activated or not depending on the F-16 model.



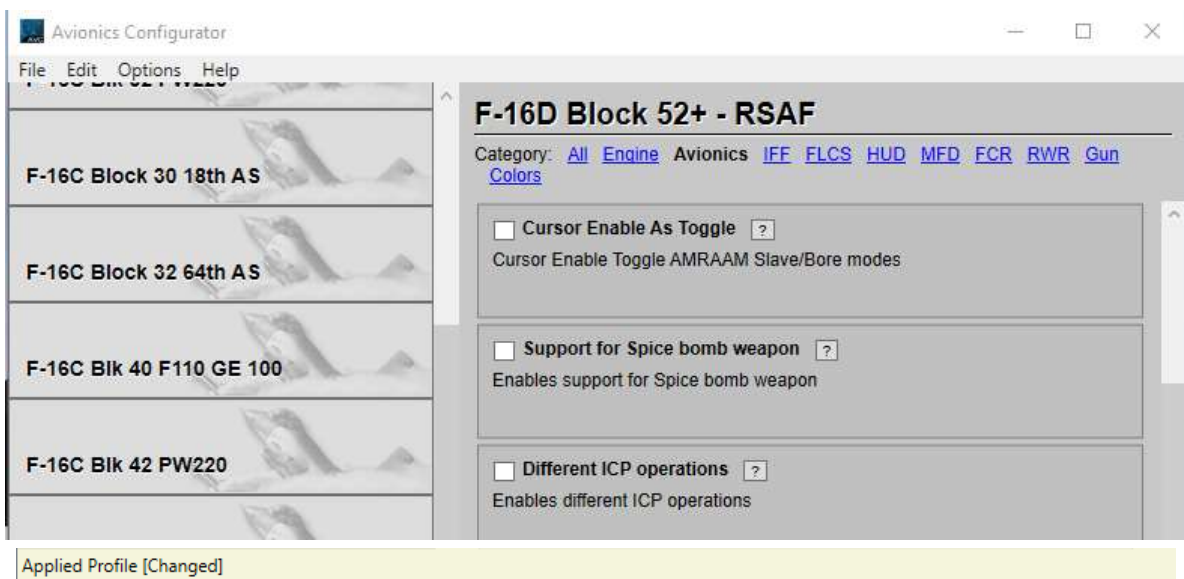


3.1 Examples of modifications

To change an option in a specific block you first select the relevant F-16 model.

Let's say the RSAF (Republic of Singapore Air Force) upgraded their F-16 block 52+ with SPICE bombs, which they bought from Israel. The stock BMS model cannot carry these weapons; we will address this situation with Avionics Configurator.

On the left list of F-16 blocks, scroll down until you find the **F-16D Block 52+ - RSAF** and click on it. Please note once you select it, the right side of the window will identify the list of avionics currently selected for the RSAF F-16 but on the left side **F-16D Block 52+ - RSAF** is no longer highlighted after you click anywhere on the right pane. So always refer to the title on the right side to know which version you are changing options for.



Scroll down the right pane until you find the relevant **Support for Spice bomb weapon** option (or click on the **Avionics** category to display a shorter list). Click on the radio button to activate this option; the yellow status line at the bottom of the window shows that the current profile has been modified [Changed].

The RSAF F-16D settings have been changed but the profile still needs to be applied for the changes to take effect. Open the File menu and select the option **Apply Profile**.



In addition to the small window popping up to tell you that the profile has been applied successfully, the status line changed again to identify that the profile is applied with the options selected.





The Apply Profile option saves your configuration in your own BMS installation. You do not need to save your profile with a different name to be able to reactivate it next time. Nevertheless, if you want to share specific profiles with other people, you may use the “Save Profile As”. This will save a Profile.Acfg file which you can then send to your VFW members, so you all fly with the same settings.

Launch BMS and select the F-16D block 52+ RSAF for flight, you will notice that the avionics of your jet have been updated and that you have now access to the SPICE page of the DED (LIST – MISC – E) as illustrated on the right DED picture. The left DED picture below is the default one before the avionics configuration changes.



But you may have realised that there is no SPICE weapon available from the loadout menu of that F-16... That is unfortunate, but the Avionics Configurator manages only the avionics side of things and not the type of weapon the F-16 can actually carry... So, you have to enable this manually in the aircraft.dat file to add the Spice bombs to the aircraft loadout menu.

Luckily in a majority of aspects, Avionics Configurator is a stand-alone tool. Let’s take another example and activate the older GUN modes for our RSAF aircraft. As you know newer blocks only have the EEGS gun mode and you cannot select the older gun modes from the SMS page. Let’s change that.

With the RSAF F-16 selected, click on the Gun category option and enable all 4 checkboxes to activate the 4 different gunsights. Apply the profile from the File menu and take the RSAF bird for a spin.





To see the changes, select Dogfight mode once in flight and OSB 2 will now allow you to select SSLC, LCOS and SNAP mode on top of the default EEGS mode.



As you may have noticed, the colours of the MFDs have changed to yellow in 4.34 for the block 50 and 52. If you prefer to switch back to the white colour used in 4.33 you can do that with Avionics Configurator as well.

Select the F-16C Blk 50 F110 GE 129 from the left side of avionics configurator and scroll down at the end of the list of avionics options on the right side of the window until you find the Color Configuration, or just click on the Colors Category.



Changing the colours of the MFD from yellow to white isn't very time consuming. Simply extend the menu on the first item: MFD_DEFAULT which is currently set to YELLOW and select MFD_WHITE. Alternatively, select DEFAULT. The background colour of DEFAULT shows the default colour that will be applied, in this case white.

The only options left specifically set to yellow are the two SOI settings. Switch these back to white (or DEFAULT) as well: MFD_SOI_BOX and MFD_NOT_SOI.






3.2 Options explanation

Most of the options are self-explanatory. In most cases, clicking on the radio button with an interrogation mark inside will open a new window with pictures and text explaining the purpose of the option.


Hud Ladder Tape Type ?
Enables different HUD ladder tape types

More info on Hud Ladder Tape Type ...



Enables different HUD ladder tape types.

More info on Cursor Enable As Toggle ...



Cursor enable switch act as a toggle (press and release) for switching between AMRAAM Slave/Bore modes rather than as a must-held (deadman) switch for selecting Bore mode.

Close





4. List of Keyboard Layouts

4.1 Default US (QWERTY) Keyboard layout

F1 ICP COM1 Fwd Chnl Deflt Radar Range Up	F2 ICP COM2 Fwd Chnl Deflt Radar Range Up	F3 ICP #4 Aircrafts Hnd RCS Return Deflt Aircrafts Hnd	F4 ICP LBT Probe Heat Up HUD Range Deflt Main Radar Up	F5 ICP A-4 Probe Heat On Aircrafts Hnd	F6 ICP A-6 RFD DFM Switch Aircrafts Hnd	F7 TR Precision FLCZ PWR Test Aircrafts Hnd	F8 TR Profile Aircrafts Hnd	F9 TR Pause Probe Heat On Aircrafts Hnd	F10 FRAPS Screen Probe Heat On Aircrafts Hnd	F11 FRAPS Benchm Probe Heat On Aircrafts Hnd	F12 TR Recorder Probe Heat On Aircrafts Hnd
---	---	---	---	--	---	---	--	---	--	--	---

Falcon BMS Keyboard Layout (QWERTY / US Int.)

Swallow Cam System Cam Pilot/Target Landing Gear	1 Head Only Aircrafts Hnd	2 Snap F8 (3D) Fwd Chnl Up Aircrafts Hnd	3 Snap F9 (3D) Fwd Chnl Up Aircrafts Hnd	4 Falcon Mode AA Engine Display Falcon Mode AG	5 Extended FOV 3D Chnl Up Aircrafts Hnd	6 Tgt to Self Cam Target Cam Tgt Custom View	7 Zooming Cam Probe Heat On Tgt to Wpr Cam	8 Friendly AC Cam Friendly AC Cam Empty AC Cam	9 Chase Cam Probe Heat On Top Gun Cam	0 Orbit Cam Orbit Cam Eject Free Cam	Backspace FALCON ICP Hnd / Mode Cycle Display Reset AG Weapon
Tab Time Accel In Time Accel Up	Q Aircrafts Hnd Fwd Chnl Up Aircrafts Hnd	W Weapon Menu Weapon Menu Weapon Menu	E Element Menu Safety Level Up Aircrafts Hnd	R Fight Menu Aircrafts Hnd Aircrafts Hnd	Y ATC Menu Aircrafts Hnd Aircrafts Hnd	U Tanker Menu Aircrafts Hnd Aircrafts Hnd	I Master Mode Cvc Aircrafts Hnd Aircrafts Hnd	O MPO Tool Aircrafts Hnd Aircrafts Hnd	P Toggle Pause Aircrafts Hnd Aircrafts Hnd	T FRACS Bright Defc Aircrafts Hnd Aircrafts Hnd	I HMCS Bright Inc Aircrafts Hnd Aircrafts Hnd
Caps Lock Time Accel In Time Accel Up	A AP Push Cap Aircrafts Hnd Aircrafts Hnd	S Snap Switch Aircrafts Hnd Aircrafts Hnd	D DP Overlay Aircrafts Hnd Aircrafts Hnd	G AVTR Sw Tag Aircrafts Hnd Aircrafts Hnd	H Gear Toggle Aircrafts Hnd Aircrafts Hnd	J HUD Status Cap Aircrafts Hnd Aircrafts Hnd	K Wheel Brake Aircrafts Hnd Aircrafts Hnd	L HUD Close Aircrafts Hnd Aircrafts Hnd	HSI Hdg Defc 1° Aircrafts Hnd Aircrafts Hnd	Return Master Caution ICP Hnd / Mode Caution Reset Time Reset	
Shift	Z Prev. Wapoint Aircrafts Hnd Aircrafts Hnd	X Next Wapoint Aircrafts Hnd Aircrafts Hnd	C MONEY Concol Aircrafts Hnd Aircrafts Hnd	V Pinky Switch Aircrafts Hnd Aircrafts Hnd	B Socbin. Tag Aircrafts Hnd Aircrafts Hnd	N HUD Toggle Aircrafts Hnd Aircrafts Hnd	M ARM Overlay Aircrafts Hnd Aircrafts Hnd	Slider Left Aircrafts Hnd Aircrafts Hnd	W/A AM MSL Aircrafts Hnd Aircrafts Hnd	Shift	
Ctrl	Keypress Modifier: Unmodified Ctrl Alt SW + Ctrl SW + Alt Ctrl + Alt SW + Ctrl + Alt	Legend: BLACK BOLD GREEN BOLD BLUE BOLD RED BOLD BLACK ITALIC GREEN ITALIC BLUE ITALIC RED ITALIC	Space WPR Release	AltGr	Ctrl						

Print Screenshot Print/Screen	Scroll Lock TS PTT	UI Functions: IVC Broadcast F1 IVC UI Comms F2 UI Exit / Abort Esc UI Screenshot Print	3rd Party Software: TR Precision F7 TR Profile F8 TR Pause F9 TR Recorder F12	TS PTT TS Broadcast TS Toggle Mike TS Tog Speaker	Scroll Lock Num * Sft Num * Ctrl Num *	FRAPS Video F8 FRAPS Screen F10 FRAPS Benchm. F11 FRAPS Overlay F12
--	------------------------------	---	--	--	---	--

Num Block Layout

Insert ICP CUR Deflt ICP FLR Lvl Up ICP Broadcst Defc	Home Cursor Sw Defc TMS Up DMS Up TMS Up	Page Up ICP CUR Defc ICP FLR Lvl Up ICP Broadcst Defc	Num Lock ICP DCR 00 Tag ICP FLR WS	Num 1 ICP 1-4-4-4 Aircrafts Hnd	Num 2 ICP 1-4-4-4 Aircrafts Hnd	Num 3 ICP 1-4-4-4 Aircrafts Hnd	Num 4 ICP 1-4-4-4 Aircrafts Hnd	Num 5 ICP 1-4-4-4 Aircrafts Hnd	Num 6 ICP 1-4-4-4 Aircrafts Hnd	Num 7 ICP 1-4-4-4 Aircrafts Hnd	Num 8 ICP 1-4-4-4 Aircrafts Hnd	Num 9 ICP 1-4-4-4 Aircrafts Hnd	Num 0 ICP 1-4-4-4 Aircrafts Hnd	Num Enter ICP 1-4-4-4 Aircrafts Hnd
---	---	---	---	--	--	--	--	--	--	--	--	--	--	--

Key Combo / Notes

Key Combo set to Alt C

- L - Load Ckpit Deflts
- P - Pilot Model
- D - Score Display
- H - HUD Rendering
- R - Random Error
- T - Reload TrackIR
- F4 - Dev Debug
- F6 - Dev Scale Down
- FB - Dev Regen
- S - Save Ckpit Deflts
- F - Frame Rate
- Q - Online Status
- O - Camp Quick Save
- J - Recenter Joystick
- F1 - Dev Location
- F5 - Dev Srt Scale
- F7 - Dev Scale Up
- FB - Dev Postion

Note: Only the first 20 Key Combos are shown here (in order of appearance).

Own Remarks:





4.2 Pitbuilder US (QWERTY) Keyboard layout

Esc Eject Disc	F1 FRAPS Default ECF COM1 MFD Map Enable MFD Map Select Probe Near On Probe Near Off Probe Near Test	F2 Day City Switch ECF COM2 TWA Low Top MFD Search Top Star Maps Home Global SW Backup Digital Switch Off MFD Home Select	F3 Bright/Less Hold ECF RT TWA Power On TWA Power Off Star Maps Home MMT 7F Map On MMT 7F Map Off MFD Home Lock	F4 ECF GEN Switch ECF LIST	F5 TCR PWR1 Test ECF A/A MDC5 Weight On MDC5 Weight Off	F6 TCR Switch MDC5 Knob On MDC5 Knob Off Trim View Right Trim Mode On Trim Mode Off Trim Mode On Trim Mode Off	F7 TR Precision ECF NAV Mode MFD Nav On MFD Nav Off Trim Mode On Trim Mode Off Trim Mode On Trim Mode Off	F8 TR Profile MFD Nav On MFD Nav Off Trim Mode On Trim Mode Off Trim Mode On Trim Mode Off	F9 TR Passes MFD Nav On MFD Nav Off Trim Mode On Trim Mode Off Trim Mode On Trim Mode Off	F10 FRAPS Screen MFD Nav On MFD Nav Off Trim Mode On Trim Mode Off Trim Mode On Trim Mode Off	F11 FRAPS Benchmark MFD Nav On MFD Nav Off Trim Mode On Trim Mode Off Trim Mode On Trim Mode Off	F12 TR Recenter MFD Nav On MFD Nav Off Trim Mode On Trim Mode Off Trim Mode On Trim Mode Off
--------------------------	--	--	--	---	--	---	--	--	---	---	--	--

Falcon BMS Keyboard Layout (QWERTY / US Int.)

Open Chnl Box Water Drain In Water Drain Out	Hot Only MFD CDSR 1 MFD CDSR 2	Snap Pin (SD) MFD CDSR 2 MFD CDSR 2	Par Pin (SD) MFD CDSR 3 MFD CDSR 3	Passbook MFD CDSR 4 MFD CDSR 4	MFD CDSR 5 MFD CDSR 5	MFD CDSR 6 MFD CDSR 6	Satellite Cam MFD CDSR 7 MFD CDSR 7	Hydro Cam MFD CDSR 8 MFD CDSR 8	Class Cam MFD CDSR 9 MFD CDSR 9	Drift Cam MFD CDSR 10 MFD CDSR 10	MFD N1 Disc MFD N1 Disc	MFD N1 In MFD N1 In	Backspace EJECT ECF Del 20 On ECF Del 20 Off		
Monitor On Left Channel Up Left Channel Down Channel X	Clear Channel On Right Channel Up Right Channel Down Channel Y	Channel V Channel V Channel V	Avi Call Ltr Off Position Lo. Wash Position Ltr Only WingPin Ltr Off	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On	WingPin Ltr On		
Tab Time Accel Up Time Accel Down	Access Menu MFD Sw On MFD Sw Off	Workman Menu PRGM Knob 1 PRGM Knob 2	Element Menu PRGM Knob 3 PRGM Knob 4	Flight Menu CMDS Mouse 4 CMDS Mouse 4	ATC Menu CMDS Mouse 5 CMDS Mouse 5	Tanker Menu CMDS Mouse 6 CMDS Mouse 6	CMOS Mode Swap CMOS Mode Auto CMOS Mode Swap	Hook On CMOS Mode Swap CMOS Mode Swap	MFD Hold CMOS Mode Swap CMOS Mode Swap	Toaster Phase CMOS Mode Swap CMOS Mode Swap	Push/In Avail/Stat CMOS Mode Swap CMOS Mode Swap	Clear Up Clear Down Clear Up Clear Down	ECF Del 20 Norm ECF Del 20		
Caps Lock Time Accel Up Time Accel Down	Public Switch MFD Sw On MFD Sw Off	Light Sw Off Light Sw Test	R/S Sw Norm R/S Sw Test	AUTR Sw Test R/S Sw Norm R/S Sw Test	ECF Headlight On R/S Sw Norm R/S Sw Test	ECF Headlight Off R/S Sw Norm R/S Sw Test	ECF Headlight On R/S Sw Norm R/S Sw Test	ECF Headlight Off R/S Sw Norm R/S Sw Test	ECF Headlight On R/S Sw Norm R/S Sw Test	ECF Headlight Off R/S Sw Norm R/S Sw Test	ECF Headlight On R/S Sw Norm R/S Sw Test	ECF Headlight Off R/S Sw Norm R/S Sw Test	Return ECF Del 20 Norm ECF Del 20		
Shift	DMS Up DMS Down DMS Up DMS Down	DMS Down DMS Up DMS Down DMS Up	DMS Left DMS Right DMS Left DMS Right	DMS Right DMS Left DMS Right DMS Left	DMS Left DMS Right DMS Left DMS Right	DMS Right DMS Left DMS Right DMS Left	DMS Left DMS Right DMS Left DMS Right	DMS Right DMS Left DMS Right DMS Left	DMS Left DMS Right DMS Left DMS Right	DMS Right DMS Left DMS Right DMS Left	DMS Left DMS Right DMS Left DMS Right	DMS Right DMS Left DMS Right DMS Left	Shift		
Ctrl	Alt	Keypress Modifier: Unmodified Cn Alt SMB + Cn Cn + L-Alt Cn + Cn + Alt										AltGr	Ctrl		
Legend: BLACK BOLD GREEN BOLD BLUE BOLD RED BOLD													Space WPN Release		

Print Screenshot	ScrL Lock TS PTT	UI Functions: NVC Broadcast NVC UI Connect UI Exit / Abort UI Screenshot / Print	3rd Party Software: TR Precision F7 TR Profile F8 TR Passes F9 TR Recenter F12	TS PTT TS Broadcast TS Toggle Mics TS Tug Speaker	Soft Lock Num * Strt Num * Ctrl Num *	FRAPS Video FRAPS Screen FRAPS Benchm. FRAPS Overlay	F9 F10 F11 F12
----------------------------	----------------------------	---	---	--	--	---	-------------------------

Num Block Layout

Insert Numeric Clear MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Home Numeric Home ECF Deep Map Up MFD Home Inc 1 MFD Home Dec 1 MFD Home Inc 2 MFD Home Dec 2 MFD Home Inc 3 MFD Home Dec 3 MFD Home Inc 4 MFD Home Dec 4 MFD Home Inc 5 MFD Home Dec 5	Page Up Numeric Home ECF Deep Map Up MFD Home Inc 1 MFD Home Dec 1 MFD Home Inc 2 MFD Home Dec 2 MFD Home Inc 3 MFD Home Dec 3 MFD Home Inc 4 MFD Home Dec 4 MFD Home Inc 5 MFD Home Dec 5	Num Lock MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num / MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num * MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num - MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off
Delete Cursor PTT On Cursor PTT Off MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	End Cursor Sw On Cursor Sw Off MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Page Dn Cursor Sw On Cursor Sw Off MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 7 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 8 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 9 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num + MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off
Num 4 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 5 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 6 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 1 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 2 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 3 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num Enter MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off
Num 0 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 0 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 0 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 0 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 0 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 0 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off	Num 0 MFD PWR LV1 On MFD PWR LV1 Off MFD PWR LV2 On MFD PWR LV2 Off MFD PWR LV3 On MFD PWR LV3 Off MFD PWR LV4 On MFD PWR LV4 Off MFD PWR LV5 On MFD PWR LV5 Off

Key Combo / Notes

Key Combo not set.

Note: Only the first 20 Key Combos are shown here (in order of appearance).

Own Remarks:





4.3 Default French (AZERTY) Keyboard layout

Esc Esc Del	F1 ICP COM1 Pilot/Comm Data Master Navace On	F2 ICP COM2 Dev City Switch Master Navace Up	F3 ICP WFF Method In Hold HUD Navace Data Main Navace On	F4 ICP LIST Probe Head Up HUD Navace Hold Main Navace Up	F5 ICP A/A Probe Head On HUD On	F6 ICP A/G CPU GEN Switch AntiDev Center	F7 TIR Precision FLCS FWR Test Arbiter Up	F8 TIR Profile	F9 TIR Pause Flaps Full Auto Hold	F10 FRAPS Screen Flaps Full Auto Hold	F11 FRAPS Benchm F-18 Flap Down Flaps Full	F12 TIR Recorder F-18 Flap Up Flaps Full
-----------------------	--	--	---	---	---	--	---	--------------------------	---	---	--	--

Falcon BMS Keyboard Layout (AZERTY / French)

0 Satellite Cam Active Cam Photo/Filmng Auto Hold	1 Heat Only Inhibit Subtle Fuel	2 Snap FN (SD) Scrny Grab Start Subtle Chakra FN Make	3 Pan FN (SD) Inhibit Tap Display Power Mode Tag	4 Parlock Parlock Mode AA Engine Display Parlock Mode All	5 Extended PDV EFDV Mode AA EFDV Mode All	6 Tgt to Self Cam Turner Calc Tag Custom View Tag Custom View	7 Incoming Cam Weapon Calc Tgt to Wpn Cam Tag Custom View	8 Friendly AC Cam Friendly AC Cam Enemy AC Cam Enemy AC Cam	9 Chase Cam Auto Calc Topology Cam Topology Plot	0 Orbit Cam Elevation Cam Eq'y Free Cam Topology Plot	1 Parlock Prev Parlock Prev AA Parlock Prev AG None Annot	2 Parlock Next Parlock Next AA Parlock Next AG None Annot	Backspace F ACK ICP Hold Mode Cycle Engines Reset All Weapon	
Home Master Dash On Master Dash Off	IMPD OSE 1 IMPD OSE 1	IMPD OSE 2 IMPD OSE 2	IMPD OSE 3 IMPD OSE 3	IMPD OSE 4 IMPD OSE 4	IMPD OSE 5 IMPD OSE 5	IMPD OSE 6 IMPD OSE 6	IMPD OSE 7 IMPD OSE 7	IMPD OSE 8 IMPD OSE 8	IMPD OSE 9 IMPD OSE 9	IMPD OSE 10 IMPD OSE 10	IMPD OSE 11 IMPD OSE 11	IMPD OSE 12 IMPD OSE 12	IMPD OSE 13 IMPD OSE 13	
Tab Time Accel 2x Time Accel 4x	A Access Menu Fuel On/Off Fuel On/Off Comms Open	Z Whisper Menu Whisper Toggle Comms Close	R Element Menu Safety Lanes Tag EJECT CPU Sw Cycle	Y Right Menu See Helix Tag RF Sw Cycle	ATC Menu Open Chat Box Recall Resources	T Tender Menu Peer Center Stop	U MarVing Unlink	I Iner Mode Cut None Default - Tag	O MPO Tag Try SWD Cam	P Toggle Pause Toggle Present	FRACS Bright Dec Comm 1 Vol Dec Comm 2 Vol Dec Precision Vol Dec	HMCS Bright Inc Comm 1 Vol Inc Comm 2 Vol Inc Precision Vol Inc	MSL Vol Dec Threat Vol Dec WV Ansk Dec	MSL Vol Inc Threat Vol Inc WV Ansk Inc
Caps Lock Time Accel 4x Time Accel 2x	AP Pitch Cyc AP Roll Cyc TFN Toggle Parlock Start	S Step Switch Inhibit Parlock Toggle Parlock Tag	OF Overhaul Safety Climb Dept Fuel Dump	AVTR Sw Tag Fuel Tank Tag ALVE Sw Cycle	G Gear Toggle Fuel Lanes Select Alarm Silence All Gear Select	H HUD Scales Cyc HUD OSC Cyc HUD Vehicle Cut HUD Control Cut	J Toggle Jamma E-Cycle Start Emergency All EMD Jett Tag	K WHead Brake Reverse Control Hook Tag	L Lock Clear Fuel Light Tag Laser Sw Tag	M HUD Hdg Dec 5° HUD Hdg Inc 5° HUD Crs Dec 5° HUD Crs Inc 5°	N HUD Hdg Dec 1° HUD Hdg Inc 1° HUD Crs Dec 1° HUD Crs Inc 1°	Alt Press - 5 Alt Press + 5	Alt Press + 5 Alt Press - 5	Return Master Caution ICP On/Off Caution Read Free Read
Shift	W Prev Weapon Front Lanes On PRGM Knob On EMD Inside On	X Next Weapon Front Lanes Off PRGM Knob Off EMD Inside Off	MMMP Cancel CAT Sw Tag	P Pony Switch MMS Sw Tag New Toggle	B Parlock Tag Scrub Clean Scrub Clean Scrub Clean Tag	N WVO Toggle RUE On/Off ANT SEL Cyc	M MM Overhaul Master Arm Cut Master Lst Tag	I HUD PPM Cyc HUD Source Up DL Sw Tag	I HUD Deprel Cyc HUD Source Up DL Sw Tag	I HUD All Cyc HUD Source Up NMP Sw Tag	I HUD Right Cyc HUD Source Up NMP Sw Tag	Shift	Shift	
Ctrl	Alt	Keypress Modifier: Unmodified Ctrl Alt Shift + Ctrl Shift + Alt Ctrl + Alt Shift + Ctrl + Alt	Legend: BLACK BOLD GREEN BOLD BLUE BOLD RED BOLD BLACK ITALICS GREEN ITALICS BLUE ITALICS RED ITALICS	Space WPN Rotation	AltGr	Ctrl								

Impr Master Dash Photo/Comm	Arret Defil IS PTT	UI Functions: SVC Broadcast F1 SVC UI Comm F2 UI Exit / Abort Esc UI Screenshot Impr	3rd Party Software: TIR Precision F7 TIR Profile F8 TIR Pause F9 TIR Recorder F12	TIR PTT Arret Defil TS Broadcast Num * TS Toggle Mike Strt Num * TS Tag Speaker Ctrl Num *	FRAPS Video F9 FRAPS Screen F10 FRAPS Benchm F11 FRAPS Overlay F12
--	------------------------------	---	--	---	---

Num Block Layout

Insert Rot Cursor Swait ICP FLUR Lvl On W Keyboard On	Page Hout Rot Cursor 2x ICP FLUR Lvl Up J Keyboard On	Num Lock ICP Drift on Top ICP FLUR Wk	Num 1 TS Broadcast TS Toggle Mike TS Tag Speaker	Num * ICP Previous Proc F1V
Suppr Cursor FF Del TMS Left TMS Left	Page Bas Cursor Sw FF B TMS Right TMS Right TMS Right	Num 7 ICP 7/MEM Down Get	Num 8 ICP BURE New Up	Num 9 ICP S/MCAL Gauss Part
Cursor Up ICP DCS Up Rot Cursor Up Track Tree Ho On	Cursor Down ICP DCS Down Rot Cursor Down Track Tree Ho On	Num 4 ICP 4-STFT New Left	Num 5 ICP INCRUB New Right	Num 0 ICP 0/TIME New Right
Cursor Left ICP DCS L/W Rot Cursor Left Track Tree Left	Cursor Right ICP DCS R/W Rot Cursor Right Track Tree Right	Num 1 ICP 1-4.8 Down 5x	Num 2 ICP 2-ALOW New Down	Num 3 ICP 3 Slide Back
Cursor Up ICP DCS Up Rot Cursor Up Track Tree Ho On	Cursor Down ICP DCS Down Rot Cursor Down Track Tree Ho On	Num 1 TWP Unlink	Num 2 TWP Spa Test	Num 3 TWP Top Stop
Cursor Left ICP DCS L/W Rot Cursor Left Track Tree Left	Cursor Right ICP DCS R/W Rot Cursor Right Track Tree Right	Num 0 ICP 0-AM-SEL	Num * ICP RCL	Num Enter ICP ENTR Default PDV ICP Hud Par Tag
Cursor Left ICP DCS L/W Rot Cursor Left Track Tree Left	Cursor Right ICP DCS R/W Rot Cursor Right Track Tree Right	Num 0 TWA Power Tag	Num * TWA Low Tag	

Key Combo / Notes

Key Combo set to: Alt C

L - Load Ckpt Defits	S - Save Ckpt Defits
P - Pilot Model	F - Frame Rate
D - Score Display	O - Online Status
H - HUD Rendering	A - Camp Quick Save
R - Random Error	J - Recenter Joystick
T - Reload TrackIR	F1 - Dev Location
F4 - Dev Debug	F5 - Dev Set Scale
F6 - Dev Scale Down	F7 - Dev Scale Up
F8 - Dev Regen	F9 - Dev Position

Note: Only the first 20 Key Combos are shown here (in order of appearance).

Own Remarks:





4.4 Pitbuilder French (AZERTY) Keyboard file

Esc Esc Lock	F1 FRAPS Broadcast ICP OSC0	F2 Day-Cry Switch ICP OSC0 TWK Low Test TWK Search Top	F3 Miscellaneous Hold ICP WF TWK Power-On TWK Power-Off	F4 CPU GEN Select ICP LIST	F5 FLCS PWR Test ICP A-A IMCS Debug Inc IMCS Debug Dec	F6 FLCS Switch ICP A-B IMCS Knob On IMCS Knob Off	F7 TR Precision ICP NAV Mode RWR Per On RWR Per Off	F8 TR Profile	F9 TR Pause	F10 FRAPS Screen	F11 FRAPS Benchm.	F12 TR Recorder
------------------------	--	---	--	---	---	--	--	-------------------------	-----------------------	----------------------------	-----------------------------	---------------------------

Falcon BMS Keyboard Layout (AZERTY / French)

Open Chat Box Master Gain On Master Gain Off	Heat Only LMPD OBB 1 RMPD OBB 1	Snop Ph (SD) LMPD OBB 2 RMPD OBB 2	Plan Ph (SD) LMPD OBB 3 RMPD OBB 3	Prelock LMPD OBB 4 RMPD OBB 4	Wing/Fus Lin Off LMPD OBB 5 RMPD OBB 5	Master Lin Off LMPD OBB 6 RMPD OBB 6	Satellite Cam LMPD OBB 7 RMPD OBB 7	Faly Cam LMPD OBB 8 RMPD OBB 8	Chase Cam LMPD OBB 9 RMPD OBB 9	Drst Cam LMPD OBB 10 RMPD OBB 10	LMPD Bt Dec RMPD Bt Dec	LMPD Bt Inc RMPD Bt Inc	Backspace EJECT ICP Dst On/Off FL Per On FL Per Off
Tab Time Accel Up Time Accel Dn	Access Menu LMPD OBB 1 RMPD OBB 1	Weapon Menu RMPD Knob 1 RMPD Knob 2	Element Menu RMPD Knob 3 RMPD Knob 4	Flyht Menu RMPD Knob 5 RMPD Knob 6	ATC Menu CMDS Mode Sby CMDS Mode Mnt	Center Menu CMDS Mode Tarm CMDS Mode Num	CMDS Mode Gnd RMPD Knob 7 RMPD Knob 8	Hook On LMPD Knob 9 LMPD Knob 10	EMO Hold LMPD Knob 11 LMPD Knob 12	Yocde Pause LMPD Knob 13 LMPD Knob 14	Perk/Brk AntiSel LMPD Knob 15 LMPD Knob 16	Deat Up LMPD Knob 17 LMPD Knob 18	ICP Dst On/Off EAT Se 1 CAT Se 1
Caps Lock Time Accel Up	Master Switch LMPD Knob 1 RMPD Knob 2	LMPD Knob 3 RMPD Knob 4	RF Sw Mode LMPD Knob 5 RMPD Knob 6	AVTR Sw Tag LMPD Knob 7 RMPD Knob 8	ICP Headlight On LMPD Knob 9 RMPD Knob 10	ICP Headlight Off LMPD Knob 11 RMPD Knob 12	TRF Handoff LMPD Knob 13 RMPD Knob 14	Emergency Jnt LMPD Knob 15 RMPD Knob 16	Wheel Status LMPD Knob 17 RMPD Knob 18	Lock Chase LMPD Knob 19 RMPD Knob 20	AP Pch Alt Hold LMPD Knob 21 RMPD Knob 22	AP Pch Alt Hold LMPD Knob 23 RMPD Knob 24	ACK LMPD Knob 25 RMPD Knob 26
Shift	DMS Up LMPD Knob 1 RMPD Knob 2	DMS Down LMPD Knob 3 RMPD Knob 4	DMS Left LMPD Knob 5 RMPD Knob 6	DMS Right LMPD Knob 7 RMPD Knob 8	Play Switch LMPD Knob 9 RMPD Knob 10	ICP Headlight On LMPD Knob 11 RMPD Knob 12	ICP Headlight Off LMPD Knob 13 RMPD Knob 14	Wheel Status LMPD Knob 15 RMPD Knob 16	Lock Chase LMPD Knob 17 RMPD Knob 18	AP Pch Alt Hold LMPD Knob 19 RMPD Knob 20	AP Pch Alt Hold LMPD Knob 21 RMPD Knob 22	ACK LMPD Knob 23 RMPD Knob 24	ICP Dst On/Off EAT Se 1 CAT Se 1
Ctrl	Alt	Keypress Modifier: Unmodified Ctrl Alt	Legend: BLACK BOLD GREEN BOLD RED BOLD	Space WPN Release	AltGr	Ctrl							

Ingr Broadcast	Arrêt Defil To PTT	UI Functions: I/C Broadcast F1 I/C UI Comm F2 UI Exit / Abort Esc UI Screenshot Ingr	3rd Party Software: TR Precision F7 TR Profile F8 TR Pause F9 TR Recorder F12	TS PTT TS Broadcast TS Toggle Mute TS Tag Speaker	Arrêt Defil Num * Stft Num * Ctrl Num *	FRAPS Video F9 FRAPS Screen F10 FRAPS Benchm. F11 FRAPS Overlay F12
--------------------------	------------------------------	---	--	--	--	--

Num Block Layout

Inser Softkey Open ICP FURC Lst On ICP FURC Lst Off	Page Hout Cursor Up ICP OSC0 ICP OSC1	Page Hout Cursor Down ICP OSC2 ICP OSC3	Num Lock BMS Up ICP FURC W	Num / TS Broadcast TS Toggle Mute TS Tag Speaker	Num * Block Test No Up ICP Precision	Num - Block Test No Up ICP Precision
Suppr Cursor FF Out ICP FURC Lst On ICP FURC Lst Off	File Cursor Sw On ICP Dst On ICP Dst Off	Page Bas Cursor Sw Off ICP Dst On ICP Dst Off	Num 7 BMS Left ICP FURC W RMPD OBB 17	Num 8 New Up ICP 8-F2 RMPD OBB 18	Num 9 Glance Falt ICP 8-CAL RMPD OBB 19	Num + Block Test No Dn ICP Inset
Left Arrow RMPD OBB 11 RMPD OBB 12	Right Arrow RMPD OBB 13 RMPD OBB 14	Up Arrow RMPD OBB 15 RMPD OBB 16	Num 4 New Left ICP 4-OTF1 RMPD OBB 11	Num 5 RMPD Knob ICP 5-OTF2 RMPD OBB 12	Num 6 View Right ICP 6-OTF3 RMPD OBB 13	Num = Block Test No Dn ICP Inset
Down Arrow RMPD OBB 17 RMPD OBB 18	Left Arrow RMPD OBB 19 RMPD OBB 20	Right Arrow RMPD OBB 21 RMPD OBB 22	Num 1 BMS Down ICP FURC W RMPD OBB 11	Num 2 New Down ICP 2-CAL RMPD OBB 12	Num 3 Glance Right ICP 3 RMPD OBB 13	Num Enter Block Test No Dn ICP Inset
Left Arrow RMPD OBB 23 RMPD OBB 24	Right Arrow RMPD OBB 25 RMPD OBB 26	Up Arrow RMPD OBB 27 RMPD OBB 28	Num 0 Block Test Left ICP 0-MAGC RMPD OBB 29	Num - Block Test Right ICP INCL	Num = Block Test Right ICP INCL	Num = Block Test Right ICP INCL

Key Combo / Notes

Key Combo not set.

Note: Only the first 20 Key Combos are shown here (in order of appearance).

Own Remarks:





4.5 Default German (QWERTZ) Keyboard Layout

Esc Exit Sim	F1 ECP COM1 FRAPS Frontal View Radar Range On	F2 ECP COM2 On-Off Switch Radar Range On	F3 ECP RFP Radar Joystick Hold HUD Range Desc Radar Range On	F4 ECP LST Radar Head Up HUD Range Hold Radar Range On	F5 ECP AA Radar Head On Rolling On	F6 ECP AG EFP GEN Switch Radar View Center	F7 TR Precision FLCS PAWI Test Radar View Up	F8 TR Profile	F9 TR Pause Pause Hold Left Hold	F10 FRAPS Screen FRAPS Full Left Hold	F11 FRAPS Benchm. F-18 View Down FRAPS On	F12 TR Recorder F-18 View Up Pause On
------------------------	---	--	---	---	--	--	--	-------------------------	--	---	---	---

Falcon BMS Keyboard Layout (QWERTZ / German)

A Kanal1 Cam Action Cam Photo Filter Lobby View	1 Nud Only Lobby View Lobby View	2 Shap Pk (C3) EFP Gen Hold Shadows	3 Plan Pk (C3) EFP Gen Hold Shadows	4 Pakback Mode AA EFP Gen Hold Shadows	5 EFP Gen Hold Shadows	6 EFP Gen Hold Shadows	7 EFP Gen Hold Shadows	8 EFP Gen Hold Shadows	9 EFP Gen Hold Shadows	0 EFP Gen Hold Shadows	Backspace F-ACK EFP Gen Hold Shadows	
Tab New Asset On New Asset Up	Q Radar Menu Radar Menu Radar Menu	W Weapon Menu Weapon Menu Weapon Menu	E Radar Menu Radar Menu Radar Menu	R Radar Menu Radar Menu Radar Menu	T Radar Menu Radar Menu Radar Menu	Z Radar Menu Radar Menu Radar Menu	U Radar Menu Radar Menu Radar Menu	I Radar Menu Radar Menu Radar Menu	O Radar Menu Radar Menu Radar Menu	P Radar Menu Radar Menu Radar Menu	U Radar Menu Radar Menu Radar Menu	R Radar Menu Radar Menu Radar Menu
Caps Lock New Asset On New Asset On	A AP Pkch Cut AP Pkch Cut AP Pkch Cut	S Shap Switch Shap Switch Shap Switch	D EFP Overide EFP Overide EFP Overide	AUTH Auth Sw Top Auth Sw Top Auth Sw Top	G Gear Toggle Gear Toggle Gear Toggle	H HUD Status Cyn HUD Status Cyn HUD Status Cyn	J Track Jammer Track Jammer Track Jammer	K Wheel Brakes Wheel Brakes Wheel Brakes	L Lock Clear Lock Clear Lock Clear	H HUD Hdg Des 0° HUD Hdg Des 0° HUD Hdg Des 0°	A HUD Hdg Des 0° HUD Hdg Des 0° HUD Hdg Des 0°	Return Master Caution EFP Gen Hold Shadows
Shift	Y Prev. Waypoint Prev. Waypoint Prev. Waypoint	X Next Waypoint Next Waypoint Next Waypoint	C EFP Cancel EFP Cancel EFP Cancel	V Vkey Switch Vkey Switch Vkey Switch	B Squad Tag Squad Tag Squad Tag	N NVG Toggle NVG Toggle NVG Toggle	M MFD Overide MFD Overide MFD Overide	R Rudder Left Rudder Left Rudder Left	R Rudder Right Rudder Right Rudder Right	Y HUD ARM MS HUD ARM MS HUD ARM MS	Shift	
Strg	Alt	Keypress Modifier: Unmodified Ctrl Alt Ctrl + Alt	Legend: BLACK BOLD GREEN BOLD BLUE BOLD RED BOLD	Space With Release	AltGr	Strg						

Druck Screenshot Print Command	Rollen TS PTT	UI Functions: AVC Broadcast: F1 AVC UI Corrm: F2 UI Exit / Abort: Esc UI Screenshot: Druck	3rd Party Software: TR Precision: F7 TR Profile: F8 TR Pause: F9 TR Recorder: F12	TS PTT: Roller TS Broadcast: Num * TS Toggle Make: Sh Num * TS Tog Speaker: Ctrl Num *	FRAPS Video: F9 FRAPS Screen: F10 FRAPS Benchm.: F11 FRAPS Overlay: F12
---	-------------------------	---	--	---	--

Num Block Layout

Einlg RCP Curser Small ECP FLUR Lst On RCP Headboard On	Pos1 Curser Sw On EFP Lst On EFP Lst On	Bild ▲ RCP Curser Zers ECP FLUR Lst Up J Keyboard Hold	Num Lock	Num 1 ECP Drft on Top ECP FLUR WS TS Toggle Make	Num + ECP Previous ECP F10V	Num - ECP Previous ECP F10V
Entf Curser FF Out EFP Lst On EFP Lst On	Ende Curser Sw On EFP Lst On EFP Lst On	Bild ▼ Curser Sw FF In EFP Lst On EFP Lst On	Num 7 EFP LMMR Zoom Out	Num 8 EFP RFLS New Up	Num 9 EFP SALCAL Status Full	Num * EFP Next ECP F10V
▲ ECP DCS Up RCP Curser Up Stick Thr Up On	▲ Rise Up Rise Up Rise Up	▲ Rise Up Rise Up Rise Up	Num 4 ECP 4-STPT New Left	Num 5 ECP INCRUS New Right	Num 6 ECP 6-TMC New Right	Num * ECP Next ECP F10V
◀ ECP DCS Wn RCP Curser Left Stick Thr Left	◀ Rise Down Rise Down Rise Down	◀ Rise Down Rise Down Rise Down	Num 1 ECP 1-LS Zoom In	Num 2 ECP 2-LOW New Down	Num 3 ECP 3 Stick Back	Num * ECP Next ECP F10V
▶ ECP DCS RD RCP Curser Right Stick Thr Right	▶ Rise Down Rise Down Rise Down	▶ Rise Down Rise Down Rise Down	Num 0 ECP 0-SEL New Power Top	Num 0 ECP 0-SEL New Power Top	Num * ECP Next ECP F10V	Num * ECP Next ECP F10V

Key Combo / Notes

Key Combo set to: Alt C

L - Load Ckpt Defits	S - Save Ckpt Defits
P - Pilot Model	F - Frame Rate
D - Score Display	O - Online Status
H - HUD Rendering	Q - Camp Quick Save
R - Random Error	J - Recenter Joystick
T - Reload TrackIR	F1 - Dev Location
F4 - Dev Debug	F5 - Dev Set Scale
F6 - Dev Scale Down	F7 - Dev Set Scale Up
F8 - Dev Regen	F8 - Dev Position

Note: Only the first 20 Key Combos are shown here (in order of appearance).

Own Remarks:

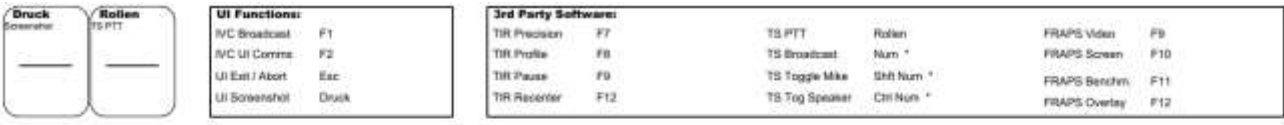
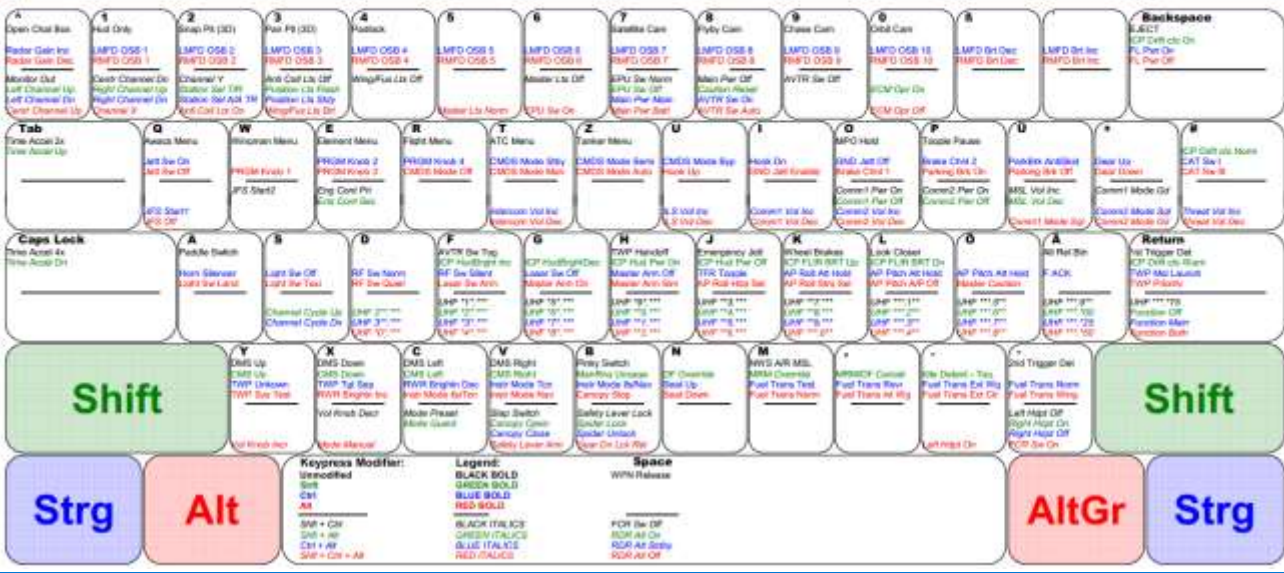




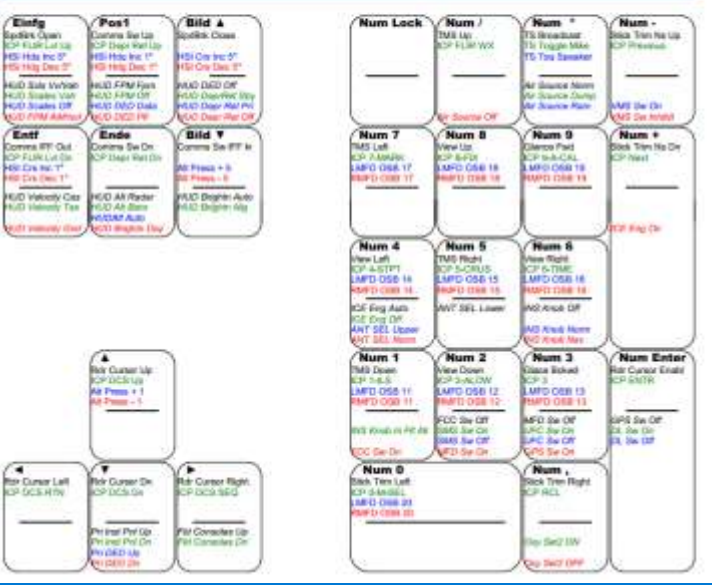
4.6 Pitbuilder German (QWERTZ) Keyboard Layout



Falcon BMS Keyboard Layout (QWERTZ / German)



Num Block Layout



Key Combo / Notes



The above graphics are created with the BMS Key file editor.xls located with its manual in Docs\01 Input Devices\02Key File Editor folder.





5. External Display Support

5.1 Falcon BMS Display Extraction

The old BMS display extraction has been removed from 4.35. The new way to extract the displays is through RTT remote.

5.2 RTT Remote

Remote Display Extractor is now the main method to extract cockpit displays.

RTT remote extracts specific cockpit displays via shared memory textures and renders them outside the BMS process. This allows the main screen to remain in any mode you choose, full screen, borderless windowed or windowed.

RTT Remote is located in your `\Tools\RTTRemote` folder and is now linked to the Launcher menu. The application is available in 32 (client only) and 64 (client & server) bit versions and – like BMS itself - does not support Windows XP or Vista.

It can either be used as a pure local client (RTTClient, standalone), or in a networked configuration. The latter consists of a server (RTTServer) part that reads the RTT data from BMS, and a client part (RTTClient) that displays it. They communicate over TCP, so you can use the client either locally or on remote networked PCs. If used remotely, the client is also capable of mirroring the "normal" shared memory areas, so you can run some instruments remotely without the need for another shared memory mirror tool.

You should enable RTT extraction by setting `set g_bExportRTTTextures` to 1 in Falcon BMS.cfg. You can either keep the batch size as default (update every other frame): `set g_nRTTExportBatchSize 1` or change it to 2 or higher to reduce the number of updates (for slower PCs = less FPS impact).

Managing the application is made by simply editing the RTTServer and RTTClient.ini files. All settings are clearly explained within the respective files:

5.2.1 RTTServer settings

```
HOST = 0.0.0.0  
PORT = 44000  
COMPRESSION = 1  
JPG_QUALITY = 80  
FPS = 30
```

The server is preset for remote use and only needs to be configured/started if you actually want to send the display data to a remote networked client. It will listen on all local IPv4 addresses by default. If you want to restrict it to a specific IP or network interface, you can adapt the "HOST" IP address. You can use both IPv4 or IPv6 addresses. The rest of the options are best left at default except maybe FPS. By default, max FPS is set at 30, i.e. that is the maximum number of updates the server sends to a client every second. If you do have a high-end system you may increase that to 60 or whatever your screen refresh rate is.

The rest of the options are clearly explained in the .ini file.





5.2.2 RTT Client settings

The client side is where you setup your displays. RTTRemote is able to display both MFDs, the HUD, the DED, the PFD, the RWR and the HMCS locally. If used remotely, RTTRemote is also able to read parts of the shared memory and thus enable 3rd party application that usually would need to run on the BMS machine itself to display further instruments on the non-BMS remote machine.

RTTClient is preset for local use. If you want to use it remotely you will have to adjust the "NETWORKED" option and "HOST" IP address.

```
NETWORKED = 0  
HOST = 127.0.0.1  
PORT = 44000
```

For local use only, you can specify the maximum number of updates RTTClient will draw via "FPS" (for remote use, see the "FPS" parameter of RTTServer instead).

```
FPS = 30
```

If you run RTTClient on a remote machine in the network, setting any of these items to 1 enables repeating the relevant part of the shared memory as if BMS would be running on the remote machine.

```
DATA_F4 = 0  
DATA_BMS = 0  
DATA_OSB = 0  
DATA_IVIBE = 0
```

This option overlays a grid on the RWR. 0 means no grid, 1 overlays a basic grid.

```
RWR_GRID = 0
```

The following section is used to define what is extracted and what is not. Obviously 1 does extract, 0 does not extract.

```
USE_HUD = 0  
USE_PFL = 0  
USE_DED = 0  
USE_RWR = 0  
USE_MFDLEFT = 1  
USE_MFDRIGHT = 1  
USE_HMS = 0
```

For each of the displays enabled above, you must find settings line for placement and eventually sizes.

Placement is done in coordinates system (DISPLAY_X= & DISPLAY_Y=) in the exact same way as calculated for the previous display extractor. The main screen origin is (0,0). Screens placed on the left have a negative X value, screens placed above the main screen have a negative Y value.

Size is optional (DISPLAY_W= & DISPLAY_H= for Width and Height respectively)

If no size values are entered, the actual (unscaled) internal BMS texture sizes are used.

```
MFDLEFT_X = -1200  
MFDLEFT_Y = 0  
MFDLEFT_W = 500  
MFDLEFT_H = 500  
MFDRIGHT_X = -600  
MFDRIGHT_Y = 0  
MFDRIGHT_W = 500  
MFDRIGHT_H = 500
```

The settings to the left will extract two MFDS sized 500x500 to the top of a screen placed to the left of the main screen. top left corner of the left MFD will be (-1200,0) and the top left corner of the right MFD will be placed at (-600,0). A 100 px gap will be visible between the 2 MFDS.





5.2.3 Using RTTRemote

Launch either RTTClient only (locally), or both the RTTServer (locally) and RTTClient (remote) applications. Both will have an icon displayed in the taskbar. The extracted displays are already active but crossed with an orange X. They will become active as you launch to 3D.

Some third-party applications will start only when BMS is running. This may create all sorts of problem like forcing you to ALT TAB out of BMS to start an application that could not be started before BMS became active. Some other applications will simply refuse to start as long as BMS is not started, which would prohibit use of shared memory readers on remote machines, even though RTTRemote has replicated the data there.

To overcome these issues, RTTRemote features a fake BMS application that may be run and trick all these third-party software into believing BMS is already running. Simply launch RTTClient64 (or32)_FakeBMS.exe instead of RTTClient.

Note that this will obviously only be useful on remote machines where BMS is indeed not running. If you start the "fake" version on the BMS machine itself, BMS will refuse to start (because it thinks it is already running).

5.2.4 Pros & Cons

Pros:

- Allows you to run BMS in any display mode (full screen included, unlike with the old tool).
- Allows you to trick other applications into believing BMS is running on remote networked machine.
- Easy to adapt to Local or Remote use.

Cons:

- If you record your flights using OBS Studio (to allow you to overlay MFDs on your screen) the latest 4.34 version of RTTRemote uses OpenGL which OBS cannot record. The version shipped with 4.33 however should still work just fine if this is an issue for you.

5.3 Third-Party extraction

5.3.1 MFDE

MFD Extractor created by Lightning allows you to extract the displays, the main and secondary instruments and even some full panels like CMD5, the Caution Panel etc. It is also networkable. MFDE is not supported anymore but remains widely used in the community.

5.3.2 Y.A.M.E.

Y.A.M.E. (Yet Another MFD Extractor) by Roccio, Focaldesign and BVT_Scorpion is a new tool that was created for 4.33 and extracts all displays, instruments, specific panels and even the CPD (large central screen that replaced all the instruments on the central console on some CCIP F-16s) and has a better compatibility with Helios (touchscreen). Unfortunately Y.A.M.E. development has stalled and the same workaround to make it work with 4.34 will be required to make it work with 4.35. Hopefully in the future development can be resumed.





6. The FPS Quest

BMS for all its improvements is still partly based on the ancient Falcon 4.0 code, which was not optimised for today's multi-threaded, multi-core CPUs and GPUs capable of parallel processing. The code renders views in a more serialised way, one after the other. So the more views we render and the more there is to see in a particular view, the slower everything gets. As such it benefits much more from a fast CPU than a multi-core one.

If you look at Task Manager or Resource Monitor you will notice that the CPU and/or the GPU are now better utilized than previous versions of BMS. Both are still idling to some degree but the BMS coders have stretched the potential of the Falcon code beyond the wildest dreams of its creators, and they have pushed the limits even further with 4.35. For example, BMS now uses a dedicated thread for all graphics rendering, which decouples it from the simulation thread and allows it feeds the GPU with as much throughput as possible. Nevertheless, we are not yet at a point where all the parallel processing capabilities of current hardware can be exploited to the fullest so a single fast GPU is preferable to a SLI or Crossfire configuration of slightly slower ones.

BMS 4.33 was graphically more demanding than 4.32, partly due to the variety and higher resolution textures of the new tiles and partly for the reasons outlined above; this was especially noticeable when using the TGP, TFR, FLIR on HUD and video WPN displays. 4.33 with comparable settings would run slower than 4.32 did; how much would vary from system to system.

4.34 & 4.35 were both further improved and optimised but on older systems you may still find your frame rate drops too much for comfort and you may have to decrease some options to gain FPS back. This section will suggest options to increase FPS without impacting the overall experience too much.

BMS will use (i.e. allocate) memory resources each time you enter 3D as the features getting loaded remain in memory until you exit BMS. So one way to lose FPS is to constantly enter and exit 3D in different scenarios. The impact should be minimal in typical use for newer PCs with lots of memory but be aware that going in and out of 3D (for example flying several consecutive missions) without restarting Falcon BMS will progressively use up memory, amongst other problems. Your first reflex upon discovering an abnormal FPS issue is to restart the software, just like you would do with any other software.

This is not a bug or memory leak per se, it's more about not having a proper "cleanup concept" of in-game objects, due to the fact that Falcon was designed in such a way that the action can continue, even when in 2D. What happens is that once stuff is loaded (even in aggregated form), it *stays* loaded - at least as an oversimplified concept; there are exceptions, of course.

It is not a trivial task to distinguish the "needs to stay loaded, because campaign/TE is still running!" stuff from the "not needed anymore, because i.e. we loaded a completely different TE" stuff. So, these "leaks" are not accidental, or by oversight; they are by design. The complexity of having proper after-3D cleanups is WAY more complicated than establishing the "always restart BMS before re-entering 3D" rule.





FPS impact will depend on your hardware components, system configuration and drivers. Ensure the drivers you are using are best suited for BMS (www.benchmarksim.org is a good place to ask). The most important advice is to try out the different options for yourself; what works on one PC might not make as much difference for yours.

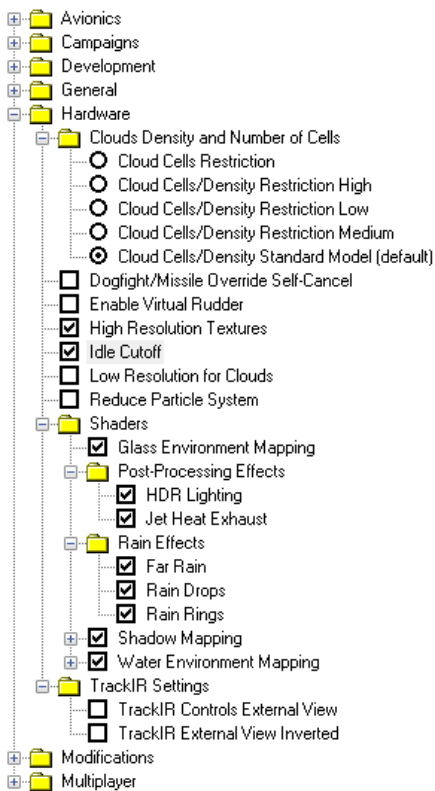
There are four BMS features that may have an impact on FPS on older hardware:

- Autogen trees/grass

- Clouds. Cloud impact on FPS has been improved in 4.35 as we draw less clouds. Moving the autogen slider position to the left and drawing fewer cumulus clouds *might* further decrease the graphical workload and increase FPS.
- Autogen trees/grass density
- Multisampling
- HDR Lighting

You may start with these 4 options and judge their impact on your system.

If you still need more FPS you will have to deactivate more eye candy features. Open the Falcon BMS Config app and select the hardware section:



The clouds density and number of cells options are a further way to decrease the clouds impact on your FPS.

Lower resolution items may help FPS so you can always try activating Low Resolution for Clouds and deactivating High Resolution Textures.

Selecting Reduce Particle System will save a few FPS when the PS is used (explosions, smoke, etc.).

Extracting displays may cost FPS even with the new RTT remote new tool, although the impact should be minimized. If you do extract your displays, ensure that extracted display windows do not overlap each other's or are spread partly on two different screens

Shaders are FPS hungry so you may want to deactivate some of them according to how serious your FPS issue is.

Rain Effects can be turned off but obviously this will only have an impact in inclement weather.

Shadow Mapping does not bring much to immersion but has FPS costs so you may want to deactivate these options.

Water Environment Mapping can be deactivated if you can live with less realistic water rendering.





7. Weather and How to create FMAPs

When it comes to creating weather, the user has 2 options:

- Create simple weather models with the deterministic or probabilistic models straight from the UI.
- Create Fmaps with Weather Commander (user created) or F4Wx (real weather data).

Fmaps are maps of the weather over the BMS terrain. Each area of the terrain can be assigned a specific cell of the usual Sunny, Fair, Poor and Inclement type. Each cell features the following information:

- Weather type
- Pressure
- Temperature
- Wind heading and speed including wind aloft parameters
- Cloud lower layer base altitude
- Cloud upper layer altitude
- Cumulus Coverage and Size
- Visibility
- Speed and direction of the airmass moving

Fmaps allow you to generate extremely complex weather types (you can now transition gradually from a sunny sky with a lot of fog in the morning to a thunderstorm front in the afternoon).

4.34 Fmaps should be compatible with 4.35 (4.33 are not)

Building Fmaps will definitely create more work but will provide a much more immersive situation.

It would be unwise in a simulation to plan a complicated COMAO, brief it for an hour or more then have to abort because a real weather fmap makes the mission objective impossible to achieve. Unlike real life where the COMAO will be put on standby until the weather cooperates, in a simulation we can make the weather support the TE designer's objectives, as he planned them.

Therefore, unlike real life, the weather situation can be adapted to the TE mission objectives.

When using that aspect of the simulated weather you cannot really always use real weather (you may but you need to ensure it's compatible with your mission's objective first). In this specific scenario, creating an Fmap for a single TE is made with **Weather Commander** by Falcas, where the user alone decides the weather parameters.

Using real weather is nevertheless appealing and is quite possible using another tool: **F4Wx** by Ahmed.

This application used real-world GRIB data taken from around the globe (obviously matching the BMS terrain) and converts the real weather data into the BMS Fmap format. This is where the complexity of the Fmap model really shines and can result in very immersive Fmaps.

Consecutive Fmaps can be created with this tool which can make a great candidate for dynamic campaign weather using the map auto-update function.

Both tools are documented below. Hopefully this is enough to give you a glimpse on how to use them and motivate you to try to learn more about them. They are very powerful tools.





7.1 Weather changes in 4.35

The weather changes are listed in the BMS manual. (new turbulences, moon phases, star map, sunrise and sunset, ...) The technical side of these changes is minimal.

7.2 Weather changes in 4.34

7.2.1 Wind change

Wind and cloud mechanisation have drastically changed in BMS:

4.34 introduced Winds Aloft features. In the real world, the wind gets stronger and turns right with altitude (degrees increase).

You can now expect this in all weather models of BMS.

When using deterministic and probabilistic weather models, the UI wind tab only gives you information about surface winds. The wind will nevertheless change with altitude, according to the default settings from the table to the right.

When using a map model the winds aloft can be changed by the user, or will be adapted according to the real-world data used.

Altitude (ft)	WindHeading	Windspeed
0	Y	X
3000	Y + 4 deg	X + 2.5 kts
6000	Y + 8 deg	X + 5 kts
9000	Y + 12 deg	X + 7,5 kts
12000	Y + 16 deg	X + 10 kts
18000	Y + 20 deg	X + 12.5 kts
24000	Y + 24 deg	X + 15 kts
30000	Y + 28 deg	X + 17.5 kts
40000	Y + 32 deg	X + 20 kts
50000	Y + 36 deg	X + 22.5 kts

7.2.2 Clouds change

The cloud model has been totally updated in 4.34, including new clouds and lighting effects.

Clouds are shared in multiplayer, which means that all players should see the same cloud cover throughout the theatre.

6 types of clouds are now modeled,

3 from the lower atmosphere: (Cumulus (FAIR), Strato-Cumulus (POOR) and NimboStratus (INCLEMENT)) and 3 from the upper atmosphere (Cirrus (SUNNY), Cirro Cumulus (FAIR) and Cirro Stratus (POOR & INCLEMENT)) .

Cloud base is always expressed in feet MSL.



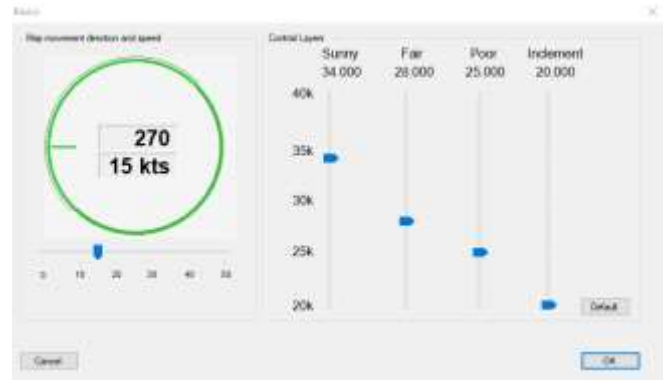


7.3 Weather Commander

Weather Commander can be downloaded from <http://www.weaponeliveryplanner.nl>

Upon opening Weather commander a map of Korea (default BMS terrain) is displayed. Other background images can be loaded from the file menu for third party terrain. A 1024 theatre is divided into 59x59 weather cells. Each cell can be characterised with the specific weather information.

The first thing to do when creating an fmap is to click the basic map data button. This opens a new window where the movement of the airmass can be defined with a direction and a speed. Contrail layers can also be set separately for each weather type. Once set, click OK to return to the main window.



Everything depends on the weather type. So the next obvious step is to paint the weather cells (sunny, fair, poor, inclement) on the map.

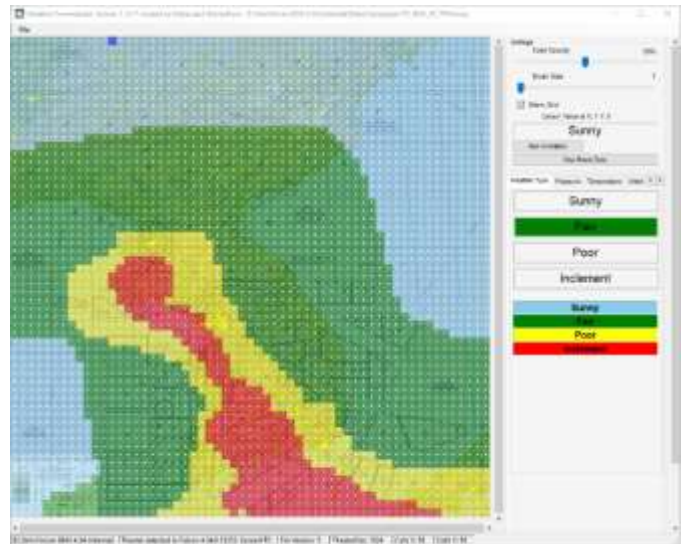
Color Opacity and Brush Size can be set with the two sliders in the Settings area. Weather Type is then simply painted over the terrain image. Varying the brush size gives you more control over transition area.

Weather Type

When the Weather Type tab is selected the map can be painted with one of the 4 BMS weather types.

Colours are fixed:

- light blue for Sunny
- green for Fair
- yellow for Poor
- red for Inclement



Pressure

The Pressure tab allows the user to set pressure for each cell from 950 to 1060Mb (28.05 to 31.30 inHg).



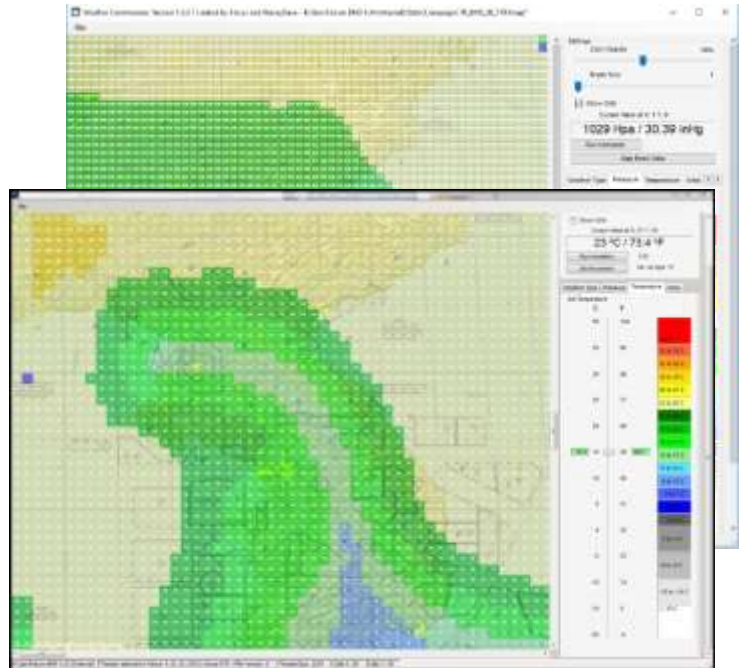


Colours are fixed according to the colour scale displayed on the right. First select a pressure you want to set then paint on the map where you want to set that particular pressure. Then do the same for the next pressure settings. Generally bad weather has a low pressure and good weather is a high pressure system. Try to avoid drastic pressure variations as these are unrealistic. Pressure variation is usually slow and very gradual. Brush thickness and opacity can be set as usual.

Temperature

Temperature at ground level can be painted on the map in the same way as pressure. A scale of Celsius and Fahrenheit temperature is colour coded. Once painted, each cell will display the colour of its associated temperature. As usual select a temperature then paint the map with the brush.

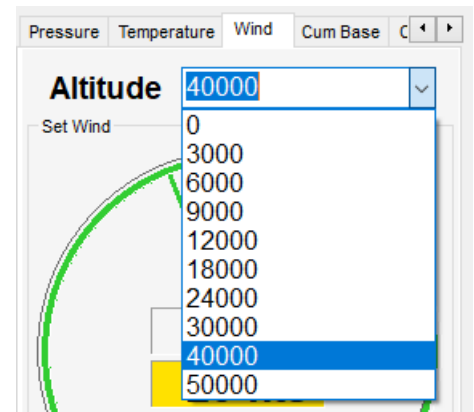
The Cursor Value gives temperature in °C and °F at the cursor position. Obviously temperature is higher when the sun is shining. Poor weather (in the day at least) has a lower temperature due to cloud cover.



Wind

Winds aloft have been introduced with 4.34 weather model, You may set wind direction and speed at different levels to accommodate the 4.34 winds aloft features. This is done with the Altitude drop down menu.

Select an altitude layer, paint the map with winds and select the next until all are covered. Remember that the wind picks up speed and degrees (turns right) as the altitude increases.

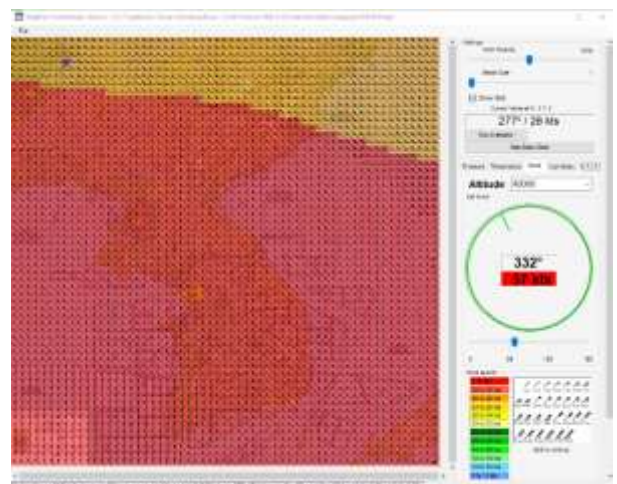


Wind direction is set by clicking the green circle from where you want the wind to blow.

Wind speed is set through the Speed slider from 0 to 150 kts. When both are set the map can be painted with the settings. As with real maps wind speed and direction is given with vectors and side bars for speed.



In the Northern hemisphere winds go from the Low pressure system to the High pressure system and are deviated to the right due to the Coriolis Effect. They turn clockwise around high pressure and anti-clockwise around low pressure systems. We can consider that wind is parallel to the isobars (points of same pressure) and the winds should be stronger as the isobars are closer to each other (if pressure changes rapidly, wind is stronger).



Cloud Base

The next tab allows the Cloud Base (corresponding to the Low clouds in the UI) to be set in feet MSL between zero and 10,000 feet.



Cloud Coverage

The Cloud density has the same meaning as the top slider in the cloud tab of the weather UI. It sets the clouds coverage in octats.

You have the choice between Few (FEW = (1-2)/8), Scattered (SCT = (3-4)/8), Broken (BKN = (5-6-7)/8) and Overcast (OVC = 8/8).

Each option has its own colour.

Please note, a Sunny weather cell will have clear skies (0/8) therefore to set no clouds at all, you must select Sunny weather type for that cell. In the same logic, you can't have inclement weather with few or SCT clouds. Therefore cells with poor and inclement type will be BKN (broken) at minimum. When you try to paint FEW over such a cell, you will notice that the cells are not painted.

Table 9-10. Sky cover. Oktas=eighths of sky covered.

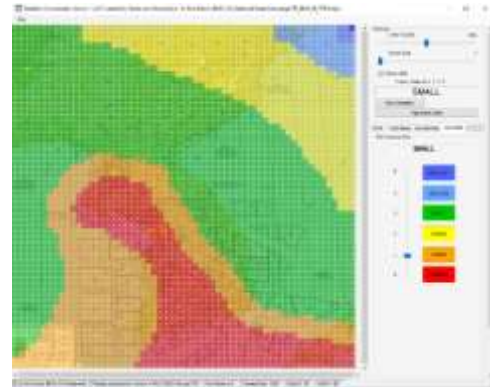
Sky Cover (oktas)	Sym-bol	Name	Abbr.	Sky Cover (tenths)
0	☉	Sky Clear	SKC	0
1	☁	Few* Clouds	FEW*	1
2	☁			2 to 3
3	☁	Scattered	SCT	4
4	☁			5
5	☁	Broken	BKN	6
6	☁			7 to 8
7	☁			9
8	☁	Overcast	OVC	10
(9)	☁	Sky Obscured		un-known
(/)	☁	Not Measured		un-known

* "Few" is used for (0 oktas) < coverage ≤ (2 oktas).

The term ceiling starts only from Broken. You cannot define ceiling under 5/8 cloud coverage. CAVOK (Ceiling and Visibility OK) is often misunderstood; you may have a CAVOK situation when the cloud coverage is above 5000 feet. An overcast layer at 9000feet but no cloud ceiling under 5000 will be defined as CAVOK as long as visibility is above 10 kilometers.

Cloud Size

The Cloud Size has the same meaning as the second slider in the cloud tab of the weather UI. It sets the clouds from Congestus to Humilis with a scale from 0 to 5 with 5 being the Humilis side of the slider and the 0 being the Congestus side of the slider.



Towering Cumulus

This tab defines if the cumulus clouds have strong vertical development. As you may know, cumulus clouds go through different development stages between the formation of storm cells. Before the storm, the cumulus start to grow high in tower forms (hence the towering) and develop strong downdrafts outside the clouds and strong updrafts inside the clouds.

There are only two settings you can apply: with (YES) or without tower (NO). Each is coloured accordingly.

Shower (not implemented in 4.35)

This tab defines if the clouds are giving shower or not. Showers are typical of unstable weather (FAIR)

There are only two settings you can apply: with (YES) or without showers (NO). Each is coloured accordingly.

Visibility

The last tab is simply used to set the visibility. You can set it from 0 to 60 km just like with the UI slider. The settings are coloured accordingly as well.

Once all layers have been painted on the map, save the file. Weather Commander will save the weather as an fmap that should be placed in the `\Data\Campaign\` folder and then can be activated in the BMS UI.

Please note, once a TE is set to weather map model, the Fmap is duplicated and renamed with the mission name.





7.4 F4Weather (F4Wx)

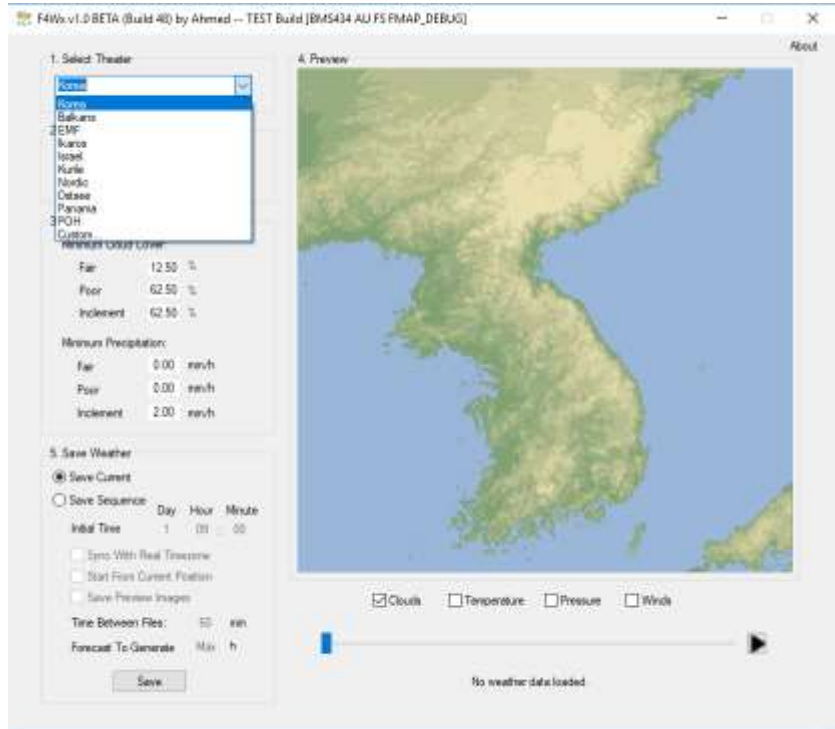
Although time consuming, creating a weather map the size of BMS KTO with Weather Commander is easy, but if you want it to be realistic you need to have some knowledge about weather in real life and weather can be complicated.

A new real weather tool emerged on the public forum and has been optimised for the 4.34 weather system. This tool allows you to download real weather data files and adapt them to BMS .fmap format very easily. F4Wx can be downloaded via the BMS forum and is still compatible with 4.35

F4Weather uses GFS data to download real weather and as you may remember from making real weather for 4.33 one of the hardest things to do was to define the real-world GPS coordinates matching those of the BMS theatre.

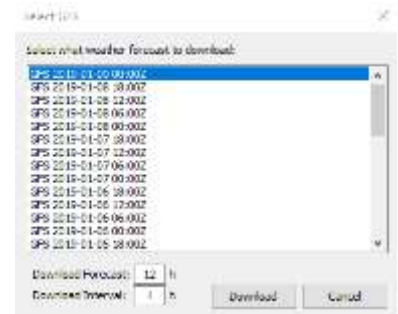
F4Weather comes preset with a list of BMS terrains the user can choose to create Fmaps for. In this example we'll obviously pick Korea.

Using the application is really easy and intuitive. As you want to replicate real-weather in BMS, the first obvious step after having set the theatre is to download the weather. This is done with the Download Weather button. Please note, you may also load weather files from other "local" sources.



Clicking on the Download Weather button opens a new window where you may select the GFS weather to download. Select the top line which should correspond to the latest weather available and input desired intervals in the two boxes below:

- Download Forecast: Default value is MAX but you can limit the number of files to download by setting an hour limit. In this case, we will limit our request to 12h forecast.
- Download interval: Default value 3 hours. Weather files are available hourly but over a long forecast request, it will create a lot of Fmaps. You can therefore limit the interval of the weather change by inputting the number of hours between 2 Weather data updates. In our example we want accurate weather over a short time (12 hours) by setting the interval to 1h.



This should generate 13 Fmaps. Click on the Download button to start downloading the weather files. It may take a while depending on how much information you chose to download.



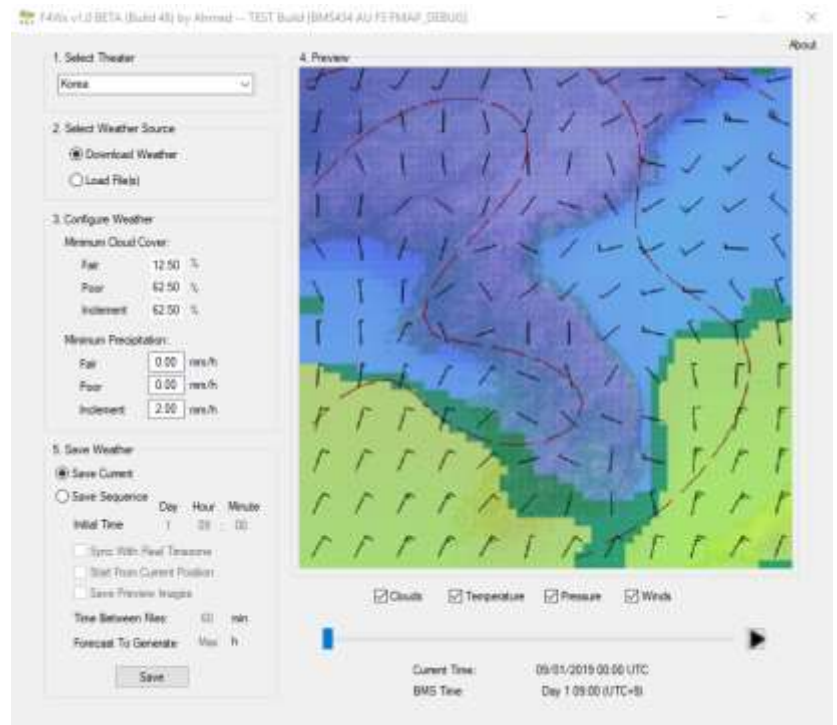


After downloading the files, the map will display the current weather. You may enable more weather options to be displayed by enabling the Temperature, Pressure and Winds checkboxes just under the map.

Placing the mouse cursor anywhere on the map opens a contextual window with the weather setting at that particular location.

The sequence of the weather can be played with the PLAY radio button on the bottom right of the window. This will show you how the real weather data you just downloaded is supposed to evolve over the next 12 hours.

The next step is simply to create the corresponding Fmaps.

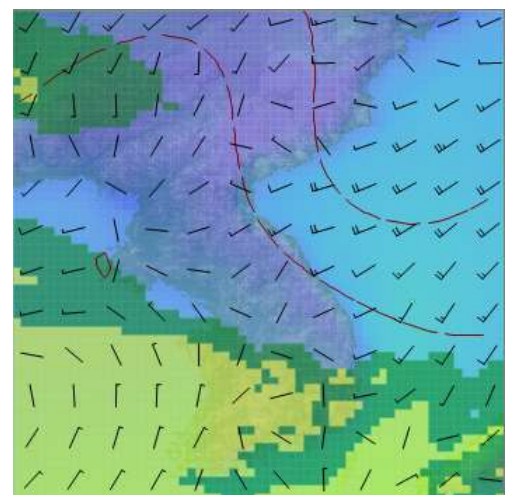


The “Save Current” option allows you to save a single Fmap according to the Time slider position. Position the time slider to a specific forecast you just downloaded and hit the Save button. One Fmap will be created. But we want here to save the full sequence of the downloaded weather forecast. Select the “Save Sequence option” and check the 3 next checkboxes (Sync with Real Timezone, Start from Current Position and Save Preview Images). The time between files and Forecast to Generate can be set as well but we will use the default settings. Click “Save”.

F4Wx will prompt you for a location folder to save the Fmaps. You may elect to copy them directly to your BMS install (`\Data\Campaign` or `\Data\Campaign\WeatherMapsUpdates`) but I always use a temporary location under the app folder (`\F4Wx\Fmaps`). As expected, 13 Fmaps were created.

Files are named according the day and hour in BMS. In this case, the sequence starts Day 1 at 09.00 (10900.fmap) and ends Day 1 at 21.00 (12100.fmap). A subfolder is also created with previews of the weather forecast for each interval you selected (if you enabled that option). This will be particularly handy for your briefings.

You may create as many Fmaps as you wish and apply them in TE or in campaign. Maps will auto-update from one to another, which is a much better way than having a single weather map that moves across the BMS terrain. When a single map is moved, the part that disappears on one side reappears on the other side which might generate weird weather artefacts.





7.5 Maps Auto Update



With the 13 Fmaps you just created with F4Wx let's create a TE scenario on a 12 hour timeframe with the BMS weather system generating a weather update every hour.

Create a TE which by default will start at 09:00 FT Day1. Once your initial TE is created, open the Weather menu and select Map Model as the Weather model. You then need to load a Fmap from the list below. Select the 10900.Fmap that you copied in your `\Data\Campaign` folder.

Enable MAPS AUTO UPDATE. This will make the TE look in the `\Data\Campaign\WeatherMapsUpdates` folder for subsequent Fmaps named according to the convention explained below and update the weather according to the Falcon clock.

Do not forget to save the Weather in the UI by clicking the "SAVE WTH" button.

Since F4Wx uses the same naming convention as the BMS UI, the files created with f4Wx work perfectly with the weather updates. In our case, the initial weather will be the data from 10900.fmap initially loaded and the weather will switch to the 11000.fmap (day1 – 10.00FT) at 10 AM falcon time.

Please note the initial Fmap must be copied to the `\Data\Campaign` folder (so it can be used as the initial weather file) and the other subsequent Fmaps must be copied to the `\Data\Campaign\WeatherMapsUpdates` folder.

You may now create a full weather briefing using the Fmap generated preview images. When doing this, Weather Commander can be a great tool to open and read the F4Wx generated Fmaps. This will provide you with many variables for creating your weather briefings.



Please note:

Minimum interval for auto loaded weather maps is now 55 minutes to guarantee correct interpolation. Maps below minimum interval will be disregarded.

7.5.1 Naming convention

You may generate your own Fmaps for the auto-update feature but you must abide by the naming convention in order for the weather to update accordingly.

The maps to be subsequently loaded must be placed in the `\Data\Campaign\WeatherMapsUpdates` folder with a specific format: **day * 10000 + hour * 100 + minute.fmap**.

If the auto-update flag is set on in the UI - WEATHER page the campaign engine will load the map when the time matches the file name.

Example:

A 30509.fmap will be loaded at day 3, 5 hours and 9 minutes.

Day:	3 * 10000	=	30000			
				+		
Hour:	5 * 100	=	500			
				+		
Minute:	9	=	9	=	30509 + .fmap =	<u>30509.fmap</u>



8. Views

The Views have been documented in chapter 9 of the BMS manual. This chapter here will cover the more technical details of managing the views in BMS.

8.1 Field of View

- Default values

Field of View:	60°
Min. FOV:	0°
Max. FOV:	80°
FOV Increment:	5° steps
- Internal vs. External FOV

You can also adjust the FOV in external views. FOV is retained when switching from external to internal views. Stick input will remain constant, regardless of FOV.
- Mouse Scroll Wheel and Middle Mouse Button

You can use the scroll wheel and middle mouse button in Falcon. By default, the wheel will control the FOV increase and decrease functions and the middle mouse button (or scroll wheel click) will set the FOV to the default value defined in the Falcon BMS.cfg.

*Note: this will only work if the mouse wheel is NOT mapped as an analogue axis!
In the Snap View Pit FOV adjustment does not work. If TrackIR Z Axis Vector Mode is set to FOV, or you have set the FOV Axis, changing the FOV with the mouse wheel will not work.*

The behavior of the scroll wheel and middle mouse buttons can be modified by adding the following lines to your Falcon BMS.cfg file:

- `set g_sScrollUpFunction "<command>"`
- `set g_sScrollDownFunction "<command>"`
- `set g_sMiddleButtonFunction "<command>"`

<command> is the name of the function you wish to execute. You can find a complete list of all available functions either in the key files or in the BMS Key File Editor.xls. Each increment of the scroll wheel will cause the corresponding command to be executed once.

Default Values

- `set g_sScrollUpFunction "FOVDecrease"`
- `set g_sScrollDownFunction "FOVIncrease"`
- `set g_sMiddleButtonFunction "FOVDefault"`

Note: this is the default behavior of the mouse wheel, so you will not find them in the Falcon BMS.cfg file.





8.1.1 How to change default FOV settings

You can set up the FOV behavior by adjusting the default values in the Falcon BMS.cfg file.

The following three settings are relevant:

- *set g_fDefaultFOV 60*
Changes your default field of view (FOV) setting. In the Config Editor you have options from 45° to 80° available in 5° offsets. Default is 60°.
If you want to have other than that, you need to change it manually in the Falcon BMS.cfg file using a text editor. The default FOV should not be greater than the *set g_fMaximumFOV* setting.
- *set g_fFOVIncrement 5*
Sets how much the field of view should change for each keypress / mouse wheel step in degrees. Default is 5° steps. You could also define 1° degree steps for finer tuning of the FOV or use bigger steps like 20°. To change this you must edit the Falcon BMS.cfg manually. Bear in mind that FOVIncrement values greater than 5° could prevent you from reaching the minimum FOV of 5°.
- *set g_fMaximumFOV 80*
This limits the maximum amount that the FOV can be increased.
You also have to change that value manually in the Falcon BMS.cfg if desired.

Note: *The higher the value, the more distorted the view is, especially above 100°
You can set FOV values between 5° and 180°.*

Here are some examples:



60° FOV (default)



80° FOV (max default)



45° FOV



100° FOV



180° FOV





8.2 View panning with the mouse

When mouselook is enabled (Clickable 3D Cockpit Default inactive and TIR inactive) the mouse wheel button has specific functions:

When the mouse wheel button is held you can move your head up / down, left / right and forward / backward. This only works if TrackIR is disabled and the 3d cockpit supports 6DoF. Mouse head translation only works in 3d pit Pan view; it does not work in 3d pit Snap view, padlock or other views.

When the mouse wheel button is depressed and held you have the following options:

- Mouse movement left / right = Pilot Head Movement left / right
- Mouse movement fwd / bwd = Pilot Head Movement forward / backward
- Mouse wheel up / down = Pilot Head Movement up / down

8.3 Touchscreen Use for Cockpit Interaction

Correct touchscreen usage is currently not possible in BMS as the input system uses *relative* mouse input only. Touchscreens need *absolute* coordinates for proper handling. If used as a secondary monitor to simulate cockpit panels you risk invoking functions in the 3d pit by accident.

To prevent this, new keystrokes have been added to enable/disable mouse buttons in the 3D cockpit entirely. This offers the possibility to avoid unwanted mouse clicks in non-exclusive mouse capture mode, e.g. for touchscreen users. The new keystroke names are:

- OTWMouseButtonsIn3dToggle: *ALT 3*
- OTWMouseButtonsIn3dEnable
- OTWMouseButtonsIn3dDisable

8.4 Custom Views

In addition to the pre-defined guided snap views you can also create custom views.

These custom views are relevant to the default 3D cockpit view (3) only. They will not work in the old 2D pit which is now the 3D snap view cockpit (2) or the HUD only view (1).




These define fixed camera views located inside the cockpit which can be focused on the MFDs, the ICP, the fault display etc. Unfortunately, you have to do some editing by yourself.





8.4.1 Capture View Position

All you need to do is move the view in the cockpit to the desired spot and open the chat box. The chatline commands `'.guidedviewdump'` or `'.gvd'` now create a `Guidedview.txt` file in the `\User\Logs` folder. If this file exists, new entries are appended.

 <code>ErrorsInClassTable.txt</code>	25.07.2015 14:22	Textdokument	1 KB
 <code>ErrorsInSFX.txt</code>	25.07.2015 14:22	Textdokument	1 KB
 <code>GuidedView.txt</code>	25.07.2015 18:26	Textdokument	1 KB

Open the `Guidedview.txt` file with an editor. Below are two example code lines:

```
guidedview <id> 0.015883 0.040278 0.018326 1.458329 -26.396080 0.007139 20.000004 <ndir> <nid> "
```

```
guidedview <id> 0.000000 -0.000000 0.000000 1.108329 -21.909981 -0.164438 30.000002 <ndir> <nid> "
```

Copy the **bold** part between `<id>` and `<ndir>`.

8.4.2 Custom view code syntax

The custom views have the following syntax:

```
customview pos.x pos.y pos.z ori.yaw ori.pitch ori.roll fov "comment" clickable
```

The underlined part must be replaced with the bold part you've just copied. The word "comment" can be replaced by a description of the view. Replace "clickable" by either "0" (= no) or "1" (= yes)

```
customview 0.015883 0.040278 0.018326 1.458329 -26.396080 0.007139 20.000004 "Right MFD" 1;
```

```
customview 0.000000 -0.000000 0.000000 1.108329 -21.909981 -0.164438 30.000002 "RWR ICP DED" 1;
```

Make sure every code line ends with a semicolon `<;>`, otherwise it will not work. You also have to take care to use the correct quotation marks.

```
" <- correct / incorrect -> "
```

8.4.3 Edit 3dckpit.dat files

The custom view code lines have to be added to the `3dckpit.dat` files, located in the `/Data/Art/Ckptart` folder. Here you find different F-16 subfolders which all contain the `3dckpit.dat` files. As we have multiple F-16 versions you have to edit all corresponding files. The custom views will only work in the pits you have edited.





Here are two examples how the above custom views look like in the pit:



Right MFD



RWR ICP DED

Note: Custom views do not work in Snap (2D) pit or HUD only / EFOV views. You have to be in the Pan (3D) pit mode.

8.4.4 Custom Views in 3d

The following callbacks are available to switch between the different custom views:

- OTWToggleCustom3dPitView: *CTL 6*: Toggles between custom view on and off.
- OTWNextCustom3dPitView: *ALT 7*: Switches to the next custom view.
- OTWPrevCustom3dPitView: *ALT 6*: Switches to the previous custom view.

8.5 How to define TopGun Views

The TopGun views are defined in the .dat files located in *Data/Sim/Acdata*. You have to edit the .dat file (*note: not the _afm files*) manually for each aircraft for which you want to change the TopGun view.

A typical code line looks like this:

```
TopGunCamera1 7 -4.72 -15.63 -4.40 +0.00 +45.00 +0.00 60.0
```

The TopGunCamera views are numbered. So the first view is TopGunCamera1, the second TopGunCamera2 etc. The "7" behind always stays the same. Besides the numbering you have to change the bold part of the code line.

x-offset y-offset z-offset x-rotation y-rotation z-rotation FOV

Note: There is no easy way to get the coordinates etc. You have to find them out by trial and error. You can use the standard TopGunf views as a starting point.





8.6 Toggle HUD Rendering (alt c : h)

A new callback (OTWToggleHUDRendering) has been added, which will toggle rendering the HUD. It works only in the HUD only, Snap (2D) Pit, Pan (3D) Pit, Empty Cockpit Shell and EFOV views. The Cockpit Display Extraction (external window rendering) is not affected by this.



HUD only w/o HUD rendering



Pan (3D) Pit w/o HUD rendering

8.7 Displays

8.7.1 Infobar (CTL 1)

In external views an infobar is shown on the bottom of the screen. It contains information about the object (aircraft, naval vessel, ground unit or weapon) the view is focused on. The infobar is displayed by default. It depends on your settings in the UI Simulation tab. You can also toggle it on and off by pressing “*Ctrl 1*”.



Note: If you toggle the infobar in 3D your last setting when leaving the 3D world will be applied to the UI setting as well. For example, if the infobar was activated in the UI before the flight and in 3D you turn it off and exit the sim the UI setting will be deactivated then. Depending on the focused object the infobar contains different information.

Aircraft will have information about callsign (Own AC = Pilots callsign - friendly aircraft = Callsign - Enemy aircraft = Type of aircraft), heading, altitude, speed, RPM, Gs and AOA.

CHASE CAMERA: Viper - heading: 275 deg, alt: 5903 ft (BARO), speed: 315 kts (IAS), rpm: 71%, 0.8G, aoa: +4.2 deg

For Ground Units only Friendly or Enemy Ground Unit Camera messages are displayed.





8.7.2 Radio Subtitles (CTL 2)

By activating this feature it is possible to display the radio messages subtitles.

This feature is exclusively activated/deactivated in the configuration screen; however it can be momentarily toggled by pressing the “ToggleSubTitles” key.

Please note that audible messages may 'lag' behind the displayed ones

Messages will be displayed for 10 seconds and up to 10 messages will be displayed at the same time. As more messages are displayed the more recent ones are added at the bottom of the display and move their way up as the older messages get removed. You can configure both the 'time to live' (TTL) and the maximum number of displayed messages by editing the `g_nSubTitleTTL` and `g_nNumberOfSubTitles` options in the Falcon BMS.cfg file. See the BMS Technical Manual for more information.



Messages are displayed in different colours, which indicate the radio channel where they originated from. The colours are user editable through config options lines and are explained later in this manual.

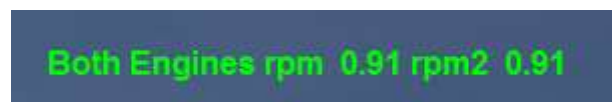
8.7.3 Flap Display (CTL 3)

When flying aircraft that do not have automatic FLAPS (LEF & TEF) the position of the flaps will be displayed in the top right corner of the screen by default. You can toggle these OFF and ON with the relevant keystroke.



8.7.4 Engine Display (CTL 4)

When flying aircraft multi-engine aircraft each engine RPM will be displayed in the top right corner of the screen. You can toggle these OFF and ON with the relevant keystroke.





8.7.5 Show Score (ALT c : d)

This function shows you the score in Dogfight module or a Tactical Engagement.

In a TE you get points if you destroy specific targets (if Victory Conditions have been set by TE creator). In Dogfight you get points for shooting down other aircraft.



Score in Tactical Engagement



Score in Dogfight

8.7.6 Frame Rate (ALT c : f)

If activated the frame rate overlay will be displayed in the upper left corner of the screen.

```
FPS: 101.5, 9.85 ms, Mem 85.92 MB, Cmd 2521  
Sim: 111.4, 8.98 ms (Draw 8.98, Expt 1.07, Wait 0.00)  
Ren: 252.3, 3.96 ms (Exec 3.89, Prsnt 0.07, Res 0.0026)
```

- FPS: The number of frames per second presented to the monitor and the duration of a frame.
- Sim: Theoretical FPS of the simulation thread based on its duration.
- Ren: Theoretical FPS of the render thread based on its duration.
- Cmd: The number of draw calls/commands created by the current scene. If this number is high, performance will be lower.
- Expt: Duration of the RTT export. This, added to the Sim duration roughly equals the overall FPS duration.

If you want e.g. 120 FPS, then SIM, REN and FPS must be above 120 FPS. It is limited by the slowest running part. If e.g. REN is only running at 50 FPS, you will never have more than 50 FPS overall.

Usually, your FPS will be a bit smaller than SIM and REN because there is still a synchronous part running between when SIM and REN are finished and until the frame is actually presented on screen. This includes things like the RTT export and virtual universe update.

8.7.7 Online Status (ALT c : o)

If activated, you can see a list of all users who are currently connected to the server in the top left corner of the screen. The number in the bracket is the consecutive numbering. The Host of an online game is always number one. The second part shows the pilot's callsign as defined in the Logbook.

```
Pilots:  
1:Viper (3D) in_sum:0kbps out_sum:0kbps
```

The third part, which is also set within brackets, shows the current status of the pilot. We have different status messages which are as follows:





- (UI): A pilot has connected to the server and the online game and is currently in the UI.
- (UI>3D): A pilot is entering the 3D world.
- (3D): A pilot is in the 3D world and in the cockpit.
- (3D>dead): A pilot has either ejected or his jet has crashed / exploded.

We have also some subtitles in relation to online status changes:

<Callsign>: (**is committing now**)

<Callsign> **joined as** <FlightCallsign&SlotNumber> **at** <FalconTime>

<Callsign> **exited from** <FlightCallsign&SlotNumber> **at** <FalconTime>

Note: The "exited from" message does NOT appear if a pilot has ejected or has been killed. You also get no message if a pilot enters a server (from desktop to UI) or leaves the server (from UI to desktop).

The OnlinePlayersDisplay 3D overlay also includes live network connection information:

- incoming bw in kbps
- outgoing bw in kbps
- ping in ms
- bandwidth limit (if applicable)

```
3D) in:24kbps out:50kbps ping:34ms  
3D) in:25kbps out:50kbps ping:74ms  
) in:30kbps out:50kbps ping:46ms (LIMITED)  
(3D) in:24kbps out:50kbps ping:48ms
```

The info is not calculated by BMS, but directly taken from the transport layer, i.e. it is accurate.

If you notice no "LIMITED" indication during game play (in brackets at the end of the line), all is fine. In case a client has insufficient bandwidth for a smooth mp experience the "LIMITED" overlay will occur, indicating that BMS msg sending is now hitting the client limits.

Also note that once "LIMITED" is shown, the ping times reported for these clients will also increase. This is an artefact of the RakNet congestion control, as hitting the BW limit simply means that messages queue up and take a while to be processed. Unfortunately, this is true for the ping measuring round trip messages as well. So you will only see the actual physical ping time as long as you don't hit the hard BW limit on any given connection.

Note: We have a new config option `g_bOnlinePlayersDisplayDefault` (default OFF/0) which specifies whether the OnlinePlayersDisplay overlay should be ON or OFF by default when entering 3D.





8.8 Pilot kneeboards

If BMS 4.34 introduced the pilot model, the legs of the pilots only had single pages kneeboards. BMS 4.35 introduced the possibility to toggle 16 pages on each kneeboard for a total of 32 pages.

Any images can be displayed on the kneeboard by way of DDS textures located in the \Data\TerrData\Objects\KoreaObj folder. Reserved DDS are from 7982.dds to 7997.dds

By modifying the DDS you may add any suitable content to the kneeboards. Each texture has a resolution of 2048x2048 pixels and is divided into 2 parts. The left part is for the left kneeboard and the right part for the right kneeboard, respectively.

For now, the old single page DDS textures 1988.dds & 8019.dds are still available for compatibility reasons, e.g. they are used by 3rd party pilot models. Please note that this can change in the future.

By default the BMS 4.35 kneeboards have checklists on the left and navigation data on the right kneeboard. The first pages of each legs are titles or a large map which are mostly placeholders for more relevant information that can be updated through Weapon Delivery Planner.

8.8.1 Managing Kneeboards with the latest reslease of WDP

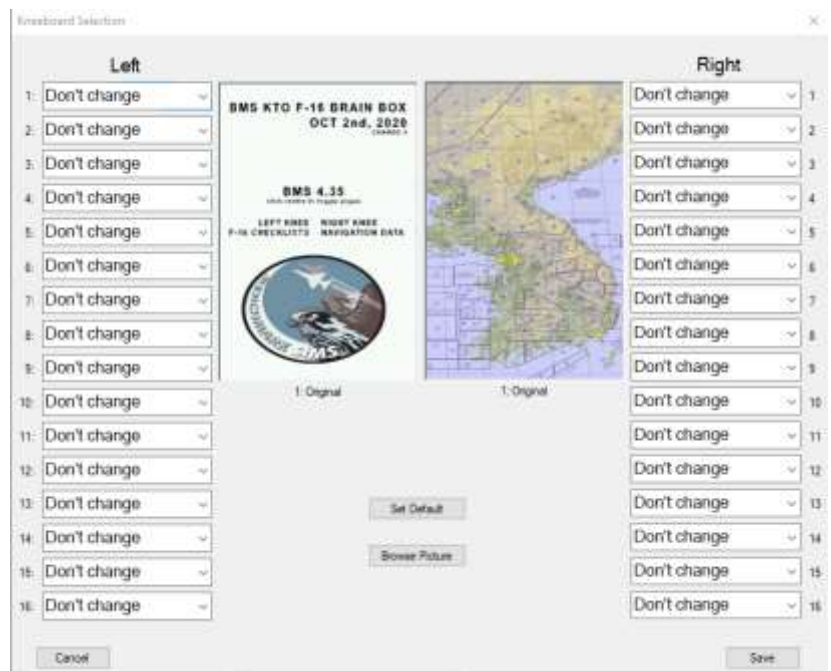
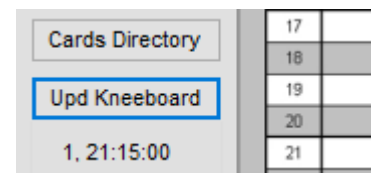
After loading the mission into WDP – make sure you have the latest version supporting BMS 4.35, select your flight and manage your datacards as usual. Once you are happy with the mission planning, you can transfer whatever content you need into your pilot leg kneeboards, here is how:

On the left margin of WDP, click the "Upd Kneeboard" button.

This open a new window displaying 16 slots of the left and 16 slots on the right representing the current kneeboards content. (left for left kneeboard and right for right kneeboard) The selected page on the left and the right are visible as images in the middle.

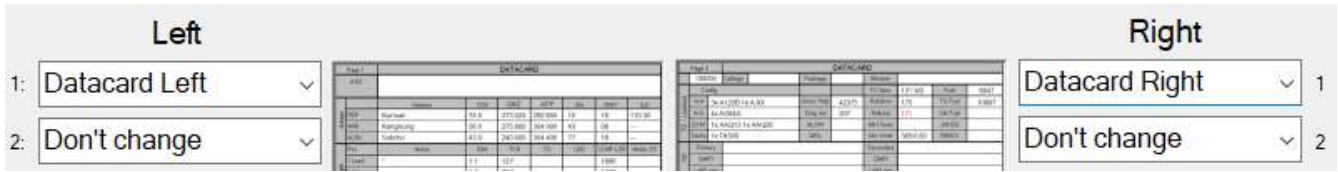
As stated above the first pages of the right and left default kneeboards are meant to be replaced the mission datacards which is what you will need the most. You may replace the other pages as well as you see fit but you will then loose the checklists and the navigation data offered by default.

Let's assume you want to add your mission datacards as advised on the first page. The WDP datacards are split into left and right side.





Select in the first left slot datacard left and select in the first right slot datacard right.



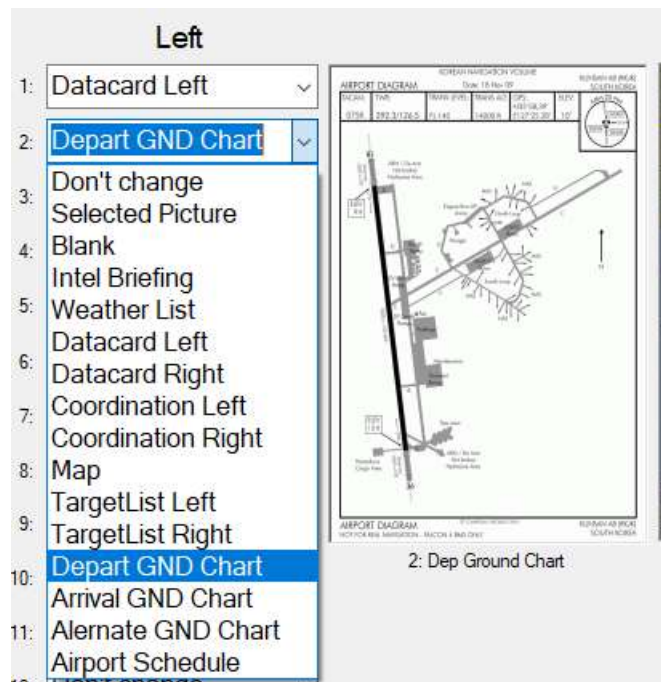
Since you do not want to change any other page, you may hit the SAVE button on the bottom right. This will override the relevant DDS with the correct images. Your mission datacard are now saved into your kneeboards.

Please note, the DDS will remain so setup until you change them again. So if on your next mission you did not upload new content or reset default content on your kneeboards, this specific datacard will remain loaded.

All 32 slots can be changed according to your needs. WDP offers a selection of possible replacements. Most of these are intuitive but some need further information:

Blank: Inserts an "Intentionally Left Blank" picture with a page number. Blank pages are not skipped and will be displayed through any other page.

Map: inserts the mission map as set in the map tab of WDP. Remember You may zoom and offset the KTO map within WDP allowing to have a more relevant map implemented on your datacard and thus also on your kneeboard. Also note that you have many options to display on that mission map from WDP such as other flight routes, PPT, airbase, bullseye, tacan and range ... You even have the option to select a white map for better readability.



Selected picture: Any picture can be loaded on any kneeboard page.

The picture must be first selected through the "Browse Picture" button. This opens a regular file browser where the relevant picture can be chosen (jpg, png, bmp and dds are supported formats).

Please note the image will be resized to 1024x2048. If you do not want to have weird size effect, adapt first your images to the proper resolution.

Once selected the images will be displayed in the middle and you may insert it on the left or right kneeboard. It is not so intuitive, hang on! Before inserting the selected picture, you must first choose a page number from the middle box and only after this you can insert it with the right or left "Insert" button.

By default the page number defaults to page 1 so if you omit this step you will constantly upload that picture on the first page.

After successful insert, the preview will disappear making room for the next picture to be uploaded in any other slot. It is therefore possible to upload many different pictures to each page of your kneeboard.



When a picture has been successfully loaded into a slot, the slot will show 'Selected Picture' and highlighting that slot will preview the loaded picture in the middle.

It is advised to cross check your slots for relevant images before sending the files to the kneeboard as it is very easy to get confused during the process and once in the pit, it is too late to further change a picture.

Unless you really want to have a picture of your family in case you do not come back from this mission (...) some squad specific SOP or cheat sheets can be handy to have on the kneeboards.

Once all your 32 pages are set as you want them, clicking the "Save" button will convert your selection into the relevant DDS files.

The DDS files will be overwritten (unless those left at "Don't Change") and default content will be lost...

Yes I know there is a "Set default" button but this is misleading as it does not reset the default vanilla BMS install DDS file but only the ones which were active before launching WDP. Once you have saved your kneeboard configuration and closed WDP, what you have saved will be the next default.

It is therefore recommended to backup the default 7982 to 7997 DDS files from your \Data\TerrData\Objects\KoreaObj folder to ensure you can restore them whenever needed.





9. Aircraft Radios

9.1 Introduction

The radio code updates started in 4.32, were further refined in 4.33 and a huge step was achieved in 4.34. No further changes have been made in 4.35. Let's have a bit of history:

The radio in Falcon4 BMS underwent significant changes in BMS 4.32 compared to earlier dialects of Falcon 4.0. These changes included differences in both the single-player and multiplayer environment.

4.33 introduced a couple of new features to the 4.32 radios. One big addition is that, as with real life radios, we are now limited in range. Radio range (including IVC) is now Line of Sight dependant. As you get further away from the other radio the quality and volume of the transmission will degrade until you can not understand, then no longer hear the other person's transmission. The other change is the possibility of using sidetone (if your hardware supports it), for an even more authentic audio experience.

4.34 introduced unique frequency according to callsign and the need for humans and AI alike to be on the same frequency to be able to communicate. More frequencies were added for controlling agencies like ATC and AWACS but most importantly AI no longer follow human frequency changes auto-magically.

With 4.34, just like real-life, if you are not on the correct frequency and issue a radio message, the message will be lost and nobody will answer.

9.2 UHF

The UHF band frequencies are from 225.000-399.975 MHz in 25 kHz stepping. Since 25 kHz stepping applies, the 5th digit of a 6-digit freq must end with 0, 2, 5, or 7 and the 6th digit must end in 5 or 0.

Examples of valid freqs: 371.075, 377.10, 271.6. Examples of invalid freqs: 339.11, 271.14.

It should be noted that the pilot inputting frequencies using 6 digits will only actually see 5 digits in the DED. With 4.34 this will happen far more often as we use many more frequencies in a limited range.

9.3 VHF

The VHF band frequencies are from (AM mode) 116.000-151.975 MHz in 25 kHz stepping. Please note BMS "own valid" range (for voice communications) is from 118.000 to 144.000 MHz.

The FM mode (30.000-87.975 MHz) is not implemented. The rules regarding digit input are the same as those which apply to the UHF band.

A large part of the VHF range is reserved for ATC and navigation and cannot be used for radio communication. ILS frequencies are within 108 – 111.975 MHz and RadioNavigation stations are from 112.0 to 117.975 MHz. ATC also uses the range between 118.0 to 135.975. That normally leaves 136.0 to 144.0 in BMS for radio communications (real life is possible up to 151.975).





9.4 What about the AI? How do they fit in?

Well it is much simpler than before.

Before 4.34 we had to know that AI members of your flight would be able to hear you and communicate back on any one of frequencies which were assigned to presets 1-14. AI package members would only hear you on the frequencies that corresponded to Package 1-5. AWACS and tankers were “smart” like your flight members. They could communicate on any of the 14 well-known default frequencies. Proximity would limit transmission to jets close by.

Now to be able to communicate with any entity you need to tune the frequency of that entity. There are no presets anymore for flight, package, team or whatever. Each entity has its own frequency and like in real life if you want to talk to them you must know and tune to that frequency in your radio.

This will be particularly noticeable the first time you try to manage your AI wingmen. If you are not on the assigned VHF frequency for your flight your wingmen will simply not hear you. As radio checks are not possible in BMS, just ask for a fuel check (or similar) and if you don't get a response chances are that your VHF is not tuned to your default flight VHF frequency.

Default VHF flight frequency presets have changed. According to the position of the flight in a package, the presets will be either #15 for the first flight (ID10), #16 for the second flight (ID20), #17 for the third flight (ID30) and you get the idea what it will be for the fourth and up to the possible fifth flight in that package. In solo play with a single flight not part of a package the default VHF preset will always be #15.

The UHF Package frequency is the assigned UHF frequency of the leading flight in that package. That leading flight is assigned ID 10. The IDM addresses and the Air to Air TACAN default channels still depend on the ID of the flight within its package. The UHF radio frequency of that package will now be assigned as the ID 10 flight UHF frequency. This is pretty transparent for users except for some very specific cases.

When a tanker package is created for instance and made up of a tanker flight and an escort flight if the tanker is the first flight in the package the package UHF will be the tanker frequency. When the user tunes to the tanker frequency prior to an air to air refuel, he will therefore hear all the escort flight UHF radio transmissions on the tanker frequency. To avoid this, the tanker when part of a package should never be assigned as the first flight; the escort should be the first flight. That way, the escort flight UHF is the primary radio and the tanker will have its own frequency, dedicated to AAR. The same is true for AWACS, to an even greater extent.

Another consequence of the new radio code is that in solo player the frequency might be much quieter than before. As you will not hear magically all the radio transmissions from multiple frequencies around you anymore but only the frequencies you are tuned to, the radio traffic will seriously decrease from previous missions. This is a double-edged sword and something future revisions of BMS will hopefully address.

Less traffic makes it easier to get a clear picture of what's going on around you without the interference from engagements happening far away from you as it was in previous versions, but on the other hand sometimes the radio traffic is so sparse on your own frequency that it's far from being enough to build a good situational awareness about what is going on around you with other flights being on their own dedicated frequency. Fortunately the new AWACS Picture calls are significantly better at helping you build/maintain SA.





9.5 IVC impact on the new Radio Code

Previously the Team frequency preset (channel 13) was a special one.

When using IVC in multiplayer from the game UI (i.e. before or after flight), the *F1* key transmitted on the Team frequency. In a force-on-force scenario, red and blue forces could (and still can) set different Team frequencies in their *Falcon BMS.cfg* file (described in chapter 12) to allow communication in the user interface (UI) with ONLY members of their team. This allowed them to brief/review their plan before committing to 3D without the other team listening in. Another feature of this is someone (a human AWACS controller or shot down pilot) in the UI could communicate with someone in the 3D world over the team freq (provided the person(s) in the 3d world are tuned to it).

To maintain the *F1* capability to communicate from the UI to the 3D world with the deletion of the TEAM channel, the *F1* default frequency was changed to a new preset common to most of the flights and named “Advisory”. It’s preset 14 and assigned to the fixed frequency 339.750 MHz.

Naturally, if both sides (teams) do not have different Team freqs, everyone will hear one another as if they were on the same team. In a MP environment, *F2* is by default ONLY used for the UI and *everyone* in the UI can hear transmissions on it. By default *F2* can never be heard from the 3d world because it uses a frequency outside the range you can select once you are in the jet. *F2* is intended for everyone to use to check their IVC when joining first chat (the COMMS window), to coordinate and synchronise launching to 3D, briefing any “global” type things like rules of engagement, any special procedures, or just BS’ing before you fly.

It is possible to change the frequency that is used for *F2* from the UI, so if you want you can reconfigure it so both *F1* and *F2* are tunable in the 3D world as well. Depending on your needs you have considerable flexibility to separate teams or have multiple “control agencies” on separate frequencies or the like.

9.6 Managing two radios

Typically in the real F-16 UHF is normally used to communicate with external agencies: ATC--ARTCC, ground, towers, approach/departure, etc. and in war-time or exercises, controlling agencies like AWACS (DCA, Strike, etc), JSTARS or other command and control agencies.

VHF is normally used for intra-flight communications but some frequencies may double as coordination net as well depending on the purpose. What does this mean to you?

Since the above are the typical set ups, we recommend talking to Towers, AI packages, or humans in other flights or packages on UHF and keeping intra-flight communications on VHF.

So, in single-player your radio set up may be:

- UHF 2 (preset 2 for Departure airbase Ground frequency) and change UHF when instructed to the following controlling agencies (tower, departure, tactical net).
- VHF 15 (the new first flight default intra-flight preset).

In a multiplayer (MP) environment, with say 2 flights of 4 aircraft in the same package the UHF would be the same as all aircraft need to be on the assigned ATC frequency (ground with preset #2 in this case) but having all the 8 players on the same VHF frequency would not work efficiently. The first flight would be on VHF preset #15 and the second flight would be on VHF preset #16.





That way each flight can have intra-flight comms on the VHF flight frequency and the 2 flight leads can talk to each other using the UHF frequency (usually after being released from the ATC).

Worth mentioning also is that 4.34 finally implements concurrent UHF/VHF radio playback. In other words, just like real life, you may hear transmissions from both radios at the same time. Two stations emitting on the same radio frequency will be blocked, but as your aircraft has two separate radios (UHF and VHF) if you receive transmissions from both at the same time, you will hear both at the same time, unlike previous versions where they were sequenced one after another or partially deleted and not heard.

During internal testing, it turned out that it is **really hard** to listen to two independent radio transmissions at the same time, especially if both are using the same voice (which can happen in BMS), and are co-located acoustically. A new config option: `g_fRadioBalance` has been introduced to shift the UHF/VHF playback volume out of centre (in opposite directions). Default is 0.0, i.e. both UHF and VHF will be centered. A positive value will shift UHF to the right, VHF to the left. A negative value will shift UHF to the left, VHF to the right. We advise you to set it to 4 (or -4) to start with and then adjust to your preference. More information can be found in chapter 13.9. Please note this is only valid for true VHF or UHF comms, ATIS which is TTS but VHF to BMS flyers is actually from the centre and cannot be side shifted.

9.7 Setting the UI Radio Frequencies

To change the team frequency, open up the *Falcon BMS.cfg* file in the root Falcon4 directory with a *plain text editor* like Notepad or even better Notepad ++.

Look for the following entry: `set g_nF1TeamUiFreq 339750`

As you can see, referencing the default frequency table above, 339750 is 339.750 MHz. That number is the default if `set g_nF1TeamUiFreq` is not set in the .cfg file. To change the Team freq, add a 6-digit number that complies with the rules mentioned in the UHF section regarding frequencies or keep it simple.

Good examples: 236800, 377800, 253700, 229025, 141325, 139000, 143925.

The team freq can either be a valid UHF or valid VHF frequency.

IMPORTANT NOTE: The team freq *does not* have to be a valid UHF or VHF frequency, but there's a catch. It *can* be any 6-digit number, but we *strongly recommend* using a valid UHF/VHF frequency, as a pilot in the 3D world will only be able to tune to a valid frequency. So while the number you choose will work for the UI, it will not for the 3D world. So just ensure you are using a valid frequency and be done with it!

The same configuration trick works for the *F2* frequency as well, so you can change it as well if you want. In this case the name of the variable to set the frequency is: `g_nF2TeamUiFreq`.



PART 2

FOR THE ADVANCED USER



10. Key Files

This chapter shall give you a basic understanding of how key files work. All available and necessary information are part of this document. If you follow the instructions you should be able to create your own key file without any major obstacles.

10.1 Categories & Sections:

We divided the key file into different categories and sections. We have to admit, it has been done before. The difference is that you can see the section names in the key file. While scrolling down (this happens sometimes too quickly to find a specific function easily) you will now have some eye catchers.

The categories and sections are meant to act like a headline. Even in the newspapers you will look for the headlines in the first place. Each category has a group of sections. As an example the LEFT CONSOLE is the category of the following sections:

TEST PANEL, FLT CONTROL PANEL, MANUAL TRIM PANEL, FUEL PANEL etc.

10.1.1 Category and Section code lines

The categories and sections are easily recognizable in the code:

Key File Description (Key file content / UI output):

SimDoNothing -1 0 0XFFFFFFFF 0 0 0 -1 "BMS - Full"



The first key file code line, if set to -1, will be displayed as shown above. This is the description of the file to see at a glance, which file is loaded.

Category example (Key file content / UI output):

SimDoNothing -1 0 0XFFFFFFFF 0 0 0 -1 "1. UI & 3RD PARTY SOFTWARE"



Section example (Key file content / UI output):

SimDoNothing -1 0 0XFFFFFFFF 0 0 0 -1 "===== 1.01 UI FUNCTIONS ====="



The categories and sections are set to value -1, so they are visible in the UI, not changeable and with a blue background color.





Separators:

In the key file itself we use the following lines for easy navigation:

```
#=====
```

They are set before and after each category and section. They are not shown in the UI, because they are commented out. Unfortunately you will lose all commented stuff (with a # at the beginning) once you save the file in the UI. So it is recommended to keep a backup of the original file.

But after a while you should be able to navigate the key file easily without these borders. This feature is just for those who are not familiar with editing keystroke files.

10.1.2 Non category & section lines in the UI:

Non changeable keys example (Key file content / UI output):

```
SimDoNothing -1 0 0FFFFFFF 0 0 0 -0 "REM: Hardcoded (not changeable)"
```



Changeable keys example (Key file content / UI output):

```
SimRadarElevationUp -1 0 0FFFFFFF 0 0 0 1 "TQS: ANT ELEV Knob - Tilt Up"
```



Invisible keys example (Key file content / UI output):

```
CommandsSetKeyCombo -1 0 0X2C 2 0 0 -2 "Key Combination"
```

<- no screenshot here, of course ☺

The UI will look like this:

No Key Assigned	BMS - Full
	1. UI & 3RD PARTY SOFTWARE
	===== 1.01 UI FUNCTIONS =====
No Key Assigned	REM: Hardcoded (not changeable)
F1	UI: IVC Broadcast (Global Comms to 2D & 3D)
F2	UI: IVC Local (Comms only to 2D)
Escape	UI: Exit Sim - Leave Menu - Abort
Sys Req	UI: Screenshot (See also section 6.06)
	===== 1.02 3RD PARTY SOFTWARE =====
No Key Assigned	REM: This is just for reference. See manual.





	===== 2.01 TEST PANEL =====
Shift F1	TEST: FIRE & OHEAT DETECT Button - Hold
Shift F2	TEST: OXY QTY Switch - Hold
Shift F3	TEST: MAL & IND LTS Button - Hold
No Key Assigned	TEST: MAL & IND LTS Button - Release

The categories and sections follow basically the arrangement made by Red Dog. So you will start in the pit at the rear left side, go to the front and then to the rear right side. But there are some differences. We sorted the callbacks in the sections and the sections itself more logical and gave them more correct names.

For more info about the categories and sections see the overview farther below.

10.2 The Terms:

Due to the reason that the description line is limited to 37 characters, we had to accept some compromises. But in most cases you will find the description as it was meant to be.

The description has been divided into three separate terms: 1. Term: 2. Term – 3. Term

1. Term: Short form of the section name followed by a colon.
2. Term: Correct designation of the switch / button / wheel / knob etc. followed by a dash.
3. Term: Correct designation of the positions / states of a switch / button etc., or: Short description of the function

Examples:

- AUX: CNI Knob – Toggle
- GEAR: HOOK Switch – DN

In some cases it does not make sense to divide the description into three terms. So you will also find descriptions with only two terms. Then it will be like this: 1. Term: 3. Term

Examples:

- RADIO: AWACS Menu
- SIM: Exit Sim





10.2.1 First Term:

As mentioned above, the 1. Term is a short form of a section name. Sometimes it is easily done. For the TEST Panel for example it is just TEST. But you have some sections where you have to be careful with naming. EXT LIGHTING panel (LEFT CONSOLE) and LIGHTING panel (RIGHT CONSOLE) for example. The term for EXT LIGHTING is „EXT“ and for LIGHTING it is „LIGHT“ in this case.

If possible, this term has no more than 4 or 5 characters. But sometimes they have more.

Each first term is unique in the key file and describes only one section. So it is possible only to look at the first term while scrolling down the key file in the UI. You will easily find what you need.

10.2.2 Second Term:

The 2nd term describes a switch / button / wheel or knob. In most cases you will find the correct designation as it is inscribed on the panel.

Examples: HSI CRS, 2-ALOW, MAIN PWR...

It is followed by the type. The types are:

- Buttons: e.g. ICP buttons on the ICP
- Switches: e.g. MASTER ARM Switch on MISC panel
- Wheels: e.g. PITCH Wheel on the TRIM panel
- Knobs: e.g. ENG FEED on FUEL panel
- Handles: e.g. EJECT Handle

10.2.3 Third Term:

The third term describes a function or the positions / states of a switch, knob etc.

For cockpit builders, you will often have things like „ON“, „OFF“ etc. But there are other functions of course (e.g. Night vision on). To distinguish between different functions we used the following designations:

Push (for buttons):

Pushing a button describes a single action.

(e.g. ICP Buttons)

Hold (for buttons and switches):

As long as you hold the button or switch, it is active. If you release the button or switch, it is inactive.

(e.g. EPU GEN Switch)

This one is also for functions which need a long input to become active. E.g. the EJECT Handle

Release (for buttons):

Release action for pushbuttons (e.g. MAL & IND LTS Button)





Toggle (for switches, knobs, buttons and functions):

Toggles through two (!) states of a switch, knob, button or function. Toggle forward and toggle back (E.g. ON / OFF).

Step Up / Down (for knobs, wheels & switches):

This is meant to step between 3 or more states of a switch, function etc. Stepping up brings you to the last state and ends there. Vice versa for step down.

(first position/ON – AUTO – OFF/last position) & (last position/OFF – AUTO – ON/first position)

Cycle Up / Down (for switches and knobs):

It cycles a switch position up. When in the last position, it jumps automatically to the first position and so on. (Vice versa for cycle down)

(ON – AUTO – OFF – ON – AUTO...) & (ON – OFF – AUTO – ON – OFF...)

Cycle (for switches and knobs):

Same as above but only in one (!) direction. You cannot cycle in the opposite direction, because there is no callback for it.

(ON – AUTO – OFF – ON – AUTO...)

Increase / Decrease (for knobs and wheels):

Incr. / Decr. is only used for knobs and wheels which change brightness, volume, degree values or pressure. It is also used for FOV.

(Increase Brightness / Decrease Volume...)

States / Positions (for knobs & switches):

This is the pitbuilders playground. You will often find stuff like ON, OFF etc. There will be no further explanation of what this switch is used for. It names only the specific state.

Function:

Describes a specific function, e.g. Sírn Exit. Sometimes we had to keep the description short. So there are some short forms. These are:

Tog. (Toggle)

Cyc. (Cycle)

Inc. (Increase)

Dec. (Decrease)

Dn (Down)

Btn. (Button)





10.2.4 Upper case vs. lower case:

All categories and sections are written always in upper case. (LEFT CONSOLE, VIEWS...)

The first term is always written in upper case. (TEST, OXY...)

If the second term refers to a switch / button etc. in the cockpit, it is also written in upper case. (MAIN PWR / PARKING BRAKE...)

If the positions / states are referring to a corresponding switch / button etc in the Pit, it is written in upper case. (ON / OFF / **UP** / DN...)

All other words only begin with a single upper-case letter. (Increase / Toggle / **Up**...)

So what is the difference between **UP** and **Up**? It's quite simple. Everything you can read in the Pit is written in upper case completely.

10.2.5 Keys vs. Mouse (right / left click, scroll wheel):

Nearly all wheels & knobs can be turned either using the mouse buttons or the mouse wheel (clockwise – left btn. or mouse wheel up / counterclockwise – right btn. or mouse wheel down).

For cycle, toggle and toggle up/down callbacks you have to use the right and left mouse button. (With some exceptions)

For pushbuttons you only need the left mouse button.





10.3 Overview categories & sections:

Category	Section	Short form
1. UI & 3RD PARTY SOFTWARE	1.01 UI FUNCTIONS	UI
	1.02 3RD PARTY SOFTWARE	3RD

Category	Section	Short form
2. LEFT CONSOLE	2.01 TEST PANEL	TEST
	2.02 ANTI G PANEL (n/i)	FLT
	2.03 FLT CONTROL PANEL	ANTI
	2.04 MANUAL TRIM PANEL	TRIM
	2.05 FUEL PANEL	FUEL
	2.06 AUX COMM PANEL	AUX
	2.07 EXT LIGHTING PANEL	EXT
	2.08 EPU PANEL	EPU
	2.09 ELEC PANEL	ELEC
	2.10 AVTR PANEL	AVTR
	2.11 ECM PANEL	ECM
	2.12 ENG & JET START PANEL	ENG
	2.13 AUDIO 2 PANEL	AUDIO2
	2.14 AUDIO 1 PANEL	AUDIO1
	2.15 MPO PANEL	MPO
	2.16 UHF PANEL	UHF
	2.17 LEFT SIDE WALL	LEFT WALL
	2.18 SEAT	SEAT
	2.19 THROTTLE QUADRANT SYSTEM	TQS

Category	Section	Short form
3. LEFT AUX CONSOLE	3.01 ALT GEAR CONTROL	ALT GEAR
	3.02 TWA PANEL	TWA
	3.03 HMCS PANEL	HMCS
	3.04 CMDS PANEL	CMDS
	3.05 GEAR PANEL	GEAR

Category	Section	Short form
4. CENTER CONSOLE	4.01 MISC PANEL	MISC
	4.02 LEFT EYEBROW	EYE
	4.03 TWP	TWP
	4.04 RWR	RWR
	4.05 LEFT MFD	LMFD
	4.06 ICP	ICP
	4.07 MAIN INSTRUMENT	MAIN
	4.08 INSTR MODE PANEL	INSTR
	4.09 FUEL QTY PANEL	QTY
	4.10 RIGHT MFD	RMFD





Category	Section	Short form
5. RIGHT CONSOLE	5.01 SNSR PWR PANEL	SNSR
	5.02 HUD	HUD
	5.03 NUCLEAR CONSENT PANEL (n/i)	NUC
	5.04 LIGHTING PANEL	LIGHT
	5.05 AIR COND PANEL	AIR
	5.06 ZEROIZE PANEL	ZERO
	5.07 KY58 PANEL (n/i)	KY58
	5.08 ANTI ICE / ANT SEL PANEL (n/i)	ANT
	5.09 AVIONICS POWER PANEL	AVIONICS
	5.10 OXYGEN PANEL	OXY
	5.11 FLIGHT STICK	STICK

Category	Section	Short form
6. MISCELLANEOUS	6.01 OTHER COCKPIT CALLBACKS	CKPIT
	6.02 SHORTCUTS	SHORT
	6.03 KEYBOARD FLIGHT CONTROLS	FCTRL
	6.04 EXTRA MFD (THIRD)	TMFD
	6.05 EXTRA MFD (FOURTH)	FMFD
	6.06 SIMULATION & HARDWARE	SIM
	6.07 WINAMP	WINAMP
	6.08 DEVELOPMENT	DEV

Category	Section	Short form
7. VIEWS	7.01 VIEW GENERAL CONTROL	VIEWGEN
	7.02 VIEW INTERNAL	VIEWINT
	7.03 VIEW EXTERNAL	VIEWEXT

Category	Section	Short form
8. RADIO COMMS	8.01 GENERAL RADIO OPTIONS	RADIO
	8.02 AWACS COMMS	AWACS
	8.03 ATC COMMS	ATC
	8.04 TANKER COMMS	TANKER
	8.05 WINGMAN COMMAND	WINGMAN
	8.06 ELEMENT COMMAND	ELEMENT
	8.07 FLIGHT COMMAND	FLIGHT

Category	Section	Short form
DirectX (Examples, depends on your settings, Note: Only Sections visible in UI!!!)	HOTAS UNSHIFTED	n/a
	HOTAS SHIFTED	n/a
	LEFT MFD	n/a
	RIGHT MFD	n/a

Remarks begin with the short form REM.

Sections labeled with (n/i) are currently not implemented.





10.4 Key files - general info:

As mentioned in the introduction, the key settings are not comparable to other common versions before. While removing systematically all old and outdated or not working callbacks, we realized soon that there are a lot of free keys available for mapping other callbacks. So we decided to remap almost everything.

Important callbacks which you will need often are easily accessible. Cockpit callbacks are grouped panel by panel. It is possible to perform a complete ramp start without using the mouse in the cockpit. The key settings are optimized for HOTAS owners, especially TM Cougar.

10.4.1 Deprecated & outdated / Dev callbacks:

Since 4.34 all outdated and deprecated callbacks have been removed. We have a bunch of new callback, though. Please refer to the “New Keyboard Commands” chapter later in this document.

10.4.2 TrackIR, Fraps, Vac & TeamSpeak:

A common weakness of any keyboard layout of modern games is the fact that they don't incorporate 3rd party software like TrackIR, Fraps, VAC or TeamSpeak. This issue is now solved, as these common tools are now taken into account by the Falcon BMS key files.

TrackIR uses some keys by default. These are e.g. F8 (TrackIR precision) or more important F12 (TrackIR recenter). These default keys along with BMS specific functions like TrackIR reload are incorporated into the key file.

Note: You can change the key for TrackIR Recenter at two different locations. The UI in BMS and the TrackIR UI. In case of mapping two different keys for recenter, both will be working.

Regarding VoIP software like TeamSpeak (should be the most common), you do not have default keys for PTT or broadcast functionality. These keys have to be set manually in the TS UI. Nonetheless we decided to implement them into our keyboard layout. You can see the key settings in our layout as suggestions. Of course you can map them to any key you want. Another solution is, to set them as DX buttons. In this case you have to assign them in TS settings. A proper place for them would be the same button on you HOTAS you have assigned VHF / UHF Comms to.

Fraps is a commonly used tool for video capturing. Unfortunately the default keys are colliding with the default keys of TrackIR. So you have to remap either the TIR or the Frap default keys.

Note: F9 (TIR Pause / FRAPS Video Capture) and F12 (TIR Recenter / FRAPS Overlay) are colliding. If you use both you have to solve it by reassigning these keys in one tool to avoid issues.





Newly implemented is the default key (F8) for VAC (Voice Activated Commands) Push to Activate (PTA). Please note that this key is in conflict with TrackIR (TIR F8 = TIR Profile). What is not incorporated to the key files is the default keys for VAC On/Off, which is Strg-End (USInt) by default. First it is also in conflict with other BMS related functions. Second, you will most likely just use one option (either PTT or On/Off).

Keep in mind that you have to set the keys directly in the software and NOT in BMS. Except for TrackIR reload and TrackIR Recenter, all key settings for TrackIR and TS are mapped to the callback SimDoNothing, so you can find them in both the key file itself and the keyboard layout.

10.4.3 The key file profiles:

There are five different key file profiles which use the same layout. They are using the key settings which can be found in the keyboard layout files.

Full:

This is the full version of the key file with all callbacks.

Pitbuilder:

This version is for pit builders. All toggle / cycle callbacks for switches / knobs which have full state callbacks are removed. The pitbuilders file has its own dedicated key assignments.

Basic:

This is the “light” version of the “Full” key file. All full state callbacks are removed. If you are not a pitbuilder and use cycle / toggle functions instead, this is the file for you.

Minimum:

This key file contains only a small number of callbacks which are essential plus some additional functions for comfort reasons. If you use the mouse in cockpit very often, this is most likely the right key file for you.

Blank:

This is the full key file without any key assignments (except hardcoded stuff, general comms and exit sim).

10.4.4 Notes on the new pitbuilder key file:

As mentioned above we have in fact two different keyboard layouts. The first one is for the average Joe Pilot, the second one is a dedicated to pitbuilders. The purpose of a different pitbuilders keyboard layout is obvious:

While non pitbuilders are used to interact with the cockpit either via mouse or by using toggle / cycle functions, the needs for a pitbuilder are different.

The main goal was to assign all currently implemented functions and taking a look into the future to take further developments into account right from the beginning. So what we have now is a complete set of all possible cockpit functions based on the F-16C/D Blk 50 cockpit layout. This makes sure, pitbuilders have a future proof keyboard layout at hand on which they can rely on.

But wait:

No promises are made whether new functions are developed or not. Although it is very likely, that most of the missing functions will be addressed someday, we cannot guarantee that all possible cockpit functions will be implemented.





10.4.5 Backup

When editing key files you will likely use the files in the User/Config folder. In case something went wrong you can easily replace the faulty key file with a good one from the BMS Cloud Storage.

The cloud is located on the Benchmarksims homepage -> Home -> Articles

10.5 The inner working of key files

If you are not familiar with editing key files you should read the following explanations of the inner working of key files (originally gathered by Dunc for Falcon AF, but revised for use with BMS).

10.5.1 Keyboard code lines

A typical key file code line looks like this:

```
AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
```

Each code line is composed of nine different parts.

Note: The parts are separated by a blank character (Spaces shown with a red underline here).

```
AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
```

What each part does in a code line is described below.

First part: The callback

```
AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
```

The first part assigns a specific BMS function to the code line. If you press the assigned keyboard key(s) it invokes the callback function and the callback related action will be executed. So in our example if you press "G" on the keyboard (0X22 = G) it toggles between gear up and gear down in the sim.

The callbacks itself are hardcoded and cannot be changed by the user. They could only be replaced by other callbacks, which do not make sense in a key file as we can simply assign other keyboard keys to this callback. (DX code lines are another story and are described later).

You can find a complete set of all callbacks which are safe to use in our _full key file. Callbacks which are not listed there are outdated / deprecated and should no longer be used as they will be most likely dropped in the future.





Second part: Sound ID

```
AFGearToggle 118 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
```

In the past the second part was used in 2D cockpits which we do not have anymore. With the release of FBMS a 4-digit sound ID could be assigned to invoke cockpit sounds when pressing the assigned key. The old 4-digit sound IDs were referenced in the 16_ckpt.dat.

These 2d pit button IDs (4-digit numbers) are now outdated and don't work anymore!

Now the second part serves as sound ID, which will be used directly from f4sndtbl.txt. It determines the KEY_DOWN sound to be played when activating the callback.

If you replace the number by -1 sound playback is disabled for that callback.

```
AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
```

More info can be found in the chapter **Why we do not hear cockpit sounds when pressing a key.**

Third part: Not in use

```
AFGearToggle -1 0 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
```

The third part is not used in the code line, as it is old 2D Pit stuff. In the past you could decide to use both, mouse and key binding (0) or only the mouse (1) in cockpit. Changes here do not have an impact. You should leave the 0 unchanged.

Fourth part: Keyboard key

```
AFGearToggle -1 0 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
```



(0X22 = g)

BMS makes use of the XT scan codes. The keyboard key code used in the BMS key files is composed of a leading 0X and the following one- or two-digit XT scan code Hex number.

Examples: 0X2 = "1" / 0X1E = "A" / 0X52 = "Num0" / 0XE = "Backspace"

There is one exception: If no key is assigned the keyboard key code is 0XFFFFFFF.



Each key has an assigned and unique scan code which will be transferred when pressing the key. The interesting part of it is that the scan codes for most keys (but not all!) are working regardless of which keyboard language is chosen, even if the key caption says something different.

For example:

0X15 is "Y" on an US International keyboard layout and "Z" on a German layout. So the XT codes are independent from the chosen keyboard language. 0X15 will be shown country-specific correctly.





I do not use all possible keys as we want to make sure, that the key files can be used by the majority of the community. The keyboard layouts around the globe are widely differing. So we concentrated on the most common keys. We do not want to list all key codes here as it would be a rather long list. Please refer to our BMS Keyboard Codes documents (can be found in the keyboard layouts folder) for more information.

A full list of all possible key codes can be found in the original Keyfile-generator.xls in the BMS installation folder/Docs/Falcon BMS Manuals. But if you want to create your own key file with the intention of spreading it, we would recommend staying with a few common keys as we did. If you use a single code which is not supported by your keyboard, it will crash BMS as soon as you try to load it in BMS UI.

Fifth part: Modifier key

AFGearToggle -1 0 0X22 **2** 0 0 1 "GEAR: LG Handle – Toggle"



You can assign modifiers to be used together with keyboard keys. In the example above the Shift key is assigned. To invoke the callback you have to press Shift + g on the keyboard.

We can use three different modifier keys, Shift, Ctrl and Alt. They can be combined which sums up in eight different modifier combinations:

Code #	Modifier(s) key(s)
0	None
1	Shift
2	Ctrl
3	Ctrl Shift
4	Alt
5	Alt Shift
6	Ctrl Alt
7	Ctrl Shift Alt

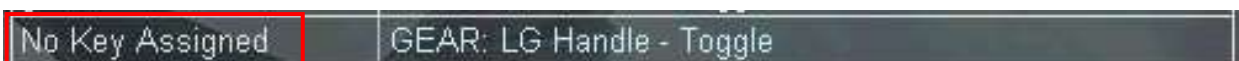
Example with a modifier combination:

AFGearToggle -1 0 0X22 **7** 0 0 1 "GEAR: LG Handle – Toggle"



If you assign a modifier but no keyboard key in the code line, the function will be shown as **"No Key Assigned"** in the UI. So it is not possible to assign a function to a modifier key only.

AFGearToggle -1 0 **0xFFFFFFFF** **2** 0 0 1 "GEAR: LG Handle – Toggle"



Also it is not distinguished between left & right modifier keys (e.g. left Shift / right Shift).





Sixth part: Key Combination key (Key Combo)

AFGearToggle -1 0 0X22 0 **0x2E 4** 1 "GEAR: LG Handle – Toggle"



Before starting here you have to understand, that the sixth and the seventh part (key combination key + modifier) cohere.

You cannot use a key combination key without a key combination modifier!

Same for the other way round, you cannot use a key combination modifier without a key combination key! But this will be addressed later.

First we have to take a look at how to set a key combo.

CommandsSetKeyCombo -1 0 **0x2E 4** 0 0 0 "SIM: CommandsSetKeyCombo Alt+C"

In our key files you have one code line which looks the same as the example above. So you assign a keyboard key + modifier key to the callback **CommandsSetKeyCombo** the same way as you normally would do with all other callbacks in a key file.

But you can't do that in the UI. Indeed it would be possible to set the keys for CommandsSetKeyCombo here. But you have no possibility to assign the key combo (Alt + C in this example) to another function in the UI. So you have to manually edit the key file with an editor either way.

Assigning a key combination key with a key combination modifier:

CommandsSetKeyCombo -1 0 **0x2E 4** 0 0 0 "SIM: CommandsSetKeyCombo Alt+C"



It is possible to set two or more different key combos. But it doesn't make sense, as there is no need for it. So we use only one key combo with our key files (Alt + C).

Once a key combo is set, we need to implement it into an existing code line:

Key combo in relation with a single keyboard key:

AFGearToggle -1 0 0X22 0 **0x2E 4** 1 "GEAR: LG Handle – Toggle"



or a keyboard key + modifier key:

AFGearToggle -1 0 0X22 2 **0x2E 4** 1 "GEAR: LG Handle – Toggle"



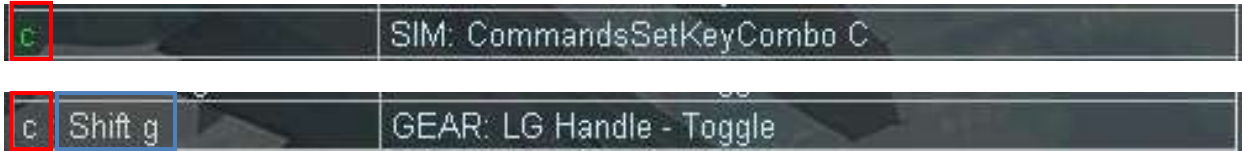


The following examples won't work:

Assigning a key combination key without a key combination modifier:

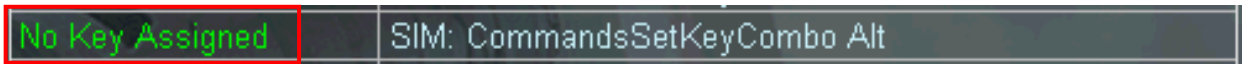
CommandsSetKeyCombo -1 0 0x2E 0 0 0 0 "SIM: CommandsSetKeyCombo C"

AFGearToggle -1 0 0X22 2 0x2E 0 1 "GEAR: LG Handle – Toggle"



Assigning a key combination modifier without a key combination key:

CommandsSetKeyCombo -1 0 0xFFFFFFFF 4 0 0 0 "SIM: CommandsSetKeyCombo Alt "



Seventh part: Key combination modifier

AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle – Toggle"

A key combination modifier has the same syntax as a normal modifier (see explanations above). Setting a key combo modifier without a key combo key does not have any effect.

Note: There is a special case for setting a key combo modifier without a key combo key. If a code line is set to "locked" (-0, see eighth part below) it is not possible to change the keys in the UI (In the example above 0X22 = "g"). You can also not assign "g" to any other function, because the locked status is preventing the deletion of the key in that code line. By assigning a key combo modifier "g" remains in the locked code line while assigning it somewhere else is possible now. This workaround is used for the "1. UI & 3RD PARTY SOFTWARE" category in the key files.

Eighth part: UI visibility

AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"

By changing the eighth part you can decide how the code line will be shown in the UI (see also Categories and Sections at the beginning of this document).

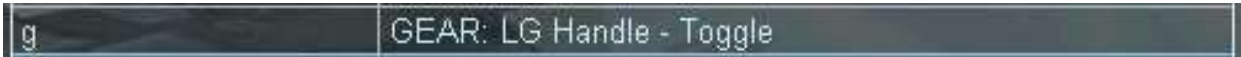
Code #	Description	Used for
1	Visible (changeable)	Standard output line (editable in UI)
-1	Headline (not changeable, blue background)	Categories & sections (easy navigation)
-0	Locked (not changeable, keys shown green)	Remarks or functions not meant to be changed by user
-2	Hidden (Invisible in UI)	Important code lines, e.g. some general radio options





Example: Visible

AFGearToggle -1 0 0X22 0 0 0 **1** "GEAR: LG Handle - Toggle"



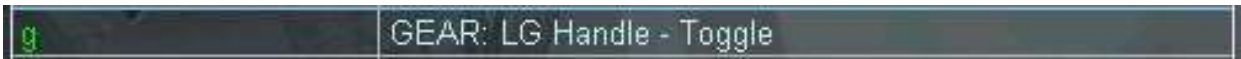
Example: Headline

AFGearToggle -1 0 0X22 0 0 0 **-1** "GEAR: LG Handle - Toggle"



Example: Visible locked

AFGearToggle -1 0 0X22 0 0 0 **-0** "GEAR: LG Handle - Toggle"



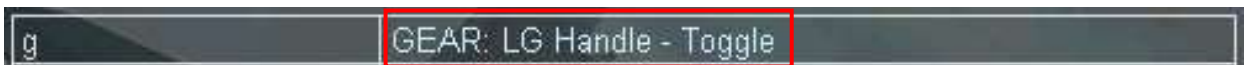
Example: Hidden (no screenshot here, of course)

AFGearToggle -1 0 0X22 0 0 0 **-2** "GEAR: LG Handle - Toggle"

Ninth part: UI description

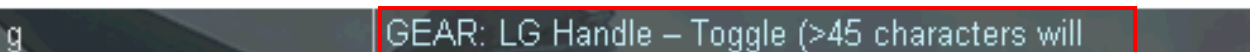
AFGearToggle -1 0 0X22 0 0 0 1 **"GEAR: LG Handle - Toggle"**

The last part of each code line defines the description shown in the BMS UI. The description line is limited to 45 characters (was 37 characters in 4.32 before) and embedded between two quotes.



When using more than 45 characters the description will be cut off as shown in the example below.

AFGearToggle -1 0 0X22 0 0 0 1 **"GEAR: LG Handle - Toggle (>45 characters will be cut off)"**





10.5.2 DirectX button code lines

A typical key file DX code line looks like this:

SimTriggerFirstDetent 0 -1 -2 0 0x0 0

Each code line is composed of seven different parts. An eighth part is optional.

Note: The parts are separated by a blank character (Spaces shown with a red underline here).

SimTriggerFirstDetent 0 -1 -2 0 0x0 0

What each part does in a code line is described below.

First part: The Callback

SimTriggerFirstDetent 0 -1 -2 0 0x0 0

The callback here works pretty much the same as described in the “Keyboard Code Lines” section with the difference, that they are invoked by pressing a DX button rather than pressing a keyboard key.

A second difference is that it does make sense to change callbacks here. If you feel unsatisfied with a DX function, just replace the callback with a new one.

Second part: The DX button ID

SimTriggerFirstDetent 0 -1 -2 0 0x0 0

The DX button ID represents a physical DX button on your input device. Each button has its own specific ID in the range of 1 to 32. Windows does not allow more than 32 different DX IDs per device.

On the contrary BMS counts the DX numbers from 0. So the first device has the IDs 0 – 31. You can assign functions to shifted and unshifted layers. How that works is described later.

Third part: Callback invocation behaviour

SimTriggerFirstDetent 0 -1 -2 0 0x0 0

DX code lines distinguish between three states when a callback is invoked. These are:

Default, Key Up, Key Down

We have four code values for the third part of the DX code line:

Code	Callback invocation
-1	Default (Both: Key up and Key down)
-2	Key down only
-4	Key up only
8	DX button assignment in BMS UI (default behaviour)

What you can do with these values is described later on.





Fourth part: DX button identifier

```
SimTriggerFirstDetent 0 -1 -2 0 0x0 0
```

This value distinguishes between DX buttons (value **-2**) and POV hats (value **-3**).

Fifth part: DX button press / release event

```
SimTriggerFirstDetent 0 -1 -2 0 0x0 0
```

```
SimTriggerFirstDetent 0 -1 -2 0x42 0x0 0
```

You can define whether the callback is invoked when a DX button is pressed (**0**) or released (**0x42**). In addition you can choose if the callback is invoked with a key down, key up or the default semantic by setting the corresponding value at the third part of the code line. More details on this subject will be provided later on in this manual.

Sixth part: Not used

```
SimTriggerFirstDetent 0 -1 -2 0 0x0 0
```

This part never changes in a DX code line and should be left untouched.

Seventh part (optional): Sound ID

```
SimTriggerFirstDetent 0 -1 -2 0 0x0 0
```

In the key files, for DX button callbacks, the former "mod1" field (last entry in a DX row, always 0 in the past) will now serve as sound ID to determine the KEY_DOWN sound to be played when activating the callback. The sound IDs can be found in the f4sndtbl.txt. Sound can be deactivated with **"-1"**. Also the value **"0"** activates no sound as no sound file is assigned.

Eighth part (optional): The description

```
SimTriggerFirstDetent 0 -1 -2 0 0x0 0 "DX: STICK – Trigger 1st Detent"
```

The description does not have a real impact. It is not shown in the UI. It only makes sense to make some personal notes about the callback, especially if you are not familiar with the callback functions.

10.5.3 DirectX POV Hat code lines

POV code lines have a slightly different syntax. A typical key file DX code line looks like this:

```
AFElevatorTrimUp 0 -1 -3 0 0x0 0
```

Each code line is composed of seven different parts. An eighth part is optional.

Note: The parts are separated by a blank character (Spaces shown with a red underline here).





AFElevatorTrimUp_0_-1_-3_0_0x0_0

What each part does in a code line is described below.

First part: The Callback

AFElevatorTrimUp 0 -1 -3 0 0x0 0

See 6.2.1

Second part: POV Hat number

AFElevatorTrimUp 0 -1 -3 0 0x0 0

BMS supports up to four different POV hats. POV Hats are numbered consecutively from 0 to 3. Most devices do not feature four different POV hats. Having just one should be common. It is possible to add a shifted layer like it is shown in the table below. The shifting offset is 2. With 4.34.1 it is now possible, to control two POV hats on two different devices. How this works is described later.

Hat #	1 POV Hat	2 POV Hats	4 POV Hats
0	unshifted	POV 1 unshifted	POV 1
1	n/a	POV 2 unshifted	POV 2
2	shifted	POV 1 shifted	POV 3
3	n/a	POV 2 shifted	POV 4

The default behaviour of the primary POV hat is changing the Point Of View. This can be changed by adding POV code lines for the unshifted layer (POV Hat number 0) and assigning different callback functions to it (e.g. TRIM).

Third part: Not used

AFElevatorTrimUp 0 -1 -3 0 0x0 0

This part is always -1.

Fourth part: POV Hat identifier

AFElevatorTrimUp 0 -1 -3 0 0x0 0

This value distinguishes between DX buttons (value -2) and POV hats (value -3).

Fifth part: Panning direction

AFElevatorTrimUp 0 -1 -3 0 0x0 0

Each POV hat has eight different directions. They are numbered consecutively from 0 to 7 (clockwise starting from the 12 o'clock position) as shown below:





Sixth part: Not used

AFElevatorTrimUp 0 -1 -3 0 0x0 0

This part is always 0x0.

Seventh part (optional): Sound ID

AFElevatorTrimUp 0 -1 -3 0 0x0 0

The sound IDs work for POV code lines as well. 0 or -1 deactivates sound.

Eighth part (optional): The description

AFElevatorTrimUp 0 -1 -3 0 0x0 0 "DX: Trim Elevator Up"

It is also possible to add descriptions to the POV hat code lines. See also 6.2.8

10.6 How to edit key files:

There are multiple ways of editing key files. Which one you prefer is your personal choice. We'd like to introduce you the possibilities to personalize it to your liking.

10.6.1 Assignments via BMS Setup menu

Key Assignments:

In the past it was not recommended to change key assignments via setup UI in BMS. This has changed as some work has been done code wise. So you may change your key assignments in the UI now. The only thing you have to keep in mind is that all comment lines will be deleted after saving. So saving in BMS UI can work.

Using an editor for that work is on the other hand maybe the first choice, even if it is inconvenient.

Note: We do not take responsibility in case something bad happens to your key file while editing it in the UI!

Let's start with a short example of a key file.

This is how it looks in an editor:

```
1 SimDoNothing -1 0 0xFFFFFFFF 0 0 0 0 "Key Assignments Via UI"
2 SimTMSUp -1 0 0XC7 1 0 0 1 "STICK: TMS Up"
3 SimTMSDown -1 0 0XCF 1 0 0 1 "STICK: TMS Down"
4 SimTMSLeft -1 0 0XD3 1 0 0 1 "STICK: TMS Left"
5 SimTMSRight -1 0 0XD1 1 0 0 1 "STICK: TMS Right"
```

And this how it looks in the UI:





No Key Assigned	Key Assignments Via UI
Shift Home	STICK: TMS Up
Shift End	STICK: TMS Down
Shift Delete	STICK: TMS Left
Shift PgDn	STICK: TMS Right

Just pick a function you would like to change a click once into the left column. As soon as you do so, the keys assignment will be highlighted in blue colour.

No Key Assigned	Key Assignments Via UI
Shift Home	STICK: TMS Up
Shift End	STICK: TMS Down

Next press the key you would like to assign instead. In this case "F1".

No Key Assigned	Key Assignments Via UI
F1	STICK: TMS Up
Shift End	STICK: TMS Down

You can do the same for keys with modifiers. Just select the left row of the function...

No Key Assigned	Key Assignments Via UI
F1	STICK: TMS Up
Shift End	STICK: TMS Down
Shift Delete	STICK: TMS Left

...and press the modifier keys, hold them and then press the keyboard key.

In this case "Shift Ctrl Alt F1".

No Key Assigned	Key Assignments Via UI
F1	STICK: TMS Up
Shift+Ctrl+Alt F1	STICK: TMS Down
Shift Delete	STICK: TMS Left

Here is what has changed in the key file itself:

```

1 SimDoNothing -1 0 0xFFFFFFFF 0 0 0 0 "Key Assignments Via UI"
2 SimTMSUp -1 0 0X3B 0 0 0 1 "STICK: TMS Up"
3 SimTMSDown -1 0 0X3B 7 0 0 1 "STICK: TMS Down"
4 SimTMSLeft -1 0 0XD3 1 0 0 1 "STICK: TMS Left"
5 SimTMSRight -1 0 0XD1 1 0 0 1 "STICK: TMS Right"

```





DirectX Assignments:

We can easily assign DX functions for the unshifted layer to our devices via the BMS UI. To give you an example I have created a short sample key file.

```
1 SimDoNothing -1 0 0xFFFFFFFF 0 0 0 0 "DX Assignment Via UI"  
2 SimTMSUp -1 0 0XC7 1 0 0 1 "STICK: TMS Up"  
3 SimTMSDown -1 0 0XCF 1 0 0 1 "STICK: TMS Down"  
4 SimTMSLeft -1 0 0XD3 1 0 0 1 "STICK: TMS Left"  
5 SimTMSRight -1 0 0XD1 1 0 0 1 "STICK: TMS Right"
```

Once loaded in the UI the same key file looks like this:

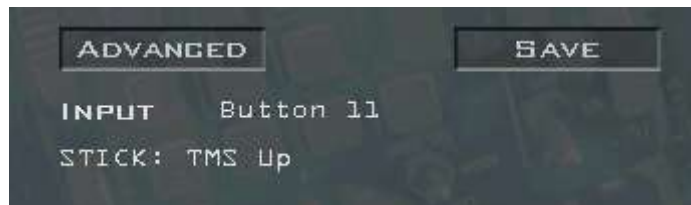
No Key Assigned	DX Assignment Via UI
Shift Home	STICK: TMS Up
Shift End	STICK: TMS Down
Shift Delete	STICK: TMS Left
Shift PgDn	STICK: TMS Right

What we need to do now is to left click on a row we want to assign a DX button to. The key of this row will be shown in blue.

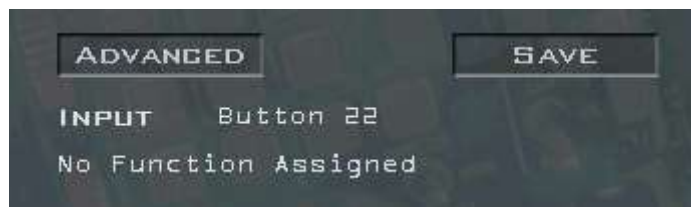
CURRENT KEYFILE: CALLBACKTEST

KEY	MAPPING
No Key Assigned	DX Assignment Via UI
Shift Home	STICK: TMS Up
Shift End	STICK: TMS Down

Now we press a button on the input device. As soon as the button is pressed the key will change to white again and below the key file list you will see something like this:



This means that you have successfully assigned a function to your input device. From now on, whenever you press that button you will see this. So you can easily figure out which function is assigned to which DX button on your device.



If you press a button on your device which has no assigned function it will be shown like in the image above.

Now, if we have assigned all four functions to DX buttons we save the key file. As you can see in the image below the DX code lines are automatically added to the key file.





```
1 SimRadarNextTarget -1 0 OXFFFFFFF 0 0 0 0 "DX Assignment Via UI"
2 SimTMSUp -1 0 OXC7 1 0 0 1 "STICK: TMS Up"
3 SimTMSDown -1 0 OXCF 1 0 0 1 "STICK: TMS Down"
4 SimTMSLeft -1 0 OXD3 1 0 0 1 "STICK: TMS Left"
5 SimTMSRight -1 0 OXD1 1 0 0 1 "STICK: TMS Right"
6 SimTMSUp 10 8 -2 0 0x0 0
7 SimTMSRight 11 8 -2 0 0x0 0
8 SimTMSDown 12 8 -2 0 0x0 0
9 SimTMSLeft 13 8 -2 0 0x0 0
10
```

Note: If you assign a DX button via the BMS UI the third part of the key file has the value "g".

10.6.2 Using an editor:

Manually Editing Keyboard Code Lines

As mentioned earlier, doing the editing in the BMS UI is not the best option considering the deletion of the comment line. If you want to keep your comments, you must use an external editor. You can edit the key files with the standard Windows editor or any other comparable program. We suggest using Notepad++, which is much better and freely downloadable from the web (google is your friend).

A part of a key file in Notepad++ will look like this:

```
1 SimDoNothing -1 0 OXFFFFFFF 0 0 0 0 "BMS Keystrokes Ver. 1.5 - Full"
2 SimDoNothing -1 0 OXFFFFFFF 0 0 0 -1 " LEFT CONSOLE"
3 #-----
4 SimDoNothing -1 0 OXFFFFFFF 0 0 0 -1 "----- TEST PANEL -----"
5 SimOverHeat -1 0 OX3B 5 0 0 1 "TEST: FIRE&OHEAT DETECT Button - Hold"
6 SimOBOGSBit -1 0 OX3C 5 0 0 1 "TEST: OXY QTY Switch - Hold"
7 SimMalIndLights -1 0 OX3D 5 0 0 1 "TEST: MAL & IND LTS Button - Hold"
8 SimMalIndLightsOFF -1 0 OXFFFFFFF 0 0 0 1 "TEST: MAL & IND LTS Button - Release"
9 SimProbeHeatMoveUp -1 0 OX3F 5 0 0 1 "TEST: PROBE HEAT Switch - Toggle Up"
10 SimProbeHeatMoveDown -1 0 OX3E 5 0 0 1 "TEST: PROBE HEAT Switch - Toggle Down"
11 SimProbeHeatOn -1 0 OXFFFFFFF 0 0 0 1 "TEST: PROBE HEAT Switch - ON"
12 SimProbeHeatOff -1 0 OXFFFFFFF 0 0 0 1 "TEST: PROBE HEAT Switch - OFF"
13 SimProbeHeatTest -1 0 OXFFFFFFF 0 0 0 1 "TEST: PROBE HEAT Switch - TEST"
14 SimEpuGenTest -1 0 OX40 5 0 0 1 "TEST: EPU/GEN Switch - Hold"
15 SimFlcsPowerTest -1 0 OX41 5 0 0 1 "TEST: FLCS PWR TEST Switch - Hold"
16 #-----
17 SimDoNothing -1 0 OXFFFFFFF 0 0 0 -1 "----- FLT CONTROL PANEL -----"
18 SimDigitalBUP -1 0 OX43 5 0 0 1 "FLT: DIGITAL Switch - BACKUP"
19 SimDigitalBUPOff -1 0 OX44 5 0 0 1 "FLT: DIGITAL Switch - OFF"
20 SimAltFlaps -1 0 OX57 5 0 0 1 "FLT: ALT FLAPS Switch - Toggle"
```

How to edit key files is best described with some examples (changes marked yellow).





```

1 1: original code line
2 AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
3
4 2: set modifier (change from 0 = "none" to 2 = "Shift")
5 AFGearToggle -1 0 0X22 2 0 0 1 "GEAR: LG Handle - Toggle"
6
7 3: set another key (change from 0X22 = "G" to 0X23 = "H")
8 AFGearToggle -1 0 0X23 0 0 0 1 "GEAR: LG Handle - Toggle"
9
10 4: set a key combo (change from 0 0 = "none" to 0x2E 4 = "Alt+C")
11 CommandsSetKeyCombo -1 0 0x2E 4 0 0 0 "SIM: CommandsSetKeyCombo Alt+C"
12 AFGearToggle -1 0 0X22 0 0x2E 4 1 "GEAR: LG Handle - Toggle"
13
14 5: set UI visibility: (change from 0 = "visible" to -0 = "locked")
15 AFGearToggle -1 0 0X22 0 0 0 -0 "GEAR: LG Handle - Toggle"
16
17 6: set UI description
18 AFGearToggle -1 0 0X22 0 0 0 1 "Enter max 3 characters"

```

Note: If you mark a value by double clicking on it (here line 18 value 0X22, dark green) other positions with the same value in the file will also be highlighted (lines 2, 5, 7, 12 & 15, brighter green).

How to avoid multiple key assignments:

If you understood what each part of a key file code line does it will be rather easy modifying existing or creating own customised files. Most likely you will only have to make the decision which functions to add and which keys (plus modifiers) to assign.

While doing that you will face the difficulty to avoid double assignment of one and the same keys.

Unfortunately you can't check for double assignment in the BMS UI. So you need other tools than that. We prefer Notepad++ for that as well.

In opposite to the double clicking feature of showing accordances it doesn't work for manually marked parts of the code (It works only for "double-clickable parts, which are separated by blank or special characters). So like in the example below the marked part is only (grey) highlighted.

```

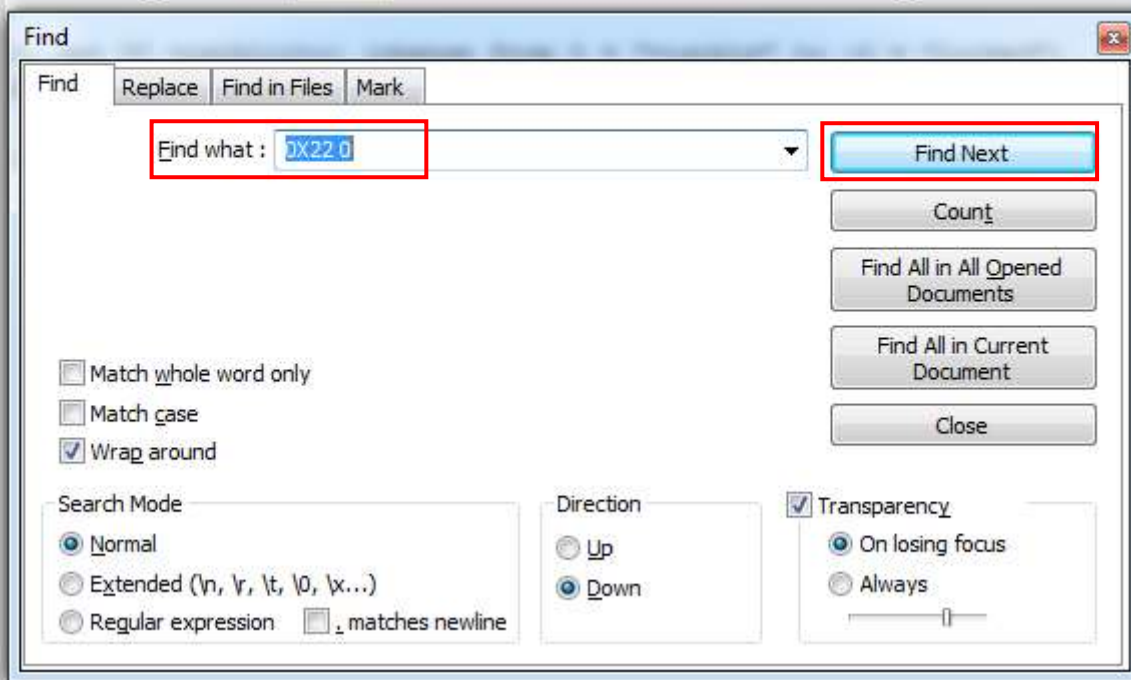
1 1: original code line
2 AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"
3
4 2: set modifier (change from 0 = "none" to 2 = "Shift")
5 AFGearToggle -1 0 0X22 2 0 0 1 "GEAR: LG Handle - Toggle"

```

To find accordances just hit Ctrl + F which opens the Find window. The highlighted code is already in the search box. Just click on "Find Next" to search for a match. If one is found the view jumps directly to the line the match was found.



```
10 4: set a key combo (change from 0 0 = "none" to 0x2E 4 = "Alt+C")
11 CommandsSetKeyCombo -1 0 0x2E 4 0 0 0 "SIM: CommandsSetKeyCombo Alt+C"
12 AFGearToggle -1 0 0X22 0 0x2E 4 1 "GEAR: LG Handle - Toggle"
```



If you found a match you should change one of the two key assignments.

As we know key files can be rather complex. So it is a good habit to do this each time you assigned a keyboard key command and / or a key modifier key. Just make sure to search for both, the fourth AND fifth part of the code line.

Manually editing DirectX Code Lines:

In the following some examples:

```
1 1. original code line
2 SimTriggerFirstDetent 0 -1 -2 0 0x0 0
3
4 2. replacing the callback (different function, same DX button ID)
5 AFGearToggle 0 -1 -2 0 0x0 0
6
7 3. set DX button ID (change from 0 = "WIN DX 1"
8 to 1 = "WIN DX 2")
9 SimTriggerFirstDetent 1 -1 -2 0 0x0 0
10
11 4. set callback invocation behaviour (change from -1 = "default"
12 to -2 = "key down")
13 SimTriggerFirstDetent 0 -2 -2 0 0x0 0
14
15 5. set DX button release event (change from 0 = "press"
16 to 0x42 = "release")
17 SimTriggerFirstDetent 0 -2 -2 0x42 0x0 0
18
```



You can avoid multiple DX button ID assignments the same way as described in the “**How to avoid multiple key assignments**” chapter.

If you assign accidentally two or more different callbacks to the one and the same DX button ID (like in the example below) only the last entry will be invoked by pressing the corresponding DX button input device.

```
1 OTWToggleFrameRate 10 8 -2 0 0x0 0
2 OTWToggleOnlinePlayersDisplay 10 8 -2 0 0x0 0
3
```

So in this example the first entry (Frame Rate) will be ignored and the last entry (Online Status) will be invoked.

10.6.3 Editing key files with the Key File Editor:

I’ve created a new Key File Editor based on Excel files (BMS Key File Editor) based on the idea of the original Keyfile-generator by Boxer. It incorporates the most important DX and key file stuff and an overview of all callbacks, logically sorted into categories and sections. You can edit almost everything. A lot of input checks are implemented.

There is an additional manual for the Editor (BMS Key File Editor Manual.pdf). Just follow the instructions.

10.6.4 DirectX shifting facility

Introduction

The HOTAS controls in the F-16 have a defined set of functionalities (see chapter “Hands-on controls”) which is often reproduced by (v)pilots in their joystick layout. However, most users have the need for controlling additional, simulator specific functionality like e.g. views or similar with their HOTAS setup.

The common practice to achieve this is the usage of a joystick “shift” button. Like the shift key on the keyboard, which controls whether a specific keystroke issues a minor letter (when used *without* shift) or a capital letter (when used *with* shift) to the PC, the shift button on the joystick controls whether a button is sending function A (when used *without* shift button) or function B (when used *with* shift button) to the simulation.

To achieve this functionality, the (v)pilot had to rely on the programming capabilities of his joystick and corresponding drivers and software in the past, making the stick act as a combined DirectX (DX) and keyboard input device. The combination of DX buttons and emulated keyboard input from the stick was needed to overcome the maximum number of buttons limitation of 32 that a DX input device is restricted to.

The current BMS version offers a built-in shifting facility that eliminates the need for such proprietary solutions. Furthermore and even a bigger advantage, it makes it possible to use a pure DX joystick setup for all functions, shifted and unshifted, by effectively (nearly) doubling the number of possible DX button inputs for a single device to 63.





Background

BMS DirectX Button Handling

BMS can handle up to 16 DX devices with 32 buttons each, making a total input of 512 DX buttons possible. To assign DX input, a special type of input line within the BMS keyfile is used e.g.:

```
SimTriggerFirstDetent 0 -1 -2 0 0x0 0  
SimPickle 1 -1 -2 0 0x0 0
```

The red part of the input lines will not be explained here. The green part of the input lines is composed from the name of the function to call and the DX button number that should trigger the execution of the function.

BMS starts enumerating the buttons with 0, so the first DX device connected to the PC uses button numbers 0 to 31, where button #1 on the stick is mapped to 0 in the keyfile, button #2 is mapped to 1 etc. If more than one DX device is connected, the 2nd device will use button numbers 32 to 63, where button #1 on the 2nd device is mapped to 32, button #2 is mapped to 33 etc.

Every DX device has its dedicated 32 button range in the keyfile, so a possible 16th DX device would use button range 480 to 511. Using this technique, BMS can distinguish between the different DX devices.

BMS DirectX POV Hat Handling

Besides the DX buttons, BMS can additionally handle up to 4 POV hat switches or 4 different POV hat layers (shifted and unshifted states). To assign POV hat input, a special type of input line within the BMS key file is used e.g.:

```
AFElevatorTrimUp 0 -1 -3 0 0x0 0  
AFElevatorTrimDown 0 -1 -3 4 0x0 0
```

The red part of the input lines does never change for POV hat definitions and will not be explained here. The green part of the input lines is composed from the name of the function to call, the POV hat number (0 to 3) and the direction in which the POV hat needs to be pushed on order to trigger the execution of the function. Regardless of the physical capabilities of the POV hat, BMS always distinguishes 8 separate directions:



Note: POV hats that have not been mapped in the keyfile automatically default to view panning.





DirectX Button Shifting

Concept and Usage

This version of BMS offers a new keyfile callback for DX buttons, which defines the mapped button to act as shift modifier. While this button is pressed down, BMS will add a fixed (but configurable) offset number to all DX button inputs.

Example: The shift button is configured to add an offset of 256 while pressed. Button #2 on the joystick (1st DX device) should usually work as weapons pickle button but should toggle the gear when used in conjunction with shift. The keyfile would need to contain the following entries to achieve this behavior:

```
SimPickle 1 -1 -2 0 0x0 0  
AFGearToggle 257 -1 -2 0 0x0 0
```

The 257 is composed from DX button #2 (1) plus the 256-shift offset. So the usage of the shift button effectively moves the DX buttons of a specific DX device into a separate DX device range. In the example above, usage of shift moves the buttons of the 1st DX device from 0-31 to 256-287. Conclusively, DX buttons that are pushed on the 1st DX device while shift is hold will look to BMS like being issued from a (virtual) 9th DX device.

Enabling Shift and Configuring Offset

To enable the shifting facility, the following parameter has to be added to the falconbms.cfg file:

```
set g_nHotasPinkyShiftMagnitude n
```

The parameter value 0 disables shifting. Setting *n* to a higher value enables it and specifies the button offset number. Although arbitrary offset numbers are supported, it is highly recommended to use a multiple of 32 for the offset. Like this, a shifted DX button range always maps to the complete button range of another (virtual) DX device.

How to change the DX Shifting Magnitude is described later on.

Mapping the Shift Button

The callback name for the DX button that should act as shift button is:

SimHotasPinkyShift

When shifting is active, this callback replaces the former “SimPinkySwitch” callback. It still fulfills the same EXPAND functionality if it is only tapped and released. When it is pressed and held down, it acts as shift button. Obviously, it is meant to be mapped to the pinky switch on the HOTAS.

In order to make shift work properly, the shift callback has to be mapped twice in the keyfile. Once for the





pinky button, and once for the pinky button plus shift offset.

Example: The pinky switch on the HOTAS uses DX button #3 and the configured shift offset is 256. Hence the keyfile needs to contain the following lines:

```
SimHotasPinkyShift 2 -1 -2 0 0x0 0  
SimHotasPinkyShift 258 -1 -2 0 0x0 0
```

Note: If the 2nd callback mapping is missing, the shift button can not be released correctly again once pressed.

If you don't use shifting at all you can also use the standard callback (SimPinkySwitch).

It is NOT possible to have more than one DX shifting layer.

But it is possible to assign the callback SimHotasPinkyShift to more than one physical DX button.

DirectX POV Hat Shifting

Concept and Usage

The shifting of POV hat buttons follows the same concept as shifting DX buttons. It uses the same shift callback that has already been defined for DX buttons. Only difference is that the shifting offset for POV hats is fixed to 2.

Example: The 1st POV hat should be used for view panning when used unshifted, and for trimming when used shifted. The keyfile would need to contain the following entries to achieve this behavior:

```
AFElevatorTrimUp 2 -1 -3 0 0x0 0  
AFAileronTrimRight 2 -1 -3 2 0x0 0  
AFElevatorTrimDown 2 -1 -3 4 0x0 0  
AFAileronTrimLeft 2 -1 -3 6 0x0 0
```

The 2 is composed from POV hat #1 (0) plus the fixed 2 shift offset. So the usage of the shift button effectively promotes the 1st POV hat to act as a 3rd (virtual) POV hat. The directional numbers do not change.

Note: It is not necessary to explicitly map the default view panning behavior to POV hat #1 (0), as every hat which is not mapped defaults to view panning behavior.

Advanced Information

Specifying the Pinky Tapping Time

The maximum pinky tapping time in milliseconds that is used to determine whether the pinky button should execute EXPAND or act as shift button can be configured within the falconbms.cfg file:





set g_nHotasShiftQuickPressTimeLimit n

The parameter value defaults to 200. If the pinky button is tapped and released within n milliseconds, EXPAND is executed. If it is not released within n milliseconds, shift is executed instead.

Shifting Multiple Devices Simultaneously

If the shift callback is mapped on more than one physical device, each individual shift button applies shift to all devices at once. E.g. pressing shift on the 1st DX device shifts buttons of the 1st as well as any other DX devices that are connected and vice versa.

Minimizing Side Effects

Most likely not every DX button or POV hat will have a shifted functionality assigned when creating a keyfile. As pointed out in the background section, any device or hat that is not used will map to default behavior. The same is true for shifted buttons and hats.

While this behavior may be desirable in some situations, it may be as well totally unwanted in other situations. A good example for an unwanted situation is the mapping of trim to a shifted POV hat, as described in the Concepts and Usage section above. In this example, only the shifted main up/down/left/right hat directions are mapped to trim, whereas the corner directions have no shifted functions assigned. Therefore, they still default to view panning, even when used shifted. When the (v)pilot tries to trim aileron and elevator at once by shift-pressing the hat to a corner, he will execute view panning instead of the desired trim.

The only way to avoid that behavior is to actually assign a function to the shifted POV hat corner directions. However, combined aileron/elevator trim keystrokes for our example are not available.

To compensate for such restrictions, a special callback has been introduced:

SimDoNothing

This callback does exactly what it says: nothing.

Like this, unwanted functionality can be deliberately deactivated. So for the trimming/view panning example, the shifted POV hat corners can be mapped to SimDoNothing to avoid unwanted view panning while trimming:

```
AFElevatorTrimUp 2 -1 -3 0 0x0 0  
SimDoNothing 2 -1 -3 1 0x0 0  
AFAileronTrimRight 2 -1 -3 2 0x0 0  
SimDoNothing 2 -1 -3 3 0x0 0  
AFElevatorTrimDown 2 -1 -3 4 0x0 0  
SimDoNothing 2 -1 -3 5 0x0 0  
AFAileronTrimLeft 2 -1 -3 6 0x0 0  
SimDoNothing 2 -1 -3 7 0x0 0
```

The same can be done with every shifted DX button or POV hat which should not execute default functionality.





Restrictions

Advanced or logical programming - as offered by many joystick vendors - can not be implemented with the BMS DX shifting facility.

It is not possible to assign functions to the shifted layer via the Setup UI.

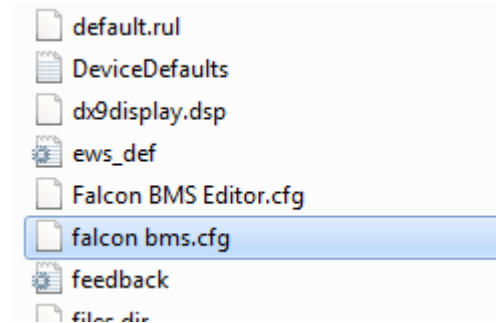
How to change DX shift magnitude in the Falcon BMS.cfg:

First, open your falcon BMS.cfg with an editor like notepad++. You find it in BMS Install Folder / User / Config.

Once opened scroll a bit down until you find the “Misc Settings” section.

Look for the code line

```
set g_nHotasPinkyShiftMagnitude 256
```



The default value is 256.

```
76 ////////////////////////////////////////////////////
77 // Misc Settings //
78 ////////////////////////////////////////////////////
79
80 set g_bVoiceCom 1
81 set g_nF1TeamUiFreq 307300
82 set g_nF2TeamUiFreq 1234
83 set g_bhudAOA 1
84 set g_bLocalEnvironmentalDate 0
85 set g_bHotasDgftSelfCancel 1
86 set g_nHotasPinkyShiftMagnitude 256
87 set g_fFOVIncrement 5
```

Change this value to **0** to disable shifting:

```
86 set g_nHotasPinkyShiftMagnitude 0
```

Setting it to 512

```
86 set g_nHotasPinkyShiftMagnitude 512
```

Of course you could specify any other value between 32 (Note: You’ll need at least one unshifted layer anyway) and 511. But this doesn’t make any sense as shifting outside the DX device limit is now possible. To sum it up:

If have not more than 8 different input devices the default setting is ok. This should apply to most users. If you have more than that you don’t necessarily have to do the math. Instead you can just set it to 512 and you are good to go.





10.6.5 DX device limitation:

The maximum number of DX devices support is 16.

Additionally, the DX button numbers can now be shifted *outside* the DX device limit, making it essentially possible to use *all* 16 DX devices with DX shifting. To do so, simply specify "[set g_nHotasPinkyShiftMagnitude 512](#)" (default is 256) in the config file (and adjust your keyfile accordingly, of course).

So instead of shifting only the 1st half DX devices 1-8 to the 9-16 range, you can now shift all DX devices 1-16 to the 17-32 range.

Using shifting or not does not matter. If you try to access a button "beyond the max range" (either by shifting or regularly), it simply is ignored. So the actual number of devices does not seem to be a problem at all - other for the fact that BMS simply ignores devices that are out of the range.

To clarify that here some examples:

First example: 8 DX devices with a shifted layer (default)

[set g_nHotasPinkyShiftMagnitude 256](#)

# of device	WIN DX #	BMS DX #	BMS shifted
Device 1	1 ... 32	0 ... 31	256 ... 287
Device 2	33 ... 64	32 ... 63	288 ... 319
Device 3	65 ... 96	64 ... 95	320 ... 351
Device 4	97 ... 128	96 ... 127	352 ... 383
Device 5	129 ... 160	128 ... 159	384 ... 415
Device 6	161 ... 192	160 ... 191	416 ... 447
Device 7	193 ... 224	192 ... 223	448 ... 479
Device 8	225 ... 256	224 ... 255	480 ... 511

Second example: 16 DX devices with a shifted layer

[set g_nHotasPinkyShiftMagnitude 512](#)

# of device	WIN DX #	BMS DX #	BMS shifted
Device 1	1 ... 32	0 ... 31	512 ... 543
Device 2	33 ... 64	32 ... 63	544 ... 575
Device 3	65 ... 96	64 ... 95	576 ... 607
Device 4	97 ... 128	96 ... 127	608 ... 639
Device 5	129 ... 160	128 ... 159	640 ... 671
Device 6	161 ... 192	160 ... 191	672 ... 703
Device 7	193 ... 224	192 ... 223	704 ... 735
Device 8	225 ... 256	224 ... 255	736 ... 767
Device 9	257 ... 288	256 ... 287	768 ... 799
Device 10	289 ... 320	288 ... 319	800 ... 831
Device 11	321 ... 352	320 ... 351	832 ... 863
Device 12	353 ... 384	352 ... 383	864 ... 895
Device 13	385 ... 416	384 ... 415	896 ... 927
Device 14	417 ... 448	416 ... 447	928 ... 959
Device 15	449 ... 480	448 ... 479	960 ... 991
Device 16	481 ... 512	480 ... 511	992 ... 1023

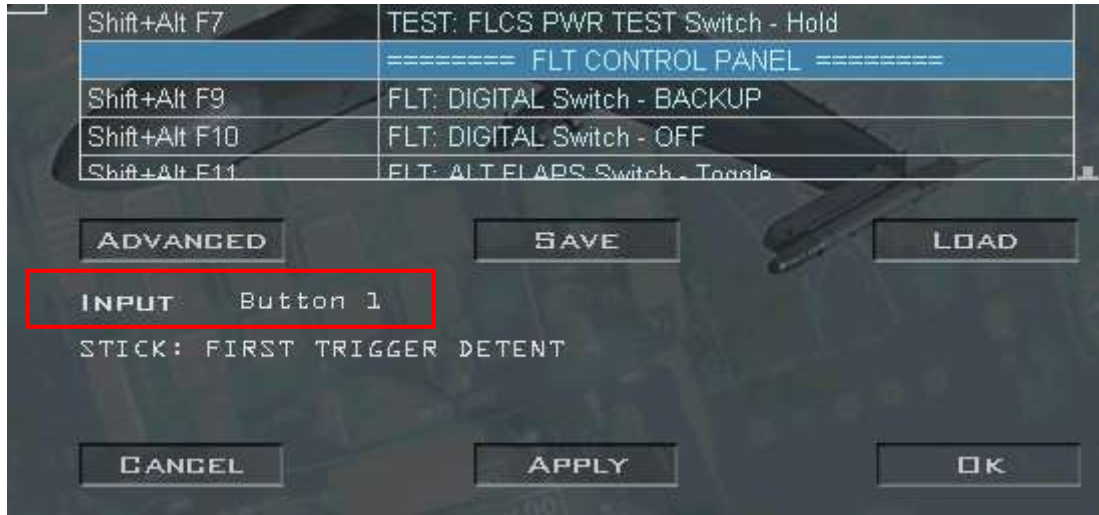




10.7 The DX Device Order:

10.7.1 How to find out your DX device order and DX button IDs:

If you want to find out the DX button number of a specific physical button or switch on your DX device there is an easy way to find out. Just enter the BMS UI and enter Setup – Controllers page. Now you have to press the button / move the switch on your device.



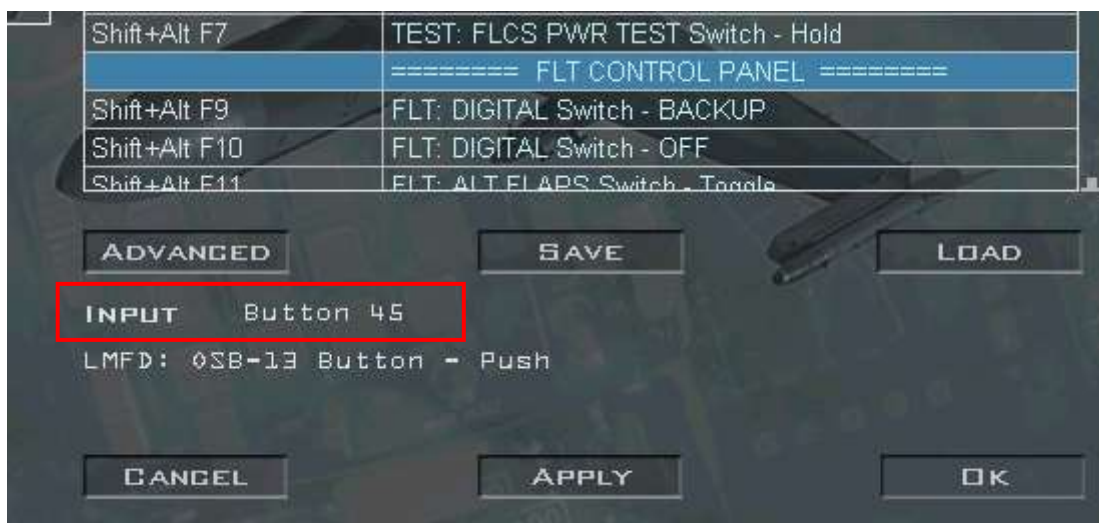
In the above image you see INPUT and Button 1 to the right.

Button 1 means in this context DirectX button 1. The DirectX button numbers in the BMS UI are shown as Windows DX numbers, which really makes it counterintuitive.

BMS counts DX button numbers slightly different than Windows. Win DX # 1 = BMS DX #0.

Win DX # 1 = BMS DX # 0. Do this for all buttons you need and note them.

You can do the same for any other device e.g. your second controller in the next example:



In the example above we see Button 45, which means Win DX # 45 and BMS DX # 44.





As mentioned above there is a difference between the Windows DX numbers and the BMS DX numbers. While Windows counts DX buttons from 1 BMS does from 0. So the BMS DX button number of the first device (Button 1 in the screenshot) is BMS DX 0. The button # 45 of the second controller is BMS # 44.

The BMS DX button numbers are used in the key file. So, if the UI says INPUT Button 1 the same device button will be DX number 0 in the key file. We can also find out the device number that way.

As you might remember we have a maximum of 32 DX buttons per device. So, if the shown button number is in the range of 1 – 32 it must be device 1. If the button number is in the range of 33 – 64 it is device # 2 and so on.

Following list should clarify that:

WIN DX #	BMS DX #	# of device
1 ... 32	0 ... 31	Device 1
33 ... 64	32 ... 63	Device 2
65 ... 96	64 ... 95	Device 3
And so on...		

Based on these numbers we also calculate the DX button IDs for the shifted layers. They have to be calculated manually or by using our Excel BMS Keyfile Editor.

10.7.2 How to avoid DX device order and xaxis changes

When you start Falcon BMS for the very first time the order of devices is stored in the "DeviceSorting.txt" in the "User\Config" directory. What is stored in that file is also shown in the Controllers selection menu.

Just enter the BMS Setup – Controllers page. Now open the Controller menu. The list shows the order of your devices. Please note that keyboard is not taken into account here. So, the first DX device in this example is the Thrustmaster HOTAS Cougar, the second DX device is F16 MFD 1 and so on. How to change that order is described in the next section.



10.7.3 How to set DX device order

A new feature has been introduced with BMS 4.33 U1.

Direct Input devices (joysticks, MFDs, boards etc.) can now be sorted to *specific* positions as desired via a new config file "DeviceSorting.txt" in the "User\Config" directory.





BMS - Pitbuilder.key	17.01.2016 23:39
BMS - README.txt	06.09.2014 10:03
default.rul	04.05.2013 19:34
DeviceDefaults.txt	05.06.2015 07:10
DeviceSorting.txt	07.01.2016 15:00
dx9display.dsp	19.02.2016 23:31
ews_def.ini	06.03.2012 18:49

This file will be created automatically if it is not existing, and it will list all devices which are currently connected to BMS. If you want to change the device order, simply close BMS, edit the file with a text editor and copy/paste the lines in the file to your liking. Once the file exists, it will always be loaded and the order in there will be honored by BMS. If you connect a new device which is not listed in the file yet, it will be appended to the existing file without changing the specified order. The file simply consists of the GUID and the device name for each device, one device per line.

Example:

```
1 {0400044F-0000-0000-0000-504944564944} "Thrustmaster HOTAS Cougar"  
2 {B351044F-0000-0000-0000-504944564944} "F16 MFD 1"  
3 {B352044F-0000-0000-0000-504944564944} "F16 MFD 2"  
4
```

If you add one device it will be appended to the end of the file:

```
1 {0400044F-0000-0000-0000-504944564944} "Thrustmaster HOTAS Cougar"  
2 {B351044F-0000-0000-0000-504944564944} "F16 MFD 1"  
3 {B352044F-0000-0000-0000-504944564944} "F16 MFD 2"  
4 {03E916C0-0000-0000-0000-504944564944} "F-16 ICP USB"
```

You can sort the device by simple copy/paste. In the example below the ICP has been moved from #4 to #2.

```
1 {0400044F-0000-0000-0000-504944564944} "Thrustmaster HOTAS Cougar"  
2 {03E916C0-0000-0000-0000-504944564944} "F-16 ICP USB"  
3 {B351044F-0000-0000-0000-504944564944} "F16 MFD 1"  
4 {B352044F-0000-0000-0000-504944564944} "F16 MFD 2"
```

If you make sure that all controllers are always connected to your computer on BMS start inadvertent changes of the device order are prevented. But there are some obstacles.





10.7.4 Basic DX device setup flow

Before we start explaining what can go wrong we first like to show the basic steps to setup your devices correctly. Please note that we do not discuss how to assign functions here as this is covered in other parts of this document.

1. Connect all DX devices to your computer (At least the ones you need for BMS).
2. Start BMS.
3. Verify you have selected the right pilot profile in the logbook page.
4. Verify all devices are shown correctly in the Controllers selection menu (Setup – Controllers)
5. Close BMS.
6. Open DeviceSorting.txt and change the order to your liking.
7. Start BMS.
8. Select your main input device (Controllers selection menu).
9. Load a key file.
10. Assign the axis to your devices in Setup – Controllers – Advanced as desired.
11. Test all axes and verify they are working correctly.
12. Verify that the devices buttons are working as expected. Just press one button per device no matter if they are two separate DirectX devices (e.g. the Warthog) or combined (e.g. Cougar). Do this for all devices connected.

Once you have done this and you do not regularly unplug / plug in devices you shouldn't have any issues in the future. If you are sure that everything is working correctly and all your preferred settings are done in the Setup pages you can also set the following files to read only:

- axismapping.dat
- joystick.cal
- callsign.pop

10.7.5 Technical details about the DX device sorting

Every change you make in the controller section WILL be active in the BMS session, regardless whether you hit CANCEL or OK, as the UI system needs to apply these changes right away in order to make the axis bars etc. show the correct values. This behaviour has not been changed/touched. Only the *persisting* of these changes to the files will no longer happen "automatically" once you e.g. change a value in a drop-down box. There currently is no way to really "cancel" - i.e. rollback - changes that you make in the controller UI screens. They will stay active as long as BMS is running no matter what (even when you hit CANCEL).

If you forget to plug in your sticks and you start BMS, it will re-enumerate the axis and re-assign them for the running session. You can even make changes in the ADVANCED CONTROLS setup screen. But unless you exit the setup screen with OK or APPLY, *none* of these changes will be persisted to the files. Hence, if you quit BMS, re-plug your missing devices, their axis will still be mapped correctly as they were before.

In addition the files "axismapping.dat" and "joystick.cal" in the <BMS>\User\Config directory will no longer be overwritten automatically. They will only be written if the "OK" or "APPLY" buttons on the UI SETUP screen are hit explicitly.





In summary: if you have a working DX button setup, it will never be "shuffled around" again like it used to. But you still have to ensure that all devices are plugged in before you start BMS!

Example: you have one stick, one ICP and 2 MFDs. The stick usually is the main input device and as such device #1 (DX button numbers 0-31). The ICP as the 2nd device uses DX button numbers 32-63, the 1st MFD as the third device has 64-95, the 2nd MFD has 96-127. Now you start BMS while forgetting to plug in the ICP.

```
1 {0400044F-0000-0000-0000-504944564944} "Thrustmaster HOTAS Cougar"  
2 {03E916C0-0000-0000-0000-504944564944} "F-16 ICP USB"  
3 {B351044F-0000-0000-0000-504944564944} "F16 MFD 1"  
4 {B352044F-0000-0000-0000-504944564944} "F16 MFD 2"
```

Missing devices will be ignored. In this case this means, that MFD 1 will be recognised as the 2nd device and the buttons will move from to 64-95 to 0-31 (and MFD 2 moves accordingly). However - and that is the whole purpose of this patch - once you realize that the ICP is missing, you can hit "Cancel", close BMS, plug in the ICP, and now it is *guaranteed* that the ICP will be seen by BMS as the 2nd device with button numbers 0-31 again. That means even by unplugging stuff and re-plugging it, the DX button numbers will not change anymore.

But if you hit the "OK" or "Apply" button it can still have some drawbacks mainly regarding the axe's assignments. As BMS fails to see the ICPs axis while the device is not plugged in it now assumes that this device never existed and sets all axes applied to the ICP device to "Keyboard". When you start BMS the next time you will realise that you have to assign these axes again.

We suggest to set the files `axismapping.dat`, `joystick.cal` and `callsign.pop` to read only once BMS is setup the way you want. But do not forget to remove that flag if you intend to make changes in the Setup menu.





10.8 DirectX device specifics:

10.8.1 How to cancel MRM/DF override Modes:

Note: This is TM HOTAS Cougar specific.

The DF Switch on the Throttle has only two different DX numbers: One for MRM override mode and one for DF override mode. There is no separate DX button for cancel (middle position). To overcome this limitation you can set a value in the Falcon BMS.cfg.

Open the cfg and scroll down to the Misc Settings section. Look for the code line

```
set g_bHotasDgftSelfCancel 0
```

and change the value to **1**. This cancels the override mode automatically when turning the DF switch back into the middle position.

10.8.2 How to overcome DX button shortcomings?

Note: This is TM HOTAS Warthog specific but can also apply to other input devices.

Some of the 3-way switches of the Warthog throttle have only 2 different DX button IDs. Let's take an example:

Function Name (on HOTAS)	Script Name (in TARGET)	DX button ID		Testcallback
		Windows	BMS device 2	
PATH	APPAT	27	58	SimRFNorm
ALT/HDG	APAH	n/a	58 / 59	SimRFQuiet
ALT	APALT	28	59	SimRFSilent

As we see only PATH (Win DX 27 / BMS DX 58) and ALT (Win DX 28 / BMS DX 59) have assigned DX button numbers. So we can only assign two different callbacks via DX. This is wrong!

There is a way to overcome the missing of the DX number for the middle function (ALT/HDG - no DX).

And here is, how it works:

default DX code line:

```
Callback 4 -1 -2 0 0x0 0
```

- 4 = DX button number
- 1 = -1 is the default key up/down behavior
- 0 = default button press/release event





DX code lines with new key up/ down semantic:

Callback1 4 -2 -2 0 0x0 0

Callback2 4 -2 -2 0x42 0x0 0

- 4 = DX button number (same as default code line)
- 2 = -2 invokes the callback with a key down semantic
- 0 = 0 invokes the callback when the DX button is pressed
- 0x42 = 0x42 invokes the callback when the DX button is released

To keep the story short:

With both these lines we can assign two different callbacks to one and the same DX button number!

One for press and one for release.

To stay with our example you can test this with your Warthog:

```
SimRFNorm 58 -2 -2 0 0x0 0  
SimRFQuiet 58 -2 -2 0x42 0x0 0
```

```
SimRFSilent 59 -2 -2 0 0x0 0  
SimRFQuiet 59 -2 -2 0x42 0x0 0
```

And this is what happens:

```
SimRFNorm 58 -2 -2 0 0x0 0  
- physical FWD position: DX button press event -> invokes SimRFNorm
```

```
SimRFQuiet 58 -2 -2 0x42 0x0 0  
- physical MID position: DX button release event -> invokes SimRFQuiet
```

```
SimRFSilent 59 -2 -2 0 0x0 0  
- physical AFT position: DX button press event -> invokes SimRFSilent
```

```
SimRFQuiet 59 -2 -2 0x42 0x0 0  
- physical MID position: DX button release event -> invokes SimRFQuiet
```

You can do this with every DX button #. Of course this is not recommended and not useful on all occasions.





10.8.3 How to use POV hats on two devices

This new feature introduced in 4.35 allows you to operate POV hats via DX on different devices. In the past, it was only possible to use POVs on your main input device. This has changed now and many HOTAS systems, e.g. the TM Warthog, will benefit from that. To make use of this feature new config options are available in Falcon BMS.cfg.

Config options:

- [*set g_nNumOfPOVs*](#)
Defines the POV hat behaviour or sets the number of POV hats.

Option	Description
-1	Default POV behaviour
0	Disables all POV hats on all devices.
1	Activates 1 POV hat
2	Activates 2 POV hats

Note: With option “-1” POV hats are only available on the primary input device. Option “1” does only make sense if you want to use just one POV hat that is not on your primary input device.

- [*set g_nPOV1DeviceID*](#)
Sets the ID of the first POV hat device.
- [*set g_nPOV2DeviceID*](#)
Sets the ID of the second POV hat device. This is only available when set g_nNumOfPOVs option is set to 2.
- [*set g_nPOV1ID*](#)
Sets the POV hat ID on the first device.

Option	Description
0	1st POV hat on POV1DeviceID
1	2nd POV hat on POV1DeviceID

- [*set g_nPOV2ID*](#)
Sets the POV ID on the second device.

Option	Description
0	1st POV hat on POV2DeviceID
1	2nd POV hat on POV2DeviceID

Note: In Warthog combined mode you still cannot use the throttle POV with DX.





How to get the device ID:

This is simple. It follows the order defined in the DeviceSorting.txt. Please note that the first ID is always assigned to the keyboard. So:

`set g_nPOV1DeviceID 1` = Keyboard (setting this does not make sense of course)

Here is the content of the DeviceSorting.txt:

```
{0402044F-0000-0000-0000-504944564944} "Joystick - HOTAS Warthog"
{0404044F-0000-0000-0000-504944564944} "Throttle - HOTAS Warthog"
{B351044F-0000-0000-0000-504944564944} "F16 MFD 1"
{B352044F-0000-0000-0000-504944564944} "F16 MFD 2"
{01000100-0000-0000-0000-504944564944} "Cougar Throttle"
{03E916C0-0000-0000-0000-504944564944} "F-16 ICP USB"
{0400044F-0000-0000-0000-504944564944} "Thrustmaster HOTAS Cougar"
{78787878-0000-0000-0000-504944564944} "Simped F16"
```

Note: In this example the F-16 ICP USB is not connected to the computer hence it is ignored by BMS.

And here is how it looks in BMS SETUP menu:



ID	Description
1	Keyboard
2	Joystick – HOTAS Warthog
3	Throttle – HOTAS Warthog
...	...
7	Thrustmaster HOTAS Cougar

You simply have to count from top to bottom. In the following example we define the POV hats of the TM Warthog:

```
set g_nNumOfPOVs 2
set g_nPOV1DeviceID 2 // Joystick
set g_nPOV1ID 0
set g_nPOV2DeviceID 3 // Throttle
set g_nPOV2ID 0
```

If we want to use the Cougar Stick instead (note: due to modding Stick and Throttle are two different USB devices!) we need this:

```
set g_nNumOfPOVs 2
set g_nPOV1DeviceID 7 // Joystick
set g_nPOV1ID 0
set g_nPOV2DeviceID 3 // Throttle
set g_nPOV2ID 0
```





Now you can assign DX function to both POV hats as described in the “DirectX POV Hat Code Lines” and Shifting fascility chapters. DX shifting functions are supported.

10.9 Key file options & specifics:

10.9.1 Changing ICP-Numpad mapping (1=7 -> 7=1):

In our key file the mapping for ICP on the numpad is OSB 1= NUM 1, OSB 7= NUM 7.

If you would like to have a more realistic 1=7, 7=1 mapping just copy the lines below and overwrite! the corresponding lines in the key file (see ICP section).

```
SimICPTILS 1 0 0X47 0 0 0 1 "ICP: 1-ILS"  
SimICPALOW 1 0 0X48 0 0 0 1 "ICP: 2-ALLOW"  
SimICPTHREE 1 0 0X49 0 0 0 1 "ICP: 3"  
SimICPStpt 1 0 0X4B 0 0 0 1 "ICP: 4-STPT"  
SimICPCrus 1 0 0X4C 0 0 0 1 "ICP: 5-CRUS"  
SimICPSIX 1 0 0X4D 0 0 0 1 "ICP: 6-TIME"  
SimICPMark 1 0 0X4F 0 0 0 1 "ICP: 7-MARK"  
SimICPEIGHT 1 0 0X50 0 0 0 1 "ICP: 8-FIX"  
SimICPNINE 1 0 0X51 0 0 0 1 "ICP: 9-A-CAL"
```

10.9.2 How to change the DX POV functions (Trim vs. View & other functions):

In our key files the unshifted POV set to TRIM. The View functions are set to the shifted layer. Here is how you can change that:

- Delete the following lines in the key file – HOTAS UNSHIFTED (They set TRIM to unshifted layer):

```
AFElevatorTrimUp 0 -1 -3 0 0x0 0  
SimDoNothing 0 -1 -3 1 0x0 0  
AFAileronTrimRight 0 -1 -3 2 0x0 0  
SimDoNothing 0 -1 -3 3 0x0 0  
AFElevatorTrimDown 0 -1 -3 4 0x0 0  
SimDoNothing 0 -1 -3 5 0x0 0  
AFAileronTrimLeft 0 -1 -3 6 0x0 0  
SimDoNothing 0 -1 -3 7 0x0 0
```

- Delete the following lines into the key file – HOTAS SHIFTED (They set Views to shifted layer):

```
OTWViewUp 2 -1 -3 0 0x0 0  
SimDoNothing 2 -1 -3 1 0x0 0  
OTWViewRight 2 -1 -3 2 0x0 0  
SimDoNothing 2 -1 -3 3 0x0 0  
OTWViewDown 2 -1 -3 4 0x0 0  
SimDoNothing 2 -1 -3 5 0x0 0
```





```
OTWViewLeft 2 -1 -3 6 0x0 0  
SimDoNothing 2 -1 -3 7 0x0 0
```

- Copy the following lines into the key file – HOTAS UNSHIFTED (They set Views to unshifted layer):

```
OTWViewUp 0 -1 -3 0 0x0 0  
SimDoNothing 0 -1 -3 1 0x0 0  
OTWViewRight 0 -1 -3 2 0x0 0  
SimDoNothing 0 -1 -3 3 0x0 0  
OTWViewDown 0 -1 -3 4 0x0 0  
SimDoNothing 0 -1 -3 5 0x0 0  
OTWViewLeft 0 -1 -3 6 0x0 0  
SimDoNothing 0 -1 -3 7 0x0 0
```

- Copy the following lines into the key file – HOTAS SHIFTED (They set TRIM to shifted layer):

```
AFElevatorTrimUp 2 -1 -3 0 0x0 0  
SimDoNothing 2 -1 -3 1 0x0 0  
AFAileronTrimRight 2 -1 -3 2 0x0 0  
SimDoNothing 2 -1 -3 3 0x0 0  
AFElevatorTrimDown 2 -1 -3 4 0x0 0  
SimDoNothing 2 -1 -3 5 0x0 0  
AFAileronTrimLeft 2 -1 -3 6 0x0 0  
SimDoNothing 2 -1 -3 7 0x0 0
```

Add other functions to unshifted / shifted layer:

Of course it is possible to assign any callback you want to the POV hat. The code lines follow always the same syntax.

AddFunctionOfYourChoice: Replace this with another callback.

0/2: Chose unshifted (**0**) or shifted (**2**) layer.

```
AddFunctionOfYourChoice 0/2 -1 -3 0 0x0 0  
SimDoNothing 0/2 -1 -3 1 0x0 0  
AddFunctionOfYourChoice 0/2 -1 -3 2 0x0 0  
SimDoNothing 0/2 -1 -3 3 0x0 0  
AddFunctionOfYourChoice 0/2 -1 -3 4 0x0 0  
SimDoNothing 0/2 -1 -3 5 0x0 0  
AddFunctionOfYourChoice 0/2 -1 -3 6 0x0 0  
SimDoNothing 0/2 -1 -3 7 0x0 0
```





10.9.3 Assigning keys to Extra MFDs (3rd & 4th MFD):

I did not assign any keys to the 3rd and 4th MFDs because the situations where you would use them are pretty rare. It is not possible to use them at all at the moment.

Be advised: They are for non F-16 MFD control only if an AC has three or more MFDs.

A suggestion for mapping is:

MFD 3: Shift+Ctrl +1, +2, +3... and Shift+Ctrl +Num1, +Num2, +Num3...

MFD 4: Shift+Ctrl+Alt +1, +2, +3... and Shift+Ctrl+Alt +Num1, +Num2, +Num3...

Another solution for DX is to add the 3rd and 4th MFD to the shifted layers of the 1st and 2nd MFDs. We included the DX code lines into the DirectX TM Cougar MFD.pdf.

10.9.4 Double entries:

There are some functions, which are present at two (or more) different locations. These are:

AFResetTrim (Reset Trim):

There are no cockpit switches for that function in a real F-16. Nonetheless this callback is implemented into Falcon most likely due to comfort reasons. The pilots might look for it at different locations. That is the reason, why you can find the Trim Reset three times.

You can find it in the MANUAL TRIM, FLIGHT STICK and in the OTHER COCKPIT CALLBACKS sections. You can change the key setting only in the CKPIT section. The state of the callback in the TRIM and the STICK section is visible, not changeable with no keys assigned. You will be referred to the CKPIT section there if you want to change the keys.

SimPickle:

Here we have two different locations, where you can shoot a weapon. The FLIGHT STICK (Pickle) and the MISC ARM PANEL (ALT REL Button). You can change the callback only in the STICK section. The ALT REL Button works without any callback! It is only visible (not changeable) in the UI to keep the key file complete.

SimRadarGainUp/Down:

You can find the radar gain functions in each of the 4 MFD sections. But you can change the key assignments only in LMFD section. In all other sections you will be referred to the LMFD section.

The TRIM Reset function in the TRIM & STICK sections, the ALT REL Btn. on the MISC Panel and the RadarGain functions in RMFD, TMFD & FMFD sections are assigned to the callback SimDoNothing, with no keys assigned (Not editable).





10.10 Troubleshooting:

10.10.1 Why does BMS crash when loading a key file?

Check your key file for syntax errors. As we do not have external tools for checking the key files (like a debugger) we have to use BMS. Of course you can check your key file manually. But this could be a rather time-consuming task.

It is better to make a backup of your key file and to open it with an editor. Now delete half of the code and check again in BMS. If the file is ok the issue occurs most likely in the second half of the code which you have deleted.

Now you can go on by halving the code of the second part and check this file again. Go on this way until you found the corrupt code line. Once found you can examine what was going wrong here.

BMS can also crash when the key file uses a keyboard key which is not compatible with your locale. A typical example is the “<” key (Key file code is 0x56) on a German keyboard layout. If you try to use this code with an US International locale, BMS will crash.



10.10.2 Stuck key:

This issue has been seen quite often in the forum. A stuck key issue appears when you are using shifted functions. In this case it does not matter if you are using the shifted function with your keyboard (key with modifier/s) or with your DX input device (Pinky Shift with DX button). A popular example is the TRIM function.

The reason for a stuck key is a mishandling of the key press sequence. Here is the way how to do it properly:

1. Press and hold modifier key (keyboard) / Pinky switch (DX Shift)
2. Press keyboard key / DX button and hold it
3. Continue holding both (1 & 2) as long as desired
4. Release keyboard key / DX button
5. Release modifier key (keyboard) / Pinky switch_(DX Shift)

If you release #4 & #5 in the wrong order the key(s) / function will most likely stuck. To prevent this either use the correct order stated above or try to avoid using shifted functions.





10.10.3 How to enable EyeFly (Free Cam):

We have a free cam (to be activated with the callback: OTWToggleEyeFly). To get it to work you have to execute the Falcon BMS.exe with the command “-ef”, like "Falcon BMS.exe" -ef. Otherwise you will get the message “EyeFly Disabled”.

10.10.4 The mouse does not work anymore in pit:

You have most likely invoked the function “Clickable Pit Mode – Toggle”. This function toggles between Mouse On and Mouse Off in the cockpit. In the key files this function is assigned to Alt + 3.

10.10.5 Keyboard keys and combinations you have to be careful with:

In general you have to be careful with all Windows related shortcuts, especially if you are using windowed mode. A complete list can be found in the BMS Key File Editor.xls on Key Code Data sheet.

Tab:

Alt + Tab
Shft Ctrl + Tab
Shft Alt + Tab
Ctrl Alt + Tab
Shft Ctrl Alt + Tab

The Tab combinations above will bring you back to desktop (if you have luck) or could crash / freeze BMS. Both unwanted actions while enjoying a flight...

Escape:

Ctrl + Esc
Alt + Esc
Shft Ctrl + Esc
Shft Alt + Esc
Ctrl Alt + Esc
Shft Ctrl Alt + Esc

Note: Escape cannot be assigned in the BMS UI! Esc is hardcoded. If you try to assign that key in the UI, it will leave the setup screen immediately.

You have to assign that key manually with an editor if desired. But do not forget to assign “Exit Sim” to another key. Otherwise the only possibilities to exit 3D are Alt + TAB (if working for you), Reset your PC (of course not recommended) or crashing / ejecting your jet (Exit menu will pop up automatically).

The above shown combinations will invoke Windows related functions and thus are not recommended to assign them. The only safe combination is Shift + Esc, which you could assign manually.

Shft Ctrl Alt F / Shft Ctrl O:

In newer Windows 10 versions these combinations do not work anymore. This might change in the future. You should avoid them for now. Standard key files (Full, Basic, Minimum) do not use these. As only Win 10 seems to be affected, the default pit builder file stays unchanged. You should take care of it yourself.





Num Lock:

If you assign the Num Lock key it will toggle between “Disable Num Pad” and “Enable Num Pad”. It has no impact on BMS as the assigned functions on the Num Pad work no matter how the Num Pad status is.

But using this key in a key file will lead to a BMS crash once you try to enter BMS Setup. So this key should not be assigned at all.

Print:

Shft + Print
Ctrl + Print
Alt + Print
Shft Ctrl + Print
Shft Alt + Print
Ctrl Alt + Print
Shft Ctrl Alt + Print

Note: PrtScr cannot be assigned in the BMS UI! When doing so the key mapping field remains empty. In the Sim nothing will happen (except hardcoded Screenshot of course).

See also **(Pretty) Screenshot vs. PrtScr** above.

You could assign this key to any function. But as Screenshots are hardcoded to Print, you would make a screenshot any time the key combinations are invoked. Because of this you should avoid the key combination. However, Pretty Screenshot is allowed for any of the Print combinations except Shift Alt + Print, which opens a Windows function.

Del & Num , (Del)

Ctrl Alt + Del
Ctrl Alt Num , (Del)

Also these combinations will bring you back to Windows.

10.10.6 (Pretty) Screenshot vs. PrtScr:

If you take a deeper look into the key files you will realize that there are no keys assigned to the callbacks ScreenShot and PrettyScreenShot. On the opposite PrtScr button is assigned to SimDoNothing. And here is the reason why:

Windows XP and Windows 7 are handling the PrtScr mapping differently. While assigning PrtScr in XP works pretty well in key files, it does not in Win 7. Because of that the PrtScr key was hardcoded. This means they are working for both and you can take screenshots even without assigning a key in the key file.

Which kind of screenshot PrtScr makes (ScreenShot **or** PrettyScreenShot) is defined in the Falcon BMS.cfg (section Misc Settings).

`set g_bPrettyScreenShot 0` = Screenshot (with text overlays)

`set g_bPrettyScreenShot 1` = Pretty Screenshot (without text overlays)

Additionally you can assign keys to the callbacks ScreenShot and PrettyScreenShot. They are set to “no keys assigned” by default.





If you assign PrtScr manually in the key file Win XP and 7 will behave differently. While it does simply nothing in Win 7 it will do two things at the same time in XP: It makes a screenshot (due to hardcoded PrtScr) and invokes the callback PrtScr is manually assigned to.

10.10.7 Why we do not hear cockpit sounds when pressing a key:

As described above the 2D Pit ID numbers are old remains from “2D Pit days”. As we do not have 2D Pits nowadays we do not need them anymore. Also the 4-digit numbers (referenced in the 16_ckpt.dat), which could invoke cockpit sounds when pressing the assigned key, are also outdated now. So if you still use them in your key file you will not hear a sound. Of course the same is true if the sound ID is set to -1.

The following list shows all 3-digit numbers and the corresponding sound files, which are currently used in the default key files and the 3dckpt.dat files. These can be found in the f4sndtbl.txt. If you want to hear sounds when pressing a key (keyboard) or button (DX device) for a specific cockpit function you have to implement them as described earlier in this document. The sound IDs in correlation with the sounds above are listed in order of their first appearance.

Sound ID	Sound file
115	cockpit\togglil.ogg
116	cockpit\rtryknob.ogg
117	cockpit\ejectlvr.ogg
118	cockpit\geardwn.ogg
119	cockpit\gearup.ogg
120	cockpit\icp1.ogg
121	cockpit\icp2.ogg
122	cockpit\icpmntry.ogg
123	cockpit\jettison.ogg
124	cockpit\knobbig.ogg
125	cockpit\knoblil.ogg
126	cockpit\mfd.ogg
127	cockpit\momntary.ogg
129	cockpit\togglbig.ogg
309	cockpit\trimwheel.ogg
310	cockpit\toggsafe1.ogg
311	cockpit\toggsafe2.ogg
312	cockpit\pushbutton_in.ogg
313	cockpit\pushbutton_out.ogg
314	cockpit\pushbutton.ogg
315	cockpit\seatarm.ogg
316	cockpit\toggguard_in.ogg
317	cockpit\toggguard_out.ogg
318	cockpit\rwrbutton.ogg
319	cockpit\magswitch_in.ogg
320	cockpit\magswitch_out.ogg
321	cockpit\altgeardwn.ogg
322	cockpit\cnpyhndl.ogg





10.11 Notes on the Development Callbacks

10.11.1 Introduction:

Before you ask:

The development callbacks will not have the slightest benefit for the casual Pilot. They are purely meant for development purposes.

So if you do not know what they are for and do not develop things such as theatres, 3d models or alike just keep your fingers away from them.

Note: They are purely meant for development purposes. That is also the reason why they are hidden from the UI.

However, before you can use the development callbacks in game you have to activate them in the Faclon BMS.cfg by setting `set g_bActivateDebugStuff` and `set g_bDevelopmentCallbacks` to **1**.

```

////////////////////////////////////
// Debug/Dev Settings (not available in the Config.exe) //
////////////////////////////////////

set g_bActivateDebugStuff 1 // Debug Setting
set g_nShowDebugLabels 0 // Debug Setting
set g_bCampLabels 0 // Debug Setting
set g_bAIprofile 0 // Debug Setting
set g_bShowMipUsage 0 // Debug Setting
set g_b3DClickableCockpitDebug 0 // Debug Setting
set g_bFake3dpit6DOF 0 // Debug Setting
set g_bDevelopmentCallbacks 1 // Dev Setting
set g_bEnableCombatAP 0 // Dev Setting
    
```

Note: Development callbacks are host controlled in MP. So if you expected a chance to cheat, sorry 😊

These are the default key assignments:

6.08 DEVELOPMENT							
SimDoNothing	yes	none	none	no	REM: See Key File Manual for details	locked	
DEV_OTWToggleLocationDisplay	yes	none	F1	yes	DEV: Location Display - Toggle	hidden	Dev Location
DEV_SimCycleDebugLabels	yes	none	F2	yes	DEV: Debug Labels - Cycle	hidden	Dev Debug
DEV_OTWSetScale	yes	none	F3	yes	DEV: Set Scale	hidden	Dev Set Scale
DEV_OTWScaleDown	yes	none	F4	yes	DEV: Scale Down	hidden	Dev Scale Down
DEV_OTWScaleUp	yes	none	F5	yes	DEV: Scale Up	hidden	Dev Scale Up
DEV_SimRegen	yes	none	F6	yes	DEV: Regenerate Mission (Dogfight only)	hidden	Dev Regen
DEV_OTWEnterPosition	yes	none	F7	yes	DEV: Enter Position (EyeFly only)	hidden	Dev Position

Note: Dev callbacks uses key combo Alt C:





10.11.2 Dev Functions:

We will now have a look at the dev callbacks.

DEV_OTWToggleLocationDisplay

This shows you Memory usage (top left corner)...



...and other simulation related information, such as your own x / y / z position, aircraft attitudes, draw calls etc. (top right corner)



X, Y, Z coordinates (Own position on the map):

Shows the current location of your AC where X=0 & Y=0 is the bottom left corner of the map and X=1024 & Y=1024 is the top right corner.

The Z value represents the altitude of the AC. Please note that the value must be always negative. Z=-0 is 0ft. MSL.

Heading, Pitch, Roll (AC attitude):

HDG : - positive values (+0 to +180) = heading 0-180°
- negative ones (-0 to -180) = heading 180-360°

Pitch: - positive values (0 to 180°) = above horizon
- negative ones (-0 to -180°) = below the horizon.
- Note: Reference is the Gun Cross, not FPM

Pitch: - positive values (0 to 180°) = right bank
- negative ones (-0 to -180°) = left bank

Other data:

Shows some information about what is processed nearby the own position. This can give you some hints and clues about the overall performance.

Very useful!

Note: Devs, please watch your draw calls. Very (!) important!





DEV_SimCycleDebugLabels



Shows some details about the entities nearby, such as name, status and owner.

DEV_OTWSetScale, DEV_OTWScaleDown & DEV_OTWScaleUp

Scale Down & Scale Up just work if you have activated Set Scale first. Invoking Scale Up after Set Scale will bring it back to normal.

Note: Btw., as a matter of fact you can and will crash if you fly into scaled up buildings...

Normal view:





Using DEV_OTWSetScale (Oops!):



Adjusted with Scale down a bit (multiple times DEV_OTWScaleDown):





DEV_OTWEnterPosition

This works just with EyeFly! You can simply enter coordinates to bring you immediately to a new position on the map. This is useful in case you want to have a quick look at a specific location on the map.

To activate EyeFly you have to append the command line “-ef” to the exe. EF is activated with the key “Ctrl 0”.

Simply adjust the coordinates with the keyboard (note: Num Pad does not work). Use decimal as separator and finish your input with Enter. The AC position is not changed at all.

You have to enter three values to change your position: X, Y and Z coordinates.

X / Y coordinates:

Both values are measured in km. The overall map size is usually 1024km x 1024km (E.g. stock KTO).

X=0 | Y=1024

X=1024 | Y=1024



X=0 | Y=0

X=1024 | Y=0

X=0 & Y=0 is the bottom left corner of the map and X=1024 & Y=1024 is the top right corner.

The centre of the map is X=512 | Y=512.

To get the coordinates of an objective of interest you can look it up in the database (Editor) or Mission Commander.

Z coordinates:

Z coordinates represent the altitude above main sea level. Please note that Z-values are always negative, thus need a leading “-“.Please be aware that the US QUERTY keyboard layout is used here. So depending on your locale it might be at a different spot on the keyboard.

Input window:



In the top row you see which data is changed with your input (X, Y, Z coordinates).

In the middle part you see the x / y coordinates. They start at 0.00 at the beginning.

In the bottom input field you see the current position first. If you have entered a new coordinate the middle





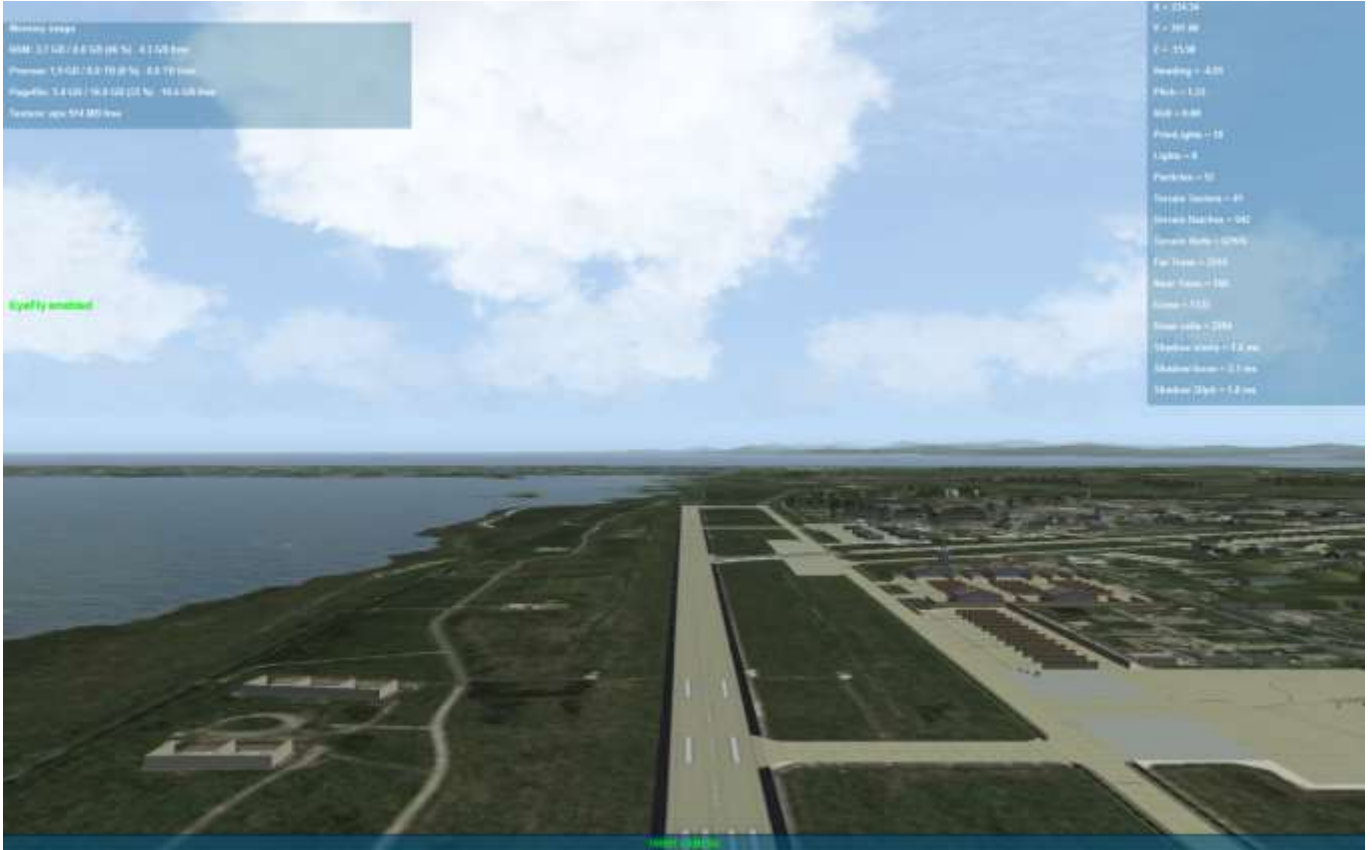
In what follows we give you a full example about how to change your location using EyeFly. In this example you start on the Runway at Kunsan.

Pressing “Alt C: F1” invokes the Location Display which shows your current position:





Switch to EyeFly Cam pressing “Ctrl. 0”:



A new popup window appears. Here you have to enter the new coordinates.



X coordinates:

As described above the middle row shows 0.00 coordinates. The bottom row shows your current X-coordinate.



X coordinates:

Simply enter the new X coordinate, in this example:

6 5 2 . 4 0

Press Enter.





Y coordinates:

X coordinate is now displayed in the middle row.
Current Y location is shown in the input field.



Y coordinates:

Enter the new Y coordinate:

3 6 6 . 7 0

Press Enter.



Z coordinates:

Y coordinate is now also displayed in the middle row.
Current Z location is shown in the input field.



Z coordinates:

Enter the new Z coordinate. Do not forget:
Z coordinates have to be entered with a leading “-“
e.g. -3500.00 = 3500.00ft AMSL:

- 3 5 0 0 . 0 0

Note: If you enter a Z coordinate which is below the terrain level you simply start on ground level.

Once you have entered the X, Y, Z coordinates your EyeFly location will be updated.





New location: This is directly above the KOTAR Range.



DEV_SimRegen

This just starts a fresh Dogfight round.





11. Notes on the 4.34 Comms system

11.1 Introduction

BMS 4.34 introduced a new comms system has a deep impact on the way we communicate with the BMS environment. This chapter shall give you a short overview about the changes. Any further details are explained in the separate Comms Handbook, Training Manual and here in the Technical Manual as well.

Let us first have a look at the place where the changes are most apparent. The briefing page.

COMM LADDER:				
AGENCY:	BALLSIGN:	UHF [QHNL]:	VHF [QHNL]:	NOTES:
INTRA-FLIGHT:	Panther1	--	143.800 MHz	Flight Management Comms
GUARD:	None	243.000 MHz [13]	121.500 MHz [13]	Distress / Emergency
COMMON:	None	339.772 MHz [15]	123.500 MHz [14]	Advisory / UNICOM
BASE OPS:	None	226.175 MHz	--	Homeplate Operations
CHECK-IN:	Sentry1	296.625 MHz [5]	--	AWACS: Global Check-In
TACTICAL:	Sentry1	293.450 MHz	--	AWACS: Package Comms
TANKER / AAR:	Texaco1	293.450 MHz	--	Boom Operator
DEP ATIS:	Kangnung ATIS	--	132.050 MHz	Departure Airbase
DEP GROUND:	Kangnung Ground	275.800 MHz	--	Departure Airbase
DEP TOWER:	Kangnung Tower	334.900 MHz	126.200 MHz [4]	Departure Airbase
DEP DEPARTURE:	Kangnung Departure	304.000 MHz	--	Departure Airbase
ARR ATIS:	Kangnung ATIS	--	132.050 MHz	Recovery Airbase
ARR APPROACH:	Kangnung Approach	304.000 MHz	--	Recovery Airbase
ARR TOWER:	Kangnung Tower	334.900 MHz	126.200 MHz [4]	Recovery Airbase
ARR GROUND:	Kangnung Ground	275.800 MHz	--	Recovery Airbase
ALT ATIS:	Sokcho ATIS	--	123.150 MHz	Alternate Airbase
ALT APPROACH:	Sokcho Approach	304.400 MHz	--	Alternate Airbase
ALT TOWER:	Sokcho Tower	236.600 MHz [10]	122.200 MHz	Alternate Airbase
ALT GROUND:	Sokcho Ground	240.400 MHz	--	Alternate Airbase

As you can see we have a number of new agencies to communicate with. The corresponding frequencies and presets are assigned automatically. This is not only in TEs, Training missions and Campaigns the case, but also behind the scenes (as you have no briefing page) in Dogfight and Instant Action modules.

The frequencies for Homeplate Operations, ATIS, Departure, Recovery and Alternate Airbases are retrieved from the stations+ils.dat. Everything else is stored in the new radiomap.dat located in the '.../campaign' folder.

Let us now go into more detail.





11.2 The Ground agencies

As for the airbase operations we have a set of new agencies. From ramp to landing the flow is like this: Base Ops, DEP Ground, DEP Tower, DEP Departure ... ARR Approach, ARR Tower, ARR Ground. Please note that Departure and Approach are in fact one and the same agency if DEP and ARR are planned to be the same airbase. Otherwies it might differ.

Each ground-based agency has a unique UHF and VHF (if applicable) frequency. However, not all agencies have both as you can see in the screenshot.

Airstrips are a bit different as they just have VHF / UHF frequencies for the Tower.

Carriers are treated the same as airbases btw. On the other hand they have in addition a LSO agency which is usually on the same frequency as the tower.

For more details please refer to the before mentioned manuals.

11.3 The Air agencies

As mentioned above the air agency frequencies are stored in the radiomap.dat. This file contains all UHF / VHF frequencies for the following agencies: Intra Flight, Guard, Common, Check In, Tactical and Tanker / AAR.

While Guard and Common are fixed frequencies, which never change, the others are tied to the Flight callsign directly. At the moment we have a total of 169 Flight callsigns. It was 160 before but we added Mirage, Rafale, Lynx, Fenwick, Tornado, Magic, Nimrod, Puma and Gazelle in 4.34.

As each callsign gets an additional number from 1 to 9 this sums up to 1521 (169 x 9) different callsigns (e.g. Falcon1, Falcon2 ... Falcon9 etc.).

Here is a brief summary of how the frequencies get assigned to the different agencies:

Guard / Common (UHF and VHF):

These are for Distress / Emergency (Guard) and Advisory / Unicom (Common) purposes. These are the only air agencies which have a UHF and VHF frequency assigned. They will never change.

Intra Flight (VHF only):

This is for flight internal comms. It is the VHF frequency tied to the flights callsign. Let us assume you are in the flight Banshee 6. The VHF tied to Banshee6 will be your Intra Flight frequency.

Check In (UHF only):

This is the global check In frequency. The UHF frequency is tied to the AWACS callsign. So if there is an AWACS on station with the callsign Dragnet1 the UHF is tied to that callsign.

Tactical (UHF only):

This is for Package Comms. The UHF frequency is tied to the primary flight of the package. This is usually the first flight in the package list (either briefing or ATO). Short example:





You task a package in the TE builder consisting of Strike, SEAD and Escort. The Strike, let us assume his callsign is Falcon1, is tasked as the first flight in the package. Tactical UHF will be tied to the callsign Falcon1.

Tanker (UHF only):

This is for communication with a tanker (obviously). Tanker is treated the same as Check In. UHF is tied to the tankers callsign.

UHF – General notes:

Each of the 1521 flight callsigns has a unique UHF frequency assigned. There are no conflicts possible with neither other flight callsigns nor ground agencies.

VHF – General notes:

This story is less prone to conflicts. At the current state we use 320 VHF frequencies in the range of 136.025 to 144.000 which are exclusively for air operations. As the frequencies are tied to the callsigns directly via the radiomap.dat you can imagine that conflicts are quite possible. We will come back on this later.

11.4 Presets

All presets are stored as shown by the examples in the following list:

DEP Airbase = ARR Airbase, ALT = Airbase			DEP Airbase = ARR Airbase, ALT = Airstrip			DEP = Airbase, ARR = Airstrip, ALT = Airbase		
Preset	Agency UHF	Agency VHF	Preset	Agency UHF	Agency VHF	Preset	Agency UHF	Agency VHF
1	Base Ops		1	Base Ops		1	Base Ops	
2	DEP Ground		2	DEP Ground		2	DEP Ground	
3	DEP Tower	DEP/ARR Tower	3	DEP Tower	DEP Tower	3	DEP Tower	DEP Tower
4	DEP Departure		4	DEP Departure		4	DEP Departure	
5	Check In		5	Check In		5	Check In	
6	Tactical		6	Tactical		6	Tactical	
4	ARR Approach		7	ARR Approach		7	ARR Approach	
3	ARR Tower		8	ARR Tower	ARR Tower	7	ARR Tower	
2	ARR Ground		9	ARR Ground		7	ARR Ground	
10	ALT Ground		10	ALT Approach		8		ARR Tower
11	ALT Approach	ALT Tower	10	ALT Tower		10	ALT Approach	
12	ALT Tower		10	ALT Ground		11	ALT Tower	ALT Tower
13	Tanker		11		ALT Tower	12	ALT Ground	
14	Common	Common	13	Tanker		13	Tanker	
15		Intra Flight	14	Common	Common	14	Common	Common
16		Intra Flight	15		Intra Flight	15		Intra Flight
17		Intra Flight	16		Intra Flight	16		Intra Flight
18		Intra Flight	17		Intra Flight	17		Intra Flight
19		Intra Flight	18		Intra Flight	18		Intra Flight
			19		Intra Flight	19		Intra Flight

As you can see the UHF and VHF presets are dependent on various factors, e.g. whether DEP, ARR or ALT is an airbase or an airstrip. What stays always the same is:

- Base Ops
- Check In / Tactical
- Tanker
- Common
- Intra Flight





11.5 DTC Storage

All frequencies will be stored in the DTC automatically. There is no need to take further actions apart saving the DTC prior entering the 3d world. But be aware, if you change the arrival or alternate airbase or add a tanker waypoint the frequencies will also change. In this case you have to save the DTC again to make sure to have the correct frequencies stored.

11.6 The VHF assignment system

As mentioned earlier in this chapter we face some problems in regards of VHF usage. While we have a set of total 1521 flight callsigns but just 320 VHF frequencies it is obvious that this cannot work as in theory roughly one fifth of every flight uses the very same VHF frequency.

In preparation on finding a solution a lot of testing was going on. We analyzed a couple of hundred save games while testing different campaigns under different circumstance. Based on that we can come up with a solution which can avoid that kind of problems with almost 100% certainty.

Analysis:

The main focus on the analysis phase was to get a worst case scenario under which a specific type of aircraft will task flights in a short period of time. The pure number of flights is not the only thing which is important but also in time frame. Flights, which T/O time is 2.5 hours apart is most likely not at risk of interfering each other as the first flight will have landed before the other takes off. The following screenshot should give you an example for the F-16:

Worst case:		Test13		Test29		Test34	
Flights		123		93		102	
Takeoff Time							
05:00	05:30	5				1	
05:30	06:00	10				7	
06:00	06:30	14				15	79 Flights are at risk to interfere (worst case). This is roughly 78% of all tasked flights max.
06:30	07:00	7				18	
07:00	07:30	15	61 Flights are at risk to interfere (worst case). This is roughly 50% of all tasked flights max.			19	
07:30	08:00	8		1		14	
08:00	08:30	12		7		13	
08:30	09:00	8		15	61 Flights are at risk to interfere (worst case). This is roughly 66% of all tasked flights max.	11	
09:00	09:30	18		16		0	
09:30	10:00	6		5		2	
10:00	10:30	6		7		0	
10:30	11:00	7		18		1	
11:00	11:30	4		5			
11:30	12:00	3		9			
12:00	12:30			6			
12:30	15:00			6			
Calculating the worst case							
max flights are		123		callsigns needed	22	96 divided by 4.5 (rounded)	
max chance of conflicts		78%		# individual frequencies	99	22 * 4,5	
Result		96		total # of callsign slots	198	22 * 9	





Based on such data, which we have done for all aircraft types (multiple times as this just shows the worst case scenario), we calculated how many callsigns and how many frequencies should be assigned to a specific aircraft type. We revised the callsign slots in the database according to the analysis we did.

Additionally we arranged the aircraft types according to their importance. Aircraft of higher priority are likely more often used by human vPilots. The higher the priority the more individual callsigns and the more discrete frequencies are assigned to that aircraft. In conclusion this means also that aircraft of lower importance are more likely ending up with VHF conflicts.

In addition this is of course also based on the result of the analysis that some aircraft are tasked less frequently in a campaign as others.

Classification of aircraft types:

So, we came up with the following classification:

Primary aircraft:

This is the F-16. It has by far the most discrete callsigns and frequencies as this is the aircraft which most likely will be flown by most people.

Secondary aircraft:

These are frequently flown by humans as well, such as F-18s, Harrier, Mig-29s etc. Secondary blue / red AC types will share the same set of frequencies. However, there is a low risk of interference due to a clever design. ☺ As a rule of thumb, there have to be more that e.g. 30 flights of both (!) F-18s and Mig-29s tasked at the very same (!) time period to get in trouble. This risk is based on the analysis really low.

Tertiary aircraft:

These are probably rarely used by humans, such as J-5, AMX, B-1, Q-5, F-117... The risk of VHF interference is high.

Quaternary aircraft:

They are unlikely used by human's aka. it does not make sense. All Transport, AWACS and Tanker flights are in this category. Also U-2, AC-130 and such. These aircraft and the assigned callsigns share one and the same VHF frequency. Even if there are humans picking up such flight, there are for sure not those many at the same time.

Non human-flyable aircraft:

All helicopters. As it stands now they cannot be flown by humans. They also share just one VHF frequency.

Drawbacks and other notes:

The whole system is optimized for campaigns and not altered TEs. See later notes on things you have to keep in mind. The casual user should not face any problems with the new comms system.

Another thing we would like to note:

Humans on either side can communicate on the same VHF / UHF frequency. However, this does not apply to AI and although not realistic the pure AI to AI comms will be suppressed for human players' convenience.





Callsign assignments to aircraft types:

While we were on it we revised the callsign assignments to aircraft types completely. This is based on research on real life callsign usage of different aircraft using most current and older data from the 80s which can be found on the internet. On the other hand compromises have to be made for stability reasons. Here is an example:

Aircraft	Callsigns	Aircraft	Callsigns	Aircraft	Callsigns	
F-16 KF-16	Falcon	F-14	Tomcat	Tanker	Texaco	
	Gamble		Wildcat		Copper	
	Hawkeye		F-18	Diamond	AWACS JSTARS	Chalice
	Plasma			Hornet		Sentry
	Warhawk	Avenger		Airlift	Dragnet	
	Cyborg	Wagon			Magic	
	Mudhen	Fist			Blazer	
	Weasel	Gypsy			Cheetah	
	Bulldog	Vulture			Lion	
	Cajun	Sting			Boxer	
	Beast	Bug			Camel	
	Satan	Phantom			Hobby	
	Mustang	Magnet	Elvis			
	Panther	Roach	Stallion			
	Banshee	Zipper	Jumbo			
	Spade	Vandal	Shark			
	Viper	Stud	Blackjack			
	Jaguar	Trojan	Jolly			
	Mako	Rider	Nighthawk			
	Jackal	Lightning	Spartan			
Cobra	Vapor	Rescue				
Hammer	Mirage	Angel				
	Rafale	Blackhawk				
	Tornado	Sparky				
				Heli		

So, basically you can say goodbye to the old and well known callsigns you are used to since years. There are some exceptions, though. One for example is that an aircraft of a specific type should get its native callsign. So the F-16 still is Falcon, F-14 Tomcat or F-4 Phantom.

11.7 How to avoid VHF conflicts

There are three ways to cause conflicts:

- Change flight callsigns via MC (TE and Campaign)
- Add more than x flights of a specific airframe while T/O time is conflicting (TE and Campaign)
- Change TO times of ATM planned flights (Campaign only)

Let us address them in short one by one.

Some TE builders tend to change flight callsigns via Mission Commander. This can introduce conflicts. Fortunately, MC has a feature to make the mission builder aware of that. So if you change any flight callsigns make sure, that at least the human occupied flights (AI flights are not a problem here) use callsigns with no VHF interferences. Otherwise you will end up with 2+ flights sharing the same VHF frequency which is for sure not a preferred scenario.





The second example is quite easy explained:

If you task 100 flights of the same aircraft type which are in the air at the same time it is obvious that problems occur. To give you an example the F-18 can have 30 discrete VHF frequencies at the same time. After that they will repeat. If you task the 31st F-18 and all of them are in the air at the same time -> disaster strikes.

Same case for flights which are tasked by the Air Tasking Manager (Campaign engine). When you set new Take Off times extensively and too many AC-types are tasked at the same time... You have guessed it.

11.8 Fallback solution and AI

As it is not 100% certain to avoid any kind of issues in regards of VHF there is a set of 15 spare frequencies which are not tied to any flight callsign thus are free to use.

These are easy memorizable frequencies within the range of 137.000 - 144.000 ending on .000 and .500:

137.000	138.500	140.000	141.500	143.000
137.500	139.000	140.500	142.000	143.500
138.000	139.500	141.000	142.500	144.000

In case of conflicts you can try to use any of them. If you change your automatically assigned VHF frequency to any of those spares there is currently no warning in case another flight uses it at the moment.

Be also advised, and this is a really important remark, if you change the VHF frequency you will immediately lose all comms to your AI wingman. So this is preferably used by human occupied flights only as the AI is unable to follow in case VHF has changed.

11.9 3rd party theater dev notes

As mentioned above the comms system is optimized for stock BMS campaigns and TEs.

You might want to have a look at the possibilities to avoid VHF conflicts in your campaign. This is especially true for PvP scenarios. You can do a lot to avoid this by just altering the radiomap.dat to your needs.

Just make sure you do not have any conflicts with the ground agencies. So, you should always cross check with the station+ils.dat to avoid conflicts in that regard.

11.10 Stock BMS VHF layout

If you want to have a look at how things are assigned by default for stock KTO and TVT just have a look at the radiomap.xls located in “\Docs\05 Other Documentation”





12. User File Structure

12.1 Structure of the TEmissionname.ini

An example is shown on the next page (wpntarget 2-98 are removed to save space).

Note that flight plan waypoint coordinates shown on the UI map also get exported to this ini file as TGT STPTs, except if an explicit TGT STPT was defined for a waypoint's index. In this case the user-defined assigned using the recon UI one is saved into the file in place of the default coordinate set from the flight plan steerpoint for the given index.

In order to distinguish a flight plan waypoint from a user-assigned TGT STPT an additional integer is exported, which is -1 in case of an undefined or TGT STPT (if undefined x,y & z coordinates will be 0), otherwise this integer is the WaypointClass::Action member, a positive integer representing the action for the waypoint (LAND, TAKEOFF, etc.).

Keep in mind that flight plan waypoints previously stored in the ini file will not be loaded into the DTC when you use the LOAD button in the DTC UI page (this would not make sense anyway). Other details from old .ini files should be loaded correctly of course.

Note: wpntarget steerpoints are new to 4.33 and are used only to simulate the pre-planned targets feed for SPICE bombs. A new set of 100 special WPN TGTs information was added to the UI and to the missions/callsign.ini files. In the UI, only select WPN TGTs from the drop-down box if you are assigning weapon targets for SPICE bombs; otherwise leave the drop down box set to STPTs and follow the guide listed above. For more information refer to the SPICE bomb section of the TO BMS1-F16CM-34-1-1.

```
[MISSION]
title=TEmissionname
[STPT]
target_0=0.000000, 0.000000, 0.000000, -1
target_1=0.000000, 0.000000, 0.000000, -1
target_2=0.000000, 0.000000, 0.000000, -1
target_3=0.000000, 0.000000, 0.000000, -1
target_4=0.000000, 0.000000, 0.000000, -1
target_5=0.000000, 0.000000, 0.000000, -1
target_6=0.000000, 0.000000, 0.000000, -1
target_7=0.000000, 0.000000, 0.000000, -1
target_8=0.000000, 0.000000, 0.000000, -1
target_9=0.000000, 0.000000, 0.000000, -1
target_10=0.000000, 0.000000, 0.000000, -1
target_11=0.000000, 0.000000, 0.000000, -1
target_12=0.000000, 0.000000, 0.000000, -1
target_13=0.000000, 0.000000, 0.000000, -1
target_14=0.000000, 0.000000, 0.000000, -1
target_15=0.000000, 0.000000, 0.000000, -1
target_16=0.000000, 0.000000, 0.000000, -1
target_17=0.000000, 0.000000, 0.000000, -1
target_18=0.000000, 0.000000, 0.000000, -1
target_19=0.000000, 0.000000, 0.000000, -1
target_20=0.000000, 0.000000, 0.000000, -1
target_21=0.000000, 0.000000, 0.000000, -1
target_22=0.000000, 0.000000, 0.000000, -1
target_23=0.000000, 0.000000, 0.000000, -1
ppt_0=0.000000, 0.000000, 0.000000, 0.000000,
ppt_1=0.000000, 0.000000, 0.000000, 0.000000,
ppt_2=0.000000, 0.000000, 0.000000, 0.000000,
```



ppt_3=0.000000, 0.000000, 0.000000, 0.000000,
ppt_4=0.000000, 0.000000, 0.000000, 0.000000,
ppt_5=0.000000, 0.000000, 0.000000, 0.000000,
ppt_6=0.000000, 0.000000, 0.000000, 0.000000,
ppt_7=0.000000, 0.000000, 0.000000, 0.000000,
ppt_8=0.000000, 0.000000, 0.000000, 0.000000,
ppt_9=0.000000, 0.000000, 0.000000, 0.000000,
ppt_10=0.000000, 0.000000, 0.000000, 0.000000,
ppt_11=0.000000, 0.000000, 0.000000, 0.000000,
ppt_12=0.000000, 0.000000, 0.000000, 0.000000,
ppt_13=0.000000, 0.000000, 0.000000, 0.000000,
ppt_14=0.000000, 0.000000, 0.000000, 0.000000,
lineSTPT_0=0.000000, 0.000000, 0.000000
lineSTPT_1=0.000000, 0.000000, 0.000000
lineSTPT_2=0.000000, 0.000000, 0.000000
lineSTPT_3=0.000000, 0.000000, 0.000000
lineSTPT_4=0.000000, 0.000000, 0.000000
lineSTPT_5=0.000000, 0.000000, 0.000000
lineSTPT_6=0.000000, 0.000000, 0.000000
lineSTPT_7=0.000000, 0.000000, 0.000000
lineSTPT_8=0.000000, 0.000000, 0.000000
lineSTPT_9=0.000000, 0.000000, 0.000000
lineSTPT_10=0.000000, 0.000000, 0.000000
lineSTPT_11=0.000000, 0.000000, 0.000000
lineSTPT_12=0.000000, 0.000000, 0.000000
lineSTPT_13=0.000000, 0.000000, 0.000000
lineSTPT_14=0.000000, 0.000000, 0.000000
lineSTPT_15=0.000000, 0.000000, 0.000000
lineSTPT_16=0.000000, 0.000000, 0.000000
lineSTPT_17=0.000000, 0.000000, 0.000000
lineSTPT_18=0.000000, 0.000000, 0.000000
lineSTPT_19=0.000000, 0.000000, 0.000000
lineSTPT_20=0.000000, 0.000000, 0.000000
lineSTPT_21=0.000000, 0.000000, 0.000000
lineSTPT_22=0.000000, 0.000000, 0.000000
lineSTPT_23=0.000000, 0.000000, 0.000000
wpntarget_0=0.000000, 0.000000, 0.000000, -1, Not set
wpntarget_1=0.000000, 0.000000, 0.000000, -1, Not set
...
wpntarget_99=0.000000, 0.000000, 0.000000, -1, Not set



12.2 Structure of the CALLSIGN.INI

The Callsign.ini file is located in the User\Config\ folder and will store a series of variables specific to the pilots. Some of them may be redundant with the TEmissionname.ini file.

It features multiple sections which are written by setting the different tabs of the UI DTC:

- **EWS:** Stores the Electronic Warfare System settings from the DTC, basically the way your 6 dispense programs are set. PGM 0 to 3 are the 4 CMDS mode released by CMS forward. PGM 4 is the slap switch and PGM 5 is the CMS left release.
- **MFD:** Stores the way your Multi Function Displays are programmed in your DTC.
- **Bullseye:** Store if the Bullseye should be displayed on your MFD.
- **IFF:** Store the IFF modes, settings and events programmed in your DTC.
- **STPT:** Stores the position of the INS (1-24) and Target (81-99) steerpoints in your DTC. Please note in the file structure, they are all named Target steerpoints and are redundant with the TEmissionname.ini file
- **Radio:** Links Preset to both VHF & UHF radios as well as the comment line for each entry.
- **Comms:** Stores the default radio settings for both COM1 & COM2 when the players enters 3D. When using WDP (Weapon Delivery Planner) to setup some options, the comms section might be more important with settings relevant to the TACAN, ILS , ...
- **ICP:** This section is created by a cockpit save state or by WDP. It will save settings entered in the UFC as default which will be reused everytime you enter the cockpit. (example: default master mode, wingspan, bingo value, A-LOW value.
- **FCC_AGB:** This section is created by a cockpit save state or by WDP. It will save weapons settings entered in the SMS CNTL page. (example: release spacing, ripple quantity, arming delay, ...)
- **Map_POP:** Stores the UI MAP legend option. These are not from the DTC but from the settings you set the last time you used the UI map options (for instance object labels or chosed to display airbases and infrastructure icons. Theses settins are saved so you can find the same set of options next time you visit the UI map.
- **NAV OFFSET:** This section is created by a cockpit save state or by WDP. It will save the VIP and VRP, OA1 & OA2 settings.
- **LASER:** This section is created by a cockpit save state or by WDP. It will save the laser settings from the UFC laser page (laser time, laser and LST code).
- **FCC_AIM:** This section is created by a cockpit save state or by WDP. It will save the air to air missile settings (aim-9 & aim-120) from the SMS pages.
- **FCC_AGM:** Same as above but got the Air to Ground missiles (mavericks – SMS page)
- **COLORS:** Avionics Configurator Colour section (See chapter 3.1)

12.2.1 Example of a callsign ini file:

[EWS]	Reqctr=1	Bingo=1
Fdbk=1	Flare Bingo=15	Chaff Bingo=24
PGM 0 Chaff BQ=5	PGM 0 Chaff BI=75	PGM 0 Chaff SQ=1
PGM 0 Chaff SI=0	PGM 0 Flare BQ=0	PGM 0 Flare BI=0
PGM 0 Flare SQ=0	PGM 0 Flare SI=0	PGM 1 Chaff BQ=3
PGM 1 Chaff BI=100	PGM 1 Chaff SQ=1	PGM 1 Chaff SI=0
PGM 1 Flare BQ=0	PGM 1 Flare BI=0	PGM 1 Flare SQ=0
PGM 1 Flare SI=0	PGM 2 Chaff BQ=9	PGM 2 Chaff BI=150
PGM 2 Chaff SQ=2	PGM 2 Chaff SI=800	PGM 2 Flare BQ=0
PGM 2 Flare BI=0	PGM 2 Flare SQ=0	PGM 2 Flare SI=0
PGM 3 Chaff BQ=1	PGM 3 Chaff BI=0	PGM 3 Chaff SQ=9
PGM 3 Chaff SI=1100	PGM 3 Flare BQ=1	PGM 3 Flare BI=0
PGM 3 Flare SQ=9	PGM 3 Flare SI=1100	PGM 4 Chaff BQ=0
PGM 4 Chaff BI=0	PGM 4 Chaff SQ=0	PGM 4 Chaff SI=0





PGM 4 Flare BQ=1	PGM 4 Flare BI=0	PGM 4 Flare SQ=1
PGM 4 Flare SI=0	PGM 5 Chaff BQ=0	PGM 5 Chaff BI=0
PGM 5 Chaff SQ=0	PGM 5 Chaff SI=0	PGM 5 Flare BQ=4
PGM 5 Flare BI=100	PGM 5 Flare SQ=1	PGM 5 Flare SI=0
Reqjam=1	PGM 0 Comment=Anti-AA Hi Pk	PGM 1 Comment=Anti-SAM Low-pk
PGM 2 Comment=Anti-SAM HI-pk	PGM 3 Comment=Sam & ManPad PopUp	PGM 4 Comment=Show ID
PGM 5 Comment=Anti-Heater Hi Pk	[MFD]	Display0-0-0=8
Display0-0-1=0	Display0-0-2=11	Display0-0-csel=2
Display0-1-0=0	Display0-1-1=0	Display0-1-2=11
Display0-1-csel=2	Display0-2-0=6	Display0-2-1=5
Display0-2-2=11	Display0-2-csel=2	Display0-3-0=0
Display0-3-1=0	Display0-3-2=11	Display0-3-csel=2
Display0-4-0=0	Display0-4-1=0	Display0-4-2=11
Display0-4-csel=2	Display1-0-0=12	Display1-0-1=10
Display1-0-2=7	Display1-0-csel=0	Display1-1-0=12
Display1-1-1=10	Display1-1-2=8	Display1-1-csel=0
Display1-2-0=4	Display1-2-1=10	Display1-2-2=7
Display1-2-csel=1	Display1-3-0=12	Display1-3-1=10
Display1-3-2=8	Display1-3-csel=1	Display1-4-0=12
Display1-4-1=10	Display1-4-2=8	Display1-4-csel=0
Display2-0-0=0	Display2-0-1=0	Display2-0-2=0
Display2-0-csel=0	Display2-1-0=0	Display2-1-1=0
Display2-1-2=0	Display2-1-csel=0	Display2-2-0=0
Display2-2-1=0	Display2-2-2=0	Display2-2-csel=0
Display2-3-0=0	Display2-3-1=0	Display2-3-2=0
Display2-3-csel=0	Display2-4-0=0	Display2-4-1=0
Display2-4-2=0	Display2-4-csel=0	Display3-0-0=0
Display3-0-1=0	Display3-0-2=0	Display3-0-csel=0
Display3-1-0=0	Display3-1-1=0	Display3-1-2=0
Display3-1-csel=0	Display3-2-0=0	Display3-2-1=0
Display3-2-2=0	Display3-2-csel=0	Display3-3-0=0
Display3-3-1=0	Display3-3-2=0	Display3-3-csel=0
Display3-4-0=0	Display3-4-1=0	Display3-4-2=0
Display3-4-csel=0	[Bullseye]	BullseyeInfoOnMFD=0
[STPT]	target_0=0.000000, 0.000000, 0.000000, -1, Not set	[target 1 to 22 removed]
target_23=0.000000, 0.000000, 0.000000, -1, Not set	ppt_0=0.000000, 0.000000, 0.000000, 0.000000,	[ppt 1 to 13 removed]
ppt_14=0.000000, 0.000000, 0.000000, 0.000000,	lineSTPT_0=0.000000, 0.000000, 0.000000	lineSTPT_19=0.000000, 0.000000, 0.000000
0.000000,	[line STPT 1 to 18 removed]	
wpntarget_0=0.000000, 0.000000, 0.000000, -1, Not set	[wpntarget 1 to 98 removed]	wpntarget_99=0.000000, 0.000000, 0.000000, -1, Not set
target_80=0.000000, 0.000000, 0.000000, -1, Not set	[target 81 to 98 removed]	target_99=0.000000, 0.000000, 0.000000, -1, Not set
lineSTPT_20=0.000000, 0.000000, 0.000000	lineSTPT_21=0.000000, 0.000000, 0.000000	lineSTPT_22=0.000000, 0.000000, 0.000000
lineSTPT_23=0.000000, 0.000000, 0.000000	[Radio]	UHF_1=254225
UHF_2=275125	UHF_3=253400	UHF_4=255625
UHF_5=292450	UHF_6=290600	UHF_7=255625
UHF_8=253400	UHF_9=275125	UHF_10=233800
UHF_11=233800	UHF_12=233800	UHF_13=293450
UHF_14=243000	UHF_15=354000	UHF_16=318100
UHF_17=359300	UHF_18=324500	UHF_19=339100
UHF_20=392900	VHF_1=138050	VHF_2=138100
VHF_3=118900	VHF_4=126200	VHF_5=134250
VHF_6=133150	VHF_7=118700	VHF_8=118900
VHF_9=132875	VHF_10=122800	VHF_11=122800
VHF_12=121200	VHF_13=121500	VHF_14=123500
VHF_15=143550	VHF_16=140150	VHF_17=138075
VHF_18=142775	VHF_19=141375	VHF_20=136150
UHF_COMMENT_1=Base ops	UHF_COMMENT_2=DEP Ground	UHF_COMMENT_3=DEP Tower
UHF_COMMENT_4=DEP Approach	UHF_COMMENT_5=(open)	UHF_COMMENT_6=Tactical
UHF_COMMENT_7=ARR Approach	UHF_COMMENT_8=ARR Tower	UHF_COMMENT_9=ARR Ground
UHF_COMMENT_10=ALT Approach	UHF_COMMENT_11=ALT Tower	UHF_COMMENT_12=ALT Ground
UHF_COMMENT_13=(open)	UHF_COMMENT_14=Guard	UHF_COMMENT_15=Frequency out of range detected!
UHF_COMMENT_16=(open)	UHF_COMMENT_17=(open)	UHF_COMMENT_18=(open)
UHF_COMMENT_19=(open)	UHF_COMMENT_20=(open)	VHF_COMMENT_1=(open)
VHF_COMMENT_2=(open)	VHF_COMMENT_3=DEP Tower	VHF_COMMENT_4=(open)
VHF_COMMENT_5=(open)	VHF_COMMENT_6=(open)	VHF_COMMENT_7=(open)





VHF_COMMENT_8=ARR Tower	VHF_COMMENT_9=(open)	VHF_COMMENT_10=(open)
VHF_COMMENT_11=ALT Tower	VHF_COMMENT_12=(open)	VHF_COMMENT_13=Emergency
VHF_COMMENT_14=UNICOM	VHF_COMMENT_15=Flight-1	VHF_COMMENT_16=(open)
VHF_COMMENT_17=(open)	VHF_COMMENT_18=(open)	VHF_COMMENT_19=(open)
VHF_COMMENT_20=(open)	[COMMS]	Comm1=7
Comm2=15	Comm1_Comment=(open)	Comm2_Comment=(open)
TACAN Channel=94	TACAN Band=0	TACAN Domain=0
ILS Frequency=10900	ILS CRS=0	[ICP]
Manual Wingspan=35.000000	Bingo_Fuel=1500.000000	MasterMode=0
Alow AGL=100.000000	Alow MSL=10000	Alow TFAdv=400
[FCC_AGB]	Profile1_Submode=8	Profile1_Fuze=0
Profile1_SGL/PAIR=0	Profile1_Release_Spacing=175	Profile1_Release_Pulse=0
Profile1_Release_Angle=45	Profile1_C1_AD1=400.000000	Profile1_C1_AD2=600.000000
Profile1_C2_AD=150.000000	Profile1_C2_BA=500	Profile2_Submode=7
Profile2_Fuze=1	Profile2_SGL/PAIR=0	Profile2_Release_Spacing=25
Profile2_Release_Pulse=3	Profile2_Release_Angle=45	Profile2_C1_AD1=400.000000
Profile2_C1_AD2=600.000000	Profile2_C2_AD=150.000000	Profile2_C2_BA=500
[MAP_POP]	MapOpt_0=1	MapOpt_1=0
MapOpt_2=1	MapOpt_3=1	MapOpt_4=0
MapOpt_5=0	MapOpt_6=0	MapOpt_7=0
MapOpt_8=0	MapOpt_9=0	MapOpt_10=0
MapOpt_11=0	MapOpt_12=0	MapOpt_13=0
MapOpt_14=0	MapOpt_15=0	MapOpt_16=1
MapOpt_17=1	MapOpt_18=1	MapOpt_19=0
MapOpt_20=1	MapOpt_21=0	MapOpt_22=1
MapOpt_23=0	MapOpt_24=0	MapOpt_25=1
MapOpt_26=0	MapOpt_27=0	MapOpt_28=1
MapOpt_29=1	MapOpt_30=0	MapOpt_31=1
MapOpt_32=0	MapOpt_33=0	
[NAV OFFSETS]	Modesel=2	VIP=6,209.4,60211,400
VIPPUP=6,207.5,33516,400	VRP=7,31.7,44962,400	VRPPUP=7,31.7,26734,400
OA1-6=197.5,46388,6629	OA2-6=197.5,46388,0	OA1-7=62.1,17609,6629
OA2-7=62.1,17609,0	[Laser]	LaserST=8
LaserTGP=1688	LaserLT=1688	[FCC_AIM]
AIM-9_Spot/Scan=0	AIM-9_TD/BP=0	AIM120_TargetSize=2
[FCC_AGM]	Maverick_AutoPwr=0	Maverick_AutoPwrDir=1
Maverick_AutoPwrWpt=3	[IFF]	Mode1 On=1
Mode2 On=1	Mode3A On=0	Mode4 On=1
ModeC On=0	ModeS On=0	Mode1 Code=11
Mode2 Code=5304	Mode3A Code=6604	Mode4 Key=1
AutoChange=2	TIME 0 Mode1 Code=02	TIME 0 Mode3A Code=0104
TIME 0 Mode4 Key=1	TIME 0 Criteria=0900	TIME 1 Mode1 Code=52
TIME 1 Mode3A Code=1354	TIME 1 Mode4 Key=1	TIME 1 Criteria=1000
TIME 2 Mode1 Code=11	TIME 2 Mode3A Code=7554	TIME 2 Mode4 Key=1
TIME 2 Criteria=1100	TIME 3 Mode1 Code=10	TIME 3 Mode3A Code=2330
TIME 3 Mode4 Key=1	TIME 3 Criteria=1200	TIME 4 Mode1 Code=12
TIME 4 Mode3A Code=3104	TIME 4 Mode4 Key=1	TIME 4 Criteria=1300
TIME 5 Mode1 Code=12	TIME 5 Mode3A Code=5054	TIME 5 Mode4 Key=1
TIME 5 Criteria=1400	TIME 6 Mode1 Code=50	TIME 6 Mode3A Code=4754
TIME 6 Mode4 Key=1	TIME 6 Criteria=1500	TIME 7 Mode1 Code=10
TIME 7 Mode3A Code=7204	TIME 7 Mode4 Key=1	TIME 7 Criteria=1600
TIME 8 Mode1 Code=13	TIME 8 Mode3A Code=1430	TIME 8 Mode4 Key=1
TIME 8 Criteria=1700	TIME 9 Mode1 Code=73	TIME 9 Mode3A Code=1704
TIME 9 Mode4 Key=1	TIME 9 Criteria=1800	TIME 10 Mode1 Code=33
TIME 10 Mode3A Code=2630	TIME 10 Mode4 Key=1	TIME 10 Criteria=1900
TIME 11 Mode1 Code=30	TIME 11 Mode3A Code=6230	TIME 11 Mode4 Key=1
TIME 11 Criteria=2000	POS 0 Mode1=1	POS 0 Mode2=1
POS 0 Mode3A=0	POS 0 Mode4=1	POS 0 ModeC=0
POS 0 ModeS=0	POS 1 Mode1=1	POS 1 Mode2=1
POS 1 Mode3A=0	POS 1 Mode4=1	POS 1 ModeC=0
POS 1 ModeS=0	POS 0 WayPoint= 0	POS 0 Direction=0
POS 1 WayPoint= 0	POS 1 Direction=0	





12.3 Notes regarding callsign.ini and TEmissionname.ini

There are some additional notes about TGT STPTs, LINES, and PPTs the pilot needs to be aware of. The first involves the nature of the callsign.ini and TEmissionname.ini.

Note 1

Upon application startup and changing the pilot (via logbook) the MFD/EWS/Radio/... information (everything in the callsign.ini) gets loaded.

Upon loading a TE, the TEmissionname.ini file gets loaded, if it exists. This loads TGT STPTs/LINES/PPT info *on top* of the MFD/EWS/Radio info. If no TEmissionname.ini exists, TGT STPTs/LINES/PPT are reset to defaults.

Upon loading a campaign mission, TGT STPTs/LINES/PPT (in the callsign.ini) are reset to the defaults (i.e., they are deleted).

Upon saving the DTC in the TE module, TGT STPTs/LINES/PPT info is saved to *both* callsign.ini and TEmissionname.ini.

Upon saving the DTC in a campaign mission, TGT STPTs/LINES/PPT info *only* gets saved to callsign.ini. This is due to technical reasons.

TGT STPTs/LINES/PPT info gets saved to callsign.ini in both the TE module and campaign module so the code in the 3D world only needs to know about the callsign.ini file.

Note 2

The second is that they are totally 'local' to each pilot's computer, meaning that while each pilot may have the TEmissionname.ini in their `\Data\Campaign` directory and see the same TGT STPTs/LINES/PPTs, changing any one of these will *only* affect the pilot that made the change. The changes will not propagate to the other pilots inside the context of the game, as each have this file on their local install.

Note 3

The third note relates to the loading of the DTC. The callsign.ini is loaded into memory automatically upon program start up (launching Falcon from the desktop) and also if a new logbook is selected/created. In other words, after a pilot creates LINES, TGT STPTs, and PPTs for a mission, hits the SAVE button in the DTC and then exits the sim (completely), it is not necessary to open the DTC and hit the LOAD button when he re-launches Falcon and wishes to fly using the elements he created. He should see them in the cockpit automatically.

Configuration file variables

Finally, a couple of variables to control behavior of DTC info load have been added.

[Set `g_bLoadDTCForTrns`](#) to 1 for the DTC to be loaded in training missions as well as other missions' types; off by default which is the original behavior – no player ini file load for trn missions.

[Set `g_bNoDTCForRampStart`](#) to 1 to suppress loading of the DTC ini file content when the player selects RAMP starts. They will have to load this up from the MFD interface (or key command if they are being lazy) manually as part of the jet start up sequence as in real-life.

This option can now also be changed in Falcon BMS Config > Campaigns.





12.4 Structure of the IFF file

Before documenting the IFF file structure it's important to recap all the IFF settings:

12.4.1 IFF MODES AND CODES FOR A GIVEN MISSION:

BMS will assign modes and codes for each aircraft of each flight. AI aircraft will follow the assigned modes & codes. For the player, the DTC will automatically be updated with the mission's modes and codes.

The assigned codes will depend on the IFF 'policy' of each team for each given mission type. This 'policy' is described for each TE and Campaign in a dedicated file, named '[MissionName].iff'. The format of this file is described in a further section.

The briefing page displays all relevant information for your assigned codes. It will also display policy options for your team.

12.4.2 MODE 4 KEYS

Mode 4 keys can rotate from 1 to 4 times per day, as given in the ".iff" file. As a result, the ground crew will load the A key as the key valid at ramp, and the B key as the key valid for the next period. In the DTC, it will automatically create TIME events switching the mode 4 key at the proper time.

For example, if the key is set to switch once a day, your M4 key will be toggled from A to B at midnight. The TIME event will be created accordingly.

Mode 4 keys are common to the whole team and are shared between allies. If 2 teams in a TE/campaign are allies, they are supposed to share mode 4 keys and reply correctly between each other.

Note: even if IRL, some nations do not have access to the NATO mode 4 functionality, Mode 4 operation in the sim can simulate other secure identification modes.

12.4.3 MODE 1 CODES

Depending on the settings in the ".iff" file, the mode 1 code can be:

- Common to the whole package
- Common to the whole team
- Common to the squadron (peacetime use)
- Assigned per mission type. With this setting, as an example, all SEAD flights will have the same M1 code.

The actual codes are attributed in a pseudo-random fashion. This attribution is repeatable (as in : if you restart a TE/Camp, you will get the same codes), but difficult to know in advance. It is possible, however, that 2 squadrons/2 packages/2 mission types get assigned the same M1 code, especially considering how few M1 available codes there are.

Mode 1 codes can also rotate periodically, up to 48 times per day or once every half hour. Again, time events will be created automatically corresponding to this rotation.

The assigned M1 codes can never be 00 or 70 (emergency code).





12.4.4 MODE 2 CODES

Mode 2 has more accessible codes than mode 1. As a result, code attribution has the following logic:

- Assigned per aircraft on the package (see below).
- Common to the whole team (similar M1)
- Common to the squadron (peacetime use, similar M1)
- Assigned per mission type (similar M1)

The setting 'per aircraft in the package' is a bit tricky to explain. Essentially, a package will get assigned a block of 20 consecutive M2 codes. Let us consider an M2 code as XXYY. XX will be common to the whole package. YY will depend on package and aircraft position in the package. There are 3 different blocks: XX04 to XX27, XX30 to XX53, and XX54 to XX77.

Let us imagine that our package gets assigned the block 4404 to 4427.

The leader of the first flight will get assigned 4404. If he has wingmen, his n°2 will get assigned 4405, his n°3 gets 4406, and his n°4 gets 4407.

The leader of the second flight will get assigned 4410. Etc.

This continues up to the 4th aircraft of the fifth flight which gets assigned 4427.

This scheme allows having different settings for each aircraft, to identify them individually. It also allows looking for a particular aircraft by setting a particular M2 interrogation code in your package. This way, it acts a bit as the TACAN A/A channel attribution for AIs.

If you want to check the individual code for an aircraft in another package (for example, a tanker), it also is possible if you select the tanker's role and go in the briefing page.

No matter how the codes are attributed, you will never see an attributed code ending in 00 or starting by 77. If you see a code ending in 00, that likely means the aircraft is in BACKUP mode (see the pilot's guide for that), or EMERGENCY if it is 7700.

Similarly to M1, it is possible that 2 squadrons/package/mission type get the same codes. It is not as likely as M1 since there are many more codes available in M2, but it is still possible.

M2 codes cannot rotate. The fact that M2 codes are not affected by TIME events in the F-16 may or may not have affected this decision.

12.4.5 MODE 3 CODES

M3 codes follow the same code attribution logic as M2's:

- Assigned per aircraft on the package. (similar M2)
- Common to the whole team (similar M1)
- Common to the squadron (peacetime use, similar M1)
- Assigned per mission type (similar M1)

M2 and M3 codes are independent. Even with the exact same settings, they will be attributed differently.

M3 codes can also rotate up to 48 times per day, at the same times as M1 codes. If assigned per aircraft in the package, the attribution scheme per block discussed above is preserved at each rotation. Again, TIME events will be created to rotate M3 codes automatically.





12.4.6 MODE ACTIVATION

Mode activation is decided in the '.iff' policy file.

Mode activation is dependent upon 3 things:

- Which team we are on, as for code attribution
- Which stage of the mission we are in.
- Which mission type it is.

First of all, the mission is separated into 3 stages:

- A 'base' stage, from takeoff to a 'switch-off' point
- A 'strike' stage, from 'switch-off' to 'switch-on' point
- And a 'return' stage, from 'switch-on' point to landing.

Thus, it is possible to set mode activation differently between 'base', 'return' and 'strike'.

Typically, for an OCA Strike, the 'base' & 'return' will have more modes enabled to allow for organizing the flights, identification by AWACS, etc.

And the 'strike' phase will have only mode 4 enabled to limit RF emissions.

The 'switch-off'/'switch-on' lines are defined as follows:

- If there is a 'push' waypoint, it acts as the 'switch-off' point
- If there is a 'split' waypoint, it acts as the 'switch-on' point
- If there are no 'push' (resp. 'split') waypoints it will choose a waypoint between takeoff and target waypoint (resp. target and landing waypoints) to define a 'pseudo'.
- If, despite that, there still are not any 'switch-off' or 'switch-on' points, the 'strike' modes are not used and the flight will only use 'base/return' modes.

The 'switch-off' and 'switch-on' points are then defined in the DTC as POS events. To determine direction, again, the code checks the flight plan to determine in which rough cardinal direction the jet is going to the target or back to base.

The modes used by every flight in the package, as well as your POS events, can be checked beforehand in the briefing page.

Each mission type can have separate mode activation logics. For example, it is possible to set every Stealth STRIKE to never enable any mode. HAVCAP or BARCAP, which stay over friendly territory, can have the same modes for 'base'/'return' and 'strike' phase –typically, M1, M2 and M4. OCA Strike, SCAR, etc. which do go over enemy territory, can have M1, M2, and M4 over friendly territory and M4 only over enemy territory. All of this can be configured in the ".iff" file.

12.4.7 AI USE OF IFF

The AIs use the exact same code and mode attribution logic as the player. They also change settings with TIME and POS events.

AI can be interrogated and will reply correctly even if the AI flight is still aggregated.

As of now, AWACS and ATC do not use IFF, not for initial check-in, ATC pattern entry, or identification.

AIs periodically interrogate on Mode 4 when doing a BVR intercept, on the target LOS.

Friendlies will not shoot at you if your IFF is not set correctly.





12.4.8 IFF POLICY FILE “.IFF” FORMAT

A “.iff” file is created or updated every time the mission is saved when in Single Player. In MultiPlayer, the file is only saved by the Host. This is meant to prevent knowing the enemy team’s policy in a PvP environment. If no ‘.iff’ file is defined, a hardcoded default is used by the code and will be saved when you save the mission. The default is different if it is a campaign or a TE. The file is defined in 8 sections for each of the 8 teams.

Please note, TEAM 0 is not used. In a TE with 2 teams (US & NK), the IFF file will have 3 team section: 0, 1 & 2. Section TEAM 0 can be left default, TEAM 1 will be US and TEAM2 will be NK.

A team section looks like the following:

```
[IFF Policy Team 0]
M1Setting=1
M1Mask=0
M2Setting=0
M2Mask=0
M3Setting=0
M3Mask=0
M1M3Rotation=24
M4Rotation=1
ProfileMission0=6363
ProfileMission1=1111
...
ProfileMission41=1111
```

M1Setting can take 4 values:

- 0: per package
- 1: per team
- 2: per squadron
- 3: per mission type.

M2Setting and M3Setting can take 4 values:

- 0: per aircraft in the package
- 1: per team
- 2: per squadron
- 3: per mission type.

M1Mask, M2Mask, and M3Mask are useful if you want to tweak a bit the code generation process. This way, the codes generated for a particular mode will be different if you use a different mask.

Note: that even if 2 teams have the same Mask value for a mode, codes generated will be different.



M1M3Rotation defines how many times per day the M1 and M3 codes rotate. It can vary between 0 (never rotates) and 48 (every half hour).

M4Rotation defines how many times per day the M4 key rotates. It can vary between 1 (once a day at midnight) and 4 (every 6 hours).

ProfileMissionX = YZZZ defines the modes applied to each mission type.

The X represents the mission type, assigned as follows:

ProfileMission0: No role assigned
ProfileMission1: BARCAP1
ProfileMission2: BARCAP2
ProfileMission3: HAVCAP
ProfileMission4: TARCAP
ProfileMission5: RESCORT
ProfileMission6: AMBUSHCAP
ProfileMission7: SWEEP
ProfileMission8: ALERT
ProfileMission9: INTERCEPT
ProfileMission10: ESCORT
ProfileMission11: DEAD
ProfileMission12: SEAD Escort
ProfileMission13: OCA Strike
ProfileMission14: Interdiction Strike
ProfileMission15: Strike
ProfileMission16: Deep Strike
ProfileMission17: Stealth Strike
ProfileMission18: STRAT BOMB
ProfileMission19: FAC
ProfileMission20: On-call CAS
ProfileMission21: Preplanned CAS
ProfileMission22: CAS
ProfileMission23: SCAR
ProfileMission24: AI
ProfileMission25: BAI
ProfileMission26: AWACS
ProfileMission27: JSTAR/ELINT
ProfileMission28: TANKER
ProfileMission29: RECON
ProfileMission30: BDA
ProfileMission31: ECM
ProfileMission32: AIRCAV
ProfileMission33: AIRLIFT
ProfileMission34: SAR
ProfileMission35: ASW
ProfileMission36: TASMO
ProfileMission37: PATROL
ProfileMission38: RECONPATROL
ProfileMission39: ABORT
ProfileMission40: TRAINING
ProfileMission41: OTHER



The mode actually applied depends on the = YYZZ part.

- YY describes the modes active over friendly territory.
- ZZ describes the modes active after the push.

If YY = ZZ, then the mode actives are the same over the whole mission, there is no 'switch off/on' points and no POS events are generated.

YY and ZZ are 2 digits encoding which modes are active. They follow the same convention as the IFF capabilities set in the aircraft.dat files. Each number is between 00 and 63, leading zeros included.

- 01 is for Mode 1 only
- 02 is for Mode 2 only
- 04 is for Mode 3 only
- 08 is for Mode 4 only
- 16 is for Mode C only
- 32 is for Mode S only (implemented only as eye-candy).

To enable multiple modes, add the codes together.

As an example: for SWEEP missions, I want to use M1, M2, and M4 over friendly territory, and only M4 after the push.

The 'ProfileMission' line to use is ProfileMission7.

For M1 + M2 + M4, we need to encode $01 + 02 + 08 = 11$.

For M4 only, we need to encode 08.

The final line looks like this:

ProfileMission7 = 1108





12.5 Structure of the ATC file

All the airbase ATC files are located in your \Data\TerrData\ATC folder
They are simple text files with a .dat format and can be edited with any application like notepad.
They are named according to the airbase they represent. Example: Taegu.dat

If an airbase does not have this file the ATC will not be able to provide vectors for instrument approach and will revert to the other possibilities (most of the time the visual procedure.)

There is another file in the ATC folder named airbase.lst that must match the list of airbase file implemented. If you add a new ATC file for a specific airbase you must list this new airbase in the airbase.lst

This chapter will explain the structure of the airbase ATC files

The file is made in different subsections. One for airbase general setting and then one for each runway at that airbase. Please note a 02/20 runway is actually 2 different sections: One for RWY 02 and another for runway 20

means remark and the line after # is not taken into account by the code. Only lines not starting with a # sign are relevant to the code.

12.5.1 Airbase general setting

```
# ATC FILE
# Modified Oct 17th 2018 by Red Dog (lead turn rwy 32)
# Campid # This is Taegu airport
#
778
#
# Nbr Runway
#
2
#
1 # forcecloserVFR: "0" normal pattern, "1" short pattern, base mandatory
#
500 #MIN ILS CLOSURE VISIBILITY (m)
#
500 #MIN ILS CLOSURE CLOUD MSL ALTITUDE (ft)
#
1500 #MIN VFR VISIBILITY (m)
#
1400 #MIN VFR CLOUD MSL ELEVATION (ft)
#
```





Which without remarks corresponds to:

778

2

1

500

500

1500

1400

Much shorter but quite less explanatory.

The first number (778) corresponds to the CAMPID which is listed in the stations+ILS.dat file (\Data\Campaign\ folder)

The second number (2) lists how many runways the airbase features (remember 1 runway is two landing direction. In this case 2 since it is a single runway (airbase with 2 runways will have 4 obviously)

The third number (1) is either 0 or 1.

0 is the standard system with all types of approaches allowed (VFR = vectors for visual approach and IFR = vectors for instrument approach).

If set to 1, IFR approaches remain fully available but VFR approaches are more restricted. Straight in is prohibited and the vectors given by ATC will always vector the returning aircraft to a tight BASE pattern. This ensures that airbases close to mountains are accessible to AI)

The fourth number (500) is a visibility value expressed in metre.

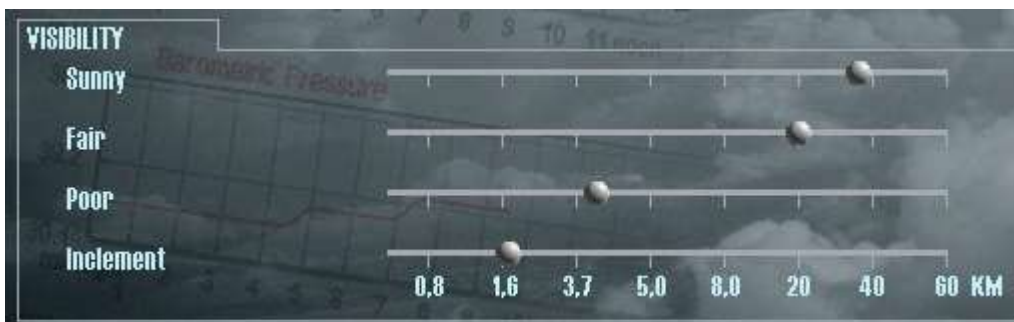
If the visibility in BMS is set to anything below that value the airbase will be closed because it is under the published weather minima for the instrument procedure.

This value corresponds to the minimum visibility for each instrument approach stated in the MINIMA section of the approach charts as illustrated in red on the picture to the right. The reason the ATC value is a bit lower than the chart value is to prevent the airport to close too soon and let a bit of dynamic weather to let pilots do the approach and eventually go around if the minima are not met.

This visibility in TEs is set either in the Fmaps or in deterministic weather cloud tab in the UI weather section.

By matching all these values when creating the weather for your TE, you can easily put the weather right on minima or overdo it and close the airport.

MINIMA:	
ILS: 553'(200'AGL)	Vis: 200ft 800m*
LOC: 773'(420' AGL)**	Vis: 500ft-1600m**
Circling: 1653'(1300'AGL)	Vis: 1300ft-2000m
*When ALS inop, increase vis to 1200m	
**When ALS inop, increase vis to 2000m	





The fifth number (500) is an altitude expressed in feet for cloud base. Under that value the airbase will be closed because the weather is below the published minima. Please note, altitude means MSL therefore the airport elevation must be taken into account. Taegu elevation is 353 feet and the ILS minima are 200 feet. $353+200=553$ feet, rounded below to 500 avoid the airbase to be closed spot on minima and may force the pilot to attempt the approach and having to go around. This value corresponds to the minimum altitude the pilot must have the runway lights in sight to continue the approach. It is given in the MINIMA section of the approach charts.

MINIMA:	
ILS: 553'(200'AGL)	Vis: 200ft-800m*
LOC: 773'(420'AGL)**	Vis: 500ft-1600m**
Circling: 1653'(1300'AGL)	Vis: 1300ft-2000m
*When ALS inop, increase vis to 1200m	
**When ALS inop, increase vis to 2000m	

The 6th number (1500) is the minimum VFR visibility. It is a visibility distance expressed in meter. If the visibility is set below that distance then VFR approaches will not be allowed and ATC will force aircraft to land IMC. It corresponds to the 1500m minimum visibility of Special VFR. Above this number, VMC is allowed.

The 7th number (1400) is the minimum cloud altitude for VFR approach. Altitudes are given in MSL and airport elevation must be taken into account. Since Taegu elevation is 353 feet, any cloud base below 1047 feet AGL will prevent VFR approaches and make the radar approach mandatory. This number is usually 1000ft AGL and correspond to special VFR ceiling.

12.5.2 The runway information section:

Each section has the same structure. As said before, the number of sections will depend on the number of active runways at that specific airbase. See the second number in the general information section. If set to 2 you will have two runway sections but one runway ID (usually 0), if set to 4 you will have 4 runway sections, hence 2 runway ID (0 and 1)

All runway section structures are the same and we will only illustrate one corresponding to RWY 32 at Taegu

```
#####  
# RUNWAY "0" - 320  
#####  
#  
#  
0 #DB runway Nbr  
#  
320 # Heading  
#  
1 #Has base approach  
0 #Has Long approach  
#  
# Overhead -1 is Left, 1 is right  
1  
#  
# FINAL  
6 0 2350 1  
#####  
# BASE  
# SAY BASE 0 "no", 1 "Right" , 2 "Left"  
0  
# BASE PT
```





```
# Lead turn by 2Nm (9 0)
9.8 -1.7 3350 1
# ENTRY PT
# Lead turn by 1Nm (10.4 -3.3)
10.3 -4.3 4000 1
#
# HOLD PT
9 -10.72 6000 1
#
# LOITER -1 is Left, 1 is right
-1
#
#####
# LONG
#
# ENTRY
0 0 0 0
#
# HOLDING
0 0 0 0
#
# LOITER -1 is Left, 1 is right
-1
#
```

The short version is

```
0
320
1
0
1
6 0 2350 1
0
9.8 -1.7 3350 1
10.3 -4.3 4000 1
9 -10.72 6000 1
-1
0 0 0 0
0 0 0 0
-1
```

Falcon Editor --- Korea KTO

File Options

Class Table Objective Data Units Sensors Vehides Weapons

Objectives Features

Index	Name	CT Index
367	Airbase 36_18	1684
48	Airbase 34_16 Dual	879
366	Airbase 34_16	1683
365	Airbase 32_14	1682
364	Airbase 30_12	1681

Point Header Data

Type: RunwayListType

Heading: 320 Runway Texture: 30

Runway Number: 0 Landing Pattern: Center

Feature Dependencies: 8 Runway Threshold 2 Runway Section

0 Runway Long Section 1 Runway Section None

OK Cancel

0 is the runway ID in the database. Single runway airbases only have ID0, but double runway airbase have 2 runways: ID0 and runway ID1. One runway has two direction (in this case ID0=140° and ID0=320°)

320 is the runway magnetic heading. So Runway 0 is assigned to RWY 32 (Rwy 14 is also Runway 0 but declared with its heading: 140°in the section before this one)

The runway ID and the magnetic heading MUST correspond to the database which is verifiable with the BMS editor in the objective Data list (of course you must know that Taegu is 32_14 Airbase (CT Index 1682)

The next two lines are either 0 or 1. They corresponds to either a 4 points system (has base approach) or a 3 points system (Has long approach)

If you want to use a 4 points system, the lines must be 1 then 0

If you want to switch to a 3 points system, the lines must be 0 then 1





The 4th setting is the direction of the overhead break. It can be set to either -1 or 1.

-1 is left and 1 is right. Usually the direction of the break is on the opposite side of the tower, but may vary according to other parameters (obstructions, noise abatement, ...) This allows to set the AI direction of the break for the overhead recovery. In this case, the break is to the right.

The next set of number correspond to the position of the FINAL point. As you can see you have 4 such lines. They correspond to the 4 points: Final, Base, Entry and Hold.

The FINAL point is the one closer to the approach end of the runway. The BASE and ENTRY are two middle points and the HOLD point is usually the IAF.

The structure is the following

X in Nm, Y in Nm, Altitude of the point in mean sea level, 1 is a setting to allow the ATC to precisely detect the aircraft. It should always stay 1.

The X,Y coordinates system originates at the runway threshold. The X axis extends from the runway threshold in the direction opposite to the runway heading the pilot would fly upon landing.

The Y axis has its zero at the runway threshold and is perpendicular to the X axis. The Y axis extends always on the right of the X axis

Points aligned with the runway centerline always have a Y value of zero.

So the final point **6 0 2350 1** is placed 6 Nm from the runway threshold, on the runway centerline at an altitude of 2350 MSL. And the system will detect the aircraft at that point.

The next setting (1) defines if the base point has to be called. Possible choices are 0, 1 & 2.

0 does not call base, 1 calls Right and 2 calls left. For most of IFR approaches, this line should stay at zero.





The next line is the position of the BASE point with the same structure explained before.

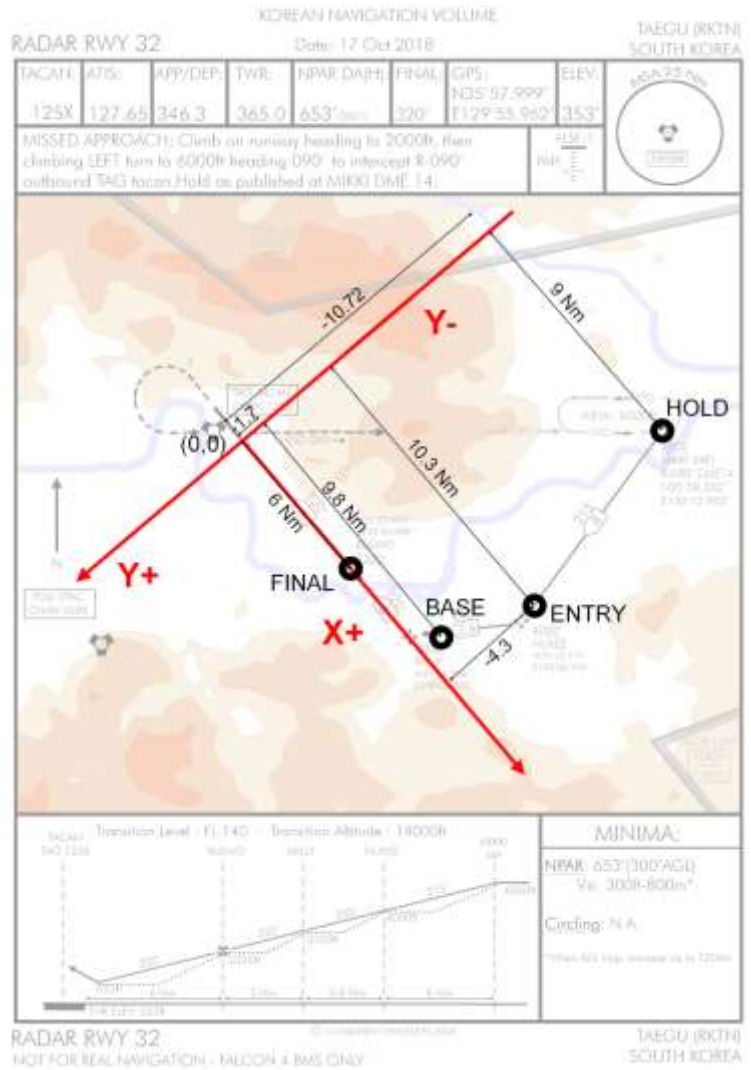
9.8 -1.7 3350 1 places the base point 9.8 Nm on the X axis, -1.7Nm on the Y axis at 3350 feet MSL.

The Chart base point would be at 9 0 3350 1 but the ATC files is placed a bit off from that position to allow the aircraft to lead it's turn and join final smoothly.

The next line is the position of the ENTRY point **10.3 -4.3 4000 1** places the entry point 10.3 Nm on the X axis, -4.3Nm on the Y axis at 4000 feet MSL.

Normally the point would be at 10.4 -3.3. the offset is to allow room for the turn.

The next line is the position of the HOLD point **9 -10.72 6000 1** places the Hold point at the IAF 9 Nm on the X axis, -10.72Nm on the Y axis at 6000 feet MSL which is the published MHA on the chart. There is no need to add any room for turns at that point since aircraft can reach this point from every direction.



As you can see on the chart the position of the ATC 4 points do not exactly match the chart radar fixes. This is not mandatory but helps aircraft to stay on the published track by leading their turn. It seriously complicates the edition of the ATC file but provide smoother approaches for both humans and AI alike.

The next setting can be set to either -1 or 1 and correspond to the direction of the turn for the holding at the IAF (HOLD point). -1 is LEFT and 1 is RIGHT. As seen on the chart the holding turns are LEFT and therefore the setting reads -1

The last three lines are respectively the coordinates of the ENTRY and HOLDING point for the 3 points approach system (Long final) and the direction of the hold turns at the Holding point.

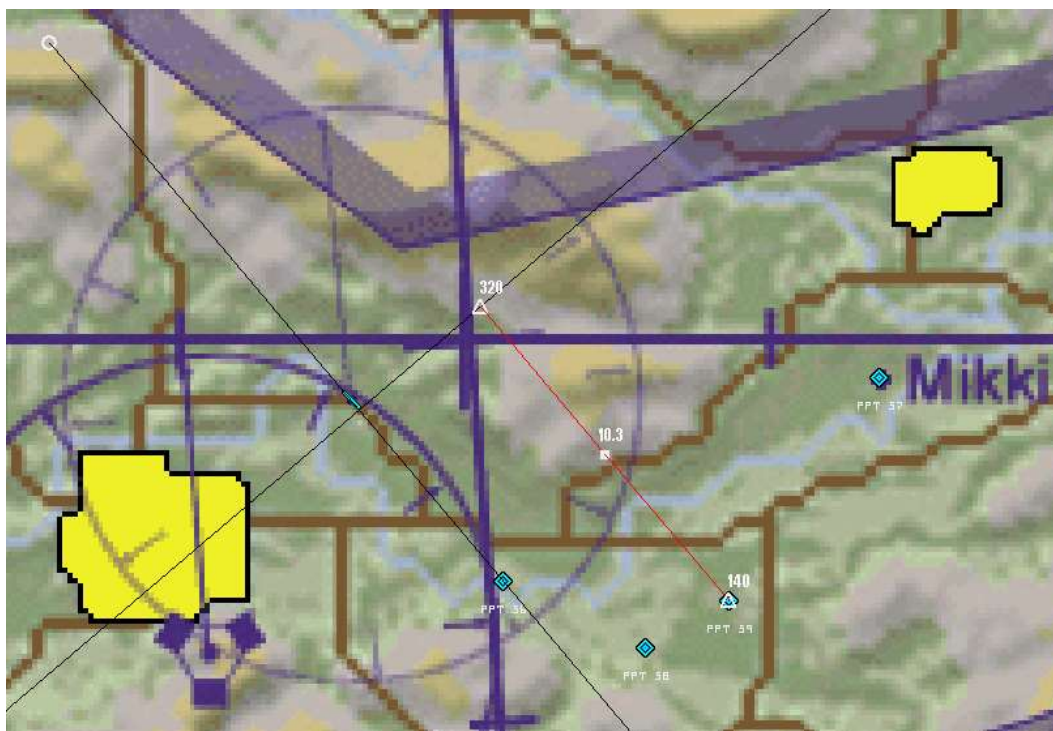
When using the 3 points approach system, the BASE point is not used. This system was created when the approach can be made simpler with only 3 points.

Most of the ATC files use the 4 points system and therefore these are most of the time left to zeroes. But some patterns in straight lines like the ones created for North Korea use the 3 points system.

Elaborating the ATC files cannot really be made on the chart as distance are not very precise. It is better to calculate the 4 points positions in the UI map using lines, PPTs and the ruler.

Here is one way to do it

- Select the ruler and draw a 30Nm line on the runway axis (140-320). Using the ruler as a reference, draw a STPT line on top of the ruler. That is your X axis.
- Repeat the operation for the Y axis by aligning the rule on runway axis plus or minus 90° (050-230) and place another STPT lines for Y. Make the two lines cross at the icon runway threshold (Yes it is not very precise either).
- Place the ruler at the 0.0 point and extend 6 Nm on X axis in the positive direction and place a PPT at the end of the ruler. That is your FINAL point
- The IAF is equally easy to find, it is usually marked as the IAF fix on the chart, in this case Mikki. Place a PPT there as well and measure the distance from X and Y with the ruler (140/320 direction for X and 050/230 direction for Y)
- Place the BASE and ENTRY points as PPTs and measure distance with the ruler as explained above.





12.6 Structure of the STATIONS+ILS.DAT file

The stations+ILS.dat is loaded when you commit to a mission and sets all the navigation data and airbase comms frequencies.

It is relevant to all airbases, single radio navigation stations, airstrips and carriers of the theatre.

The file structure has been modified for 4.35 to better define the frag for each ATC agency to ease up callsign assignment.

The structure of the file is as follows:

```
<campaign ID> <tacan ch#> <tacan band> <callsign> <range> <tacan_type> <tower_freq_UHF>  
<tower_freq_VHF> <runway1_ils_freq> <runway2_ils_freq> <runway3_ils_freq> <runway4_ils_freq>  
<OPS_freq_UHF> <GROUND_freq_UHF> <APPROACH_freq_UHF> <LSO_freq_UHF> <ATIS_freq_VHF>  
<TOWER_FRAG> <GROUND FRAG> <APPROACH FRAG> <ATIS STATION FRAG>
```

- <campaign ID>: ID of the airbase the following information are for.
- <tacan ch#>: Integer for the tacan chanel (1-126)
- <tacan band>: either X or Y. In BMS X band is more for ground station while Y is more for airborne stations. Although both should work in either role.
- <callsign>: Approach index, set the ATC name of the airbase agency.
- <range>: The Tacan range in Nm. Please note, signal is line of sight. Range might be shorter if obstructed by terrain. As a rule of thumb, Terminal stations have a range of 50-25Nm, En-route stations have a range of 100-200NM. Maximum optical range is around 200Nm.
- <tacan_type>: Always 1.
- <tower_freq_UHF>: Integer that represents a valid UHF radio frequency for tower (kHz*1000)
- <tower_freq_VHF>: Integer that represents a valid VHF radio frequency for tower (kHz*1000)
- <runway1_ils_freq>: Integer that represents a valid ILS frequency for Runway 1 (kHz*100)
- <runway2_ils_freq>: Integer that represents a valid ILS frequency for Runway 2 (kHz*100)
- <runway3_ils_freq>: Integer that represents a valid ILS frequency for Runway 3 (kHz*100)
- <runway4_ils_freq>: Integer that represents a valid ILS frequency for Runway 4 (kHz*100)
Note: the sequence of the runway ILS depends on the Campaign ID. For instance, there are two models of generic 05/23 runway airbase: ID 978 & 1779. The ILS sequence can be found in the table on the next page..
- <OPS_freq_UHF>: Integer that represents a valid UHF radio frequency for operations (kHz*1000)
- <GROUND_freq_UHF>: Integer that represents a valid UHF radio frequency for ground (kHz*1000)
- <APPROACH_freq_UHF>: Integer that represents a valid UHF radio frequency for approach (kHz*1000)
- <LSO_freq_UHF>: Integer that represents a valid UHF radio frequency for landing signal officer z (kHz*1000). The LSO frequency is actually never used for land airbase and are not used either for carrier as the LSO talks to the tower frequency.
- <ATIS_freq_VHF>: Integer that represents a valid VHF radio frequency for tower (kHz*1000)
- <TOWER_FRAG>: Defines the Frag ID for Tower. (previously read from the eval.xml file)
- <GROUND FRAG> : Defines the Frag ID for Ground. (previously read from the eval.xml file)
- <APPROACH FRAG>: Defines the sound Frag for Approach. (previously read from the eval.xml file)
- <ATIS STATION FRAG>: Defines the the TTS Eval.dat frag number for the ATIS.





12.7 Structure of the RADIOMAP.DAT file

The Radiomap.dat is a new file for BMS 4.34 located in the data\campaign folder.

The assignments of frequencies to flights is now based on callsigns.

The Radiomap.dat which is a file in text format links VHF and UHF frequencies to every possible callsign the game may use.

This operates similarly to the stations+ils.dat file in that it is loaded when you commit to a mission so it can be theater specific if needed.

There are enough UHF frequencies that each callsign gets a unique frequency in the UHF band.

The VHF band has fewer possible frequencies so it is possible for more than one flight to get the same frequency on VHF in some cases.

An attempt has been made to reduce the possibility of this as far as possible by careful assignments in the file. Still, the file is text format and you can edit it if you prefer to see different assignments.

There are a few non-callsign agencies that the code recognizes (GUARD, UNICOM, ADVISORY, COMMON TWR, COMMON APPROACH, ...) which also have lines in the file under the GENERIC section.

The format is the following:

<agency_name>, <VHF_frequency>, <UHF_frequency>

- *<agency_name>*: a string that identifies the flight callsign or equivalent.
- *<VHF_frequency>* Integer that represents a valid VHF radio frequency (kHz*1000)
- *<UHF_frequency>* Integer that represents a valid UHF radio frequency (kHz*1000)

"NONE" is used to indicate no frequency assigned to an agency in that band.

Note: frequencies are range checked so make sure to use legitimate values for the frequency band in question.

Radio range consideration for STATIONS+ILS.DAT and RADIOMAP.dat file:

The full spectrum of the VHF range is from 108 to 151.975 Mhz

(BMS still uses .25 steps whereas real life is switching to 8.33)

- VHF range for navigation stations is from 108.0 to 117.975 Mhz, this range is reserved for stations+ILS.dat file. (108.0 – 111.975 for ILS and 112.0 to 117.975 for radionavigation stations)
- VHF Range for ATC is from 118.0 to 135.975 Mhz, this range is also reserved for stations+ILS.dat file.
- As you can see that leaves only a small range from 136.0 to 151.975 for intraflight frequencies according to callsigns which can be assigned in the radiomap VHF range.

UHF range for ATC is from 225.0 to 399.975 of which a few frequencies in no specific range are assigned to ATC frequencies. That leaves much more UHF frequencies (than VHF) available for interflight in the radiomap.

Last consideration when you plan to change the default assigned frequencies:

Always try to use unique frequency. Duplicating frequency may lead to all sorts of unexpected problems.





13. List of the Config options

The first part of the Falcon BMS.cfg are the options the user can easily set with the Falcon BMS Config app launched from the **Configuration** option of the Launcher. These options are commented in the app.

13.1 Avionics

g_fCursorSpeed

Set Cursor speed from 50 to 200%. Default is 100%.
Available options: 0.5 – 0.75 – 1.0 – 1.5 – 2.0

g_bMFDHighContrast

Disable or Enable High Contrast MFDs to help colour recognition. Default is 0 (Disable). Possible setting 0/1.

g_fRadarScale

Change the size of the radar blips. Default is 75% (0.75).
Available options: 0.25 – 0.5 – 0.75 – 1.0 – 1.5

g_fHSDSymbolSize

Sets size of the HSD symbols. Default is 0.025 which might be too small for lower resolutions, in which case set to 0.05.

13.2 Campaigns (the options below also influence Tactical Engagement)

g_bBarcapNoBarcap

If set to 1/true, AI BARCAP flights will not react against other BARCAP flights anymore. Default 0/false.

g_bNoDTCForRampStart

With this option enabled the DTC will not be automatically loaded during Ramp start; the user will have to load it. Automatic DTC load always happens at Taxi & Takeoff start. Possible options: 0/1. Default: 1 (not auto loaded at Ramp).

13.3 General

g_nPrintToFile

Enables output of detailed briefing data to "briefing.txt" file located in `\User\Briefings`. That file may then be externally printed. Available options: 0/1

g_bAppendToBriefingFile

When enabled this option adds briefing data to the same Briefing.txt. When disabled it overwrites the briefing data. so Briefing.txt only contains the last brief. Available options: 0/1

g_bBriefHTML

Enable HTML format for briefing. The HTML files are stored in `\User\Briefings`. One file per briefing. Naming is done with `date_time_briefing.html`. Please note when this option is enabled the briefing.txt files are not generated anymore.

g_bLogEvents


Enables output of detailed debriefing data to "debrief.txt" file located in `\User\Briefings` file. Newest debrief are always added at the end of the file. Available options: 0/1





<i>g_bEnableDisplacementCam</i>	Enable or Disable Displacement Camera that simulates camera instability in the Orbit view. Available options: 0/1.
<i>g_b3DClickableCursorAnchored</i>	When enabled, the mouse cursor in cockpit is anchored on a hotspot while moving view. Available options: 0/1. <i>(See the view chapter for more information)</i>
<i>g_fDefaultFOV</i>	Sets Cockpit Field Of View Angle. Available options: 40° – 45° – 50° – 55° – 60° (default) – 65° – 70° – 75° – 80°
<i>g_bFullScreenNVG</i>	Enable or Disable full screen NVG. When disabled the FOV of the NVG will more realistically implement the tunnel vision effect. Available options: 0/1.
<i>g_bNoPadlockBoxes</i>	If set to 1/true, there will be no box at all around padlock objects. Default 0/false.
<i>g_bPngScreenshots</i>	If enabled the screenshot and prettyscreenshot callbacks generate .png format images. If disabled both callbacks generate .jpg format images. Available options: 0/1.
<i>g_nPadlockBoxSize</i>	Adjusts the size of the padlock box (if padlock is enabled). Available settings: 0/2. Default option: 2. <i>(See the view chapter for more information)</i>
<i>g_bPlayIntroMovie</i>	Plays intro movie if enabled. Go straight to UI if disabled. Available options: 0/1 – Default is Enabled (1).
<i>g_bRampTaxiStart</i>	If set to 1/true, TAXI spawns on RAMP instead of taxiway. Default 0/false.
<i>g_bCalibrationHg</i>	Determines unit of atmospheric pressure in general and cockpit altimeter in particular. Available options 0/1. When set to 0 units are SI: hPa (millibars). If set to 1, units are Imperial (inHg).

13.4 Hardware



<i>g_bHotasDgftSelfCancel</i>	For DX programming joysticks. When enabled the Dogfight/Missile Override callbacks (DGFT/SRM/MRM) will call "override cancel" automatically upon release. This can be used for joysticks like the TM HOTAS Cougar, where the override switch on the throttle is a 3-position switch, but only the 2 outwards positions actually create a DX button input. This option enables the centre position in case of DX programming. If set to disabled, the centre position will need to be programmed with callback. Available options: 0/1.
<i>g_bUseVirtualRudder</i>	This option is only useful for people who use racing wheel pedals and are NOT able to combine them into one axis via the wheel's drivers. By activating this option AND assigning the

pedals to Brake and Right Toebrakes axis in the UI, the sim will emulate a rudder axis by combining the inputs of these two devices. Available options: 0/1 - Default is disabled.

g_bHiResTextures

This option enables higher resolutions DDS for object skins and miscellaneous textures. When enabled, the content of both the `misctex_HiRes` and `KoreaObj_HiRes` will be used rather than the `misctex` and `KoreaObj` folders. If there is no HiRes texture available in these folders it will use the default texture instead. Available options: 0/1. **New default is now true (1)**

g_bUseAnalogIdleCutoff

This option when enabled suppresses the need for the idle detent callback. The detent is simulated by the course of the throttle axis. The idle point must be declared with a right click on the SET AB button in the SETUP > CONTROLLER page of the UI. Enable this option if you have a strong physical detent at the idle point of your throttle. Available options: 0/1 – default is 0.

g_bReducePSFires

This option when enabled reduces by 50% (average) the number of some main PS effects like ground units and features like smokestacks and explosions in order to reduce the FPS hit caused by the Particle System. Available options: 0/1.

13.5 Shaders

g_bEnvironmentMapping

Enable (1) or disable (0) Environment Mapping for glass.

g_bHdrLighting

Enable (1) or disable (0) High Dynamic Range (post-processing) lighting effects.

g_bHdrLightingStar

This option is relevant only when HDR lighting is enabled. When Enabled (1) this option renders bright lights with a 'star' effect, i.e. light rays originating from the light source.

Available options: 0/1.

Please note: this option is not in the applet.

g_bUseHeatHazeShader

Enable (1) or disable (0) Jet Heat Exhaust effects.

g_bShowFarRain

Enable (1) or disable (0) rendering of additional rain

g_bShowRainDrops

Enable (1) or disable (0) rain drops on camera (outside view) and canopy (internal view).

g_bShowRainRings

Enable (1) or disable (0) rain rings on the ground.

g_bShadowMapping

Enable (1) or disable (0) shadows for the cockpit and a small area around the viewer. Disabling this option will make the submenu options unavailable.



g_bShadowOnSmoke

Enable (1) or disable (0) shadows on particle system effects.
Please note: This option has a huge FPS impact.

g_bWaterEnvironmentMapping

Enable (1) or disable (0) environment mapping for water.
Disabling this option will make the submenu options unavailable.

g_bEnvMapRenderClouds

Enable (1) or disable (0) rendering of the clouds into the water environment map.

g_bEnvMapRenderFocusObject

Enable (1) or disable (0) the rendering of the focused object into the water environment map.

13.6 Track IR settings

g_bExternalTrackIR

When this option is enabled the TIR will slew the external view, replacing the mouse. Vector expansion will control zoom in external view as well. When this option is disabled the TIR is not active in external view and slewing is done with the mouse. Available options: 0/1.

g_bInvertExternalTrackIR

Relevant only when ExternalTrackIR is enabled. If enabled TIR azimuth slewing will be inverted. Available options: 0/1.

13.7 Multiplayer

g_bAllowMP_Freeze

Allows the MP host to specify whether "freeze" pause can be used (default = 0/disabled). Available options: 0/1.

g_bAllowMP_NVG

Allows the MP host to specify whether NVGs can be used by clients or not (default is 1/enabled = NVGs can be used by clients). Available options: 0/1.

g_bAllowMP_NVGFullscreen

Allows the MP host to specify whether NVGs can be used in full screen mode by clients or not (default is 1/enabled = client able to choose either option through his own config file). Available options: 0/1.

g_bRequireSameAcdataMP

4.33 Anti-cheat option: Allows the MP host to specify whether ACDATA and MISDATA files need to be the same in MP (default = 0 disabled = files can be different). This option was created mainly as a way for larger MP event servers to ensure tweaks to aircraft or weapons are not permitted.





Please note: changing settings with the Avionic Configurator will change the ACDATA content, preventing connection if this option is checked. If you enable this option you may have to ensure that a common (approved) set of avionic configuration files is provided.

<i>g_bRequireSameTileSetMP</i>	Allows the MP host to specify whether clients have to use the same tile (default is 1 = enabled). Available options: 0/1.
<i>g_bAllowMP_Smoke</i>	Allows the MP host to specify whether Smoke can be used (default = 1/enabled). Available options: 0/1.
<i>g_nMPStartRestricted</i>	Allows the MP host to specify which start-up options are allowed: 0=RAMP/TAXI/TAKEOFF (default), 1=RAMP/TAXI, 2=RAMP ONLY

13.8 Other Options

These options are not accessible from the app and must be edited directly in the Falcon BMS.cfg file.

<i>g_sTileSet "POLAK"</i>	This option defines which tile set to use. Tile sets are declared by by name according to the tile set's subfolder name. <i>(See theatre dev notes later on in the annex).</i>
<i>g_nTaxiLaunchTime</i>	Sets the Time in minutes before the take off time the player enters the cockpit for the TAXI start option (default is 4 minutes). This option is valid in SP only but affects only human flights. In MP the host setting overwrites the client's.
<i>g_nReagTimer</i>	Time in minutes that aircraft disappear at airbases after despawn. Default time is 2 (minutes).
<i>g_nDeagTimer</i>	Time in minutes for aircraft to appear at airbase before their taxi time. Default time is 2 (minutes).
<i>g_npercentage_available_aircraft</i>	Determines what proportion of your squadron's roster will be available (in %) for campaigns. This reduces the number of aircraft available but simulates maintenance or other planned mission requirements. Valid options 0-100. Default is 75%.
<i>g_nminimum_available_aircraft</i>	Allows you to limit the effect of a low 'percentage' (lower-limit of available aircraft). The ATO will frag missions until the number of aircraft available is under this limit. Valid option: integer. Default 4
<i>g_bEnableABRelocation</i>	Enable (1) or disable (0) airbase relocation. Squadron will relocate to other airbases if their own airbase is damaged to a certain level.



g_bServerHostAll

This option should always be set to 1 (Enabled). When enabled the MP host technically owns ALL units. When set to 0 the client requesting a unit to deaggregate will own it, making it responsible for distributing it over the network. Previous experience showed that this can minimise bandwidth demand on the host, but that it will create many more problems and sync issues. Options: 0/1.

g_nForceMinClientBwSetting

Allows the MP host to specify a minimum client bandwidth (default 0 = no value forced). Available options: 0 or forced bandwidth (e.g.: 500, 1000, etc).

g_bNoAiForHumanControlledSqd

Allows the MP host to specify whether AI pilots can be assigned to human controlled squadrons. This option is designed to prevent AI being assigned in squadrons flagged as human controlled during PvP or TvT type MP events. Default 0/false. Available options: 0/1.

g_nRemoteControlSurfacesInterval

Time in milliseconds between control surface updates over multiplayer. Valid range 20-1000. Default is 200, 0 = disabled.

g_bEnforceBandwidthLimits

If set to 1/true, the upload bandwidth setting from the UI will not be taken as "information for the network stack", but actually be "enforced", i.e. if you have a 50 MBit line but set your upload to 10MBit, BMS will only use 10MBit for upload, even if it could use more; should not be set unless you *need* to hard limit your upload rate for some reason. Default 0/false.

g_nF1TeamUiFreq

Set frequency for F1 UI voice comms (for Force on Force type missions). By default 339.750 is also UHF preset 14 in 3D allowing people in UI to communicate with ppl in 3D as long as they monitor the same frequency. Available options: valid UHF frequency range (6 digits, no decimal).

g_nF2TeamUiFreq

Frequency for F2 UI voice comms (for Force on Force type missions). Since default 1234 is not a valid UHF or VHF frequency it is not possible to communicate with people in 3D with F2. You can assign valid frequencies here to overcome that issue although it is intentional that F2 cannot be heard in 3D.

g_bhudAOA

Enable (1) or Disable (0) the HUD AOA symbols for non F-16 aircraft. Available options: 0/1.

g_bLocalEnvironmentalDate

If set to 0 (Disabled) the reference time for light calculations is day 135 of 2004. If set to 1 (Enabled) the reference time for light calculations is your system time. Available options: 0/1.

g_nHotasPinkyShiftMagnitude

Enable DirectX shifting and specifies button offset. DX button numbers can now be shifted *outside* the DX device limit, making



	it essentially possible to use <i>all</i> 16 DX devices with DX shifting. To do so, change offset to 512 (default is 256) in the config file (and adjust your keyfile accordingly of course).
<i>g_fFOVIncrement</i>	Defines how much the field of view should change for each keypress in degrees. Default value = 5.
<i>g_fMaximumFOV</i>	Limits the maximum angle that the FOV can be increased. Default value = 80°.
<i>g_bNoAAAEventRecords</i>	When this option is enabled (set to 1) it removes AAA shots from debrief. If disabled (set to 0) all AAA shots will be listed in debrief. Available options: 0/1.
<i>g_bACMIRecordMsgOff</i>	When enabled (1) this option turns off the ACMI RECORDING msg on top of the 3D view. Available options: 0/1.
<i>g_nPadlockBoxThickness</i>	Determines the thickness in pixel of the padlock box. See view chapter 9 for further information.
<i>g_nDynamicVoices</i>	Sets the maximum number of voices allocated by the sound code. Default = 32.
<i>g_nSoundUpdateMS</i>	Sets how many milliseconds must elapse before the sound code updates. Default 10.
<i>g_nSoundSwitchFix</i>	This option when set to 1 (enabled) may fix the problem of AI comms disappearing.
<i>g_bPilotEntertainment</i>	Enable (1) or Disable (0) the user to control WinAmp.
<i>g_nWinAmpInitVolume</i>	Sets the initial WimAmp volume. Default = 204.
<i>g_fAmbientmin</i>	This option allows tweaking the sky brightness at night. Valid range from 0.0 (all black) to 1.0(all white). Default setting: 0.1.
<i>g_bUseTerrainNightLightsTextureFilter</i>	This controls which texture filter to use for the terrain tile night lights (1=anisotropic filter - 0=point filter). It is again an option for servers with reduced GPU capabilities. Running BMS on a server may benefit from less VRAM usage with this option set to 0. Available options: 0/1. Default is 1.
<i>g_bUseTracerColors</i>	Enable (1) or Disable (0) green tracers for OPFOR (red) aircraft. Default = 1 (Enabled). Available options: 0/1.
<i>g_bEnableRandomFailures</i>	Enable (1) or Disable (0) Random failures. Available options: 0/1.
<i>g_fMeanTimeBetweenFailures</i>	Set the time in flight hours (logbook) between random failures. If set to 0 (default) random failures are disabled.



g_bF16MfdHasRwr	Enable (1) or Disable (0) the unrealistic RWR MFD page for F-16's. Available options: 0/1. Default is 0.
g_fMouseSensitivity 1.0	This option sets the 3D mouse sensitivity multiplier. Default value is 1.0. Any value under 1.0 will cut the sensitivity in half. The multiplier should be just above 1.0 to overcome the fact that the 3D cursor is a bit smaller than the mouse. A value of 2.0 will double the speed.
g_bMouseWheelKnobs	Enable (1) or Disable (0) the mouse wheel to turn cockpit knobs. Default is 1 (Enabled). Available options: 0/1
g_bEnableExclusiveMouseCapture	When enabled (1) this option captures the mouse when in 3D, so that you don't click something on the desktop when running in windowed mode. Default is 1 (Enabled). Available options: 0/1.
g_bMouseButton4TogglesClickablePit	If enabled and if a 4 th mouse button is available it will toggle between mouselook and clickable pit. FYI the mouse wheel is button 3. Available options: 0/1. Default = 0.
g_bUseIvcUiVolume	If enabled this option unties IVC volumes from the COMM1 (UHF) and COMM2 (VHF) volume sliders. UI IVC Volume is controlled independently. Options: 0 (disabled)/1(enabled). Default is 1.
g_nIvcUiVolume	Sets UI IVC Volume in db, if controlled independently (above option is 1). Valid range -6 to +6. Default is 5.
g_bPreventScreensaver	Prevent (1) activation of screensaver/powersave mode. Available options: 1=prevent, 0=do not prevent. Default = 1.
g_bPrettyScreenShot	If enabled (1) PrtScr key will make pretty screenshots (without text overlays) instead of normal screenshots with overlays. Available options: 1=pretty, 0=normal. Default = 0.
g_bExportRTTTextures	This enables the shared texture memory area for HUD/MFDs/HMCS/RWR/DED/PFL. This is independent from BMS external window usage. Available options: 0/1. Default = 1. Please note: this option must be enabled (1) for MFDE to work.
g_nRTTExportBatchSize	This option determines how often the shared texture memory area (if it is enabled with the above option) will be updated every Nth frame. Default is 2. This option can be changed by the MFDE config option. For best results set to 1.
g_nCampPeriodicSaveMinutes	Sets the time in minutes between automatic campaign saves. Valid range in minutes. 0 = disabled, which is default.

<i>g_sCampPeriodicSaveName "Auto"</i>	Defines the "prefix" for the automatic cyclic save name (when auto save is enabled). Campaign date and time will be added automatically to the saved name.
<i>g_nTacPeriodicSaveMinutes</i>	Sets the time in minutes between automatic TE saves. Valid range in minutes. 0 = disabled, which is default. Works similar to the existing campaign autosave, however the save name prefix is fixed to the original TE title
<i>g_nActionCameraTimer</i>	Sets the duration in milliseconds for the action camera before switching views. Default 8000, i.e. 8 seconds.
<i>g_nJetHeatShaderMaxSpeed</i>	Define a speed in knots above which the jet blur effect disappears. Valid range in Kts. Default 250 knots.
<i>g_nMessageScrollTime</i>	Sets the time in seconds until a chat message scrolls out of view. Default = 15.
<i>g_bPlayDogfightBits</i>	Enable (1) or Disable (0) playback of the various sound comments in the dogfight scores menu. Default = 1.
<i>g_fViewlimitPitchDown</i>	Is used to used to override ALL "viewlimit_pitch_down" values in the 3dckpit.dat files (only value increases are possible, lower values will be ignored). Maximum value is 90.0. This is helpful for multiscreen and/or TrackIR users.
<p>Note: you set it in the config ONCE if you want a non-default value for EVERY aircraft. If you really want specific values for specific aircraft simply edit the relevant dat files. Default =0.0</p>	
<i>g_sLogsDirectory ""</i>	If filled this option redirects the logfile output from <code>\User\Logs</code> to the specified directory in <code>" "</code> . Default value is empty <code>""</code> hence the default location for the log files.
<i>g_sAcmiDirectory ""</i>	If filled this option redirects the ACMI directory from <code>\User\Acmi</code> to the specified directory. Default value is empty <code>""</code> hence the default location for the acmi files.
<i>g_sPicturesDirectory</i>	If set, redirects the pictures directory from <code><BMS>\User\Pictures</code> to the specified directory. Default <code>""</code> .
<i>g_sBriefingsDirectory</i>	If set, redirects the briefing output from <code><BMS>\User\Briefings</code> to the specified directory. Default <code>""</code> .
<i>g_sPatchesDirectory</i>	If set, redirects the patches directory from <code><BMS>\User\Patches</code> to the specified directory. Default <code>""</code> .
<i>g_nAnisotropicValue</i>	Sets the max anisotropic filter value to use if anisotropic filtering is ON in the setup UI. Valid values: 0 = max available

(default), 2, 4, 8, 16 (not recommended for AMD). This is a potential fix for coastlines shimmering with AMD cards.

g_fSmartScalingThreshold

If SmartScaling is enabled via the UI it sets the distance in nautical miles from your own POV where Smartscaling will start, for pilots who want to use SmartScaling for better visual identification at far distances, but do NOT want any scaling in very close proximity, i.e. to make sure close formation or HUD cues are not distorted and tyres sit on the taxiway. Default 0.0.

g_bDisableCommsMenu

If set to 1/true, the comms menu (tower/awacs/wingman etc.) will no longer be drawn. Default 0/false.

In addition to the *g_bDisableCommsMenu* config option, the comms menu can also be switched on and off in-game now using the new chatline dot command ".commsmenu N" (N = 0: menu off, N = 1: menu on). This should make it much easier to debug "failing" voice commands in-game.

g_bAutoLoadCommPlanToDTC

Enables auto-loading of the Comm Plan frequencies associated with your flight in the DTC. Default 1/true.

g_nJpegCompression

Sets the compression level for JPG screenshots, 0-100 (default 90)

g_nMaxBRAARange

This determines the distance at which the radio communications will use BRAA instead of BE, default 25

g_nBumpIntensityMinTime

This determines the minimum bump intensity "peak time", how long a "bump" value is visible in shared memory (default 100)

g_bOnlinePlayersDisplayDefault

Specifies whether the OnlinePlayersDisplay overlay should be ON (1) or OFF (0) by default when entering 3D. Default 0/false.

g_bShowFrameRateDefault

Specifies whether the frame rate overlay should be ON (1) or OFF (0) by default when entering 3D. Default 0/false.

g_fRadioBalance

The value in DB on how much to shift the UHF/VHF playback volume out of center (into opposite directions), range +/-100.0. Default is 0.0, i.e. both UHF and VHF will be centered. A positive value will shift UHF to the right, VHF to the left. A negative value will shift UHF to the left, VHF to the right.


The rationale behind this new option is simple: 4.34 finally implements the "concurrent UHF/VHF radio playback".

During internal testing, it turned out that it is **really hard** to listen to two independent radio transmissions at the same time, especially if both are using the same voice (which can happen in BMS), and are co-located acoustically. So while such a "channel separation" does probably not exist in the real jet, it is one



	"sanity option" to allow for easier perception in concurrent playback cases.
<i>set g_nMaxAivsIVCOffset</i>	specifies the maximum +/- offset for the Ai vs IVC balance (range 0 - 10000, default 500)
<i>g_bLabelShowDistance</i>	Enable (1) or disable (0) the distance (in Nm) information on the labels (if labels are turned on). Available options: 0/1.
<i>g_nNearLabelLimit</i>	Sets the near label limit in Nm. You can limit the number of objects shown with near labels by reducing the value. If an object with default setting would be normally shown at a distance 50nm and you set it to 25, it will appear when within 25 nm of your jet. Default = 100.
<i>g_bSmartCombatAP</i>	Enable (1) or disable (0) the Combat autopilot to shoot AA missiles. Available options: 0/1.
<i>g_nNumberOfSubTitles</i>	Determines the maximum number of simultaneously displayed subtitles. Default = 10.
<i>g_nSubTitleTTL</i>	Sets the time in milliseconds a radio subtitle is displayed. Default = 10000. TTL = Time to Live.
<i>g_fSubtitleWrapWidth</i>	Radio subtitle text line width before it gets wrapped, 0.0 (nothing) to 2.0 (full view width) Default = 0.6.
<i>g_bRealisticMavTime</i>	Enable (1) or disable (0) realistic maverick seeker head gyro spool up time of 3 minute. Available options: 0/1.
<i>g_fMavFOVLevel</i>	This option sets the horizontal FOV in degrees for the narrow WPN view of the maverick. Valid range: degrees. Default = 4.
<i>g_fMavEXPLLevel</i>	This option sets the horizontal field of view in degrees for the EXP views of the Maverick WPN page. The smaller the value the higher the zoom. Valid ranges: degrees. Default 2.0.
<i>g_bNoRPMOnHud</i>	This option when set to 1 removes the RPM indication on your HUD as in the real aircraft. 0 displays RPM on HUD. Default =1
<i>g_fAIRefuelSpeed</i>	This option allows speeding up AI refuelling by multiplying the AAR time by the set variable. Default = 1 normal speed.
<i>g_bCanopyOpenForRampStarts</i>	Enable (1 default) or disable (0) the open canopy at ramp start.
<i>g_bServer</i>	This option when set to 1 (enable) puts Falcon BMS into Multiplayer Server mode. A server mode session can't enter the 3D world. Available options: 0/1. Default = 0.
<i>g_bUsePsTracers</i>	Enable (1) or disable (0) the particle system for tracers. Available options: 0/1. Default = 1.





<code>g_bAnyWaypointTask</code>	Enable (1) or disable (0) the option to assign any task to any waypoint. Available options: 0/1. Default = 1.
<code>g_bAIGloc</code>	Enable (1) or disable (0) AI G-Loc. Default = 1.
<code>g_bAIjamLogic 0</code>	Enable (1) or disable (0) new AI jamming logic code. The new logic should be more realistic, but has not been fully tested yet. The new logic tries to adapt jamming according to the opposition capabilities (to prevent HOJ for instance). The old logic was using the jammer according to flight lead usage or RWR spikes (i.e. switching on the jammer in reaction to a spike, even when in HOJ range of capable weapons). Default = 0.
<code>g_bScramble 1</code>	Enable (1) or disable (0) Scramble missions in Campaign. Default = 1.
<code>g_b3DClickableCursorChange</code>	When enabled (1) the mouse cursor changes over an 3D cockpit hotspot to let the pilot know he is over a clickable area. Available options: 0/1. Default = 1.
<code>g_sRadioStandardCol "0xFFFF0000"</code>	Defines the colour of the standard comms subtitles to a specific hex value. Default = 0xFFFF0000 = bright blue.
<code>g_sRadioTowerCol "0xFF00FF00"</code>	Defines the colour of the tower comms subtitles to a specific hex value. Default = 0xFF00FF00 = bright green.
<code>g_sRadioflightCol "0xFF0000FF"</code>	Defines the colour of the to/from flight comms subtitles to a specific hex value. Default = 0xFF0000FF = bright red.

You can also set specific colours (hex codes as above) for more radio subtitles:

<code>g_sRadioToFromPackageCol</code>	for comms to and from package.
<code>g_sRadiotoPackageCol</code>	for comms to package.
<code>g_sRadioTeamCol</code>	for guard (team) comms.
<code>g_sRadioProximityCol</code>	for proximity comms.
<code>g_sRadioWorldCol</code>	for broadcast comms.

Please note: BMS has a specific hex code structure valid for all hex codes in the config files: 0xAABBGGRR. AA are the alpha channel values (00=fully solid – FF=fully transparent), BB are the Blue values, GG are the Green values and RR are the Red values. The alpha channel values may be optional in some config lines.

<code>g_nTrackIRTimeout</code>	Sets the time in milliseconds to let TIR initialize. If no TIR is detected after that time TIR will be deactivated. (TIR SDK value) Default = 1000.
--------------------------------	---



<i>g_sThreatCircleColor_RadarHigh "0xFF0000"</i>	Sets the colour of the UI map threat circles for 'Radar High' as a hex value. Default is 0xFF0000 = blue. Hex format is 0xBBGGRR (no alpha channel).
<i>g_sThreatCircleColor_RadarLow "0xFF0000"</i>	Sets the colour of the UI map threat circles for 'Radar Low' as a hex value. Default is 0xFF0000 = blue. Hex format is 0xBBGGRR (no alpha channel).
<i>g_sThreatCircleColor_ADALow "0xFF00FF"</i>	Sets the colour of the UI map threat circles for 'Air Defense Low' as a hex value. Default is 0xFF00FF = magenta. Hex format is 0xBBGGRR (no alpha channel).
<i>g_sThreatCircleColor_ADALow "0xFF00FF"</i>	Sets the colour of the UI map threat circles for 'Air Defense Low' as a hex value. Default is 0xFF00FF = magenta. Hex format is 0xBBGGRR (no alpha channel).
<i>g_nThreatCircleContrast_RadarHigh</i>	Sets the strength of the colour mixing if 2 or more threat circles of type 'Radar High' overlap. Range 0-100. Default = 40.
<i>g_nThreatCircleContrast_RadarLow</i>	Sets the strength of the colour mixing if 2 or more threat circles of type 'Radar Low' overlap. Range 0-100. Default = 40.
<i>g_nThreatCircleContrast_ADALow</i>	Sets the strength of the colour mixing if 2 or more threat circles of type 'ADA Low' overlap. Range 0-100. Default = 40.
<i>g_nThreatCircleContrast_ADALow</i>	Sets the strength of the colour mixing if 2 or more threat circles of type 'ADA Low' overlap. Range 0-100. Default = 40.
<i>g_bSaveLegacyDb</i>	If set to 1/true, synchronizes the bin files to match the xml files when the game/editor is started. Default 0/false.
<i>g_nTTSSpeedAdjust</i>	Sets the TTS talking speed, default speed is 0, range -10 to 10.
<i>g_sPPTRingColor</i>	The color of the UI map PPT rings, set as hex, "0xBBGGRR" Default "0x0000FF".
<i>g_nPPTRingWidth</i>	The width in pixels of the UI map PPT rings. Default 1.

The last few lines are Debug config options and should be ignored (i.e. left disabled by regular users).

g_bActivateDebugStuff Enable (1) or disable (0) Debug options.

set g_nShowDebugLabels, set g_bCampLabels, set g_bAIprofile, set g_bShowMipUsage, set g_b3DClickableCockpitDebug, set g_bFake3dpit6DOF are all debug options

g_bDevelopmentCallbacks If set to 1/true, the development callbacks in the keyfile are active (depends on *g_bActivateDebugStuff*) Default 0/false.





[g_bEnableCombatAP](#)

If set to 1/true, the autopilot will act as CombatAP again (depends on [g_bActivateDebugStuff](#)) Default 0/false.

The following 3 settings are not being used by BMS but by the editor which also reads the config file. They allow the editor to trigger third party applications from the editor by giving them the correct path. Simply add your computer path to the correct application:

[set g_sPathFlightModelManager](#) "D:\FlightModelManager\Flight Model Manager.exe

Set local path for the FMM (Flight Model Manager)

[set g_sPathMissileDataManager](#) "D:\MissileDataManager\Missile Data Manager.exe"

Set local path for the MDM (Missile Data Manager)set

[g_sPathVehicleDataManager](#) "D:\VehicleDataManager\Vehicle Data Manager.exe"

Set local path for the VDM (Vehicle Data Manager)set

There are others config options that are not included in the config file but which may still be useful to some users. The code still supports them and they can be added in your config file should you need them. It is important to realise that if an option is not in the config file it is not officially supported by BMS, so you use it at your own risk. Assume that they are MP critical and all members of a MP flight should have the same config lines enabled.

[g_fAVTRSeconds 30](#)

Sets the time in seconds that the Auto ACMI feature (on the AVTR panel) should run after each trigger press. Default is 30 seconds.

[g_nRampMinutes](#)

Sets the time in minutes before Takeoff time players will join 3D. Default is 20. With newer F-16 blocks equipped with EGI the alignment time is reduced from 8 to 4 minutes. In some cases 20 minutes ramp start might be too long and can be reduced with this option.

[g_bUseDeprRetAsBrake](#)

Enables use of a unipolar avionics' axis mapping to the analog braking function. Specifically, if you have a set of driving sim pedals that you want to use with Falcon BMS this can give you a way to use them to simulate the F-16 rudder bar and brakes. [g_bUseDeprRetAsBrake](#) only works with [g_bUseVirtualRudder](#) (lets you use the left and right toe brake axis mappings to simulate rudder input) set to 1 (true). What you can then do is map your driving clutch and accelerator pedals in the setup UI to the right and left (respectively; try it before you say "that's backwards...") and the driving brake pedal to the DEPR RET axis, with the net result of analog rudder input by using both pedals and single channel analog brakes in the jet by pressing on the driving brake control.

[g_nKnobAccelerationDelta](#)

When set to 0 it will make the "By1" and default "By5" callbacks work without acceleration, i.e. By1 =1 and By5=5. Default = 60 milliseconds in which the next input has to occur.

If [g_nKnobAccelerationDelta](#) is **not** 0 it will not change the



"By1" callbacks at all. Instead the "Normal" callbacks will behave as "By1" if used once and accelerated to "By5" if used in rapid succession. Basically if you move the controls slowly you get By1 increment, if you move them faster you get By5 increments. Use *SimHsiCourseInc*, *SimHsiCourseDec*, *SimHsiHeadingInc*, *SimHsiHeadingDec*, *SimAltPressInc*, *SimAltPressDec* "normal" callbacks for acceleration to work, not the By1 callbacks.

g_bAllowAllRefreshRates

When set to 1 this option disables the 60 Hz limit of BMS, allowing BMS to display resolutions with lower refresh rates (when using HDMI on larger monitors for instance). Default value is 0; Range 0/1.

g_nAbLightsSwitchOffDelay

Sets the time in seconds that the ATC lights stay on for. Default is 150 seconds.

g_bLogInputFunctions

When set to 1/TRUE, the MonoLog will have entries for every input function that is being called, in the format: "INPUT: <function> (DOWN|UP)". Can be helpful for keyfile debugging etc. Default 0/OFF.

13.9 New 4.35 (& 4.34.1, 2, 3 & 4) Options

g_bVisorUpByDefault

Variable to set visor in the up position upon entering 3D cockpit in other option than ramp. Ramp always starts with visor Up regardless of this setting. Range 0/1 Default is false (0)

g_bShutDownSAMWhenEmpty

Variable to shut down SAM Radar when no radar missiles are left in the battery. Range 0/1 Default is false (0)

g_nFairCloudRestriction

Make Clouds density and number of cells in Fair weather configurable via cfg, that will override all sliders settings or map settings. This saves a lot of FPS for small configurations. Range 0 to 4 (default 0).

0 Standard model, from UI slider / Weather Map or Server Setting,

1 reduces the number of cells displayed from 9 to 4,

2 forces the density to max 3/8,

3 forces the density to max 2/8

4 forces the density to max 1/8



[g_nPoorInclCloudRestriction](#)

Make Clouds density and number of cells in Poor and Inclement weather configurable via cfg, that will override all sliders settings or map settings. This saves a lot of FPS for small configurations. Range 0 to 3 (default 0)

0 Standard model, from UI slider / Weather Map or Server Setting

1 reduces the number of cells displayed from 9 to 4

2 forces the density to max 6/8

3 forces the density to max 4/8

Note : *As this will force different cloud display from the server , server can allow to use it or not.*

[g_bForceMPCloudSettings](#)

Server setting to allow [g_nFairCloudRestriction](#) & [g_nPoorInclCloudRestriction](#) on client's side. Range 0/1 default 0

[g_nCloudRotatingMethod](#)

Allow users to choose the method of cloud rotation they wish.

Code logic is:

in cockpit view: face camera direction when close and face camera position when far.

In other views: Face camera direction.

Range from -1 to 2. Default (-1)

-1: method is defined by code logic

0: cockpit method

1: face camera position

2: face camera direction.

[g_bEnforceServerIn3dWorldBeforeClients](#)

Setting to force the MP host to be the first in 3D. 1 minute before the first client, independently of his own take off time or ramp start setting. Range 0/1. Default is true (1)

[g_fTankerFerryDistance](#)

Possibility to set up the tanker for FERRY refueling. In case refuel waypoints are distant from more than [g_fTankerFerryDistance](#) value, the tanker will no more apply the standard 20 NM racetrack but will fly a track from waypoint to waypoint. This allow to refuel -en route- should a TE designer chose to do so. This should not be the case when tankers are generated by ATO since the distance between waypoint is smaller than default [g_fTankerFerryDistance](#).

Range: Value in Nautical mile. Default is 60

[g_nNumOfPOVs](#)

Setting to override number of POV hats.

Should be in 0..2 range, -1 is override disabled

[g_nPOV1DeviceID](#)

Setting to override POV1 hat device ID,

-1 is override disabled





<i>g_nPOV1ID</i>	Setting to override POV1 hat ID. Should be in 0..1 range
<i>g_nPOV2DeviceID</i>	Setting to override POV2 hat device ID, -1 is override disabled
<i>g_nPOV2ID</i>	Setting to override POV2 hat ID. Should be in 0..1 range

The following config settings are meant to help players with less precise controllers. Allows to set exponential curves and saturation for PITCH (Y), ROLL (X), YAW (rudder) and CURSOR X&Y (Cursors) axis. The latest should help with the usual TM microstick issues. These settings work properly with the advanced UI axis settings.

<i>set g_nAxisExp_AXIS_PITCH</i>	Variable to set exponential curve for PITCH axis. Should be in 0-100 range. 0 is disabled
<i>set g_nAxisSat_AXIS_PITCH</i>	Variable to set exponential saturation for PITCH axis. Should be in 0-99 range. 0 is disabled
<i>set g_nAxisExp_AXIS_ROLL</i>	Variable to set exponential curve for ROLL axis. Should be in 0-100 range. 0 is disabled
<i>set g_nAxisSat_AXIS_ROLL</i>	Variable to set exponential saturation for ROLL axis. Should be in 0-99 range. 0 is disabled
<i>set g_nAxisExp_AXIS_YAW</i>	Variable to set exponential curve for YAW axis. Should be in 0-100 range. 0 is disabled
<i>set g_nAxisSat_AXIS_YAW</i>	Variable to set exponential saturation for YAW axis. Should be in 0-99 range. 0 is disabled
<i>set g_nAxisExp_AXIS_CURSOR_Y</i>	Variable to set exponential curve for CURSOR Y axis. Should be in 0-100 range. 0 is disabled. Suggested value for fixing a faulty microstick: 30
<i>set g_nAxisSat_AXIS_CURSOR_Y</i>	Variable to set exponential saturation for for CURSOR Y axis.. Should be in 0-99 range. 0 is disabled
<i>set g_nAxisExp_AXIS_CURSOR_X</i>	Variable to set exponential curve for CURSOR X axis. Should be in 0-100 range. 0 is disabled. Suggested value for fixing a faulty microstick: 30
<i>set g_nAxisSat_AXIS_CURSOR_X</i>	Variable to set exponential saturation for for CURSOR X axis.. Should be in 0-99 range. 0 is disabled





13.10 Options removed from 4.35

<i>g_bHelosReloc</i>	<p>Obsolete due to the new AWACS code</p> <p>Enable (1) or disable (0) helicopter squadrons to relocate faster. Default = 1.</p>
<i>g_bRealisticAttrition</i>	<p>Obsolete</p> <p>Enable (1) or disable (0) the campaign engine to subtract destroyed vehicles and aircraft along with any munitions carried. Default = 1.</p>
<i>g_bNewThreadTiming</i>	<p>Obsolete</p> <p>If experiencing hiccups/stuttering in the UI with multi-core processors, try setting this to 0. Available options: 0/1. Default is 1</p>
<i>g_bVoiceCom</i>	<p>Obsolete</p> <p>Enable (1) or disable (0) Internal Voice Communications (IVC). Available options: 0/1.</p>
<i>g_bDoubleRTTResolution</i>	<p>Obsolete as it is managed with RTT options</p> <p>This option when enabled doubles the resolution of the cockpit displays (MFDs, HUD, HMS, RWR, DED, PFL), internal and external. Disable it only if your integrated GPU has less VRAM and/or if you encounter related CTDs. Available options: 0/1.</p>
<i>g_bEnableTTS</i>	<p>Obsolete as TTS seems to be stuck for ATIS implementation and therefore shoun't be removed</p> <p>If set, globally enables the Text To Speech (TTS) functionality. Default 1/true.</p>
<i>g_bTripleBuffering</i>	<p>Obsolete as it is managed in the Ui with 4.35</p> <p>Enables (1) or disable (0) DirectX triple buffering. This should give a slight FPS boost on most graphic cards, especially if vSync is on or if external displays are in use.</p>
<i>g_bUseExternalWindows</i>	<p>Obsolete as it is managed with RTT</p> <p>Enables rendering of cockpit displays to external windows and forces windowed mode on (when enabled). External display management is made with the Cockpit Display Extraction app available from the Launcher menu. Options: 0/1 — Default is 0.</p>
<i>g_bExternalWindowsOnTop</i>	<p>Obsolete as it is managed with RTT</p> <p>When enabled this option ensures that external windows are always rendered on top of other windows and the taskbar. Available options: 0/1 — Default is disabled.</p>



<i>g_bFilterExternalWindows</i>	<i>Obsolete as it is managed with RTT</i> Enable (1) or disable (0) anisotropic filtering to external windows (if in use). Default =1.
<i>g_bPixelLighting</i>	<i>Obsolete due to DX11 migration</i> Enable (1) or disable (0) light to be computed for every pixel. This option gives best results.
<i>g_bVertexLighting</i>	<i>Obsolete due to DX11 migration</i> Enable (1) or disable (0) light to be interpolated between vertexes. This option gives best FPS. <i>Please note: if enabled some effects will not be visible.</i>
<i>g_bUseMotionBlurShader</i>	<i>Obsolete due to DX11 migration</i> Enable (1) or disable (0) Motion Blur effects.
<i>g_fMotionBlurFactor</i>	<i>Obsolete due to DX11 migration</i> If the Motion Blur shader is enabled this option controls how strong the blur effect will be (the higher the number, the stronger the effect. 1.0 is MAX. Default is 0.2
<i>g_bCockpitShadows</i>	<i>Obsolete due to DX11 migration</i> Enable (1) or disable (0) shadows for the cockpit.
<i>g_bFocusShadows</i>	<i>Obsolete due to DX11 migration</i> Enable (1) or disable (0) shadows an area around your aircraft.
<i>g_bWaterNormalMapping</i>	<i>Obsolete due to DX11 migration</i> Enable (1) or disable (0) a normal map to animate and more accurately compute reflections on water surfaces. Disabling this option will make the submenu options unavailable.
<i>g_bSkipAggregationBWCheck</i>	Not needed anymore If set to 1 (on) this will remove the static bandwidth check for deagg/reagg network message (default 1/on). Host controlled. This should reduce the risk of 'ghost flights' for low BW users.
<i>g_bForce16bitDisplay</i>	Not needed anymore This option when enabled (1) would force the display device to use 16 bits. This is pretty irrelevant for genuine users but may prove useful when trying to run BMS on a server that may have restricted GPU capabilities. Available options: 0/1. Default is 0.
<i>g_bStrictFogOfWar</i>	Not needed anymore



13.11 Options removed from 4.34

g_bAWACSRequired

Obsolete due to the new AWACS code

User will not get AWACS comms unless an AWACS is present in flight. Available options: 0/1. Default is 1 (AWACS required).

g_bUseAggressiveIncompleteA2G

Made active by default in the code.

Available options: 0/1. When enabled this option lets AI flights with unfinished A-G tasks engage enemy aircraft even if their ground task is not yet finished. They will not jettison stores. If this option is disabled the AI will concentrate on the ground task and try to avoid the enemy aircraft by cranking away. Default: 0

g_bAllowAICommsDrop

Obsolete due to the new Radio code.

This option enables the AI flight comms radio message filter. When enabled (1) you will only hear intra-flight (VHF) radio traffic from AI in your own flight. Intra-flight AI radio traffic (VHF) from other AI flights in your package will not be heard. This simulates different VHF frequencies for different flights within the same package. Available options: 0/1. Default is 1 (enabled).

g_bAllowAICommsDrop_NoRadioPower

Obsolete due to the new Radio code.

When enabled (1) this option ties the AI flight comms radio message filter to the status of the user's radio system. If the user radios are off AI flight radio traffic will not be heard. Available options: 0/1. Default is 1 (enabled).

g_bCenterUI

Obsolete because the main 2D is managed in the Cockpit Display Extraction tool

enabled (1) the 2D UI in window mode (1024x768) will be centred horizontally and vertically on the desktop (for triple screen users for instance). Available options: 0/1. Default = 0.

g_nEnableNewLineup:

obsolete due to new ATC/taxi/ground control code





13.12 TrackIR Axis Customisation

Additional support to swap TIR axes around. You may zoom in with the yaw axis, map pitch to x; whatever you want.

.cfg file entries:

g_nTrackIRYawMapping
g_nTrackIRPitchMapping
g_nTrackIRRollMapping
g_nTrackIRXMapping
g_nTrackIRYMapping
g_nTrackIRZMapping

Map to these constants:

TIR_YAW = 0
TIR_PITCH = 1
TIR_ROLL = 2
TIR_X = 3
TIR_Y = 4
TIR_Z = 5

So, to swap pitch and yaw you would write:

set g_nTrackIRYawMapping 1
set g_nTrackIRPitchMapping 0

Please note: Do not include these lines if you do not want to change anything, or use -1 (which means the exe ignores this line). None of these entries appear in the Config Editor: approach with caution!





14. Shared Memory Changes

14.1 New 4.35.0 changes

FalconSharedMemoryArea2 updated to v17

- Added SolenoidStatus in MiscBits of FlightData2 (0 working - 1 not working)
- Added FLCs Branches Bits (A, B, C, D) in MiscBits of FlightData2 (previously only 1 bit was present for the 4 branches preventing proper failure response)
- Added Inlet_Icing in the LightBits3

Added completely new area DrawingData/FalconSharedMemoryAreaDrawing:

- Dynamically sized (check FlightData2->DrawingAreaSize for the actual size)
- Area will hold vector drawing commands for HUD, RWR, HMS as parsable strings.

14.2 Changes for 4.34.1, not previously documented

Added more data to the shared memory area "FalconSharedMemoryArea2" (Updated to v16):

- Radar altitude
- Bingo fuel level
- Cara Allow setting
- Bullseye X/Y
- BMS version information
- Indication which Betty sound is playing
- Turn rate in degrees/second

Added a completely new shared memory area "FalconSharedMemoryAreaString" (v3)

- BMS exe name, full path
- Key file name in use, full path
- Various BMS directories in use
- Name of the current theatre
- Various theatre directories in use
- Current aircraft name/type and NCTR
- Current 3dbuttons.dat/3dckpit.dat full path
- NavPoints





14.3 4.34.0 changes

Some additions were made in FLightData2:

- Added ownship Latitude and Longitude,
- Added bumpintensity
- Added Rtt values size and area 0 to 6. These are specific to the RTT remote new application

Updated Falcon Shared Memory Area2 to v13: added IFF panel status

- IFF panel backup Mode1 digit 1
- IFF panel backup Mode1 digit 2
- IFF panel backup Mode3A digit 1
- IFF panel backup Mode3A digit 2

14.4 From 4.32 to 4.33

- A realistic delay of 3 seconds was implemented on the MASTER CAUTION light (see below).
- New behaviour for TWP missile launch TEST button and SYS TEST button (see below).
- New bits in Lightbits 3:
 - ATF_Not_Engaged = 0x10000000 for the TFR caution light.
 - SYS TEST button 0x1000000 for TWP system test.
 - MCAnnounced = 0x2000000 for MC delay.
 - MLGWOW = 0x4000000 & NLGWOW = 0x8000000 for AFM needs (WOW switches).
- Flying bit (0x80000000) is now updated accordingly for Ramp start when the user goes to 3D. This bit is useful to state when the user is in the cockpit (it can be used to start extracting data for instance).
- DED will have a realistic 30 second warm-up time after UFC is powered on. It is therefore normal to see a delay between UFC power on and the DED starting to display information.
- A delay was implemented before illumination of the JFS run light after JFS start-up. As a consequence, if you used the JFS RUN light bit (JFSOn = 0x200000) to drive the JFS magnetic switch you must change the bit to the new JetFuelStarter = 0x40 bit.
- The WOW lightbit (0x10) has been renamed to ONGROUND bit with the same address.
- Flightdata2 was completely overhauled with many more features:
 - Pressure setting for the altimeter in hPa or inHG.
 - PNEU flag for the altimeter.
 - Cabin pressure value.
 - BUP UHF presets and frequencies.
 - CMDS mode state (OFF, STBY, MAN, SEMI, AUTO, BYP).

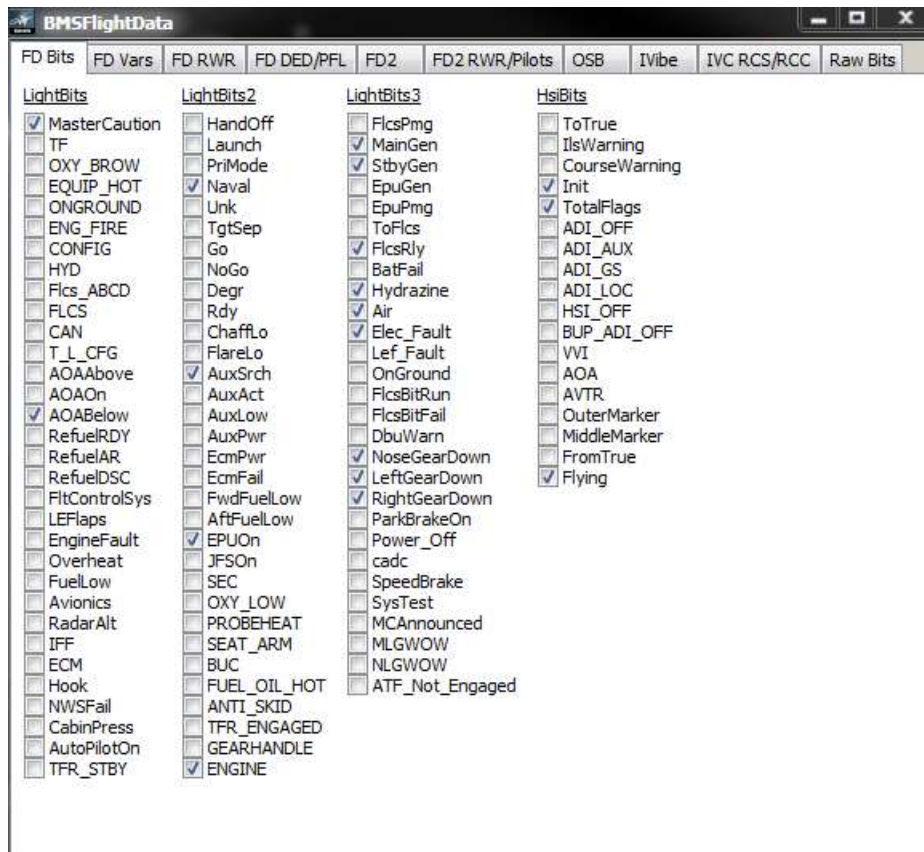




- HSI mode (ILS/TCN, TCN, NAV, ILS/NAV).
- Aircraft type.
- Current time and human player ID (location).
- Hydraulic A&B pressure value.
- LEF & TEF position.
- VTOL nozzle angle.
- New blinking light classification (the sharedmem is now able to state if a lightbit must blink).

Please note: the actual blinking implementation is left to the third party apps developers' discretion. Implemented blinking lightbits: Outer and Inner marker, Probeheat, Aux Search, Launch, PriMode, Unk). See the flightdata.h file now located in \Tools\SharedMem\ folder for further information.

The \Tools\SharedMem\ folder also contains a raw shared memory reader application that may be useful for testing sharedmem content: BMSFlightData.exe



15. New Keyboard Commands

15.1 Quick Reference

Since the first official Falcon BMS release (4.32), a number of additional keystrokes were added to the game.

These are listed below, including the key mappings for the two different keyboard layouts. Also a couple of old and outdated callbacks have been removed. Check renamed and deleted callbacks at the end of this chapter.

Please note that the following keybindings show US International keyboard assignments.

New Callback	Ver.	Non-Pitbulders File		Pitbulders Key File		Description
		Modifier	Key	Modifier	Key	
SimILSup	4.35.0	Shft Ctrl	Y	Ctrl Alt	U	AUDIO2: ILS Knob – Volume Incr.
SimILSDown	4.35.0	Shft Alt	Y	Shft Ctrl Alt	U	AUDIO2: ILS Knob - Volume Decr.
SimJfsStartCycle	4.35.0	Shft	J	none	none	ENG: JFS Switch - Cycle 1 / OFF / 2
SimJfsStartInc	4.35.0	none	none	none	none	ENG: JFS Switch - Step Up
SimJfsStartDec	4.35.0	none	none	none	none	ENG: JFS Switch - Step Down
SimJfsStartUp	4.35.0	none	none	Ctrl Alt	Q	ENG: JFS Switch - START 1
SimJfsStartMid	4.35.0	none	none	Shft Ctrl Alt	Q	ENG: JFS Switch - OFF
SimJfsStartDown	4.35.0	none	none	Shft Ctrl	Q	ENG: JFS Switch - START 2
AFCanopyInc	4.35.0	Alt	Q	Shft Alt	V	LEFT WALL: CANOPY - Open
AFCanopyStop	4.35.0	none	none	Alt	B	LEFT WALL: CANOPY - Stop
AFCanopyDec	4.35.0	Alt	W	Ctrl Alt	V	LEFT WALL: CANOPY - Close
AFCanopyLockToggle	4.35.0	Alt	S	none	none	LEFT WALL: SPIDER - Toggle Open/Close
AFCanopyLock	4.35.0	none	none	Shft Alt	B	LEFT WALL: SPIDER - Lock
AFCanopyUnlock	4.35.0	none	none	Ctrl Alt	B	LEFT WALL: SPIDER - Unlock
SimSeatUp	4.35.0	none	none	Ctrl	N	SEAT: Move Up
SimSeatDown	4.35.0	none	none	Alt	N	SEAT: Move Down
SimThrottleIdleDetentForward	4.35.0	none	none	none	none	TQS: CUTOFF RELEASE - Idle Detent - Idle
SimThrottleIdleDetentBack	4.35.0	none	none	none	none	TQS: CUTOFF RELEASE - Idle Detent - Off
SimThrottleIdleDetentLeft	4.35.0	none	none	none	none	TQS: CUTOFF RELEASE - Left Engine
SimThrottleIdleDetentRight	4.35.0	none	none	none	none	TQS: CUTOFF RELEASE - Right Engine
SimLandingLightCycle	4.35.0	Shft	L	none	none	GEAR: LIGHTS Switch - Cycle
SimLandingLightInc	4.35.0	none	none	none	none	GEAR: LIGHTS Switch - Step Up
SimLandingLightDec	4.35.0	none	none	none	none	GEAR: LIGHTS Switch - Step Down
SimLandingLightUp	4.35.0	none	none	Alt	A	GEAR: LIGHTS Switch - LANDING
SimLandingLightMid	4.35.0	none	none	Ctrl	S	GEAR: LIGHTS Switch - OFF
SimLandingLightDown	4.35.0	none	none	Alt	S	GEAR: LIGHTS Switch - TAXI
SimBrakeChannelToggle	4.35.0	Alt	B	none	none	GEAR: BRAKES - Toggle
SimBrakeChannelUp	4.35.0	none	none	Alt	O	GEAR: BRAKES - Channel 1
SimBrakeChannelDown	4.35.0	none	none	Ctrl	P	GEAR: BRAKES - Channel 2
SimParkingBrakeCycle	4.35.0	Alt	P	none	none	GEAR: PARKING BRAKE Switch - Cycle
SimParkingBrakeInc	4.35.0	none	none	none	none	GEAR: PARKING BRAKE Switch - Step Down
SimParkingBrakeDec	4.35.0	none	none	none	none	GEAR: PARKING BRAKE Switch - Step Up

New Callback	Ver.	Non-Pitbulders File		Pitbulders Key File		Description
		Modifier	Key	Modifier	Key	
SimParkingBrakeUp	4.35.0	none	none	Alt	P	GEAR: PARKING BRAKE Switch - ON
SimParkingBrakeMid	4.35.0	none	none	Ctrl	[GEAR: PARKING BRAKE Switch - ANTI SKID
SimParkingBrakeDown	4.35.0	none	none	Alt	[GEAR: PARKING BRAKE Switch - OFF
AFEmergencyGearHandleUnlock	4.35.0	Shft Alt	G	Shft Ctrl Alt	B	GEAR: DN LOCK REL - Push
SimAntilceCycle	4.35.0	Shft	N	none	none	ICE: ENGINE Switch - Cycle
SimAntilceInc	4.35.0	none	none	none	none	ICE: ENGINE Switch - Step up
SimAntilceDec	4.35.0	none	none	none	none	ICE: ENGINE Switch - Step Down
SimAntilceUp	4.35.0	none	none	Shft Ctrl Alt	Num +	ICE: ENGINE Switch - ON
SimAntilceMid	4.35.0	none	none	Shft Ctrl	Num 4	ICE: ENGINE Switch - AUTO
SimAntilceDown	4.35.0	none	none	Shft Alt	Num 4	ICE: ENGINE Switch - OFF
SimAntennaSelectCycle	4.35.0	Ctrl	N	none	none	ANT: IFF UHF Switch - Cycle
SimAntennaSelectInc	4.35.0	none	none	none	none	ANT: IFF UHF Switch - Step Up
SimAntennaSelectDec	4.35.0	none	none	none	none	ANT: IFF UHF Switch - Step Down
SimAntennaSelectUp	4.35.0	none	none	Ctrl Alt	Num 4	ANT: IFF UHF Switch - UPPER
SimAntennaSelectMid	4.35.0	none	none	Shft Ctrl Alt	Num 4	ANT: IFF UHF Switch - NORM
SimAntennaSelectDown	4.35.0	none	none	Shft Ctrl	Num 5	ANT: IFF UHF Switch - LOWER
SimRightKneePadInc	4.35.0	Ctrl	Insert	none	none	CKPIT: Right Kneeboard - Inc
SimRightKneePadDec	4.35.0	none	none	none	none	CKPIT: Right Kneeboard - Dec
SimLeftKneePadInc	4.35.0	Ctrl	Page Up	none	none	CKPIT: Left Kneeboard - Inc
SimLeftKneePadDec	4.35.0	none	none	none	none	CKPIT: Left Kneeboard - Dec
SimIFFMasterInc	4.34.0	Shft Ctrl	F8	Shft Ctrl	F8	AUX: MASTER Knob - Step Up
SimIFFMasterDec	4.34.0	Shft Ctrl	F7	Shft Ctrl	F7	AUX: MASTER Knob - Step Down
SimIFFMasterOff	4.34.0	none	none	none	none	AUX: MASTER Knob - OFF
SimIFFMasterStby	4.34.0	none	none	none	none	AUX: MASTER Knob - STBY
SimIFFMasterLow	4.34.0	none	none	none	none	AUX: MASTER Knob - LOW
SimIFFMasterNorm	4.34.0	none	none	none	none	AUX: MASTER Knob - NORM
SimIFFMasterEmerg	4.34.0	none	none	none	none	AUX: MASTER Knob - EMER
SimIFFCodeSwitchZero	4.34.0	none	none	none	none	AUX: M-4 CODE Switch - ZERO
SimIFFCodeSwitchHold	4.34.0	none	none	none	none	AUX: M-4 CODE Switch - HOLD
SimIFFMode4ReplyCycle	4.34.0	Shft Ctrl	F11	Shft Ctrl	F11	AUX: REPLY Switch - Cycle
SimIFFMode4ReplyInc	4.34.0	none	none	none	none	AUX: REPLY Switch - Step Up
SimIFFMode4ReplyDec	4.34.0	none	none	none	none	AUX: REPLY Switch - Step Down
SimIFFMode4ReplyBravo	4.34.0	none	none	none	none	AUX: REPLY Switch - B
SimIFFMode4ReplyAlpha	4.34.0	none	none	none	none	AUX: REPLY Switch - A
SimIFFMode4ReplyOff	4.34.0	none	none	none	none	AUX: REPLY Switch - OUT
SimIFFMode4MonitorToggle	4.34.0	Shft Ctrl	F12	Shft Ctrl	F12	AUX: MONITOR Switch - Toggle
SimIFFMode4MonitorAud	4.34.0	none	none	none	none	AUX: MONITOR Switch - AUDIO
SimIFFMode4MonitorOff	4.34.0	none	none	none	none	AUX: MONITOR Switch - OUT
SimIFFBackupM1Digit1Inc	4.34.0	Shft Alt	F2	Shft Alt	F2	AUX: IFF MODE 1 - Cycle Up Left Digit
SimIFFBackupM1Digit1Dec	4.34.0	none	none	none	none	AUX: IFF MODE 1 - Cycle Down Left Digit
SimIFFBackupM1Digit2Inc	4.34.0	Shft Alt	F3	Shft Alt	F3	AUX: IFF MODE 1 - Cycle Up Right Digit
SimIFFBackupM1Digit2Dec	4.34.0	none	none	none	none	AUX: IFF MODE 1 - Cycle Down Right Digit
SimIFFBackupM3Digit1Inc	4.34.0	Shft Alt	F4	Shft Alt	F4	AUX: IFF MODE 3 - Cycle Up Left Digit
SimIFFBackupM3Digit1Dec	4.34.0	none	none	none	none	AUX: IFF MODE 3 - Cycle Down Left Digit
SimIFFBackupM3Digit2Inc	4.34.0	Shft Alt	F5	Shft Alt	F5	AUX: IFF MODE 3 - Cycle Up Right Digit
SimIFFBackupM3Digit2Dec	4.34.0	none	none	none	none	AUX: IFF MODE 3 - Cycle Down Right Digit

New Callback	Ver.	Non-Pitbulders File		Pitbulders Key File		Description
		Modifier	Key	Modifier	Key	
SimIFFEnableCycle	4.34.0	Shft Alt	F1	Shft Alt	F1	AUX: IFF ENABLE Switch - Cycle
SimIFFEnableInc	4.34.0	none	none	none	none	AUX: IFF ENABLE Switch - Step Up
SimIFFEnableDec	4.34.0	none	none	none	none	AUX: IFF ENABLE Switch - Step down
SimIFFEnableM3MS	4.34.0	none	none	none	none	AUX: IFF ENABLE Switch - M3/MS
SimIFFEnableOff	4.34.0	none	none	none	none	AUX: IFF ENABLE Switch - OFF
SimIFFEnableM1M3	4.34.0	none	none	none	none	AUX: IFF ENABLE Switch - M1/M3
SimThrottleIdleDetentForward	4.34.0	none	none	none	none	TQS: CUTOFF RELEASE - Idle Detent - Idle
SimThrottleIdleDetentBack	4.34.0	none	none	none	none	TQS: CUTOFF RELEASE - Idle Detent - Off
SimNVGModeOn	4.34.0	none	none	none	none	CKPIT: Nightvision - On
SimNVGModeOff	4.34.0	none	none	none	none	CKPIT: Nightvision - Off
SimVisorToggle	4.34.0	Alt	V	Alt	V	CKPIT: Visor - Toggle
SimSmokeOn	4.34.0	none	none	none	none	CKPIT: Smoke - On
SimSmokeOff	4.34.0	none	none	none	none	CKPIT: Smoke - Off
SimF18FCSGainToggle	4.34.0	Shft Ctrl Alt	G	Shft Ctrl Alt	G	CKPIT: F-18 FCS GAIN Switch - Toggle
SimF18FCSGainNORM	4.34.0	none	none	none	none	CKPIT: F-18 FCS GAIN Switch - NORM
SimF18FCSGainORIDE	4.34.0	none	none	none	none	CKPIT: F-18 FCS GAIN Switch - ORIDE
SimF18FCSTOTrim	4.34.0	Shft Ctrl Alt	T	Shft Ctrl Alt	T	CKPIT: F-18 FCS T/O TRIM Button
SimLaunchBarToggle	4.34.0	Shft Ctrl Alt	L	Shft Ctrl Alt	L	CKPIT: F-18 LAUNCH BAR Switch - Toggle
SimLaunchBarEXTEND	4.34.0	none	none	none	none	CKPIT: F-18 LAUNCH BAR Switch - EXTEND
SimLaunchBarRETRACT	4.34.0	none	none	none	none	CKPIT: F-18 LAUNCH BAR Switch - RETRACT
SimF18ThrottleATC	4.34.0	Shft Ctrl Alt	A	Shft Ctrl Alt	A	CKPIT: F-18 Throttle - ATC Button
SimPilotToggle	4.34.0	none	Alt C: P	none	Alt C: P	SIM: Toggle Pilot Model
SimTEFCMDInc	4.33.1	Shft	F12	none	none	CKPIT: F-18 FLAP Switch - Step Up
SimTEFCMDDec	4.33.1	Shft	F11	none	none	CKPIT: F-18 FLAP Switch - Step Down
SimTEFCMDAuto	4.33.1	none	none	none	none	CKPIT: F-18 FLAP Switch - AUTO
SimTEFCMDHalf	4.33.1	none	none	none	none	CKPIT: F-18 FLAP Switch - HALF
SimTEFCMDFull	4.33.1	none	none	none	none	CKPIT: F-18 FLAP Switch - FULL
SimDigitalBUPBackup	4.33.0	none	none	Shft Alt	F2	FLT: DIGITAL Switch - BACKUP
SimManualFlyup	4.33.0	Ctrl Alt	F3	none	none	FLT: MANUAL TF FLYUP Switch - Toggle
SimXBandAuxComDigit	4.33.0	none	none	Shft Ctrl Alt	1	AUX: CHANNEL - Toggle Band X
SimYBandAuxComDigit	4.33.0	none	none	Shft Ctrl	2	AUX: CHANNEL - Toggle Band Y
SimEcmPower	4.33.0	Ctrl Alt	W	none	none	ECM: OPR Switch - Toggle
SimJfsStart_Off	4.33.0	none	none	Shft Ctrl Alt	Q	ENG: JFS Switch - OFF
SimJfsStart_Start2	4.33.0	none	none	Shft Ctrl	W	ENG: JFS Switch - START 2
SimEngCont	4.33.0	Ctrl Alt	Y	none	none	ENG: ENG CONT Switch - Toggle
SimILS	4.33.0	Shft Alt	Y	none	none	AUDIO2: ILS Knob - Toggle
SimAud1Com1	4.33.0	Shft Alt	O	none	none	AUDIO1: COMM 1 Mode Knob - Toggle
SimAud1Com2	4.33.0	Shft Alt	P	none	none	AUDIO1: COMM 2 Mode Knob - Toggle
SimBupUhfFreq1Inc	4.33.0	Shft Ctrl	D	none	none	UHF: A-3-2-T Rotary X___.___ - Step Up
SimBupUhfFreq1Dec	4.33.0	Shft Ctrl	S	none	none	UHF: A-3-2-T Rotary X___.___ - Step Down

New Callback	Ver.	Non-Pitbulders File		Pitbulders Key File		Description
		Modifier	Key	Modifier	Key	
SimBupUhfFreq1_2	4.33.0	none	none	Shft Alt	D	UHF: A-3-2-T Rotary 2__._
SimBupUhfFreq1_3	4.33.0	none	none	Ctrl Alt	D	UHF: A-3-2-T Rotary 3__._
SimBupUhfFreq2Inc	4.33.0	Shft Ctrl	F	none	none	UHF: Manual Frequency _X_.__ - Cycle Up
SimBupUhfFreq2Dec	4.33.0	none	none	none	none	UHF: Manual Frequency _X_.__ - Cycle Down
SimBupUhfFreq2_0	4.33.0	none	none	Shft Ctrl Alt	D	UHF: Manual Frequency _0_.__
SimBupUhfFreq2_1	4.33.0	none	none	Shft Ctrl	F	UHF: Manual Frequency _1_.__
SimBupUhfFreq2_2	4.33.0	none	none	Shft Alt	F	UHF: Manual Frequency _2_.__
SimBupUhfFreq2_3	4.33.0	none	none	Ctrl Alt	F	UHF: Manual Frequency _3_.__
SimBupUhfFreq2_4	4.33.0	none	none	Shft Ctrl Alt	F	UHF: Manual Frequency _4_.__
SimBupUhfFreq2_5	4.33.0	none	none	Shft Ctrl	G	UHF: Manual Frequency _5_.__
SimBupUhfFreq2_6	4.33.0	none	none	Shft Alt	G	UHF: Manual Frequency _6_.__
SimBupUhfFreq2_7	4.33.0	none	none	Ctrl Alt	G	UHF: Manual Frequency _7_.__
SimBupUhfFreq2_8	4.33.0	none	none	Shft Ctrl Alt	G	UHF: Manual Frequency _8_.__
SimBupUhfFreq2_9	4.33.0	none	none	Shft Ctrl	H	UHF: Manual Frequency _9_.__
SimBupUhfFreq3Inc	4.33.0	Shft Ctrl	G	none	none	UHF: Manual Frequency __X.____ - Cycle Up
SimBupUhfFreq3Dec	4.33.0	none	none	none	none	UHF: Manual Frequency __X.____ - Cycle Down
SimBupUhfFreq3_0	4.33.0	none	none	Shft Alt	H	UHF: Manual Frequency __0.____
SimBupUhfFreq3_1	4.33.0	none	none	Ctrl Alt	H	UHF: Manual Frequency __1.____
SimBupUhfFreq3_2	4.33.0	none	none	Shft Ctrl Alt	H	UHF: Manual Frequency __2.____
SimBupUhfFreq3_3	4.33.0	none	none	Shft Ctrl	J	UHF: Manual Frequency __3.____
SimBupUhfFreq3_4	4.33.0	none	none	Shft Alt	J	UHF: Manual Frequency __4.____
SimBupUhfFreq3_5	4.33.0	none	none	Ctrl Alt	J	UHF: Manual Frequency __5.____
SimBupUhfFreq3_6	4.33.0	none	none	Shft Ctrl Alt	J	UHF: Manual Frequency __6.____
SimBupUhfFreq3_7	4.33.0	none	none	Shft Ctrl	K	UHF: Manual Frequency __7.____
SimBupUhfFreq3_8	4.33.0	none	none	Shft Alt	K	UHF: Manual Frequency __8.____
SimBupUhfFreq3_9	4.33.0	none	none	Ctrl Alt	K	UHF: Manual Frequency __9.____
SimBupUhfFreq4Inc	4.33.0	Shft Ctrl	H	none	none	UHF: Manual Frequency __X.____ - Cycle Up
SimBupUhfFreq4Dec	4.33.0	none	none	none	none	UHF: Manual Frequency __X.____ - Cycle Down
SimBupUhfFreq4_0	4.33.0	none	none	Shft Ctrl Alt	K	UHF: Manual Frequency __0.____
SimBupUhfFreq4_1	4.33.0	none	none	Shft Ctrl	L	UHF: Manual Frequency __1.____
SimBupUhfFreq4_2	4.33.0	none	none	Shft Alt	L	UHF: Manual Frequency __2.____
SimBupUhfFreq4_3	4.33.0	none	none	Ctrl Alt	L	UHF: Manual Frequency __3.____
SimBupUhfFreq4_4	4.33.0	none	none	Shft Ctrl Alt	L	UHF: Manual Frequency __4.____
SimBupUhfFreq4_5	4.33.0	none	none	Shft Ctrl	;	UHF: Manual Frequency __5.____
SimBupUhfFreq4_6	4.33.0	none	none	Shft Alt	;	UHF: Manual Frequency __6.____
SimBupUhfFreq4_7	4.33.0	none	none	Ctrl Alt	;	UHF: Manual Frequency __7.____
SimBupUhfFreq4_8	4.33.0	none	none	Shft Ctrl Alt	;	UHF: Manual Frequency __8.____
SimBupUhfFreq4_9	4.33.0	none	none	Shft Ctrl	'	UHF: Manual Frequency __9.____
SimBupUhfFreq5Inc	4.33.0	Shft Ctrl	J	none	none	UHF: Manual Frequency __.XX - Cycle Up
SimBupUhfFreq5Dec	4.33.0	none	none	none	none	UHF: Manual Frequency __.XX - Cycle Down
SimBupUhfFreq5_00	4.33.0	none	none	Shft Alt	'	UHF: Manual Frequency __.00
SimBupUhfFreq5_25	4.33.0	none	none	Ctrl Alt	'	UHF: Manual Frequency __.25

New Callback	Ver.	Non-Pitbulders File		Pitbulders Key File		Description
		Modifier	Key	Modifier	Key	
SimBupUhfFreq5_50	4.33.0	none	none	Shft Ctrl Alt	'	UHF: Manual Frequency _____.50
SimBupUhfFreq5_75	4.33.0	none	none	Shft Ctrl	Return	UHF: Manual Frequency _____.75
SimBupUhfManual	4.33.0	none	none	Shft Ctrl Alt	X	UHF: MODE Knob - MNL
SimRwrPowerOn	4.33.0	none	none	Ctrl	F3	TWA: POWER Button - On
SimRwrPowerOff	4.33.0	none	none	Alt	F3	TWA: POWER Button - Off
SimEWSMwsPower	4.33.0	Ctrl Alt	D	none	none	CMDS: MWS Switch - Toggle Power
SimEWSMwsOn	4.33.0	none	none	Ctrl	F9	CMDS: MWS Switch - Power ON
SimEWSMwsOff	4.33.0	none	none	Alt	F9	CMDS: MWS Switch - Power OFF
SimEWSO1Power	4.33.0	Ctrl Alt	F	none	none	CMDS: O1 Switch - Toggle Power
SimEWSO1On	4.33.0	none	none	Ctrl	F10	CMDS: O1 Switch - Power ON
SimEWSO1Off	4.33.0	none	none	Alt	F10	CMDS: O1 Switch - Power OFF
SimEWSO2Power	4.33.0	Ctrl Alt	G	none	none	CMDS: O2 Switch - Toggle Power
SimEWSO2On	4.33.0	none	none	Ctrl	F11	CMDS: O2 Switch - Power ON
SimEWSO2Off	4.33.0	none	none	Alt	F11	CMDS: O2 Switch - Power OFF
SimEwsJettOn	4.33.0	none	none	Ctrl	Q	CMDS: JETT Switch - ON
SimRWRLaunch	4.33.0	Alt	Num 5	Ctrl	Return	TWP: MISSILE LAUNCH - Push
SimRWRSysTest	4.33.0	Alt	Num 2	Alt	Z	TWP: SYS TEST - Push
SimSetWX	4.33.0	Shft	Num /	Shft	Num /	ICP: FLIR - WX Mode
SimFlirLevelUp	4.33.0	Shft	Page Up	Shft	Insert	ICP: FLIR Rocker ▲ - Level Up
SimFlirLevelDown	4.33.0	Shft	Insert	Shft	Delete	ICP: FLIR Rocker ▼ - Level Down
SimBrtWheelUp	4.33.0	Shft Alt	Num +	Shft	K	ICP: BRT Wheel - Increase FLIR Intensity
SimBrtWheelDn	4.33.0	Shft Alt	Num -	Shft	L	ICP: BRT Wheel - Decrease FLIR Intensity
SimAltPressIncBy1	4.33.0	Shft Alt	'	Ctrl	▲	MAIN: Altimeter Pressure Knob - Incr. (1°)
SimAltPressDecBy1	4.33.0	Shft Alt	;	Alt	▲	MAIN: Altimeter Pressure Knob - Decr. (1°)
OTWMouseButtonsIn3dToggle	4.33.0	Alt	3	none	none	CKPIT: Toggle Mouse Btns in 3D
OTWMouseButtonsIn3dEnable	4.33.0	none	none	none	none	CKPIT: Enable Mouse Btns in 3D
OTWMouseButtonsIn3dDisable	4.33.0	none	none	none	none	CKPIT: Disable Mouse Btns in 3D
AFWingFoldToggle	4.33.0	Shft	W	none	none	CKPIT: Wing Fold - Toggle
AFWingFoldUp	4.33.0	none	none	none	none	CKPIT: Wing Fold - Up
AFWingFoldDown	4.33.0	none	none	none	none	CKPIT: Wing Fold - Down
SimStepFormationLightsUp	4.33.0	Shft	X	none	none	CKPIT: Formation Lights - Step Up
SimStepFormationLightsDown	4.33.0	Shft	Z	none	none	CKPIT: Formation Lights - Step Down
AFTtriggerReverseThrust	4.33.0	Alt	T	none	none	FCTRL: ENGINE - Togg. Thrust Reverser
OTWToggleHUDRendering	4.33.0	none	H	none	none	SIM: Toggle HUD Rendering
WinAmpTogglePause	4.33.0	none	none	none	none	WINAMP: Toggle Pause

15.2 Changes For 4.35.0

The cockpit procedure changes are documented in the Dash-1 manual. The purpose of this small chapter is to focus on callback changes and updates in 4.35.

Please see chapter 14.1 for default key assignments.

Eng & Jet Start Panel – Jet Fuel Starter

Callbacks added:

JFS Start 1 (only 1 JFS accumulator is used) implemented.

Callbacks removed:

SimJfsStart	->	replaced by SimJfsStartCycle
SimJfsStart_Off	->	replaced by SimJfsStartMid
SimJfsStart_Start2	->	replaced by SimJfsStartDown

Audio 2 panel - ILS Volume Knob

Callbacks added:

The ILS Volume Knob is now a nine-posit switch (can be assigned to an analogue axis as well) rather than a toggle between on / off

Callbacks removed:

SimILS		
SimILSOn	->	replaced by SimILSUp
SimILSOff	->	replaced by SimILSDown

Gear Panel

Callbacks added:

Added new brake channel switch (channels 1 & 2: FLCC A / B / C / D).

Added ANTI SKID system. Switch is now a 3-position switch.

Runway conditions coded following real landing distance charts in case of wet runway or icing conditions.

Additional support for landing light. Switch is now a 3-position switch.

Improved modelling of the GearHandle. New DN LOCK function added: To be used if solenoid is faulty, first click on DN LOCK, then within 2 seconds move the gear handle.

Callbacks removed:

SimParkingBrakeToggle	->	replaced by SimParkingBrakeCycle
SimParkingBrakeOn	->	replaced by SimParkingBrakeUp
SimParkingBrakeOff	->	replaced by SimParkingBrakeDown
SimLandingLightToggle	->	replaced by SimLandingLightCycle
SimLandingLightOn	->	replaced by SimLandingLightUp
SimLandingLightOff	->	replaced by SimLandingLightMid

Canopy & Spider

Callback added:

Added new canopy open / close logic. Canopy opening / stop / closing is activated with a new 3 posits switch.

Callbacks removed:

AFCanopyToggle		
AFCanopyOpen	->	replaced by AFCanopyInc
AFCanopyClose	->	replaced by AFCanopyDec

TQS – Idle Detent

Callback added:

Added new keystrokes for handling dual engines idle detent. It works like a momentary switch. When pressed, Idle Cut-Off is activated; otherwise, throttle is set past Idle detent (toggle function). It is optimised for TM Warthog. For other devices you may have to use the special press / release events.

SimThrottleIdleDetentLeft, SimThrottleIdleDetentRight

Here is an example for the TM Warthog, left engine: SimThrottleIdleDetentLeft 61 -1 -2 0 0x0 314

Attention: This feature was initially released in 4.34.1. The callbacks SimThrottleIdleDetentCutOff and SimThrottleRightIdleDetentCutOff have been renamed for consistency reasons.

Seat Adjust

Callback added:

New callbacks for Seat Adjust move up (SimSeatUp) and down (SimSeatDown). The switch in the F-16 3d cockpit is located on the right sidewall next to the SNSR PWR panel.

Anti Ice / Ant Sel Panel

Callback added:

Icing conditions implemented depending on total temperature and humidity.

IFF / UHF antenna selector implemented (Note: the callbacks affect both switches and are bogus).

Left & Right Kneeboard

Callback added:

Support for left and right kneeboard pages (16 pages per kneeboard).

The default key assignments are:

SimRightKneePadInc	Ctrl PgUp	SimRightKneePadDec	Not assigned
SimLeftKneePadInc	Ctrl Insert	SimLeftKneePadDec	Not assigned

15.3 Changes For 4.34.0

GAIN Switch (F-18)

Added: [SimF18FCSGainToggle](#), [SimF18FCSGainNORM](#) & [SimF18FCSGainORIDE](#)

The GAIN switch is located on the left console, FCS panel. This function does the following:

ORIDE:

When the GAIN switch is in ORIDE and the FLAP switch in AUTO, the leading and trailing edge flaps are fixed to 3° down and will not vary with airspeed and AOA.

NORM:

When the GAIN switch is in NORM position FLAP operations are as described under 3.1 FLAP Switch (F-18) farther below.

For detailed information about the function please refer to a F-18 Flight Manual, such as A1-F18AC-NFM-000, chapter 2.8.4.2 GAIN Switch, pages I-2-55 & 56.

The new GAIN switch functions are located in section [6.01 Other Cockpit Callbacks](#) in the key files and editor.



Added Hotspot:

[SimF18FCSGainToggle](#)
[SimF18FCSGainNORM](#)
[SimF18FCSGainORIDE](#)

➔ Note:

The seal is animated but the switch is not implemented

The default key assignment for [SimF18FCSGainToggle](#) is **Shift Ctrl Alt G**.

T/O TRIM Button (F-18)

Added: [SimF18FCSTOTrim](#)

The T/O Trim button is in the center of the rudder trim knob on the FCS panel. With WOW, holding the button pressed it sets control surfaces for Take Off.

For detailed information about the function please refer to a F-18 Flight Manual, such as A1-F18AC-NFM-000, chapter 2.8.2.2.7 T/O Trim Button, page I-2-47.

The new T/O TRIM button function is located in section [6.01 Other Cockpit Callbacks](#) in the key files and editor.



Added Hotspot:
SimF18FCSTOTrim

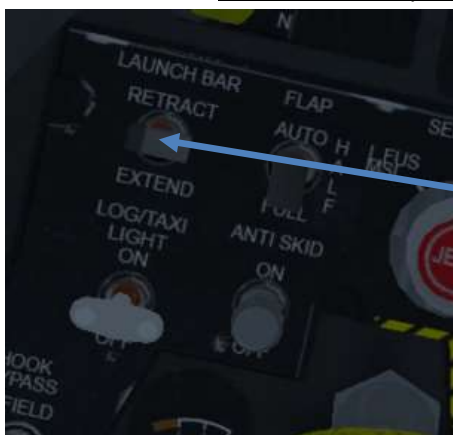
➔ **Note:**
The rotary has no hotspot.

The default key assignment for [SimF18FCSTOTrim](#) is **Shift Ctrl Alt T**.

LAUNCH BAR Switch (F-18)

Added: [SimLaunchBarToggle](#), [SimLaunchBarEXTEND](#) & [SimLaunchBarRETRACTSim](#)

The Launch Bar switch on the forward left console has been added. Now you have to extend the Launch Bar in order to get hooked to the catapult. After launch and prior gear retraction the Launch Bar must be retracted again. Otherwise gear retraction is impossible. The new LAUNCH BAR switch function are located in section [6.01 Other Cockpit Callbacks](#) in the key files and editor.



Added Hotspot:
SimLaunchBarToggle
SimLaunchBarEXTEND
SimLaunchBarRETRACTSim

➔ **Note:**
The Switch is not animated.

The default key assignment for [SimLaunchBarToggle](#) is **Shift Ctrl Alt L**.

Throttle ATC Button (F-18)

Added:

SimF18ThrottleATC

The automatic throttle control is a two-mode system that automatically maintains angle of attack (approach mode) or airspeed (cruise mode). In the real jet the button is located on front side of the left throttle.

ATC APPROACH Mode

With the FLAP switch set to HALF or FULL the thrust is set to maintain the AOA when the ATC button is pressed.

ATC CRUISE Mode

With the FLAP switch set to AUTO the thrust is set to maintain the current airspeed when the ATC button is pressed.

For detailed information about the function please refer to a F-18 Flight Manual, such as A1-F18AC-NFM-000, chapter 2.1.2.1 ATC Approach Mode, pages I-2-8 to I-2-10.

The new ATC button function is located in section 6.01 Other Cockpit Callbacks in the key files and editor.

The default key assignment for SimF18ThrottleATC is **Shift Ctrl Alt A**.

Toggle Pilot Model

Added: SimPilotToggle

Now it is possible to toggle the pilot legs model in any of the F-16 3d cockpits. To make it work in other cockpits as well the model has to be defined in the 3dckpit.dat with cockpitlegsmode. If no model is defined as cockpitlegsmode nothing will happen. The new Toggle Pilot Model function is located in section 6.06 Simulation & Hardware in the key files and editor.



The default key assignment for SimPilotToggle is **Alt C: P**.

Left Console, AUX COMM Panel

Added:

IFF MASTER Knob:

Controls power to the IFF transponder/ interrogator unit. Positions: OFF, STBY, LOW, NORM, EMERG .

IFF MODE 4 REPLY Switch:

Control Mode 4 reply when C&I switch is in BACKUP. Positions: OFF, A, B.

IFF MODE 4 MONITOR Switch:

Controls audio feedback if a mode 4 interrogation is detected. Positions: OFF, AUDIO.

IFF MODE 4 CODE Switch:

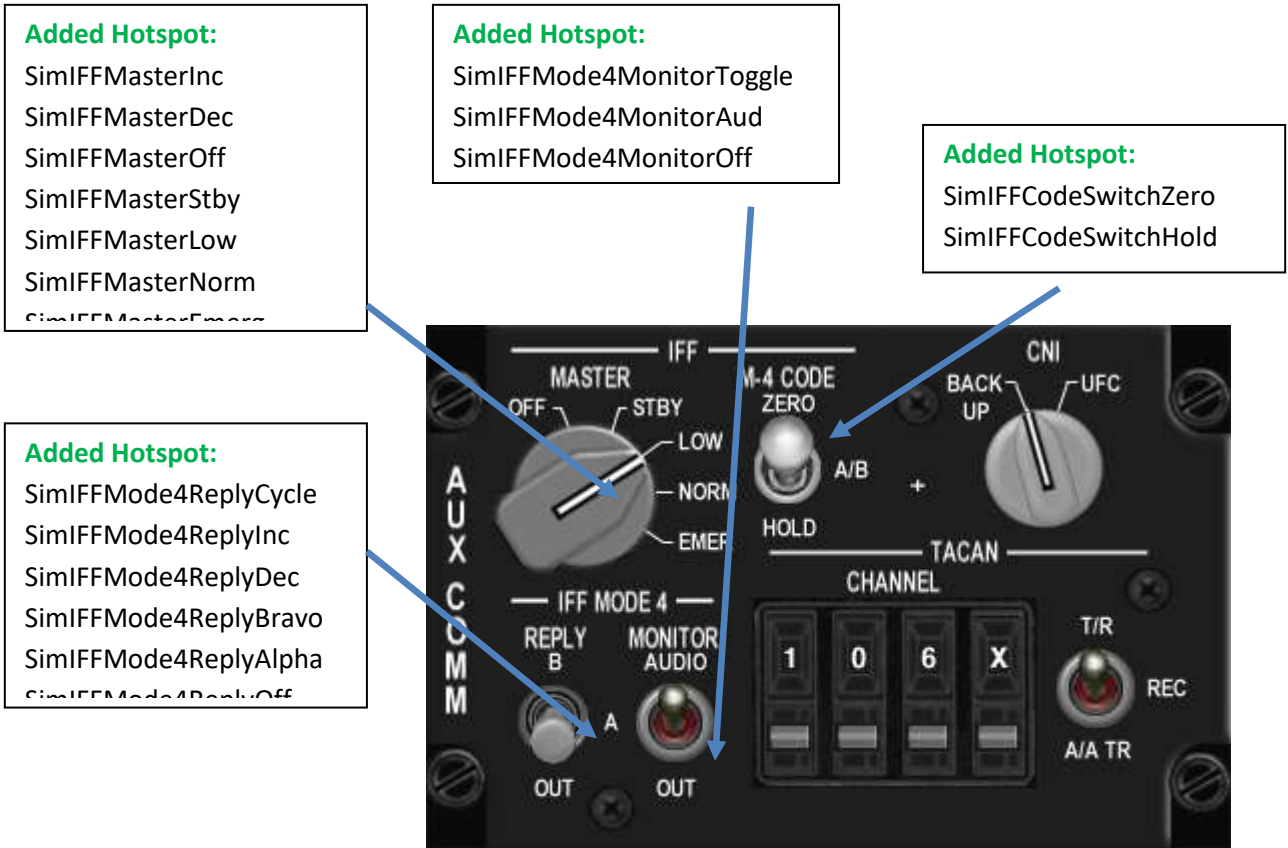
Controls Mode 4 keys. Positions: ZERO, A/B, HOLD

IFF ENABLE Switch:

On equipped jets only: Enable control of modes if the C&I switch is in backup. Positions: M1M3, OFF, M3MS

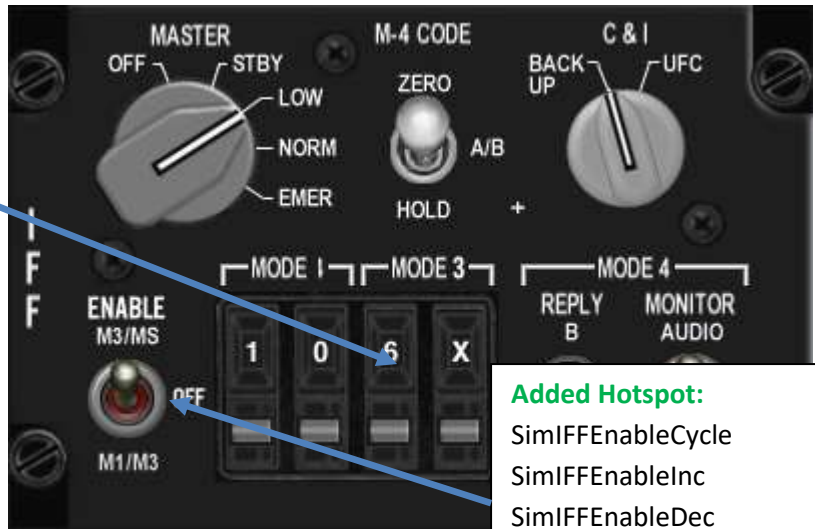
IFF Backup Digits:

On equipped jets only: Change mode 1/3 code.



Added Hotspot:

SimIFFBackupM1Digit1Inc
 SimIFFBackupM1Digit1Dec
 SimIFFBackupM1Digit2Inc
 SimIFFBackupM1Digit2Dec
 SimIFFBackupM3Digit1Inc
 SimIFFBackupM3Digit1Dec
 SimIFFBackupM3Digit2Inc
 SimIFFBackupM3Digit2Dec



Added Hotspot:

SimIFFEnableCycle
 SimIFFEnableInc
 SimIFFEnableDec
 SimIFFEnableM3MS
 SimIFFEnableOff
 SimIFFEnableM1M2

The default key assignments are:

<u>SimIFFMasterInc</u>	Shft Ctrl F8	<u>SimIFFBackupM1Digit1Inc</u>	Shft Alt F2
<u>SimIFFMasterDec</u>	Shft Ctrl F7	<u>SimIFFBackupM1Digit2Inc</u>	Shft Alt F3
<u>SimIFFCodeSwitchZero</u>	Shft Ctrl F10	<u>SimIFFBackupM3Digit1Inc</u>	Shft Alt F4
<u>SimIFFCodeSwitchHold</u>	Shft Ctrl F9	<u>SimIFFBackupM3Digit2Inc</u>	Shft Alt F5
<u>SimIFFMode4ReplyCycle</u>	Shft Ctrl F11	<u>SimIFFEnableCycle</u>	Shft Alt F1
<u>SimIFFMode4MonitorToggle</u>	Shft Ctrl F12		

Toggle Visor

Added:

SimVisorToggle

This toggles the helmet visor up / down e.g. for better readability of the HUD in bright conditions. The view is darkened a bit. The tint is configurable in 3dckpt.dat file via *visor_tint_color*.



Visor up



Visor down

The default key assignment for SimVisorToggle is **Alt V**.

Idle Detent

Added: SimThrottleIdleDetentForward, SimThrottleIdleDetentBack

Added two new callbacks for the Idle Detent: One only starts the engine, one only shuts it down
Complements the existing SimThrottleIdleDetent callback, which can do both depending on engine state,
but as such, can be ambiguous.

Note: These callbacks only work if the analog axis mapped to the engine is below 10% of its value, or if the throttle is mapped on the keyboard. For 2-engine jets, both engines are concerned if they met the criteria above.

Both callbacks have no default keys set.

Nightvision On / Off

Added:

SimNVGModeOn, SimNVGModeOff

Added two new dedicated on/off callbacks to supplement the existing toggle ToggleNVGMode.

Both callbacks have no default keys set.

Smoke On / Off

Added:

SimSmokeOn, SimSmokeOff

Added two new dedicated on/off callbacks to supplement the existing toggle ToggleSmoke.

Both callbacks have no default keys set.

15.4 Changes For 4.33 Update 1

FLAP Switch (F-18)

To complement the F-18 FM overhaul a new switch function has been implemented.

AUTO

With weight off wheels, leading and trailing edge flaps are scheduled as a function of AOA. With WOW, leading and trailing edge flaps and aileron droop are set to 0°.

HALF

Below 250 knots, leading edge flaps are scheduled as a function of AOA. Trailing edge flaps and aileron droop are scheduled as a function of airspeed to a maximum of 30° at approach airspeeds. Above 250 knots, the flaps operate in the auto flap up mode and the amber FLAPS light comes on. On the ground, the leading edge flaps are set to 12°. The trailing edge flaps and aileron droop are set to 30°. With the wing unlocked, aileron droop is set to 0°.

FULL

Below 250 knots, leading edge flaps are scheduled as a function of AOA. Trailing edge flaps and aileron droop are scheduled as a function of airspeed to a maximum of 45° flaps and 42° aileron droop at approach airspeeds. Above 250 knots, the flaps operate in the auto flaps up mode and the amber FLAPS light comes on. On the ground, the leading-edge flaps are set to 12°. The trailing edge flaps are set to 43° to 45° and aileron droop to 42°. With the wings unlocked, aileron droop is set to 0°.

The new Flap switch functions are located in section [6.01 Other Cockpit Callbacks](#) in the key files and editor.



Added Hotspot:

- SimTEFCMDInc
- SimTEFCMDDec
- SimTEFCMDAuto
- SimTEFCMDHalf
- SimTEFCMDFull

The default key assignment for [SimTEFCMDInc](#) is **Shft F12** and for [SimTEFCMDDec](#) **Shft F11**.

15.5 Conversion From 4.32 To 4.33

Left Console, FLT CONTROL Panel

Redesignated:

SimDigitalBUP is now a toggle between BACKUP and OFF (was dedicated BACKUP before)

Added:

SimDigitalBUPBackup added as dedicated BACKUP call (SimDigitalBUPOff dedicated OFF call unchanged)

Added:

SimManualFlyup added as toggle between Enable and Disable (SimManualFlyupEnable/Disable dedicated states unchanged)



Left Console, AUX COMM Panel

Added:

SimXBandAuxComDigit and SimYBandAuxComDigit added as dedicated X/Y tacan selectors
(SimCycleBandAuxComDigit toggle unchanged)



Left Console, ECM Panel

Added:

SimEcmPower added as toggle between On and Off (SimEcmPowerOn/Off dedicated states unchanged)



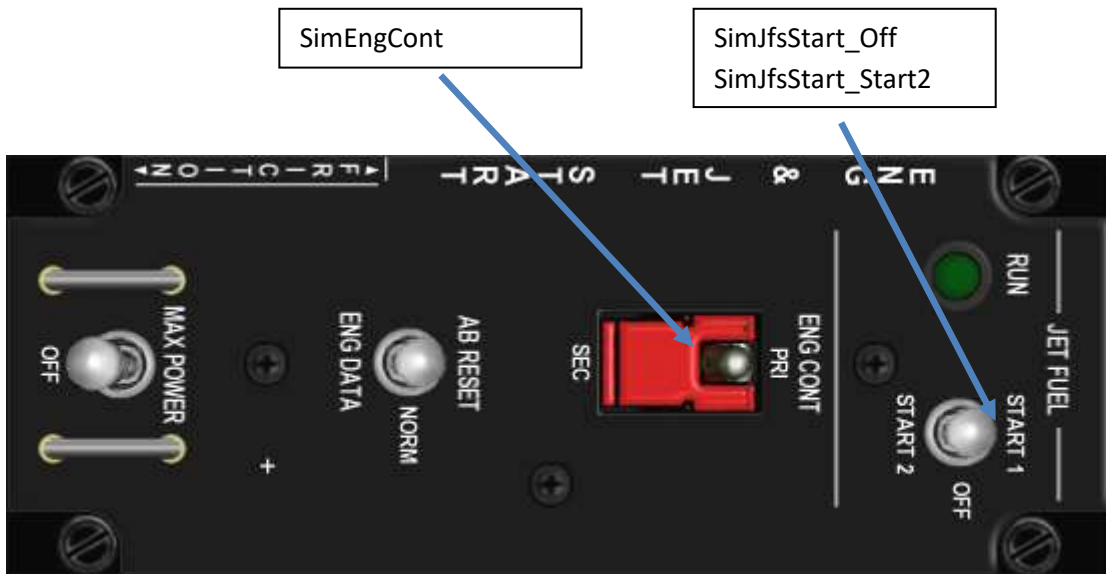
Left Console, ENG & JET START Panel

Added:

SimJfsStart_Off and SimJfsStart_Start2 added as full state callbacks for the JFS switch (START2 and OFF positions) to supplement SimJfsStart toggle.

Added:

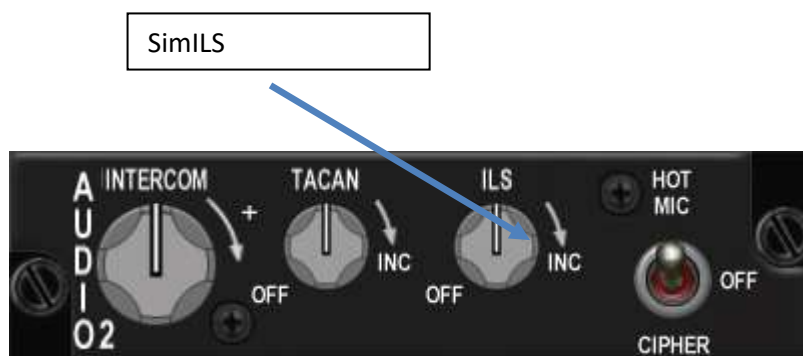
SimEngCont added as toggle between Pri and Sec (SimEngContPri/Sec dedicated states unchanged)



Left Console, AUDIO2 Panel

Added:

SimILS added as toggle between On and Off (SimILSON/Off dedicated states unchanged)



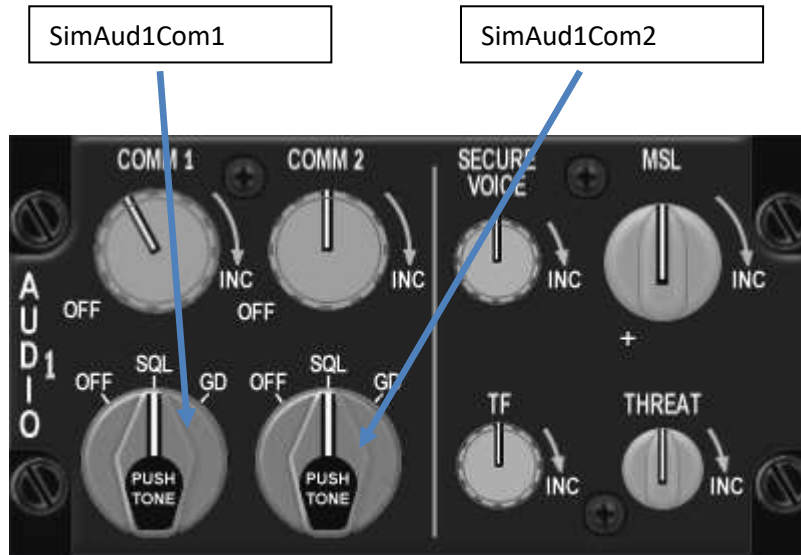
Left Console, AUDIO1 Panel

Added:

SimAud1Com1 added as toggle between Sql and Gd (SimAud1Com1Sql/Gd dedicated states unchanged)

Added:

SimAud1Com2 added as toggle between Sql and Gd (SimAud1Com2Sql/Gd dedicated states unchanged)



Left Aux Console, TWA

Added:

SimRwrPowerOn and SimRwrPowerOff (dedicated positions for the SimRwrPower toggle)



Left Console, UHF Panel:

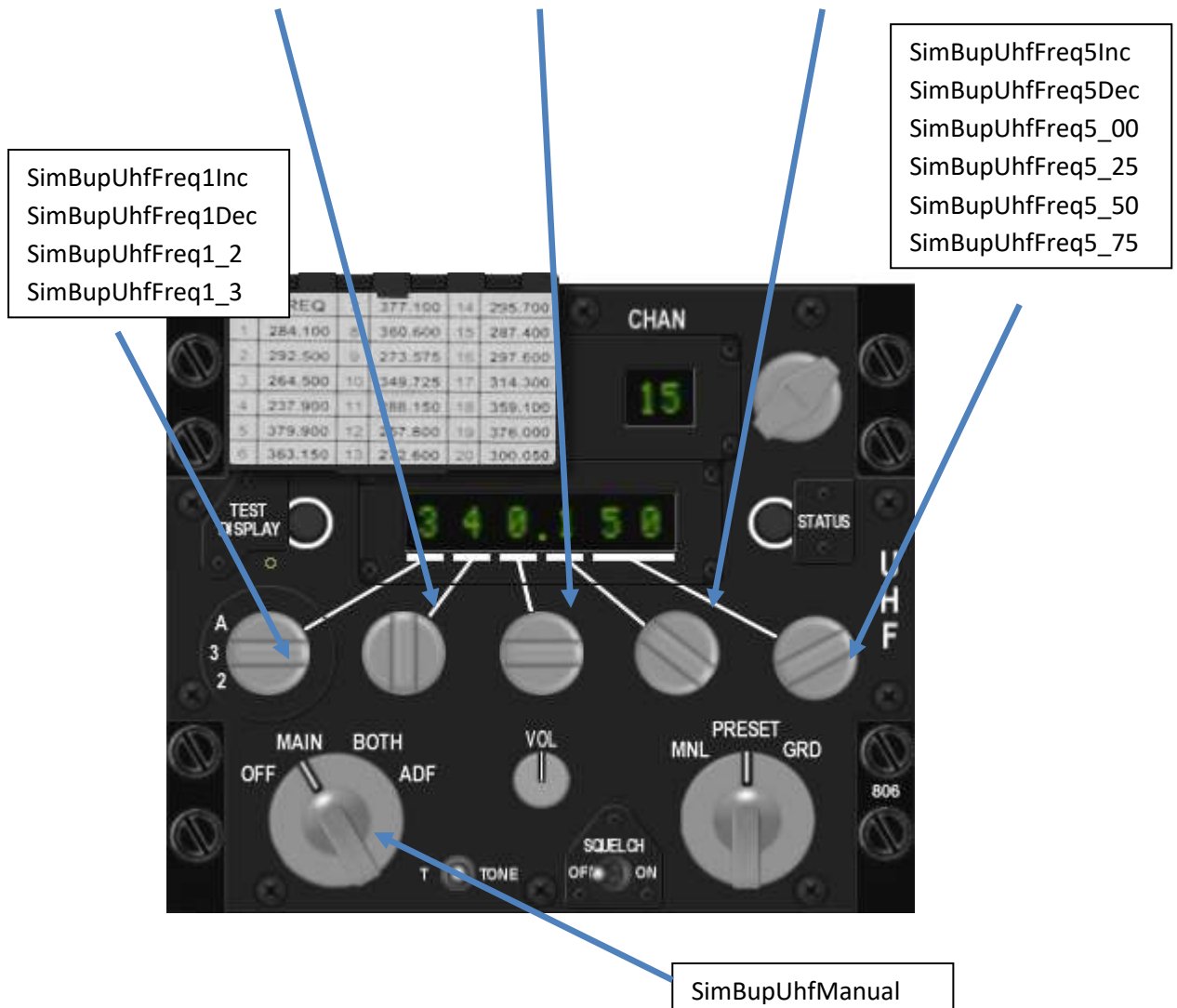
Added:

SimBupUhfManual (for the BUP Mode Knob; in addition to the existing SimBupUhfPreset and SimBupUhfGuard)

Added:

Added dedicated keystrokes for the BUP UHF manual frequency tuning and increment/decrement keystrokes (for the frequency digit knobs, left to right)

SimBupUhfFreq2Inc	SimBupUhfFreq3Inc	SimBupUhfFreq4Inc
SimBupUhfFreq2Dec	SimBupUhfFreq3Dec	SimBupUhfFreq4Dec
SimBupUhfFreq2_0	SimBupUhfFreq3_0	SimBupUhfFreq4_0
SimBupUhfFreq2_1	SimBupUhfFreq3_1	SimBupUhfFreq4_1
SimBupUhfFreq2_2	SimBupUhfFreq3_2	SimBupUhfFreq4_2
SimBupUhfFreq2_3	SimBupUhfFreq3_3	SimBupUhfFreq4_3
SimBupUhfFreq2_4	SimBupUhfFreq3_4	SimBupUhfFreq4_4
SimBupUhfFreq2_5	SimBupUhfFreq3_5	SimBupUhfFreq4_5
SimBupUhfFreq2_6	SimBupUhfFreq3_6	SimBupUhfFreq4_6
SimBupUhfFreq2_7	SimBupUhfFreq3_7	SimBupUhfFreq4_7
SimBupUhfFreq2_8	SimBupUhfFreq3_8	SimBupUhfFreq4_8
SimBupUhfFreq2_9	SimBupUhfFreq3_9	SimBupUhfFreq4_9



Left Aux Console, CMDS Panel

Redesignated:

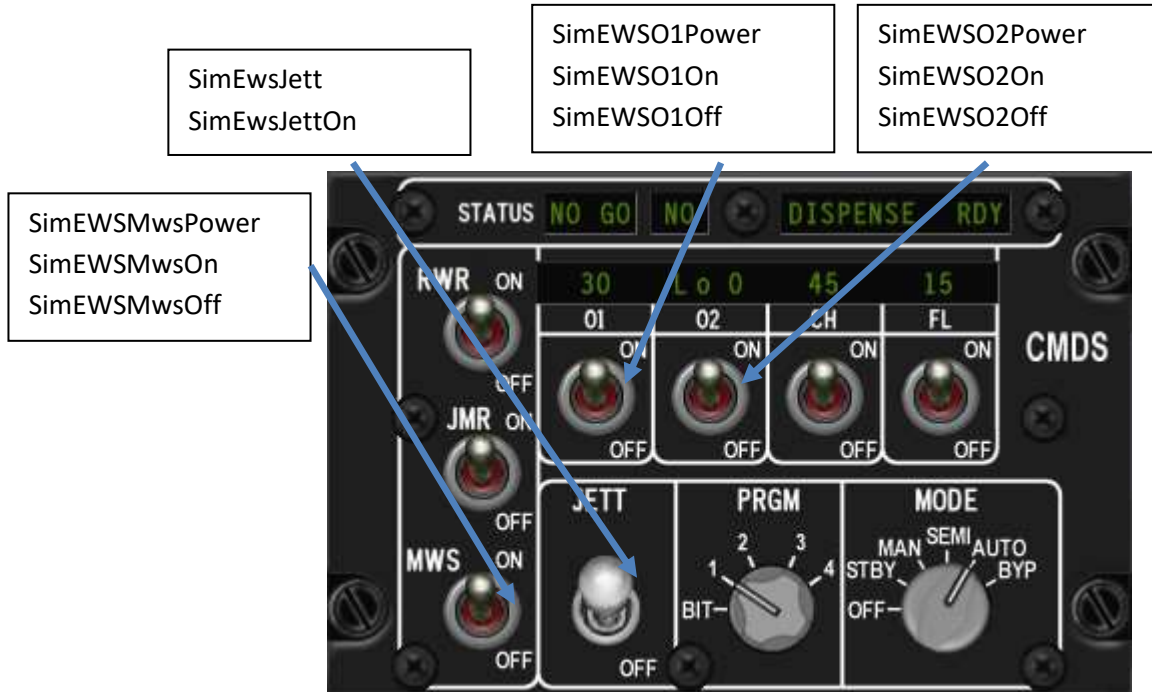
SimEwsJett is now a toggle between ON and OFF (was dedicated ON before)

Added:

New callbacks for the MWS, O1 and O2 switches. Added hotspots in Pit also.

Added:

SimEwsJettOn added as dedicated ON call (SimEwsJettOff dedicated OFF call unchanged)



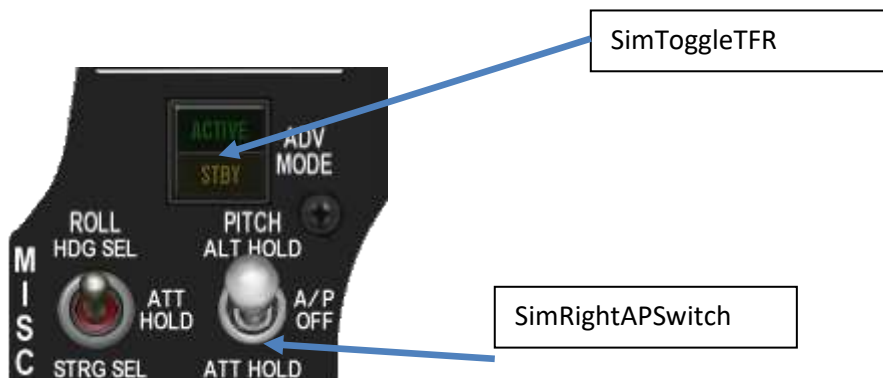
MISC Panel

Redesignated:

Enhanced the SimRightAPSwitch cycle to act as toggle for non-three-axis AP modes as well, similar to SimToggleAutopilot, which is obsolete now.

Added:

New hotspot for SimToggleTFR.



Center Console, Threat Warning Prime

Added:

two missing TWP callbacks (SimRWRLaunch & SimRWRSysTest).

Removed:

SimRWRSetNaval callback.

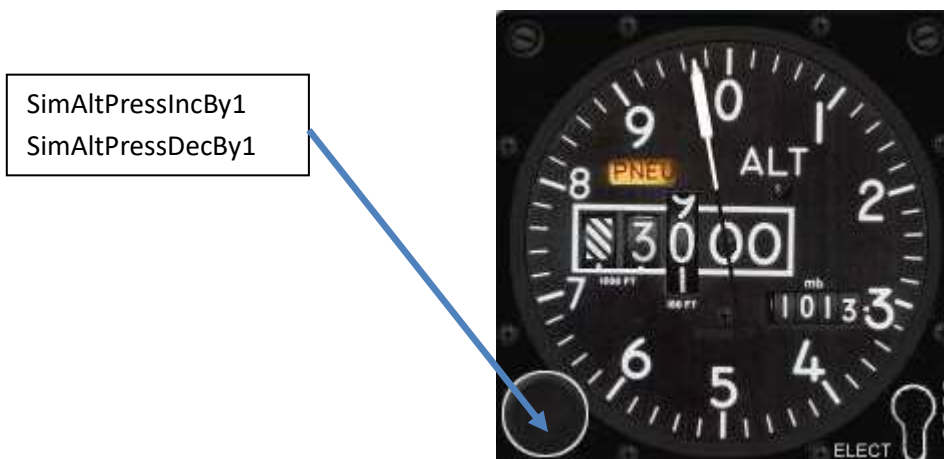
There is no such a button in RL. ALR-56M does not have SEA mode at all and naval is always visible. ALR-69 has SEA mode and its toggled by pressing SysTest button and within one second press UNKNOWN button. Then small ship icon will be displayed at bottom of RWR scope as SEA mode indication.



Center Console, Altimeter Rotary (AltPress)

Added:

SimAltPressIncBy1 and SimAltPressDecBy1 as dedicated "By1" increase / decrease callbacks for the Altimeter rotary to supplement existing SimAltPressInc / SimAltPressDec callbacks (which increase / decrease AltPress by 5).



Center Console, ICP

Added:

SimBrTWheelUp and SimBrTWheelDn to control HUD FLIR intensity.

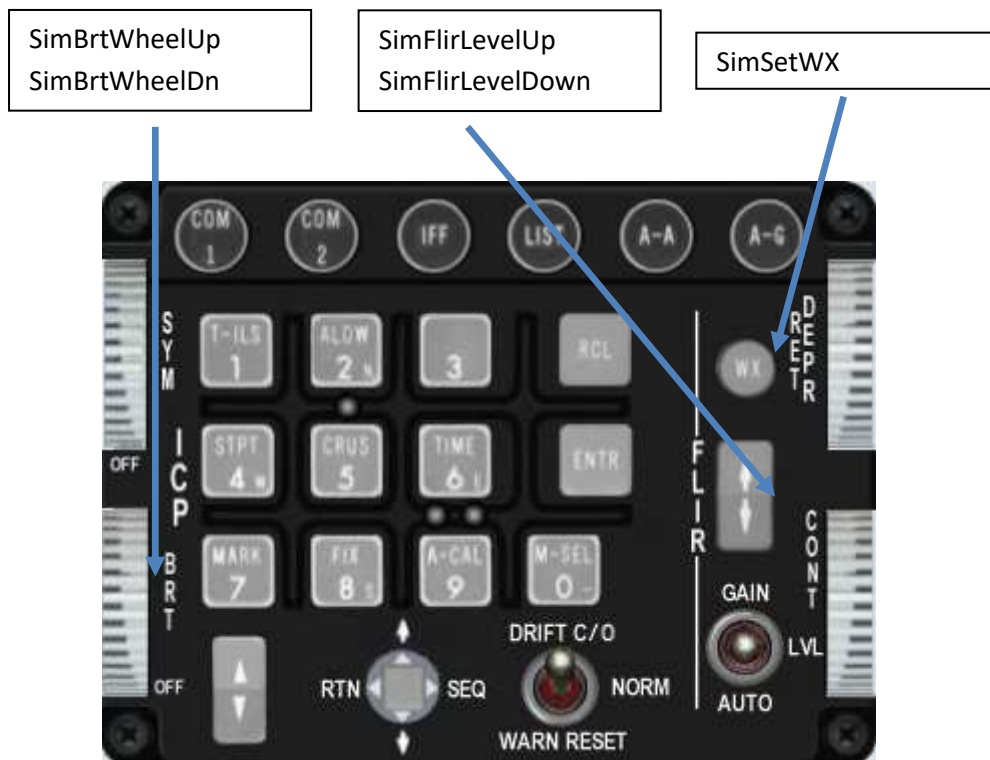
Added:

SimFlirLevelUp and SimFlirLevelDown to control FLIR level.

Added:

SimSetWX for ICP WX button.

The corresponding hotspots were added to the pit as well.



Thrust Reverser:

New callback added to trigger reverse thrust (AFTriggerReverseThrust).

Wing Fold:

Added new callbacks (AFWingFoldToggle, AFWingFoldUp, AFWingFoldDown) to be used with e.g. F-18 Hornet or E-2C .

Formation Lights:

Added new callbacks ([SimStepFormationLightsUp](#), [SimStepFormationLightsDown](#)) to control the formation lights. Currently three states (0%, 50% and 100%) are implemented for various AC.

Pilot entertainment system:

Added possibility to pause/resume (in addition to start/stop), new callback [WinAmpTogglePause](#).

Changed DCS SEQ to call [TogglePause](#) (pause/resume) instead of [TogglePlayback](#) (start/stop).

Mouse buttons:

Added keystrokes to enable/disable the mouse buttons in the 3D world. This offers the possibility to avoid unwanted mouse clicks in non-exclusive mouse capture mode, e.g. for touchscreen users. The new keystroke names are: [OTWMouseButtonsIn3dEnable](#), [OTWMouseButtonsIn3dDisable](#), [OTWMouseButtonsIn3dToggle](#).

OTWToggleHUDRendering:

Added new callback [OTWToggleHUDRendering](#), which will toggle rendering the HUD in the 3D cockpit view. The external window rendering is not affected by this.

AFTriggerCatapult:

As catapult works automatic now, [AFTriggerCatapult](#) is redesignated to release the catapult trigger (in case of malfunctions etc.) instead of triggering the catapult launch.

SimCursorStopMovement

Added new callback [SimCursorStopMovement](#), which will simply set all radar cursor input to 0. This is not a real-life functionality, but a faulty microstick in the Cougar for example *is* a common problem. Given the fact that the Cougar is not in production anymore but still rather common for BMS users this is a possible manual workaround for people that experience "stuck cursors".

15.6 Renamed Callbacks

These are Development Callbacks which are only available if new config option [g_bDevelopmentCallbacks](#) is set to 1. You have to add this line manually to the .cfg file:

OTWStepHeadingScale	->	DEV_OTWStepHeadingScale
SimRWRSetNaval	->	DEV_SimRWRSetNaval
SimTISLPower	->	DEV_SimTISLPower
SimToggleDropPattern	->	DEV_SimToggleDropPattern
OTWStepHeadingScale	->	DEV_OTWStepHeadingScale
SimRWRSetNaval	->	DEV_SimRWRSetNaval
SimTISLPower	->	DEV_SimTISLPower
SimToggleDropPattern	->	DEV_SimToggleDropPattern
SimToggleInvincible	->	DEV_SimToggleInvincible
OTWTimeOfDayStep	->	DEV_OTWTimeOfDayStep
OTWToggleGLOC	->	DEV_OTWToggleGLOC
OTWToggleLocationDisplay	->	DEV_OTWToggleLocationDisplay
OTWToggleAeroDisplay	->	DEV_OTWToggleAeroDisplay
OTWToggleWindDisplay	->	DEV_OTWToggleWindDisplay
OTWScaleDown	->	DEV_OTWScaleDown
OTWScaleUp	->	DEV_OTWScaleUp
OTWSetObjDetail	->	DEV_OTWSetObjDetail
OTWToggleClouds	->	DEV_OTWToggleClouds
OTWEnterPosition	->	DEV_OTWEnterPosition
OTWSetScale	->	DEV_OTWSetScale
OTWStateStep	->	DEV_OTWEyeFlyStateStep
SuperCruise	->	DEV_SuperCruise
SimSpeedyGonzalesUp	->	DEV_SimSpeedyGonzalesUp
SimSpeedyGonzalesDown	->	DEV_SimSpeedyGonzalesDown
SimCycleDebugLabels	->	DEV_SimCycleDebugLabels
SimSetBubbleSize	->	DEV_SimSetBubbleSize
SimRegen	->	DEV_SimRegen
KevinsFistOfGod	->	DEV_ATMRequestDivert
OTWToggleAutoScale	->	DEV_OTWToggleFullScreen
WingmanEchelon	->	WingmanEchelonRight
ElementEchelon	->	ElementEchelonRight
FlightEchelon	->	FlightEchelonRight
WingmanForm1	->	WingmanLine
ElementForm1	->	ElementLine
FlightForm1	->	FlightLine
WingmanForm2	->	WingmanEchelonLeft
ElementForm2	->	ElementEchelonLeft
FlightForm2	->	FlightEchelonLeft
WingmanForm3	->	WingmanDiamond
ElementForm3	->	ElementDiamond
FlightForm3	->	FlightDiamond

15.7 Deleted Callbacks

The following are old remains from ancient 2D-Pit times or just empty functions which do not work anymore / never have been fully implemented.

Sticky View functions:

OTW1000View	OTWHud1100View	OTWSelectHybridPitMode
OTW1200DView	OTWHud1200View	OTWToggleHybridPitMode
OTW1200HUDView	OTWHud200View	SimToggleAltView
OTW1200LView	OTWHud300View	SimToggleCockpit
OTW1200RView	OTWHud400View	SimToggleGhostMFDs
OTW1200View	OTWHud600LView	SimToggleRearView
OTW200View	OTWHud600RView	OTWSnap900
OTW300View	OTWHud800View	OTW900View
OTW400View	OTWHud900View	OTWSnap300
OTW800View		

Tilt View functions:

OTWViewDownLeftTiltDown	OTWViewRightTiltDown	OTWViewDownLeftTiltUp
OTWViewRightTiltUp	OTWViewDownRightTiltDown	OTWViewUpLeftTiltDown
OTWViewDownRightTiltUp	OTWViewUpLeftTiltUp	OTWViewDownTiltDown
OTWViewUpRightTiltDown	OTWViewDownTiltUp	OTWViewUpRightTiltUp
OTWViewLeftTiltDown	OTWViewUpTiltDown	OTWViewLeftTiltUp
OTWViewUpTiltUp		

Other functions:

BreakToggle	KneeboardTogglePage	SoundOff
SimDbuOff	SimABReset	SimAutoAVTR
SimDbuOn	SimEPUGEN	SimICPLink
SimMaxPower	SimIFFIn	SimIFFPower
SimProbeHeat	SimRadarNextTarget	SimRadarPrevTarget
SimStepSMSRight	SimSelectiveJettison	SimStepSMSLeft
SimTestSwitch	SimToggleChatMode	SimToggleUHFMaster
SimToggleRealisticAvionics	SimToggleRadioVolume	OTWShowTestVersion
OTWToggleScales	Cycle3DPitPanMode	Select3DPitPanMode
Select3DPitSnapGuidedMode	Select3DPitSnapAbsMode	Select3DPitSnapRelMode
Select3DPitGlanceMode	SelectSticky3dPitSnapViews	ToggleSticky3dPitSnapViews
SelectNonSticky3dPitSnapViews		

Comm functions:

ATCTaxiing	FACCheckIn	FACRequestMark
ATCReadyToGo	FACWilco	FACRequestTarget
ATCRotate	FACUnable	FACRequestBDA
ATCGearUp	FACReady	FACRequestLocation

ATCGearDown
ATCBrake
ElementForm4

FACIn
FACOut
FlightForm4

FACRequestTACAN
WingmanForm4

Dev functions:

DEV_OTWToggleFullScreen
DEV_OTWStepHeadingScale
DEV_SimRWRSetNaval

DEV_SimToggleDropPattern
DEV_SimToggleInvincible
DEV_OTWToggleGLOC

DEV_OTWSetObjDetail
DEV_OTWToggleClouds
DEV_SimSetBubbleSize

15.8 Outdated Callbacks

The following callbacks are still in the code and do work properly. Nonetheless, they will be for sure removed from the code in the future as we have newer callbacks now. So take this as a serious advice. It's (on the other hand) just fair to remind you about further code changes in that regard. So be warned. Take the opportunity to get rid of these old functions.

SimDesignate	->	Same as TMS up
SimDropTrack	->	Same as TMS down
SimACMBoresight	->	Same as TMS up, also controlled via MFD
SimACMVertical	->	Same as TMS right, then down, also controlled via MFD
SimACMSlew	->	Invoked by cursor, also controlled via MFD
SimACM30x20	->	Same as TMS right, also controlled via MFD
OTWStepMFD1	->	Same as DMS left
OTWStepMFD2	->	Same as DMS right
SimECMStandby	->	Same as CMS right
SimECMConsent	->	Same as CMS down
SimDropProgrammed	->	Same as CMS up
SimNextAAWeapon	->	Same as SimMissileStep, also controlled via MFD
SimTrigger	->	Same as second trigger detent
SimMaverickFOVStep	->	Same as Pinky switch, also controlled via MFD
SimRadarFOVStep	->	Same as Pinky switch, also controlled via MFD
SimSOIFOVStep	->	Same as Pinky switch, also controlled via MFD
SimFCCSubModeStep	->	Same as DMS up / down
SimCmsLeft	->	Same as CMS left
SimNextNavMode	->	Same as ICP 1 (SimICPTILS)
SimRadarStandby	->	Replaced by RF switch, also controlled via MFD
OTWTogglePitchLadder	->	Replaced by FPM Switch - Cycle (SimHUDFPM)
SimToggleExtLights	->	No such function in Pit, see EXT LIGHTNING panel section
SimObogsBit	->	Same as SimOBOGSBit
SimToggleAutopilot	->	Function integrated in SimRightAPSwitch

Note: Since 4.34 all of the above callbacks have been deleted from the code. Again: You have been warned 😊

PART 3

FOR THE THIRD PARTY DEVELOPER

16. Training Scripts

This chapter describes how to create basics training scripts. Falcon BMS includes the ability to create interactive training missions. This is accomplished by means of "training scripts." These scripts are simple text files (either with a .txt or a .run extension) that are loaded with the training mission and contain the interactive training instructions. Training scripts are written in a simple programming language that is specifically designed to make writing scripts easy.

16.1 Script Files:

BMS features two different kinds of script files: Manual & automatic scripts. You can use both script-types at the same time. There is one very important difference between the ".run" and ".txt" scripts: only the ".txt" scripts will work properly with regards to user input, input blocking etc... The ".run" scripts will not execute "WaitInput" or similar. The reason here is that ".run" was not meant to be used for "training", but for "automatic setup" (without user interaction or feedback) of the A/C after mission start. You can execute commands via SimCommand etc... but you can not evaluate whether the user did press a button etc.. this only works in the ".txt" scripts.

Use the ".run" skripts to setup the A/C properly for mission start, even if the user chooses not to use the manual ".txt" training skript, which can contain additional things like tutorials, additional hints and such. The difference between manual and automatic scripts is as follows:

16.1.1 Manual Scripts:

Manual scripts have to be started by the user by checking the "Enable Training Script" checkbox which becomes available in the bottom right corner of the UI once a training mission is selected. The default is OFF. If the checkbox isn't activated, the script will not be executed. Use manual scripts if you want to let the user decide, if he wants to have training scripts activated or not.



- Manual scripts have a **.txt** extension.
- They work just with **.trn** missions.
- **All** functions are supported.

16.1.2 Automatic Scripts:

With 4.33 a new kind of training scripts is introduced. These kind of scripts run automatically. So the user will not have a possibility to enable or disable them. The automatic scripts are useful for training missions you want to make sure they are started with pre-defined conditions (e.g. specific cockpit setup).

- Automatic scripts have a **.run** extension.
- They work with **.trn** and **.tac** missions.
- Function support is limited (no user interaction functions available, e.g. WaitInput)

***Note:** If the mission has an automatic but no manual training script file assigned, no check box will be shown. Check Boxes will only appear, if a manual script is assigned to the mission file.*

16.1.3 How to assign script files to a mission file

Script files must have the same file name as the mission they are created for, except the file extension, which is either **.txt** or **.run**.

Example:

YourTrainingMission. trn	=	Training Mission File
YourTrainingMission. txt	=	Manually started Training Script
YourTrainingMission. run	=	Automatically started Training Script

In BMS most of the training missions have training scripts. These are saved in the folder

"...\Data\Campaign"

Training scripts do not work with Campaign missions. They are meant for training purposes only. However, the scripts do also work in Tactical Engagement missions. When the training mission or TE loads, Falcon will check for the existence of a training script file to go along with the mission.

For editing purposes a simple text editor can be used for that task.

16.2 Basic Concepts

16.2.1 Variables:

The script language has no concept of variables. All functions take actual values. For the purpose of interactive training, this lack of variables in the language should not be a significant limitation.

16.2.2 Script vs. Falcon Process:

The language is designed to never block the main Falcon process. This is accomplished by only running one command in the language per frame. One effect of this is that bigger scripts do NOT slow the simulation down. This is because, regardless of the size, only a single command executes per frame.

The exception to above is that certain functions can execute in "parallel" with others. For example, there are two print commands. Both "Print" and "WaitPrint" put a string on the screen for a set duration. The difference between the two is that "Print" immediately moves onto the next command on the next frame. "WaitPrint" on the other hand, pauses the execution of the script until the time expires.

16.2.3 Boolean Functions:

The language has a unique and very limited form of return values. Specifically, most functions when they run have a return value of true. Certain functions, however, return false under certain circumstances (usually when they time out). The only way that return values can be used is by the use of them if and while type functions. These functions simply check the return value - either true or false - of the last executed function to determine whether to execute. This is ONLY use of return values.

16.2.4 Code Sections:

The code in a script can be divided into "sections". These sections are defined by simply putting the name of the section on a line by itself. There are several ways to invoke such sections. This will be described later on.

16.2.5 Code Lines:

The action of the language is composed entirely of functions. There are no operators. So, each line of a script can contains one of the following things:

- A function name followed by arguments
- A section name
- A comment starting with //
- A blank line

16.2.6 Mission Start with Scripts:

Unlike older Falcon flavors missions using scripts either with manual scripts enabled or automatic scripts **DO NOT** start out in PAUSED mode anymore.

It is possible to use both script types at the same time.

- Automatic scripts work with both Tactical Engagement (.tac) and Training Missions (.trn).
- Manual Scripts just work with Training Missions (.trn)

16.3 Screen Coordinates / Cursor Position:

The screen is divided in coordinates. Each coordinate is composed of a set of two floating point numbers, which represent the x- and y-axis. The first number represents the x-axis (left-right) the second one the y-axis (up-down). Both axis values are divided by a blank character (x.x y.y).

Example:

```
SetCursor 0.0 0.95
```

The first number (0.0) is the position on the x-axis, the second (0.95) is the position on the y-axis.

Positive numbers move the cursor right (x-axis) or up (y-axis). Negative numbers move the cursor left (x-axis) or down (y-axis).

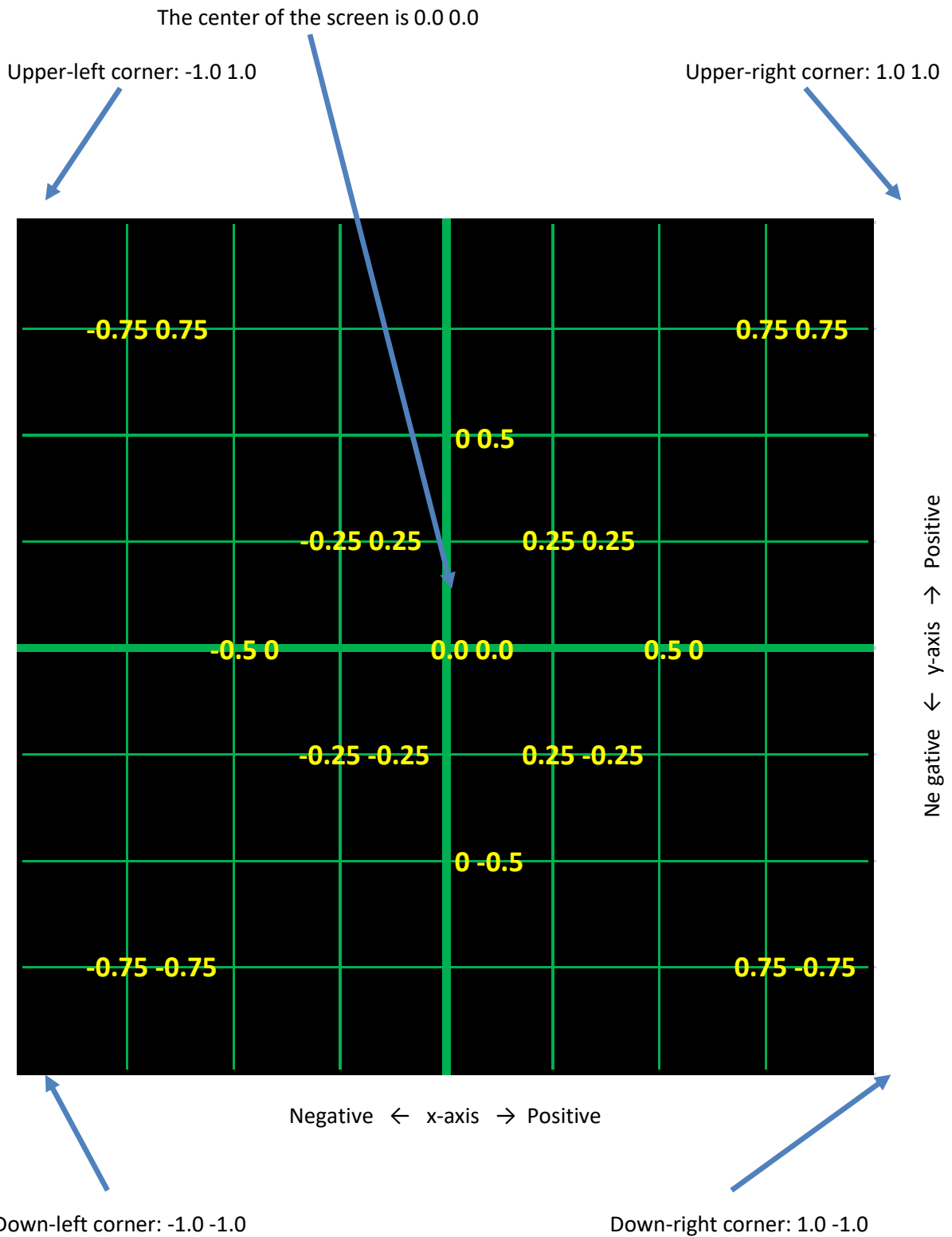
This coordinate system is completely independent of the screen resolution (1024*768, 800*600...). This ensures, every pilot, no matter of his screen setup, sees the very same result.

Whenever you want to type or draw something on the screen, you can move your cursor to a specific position, where the action should occur. If no cursor position is defined, the output will appear on the centre of the screen (0.0 0.0).

Note: Instead of writing 0.0 you can also write just 0.

Screen coordinates:

This picture illustrates the coordinate system:



The center of the screen (0.0 0.0) is the default cursor position.

16.4 Functions and Arguments:

Each function can have one or more arguments (function <argument1> <argument2> ...). Functions and arguments are separated by a blank character. The following is a list of all types of arguments accepted by the system.

<time>	A floating point number. E.g. 1.0 is 1 second, 0.001 is 1 millisecond.
<integer>	A whole number like 1 or 2.
<float>	A floating point number, e.g. 1.0 / -0.25 / 0.5 / -0.05 etc.
<string>	A string of characters surrounded with quote marks. For example "Hello, World".
<hex>	A hexadecimal number which sets a color, like 0xffffffff (=white).
<command>	The name of command as found in the keystrokes files. An example is SimTogglePaused.
<section>	The name of a section in the script.

Example:

FunctionName _<Argument1> _<Argument2> _...

16.5 Function Reference

16.5.1 Cursor Positioning Functions:

SetCursor

Syntax:

SetCursor <float (x)> <float (y)>

Description:

Sets the cursor position to the location specified in <float (x)> and <float (y)>. The range of the coordinate system is from -1.0 to 1.0. For example x = -1.0 is the left of the screen, 0.0 is the center and 1.0 is the right side of the screen. This cursor location is used for all print and draw functions. The cursor position stays the same all the time until it is relocated to another screen location by using either SetCursor or MoveCursor.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0.0 0.0	Min Value	-1 (x & y)	Max Value	1 (x & y)
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint, Oval				

Example: **SetCursor** 0.0 0.95

MoveCursor

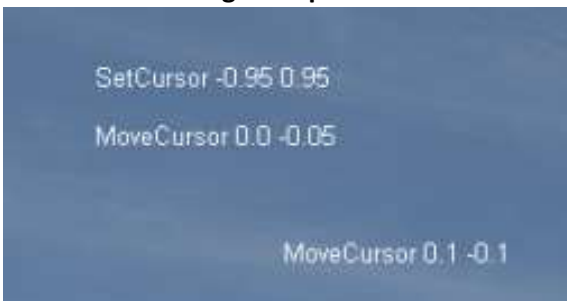
Syntax: **MoveCursor** <float (x)> <float (y)>

Description: Moves the cursor from its current position by x and y amount. The range of the coordinate system is from -1.0 to 1.0. For example x = -1.0 is the left of the screen, 0.0 is the center and 1.0 is the right side of the screen. This cursor location is used for all print and draw functions. The cursor position stays the same all the time until it is relocated to another screen location by using either SetCursor or MoveCursor.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	-1 (x & y)	Max Value	1 (x & y)
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint, Oval				

Example: **MoveCursor 0.0 -0.05**

Cursor Positioning Example:



As seen in the screenshot first the SetCursor function sets the cursor position (1) to the user defined spot on the screen (in this case the top left corner of the screen).

The cursor is then moved slightly to the bottom (2) and after that a bit further to the bottom right (3).

16.5.2 Output Functions:

Print:

Syntax: **Print** <time> <string>

Description: Prints a string to the screen using the current cursor settings for the duration of <time>. Execution moves immediately onto the next command while leaving the message printed for the duration.

Note: If you have not defined the location on the screen (see **SetCursor**), the message appears in the centre of the screen (0.0 0.0).

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **Print 20 "Hello World"**

```

...
Code Line
Print 20 "Hello World"
Code Line
...
    
```

"Hello World" is displayed for 20 seconds while the script continuous.



WaitPrint

Syntax:

WaitPrint <time> <string>

Description:

Functions identically to "Print" except that the execution does not advance to the next command until <time> has expired.

Pauses Script	✓	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

WaitPrint 40 "Hello World" "Hello World" is displayed for 40 seconds AND the script pauses for that time until it continuous.

```

...
Code Line
WaitPrint 40 "Hello World"
Script pauses 40 sec.
Code Line
...

```

Oval

Syntax:

Oval <time> <float (radius x)> <float (radius y) optional>

Description:

Draws an oval on the screen for <time> duration at the current position of the cursor. The size of the oval is specified by the two arguments <float (x)> and <float (y)>. If only one argument is supplied, then a circle is drawn with that radius specified by <float (x)>. The cursor position is at the bottom of the object (marked by the green x), not in its center.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	-1 (x & y)	Max Value	1 (x & y)
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

Oval 40 0.05 (circle)

Oval 40 0.1 0.05 (oval)



Line

Syntax: **Line** <time> <float (x1)> <float (y1)> <float (x2)> <float (y2)>

Description: Draws a line for <time> duration from x1, y1 to x2, y2 coordinates. This function is not dependent from the current cursor location.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	-1 (x & y)	Max Value	1 (x & y)
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **Line** 40 0.55 -0.10 0.75 -0.50



16.5.3 Layout Functions

SetColor

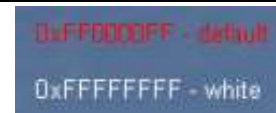
Syntax: **SetColor** <hex>

Description: Sets the current cursor color to the color specified by <hex>. This cursor color is used for all print and draw functions. The setting affects all following draw & text functions until another value is defined either using SetColor, SetFontColor or SetDrawColor.

Note: See "Appendix – Color Code Examples" for some more colors and hints.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0xFF0000FF	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint, Line, Oval (All Text & Draw Functions)				

Example: **SetColor** 0xFFFFFFFF



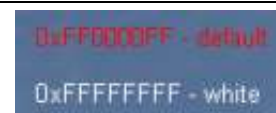
SetFontColor

Syntax: **SetFontColor** <hex>

Description: Identical to SetColor but affects just text functions. The setting affects all following text functions until another color is defined.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0xFF0000FF	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint (Text Functions Only)				

Example: **SetFontColor** 0xFFFFFFFF



Note: See "Appendix – Color Code Examples" for some more colors and hints.

SetDrawColor

Syntax: **SetDrawColor** <hex>

Description: Identical to SetColor but affects just draw functions. The setting affects all following draw functions until another color is defined.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0xFF0000FF	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Line, Oval (Draw Functions Only)				

Example: **SetDrawColor** 0xFFFFFFFF



Note: See “Appendix – Color Code Examples” for some more colors and hints.

SetFlash

Syntax: **SetFlash** <Hex>

Description: Sets the rate of flashing that should occur for all text and draw functions. "SetFlash 0" disables flashing. The higher the number (in milliseconds) behind 0x is, the slower the flashing interval is (e.g. 0x100 = 100 ms = fast flashing / 0x900 = 900 ms = slow flashing). The setting affects all following draw & text functions until another value is defined.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0	Min Value	0x1	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint, Line, Oval (All Text & Draw Functions)				

Example: **SetFlash** 0x100

SetFontFlash

Syntax: **SetFontFlash** <Hex>

Description: Same as SetFlash, but for text functions only. The setting affects all following text functions until another value is defined.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0	Min Value	0x1	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint (Text Functions Only)				

Example: **SetFontFlash** 0x100

SetDrawFlash

Syntax: **SetDrawFlash** <Hex>

Description: Same as SetFlash, but for draw functions only. The setting affects all following draw functions until another value is defined.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	0	Min Value	0x1	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Line, Oval (Draw Functions Only)				

Example: **SetDrawFlash** 0x100

SetFont

Syntax: **SetFont** <integer>

Description: Sets the font size specified by <integer>. This setting is used for all following print functions until another value is defined. The following values are valid:

- 0 = default font size, bold
- 1 = bigger font size, bold
- 2 = default font size



Pauses Script	X	Stops Script	X	Return Value	X
Default Value	0	Min Value	0	Max Value	2
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint (Text Functions Only)				

Example: **SetFont** 1

SetLineWidth

Syntax: **SetLineWidth** <float>

Description: This sets the line width defined as a floating point number from 0 to 1. Default = 0.005. This setting is used for all following draw functions until another value is defined.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	0.005	Min Value	0.0	Max Value	1.0
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Line, Oval (Draw Functions Only)				

Example: **SetLineWidth** 0.01

Left Line: Default width 0.005

Right Line: Set to width 0.01



SetFontBGColor

Syntax:

SetFontBGColor <hex>

Description:

You can set the background color for text functions. The default is set to transparent. It affects all following text functions until another value is defined. To be used together with SetTextBoxed.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0x00000000	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint (Text Functions Only)				

Example:

SetFontBGColor 0xFFFFFFFF

SetTextBoxed

Syntax:

SetTextBoxed <Integer>

Description:

Sets the type of text boxing that should apply to print statements. Valid values are 0 & 2.

- 0 = no text boxing
- 2 = text is boxed by lines



This function affects all print functions until another value is defined. You have to define a background color using SetFontBGColor first.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	0	Min Value	0	Max Value	2
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint (Text Functions Only)				

Example:

SetTextBoxed 2

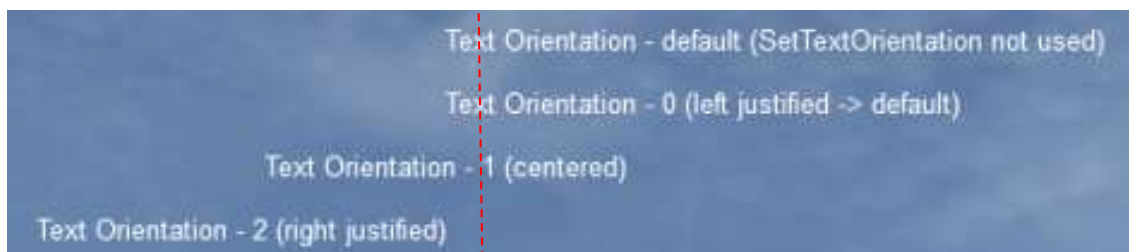
SetTextOrientation

Syntax:

SetTextOrientation <Integer>

Description:

Sets the orientation of the text relative to the cursor position. Acceptable values are 0, 1 and 2. 0 = left justified text, 1 = centered text, 2 = right justified text. This function affects all following print functions until another value is defined.



The red dotted line represents the horizontal alignment of the cursor position.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	0	Min Value	0	Max Value	2
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint (Text Functions Only)				

Example: `SetTextOrientation 2`

Clear

Syntax:

`Clear`

Description:

Immediately removes all drawn elements from the screen (Such as those created by "Print" "Line" and "Oval" statements).

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint, Line, Oval (All Text & Draw Functions)				

Example: `Clear`

ClearLast

Syntax:

`ClearLast <Integer - optional>`

Description:

If no argument is provided, the last created drawn element will be removed from the screen (Such as those created by "Print" "Line" and "Oval" statements). If a number is specified, then the nth last element will be removed. For example, "ClearLast 2" will remove the second to last drawn element from the screen.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	Print, WaitPrint, Line, Oval (All Text & Draw Functions)				

Example: `ClearLast`

16.5.4 User Interaction Functions:

WaitInput

Syntax:

WaitInput <time> <command>

Description:

Pauses execution of the script until the command specified is executed by the user. <time> is the amount of time to wait for the input. If the function times out, it returns false. If the user executes the command within the time limit, it returns true. This function accepts an unlimited number of commands as arguments. In this case the function returns true, if either of them is executed by the user. The function accepts either keystroke or DX inputs.

Pauses Script	✓	Stops Script	✗	Return Value	true or false
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✗	TRN	(.run) ✗	(.txt) ✓
Affected Functions	✗				

Example:

WaitInput 111 SimPickle SimTriggerSecondDetent

If the user fails to execute SimTriggerSecondDetent or SimPickle the function times out and returns False. The script continues after 111 seconds.

If he executes e.g. SimPickle after e.g. 20 seconds, the function returns True and the script continues after 20 seconds.

WaitMouse

Syntax:

WaitMouse <time> <float (target x)> <float (target y)> <float (distance)>

Description:

Pauses execution of the script for <time> duration or until the user moves the mouse cursor within <float (distance)> of the screen coordinates of <float (target x)> and <float (target y)>. If the function times out, it returns false.

Pauses Script	✓	Stops Script	✗	Return Value	true or false
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

WaitMouse 40 -0.34 0.75 0.05



*Note: Please note that you do not have any visual cues where the user should place his mouse cursor. So it recommended using this function together with **SetCursor** and **Oval** functions. Additionally the hotspot is fixed on the screen. When you pan your POV the hotspot moves as well.*

Block

Syntax: **Block** <command >

Description: Blocks the system from recognizing any commands listed as arguments. This function accepts an unlimited number of commands as arguments. If no arguments are provided, all commands are blocked.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✗	TRN	(.run) ✗	(.txt) ✓
Affected Functions	✗				

Example: **Block** SimTriggerSecondDetent

Allow

Syntax: **Allow** <command >

Description: Allows the system to recognize any commands listed as arguments. This function accepts an unlimited number of commands as arguments. If no arguments are provided, then all commands are allowed.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✗	TRN	(.run) ✗	(.txt) ✓
Affected Functions	✗				

Example: **Allow** SimTriggerSecondDetent

Note: Use the Block and Allow functions with care. E.g. if you Block everything also “SimEndFlight” (Exit Sim Menu) will be blocked. In this case the user has no chance leaving the sim other than “Alt – Tab”, “Ctrl- Alt – Del” or crashing the jet to get the exit menu. On the other hand, if you just allow e.g. SimTriggerSecondDetent as in the example above it is the same: No Exit Sim Menu available.

16.5.5 Section Functions:

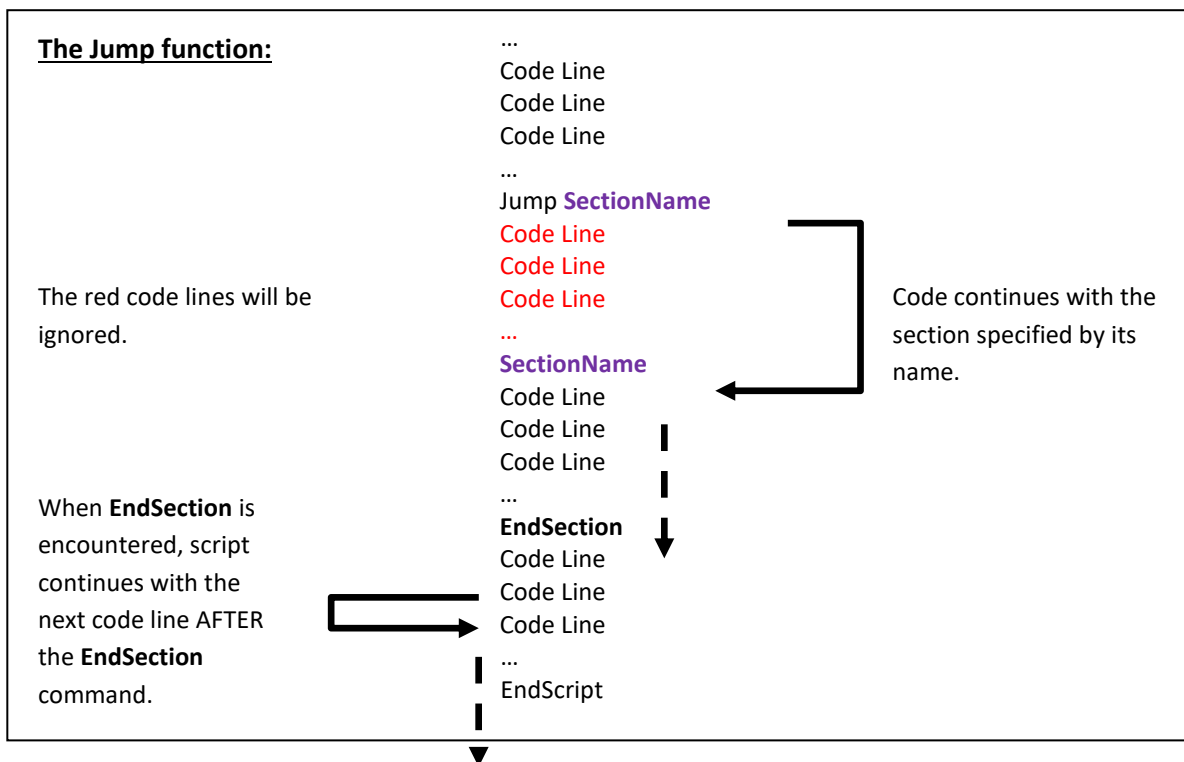
Jump

Syntax: **Jump** <section>

Description: Immediately jumps to the section identified by <section>. NOTE: This does not add anything to the call stack.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: **Jump** SectionName



JumpIf

Syntax: **JumpIf** <section>

Description: If the prior return value is true jumps immediately to the section identified by <section>. Note: This does not add anything to the call stack.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: **JumpIf** SectionName

JumpIfNot

Syntax: **JumpIfNot** <section>

Description: If the prior return value is false jumps immediately jumps to the section identified by <section>. Note: This does not add anything to the call stack.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **JumpIfNot** SectionName

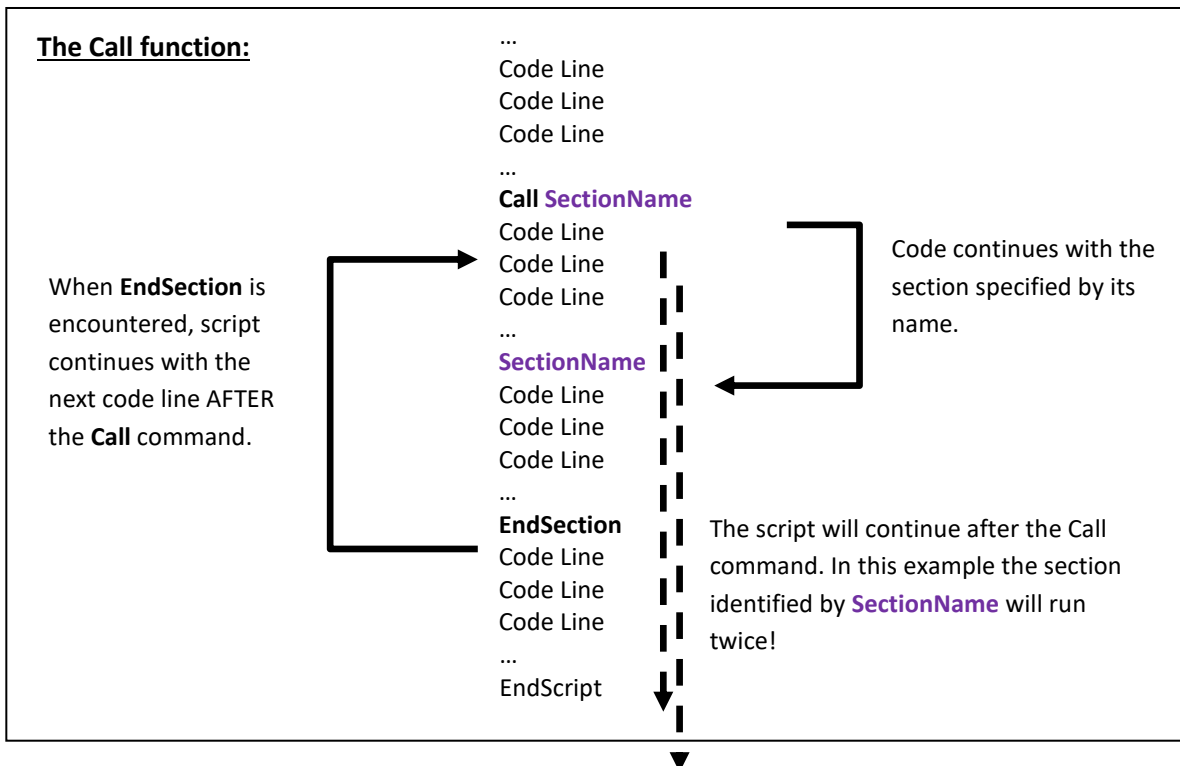
Call

Syntax: **Call** <section>

Description: Immediately jumps to the section identified by <section>. The current location is added to the call stack, and execution will return to this location when an "EndSection" function is encountered.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **Call** SectionName



CallIf

Syntax: **CallIf** <section>

Description: If the return value of the prior statement is true, then the section identified by <section> will be called. Once an "EndSection" is encountered, execution will return to the instruction after "Call".

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: **CallIf** SectionName

CallIfNot

Syntax: **CallIfNot** <section>

Description: Functions identically to "CallIf" except that a false return value triggers the call.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: **CallIfNot** SectionName

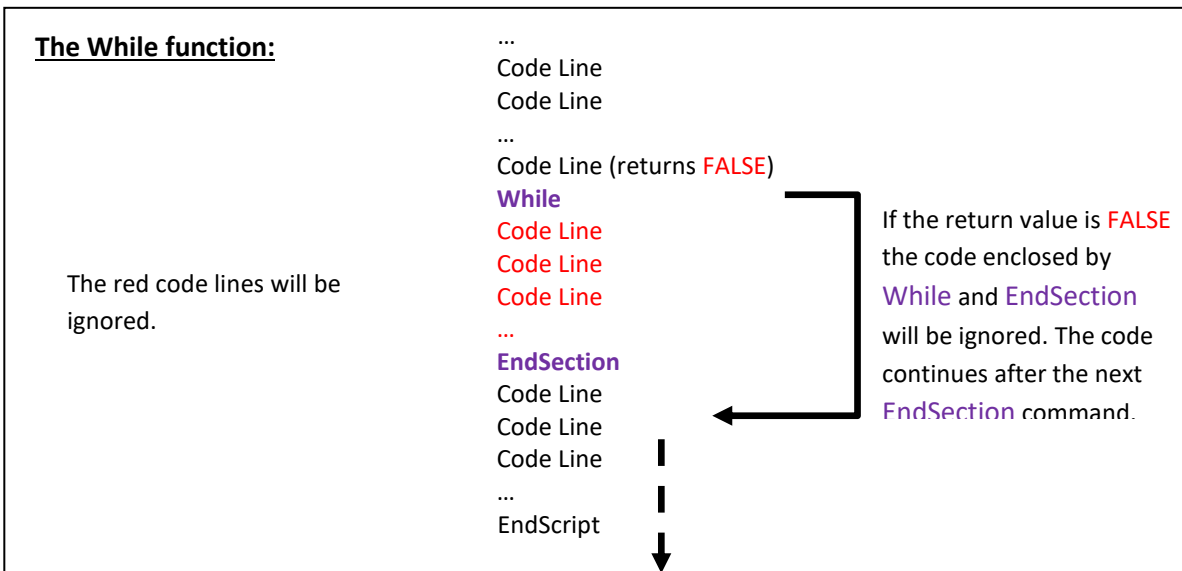
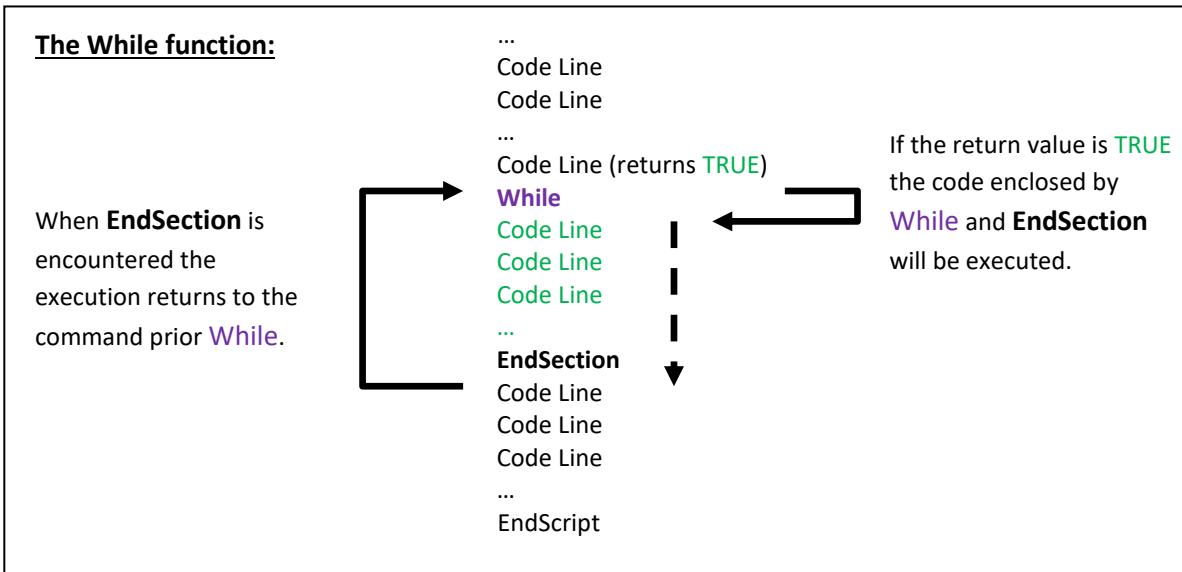
While

Syntax: **While**

Description: If the prior return value is true, runs the commands until the next "EndSection". When "EndSection" is found, execution returns to the command prior to the "While" function. If the prior return value is false, then execution moves to the command after the next "EndSection"

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: **While**



WhileNot

Syntax:

WhileNot

Description:

Operates identically to the "While" statement, with the exception that a return value of false causes the while to execute.

Note: Be careful using this with automatic scripts (.run). This could create a permanent loop which cannot be exited.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

WhileNot

EndSection

Syntax:

EndSection

Description:

This function can be used to end a section. If there is any previous function on the call stack (generally created by using the "call" function), then execution will return to the instruction immediately after the "call" when "EndSection" is encountered. If there is nothing on the call stack, this line is ignored.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

EndSection

16.5.6 Sound Functions:

Sound

Syntax:

Sound <string>

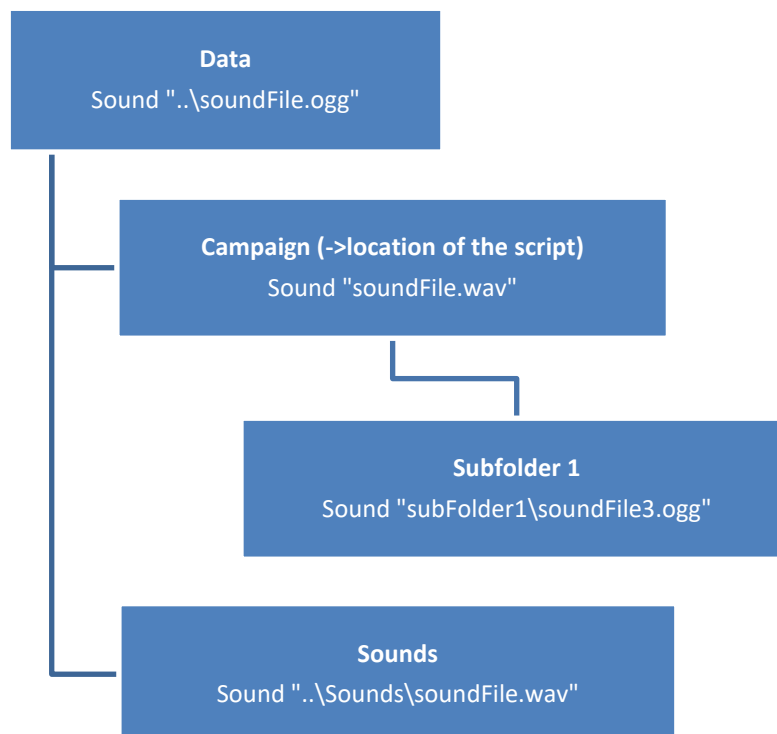
Description:

Plays the sound specified by <string>. The sound file location needs to be specified relative to the .txt/.run script file and must be enclosed by quote tags. Execution of the script continues while the sound plays. The volume of the sound is the same as the "Intercom" volume level. Valid sound files: ogg & wav.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

Sound "..\subFolder\soundFile.ogg"



WaitSound

Syntax: **WaitSound** <string>

Description: Same as Sound but the execution of the script pauses until the sound finishes playing.

Pauses Script	✓	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **WaitSound** "soundFile.wav"

WaitSoundStop

Syntax: **WaitSoundStop** <time>

Description: Pauses execution of the script for <time> duration or until the last sound executed by "Sound" stops playing. If the function times out, it **returns false**.

Pauses Script	✓	Stops Script	✗	Return Value	true or false
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **WaitSoundStop** 60

16.5.7 Cockpit Orientation Functions:

SetPanTilt

Syntax: **SetPanTilt** <float (x)> <float (y)>

Description: Sets the current 3d view to the coordinates specified by <float (x)> and <float (y)>. These coordinates are in radians and 0.0 is straight ahead in the 3d cockpit. The next table shows some examples:

Degrees	Radians
22°	0,38397243543875250692321196906749
45°	0,78539816339744830961566084581988
90°	1,5707963267948966192313216916398
180°	3,1415926535897932384626433832795
360°	6,283185307179586476925286766559
Positive numbers move the view to the left or down, negative ones to the right or up.	

Please note that this function does work only in "3d Pan View". In 2d SNAP View" it simply does nothing. Please also be advised that the user can change the POV either by using key strokes, POV hat or a headtracking device like TrackIR. If the latter one is used by the pilot there is a high chance that the desired view will not stay as it was intended by the script designer. The POV jumps to the spot specified in the script but as the user has full control his POV will change immediately.

Pauses Script	✗	Stops Script	✗	Return Value	✗
---------------	---	--------------	---	--------------	---

Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: `SetPanTilt -0.40 1.13`

MovePanTilt

Syntax: `MovePanTilt <float (x)> <float (y)>`

Description: Offsets the current 3d view from its current position by the amounts specified by <float (x)> and <float (y)>. These coordinates are in radians. See also SetPanTilt notes.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: `MovePanTilt 0.03 -0.12`

The pictures below illustrate how the functions work. For easy recognizability a red circle in the center of the screen has been added. First the POV was set from the center (0.0) to the down left. Then slightly up by MovePanTilt.



SetPanTilt -0.40 1.13



MovePanTilt 0.03 -0.12

Hillite3DButton

Syntax: `Hillite3DButton <time> <command> <integer – optional>`

Description: This function will draw two circles around a function in the cockpit (knob, button, rotary etc.) specified by <command> for the duration of <time>. Of course it must be a command which is available on the cockpit panels. Otherwise nothing will happen. The highlighted cockpit function will be shown in both Snap (2D) and Pan (3D) pit like shown in the picture. The view is not fixed on this cockpit function and if it is not visible in the current POV, it will not jump to that position. The pilot has to pan the view to that highlighted function by himself.



It is also possible to specify an optional <integer> value. If a command is present more than once in the cockpit you can specify the nth hotspot in the cockpit.

Example: The safety arm lever in the F-16 pit has three hotspots. You can now specify which hotspot shall be highlighted.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: `Hilite3DButton 10 SimICPAG`

WaitHilite3DButton

Syntax: `WaitHilite3DButton <time> <command> <integer – optional>`

Description: Same as Hilite3DButton with the exception that the script pauses for the specified amount of time and waits for the pilot to execute (either with mouse click in 3d pit or via keystroke, DX) the cockpit function specified by <command>. If the user executes the highlighted cockpit function (switch, button, knob, wheel...) within the time limit, it returns true. If time expires, the function returns false and the script continues.

Pauses Script	✓	Stops Script	X	Return Value	true or false
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	X	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: `WaitHilite3DButton 10 SimICPAG`

16.5.8 Command & Fault Functions:

SimCommand

Syntax: `SimCommand <command>`

Description: Causes <command> to be executed just as if the user had pressed the keystroke. All callbacks in the full key file are valid commands here. SimCommand issues both KEY_PRESS and KEY_RELEASE events in sequence.

Pauses Script	X	Stops Script	X	Return Value	X
Default Value	X	Min Value	X	Max Value	X
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	X				

Example: `SimCommand SimTogglePaused`

SimCommandPress

Syntax: **SimCommandPress** <command>

Description: Causes <command> to be executed (KEY_PRESS event only). To be used in conjunction with SimCommandRelease. The command will be executed until the KEY_RELEASE event with the same command is triggered.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **SimCommandPress** SimEject

SimCommandRelease

Syntax: **SimCommandRelease** <command>

Description: Causes <command> to be released (KEY_RELEASE event only). To be used in conjunction with SimCommandPress.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **SimCommandRelease** SimEject

Note: Some commands require a long input, such as Eject or Pickle. SimCommand alone won't work here as it executes the command with a press and a release event in very short time frame, thus e.g. Eject will not work as it needs to be held longer. Use SimCommandPress & SimCommandRelease with a Wait function in between.

SetFault

Syntax: **SetFault** <integer>

Description: Sets a fault specified by its individual fault id. You can find a list of possible faults in the appendix later on.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **SetFault** 147

ClearFault

Syntax: **ClearFault** <integer>

Description: This is the opposite function. You delete a specific fault.

Please note that the fault still remains in the Test page and PFL. Master Caution is still active. You simply have to hit Master Caution button once and clear the Test page.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **ClearFault 147**

16.5.9 Initial Mission Setup Functions:

SetMavCoolTime

Syntax: **SetMavCoolTime** <float>

Description: The realistic Maverick spool up time is 3 minutes. If you don't want to wait that long you can reduce the time. This function sets the spool up time as a floating point number measured in seconds. To have the realistic time you'll have to set 180.0.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	0	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **SetMavCoolTime 60.0**

SetGunAmmo

Syntax: **SetGunAmmo** <Integer>

Description: Sets the amount of cannon rounds. Initially introduced to set the gun rounds to zero at training start, but can also be used to raise the amount of ammo or even as a rearm function, as it can be used multiple times in the script. The number of rounds is not limited and can be higher as what can usually be loaded. For example, the F-16 has a maximum of 510 rounds. If you set the value to 10000, you have 10k rounds available.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	300	Min Value	0	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **SetGunAmmo 0**

SetFuel

Syntax: **SetFuel** <integer1> <integer2> <integer3> ... <integer13> <integer14>

Description: With this command you can set the fuel capacity for each fuel compartment independently. Keep in mind that the cockpit fuel gauge has a built-in random inaccuracy so the capacity shown can be off. The setting has also an impact on the total aircraft weight. Max values can be found in the aircrafts afm.dat file.

Integer	afm.dat value	Description
1	GCfuelFwdREs	Internal Forward Reserve
2	GCfuelAftREs	Internal Aft Reserve
3	GCfuelFwd1	Internal Forward 1
4	GCfuelFwd2	Internal Forward 2
5	GCfuelAft1	Internal Aft 1
6	GCfuelWingfr	Internal Wing Forward Right
7	GCfuelwingAl	Internal Wing Aft Left
8	GCfuelext-R FWD	External Tank Right Wing – Forward
9	GCfuelext-R CTR	External Tank Right Wing – Center
10	GCfuelext-R AFT	External Tank Right Wing – Aft
11	GCfuelext-L FWD	External Tank Left Wing – Forward
12	GCfuelext-L CTR	External Tank Left Wing – Center
13	GCfuelext-L AFT	External Tank Left Wing – Aft
14	GCfuelexT-C	External Tank Centerline

Note: There are no sanity checks if you have entered the correct values according to the tank capacity. Its up to you to calculate and enter the correct data here.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	0	Max Value	See afm.dat
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **SetFuel** 0 500 100 100 100 100 100 0 0 0 0 0 0

16.5.10 Time Management Functions:

Wait

Syntax: **Wait** <time>

Description: Pauses execution of the script until the specific REAL <time> expires. Pause mode does not interrupt the function. The time runs in real time until <time> has expired.

Pauses Script	✓	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **Wait** 60

WaitGameTime

Syntax: **WaitGameTime <time>**

Description: Pauses execution of the script until the specific GAME <time> expires. Pause mode will interrupt the function because the In game time is stopped. If the sim is resumed, the function starts counting again until the specified time has expired. In Freeze mode the in game clock is still running, so has no impact on this function.

Pauses Script	✓	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **WaitGameTime 60**

WaitForNoWOW

Syntax: **WaitForNoWOW**

Description: This prevents the script from continuing for an unspecified amount of time until an aircraft gets airborne (= No Weight On Wheel anymore). So it only does make sense to use this, while the aircraft is still on the ground when entering 3D. If you start the mission in the air this function has no impact at all (With the exception if you pause the script for a specific amount of time and the aircraft landed meanwhile). After Take Off, if there is no weight on wheel anymore the script continues as desired.

Pauses Script	✓	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **WaitForNoWOW**

16.5.11 Other Script Functions:

EndScript

Syntax: **EndScript**

Description: Immediately ends the script. All following functions will be ignored. Although it's a good habit to set it at the end of the script, it is not necessarily needed to do so. If there is no function left to be executed, the script simply does nothing.

Pauses Script	✗	Stops Script	✓	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example: **EndScript**

16.5.12 Boolean Functions:

If

Syntax:

If

Description:

If the return value of the last command is true, then execution moves to the next instruction. If the return value is false, then the next instruction is skipped.

Please note that the function works indeed with .run scripts but the functions which return true / false are limited. This makes it almost unusable with .run scripts.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

If

<p><u>Example 1:</u></p> <p>If the function prior If returns TRUE, the next code line will be executed.</p>	<pre> ... Code Line Code Line (returns TRUE) If Code Line (executed) Code Line ... </pre>
---	--

<p><u>Example 2:</u></p> <p>If the function prior If returns FALSE, the next code line will be not executed.</p>	<pre> ... Code Line Code Line (returns FALSE) If Code Line (not executed) Code Line ... </pre>
--	---

IfNot

Syntax:

IfNot

Description:

Functions identically to "If", except the next instruction is executed if the return value of the last statement is false.

Pauses Script	✗	Stops Script	✗	Return Value	✗
Default Value	✗	Min Value	✗	Max Value	✗
Works with	TAC .run	✓	TRN	(.run) ✓	(.txt) ✓
Affected Functions	✗				

Example:

IfNot

16.5.13 Removed Functions Reference:

The following functions have been removed from the code as these are pure 2D cockpit functions.

- SetViewCallback
- SetViewDial
- SetCursorCallback
- SetCursorDial
- WaitCallbackVisible
- WaitDialVisible

16.6 Training Script Appendix

16.6.1 Color Code Examples

A color code consists of four different values, in order: Alpha, Blue, Green and Red -> 0xAABBGGRR

So the order for RGBA is reversed in BMS. The 2 first digits are "alpha", where FF is "fully solid" and 00 is "fully transparent". Below we have some examples which can be used in Falcon. Of course a lot of other colors are possible. They can be used for text as well as drawings or background colors.

Transparent	=	0x00000000
Black	=	0xFF000000
White	=	0xFFFFFFFF
Red	=	0xFF0000FF
Green	=	0xFF00FF00
Blue	=	0xFFFF0000
Cyan	=	0xFF00FFFF
Magenta	=	0xFFFF00FF
Yellow	=	0xFF00FFFF



The hex-values (after 0x) are not case sensitive. So 0xffffffff (white) works as well. Default color is 0xFF0000FF (red).

Note: Alpha is just working for Fonts for now. If you change the Alpha for drawing functions (SetColor, SetDrawColor) they do not have any effect.

16.6.2 Fault Codes

You can set a various number of different faults the pilot has to handle with. This is especially useful for all kinds of emergency training scenarios as well as for degrading the jets capabilities for specific training purposes. On the following pages the symptoms of these faults are described.

The attentive observer might notice that a couple of fault ids are missing. These are the ones which can't be set via SetFault. However, there are also some „Fake Faults“ where the faults have no noticeable effect (e.g. Take Off & Landing messages in TEST page) or just produce an eye candy caution light illumination (e.g. HOOK without having lowered the hook).

Some faults are dependant on special conditions or even the type of aircraft. E.g. you can't set a TFR fault if the jet is not TFR-capable at all or does not carry a correspondig pod (if applicable). Some faults may produce follow-up faults later in the flight.

Faults can be cleared via the ClearFault function. Check out the “Clear” column in the tables. However, this is not true for all faults. For some it is not possible. So use them with care.

	ID	Clear	TEST Page	PFL	CAUTION PANEL	WARNING LIGHTS	CAUSE / SYMPTOMS
FLCS FAULTS	97	✓	FLCS 055 1	FLCS BIT FAIL	None	None	Failed FLCS BIT / Fault only on ground
	99	✓	FLCS 050 1	FLCS A/P FAIL	None	FLCS	Autopilot unavailable (A/P must be on to see this failure)
	100	✓	FLCS 003 1	FLCS BUS FAIL	FLCS FAULT & AVIONICS FAULT	None	FLCS System failure
	101	✓	FLCS 021 1	FLCS DUAL FAIL	None	FLCS	FLCS electronics, sensor or power failure (n/i)
	102	✓	FLCS 049 1	FLCS SNGL FAIL	FLCS FAULT	None	Single electronic or sensor failure in FLCS (only on ground)
	103	✓	FLCS 014 1	STBY GAIN	None	FLCS	Dual Air Data failure / FLCS in Standby Gains
	104	✓	FLCS 013 1	FLCS ADC FAIL	FLCS FAULT	None	Air Data Input Signal failure, second ADC fail triggers STBY GAIN
	105	✓	FLCS 071 1	FLCS MUX DEGR	FLCS FAULT	None	FLCC MUX interface degraded, when FLCS BIT attempted without FCC/MMC power (only on ground)
	106	✓	FLCS 054 1	FLCS FLUP OFF	FLCS FAULT	None	FLCS auto fly-up is inhibited / No fly-up in manual TF
	107	✓	FLCS 036 1	ISA ALL FAIL	FLCS FAULT	None	Controls servo actuators malfunction / Flight control problems
	108	✓	FLCS 034 1	ISA RUD FAIL	FLCS FAULT	None	Rudder servo actuators malfunction / Rudder problems

	ID	Clear	TEST Page	PFL	CAUTION PANEL	WARNING LIGHTS	CAUSE / SYMPTOMS
ENGINE FAULTS	19	X	MC04 326 1 MC13 326 1 IDM 001 1 RWR 018 1 RWR 021 1 SMS 103 1	RWR DEGR CMDS DEGR SMS STA1 DEGR	Engine off: ELEC SYS, SEC & IFF After engine restart: AVIONICS	ENGINE & HYD/OIL PRESS	Single Engine AC & left Engine of multi-engine AC. Produces a flameout / Engine shutdown. Caution Panel and Warning Lights are illuminated while the engine is off. The Test page and PFL faults are shown after the engine has been restarted. Not all of them may appear.
	20	X	None!	None!	None!	None!	Right Engine of multi-engine AC. Produces a flameout / Engine shutdown. No indication of that. Only noticeable if you watch RPM.
	22	✓	None	None	None	ENG FIRE	Single Engine AC & right Engine of multi-engine AC -> produces an engine fire
	23	✓	None	None	None	None	Single Engine AC & left Engine of multi-engine AC -> black smoke only
	109	✓	ENG 015 1	ENG A/I FAIL	ENGINE FAULT & OVERHEAT	HYD/OIL PRESS	Engine anti-ice valve failed (GE129)
	110	✓	ENG 215 1	ENG A/I FAIL	ENGINE FAULT	HYD/OIL PRESS	Engine anti-ice valve failed (PW229)
	111	✓	ENG 018 1	ENG A/B FAIL	ENGINE FAULT & SEC	HYD/OIL PRESS	Afterburner system failure (GE129) / Afterburner not available
	112	✓	ENG 218 1	ENG A/B FAIL	ENGINE FAULT	HYD/OIL PRESS	Afterburner system failure (PW229) / Afterburner not available

	ID	Clear	TEST Page	PFL	CAUTION PANEL	WARNING LIGHTS	CAUSE / SYMPTOMS
AVIONICS FAULTS	113	✓	RALT 003 1	RALT BUS FAIL	AVIONICS FAULT & RADAR ALT	None	Radar Altitude system failure / No Radar altimeter (RDR ALT must be powered to see this fault)
	114	✓	HUD 003 1	HUD BUS FAIL	AVIONICS FAULT	None	HUD system failure / No HUD (HUD must be powered to see this fault)
	115	✓	FMS 004 1	FMS FAIL	AVIONICS FAULT	None	Fuel Management system failure / Fuel Bingo capability degraded
	116	✓	DLNK 005 1	DLNK FAIL	AVIONICS FAULT	None	Datalink system failure / Datalink non-operational

	117	✓	BLKR 003 1	BLKR BUS FAIL	AVIONICS FAULT	None	RWR system failure / RWR is blind
	118	✓	FCC 004 1	FCC FAIL	AVIONICS FAULT	None	FCC failure / Fire Control Computer non-operational
	119	✓	MC04 300 1	MMC DEGR	AVIONICS FAULT	None	???
	121	✓	MC04 300 1	MMC DEGR	AVIONICS FAULT	None	???
	123	✓	AMUX 003 1	AMUX BUS FAIL	AVIONICS FAULT	None	AMUX Bus failure / FCC is forced to NAV
	124	✓	BMUX 003 1	BMUX BUS FAIL	AVIONICS FAULT	None	BMUX Bus failure / FCC is forced to NAV
	ID	Clear	TEST Page	PFL	CAUTION PANEL	WARNING LIGHTS	CAUSE / SYMPTOMS
AVIONICS FAULTS (Cont.)	125	✓	DMUX 003 1	DMUX BUS FAIL	AVIONICS FAULT	None	BMUX Bus failure / HUD, HMS, MFDs (TEST n/a) non- operational
	126	✓	BLKR 060 1	EPOD SLNT DEGR	AVIONICS FAULT	None	Check wegen Dash 1 -> not available
	127	✓	CADC 003 1	CADC BUS FAIL	AVIONICS FAULT	None	Loss of CADC parameters to avionics system / no Airdata available
	128	✓	UFC 003 1	None	None	None	UFC malfunction: No PFL, RWR, DED
	129	✓	CMDS 003 1	CMDS BUS FAIL	AVIONICS FAULT	None	CMDS bus failure / CMDS non- operational
	130	✓	CMDS 006 1	CMDS INV DEGR	AVIONICS FAULT	None	CMDS failure with flares / Flare release non-operational
	131	✓	CMDS 004 1	CMDS DSPN DEGR	AVIONICS FAULT	None	CMDS failure with flares / Flare release non-operational
	133	✓	TCN 004 1	TCN FAIL	AVIONICS FAULT	None	TACAN system failure / TACAN non-operational
	134	✓	FCR 003 1	FCR BUS FAIL	AVIONICS FAULT	None	FCR failure / FCR non- operational
	135	✓	FCR 094 1	FCR XMTR FAIL	AVIONICS FAULT	None	FCR transmit operation failure / FCR not emitting
	136	✓	IFF 003 1	IFF BUS FAIL	AVIONICS FAULT	None	IFF system failure / IFF non- operational
	137	✓	MSL 004 1	MSL SLAV FAIL	AVIONICS FAULT	None	Missile Slave failure / Missile seeker will not follow radar line of sight
	138	✓	RWR 003 1	RWR BUS FAIL	AVIONICS FAULT	None	RWR bus failure / RWR non- operational (RWR must be powered to see this fault)
	140	✓	GPS 003 1	GPS BUS FAIL	AVIONICS FAULT	None	GPS failure / GPS non- operational
	141	✓	INS 003 1	INS BUS FAIL	AVIONICS FAULT	None	INS failure / INS non-operational
	143	✓	EGI 013 1	EGI NAV FAIL	AVIONICS FAULT	None	EGI failure / GPS, INS non- operational
145	✓	IDM 003 1	IDM BUS FAIL	AVIONICS FAULT	None	IDM failure / Loss of IDM	
146	✓	MFDS 168 1	MFDS LFWD FAIL	AVIONICS FAULT	None	MFD Left failure / Left MFD non- operational	

147	✓	MFDS 177 1	MFDS RFWD FAIL	AVIONICS FAULT	None	MFD Right failure / Right MFD non-operational
148	✓	SMS 003 1	SMS BUS FAIL	AVIONICS FAULT	None	SMS bus failure / All functions lost except EJ, SJ
149	✓	SMS 103 1	SMS STA1 DEGR	AVIONICS FAULT	None	Weapon Station 1 degraded / Station is not operating correctly
150	✓	SMS 104 1	SMS STA2 DEGR	AVIONICS FAULT	None	Weapon Station 2 degraded / Station is not operating correctly
151	✓	SMS 105 1	SMS STA3 DEGR	AVIONICS FAULT	None	Weapon Station 3 degraded / Station is not operating correctly
152	✓	SMS 106 1	SMS STA4 DEGR	AVIONICS FAULT	None	Weapon Station 4 degraded / Station is not operating correctly
153	✓	SMS 107 1	SMS STA5 DEGR	AVIONICS FAULT	None	Weapon Station 5 degraded / Station is not operating correctly

	ID	Clear	TEST Page	PFL	CAUTION PANEL	WARNING LIGHTS	CAUSE / SYMPTOMS
AVIONICS FAULTS (Cont.)	154	✓	SMS 108 1	SMS STA6 DEGR	AVIONICS FAULT	None	Weapon Station 6 degraded / Station is not operating correctly
	155	✓	SMS 109 1	SMS STA7 DEGR	AVIONICS FAULT	None	Weapon Station 7 degraded / Station is not operating correctly
	156	✓	SMS 110 1	SMS STA8 DEGR	AVIONICS FAULT	None	Weapon Station 8 degraded / Station is not operating correctly
	157	✓	SMS 111 1	SMS STA9 DEGR	AVIONICS FAULT	None	Weapon Station 9 degraded / Station is not operating correctly
	158	✓	SMS 87 1	SMS STA1 FAIL	AVIONICS FAULT	None	Weapon Station 1 failed / Station non-operational
	159	✓	SMS 88 1	SMS STA2 FAIL	AVIONICS FAULT	None	Weapon Station 2 failed / Station non-operational
	160	✓	SMS 89 1	SMS STA3 FAIL	AVIONICS FAULT	None	Weapon Station 3 failed / Station non-operational
	161	✓	SMS 90 1	SMS STA4 FAIL	AVIONICS FAULT	None	Weapon Station 4 failed / Station non-operational
	162	✓	SMS 91 1	SMS STA5 FAIL	AVIONICS FAULT	None	Weapon Station 5 failed / Station non-operational
	163	✓	SMS 92 1	SMS STA6 FAIL	AVIONICS FAULT	None	Weapon Station 6 failed / Station non-operational
	164	✓	SMS 93 1	SMS STA7 FAIL	AVIONICS FAULT	None	Weapon Station 7 failed / Station non-operational
	165	✓	SMS 94 1	SMS STA8 FAIL	AVIONICS FAULT	None	Weapon Station 8 failed / Station non-operational
	166	✓	SMS 95 1	SMS STA9 FAIL	AVIONICS FAULT	None	Weapon Station 9 failed / Station non-operational
	167	✓	HMCS 003 1	HMCS LBUS FAIL	AVIONICS FAULT	None	Left bus HMCS failure / Loss of HMCS
	168	✓	HMCS 004 1	HMCS RBUS FAIL	AVIONICS FAULT	None	Right bus HMCS failure / Loss of HMCS

	ID	Clear	TEST Page	PFL	CAUTION PANEL	WARNING LIGHTS	CAUSE / SYMPTOMS
OTHER FAULTS	13	X	None	None	None	ELEC SYS	Main Generator "Soft" Failure.
	14	X	None	None	None	ELEC SYS	Main Generator "Hard" Failure.
	15	X	None	None	None	ELEC SYS	Standby Generator "Soft" Failure. Main Generator must fail first.
	16	X	None	None	None	ELEC SYS	Standby Generator "Hard" Failure. Main Generator must fail first.
	31	X	None	None	None	HYD/OIL PRESS	Hydraulic system A compressor failure
	32	X	None	None	None	HYD/OIL PRESS	Hydraulic system B compressor failure
	47	✓	None	None	None	NWS FAIL	Gears must be lowered to trigger this Fault.

17. Third Party Theater DEV notes

17.1 Tilesets

Tile sets will now be selected by NAME via a new *g_sTileSet* option (the former '*g_nTileSet*' number is gone). Default value is "POLAK".

This enables 3rd party tilesets to co-exist on the BMS installation without overwriting each other and without the need for the code to have a fixed set of options upfront.

Info for tileset creators: the following files and directories have to be present and will be loaded if a tileset name (example: "AWESOME") is set:

- Data\Terrdata\korea\texture\texture_AWESOME.bin
- Data\Terrdata\korea\texture\texture_AWESOME*
- Data\Terrdata\korea\terrain\THEATER_AWESOME.L2
- Data\Terrdata\korea\terrain\THEATER_AWESOME.O2

In addition, the following autogen tree textures can be used (BMS will fall back to the default trees1_summer.dds and trees2_summer.dds if not present):

- Data\Terrdata\misctex\trees1_summer_AWESOME.dds
- Data\Terrdata\misctex\trees1_winter_AWESOME.dds
- Data\Terrdata\misctex\trees1_spring_AWESOME.dds
- Data\Terrdata\misctex\trees1_fall_AWESOME.dds
- Data\Terrdata\misctex\trees2_summer_AWESOME.dds
- Data\Terrdata\misctex\trees2_winter_AWESOME.dds
- Data\Terrdata\misctex\trees2_spring_AWESOME.dds
- Data\Terrdata\misctex\trees2_fall_AWESOME.dds

17.2 Add-On Theaters - Tilesets

Theater definition files (TDF) can now override the default tileset to use (from the cfg) by using the parameter "tileset". The flow is as follows:

- 1) Look for a "tileset" entry in the ".tdf" file. If found, use it. If not,
- 2) Look for the tile name config entry in the regular "Falcon BMS.cfg" file. If found, use it. If not,
- 3) Look for a "generic" tileset named "theater" (without any "_<name>" postfix). If found, use it. If not,
- 4) Fail.

17.3 Add-On Theaters - File locations

Add-On Theaters can specify paths to OVERRIDE the default BMS paths for all kinds of things, e.g.:

[code]# Where the campaign directory is

campaigndir Add-On Korea Strong DPRK\campaign

where the terrain data is

terraindir Add-On Korea Strong DPRK\Terrdata\korea

which tileset to use - overrides the global setting in the cfg file

tileset POLAK

where art is loaded from artdir Add-On Korea Strong DPRK

movies

moviedir Add-On Korea Strong DPRK

uisounds

uisounddir Add-On Korea Strong DPRK\sounds\ui

default objects

objectdir Add-On Korea Strong DPRK\Terrdata\objects

misc textures

mistexdir Add-On Korea Strong DPRK\Terrdata\mistex

3d object files

3ddatadir Add-On Korea Strong DPRK\Terrdata\objects

sounds dir

sounddir Add-On Korea Strong DPRK\sounds

simdata dir

simdatadir Add-On Korea Strong DPRK\Sim[/code]

17.4 HiRes Textures

In addition to all these locations above, setting the "HiRes" option will look for a <name>_HiRes folder at any given location used above first and if there is a file load it. If not, it will fall back to the folder <name>.

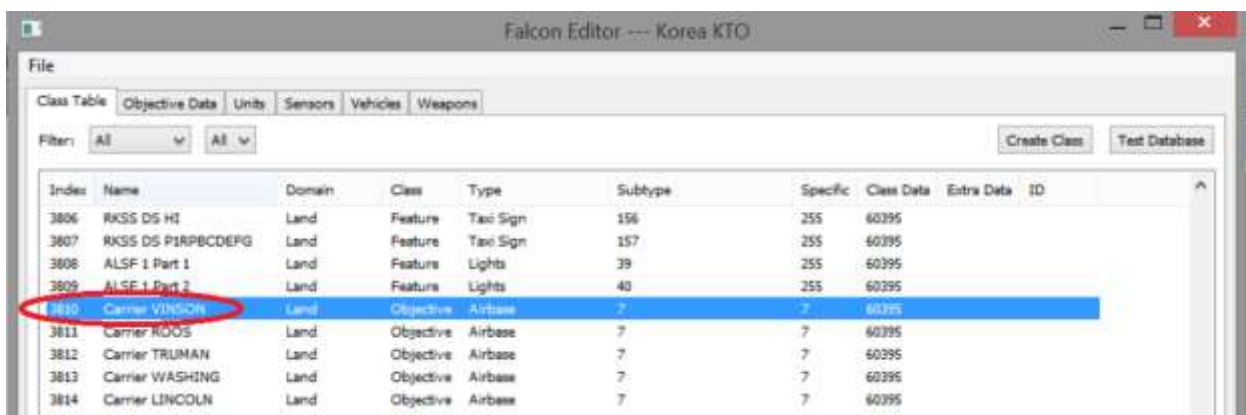
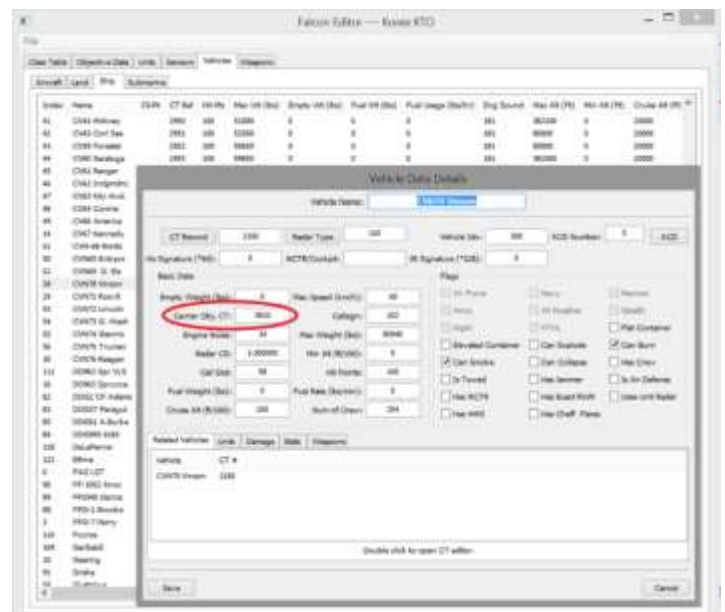
17.5 Adding Carrier to campaign

17.5.1 Database: Creating Objectives associated with Airbases

The carriers are associated automatically with their corresponding airbase/objective in the database.

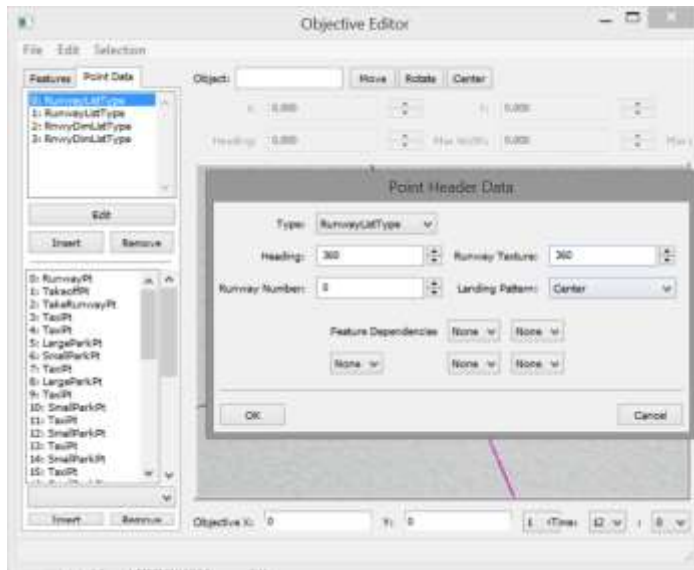
A value is accessible with the BMS Editor to show which Class Table Airbase is associated with the carrier. For a regular vehicle it corresponds to the fuel value – for a carrier, the BMS editor labels changes to display “Carrier Obj. CT”.

Each carrier must be linked with a unique Airbase/Objective. This Airbase/Objective can then be treated just like a regular land airbase.



17.5.2 Database: Carrier Objective Data

The first Point Data corresponds to the Take-Off runway with Spawn, Taxi and Take-Off points. The second is the Landing runway with taxi and parking points. The order (Take-Off 1st, Landing 2nd) is mandatory.



Note that the Take-Off runway should be oriented 360 while the Landing runway should be at 170 for a modern carrier, or 180 for a vintage.

The two following Point Data corresponds to the definition of the dimensions of Take-Off and Landing areas.

For the Take-Off area dimensions, you need to place the bottom edge center of your rectangle in between the two catapults. Both side edges should be at 10ft to 15ft from each side of the catapults.

We recommend you take the existing models as an example. Landing area is the spot for the arrestor cables.

We recommend the use of the BMS Editor to create or move those Point Data.

If you really need to change these data, we recommend to create a **temporary feature** (namely the CT of the carrier in itself) attached to your objective, so that you will be able to visualize each point and runway dimensions on

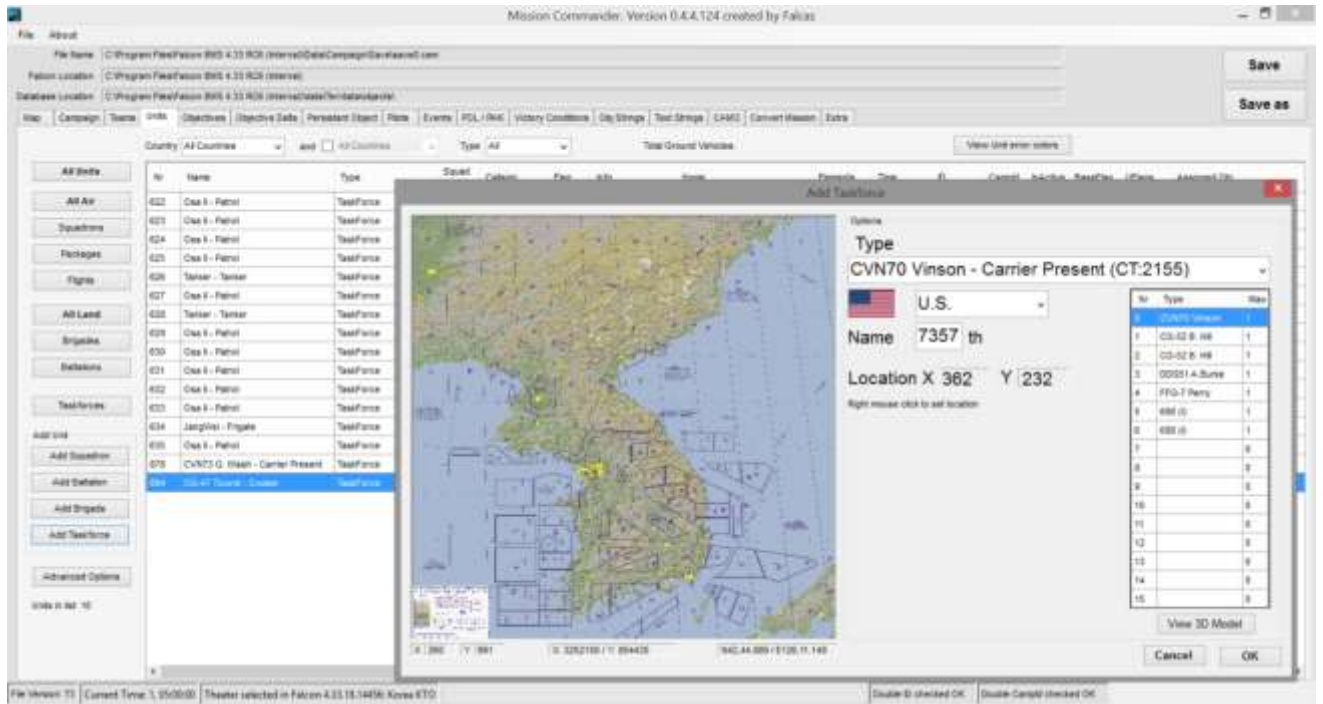
your carrier.

17.5.3 Campaign Files: Placing Objectives in the Campaign

Once you added (a) task force(s) including a carrier in the campaign you need to define the associated objectives for each of the carriers.

There are a few requirements. First is that it needs **the correct OCD Id (CT+100)**, so in our example, 3810 + 100 for the Vinson. Use correct Id and CampId and **place the objective at X=0 and Y=0**.

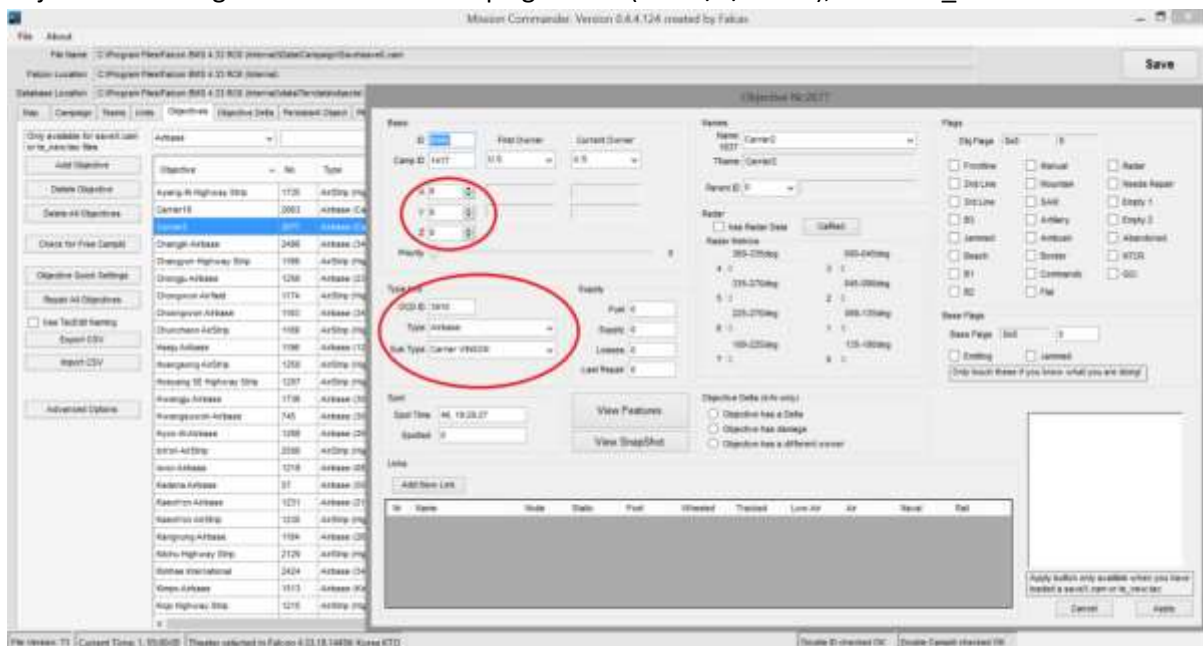
The TACAN and UHF/VHF stations are linked with the CampId number; this is why you cannot have the same carrier twice in a TE or campaign.



These objectives will never be displayed on the map but will be automatically associated with the carriers when needed. If you don't create these objectives in the tac_new.tac or campaign files (save#.cam), the carrier code will not work.

The good news is that for KTO, Mission Commander takes care of these requirements: once your task force is created, you can add an Objective, type airbase, subtype the carrier of the task force you just created, and both OCD Id and X/Y values will be correct. Additionally, in the "Units" tabs, if your carrier is improperly defined in the DB, it will be highlighted in orange. For other theaters, at the moment, you have to do this process by hand in TacEdit.

Use default KTO BMS te_new.tac as a basic example. Be aware that Mission Commander only allows adding objectives starting from the default campaign saves (save0/1/2.cam), or the te_new.tac.



18. IVC

Warning: This chapter hasn't been updated since 4.32

18.1 Purpose & Implementation Overview

Like many other features of Falcon4 BMS, realism is one of the driving reasons for the radio changes. Along with this, the Internal Voice Communications (IVC) for multiplayer, which was first introduced as a feature of SuperPAK 3, has vastly grown in flexibility and realism.

The current implementation in the game code is based on use of an external voice client program run by each player and a remote voice server program that may either be hosted by one player or resident on a separate server system.

There are a number of potential candidates for the client and server voice programs and the code inside the game is designed to be agnostic to the choice of voice client in particular while still offering the ability to control the client from within the game, using the realistic radio and HOTAS controls. This is accomplished by use of a shared memory structure for the game code to provide status to the client program. Any client modified to read the shared memory state can in theory be used to deliver a good voice communication solution.

To make things a little simpler, one such client/server program combination is provided as part of the game install. We are very much indebted to TeamSpeak Systems GmbH/Triton CI Associates for permission to use the TeamSpeak 3 SDK for this development work.

In essence, the voice system implemented for Falcon4 BMS includes therefore three components:

- The voice server program, based on the TS3 SDK; and
- The voice client program, also based on the TS3 SDK; and
- Code in the game designed to provide command and control to the local client program.

If you are wondering about the former IVC implementation that was based on the Microsoft DirectPlay Voice system, this has been removed from the game. Since Microsoft withdrew support for this system some time ago and it wasn't possible to make it work on Windows 7 or newer versions of the operating system, we needed a new approach and that is what you see in the TS3 SDK based programs.

One important note up front: the client and server programs based on the TS3 SDK are NOT – repeat – NOT compatible with the normal TeamSpeak client and server programs available from the TeamSpeak web site.

The SDK programs have a different enough implementation that you can't mix-and-match components. In effect, the client and server programs included with the Falcon4 BMS install are only usable with other copies of the exact same programs. Please don't ask how to change this so your existing TS3 (or TS2 for that matter) server or client can participate in the voice communication systems – it's not possible.

The folks at TeamSpeak/Triton have really done us a good turn with the licensing on this code now. You may all use it with Falcon4 BMS – we can theoretically have as many copies running at one time as we like.

We can run the server as long as we like and as often as we like. Obviously Triton intends for us to be able to get what we need from this tool set to support Falcon4 BMS. If we abuse this Triton has the ability to revoke the license; so please, be mindful of this privilege for all our sakes.

18.2 IVC dos and don'ts

- Do NOT decompile or otherwise reverse engineer the client or server exe's.
- Do NOT decompile or otherwise reverse engineer the dll's that come with the exe's.
- ONLY use the client and server code in concert with Falcon4 BMS.
- Do NOT try to use the client with a standard TS3 server – this won't work anyway.
- Do NOT try to use the server with standard TS3 clients – this won't work anyway.
- Virtual squadrons may run voice servers 24x7 but please run as few as practical.
- Don't start the server exe yourself unless you really need to do that.
- If you run your own voice server standalone for a game, please terminate it when your mission finishes.
- There is a 32 player limit on any one voice server at any one time.

Every one of you has a responsibility to the others in our community; anyone can screw this up for everyone else with a moment's selfishness or thoughtless behaviour; be wise, please and we should have no problems.

18.3 Using the Voice Server Program

It's actually dirt simple. All you really need to know about running the server is that you start it and you are done. At this point it's ready and waiting for clients.

When you start the server exe, a cmd/DOS box pops up. That's mostly there to show status. It has a small set of 'commands' that you can enter in the box. Unless you need to for some reason, the best advice is to leave those alone – they come from the TS3 SDK example server code and we left them for possible use in debug. We haven't tested them other than to discover that at least two of the options `_will_` crash the server exe.

If you run the exe on a machine that has a fully qualified domain name or an IP address that is routable (e.g. a machine that is direct connected to the internet) no other preparation is required – just give that IP address to your prospective flying partners for them to use as the voice server address and let them connect.

If you run the server exe on a system that is behind a NAT router you will have to forward the 3 ports above to the LAN address of the system running the server exe. The code uses only UDP so there is no need to forward TCP. Failure to forward the ports will render the server accessible to LAN clients only. Assuming that you have correctly set up port forwarding, tell your prospective flying partners the WAN IP of your NAT router and tell them to use that as the voice server IP address. Usually that address is one that's routable, given to you by your ISP, and most often delivered via DHCP when your router connects to the ISP (but not always, YMMV).

The server supports a number of command line options to control its behaviour. One of those is `-h` which prints out the following usage message on the console and then exits the program.

```
Usage: IVC Server [-h] [-i <arg>] [-p <arg>] [-w <arg>]
  Help - -h: print this usage message and exit
  Addr - -i <addr>: sets the IP address that the server should bind to
  Port - -p <NNNNN>: sets the 4-digit value as the base port to listen on for this server
(port, port+1
and port+2 are used)
  Word - -w <str>: sets the string value as the code word clients must use to access this
server
```

On some systems with more than one network connection, it is possible that there is more than one network IPv4 address by which the system is known for remote connections. The `-i` option allows you to specify the IPv4 address on which the server will listen for incoming connections and voice data traffic. The format of the address is the typical 4-number integer representation where each number is separated by a dot character. For example you might use `-i 127.0.0.1` (although that's probably not very useful in practice!).

As mentioned above the server uses three UDP IP ports with well-known defaults. In some cases it may be useful to have the server listen on different ports and the `-p <NNNNN>` option makes this possible. The argument for this option is a single integer that defines the lowest port number or a set of three consecutive ports that the server will use. For example `-p 22222` tells the server to use ports 22222, 22223 and 22224 for connections. The help description above says '4-digit' but actually any valid port number will work so long as the set of three ports keyed off the number supplied as the argument are not in use by other programs.

The server does support code word access and that is enabled with the `-w <str>` option. This is a supplemental credential that you can customize to your server and that all clients would then need to present in order to connect. This can be used to limit access to a given server to people that know the code word to use in order to connect to the server. To set an access code word, start the server exe with a single word following the `-w` command line argument. That word can be arbitrary length but only the first 8 characters are considered. It is possible to use non US/English characters in the code word but results may be mixed so it's recommended that you stick to the ASCII printing characters a-z, A-Z, 0-9 and punctuation marks for best interoperability.

When I start the server exe, the cmd/DOS window pops up and this is what it should say if it's working correctly:

```
TeamSpeak Server 3.0.0-beta6 [Build: 11633] SDK
(c)TeamSpeak Systems GmbH

Logging to file started (no console logging on Windows)
Server running
Server lib version: 3.0.0-beta6 [Build: 11633] SDK
Create virtual server using keypair ''
Create virtual server with 32 slots
Create virtual server using keypair ''
Create virtual server with 32 slots
Create virtual server using keypair ''
Create virtual server with 32 slots

Falcon BMS IVC Server commandline interface
[q] - Quit
[h] - Show this help
[v] - List virtual servers
```

```
[c] - Show channels of virtual server 1  
[l] - Show clients of virtual server 1  
[n] - Create new channel on virtual server 1 with generated name  
[N] - Create new channel on virtual server 1 with custom name  
[d] - Delete channel on virtual server 1  
[r] - Rename channel on virtual server 1  
[m] - Move client on virtual server 1  
[C] - Create new virtual server  
[E] - Edit virtual server  
[S] - Stop virtual server
```

```
Enter Command (h for help)>
```

I have done nothing other than start the server to get the above output. I don't type in any commands once the window appears. The server is now open for business. There are 3 virtual servers running in the context of this one exe – one virtual server each for UHF, VHF and GUARD radio frequency sets.

Notice, no mention of the code word in the output there... you'll have to keep track of whether you started the server with one or not.

For the list of command line interface commands the ones that list states are probably safe. The ones that change the state of the server you should avoid; at least for now. The exception to this rule of course is “q” which you are encouraged to use as soon as practical once you have finished using the server in order to shut it down.

The person with the most bandwidth and fastest machine should be the voice host, and maybe even the mission host as well depending on the clients. In large missions with humans, running a voice host and the mission can induce quite a load on the host machine and lower frame rate can be expected. When possible, have one person with a lot of bandwidth and CPU power host voice and another with high bandwidth/CPU power host the mission. Ideally, the best way to use IVC is with a separate dedicated voice server system.

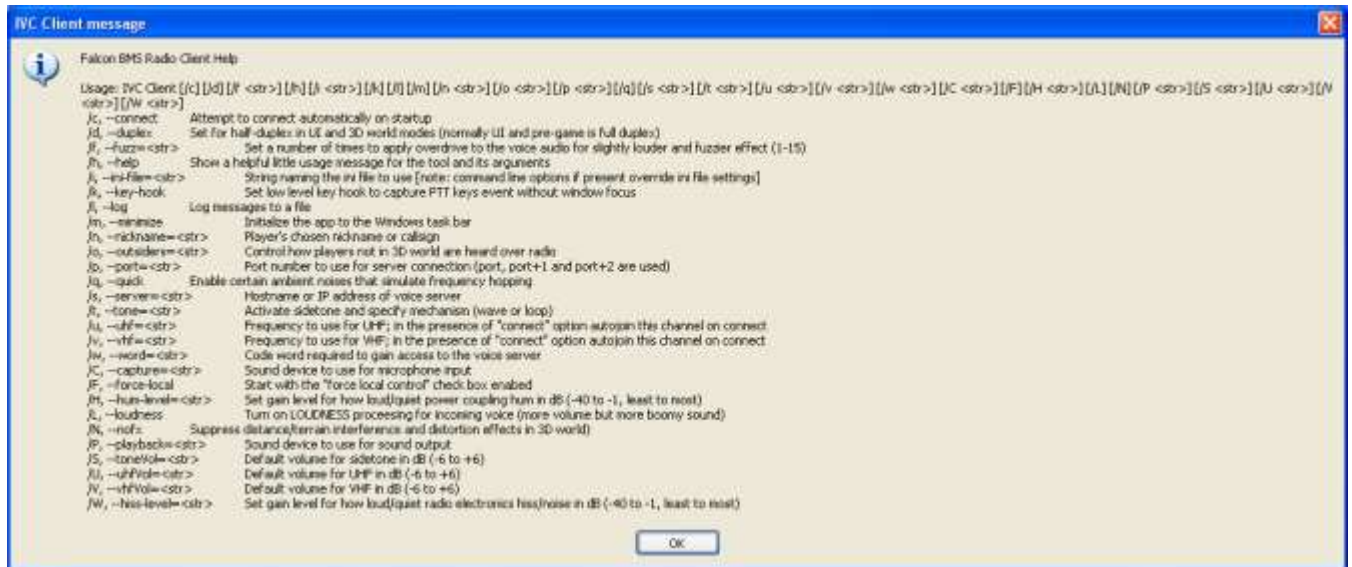
18.4 Using the Voice Client

18.4.1 Command Line Switches and Options

The IVC client includes some new capabilities. In particular I have added a number of command line switches and options that you can use. These are useful for one time configuration of the client but as I hope you will imagine, saving sets of appropriate command line strings as part of a shortcut to the client applet will allow you to parameterize the startup operation in a pretty flexible way.

For instance, a virtual squadron may have a couple of different voice servers in various parts of the world at some point. To that end, I will probably have two shortcuts that auto-connect me to each one separately with the appropriate frequencies pre-selected and set up so I do no more than launch the client and then start mashing the PTT to talk to the other pilots.

So, how's this all going to work you ask?? Good question! Here's a picture of the usage/help text dialog:



[Note: this image was taken on Windows XP. For some reason the dialog aspect ratio default makes it look a lot less tidy on Windows 7 or newer and changing that in the code library used for the client seems "impossible". The information is the same in all cases though.]

You can enter command line options either by launching the applet from the cmd prompt or by setting the command line string in a Windows shortcut.

You can use any or all of these switches and options in combination; a few have interactions with each other as will be noted.

You can use the /<letter_name> syntax or the longer --<switch_name> choice but note that with the slash version you just follow with space and a string for the switches that require an option <str> to be present whereas the --<switch_name> form requires the =<str> (no space) construction so far as I know (latter not tested much because frankly I think the / notation is easier). I did discover by accident that -<letter_name> (i.e. single dash plus letter name) seems to work too, by the way.

If you put in a switch that is not recognized, fail to provide an option <str> where one is required or otherwise mangle the command line so the applet can't figure out what you meant, then you get the above dialog box and the client will exit immediately after you click on the OK button.

Case IS significant for these options. Be careful as some are upper case now in addition to the lower case ones.

Here's a longer description of the available choices:

- **Connect:** /c or --connect
If present, this will cause the client applet to attempt to connect immediately the program starts. Be careful using this one alone because by default your nickname is "noname" and the server address entry isn't either a legitimate ip address or translatable hostname. In other words, expect this to fail unless you use the other appropriate options as well as this one.

Example: "<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /c

- **Duplex:** /d or --duplex

If present this causes the client to operate on half duplex basis in all modes, pre-game, Falcon4 BMS UI and 3D world. If not present, the client operates on full duplex basis for pre-game and Falcon4 BMS UI but you get half duplex in 3D world. Half duplex means that when you transmit, all incoming sound is muted. Full duplex is like the telephone: you can talk over each other if you want. Half duplex is how the radio should work for the 3D world to model the real thing. The sound effects follow this option too – if you enable it for pre-game for instance, then you get mic clicks mixed in with all transmissions.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /d

- Radio distortion: /f or --fuzz=<str>

If present this option specifies the degree to which incoming voice audio is distorted. The range of values you can provide in the string argument is integers one to 15 inclusive. Using one applies slightly more distortion to the signal than the default and 15 applies even more still. The effect to control the degree of mild overdrive applied to the voice data stream. Use of this option is purely an aesthetic choice for you. The default degree of distortion signal processing probably works for most people but if you would like to further "dirty up" the voice signals, this option gives you some alternatives to try.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /f 5

- Help: /h or --help

This one is special...you can put it anywhere in the command line and all other options and switches WILL BE IGNORED. This causes the usage dialog box to show and from there the program will exit. Do not pass "GO!" do not collect 200 local currency units, no soup for you!

The rest of the switches and options act alone or in combination provided there is no /h... You can enter switches and their option strings (if required for a switch) in any order you like...order is not significant.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /h

- INI File: /i or --ini-file

How to setup .ini files will be explained in the next chapter.

- Key Hook: /k or --key-hook

If present this option causes the applet to install a low level Windows key hook. This will gobble up all key presses of **F1**, **F2** and **F3** keys into the applet. It will do this regardless of which window/application has focus at the time. What this means is that you can minimize the client or put it in the background and the PTT keys will still transmit when you press them. So for instance, many applications in Windows respond to **F1** as the "help" key – if the key hook option is present for the radio applet, **F1** will not be delivered to any application other than the radio applet so you won't be able to use it for opening help for other applications.

Another useful property of doing this is that it means you can start the client, connect and start talking all the way into the Falcon4 BMS UI and on into 3D without interruption. Without the hook, when you start Falcon4 BMS (especially in full screen mode; or if you click on some other application and give it foreground focus) you can't transmit again until the COMMS->connect

operation completes successfully. With the hook, the PTTs make the client transmit regardless of what window has focus at the expense of stealing **F1** / **F2** / **F3** from any and all other applications.

One extra note: when you go through COMMS->connect and reach the connection established dialog box, the low level hook is removed if it's active at this point.

That allows you to still use **F1** / **F2** / **F3** for normal avionics key bindings in the 3D world. For the avoidance of doubt the hook is set when the applet is launched and remains in place until the applet closes (yes, it's active even after Falcon4 BMS closes because the applet resets the hook when you leave Falcon4 BMS...so no **F1** / **F2** / **F3** in other apps if you don't also close the applet but it will release the keys back to Windows if you then close the applet as well as Falcon4 BMS).

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /k

- **Radio log:** `/l` or `--log`
Primarily a debugging option so most of you won't need it unless we run into problems. Using this switch forces the applet to write a text file called "radio-log.txt" into the ivc sub-directory in your Falcon4 BMS install. It's full of boring spooze about what the client library and server are up to behind the scenes and there really isn't a lot of interest in there for regular pilots. It's mostly there in case I ask you to enable it to help debug problems at some point.

Note: the log file is one-shot so every time you start the applet any previous content is deleted. Keep this in mind if you are asked for log files to match specific game circumstances.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /l

- **Minimize:** `/m` or `--minimize`
This launches the applet to the taskbar instead of showing the full window on the screen. The BMS code launches the client this way when it auto-launches for you and that's mostly why the option exists but when used in concert with other options you can get the effect of being magically connected to the voice server and talking away all while the client UI is tucked away out of sight.

NOTE: I was trying to get the auto-launch of the client by the Falcon4 BMS code to be silent/background but this doesn't always work properly. When Falcon4 BMS auto-launches minimizes the client but it may also alt-tab away from the BMS window. Good news is that alt-tab is fixed to work properly now so the remedy is just to alt-tab back to the Falcon4 BMS window and you should see the game UI again (no more black-screen/partial screen paints). If this behavior offends your sensibilities you can either: a) educate me how to do this kind of launch from a full screen DX application so that it doesn't alt-tab; or b) if you can't do a) then I recommend just pre-launching the client.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /m

- **Nickname:** `/n <str> Or --nickname=<str>`
This switch causes the string you supply to be used as the content for the "nickname" field in the applet's UI – this is the equivalent of your logbook name for Falcon4 BMS purposes. So for example enter `/n Viper` on the command line. Note this is different than `/n viper` – logbook names *are* case sensitive. Oh and don't put a string that is longer than 60 characters. No really, don't put in any long strings.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" `/n Viper`

- **Radio filter:** `/o <str> Or --outsiders=<str>`
This switch affects the way your copy of the client filters incoming voice messages from remote players when you are already in a network game and flying in the 3D world. Specifically this switch affects messages from players who are not connected to the game and in the 3D world when they transmit a message to you. There are several possible alternatives. The default if no switch is supplied is equivalent to specifying `/o seat` which tells the client to suppress all incoming messages unless the transmitting player is a) connected to the same network game, voice server and frequency as you and b) has taken a seat in a flight on the Air Tasking Order list (i.e. their logbook name appears under a jet in a flight that you could see in your copy of the ATO window were you to be looking at it).

For clarity this means players using the IVC client standalone or who have connected to the host but not yet joined the network game (sometimes known as "first chat") will NOT be audible to you if you are in the 3D world already. By contrast, players joined to your network game and seated in a flight will be audible to you if your radio is tuned to the frequencies used in the UI (see Setting the UI Radio Frequencies below for more information on setting team frequencies).

In addition to filtering in and out messages as above, the seat option also applies radio sound effects (see chapter "Radio Sound Effects" below) using the position of your aircraft and the position of the remote player's flight in the 2D world coordinates to determine audio quality.

The other possible choices for the switch argument are:

- **None**
This means that you will not hear any remote player unless they are connected to your network game *and* they are also in the 3D world on the same voice server and frequency as you.
- **Awacs**
This choice is similar to the default seat option. The only difference is that instead of using the position of the flight the remote player has selected for radio sound quality calculations, the position of the AWACS assigned to cover your flight is used. If no AWACS is assigned for the mission and this option is selected then the range used will be nominal (i.e. you hear little to no signal degradation).
- **all**
This means that you will always hear any remote player regardless of whether they are even running Falcon4 BMS or not with the only proviso being they are connected to the same voice server and speaking on a frequency to which you are tuned of course.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" `/o awacs`

- **IP port address:** `/p <str> or --port=<str>`

This switch causes the numeric value of `<str>` to be used as the base IP port address for the client to use in trying to talk to the server. The client uses 3 ports so this one `<str>` number is the first and the others are one and two more than that respectively. So if I put in `/p 8086` the client will use ports 8086, 8087 and 8088 to reach the voice server (it may also transport your hardware to somewhere between 1978 and 1979 and reduce your system to 8-bit processing mode...don't say I didn't warn you!). In general there is presently no way to change the ports that the server binary listens on anyway so you are best off ignoring this option anyway...it's there for future growth. The string you enter must be a decimal number made up of the characters zero through 9 inclusive.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /p 1234`

- **HAVE QUICK ambient noise:** `/q or --quick`

If present this switch adds some ambient noise to the incoming voice stream on the UHF radio that simulates the clicking effect that you hear when the radio is in HAVE QUICK II frequency hopping mode. At some point, the presence of this effect will be tied into the 3D cockpit controls for the radio in the game (pending with other UHF radio rework in the game code). Until that time, if you wish to simulate use of HAVE QUICK radio you can use this switch.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /q`

- **Server IP:** `/s <str> or --server=<str>`

This switch causes the applet to use `<str>` to fill into the server box in the applet UI. This string can either be a routable IP address or a fully qualified domain name. It is possible to use a more local name so long as it is routable by your system.

If you happen to be running a server binary on your system own you must still enter the address for it – you can use `localhost` or `127.0.0.1` and that should work in that case. This option is useful in combination with `/c` because in that case it will launch the applet and go ahead and attempt a connection to the server you specify with this `/s` option in the `<str>` field. FQDN's must be less than 60 chars long also.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /s 123.456.78.9`

- **Sidetone:** `/t <str> or --tone=<str>`

This option enables sidetone and allows you to select one of two methods for delivering the capability. The only legal values for the `<str>` value are "loop" and "wave". For more explanation of sidetone, see the dedicated section below.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /t loop`

- **UHF frequency:** `/u <str> or --uhf=<str>`

This makes the applet fill `<str>` into the UHF frequency box in the applet UI. This option is useful in combination with the `/c` and `/s` options because it will cause the applet to auto-connect to the server specified with the `/s` option and then auto-join the UHF channel specified with the `/u` option [for you TeamSpeak users, all channels are type "temporary" so they are created and torn down for

you, no server side management required]. The `<str>` in this case can be anything but keep in mind that the game universe uses 6 digit decimal integers to represent frequencies -- essentially the MHz value times 1000 to remove the decimal – Osan tower is 308.800MHz so the `<str>` to use so you can hear players on that frequency is: `/u 308800`.

By the way, if you leave this option off and still use `/c` and `/s`, you are connected and the UHF radio ends up tuned to a channel imaginatively called "Default channel". Channel name `<str>` values must be 60 characters or less. Remember, you can only talk to people on the same channel as you so if you use this option to select something other than the default channel to join, initially you will end up unable to talk to people who just start the client without this option.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /u 123456`

- VHF frequency: `/v <str> or --vhf=<str>`

This makes the applet fill `<str>` into the VHF frequency box in the applet UI. This option is useful in combination with the `/c` and `/s` options because it will cause the applet to auto-connect to the server specified with the `/s` option and then auto-join the VHF channel specified with the `/v` option.

The `<str>` in this case can be anything but keep in mind that the game universe uses 6 digit decimal integers to represent frequencies – essentially the MHz value times 1000 to remove the decimal. By the way, if you leave this option off and still use `/c` and `/s`, you are connected and the VHF radio ends up tuned to a channel imaginatively called "Default channel".

This is NOT the same as the UHF default channel. They are separate radios don't forget! Channel name `<str>` values must be 60 characters or less. Remember, you can only talk to people on the same channel as you so if you use this option to select something other than the default channel to join, initially you will end up unable to talk to people who just start the client without this option.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /v 123456`

- Password: `/w <str> or --word=<str>`

Use this option to specify a code word that is required by some servers to gain access for connection. Only the first 8 letters of this string are use (if 8 or more are provided, less is OK if that's what your server needs). If you need a null password for some server you can explicitly add that with `/w ""`

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /w "password"`

- Sound input device: `/C <str> or --capture=<str>`

This option allows you to pre-select the sound device that will be used for microphone input to the IVC client. The strings to use for naming the devices are going to be specific to your system. You will notice from the client UI that the list of possible options for this device is shown in the dialog box – so long as the string you provide with the command line option matches one of those strings, pre-selection should work.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /C Microphone`

- Force local: `/F or --force-local`

If present this switch causes the IVC client to launch with the "Force local control" checkbox enabled. This is useful for scenarios where you intend, for example, to play the role of AWACS controller using the IVC client to communicate with players in the 3D world while you remain in the 2D UI watching the theater map view. If you use this switch you can join the network game and the game code will not take over control of the client. That in turn enables you to enter specific radio frequencies in the IVC client as required to talk to one or more player flights during the mission.

NB: you can also achieve the same result by connecting to the game with IVC disabled in the game's COMMs dialog if you previously started the IVC client before launching the game. Using /F may be more convenient in that it doesn't require you to remember to disable IVC in the COMMs dialog or to keep multiple phonebook entries for the same game host with and without IVC enabled.

Example: "`<YourInstallPath>\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /F

- AC power hum level: `/H <str> or --hum-level=<str>`

If present this takes a numeric argument in the range -40 to -1 (quietest to loudest). There's a 400Hz tone you hear that simulates leak of AC power hum into the audio feed to your ears. This option can be used to adjust the volume of this tone to taste. The default value is presently -4 for this one though I have it on reliable authority from experts that do this all day that -18 might be a better choice.

Example: "`<YourInstallPath>\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /H -10

- Loudness: `/L or --loudness`

What this does is turn on an audio compressor effect for incoming radio voice. The compressor reduces the dynamic range of the voice sound data and then it boosts the whole resulting signal waveform in amplitude which effectively makes it sound a little less "hifi" but overall higher in volume. Since we're not aiming for hifi anyway, this should be useful to you if you want to make the remote players sound a little louder than they do by default.

Think of this like a LOUDNESS button on a home stereo or car stereo system; apparently a lot of hifi kit and quite a bit of transceiver equipment do this same trick to squeeze max volume out of received signal. To me it sounds a bit more "boomy" with -L than without but it's definitely louder in volume. Off by default.

Example: "`<YourInstallPath>\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /L

- Sound effects: `/N or --nofx`

If present this switch will cause all sound effects processing to do with signal strength and interference to be omitted. This results in clearer sound reception although volume attenuation at extreme ranges is still present. In testing, it was noted that in some cases of mixed language speakers, heavy accents are already enough challenge to communications without the additional difficulties in hearing due to audio degradation. This switch can help if you need to set up a connection that is relatively clear in the 3D world but still sounds like radio transmission.

Example: "`<YourInstallPath>\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /N

- Sound output device: `/P <str> or --playback=<str>`

This option allows you to pre-select the sound device that will be used for speaker or headphone output from the IVC client. The strings to use for naming the devices are going to be specific to your system. You will notice from the client UI that the list of possible options for this device is shown in the dialog box – so long as the string you provide with the command line option matches one of those strings, pre-selection should work.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /P Speaker

- Sidetone volume level: `/S <str> OR --toneVol=<str>`

Use this option to change the default volume level for the sidetone played back to you as you talk. This option does nothing unless sidetone is enabled (see /t above). The range of usable values is +6 to -6 (yes, you can put the '+' and '-' symbols in <str>). The default volume, which the TS code sets to "normal maximum", is accomplished by either leaving out this option or explicitly using it and providing '0' as the option string.

Note: this means that a value of +6 actually amplifies the incoming voice level so it's LOUDER than normal...mind your ears. Minus values reduce the volume below the norm. [NB: this range of adjustment in the minus area is less than the range that can be commanded via control in the game – disagreements on volume levels with pre-game options and in-game levels can result in jumps in volume level as you move from one environment to another...it's up to you to manage this if you don't want the jumps!]. For the technically minded these are interpreted to be a decibel value...which is why minus means quieter and so on.

Note further that it's possible that -6 (i.e. -6dB below normal max volume) could be quite loud, especially for the "wave" based mechanization; the volume level is controllable in the game universe via the INTERCOM knob.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /S +1

- UHF volume: `/U <str> OR --uhfVol=<str>`

Use this option to change the default volume level for the UHF radio channel. This also affects the GUARD receive channel as well. The range of usable values is +6 to -6 (yes, you can put the '+' and '-' symbols in <str>). The default volume, which the TS code sets to "normal maximum", is accomplished by either leaving out this option or explicitly using it and providing '0' as the option string.

Note: this means that a value of +6 actually amplifies the incoming voice level so it's LOUDER than normal...mind your ears. Minus values reduce the volume below the norm. [NB: this range of adjustment in the minus area is less than the range that can be commanded via control in the game – disagreements on volume levels with pre-game options and in-game levels can result in jumps in volume level as you move from one environment to another...it's up to you to manage this if you don't want the jumps!]. For the technically minded these are interpreted to be a decibel value...which is why minus means quieter and so on.

Example: "`<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe`" /U +2

- VHF volume: `/V <str> OR --vhfVol=<str>`

Use this option to change the default volume level for the UHF radio channel. The range of usable values is +6 to -6 (yes, you can put the '+' and '-' symbols in `<str>`). The default volume, which the TS code sets to "normal maximum", is accomplished by either leaving out this option or explicitly using it and providing '0' as the option string.

Note: this means that a value of +6 actually amplifies the incoming voice level so it's LOUDER than normal...mind your ears. Minus values reduce the volume below the norm. [NB: this range of adjustment in the minus area is less than the range that can be commanded via control in the game -- disagreements on volume levels with pre-game options and in-game levels can result in jumps in volume level as you move from one environment to another...it's up to you to manage this if you don't want the jumps!]. For the technically minded these are interpreted to be a decibel value...which is why minus means quieter and so on.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /V -1`

- Hiss level: `-W <str> or --hiss-level=<str>`

Option, takes a numeric argument in the range -40 to -1 (quietest to loudest). This is similar to the AC power hum, ambient noise in the incoming transmission in more of a hissing form. This option can be used to adjust the volume of this tone to taste. The default value is presently -5 for this one though I have it on reliable authority from experts that do this all day that -18 might be a better choice for this option also.

Example: `"<YourInstallPath>:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /W +10`

One useful way to use these is with shortcuts. Perhaps make a folder of shortcuts for each of the voice servers you expect to work with. Let's assume that I want to auto-join the server and use the same UHF/VHF frequencies that BMS is set up to use by default in the game UI chat/mission screens. Given the server address we are using as an example I'd make a shortcut that has this command line:

```
"c:\FalconBMS\Bin\x86\ivc\IVC Client.exe" /c /k /n Viper /s ivc.mydomain.org /u 307300 /v  
1234
```

I could add a `/m` in there as well to have it launch to the task bar instead of opening a window initially as well perhaps. I'd probably rename the shortcut to something pithy like "mydomain IVC" and call it good from there.

18.4.2 INI File Setup

Earlier versions of the IVC client only supported command line options as described above. This capability is now supplemented by support for an ini file to contain the same option settings.

In all cases if a command line argument is present for some option or switch, that command line option takes precedence over the content of an ini file. You can specify the name of the ini file with the "`-i ini-file`" option which takes the name of an ini file (without the .ini extension) as an argument to select which file to use. In the case that no ini file is specified, the app looks for the default one which is named for the app ("IVC Client", which would be named with the ini extension).

Note: the app looks for the default ini file every time unless there is a `-i` option on the command line.

If you are really feeling perverse, using `-i nonexistentfile.ini` (where no such file exists) will cause the app to bypass the default file and since there would be no file by that name, you'd skip any parameterization by ini file content. In the case that there's no `-i` option, it always looks for an "IVC Client.ini" in the directory where the "IVC Client.exe" file is located and will load that such that any options and switches in there will take effect provided no equivalent command line options are present (remember: command line options and switches always take precedence over any ini file setting if both are present).

One other way to say all this which might help, here's how the client deals with options and switches procedurally:

- The exe looks for a "`-i filename`" option on the command line
 - if it finds `-i` then it uses `filename.ini` and tries to open that to look for more options and switches
 - if it doesn't find the `filename.ini` then no ini file content is used in what follows
- if there is no `-i` option, then it looks for the default of "IVC Client.ini" for more options and switches
 - if the default ini file is not found, then no ini file content is used in what follows
- now it looks for each option/switch in turn:
 - is the option/switch specified on the command line?? if "yes" use that and go on to the next option/switch
 - if not found on the command line, look in the ini file for a value to use. If found, use and go on to the next option/switch
 - if not found on the command line or in the ini file in use, then use exe's internal default
 - keep doing this loop until all possible options/switches have been considered

In general the easiest thing is to place any ini file(s) you want to have in the directory with the IVC Client executable since the code looks in the install directory of the app. Format for lines in the ini files is:

`<key> = <value>`

Thus as an example placing the following line in the ini file:


```
duplex = 1
```

Sets the client to half duplex mode even before the game enters the 3D world regime.

The key values are the same as the long form of the command line option/switch names so the above example is the same as setting a `-d` switch on the command line for the app.

For switches like `-d` that have no arguments they are enabled in the ini file with a value of 1 (and only 1 will do; anything else will be interpreted as disable for that capability).

Other keys that are equivalent to command line options that take an argument use the same format for the value as on the command line. So an example there would be:

```
nickname = starbuck
```

which sets the username for you.

Only one ini file is used at a time so if you specifically select one with the `-i` command line option, the values in the default (app named) ini file are ignored.

Some general advice and suggestions about the ini file having played with this a bit in testing.

If you are a person that starts the app before going into the game, the ini file can be a handy way of setting up automated connections to a pre-selected server. Put:

```
server = thehostname  
  
nickname = starbuck  
  
connect = 1
```

in the file called "IVC Client.ini" that is located in the same directory as the IVC client executable file and the app will automatically pick up your name and connect you to the host called thehostname.

You could have separate ini files for different hosts that have "`server =`" lines that match the hostnames for each that you connect to and then you get a separate profile per voice host. Just use a `-i` command line option to select the file `this_host.ini` from `that_host.ini` for example as "`-i this_host`".

What I have decided to do is put the options and switches that are common to all connections I hook up to in the default file and then use separate shortcuts with command line options to pick out the host I want to connect to at any given time. One advantage to this strategy comes up in the rare case that the client crashes in mid-game -- if that happens Falcon4 BMS will attempt to relaunch the client for you automatically...having the choices I want in the ini file for all connections means that these are applied in the restart case (remember: you can't set command line options or switches once you are into the game, Falcon4 BMS just launches the IVC client with defaults for you if you didn't start it before the game) because even in that case the app does read ini file content on restart. This isn't a problem because the parameters you entered for server and the channels you are tuned to etc. are all communicated to the app on restart directly by the game code...in other words the default ini file content there allows a restart to ensure you have the same settings even if you launched the app with command line options and switches or a host-specific ini file.

Here's a bit larger example of ini file content:

```
# An ini file with a variety of switches and options
duplex = 0
server = 192.168.0.132
nickname = viper
connect = 1
# "Front pink input" in mixer, front mic input jack on SPEAR machine
tone = loop:3
loudness = 1
uhf = 307300
vhf = 1234
key-hook = 1
quick = 1
hum-level = -18
hiss-level = -18
toneVol = +6
uhfVol = +2
vhfVol = -1
outsiders = awacs
log = 1
```

Note that a line prefixed with a # is treated as a comment and ignored by the client. What the other lines do is:

- disable the duplex option (yes, I know that's off by default so same as line not being there but this is an example of how you could quickly turn this capability on and off...simple edit to "1" for the value and duplex would be then set to on);
- sets localhost to the place to look for a server to connect to (yes, "localhost" works fine for this if you have a server running on the same machine as the client);
- sets my nickname;
- sets the app to try a connect on start-up;
- enables sidetone using the loopback mechanism and since I'm on Win7 for this machine it specifies the 4th (zero base counting remember) device in the record mixer set as the one to turn on for transmit feedback to yourself;
- enables the audio compressor loudness;
- sets initial UHF frequency to 307.300MHz;
- sets initial VHF frequency to 1234 (which is the UI **F2** default);
- sets the key hook switch that means that **F1**, **F2** and **F3** key presses are sent to the app regardless of whether it has focus until you enter 3D world in the game (at which point this hooking is suspended to allow you to use those keys for in-game functions);
- enables the simulated have quick channel hop clicks;
- sets the hum volume to -18dB...rumor has it this is the "best" choice;
- sets the hiss volume to -18dB...rumor has it this is the "best" choice here too;
- sets the sidetone volume to max (-6dB to +6dB range of adjustment);
- sets the UHF volume to 2dB above default;
- sets the VHF volume to 1dB below default;
- makes all players on my voice server and frequency who are connected to my network game but not in 3D world sound like they are transmitting from the AWACS jet while I am in 3D world; and
- enables debug logging to a file called radio-log.txt (also found in the IVC client binary's directory).

18.4.3 Client User Interface



If you start the client applet with no command line arguments, you should see a window like the one above as a result. **Note: the listed sound devices will vary depending on your system hardware.** Focus first on the Server Connection group of controls.

In the Nickname box fill in your callsign or equivalent. Note that when you get into a multiplayer game the content in this field is overwritten with the name that matches your logbook as selected in the game configuration. It's going to be less confusing if you enter the same name (case sensitive) here as the one you use for your Falcon4 BMS logbook. "noname" is the default nickname pre-entered for you if you don't change this field's content.

In the Server IP/DNS box you type in the routable IP address of the target voice server or a fully qualified domain name for that server. Note that for a server running on the same system as the client applet, you can enter 127.0.0.1 or "localhost" – either choice will work to connect you in that case.

The connection status box is informational. As shown above you begin in not connected state unless you use some of the command line options listed above to automate an initial connection.

Beside the status box is the connect button. Click this to initiate a connection. **Note that you will only get a useful response if the server box contains the correct address or hostname for a system running the voice server.** Pressing connect before entering the address will cause an error.

You will notice that the Radio Frequencies group of controls is greyed out when the applet starts. These controls become active once a successful server connection is initiated. If you had used command line options to pre-fill the radio frequencies, you should see those values in the boxes but they are still greyed out until a connection is made to a valid server.

Once the frequencies group does activate, you can enter channel values and begin transmitting. If there are no command line options used, you will initially be connected to the server and each of your radios placed

in a frequency called "Default channel" (UHF and VHF transceivers both have separate default channels so you have to transmit on the right radio to talk and be heard).

In the UHF Freq box you can type in any string value. When you click on the Change FRQ button to the right of that box, you will be connected to the channel named by the string on the voice server. Choice of string is completely up to you but the most useful way to use the applet is to enter strings that represent entries in the radio frequency domain that you can use inside the game. To do that, enter a string with six digits that represent the MHz frequency multiplied by 1,000. For example, if you wanted to dial in Osan tower in your UHF channel, enter the string 308800 to represent 308.800 MHz. You must click on Change FRQ for a newly entered string to put you in a new channel. Hitting ENTER after typing the string moves the window focus from the text box to the Change FRQ button so you can type "308800", ENTER then ENTER again and that will be a shortcut for clicking on the Change FRQ button as well. The UHF band frequencies are from 225.000-399.975 MHz in 25 kHz stepping, just like the real thing.

The VHF box and Change FRQ button operate exactly the same way but the recommended frequency range to use is different. The VHF band frequencies used in the game are from 116.000-151.975 MHz in 25 kHz stepping.

The volume boxes allow you to enter numeric values to a maximum of 6.0 either side of zero. These are decibel values that modify the radio output volume. Zero is the default sound level. Plus 6.0 is the maximum amplification above the default (caution: sound may overdrive and clip depending on your hardware). Minus 6.0 attenuates the volume level significantly. *Note that the UHF volume also controls the volume of the GUARD transceiver – these cannot be modified separately.* The intercom volume reflects the level of sidetone volume and hence this value will have no effect unless sidetone is activated (see below).

The three radio buttons to the right of the control group are informational. These activate when you press and hold one of the push to talk keys: **F1**, **F2** and **F3** for UHF, VHF and GUARD transceivers respectively. If you want to talk for a long time and not to have to hold down the PTT key for a given channel, then you can click on the active check box for that radio which is the equivalent of holding down the PTT. Click a second time to release it or press and release the PTT; either will clear the check mark and cease transmission.

Oh and let me forestall one thing right away: there is no way to change the PTT keys for the applet **F1**, **F2** and **F3** is all you get. The Falcon4.0 precedent rules!

There is also a checkbox for Automatic Gain Control – this is enabled by default and seems to work reasonably well most of the time. Being able to turn this off for the applet may help some users so the option is there if you end up needing it.

The last gadget in the radio frequencies grouping is a "force local control" check box. Most people should stay away from this. The purpose of it is to prevent the applet code from looking for Falcon4 BMS and hence to prevent the client from being slaved to the Falcon4 BMS code. This can be useful in scenarios like debug or when you are playing the role of AWACS controller but probably not useful outside these types of usages. If you click it while Falcon4 BMS runs, then you will likely be disconnected from the server if you are connected at the time. It's not totally clear that it's safe or reliable to disconnect in this fashion however...changing this setting while the game runs is at your own risk: you have been warned!

Below the Radio Frequencies group is the Sound Device control group. This presents two list boxes with choices for the microphone input device and the audio output device. The strings here represent what you would use if you wanted to use the command line options listed above to preselect which device(s) to use. In the applet UI, you can use the lists by clicking on the devices you want to direct the client to use for sound capture and audio playback. Note that while the selection for each is independent, most testing and likely the most reliable way to work with this, not to mention best performance, focuses on using the same physical sound card for capture and playback. Using the same for both is therefore recommended best practice.

Finally, note the version number in the bottom right hand corner: all players in a session connected together via a server need to be using the same version of the client application. The current version to which this documentation refers is 1.2.8.

18.4.4 Interactions between the Client and the Game

In practice there are two ways to start voice comms to use with the game. You can either launch the applet before the game starts or the game can start the client applet for you. Only one instance of the voice client applet can run on your system at any one time.

The launcher gives you the opportunity to start the voice client applet separately. This allows you to connect for voice communications in advance of starting the game and making a multiplayer connection. Among other things, this may be useful to talk players new to multiplayer through the process of making a connection.

Alternatively if you enable voice communications and the client applet is not running as you press the “connect” button then the applet is launched for you automatically in the background. In some cases an alt-tab may be required to bring the game UI to the foreground again once the applet has launched. This mode more closely resembles the operating model of the earlier DirectPlay voice system and may be more comfortable for some players.

If the client is running before the game then it runs independently up until the “connect” button in the COMMs UI page in the game is pressed to initiate a multiplayer game. At that time, provided the “force local control” gadget in the client applet is not checked, the applet will slave itself to the Falcon4 BMS code. When the applet is slaved the local user interface controls are disabled and all parameters are driven from the game. Among other things, this means that if the COMMs UI dialog has any different parameters from those used to start the applet (such as logbook name or voice server address), then the applet is directed to disconnect and then reconnect with the settings from the game UI.

There is some amount of error recovery built in to the voice system so if for some reason the connection to the voice host is lost or dropped, then attempts are made to re-establish communications. Absent any error though, the connection is maintained as long as the game runs. When you exit Falcon4 BMS, the connection to the voice host is terminated and local control is return to the UI for the client applet.

18.5 Radio Sound Effects

This revision of the 4.32 code includes three wav files that provide some audio sound effects.

There's a wav for each of the mic activation clicks on start and finish of a transmission. Some folks don't like the idea of the lead-in or end-transmission clicks – if that's you, simply rename or remove the corresponding file and the sound won't be played.

In previous versions there was a fixed wav file that provided an ambient background noise when you transmitted in the 3D world. This has been removed and replaced by a more dynamic system digital signal processing code which is applied to incoming voice data before it is played back to your sound output device.

There is also a 'block' wav. This one is LOUD and sort of obnoxious by design; in case you are wondering it is taken from a real radio clip. This one is played when more than one person is transmitting on the frequency you are listening to and represents co-transmission interference. Obviously you can only run into this one if three or more people are on the frequency; you listening and at least two others talking over each other.

Note that the blocking sounds interact with the half-duplex transmit mode that is also included in this update: when any person transmits all incoming sound on the frequency is muted for that person. Thus in the case of two (or more) folks talking over each other now, they hear nothing but everyone else not already talking on the channel gets an earful. Be careful what you wish for!

In the past, the option to force half-duplex operation meant that you heard background noise and blocking sounds in the UI as well as in the 3D world. This is no longer the case: sound effect processing is limited to operation when you are in the 3D world although you can still force the UI to operate in half duplex mode.

It may be worth noting that this is one of the bigger compromises in the model: the block sound is taken from a real aviation radio recording so it's realistic but it is not realistic that this would be what you hear every time for every combination of transmitters stepping on each other's transmissions. Making this effect more dynamic is an area for possible future enhancement but for now there is a single, uniform effect to indicate that the channel was blocked.

The format of the fixed wav files is: 48kHz sample rate, 16-bit, single channel audio (and for the really technically inclined the "format length" value in the header can be 16 or 18 and should still work so long as the data is consistent although actual "extra format" info is not used if present). If you choose to replace or edit the files supplied, your replacements should follow the same format.

18.5.1 Digital Signal Processing Sound Effects

When players talk to each other in the 3D world, the voice client applet will apply DSP effects to the voice samples in an effort to render a more realistic impression of radio sound.

For the most part this consists of attenuating and then filtering the otherwise-clean recorded sound and adding ambient and radio system derived noise.

Among other things this works best in the 3D world because player positions relative to each other are used to calculate the type and extent of effects to apply in real time. As the implementation of these calculations is described, keep in mind that this is a model of how real radios operate designed to give you a more realistic impression. Said another way, this is not an attempt to model with extreme accuracy the very complex and tricky business of getting one pilot to talk to another pilot using real world MILSPEC equipment. Making a completely accurate simulation of radio traffic would require more code and CPU budget than we can reasonably afford for this one feature of our game world. As such, there are some compromises built into the model presented. However, I hope the end result is an improved experience relative to the formerly rather sterile radio system from earlier versions. Expect this model to evolve with experience and as better data and documentation become available.

The primary input for the DSP calculations is the position of the speaker's aircraft relative to your jet as receiver in the 3D world. This is presented in the form of a slant range and AGL height for the purposes of the model (this is all taken care of for you by the applet in concert with the game code).

There are a number of factors that are blended into calculating the exact parameters used for DSP effects as one player talks to another. These include:

- Transmitter power models typical aviation radio sets for tactical aircraft and ultimately power limits potential effective range for any signal; and
- receiver sensitivity models the ability of the radio to pull signal out of the radio spectrum and turn it back into audible voice again based on typical values for tactical aircraft radio sets; and
- channel frequency is taken into account such that other things being equal, VHF signals are stronger than UHF signals at any given range so VHF has the longer absolute effective range; and
- radio horizon is modelled so that if there is no line of sight between players because of simple curvature of the earth, then you will not hear an incoming transmission; and
- similarly, if there is no direct line of sight between players, then you will not hear any incoming transmission; and
- free space signal path loss is used to determine signal strength from the transmitting player as it arrives at the receiver; and
- terrain and some atmospheric interference impact modelling is used to further refine signal strength from the transmitting player based on range and height above terrain for receiver and transmitter.

In practice, the terrain interference model and the radio line of sight constraints are the ones that you will run into most often. If you are flying nap of the earth through rolling terrain or mountains you may lose line of sight for transmitters who may in fact be well within effective radio range as measured by the other constraints. Equally, players flying along at a few hundred feet AGL but separated by tens of miles may find themselves unable to talk without first climbing a few thousand feet to get out of the ground clutter.

The effects are calculated dynamically such that radio signal perceived quality will degrade as signal strength falls. Unless you run into one of the hard line of sight constraints the path loss models progressively reduce signal strength up to the point where remaining strength is below the modelled receiver sensitivity. As signal strength approaches receiver sensitivity threshold, the received audio will reduce in volume and there will be more distortion and in the limit the signal will break up and become intermittent before disappearing entirely once the threshold of receiver sensitivity is breached.

[Again for the technically inclined: the terrain interference model is loosely based on the Egli model and other similar terrestrial signal propagation models, but adapted for aviation antenna heights based on input from pilots about expected radio performance. Most of the path loss models that deal with terrain effects that are well documented in public sources tend to focus on terrestrial based antenna systems for both transmitter and receiver (TV, cellular etc.). Clearly though, terrain interference has the potential to affect tactical aviation radios; although that's likely to happen at longer ranges and higher AGL values than would be typical for the terrestrial radio models.]

18.5.2 Sidetone

Sidetone refers to the idea of a small amount of your own voice content spoken into the microphone being fed back to you in real time through your speakers or headphones. Almost everyone will be familiar with the phenomenon even if not with what it is called: if you have ever used a plain old telephone you will have heard yourself in the earpiece as you speak into the receiver/handset. The reason this exists is to help you modulate how loud you are as you speak into the microphone elements – take away side tone and you have no reference when people on the receiving end ask you to speak up. What's more, in most practical cases people have a tendency to yell into microphones to ensure they get heard if there's no side tone feedback to guide them. Almost every radio transceiver, including gear used in tactical aviation, implements the side tone capability.

In explaining the implementation and how to use it to the test team, it became apparent that you may need to consider it something of a feature for advanced users who have a good grasp of the Windows sound subsystem and how the design of sound mixing works under Windows. It's simpler under Windows XP than under Windows 7 or newer. In the latter case, even finding the correct parts of the sound subsystem to use for this capability can be challenging. There really doesn't seem to be a way to make this simple so if you want to use this, be prepared to have your knowledge of sound in Windows tested. If that seems daunting as a prospect it may in fact be better to simply skip over this section.

The IVC client applet now has the ability to create the effect of side tone for you as an option. As described above this is enabled with the `/t` or `--tone` command line option. There are two basic mechanizations for this effect: one uses the captured sound from your microphone and plays it back for you and the other makes use of hardware loopback inside your sound card device that routes audio from the microphone to your output device. The former is enabled with the "wave" argument (i.e. `--tone=wave`) and the latter with the "loop" argument (i.e. `--tone=loop`). Unless the client recognizes one of those two exact command line option formulations then sidetone will remain disabled completely.

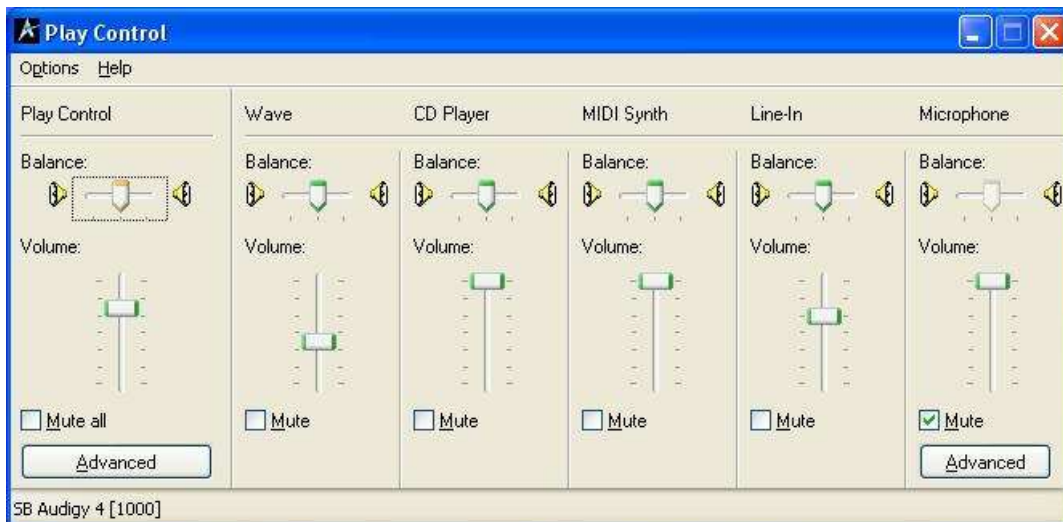
In all cases, use of the loopback capability of your sound card is preferred. This ensures that you hear your voice with imperceptible delay. The reason the wave option exists is because of the fact that not all sound card devices include the hardware loopback capability as part of their design. For example, many USB headset type devices will not give you any hardware loopback capability.

The drawback with the wave system is that it's effectiveness is very dependent on how good the implementation of the sound card driver stack is – in many cases, when you enable wave based sidetone, there is delay in recording the audio into the client's memory and then playing it back and that delay means that you may hear your voice fed back a half second or more delayed...sort of like a giant echo on a phone line...obviously very distracting and arguably not really useful. Still, some sound card devices may do a good enough job at low latency record/playback to make wave an option if hardware loopback is unavailable to you.

Some experimentation may be required to find the best solution given the sound hardware in your system and it may be the case that there is no usable solution for you to enable sidetone without changing the sound hardware in some instances.

Enabling sidetone is further complicated by the redesign of the Windows operating system sound mixing capabilities between the Windows XP and Windows Vista product generations.

Under Windows XP, if you enable loop sidetone in the client applet, the code will find a hardware loopback microphone monitor if one is present. The applet will remember mute status and the volume level the slider was set to before the applet was launched and will restore those on exit. You can actually bring up the Windows mixer controls window and start the client applet to see this happening.



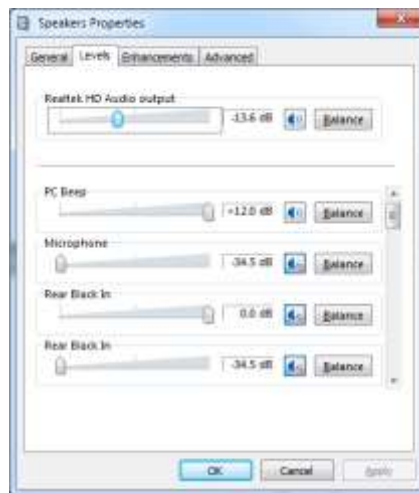
In the above example the window shows the playback audio volume controls and the one to the far right labeled "Microphone" controls the hardware loopback. Note in this case it is muted and the volume is set at max. One tip is to open this window before even starting the IVC applet and uncheck the mute with the volume at the lowest level and then move the slider up as you speak into your microphone to get a sense of how much sidetone volume is comfortable for you. You can use that to gauge how to set the initial volume command line parameter (`--toneVol`, see above).

When the client starts with sidetone enabled and available you will see the volume level slider move to the default or the level specified by command line option and the mute will be checked. Then when you press the PTT you can see the checkbox for the mute clear while you talk and then set again as you release the PTT. The slider controls the level of sidetone and the mute is used to enable and disable it as you talk, much as it would work for a real radio set.

If your sound device has no hardware loopback or equivalent, you may not be able to find a slider/mute for the microphone in the playback volume controls. In that case, wave may be your only option short of changing the sound device.

Unfortunately, the situation is a good deal trickier on Windows Vista and subsequent Windows versions. Microsoft changed the mixer controls such that there is no way (literally: the devs at Microsoft central in Redmond confirmed this directly) to programmatically discover the mute and slider associated with a particular microphone and sound card device into which that is plugged. At least no way to do it

automatically without some additional context or a hint in lay terms. This means the format of the `loop` option under Windows 7 has a little more information tacked on the end to give that hint.



You find the above page of Speaker Properties Levels settings by right clicking on the speaker icon in the system tray and selecting “playback devices” to bring up the dialog and then selecting the Levels tab. The mute control and slider that you are after is on this page, provided of course there is one.

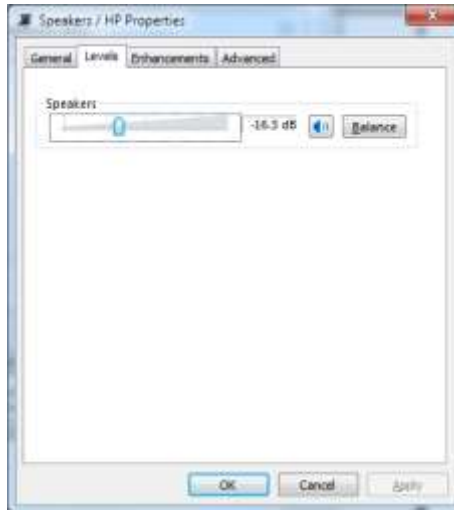
The essence of making this work is to give a number hint as part of the `loop` argument – it is programmatically possible to find and control the n^{th} entry on this page. So when you start the client you would use the `--tone=loop:3` format, adding a colon and a number to hint to the client that the correct slider to work with is the 3rd (in this example) listed in the Levels tab. The index numbers are zero-based, meaning the first is “0”, the second one is “1” and the third “2” and so on. Thus it’s possible that you may need to use `--tone=loop:0` if the very first gadget listed in the levels tab is the one you want to control. This colon + number suffix is ignored if present in the command line argument on Windows XP, it’s only needed for Windows 7.

Now there’s a bit more to it unfortunately. Not all the sound control gadgets that Windows will list when the code enumerates this collection are actually made visible in the Levels tab. It appears that the master volume (in the picture above that’s the top one above the horizontal rule line across the tab content) is not counted on some (all??) systems as part of the collection. If that wasn’t tricky enough, in the example above taken from my laptop you might imagine that the one labeled “Microphone” is the one I want but it’s not – that is the built in microphone on the machine chassis and with my headset plugged in to the side of the machine it’s actually the first of the two labeled “Rear Back In” (the second of those entries is actually the line in input socket...it’s on the side of the machine too, but hey, what do I know...the fact that there are two controls for separate devices labeled the same exact thing will tell you why an index number and not a string to declare the gadget you want to use is required...and in case you are wondering it’s the brains that wrote the Windows driver code for this hardware device that get the “credit” for the exact duplicate names with misleading geography content for different connections to their device, not Microsoft in this case).

So in my example system, as shown in the picture above, there’s actually one invisible device and I need the first “Rear Back In” and the master volume isn’t counted. So by trial and error, and unfortunately there’s no better way to do this, I found that “3” is the correct value to hint to the client to use with my headset plugged in and “2” is the right number to use if I unplug the headset and use the built in microphone. Again as above, the best advice for figuring this out seems to be to start the client standalone with a command

line option to enable sidetone and indicate a device number and then see which if any mute controls flip state when you press the PTT by watching the mixer levels page of the Speaker Properties dialog. It's actually easier to see in practice than it is to describe.

As previously stated many USB headset devices have no loopback and limited sound mixing – I've seen examples of the above dialog box from Windows 7 machines like the one below where all there is present is a master volume. If that's true for your system, you may not be able to use loopback based sidetone at all without changing your hardware.



If you are working with Windows 7 and you have no hardware loopback there is a way to audition the effect of the "wave" type sidetone implementation.

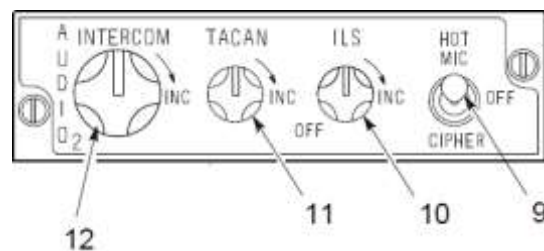


The above dialog can be found by right clicking on the sound icon in the system tray and selecting recording devices. From there, highlight the microphone device you plan to use and click the properties button to get this dialog. Notice in the "Listen" tab there is a check box for "listen to this device". In effect that is the same as the "wave" implementation in the IVC client applet. Thus selecting that check box should activate your microphone so you can hear yourself. If the delay is not noticeable or minimal to the point it's not

distracting then using “wave” in the client applet may work for you. If the effect here is too distracting it will likely be no better in the client.

One last piece of advice about making loopback sidetone work for you: it will likely work better if you select the same sound card device for input and output in the client applet UI or via the command line arguments for that. Hardware loopback is typically implemented internal to the device hardware and the chances of loopback working where one sound card records your voice and a separate sound card plays it back are much more remote.

In addition to the output of the recorded voice sounds from the microphone input, the client applet also mixes in a little noise while the push to talk switch is held. This simulates the system and ambient noise that are typical in aviation radio mic regimes, even ones with noise cancelling capabilities. In the `bin` directory there is a WAV file called `opencircuit.wav`. This is a short clip that is mixed in as a continuous loop with the sidetone voice audio as you talk. You can turn off use of this effect by removing or renaming the file. Additionally, if you want to change the sound that is mixed in as you talk you can edit or replace the contents of the file to have your own preferred sound effect mixed into the playback stream (file format is as above for the other wav files the client uses).



It is possible to control the volume level of the sidetone played through the headset channel when you are in the game 3D world environment. This is done with the INTERCOM (12) knob on the AUDIO2 panel shown above. There is an analog control channel mapping for this control present in the game setup UI or you can use keyboard callback mappings.

Please note that this is a volume control – for this to work correctly, meaning for you to perceive smooth change in volume level as you turn an analogue control device (presumably a potentiometer in most cases) the device must use a logarithmic taper (sometimes referred to as an “audio pot”). If you use a linear taper potentiometer for this the effect you will likely hear is most of the settings of the knob delivering no volume at all and then at one end of the adjustment range you go from nothing to full volume in a very small region of change. This is not a bug, it’s how your ears work with sound pressure and perceived volume. Keyboard mappings for the same control deliver logarithmic performance but with obviously fewer steps in granularity of control.