

Falcon BMS Key File Manual

Introduction:

I found it always pretty hard to go through the key settings to find a specific function. In most cases it's due to lack of clarity and a missing segmentation. For this reason I decided to create a new keystroke file from scratch. Many did it before, but the output was not really suitable for me (and maybe others). But while continuing my work I realized, that there are even more things to do. E.g. the keyboard layout.

Since the beginning of community modded Falcon versions we are using more or less the same key settings. If you take a closer look to the keyboard layout (even in BMS) you may realize, that the settings are not very logical.

When BMS was introduced to the public it became soon clear, that many things regarding to the game itself have changed. I don't mean the "look and feel" of this software. The difference is: BMS has in opposite to other current Falcon versions a real future. And so I decided to revamp the key layout as well.

The main idea behind this project was to categorize the key file by using „headlines“ and strict naming conventions. It should be easily possible, to navigate through the key file and to find things quick. Therefore the description of each callback is following some rules which will be described below.

The keyboard layout has changed in many ways and is not comparable to any other versions before. If you expect to use it without any work which have to be done by yourself, you are definitely on the wrong track.

Because of there's a lot of hard work (mostly brain work) and testing in it (every callback is tested properly) I think it's worth to share it.

I hope, you'll enjoy it.

Kolbe

Update 1, December 2012

This is the first version of my updated manual. It is most likely that it will be revised in the future.

Falcon BMS Key File Manual

Index:

Introduction:	1
Categories & Sections:	3
The Terms:	5
1. Term:	5
2. Term:	6
3. Term:	6
Upper case vs. lower case:	8
Keys vs. Mouse (right / left click, scroll wheel):	8
Overview categories & sections:	9
Key files - general info:	10
Deprecated & outdated / Dev callbacks:	11
TrackIR & TeamSpeak:	11
The key files:	11
How to edit key files:	12
Editing key files in BMS UI:	12
Editing key files with Excel:	12
Using an editor:	12
How key files work:	13
How to avoid multiple key assignments:	20
How to implement DX code lines into the key file:	21
How to find out your DX device order and DX button IDs:	21
DirectX Assignment via BMS UI:	23
DirectX Assignment via Excel:	24
Manually editing DirectX Code Lines:	24
DX button ID limitation:	27
How to change DX shift magnitude in the falcon bms.cfg:	28
Activate Shifting:	29
DirectX device specifics:	29
Key file options & specifics:	31
Changing ICP-Numpad mapping (1=7 -> 7=1):	31
How to change the DX POV functions (Trim vs. View & other functions):	32
Assigning keys to Extra MFDs (3 rd & 4 th MFD):	33
(Pretty) Screenshot vs. PrtScr:	33
Double entries:	34
Keyboard keys and combinations you have to be careful with:	35
Why we don't hear cockpit sounds when pressing a key:	35
Troubleshooting:	36
Finally:	37

Falcon BMS Key File Manual

Categories & Sections:

I divided the key file into different categories and sections. I have to admit, it was also 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.

The categories and sections are easily recognizable in the code:

Category example (Key file content / UI output):

SimDoNothing -1 0 0FFFFFFF 0 0 0 -1 " LEFT CONSOLE"



Section example (Key file content / UI output):

SimDoNothing -1 0 0FFFFFFF 0 0 0 -1 "===== TEST PANEL ====="



Each description between the quotes has a length of 37 characters and as much "=" as possible.

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

In the key file itself I used 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 out commented. Unfortunately you will lose all out commented stuff (with a # at the beginning) once you saved the file in the UI. So it is recommended to keep a backup of the original file.

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

Falcon BMS Key File Manual

Non categorie & section lines in the UI:

Non changeable keys example:

SimDoNothing -1 0 0FFFFFFF 0 0 0 -0 "BMS Keystrokes Ver. 1.3 - Pit"

No Key Assigned	BMS 4.32: Keystrokes Ver. 1.3 - Pit
-----------------	-------------------------------------

Changeable keys example:

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

No Key Assigned	TQS: ANT ELEV Knob - Tilt Up
-----------------	------------------------------

Invisible keys example:

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

<- This is a screenshot. Believe me ☺

The UI will look like this:

CURRENT KEYFILE BMS_KEYSTROKES_VER_1_5_FULL	
KEY	MAPPING
No Key Assigned	BMS Keystrokes Ver. 1.5 - Full
	LEFT CONSOLE
	===== TEST PANEL =====
Shift+Alt F1	TEST: FIRE&OHEAT DETECT Button - Hold
Shift+Alt F2	TEST: OXY QTY Switch - Hold
Shift+Alt F3	TEST: MAL & IND LTS Button - Hold
No Key Assigned	TEST: MAL & IND LTS Button - Release
Shift+Alt F5	TEST: PROBE HEAT Switch - Toggle Up
Shift+Alt F4	TEST: PROBE HEAT Switch - Toggle Down
No Key Assigned	TEST: PROBE HEAT Switch - ON
No Key Assigned	TEST: PROBE HEAT Switch - OFF
No Key Assigned	TEST: PROBE HEAT Switch - TEST
Shift+Alt F6	TEST: EPU/GEN Switch - Hold
Shift+Alt F7	TEST: FLCSS PWR TEST Switch - Hold
	===== FLT CONTROL PANEL =====
Shift+Alt F9	FLT: DIGITAL Switch - BACKUP

Falcon BMS Key File Manual

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. Also the mentioned “outdated” and “not implemented” callbacks are not to find in this key file.

But there are some differences. I 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 at the end of this manual.

The Terms:

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

The description was 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 doesn't 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

1. Term:

As mentioned above the 1. Term is a short form of a section name. Sometimes it's easily done. For the TEST Panel for example it is just TEST. But you have some sections, where you have to be careful with names giving. 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 not 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 also possible, only to look at the first term while scrolling down the key file in the UI. You will easily find what you need.

2. Term:

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

Examples: HSI CRS, 2-ALLOW, 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

3. 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 the different functions I used for their description 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)

Falcon BMS Key File Manual

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).

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

This is meant to toggle between 3 or more states of a switch, function etc. Toggling up brings you to the last state and ends there. Vice versa for toggle 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 begins automatically at 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 pit builders play yard. 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.

Falcon BMS Key File Manual

Sometimes I 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)

U (Up – lol, just kidding... ☺)

Upper case vs. lower case:

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

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

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

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

All other words only begin with an upper case. (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 cases.

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 need only the left mouse button.

Falcon BMS Key File Manual

Overview categories & sections:

Category	Section	Short form
LEFT CONSOLE	TEST PANEL	TEST
	FLT CONTROL PANEL	FLT
	MANUAL TRIM PANEL	TRIM
	FUEL PANEL	FUEL
	AUX COMM PANEL	AUX
	EXT LIGHTING PANEL	EXT
	EPU PANEL	EPU
	ELEC PANEL	ELEC
	AVTR PANEL	AVTR
	ECM PANEL	ECM
	ENG & JET START PANEL	ENG
	AUDIO 2 PANEL	AUDIO2
	AUDIO 1 PANEL	AUDIO1
	UHF PANEL	UHF
	MPO PANEL	MPO
	LEFT SIDE WALL	LEFT WALL
	SEAT	SEAT
	THROTTLE QUADRANT SYSTEM	TQS

Category	Section	Short form
LEFT AUX CONSOLE	ALT GEAR CONTROL	ALT GEAR
	TWA PANEL	TWA
	HMCS PANEL	HMCS
	CMDS PANEL	CMDS
	GEAR PANEL	GEAR

Category	Section	Short form
CENTER CONSOLE	MISC PANEL	MISC
	LEFT EYEBROW	EYE
	TWP	TWP
	LEFT MFD	LMFD
	ICP	ICP
	MAIN INSTRUMENT (CP)	MAIN
	INSTR MODE PANEL	INSTR
	FUEL QTY SEL PANEL	QTY
	RIGHT MFD	RMFD

Category	Section	Short form
RIGHT CONSOLE	SNSR PWR PANEL	SNSR
	HUD	HUD
	LIGHTING PANEL	LIGHT
	AIR COND PANEL	AIR
	ZEROIZE PANEL	ZERO
	AVIONICS POWER PANEL	AVIONICS
	OXYGEN PANEL	OXY
	FLIGHT STICK	STICK

Falcon BMS Key File Manual

Category	Section	Short form
MISCELLANEOUS	OTHER COCKPIT CALLBACKS	CKPIT
	KEYBOARD FLIGHT CONTROLS	FCTRL
	SIMULATION & HARDWARE	SIM
	WINAMP	WINAMP

Category	Section	Short form
VIEWS	VIEW GENERAL CONTROL	VIEWGEN
	VIEW INTERNAL	VIEWINT
	VIEW EXTERNAL	VIEWEXT

Category	Section	Short form
RADIO COMMS	GENERAL RADIO OPTIONS	RADIO
	AWACS COMMS	AWACS
	ATC COMMS	ATC
	TANKER COMMS	TANKER
	WINGMAN COMMAND	WINGMAN
	ELEMENT COMMAND	ELEMENT
	FLIGHT COMMAND	FLIGHT
	FAC RADIO CALLS	FAC

Category	Section	Short form
DirectX Note: Not visible in UI	HOTAS UNSHIFTED	
	HOTAS SHIFTED	
	LEFT MFD	
	RIGHT MFD	

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 I realized soon, that there are a lot of free keys to map other callbacks.

So I decided to remap almost everything.

Important callbacks which you will need often are easy 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.

There are a lot of functions where no keys are assigned. These are mostly the functions which are relevant for pit builders e.g. full state callbacks.

Falcon BMS Key File Manual

Deprecated & outdated / Dev callbacks:

I am not using a single old callback with my key files, as they will be removed completely in the future. Also all debug and testing callbacks are not implemented into my key files. If you still (or want to) use them, please refer to RedDogs BMS Callbacks reference.

TrackIR & TeamSpeak:

A common weakness of any keyboard layout is the fact, that they don't incorporate 3rd party software like TrackIR or TeamSpeak. This is not an exclusive issue with Falcon keystrokes only.

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 are working.

Regarding VoIP software like TeamSpeak (should be the most common) you don't have default keys for PTT or broadcast functionality. These keys have to be set manually in the TS UI. Nonetheless I decided to implement them into my keyboard layout. You can see the key settings in my layout as suggestions. Of course you can map them to any key you want.

Except of TrackIR reload and TrackIR Recenter all keysettings 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.

The key files:

There are four different key files which use the same layout. They (except Blank & Stock) are using the key settings which can be found in the keyboard layout files.

1. BMS_Keystrokes_Ver_1_5_Full

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

2. BMS_Keystrokes_Ver_1_5_Pitbuild

This version is for pit builders. All toggle / cycle callbacks for switches / knobs which have full state callbacks are removed.

3. BMS_Keystrokes_Ver_1_5_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.

4. BMS_Keystrokes_Ver_1_5_Blank

This is the Full version but with no keys assigned, except the comms menus (Tanker, AWACS...), PrtScr (hardcoded Screenshot), UI Comms F1 & F2 (hardcoded) and Exit Sim. All other callbacks are set to visible / changeable.

Falcon BMS Key File Manual

5. BMS_Keystrokes_Ver_1_5_Minimum

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

6. BMS_Keystrokes_Ver_1_5_Stock

This key file uses the key settings of the stock BMS.key file. But be advised: ALL outdated & deprecated callbacks have been removed!

How to edit key files:

Editing key files in BMS UI:

It is not recommended to change key assignments in setup UI in BMS.

First all comment lines will be deleted after saving the file (code lines / remarks beginning with #). Secondly all "SimDoNothing" callbacks (and some others) will be changed into "SimRadarNextTarget".

So saving in BMS UI can and will damage your key files. Using an editor for that work is the best choice, even it is inconvenient.

Editing key files with Excel:

I've created new Excel files (BMS-Keyfile-Generator & BMS-DX-Generator) based on the idea of the original Keyfile-generator.

It incorporates the most important DX and key file stuff and an overview of nearly all callbacks, logically sorted by categories and sections. You can edit almost everything.

The BMS-Keyfile-Generator comes as a plain excel file and a more advanced one which uses VBA code.

The files (ver. 1.5) are still work in progress. Just follow the instructions.

Using an editor:

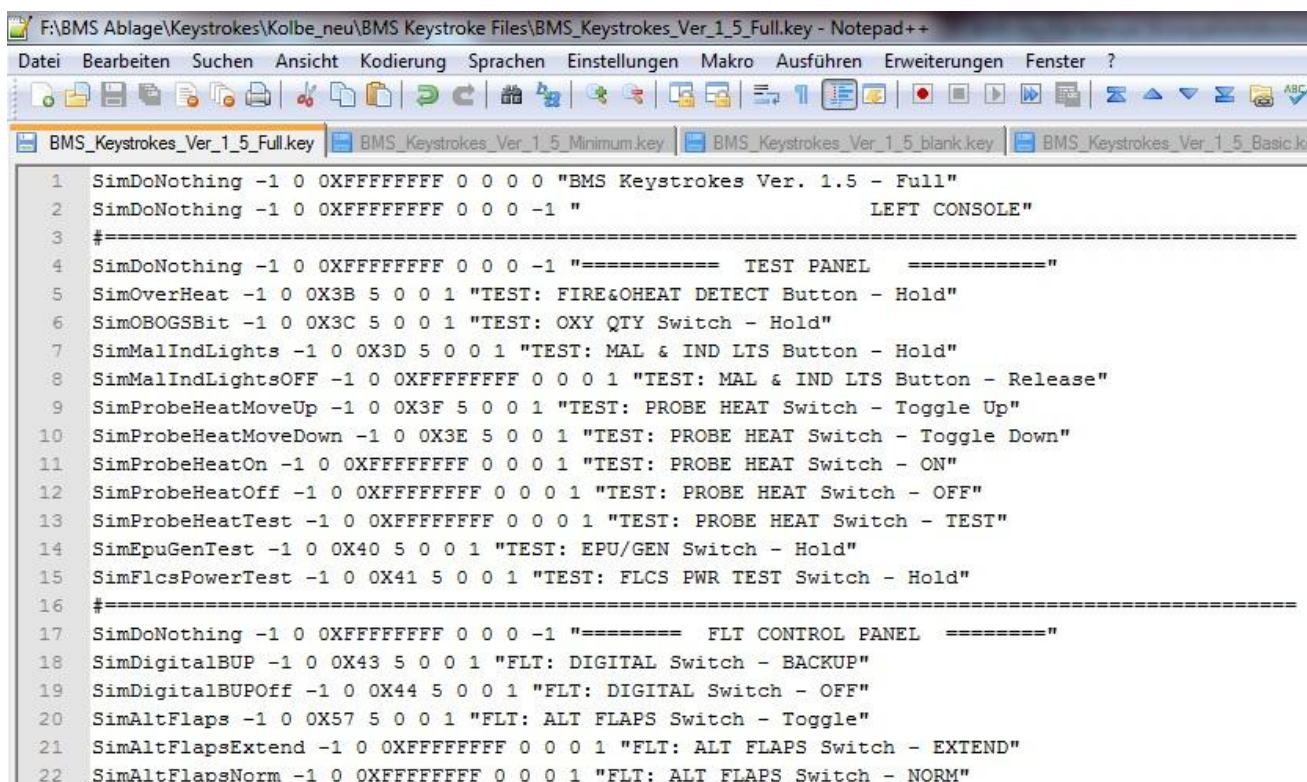
As mentioned earlier it is not recommended to change the key settings in BMS setup. It might work but can also damage your key file.

You can edit the key files with the standard Windows editor or any other comparable program. I suggest using Notepad++, IMHO the best freeware editor.

It can be downloaded [here](#).

Falcon BMS Key File Manual

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



```
1 SimDoNothing -1 0 0XFFFFFFF 0 0 0 0 "BMS Keystrokes Ver. 1.5 - Full"
2 SimDoNothing -1 0 0XFFFFFFF 0 0 0 -1 "LEFT CONSOLE"
3 #=====
4 SimDoNothing -1 0 0XFFFFFFF 0 0 0 -1 "===== TEST PANEL ====="
5 SimOverHeat -1 0 0X3B 5 0 0 1 "TEST: FIRE&OHEAT DETECT Button - Hold"
6 SimOBGSBit -1 0 0X3C 5 0 0 1 "TEST: OXY QTY Switch - Hold"
7 SimMalIndLights -1 0 0X3D 5 0 0 1 "TEST: MAL & IND LTS Button - Hold"
8 SimMalIndLightsOFF -1 0 0XFFFFFFF 0 0 0 1 "TEST: MAL & IND LTS Button - Release"
9 SimProbeHeatMoveUp -1 0 0X3F 5 0 0 1 "TEST: PROBE HEAT Switch - Toggle Up"
10 SimProbeHeatMoveDown -1 0 0X3E 5 0 0 1 "TEST: PROBE HEAT Switch - Toggle Down"
11 SimProbeHeatOn -1 0 0XFFFFFFF 0 0 0 1 "TEST: PROBE HEAT Switch - ON"
12 SimProbeHeatOff -1 0 0XFFFFFFF 0 0 0 1 "TEST: PROBE HEAT Switch - OFF"
13 SimProbeHeatTest -1 0 0XFFFFFFF 0 0 0 1 "TEST: PROBE HEAT Switch - TEST"
14 SimEpuGenTest -1 0 0X40 5 0 0 1 "TEST: EPU/GEN Switch - Hold"
15 SimFlcsPowerTest -1 0 0X41 5 0 0 1 "TEST: FLCS PWR TEST Switch - Hold"
16 #=====
17 SimDoNothing -1 0 0XFFFFFFF 0 0 0 -1 "===== FLT CONTROL PANEL ====="
18 SimDigitalBUP -1 0 0X43 5 0 0 1 "FLT: DIGITAL Switch - BACKUP"
19 SimDigitalBUPOff -1 0 0X44 5 0 0 1 "FLT: DIGITAL Switch - OFF"
20 SimAltFlaps -1 0 0X57 5 0 0 1 "FLT: ALT FLAPS Switch - Toggle"
21 SimAltFlapsExtend -1 0 0XFFFFFFF 0 0 0 1 "FLT: ALT FLAPS Switch - EXTEND"
22 SimAltFlapsNorm -1 0 0XFFFFFFF 0 0 0 1 "FLT: ALT FLAPS Switch - NORM"
```

How key files work:

If you are not familiar with editing key files you should read the following explanations of the inner working of key files.

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.

Falcon BMS Key File Manual

The callbacks itself are hardcoded and cannot be changed by the user. They could only be replaced by other callbacks, which doesn't 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 my _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: 2D Pit ID number

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

The second part is outdated as it was used in 2D cockpits which we don't have anymore since BMS was released. In the stock BMS.key and keystrokes.key files you can often find 4-digit numbers instead of the -1.

Example:

SimCBEOSB_1L **1025** 0 0X2 6 0 0 1 "LMFD OSB-1"

The 4-digit numbers represent an ID of a button or switch in a 2D cockpit. So basically each button or switch in a 2D Pit had a 2D Pit ID number which is defined in the 16_ckpit.dat. It assigns mouse functions only and is not related to keyboard bindings, which work independently.

In our example the number **1025** is assigned to the OSB button 1 of the left MFD. It means that you can invoke the callback (SimCBEOSB_1L) by clicking with the mouse on the OSB-1 button in the 2D Pit. If you replace the number by -1 you will lose the mouse function while pressing the assigned key will still work.

But again: The 4-digit number is outdated and not used with the BMS 3D cockpit anymore.

But wait.

One more thing to say. The 4-digit numbers invoke cockpit sounds when pressing the assigned key. But this is described in depth in the chapter **Why we don't hear cockpit sounds when pressing a key**

✓ Third part: Not in use

AFGearToggle -1 **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 don't have an impact.

You should leave the **0** unchanged.

Falcon BMS Key File Manual

✓ Fourth part: Keyboard key

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

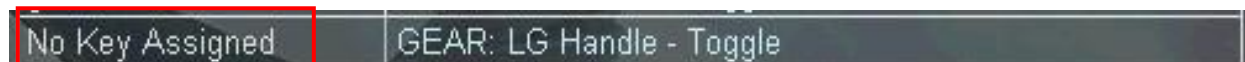


(0X22 = g)

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 0XFFFFFFFF.



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 don't use all possible keys as I want to make sure, the key files can be used by the majority of the community. The keyboard layouts around the globe are widely differing. So I concentrated on the most common keys.

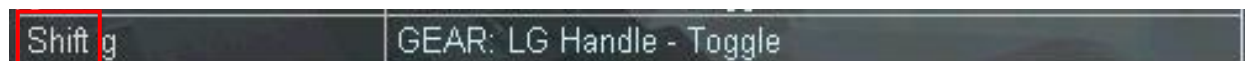
I don't want to list all key codes here as it would be a rather long list. Please refer to my BMS Keyboard Codes documents (can be found in the keyboard layouts folder) for more information.

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 I would recommend staying with a few common keys as I 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.

Falcon BMS Key File Manual

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	Ctrl
2	Shift
3	Alt
4	Ctrl Shift
5	Ctrl Alt
6	Alt Shift
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!

On the contrary 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"

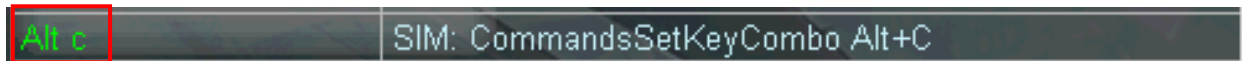
In my key files you have one code line which looks the same as the example above. So you assign a keyboard key + modifier key to the callback **CommandsSetKeyCombo** the same way, you normally would do with all other callbacks in a key file.

Falcon BMS Key File Manual

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 chance 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. So I use only one key combo with my key files (Alt + C).

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

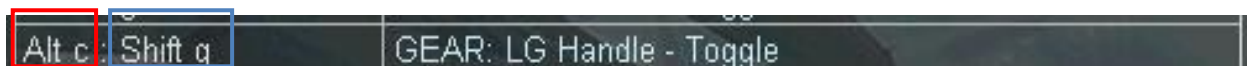
Key combo in relation with a single keyboard key:

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"

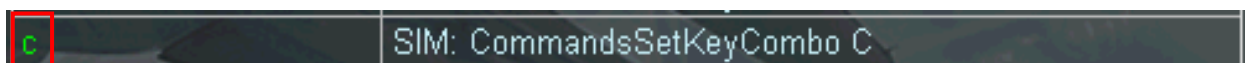


The following examples won't work:

Assigning a key combination key without a key combination modifier:

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

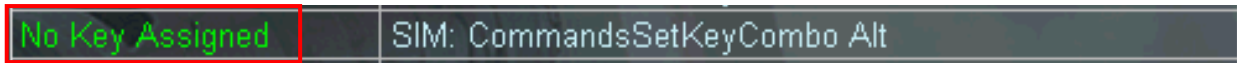
AFGearToggle -1 0 0X22 2 0x2E 0 1 "GEAR: LG Handle – Toggle"



Falcon BMS Key File Manual

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 0 1 "GEAR: LG Handle – Toggle"

A key combination modifier has the same syntax as normal modifier (see explanations above).

✓ Eighth part: UI visibility

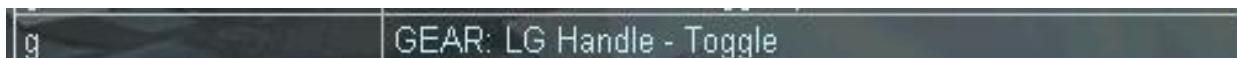
AFGearToggle -1 0 0X22 0 0 0 0 1 "GEAR: LG Handle - Toggle"

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

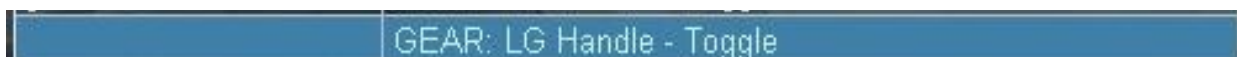
Example: Visible

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



Example: Headline

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



Falcon BMS Key File Manual

Example: Visible locked

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



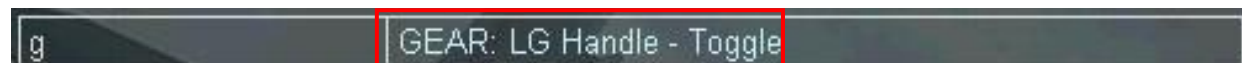
Example: Hidden (no screenshot here, of course)

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

✓ Ninth part: UI description

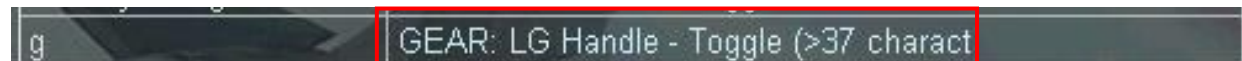
AFGearToggle -1 0 0X22 0 0 0 1 "GEAR: LG Handle - Toggle"

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



When using more than 37 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 (>37 characters)"



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 37 characters"
```

Falcon BMS Key File Manual

Note: If you mark a value by double clicking on it (here line 18 value 0X22, dark green) accordances in the file will also be shown (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.

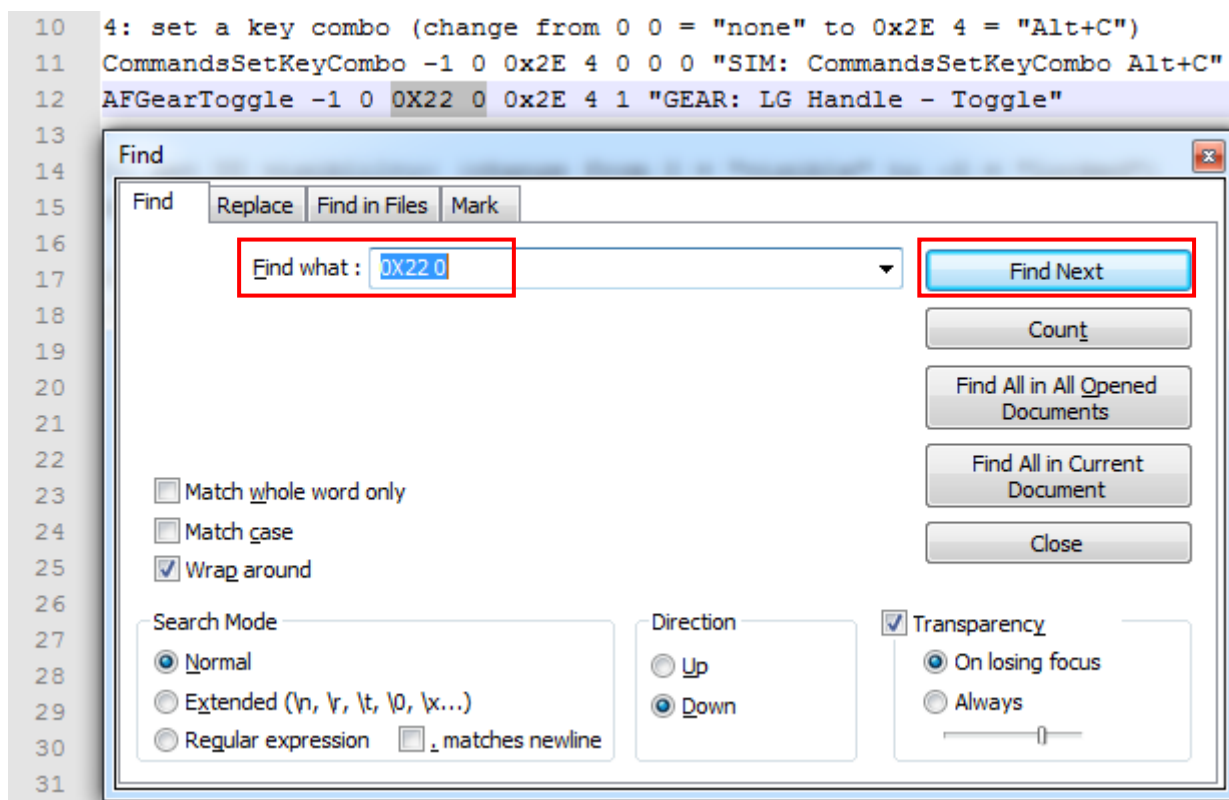
While doing that you will face the difficulty to avoid double assignation of one and the same keys.

Unfortunately you can't check for double assignation in the BMS UI. So you need other tools than that. I 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.



Falcon BMS Key File Manual

If you found a match you should change one of the both 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.

How to implement DX code lines into the key file:

The device specific DirectX part was excluded from this manual. I decided to implement the info into different, device related documents. You can find them in the different device folders. As of now only two devices (HOTAS Cougar and TM Cougar MFDs) are implemented. It is planned to cover the most common HOTAS systems in the future.

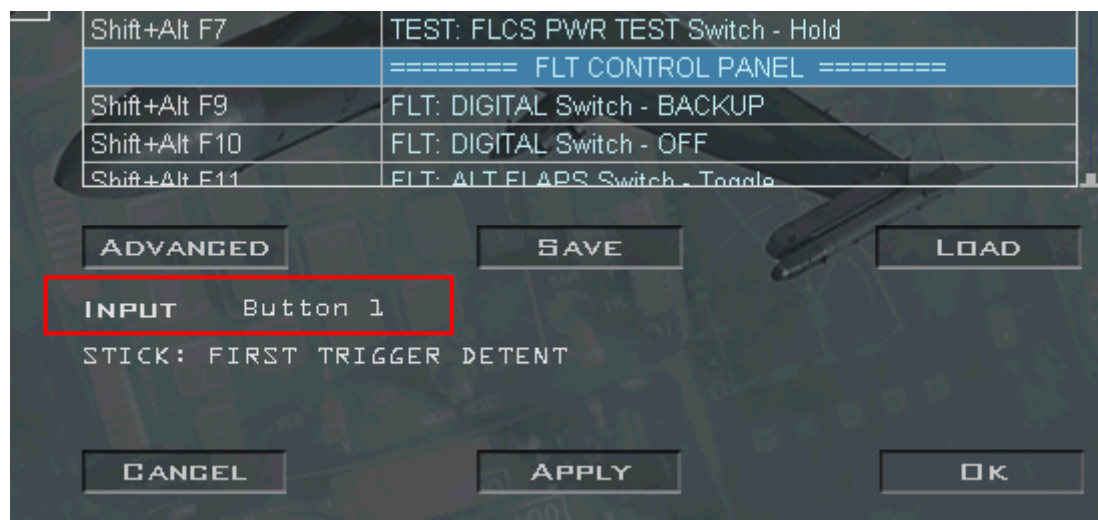
How to find out your DX device order and DX button IDs:

To make the devices work properly it is necessary to know the order of the installed DX devices.

I have to admit, in the first version of my updated manual I showed a fully illustrated way on how to find that out with the help of Windows. Things like Control Panel, Devices and Printer, Game Controller Settings. Five wasted pages...☺

There is much easier way to find that out. In fact with the help of BMS.

Just enter the BMS UI and enter Setup – Controllers page. Now we need to press only one button per device.



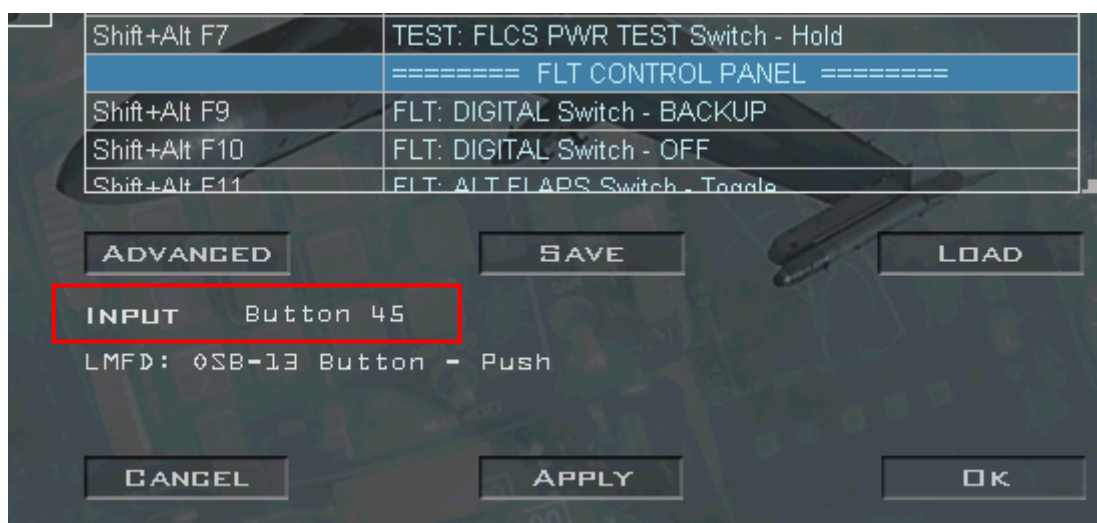
In the above image you see INPUT and Button 1 to the right.

Button 1 means in this context DirectX button 1. The DirectX button numbers in the BMS UI are shown as Windows DX numbers. From reading the [DirectX Shifting Facility Article](#) you should remember that BMS counts DX button numbers slightly different than Windows. You haven't read it yet? No Problem, I will explain it. Win DX # 1 = BMS DX # 0.

All clear? No? Too bad. Read the [article](#) ☺

Falcon BMS Key File Manual

So far so good. Let's try that again with our second controller:



Now we see Button 45, which means Win DX # 45 and BMS DX # 44.

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, your primary input device so to speak. If the button number is in the range of 33 – 64 it is device # 2 and so on.

So in our example above we now know that the first device is the one with the button # 1 and the second with the button # 45.

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.

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...		

So why do we need to know the numbers when BMS shows them well in the UI. That's because the BMS numbers are used in the key file. So if the UI says INPUT Button 1 the same device button will be shown as DX number 0 in the key file.

On the other hand we need these numbers to calculate the DX button IDs for the shifted layers. They have to be calculated manually or by using my Excel BMS DX-Generator.

But more about this later.

Falcon BMS Key File Manual

DirectX Assignment via BMS UI:

We can easily assign DX functions to our devices via the BMS UI. To give you an example I've 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"
```

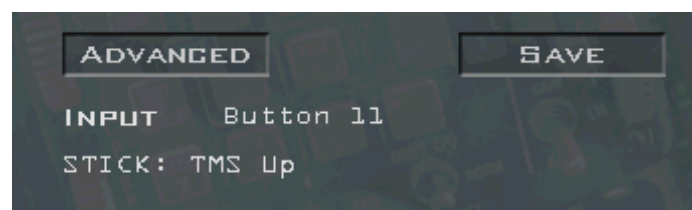
Once loaded in the UI the same key file looks like this:

CURRENT KEYFILE CALLBACKTEST	
KEY	MAPPING
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

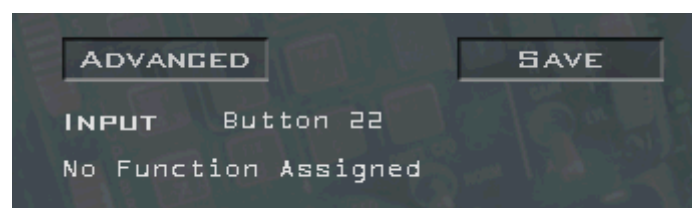
What we need to do now is to left click on a row we want to assign a DX button to. The key of this row will be shown in blue.

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'll 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'll 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.

Falcon BMS Key File Manual

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 0FFFFFFF 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"
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 "8".

DirectX Assignment via Excel:

I've created an Excel file where you can easily assign callbacks to your DX device. Just follow the instructions given in the separate sheets.

This file is still WIP and will be updated in the future to cover all common HOTAS devices. Until now it only supports TM HOTAS Cougar and TM Cougar MFDs.

Manually editing DirectX Code Lines:

Before starting to describe, how to manually edit the DX code lines I assume you have read and understood the [DirectX Shifting Facility Article](#). I won't explain how shifting works in general.

A typical key file DX code line looks like this:

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

Each code line is composed of six different parts.

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 section "How key files work" 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.

Falcon BMS Key File Manual

✓ 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 doesn't 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: Not used

SimTriggerFirstDetent 0 -1 -2 0 0x0 0

Back in old days you could decide to use the binding as DX button (value -2) or as a POV hat (value -3). But this isn't implemented in BMS so you should leave the default value -2 unchanged.

✓ 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.

Falcon BMS Key File 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): The description

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

The description doesn't 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.

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.

Falcon BMS Key File Manual

DX button ID limitation:

BMS can handle max. 512 DX buttons. This means either 16 separate devices without shifted layers, 8 devices with shifted layers for each device or more than 8 DX devices but in this case with a proportional reduced number of shifted layers, e.g. 12 DX devices (8 without shifted layers, 4 with shifted layers).

To clarify that here some examples:

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

# 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

Note: This is the default configuration. If you don't have more than 8 different DX devices (and most of us won't) you don't need to worry about other settings.

✓ Second example: 16 DX devices without a shifted layer

# of device	WIN DX #	BMS DX #	BMS shifted
Device 1	1 ... 32	0 ... 31	n/a
Device 2	33 ... 64	32 ... 63	n/a
Device 3	65 ... 96	64 ... 95	n/a
Device 4	97 ... 128	96 ... 127	n/a
...
Device 13	385 ... 416	384 ... 415	n/a
Device 14	417 ... 448	416 ... 447	n/a
Device 15	449 ... 480	448 ... 479	n/a
Device 16	481 ... 512	480 ... 511	n/a

Note: If you have 16 different DX devices you won't have shifted layers for them. Additionally you have to change the value for **set g_nHotasPinkyShiftMagnitude** in the falcon bms.cfg to **0**. How that works is described later on.

Falcon BMS Key File Manual

- ✓ Third example: 12 DX devices (8 without a shifted layer, 4 with a shifted layer)














# of device	WIN DX #	BMS DX #	BMS shifted
Device 1	1 ... 32	0 ... 31	384 ... 415
Device 2	33 ... 64	32 ... 63	416 ... 447
Device 3	65 ... 96	64 ... 95	448 ... 479
Device 4	97 ... 128	96 ... 127	480 ... 511
Device 5	129 ... 160	128 ... 159	n/a
...
Device 10	289 ... 320	288 ... 319	n/a
Device 11	321 ... 352	320 ... 351	n/a
Device 12	353 ... 384	352 ... 383	n/a

Note: The 12 devices use the BMS DX IDs 0 – 383. So in this example the shifting begins at 384. We have to change the value for the shift magnitude (see below) to 384. Because of the DX button limitation we can only have four shifted layers, but only for the first four devices. All devices after #4 would require a higher DX button number than 511, which is not possible with BMS.

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

	axismapping.dat	26.11.2012 14:47	DAT-Datei	1 KB
	BMS	13.06.2012 16:18	KEY-Datei	43 KB
	BMS_Keystrokes_Ver_1_5_Full	09.12.2012 20:32	KEY-Datei	74 KB
	Callbacktest	20.12.2012 19:07	KEY-Datei	16 KB
	default.rul	02.01.2011 19:32	RUL-Datei	1 KB
	DeviceDefaults	02.07.2011 08:56	Textdokument	4 KB
	dx9display.dsp	20.12.2012 19:14	DSP-Datei	1 KB
	ews_def	19.10.2010 23:26	Konfigurationsein...	1 KB
	Falcon BMS Editor.cfg	26.11.2012 15:37	CFG-Datei	1 KB
	falcon bms.cfg	03.12.2012 21:09	CFG-Datei	13 KB
	feedback	19.10.2010 23:26	Konfigurationsein...	4 KB
	files.dir	19.10.2010 23:26	DIR-Datei	7 KB
	joystick.cal	26.11.2012 14:47	CAL-Datei	1 KB

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

Falcon BMS Key File Manual

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
```

or to any value desired (e.g. **384** like in our example above).

```
86 set g_nHotasPinkyShiftMagnitude 384
```

Activate Shifting:

To activate shifting you have to use the callback SimHotasPinkyShift. Just assign it to a button on your joystick (in most cases the pinky button but depends on the input device) and make sure to use this callback for the shifted DX button ID as well (DX button number plus offset defined in the code line “set g_nHotasPinkyShiftMagnitude”)

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

The shifting applies to ALL DX devices simultaneously.

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 device specifics:

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.

Falcon BMS Key File Manual

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.

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.

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 keyup/down behaviour

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

0x42 = 0x42 invokes the callback when the DX button is released

Falcon BMS Key File Manual

To keep the story short.

With these both 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  
DX button press event -> SimRFNorm
```

```
SimRFQuiet 58 -2 -2 0x42 0x0 0  
DX button release event -> SimRFQuiet
```

```
SimRFSilent 59 -2 -2 0 0x0 0  
DX button press event -> SimRFSilent
```

```
SimRFQuiet 59 -2 -2 0x42 0x0 0  
DX button release event -> SimRFQuiet
```

You can do this with every DX button #. Of course this is not recommended and not useful in all occasions.

Key file options & specifics:

Changing ICP-Numpad mapping (1=7 -> 7=1):

In my 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-ALOW"  
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"
```

Falcon BMS Key File Manual

How to change the DX POV functions (Trim vs. View & other functions):

In my key files the unshifted POV acts with its default behavior (views). The TRIM functions are set to the shifted layer. Here is how you can change that:

Set TRIM to unshifted layer:

Delete the following lines in 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
```

Copy the following lines into 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
```

Set Views to shifted layer:

Copy 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
```

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.

Falcon BMS Key File Manual

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**

Assigning keys to Extra MFDs (3rd & 4th MFD):

I didn't assign any keys to the 3rd and 4th MFDs because the situations where you would use them are pretty rare. As far as I know 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. I included the DX code lines into my DirectX TM Cougar MFD.pdf.

(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 doesn't 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)

Falcon BMS Key File Manual

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.

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's 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).

Falcon BMS Key File Manual

Keyboard keys and combinations you have to be careful with:

Alt Tab:

Should be well known but it's not a mistake to mention it here.

Don't assign a function to Alt + TAB. This will bring you back to desktop (If you have luck) or could crash / freeze BMS. Both unwanted actions while enjoying a flight...

Escape:

It can't be assigned in the BMS UI. You have to do that manually if desired. But don't 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)

Num Lock:

If you assign the Num Lock key it will toggle between "Disable Num Block" and "Enable Num Block". It has no impact on BMS as the assigned functions on the Num Block work no matter how the Num Block status is. But be aware of it when working with other applications after a flight.

Print:

See **(Pretty) Screenshot vs. PrtScr** above.

Why we don't hear cockpit sounds when pressing a key:

As described above the 2D Pit ID numbers are old remaining from "2D Pit days". As we don't have 2D Pits anymore we don't need them nowadays. On the other hand this statement is partly wrong as the 4-digit numbers can invoke cockpit sounds when pressing the assigned key. The 2D Pit ID numbers are assigned in the 16_ckpt.dat.

In 3D Pit we have a total of 29 different sounds which are assigned to the different knobs, buttons, switches etc in the cockpit. The sounds are assigned in the 3dbuttons.dat. So all of the sounds work when we are using the mouse in the cockpit.

Unfortunately only 12 different sounds work with the 4-digit 2D Pit ID numbers. But only 10 of them are among the 29 needed sounds for a full implementation of the sounds invoked by the 4-digit IDs. In short, we can only assign 10 out of 29 sounds by using the 4-digit numbers in our key file.

That's the reason why I decided not to use any of the working 4-digit numbers. It is incomplete and it would sound odd if you hear different sounds when using the keyboard and the mouse for the same cockpit function.

Falcon BMS Key File Manual

The following list shows the working 4-digit numbers and how they sound like. If you want to hear sounds when pressing a key for a specific cockpit function you have to implement them manually.

<u>2D Pit ID</u>	<u>Sounds like (example)</u>
1001	MFD OSB Button
1005	Inst. Mode Knob
1007	Landing Lights
1008	Master Caution
1009	Emerg. Jettison
1011	Oxygen Supply
1020	ENG Feed Knob
1074	Eject Handle
1075	Gear up
1112	ICP Buttons
1077 / 1210	Unknown

In fact, there are of course a lot of more 4-digit numbers (on or about 240) which have sound assignments. But several numbers are assigned to the one and the same sound. The 2D Pit IDs in correlation with the sounds above are listed in order of their first appearance.

At this point I would like to mention that there is a mod with a solution to overcome the sound limitation. You can find it [here](#). But use it on your own risk.

Troubleshooting:

Why does BMS crash when loading a key file?

Check your key file for syntax errors. As we don't 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's 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.



Falcon BMS Key File Manual

The mouse doesn't 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.

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 doesn't 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.

Finally:

So here we are. I hope you find my work useful. You might also take a look at the keystrokes layout documents. Of course you are free to change both of them to your own liking. If you have any suggestions or found mistakes, please contact me.

My work is not yet finished. So it is likely that it will be updated in the near future. Updates and news are announced and discussed in the following forum thread:

[Forum Link](#)

I apologize for any mistakes (maybe phrase or spelling) I made. English is not my native language. But I gave my very best ☺

Thanks to DragonFly (49th) and Dunc (BMS) for reviewing this document and the key files.

Have fun...

Kolbe