Thrustmaster TARGET Profile for Train Sim World

Introduction

This document explains how to use the included Thrustmaster TARGET script with Train Sim World (TSW).

There are a lot of game controllers such as HOTAS throttles or generic throttle quadrants which would make great controllers for Train Sim World (TSW). But TSW does not support use of DirectX Game controllers other than the Xbox controller. In order to map a generic game controller to run TSW, you must use axis to keyboard mapping, where moving a game controller axis results in pressing keyboard keys which manipulate the locomotive levers in TSW. For reasons explained in the technical details later in this document, this is significantly more difficult than it might seem. The included Thrustmaster TARGET script, however, accomplishes the task with great accuracy.

The script is designed mainly to use a Thrustmaster Warthog Throttle to manipulate the locomotive control levers and some buttons and switches. The Thrustmaster TWCS is also configured in the script to act as the Dynamic Brake for most locomotives, but this is not required if you only have a Warthog Throttle. You can easily modify the script to work with any TARGET compatible devices. The Warthog is the best Thurstmaster device to use because it has 3 lever type axes, but the TWCS Throttle or the HOTAS Cougar Throttle could also be used exclusively. You would just have to get creative on how you mapped the one throttle axis. More is explained about that in the Technical Details section.

The script provides the following features:

- Warthog Right Throttle axis controls the engine throttle or throttle/brake lever depending on the engine type.
- Warthog Left Throttle axis controls the dynamic brake, automatic brake, and independent brake, depending on the setting of the FLAPS switch. It is easy to use all three modes on one lever due to a memory feature that remembers the position of each brake lever in TSW so you can quickly move the Warthog axis to match the lever in TSW without getting out of sync.
- Warthog Friction Control axis controls the reverser
- TWCS Throttle axis controls the dynamic brake on most engines. Use of the TWCS is optional.
- Many commonly used functions (e.g. horn, bell, windshield wipers, etc.) are mapped to buttons and switches on the Warthog Throttle.
- All engines supported in a single script with the ability to change engine profiles without leaving TSW. The script remembers the last profile you used and starts on that profile when reloaded.
- Audible feedback is provided using beeps and spoken words. It is possible to customize the script further with text to speech, beeps, and playing .WAV files for audible feedback.
- A way to make TrackIR work is also offered using a FreePIE script. Use of this script is optional and not part of the TARGET Script.
- Profiles are provided to work with the Logitech/Saitek Throttle Quadrant as a convenience. These are optional and not part of the TARGET Script. They do not work very well because the Saitek Programming tool cannot do what TARGET can do, but used properly, they can be of benefit alongside the Warthog Throttle.

What's New in V3.1

See the file "Drakoz_TrainSimWorld-Release_Notes.txt" for complete details about updates.

Version 3.1 of the script has the following new features:

- Added the following locomotives
  - Amtrak ex-Metroliner Cab Car IVb
  - BR Class 31
  - BR Class 33
  - BR Class 37 (uses same profile as BR40)
  - BR Class 52
  - DB BR Class 155
  - DB BR Class 146.2 (uses same profile as DB BR 185.2)
  - DB BR 766.2 and 767.2 Control Cars
  - DB BR 1442-1 Talent 2
- Fixed an issue with locomotive selection when using the THR_FC lever. Now, THR_FC will be ignored until it is moved to the center first when changing profiles.
- Fixed a couple issues with the FLAPs switch when changing profiles. No longer need to cycle the FLAPs switch when changing profiles.
- Fixed direction of BR 101 Reverser control.
- Set Warthog Throttle Speed Brake switch to decrease the Sand Switch when pushed forward (so now SPDB = Sand, and SPDF = SanDecreasePress).
- Added Warthog Throttle LTB button to also press the AWS Reset function.
- For locomotives with a Brake Pin and Neutral/Shutdown position, wait for Pin to disengage before entering Neutral/Shutdown. This affects locomotives like BR 31, 33, 37, 40, 45.
- A few minor improvements to the axis mapping tables for various locomotives.

What's New in V3.0

Version 3.0 of the script has the following new features:

- Doubled the number of supported engines (and a few more coming soon).
- The common functions for all engines were mapped in a more consistent manner (e.g. wind shield wipers, cruise control, locomotive headlight/taillight control, cab or console lights, sand, etc.).
- The Thrustmaster TWCS Throttle has been added to the script to work in aid to the Warthog.
- The script was significantly modified to make it easier to add or change which game controller axis controls which levers in TSW. It is now very easy to change from using a Warthog to using a TWCS or HOTAS Cougar for example as well as combine axes for multiple Thrustmaster devices to work together.
- New audio feedback features were added including text to speech which is used to announce the profile that has been selected, and the ability to play .WAV files. The WAV file feature is disabled by default, but it is available for user customization.
- Engine profile can be selected up or down the list using two buttons (no more accidentally passing the profile and having to count through all profiles over again to get to the one you want), or quickly select the profile using the Warthog THR_FC lever so you don't have to press a button 20 times to get to a profile.

- The script uses a configuration file to save the last profile selected and will default to that profile next time the script is loaded.
- When selecting a locomotive profile, the order of the selection list is now sorted by world region (e.g. N. America, Britain, Germany) and then by locomotive type number (e.g. BR Class 40, 45, 47).
- Speed improvements made to make the axis to keyboard mapping more responsive (not as much lag).
- Added the ability to map an axis to output complex keyboard combos (e.g. SHIFT-A, CTRL-A, ALT-A). This was important to support cruise control features on the German locomotives.

Supported Locomotives

Locomotive support is regardless of DLC route. For example, the GP38-2 is used on the CSX Heavy Haul, NEC New York, and Canadian National Oakville DLC routes, but all are supported by the same engine profile in the TARGET Script.

- LIRR M7 EMU
- GP38-2, GP40-2, SD40-2 (GP9 should also work, but not fully tested)
- EMD F40PH-2CAT
- EMD F40PH-2CAT CAB CAR
- Amtrak ACS-64
- Amtrak SW1000R
- GE AC4400CW
- Amtrak ex-Metroliner Cab Car IVb
- BR Class 08
- BR Class 09
- BR Class 31
- BR Class 33
- BR Class 37
- BR Class 40
- BR Class 43 HST
- BR Class 45
- BR Class 47
- BR Class 52
- BR Class 66
- BR Class 101
- BR Class 166
- DB BR 143
- DB BR 155
- DB BR 182
- DB BR 146.2 and DB BR 185.2
- DB BR 766.2 and 767.2 Control Cars
- DB BR 1442-1 Talent 2

Adding Support for New Locomotives

Support for other locomotives depends on my time and funds to buy new DLC. Unfortunately, due to the nature of how this script works, and because most engines work differently, support for new engines must be added on a case by case basis.

Contact me and with your help, we can add support for DLC that I don't own. Or, if you know how to use TARGET scripting, it isn't difficult to add support on your own (see the Technical Details section to see how). I am willing to help either way, so don't hesitate to contact me for assistance. If you add support for an engine, please send me your results so I can release to everyone else.

If you want to support this effort financially, you can Paypal me at mike@akhara.com.

Donations are put back into buying more TSW DLC so I can support all the locomotives. But to clarify, I don't do this for money or recognition. I started out wanting to use a Warthog to control TSW. But I take the time to document this and publish it both because it greatly enhances the enjoyment of TSW, but also because this script demonstrates the power of Thrustmaster TARGET to those looking for ideas to use on their own scripts. If what I have done teaches others, then that is my best reward.

How to Customize the Script to Fit Your Tastes

It is not necessary to understand how to program TARGET scripts to use the script as it is, but it will help a great deal if customizing the script is desired. Read the Thrustmaster TARGET Scripting manual, and read this entire document, and you should be able to figure most things out well enough. This script is no different than any other TARGET script with respect to mapping buttons and switches. And since v3.0 of the script, it is easy now to change axes without understanding C programming or even much of the underlying keyboard mapping stuff. More will be explained in the Technical Details section below.

Installation

Installation is a manual process that isn't much different than creating your own TARGET script from scratch. Unpack the downloaded .ZIP file to a temporary folder and you will find the following files and foldes.

- *TARGET Script files*
  The Thrustmaster Warthog Throttle Script files. *Drakoz_TrainSimWorld.ttm* and *Drakoz_TrainSimWorld_Warthog.tmc*
- *Train Sim World – TARGET Script Doc.pdf*
  The document you are reading now.
- *TrainSimWorld_Generic_Warthog Controller Layout.xlsx*
  Excel file that shows the layout of all controls mapped in the TARGET script file. Also includes mapping for the Saitek Throttle Quadrant if you use those profiles.
- *wBeep files*
  The wBeep.exe program used in the TARGET script – for audible feedback when using the script.
- *TrackIR Support using FreePIE*
  A FreePIE python script and instructions to use FreePIE to get TrackIR support in TSW. See the README file for details on how to install it and use it. It is optional if you want to use this. Here's a video link that demonstrates how this works: https://youtu.be/V6_B484yDcM
- *Logitech-Saitek Throttle Quadrant Profiles*
  Profiles for the Logitech/Saitek Pro Flight Throttle Quadrant and Pro Flight Yoke (which has a Throttle Quadrant). It is optional if you want to use these.

To install the TARGET script…

Copy the Script files:

- Copy the *Drakoz_TrainSimWorld.ttm* and *Drakoz_TrainSimWorld_Warthog.tmc* files from "*Target Script files*" to where you normally store your TARGET script files. By default, this should be:
  c:\users\<your username>\AppData\Roaming\Thrustmaster\TARGET\Scripts
  The AppData folder is normally hidden. You can access it easily by hitting the Windows key and typing %appdata% <enter> (for Windows 7, 8, 10) and it will open the Appdata folder. Or it is OK to save the files anywhere as long as the TARGET Script editor can brose to them (see below).

Create the Sounds folder and copy wBeep.exe:

- Create the following folder:
  - C:\bin\Sounds\
- Copy wBeep.exe to this new Sounds folder.
  - If you don't want to use or install wBeep.exe, see *How to Disable Audio Features of the Script* toward the end of this document.
  - Note, for older versions of this script, you were instructed to copy wBeep.exe to C:\bin. I am changing that as of v3.0 of this script so that all sound related functions are now kept in the C:\bin\Sounds folder. This includes sWavPlayer.exe if you want to use that to play .WAV files. See below for more details about playing .WAV files. It is disabled by default, so you can ignore sWavPlayer.exe for now.

- If you want to change the location of the Sounds folder, see *How to Change the Location of the Sounds Folder* toward the end of this document.

Using the Logitech/Saitek Throttle Quadrant Profiles

- If you want to use the Logitech/Saitek profiles, see the README file in that directory for where to copy them. You only need one file or the other (the Pro Flight Yoke or Pro Flight Quadrant files) as each profile does the same thing. These profiles are easily modified through the Logitech Profile software.

To run the TARGET script…

- It is assumed you have already installed and set up TARGET and your Thrustmaster devices.
- Make sure your Thrustmaster HOTAS Warthog Throttle is plugged in. The Warthog Joystick is not needed, but it does not hurt to have it plugged in. The same is true for any other Thrustmaster TARGET compatible controllers. This script will ignore them by default.
- If you want to use the Thrustmaster TWCS Throttle, make sure it is plugged in as well, though use of the TWCS is optional. If not plugged in, it will not affect the script.
- Run the TARGET Script Editor (not the TARGET GUI).
- Select the TOOLS tab.
- Make sure FILE PATHS in TARGET points to where you copied the .tmc and .ttm files. If you copied the script files to a non-standard place, this is where you need to tell TARGET where they are.
    - Click on Options
    - Select "FILE PATHS" tab
    - Make sure the *Drakoz_TrainSimWorld.ttm* and *Drakoz_TrainSimWorld_Warthog.tmc* files are in a folder that is in the list of INCLUDE FILE LOCATIONS.
- Load the *Drakoz_TrainSimWorld_Warthog.tmc* file.
    - Click on MENU
    - Click on OPEN
    - Find the file "Drakoz_TrainSimWorld_Warthog.tmc" and open it.
    - The .ttm file will be loaded automatically for you when you run the script.
- Compile and run the script by clicking "RUN".
- If all was installed correctly, it should compile with no errors, and you should hear the Windows "USB device connected" sound followed by a beep. The Windows USB sound is because TARGET just created the Thrustmater Combined device. The beep is part of the script using the wBeep.exe program. If you do not hear a beep, recheck that you have copied wBeep.exe to C:\bin\Sounds (the script will not run without wBeep.exe unless you have disabled beeps) or check your audio settings. The beep sounds are part of the Windows "System Sounds" volume in the Windows Volume Mixer. This the same volume setting that controls the volume for all system sounds like alert notifications, hardware connected/disconnected sounds, etc. Make sure the System Sounds volume is high enough (this should be true by default).
- If it fails to run for some other reason, it is probably a TARGET Script Editor setup thing, or you forgot to plug in your Warthog. Check the error log in the TARGET Script Editor, resolve the problem and try again. If all else fails, ask for help in the forums where you downloaded this.
- The script is now running and ready to be used in TSW. Please see the section "Using the TARGET Script below before continuing.

A WARNING!!!! Once running, be careful about moving the axes on your Warthog Throttle when Train Sim World or the TARGET Script Editor are not the selected application as it will send lots of keystrokes to the computer with unintended consequences. To see what I mean, open Notepad and move the axis back and forth. You'll see a spew of characters show up on the screen in Notepad.

To Install the Logitech Profiles (optional)

- See the README file in the *Logitech-Saitek Throttle Quadrant Profiles* folder.
- These profiles are only needed if you have and plan to use a Logitech Throttle Quadrant or the Throttle Quadrant that came with a Pro Flight Yoke. They are functionally the same, so I created a version of the profile for each as I have both. I only use one or the other, not both at the same time.
- Again, these profiles were provided as a convenience because the Warthog Throttle must share the left lever to do 3 different brake modes on a locomotive. But using the Saitek Throttle Quadrant, you can control some of the braking there instead of using the FLAPS switch on the Warthog to change brake mode. But the Saitek profile doesn't work anywhere near as well as the TARGET Script. The Saitek profiles have nothing to do with the TARGET script otherwise.

Configuration File – Drakoz_TrainSimWorld_Config.cfg

The first time you change locomotive profiles in the script, a configuration file will be written to the same folder where the script is stored (file name: Drakoz_TrainSimWorld_Config.cfg). The file stores the locomotive you selected so that next time you run the script; it will default to the last locomotive you used.

If desired, it is OK to delete this file for whatever reason. It will be recreated the next time the script runs and you change profiles.

Using the TARGET Script – Summary

- Run the TARGET Script as explained above.
- Select the desired locomotive profile – See *"Changing the Locomotive Profile"* below for more details.
- Load Train Sim World and select a scenario or service.
- Before entering the scenario, set the axes and switches on the Warthog to match the current state of the engine.
- See the included Excel file for button and axes configuration.

Please see below for detailed usage instructions.

Function Reference – *TrainSimWorld_Generic_Warthog Controller Layout.xlsx*

Open the *TrainSimWorld_Generic_Warthog Controller Layout.xlsx* file which has a layout of the controller and how the buttons, switches, and axes are mapped. Here are some notes on how to read this file.

- Each button and axis on the Throttle is listed in the gray boxes.
- **Black** text lists the **primary function** of a button (e.g. push the APENG button and it blows the horn).
- **Red** text is a **shifted function** controlled by the Thrustmaster IO shift button. The shift button is MSP (Mic Switch Hat center button). E.g. press MSP and the LDGH button, and it will change to the next engine profile (more on engine profiles below).
- **Orange** text is a **Press and Hold function**. E.g. Press and hold the Mic Switch HAT forward, and it will switch to Free View (view 8 in TSW).
- **Blue** text indicates **mode based functions** that change based on the position of the **Flaps Switch**. This is the Thrustmaster UMD mode switch.
- **Light Blue** text is **setup notes** or additional functional notes related to the sim.
- The file also includes mapping information for the Saitek Throttle Quadrant and the Logitech G700S mouse which I use. You can ignore the G700S settings as that is not part of this collection of files.

Using the TARGET Script – In Detail

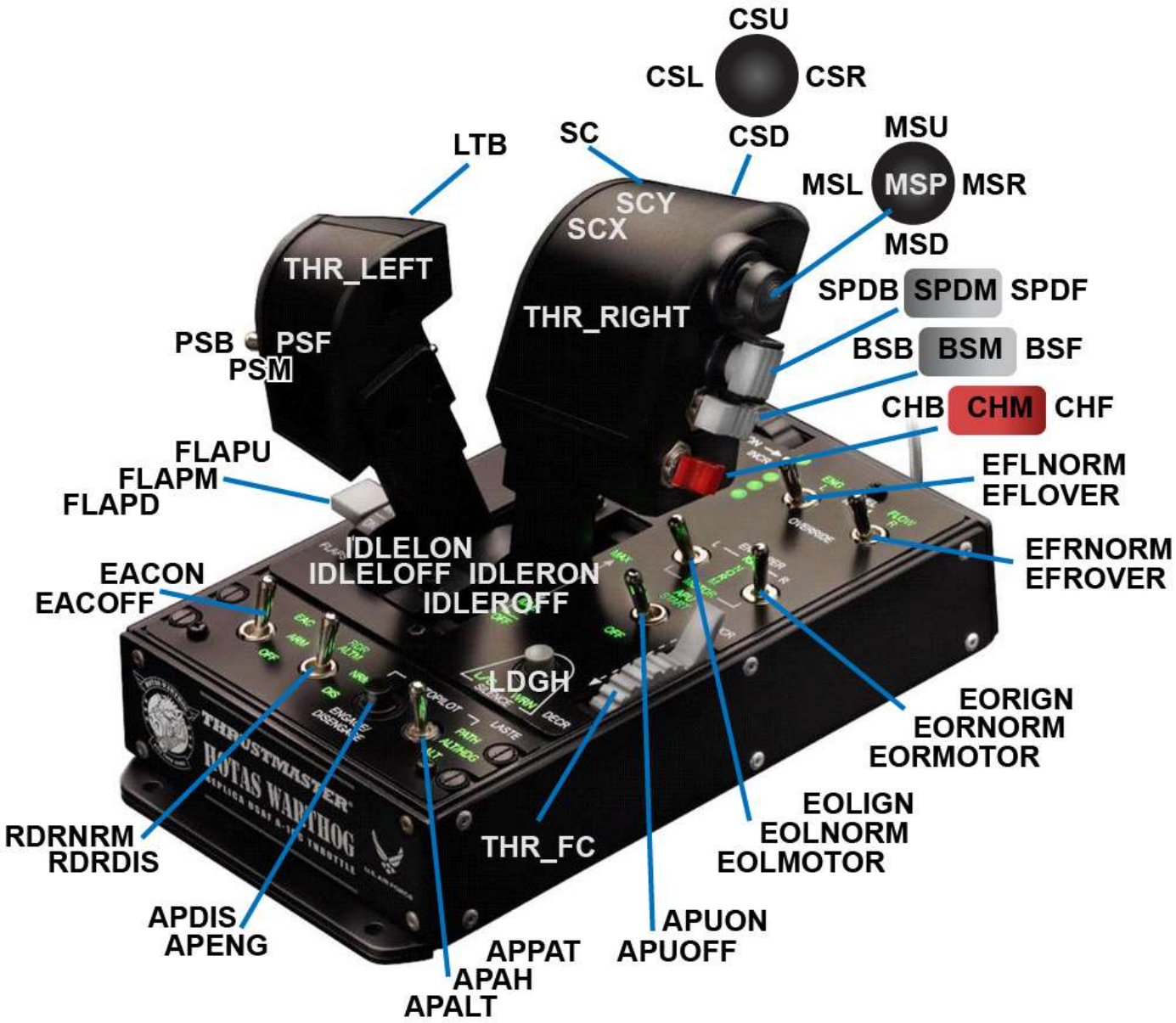
The sections below explain in detail how to use the TARGET Script. If you prefer watching videos instead of reading, you can watch the Youtube playlist linked below and things should make a lot more sense. Note, the early videos were made using the v2.0 script, but the method of changing profiles has been updated. So, at least read the section below about changing profiles.

https://www.youtube.com/playlist?list=PLqB6OrmhBjeGIxAuxg5x6TB4YHyA7xcHs

Below, there are sections that explain generally how to use the script (covering functions that are common for all locomotives). Following that are summary sections for each locomotive profile explaining specific details about each profile.

References are made to the names of buttons and axes on the Warthog and TWCS. Most of the names used are the default names as used in TARGET and are documented in the TARGET documentation. The pictures below are summaries of these axes and button names.

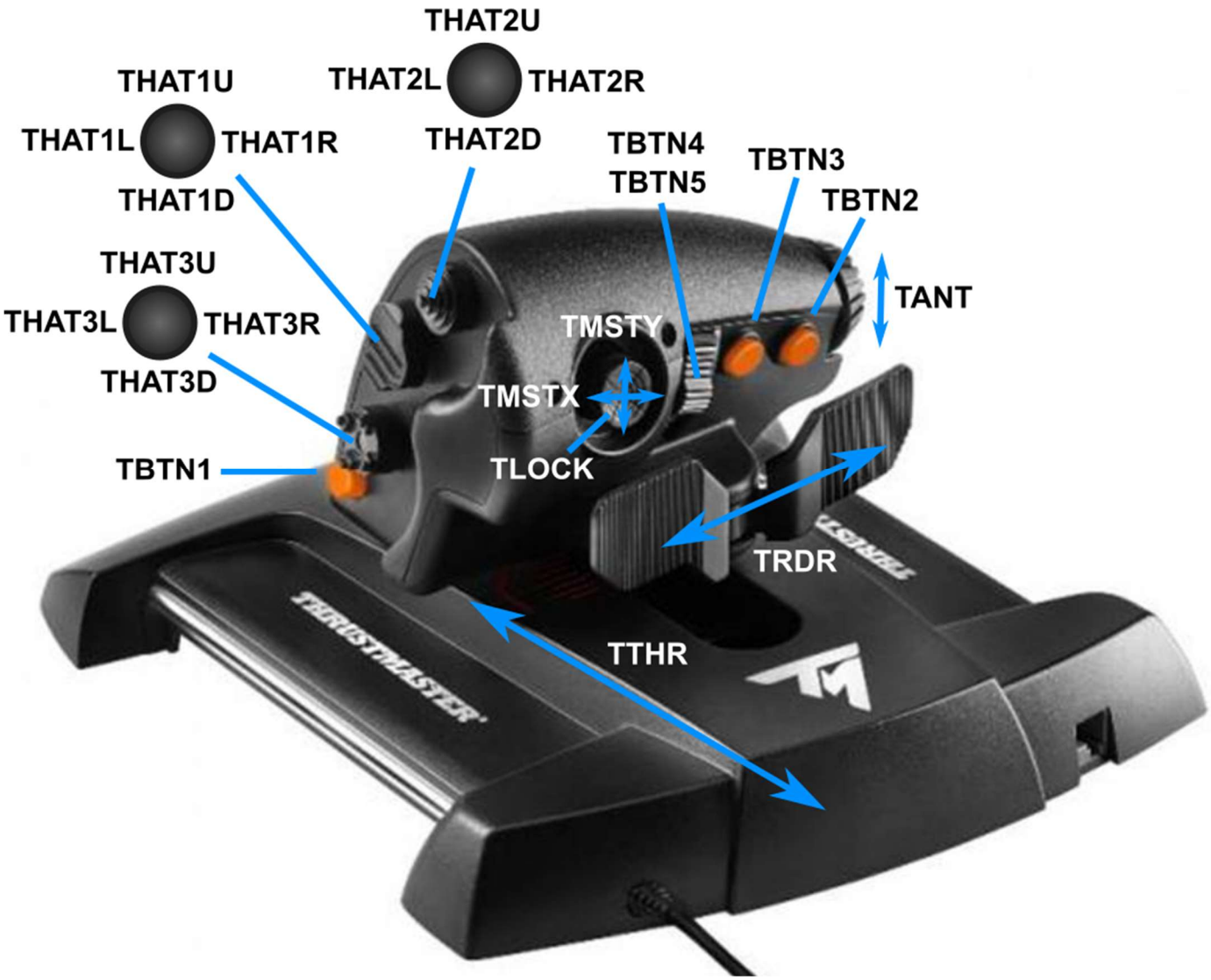
Thrustmaster Warthog Button and Axis Layout



Notes about the Warthog Throttle Button Layout:

- IDLELON/IDLERON are the buttons actuated when you pull the left or right throttle (THR_LEFT or THR_RIGHT) all the way back, lift the lever up and pull further back. Don't confuse IDLExON with IDLExOFF. The ON state is when the lever is lifted and pulled all the way back. The OFF state is when the lever is pushed forward out of the lifted zone. The diagram above has these reversed.
- SC, SCX, SCY – SCX/SCY is the "mouse" joystick on the front of the right throttle lever. SC is the center push button for SCX/SCY.

Thrustmaster TWCS Button and Axis Layout



Notes about the TWCS Throttle Button Layout:

- TMSTX, TMSTY, TLOCK – TMSTX/TMSTY is the “mouse” joystick on the front of the throttle lever. TLOCK is the center push button for TMSTX/TMSTY.
- The TWCS has 3 axes, TTHR (the main throttle axis), TANT (the pinky axis knob on the left side of the throttle axis), and TRDR (the spring loaded rudder axis on the front of the throttle axis). Because TRDR is spring loaded, it always re-centers. So, it cannot be used like a normal axis to control an engine lever.

The TWCS has a connector on the front left corner. This connector allows attaching a Thrustmaster TFRP rudder pedal set to the TWCS, adding rudder and left/right brake axes. Such a rudder set is not generally useful for TSW, but you could wire up your own 3 axis throttle quadrant for example and connect it to this connector, adding 3 more axis to the TARGET Script!! Hence it would be possible to run most the locomotives with just a TWCS throttle and your own custom built 3 axis throttle quadrant. This can be a much less expensive solution vs. buying a Warthog.

Changing the Engine Profile

For the v2.x TARGET script, changing profiles was accomplished by pressing and holding the Warthog MSP button (you will hear a “bip” sound to confirm it was pressed). Then with each press of the LDGH button, you will advance to the next profile. It works the same with the v3.0 script, but now you can press the APENG button to move to the previous profile. Also, it no longer cycles from the last profile to the first profile. Instead, you must use APENG to go back down to the first profile.

Also, new in v3.0, the Warthog THR_FC axis can select the profiles as well. This is perhaps more convenient than using the push buttons.

- Push the lever forward a little from center (between 60% and 90% range) and the TARGET Script will cycle to each next profile about once a second.
- Push THR_FC all the way forward (between 90% and 100%), and it will cycle through the profiles more quickly.
- The same is true for cycling backward by pulling the THR_FC lever back (40% to 10% for slow cycle, 10% to 0% for fast cycle).
- It will stop cycling through profiles once you have reached either the first or last profile.

Feedback to tell what profile is selected:

- Watch the TARGET Script Editor output window for messages which identify which profile has been selected.
- The LEDs on the Warthog Throttle will display a binary number indicating the current profile number (see table below to decode the pattern).
- A tone is sounded for each profile, increasing in tone as you increase to the next profile, decreasing when going to the previous profile. With the pitch sounds, or LEDs you can identify which profile you have switched to without leaving or alt-tabbing out of TSW.
- When you release the MSP button, the script will speak the name of the chosen profile as confirmation.

A note about the Text to Speech function in the TSW TARGET Script...

Text to Speech might not work for some people depending on your Internet Security settings, or if you have removed, or never run, or never installed Internet Explorer. Or, it may trigger a false positive on your virus program. The issue is because I am using a Microsoft Windows program (mshta.exe) and VBScript (Visual Basic Script) to execute a speech command from the Windows Speech API. Because this is part of Internet Explorer, tighter than default security settings, or other issues with Internet Explorer may prevent it from working or even flag as a false positive. If you are concerned about all this, or just don't want to deal with it, see below for how to disable Text to Speech in my script. If it becomes a problem for many people, I will disable it by default or find an alternate solution using a separate program.

How to get Speech working

If Speech isn't working for you, try running Internet Explorer once and exiting it. I mean Internet Explorer, not Microsoft Edge or Chrome. This may configure your Internet Security Settings and allow things to work. If you have changed your Internet Security settings to be more stringent than the default, that may also affect it. But

there are too many security settings for me to suggest what you should change if you want it to work. The best I can suggest is to set your security settings back to the default. Also, if you removed or never installed Internet Explorer, you may need to reinstall it. The mshta.exe program is installed by default with Internet Explorer when Windows is installed, but if you selected an advanced install, it is possible to not install Internet Explorer, or you can remove it later in Windows Settings, or the Control Panel, and possibly prevent this from working.

To check your Internet Security settings, hit the Windows key, and type "Internet options" without the quotes and it should show the Internet Options control panel in the top of the list. Click on that. Then go to the Security tab. The security levels you have set for Local Intranet (or maybe Trusted Sites) is what affects things like my Text to Speech function.

How to disable Text to Speech

If you are concerned about all this, or you just don't want text to speech in the TSW script, you can disable the function in my script by changing the line that says:

define SpeakEnabled 1

to the following:

define SpeakEnabled 0

Want to know more...

If you want to see my code that does all this, look for the Speak() function toward the end of the .tmc file. I have comments and web links there that explain what I am doing, and what mshta.exe is. It is a very useful function, but if too many people have problems getting it to work, I may have to use a different solution.

The Profiles are (in order of selection):

The Locomotive profiles are sorted in order of world region (North America, Britain, Germany) first, and within each region, they are sorted by the locomotive's number (e.g. BR Class 40, 43, 45, 47, etc.). The table below shows the list, in order of selection. As new locomotives are added, this order will change to insert new locomotives in the list.

Text in TARGET Window		**Engines Supported**	**LED Pattern [1]**
Profile 1:	pro_LIRRM7	LIRR M7 EMU	LED 00001 = 1
Profile 2:	pro_SD40GP38	GP38-2, GP40-2, SD40-2, GP9	LED 00010 = 2
Profile 3:	pro_F40PH	EMD F40PH-2CAT	LED 00011 = 3
Profile 4:	pro_F40PHCAB	EMD F40PH-2CAT CAB CAR	LED 00100 = 4
Profile 5:	pro_ACS64	Amtrak ACS-64	LED 00101 = 5
Profile 6:	pro_SW1000R	Amtrak SW1000R	LED 00110 = 6
Profile 7:	pro_AC4400CW	GE AC4400CW	LED 00111 = 7
Profile 8:	pro_ExMetro	Amtrak ex-Metroliner Cab Car IVb	LED 01000 = 8
Profile 9:	pro_BR08	BR Class 08	LED 01001 = 9
Profile 10:	pro_BR09	BR Class 09	LED 01010 = 10
Profile 11:	pro_BR31	BR Class 31	LED 01011 = 11
Profile 12:	pro_BR33	BR Class 33	LED 01100 = 12
Profile 13:	pro_BR37BR40	BR Class 37 and 40	LED 01101 = 13
Profile 14:	pro_BR43	BR Class 43 HST	LED 01110 = 14
Profile 15:	pro_BR45	BR Class 45	LED 01111 = 15
Profile 16:	pro_BR47	BR Class 47	LED 10000 = 16
Profile 17:	pro_BR52	BR Class 52	LED 10001 = 17
Profile 18:	pro_BR66	BR Class 66	LED 10010 = 18
Profile 19:	pro_BR101	BR Class 101	LED 10011 = 19
Profile 20:	pro_BR166	BR Class 166	LED 10100 = 20
Profile 21:	pro_DBBR143	DB BR 143	LED 10101 = 21
Profile 22:	pro_DBBR155	DB BR 155	LED 10110 = 22
Profile 23:	pro_DBBR182	DB BR 182	LED 10111 = 23
Profile 24:	pro_DBBR185	DB BR 146.2 and BR 185.2	LED 11000 = 24
Profile 25:	pro_DBBR767	DB BR 766.2 and 767.2 Control Cars	LED 11001 = 25
Profile 26:	pro_DBBR1442	DB BR 1442-1 Talent 2 EMU	LED 11010 = 26

[1] A 1 means the LED is lit, a 0 means it is dark.

Using the TARGET Script – Detailed Usage Notes

Following is a detailed description of how to use the TARGET script to control various locomotives.

What is the TARGET Script Doing?

Since TSW does not support DirectX game controllers (other than an Xbox controller), it is impossible to directly map a joystick or throttle axis to control a locomotive control lever. Instead, the only option is to program the game controller to press keyboard keys as you move its axis. This is called Axis to Keyboard mapping.

For example, increasing or decreasing the engine throttle in TSW is accomplished using A and D respectively. The TARGET Script configures the Warthog Right Throttle axis to control the locomotive throttle. As the Right Throttle axis is moved backwards, the script presses the A key several times to step through the TSW engine throttle positions. Move the Right Throttle axis forward and the script presses the D key several times. What happens in order to make this work reliably is covered in greater detail in the Technical Details section below. For now, it is only important to understand that the TARGET script is doing the same thing you would do in order to control the engine with your keyboard.

For buttons and switches, again the TARGET Script is pressing the same keyboard keys you would use to control these buttons and switches in TSW.

If you want to see this in action, click on the Event Tester icon in the TARGET Script editor to open a program that will show you what keys are being pressed and for how long.

Warthog Axis out of Sync with In-Game Lever

Occasionally, a Warthog axes will get out of sync with the engine lever in TSW. Great effort has been taken to avoid this, so it is possible to throw the Warthog axes back and forth quickly and have every axis movement accurately translate to moving the lever in game. This is the big difference between using this script and simply using key mapping features in other programming software like I did for the Logitech/Saitek profiles. For example, it should be possible to throw an axis full forward to full backward several times, and though there is a lag between moving the Warthog axes and the in-game lever, the lever in game should eventually follow your every movement.

Because this is only an axis to keyboard mapping script, occasionally a keypress is missed by TSW. This causes the Warthog axis to be out of sync with the sim. It happens rarely, but still more often than I would like, and it can be worse with some engines vs. others. So, if something isn't working as expected, check to see if the Warthog axis is out of sync.

When they get out of sync, either use the keyboard keys to move the in-game lever to match the Warthog, or press ESC to pause the game, and move the Warthog axis to match the lever in game.

For most locomotive control levers, the TARGET script has been designed to resync automatically when the Warthog axis is pushed to the maximum or minimum limits of travel. An extra long press of the key is given to make sure that if a keypress was missed, the Warthog will resync the TSW lever with the Warthog.
Also, on most profiles, you can resync at the ends of travel (lever at max or min position) by wiggling the Warthog lever back and forth by about 1% of travel. This will resend the required key press to move the TSW engine lever to match the Warthog lever.

For some brake levers, it is possible to wiggle the Warthog axis similarly, but you must wiggle it a full step back and forth, not just 1%. Then the long press of the key used by TSW will step the in-game lever similarly to its max or min position and it should be in sync.

Finally, when either of the Warthog throttle axes are pulled all the way back, you can lift the axis and pull it into the Idle position (IDLELON or IDLERON) which will press and hold the programed key until you move the lever off the Idle position. This was done to allow resyncing but is also used for real locomotive functions like bail off on the Independent brake, or Release/Overcharge on some British Rail driver (automatic) brake levers like on the BR Class 47.

For example, if the GP38-2 throttle is out of sync (e.g. the GP38-2 throttle is on 2 when you intended to reach 1), you can quickly solve this by pushing the Warthog Right axis all the way to min throttle. This will give an extra long press of the 'd' key which will move the GP38-2 throttle to IDLE, synchronizing the axis and the GP38-2 throttle lever. Then move the Warthog Right axis back to the "1" position, and the GP38-2 will follow correctly. The same feature happens when moving the Warthog to maximum throttle, or for the automatic brake. Though for the automatic brake, it is not suggested to push the brake lever all the way forward to solve a sync issue as this will engage the emergency brake, stop the train, and you'll have to deal with the emergency brake recovery procedure, unhappy passengers, and perhaps getting fired by the boss.

These resync features apply to the independent, automatic, and dynamic brakes for most engines. For the reverser, just cycle the Warthog THR_FC lever back and forth, and it will resync with the reverser in game.

Lever Notch Feedback

Many locomotive control levers in TSW have notches. For example, the throttle lever on the GP38-2 has 9 notches, IDLE and 1 through 8, or the AC4400CW has a hump between the throttle and dynamic braking sections. Sadly, the Warthog Throttle does not have these physical notches to give feedback. Even the RailDriver product does not match these notches perfectly for every engine. If you are really hardcore (as if using a Warthog to run TSW isn't already pretty hard core), you can change out the Warthog Afterburner notch piece with a 3D printed notch piece that will give physical notch feedback. See the forums for this TARGET script more info.

Feedback as you move the in-game levers is given by HUD messages or the HUD compass in the lower right corner, but operating the engines with the HUD off, you lose this feedback. For some levers, a click can be heard in TSW that indicates lever movement. This is not consistent with all engines. Hence, the TARGET script uses various tones and beeps (assuming you installed wBeep.exe and have not disabled beeps in the script) to indicate the most important notch points such as the hump or notch between throttle and braking on the AC4400CW, Class 166, and ACS-64 throttle/brake lever, or important positions on the automatic brake lever such as Release, Handle Off and Emergency Brakes.

Because there are no physical notches on the Warthog, when you first start using this script, start out moving the Warthog axes slowly while getting used to the various feedback (HUD, in game clicks, and the TARGET Script beeps). This will help prevent overshooting the desired lever position in game. After some practice, muscle memory will kick in and using the Warthog will feel very natural.

Reverser (Warthog THR_FC Axis)

The Warthog Throttle Friction Control axis (THR_FC axis) is mapped to control the engine reverser lever for all engines.

If the Warthog THR_FC axis gets out of sync with the engine's reverser (which happens usually when starting a new scenario without first setting the THR_FC to match the reverser in game or when changing engine profiles) just move the THR_FC axis lever full forward, then full backward, then back to center, and the THR_FC axis will be in sync with TSW again.

Many engines have an Engine Off position (e.g. Class 43 HST, ACS-64, and Class 166). The THR_FC axis is programmed to reach this position only in the last 2% of axis travel. Moving the THR_FC only 10% forward or 10% backward from center, however, will select forward and reverse while avoiding accidentally moving to the Engine Off position.

Throttle Lever (Warthog Right Axis)

The Warthog Right Throttle axis is mapped to control the throttle lever for locomotives that have a dedicated throttle lever (e.g. the GP/SD engines, BR Class 43 HST, etc.). For locomotives with AFB (cruise control), the throttle lever may be mapped to control AFB, or to control the power setting. See the description for individual locomotives to see how it is mapped.

Combined Throttle & Brake Lever (Warthog Right Axis)

This applies to engines like the AC4400CW, Class 166, and ACS-64. These engines have a combined throttle and brake lever with several notched throttle positions (AC4400CW or Class 166), or a continuous (non-notched) throttle zone (ACS-64), a notch or hump in the middle, and several notched brake positions (e.g. Class 166), or a continuous (non-notched) brake zone (e.g. AC4400CW, ACS-64). The Warthog Right axis maps directly to the throttle/brake lever on these engines.

When using these engines, it is best to move the Right axis slowly to not overshoot the center detent or hump between throttle and braking zones. This is especially important for engines like the AC4400CW as the center hump between throttle and braking requires a 1 second key press to overcome the hump. Hence it is easy to get confused as to where the Warthog lever is vs. the lever in TSW. There is some delay between moving the Warthog Right axis and the lever movement in the sim making it easy to get out of sync or overshoot the desired position. For the AC4400CW, extra beeps are used to help identify the movement over this hump. It is a good idea to practice moving back and forth over the hump to get used to the feel and the beeps and it will become second nature very quickly.

Be aware that if the reverser on the AC4400CW or similar engines is in neutral, TSW will not allow the throttle/brake lever to move into the braking region. Of course, the Warthog axis has no such limitation which gets things out of sync immediately.

Dynamic/Automatic/Independent Brake Lever Selection using the FLAPS Switch (Warthog Left Axis)

There are three types of brake levers in TSW, the dynamic brake, automatic (or driver's) brake, and independent (or straight) brake. Not all engines have all three levers, and some engines combine one or more of these brake functions with the throttle lever (e.g. AC4400CW, BR Class 166, ACS-64). All three brake levers are controlled by the Warthog Throttle Left axis. You select which brake lever you want to control with the position of the FLAPS switch.

The Warthog Left axis is mapped as follows:

FLAPS Switch Position	**Left Axis Controls**
FLAPU (forward position)	Dynamic Brake
FLAPM (middle position)	Automatic (Driver's or train) Brake
FLAPD (backward position)	Independent (Straight or locomotive) Brake

NOTE: When changing profiles, it is often necessary to move the FLAPS switch back and forth in order to enable all the axes. This is a bug with the script that will be resolved in a future release.

For engines that only have a single brake lever (e.g. Class 43 HST), the position of the FLAPS switch does not matter. For other engines, only those positions that match the engine are functional.

The TARGET script remembers the position of all three in game brake levers. When switching from one to another with the FLAPS switch, the script will prevent the Warthog Left axis from affecting the in game lever until it has synchronized with the in game lever.

For example, in a GP38-2 going down a hill with the automatic and independent brakes set to release, and the dynamic brake set to position 7, say you are still gaining speed, but can't increase the dynamic brake more due to a brake warning (over current) light. Adding a little automatic braking will slow things down. To do this with the Warthog Left axis, flip the FLAPS switch to FLAPM to select the automatic brakes and pull the Left axis all the way back. You will hear a beep indicating that the Left Axis is now in sync with the automatic brake lever in game. Apply a little automatic braking to slow down, then move the automatic brake lever back to release. Flip the FLAPS switch back to FLAPU to select dynamic brakes and move the Left axis forward a little. Notice the dynamic brake lever in game does not move. Keep pushing the lever forward until you hear a beep. Now the Left axis is in sync with the in game dynamic brake lever, and lever movement is enabled again.

For those that want separate lever controls, that is why the Logitech/Saitek Throttle Quadrant profiles were provided. They allow controlling the automatic (left lever) and independent (middle lever) brakes. When using the Throttle Quadrant, set the Warthog FLAPS switch to FLAPU to control the dynamic brake with the Warthog Left axis. I did not set up the Throttle Quadrant right lever to control the dynamic brakes because I always use the Warthog for dynamic brakes regardless. Or for the AC4400CW, I use the Warthog Left axis to control the automatic brakes, and the Saitek Throttle Quadrant only to control the independent brake.

In v3.0 (and beyond) of the script, the TWCS was added by default to act as the dynamic brake for those that have both a Warthog and TWCS. Since you would not use the independent and automatic brake at the same time, this means it is easy to have a dedicated lever for each brake, only changing the Warthog Left Throttle lever with the FLAPS switch as needed for running a train (automatic brake) vs. just a set of engines (independent brake).

Here are some specifics about each brake setting.

Dynamic Brake

This applies to engines like the SD40-2, GP38-2 and GP40-2. These engines use a dedicated dynamic brake lever which has OFF and SETUP notched positions and an un-notched continuous region (e.g. marked 1 through 8). When FLAPU is selected, the Warthog Left axis maps directly to the dynamic brake lever in game.

Automatic (Driver's) Brake

Almost all the engines in TSW have an automatic or driver's brake, or a brake that is mapped to work with the automatic brake keys in TSW (; and '). The automatic brake usually has several notched positions (e.g. Release, Initial Reduction, Suppression, Handle Off, and Emergency) as well as a continuous non-notched region between Initial Reduction and Suppression. When FLAPM is selected the Warthog Left axis maps directly to the automatic brake lever in game.

To actuate Release/Overcharge for engines that have that feature, with the Warthog Left axis pulled all the way back, lift the Left axis up and pull back to the Engine OFF position (called IDLELON). As long as the axis is in IDLELON, the automatic brake will be held in Release/Overcharge (the ; key will be held down). Push the Left axis forward to IDLE and Release/Overcharge is released.

Many British Rail locomotives have a Neutral/Shutdown position at the top of travel for the automatic or driver's brake. Several of these locomotives also have a Brake Pin on the driver's brake lever.

When moved to the Running position, the Brake Pin on the Driver's Brake is engaged. When the Brake Pin is engaged, you cannot move the Driver's Brake Lever into Neutral/Shutdown. You must either disengage the pin manually with the mouse (pull the pin up) or wait about 5 seconds for the Brake Pin to disengage automatically. It takes about 5 seconds for the pin to pop up for most locomotives.

Using the Warthog Left axis, if you try to engage Neutral/Shutdown while the Brake Pin is engaged be patient as the script will not immediately move into Neutral/Shutdown. It has an intentional 5 second delay in order to wait for the Brake Pin to disengage. Then the script will then move the lever into Neutral/Shutdown.

The script will sound two high pitch beeps, 5 seconds apart, when trying to move into Neutral/Shutdown. If the Brake Pin is disengaged, the brake lever will move to Neutral/Shutdown immediately coincident with the first high pitch beep. The second high pitch beep will occur 5 seconds later indicating that the 5 second delay is over. If the brake lever couldn't move into Neutral/Shutdown on the first high pitch beep, by the time you hear the second high pitch beep, the Brake Pin should have been disengaged, and the brake lever should now move to Neutral/Shutdown.

If you move the Wathog Left axis into Neutral/Shutdown and then back out immediately (to Emergency or any position less than Emergency), you must still wait for the 5 second delay even though the pin is still disengaged. Listen for the two high pitch beeps to know when the 5 second delay ends, at which point the locomotive brake lever will move out of Neutral/Shutdown as commanded.

If for any reason the locomotive driver's brake lever still does not move into Neutral/Shutdown, wiggle the Warthog Left Throttle lever about 2% of travel at max forward position and it will cause the Driver's Brake Lever to go into Neutral/Shutdown. If you move the lever too far, you may cause another 5 second delay. Again, listen for the 2 high pitch beeps if this happens.

Unfortunately, there is nothing that can be done in the script to overcome this delay. The Brake Pin has no keyboard keys which can be used to disengage it, and there is no way for the script to know if the pin is engaged or not in TSW. Hence, we must wait the 5 second delay whether it was needed or not.

Independent Brake

Most engines have an independent brake. When FLAPD is selected the Warthog Left axis maps directly to the independent brake lever in game.

To actuate Bail Off, with the Warthog Left axis pulled all the way back, lift the Left axis up and pull back to the Engine OFF position (called IDLELON). As long as the axis is in IDLELON, the independent brake will be held in Bail Off (the ; key will be held down). Push the Left axis forward to IDLE and Bail Off is released.

Views – Mic Switch

The Mic Switch is mapped to control TSW views and enable/disable HUD elements. See the Excel file for all mappings. Here are a few notes on the Mic Switch.

- MSR (push Mic switch forward) – In Cab View – TSW 1 key
- MSR Press and Hold – Free View – TSW 8 key
- MSL (push Mic switch backward) – Exterior View – TSW 3 key
- MSL Press and Hold – Boom View – TSW 2 key
- MSD (press Mic switch down) – Toggle HUD Markers toggles the 3 icon markers that show next way point, next speed zone, and next signal (TSW CTRL-1, CTRL-2, CTRL-3).
- MSD Press and Hold – HUD Toggle turns all HUD elements on or off – TSW F1 key.
- MSU (press Mic switch up) – 2D Map View – TSW 9 key
- MSU Press and Hold – TSW Speedometer/Compass Toggle (CTRL-5) toggles display of the HUD elements in the lower right corner in TSW.

Using the Mic Switch for views, the Coolie Switch as the arrow keys (see below), and some mappings on my Logitech G700S Mouse (see the Excel file) allows me to control most external views and First Person activities without touching the keyboard.

Arrow Keys – Coolie Switch

I normally have my left hand on the Warthog Throttle and right hand on the mouse. I mapped the following Warthog buttons so I don't have to use the keyboard arrow keys to move around. I can quickly switch views using the Mic Switch, and then fly around.

The keyboard arrow keys are mapped to the Coolie HAT switch. This controls the UP, DOWN, LEFT, and RIGHT arrow keys for switching views in the cab or moving around in external views.

MSP + the Left or Right Coolie buttons (CSL and CSR), does a CTRL-LEFT ARROW or CTRL-RIGHT ARROW which is used in external views.

The Slew Control center button (SC) maps to the keyboard SHIFT key. When SHIFT is held with an arrow key, you move faster in external views. This also works with First Person walk around mode.

Headlight Switches

TSW has a forward headlight switch (keyboard hot key h and SHIFT-h) and a backward headlight switch (keyboard hot key CTRL-h and SHIFT-CTRL-h). These switches work differently for different engines. The Warthog Engine Operate Switches (ENG OPER Left and Right) are mapped to control these switches. Actuate the switches multiple times to move the headlight switches forward or backward several positions (e.g. off to dim to bright). For example, the switches should be in the center position normally when you load a new locomotive. To turn on the headlights from off to dim to bright (on certain locomotives), move the right ENG OPER switch forward and release it back to center twice (it is spring loaded). To reduce the headlight from bright to dim to off, move the switch in the opposite direction, and then manually move it back to center twice. In this direction, the switch is not spring loaded, so you must move it back and forth manually.

Windshield Wiper Switch

The wipers work differently for different engines. Some have simple on or off switches while other engines have a variable speed control (0 to 100%), and some have multi-function switches with off, on, park, etc. where you have to park the wipers to get them out of the way.

The Warthog Autopilot APPAT and APALT toggle switch positions are intended to increase (APPAT) or decrease (APALT) the wiper switch position or wiper speed on the locomotive. Quickly flip the toggle switch forward to APPAT and back to the middle position and it will advance the wiper switch one position or increase the wiper speed by about 25%. Quickly flip the toggle switch backward to APALT and back to the middle position and it will move the wiper switch back one position or decrease the wiper speed by about 25%. If the toggle switch is left in the APPAT or APALT, the TARGET script will execute only 5 presses of the 'v' or 'shift-v' keys, which is enough to set the wiper switch to max position/speed, or min position/speed.

Cabin and Instrument Lighting

The Warthog Engine Fuel Flow Switches (EFLNORM and EFRNORM) control the engine cabin lights (keypress L) and gauge lights (keypress I) respective. When walking around EFLNORM toggles the flashlight on and off.

Some engines have multiple settings for these lights. In that case, the Warthog was programmed to pick the most useful setting.

Horn 1 and Horn 2

Some locomotives have two horns. For all locomotives, the Warthog Autopilot button (APENG/APDIS) is mapped to blow the primary horn (horn 1, equivalent to pressing the spacebar). But for locomotives with two horns, the script also maps the Left Hand Pinky Switch (PSF and PSB) to blow horn 1 and horn 2.

AWS, SIFA and PZB

The AWS and SIFA Acknowledge Button (Q) is mapped to the Warthog LDGH button for all locomotives. PZB, however, has 3 buttons, PZB Override, PZB Release, and PZB Acknowledge (DEL, END, and PGDN respectively). These buttons are not mapped to the Warthog Throttle because it is simply easier to use the keyboard keys. The Warthog does not have a triplet of buttons that are arranged to cover this function very well.

Sand

The Warthog RDR ALTM and Speed Brake switches are mapped to control sand for most locomotives. But it seems sand is not needed. Wheel slip is either not modeled, or not modeled well enough to matter. DTG may add sand modeling later, but as yet, it does not seem to matter. Because I use these same buttons/switches when using my Warthog in Train Simulator, I mapped them for TSW as well for consistency. Of course, you can remap them as desired if you don't want this.

Other Switches and Buttons

All other switches and functions in TSW should be easy to figure out. See the Excel cheat sheet for details.

Note there are many cases where a toggle switch position (on or off, up or down) does not matter. For example, the locomotive bell is turned on and off by pressing the B key. The Warthog EAC switch is set to control the bell. But all the EAC switch does is press the B key any time you flip it, up or down. So, it doesn't matter which way the switch is flipped.

Using the TARGET Script – Profile Specific Usage

The following sections explain specific usage details for each locomotive profile.

Profile 1: LIRR M7 EMU

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Reverse, Neutral, Forward, Key Out. Key Out is mapped to the upper 2% range of the Warthog THR_FC axis.
- The M7 has only one control lever – the throttle/brake lever.
    - This is controlled by the Warthog Right Throttle Axis. All other axis, including the TWCS are disabled for this profile.
    - If new to this locomotive, move the throttle/brake lever slowly while you are getting used to its settings and how the Warthog controls it. The TARGET Script produces beeps for certain positions help make it clear what is happening.

Profile 2: EMD GP38-2, GP40-2, SD40-2, GP9

Here is a description of controls specific to this profile.

- The GP9 and CN GP38-2 have not been fully tested yet, but appear to work correctly.
- The CN Oakville GP38-2 has some minor differences vs. the GP38-2 in other DLC. Although the CN GP38-2 should work, some switches (e.g. headlight controls) work differently on the Warthog.
- The CN GP38-2 headlight switches works a little differently.
    - There are separate Front and Rear headlight on switches. These switches are controlled by the ENG OPER switches when pressed forward (in their momentary position) to toggle the state of each switch.
    - The Headlight brightness switch (lower right corner of main panel) cannot be controlled by the TARGET Script because there is no key combo in TSW that will actuate this switch. If you want to use this switch, you will have to use the mouse.
- Lead Axle Sand
    - For the SD40-2 and GP40-2, lead axle sand is toggled on/off by flipping the RDR ALTM switch **ON** (RDRNRM position) each time you want to **TOGGLE** the state of the sander. There is no effect when the RDR ALTM switch is turned off (RDRDIS position). In other words, if lead axle sand is off, flip the RDR ALTM switch up and lead axle sand will be on. Flip RDR ALTM down, and nothing happens. Now to turn sand off, repeat the operation. Flip the RDR ALTM switch up, and sand is turned off.
    - For the GP38-2, the RDR ALTM switch works normally. The RDR ALTM switch position matches the lead axle sand switch in the locomotive.
- Sand
    - Turned on by holding the momentary Speedbrake (SPDB) button on. Turned off when SPDB is released. This is the default behavior for most locomotives.

Profile 3: EMD F40PH-2CAT

Here is a description of controls specific to this profile.

- F40PH has a STOP position on the throttle lever which shuts down the engine. Going into this mode accidentally is undesirable. It stops the engine, which must then be restarted.
    - To go into STOP, TSW wants to see two presses of the D key rapidly. The TARGET Script is programmed to avoid doing this when not intended.
    - If you want to move the throttle lever to the STOP position, make sure the Warthog Right Throttle axis (and engine throttle) is all the way forward. Press and hold MSP (the IO Shift button) and pull the Warthog Throttle axis back about 2-3%, then forward again (this is practically a wiggle). This will cause a rapid double D to be pressed and the engine throttle should go into the STOP position.
    - After doing this, the throttle will go into the STOP position every time. To stop it from doing that, cycle the Warthog Right Throttle axis all the way back and all the way forward. Now it will no longer go into STOP.
    - After entering STOP, you will also need to restart the locomotive.

Profile 4: EMD F40PH-2CAT CAB CAR

Here is a description of controls specific to this profile.

- The Cab Car does not have an Independent brake. But it does have a parking brake.
    - For the TARGET Script, when the FLAPS switch is set to FLAPD (the normal position for the independent brake), the Warthog Left Axis will control the parking brake.
    - Axis Forward (above 60%) is brake released. Axis backward (below 40%) is brake applied.
- Sander – The F40PH Cab Car has a sander lever. Normally CTRL-X will engage sand, and SHIFT-X will disengage it, but right now for the F40PH Cab Car, SHIFT-X does not work. Hence, sanding by the TARGET script has been disabled. If you want to use sand, it is best to use the mouse to actuate it.
- The F40PH Cab Car does not have the STOP position on the throttle like the F40PH locomotive.
- After blowing the horn, the bell remains on. To turn it off, just toggle the Warthog EAC switch. It does not matter if the switch is forward or backward. In both cases, flipping the switch presses the ‘b’ key which toggles the bell on or off.

Profile 5: Amtrak ACS-64

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Reverse, Neutral, Forward, Out. Out is mapped to the upper 2% range of the Warthog THR_FC axis.
- The ACS-64 has a combined throttle/brake lever.
    - This is controlled by the Warthog Right Throttle Axis.
    - If new to this locomotive, move the throttle/brake lever slowly while you are getting used to its settings and how the Warthog controls it.

- The TWCS is mapped to control the Independent Brake since the dynamic brake is part of the throttle/brake lever.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- After blowing the horn, the bell remains on. To turn it off, just toggle the Warthog EAC switch. It does not matter if the switch is forward or backward. In both cases, flipping the switch presses the 'b' key which toggles the bell on or off.

Profile 6: Amtrak SW1000R

Here is a description of controls specific to this profile.

- Headlight Switches (ENG OPER switches).
    - The SW1000R has separate Front and Rear headlight on/off switches. These switches are controlled by the ENG OPER switches when pressed forward (in their momentary position) to toggle the state of each switch.
    - The Headlight brightness switch (lower right corner of main panel) cannot be controlled by the TARGET Script because there is no key combo in TSW that will actuate this switch. If you want to use this switch, you will have to use the mouse.
- Lead Axle Sand and Sand
    - The SW1000R does not have lead axle sand. Hence, the RDR ALTM switch and the Speedbrake (SPDB) buttons both actuate normal sand.
- Sand
    - Turned on by holding the momentary Speedbrake (SPDB) button on. Turned off when SPDB is released. This is the default behavior for most locomotives.
- Bell
    - The SW1000R bell is turned on when you blow the horn. To turn it off, cycle the Warthog EAC switch on, then off.
- Windshield Wipers
    - On the SW1000R, the Warthog Autopilot APPTH/APALT switch, which is used to control the wipers, works normally but only controls the wiper on the engineer's front window. There are no key combos in TSW to control the other wipers. They must be controlled manually with the mouse.

Profile 7: GE AC4400CW

All controls are mapped and work as normal for the AC440CW except that the 4400 has a single throttle/dynamic brake lever. This lever has a hump between the throttle and braking side. If new to this locomotive, make small and slow inputs to the Warthog Throttle axis to give yourself time to understand the AC4400 throttle/brake lever and how the Warthog controls it. Listen to the beep feedback provided by the TARGET Script. When crossing over the hump, there is a long low tone to indicate the transition, followed by a slightly higher tone to indicate you have completed the transition and reached the Idle position (true for going into braking as well as leaving braking mode). It is best to always cross the hump as one operation, wait for the proper tones so you know the crossover has completed, and then increase throttle or braking as desired.

Even with experience, you'll find that just pulling the Warthog lever past the hump quickly and then adding braking or throttle in one operation can cause problems.

In fact, if you want to operate realistically, you would never just blow past the hump into dynamic braking anyway. Look at the placard just over the throttle/brake lever in TSW. Like most dynamic braking systems, the placard explains you are supposed to hold the braking system at idle for a few seconds before increasing the dynamic braking. TSW does not enforce this requirement as best I can tell.

Also, remember, you cannot engage the brake side of the AC4400 throttle/brake lever if the reverser is in neutral. TSW will not allow the lever to move. No such limitation exists for the Warthog axis. Hence it is very easy to get out of sync if you do it wrong. Of all the engines in TSW, this one is potentially the most confusing because of these factors.

Profile 8: Amtrak ex-Metroliner Cab Car IVb

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Reverse, Neutral, Forward, Out. Out is mapped to the upper 2% range of the Warthog THR_FC axis.
- Parking Brake
    - The is no independent brake on the ex-Metroliner. Instead, it has a parking brake. The Warthog Left axis controls the parking brake lever when the FLAPs switch in in the down position.
    - The TWCS is mapped to control the Parking Brake as well.
- After blowing the horn, the bell remains on. To turn it off, just toggle the Warthog EAC switch. It does not matter if the switch is forward or backward. In both cases, flipping the switch presses the 'b' key which toggles the bell on or off.

Profile 9: British Rail Class 08

Here is a description of controls specific to this profile.

- Master Key
    - This is a 4 position lever with positions OFF, ON, ENGINE ONLY, and START.
    - Instead of using the Warthog APUON switch, the Master Key is controlled by the Warthog ENG OPER Left. Press the switch forward to move the Master Key forward, and toggle it backwards, then to center to move the Master Key backward.
    - To start the locomotive engine, it is necessary to hold the ENG OPER switch forward for a few seconds until the engine starts. Then it can be released and the Master Key will move back to the Engine Only position.
- Sand
    - The Speedbrake switch controls sand. SPDB will move the sand lever to the next setting (Front Sander, Rear Sander). SPDF will move the sand lever to the previous setting (Front Sand, Off).
    - The RDR ALTM switch is disabled for this profile (does not control sand).

Profile 10: British Rail Class 09

Here is a description of controls specific to this profile.

- Master Key
  - This is a 4 position lever with positions OFF, ON, ENGINE ONLY, and START.
  - Instead of using the Warthog APUON switch, the Master Key is controlled by the Warthog ENG OPER Left. Press the switch forward to move the Master Key forward, and toggle it backwards, then to center to move the Master Key backward.
  - To start the locomotive engine, it is necessary to hold the ENG OPER switch forward for a few seconds until the engine starts. Then it can be released and the Master Key will move back to the Engine Only position.
- Sand
  - The Speedbrake switch controls sand. SPDB will move the sand lever to the next setting (Front Sander, Rear Sander). SPDF will move the sand lever to the previous setting (Front Sand, Off).
  - The RDR ALTM switch is disabled for this profile (does not control sand).

Profile 11: British Rail Class 31

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Off, Reverse, Engine Only, Forward. Off is mapped to the lower 2% range of the Warthog THR_FC axis.
- The Driver's (Automatic) Brake
  - The Driver's Brake lever has a spring loaded Release/Overcharge setting at the bottom end of travel. Actuate this by putting the Warthog Left axis in the IDLELON position (similar to Bail Off), and it will hold the Brake in Release/Overcharge as long as in that position.
  - This locomotive has a Brake Pin on the Driver's Brake. See the notes about the Brake Pin on British Rail Locomotives in the explanation of the Automatic (Driver's) Brake earlier in this document.
- Windshield Wiper
  - The wiper is a knob with 3 stages, Park which is momentary, Off, and Run. Run can be rotated from very slow to fast.
  - The Warthog APPTH/APALT switch is used to control this knob, and works the same as other locomotives, but because of the way the knob works, you may want to use the switch differently.
  - If you flip the switch up (APPTH), it will quickly turn the wiper knob to full speed. If you flip the switch down (APALT), it will quickly turn the wiper knob to Park. Move the switch back to the middle (from Park), and the knob will move to Off.
  - To adjust the wiper speed in the run position, quickly flip the switch up or down and then back to the middle. This will turn the wiper knob a small amount, adjusting the wiper speed.

Profile 12: British Rail Class 33

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Off, Reverse, Engine Only, Forward. Off is mapped to the lower 2% range of the Warthog THR_FC axis.
- The Driver's (Automatic) Brake
    - This locomotive has a Brake Pin on the Driver's Brake. See the notes about the Brake Pin on British Rail Locomotives in the explanation of the Automatic (Driver's) Brake earlier in this document.
- Windshield Wiper
    - The wiper is a knob with 3 stages, Off, Park, and Run.
    - The Warthog APPTH/APALT switch is used to control this knob, and works the same as other locomotives, but because of the way the knob works, you may want to use the switch different.
    - If you flip the switch up (APPTH), it will cycle to Park and then Run. If you flip the switch down (APALT), it will cycle to Park and then Off.
    - To toggle to Park from Run or Off, momentarily flip the switch up or down (as needed) and back to center, and the wipers will stop at Park to park the wipers. Once parked, move the switch back to turn off the wipers.

Profile 13: British Rail Class 37 and 40

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Off, Reverse, Engine Only, Forward. Off is mapped to the lower 2% range of the Warthog THR_FC axis.
- The Driver's (Automatic) Brake
    - The Driver's Brake lever has a spring loaded Release/Overcharge setting at the bottom end of travel. Actuate this by putting the Warthog Left axis in the IDLELON position (similar to Bail Off), and it will hold the Brake in Release/Overcharge as long as in that position.
    - This locomotive has a Brake Pin on the Driver's Brake. See the notes about the Brake Pin on British Rail Locomotives in the explanation of the Automatic (Driver's) Brake earlier in this document.
- Windshield Wiper
    - The wiper is a knob with 3 stages, Park which is momentary, Off, and Run. Run can be rotated from very slow to fast.
    - The Warthog APPTH/APALT switch is used to control this knob, and works the same as other locomotives, but because of the way the knob works, you may want to use the switch differently.
    - If you flip the switch up (APPTH), it will quickly turn the wiper knob to full speed. If you flip the switch down (APALT), it will quickly turn the wiper knob to Park. Move the switch back to the middle (from Park), and the knob will move to Off.
    - To adjust the wiper speed in the run position, quickly flip the switch up or down and then back to the middle. This will turn the wiper knob a small amount, adjusting the wiper speed.

Profile 14: British Rail Class 43 HST

Here is a description of controls specific to this profile.

- For the Class 43 HST engine, the brake lever is just a simple notched lever (no separate automatic, independent, or dynamic brake). The Warthog Left axis maps directly to this lever in game and is chosen regardless of the FLAPS switch position.

- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- The Master key is controlled by the Warthog APU Start switch.
- The Reverser has 4 positions: Off, Reverse, Neutral, Forward. Off is mapped to the lower 2% range of the Warthog THR_FC axis.
- The BR43 gauge lights have 4 positions. You must press the i key 4 times or hold i down for 1.25 seconds to get to 100% brightness. For the script, the EFRNORM switch turns the gauge lights on to full brightness and ERFOVER turns the gauge lights off (both hold the i key for 1.25 seconds).
- The wind shield wiper on the BR 43 has 4 positions, Parked, Off, On, Wash. Like all engines, windshield wipers are controlled by the Warthog Autopilot APPAT and APALT switch.
    - Push the switch forward (APPAT) and leave it there, and the BR43 wiper switch will cycle from Parked to Off to On to Wash. Flip the Autopilot switch backward (APALT), and the BR43 wiper switch will cycle from Wash to On to Off to Parked.
    - In the real engine, you would move the switch from Off to On to turn on the wipers. You would use Wash only to spray the window (not simulated in TSW), so this setting can be ignored. To turn the wipers off, you would flip the switch to Parked to park the wipers, and then once parked, move it to Off. By flipping the Warthog Autopilot switch forward or backward and then quickly back to center, you can have proper control of the BR43 wiper's switch to turn it on, park it and turn it off.
- The TWCS is disabled for the BR43 as it is redundant.

Profile 15: British Rail Class 45

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Off, Reverse, Engine Only, Forward. Off is mapped to the lower 2% range of the Warthog THR_FC axis.
- The Driver's (Automatic) Brake
    - This locomotive has a Brake Pin on the Driver's Brake. See the notes about the Brake Pin on British Rail Locomotives in the explanation of the Automatic (Driver's) Brake earlier in this document.
    - The Brake Pin on the BR 45 is engaged when you move to the Overcharge/Release position, not the Running Position. This is different compared to most other BR Class locomotives, but the 5 second delay and high pitch beeps are still used on this locomotive.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- Windshield wipers – The wiper switch has 3 positions, Off, Park, On. The Warthog Autopilot APPTH and APALT switch, if flipped up or down, will cycle through all positions, stopping at On or Off. Or you can flip and release the toggle switch quickly to manually cycle through the positions, for example in order to reach Park to park the wipers.
- The BR 45 only has a single sand function. It is actuated by both the Speedbrake momentary switch as well as the RDR ALTM Switch.

- The Right ENG OPER switch controls the headlights (on/off). The Left ENG OPER switch controls the Right Tail Light, but there is no keyboard combo in TSW to control the Left Tail Light. Hence, you may be better to just control these using the mouse by flipping the switches on the upper left switch panel.
- The TWCS is mapped to control the Independent Brake since this locomotive does not have a dynamic brake.

Profile 16: British Rail Class 47

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Off, Reverse, Engine Only, Forward. Off is mapped to the lower 2% range of the Warthog THR_FC axis.
- The Driver's (Automatic) Brake
    - The Driver's Brake lever has a spring loaded Release/Overcharge setting at the bottom end of travel. Actuate this by putting the Warthog Left axis in the IDLELON position (similar to Bail Off), and it will hold the Brake in Release/Overcharge as long as in that position.
    - The Brake Pin on the BR 47 is not simulated. Hence there are no delays moving into the Neutral/Shutdown position.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- Windshield wipers – The wiper switch has 3 positions, Off, Park, On. The Warthog Autopilot APPTH and APALT switch, if flipped up or down, will cycle through all positions, stopping at On or Off. Or you can flip and release the toggle switch quickly to manually cycle through the positions, for example in order to reach Park to park the wipers.
- The BR 47 has no sand function.
- The TWCS is mapped to control the Independent Brake since this locomotive does not have a dynamic brake.

Profile 17: British Rail Class 52

Here is a description of controls specific to this profile.

- The Reverser has 5 positions: Engine Only, Reverse, Off, Forward, Engine Only. The two Engine Only settings are mapped to the upper and lower 2% range of the Warthog THR_FC axis.
- The Driver's (Automatic) Brake
    - The Driver's brake has a Release position. To reach this position, live the Warthog Left Axis up and pull back into IDLELON.
    - The Driver's Brake has a Shutdown position, but does not have a Brake Pin, hence there are no issues with delays engaging Shutdown.
- Windshield Wiper
    - The wipers have 3 positions, Off, Park, and Run.
    - The Warthog APPTH/APALT switch is used to control these knobs (both left and right wiper knobs move at the same time) and works the same as most other locomotives. Flip the Auto Pilot Switch forward (to APPTH), and the wipers will go from Off to Park to Run at 1 second intervals between

changing the setting. Flip the switch backwards (to APALT) and the wipers will go from Run to Park to Off. To move only one position (e.g. Run to Park), flip the switch backwards and then back to center quickly and it will stop at Park so you can park the wipers.

- The TWCS is mapped to control the Independent Brake.

Profile 18: British Rail Class 66

Here is a description of controls specific to this profile.

- Note, the throttle lever on the BR66 is opposite vs. most engines. Back is idle throttle, forward is full throttle. The Warthog Right Throttle Axis has been mapped to mimic this action.
- The automatic brake is a three-position lever with Release, Hold, and Apply. The lever is spring loaded to default to the Hold position. The Warthog Left axis maps so that the lower 20% range is the Release setting, the 20-40% range is the Hold setting, and 40%-100% is the Apply setting. This works out so that when the Warthog lever is straight up, the locomotive lever is centered. Of course, the Warthog is not spring loaded, so don't forget to move the Left axis back to the Hold position.
- The independent brake is a three position lever with Release, Hold, and Apply. The Warthog Left axis maps so that the lower 20% range is the Release setting, the 20-40% range is the Hold setting, and 40%-100% is the Apply setting. Again, this works out so when the Warthog Left Axis is straight up, the locomotive lever is centered. There is no Bail Off.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- Speed Control Settings:
    - When using the speed controls (e.g. when loading/unloading rock), the Warthog Boat Switch Starts or Stops (enables/disables) speed control. The Warthog China Hat is used to increase or decrease the speed. These are the two Speed Control switches on the left console in the BR66 engine.
    - It is still necessary to enable and configure speed control using the overhead console, but once enabled, the above controls can be used.
- The TWCS is configured to control the Independent brake using the same control ranges as explained for the Warthog Left axis.

Profile 19: British Rail Class 101

Here is a description of controls specific to this profile.

- Driver's (automatic) Brake
    - Controlled by the Warthog Left Axis when the FLAPS switch is set to FLAPM.
    - The Driver's brake has a center notch called the lap position. The script will always sync to the lap position in case things get out of sync because TSW has a special key (/) to go to this position exactly.
    - Pull Warthog Left axis back to release the brake.
    - Push Warthog Left axis forward to apply the brake.

    - Go to the Lap position to hold (lap) the brake in its current setting.
    - If new to this locomotive, move the brake lever slowly while you are getting used to its settings and how the Warthog controls it.
- Gear Shift
    - Controlled by the Warthog Left Axis when the FLAPS switch is set to FLAPU.
    - Controlled by the Warthog China Hat Switch or the TWCS Throttle as an alternative.
- The Warthog Left Axis is not used in the FLAPD position.
- Light Switches (ENG OPER) - The Warthog ENG OPER left and right switches control the locomotive left and right light switches. Forward turns on the White lights. Backwards turns on the Red Lights.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- Windshield wipers – The wiper switch has 3 positions, Off, On, and Park. The Warthog Autopilot APPTH and APALT switch, if flipped up or down, will cycle through all positions, stopping at Park or Off. Or you can flip and release the toggle switch quickly to manually cycle through the positions, for example in order to reach On. This is the same as most other locomotives except that the Park setting is in the middle setting.
- The BR 101 has no sand function.

Profile 20: British Rail Class 166

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Reverse, Neutral, Forward, Off. Off is mapped to the upper 2% range of the Warthog THR_FC axis.
- The BR166 has only one control lever – the throttle/brake lever.
    - This is controlled by the Warthog Right Throttle Axis. All other Axis, including the TWCS are disabled for this profile.
    - If new to this locomotive, move the throttle/brake lever slowly while you are getting used to its settings and how the Warthog controls it. The TARGET Script produces beeps for each position change to help make it clear what is happening.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- The TWCS is disabled for the BR Class 166.

Profile 21: DB BR 143

Here is a description of controls specific to this profile.

- The Reverser has 3 positions: Reverse, Neutral, Forward.
- Throttle, AFB, and Cruise Control
    - The Throttle and AFB control levers are on the left side of the console. The levers are labeled as the AFB control and the Tractive/Braking Force Effort control.

    - The AFB control is the primary control used to manage locomotive speed. It sets the cruise control speed as indicated by the blue mark on the TSW indicator in the lower left corner of the screen.
        - The Warthog Right axis is mapped to control the AFB because it is the primary speed control.
    - The Tractive/Braking Force Effort control is the lever that sets the power level, but it is a secondary control on the DB BR 143 regarding speed control. Normally it is set to 100% and left in that position.
        - The Warthog Left Axis is mapped to control the Tractive/Braking Force Effort control when the FLAPs switch is in the upper (FLAPU) position.
        - The Warthog Boat Switch can also be used to control the Tractive/Braking Force Effort as this is the default for most locomotives.
        - The TWCS Throttle is configured to control the Tractive/Braking Force Effort Lever.
    - The mapping of these functions is opposite compared to locomotives like the DB BR 182 and 185.2. On those locomotives, either the throttle, or the AFB lever may be used to set locomotive speed, depending on the circumstances (the AFB can be disabled). But on the 143, the AFB is not normally disabled.
        - AFB can be disabled, it seems, using the China Hat switch to move the Auxiliary Control Lever, but for the 143, it seems as soon as you move the AFB lever again, AFB is re-enabled. This may be a bug with the 143, or maybe I just don't understand how to use it.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- Bell – the 143 does not have a bell.

Profile 22: DB BR 155

Here is a description of controls specific to this profile. The DB BR 155 is one of the more odd locomotives in TSW. It is suggested to take a little more time to understand its controls and be patient while building experience with controlling the DB BR 155 with the Warthog.

- The Reverser has 3 positions: Reverse, Neutral, Forward.
- Throttle Wheel
    - The throttle is a large wheel (like a steering wheel) on the DB BR 155. More oddly, the throttle has two methods of operation, specific setting, or Down, Hold, Up operation.
    - Rotate the Throttle CW from Throttle 0 and you increase throttle to one of 30 possible specific steps. This is like the throttles on most other locomotives.
    - Rotate the Throttle CCW from Throttle 0 and you reach three positions, Down, Hold, and UP. In the UP position, the throttle level is increased, in the Hold position, the current level is maintained, and in the Down position, the throttle level is decreased.
    - You can always set the Throttle to off by moving to the Throttle 0 position, but be aware, it will take a moment for the wheel to move to Throttle 0 if coming from a high setting (e.g. 30), and/or it will take a moment for the locomotive to reduce to Throttle Setting 0 even after you have moved the wheel to Throttle 0 (as shown by the digital readout on the dash).
    - You can monitor the throttle setting using the actual Locomotive gauges or the TSW HUD information. It is useful to understand both while learning this locomotive.

        - On the locomotive, you can see the power setting both by the power level gauge (the left gauge of the two main circular gauges on the dash) as well as the digital power setting number just below and to the right of the power level gauge.
        - For the TSW HUD, you can use the HUD messages in the upper right part of the screen which notify of throttle position changes (the actual position of the throttle wheel) as well as the HUD info in the lower right corner that shows the throttle power setting number (same as the digital readout on the dash).
        - It is important to note you can set the throttle wheel to one setting (e.g. 24) and the digital power setting number will slowly rise to meet this number, but it may take a long time as the locomotive speeds up. Hence, the Throttle wheel number and the digital throttle setting number are not the same thing. This can be a source of confusion when adjusting the throttle using the Warthog.
    - The Warthog Right axis controls the Throttle Wheel. When pulled all the way back, the throttle wheel is set to UP (and the digital throttle setting number will slowly increase). Move the Warthog Right axis forward 3 positions, and the throttle wheel will be set to Throttle 0. Move the Warthog Right axis all the way forward, and the throttle wheel is set to Throttle 30. Listen to the beeps to help understand what notch you are on. Beeps occur when entering/exiting Throttle 0, Down, Hold, and UP, but not on Throttle 1 to 30.
    - Throttle Setting by moving the wheel from Throttle 0 to Throttle 30.
        - This is straight forward, and works like most other locomotives. The digital throttle power setting on the dash will not always match the throttle number set on the wheel as it takes time for the locomotive to speed up to the chosen setting. Hence, be slow in your movements of the Warthog Right axis to slowly increase the throttle wheel to the desired setting.
        - When coming back down to Throttle 0, it is easy to overshoot Throttle 0 and instead move to Throttle Down or Throttle Hold. Again, move the Warthog lever slowly while you gain the experience running this locomotive. Throttle 0 is set when the Warthog Right axis is moved just backward from straight up.
    - Throttle Setting using Down, Hold, and Up
        - To increase throttle, pull the Warthog Right axis backward to the UP position. As long as the throttle is in the UP position, the throttle setting number will increase.
        - To hold the current speed setting, push the Warthog Right axis slightly forward to the Hold position.
        - To reduce throttle, push the Warthog Right axis slightly forward to the Down position. As long as the throttle is in the Down position, the throttle setting number will decrease until is reaches 0.
        - In TSW, the throttle wheel will remain at the UP position only momentarily (as long as the D key is pressed and held). The Warthog, of course, does not mimic this momentary function. So it will be necessary to get used to moving the Warthog axis back and forth manually.
        - Throttle 0, Down, Hold, and UP are separated on the Warthog Right axis by 10% of axis movement.
- Driver's (Automatic Brake)
    - The Warthog APU Start switch controls the Brake Key (like a Master Key, but in this case, it is a key for the brakes).

    - The Warthog Left axis controls the Driver's brake when the FLAPs switch is in the middle position.
    - To turn off the Brake Key, it is required to move the Driver's Brake to the Off position first.
- Direct (independent) Brake
    - The Warthog Left axis controls the Direct Brake when the FLAPs switch is in the down position.
    - There is no bail off function on the DB BR 155.
- Braking and Traction Force Selectors
    - The Braking Force Selector is the lever to the right of the Throttle Wheel. This lever is controlled by the Warthog China Hat switch.
    - The Traction Force Selector is the knob to the lower right of the Throttle Wheel. This knob is controlled by the Warthog Boat Switch.
- Inside Lighting
    - The instrument and Cab Lights work as normal (are mapped similar to other locomotives), but flipping the Warthog Right Fuel Flow switch forward will only illuminate some of the instrument gauge lights. The two dial gauges (the center round gauges) will not be lit. To turn on the lights for these gauges, you must flip the switch just below and to the right of the right hand gauge with the mouse. There is no keyboard combo to control these lights.
    - The illumination of the area around the Throttle Wheel, Braking Force Selector, and Traction Force Selector is not a switch. This illumination is controlled by the two red brightness knobs to the right and below the Throttle Wheel. Flipping the Warthog Right Fuel Flow Switch forward/backward will turn these knobs to 100% or 0% brightness.

DB BR 182

Here is a description of controls specific to this profile.

- The DB BR 182 reverser, throttle, AFB, and all brakes work the same as on the DB BR 185.2. Read the 185.2 description for these functions.
    - There is a lot of lag when using the Warthog to control the AFB on the 182. This is because the lever has a lot of steps (10KPH to 230KPH in steps of 5KPH). While getting used to it, move the lever slowly or you will overshoot the desired position which can get frustrating.
- The Headlights are turned on and off with the Right ENG OPER switch, but there is no key combo that can change the light configuration. Hence no switch on the Warthog is programed for that feature. Instead, the light configuration (headlight, taillight, etc.) must be manually actuated with the mouse. The switch for this is on the panel behind the engineer's seat.
- The Instrument Lights are turned on/off with the Engine Fuel Flow Right Switch as normal for most locomotives.
- The Cab Light is turned on with the Engine Fuel Flow Left Switch as it normal for most locomotives. But unlike the DB BR 185.2, the DB BR 182 can only turn on the cab light. The Desk Light can only be turned on by manually actuating the Desk Lamp switch with the mouse.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- Bell – the 182 does not have a bell.

Profile 23: DB BR 146.2 and BR 185.2

Here is a description of controls specific to this profile.

- The Reverser has 4 positions: Reverse, Off, Neutral, Forward. Off is mapped to the 15% to 35% range of the Warthog THR_FC axis. This makes it easy to set Reverse by pulling the THR_FC Axis all the way back, Neutral by setting the axis at the center detent, and Forward by pushing the axis all the way forward. This bypasses the Off position for normal operations.
- Throttle, AFB, and Cruise Control
    - The Throttle and AFB control levers are on the left side of the console.
    - The DB BR 146/185 throttle is controlled by the Warthog Right Axis. The 146/185 Throttle lever is pushed forward to increase throttle in TSW. The Warthog Right axis is programmed to mimic this direction of travel.
    - The AFB lever, to the left of the throttle lever, controls the speed for cruise control. This lever is controlled by the Warthog Left axis when the FLAPs switch is in the down (FLAPD) position. Normally that position is used to control the independent brake on other locomotives, but for the 146/185, it makes more sense to use it for the AFB Lever. The Driver's (independent) Brake is controlled by the Speed Brake button (see below).
    - AFB (cruise control) is enabled/disabled using the China Hat switch. Both the forward and backward direction of the China hat switch do the same thing – toggle AFB on and off.
    - If you need help understanding how to use AFB in general on the DB BR 146/185, see this video by Matt from Dovetail. It can be confusing knowing when and how to enable it and use it.
        https://www.youtube.com/watch?v=UjA8yU3gZL0
    - The TWCS Throttle is configured to control the AFB Lever.
    - The Boat Switch is also configured to control the AFB Lever (as this is a default for most locomotives).
- Train Brake and Electric Brake
    - The DB BR 146/185 Train (automatic) Brake and Electric (dynamic) Brake (the two levers which are side by side on the right side of the console) are normally locked together. To operate these brakes, you would normally move the Train Brake (the left lever of the two), and the Electric Brake will follow.
    - Similar to other locomotives, the Warthog Left axis controls the brakes as follows:
        - The Train Brake is controlled by the Warthog Left axis when the FLAPs switch is set to the middle (FLAPM) position.
        - The Electric Brake is controlled by the Warthog Left axis when the FLAPs switch is in the up (FLAPU) position.
    - Normally you would only move the Train Brake, and the Electric Brake will follow. But if you move the Electric Brake lever, it will decouple the two levers, and the Electric Brake lever can be moved independently. To resync the Electric Brake to the Train Brake, move the Train Brake lever past the Electric Brake's position, and the two will lock to each other again. This is all a function of how TSW works. The TARGET script correctly implements these features.
    - Train Brake and Electric Brake Release.
        - Both the Train Brake and Electric Brake have a Release position. Actuate this position by moving the Warthog Left axis to the IDLELON position.
        - If you move the Train Brake to Release, the Electric Brake will follow.

        - If you move the Electric Brake to release on its own, the Train Brake will not follow, and the two levers will be decoupled. But they will be recoupled as soon as you move the Train Brake to match the Electric brake's position.
    - The Train Brake and Electric Brake levers on the DB BR 146/185 pull **backwards** to increase braking, which is opposite vs. most locomotives in TSW. The Warthog Left axis is programmed to push **forward** to increase braking (opposite to the 146/185 brake levers). This was done to allow the IDLELON position to be used on the Warthog to enable Brake Release.
- Direct (independent) Brake
    - The Direct Brake is a 3 position lever in the DB BR 146/185. The positions are Release, Hold, and Apply (lever forward, middle, backwards respectively).
    - Normally the independent brake is programmed on the Warthog's Left axis with FLAPs in the up position (FLAPU). But since we need that lever for the AFB Cruise control, for the DB BR 146/185, the Direct (independent) Brake is controlled by the Speed Brake switch as this switch perfectly mimics how you would use the Direct Brake.
        - Speed Brake back (SPDB) increases braking. This is a momentary position on the switch. When released, it will automatically switch back to the middle (SPDM) position, holding the Direct Brake in the Hold position.
        - Speed Brake Forward (SPDF) decreases braking. This is not a momentary position. Normally when you want to release the brakes, you would move the Speed Brake switch to the forward position and leave it there.
- Sand - The RDR ALTM switch controls sand. The Speed Brake switch which controls sand on some locomotives is not used for the 146/185 since it is programmed to control the Direct Brake.
- Master & Instrument Lights and Console Lights
  The Master & Instrument Light switch turns on both the Instrument lights on the console as well as the external headlights. Signal/Headlight Selection and Headlight Brightness is controlled by a different switch (see below). What follows is an explanation of how to control these lights with the Warthog.
    - Short explanation:
        - Flip the Warthog Fuel Flow Right Switch forward and you will turn on both the Master & Instrument Lights and the Console Lights. Cycle this switch forward and back twice and you will cycle through all the Instrument and Console Lights settings.
        - Normally you would just flip the Warthog Fuel Flow Right switch forward and leave it.
    - Long explanation:
        - The Master & Instrument lights are the main console lights (turns on everything on the main console desk). The switch that controls these lights has 3 positions, forward or backwards turns the lights on. The center position turns the lights off.
        - TSW uses the I and SHIFT-I key combos to control this switch. The I key moves the switch forward one position. SHIFT-I moves the switch back one position.
        - The Console Light is a small light that illuminates the controls below the left side of the main console desk. The control switch for these lights is either on or off.
            - TSW also uses the I key to **toggle** this switch, but not the SHIFT-I key combo.
        - The result of this is, if all these lights are off, pressing the I key will turn both on. Pressing I again will turn the Console Light off, but not the Master & Instrument Lights.
        - To control these lights with the Warthog, like all other locomotives, the Engine Flow Right Switch controls the instrument lights. Why explain all this? Because if the lights are off, if you flip the switch forward, both the Master & Instrument Lights and Console Lights will turn on. Flip the switch backward, and the Master & Instrument Lights turn

off, but not the Console Lights. To turn off the Console Lights, flip the switch forward and backward again, and all these lights will be off. This is because flipping the Warthog switch forward presses the I key. Switching it backward presses the SHIFT-I key.

  - Hopefully that all makes sense now, just in case anyone cared. 😊 But remember, the other key detail is, this switch turns on the headlights, which is not like the configuration for most other locomotives.
- Signal Lights and Headlight Brightness
  - The Signal and Headlights are selected in the DB BR 146/185 by a rotary switch on the rear panel behind the engineer's seat. It continuously rotates CW or CCW.
    - The Warthog ENG OPER Left switch controls the Signal Lights setting. Press it forward to move the switch CW, backwards to move the switch CCW.
  - The Signal Light or Headlight Brightness is controlled by a switch on the main console desk. The switch has 4 positions, Signal Light Reduced (momentary), Signal Light Normal, Headlight Reduced, Headlight Bright.
    - The Warthog ENG OPER Right switch controls the Signal Light and Headlight Brightness. Toggle the ENG OPER Right switch forward (EFRNORM) to increase the switch setting, and backwards (EFROVER) to decrease the setting. If you hold the switch forward or backwards, the brightness switch will cycle to the most forward or most backwards state respectively. Because the Signal Light Reduced switch setting is momentary, when you push the ENG OPER Right switch back to the middle position, the light intensity will revert back to Signal Light Normal.
- Desk and Cab Lights
  - The Warthog Fuel Flow Left switch controls the Desk or Cab Lights. Normally, the switch toggles the Desk Light on/off. But when using MSP + the Fuel Flow Left Switch, it toggles the Cab Light. See below using the Fuel Flow Left switch position names:

    EFLNORM – turn Desk Light on
    EFLOVER – turn Desk Light off
    MSP+EFLNORM – Turn Cab Light on
    MSP+EFLOVER – Turn Cab Lights off
  - Normally you would just flip the Warthog Fuel Flow Left switch forward and leave it to get the desk light on.
- Horn controls
  - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
  - The low pitch horn sounds when the pinky switch is flipped backwards.
- Bell – the 146/185 does not have a bell.

Profile 24: DB BR 766.2 DBpbzfa and 767.2 DABpbzfa Control Cars

Here is a description of controls specific to this profile.

- The APU Start switch controls the Brake Key on the 766 and 767 cab cars.
- The Reverser has 4 positions: Reverse, Off, Neutral, Forward. Off is mapped to the 15% to 35% range of the Warthog THR_FC axis. This makes it easy to set Reverse by pulling the THR_FC Axis all the way back, Neutral by setting the axis at the center detent, and Forward by pushing the axis all the way forward. This bypasses the Off position for normal operations.

- Train Brake and Electric Brake
    - The DB BR 766 and 767 Train (automatic) Brake and Electric (dynamic) Brake (the two levers which are side by side on the right side of the console) are normally locked together. To operate these brakes, you would normally move the Train Brake (the left lever of the two), and the Electric Brake will follow.
    - Similar to other locomotives, the Warthog Left axis controls the brakes as follows:
        - The Train Brake is controlled by the Warthog Left axis when the FLAPs switch is set to the middle (FLAPM) position.
        - The Electric Brake is controlled by the Warthog Left axis when the FLAPs switch is in the up (FLAPU) position.
    - Normally you would only move the Train Brake, and the Electric Brake will follow. But if you move the Electric Brake lever, it will decouple the two levers, and the Electric Brake lever can be moved independently. To resync the Electric Brake to the Train Brake, move the Train Brake lever past the Electric Brake's position, and the two will lock to each other again. This is all a function of how TSW works. The TARGET script correctly implements these features.
    - Train Brake and Electric Brake Release.
        - Both the Train Brake and Electric Brake have a Release position. Actuate this position by moving the Warthog Left axis to the IDLELON position.
        - If you move the Train Brake to Release, the Electric Brake will follow.
        - If you move the Electric Brake to release on its own, the Train Brake will not follow, and the two levers will be decoupled. But they will be recoupled as soon as you move the Train Brake to match the Electric brake's position.
    - The Train Brake and Electric Brake levers on the DB BR 766/767 pull **backwards** to increase braking, which is opposite vs. most locomotives in TSW. The Warthog Left axis is programmed to push **forward** to increase braking (opposite to the 766/767 brake levers). This was done to allow the IDLELON position to be used on the Warthog to enable Brake Release.
- Direct (independent) Brake
    - The Direct Brake is a 3 position lever in the DB BR 766/767. The positions are Release, Hold, and Apply (lever forward, middle, backwards respectively).
    - Normally the independent brake is programmed on the Warthog's Left axis with FLAPs in the up position (FLAPU). But since we need that lever for the AFB Cruise control, for the DB BR 766/767, the Direct (independent) Brake is controlled by the Speed Brake switch as this switch perfectly mimics how you would use the Direct Brake.
        - Speed Brake back (SPDB) increases braking. This is a momentary position on the switch. When released, it will automatically switch back to the middle (SPDM) position, holding the Direct Brake in the Hold position.
        - Speed Brake Forward (SPDF) decreases braking. This is not a momentary position. Normally when you want to release the brakes, you would move the Speed Brake switch to the forward position and leave it there.
- Headlights, Instrument Lights, Signal Lights, and Cab Lights
    - The Headlights and Instruments Light switch is controlled by the Warthog Right Fuel Flow Switch. This turns on the instrument lights as well as the external headlights.
    - The Warthog Right ENG OPER switch controls the Headlights switch.
        - There are 4 positions on this switch in the cab car, Signal Lights Reduced, Signal Lights Normal, Headlights Reduced, and Headlights Bright.

        - Toggle the Right ENG OPER switch forward to move the cab car Headlight switch forward by one position. Toggle the Right ENG OPER switch backwards to move the cab car Headlight switch backwards by one position. Do this multiple times to cycle through the switch positions.
    - The Left ENG OPER switch is not used on these cab cars.
    - The Headlight configuration is controlled by a switch on the back panel, but there is no key combination that can control this switch. Hence, it is necessary to set this switch using the mouse.
    - The Warthog Left Fuel ENG Switch controls the Cab Lights. It only turns on the Cab Lights. You can also turn on the desk lights manually. If done sone, then the Left Fuel Eng switch will now control the Desk Lights.
- Horn controls
    - The high pitch horn sounds with the APENG button or when the Pinky Switch is flipped forward.
    - The low pitch horn sounds when the pinky switch is flipped backwards.
- Windshield wipers
    - The 766 and 767 have a 4 position wiper switch, Park, Off, Interval, and Lasting.
    - If you flip the Autopilot Switch forward (APPTH), the script will flip the cab car's wiper switch forward one position every 1 second. If you flip the Autopilot Switch backwards (APALT), it does the same, moving the wiper switch back one position per second.
    - You can quickly flip the switch manually to move forward or backward a single setting, or flip it, wait for the desired setting to be chosen, and then flip it back to center.
- Sand
    - The Speedbrake switch controls sand. SPDB enables sanding front, and SPDF enables sanding back.
    - The RDR ALTM switch enables sanding front.
- Bell – the 766 and 767 control cars to not have a bell.

Profile 25: DB BR 1442-1 Talent 2 EMU

Here is a description of controls specific to this profile.

- The Throttle/Brake lever is controlled by the Warthog Right Axis.
    - The throttle/brake lever has a center detent position which is 0 throttle, 0 braking.
    - There is also an Emergency Brake position when the lever is pulled all the way back.
    - Because of the nature of how the Warthog controls the throttle/brake lever, if you are new to using the Warthog TARGET Script, or the DB BR 1442, take your time and move the levers slowly while you figure out these notched positions . It is easy to over shoot the center detent or accidentally go into Emergency Braking when not intended.
    - Listen to the beeps (assuming you haven't disabled beeps) to help learn where the center position and Max Braking detents are to avoid frustration.
    - To sync the Warthog Right Axis and the DB BR 1442's throttle/brake lever, it is best to put the reverser in the Neutral position, push the Warthog right axis all the way forward or backward as needed, then recenter. Then you can enable forward or reverse travel on the reverser. Or use the A/D keys to match the in game lever with the Warthog axis.
- The Indirect Brake is controlled by the Warthog Left Axis.

    - In TSW, the lever is pulled back to increase braking, but the Warthog Left axis is configured to push forward to increase braking. This was done for consistency with other locomotives and to allow the Warthog IDLELON position to be used to sync the Warthog axis to the in game lever.
    - The FLAPs switch must be in the center position to control the Indirect Brake. The FLAPs up and down positions have no function.
- The TWCS is not used with this profile.
- Tempomat (Cruise) Control
    - Tempomat is not explained in the TSW manual for the DB BR 1442, but here is the basic way to use it.
    - The China Hat (CHF/CHB) enables/disables the Tempomat function (press it forward or backward, it doesn't matter).
    - The Boat Switch (BSF/BSB) controls the speed setting. This moves the blue cruise control line back and forth on the TSW Speed Compass as with other locomotives that support cruise control.
    - To use Tempomat:
        - Accelerate to the desired speed and enable Tempomat by pressing the Tempomat Enable/Disable button (China Hat Switch). A blue line will appear on the TSW Speed compass indicating the cruise control speed (it will match your current speed). The locomotive will be maintained at that speed.
        - To speed up or slow down, use the Boat Switch to adjust the position of the blue line up or down as desired.
        - To disengage Tempomat, move the throttle lever to the 0 Throttle position and press the Tempomat Enable/Disable button to turn it off. Then move the Throttle/Brake Lever to the desired setting to control your speed.
        - If you move the Tempomat speed to 0 KPH, Tempomat will disengage once the locomotive slows to a stop. To reengage Tempomat, use the procedure above.
        - Be aware, it seems there are a few features about the Tempomat that are not fully implemented or buggy in TSW. Some of this probably just be lack of documentation on how to use it, but the above procedure is easy and works well.
- The Headlights are controlled by either of the ENG OPER switches (they both do the same thing.
    - The headlights have 4 positions, Off/Taillights, Marker Lights, Dimmed Bean, and High Beam.
    - Flip forward and release the ENG OPER switch to move the Headlight switch to the next forward position. Flip backward and then back to center to move the Headlight switch to the previous position.
- The Cab Lights are controlled by the Left Fuel switch (EFLNORM/EFLOVER) as normal except…
    - Flip the switch forward/back and it turns on the desk lamp on/off.
    - When MSP is pressed, flipping the switch forward/back controls the cab light on/off.
- Instrument Lights are controlled by the Right Fuel switch (EFRNORM/EFROVER) as normal except…
    - There is no actual instrument light on/off switch. Instead, there is a dimmer control. Flipping the Warthog switch forward or backward simply adjusts the dimmer control up or down.
    - The switch sets to dimmer to 80% max as it seemed 100% was a little to bright.

How to Disable Audio Features of the Script

Disable wBeep.exe and the Beep() Function

If you don't want to use wBeep.exe to get audible feedback from the scripts, make the following change. This variable is found near the top of the ***Drakoz_TrainSimWorld_Warthog.tmc*** file. When disabled, it is not necessary to copy wBeep.exe to C:\bin\Sounds on your computer. I strongly recommend trying the beeps first before deciding to disable it. Because there are no detents on the Thrustmaster Throttle Left and Right axis, the beeps are a solution to give feedback. To understand more about wBeep.exe and what it is doing, see the section titled *Beep() Function – Calling wBeep.exe* in the *Technical Details* section.

Default setting – this enables beeps:

```
// Disable Beep() output - disabled =0, enabled =1
define     BeepEnabled          1
```

To disable beeps, make the following change (replace 1 with 0):

```
// Disable Beep() output - disabled =0, enabled =1
define     BeepEnabled          0
```

Disable Text to Speech and Speak() Function

If you want to turn off the speech function for the script, make the following change. This variable is found near the top of the ***Drakoz_TrainSimWorld_Warthog.tmc*** file.

Default setting – this enables speech:

```
// Disable Speak() output - disabled =0, enabled =1
define     SpeakEnabled         1
```

To disable beeps, make the following change (replace 1 with 0):

```
// Disable Speak() output - disabled =0, enabled =1
define     SpeakEnabled         0
```

How to Change the Location of the Sounds Folder

The default location for storing the Sounds (.wav file, wBeep.exe, and sWavPlayer.exe) is c:\bin\Sounds. If you want to change this to a different location, make the following change. This variable is found near the top of the ***Drakoz_TrainSimWorld_Warthog.tmc*** file.

The default folder location is specified as follows:

```
define     SOUNDS_Folder        "C:\\bin\\Sounds\\"
```

To change the location, replace “C:\\bin\\Sounds\\” with your new folder location. Make sure to enclose the entire folder path in quotes, use \\ in place of \, and make sure the path ends with \\, just as shown above.

Using the Logitech/Saitek Throttle Quadrant Profiles

Refer to the Excel spreadsheet to see the mapping for the Saitek Throttle Quadrants. Although you can control all the brake levers with the Warthog Left axis, it is much nicer to have a Throttle Quadrant set up so you have three separate levers to control the brakes.

The Saitek left and middle (X and Y) axes were configured to use Directional Axis mode. This mode is very simple. Move the lever up and it presses one key. Move the lever down and it presses another key. The right (Z) axis is not used.

When moving the Throttle Quadrant levers, move them slowly. Unlike the Warthog that guarantees that every key press will be sent to TSW, which keeps things in sync and gives great precision, the Saitek programming software will skip keypresses if you move the lever too quickly and fall short when you reach the end of travel. It works OK for the automatic brake and independent brake, but this way of doing keypresses simply does not work for the throttle or throttle/brake lever in TSW.

The Throttle Quadrant, however, has a nice feature missing on the Warthog – the Reverser position at the bottom of the lever throw. The Warthog has a similar position, but the Throttle Quadrant’s position has a physical notch you have to overcome to use it which makes it perfect for the automatic and independent brakes.

Automatic Brake using the Saitek Throttle Quadrant

The Throttle Quadrant’s left lever is mapped to move the engine’s automatic brake lever, but it uses only short keypresses. It is not intended to overcome the notched areas that require long presses. Instead, the T1 and T2 buttons perform a long press to get past any notches.

When the Left lever is pulled back to the Saitek’s Reverser position, it does two long presses which will move the automatic brake from Service to Minimum Reduction to Release. Then when moved out of the Reverser position, two long presses are made again, which moves from Release to Service.

To move to the Minimum Reduction, Handle Off, or Emergency positions, press the T1 or T2 buttons as needed to create the long presses.

Independent Brake using the Saitek Throttle Quadrant

The Throttle Quadrant’s middle lever is mapped to move the engine’s independent brake lever. It only does short presses as long presses are not needed.

To actuate Bail Out, press the middle lever down into the Saitek’s Reverser position. To release Bail Out, pull the lever out of the reverser position.

Technical Details

This section will be slowly updated over time to explain various technical details of the script. If you have any difficulty understanding this script, don't hesitate to post to the forums, or email me (see my contact info at the end of this document) and I am happy to explain things.

This section assumes you have some familiarity with TARGET scripting (not the TARGET GUI, but the TARGET Scripting Editor). It also helps to understand C Programming at least a little. If you don't have such experience, don't be afraid to dive in. That's how you learn. But you'll want to keep the TARGET Scripting Documentation handy to understand things. TARGET is a C like language. It has a lot of power, but lacks a few basic features of C. For the average TARGET Script user, however, Thrustmaster has simplified the language down to relatively simple MapKey(), MapAxis() and KeyAxis() commands.

Originally the information below was written thinking some people may want to modify this script to match their Thrustmaster devices, for those that don't own a Warthog. Though this is still possible, keep in mind that because the Warthog Throttle has 3 axes, this is a significant advantage vs. the other TARGET compatible devices.

In fact, the only TAGET compatible devices that make sense to use as a train controller are the Warthog Throttle, HOTAS Cougar Throttle, or the TWCS. It doesn't make much sense to use a Joystick to control TSW because the joystick is spring loaded. It will snap back to center when you release it and holding a joystick against the spring while running a train is not likely to be very comfortable. Hence, the compatible joysticks (the Warthog Joystick, HOTAS Cougar Joystick, and T.16000M Joystick) are mainly useful to provide more programmable buttons, but not as useful to control locomotive levers. The T.16000M joystick has a small axis on it, which may be helpful, though. Other TARGET compatible devices mainly will just add buttons. The Thrustmaster MFD panels, for example, add 20 buttons per panel, 40 buttons total.

Thrustmaster rudder pedals, like the joysticks, are similarly not useful for running a train because the pedals and rudder axis are spring loaded.

Editing the TARGET Script to Change Button/Switch Assignments

If all you care about is changing a few of the button assignments, you only need to understand basic TARGET Script functions like MapKey(), but also be aware of the following that is specific to my script.

A normal TARGET script has a main() section where the user implements their MapKey() statements. In the case of the TSW TARGET script, the normal MapKey() statements are not implemented in main(), but are instead implemented in a separate function called CommonConfig(), or in locomotive specific profile functions (e.g. pro_ConfigureSD40GP38). To modify the configuration for all locomotives, make changes to CommonConfig(). To modify the configuration for specific locomotives, make changes in the individual profile functions (e.g. pro_ConfigureSD40GP38). Changes should not be made in main(). Doing so may break the profile swapping features, and also, such changes will not be updated correctly when changing profiles. To understand better, see the order of operations below.

The order of operations in the script are:

1. The main() function calls pro_CallCustomConfigs() to call the required configuration specific to the selected locomotive profile.
2. pro_CallCustomConfigs() calls CommonConfig() to set up the configuration that is common to all locomotives.
3. Then pro_CallCustomConfigs() calls the profile configuration function specific to the currently selected locomotive – for example pro_ConfigureSD40GP38().

When you change to a different locomotive profile, steps 2 and 3 above are performed again.

Because of the order of operations, any mapping (e.g. MapKey functions) in CommonConfig() that are duplicated in the locomotive profile functions will override the mapping in CommonConfig(). That is why CommonConfig() is called first, followed by the locomotive profile function. For example, EACON is mapped to blow the horn for all engines in the CommonConfig() function. If you want to change EACON to turn on the cab lights only for one particular engine, add a MapKey() function to the locomotive profile function and it will replace the mapping to blow the horn only for that engine, even though it was previously mapped in CommonConfig(). When switching the profile back to another engine, EACON will go back to blowing the horn as CommonConfig() is executed again every time an engine profile change occurs.

Editing the TARGET Script to Change Axis Assignments

All axis assignments in the TSW TARGET script occur only in the locomotive profile functions. No axis mappings occur in CommonConfig(). This is because almost every locomotive in the script has different mapping for the axis, even the reverser.

Remember, this script maps axes to press keyboard buttons. No axis is mapped to control a DirectX axis because TSW does not support DirectX controls. Hence axis mapping is done using the TARGET KeyAxis() function. Changing the mapping of the Thrustmaster axes to control different locomotive levers is done by changing the axis configured in the KeyAxis() statements.

Two forms of the KeyAxis() function are used in the script.

One form is the normal form using AXMAP1(). This is used for the reverser on almost all locomotives. Here is an example:

```
KeyAxis(&Throttle, THR_FC, 'o', AXMAP1(LIST(0,40,60,100),
    CHAIN(LOCK+DOWN+Reverser_Backwards, D(400), UP+Reverser_Backwards, D(50), LOCK),
    CHAIN(LOCK+DOWN+Reverser_Forward, D(400), UP+Reverser_Forward, D(50), LOCK))
);
```

AXMAP1() creates a simple mapping. As the axis is moved forward, one key is pressed (e.g. W to move the reverser forward), and as the axis is moved backward, a second key is pressed (e.g. D to move the reverser backward). Use of the AXMAP1() form is fully documented in the TARGET documentation including use of the LIST() function to set the zones where the keys are pressed, and the CHAIN() function. The CHAIN() functions need a little explaining, though, and will be explained in more detail below.

The other form uses a custom form of AXMAP called AXMAPEXEC("KeyAxisDirectional(…);"). Here is an example:

```
KeyAxis(&Throttle, THR_RIGHT, 'o', AXMAPEXEC("KeyAxisDirectional(&Throttle, THR_RIGHT,
&pro_SD40GP38_ThrZoneList, &pro_SD40GP38_ThrActionUP, &pro_SD40GP38_ThrActionDN);"));
```

This is not documented in the TARGET documentation as it is custom to this script. If you are familiar with AXMAP1() vs. AXMAP2(), AXMAPEXEC() is used to call KeyAxisDirectional() which performs the equivalent of AXMAP1 and AXMAP2 at the same time. It is that capability that allows this script to work so well. AXMAP1() can press one key when the axis is moved forward, and a different key when the axis is moved backward. AXMAP2() can press a different key for different zones in a LIST(), but the same key is pressed when you enter a zone moving the axis forward and backwards. KeyAxisDirectional() has the ability to press different keys for different zones, but it can also press a different key when the axis is moving forward vs. moving backward for each of those zones. Additionally, KeyAxisDirectional() can press those keys for a different period of time for each zone. For example, the Automatic Brake often has several stepped positions (e.g. Release, Initial Application, Full application, Handle Off, Emergency Brake). To move from one step to another requires a different length key press to overcome the notch to move positions. Neither AXMAP1 or AXMAP2 can do that, but KeyAxisDirectional can.

Here is a breakdown of KeyAxisDirectional() as shown above.

- KeyAxis(&Throttle, THR_RIGHT, 'o', …..)
    - This is the normal TARGET KeyAxis() statement. In this example, the Warthog Throttle (&Throttle) Right Axis (THR_RIGHT) is selected to control the function. The 'o' means this axis will only work when the IO shift state is out (meaning the MSP button is not pressed) but will work in all three UMD layer states. This is all well documented in the TARGET documentation.
- AXMAPEXEC()
    - This is a new form of AXMAP created for this script.
    - AXMAPEXEC() works exactly the same as EXEC() which is completely documented in the TARGET documentation. It is a special form of EXEC() that can be used in KeyAxis() where you would normally have to use AXMAP1 or AXMAP2. This is required because EXEC() cannot be used in KeyAxis() normally unless it is part of AXMAP1 or AXMAP2.
    - In this case, the sole purpose of AXMAPEXEC() is to call KeyAxisDirectional(), but AXMAPEXEC() could be used to call any function due to an axis moving. Such functions would then be used to process the result of that axis movement, which is what KeyAxisDirectional() does.
- KeyAxisDirectional(&Throttle, THR_RIGHT, &pro_SD40GP38_ThrZoneList, &pro_SD40GP38_ThrActionUP, &pro_SD40GP38_ThrActionDN);
    - &Throttle and THR_RIGHT means the Warthog Right Throttle axis is selected for this configuration.
    - &pro_SD40GP38_ThrZoneList is the list of zones by percentage, similar to what the LIST() function provides in a normal AXMAP1 or AXMAP2.
    - &pro_SD40GP38_ThrActionUP is the list of key presses and key press durations that occur as the configured axis moves up or forward.
    - &pro_SD40GP38_ThrActionDN is the list of key presses and key press durations that occur as the configured axis moves down or backward.

KeyAxisDirectional() is explained in further detail below. Suffice to say here, changing which axis controls which locomotive lever is largely as simple as changing &Throttle and THR_RIGHT to a different Thrustmaster device and axis. Here is an example using the TWCS Throttle to control the dynamic brake on the GP38/SD40/GP40 locomotives.

```
KeyAxis(&TWCSThrottle, TTHR, 0, AXMAPEXEC("KeyAxisDirectional(&TWCSThrottle, TTHR,
&pro_SD40GP38_DynZoneList, &pro_SD40GP38_DynActionUP, &pro_SD40GP38_DynActionDN,
AXIS_REVERSED);"));
```

Here, &TWCSThrottle and TTHR are used to set the TWCS Throttle as the controlling axis. The 0 simply means that this mapping works for all shift layer states (IO and UMD, which means it works regardless of the position of the FLAPs switch or MSP button). Additionally, the AXIS_REVERSED flag reverses the direction of the axis, which is needed because sometimes different Thrustmaster devices' axes are mapped opposite in direction to other devices.

KeyAxisDirectional() and Building the ZoneList, ActionUP, and ActionDN Tables

This section explains how to build the tables used by KeyAxisDirectional(). These tables are created in the **Drakoz_TrainSimWorld.ttm** file. The examples below are simple examples whereas the tables in the actual script are often very complex. The complexity of the tables in the script is a result of the many different techniques needed to make the tables work with different locomotives.

The ZoneList, ActionUP, and ActionDN tables are made using C arrays. For those not familiar with C programming, an array is just a table of data.

ZoneList

The Zonelist is an array of Integers. Here is an example:

```
int pro_SD40GP38_ThrZoneList[] = {0,111,222,333,444,556,667,778,889,1000};
```

The array name is **pro_SD40GP38_ThrZoneList** and the **int** means the array is a list of integers. The [ ] means pro_SD40GP38_ThrZoneList is an array, and the { } are used to show the start and end of the array elements, where each element is separated by a comma.

In this example, this array has 10 numbers, which means 9 zones. The numbers are percent of axis travel where 111 = 11.1%, 1000 = 100.0%, etc. So, the zones are from 0% to 11.1%, 11.1% to 22.2%, etc. Note that AXMAP1 and AXMAP2 use the LIST() function to create a list of zones. LIST has a resolution of 1% (e.g. 25 for 25%). KeyAxisDirectional() has a resolution of 0.1% to give greater precision of axis control.

The list can contain as many numbers as needed to achieve the desired control. The example above is for the SD40/GP38/GP40 locomotive throttle, which has 8 throttle positions plus an IDLE position (9 zones). Calculating these zones is easy for small lists, but many of the zone lists in the script have upwards of 40 to 60 zones. To aid in calculating these lists, I use an Excel spreadsheet to both divide up and calculate the zone sizes, as well as print the zone sizes out as 3 digit numbers so I can quickly copy and paste the numbers from Excel into my .tmh file.

Keep in mind it is the zones we care about, not the percentage numbers. The ActionUP and ActionDN lists will give the key press action for each zone as defined in the ZoneList array. When an axis is moved into a zone as defined by the percent of axis travel, the appropriate key press action occurs for that zone.

ActionUP, ActionDN Tables

The ActionUP and ActionDN tables are almost exactly the same in their layout. Here is a simple example to match the GP38/GP40/SD40 zonelist given above. Two tables are shown, each with 9 zones. The ActionUP table is given the name **pro_SD40GP38_ThrActionUP**, and ActionDN table is given the name **pro_SD40GP38_ThrActionDN**. The table is similar to the ZoneList array above in that the entire array is enclosed in { }, and each element is separated by a common. As the comments show in the example, the ActionUP table is for when the axis is moving up or forward, which is intended to move the locomotive throttle in a decreasing direction. Hence the D key is pressed as we move to each zone (shown by the 'd') to decrease the throttle. The ActionDN table is the same except the A key is pressed (shown by 'a'). In the TSW TARGET .ttm files, instead of using 'd' or 'a' letter codes, the script is using defined words (defined using the TARGET define keyword) like *Throttle_Decrease* and *Throttle_Increase*. But the example below uses letter codes 'd' and 'a' to keep the example simple. The \/ and ^ characters in the comments represent the direction of movement through the table (\/ means moving down, ^ means moving up). Of course, anything after the // is a comment and does not affect the table.

```
//  action, delay,  freq,   dur
axisAction pro_SD40GP38_ThrActionUP[9] = {
    0,      0,      100,    0,      // zone 1 - Throttle 8 (ignored)
    'd',    100,    100,    0,      // zone 2 - Throttle 7
    'd',    100,    100,    0,      //      \/
    'd',    100,    100,    0,      //      \/  (axis moving up, throttle decreasing)
    'd',    100,    100,    0,      //      \/
    'd',    100,    100,    0,      //      \/
    'd',    100,    100,    0,      //      \/
    'd',    100,    100,    0,      // zone 8 - Throttle 1
    'd',    100,    100,    50      // zone 9 - IDLE
};

//  action, delay,  freq,   dur
axisAction pro_SD40GP38_ThrActionDN[9] = {
    'a',    100,    140,    0,      // zone 1 - Throttle 8
    'a',    100,    140,    0,      // zone 2 - Throttle 7
    'a',    100,    140,    0,      //      ^
    'a',    100,    140,    0,      //      ^
    'a',    100,    140,    0,      //      ^
    'a',    100,    140,    0,      //      ^   (axis moving down, throttle increasing)
    'a',    100,    140,    0,      //      ^
    'a',    100,    140,    50,     // zone 8 - Throttle 1
    0,      0,      140,    0       // zone 9 - IDLE (ignored)
};
```

Each row of each array represents a single zone. Each zone has 4 pieces of data, the *action*, *delay*, *frequency*, and *duration*. All values in these arrays are separated by a comma, but note, there should be no comma after the last number on the last row of the array. Remember while reading this that when the axis moves in the up direction, the ActionUP array is used, and when moving in the down direction, the ActionDN array is used. Keeping track of what is "up" and what is "down" will get confusing, so keep your comments clear so you don't come back weeks later and wonder what all these numbers mean.

Here is an explanation of the values for each row.

- Action – This is the key to press when this zone is entered.

    - Example: If the axis is currently in zone 1, and the axis is moved to zone 2, the D key will be pressed as defined by the ActionUP array zone 2 line. If the axis is then moved back to zone 1, the A key is pressed as defined by the ActionDN array zone 1 line.
    - If the action is 0, no key is pressed, the zone is just a place holder.
    - An action can only be a key press, or a key press with modifiers (e.g. 'a', L_SHIFT+'d', R_SHIFT+R_CTL+'f'). It is not allowed to use complex TARGET key press commands like PULSE, SEQ(), CHAIN(), etc. Of course, it is OK to use define statements to create macro names (e.g. *Throttle_Decrease* or *Throttle_Increase*) to make the table more clear, or easier to edit.
    - DEADZONE (not shown in the above example) is a special action. See below for an explanation.
    - Note, the first zone of an ActionUP table, and the last zone of an ActionDN table never press a key by the very nature of how the table works. So, the actions for these zones are set to 0. The action could be set to anything, but it would not matter. These zones are ignored.
- Delay – This is the time in ms that determines how long the key is pressed for that zone. In the example above, most zones are set to 100ms. Here are the other special delays values that can be used.
    - -1 – Press and Hold Key – If the action contains a key press and the delay is -1, then the key is pressed and held until released with a delay of 0 in a different zone. This is equivalent to the DOWN+ modifier in TARGET.
    - 0 – Release Key – If the action contains a key press, and the delay is set to 0, then the key is released. Be careful using press and hold as a key can be pressed and held indefinitely if the axis is not moved to a zone that releases the key. This is equivalent to the UP+ modifier in TARGET.
    - Delay with no Action – If the action is 0 (no key to press), but delay is > 0, then the delay will occur before the next zone is allowed to happen. This is equivalent to the D(x) delay function in TARGET.
- Frequency (freq) and Duration (dur) – ActionUP and ActionDN tables can generate beeps for each zone. This gives audible feedback while moving through the zones. The frequency and duration values are sent to the Beep() function. Frequency is in Hz, and duration is in ms. If either frequency or duration are 0, then no sound will be generated.

People familiar with C will notice the ActionUP and ActionDN arrays are in fact single dimensional arrays of type struct axisAction. The definition of axisAction is:

```
struct axisAction {
    int action;
    int delay;
    int freq;
    int dur;
}
```

This means that each zone or row in the tables above is a single element of the array, where each element contains an *action*, *delay*, *freq*, and *dur* value. Hence it is necessary to put brackets, [], after the ActionUP or ActionDN array name which include the number of zones in the array. In the examples above, there are 9 zones, hence we see pro_SD40GP38_ThrActionUP[9] and pro_SD40GP38_ThrActionDN[9]. Unlike the ZoneList array, it is required to enter the number of elements in the arrays for ActionUP and ActionDN.

DEADZONE as an Action

DEADZONE is a special value for an action. DEADZONE is actually defined as -1, so you could just use a -1 for the action to achieve the same result. The purpose of DEADZONE is to create a case where a zone is ignored

until you move past it. If you don't move past the DEADZONE, but instead move into the zone and then back to the previous zone, then moving back will NOT cause an action key press. This is useful for some rare cases where it is desired to create a special kind of delay or hysteresis. It is best to explain with an example.

Here is an example of this:

```
axisAction ActionUP[5] = {
    0,          0,      100,    0,      // zone 1   \/
    'a',        100,    100,    0,      // zone 2   \/
    DEADZONE,   100,    100,    0,      // zone 3   \/
    'b',        100,    100,    0,      // zone 4   \/
    'c',        100,    100,    0       // zone 5   \/
};

//  action,     delay,  freq,   dur
axisAction ActionDN[5] = {
    'z',        100,    140,    0,      // zone 1   ^
    'y',        100,    140,    0,      // zone 2   ^
    DEADZONE,   100,    140,    0,      // zone 3   ^
    'x',        100,    140,    0,      // zone 4   ^
    0,          0,      140,    0       // zone 5   ^
};
```

Where zone 3 has the action set to DEADZONE.

If we start with the axis in zone 1 and then move to zone 3, key press actions will occur for 'a' and 'b', but nothing will happen for zone 3 because it is the DEADZONE. If we move to zone 4, then action 'c' will occur. If we move back to zone 1, then nothing happens for zone 3, 'y' and 'z' are pressed for zones 2 and 1 respectively.

As a second example, though, if we start with zone 1 and move to zone 3, we get the actions 'a' and 'b', and again nothing happens for zone 3. If we then move back to zone 2, still no action occurs. The reason for this is because we did not move beyond zone 3, the DEADZONE, before reversing. See the DB BR 1442 Talent 2 Indirect (automatic) Brake for a real example of how DEADZONE is used. It was needed to solve an issue with reversing the axis at the wrong zone, which caused the brake lever to get out of sync because the Talent 2's brake lever didn't work using the normal hysteresis used on other locomotive profiles.

Note, DEADZONE (all caps) is a special key word for the action field. Throughout the configuration tables, there are many comments calling a zone a "deadzone" (lower case in the comments section). This is just a comment to clarify that a particular zone has no action (action = 0), but it is not the same as using the DEADZONE keyword for an action.

CHAIN Commands to Deal with Long Delays, or Modifyer Keys

Previously the following example was given:

```
KeyAxis(&Throttle, THR_FC, 'o', AXMAP1(LIST(0,40,60,100),
    CHAIN(LOCK+DOWN+Reverser_Backwards, D(400), UP+Reverser_Backwards, D(50), LOCK),
    CHAIN(LOCK+DOWN+Reverser_Forward, D(400), UP+Reverser_Forward, D(50), LOCK))
);
```

The purpose of the CHAIN() command in this example is to add a longer than normal delay for a key press.

Normally with TARGET, when you use MapKey() or KeyAxis() to press a key (using the PULSE+ modifer), the key is pressed for the default time as set by SetKBRate(). The problem with this for TSW is, there is no default key press time that works for all functions. Hence in the TSW TARGET Script, often a CHAIN command is used like above to press a key, wait a given delay, and then release the key. In the above example, a 400ms delay is added between pressing the Reverser_Backwards or Reverser_Forward keys and releasing these keys. Sadly, TARGET doesn't have an easier way to do this, so the somewhat complicated CHAIN commands shown above is the method that must be used.

Some notes about the above example that are specific to TSW:

- LOCK is used to make sure that the entire series of key presses, delays, and releases are performed without being interrupted by any other key press functions. This is very important for the TSW Script because it is part of what makes sure that any action you take with a controller, the key presses are given to TSW in order, without missing a key press.
- The D(50) at the end of the CHAIN is to make sure there is a slight delay after one key press sequence before another key press occurs.

There are other examples where CHAIN is used in the TSW script similar to the following:

```
CHAIN(LOCK+DOWN+L_CTL, D(100), DOWN+'a', D(200), UP+'a', D(100), UP+L_CTL, D(50), LOCK)
```

In addition to the notes above, this new example is necessary for some locomotive functions because of the modifier keys. Normally, with most games, the following should work.

```
CHAIN(LOCK+DOWN+L_CTL+'a', D(200), UP+L_CTL+'a', D(100), LOCK)
```

Where CTRL-A (the CTRL key and the A key) are pressed and released at the same time with a 200ms delay between pressing and releasing the keys. But often TSW does not work if the modifier key (e.g. CTRL or SHIFT) is pressed at the same time as the letter key (e.g. 'a'). To solve that, the first example is needed. In this example, we press the Left CTRL key first, wait 100ms, then press the 'a' key. The release sequence is similar, releasing 'a' first, wait 100ms, then release CTRL.

These techniques using CHAIN to stagger the pressing and releasing of keys are common techniques explained in the TARGET documentation. It was important to explain it here, though, to point out that TSW requires this technique much of the time.

Old Technical Details Section

This section applies to version 2 of the script. It has not been updated to match version 3, but is provided as the information is still somewhat useful, but may often be incorrect or misleading. For example, CallKeyAxisDirectional() has been replaced with the KeyAxis(xxx, AXMAPEXEC("KeyAxisDirectional();"); functions. Also, the parameters for KeyAxisDirectional() have been simplified compared to the version explained below.

The information below was written before my final release of the script, so some details may not match the actual script. Refer to my actual TARGET scripts to see exactly what I did. But the information below is still correct from a learning perspective.

To follow along, open up the .tmc and .ttm TARGET scripts for Train Sim World in the TARGET Script. Or use a program like Notepad++ and tell it that .tmc and .ttm files are C code to add proper coloring. Notepad++ is a much better text editor than using the TARGET Script editor or Windows Notepad. You can edit the files in Notepad++ and load them in the TARGET Script Editor at the same time. Make edits in Notepad++, save the file, and press the RUN button in the TARGET Script Editor to compile and run file. It will always compile and run the saved file even if the open file in TARGET doesn't match anymore. Just never save the file in the TARGET Script Editor. That will over write the saved copy from Notepad++.

Changing the IO Shift and UMD Mode Buttons

The exception to all this is the MSP button must be mapped in main() and nowhere else because it is the TARGET IO Shift button. If you want to use a different key for the IO Shift key, change both the following two lines:

```
// IO Shift and UMD Setup (%DEV1, I button, $DEV, U, D, Toggle settings)
SetShiftButton(&Throttle, MSP, &Throttle, FLAPU, FLAPD, 0);
```

and

```
// MSP is the IO Shift Key.  Only map it here. Do no map in custom profiles or CommonConfig)()
//   Doing so will break rotating through profiles.
//   Makes a bip sound when pressing MSP
MapKey(&Throttle, MSP, EXEC("Beep(8000,30);"));
```

To change the UMD Mode Switch, modify the SetShiftButton() function shown above, but also make sure to update the following MapKey() functions (found in CommonConfig()) to point to your new chosen mode buttons (e.g. replace FLAPU, FLAPM, and FLAPD with your chosen switch positions):

```
// Flaps Switch
// UMD Switch - also used to select function of Warthog Left Throttle Axis with
KeyAxisDirectional()
MapKey(&Throttle, FLAPU, 0, EXEC("SwitchUMD(SWITCHFORWARD);"));
MapKey(&Throttle, FLAPM, 0, EXEC("SwitchUMD(SWITCHMIDDLE);"));
MapKey(&Throttle, FLAPD, 0, EXEC("SwitchUMD(SWITCHBACK);"));
```

The above EXEC() statements call the function SwitchUMD() which important for allowing Left axis to control different engine levers and keep them in sync when switching modes.

Long Keypresses required by TSW

Many of the keyboard keys used to control TSW require a longer than average press, and the length of that press is different for different locomotives. For example, controlling the headlight switch (h) requires a 200-300 ms key press for some engines and a 300-400 ms keypress for other engines.

Normally, in TARGET scripts, you map a key with MapKey() as follows:

```
MapKey(&Throttle, LTB, 'h');
```

When the LTB button is pressed, the h key will be pressed and held as long as you press and hold the LTB button. Often, the PULSE+ modifier is used as follows:

```
MapKey(&Throttle, LTB, PULSE+'h');
```

When LTB is pressed, no matter how long you press and hold LTB, the h key is pressed only for a moment. It is pulsed for a period of milliseconds defined by the default time for a PULSE. This is set by the SetKBRate() function, or 25ms, which I believe is the default if there is no SetKBRate() function.

The first method above works for TSW but is sometimes inconvenient as you must always remember to press and hold the button long enough to have the desired result. Some functions in TSW require such a long press that I often "missed' getting the button or lever to move because I didn't press the button long enough. So, I tend to program using the PULSE+ modifier. The problem with the PULSE+ modifier is there is only one default delay for all PULSE+ key presses. But as pointed out, some keys require a 100 or 200 ms press, while others require a 300 or 400 ms press.

The solution is to use a CHAIN as follows:

```
MapKey(&Throttle, LTB, CHAIN(LOCK+DOWN+'h', D(250), UP+'h', D(50), LOCK);
```

Using DOWN+ and UP+ in a chain command like this allows us to set a specific delay for each button. DOWN+ presses the h key down. UP+ releases the h key. The delay between pressing and releasing h is set by D(250) in the middle which causes a 250ms delay. The LOCK+ modifier makes sure that the entire string of commands in the CHAIN() are executed without interruption by additional presses of the LTB button. The LOCK at the end unlocks it to allow other LTB presses to fire off a new pressing of the h key. The D(50) at the end is just to make sure that if you do multiple presses of the button, there is a slight delay between them. This is because when using LOCK, if you press the LTB button 10 times rapidly (faster than the D(250)), TARGET will queue up all 10 presses of the button and spit them out one after the other. Without LOCK, they would be sent coincidentally, which would cause chaos. But you also need TSW to register the button as being unpressed, and a small delay between each button press is required for that. Hence the D(50).

So if you desire to modify the MapKey() functions to fit your tastes, keep the above in mind as you may have to use the CHAIN(..DOWN+…UP+) version of a MapKey() to get the key to work correctly.

Changing the Lever Mapping and KeyPress Delays for Each Engine

This topic is too big to explain in detail, but I'll make a "few" comments and then direct the reader to follow the existing script as an example.

Engine Brake and Lever controls are mapped using the CallKeyAxisDirectional() function. This function simply calls KeyAxisDirectional() with the proper data tables specific to each engine. The data tables are configured in the .ttm file.

CallKeyAxisDirectional() looks surprisingly similar to EventHandle(). That is because CalKeyAxisDirectional() is called by EventHandle(). Every time a button is pressed, or a switch is flipped, or an axis moves on the Warthog, the TARGET service calls EventHandle(). This is our backdoor to write custom code to take over. When we are done, EventHandle() then calls DefaultMapping() to continue handling the event normally.

EventHandle() takes in three parameters, type, o, and x. Parameter o is the device (e.g. &Throttle). Parameter x is the button or axis that made the event (e.g. MSP for the Mic center push button, or THR_RIGHT for the Warthog Right Throttle axis). Parameter type is not used. In EventHandle(), CallKeyAxisDirectional() is called with the same o and x parameters. CallKeyAxisDirectional() then determines which engine profile is active, and which axis was moved, and sends the data to KeyAxisDirectional() which performs the actual key presses to make the TSW engine levers move. So to be clear, KeyAxisDirectional() is really the main point of all this. Everything else wrapped around it is just the fluff to make it work conveniently (customizable profiles, profile switching, formatted tables that are easy to edit, etc.).

KeyAxisDirectional() performs the equivalent of an TARGET AXMAP1() and AXMAP2() function at the same time. Similar to AXMAP1, KeyAxisDirectional() presses one key when an axis moves up, and another key when the axis moves down. If LIST() is used with AXMAP1(), several zones of different size can be configured so the key press occurs when crossing the oddly sized zones. Similar to AXMAP2(), KeyAxisDirectional() can press a different key depending on what zone the axis is in. Hence KeyAxisDirectional() can send one set of keys based on zones when moving up, and another set of keys based on zones when moving down. It is designed to use the same zones for axis up and axis down. Though it wouldn't be difficult to modify KeyAxisDirectional() to allow different zones for axis up vs. axis down, such a feature is not needed for TSW.

For AXMAP1() and AXMAP2(), these zones are defined by LIST(), followed by a comma separated list of key press events for each zone. For KeyAxisDirectional(), the zones and the key presses are defined by the tables in the .ttm file, but it is the same basic concept.

Finally, the tables for KeyAxisDirectional() have a pair of columns to specify a frequency and duration for a tone or beep so that beeps can be heard at specific zones to give feedback to the user.

Here is the definition of KeyAxisDirectional():

```
int KeyAxisDirectional(alias dev, int axis, alias zone, alias zonesList, alias actionUP, alias
actionDN)
```

An explanation of each parameter:

- dev – Device (e.g. &Throttle). dev is an alias for Throttle, Joystick, HCougar, etc. The variable is passed into KeyAxisDirectional() using the "address of" notation &. Hence &Throttle means the address of Throttle. Throttle is an array of int that contains all the current button and axis states for the Warthog

Throttle. For example, Throttle[MSP] equals the current state of the MSP button. The & notation is used throughout TARGET scripts to pass global variables around and allow functions to edit those variables without knowing what they are editing. Normally in C Programming this is done with pointer notation (*), but TARGET does not support pointers. It only supports &.

- axis – This is the axis from dev (e.g. THR_RIGHT) that moved and must be processed by KeyAxisDirectional().
- zone – This is an alias to the current zone for the axis. Zone is a number between 1 and the number of zones defined by zoneList. If zone = 0, the axis hasn't moved since we started the script (there is no current zone yet).
- zonesList - alias to array of int listing zone boundaries by percent (e.g. {0,250,500,750,1000} = 0%, 25%, 50%, 750%, 100%). Whereas LIST() only supports whole numbers (e.g. 25%), KeyAxisDirectional() supports fractions of a percent (e.g. 25.5%). But integers are used (e.g. 250) to avoid using floating point math and keep things efficient in the middle of the EventHandle() routine. The number of zones defined is 1 less than the number of elements in zonesList. E.g. zonesList = {0,250,500,750,1000} is 5 elements, but there are only 4 zones as the zones are between the numbers, not on the numbers. This is the same as LIST().
- actionUP, actionDN - alias to an array of type "struct axisAction" which contains actions and beeps for each zone. Each element of the array applies to a zone defined by zonesList and contains the data action, delay, freq, and dur. The action value is the key to press (e.g. 'h'), delay is how long to hold it down in ms, freq and dur are the frequency (in Hz) and duration (in ms) of a beep for that zone. actionUP is for increasing axis values (the axis moved up), actionDN for decreasing axis values (the axis moved down). If action is 0, no action is performed (no key press). If delay is -1, it means press and hold the action. If delay is 0, it means release the action (be careful with this – always make sure there is a path to release the key!). If dur is 0, no beep is performed. The value of freq doesn't matter if dur is 0.

The parameters dev and axis are passed in directly from EventHandle(). The parameters zone, zonesList, actionUP, and actionDN are defined on a per engine profile basis using the tables in the .ttm file. CallAxisDirectional() figures out which engine profile is the current profile and which axis moved, and passes all this to KeyAxisDirectional() which then performs key presses without having any knowledge of what profile is current, or what axis just moved. It only cares about the current position of the axis (newzone) relative the previous position of the axis (zone), determines if the axis crossed a zone barrier as determined by zonesList , and if so, which direction the axis moved. KeyAxisDirectional() then performs the appropriate actionUP or acitonDN key presses (and beeps) to move the lever in TSW.

The parameter zone (which is called the previous zone) is modified by KeyAxisDirectional() to contain the new zone (newzone) that the axis moved to (if it moved zones at all). All other parameters are not changed by KeyAxisDirectional().

Looking at KeyAxisDirectional(), you'll see that it does the following in this order:

- Determines what zone the axis is currently in (newzone).
- If the previous zone = 0, set zone = newzone and exit. There was no previous zone. First time through the function for this axis/engine combo.
- If the zone didn't change, exit.

    - But first check UMDJoySynced[UMDMode] to see if the axis has now synced. If so, beep and clear the UMDJoySynced[UMDMode] flag – enabling the axis to move the levers in TSW.
- If the zone changed, check UMDJoySynced[UMDMode] to see if the axis is not synced. If not synced, exit – perform no action.
- Determine if Axis moved UP or Down and perform the actionUP or actionDN action
    - Beep according to the action table
    - Perform the action (key press)
- Set zone = newzone and exit

UMDJoySynced[] is an array of 3 elements, one for each mode of the UMD switch (e.g. FLAPU, FLAPM, FLAPD) and hence one for each engine brake lever in TSW (dynamic, automatic, independent). When the UMD mode switch is changed, all elements of UMDJoySynced[] are set to 0 indicating that all 3 brake levers are out of sync with the Warthog Left axis. The next time the Left axis is moved, KeyAxisDirectional() determines if the axis is in the current zone for the brake lever in TSW. If not, then KeyAxisDirectional() ignores the axis movement and exits. If true, then UMDJoySynced[UMDMode] is set to 1 and axis movement is now allowed to move the engine brake lever in TSW. UMDMode is simply the current brake mode as determined by the Flaps switch.

Sometimes it is possible to move the Warthog Left axis faster than the script can keep up. You really have to throw the Warthog lever fast to do this, or it happens due to a Windows interruption. KeyAxisDirectional() figures this out and makes sure that the proper action is performed for every zone that was crossed. No actions or key presses are missed due to moving the Warthog too quickly. This is one of the key problems using the common key mapping software for other game controllers. The Saitek Throttle Quadrant is nice to have, but if you move the lever too fast, the Saitek software starts missing zones and sends too few key presses, causing the Saitek lever to get out of sync with TSW. Or it will mash the key presses together and send them at the same time - chaos. This TARGET script prevents that.

Again, it isn't perfect. There are still incidents where TSW seems to miss a key press. This is related to the time delay for each key press. TARGET is very good at guaranteeing the D() delays within the bounds that Windows allows. But the TARGET service cannot make it perfect, resulting in some key presses being too short and TSW fails to register them.

There is a range of delay with TSW where too short, and the key doesn't register, too long and the key will register twice and move a lever two steps. For example, for the a key to increase the throttle, for some engines the delay is between 60 and 210 ms. I will choose a delay on the high end of that spectrum (e.g. 170) in the hopes that if an interruption occurs, it will still meet the 60 ms window. But choose too close to the 210 ms value, and TSW might interpret the key press as a long key press and move the throttle two steps. The entire reason I had to write all this scripting was to overcome this issue. It is variable from one lever/button to the next and even from one engine to the next for the same lever/button control. Why Dovetail did it this way is beyond me. No other simulator/game I mess with has this problem. But it is nothing new. Trains Simulator 20xx has the same issue. Which means that this TARGET script could be updated to work with Train Simulator also, but there is already a good mod for Train Simulator to handle all this far more elegantly and for all game controllers, not just Thrustmaster. The point is, this TARGET script is capable of doing this for any train simulator (RUN 8, TS 20xx, etc.) using the same techniques.

There is a lot more that could be said about the tables in the .ttm file and why they were done the way they were done. There is no great insight I can give here. It was just hours of time spent figuring out the required

key delays for each lever on each engine and finding the right combination and number of zones and keypress delays to get the result I wanted. Now that the groundwork is done, it is very easy to add new engines as they come out.

Engine Profiles and Too Many Engines

Right now, there are only a few engine profiles, so cycling through them with a button press, beeps, and LEDs for feedback is OK. But as more engines are released by Dovetail Games, it will be ridiculous pressing the LDGH button a couple dozen times to get to your engine. And who really wants to read binary code on an LED display (well, actually, if you read this far, you are probably someone that doesn't mind binary). Anyway, this will be addressed in the future either by splitting up the script into multiple scripts based on engine type and having to run each script manually in TARGET, or by asking for user input through the TARGET Script output window, which I believe is possible.

The real issue is, I don't plan to buy every DLC for TSW, but I can help define the engine tables in the .ttm file with input from other users. That is really the reason I took the time to write this document – so that people can help define the engine profile tables for DLC that I don't own.

Beep() Function – Calling wBeep.exe

Beep() is a function that simply calls wBeep.exe using the system() command. Here is the coding for Beep():

```
//*************************************************************
// Beep()
//  Beep function - calls wbeep.exe which must be located at the path below.
int Beep(int freq, int duration)            // Beep(frequency, duration in ms)
{
    char buffer;Dim(&buffer, 64);
    sprintf (&buffer, "spawn c:\\bin\\wbeep.exe %d %d", freq, duration);
    system(&buffer);
}
```

The sprintf() function "builds" the command to be sent to the Windows command prompt to call wBeep.exe. The command it executes is:

```
spawn c:\bin\wbeep.exe %d %d
```

Where the %d %d is the frequency and duration of the beep, and spawn is just a command to run the program as its own process. To use the system() function correctly with a Windows command prompt pathname, it is required to escape the \ with a \\. If the location for wBeep.exe is changed, the path should be modified in the Beep() function, remembering the \\ in place of \.

See the readme file in the "*wBeep files*" directory for more details including the source code of the wBeep.exe program and how to make and compile it yourself. The program literally just passes the frequency and duration to the Windows Beep(f, d) system call using 1 line of code.

I have considered upgrading to something that can play .wav or .mp3 files to allow for a variety of custom sounds. Other people have done this with TARGET Scripts using the same technique used for wBeep.exe. See this forum topic at SimHQ.com for further details on wBeep.exe and other ideas.

http://simhq.com/forum/ubbthreads.php/topics/3852299

Conclusion on the Technical Details

Most of the rest of the TSW TARGET script is using common methods that are clearly explained in the TARGET Script Editor Basics manual.

Support

I am posting these files in the following location. Any questions/comments? Please post on the Dovetail forums or email me at my address listed below.

DoveTail Games TSW forums:
https://forums.dovetailgames.com/threads/thurstmaster-target-script-for-warthog-throttle-saitek-tq-profile.3634

Or if all else fails, email me…
Michael Lohmeyer
mike@akhara.com

Other Works and Links of Interest

The UKTrainSim forum is where I found the mods by CobraOne and Havner and older works by other people (see the Links below). Without their work, I would never have considered Train Simulator 20xx anything more than a novelty. I took a gamble and bought TSW hoping I could make TARGET do what I need. Sadly, extending this capability to other game controllers is a much bigger task. I would suggest starting with AutoHotKeys as it can do it, but also tell Dovetail that you want game controller support. My TSW TARGET Script has nothing to do with CobraOne and Havner's work (no code in common), but I was inspired by their desire to solve the problem.

TS2015 Raildriver Interface

CobraOne's mod – a plugin/mod for Train Simulator 20xx that provides better RailDriver support than the actual RailDriver software, as well as support for all DirectX game controllers including axis and button mapping. Also includes an overlay that provides useful information for people that want to turn off the TS 20xx HUD information at the bottom of the screen.
http://forums.uktrainsim.com/viewtopic.php?f=361&t=139830

TrainSim Helper (Joystick/Overlay) release thread

Havner's mod – a plugin/mod for Train Simulator 20xx that provides support for all DirectX game controllers but including only axis mapping to TS 20xx. Button mapping is not provided, nor is the enhanced RailDriver support provided. Also provides the overlay feature like in CobraOne's mod. In fact, the overlay started here and CobraOne adopted it into his mod. Havner and CobraOne have collaborated a great deal on these two items and deserve huge recognition for their efforts.
http://forums.uktrainsim.com/viewtopic.php?f=361&t=139304