VIRPIL CONTROLS
COMMUNITY SERVERS



VIRPIL CONTROLS

VPC Link Tool Documentation

Contents

INSTALLATION AND LAUNCH 3
SOFTWARE PROBLEMS 5
USEFUL LINKS 5
General description 6
Main window 7
Edit rule window 11
Argument type Source interface 13
Keyboard type Source interface 14
Axis type Source interface 15
Button type Source interface 16
Options window 17
Application options 18
Elite Dangerous options 19
War Thunder options 20
Il-2 Great Battles options 21
Device profile generator window 23
Get started with Link Tool 25
When device is connected in USB joystick mode 26
When device is connected in OUT-SLAVE mode 28
How to create LED rules 30
Argument rules 30
Integrations 36
DCS World Integration script 36
X-Plane 11 Integration script 43
Il-2 Sturmovik simulations family integration 51
VIRPIL Devices LED Addresses 54

2 INSTALLATION AND LAUNCH

INSTALLATION AND LAUNCH

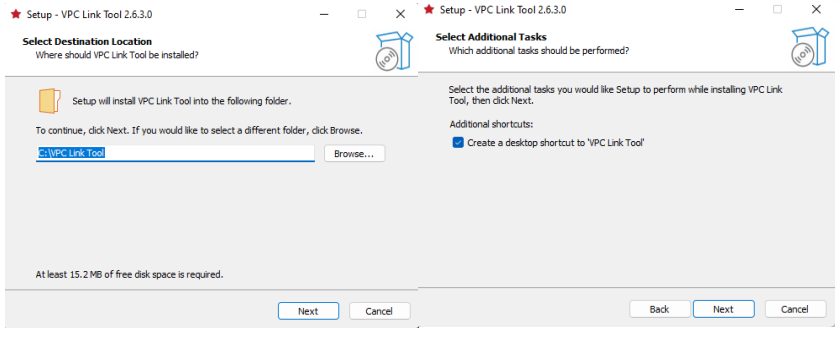
For the beginning, visit VIRPIL Updates Portal in order to download and install VPC Link Tool application.



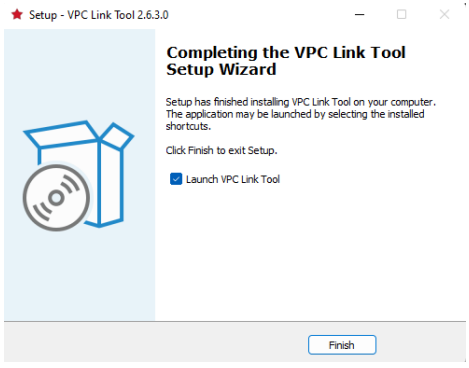
There are 2 download options:

- By clicking MORE button, you will be redirected to Link Tool page. This page contains a few sub-pages (Description, Releases, Profiles). Select specific or latest LT version to download.
- By clicking DOWNLOAD button you can get latest LT release.

After downloading setup file, launch it. You will be prompted to select app destination and link creation.



When setup will finish LT installation, LAUNCH window appears on the screen.


Setup - VPC Link Tool 2.6.3.0
Completing the VPC Link Tool Setup Wizard
Setup has finished installing VPC Link Tool on your computer. The application may be launched by selecting the installed shortcuts.
Click Finish to exit Setup.
Launch VPC Link Tool
Finish

SOFTWARE PROBLEMS

If you encounter any problems, we suggest you back up device profiles, located in *devices* folder and delete VPC Link Tool entirely from PC. Download latest stable release, install it and go through LED profiles creation process, which we will be covering in the next chapters.

If problems persist, we suggest consulting our online technical support forums at

https://forum.virpil.com/index.php?/topic/2561-vpc-link-tool/

USEFUL LINKS

VIRPIL Homepage:
https://virpil.com/
VIRPIL forum:
https://forum.virpil.com/
VIRPIL Updates Portal:
https://updates.virpil.com/
VIRPIL Servers:
https://virpil-servers.com/

General description

Link tool is a freeware, Windows Application type software, designed by VIRPIL controls, is used to set different colors for all LED-capable devices, provided by VIRPIL. The main concept of the LT is to read different aircraft arguments from famous simulations, such as:

- Il-2 Sturmovik 1946 Forgotten Battles
- Il-2 Sturmovik Cliffs of Dover / Desert Wings
- Il-2 Sturmovik Great Battles
- War Thunder
- DCS World – *requires LUA integration script*
- Microsoft Flight Simulator 2020 (MSFS2020) – *requires variables profiles*
- Microsoft Flight Simulator X (FSX) - *requires variables profiles*
- Microsoft Flight Simulator Deluxe Edition– *requires variables profiles*
- Lokheed Martin Prepar3D v4/v5 – *requires variables profiles*
- Elite Dangerous
- X-Plane 11 - *requires LUA integration script*

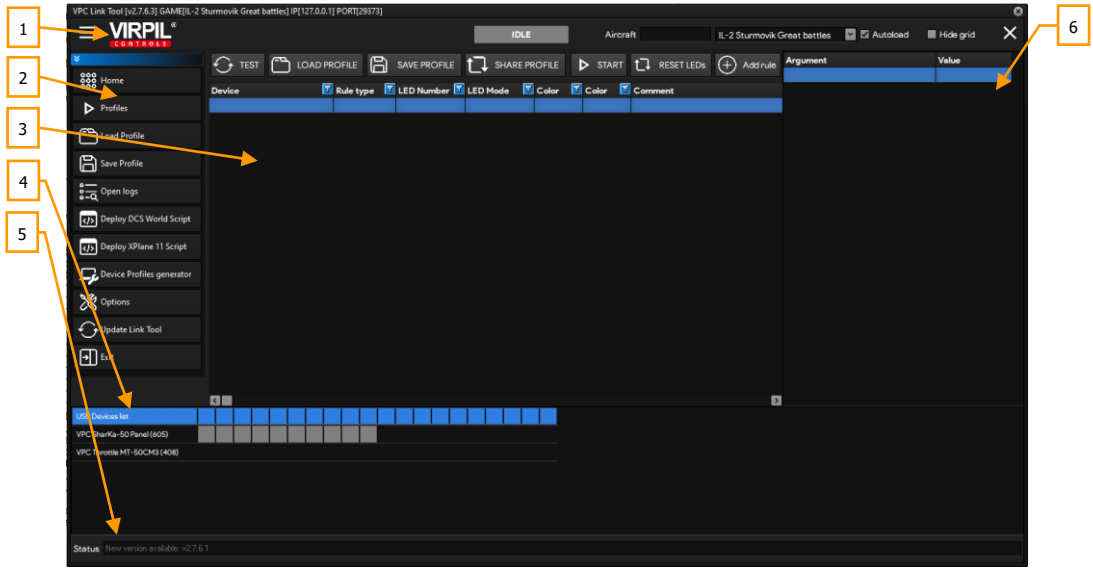
Basically, LT reads a key-value pairs from game and allows to create a LED rules, depending on arguments in-game state. We will be covering how to connect LT with all supported simulations in Integrations chapter.

Another capability of Link Tool is:

- creating LED rules, based on keyboard shortcuts;
- creating LED rules, based on any USB HID device Axes positions;
- creating LED rules, based on any USB HID device Buttons states (pressed/released).

Main window

The main LT window consist of 6 parts:



1. TOP NAVIGATION BAR. This section contains a menu toggle button (three horizontal lines, located in the left corner), application logo, LT status window (gray IDLE status, see picture above), aircraft title input field (non-editable, fills in from simulation data set), simulation select box, 2 quick options and X-button (close application).
    - Autoload option enables or disables LED profiles autoload function. When option is enabled, LT reads aircraft title input field and makes an attempt to autoload *[aircraft tile].led.json* file from *LT/autoload* folder. In case if such file does not exists in autoload folder, LT skips autoload process.
    - Hide Grid option is used to Hide arguments grid in order to increase LT overall stability and performance at runtime. When LT is started, calculation engine runs a lot of under-the-hood operations. Periodical UI updates (updating arguments values) is a quite "heavy" procedure, which uses a lot of CPU/memory resources. In case if you notice some UI freezing or delays, this option could help to increase LT performance. When option is enabled, arguments grid will be hidden, and values are not updated in the grid, but they are still being updated "under-the-hood".

    There are 2 simulation in the list that needs additional scripts – XPlane 11 and DCS World. Basic integrational scripts can be pulled from VIRPIL Official code repository by clicking Deploy buttons in the left sidebar menu.

    There are another 4 simulations in the list that needs additional arguments profiles to be created:

- Microsoft Flight Simulator X
- Microsoft Flight Simulator Deluxe Edition
- Microsoft Flight Simulator 2020
- Lokheed Martin Prepar3D v4/v5.

Elite Dangerous game integration with Link Tool requires a path to Status.json file to be set at Options UI.

Il-2 Sturmovik simulations family integration requires additional changes in game configuration files.

We will discuss profiles creation, scripts deployment and games integration procedures in the next sections.

2. LEFT SIDEBAR MENU. This block contains classic drawer menu (can be hidden). Buttons functions are:
   I. LOAD PROFILE – this button is used to load rules profile, saved before at some points.
   II. SAVE PROFILE – it is used to save rules profile as *.led.json* file extension. If AIRCRAFT window is not empty, new LED profiles will be named as *[aircraft].led.json*. But you are still able to set your desired profile name.
   III. OPEN LOGS – it is used to open logs file directly from LT. Logs are saved automatically when app is in DEBUG mode.
   IV. DEPLOY DCS WORLD Script – this menu option is used to download the latest version of the integration script from VIRPIL repository and install it (script is responsible for data transfer from DCS to LT).
   V. DEPLOY XPLANE 11 Script - this menu option is used to download from VIRPIL repository and install the latest version of the integration script, which is responsible for data transfer from DCS to LT.
   VI. DEVICE PROFILES GENERATOR – this option opens LED Devices Profiles Generator. Each VIRPIL device, registered by LT, should have its LED definitions. We will discuss device profile generation flow in the next chapters of this document.
   VII. OPTIONS – this menu option is used to open OPTIONS window, which contains general app options, such as Debug mode, profiles autoload, different simulation connection data (hosts, ports), default LED color, etc.
   VIII. UPDATE LINK TOOL – this menu option is responsible for version check. Link Tool checks for updates on every startup cycle, but if you want to check updates manually, this button will help you.
   IX. EXIT – this option is used to close LT and set all LED's to its default colors (as set by firmware).
3. RULES SECTION. This is basically a table with filtering functionality, which is representing a set of different rules (AXIS position, USB BUTTON press, KEYBOARD button press, simulation argument conditions) for different devices. Top row of this section consists of next buttons:
   I. RUN TEST – this action runs single LT cycle procedure, which consists of 4 steps:

- ASK DATA. On this step LT is making a data request to simulation (if required).
- GET DATA. On this step LT is transforming received data, depending on exchange protocol and displays it in Arguments grid.
- CALCULATE LED's. On this step LT is making LED's calculations by reading rules line by line from top to the bottom of the grid. When LT determines rule as VALID, appropriate cell, which indicates LED address (number/position), will be filled by color, as set in the rule.
- SET LED's. On this step LT goes through all devices, which are registered in devices grid and sends request with a stack of RGB colors, that should be applied to specific LED's.

II. LOAD PROFILE – the same function as described for the left drawer menu.
III. SAVE PROFILE – the same function as described for the left drawer menu. This function has a LCTRL + S shortcut.
IV. START. Starts LED's calculation cycle.
V. RESET LED's. Sets all VPC devices LED's to default color, as set in OPTIONS menu.
VI. ADD RULE. Adds new empty rule to Rules grid.

A table in this section represents a list of LED rules, which can be applied to all LED-capable VIRPIL devices. This table consists of:

- Device column. It is a VIRPIL device title.
- Rule Type column. This column represents a rule type. There are a few rule types: Argument, Button, Axis, Keyboard.
- LED number column. This column indicates a number of an LED (onboard address)
- LED mode column. There are 2 LED modes, which can be set for specific LED: Steady or Flashing.
- Color #1 column. This column contains a visual presentation of the color, which will be applied to specific LED and is used when STEADY mode is selected in the rule options.
- Color #2 column. This column is a visual presentation of the color, which will be applied as second color to specific LED in FLASHING mode and is not used when STEADY mode is selected in the rule options.
- Comment column. This column can contain any of your information. Basically, it is a description for the rule to not forget what actions, situations this rule calculates.
- Status column. This column represents a rule status (Enabled/Disabled). If Status is set to Disabled - LT will not account that rule and skip it in every iteration of LT cycle.

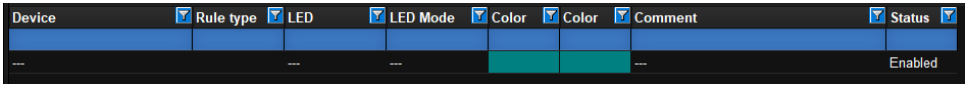
4. DEVICES GRID. This area displays a list of VIRPIL devices, recognized by LT. There are a LED cells on the right side of the Device name, which represents current LED state (color, applied to LED by LT). On the picture above you can see VPC SharKA-50 Panel detected. 10 gray cells mean that LT detected LED profile (which is manually created by user) with 10 LED's, located on a device board (Gray color is selected as Default color in OPTIONS menu):
   - Transparent state: LED is not recognized by LT.
   - Colored state: LED is recognized by LT (because LED profile was created before). Default LED color, which can be set in OPTIONS menu, is applied for every cell (LED) as basic color state.
5. STATUS BAR. This section displays info/error/warning messages or LT state in some cases.

6. ARGUMENTS GRID. This section represents a key/value data, which is exported from selected simulations. Left column of this table contains an argument name, right column represents argument's value (in case, if data is successfully obtained from selected simulation).

Edit rule window

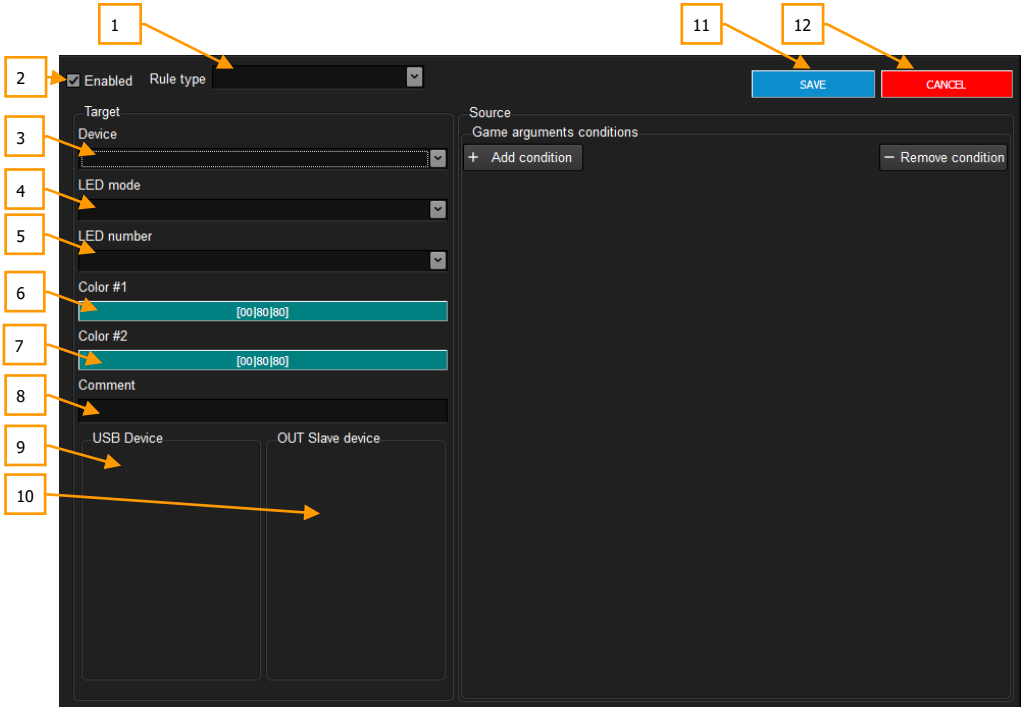
This part of LT interface is used to edit single LED rule.

Press ADD RULE button to add new empty rule into a grid.



A new record will be added to the end of the rules table with some 3-dash values in the cells. Such rule is totally empty and needs to be edited. Click twice on created row to edit a rule.

A new section will appear on the screen:



This section is separated into 2 parts: Target and Source.

Target section contains a few input fields, which determines target device, where LED state will be affected.

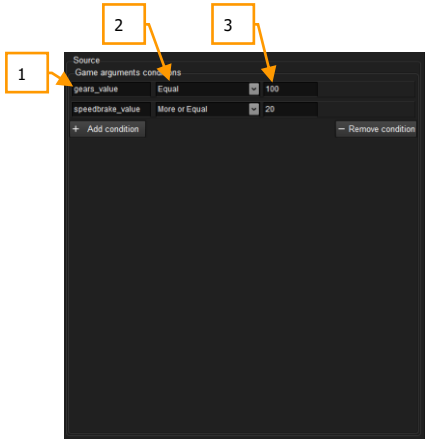
Source section content depends on selected rule type and is used to define conditions which will affect selected LED in Target section. This part of EDIT interface can define a keyboard combination, arguments conditions, pressed buttons or axis positions, that are read from any USB HID devices.

1. A drop-down menu is used to set a type of the rule. There are a few rule types that LT supports:
    - ARGUMENT. This type is used to create rules, which set LEDs states based on arguments values, which are obtained/exposed from selected simulation. In the Source On the picture above Arguments conditions UI is shown by default. Buttons on the left and right side are used to add or remove conditions rows. In general, all conditions are interpreted as a stack with AND operator. Example: Condition1 AND Condition2 AND Condition3. In this case rule will be valid in case if all 3 conditions are valid.
    - KEYBOARD. This type is used to set selected LED's color/mode, based on specific keyboard combination state (key combo is pressed). Example: if LCtrl + H is pressed, LED #3 can blink with bright Green color.
    - AXIS. This type is used to set selected LED color/mode, based on any USB HID device axis position. There are 3 condition types for axis rules: axis value is more or less in relation to specific value or between desired axis range.
    - BUTTON. This type is used to set selected LED color/mode, based on selected button state (pressed/not pressed) on any USB HID device. For example, if button 3 on USB Gamepad is not pressed – make VPC SharKA-50 Panel LED #2 lit with steady RED color.
2. Rule status checkbox. Any rule can be disabled and will not be accounted during LED's calculation cycle.
3. Device select drop-down menu. A list of available VIRPIL devices, detected by LT.
4. LED Mode select menu. There are 2 types of LED modes: Steady and Flashing. In case of Steady option selected LED will lit Steady with a color selected by Color #1 option. In case of Flashing option selected LED will change colors from Color #1 to Color #2 on every LT iteration.
5. LED number. This menu is used to select desired LED on selected device. If LEDs list is empty, then, possibly, LED profile was not created for selected device or device is not selected.
6. Color #1. This option is used to select color, which will be applied as main color in Steady mode or primary color in Flashing mode.
7. Color #2. This option is used to select secondary color, when operating in Flashing mode.
8. Comment field is used to set some custom rule description (example: argument description or entered keyboard combo).
9. USB Device picture will be shown in this section when device is selected in the menu, located at the top of Target section.
10. OUT Slave device image is shown in case when device is selected in the top part of Target section and is defined as SLAVE device in LED profile.
11. SAVE button is used to save changes to LED rule. It does not save LED rule into file or so. Also, it is closing EDIT interface and returns back to rules grid interface.
12. CANCEL button is used to cancel rule changes and returns back to rules grid interface.

Argument type Source interface

This section is used to determine custom conditions based on arguments values, arrived from selected simulation. 13 conditions (at the moment) can be defined by the user.

Every condition consists of 3 editable fields:



1. Argument name (String value). This field is editable.
2. Condition (predefined String value). This field is not editable.
3. Argument value. This field is editable and should be always Integer or Float type value (1 or 1.0).

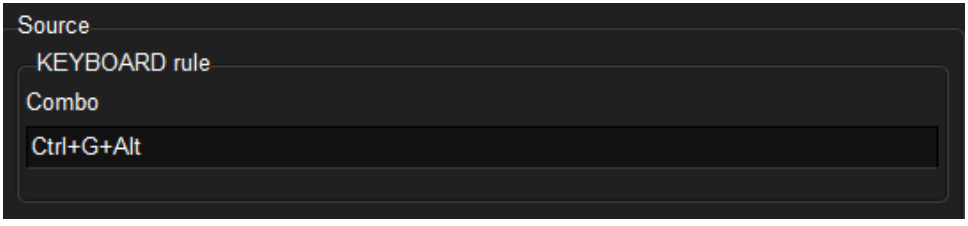
On the picture above your can see 2 conditions created for gears_value and speedbrake_value arguments. In this case rule will be valid only in case when (gears_value will be equal to 100) AND (speedbrake_value will be MORE or EQUAL to 20). Both conditions should be valid in the same time.

Generally, you can select one of 6 logical operators for every condition:

- Less
- More
- Equal
- Not Equal
- More or Equal
- Less or Equal

Keyboard type Source interface

This section is used to set keyboard combination.

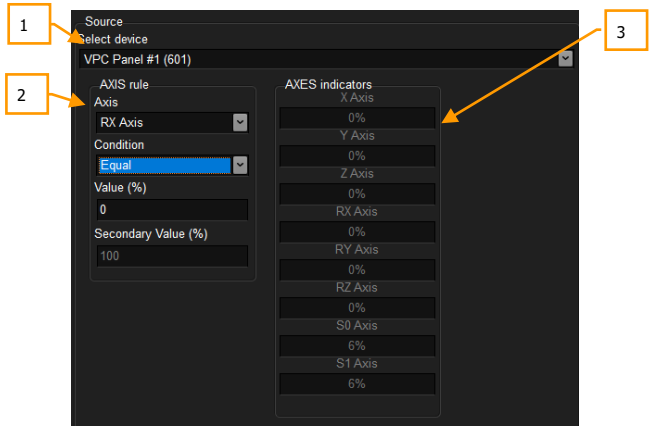


Left click on Combo field will place a focus on it, so you will be able to set keyboard combination. If Combo field is out of focus, pressed keyboard combination will not be saved.

Note! Link Tool does not emulate keyboard buttons press events. Rule will be valid only in case, when selected combination was/will be pressed by keyboard (not a joystick/gamepad or so).

Axis type Source interface

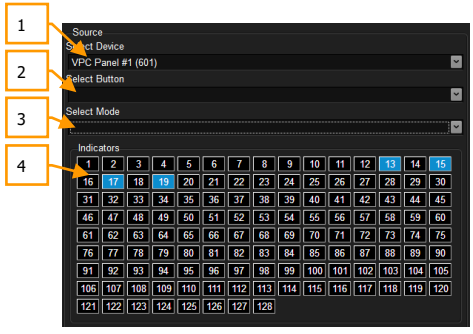
This section is used to set a desired axis and condition depending on selected axis value and consists of:



1. Select Device dropdown menu. This menu contains all USB HID devices, detected by LT, including 3rd-party USB devices.
2. AXIS rule section. This section is used to select desired AXIS, condition and primary/secondary AXIS values. Condition menu contains the same list as Arguments conditions list. It also has additional condition type BETWEEN. In case of selected BETWEEN condition – rule will account primary and secondary AXIS values at the same time. All other condition tyles does not account secondary axis value.
3. AXES indicators. This section shows an axis position in % in real time by moving Stick, Throttles or any axis-type controls, located on your device.

Button type Source interface

This section is used to set rule depending in different USB devices buttons states. At this moment there are 2 button states: Pressed, Released.

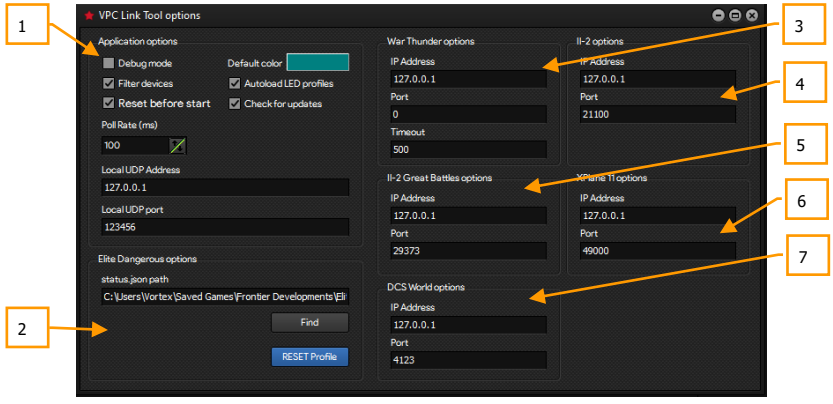


1. Select Device dropdown menu. This menu contains all USB HID devices, detected by LT, including 3rd-party USB devices.
2. Select button dropdown menu contains available buttons list, depending on selected device. If buttons list is empty – it means that device is not selected or LT does not detect any available buttons, located on selected device.
3. Select mode menu. This section is used to select button state between 2 available values: Pressed, Released.
4. Indicators section is used to test or observe pressed/released buttons on selected device, recognized by LT.

NOTE! At this moment LT supports only one button to be pressed/released at the same time during rule calculation cycle.

Options window

This section appears as another modal interface, which stops LT cycle (in case if LT is running) and blocks all interactions with main interface. Options can be opened by *LCtrl + O* combination.



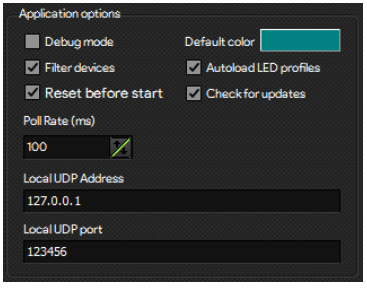
Options interface consists of:

1. General Application options
2. Elite Dangerous connection options
3. War Thunder connection Options
4. Old Il-2 connection options
5. Il-2 Great Battles connection options
6. X-Plane 11 connection options
7. DCS World connection options

Note! After making any changes just close this window, so all settings will be saved automatically.

Application options

These options are used to configure application general state. Let's have a closer look on these options list.



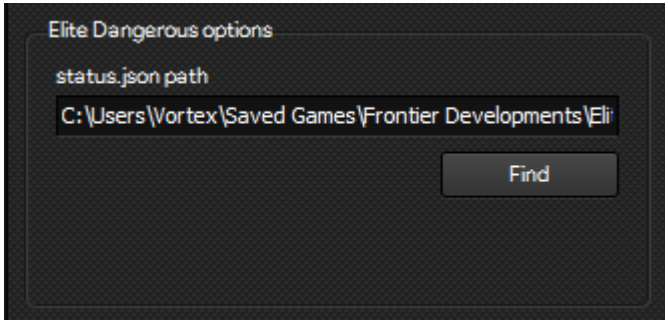
- Debug mode. This checkbox is used to enable DEBUG mode (requires LT restart). In this mode Windows Command-Line Interface appears during application startup cycle. It's a simple CLI terminal, opened in read-only mode. LT will print debug messages into CLI terminal. It's very useful information when talking-solving issues with VIRPIL support. Please, keep in mind! DEBUG mode makes a big impact on application performance. **We do not recommend to play with DEBUG mode enabled**.
- Filter devices (deprecated). This option was used in earlier versions of LT to filter joysticks and show only VIRPIL produced devices, and will removed in the future.
- Reset before start. This option is used to RESET all LEDs to Default color before running a LT calculation cycle.
- Poll Rate. This option is used to determine how often LT will read all rules and apply colors to them. Less rate – faster calculations, and vice versa.
- Default color. This option is used to set Default color, which will be applied to all LEDs when RESET BEFORE START option is enabled.
- Autoload LED profiles. This option is used for automatic rule-profiles load when LT detects any change in Aircraft field, located in the top navigation bar. All LED rules profiles should be located in *VPC Link Tool / autoload* folder and should be named as *{aircraft}.led.json* .
- Check for updates. When option is enabled LT makes a request to Updates Portal in order to check for a new version after opening application. If new version detected LT offers to download and install it. If you do not want to receive version notifications, just make sure this box is not checked.

Local UDP Address and Port options are deprecated and will removed in the future.

Elite Dangerous options

In order to make LT read all available arguments you need to set correct path to *Status.json* file. It contains different ship data values, which are updated very often.

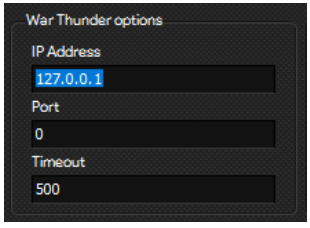
File can be found in *C:\Users\[YOUR_USER]\Saved Games\Frontier Developments\Elite Dangerous* folder.



FIND button is used to set Status.json file path using system window dialog. After selecting Status.json file full path to it will be shown in PATH input field.

War Thunder options

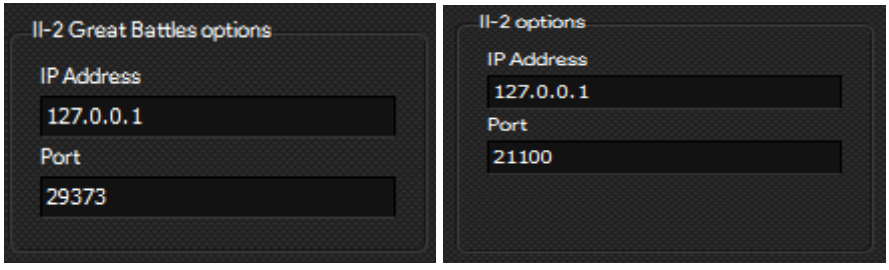
This section is used to control WT HTTP server information. Basically, WT exposes aircraft data using internal HTTP server, provided by simulation itself.



- **IP Address**. This field is used to set WT local IP address. It is always equal to localhost address.
- **Port**. This option is not used anymore and will be deleted in the future.
- **Timeout**. This option is used to determine how long Link Tool will wait any response from WT data server.

Il-2 Great Battles options

This section is used to configure UDP server address and port in order to receive aircraft telemetry data, broadcasted by IL-2 GB game.

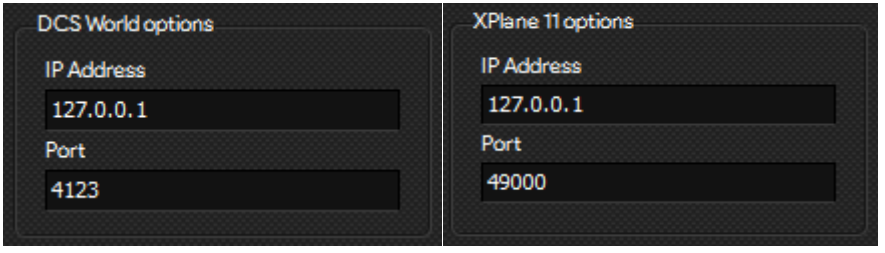


Il-2 options section is used to set UDP server address and port, which are provided by Il-2 Sturmovik 1946 Forgotten Battles and Il-2 Sturmovik Cliffs of Dover / Desert Wings games.

We will be covering Il-2 configurations in Integration chapters.

DCS World options

This section is used to set UDP server address and port, provided by DCS World integration script. By default, UDP Client, created by DCS World integration script is using 127.0.0.1:4123 address to transfer data to LT. These options should be the same in LT and LUA scripts.

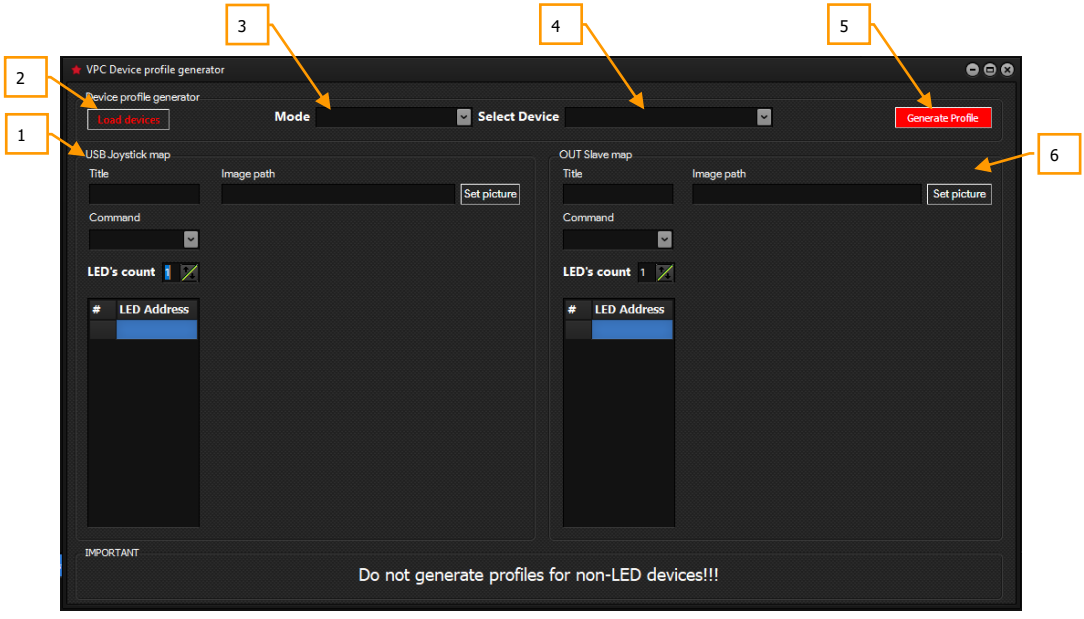


X-Plane 11 is using almost the same way to broadcast aircraft telemetry data as DCS World.

We will be covering LUA scripts installation and configuration in the next chapters.

Device profile generator window

This interface is used to create LED profiles for VIRPIL devices when you open LT for the first time. Every created profile defines how much LEDs are located on a Grip, Panel or Throttle.



1. USB Joystick configuration section. This section is used to configure selected main LED device.
   - I. TITLE field is used to set device name, which will be shown in Devices Grid.
   - II. Image path field is used to set device picture, which will be shown when editing rule as device tooltip (pictures contains LEDs numbers/addresses)
   - III. Command menu. This option is used to determine a device type. As a result, LT will send special code to VIRPIL device in order to set LEDs colors correctly. If device is a throttle, but profile contains wrong command (f.e, 101- Grips), LEDs colors won't be applied correctly.
   - IV. LEDs Count input field. This is a number field, which is used to determine how much LEDs are located on selected device. Changing this value will affect table, located under input field. Minimum value is 1.
   - V. LED Addresses table. This table is used to define LED numbers (addresses), which will be saved into LED profile. So, LT will recognize every LED and you will be able to create LED rules for specific device.
2. LOAD Devices button. This button is used to refresh devices list, which is shown in Select Device Menu.

3. Mode menu. This option is used to select device mode. As you may know, every VIRPIL device can operate in two modes: as a single USB and a SLAVE device, which is connected as AUXILARY to the main device by using OUT-AUX cable. If main device does not have SLAVE any other panels connected as SLAVE device, select *USB Joystick mode*. In case if you have SLAVE device connected via OUT-AUX cable, select *USB Joystick + OUT Slave* mode.
4. Device menu. This menu contains VIRPIL devices list, recognized by LT on a startup cycle.
5. GENERATE profile button. This button is used to save/rewrite all configurations to a JSON file, which will be located in devices folder. Profile name will look like *[PID].json* . For example, *408.json*.

OUT Slave configuration section. This section is a copy of USB Joystick section and is used to determine all necessary configurations for SLAVE device. Every slave device should receive command 103, that's why SLAVE command menu contains only one option. This section should be used in pair with *USB Joystick + OUT Slave* mode for device, which is connected by OUT-AUX cable.

Note! You may probably face a question: "Where I can get LEDs numbers or addresses?". The answer is simple. All LED devices pictures are located in assets folder and can be look like:

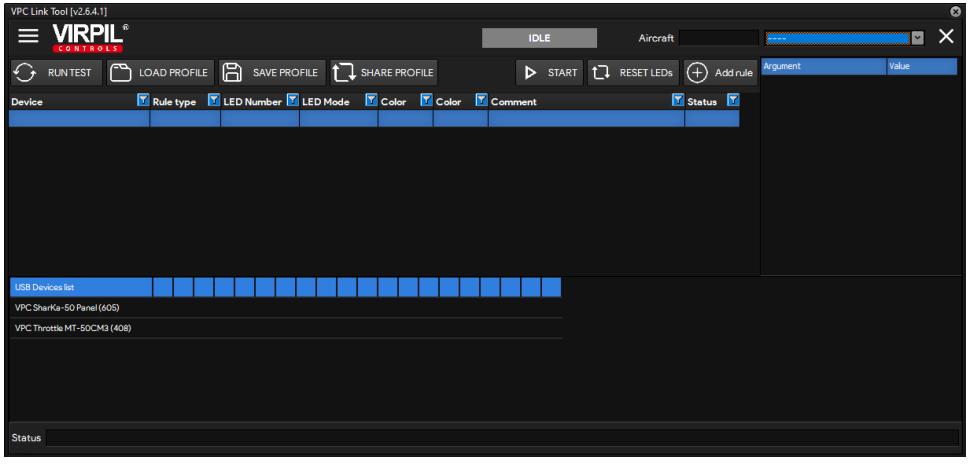


On the picture above, there are 6 LEDs (marked by RED color) located on CM2 throttle. Every LED has its number. Actually, this number represents LED address on the board.

Do not create any profiles for non-LED devices (f.e. Rudder Pedals). Otherwise, Link Tool will not send proper requests to other LED-capable devices.

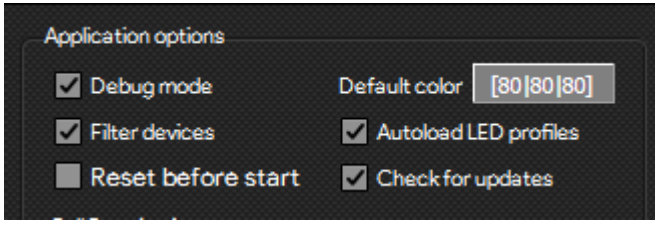
Get started with Link Tool

After first LT launch on your system or getting new VIRPIL LED-capable device, you will need to create a LED profile. Basically, it's a definition of LED buttons, located on any of your VIRPIL device.



As you can see on the picture above, LT detected two VIRPIL devices. Let's configure some options first. Using left menu, click OPTIONS button or press LCtrl + O to open Options menu.

At this point, we will set Default color setting to any visible color, so you will be able to see all defined LED cells in Devices list. This option is also affecting LED default state in case if there is no any working rule for LED.

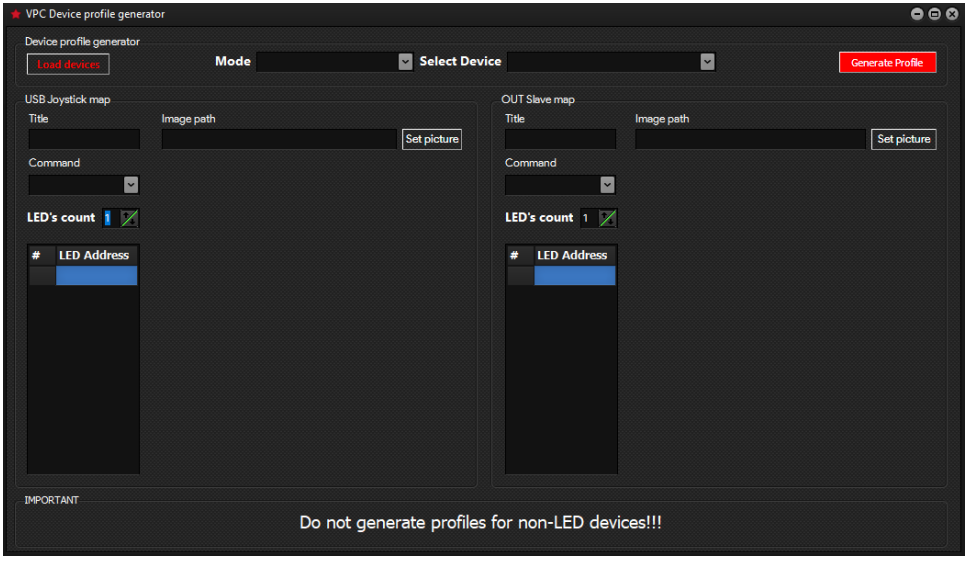


After changing default color – restart LT.

Next, we need to define all LEDs for every LED-capable device. Open Device profile generator by pressing LCtrl + D or by clicking Device Profiles Generator button, located in the left menu.

New window will appear on your screen.

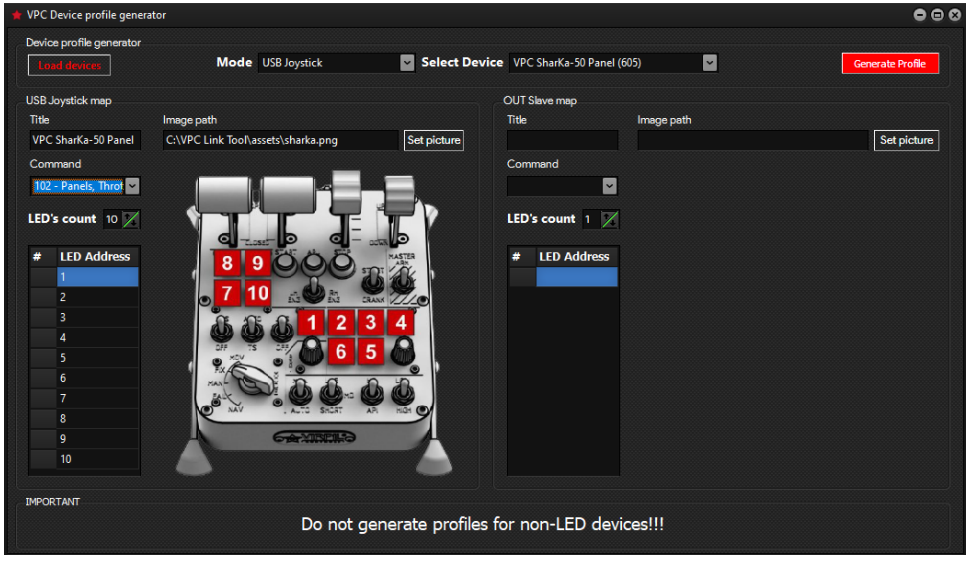
When device is connected in USB joystick mode



Now let's create a LED profile for VPC SharKA-50 panel.

1. Select mode. In case if device is connected straight to the PC by using USB cable, select USB-joystick mode.
2. Select device. Using dropdown menu select VPC SharKA-50 panel (605) option. Red cross device image should appear in the Image field.
3. Set image. Using Set Picture button, navigate to assets folder of LT installation location and select sharka.png file. Now you should be able to see all LED numbers, located on SharKA panel, which we will use in the next steps.
4. Double check USB Joystick title in the top left part of USB Joystick MAP section. Title field should be filled automatically. You are also able to set your custom device name.
5. Set Command type. Basically, if device is connected straight to the PC in USB Joystick mode, depending on its type, you need to select proper command, which will be sent to device. 101 command is used for Grips/Sticks. 102 command is used to send commands to Panels/Throttles. In our case, SharKA-50 panel requires to set command to 102.
6. Set LEDs count. Using picture, selected in the step #3, determine how much LEDs you have on board of your device. In our case of SharKA-50 panel, there are 10 LEDs.
7. Link Tool automatically will set all LEDs addresses (you can find all addresses on the picture, selected in step #3).

After finishing all configurations in device profiles generator, you should see next picture:



As you can see, OUT Slave map section is not set, because SharKA-50 Panel is connected to the PC by using OUT-USB cable (in USB Joystick mode).

Click Generate profile button. Link Tool will restart automatically. After restart, you should see your VPC SharKA-50 panel in the devices list with 10 LEDs (ten gray cells, Gray is set to default LED color).



When device is connected in OUT-SLAVE mode

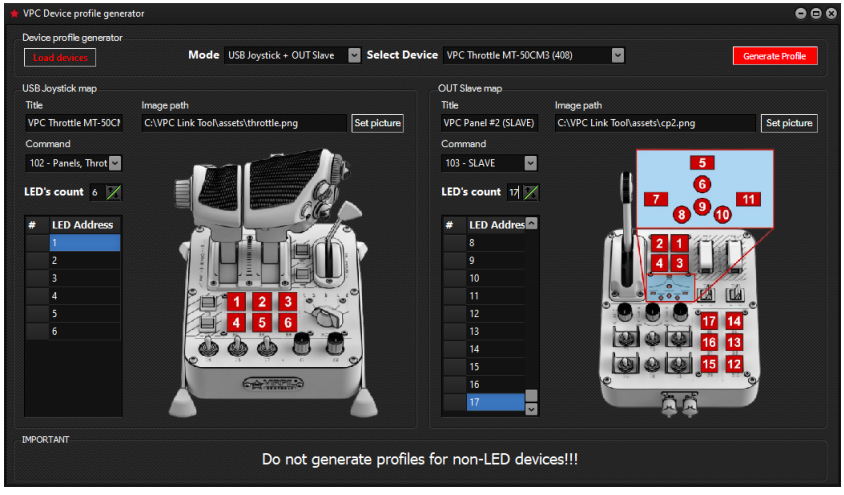
Let's assume, that we have a set of VPC Panel #2 + VPC Throttle MT-50. Panel #2 is connected to throttle using OUT Slave mode in order to minimize busy USB ports count on your workstation. Take a look on Link Tool Devices List.



As you may notice, we have only Throttle visible in the list. At the same time, Panel #2 is missing. Also, LEDs for Throttle are not defined – dark row on the right side, after Throttle cell with its fully qualified name.

Lets fix this issue making a new LED Profile. Open Device Profiles Generator by pressing LCtrl + D.

1. Select mode. When creating profile for OUT-Slave devices chain, select USB Joystick + OUT Slave value.
2. Select device. In our case, you need to select main device in OUT Slave chain. This is VPC Throttle device.
3. Select picture.
4. Set Command. In our case, 102 command should be selected.
5. Set LEDs count. From step #3 you can determine that CM3 throttle has 6 LEDs on board. LT will set all LED addresses automatically. In case of custom addresses, use device picture. VIRPIL team after releasing any new device also provides a picture of LED addresses in case if device has any LEDs.
6. Set OUT Slave device title. In our case, we need to set it as VPC Panel #2 (SLAVE). Use (SLAVE) suffix to "tell" LT that device is connected in OUT-Slave mode.
7. Set SLAVE device image (cp2.png) .
8. Set command. All SLAVE devices are using 103 command.
9. Set LEDs count. In case of VPC Panel #2 – 17 LEDs on board.

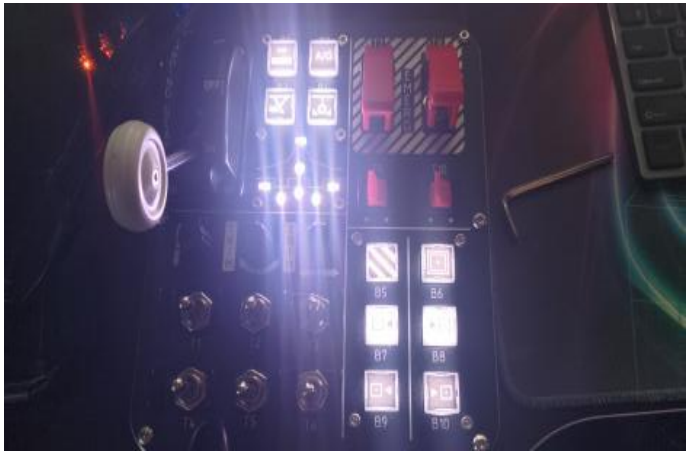


After configuring all steps, we covered earlier, you should see a picture, as shown above.

Press Generate Profile button. Link Tool will restart automatically. After restart you should be able to see 2 USB Devices in the list with 6 and 17 LEDs as we set in profile.

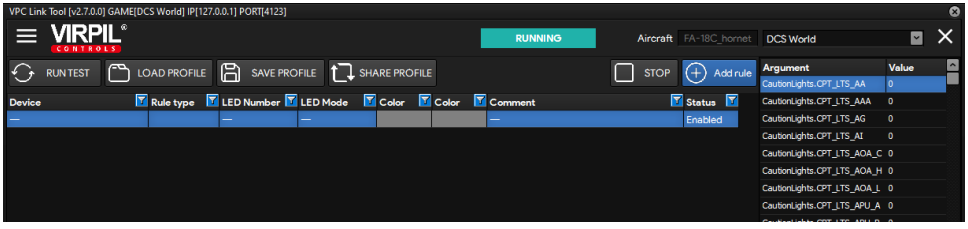


In order to test if everything was done correctly, you can press RESET LEDs button. All LEDs on both Panel and Throttle should become Gray.



How to create LED rules

First, lets create an empty rule entry. Press ADD RULE button, which is located at the top right side of the top row in the main window.

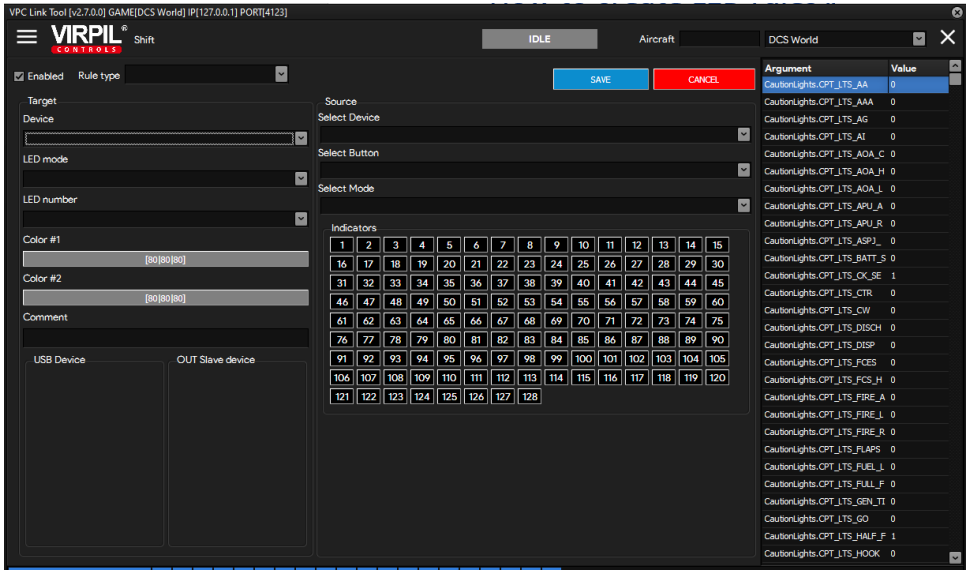


After creating new rule entry, double left click on it will open EDIT RULE interface. Link Too will be stopped. But you still will be able to see arguments list you got in the last calculation cycle.

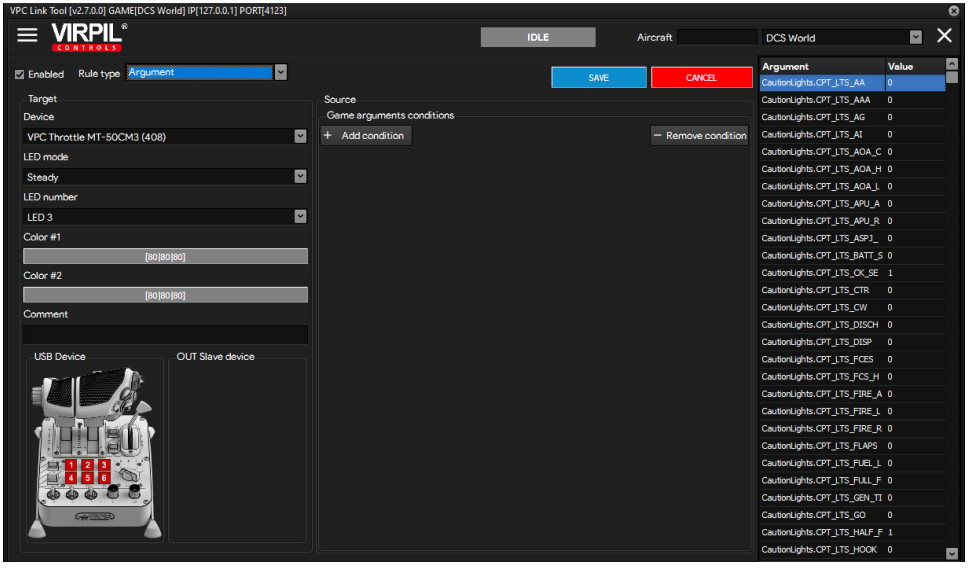
Argument rules

This type of rules is based on simulation data, which was received by LT during its calculation cycles. In our example we will be running DCS World (FA-18C module) in order to demonstrate how rules can be created.

Imagine, we have DCS World mission running along with LT. Arguments list is still visible while editing rule.



1. Select Rule type to match Argument value.
2. Select target Device. Its image will appear in the USB or OUT-Slave sections, depending on device state. In our case, USB Device will be shown. So, you are able to see its picture and LEDs, located on the board.
3. Select MODE. Depending on your needs, select Steady or Flashing mode.
4. Select desired LED Number in the Target Device section.
5. Select Color #1. This color will be used in both Steady and Flashing modes.
6. Select Color #2. This color is used only in case if Flashing mode was selected in step #3. There is no need to select Color #2 option in case If Steady mode was previously selected.
7. If you wish, you can put some comments so you won't forget selected LED functionality or relation to some parts of cockpit or a system.



At this step we configured only Target Device. It means, we programmed a way how LED will react.

Now it's time to add some conditions. This should be done in Source section. Basically, conditions represents a set of IF statements, which are connected with AND statements between each other. For example:

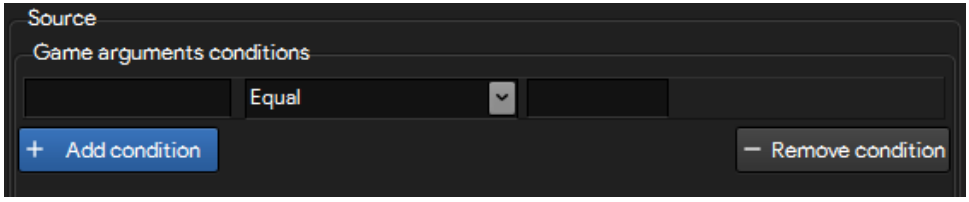
If CONDITION is TRUE

AND If CONDITION is TRUE

AND If CONDITION is TRUE

…And so on.

Add new condition by pressing ADD CONDITION button in the top left side of Source section. New empty row will appear on the screen. You can create up to 13 conditions for a single rule.

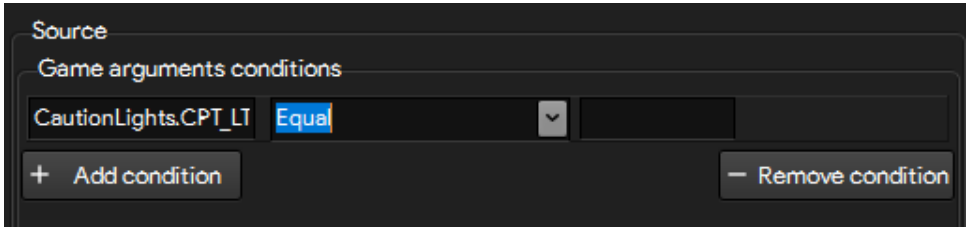


First empty field on the left side of the row represents an Argument name. This is a string value, which is equal to argument name, which you can see on the right side of LT in arguments grid.

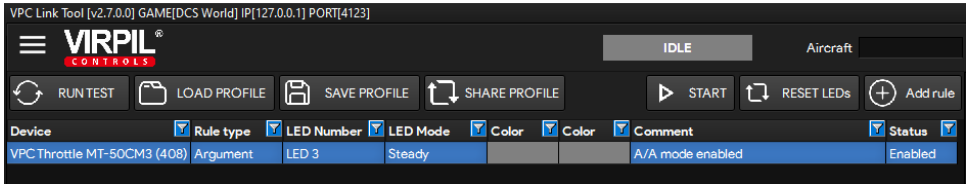
Second, central dropdown field in the row represents a selected condition statement. By default, EQUAL statement is selected.

Third empty field, which is located on the right side of the row represents argument Value. This is a possible number (integer or float) value of a desired argument. In our example we will create a rules, which will control LED 3 on VPC Throttle CM3, depending on A/A mode state of the FA-18C Hornet module. This argument is called *CautionLights.CPT_LTS_AA* . So, lets write down this argument into a first input field. We already have EQUAL statement selected. And third part will be to fill value input field with a possible argument value.

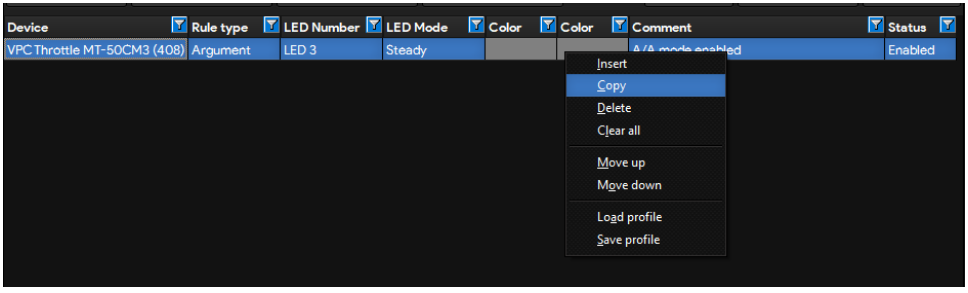
While running DCS World and Link Tool alongside, you can notice that *CautionLights.CPT_LTS_AA* can be 0 or 1. When A/A mode is selected, A/A LED Button will lit in the cockpit. The same behavior we expect from LED 3. So, lets set value 1 into the 3rd input field. At the end, you should see next:

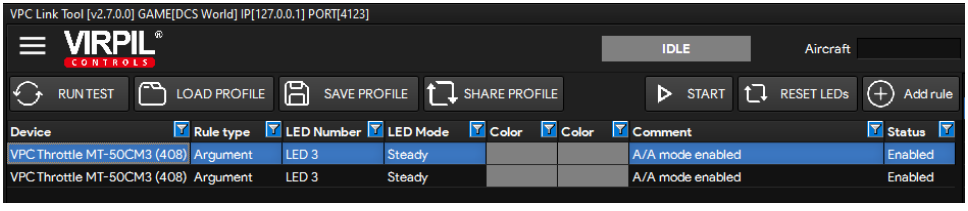


Click SAVE button in order to save new Rule. You will be redirected to rules list.

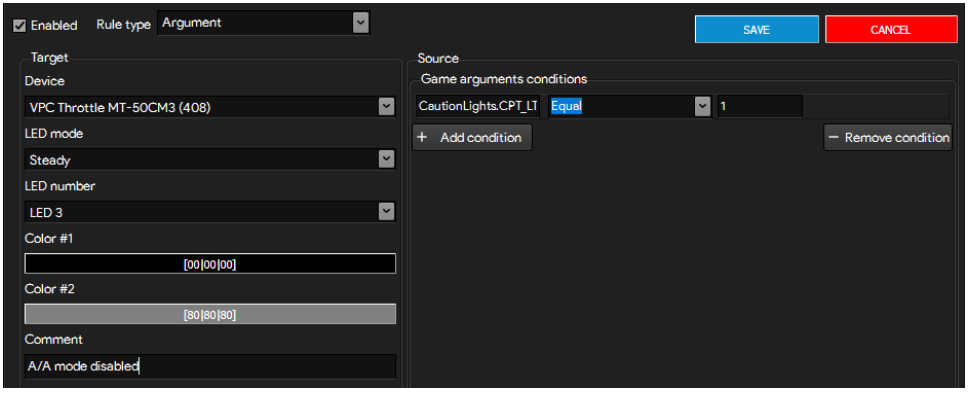


Basically, we create a new rule for ENABLED lamp state. Now, lets create a new rule for DISABLED lamp state. For that, you can COPY your rule and edit new one created. To do that, press right mouse click on your first rule and select Copy option.

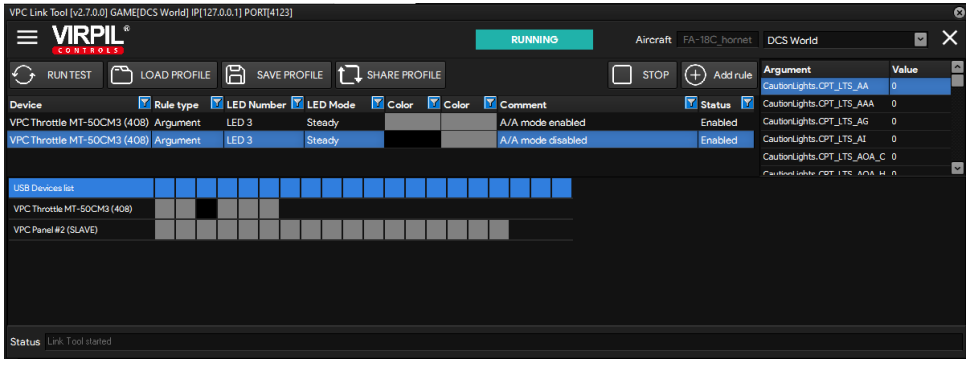




Now, edit second rule. Double left mouse click will bring you to EDIT RULE interface. Change condition value from 1 to 0 and change Color #1 to Black. Hit SAVE button.



Start Link Tool by pressing START button. In order to check LED state – use devices grid, which is located at the bottom of main interface of LT.



As you can see on the picture above, LED 3 on CM3 throttle is Black, because current value of CPT_LTS_AA argument is equal to 0.



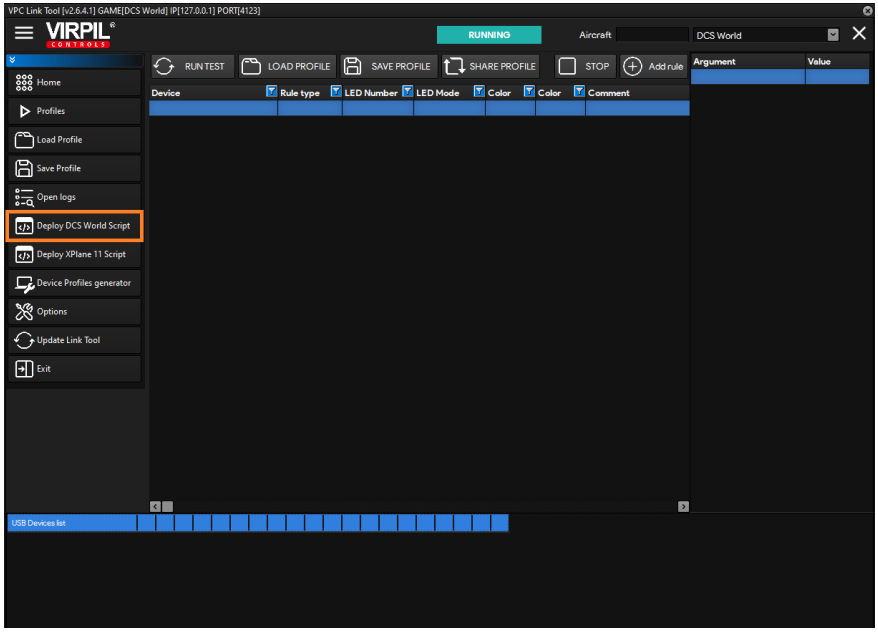
Integrations

In this section we will cover how to get telemetry data from DCS World, X-Plane 11, MSFS 2020, Il-2 and other simulations in Link Tool. For some simulators in order to achieve this task it is required to get integrational scripts from VIRPIL Official Repository, put them in the right folder and run simulation.

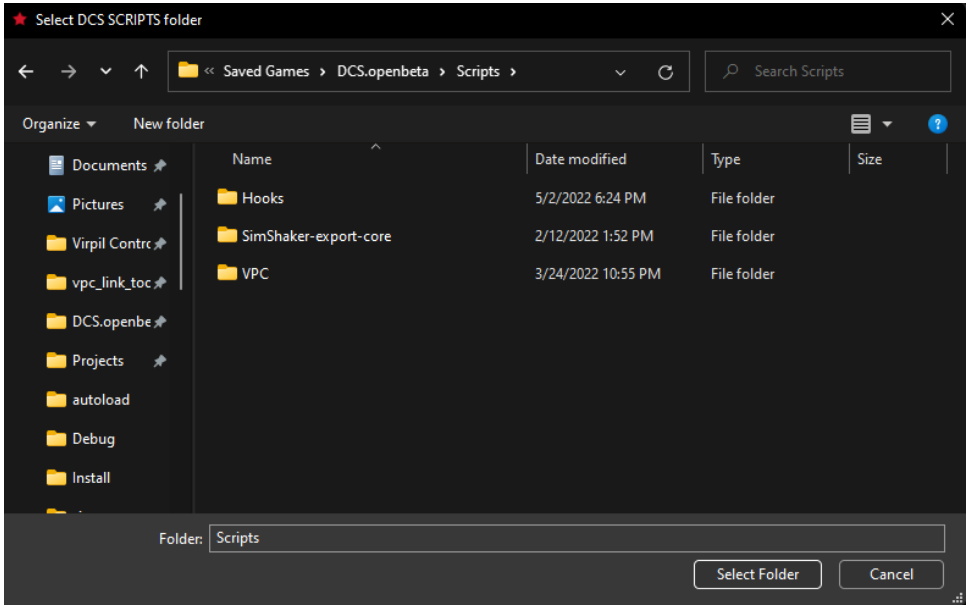
All aircraft telemetry data is exposed from DCS World, Il-2 and X-Plane 11 simulations using UDP protocol. DCS/XPlane integrational scripts create a UDP Client, which is sending user defined data to UDP server, provided by LT. Server address and port can be set in LT Options, in special sections, which we already discussed in the previous topics.

DCS World Integration script

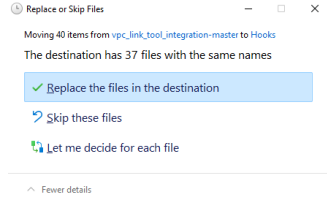
The easiest way to install DCS World script is to use LT functionality. In the left menu you can find button, called Deploy DCS World Integration script.



After pressing this button LT will ask to set a path to special *Scripts* folder, provided by DCS World, which is located in *C:|Users|YOUR_USER|Saved Games* folder. If Scripts folder does not exists, create it first.

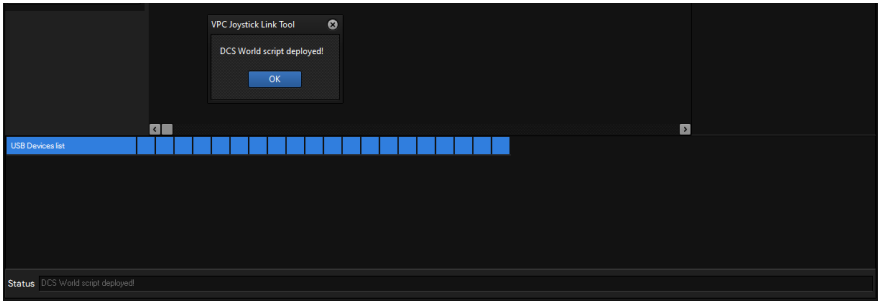


After selecting Scripts folder LT will pull all needed scripts into Hooks folder. If you already have VIRPIL scripts installed, Windows utils will ask you to replace old files with new ones.



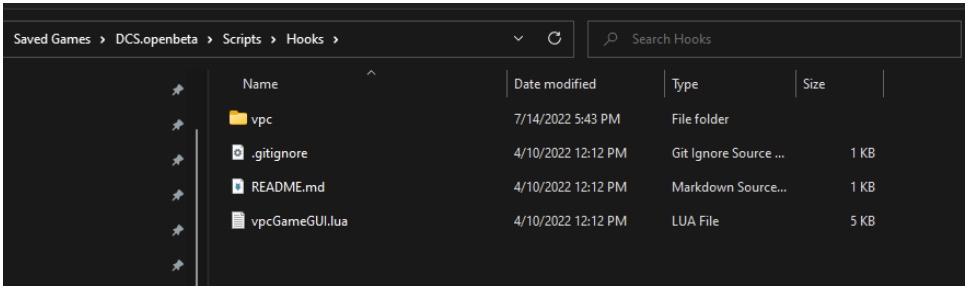
Select *Replace the files in the destination* option.

LT will inform you about success or failed deploy operation by showing modal window and displaying text in LT Status Bar.



Deployment process works in the same way for both DCS World Release and Beta versions. The only difference is that Beta version use files are located in *C:|Users|YOUR_USER|Saved Games|DCS.openbeta* folder, and Release version files are located in *C:|Users|YOUR_USER|Saved Games|DCS|Scripts* folder. In case if DCS World Home folder path (f.e. Saved Games\DCS.openbeta) was changed, please select its destination instead.

After deploying scripts, you can discover files, provided to connect DCS World and LT.

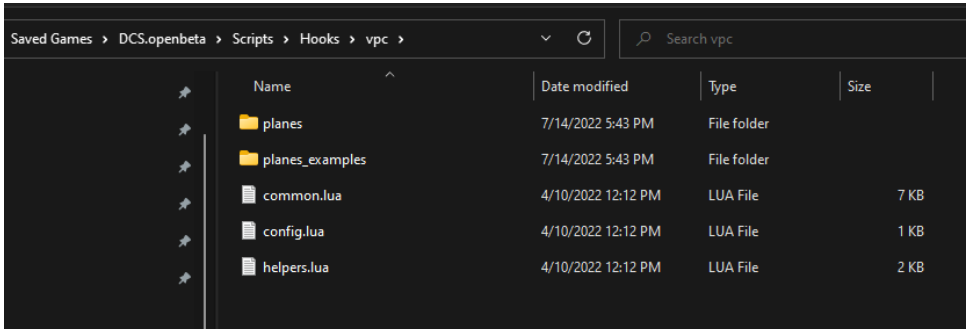


Let's talk about some files. *vpcGameGUI.lua* is an entry point for script execution in DCS Environment. This file is executed once you open DCS World.

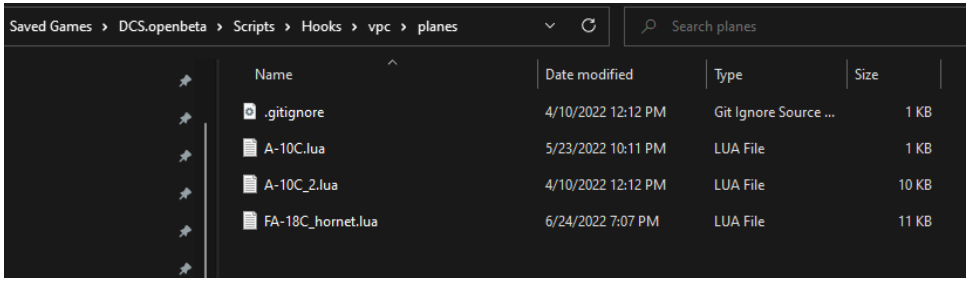
Note! Do not edit this file unless you understand LUA code.

Basically, all data, that is coming from DCS World is separated into 2 parts:

- Common aircraft telemetry. This is a set of data, which is always exposed for every flyable DCS World module in both Single and Multiplayer modes.
- Plane specific telemetry. This is a set of cockpit arguments, which can be exposed from both Single and Multiplayer modes.



First folder, called *planes*, contains a plane specific scripting files. With those files you can expose desired plane-cockpit arguments to Link Tool. Folder *planes_examples* contains ready-to-go scripts examples for some DCS modules and can be copy-pasted into planes folder in order to expose some predefined cockpit arguments.



As you can see on the picture above, there are 3 scripts defined for A-10C, A-10C 2 and FA-18C modules.

NOTE! If plane specific file in this folder is missing, integration script creates a new one when entering aircraft cockpit for the first time, based on boilerplate file *template.lua*, which is located in *Saved Games | DCS.openbeta | Scripts | Hooks | vpc | planes_examples* folder.

template.lua - Notepad

File Edit View

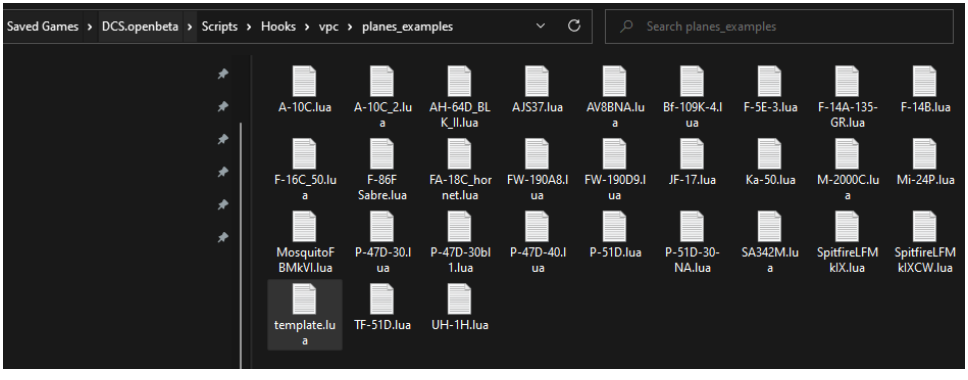
```
return function (vpcPanel)
    local data = {};
    ---------------------------- WRITE YOUR CODE HERE -----------------------------------
    -- ARGUMENT_ID - integer value
    -- ARGUMENT_NAME - string value
    --
    -- if (vpcPanel:get_argument_value(ARGUMENT_ID) == 0) then
    --     data["ARGUMENT_NAME"] = "0"
    -- else
    --     data["ARGUMENT_NAME"] = "1"
    -- end
    --
    --
    -- -- OR
    -- data["ARGUMENT_NAME"] = vpcPanel:get_argument_value(ARGUMENT_ID)
    -- data["ARGUMENT_NAME"] = vpcPanel:get_argument_value(ARGUMENT_ID)
    -- data["ARGUMENT_NAME"] = vpcPanel:get_argument_value(ARGUMENT_ID)
    -------------------------------------------------------------------------------------

    return data; -- Don't forget to return data from function in order to display args inside LINK TOOL
end
```

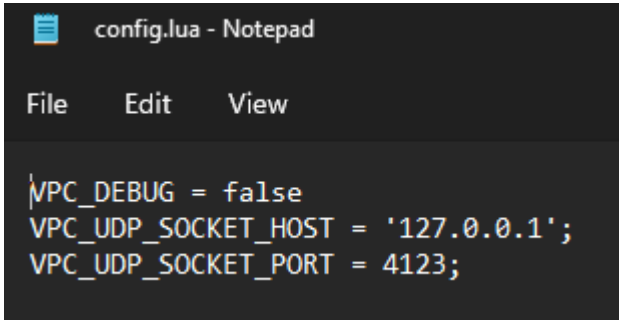
This is how boilerplate looks like.

This is an anonymous function, that takes a *vpcPanel* object as parameter and returns a LUA table. All you need is to know cockpit argument ID, so you can build your own logic and expose any desired data from DCS to LT.

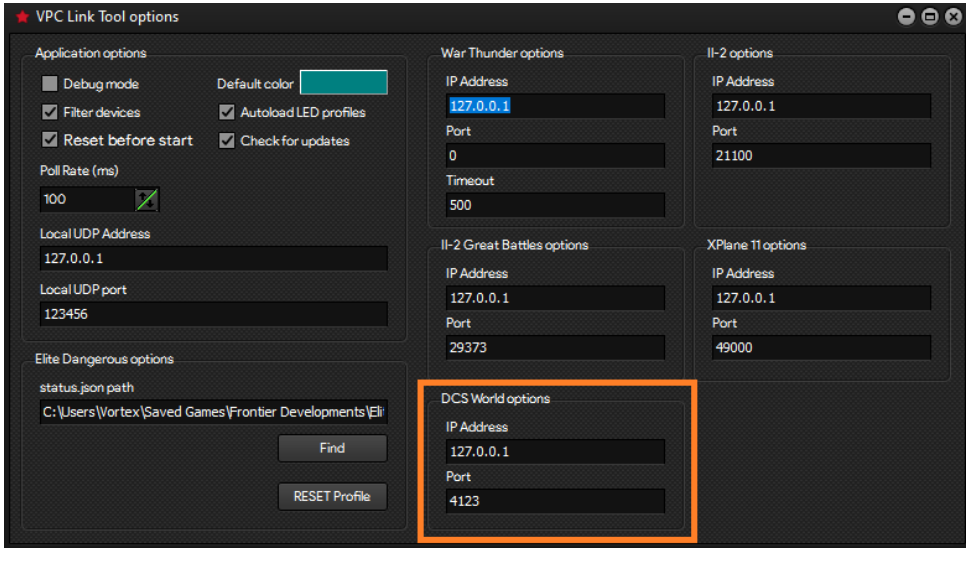
You can find a lot of code examples in *planes_examples* and copy them into *planes* folder. All files are ready-to-go in both Single And Multiplayer modes.



In case, if you do not get any data from DCS World in LT, try to change UDP Server address and port for LUA client. Go to *C:|Users|YOUR_USER|Saved Games|DCS.openbeta|Scripts|Hooks|vpc* folder and open *config.lua* file.



Change VPC_UDP_SOCKET_HOST and PORT options to match the same values as set in LT Options.



During developing a new cockpit arguments code you would probably use VPC_DEBUG option. When this option is set to **TRUE**, it hot reloads all LUA code, created by user inside DCS World environment every time you exit single player mission and start it again. So, you don't need to restart DCS World every time you make a change in scripts.

NOTE! Do not set VPC_DEBUG option to true In daily usage cases. This options conflicts with a lot of community utils, list SRS or TacView.

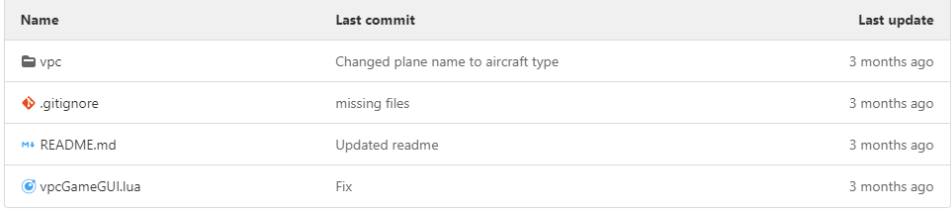
How to connect DCS World and Link Tool

1. Open Left Sidebar menu and click Deploy DCS World Script
2. Select Scripts folder from C\Users\YOUR_USER\Saved Games\DCS location (if folder is missing, create it)
3. Wait until Link Tool will finish deploying files.

There are another two ways you can download and install DCS World Integration script.

1. Download code straight from VIRPIL Official Repository using DOWNLAOD button (archive will be downloaded) and unarchive code files into *C|Users|YOUR_USER|Saved Games|DCS|Scripts* folder.
2. Advanced GIT users can clone code repository using GIT Software (Git SCM GUI, Github Desktop, etc…) straight into *C|Users|YOUR_USER|Saved Games|DCS|Scripts* folder and make pull from time to time in order to get latest script updates. This can be done by running a set of commands in CMD terminal:
    - mkdir C:\Users\YOUR_USER\Saved Games\DCS.openbeta\Scripts
    - cd C:\Users\YOUR_USER\Saved Games\DCS.openbeta\Scripts
    - git clone git@git.dev.virpil.com:dcs-world/vpc_link_tool_integration.git .\

Scripts folder should have next files structure:



Name	Last commit	Last update
vpc	Changed plane name to aircraft type	3 months ago
.gitignore	missing files	3 months ago
README.md	Updated readme	3 months ago
vpcGameGUI.lua	Fix	3 months ago

In any other cases, integration script won't be loaded into DCS World environment, code won't be executed and LT won't receive any telemetry data.

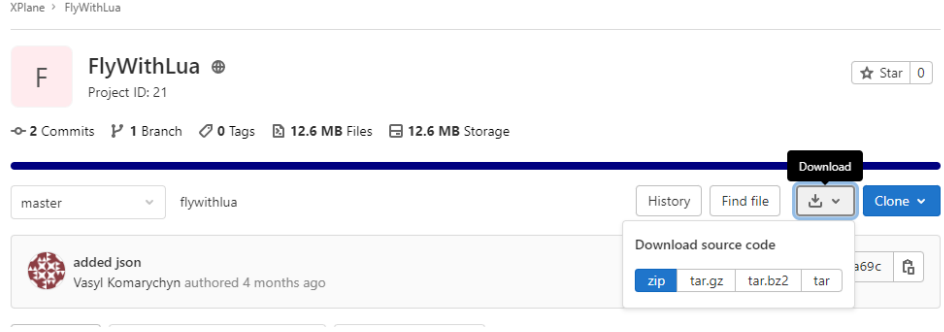
In case if LT doesn't receive any telemetry data, try to change PORT number in config.lua and LT Options and restart DC World. If LT still does not receive any data after pressing START button, please contact our VIRPIL SUPPORT Team.

X-Plane 11 Integration script

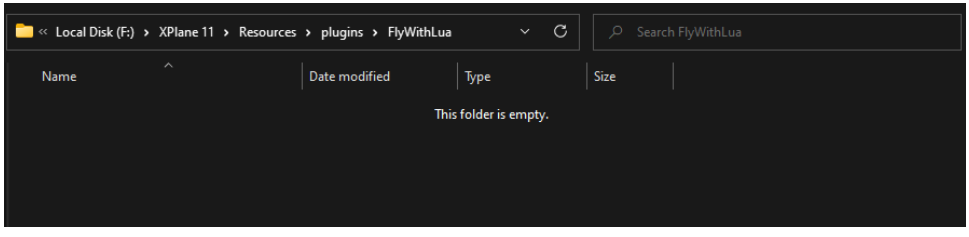
In this section we will cover how to transfer telemetry data from X-Plane 11 to LT. In order to do it using LUA Scripting language, please, download or clone FlyWithLua library from [VIRPIL Official Repository].

In case of downloading:

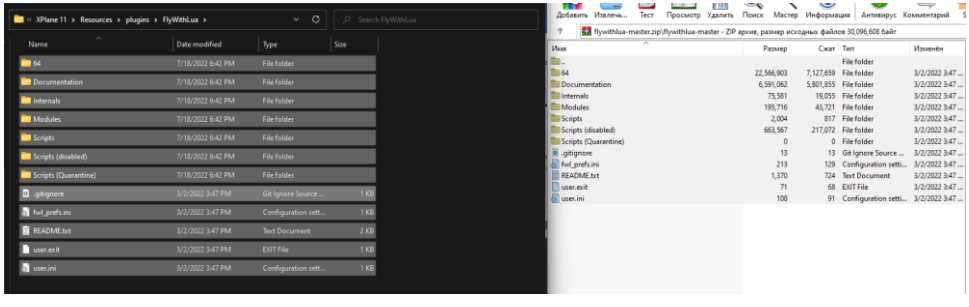
go to https://git.dev.virpil.com/xplane/flywithlua and hit Download button, select ZIP type and wait until archive file will be downloaded.



Go to X-Plane 11 location and create a folder FlyWithLua in XPlane 11\Resources\plugins folder as shown on the picture under.



Unzip all files from downloaded archive (flywithlua-master.zip) into FlyWithLua folder.



In case of cloning project using GIT, open CMD (Command Prompt) and run a set of commands:

```
C:\Users\Vortex>F:

F:\>cd "XPlane 11"

F:\XPlane 11>cd Resources

F:\XPlane 11\Resources>cd plugins

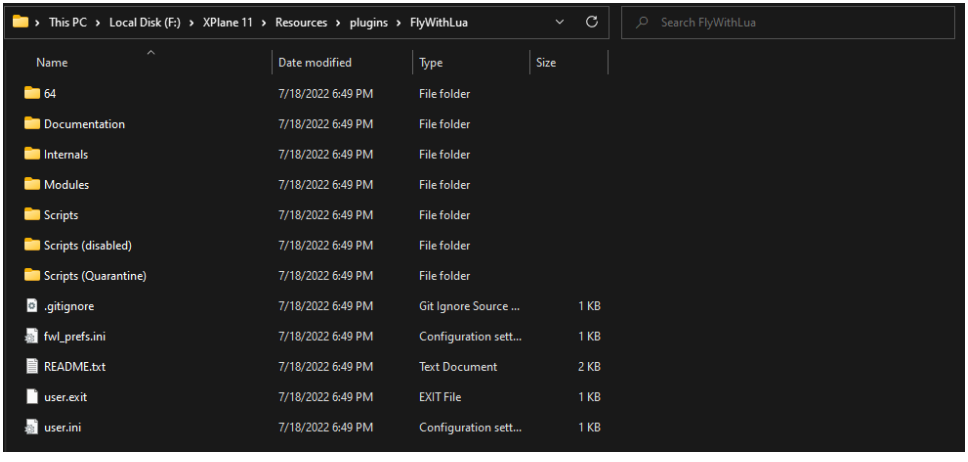
F:\XPlane 11\Resources\plugins>
F:\XPlane 11\Resources\plugins>mkdir FlyWithLua

F:\XPlane 11\Resources\plugins>cd FlyWithLua

F:\XPlane 11\Resources\plugins\FlyWithLua>git clone git@git.dev.virpil.com:xplane/flywithlua.git .\
Cloning into '.'...
remote: Enumerating objects: 188, done.
remote: Total 188 (delta 0), reused 0 (delta 0), pack-reused 188Receiving objects:  89% (168/188), 11.54 MiB | 1.98 MiB/
Receiving objects:  95% (179/188), 11.54 MiB | 1.98 MiB/s
Receiving objects: 100% (188/188), 12.46 MiB | 1.64 MiB/s, done.
Resolving deltas: 100% (13/13), done.

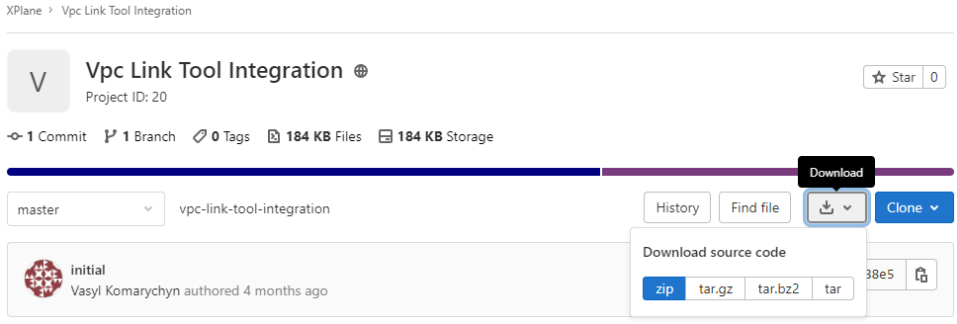
F:\XPlane 11\Resources\plugins\FlyWithLua>
```

Basically, you need move to correct folder, create FlyWithLua folding using MKDIR command and clone project into this folder. At the end, you should see the same folders/files structure:

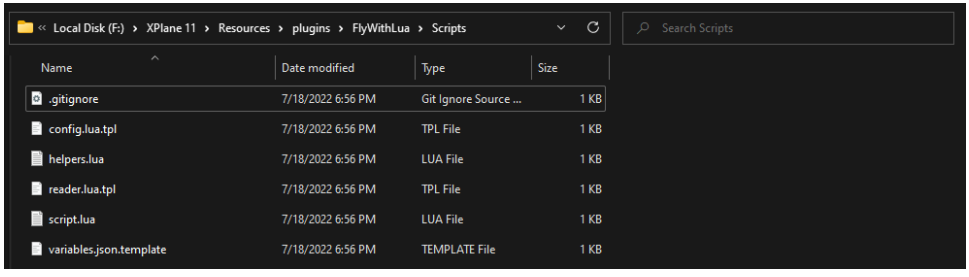


At this point, FlyWithLua library is installed and ready to go. Now, we need to install basic integration script for XPlane 11. This can be done with both download or clone ways from VIRPIL Repository.

Go to https://git.dev.virpil.com/xplane/vpc-link-tool-integration and download ZIP archive of the source code.



Unzip files into XPlane 11\Resources\plugins\FlyWithLua\Scripts so it should looks like on the picture under:



Copy all contents from variables.json.template into variables.json file and your are ready to do some scripting.

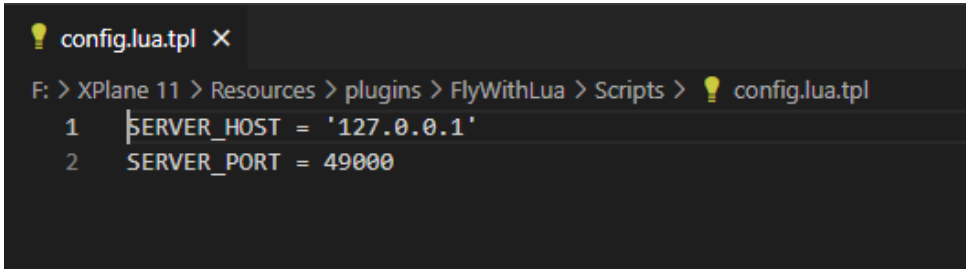
For any further coding we will be using XPlane Datarefs website. It contains a description of all arguments, that are available in XPlane simulator. Go to https://developer.x-plane.com/datarefs/.

As you may see, there is a big list of available categories. But we will discuss a few vars.

Datarefs for X-Plane 1150, compiled Thu May 7 20:05:46 2020

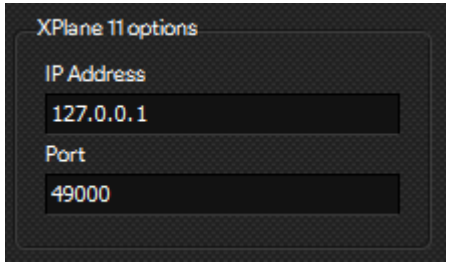
Name	Type	Version	Writable	Units	Description
+ sim/					
+ sim/aircraft/					
+ sim/airfoils/					
+ sim/atc/					
+ sim/cockpit/					
+ sim/flightmodel/					
+ sim/graphics/					
+ sim/joystick/					
+ sim/multiplayer/					
+ sim/network/					
+ sim/physics/					
+ sim/operation/					
+ sim/test/					
+ sim/time/					
+ sim/version/					
+ sim/weapons/					
+ sim/weather/					
+ sim/world/					
+ sim/aircraft2/					
+ sim/cockpit2/					
+ sim/flightmodel2/					
+ sim/graphics/VR/					

At this point lets configure UDP Server data. Open config.lua.tpl file from *XPlane 11|Resources|plugins|FlyWithLua|Scripts* location.

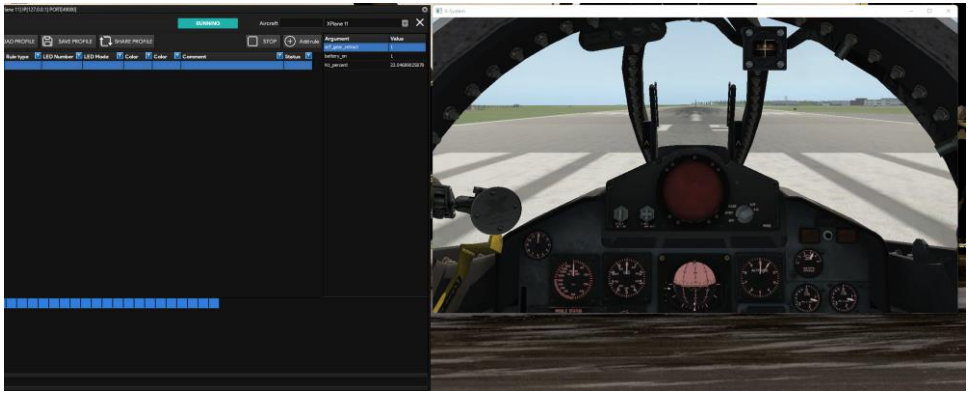


In this file you can find UDP host and port number, to which XPlane will transfer arguments, defined in *variables.json* file. In case, if this file is missing, please, create a copy from *variables.json.template* file.

Configure LT options and set UDP server address and port for XPlane 11 to be the same as set in *config.lua.tpl* file.

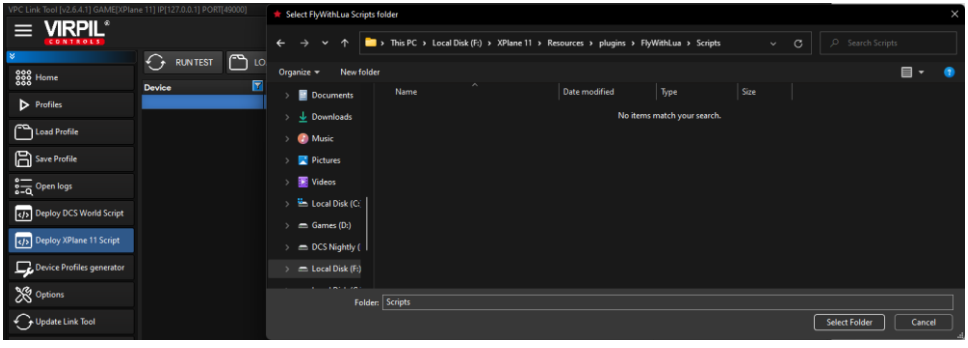


Now, run Link Tool with selected XPlane 11 simulation and run XPlane itself. If you did all steps without any errors, you should see 3 arguments in LT.

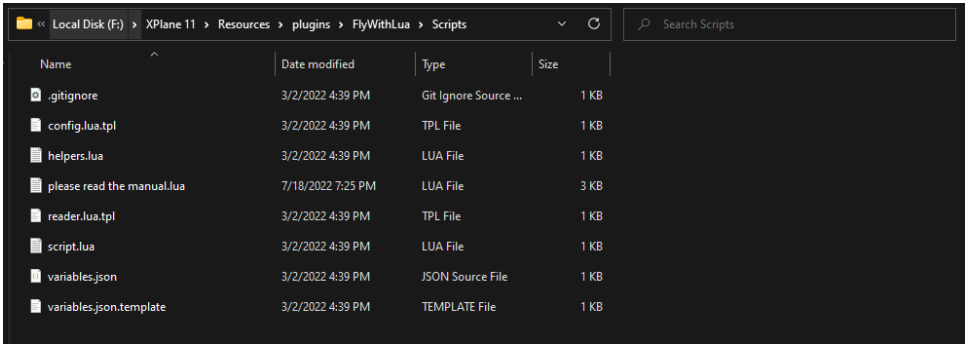


Integration script can be deployed in other way. But you still need FlyWithLua to be installed first.

So, to download integration script, open LT, in the left menu click "Deploy XPlane 11 Script" button, select Scripts folder, which is located in *XPlane 11\Resources\plugins\FlyWithLua* and press Select folder button.



When files will be downloaded, LT will inform you with message : "XPlane 11 script deployed". So, now you can open scripts location and check if files structure is correct.



All files are in place, so you are able to do some magic. Lets see how to add new arguments to LT.

Open file *variables.json* file. This is a place, where all variables are read while XPlane 11 is running.

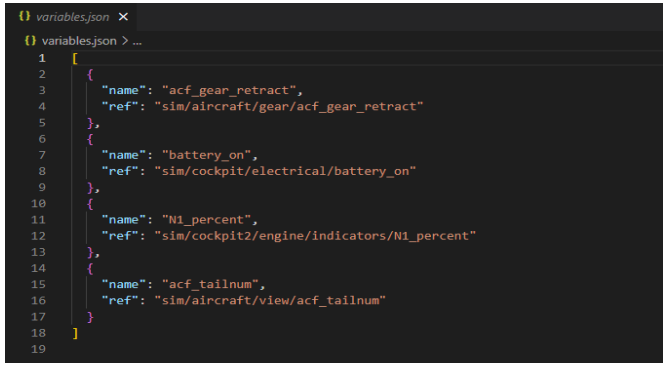


As you can see, on the picture above, content of this file is a JSON Array of objects. Every single object of this array contains 2 keys: name and ref. Key name represents a name of argument from XPlane Dataref table. For example, acf_tailnum.

acf_tailnum	byte[40]	660+	yes	string	Tail number

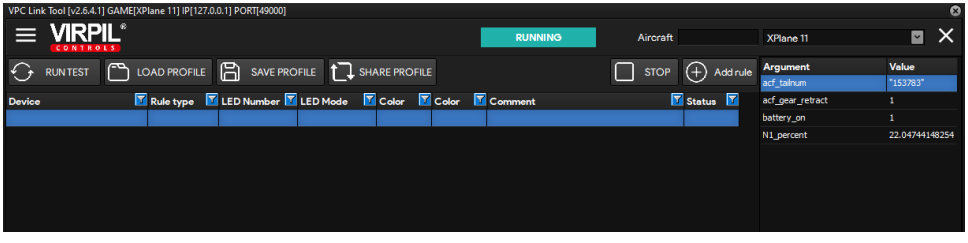
Key ref represents a full path to argument, including category name. In our case, *acf_tailnum* argument is located in *sim/aircraft/view/* category. So, ref value should look like *sim/aircraft/view/acf_tailnum*.

After you found desired argument in Dataref's table, you should define new JSON object in variables.json file. This object in case with acf_tailnum argument should look like :



After making changes in variables file, you can reload all LUA scripts, using FlyWithLua functionality in XPlane 11 in order to see new arguments in LT.





As you can see on the picture above - there is a new argument (acf_tailnum) with value 153783.

You probably, may notice some useless messages on the screen. These messages are generated by manual.lua file, which is located in Scripts folder of FlyWithLua plugin and reload all LUA scripts.

Il-2 Sturmovik simulations family integration

In this section we will discuss LT integration process with a different Il-2 Surmovik versions.

The only exception for LT integration is an Il-2 Sturmovik: Cliffs of Dover simulator. It is ready-to-go for telemetry exchange with LT and does not requires any configuration actions.

Il-2 Sturmovik: Great Battles

In order to connect Great Battles and LT, it is required to change game configuration file called startup.cfg. It can be found in data folder (f.e. C:/Il-2 Great Battles/data/startup.cfg). Open this file using Notepad++ or similar editor.

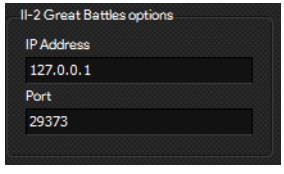
Scroll down and find *[KEY = telemetrydevice]* row. This is a telemetry device section which is used to edit UDP server options, which are used to expose telemetry data from simulator. If this section is missing in the startup.cfg file, just add it manually:

```
[KEY = telemetrydevice]
    addr = "127.0.0.1"
    decimation = 2
    enable = true
    port = 29373
[END]
```

Note. Keep code formatting, so it will be easier to read current configuration file.

Now you can save file and close it. Run simulation. If Il-2 was up and running – restart it in order start/reload telemetry UDP Server, provided by simulation.

After editing startup.cfg file – additional check/configuration is required for LT. Open Link Tool and navigate to options UI using left side menu or using **LCTRL + O** keyboard shortcut. Check or Edit Il-2 Great Battles section in order to match settings, provided for telemetry device in startup.cfg file.



Field `addr` is related to IP Address input field and stands for UDP Server host address. This field always should be as localhost (127.0.0.1).

Field port is related to Port input field. This option can be changed to any available/desired value. But it always should match a value, provided in startup.cfg file. This option stands for UDP Server port, on which Il-2 is broadcasting plane telemetry data.

Possible issues

Sometimes you might face issue, like "Link Tool does not show arguments list". First place you should look at – is a port configuration in both LT Options UI and startup.cfg file.

Another possible issue could be a mess in arguments grid (empty rows or empty values or so). This issue could be related to startup.cfg file. It can contain other configurations for different devices you use in your daily flights. For example, motiondevice configuration section, which is used for SimShaker products like JedPads or similar:

```
[KEY = motiondevice]
   addr = "127.0.0.1"
   decimation = 2
   enable = true
   port = 29373
[END]
```

In this case you may notice, that port 29373 is used for both motion and telemetry device sections. This configuration makes a collision of both data structures (telemetry and motion) when LT reads incoming packets from Il2.

Solution #1: change port number for telemetrydevice in both Il-2 and Link Tool options.

Solution #2: change motiondevice port in startup.cfg and reconfigure SimShaker or other 3rd Party's software you are using along LT in order to use different UDP port.

```
[KEY = motiondevice]
   addr = "127.0.0.1"
   decimation = 2
   enable = true
   port = 29374
[END]
[KEY = telemetrydevice]
   addr = "127.0.0.1"
   decimation = 2
   enable = true
   port = 29373
[END]
```

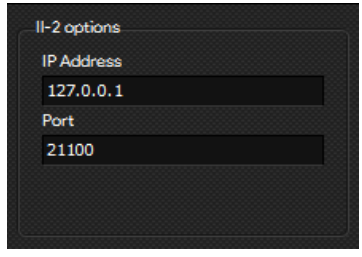
IL-2 Sturmovik 1946 FB

Integration with 1946 FB is similar to the Great Battles.

First, find and open conf.ini file in 1946 FB simulation folder. Scroll down and search for DeviceLink configuration section and edit it. If this section is missing, then add it as follows:

[DeviceLink]
port=21100
IPS=127.0.0.1
host=127.0.0.1

Start or restart simulation. After adding or editing DeviceLink configuration LT will be able to read telemetry data on 21100 port. Check LT options and make sure Il-2 options section is matching DeviceLink configuration.



VIRPIL Devices LED Addresses

