FLIGHT SIMULATION ASSOCIATION

KCVG Chapter

Switch It Up!

Cockpit Controls with Air Manager

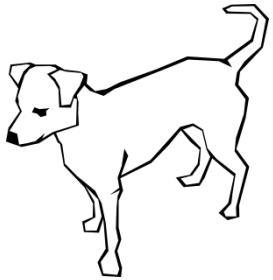
Jan 16 2024 Meeting

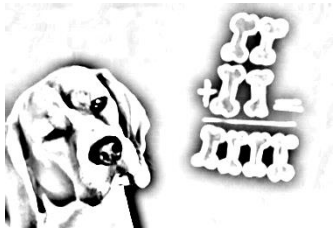
Mike Fullington

Cockpit Building…

A Simple Equation... using a few inexpensive items

Sim Innovations

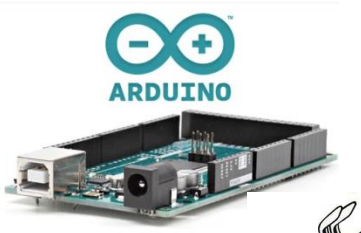




Outline

- Creating a hardware plugin
  - Download or roll your own
- The script / code (LUA)
  - HW_xxx_ADD functions
  - The Function function
  - The “talking” functions – xpl_command, fs2020_ variable_write…
- Adding the hardware - Flashing arduino / raspberry pi – id’ing port
- Navigating your function – info and script buttons
- HW Function Details…
- Getting commands and datarefs
- Creating a panel and adding our hardware function / plugin

What we have to do…

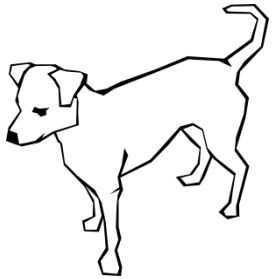


We have to give our board a name and teach it how to speak Air Manager

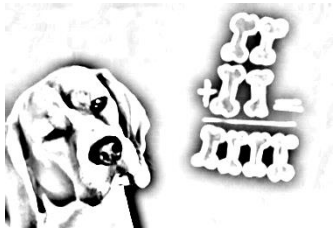
Flash the hardware and add a device

Turns electrical activity into computer instructions

Sim Innovations



We have to tell AM who to listen to. Listening for a change.



We have to tell AM how to make a decision

We have to tell AM what to tell XP or MSFS

The Air Manager plug-in listens and translates into Xplane or MSFS

The Logic…

logic.lua

```lua
function battery_switch_callback(position)

    if position == 0 then
        xpl_command("sim/electrical/battery_1_off")
        fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
        fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
    else
        xpl_command("sim/electrical/battery_1_on")
        fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
        fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
    end

end

hw_switch_add("Battery switch", 1, battery_switch_callback)
```

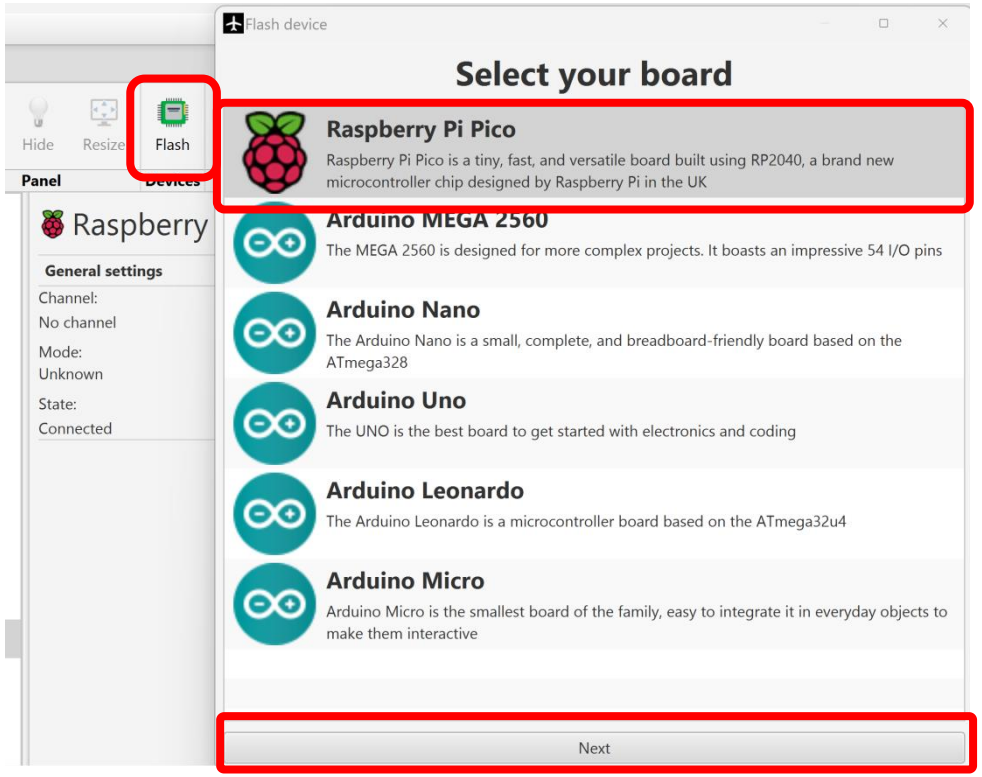
Option 1 - Map hardware later…

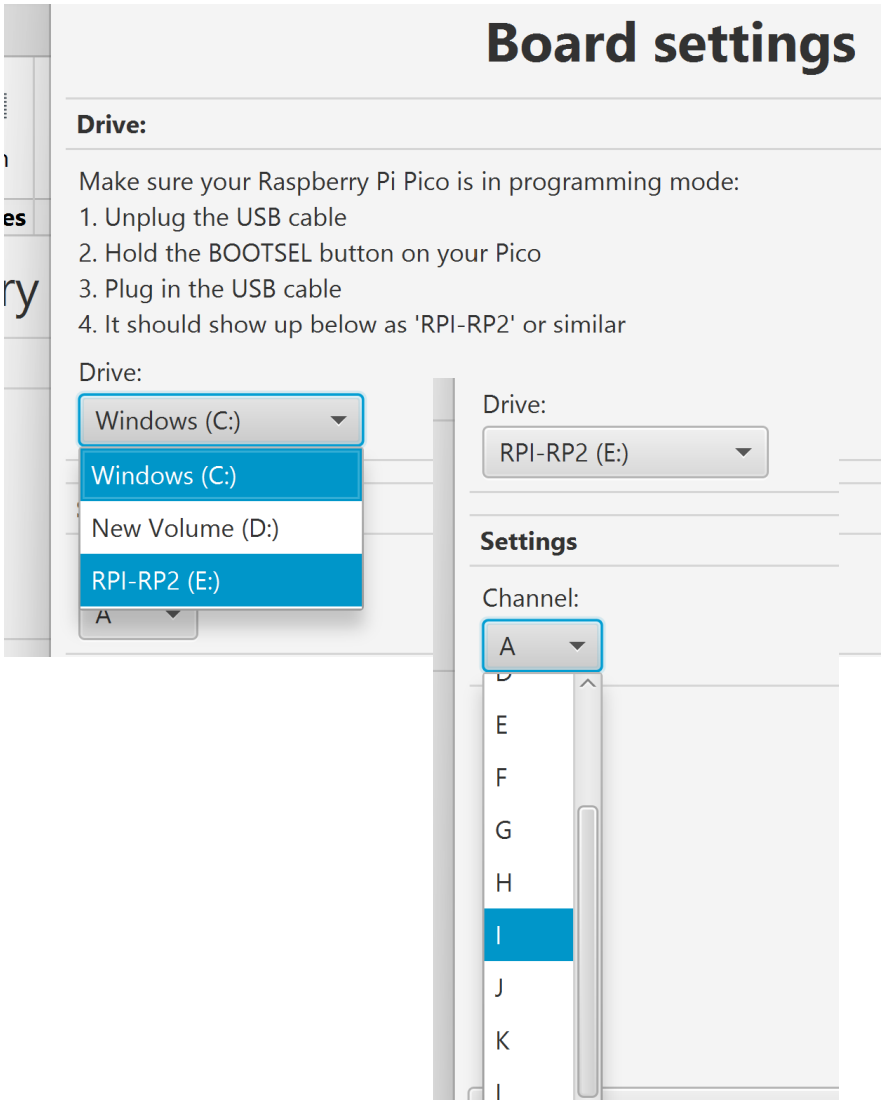
Option 2 - Map hardware now…

```lua
hw_switch_add("RPI_PICO_I_GP17", battery_switch_callback)
```

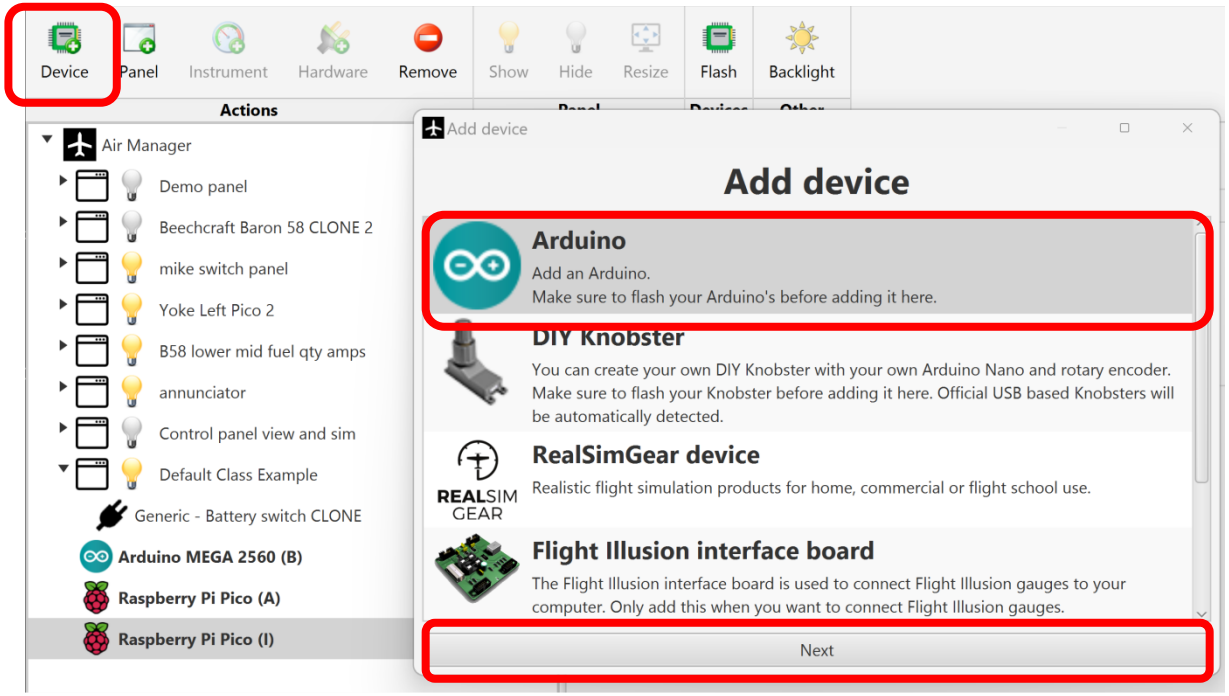
Can have as many of these as you wish in a single LUA file…
Just group the hw_xxxx_add at the end, with multiple functions above

We have the logic, now we need to add the actual hardware (Arduino, Raspberry Pi, etc)
First, name it - flash the board so AM can see it…

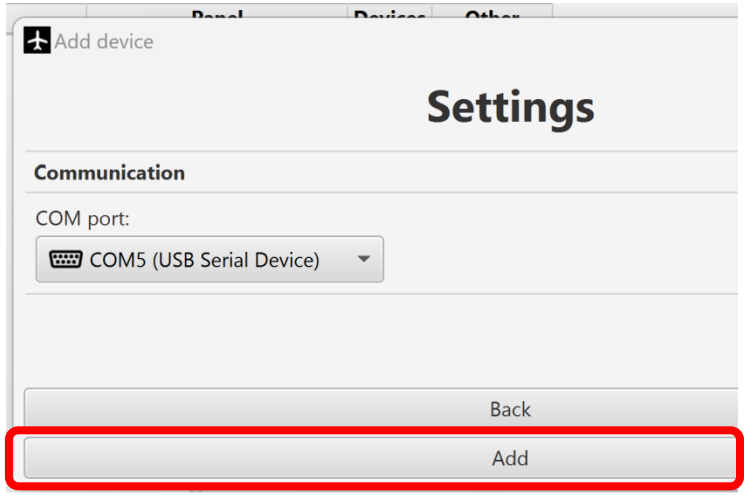




For Arduino’s there’s one more step…

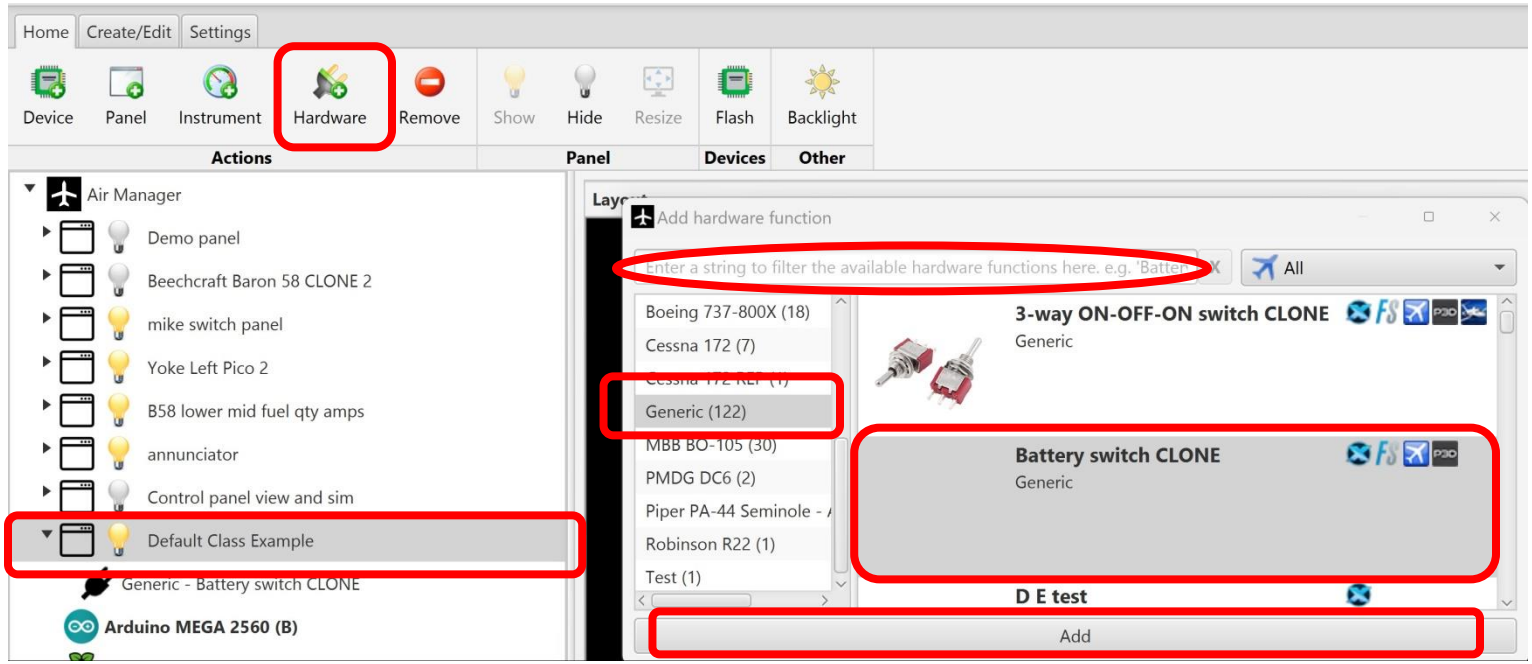


And you have to determine which com port it is on…

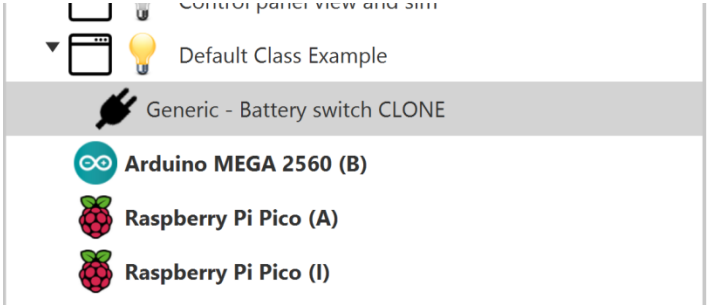


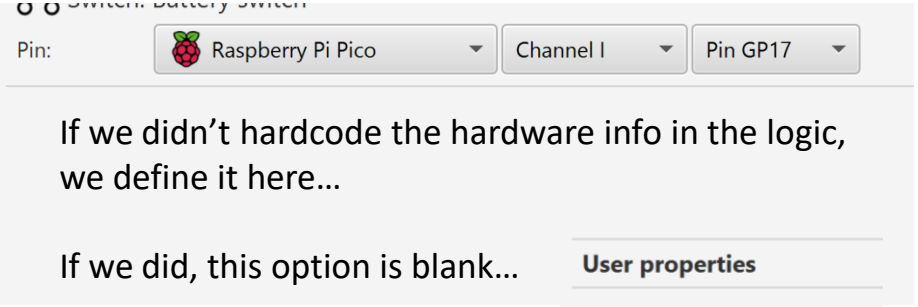
Create a panel and add our hardware function to it (or add it to an existing panel)…

Panel creation – +Panel button, not shown here – panels don't have to have instruments…



Select your panel, click "Hardware", search or scroll the list of available hardware, select it and click "Add"

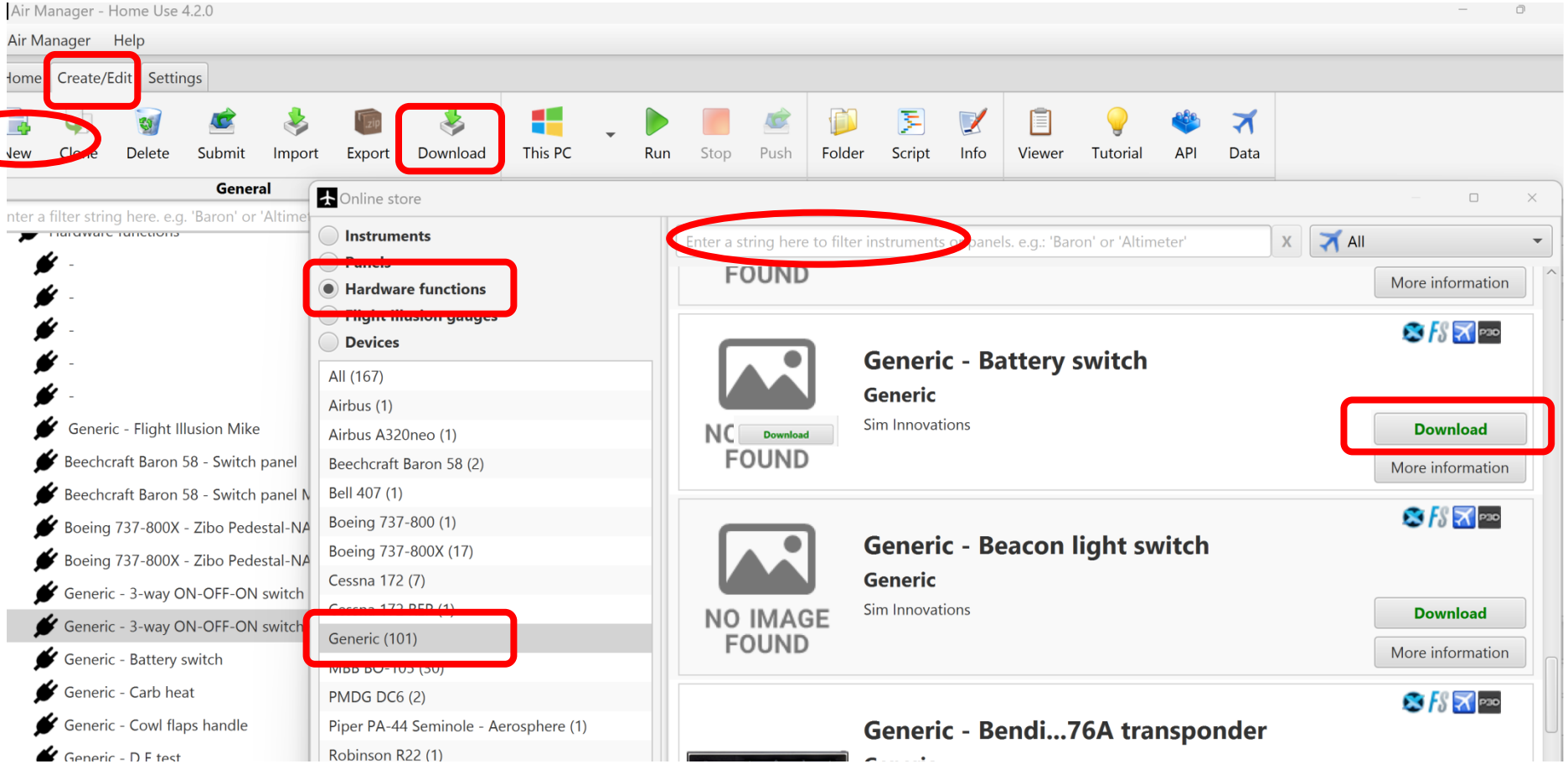




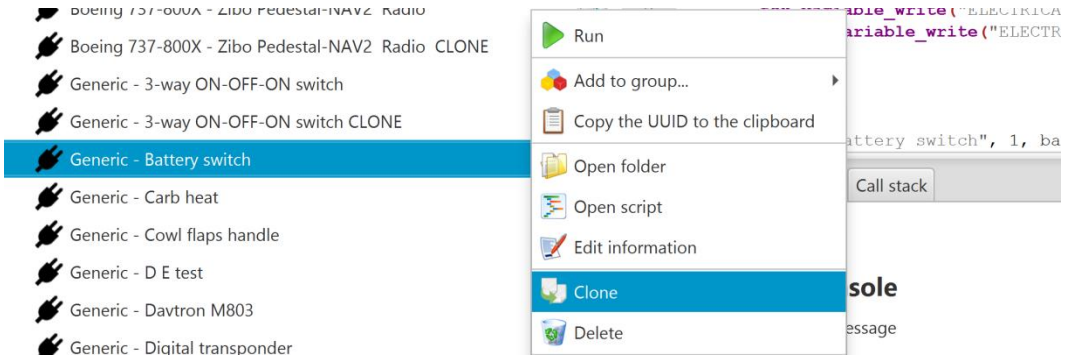
If we didn't hardcode the hardware info in the logic, we define it here…

If we did, this option is blank…

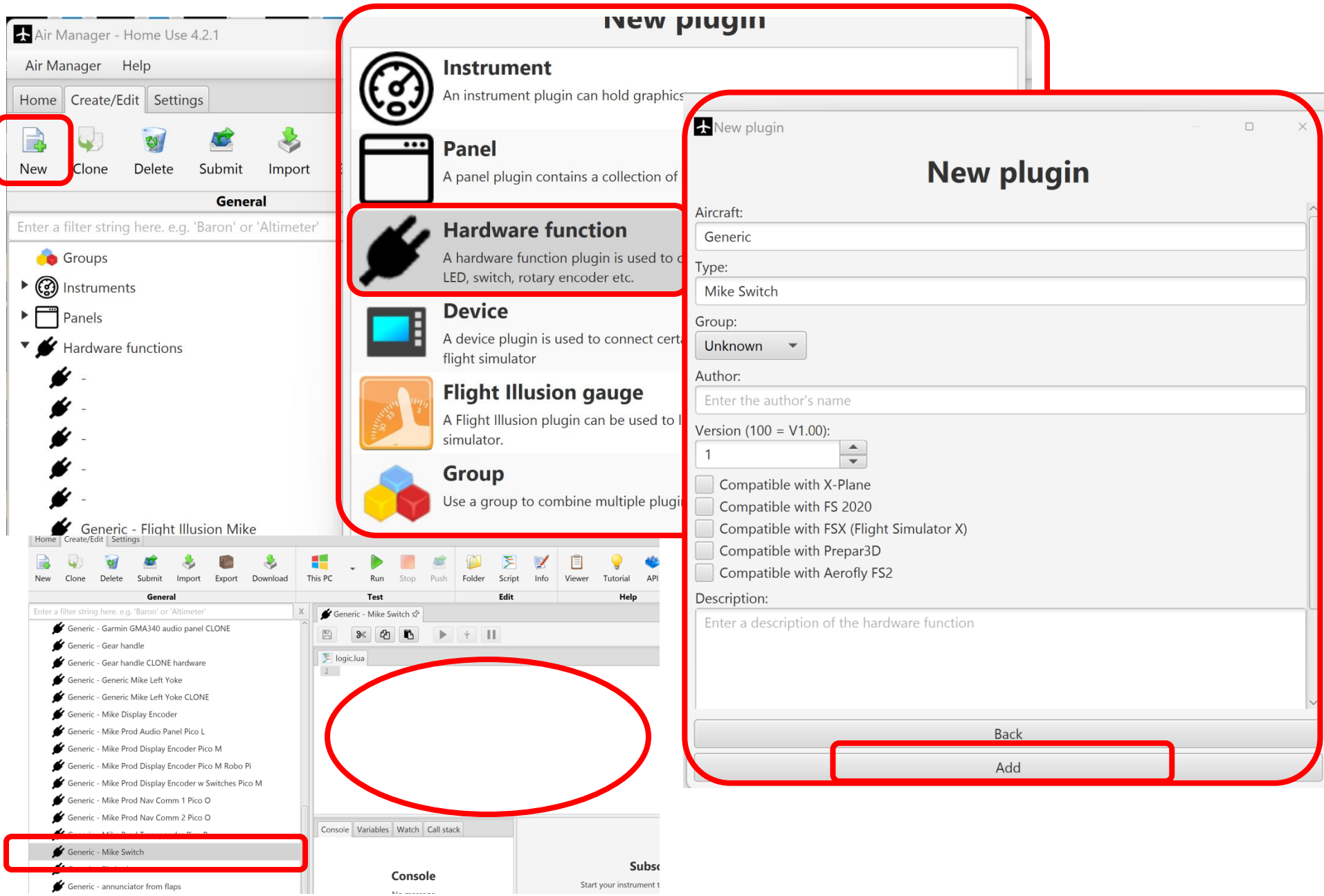
Add a new hardware function by downloading… Take advantage of what SI has already!



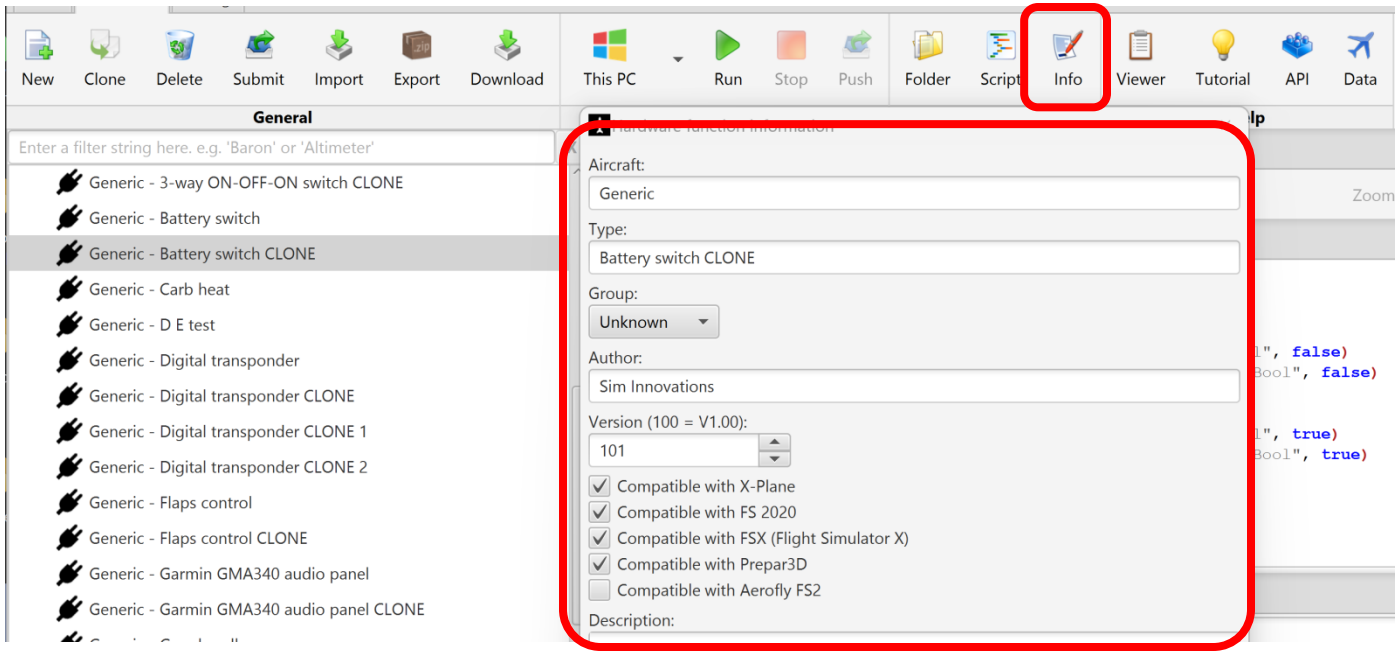
Then clone it by right clicking it…

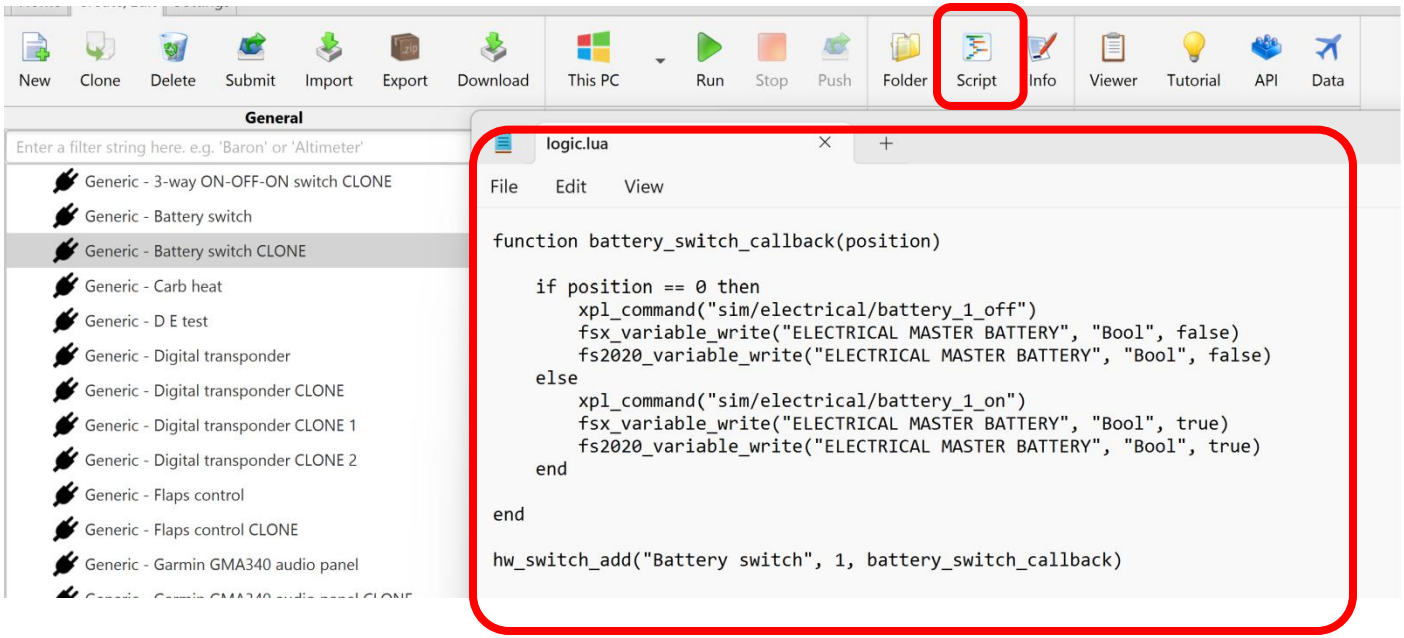


Add a new hardware function – roll your own…



Navigating your function - info and script buttons…





HW Function Details…

```
16  hw_switch_add("RPI_PICO_I_GP17", battery_switch_callback)
```

siminnovations.com/wiki/index.php?title=Hw_switch_add

Description

```
hw_switch_id = hw_switch_add(name, nr_pins, callback) (from AM/AP 3.5)
hw_switch_id = hw_switch_add(hw_id_0, hw_id_1, hw_pos_n,callback)
```

switch_add is used to add a hardware switch. Every position on the switch should use one hardware input.

Named

Available from AM/AP 3.5.

Give your hardware objects a name (.e.g. 'Power button' or 'Strobe LED').
Air Manager will present the user with a view where the assignment of pins can be done.

Arguments

```
hw_switch_id = hw_switch_add(name, nr_pins, callback)
```

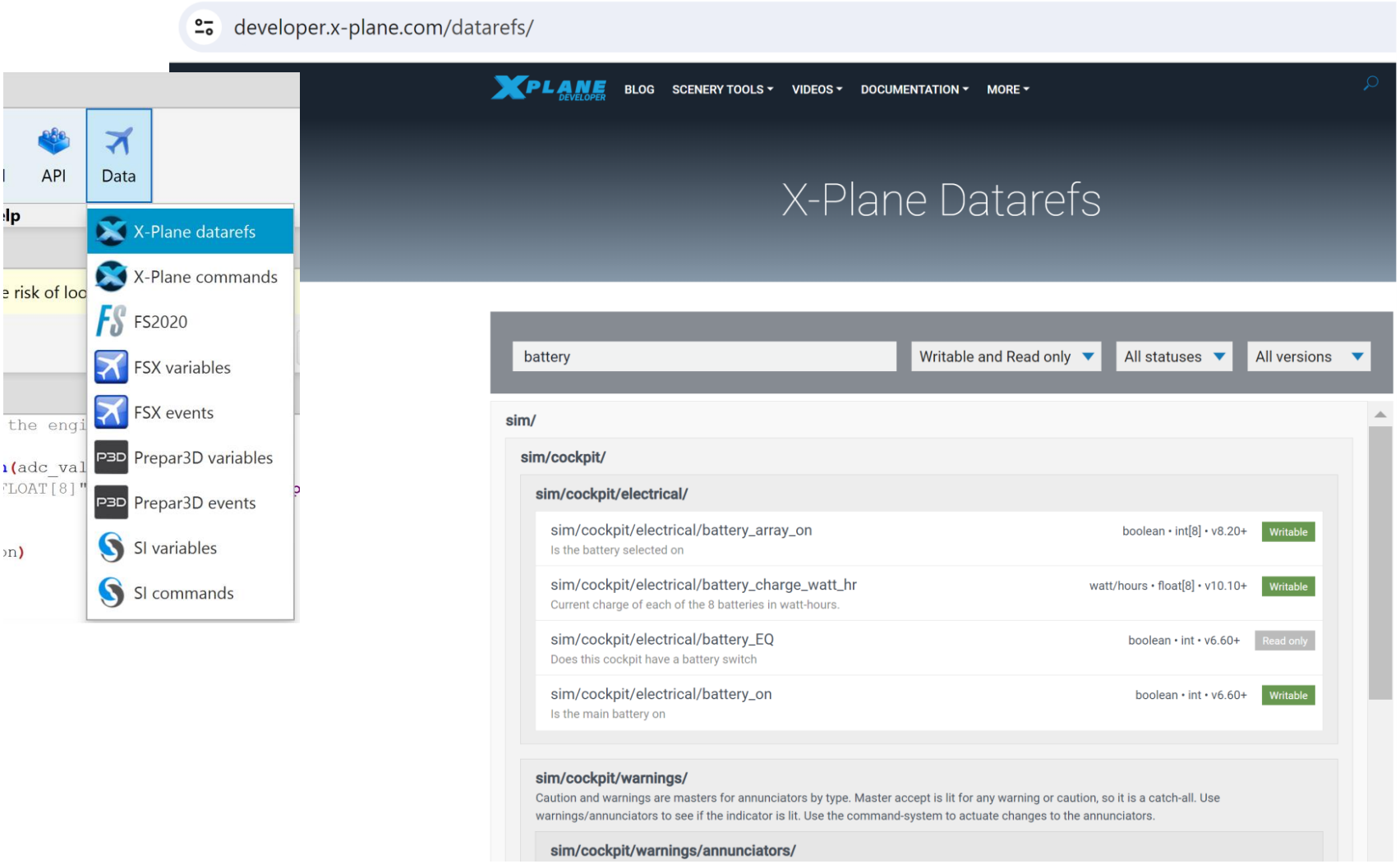
#	Argument	Type	Description
1	**name**	*String*	A functional name to define the switch.
2	**nr_pins**	*Number*	Number of input pins used for this switch.
3	**callback**	*Function*	This function will be called when the switch changed position.

Return value

Argument	Type	Description
hw_switch_id	*ID*	This value can be used for further reference.

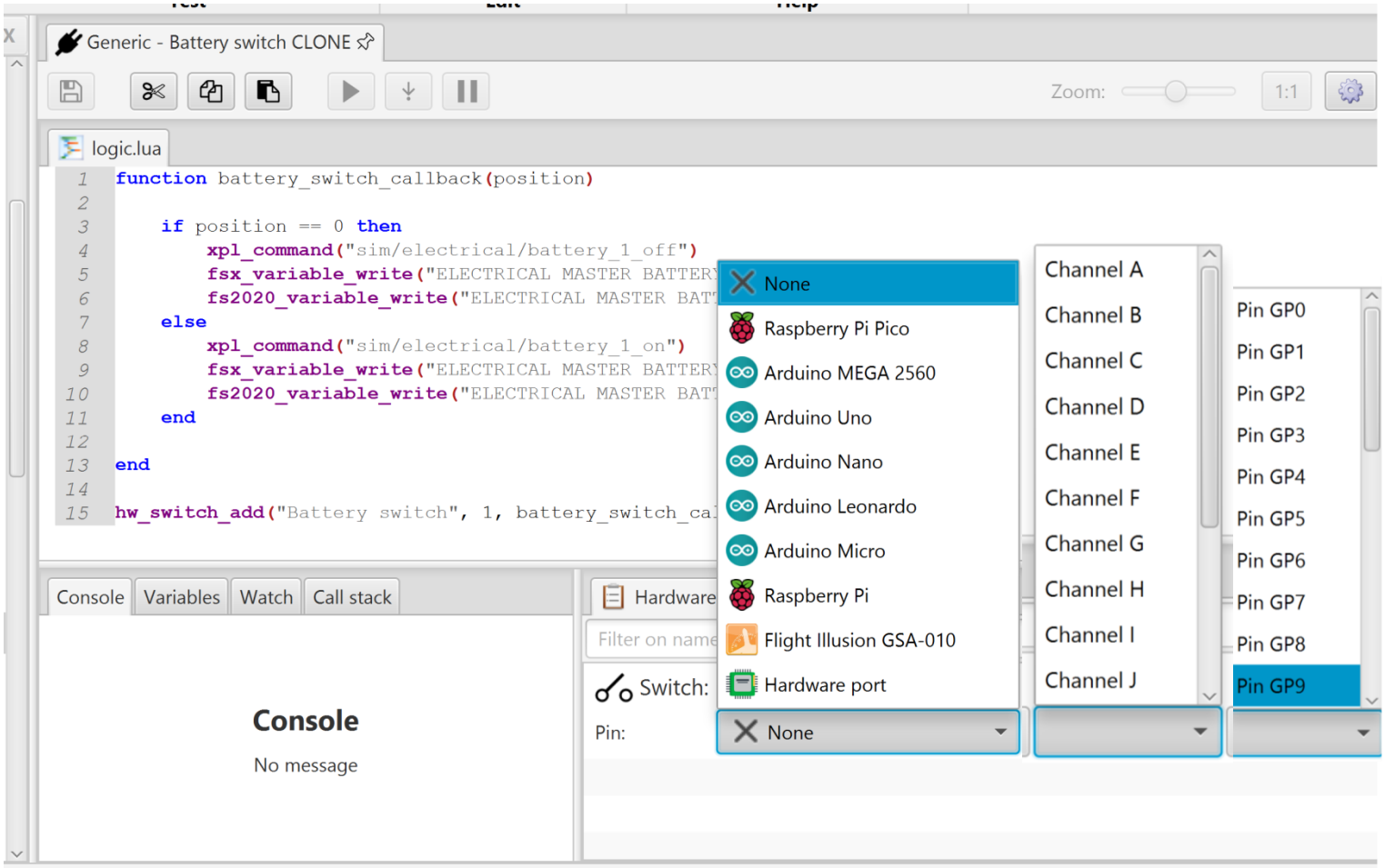
- Hw led set
- Hw led add
- Hw output add
- Hw output set
- Hw dial add
- Hw button add
- Hw connected
- Hw switch add
- Hw output pwm add
- Hw adc input add
- Hw dac output set
- Hw input add
- Hw stepper motor add
- Hw input read
- Hw output write
- Hw message port add
- Hw chr display set text
- Hw dial set acceleration
- Hw switch get position
- Hw adc input read

Getting commands and datarefs…



That’s it – you’re ready to go!

Backup


Generic - Battery switch CLONE
Zoom:
1:1
logic.lua
function battery_switch_callback(position)
if position == 0 then
xpl_command("sim/electrical/battery_1_off")
fsx_variable_write("ELECTRICAL MASTER BATTERY
fs2020_variable_write("ELECTRICAL MASTER BAT
else
xpl_command("sim/electrical/battery_1_on")
fsx_variable_write("ELECTRICAL MASTER BATTERY
fs2020_variable_write("ELECTRICAL MASTER BAT
end
end
hw_switch_add("Battery switch", 1, battery_switch_ca
None
Raspberry Pi Pico
Arduino MEGA 2560
Arduino Uno
Arduino Nano
Arduino Leonardo
Arduino Micro
Raspberry Pi
Flight Illusion GSA-010
Hardware port
Channel A
Channel B
Channel C
Channel D
Channel E
Channel F
Channel G
Channel H
Channel I
Channel J
Pin GP0
Pin GP1
Pin GP2
Pin GP3
Pin GP4
Pin GP5
Pin GP6
Pin GP7
Pin GP8
Pin GP9
Console
Variables
Watch
Call stack
Console
No message
Hardware
Filter on name
Switch:
Pin:
None

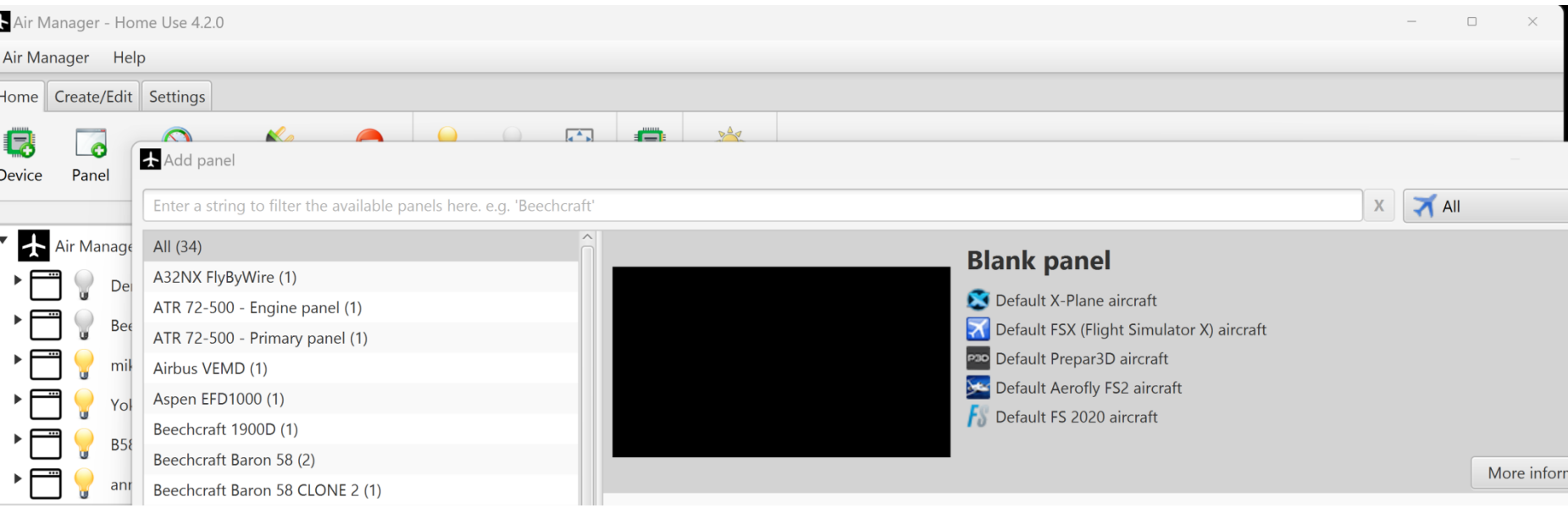
https://www.siminnovations.com/index.php?option=com_content&view=article&id=21

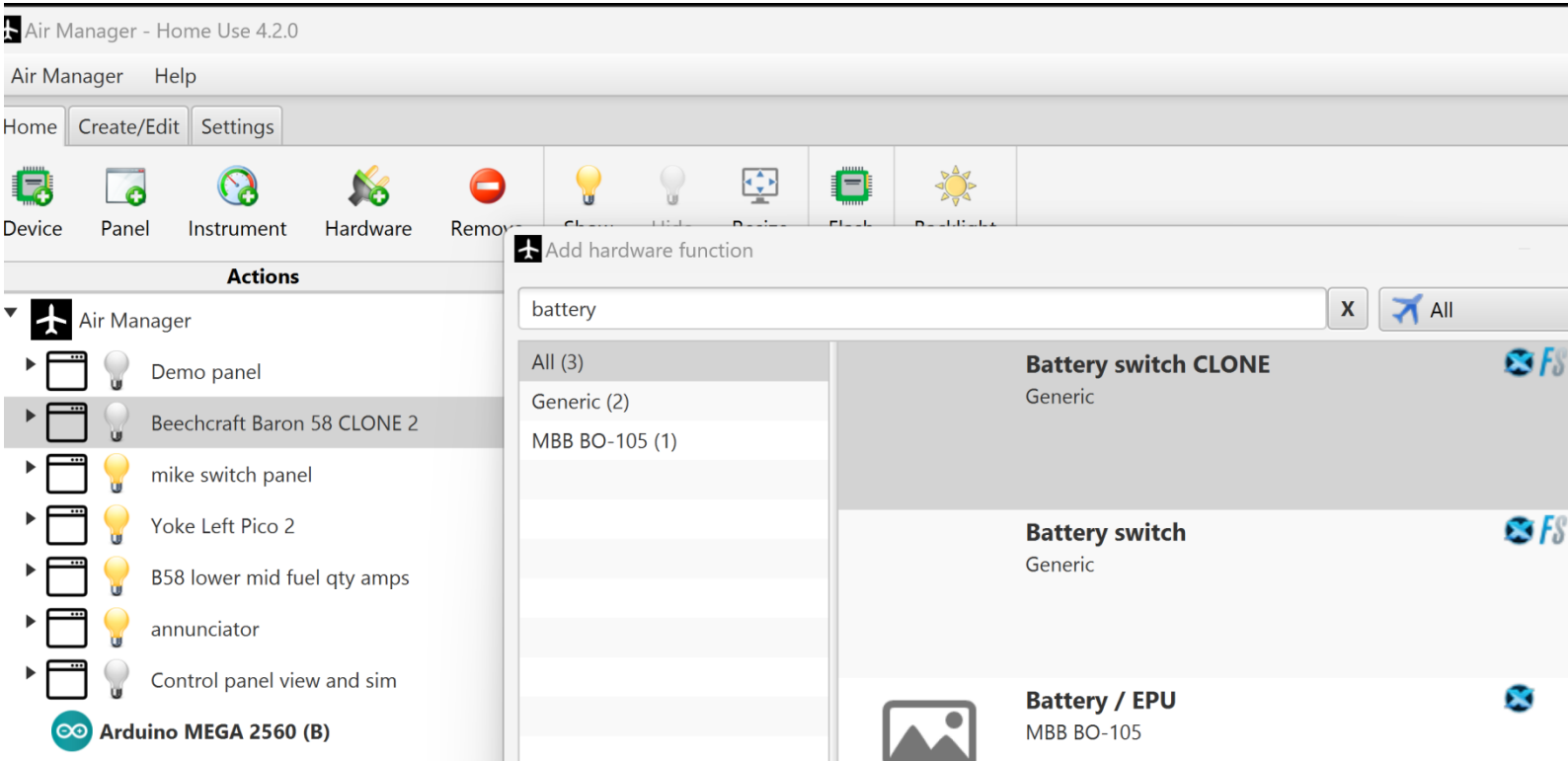
Switch

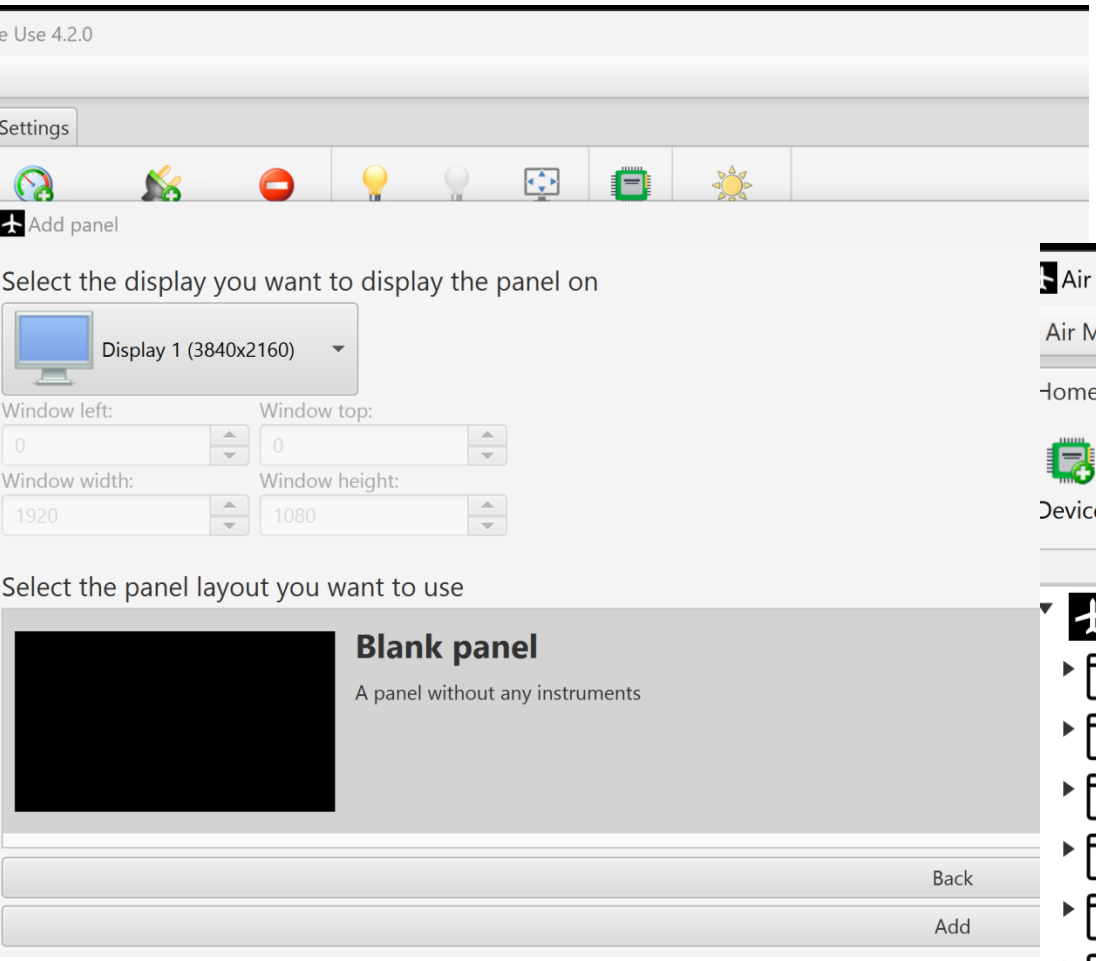
An example on how to connect a switch, this uses the API function hw_switch_add. In this case we use a switch to turn the strobe light on and off.

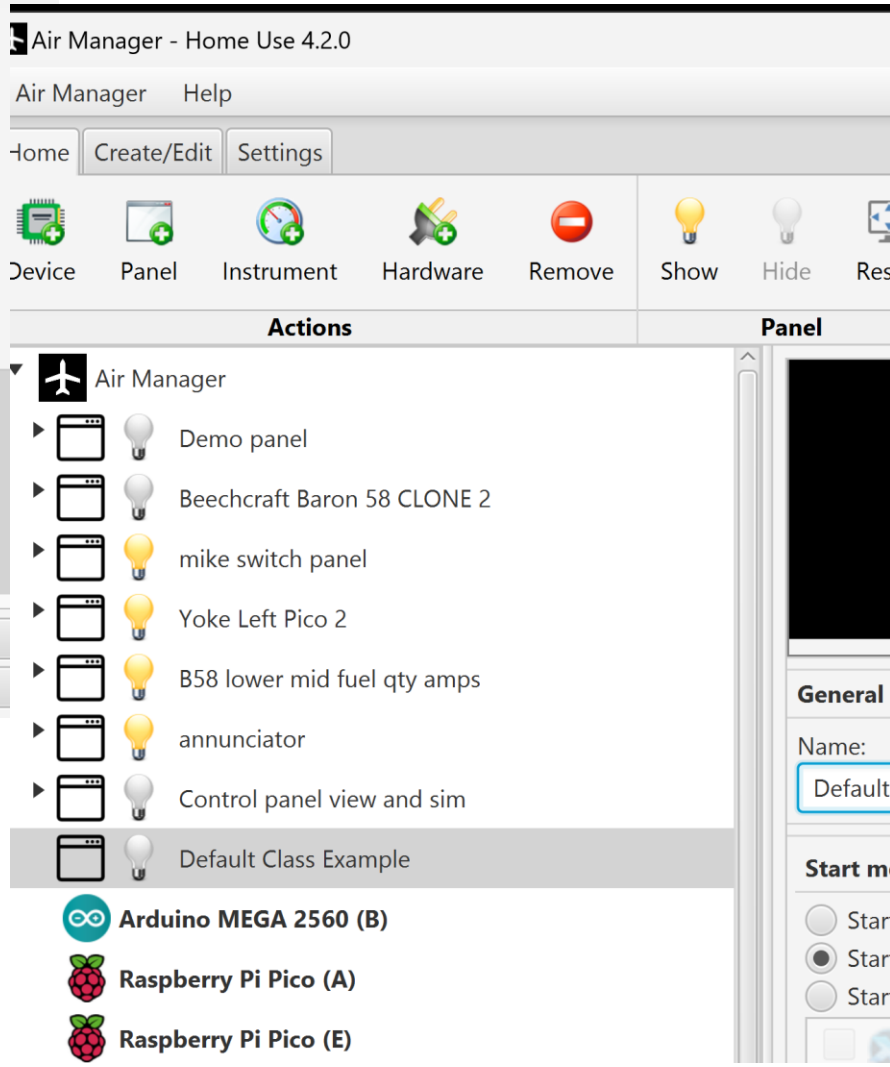
logic.lua

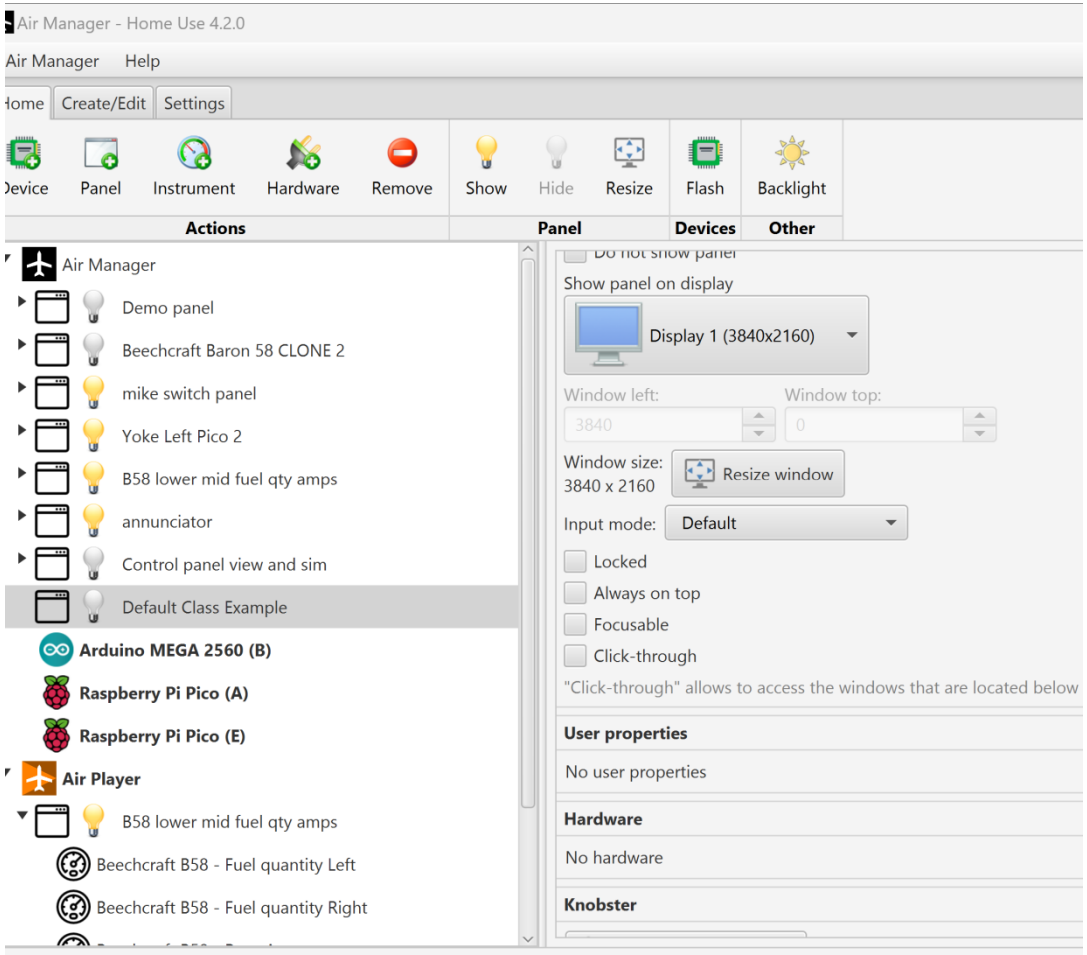
```lua
function battery_switch_callback(position) -- Function called from hw_switch_add below

    if position == 0 then
        xpl_command("sim/electrical/battery_1_off")
        fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
        fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
    else
        xpl_command("sim/electrical/battery_1_on")
        fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
        fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
    end

end

-- hw_switch_add("Battery switch", 1, battery_switch_callback)  -- Arduino,Pi assigned on home
hw_switch_add("RPI_PICO_I_GP17", battery_switch_callback) -- Defined here
```


Air Manager - Home Use 4.2.0
Air Manager
Help
Home
Create/Edit
Settings
Device
Panel
Add panel
Enter a string to filter the available panels here. e.g. 'Beechcraft'
X
All
All (34)
A32NX FlyByWire (1)
ATR 72-500 - Engine panel (1)
ATR 72-500 - Primary panel (1)
Airbus VEMD (1)
Aspen EFD1000 (1)
Beechcraft 1900D (1)
Beechcraft Baron 58 (2)
Beechcraft Baron 58 CLONE 2 (1)
Blank panel
Default X-Plane aircraft
Default FSX (Flight Simulator X) aircraft
Default Prepar3D aircraft
Default Aerofly FS2 aircraft
Default FS 2020 aircraft
More information


Air Manager - Home Use 4.2.0
Air Manager
Help
Home
Create/Edit
Settings
Device
Panel
Instrument
Hardware
Actions
Air Manager
Demo panel
Beechcraft Baron 58 CLONE 2
mike switch panel
Yoke Left Pico 2
B58 lower mid fuel qty amps
annunciator
Control panel view and sim
Arduino MEGA 2560 (B)
Add hardware function
battery
X
All
All (3)
Generic (2)
MBB BO-105 (1)
Battery switch CLONE
Generic
Battery switch
Generic
Battery / EPU
MBB BO-105


e Use 4.2.0
Settings
Add panel
Select the display you want to display the panel on
Display 1 (3840x2160)
Window left:
Window top:
0
0
Window width:
Window height:
1920
1080
Select the panel layout you want to use
Blank panel
A panel without any instruments
Back
Add


Air Manager - Home Use 4.2.0
Air Manager
Help
Home
Create/Edit
Settings
Device
Panel
Instrument
Hardware
Remove
Show
Hide
Actions
Panel
Air Manager
Demo panel
Beechcraft Baron 58 CLONE 2
mike switch panel
Yoke Left Pico 2
B58 lower mid fuel qty amps
annunciator
Control panel view and sim
Default Class Example
Arduino MEGA 2560 (B)
Raspberry Pi Pico (A)
Raspberry Pi Pico (E)
General
Name:
Default


Air Manager - Home Use 4.2.0
Air Manager
Help
Home
Create/Edit
Settings
Device
Panel
Instrument
Hardware
Remove
Show
Hide
Resize
Flash
Backlight
Actions
Panel
Devices
Other
Air Manager
Demo panel
Beechcraft Baron 58 CLONE 2
mike switch panel
Yoke Left Pico 2
B58 lower mid fuel qty amps
annunciator
Control panel view and sim
Default Class Example
Arduino MEGA 2560 (B)
Raspberry Pi Pico (A)
Raspberry Pi Pico (E)
Air Player
B58 lower mid fuel qty amps
Beechcraft B58 - Fuel quantity Left
Beechcraft B58 - Fuel quantity Right
Show panel on display
Display 1 (3840x2160)
Window left:
3840
Window top:
0
Window size:
3840 x 2160
Resize window
Input mode:
Default
Locked
Always on top
Focusable
Click-through
"Click-through" allows to access the windows that are located below
User properties
No user properties
Hardware
No hardware
Knobster