



KCVG Chapter

Switch It Up!

Cockpit Controls with Air Manager

Jan 16 2024 Meeting

Mike Fullington

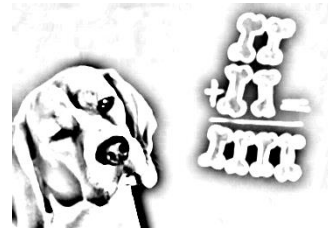
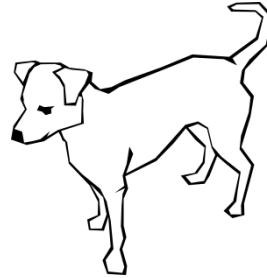
Cockpit Building...



A Simple Equation... using a few inexpensive items



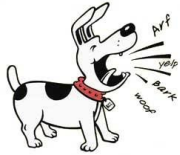
or



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

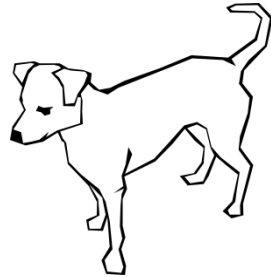
What we have to do...



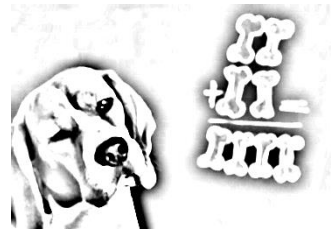
We have to give our board a name and teach it how to speak Air Manager

Flash the hardware and add a device

Turns electrical activity into computer instructions



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



```
1 function battery_switch_callback(position)
2
3     if position == 0 then
4         xpl_command("sim/electrical/battery_1_off")
5         fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
6         fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
7     else
8         xpl_command("sim/electrical/battery_1_on")
9         fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
10        fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
11    end
12
13 end
14
15 hw_switch_add("Battery switch", 1, battery_switch_callback)
```



Option 1 - Map hardware later..



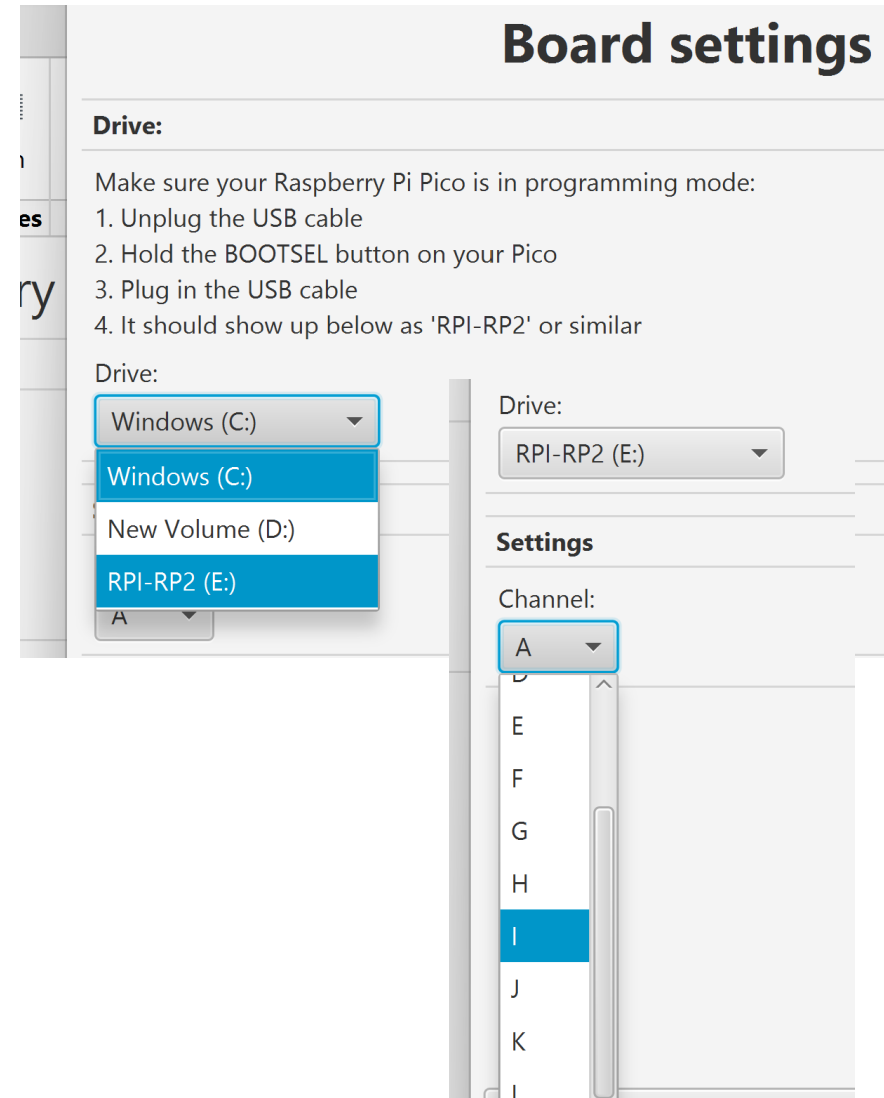
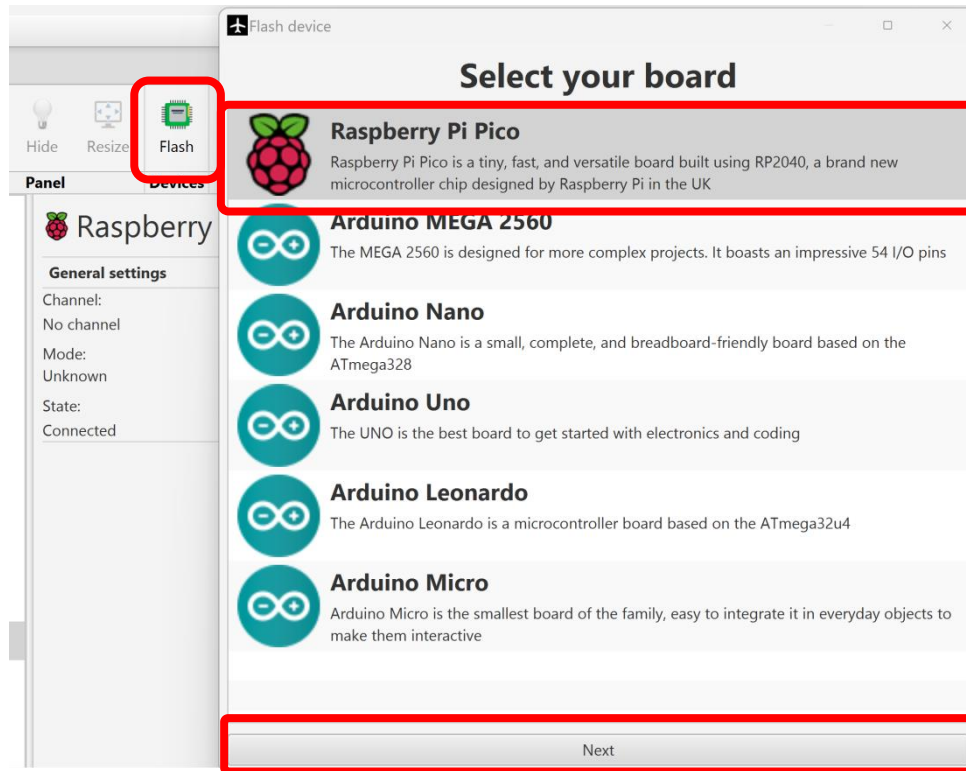
Option 2 - Map hardware now...

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

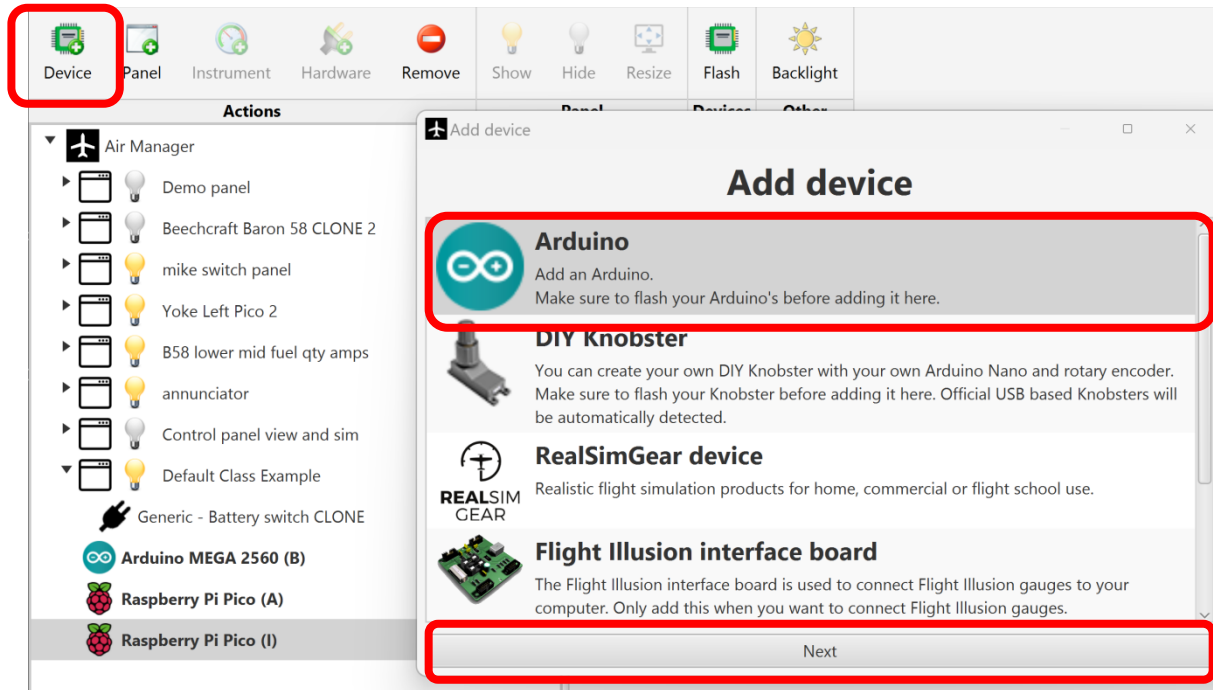
Can have as many of these as you wish in a single LUA file...

Just group the hw_xxxx_add at the end, with multiple functions above

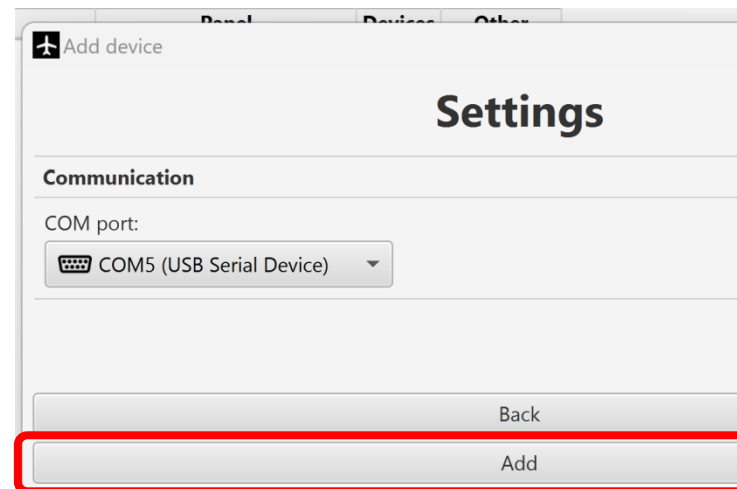
We have the logic, now we need to add the actual hardware (Arduino, Raspberry Pi, etc)
First, name it - flash the board so AM can see it...



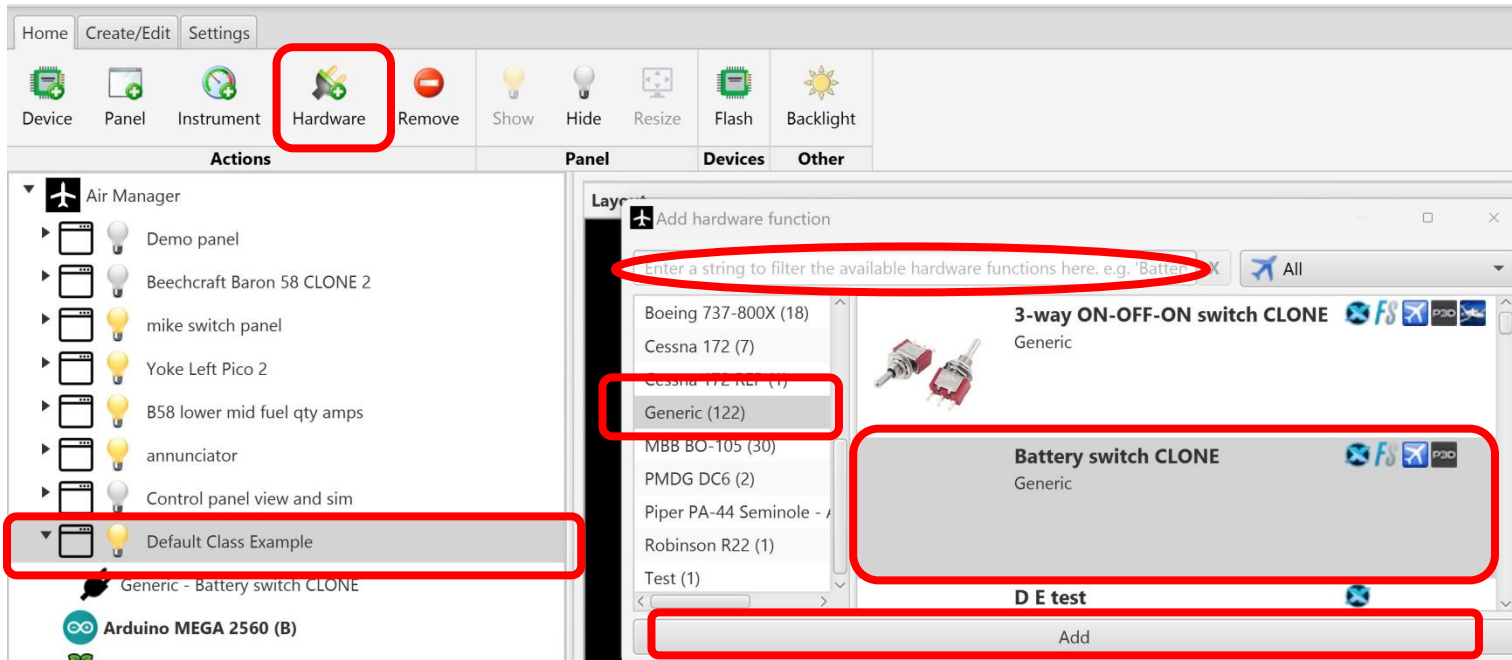
For Arduino's there's one more step...



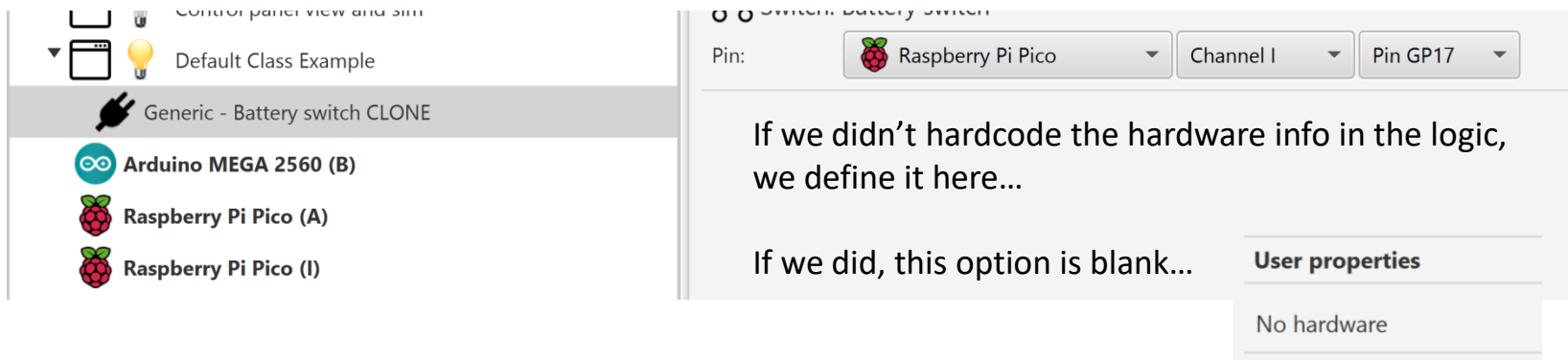
And you have to determine which com port it is on...



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...

User properties

No hardware

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

The screenshot shows the Air Manager software interface. The top menu bar includes 'Air Manager' and 'Help'. Below it, a toolbar contains icons for 'New', 'Clone', 'Delete', 'Submit', 'Import', 'Export', 'Download', 'This PC', 'Run', 'Stop', 'Push', 'Folder', 'Script', 'Info', 'Viewer', 'Tutorial', 'API', and 'Data'. The 'Download' icon is highlighted with a red circle. The main window is divided into several sections. On the left, a sidebar lists 'Hardware functions' with a search filter. The 'Generic - 3-way ON-OFF-ON switch' is selected, and its details are shown in a panel on the right. This panel includes a search bar, a 'Download' button (highlighted with a red circle), and a 'More information' button. The 'Download' button is also highlighted with a red circle. The 'Generic - Battery switch' and 'Generic - Beacon light switch' are also visible in the list.

Then clone it by right clicking it...

The screenshot shows a context menu for a hardware function. The menu items are: 'Run', 'Add to group...', 'Copy the UUID to the clipboard', 'Open folder', 'Open script', 'Edit information', 'Clone', and 'Delete'. The 'Clone' option is highlighted with a blue bar. The background shows a list of hardware functions, with 'Generic - Battery switch' selected.

Add a new hardware function – roll your own...

The image shows the Air Manager software interface with several windows and panels. The main window is titled "Air Manager - Home Use 4.2.1" and has a menu bar with "Air Manager" and "Help". Below the menu bar are tabs for "Home", "Create/Edit", and "Settings". A toolbar contains icons for "New", "Clone", "Delete", "Submit", and "Import". The "New" icon is highlighted with a red box.

A "New plugin" dialog box is open, showing a list of plugin categories: "Instrument", "Panel", "Hardware function", "Device", "Flight Illusion gauge", and "Group". The "Hardware function" category is selected and highlighted with a red box. Below the categories are several checkboxes for compatibility with various flight simulators: "Compatible with X-Plane", "Compatible with FS 2020", "Compatible with FSX (Flight Simulator X)", "Compatible with Prepar3D", and "Compatible with Aerofly FS2". The "Add" button at the bottom right of the dialog is also highlighted with a red box.

In the background, the "General" panel shows a list of existing plugins. The "Hardware functions" folder is expanded, and the "Generic - Flight Illusion Mike" plugin is selected. A red oval highlights the "Generic - Mike Switch" plugin in the list.

The "New plugin" dialog box contains the following fields and options:

- Aircraft: Generic
- Type: Mike Switch
- Group: Unknown
- Author: Enter the author's name
- Version (100 = V1.00): 1
- Compatible with X-Plane
- Compatible with FS 2020
- Compatible with FSX (Flight Simulator X)
- Compatible with Prepar3D
- Compatible with Aerofly FS2
- Description: Enter a description of the hardware function

Navigating your function - info and script buttons...

The screenshot shows the top toolbar of the software. The 'Info' button, represented by a document icon with a pencil, is highlighted with a red circle. Below the toolbar, a list of functions is visible, with 'Generic - Battery switch CLONE' selected. A red box highlights the 'Info' panel for this function, which contains the following details:

- Aircraft: Generic
- Type: Battery switch CLONE
- Group: Unknown
- Author: Sim Innovations
- Version (100 = V1.00): 101
- Compatibility checkboxes:
 - Compatible with X-Plane
 - Compatible with FS 2020
 - Compatible with FSX (Flight Simulator X)
 - Compatible with Prepar3D
 - Compatible with Aerofly FS2
- Description:

The screenshot shows the top toolbar of the software. The 'Script' button, represented by a document icon with a pencil, is highlighted with a red circle. Below the toolbar, a list of functions is visible, with 'Generic - Battery switch CLONE' selected. A red box highlights the 'Script' panel for this function, which displays the following Lua code:

```
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)
```

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	<i>String</i>	A functional name to define the switch.
2	nr_pins	<i>Number</i>	Number of input pins used for this switch.
3	callback	<i>Function</i>	This function will be called when the switch changed position.

Return value

Argument	Type	Description
hw_switch_id	<i>ID</i>	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...

developer.x-plane.com/datarefs/

- API
- Data
 - X-Plane datarefs
 - X-Plane commands
 - FS FS2020
 - FSX variables
 - FSX events
 - P3D Prepar3D variables
 - P3D Prepar3D events
 - SI variables
 - SI commands

X-PLANE DEVELOPER BLOG SCENERY TOOLS VIDEOS DOCUMENTATION MORE

X-Plane Datarefs

battery Writable and Read only All statuses All versions

sim/

sim/cockpit/

sim/cockpit/electrical/

sim/cockpit/electrical/battery_array_on	boolean • int[8] • v8.20+	Writable
Is the battery selected on		
sim/cockpit/electrical/battery_charge_watt_hr	watt/hours • float[8] • v10.10+	Writable
Current charge of each of the 8 batteries in watt-hours.		
sim/cockpit/electrical/battery_EQ	boolean • int • v6.60+	Read only
Does this cockpit have a battery switch		
sim/cockpit/electrical/battery_on	boolean • int • v6.60+	Writable
Is the main battery on		

sim/cockpit/warnings/

Caution and warnings are masters for annunciators by type. Master accept is lit for any warning or caution, so it is a catch-all. Use warnings/annunciators to see if the indicator is lit. Use the command-system to actuate changes to the annunciators.

sim/cockpit/warnings/annunciators/

That's it – you're ready to go!



Backup

Generic - Battery switch CLONE

Zoom: 1:1

logic.lua

```
1 function battery_switch_callback(position)
2
3   if position == 0 then
4     xpl_command("sim/electrical/battery_1_off")
5     fsx_variable_write("ELECTRICAL MASTER BATTERY", 0)
6     fs2020_variable_write("ELECTRICAL MASTER BATTERY", 0)
7   else
8     xpl_command("sim/electrical/battery_1_on")
9     fsx_variable_write("ELECTRICAL MASTER BATTERY", 1)
10    fs2020_variable_write("ELECTRICAL MASTER BATTERY", 1)
11  end
12
13 end
14
15 hw_switch_add("Battery switch", 1, battery_switch_callback)
```

Console Variables Watch Call stack

Hardware

Filter on name

Switch: Hardware port

Pin: None

- 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

[https://www.siminnovations.com/index.php?option=com_content
&view=article&id=21](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
1 function battery_switch_callback(position) -- Function called from hw_switch_add below
2
3     if position == 0 then
4         xpl_command("sim/electrical/battery_1_off")
5         fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
6         fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", false)
7     else
8         xpl_command("sim/electrical/battery_1_on")
9         fsx_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
10        fs2020_variable_write("ELECTRICAL MASTER BATTERY", "Bool", true)
11    end
12
13 end
14
15 -- hw_switch_add("Battery switch", 1, battery_switch_callback) -- Arduino, Pi assigned on home
16 hw_switch_add("RPI_PICO_I_GP17", battery_switch_callback) -- Defined here
```

Device Panel

Add panel

Enter a string to filter the available panels here. e.g. 'Beechcraft'

All

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

- 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

Device Panel Instrument Hardware Remove Clear Hide Design Flight Panel Lights

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

All

- All (3)
- Generic (2)
 - MBB BO-105 (1)

Battery switch CLONE Generic

Battery switch Generic

Battery / EPU MBB BO-105

Settings



Add panel

Select the display you want to display the panel on

Display 1 (3840x2160)

Window left: 0, Window top: 0, Window width: 1920, Window height: 1080

Select the panel layout you want to use

Blank panel
A panel without any instruments

Back

Add

Air Manager Help

Home Create/Edit Settings



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

Start m

Start buttons with radio icons

