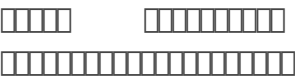




☐☐ OEM☐☐☐

- [☐☐☐☐](#)
- [☐☐☐☐](#)
- [☐☐](#)
- [☐☐☐☐☐☐](#)
- [OEM ☐☐](#)
- [☐☐](#)
- [☐☐☐☐](#)
- [SkyRocket](#)
- [Solo ☐☐☐☐](#)





ArduPilot. GPS. ,
GPS, ** ** ** GCS.

** ArduPilot



**

- ArduPilot (n
- ,
- (GCS) ,



	?	Complete System?	URL
Alpha Unmanned Systems GTRACK	Yes	Yes	common-alphaunmannedsystems-vcs
ARKBIRD AAT	No	Yes	https://www.arkbirdfpv.com/
Mainlink MF18	Yes	No	https://www.szmainlink.com/mf18-antenna/
Motionew AAT v1 + Datalink Box	Yes	?	https://www.motionew.com/
Motionew CommuniNet MND-1410 AAT System	Yes	?	https://www.motionew.com/
Motionew Crossbow AAT	Yes	No	https://www.motionew.com/shop/data-link-video-link/antenna/antenna-tracker/

✈	Antenna ?	Complete System?	URL
Motionew Mini Crossbow AAT	Yes	No	https://www.motionew.com/
MyFlyDream Crossbow AAT	Yes	No	https://www.myflydream.com/
MyFlyDream miniCrossbow	Yes	No	https://www.myflydream.com/
SoarApex ATS20	Yes	No	https://soarapex.com/

Complete System: includes necessary vehicle equipment

**Mission Planner-based GPS Tracking **

- Uses the *Mission Planner* GCS to determine the direction to aim the antenna.
- The PC running Mission Planner must have GPS.
- You will need a special servo driver board to control the servos.

[copywiki destination="plane,copter,rover"]

Antenna Tracking



,

:

,

"

"

.

[[[[]]]] = "[[]] , [] , [] 8.
[[[[]]]] < common-rtf >

$$[\boxed{1} \boxed{2} \boxed{3} \boxed{4}] [\boxed{5} \boxed{6}] [\boxed{7} \boxed{8} \boxed{9} \boxed{10}] = "[\boxed{1} \boxed{2}]".$$

“ ” < >

[/] [] = "cop ".

Navigation icons: back, forward, search, etc.

[illegible]

Navigation icons: back, forward, search, etc.

```
[/[] ]  
    [] < common- store > []  
[[] [] "[] đây"  
    Training Centers <common-training-centers>  
[/site]  
    Wiki Editing Guide <common-wiki_editing_guide>  
[site wiki="planner"]  
    Archived - Recording and Playing Back Missions<common-recording-and-playing-back-missions> Archived - Loading ChiBios Firmware Onto Pixhawk<common-loading-chibios-firmware-onto-pixhawk>  
[/site]
```

[site wiki="plane"]
Common Airframe Builds <common-common-airframe-builds> Release Notes <project-news>

[/site]

```
[site wiki="antennatracker"]
project-news
```

[/site]

[site wiki="dev"]
trademark

[/site]

[site wiki="copter,plane,rover,dev,sub"]
Archived Topics <common-archived-topics>

[/site]

```
[copywiki  
destination="copter,plane,rover,sub,blimp,planner,planner2,dev,antennatracker,mavproxy,ardupilot"]
```



, " " " , β .
 ,
 . , β , .

4.7β " "(4.8dev), 4.7β .
 [" ,copter,rover,sub". ===== [/]



[" ,copter,rover,sub"
 Currawong ESC < common-velocity-can-escs >
 [/]
 [copywiki " ,copter,rover,blimp,sub"].
 [

OEM

ArduPilot OEM, , :

- , gimbals, , .
Mission Planner MAVProxy" ,
/IMU/ RC .
- ROM Lua , SD . , ArduPilot
Lua .
-
-
- , **APJ** .(4.5 ,
)
- [LUA](#) MAVLink

Customization Steps

This section assumes that the OEM has set up the build environment (**building-the-code**) and cloned the ArduPilot GitHub repo locally (**where-to-get-the-code**), to build its customized version of the firmware.

1. Create a branch with the version of firmware you wish to base the customization upon. This will usually be the current stable version. To do this for ArduPlane Stable, for example, assuming you are already in the ArduPilot directory on your PC:

“

```
git fetch https://github.com/ArduPilot/ardupilot.git <version>
      where <version> is the tag for the stable version for the
      desired vehicle: ArduPlane-stable, ArduCopter-stable,
      APMrover2-stable,etc.

git checkout -b <your branch name> FETCH_HEAD
git submodule update --init --recursive
```

2. In the `libraries/AP_HAL_ChibiOS/hwdef` directory, create a new subdirectory for your customized board definitions. In this example, the directory will be named `OEM_CubeOrange` to create a derivative for that board.

3. Create a new `hwdef.dat` file in this format. In this case, our ready-to-fly plane will be using a CubeOrange autopilot and only requires a single line:

```
“  
include ../CubeOrange/hwdef.dat
```

4. The firmware name can be customized by adding one line to the `hwdef.dat` file.

```
“  
define AP_CUSTOM_FIRMWARE_STRING "MyMagicFrame"
```

□ □ □

Custom frame type strings can be created through Lua the scripting method `motors:set_frame_string("Custom frame name")`.

5. Now in that same directory, copy the base board's `hwdef-bl.dat` bootloader file, and then include a file named `defaults.parm`. This file will be the parameter overrides of the standard defaults to match your system's configuration. Things like output function assignments, auxiliary RC switches, flight and tuning parameters, etc.

□ □ □

a parameter can be made readonly (ie user cannot change its value) by marking it `@READONLY` in the file

□ □ □

The `defaults.parm` file should be as small as possible. Some boards only allow 1024 bytes total for this file. Every ASCII byte in the file counts against this limit (except for comment lines). Use integer values where possible. Below is a simple example. Serial port protocols, baud rate, and options defaults can be set directly in the hwdef, as well as NTF_LED_TYPES, and battery monitor defaults, and should be done there instead of a defaults file.

```
# setup for NTF LEDs on output5  
SERVO5_FUNCTION 120  
NTF_LED_TYPES 256
```

1. You can also embed **Lua scripts** in the ROM of the chip that will automatically run. Since Lua is currently only run on autopilots with a lot of flash space, they are only restricted in

total aggregate size to available free flash memory. Put the scripts in a sub-directory called `scripts`, i.e. `libraries/AP_HAL_ChibiOS/hwdef/OEM_CubeOrange/scripts`. Files must end in `.lua`.

Alternatively, you can specify the LUA applet or driver at compile by using the "--embed-<filename>" command, for example:

“

```
./waf plane --embed-LTE_modem
```



The user may also run Lua scripts off the SD card, so care should be taken in naming the embedded script file names to not conflict with potential user file names. It is recommended that the file names of embedded Lua scripts be provided in the product documentation for the user.

1. You can also embed small pieces of documentation in the ROM of the chip that are readable when examining the @ROMFS folder via MAVFtp. These can be pictures or small informational documents. These must fit within the free flash space of the autopilot. These files can be located in sub-directories in `libraries/AP_HAL_ChibiOS/hwdef/OEM_CubeOrange` (e.g. `libraries/AP_HAL_ChibiOS/hwdef/OEM_CubeOrange/AircraftManual`).
2. Now build as normal with OEM-CubeOrange as the board name in the configuration. The default parameters, Lua scripts, and the custom firmware name will be embedded appropriately.

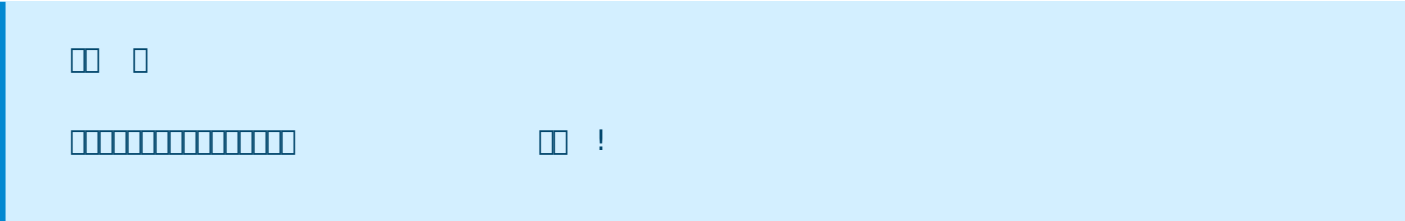
Alternative To Customizing hwdef.dat

Instead of creating a separate branch and modifying the hwdef file, you can also insert Lua scripts or even informational files into the ROMFS of the build. Simply go to the top level of your ArduPilot git checkout and create a sub-folder named "ROMFS_custom" in the root of the source tree. Place your LUA scripts in a sub-folder in this directory, named "scripts" (i.e. the path `ardupilot/ROMFS_custom/scripts`). You can have other sub-folders for informational files and these will be included and viewable when examining the @ROMFS folder with MAVFtp.

```
[copywiki destination="plane,copter,rover,sub,dev"]
```




dev: 2 , ,



“ ” **Software(SITL) ** , ,
Linux, Windows Mac OSX .

" " .

[" "



[{"name": "VTOL"}]. - VTOL

../../images/case-vtol-quadplane.jpg

["sub"] : [{"name": "sub"}]. -

../../images/case-sub.jpg

-

<https://youtu.be/T FMx7C5P 8> isbn= ().

“

:

100%

-

<https://youtu.be/BV91zgxEFHs> ().

“

:

100%

-

<https://youtu.be/8lOn9yDXBLM> ().

“

:

100%

- Boat, equipment, and infrastructure inspection

https://youtu.be/LCWA_9RFQ24

	width:	100%
--	---------------	------

- Biological sampling and surveying

<https://youtu.be/t6J94qbZqKk>

“	width:	100%
---	---------------	------

- Underwater retrieval

https://youtu.be/_tzUh62LPjA

“	width:	100%
---	---------------	------

- Academic and research projects

https://youtu.be/X8CUwg6_y7E

“	width:	100%
---	---------------	------

- ROV and AUV competitions

[/site] [site wiki="rover"] - Bathymetry Boat

../../../../images/case-boat.jpg

<https://youtu.be/p24jFOGKxb8>

“	width:	100%
---	---------------	------

- Autonomous Mowers and Tractors

../../../../images/case-tractor.jpg

[/site] - [Autonomous Vehicle Collision Avoidance](#)

../../../../images/case-cmu-nrec-drone.png

<https://youtu.be/k6jKkpmj4-k>

“

width:	100%
---------------	------

```
[site wiki="plane"]
```

../../../images/case-skymapper.jpg

[/site]

First Person View (FPV) <common-fpv-first-person-view> Non-GPS Positioning Systems
<common-non-gps-navigation-landing-page>

```
[site wiki="copter,plane"]
```

Crop Spraying <common-sprayer>

```
[/site] [site wiki="copter"]
```

“ Indoor Flying <indoor-flying>

[/site]

Vehicle Types

ArduPilot can support many vehicles types: Planes, Multicopter, Helicopter, Rover, Subs, Boats, Blimps, Sailboats, VTOL Planes....and its even used on an orbiting satellite!

Its not only used for autonomous operation but as basic stabilization for pilot controlled vehicles.

See all the supported **vehicle types and frames** here **[site wiki="plane"]** Records =====

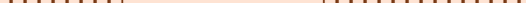
SkyRocket

```
..../images/skyrocket-skyvipergpsdrone.png_
```

Team	Score	Rank
SkyRocket	100.00	1
ArduPilot	90.00	2
RTF	80.00	3



SkyRocket 2021

[illegible][illegible][illegible][illegible][illegible][illegible][illegible][illegible][illegible][illegible][illegible]

ArduPilot 3.10.0

2017 [illegible]

SkyRocket[] [] [] [] [] [] [] [] . □ 2017[] [] [] [] ,□ V245GPS[] [] [] [] ([] [] [] []) [] [] []
ArduPilot.

2018 models

In August 2018, SkyRocket released three new ArduPilot powered drones, the Fury, Scout, and Journey. The Fury and Scout utilize "SurfaceScan", built on ArduPilot's OpticalFlow feature, to allow for indoor flight stability and position holding, unprecedented in this price range. The Journey, with GPS rather than OpticalFlow, takes lessons learned from the V2450GPS and builds on those for a solid re-release for the full GPS-enabled ArduPilot experience. The flight parameters and firmware for both the Scout and Journey can be easily modified as the user sees fit through the embedded wifi access point, powered by APWeb or using your favorite ground control software. Much of the information on this page, which was written for the V2450GPS, will at least partially apply to these new models, with updates annotated as they are discovered.

2021 models

The Journey is not sold anymore. Instead a Sky Viper Journey SE is shipped. Resellers might still have the Journey listed but have been observed to ship the Journey SE.

Discussion

We have a [SkyViper section](#) on the ArduPilot discussion forum. Have a look there to see what people are up to and ask questions.

More developer specific information including how to compile the sonix and ardupilot firmware can be found on the [developer wiki](#).

Toy Mode

The Skyviper has Toy Mode set by default. Toy Mode handles the following functions, specific to the Sky Viper 2450GPS:

- Handles the button presses from the transmitter
- Magically trims the sticks when they're idle and the SV is disarmed
- Toggles the fence on and off depending on the situation
- Basically, if GPS is "good", the fence is armed, if GPS is "not good", the fence is disarmed (for obvious reasons)
- Toy Mode also handles moving you between ALT_HOLD and LOITER automatically depending on GPS Status.
- Handles the LED bling (source: peterbarker)
- Toy Mode automatically adjusts the thrust based on voltage
- Toy Mode processes an arming script when armed using the throttle control to prevent sudden climbs (idles motors for a moment before increasing speed)
- Performs some automatic Compass tuning if needed

Hardware

- STM32 CPU
- 5x serial ports
- 1x I2C
- 1x SPI
- ICM20789 IMU including 3-axis accelerometer, gyro and barometer
- Ublox M8 GPS
- 1S battery (4.2V max, replacement batteries are readily available on Amazon and other places)
- brushed motors; 8.5x20mm with a kV between 16,000 and 17,000 as measured.
- The small pinion is 13T and the larger one is 73T, which provides a gear ratio of approximately 5.6
- the camera can be manually adjusted to point forward, down or anywhere in between
- 2.4Ghz wifi for telemetry and video
- 145g
- flight time of about 11min
- top speed of between 8m/s ~ 10m/s
- video streaming uses a Sonix board with ARM CPU running FreeRTOS and OmniVision OV9732 chip

[../../images/skyrocket-flight-controller.png](#)

[sUAS news](#) interview with Tridge and Matt (from SkyRocket):

<https://youtu.be/3RdEELDleVs>



More Info



Solo

ArduCopter 3.5.0, 3DR Solo.



3DR Solo, Pixhawk 2.0, ArduCopter 3.3, 3DR Solo, Pixhawk, UAS, ArduCopter, 2016.

3D (3D)

3DR Solo, ArduCopter, 3DR Solo, ArduCopter, RTK GPS, ADS-B, lidar, IR, GPS.

../images/solo_skids_here.jpg ../images/solo_logos.jpg ../images/solo_rtk.jpg
../images/solo_nightraptor.jpg

36 Andrew Emmett Matt Lawrence Stephan Schindewolf Paul Dinardi,



Pixhawk 2.1 (CubePilot CubeGreen/CubeBlack) 5 Cube)
ArduCopter 3.5.0 3DR Solo. IMU 5, Solo, 5.

../images/solo_greencube.jpg ../images/solo_cube_installed.jpg

You *can* install ArduCopter master on the stock Pixhawk 2.0, but it is highly discouraged. It will install, and it will fly. But you are at fairly high risk for motors shutting down in flight, leading to a serious crash. This is because of an electrical hardware flaw in the Solo's motor pods. The old stock firmware has a software patch to *mostly* mitigate this flaw. This is what you may hear referred to as "slew rate protection". You can view the code for the slew rate protection in the stock 3DR firmware [here on their GitHub site](#). The production versions of ArduCopter, including ArduCopter 3.5 do not have this slew rate protection. It is severely handicapping and difficult to manage for a world of vehicles that use ArduCopter besides the Solo. The 5 volt signalling used in the Pixhawk 2.1 (CubePilot CubeGreen/CubeBlack) effectively solves the electrical problem on the motor pods.

There are other potential ways to mitigate the electrical problem with the motor pods without buying a new Cube. You could use conventional DIY ESCs and bypass the ones built into the motor pods. Or you could build a level converter that steps the signalling voltage up from 3v to 5v. None of these solutions are commercially available as a kit, but can be done on a DIY basis if you are creative. You would not have the benefit of enhanced hardware in The Cube, but it would be just as safe and reliable to fly.

You cannot use the old stock 3DR Solo firmware on the Cube. It is entirely incompatible. This also means you cannot do a factory reset on the Solo with The Cube still in the Solo. The factory reset tries to reload the old Solo firmware, which is incompatible. If you need to Factory Reset, you will need to put the old stock Pixhawk 2.0 back in, run the full factory reset and update, then put the Green Cube back in. This is annoying, but there is no way around it now or in the foreseeable future. In short, do not need to factory reset. Which also means do not lose your WiFi password! Do not throw away your old stock cube!

Resources

There are several great resources online for modification ideas, vendors, beta testing firmware, troubleshooting, and support

- [Solo Beta Test Facebook group](#)
- [Solo Mod Club Facebook group](#)
- [Solex Users Facebook group](#)
- [ArduPilot Discuss Forums](#)
- [ArduPilot copter Wiki](#)
- [3DR Pilots Forum](#)
- ****Solo Battery Calibration Process ****

Upgrade Process

Preparation

Before beginning the upgrade to ArduCopter 3.5 with the Pixhawk 2.1 (CubePilot CubeGreen/CubeBlack) on your Solo, you need to complete some important requisites.

- Complete Solo and controller in good working order, paired, flyable, and fully charged.
- Solo and controller both up to date with current 3DR Solo firmware from the initial pre-flight update. The current 3DR firmware is 2.4.2.
- Flight tested, working properly in all respects. An untested or malfunctioning Solo should not be used for this process. It won't fix it.
- Pixhawk 2.1 (CubePilot CubeGreen/CubeBlack)
- Philips and flat head screwdrivers
- Solex App for Android or a Windows PC for Mission Planner and WinSCP

Instructions

There are two ways to do the upgrade that have detailed instructions published here.

- ****Initial installation using the Solex app****. This is by far the most straight forward and highly recommended method. Solex has the means to load and reset firmware and parameters, and has access to all the necessary files online directly.
- ****Initial installation using Mission Planner and WinSCP****. This method is a little more involved but is just as successful. You will need to download zip files, use Mission Planner to change settings, and use WinSCP to transfer files. If you do not have the Solex app, this is the method you will need to follow.

FIRST FLIGHT

Once the upgrade process is complete, you're ready to make your first flight using ArduCopter 3.5 on the Solo. It is recommended that your first flight be conducted at a location and time that allows you to test some basic functions and safety systems. Choose a wide open area free of obstructions,

crowds of people, lakes, etc.

Solo/Solex App Settings

You will need to go through all the settings in the 3DR Solo App (and the Solex app if you use that too) to verify and update sliders, options, and settings. Hot items to set include in this sweep include but are certainly not limited to:

- RTH altitude
- RTH/RTM* & Rewind
- Maximum altitude
- A/B Buttons
- Advanced Flight Modes
- Speed sliders
- GoPo settings

Go Airborne

With all of the above complete, it is time to take your first flight on ArduCopter master!

- Take off and verify the Solo flies stable and predictably.
- Test all axes... pitch, roll, yaw, climbs, descends, and even all at once.
- Test the flight modes you have on the A & B buttons
- Make sure you are getting a good GPS lock
- Make sure the distance, altitude, speed, modes, and GPS data displayed on the app and controller are correct and as you expect to see.
- Let the battery run down to the failsafe while hovering safely nearby. Observe its behavior and verify it appropriately executed the RTH/RTM procedure.



If you need help troubleshooting a problem during with this process or have questions, the [Solo Beta Test Facebook group](#) is the best place to go.

ArduCopter Parameters

There are over 700 parameters in ArduCopter. For everyday use of the Solo, you still do not need to worry about any of them. They're all pre-set for you in the above processes and by way of defaults. All the configuration parameters for the Solo that require a value different from the ArduCopter defaults can be found in the [ArduPilot GitHub repository /tools/frame_params/ directory](#). These are the parameters that are loaded during the upgrade process. If you are not familiar with editing parameters and have no special use case to warrant changing them, altering them is not recommended.

However, there are some advanced and special use cases that may require changing some parameters. Some key parameters for advanced users are detailed below. This list will likely grow as new use cases and modifications develop.

COMPASS_ORIENT is the orientation of the external compass.	
Value	Meaning
38	Stock compass in right rear leg
0	HERE External Compass

FS_THR_ENABLE controls how the Solo respond to a loss of signal from the controller.	
Value	Meaning
0	No failsafe. This should not be used.
1	RTH/RTM will initiate if GPS available. If no GPS, Solo will land.
2	Continue with Smart Shot or auto mission. Otherwise RTH/RTM if GPS available. If no GPS, Solo will land.
3	Land only, no RTH/RTM. This is useful for indoor flying.

FS_BATT_ENABLE controls the low battery failsafe action for Copter-3.5 (and earlier). For Copter-3.6 (and higher) check the **BATT_FS_LOW_ACT** parameter. The low battery failsafe kicks in when the values set in parameters <code>FS_BATT_VOLTAGE</code> or <code>FS_BATT_MAH</code> are breached.	
Value	Meaning
0	No failsafe action
1	Land immediately, no RTH/RTM. Useful for indoors.
2	RTH/RTM

FS_BATT_VOLTAGE is the low battery voltage threshold for Copter-3.5 (and earlier). For Copter-3.6 (and higher) check the **BATT_LOW_VOLT** parameter. When the battery voltage drops below this point, the low battery beeper sounds and it will do what you have <code>FS_BATT_ENABLE</code> set for. This value is expressed in volts. The default is 14.0. You can adjust this higher or lower depending on use case and preference.	
--	--

Value	Meaning
0	No low voltage alarm or failsafe
14	14 volts

FS_BATT_MAH is the battery capacity remaining threshold expressed in milliamperes (MAH) for Copter-3.5 (and earlier). For Copter-3.6 (and higher) check the ****BATT_LOW_MAH **** parameter. When the battery remaining capacity drops below this point, the low battery beeper sounds and it will do what you have **FS_BATT_ENABLE** set for. The default is 520. On the solo, that is on average about 1.5 minutes of flying time remaining. You can adjust this up or down to fit your preference and use case. Setting it for 0 will disable capacity remaining based alarms and failsafes.

Value	Meaning
0	No battery capacity remaining alarm or failsafe
520	520 MAH

****WP_YAW_BEHAVIOR **** specifies the yaw behavior in auto missions and RTH/RTM.

Value	Meaning
0	No change. The Solo's yaw will keep pointing in the same direction unless you change it.
1	Face the next waypoint regardless of direction of flight.
2	Face the next waypoint except in RTH/RTM.
3	Face forward along the GPS course.

NTF_OREO_THEME controls the Solo's motor pod LED theme. This is for ArduCopter 3.5.0 and higher only.

Value	Meaning
0	Disabled
1	Aircraft theme with red/green front and white strobes rear.
2	Rover theme with white front & red rear (like a stock Solo used to be).

****AHRS_GPS_USE **** is for enabling or disabling the GPS on you Solo. The default is 1 for enabled. The primary use case for disabling the GPS is for indoor flight. If the GPS is disabled, it cannot and will not try to use it for flight, failsafes, or any other function. That means RTH mode is not available. You must be familiar with how the failsafes work without a GPS. In most cases, the Solo will land since it cannot RTH.

Value	Meaning
0	GPS disabled
1	GPS enabled

LOG_DISARMED enables and disables dataflash (*.bin) logging when the Solo is disarmed. It is currently enabled by default since it can be very useful for testing and troubleshooting. But it does result in large and often unnecessary logs. If you are comfortable and confident in your Solo, you disable logging while disarmed. The dataflash logs be much cleaner and volumnous.	
Value	Meaning
0	Disabled / No dataflash logging while disarmed
1	Enabled / Dataflash logging while disarmed and armed.

Further Information

