

YOLOV5 Implementation

Computer Vision Serial Interface application

Some time you need computer vision on your drone in order to interact with your subject or make automatic decision

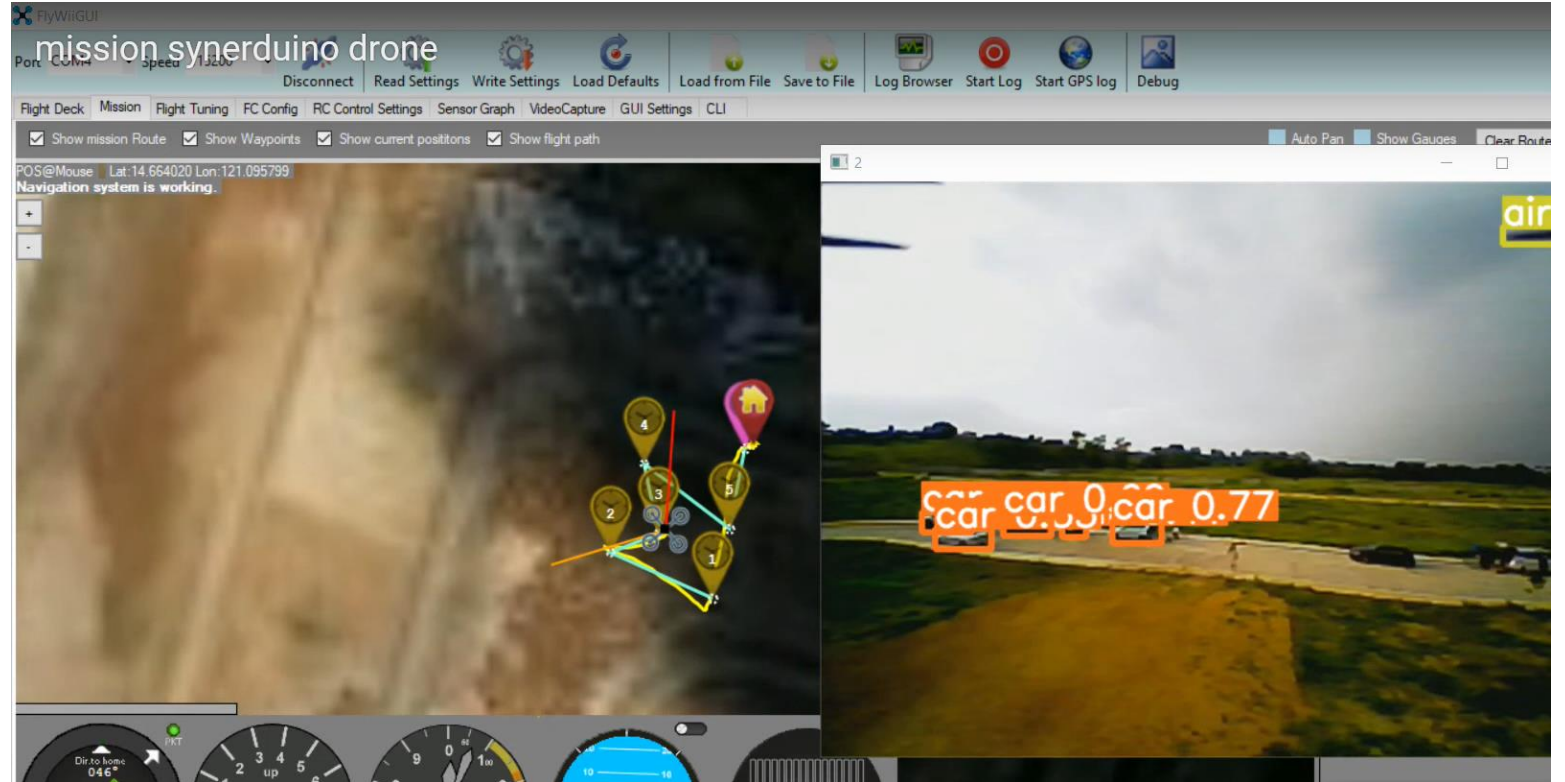
- This works in conjunction with the Synerduino Board and INAV PLC
- Requires FPV camera and OTC UVC Receiver

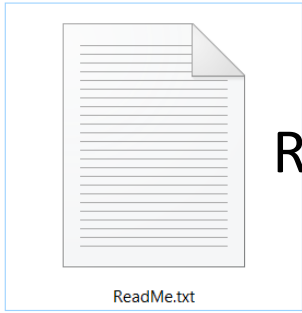
Of the Following Board

- Synerduino Arduino
- Synerduino STM

Computer Vision Serial Interface application

Teaching the Machine what
its Looking For and the
Condition of the subject its
looking at





Readme.txt contains all the information needed for setting up your Yolov5 synerduino



yolov5-install-requirements.bat

Install requirement .bat Batch install all your Yolov5 requirements

Edit : `pip install -r SynerCV2/requirements.txt`



yolov5-custom-detect.bat

This Runs the application Yolov5 with Synerduino Detection

`python SynerCV2/detectardu20.py --source 0 --weights yolov5s.pt --img 640 --class 0`



yolov5-custom-train.bat

See: Yolov5 tutorial on custom Train data and image Annotators

<https://www.makesense.ai/>



datas

Datas - contains all the Parameters for configuring your CV Serial Coms and instructions to send to the Arduino

datas folder contains the readme file for settings configuration and documentation

datas/ baud.txt - change baud

datas/ com.txt - change Serial com

datas/ ipt.txt - object location on screen

datas/ noct.txt - names or classes

datas/ soct.txt - number of objects count



SynerCV2

Syner CV2 - contains all the Yolov5 applications
detectardu20.py – Contains the Python codes and the PT Train image files



ArduinoCV

ArduinoCV - this contains the Arduino sketches and adp files of Ardublock (Arduino 1.8.18 compatible)

Tool folder containing Ardublock must be extracted in
thisPC/Documents/Arduino/Tools

This sketch is used to convert the Serial communication coming from the YOLOV5 and Converting it into PWM or ADC output for the Arduino companion board to read and send appropriate input to Synerduino board to registred

Synerduino Arduino

Arduino Nano

- D3 – (ADC V) – A0
- D5 – (RC PWM) - A14
- D6 – (RC PWM) - A15

Synerduino STM

This would be utilized as output PWM feed from Arduino to Synerduino and INAV PLC

Arduino Mega

- D12 (ADC V) - Voltage ADC
- D13 (ADC I) –Current ADC
- D3 (BEEP/RSSI) –RSSI ADC

Synerduino STM PWM SBUS Converter Method

Also utilizing an Arduino but this time using the Sbus Converter to respond to RC PWM inputs Generated by the Arduino in respond to the serial input

Companion board

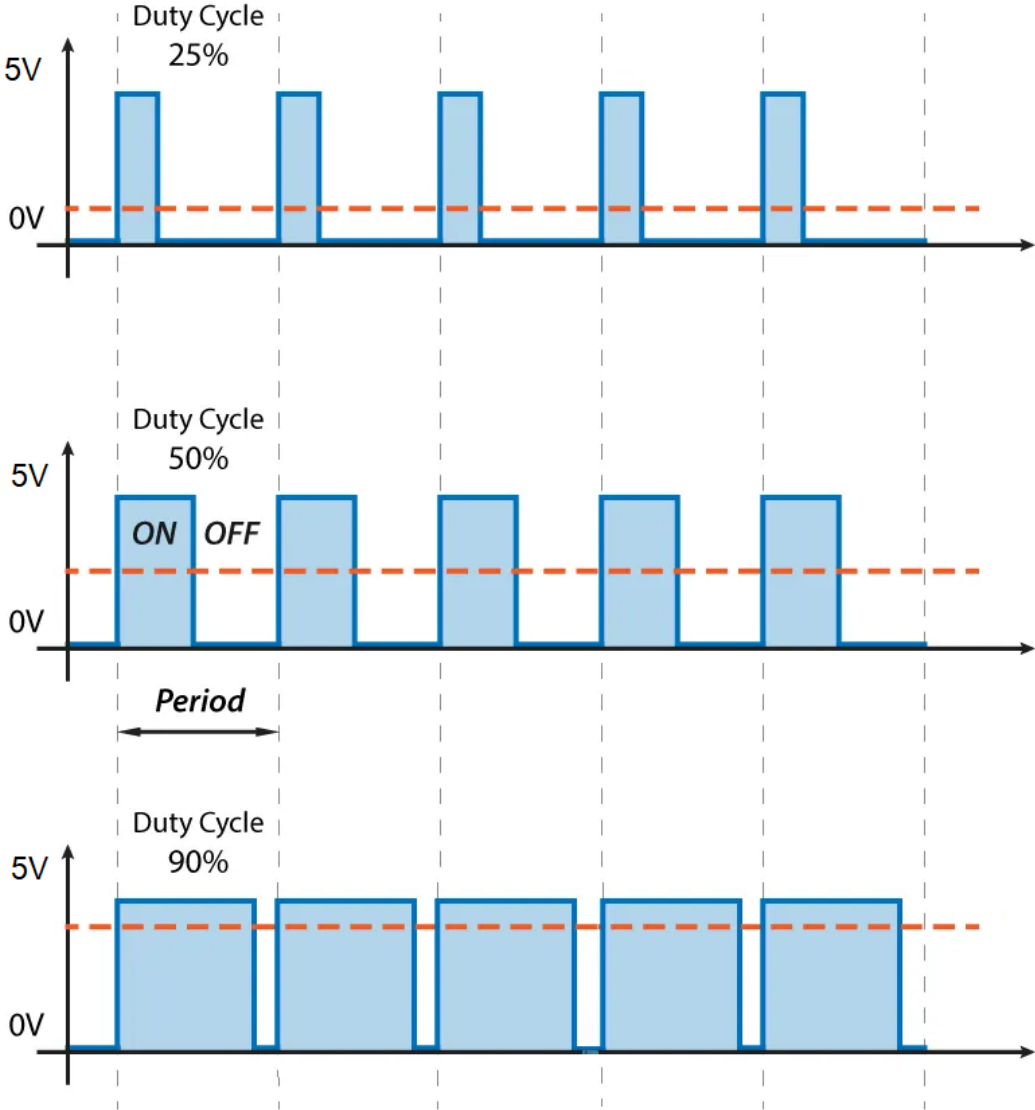
There are several reason you may add a companion controller or computer to perform tasks separate from the Synerduino but need to be able to communicate with it to perform flight modifying action or Data Logging or even extended sensor array that is more than the standalone synerduino board physically accommodate

For Companion board add on options there are several Levels of installation

- ADC – Analog 0V-5V input
- Sbus/PWM – RC PWM 50 hz (1000ms to 2000ms)
- Serial – MSP Telemetry
- PWM Output

ADC Companion

Pulse Width Modulation

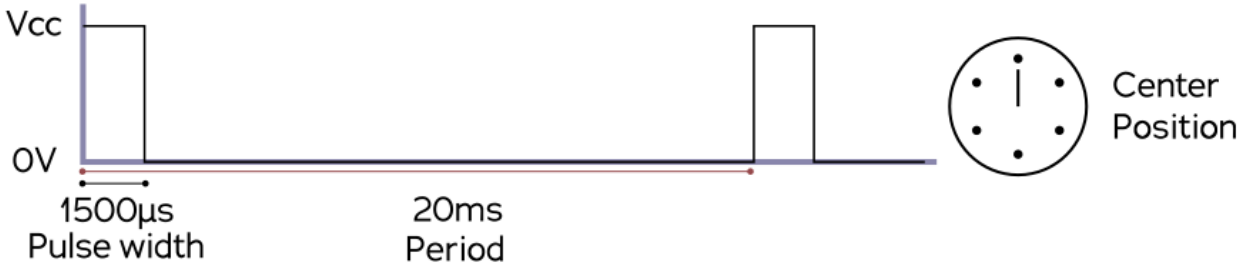


The average voltage depends on the duty cycle, or the amount of time the signal is ON versus the amount of time the signal is OFF in a single period of time.

This works with most Arduino PWM motor driver scripts

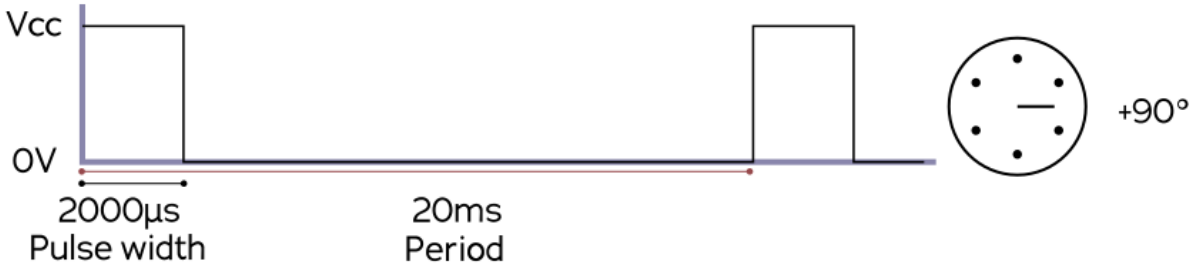
0-225

SBUS/PWM Companion



As RC servo standard this can be taken as an advantage for multiple data points for inputs

This works on Most Arduino Servo Scripts



FPV Standalone / Synerduino Arduino

This requires no introduction as it uses a BEC to supply a standalone FPV25mw camera with integrated VTX

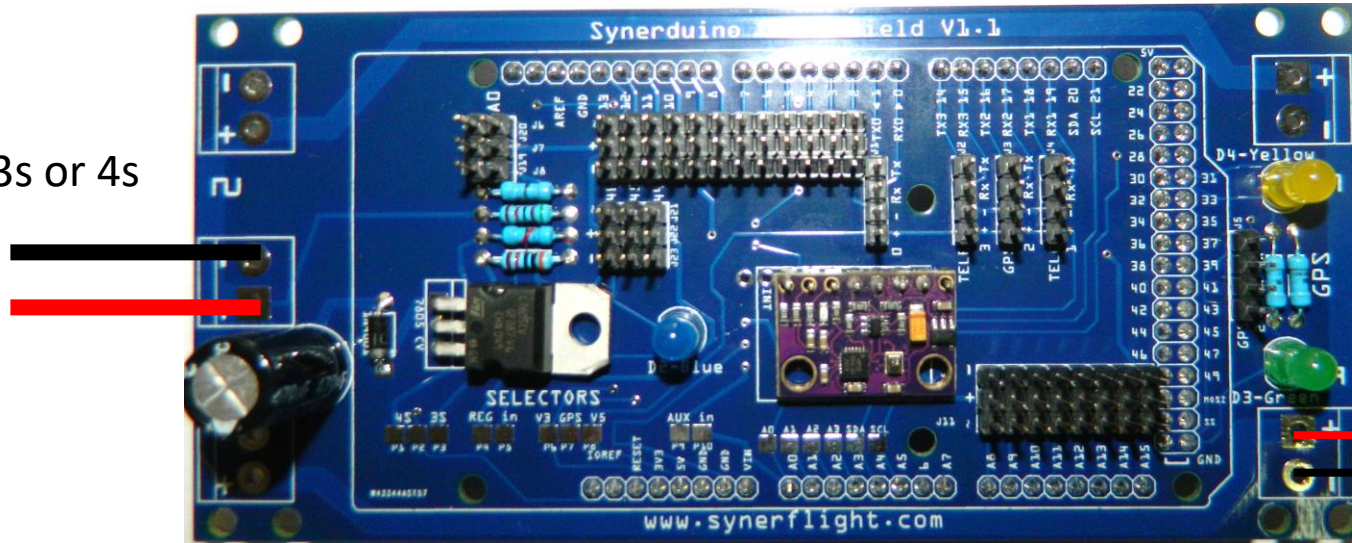
This also can be apply to split camera and VTX sets as well (some Standalone VTX can support 2s to 6s meaning they can directly hook up to the main batter Pads with requiring a BEC supplement)



FPV camera 25mw Standalone



Battery 3s or 4s



BEC or Buck converter supplying extra power

Hardware Setup / Synerduino Arduino

The is useful for analog base inputs

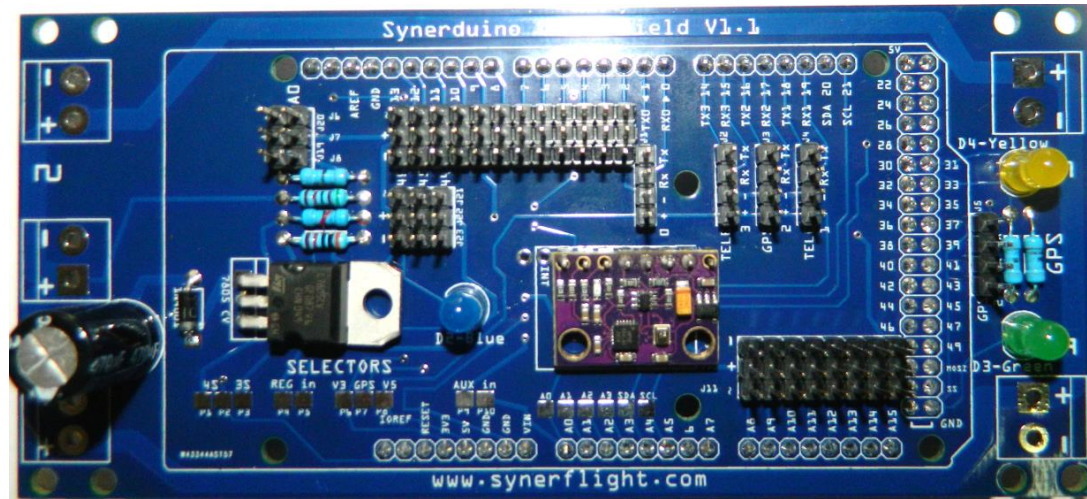
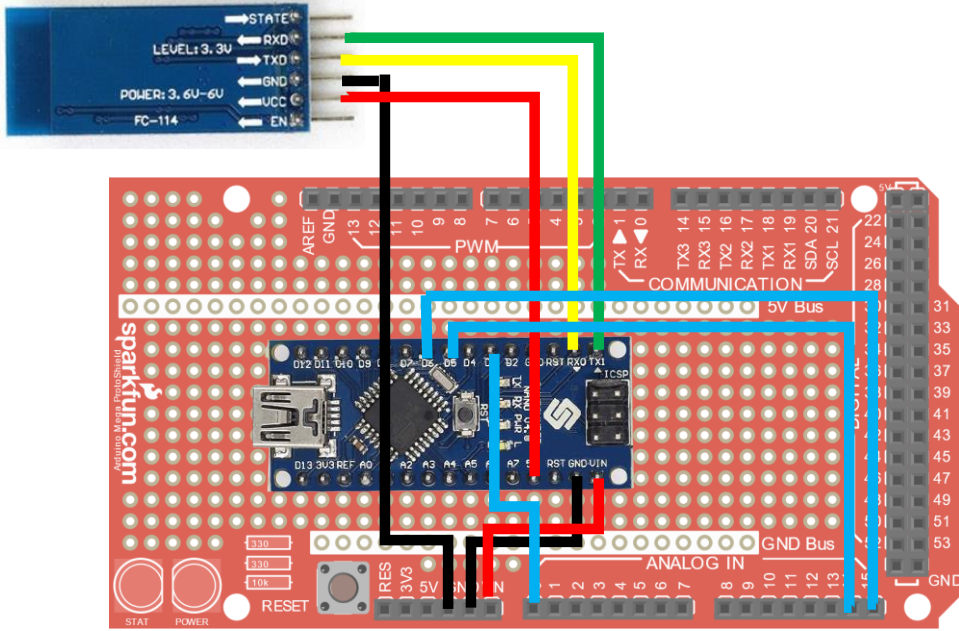
PWM, or pulse width modulation is a technique which allows us to adjust the average value of the voltage that's going to the electronic device by turning on and off the power at a fast rate.

This method is the most simplest interface that one can incorporate even the most basic of logic circuits for simple tasks

Arduino Nano

The Arduino nano on top of a Prototyping shield serves 2 purpose

- Provide a bypass from the Arduino mega to the Synerduino shield
- Allows PWM from the D3 of the Nano to interface with the A0 of the Mega
- Purchase an Arduino NANO without the headers installed (Omit the Top SPI headers) and directly solder the Nano to the Development shield via wire to ensure it still fits under the Synerduino shield



An 2nd Bluetooth or Serial device required to interface with a computer running Yolov5

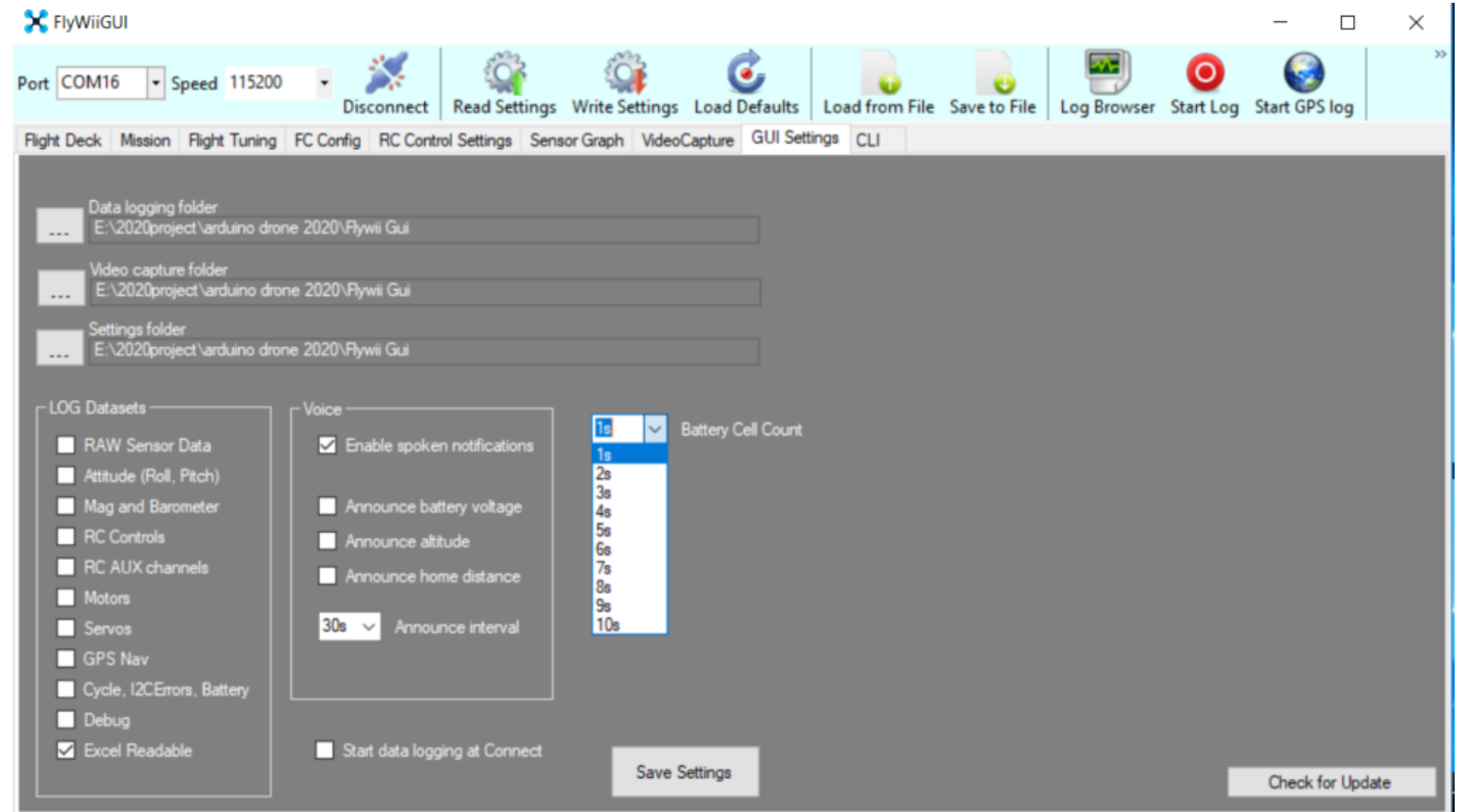
Serial 0 of the Arduino Board

Note: this development shield is the 2nd layer under the Synerduino board
And the Arduino mega on the 3rd Layer

Synerduino Arduino

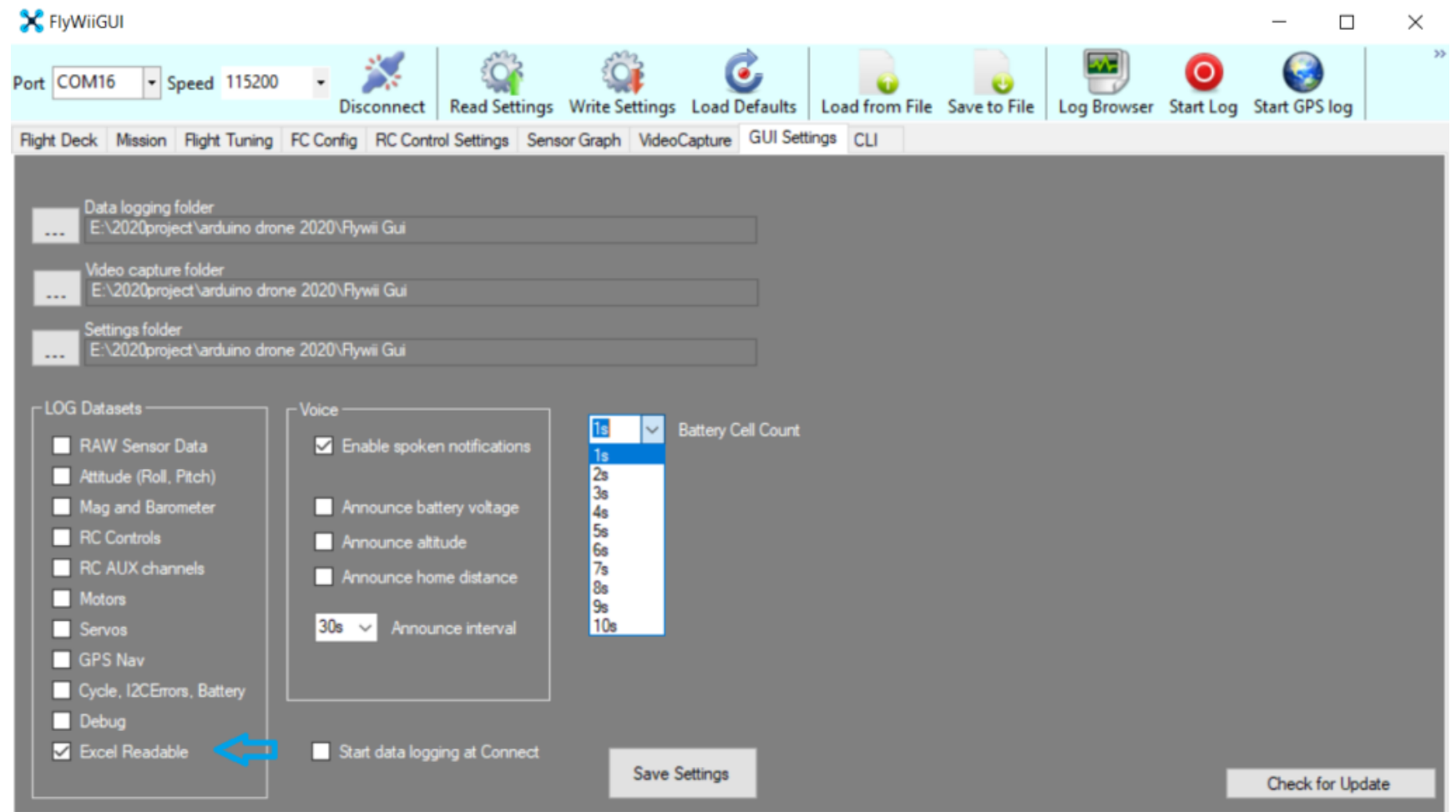
FlyWii GUI Setup

First connect the Drone and go to GUI settings and Change the Battery Cell Count to 1s as we are using it for sensor signal mode.



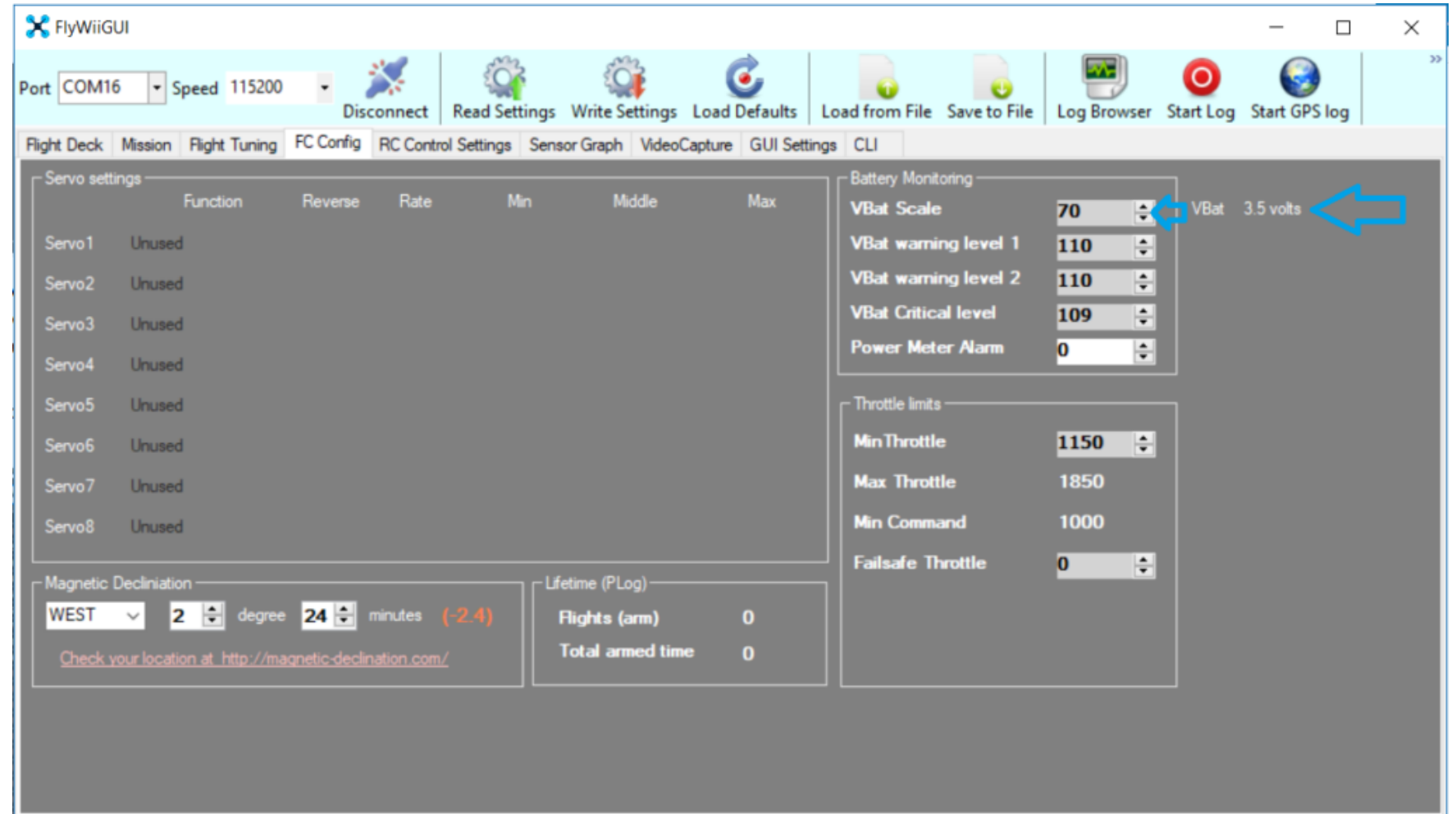
Synerduino Arduino

Available in FlywiiGUI18
select Excel Readable and
Save settings it will run when
you hit the Start Log button
on top or when you setup
start logging on
connect(warning it will start
recording the moment serial
connection is on and would
incur useless idle data)



Synerduino Arduino

Change the Vbat scale to match the voltage range your sensors is capable of delivering (in my case 70) 0V-4.6V analog **Use the Vbat scale to adjust the output till the number Matches the ppm concentration numerical value of the Control Sensor**



The screenshot shows the FlyWiiGUI software interface. The top bar includes a port selection (COM16) and speed (115200). The main menu has tabs for Flight Deck, Mission, Flight Tuning, FC Config, RC Control Settings, Sensor Graph, VideoCapture, GUI Settings, and CLI. The FC Config tab is active, displaying several configuration panels:

- Servo settings:** A table with columns for Function, Reverse, Rate, Min, Middle, and Max. All servos (Servo1-Servo8) are currently set to "Unused".
- Battery Monitoring:** Contains sliders for VBat Scale (70), VBat warning level 1 (110), VBat warning level 2 (110), VBat Critical level (109), and Power Meter Alarm (0). A label "VBat 3.5 volts" is shown next to the VBat Scale slider, with two blue arrows pointing to it.
- Throttle limits:** Contains sliders for Min Throttle (1150), Max Throttle (1850), Min Command (1000), and Failsafe Throttle (0).
- Magnetic Declination:** Shows a dropdown for "WEST", a value of "2" degree, and "24" minutes, with a red "-2.4" indicator.
- Lifetime (PLog):** Shows "Flights (arm)" and "Total armed time" both set to 0.

Synerduino Arduino

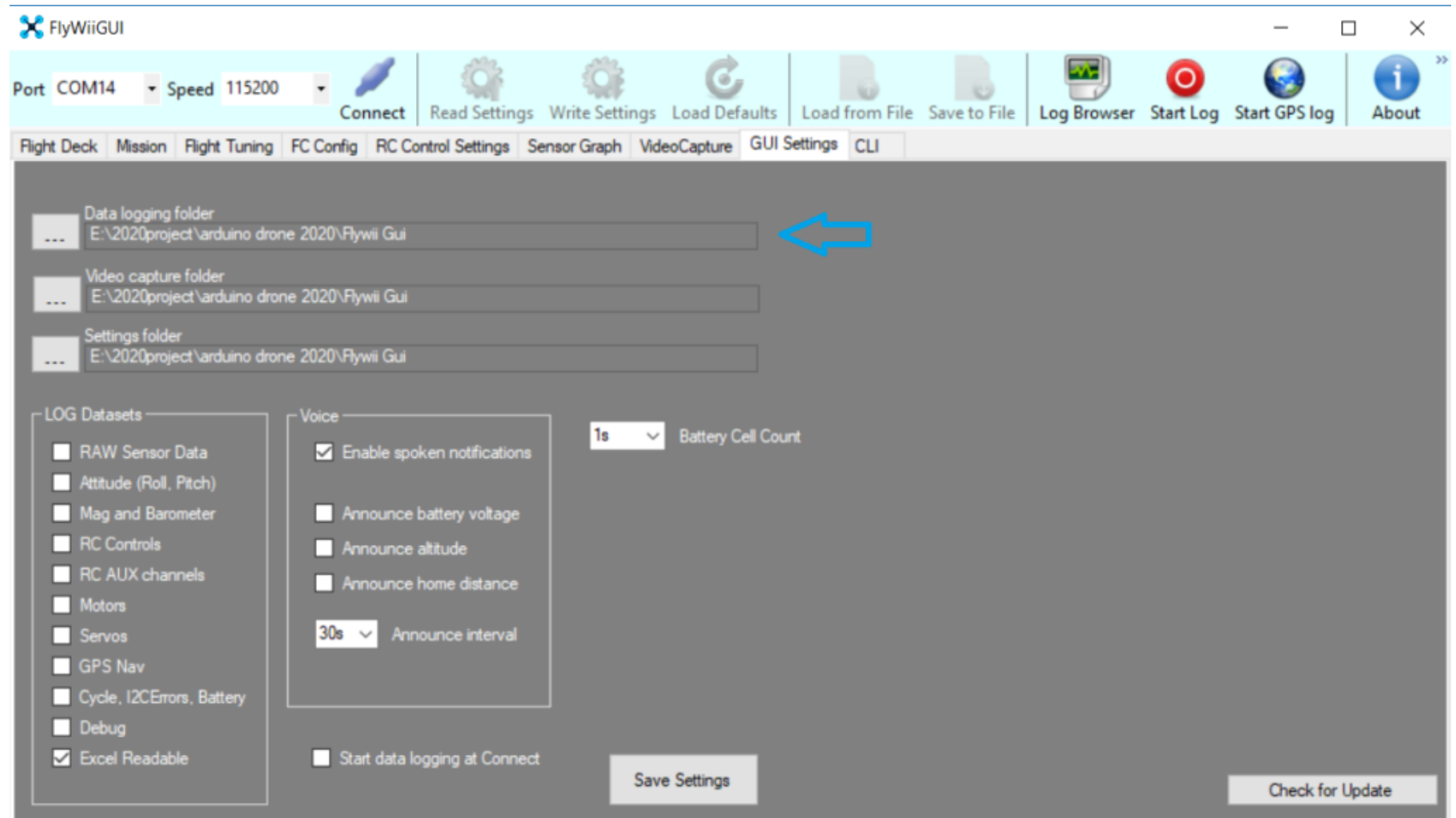
For A0 (Aux input)

Aux sensor should do a read out like this depending if your sensor is resistance base it go up or down



Synerduino Arduino

In GUI Settings you need to indicate which folder the Log data is to be saved to



Synerduino Arduino

Select Log button anytime you want to start and when you want to see the logs select log browser

For Information on how to export as CSV and create Charts and Graphs on Spread sheet

see :

Add on Integration tab Gas Sensors and ADC Data Logging



FPV Standalone / Synerduino STM

- Requires FPV camera and OTC UVC Receiver

FPV Standalone

This requires no introduction as it uses a BEC to supply a standalone FPV25mw camera with integrated VTX

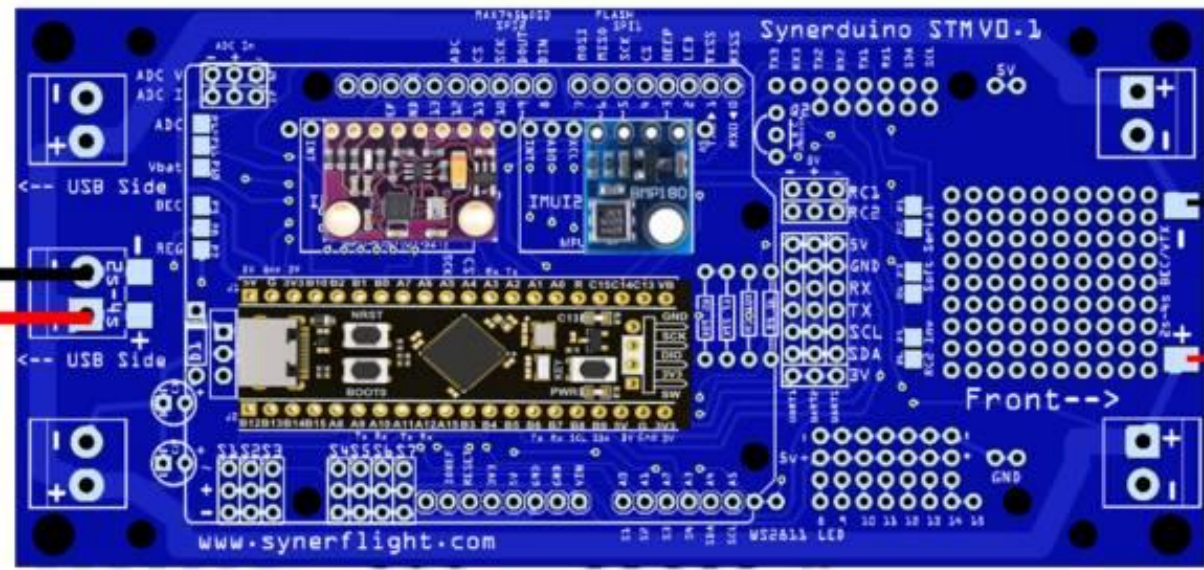
This also can be apply to split camera & VTX sets as well (some Standalone VTX can support 2s to 6s meaning they can directly hook up to the main batter Pads with requiring a BEC supplement)



FPV camera 25mw Standalone



Battery 3s or 4s



BEC or Buck converter supplying extra power

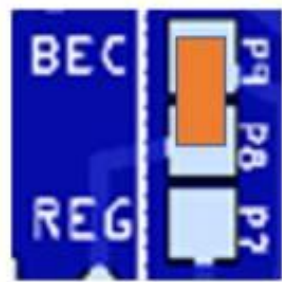
FPV Standalone / Synerduino STM

FPV with SERIAL OSD

- Requires FPV camera and OTC UVC Receiver



The Telemetry can also use the Serial OSD module



Power Pads is selected to BEC to use external BEC

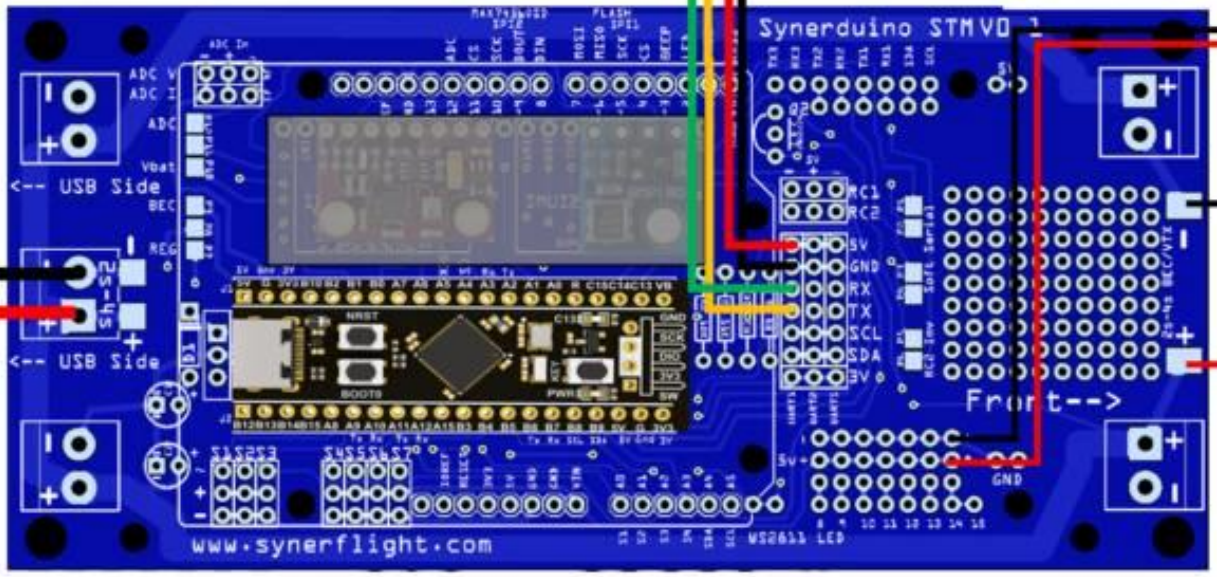
Battery 3s or 4s

UART 1 needs to be configure in Ports to OSD flysky serial

Frsky OSD module



FPV camera 25mw Standalone



BEC or Buck converter supplying extra power

Hardware Setup / Synerduino STM

This is useful for analog base inputs

PWM, or pulse width modulation is a technique which allows us to adjust the average value of the voltage that's going to the electronic device by turning on and off the power at a fast rate.

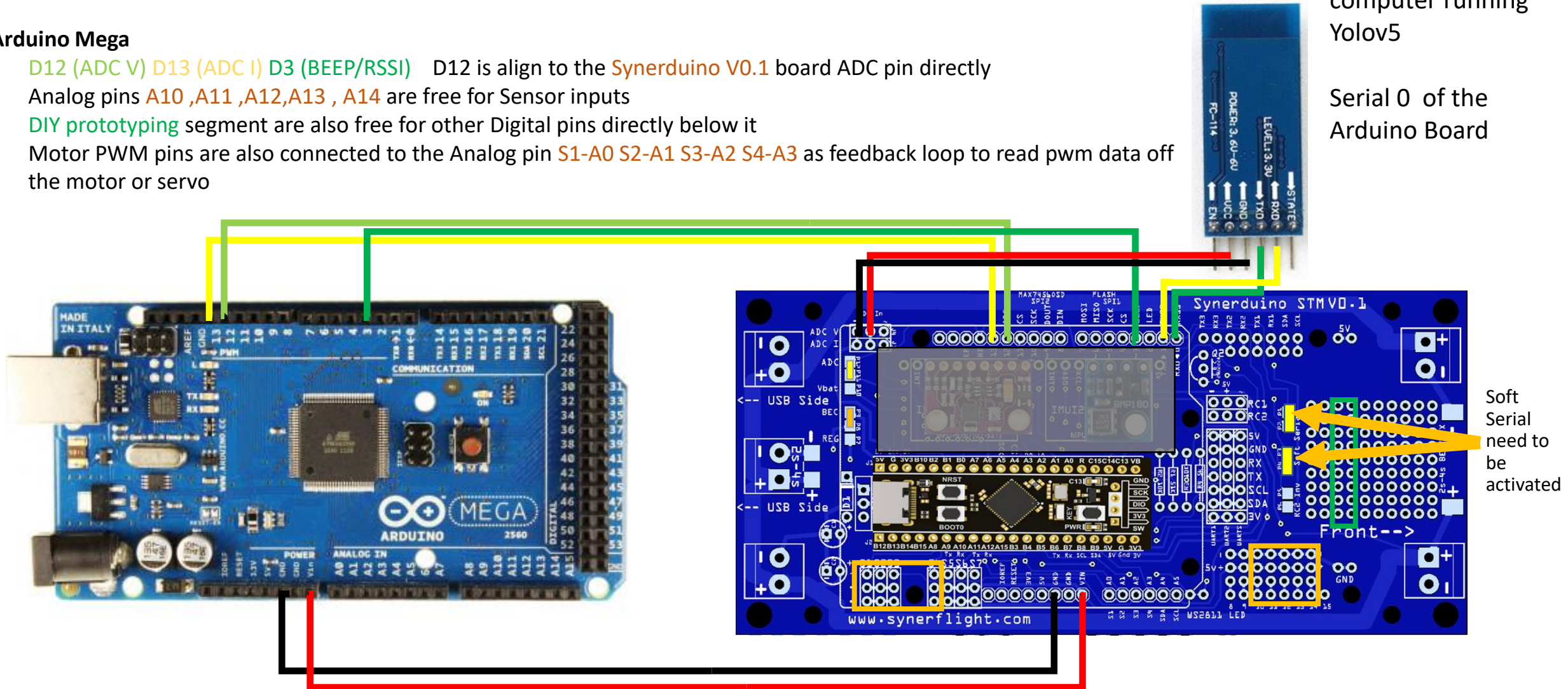
This method is the most simplest interface that one can incorporate even the most basic of logic circuits for simple tasks

Arduino Mega

- **D12 (ADC V) D13 (ADC I) D3 (BEEP/RSSI)** D12 is align to the Synerduino V0.1 board ADC pin directly
- Analog pins **A10 ,A11 ,A12,A13 , A14** are free for Sensor inputs
- **DIY prototyping** segment are also free for other Digital pins directly below it
- Motor PWM pins are also connected to the Analog pin **S1-A0 S2-A1 S3-A2 S4-A3** as feedback loop to read pwm data off the motor or servo

An 2nd Bluetooth or Serial device required to interface with a computer running YoloV5

Serial 0 of the Arduino Board



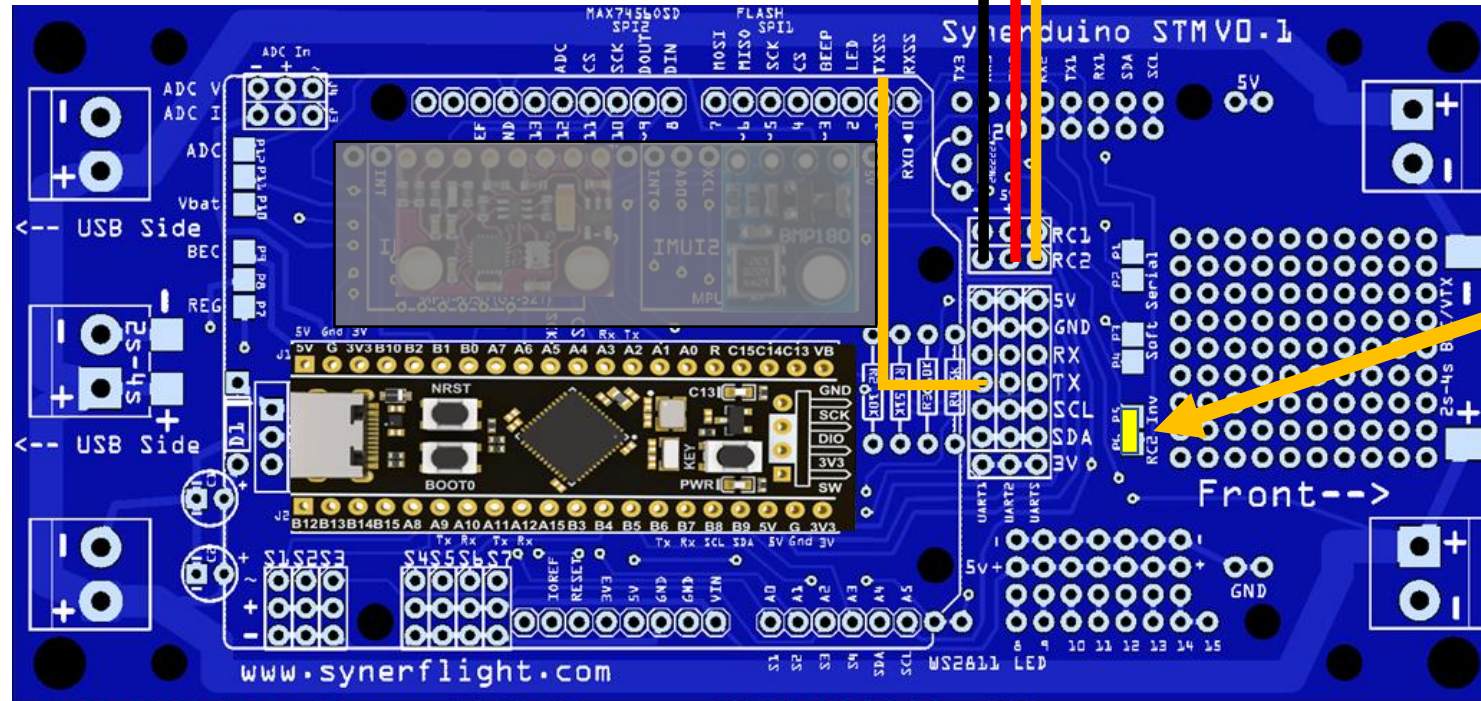
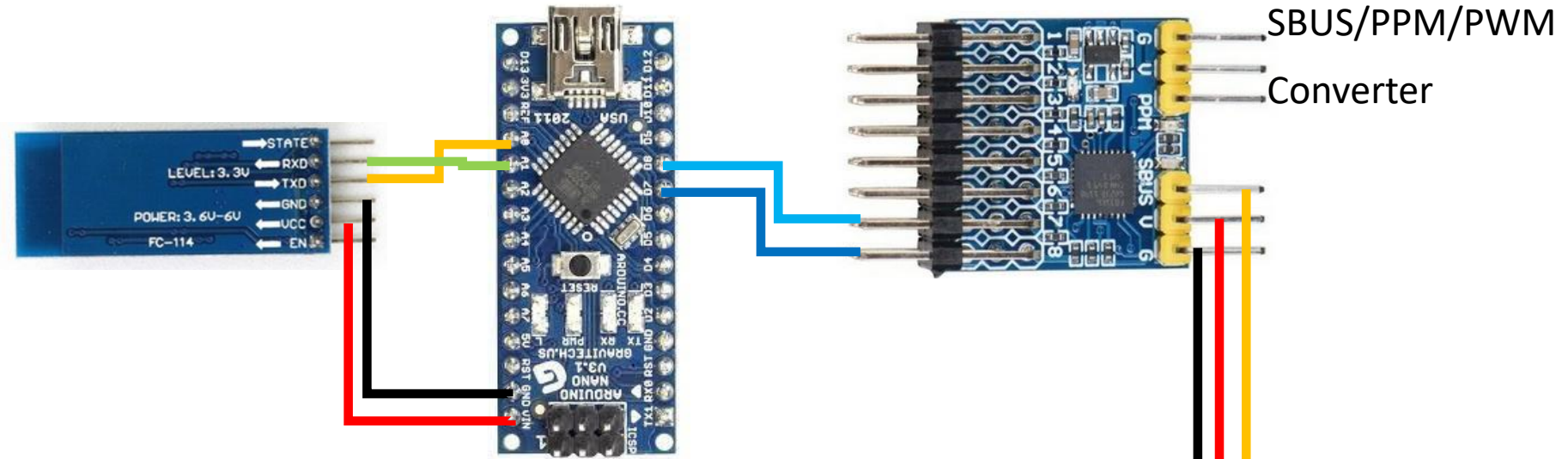
UART Serial Devices

SBUS Converter Method

For those who Uses PWM or PPM Receiver Require to add an Additional PWM/PPM/SBUS Converter to RC2/UART2

This method also does work should you require precise PWM input

Same as Radio Transmitters with PL Jacks



Synerduino STM

INAV Programming (PLC)

This is the definitive feature of INAV combine with the Synerduino Shield .

This PLC function allows you to program upto 8 GVAR and instructions from timer to sensor conditions to trigger a Flight mode action or control action of your Drone

The screenshot shows the INAV Configurator software interface. The top status bar displays a battery level of 5.48V and various sensor icons (Gyro, Accel, Mag, Baro, GPS, Flow, Sonar, Speed, IMU2). The left sidebar contains navigation options: Configuration, Failsafe, PID tuning, Advanced Tuning, Programming (selected), Receiver, Modes, Adjustments, GPS, Magnetometer, Mission Control, OSD, LED Strip, Sensors, Tethered Logging, Blackbox, and CLI. The main area displays the Logic Conditions configuration page. It shows 8 GVAR values: GVAR 0: 8, GVAR 1: 549, GVAR 2: 0, GVAR 3: 0, GVAR 4: 0, GVAR 5: 0, GVAR 6: 0, GVAR 7: 0. Below this is a table of logic conditions:

#	Enabled	Operation	Operand A	Operand B	Active	Flags	Status
0	<input checked="" type="checkbox"/>	Increase GVAR	Value	0	Value	1	Always
1	<input checked="" type="checkbox"/>	Greater Than	Global Variable	0	Value	55	Always
2	<input checked="" type="checkbox"/>	Set GVAR	Value	0	Value	0	Logic Condition 1
3	<input checked="" type="checkbox"/>	Set GVAR	Value	1	Flight	Vbat [centi-Volt] [1V = 100]	Always
4	<input checked="" type="checkbox"/>	Greater Than	Global Variable	1	Value	545	Always
5	<input checked="" type="checkbox"/>	Override RC Channel	Value	6	Value	55	Logic Condition 4
6	<input type="checkbox"/>	True					
7	<input type="checkbox"/>	True					
8	<input type="checkbox"/>	True					

The bottom status bar shows system metrics: Packet error: 0, I2C error: 0, Cycle Time: 517, CPU Load: 22%, MSP version: 2, MSP load: 2.0, MSP round trip: 66, HW round trip: 17, Drop ratio: 7%, and version 5.0.0. The system clock shows 9:22 PM on 17/10/2022.

ADC Companion

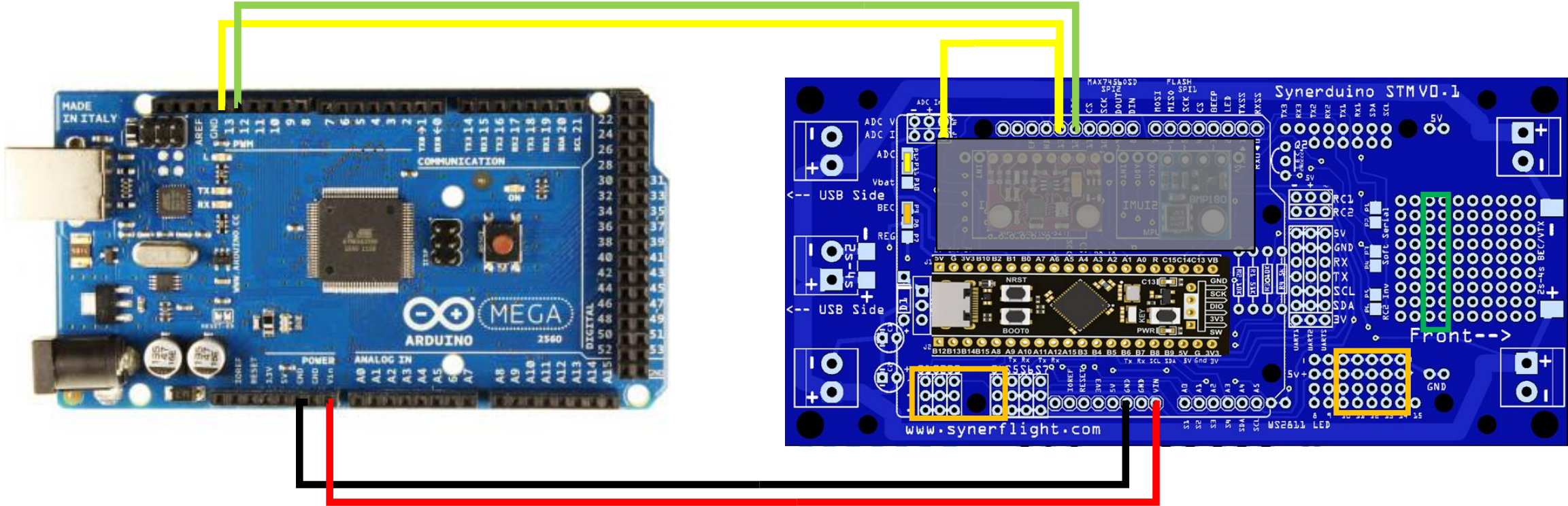
The is useful for analog base inputs

PWM, or pulse width modulation is a technique which allows us to adjust the average value of the voltage that's going to the electronic device by turning on and off the power at a fast rate.

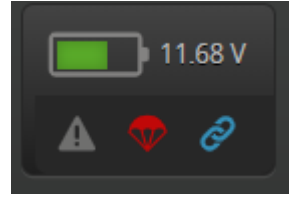
This method is the most simplest interface that one can incorporate even the most basic of logic circuits for simple tasks

Arduino Mega

- D12 (ADC V) D13 (ADC I) D12 is align to the Synerduino V0.1 board ADC pin directly
- Analog pins A10 ,A11 ,A12,A13 , A14 are free for Sensor inputs
- **DIY prototyping** segment are also free for other Digital pins directly below it
- Motor PWM pins are also connected to the Analog pin S1-A0 S2-A1 S3-A2 S4-A3 as feedback loop to read pwm data off the motor



Data Analytic Intervention Via Companion board



Configuration

Similar Low battery failsafe the DAI Starts by configure your Battery Cell Count ,Current and Voltage scale this way you can adjust to your sensors while calibrating the value output.

But in this case instead of the Battery its ADC data would be inputed from the Companion Controller (Arduino)

This would define your signal point on where to trigger.

ADC V (Battery Voltage Monitoring) could be input 1 D12

ADC I (Battery Current Monitoring) could be input2 D13

You can set your Arduino to input a specific PWM indicating a Data is send

Voltage and Current Sensors	
<input checked="" type="checkbox"/>	Battery voltage monitoring
ADC	Voltage Meter Type
Raw	Voltage source to use for alarms and telemetry
450	Voltage Scale
11.64	Battery Voltage
<input checked="" type="checkbox"/>	Battery current monitoring
ADC	Current Meter Type
400	Current Meter Scale
0	Offset in millivolt steps
48.24	Battery Current

Battery Settings	
3	Number of cells (0 = auto)
4.25	Maximum cell voltage for cell count detection
3.3	Minimum Cell Voltage
4.2	Maximum Cell Voltage
3.5	Warning Cell Voltage
mAh	Battery Capacity Unit
0	Capacity
	Warning Capacity (remaining %)
	Critical Capacity (remaining %)

Synerduino STM

SBUS/PWM Companion

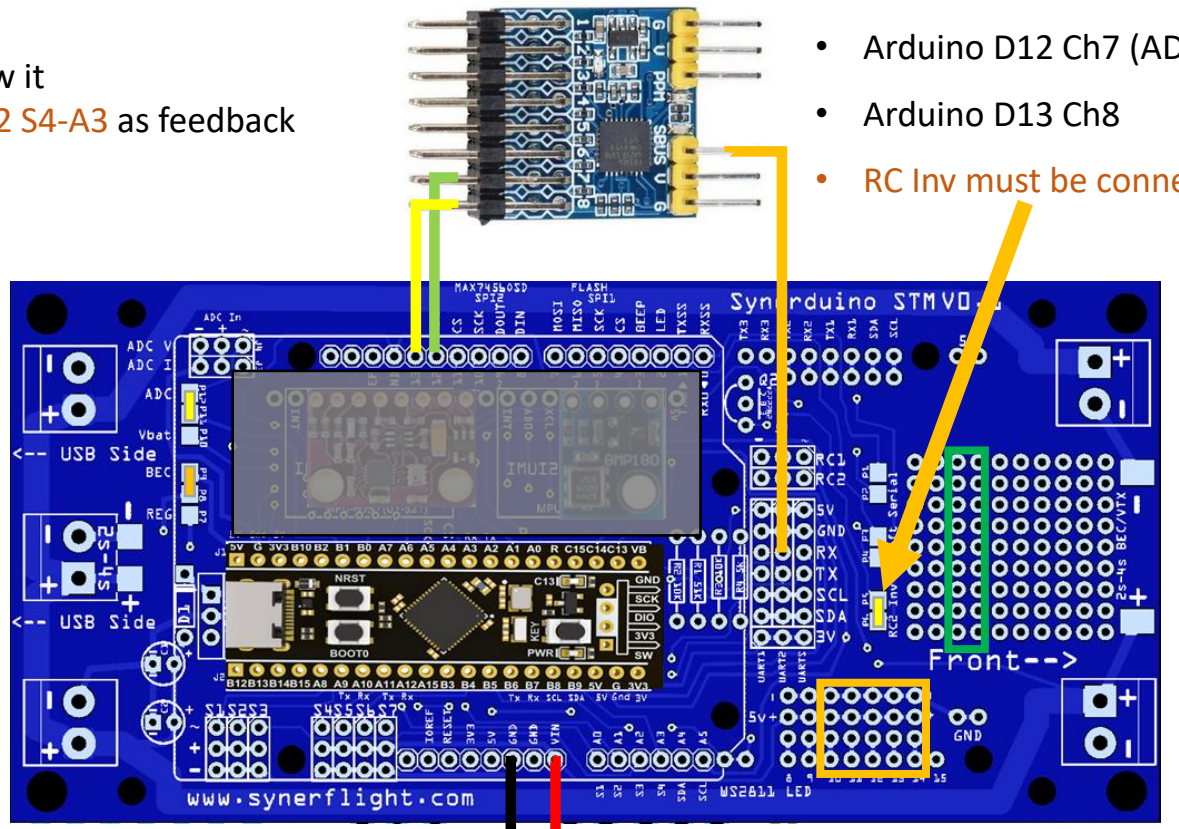
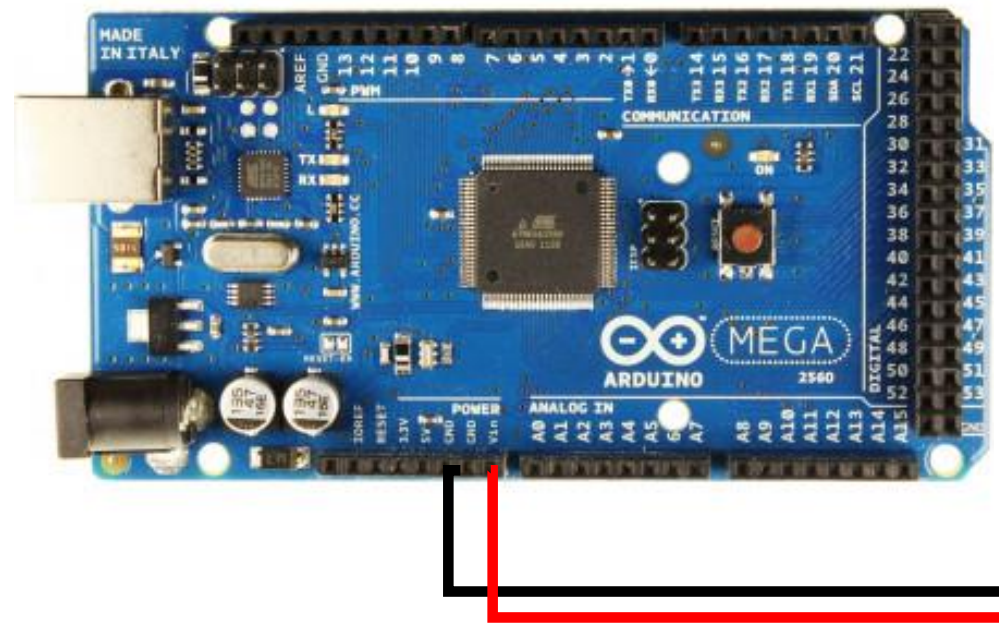
In the world of Robotics /RC /Drones
 There is always a standard of PWM called the RC PWM its set at 50hz
 And range from 1000us to 2000us
 Of pulse this controls the value of the servos

This is said to be useful as interface when you need to have a semi digital range you want to instruct vehicle to do

- Analog pins **A10 ,A11 ,A12,A13 , A14** are free for Sensor inputs
- **DIY prototyping** segment are also free for other Digital pins directly below it
- Motor PWM pins are also connected to the Analog pin **S1-A0 S2-A1 S3-A2 S4-A3** as feedback loop to read pwm data off the motor

This Setup requires an SBUS/PPM/PWM converter

- Arduino D12 Ch7 (ADC V)
- Arduino D13 Ch8
- **RC Inv must be connected**



Data Analytic Intervention

SBUS Converter Method



In Sbus Converter Method We Wanted the RC Channel 7 and 8 to be free for the Mode Trigger this would be active when a PWM value is send by the companion board (Arduino)

The PWM is RC Servo PWM this is useful if you need a properly define value

The screenshot shows the INAV Configurator software interface. The top bar displays system status: battery at 11.64 V, and various sensor icons (Gyro, Accel, Mag, Baro, GPS, Flow, Sonar, Speed, IMU2). The main content area is titled "Receiver" and includes a "DOCUMENTATION" link. A yellow warning box contains text about configuring the serial port and channel map. Below this, the "Channel Map" section shows a list of channels with their corresponding values:

Channel	Value
Roll [A]	1500
Pitch [E]	1500
Yaw [R]	1500
Throttle [T]	885
CH 5	1675
CH 6	1500
CH 7	1500
CH 8	1500
CH 9	1500
CH 10	1500
CH 11	1500
CH 12	1500
CH 13	1500

The "Receiver Mode" section is set to "SERIAL" with a note: "Note: Remember to configure a Serial Port (via Ports tab) for the serial receiver". The "Serial Receiver Provider" is set to "SBUS". The "Serial Port Inverted" option is "OFF", and "Serial receiver half-duplex" is "AUTO". The "RC Smoothing" section is set to "OFF". At the bottom, a status bar shows system metrics: Packet error: 0, I2C error: 0, Cycle Time: 508, CPU Load: 17%, MSP version: 2, MSP load: 0.3, MSP round trip: 26, HW round trip: 17, Drop ratio: 0%. The bottom right corner shows the time 10:42 AM on 04/01/2023.

Data Analytic Intervention

In this sample we use ADC I (Flight Current) as a sample where we hook up the sensor to

0# set GVAR
Operand A = Value 0
Operand B = Flight Current / Vbat Volt
Active = Always

First we need to visualize the Current input in this case GVAR0 4849

1# Greater than
Operand A = Flight Current / Vbat Volt
Operand B = Value 2670
Active = Always

Operand B can be change depending on the Arduino input value want to trigger from

2# Override RC Channel
Operand A = Value 7
Operand B = Value 200
Active = Logic condition 1

Here is where the magic happens when the conditions are met in Logic #1 with an active Status this would trigger the Value 7 (Ch7) to set PWM to 200 overriding the RC input and triggering the servo or Payload

The screenshot shows the INAV Configurator software interface. At the top, there's a status bar with battery voltage (11.58 V) and various sensor icons (Gyro, Accel, Mag, Baro, GPS, Flow, Sonar, Speed, IMU2). Below that, a log shows several 'EEPROM saved: Programming' entries. The main area displays GVAR settings for GVAR 0 through GVAR 7. GVAR 0 is set to 4849. Below the GVAR settings is a table of logic conditions:

#	Enabled	Operation	Operand A	Operand B	Active	Flags	Status
0	<input checked="" type="checkbox"/>	Set GVAR	Value 0	Flight Current [centi-Amp] [1A = 100]	Always		
1	<input checked="" type="checkbox"/>	Greater Than	Flight Current [centi-Amp] [1A = 100]	Value 6270	Always		<input type="radio"/>
2	<input checked="" type="checkbox"/>	Override RC Channel	Value 7	Value 300	Logic Condition 1		<input type="radio"/>
3	<input type="checkbox"/>	True					
4	<input type="checkbox"/>	True					
5	<input type="checkbox"/>	True					
6	<input type="checkbox"/>	True					
7	<input type="checkbox"/>	True					
8	<input type="checkbox"/>	True					
9	<input type="checkbox"/>	True					

The bottom status bar shows system metrics: Packet error: 0, I2C error: 0, Cycle Time: 503, CPU Load: 17%, MSP version: 2, MSP load: 0.0, MSP round trip: 240, HW round trip: 17, Drop ratio: 0%, and version 6.0.0-FP2.

Note: if your RC Channel is occupied by an Receiver channel lets say 7 or 8 and you don't want to override it. You can set the next free channel 9 - 10

Synerduino STM

Data Logging

For Information on how to export as CSV and create Charts and Graphs on Spread sheet

See:

Add on Integration Synerduino STM ADC sensors in the Implementation Tab

Tethered Logging Page

