# Programming AMRTs for reliable F5J performance

#### Feature article by Tuan Le (fnnwizard@gmail.com) October 2021

**Forward:** We are very fortunate to have this article contributed by Tuan Le. Tuan is well known in the RC soaring community for his analytical and technical skills (in addition to being an excellent soaring competitor and all-around great guy!). This summer Tuan did a deep dive into how to program your TX so that it works predictably with your AMRT (altimeter/motor timer). One of the things he realized is that without a tool to display your motor control channel signal you were "flying blind" when it came to properly setting up your TX's motor control signal range.

So Tuan rolled up his shirtsleeves and designed a clever control signal reader that displays the pulse width modulation (PWM) signal in microseconds. The construction of that reader is in this article. He also included a section on how to program a BL Heli ESC which is available in a separate PDF via the link at the end of this article. Thanks much for the article and sharing your design work, Tuan! --Editor

#### The AMRTs we use have specific settings per FAI code:

#### 3.4 Motor Run Timer

- 3.4.1 The Motor Run Timer function must operate with the accuracy specified in item 2.3.5.
- 3.4.2 On initial power-up of the AMRT, the Motor Run Timer must be protected from false triggering during the period defined in Item 2.4 (c). After this period, the Motor Run Timer shall commence operation and 'Start Height' determination shall be initiated on the first occasion that the command signal exceeds 1.2 milliseconds.
- 3.4.3 After the Motor Run Timer has been triggered, the timing of the motor run ceases when either the command signal reduces below 1.2 milliseconds, or the elapsed time reaches 30 seconds and a definite stop command must be applied to the ESC by the AMRT.
- 3.4.4 The 'Start Height' determination process shall continue for a further 10 seconds after the instant of "motor stop".
- 3.4.5 Once the definite stop command has been applied to the ESC, the system must not respond to any further changes in the motor command signal.
- 3.4.6. A small amount of hysteresis is permitted in the sensing of the 1.2 milliseconds motor command signal to eliminate malfunction with jittering command signals.

From this code, we see that the "trigger" value for motor start and stop is at 1200us (1.2ms) with "a small amount of hysteresis" on either side of 1200us. I have found out the hysteresis is +- 20us. So to the up side, Altis sees motor on at 1220us, and sees

motor off at 1180us.

But the motor off has another step in there and can be verified with its logger. What happens is that the Altis measures and records the TX signal low throttle position when first booted up. It uses that as the default setting when we cross the 1180 threshold throttle off.

For example, if our throttle off position is 1000us, upon moving our throttle from above 1200us to below 1180us, the Altis immediately cuts it further to 1000us.

I've posted this info on RCG showing how an unscrupulous person can go about tricking the Altis to gain unfair advantage in a competition. https://www.rcgroups.com/forums/showpost.php?p=47499577&postcount=21

Since most Tx/Rx combo have user settings capable of transmitting PWM signal between 900us - 2100us, the fixed 1200us setting specified can cause issues for the AMRT.

If we assume this hysteresis to be 20us in either direction for all AMRTs, (it may or may not be due to manufacturing tolerances of all the components for F5J) then the following programming can be used for effective control of Motor, ESC, AMRT.





Figure 1 - PWM programming range

Tx/Rx **THROTTLE OFF**, using switch etc., should read lower than 1180us. I like to add some extra buffer of 30us so anything between 900us - 1150us.

Tx/Rx **THROTTLE IDLE** should be higher than 1200us. How high depends on your ESC/Motor/Prop/Battery setup. It basically is where neutral drag is for the spinning prop. I would guess it'll be 1300us - 1500us for most setups. This needs to be set in conjunction with ESC IDLE.

Tx/Rx THROTTLE MAX using slider, switch, stick etc should be 2000us or higher.

Just remember that it must not be lower than 1180us (again depending on jitter of Tx/Rx, I add about 25us as a buffer here), since that is where Altis sees and sets ESC off to, see below.

ESC THROTTLE OFF should be just slightly higher than 1200us (Mine is set at 1205us)

ESC THROTTLE IDLE again depends on power setup. I would guess it'll be between 1300us - 1500us for most setups but it can be in just about any range and still work since you can always program accordingly in the TX.

ESC THROTTLE MAX should be lower than TX Max, 2000us or less (Mine is set at 1980us).

Ok, now that we have an idea what needs to done, we can setup the TX/RX accordingly. There are tools out there that will read the signals, as well as some TX's have the feature built in. But we can also use an Arduino Nano to help us find that PWM signal. Programming an Arduino NANO to read PWM from a plane's RX



I bought the Nanos without headers and soldered the female end of a servo extension to the board. This connection goes into the throttle channel of the receiver to read its PWM signal. Connections are:

- Servo Ground to 1 of the ground pads on the Nano
- Signal (orange wire) to D2 pad
- Positive to +5v on Nano Pad

## Preparing the NANO for programming BLH ESCs



The male end will be used to program the BLH ESC's later on. Make these connections:

- Ground (brown wire) soldered to another GND pad of the Nano.
- Signal to D3 pad on the NANO
- DO NOT CONNECT the positive wire (you can see I cut that)

Finish with some heat shrink to prevent shorts.

Now we need to prep the software:

Nano Driver link:

https://drive.google.com/file/d/1oDpJ2ZBGofbW4FNgFWabktUkfJhaJLi\_/view? usp=sharing

Arduino App link: https://downloads.arduino.cc/arduino-1.8.15-windows.exe Download and Install. Then Open the App. We get this default screen:



Go to Tools and set three things there:

- 1. Select Board = Arduino Nano
- 2. Processor = ATMega328P (Old Bootloader)
- 3. Port: Choose Com port corresponding to what is shown in your device manager. In Ports (COM & LPT), look for USB-SERIAL CH340, it'll show the COM #

Then copy this code:

```
double channel;
void setup() {
pinMode(2, INPUT);
Serial.begin(9600);}
void loop() {
```

### channel = pulseIn(2, HIGH); Serial.println(channel);}



Then highlight the above default code in yellow and paste the copied code into box overwriting the default lines.



Click the check circle in white above. This Opens a box to Save. Name it Pwm Reader or whatever.

It'll then verify and compile.



Then plug the Nano into your computer using a USB cable.

Click the right arrow highlighted in white above to load the code into the Nano.

Give it a few seconds. When it's done uploading the code (sketch).

Plug NANO into RX channel you want to monitor.

Select Tools, Serial Monitor

You should now see the PWM signal value in microseconds being displayed (as in the example screen below):

COM3		-		×
1				Send
18:25:13.130 -> 1094.00				
18:25:13.130 -> 1094.00				1
18:25:13.130 -> 1094.00				
18:25:13.184 -> 1094.00				
18:25:13.184 -> 1094.00				
18:25:13.184 -> 1094.00				
18:25:13.231 -> 1094.00				
18:25:13.231 -> 1088.00				
18:25:13.231 -> 1094.00				
18:25:13.231 -> 1094.00				
18:25:13.284 -> 1094.00				
18:25:13.284 -> 1095.00				
18:25:13.284 -> 1094.00				
18:25:13.331 -> 1095.00				
18:25:13.331 -> 1095.00				
18:25:13.331 -> 1095.00				
18:25:13.384 -> 1094.00				
18:25:13.384 -> 1094.00				
18:25:13.384 -> 1088.00				
18:25:13.384 -> 1094.00				
18:25:13.431 -> 1094.00				
18:25:13.431 -> 1094.00				
18:25:13.431 -> 1094.00				
18:25:13.485 -> 1095.00				
18:25:13.485 -> 1094.00				
18:25:13.485 -> 1095.00				
18:25:13.532 -> 1094.00				
18:25:13.532 -> 1094.00				
18:25:13.532 -> 1094.00				
18:25:13.532 -> 1095.00				
18:25:13.585 -> 1094.00				
18:25:13.585 -> 1094.00				
18:25:13.585 -> 1089.00				
18:25:13.632 -> 1094.00				
18:25:13.632 -> 1094.00				
18:25:13.632 -> 1094.00				
18:25:13.685 -> 1094.00				
18:25:13.685 -> 1095.00				
18:25:13.685 -> 1094.00				
18:25:13.685 -> 1094.00				
18:25:13.732 -> 1094.00				
18:25:13.732 -> 1094.00				1
18:25:13.732 -> 1094.00				
Autoscroll Show timestamp	Newline V	9600 baud 🗸	Clear	output

Please Note: the displayed PWM value could have errors between 1-10us. For our purposes this doesn't matter much.

Now you can set the throttle channel endpoints for your arming switch and whatever control for variable throttle you are using per Figure 1 above. Keep in mind the variable throttle control should not turn off the motor at its lowest setting.

If you want to monitor other channels on the RX, just plug servo connector into the corresponding channel. You will need to close and restart the serial monitor screen to see the new results.

# Programming a BLH ESC

Download this App:

https://drive.google.com/file/d/1sj4R5AhuPm7YktLXzl2rvbQn4gocl5bm/view? usp=sharing

BLHeliSuite32 32.8.0.7 [4way-if @COM3]			-	×
ESC setup Select BLHeli_32 Interface Options ? BLHeli_32 info	Save Screenshot	Styles		 
ESC Setup ESC overview ESC Flash	Make interfaces			
Make Arduino Interface Boards				
Arduino Board:				
Nano w/ ATmega328 (old Bootloader) 🛛 🗸				
Baud:				
\$7600 ~				
Arduino BLHeli Bootloader     1 C				
O Arduino 4way-interface K 2 D B 3				
And ins Illine Name 228 hav				
ALOUID TAME TABLO 250 LIEX				
Arduino BLHeli Bootloader				
Port: COM3 USB-SERIAL CH340	$\sim$			

Now execute these steps:

- Select 4way-if under Interface Tab Hi-Lite Blue in Picture
- Select Make interfaces Tab
- Select Nano Board with Old Bootloader
- Baud Rate 57600
- Click Arduino BLHeli Bootloader

# Then the program will ask:



Hit Yes to Confirm.

#### Result:

Information	tion X
6	Arduino flashed sucsessfully.
•	Please select "BLHeli32 Bootloader (USB/Com)" Interface.
	ОК
¥	

Hit Ok.

# Configuring the BLHeli ESC

<u>C</u> setup	Suite32 32. S <u>e</u> lect BL	8.0.7 [USB/Com @ Heli_32 Interface	COM3] Options	? <u>B</u> LHeli_32 info	S <u>a</u> ve Screenshot	<u>S</u> tyles	_	>
ESC S	etup	ESC overview	,	ESC Flash	Make interfaces			
Make Ardu	uno Interfac	e Boards						
Arduino B	oard:							
Nano w/	ATmega328	(old Bootloader)		~				
Baud:								
57600		~						
			1.0					
Arduin	o BLHeli Boo	tioader	LU					
		SPIN .						
and the second	100	and the second se						
and the second		· · · · · · · · · · · · · · · · · · ·						
) Arduin	o 4way-inte	rface K	2 D B	3				
Arduino 1	Wire_Nano	_328.hex		~				
				and a second				
Ardu	uino BLHeli E	lootloader						
et [00]	M21108.000	TAL CH240						

Back to Select Interface in blue above, choose USB.

BLHeliSuite32 32.8.0.7 [USB/Com @COM3] ESC setup: Select RI Heli 32 Interface: Ontions: 7 RI Heli 32 info: Save Screenshot: Styles	-	×
ESC Setup ESC overview ESC Flash Make interfaces		
Make Arduino Interface Boards		
Arduino Board:		
Nano w/ ATmega328 (old Bootloader) 🗸 🗸		
Baud:		
57600 ~		
Arduino BLHeli Bootloader     ILC		
Arduino 4way-interface K 2 D B 3		
		1
Arduno1Wire_Nano_328.hex		
Arduino BLHeli Bootloader		
Port: COM3 USB-SERIAL CH340		

Click ESC Setup in blue above

SC 1 Unkn	esc ove	erview	ESC Flas	h	Make interfa	ces					
Name		Info for ?? BLHeli	? Motors 32 Revisior	n: xx.x	Misc	ttle Cal En	able				
Rampup Power		Motor I	Direction		Minimu	m Throttle		Star	tup Beep \	/olume	
(	>	<	Horman	>	<	1010	>	<		;	•
Temperature Pro 140 C	otection	Demag	Compensat Low	tion	Maximu	m Throttl 1960	e	Bead	on/Signal 80	Volume	
c 🔛	>	<		>	<	1	>	<	101	2	,
Low RPM Powe On	r Protect	Motor	iming 16 deg		Center	Throttle 1500		Bead	on Delay 10:00 m	in	
sine Modulation Off	Mode	<	m Accelera 1aximum	tion	<	n Stop Off	>	PWN	1 Frequenc 24 kHz	y Low	
Auto Telemetry Off		Stall Pr	otection Normal		SBUS C	hannel Off		PWN	1 Frequenc	y High	
C	>	<		>	<		>	<		,	•
								Mus	ic Note Cor Music O	nfig ff	
								1500	Music Edi	tor	
								-			
Read Setup	Write :	Setup	Flash BLHe		erify BLHeli	]					

Make Sure Port Matches, (Blue Hi-Lite) Connect Speed Control to Nano via servo wire, then Click Read Setup.

### This Screen Pops up



### Connect Battery to ESC and wait a few seconds



### Click Ok

You can now adjust each of the parameters.

Ampup Power     Motor Direction     Immum Throttle     Startup Beep Volume       100 %        40       >          40       >          8       ampup Power     Immum Throttle     1205     40        >             ampup Power     Demag Compensation     Maximum Throttle     Beacon/Signal Volume       140 C     Low	
Image: Second	
Image: Compensation     Demage: Compensation     Maximum Throttle     Beacon/Signal Volume       140 C     10w     1980     180	
Image: Source of the source	
Notor Timing     Center Throttle     Beacon Delay       On     Auto     1500     12:44 min       C     >     C     >	
On         Auto         1500         12/44 min           <         >         <         >         <         >	
ow Voltage Protection Maximum Acceleration Brake On Stop PWM Frequency	
3.40 V Maximum 65 % 48 kHz	
Current Protection Current Sense Calibration Non Damped Mode Music Note Config	
A state of the	
Sine Modulation Mode Auto Telemetry Stall Protection	
Off Off Normal	
· · · · · · · · · · · · · · · · · · ·	

These are some of my settings

**NOTE!** Make sure **Throttle Cal Enable** above Minimum throttle setting is UNCHECKED! Click Write Setup when you want to transfer settings to ESC

This Pops Up



Click OK, mine says nothing to write because it already is setup this way.

	for Multicopter Motors BLHeli 32 Revision: 32 7	🗆 Throttla Cal Enable 🖉	Off Off On
Rampup Power 100 %	Motor Direction Reversed	Minimum Throttle	Startup Beep Volume 40
Content of the second s	Compensation Low Compensation Low Compensation Low Compensation Low Compensation Low Compensation Low Compensation Low Compensation Low	< > S Maximum Throttle 1980 < > S Center Throttle 1500	Beacon/Signal Volume 180 Seacon Delay 12:44 min
Cow Voltage Protection 3.40 V	Maximum Acceleration Maximum < >	Stake On Stop     65 %	< >
Off >	Current Sense Calibration +/- 0%	Non Damped Mode Off Control Of	Music Note Config Music Off Music Editor
Sine Modulation Mode Off	Auto Telemetry Off Control of the second sec	Stall Protection	

Click Read Setup again to verify settings have been transferred

Done Programming BLH ESC

Using BLH with Android

Search for BLHELI\_32 in the play store, and load it on an Android device.

You will need a USB cable that goes from the phone to the Nano. Launch the app and plug in the Nano



Tap on Scan USB



Tap Connect

10:32 #	<u>"</u>	* <u>*</u> * all all 4	8%	
BLHel	i_32 Settings	i		:
	Connected ESC	s:		
	ESC 1: Unknow	'n		
	ESC 2: Unknow	'n		
	ESC 3: Unknow	'n		
	ESC 4: Unknow	'n		
	Motor Timing: 1	6 deg		
<	<del>.</del>	•	>	
	Pwm Frequency	Low: 24 kHz		
<			$\rightarrow$	•
	Pwm Frequency	High: 24 kHz		
<	•		>	
	Rampup Power:	50 %		
<			>	•
BLH	eli_32			
Allow	BLHeli_32 to a	iccess USB S	Serial?	
	Cancel		ок	
	111	0	<	

Allow

10:32 🦀 🕰	<b>∜</b> .ıll .ıll 48%	-
BLHeli_32 S	Settings	1
Conne	cted ESCs:	
ESC 1:	TEKKO32_F3_Metal_B Rev32.7	
ESC 2:	Unknown	
ESC 3:	Unknown	
ESC 4:	Unknown	
Motor	Timing: Auto	
< •		- >
Pwm F	Frequency: 48 kHz	-• >
Pwm F Not su	Frequency High: apported	
Rampu	up Power: 100 %	->
Demag	g Compensation: Low	- >
USB Device: S	Single ESC	*
BLE Device: S	Scan Bluetooth	*
Disconnect	Read Writ	e
111	0	<

Now you can do the same Read/Write functions.