Operation manual for BLHeli_32 ARM

BLHeli_32 firmware is the third generation BLHeli, following base BLHeli and BLHeli_S.

BLHeli_32 is designed for superior functionality and performance, primarily in multirotors and runs on ARM 32bit MCUs.

All codes implement damped light mode as default.
Damped light does regenerative braking, causing very fast motor retardation, and inherently also does active freewheeling.

The code supports features to prevent sync loss. There are tuneable parameters that can make the code run well even in the most demanding situations, although default settings will work excellently in normal operating environments.

The code supports regular 1-2ms pulse width input, as well as Oneshot125 (125-250us), Oneshot42 (41.7-83.3us) and Multshot (5-25us). Dshot is supported at any rate up to at least Dshot1200. The input signal is automatically detected by the ESC upon power up.

The code also supports a beacon functionality, where the ESC will start beeping after a given time of zero throttle. This can be very useful for finding lost crafts.

Programming parameters:

Rampup Power:
Rampup power can be set to relative values from 3% to 150%. This is the maximum power that is allowed when ramping up at low rpms and during startup. For low rpms, the maximum power to the motor is limited, in order to facilitate detection of low BEMF voltages.

Rampup power also affects bidirectional operation, as the parameter is used to limit the power applied during direction reversal.

During startup, the actual applied power depends on throttle input, and can be lower than the maximum level set by the rampup power parameter, but the minimum level is a quarter of the maximum level.

Motor Timing:
Motor timing can be set between approximately 1^0 and approximately 31^0 in approximately 1^0 increments (actual accurate values here are 15/16ths of a degree).

Typically a medium setting will work fine, but if the motor stutters it can be beneficial to increase timing. Some motors with high inductance can have a very long commutation demagnetization time. This can result in motor stop or stutter upon quick throttle increase, particularly when running at a low rpm. Setting timing higher will allow more time for demagnetization, and often helps.

This parameter can also be set to auto. I this case the code monitors demagnetization time, and keeps timing as low as possible without having issues with demag. On well behaved motors, timing can be low in the entire power range, and thereby max power can be reduced. On not so well behaved motors, timing is increased as needed, and thereby improves margins against sync loss.

PWM frequency:
Motor PWM frequency can be programmed in a range that is preconfigured by the ESC manufacturer.

Code revisions from Rev32.8 and on support variable pwm frequency.
Then you can configure motor pwm frequency to increase with increasing throttle.
The benefits of variable pwm frequency are:
- Low frequency for low throttle gives good active braking where it is most needed
- High frequency for higher throttle makes running smoother
- The variable frequency will work as dithering making running even more smooth

Code revisions from Rev32.9 and on support variable pwm frequency where the pwm frequency is controlled by motor RPM. This mode is called “By RPM”.
This mode can be invoked by setting pwm high frequency to maximum.
By letting pwm frequency be controlled by RPM instead of throttle, some artifacts that can arise from throttle control can be alleviated.

Demag Compensation:
Demag compensation is a feature to protect from motor stalls caused by long winding demagnetization time after commutation. The typical symptom is motor stop or stutter upon quick throttle increase, particularly when running at a low rpm. As mentioned above, setting high commutation timing normally helps, but at the cost of efficiency.

Demag compensation is an alternative way of combating the issue. First of all, it detects when a demag situation occurs.
- In this situation, there is no info on motor timing, and commutation proceeds blindly with a predicted timing.
- In addition to this, motor power is cut off some time before the next commutation.

A metric is calculated that indicates how severe the demag situation is. The more severe the situation, the more power is cut off.
When demag compensation is set to off, power is never cut.
When setting it to low or high, power is cut. For a high setting, power is cut more aggressively.
Code revisions from Rev32.9 and on also support a setting called “Very High”, for which power is cut even more aggressively.

Generally, a higher value of the compensation parameter gives better protection.
If demag compensation is set too high, maximum power can be somewhat reduced for some motors.

Sine Modulation Mode:
Sine modulation mode can give a few percent more efficient running, as well as smoother running.

It is a pretty subtle effect. Power is modulated with a sine shape, following the top of a sine wave through the commutation cycle so that the power when commutating shall be ideal for a motor with sine shaped BEMF. Power is varied between sin(60deg)=87% when commutating to sin(90deg)=100% in the middle of a commutation cycle, and then down again to sin(120deg)= 87% at the end of the commutation cycle.

Maximum power is the same for sine mode as for regular mode, as when approaching full power the ESC will transition smoothly into regular mode.

Varying power can only be done by varying pwm, so a high pwm frequency is preferable for accurate sine mode operation. Still, even with 48kHz pwm frequency, a reasonable accuracy of the sine modulation can only be achieved up to some 100k erpm. At higher erpms, the motor still runs fine, but the quality of the sine modulation is degraded.

Due to the increased MCU processing for sine mode, max erpms is lower for sine mode than for regular mode. Still it will generally be more than 300k erpm even for sine mode.

Sine mode is implemented from Rev32.6.

Note that if sine mode is chosen, then variable pwm frequency is disabled.

Maximum Acceleration:
Maximum acceleration can be set between 0.1%/ms and 25.5%/ms. It can also be set to maximum, in which case acceleration is not limited. Limiting acceleration is primarily intended as a backup parameter that can be used in cases where too hard acceleration gives desyncs.

When setting to e.g. 10%/ms, it means that the power applied to the motor is not allowed to increase by more than 10% per millisecond.

Motor Direction:
Motor direction can be set to fwd, rev, bidirectional 3D, bidirectional 3D rev, bidirectional soft and bidirectional soft rev. In bidirectional mode, center throttle is zero and above is fwd rotation and below is reverse rotation. When bidirectional operation is selected, throttle calibration is disabled.

There are two bidirectional modes from Rev32.6, bidirectional 3D and bidirectional soft. The 3D mode applies more power when reversing direction, and also limits minimum throttle to 6%. The soft mode applies less power when reversing, and does not limit minimum throttle.

Startup Beep Volume:
Sets the volume of beeps during powerup.

Beacon/Signal Volume:
Sets the volume of beeps when beeping beacon beeps. The ESC will start beeping beacon beeps if the throttle signal has been zero for a given time. Note that setting a high volume can cause hot motors or ESCs!
Also sets the volume used for Dshot initiated signal tones.

Beacon Delay:
Beacon delay sets the delay before beacon beeping starts.

Throttle Cal Enable:
If disabled, throttle calibration is disabled.

Minimum throttle, maximum throttle and center throttle:
These settings set the throttle range of the ESC. Center throttle is only used for bidirectional operation. The values given for these settings are for a normal 1000us to 2000us input signal, and for the other input signals, the values must be scaled.

For Dshot input signal, these settings have no effect.

Temperature Protection:
Temperature protection can be enabled or disabled. And the temperature threshold can be programmed. The programmable threshold is primarily meant as a support for hardware manufacturers to use, as different hardwares can have different tolerances on the max temperatures of the various components used.

The ESC measures temperature within the MCU and limits motor power if the temperature is too high. Motor power is limited over a range:
- If the temperature is above the threshold, motor power begins to be limited.
- If the temperature is above the threshold plus approximately 15^0C, motor power is limited to 25%. Motor power is not limited below 25%.

Low RPM Power Protect:
Power limiting for low RPMs can be enabled or disabled. Disabling it can be necessary in order to achieve full power on some low kV motors running on a low supply voltage. However, disabling it increases the risk of sync loss, with the possibility of toasting motor or ESC.
Code revisions from Rev32.9 and on have a mode called "On Adaptive". This setting is intended for large low kV motors running on a fairly low battery voltage. But it can be used, and is indeed suitable for any motor kV and battery voltage. In this mode, the code calculates the kV*voltage and adjusts the low rpm power protection accordingly

Low Voltage Protection:
Low voltage protection can be set between 2.5V and 4.0V per lipo cell. Or it can be disabled. When enabled, it will limit power applied to the motor if the battery voltage drops below the programmed threshold. This feature is primarily intended for fixed wing crafts.

Current Protection:
Current protection can be enabled to limit current. If enabled, then current will be limited to maximum the programmed value. The reaction time of the current limiting is quite fast, so current will also be limited during accelerations.

The value given for current protection, is per ESC. So if setting limit to e.g. 40A for each of the ESCs in a quad (using BLHeliSuite32 or the BLHeli_32 Android app), then the total current limit for the four ESCs will be 160A.

Brake On Stop:
Brake on stop can be set between 1% and 100%, or disabled. When not disabled, the given brake force will be applied when throttle is zero. For nonzero throttle, this setting has no effect. This feature is primarily intended for fixed wing crafts with folding props.

On some ESCs this setting is not linearly programmable, it will just be enabled (at 100% force for any setting 1%-100%) or disabled (this applies to ESCs that have "EN/PWM" style fet drivers).

Auto Telemetry:
When auto telemetry is enabled, the ESC will autonomously output telemetry at 32ms intervals, regardless of whether or not there are telemetry requests from the input signal.

Autonomous telemetry is implemented from Rev32.6.

LED Control:
LEDs can be controlled on ESCs that support it. Up to 4 LEDs can be turned on or off.

Stall protection:
From revision 32.7, stall protection can be programmed to normal or relaxed. Relaxed stall protection increases the risk of damage to ESC or motor but can recover faster when props hit obstacles. For revision 32.6 and earlier, stall protection is relaxed.
Code revisions from Rev32.9 and on have a tweak to the relaxed stall protection mode, where there is no boost on startup for this mode. So if you are flying with really low throttle and the motors stop e.g. due to reverse flow, then they will just gently start up again on the low throttle.

Regenerative braking / active freewheeling:
Damped light mode is implemented by doing regenerative braking, and inherently active freewheeling is also implemented. Then losses due to braking are counteracted by the reduced losses of active freewheeling.

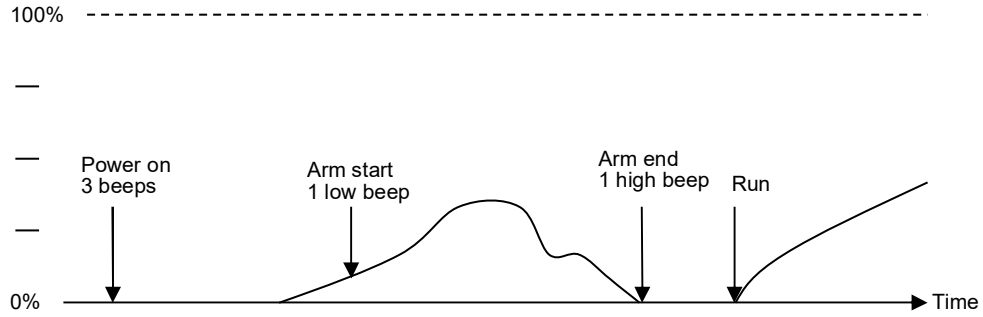
From code revision 32.4 and onwards, it is possible to select nondamped operation (for most ESCs). This will degrade performance in multirotor applications but can be desirable for fixed wing environments.

S.BUS:
From code revision 32.8 and onwards, S.BUS as input signal is supported. The S.BUS channel is selected with BLHeliSuite32 or the BLHeli_32 Android app. If a valid S.BUS channel (0 to 16) is selected, then the input signal will be interpreted as S.BUS.

S.PORT:
From code revision 32.8 and onwards, FrSky compatible S.PORT telemetry is supported. The S.PORT physical ID is selected with BLHeliSuite32 or the BLHeli_32 Android app. If a valid S.PORT physical ID (1 to 28) is selected, then the telemetry format will be S.PORT. Note that only ESCs that use USART1 (port PB6) for telemetry support S.PORT. If the "S.PORT Physical ID" programming parameter shows up in BLHeliSuite32, then your ESC supports it.

Arming sequence:

The figure below shows an example of throttle value versus time.



At power on, an activated ESC beeps 3 beeps.

When throttle signal is detected, it beeps one low tone beep. This signals that input signal is detected. Then, when or if throttle is zero, it beeps one high tone beep. This signals the end of the arming sequence, and the ESC is ready to run.

Also, if more than 50% throttle is detected at arm start, the ESC starts throttle calibration.

If the esc is armed and sees zero throttle for a given time, it beeps beacon beeps, which are approximately one beep per three seconds.

Input signal:

Available throttle calibration range is from 1000us to 2000us, and the difference between minimum and maximum throttle must be more than 140us (70us in bidirectional mode). If a calibration is done where the difference is less than 140us (70us), the maximum will be shifted so that the difference is 140us (70us).

Oneshot125 mode works just the same as regular 1-2ms mode, the only difference is that all timing is divided by 8. And the same for Oneshot42, where all timing is further divided by 3. Multishot also works similarly, except the input signal range is 5-25us.

Dshot is supported at any rate, up to at least Dshot1200. When the input signal is Dshot, throttle calibration is disabled, and the throttle calibration values are ignored. Code revisions from Rev32.10 and on support extended Dshot telemetry.

Input signal rates up to at least 32kHz are supported. But please note that higher input signal rates put a heavier load on the MCU, and will reduce the maximum erpm that the ESC can handle.

For a 48MHz clock MCU, the minimum input signal rates are about 40Hz for 1-2ms PWM, about 1000Hz for Dshot and about 300Hz for all other input signal types. For faster MCUs, these frequencies will scale up proportionally to MCU clock frequency.

Input signal statistics:

From code revision 32.4 and onwards, input signal statistics can be read out using BLHeliSuite32 or the BLHeli_32 Android app. This can be used to diagnose potential noisy input signal. The number of good frames and bad frames are reported.

Note that good frames count is limited to 32bit (4294967295 maximum), and that activation of beacon will generate bad frames.

Telemetry:

From code revision 32.1 and onwards, telemetry is supported. Telemetry is designed to be compatible with the specifications from KISS 24A, and delivers the following data:
- Temperature [^{0}C]
- Voltage [V]
- Current [A]
- Temperature [Ah]
- Rotation speed [electrical rpm]

Temperatures below 0^0C are not supported, they will be shown as 0^0C.
For conversion from electrical rpm to mechanical rpm, divide by (motor poles)/2.

Note that rotation speed measurements are erroneous below 1000 electrical rpm for revision 32.6 and earlier.

Maximum supported speeds:

For a 48MHz ST MCU, the maximum speeds are approximately:

- Multishot at 8kHz: 510k erpm
- Multishot at 16kHz: 450k erpm
- Multishot at 32kHz: 420k erpm

- Dshot at 8kHz: 470k erpm
- Dshot at 16kHz: 420k erpm
- Dshot at 32kHz: 310k erpm

- Dshot at 16kHz with sine: 280k erpm

For a 72MHz GigaDevice MCU, these speeds are approximately doubled (increased clock speed and no flash memory wait states).

A 14 pole 2300kv motor on 4S will theoretically run up to (14/2)*2300*4*4.2 erpm = 270k erpm

Beeps - Normal operation:

Power up:

Once

Throttle signal detected (arming sequence start):

Once

Zero throttle detected (arming sequence end):

Once

After this, the motor will run.

Beeps - Throttle calibration:

Power up:

Once

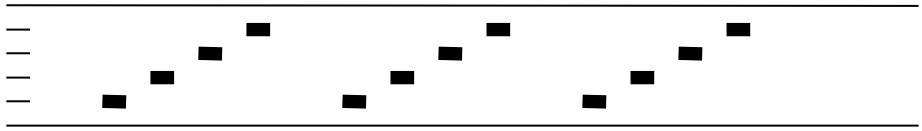
Throttle signal detected (arming sequence start):

Once

When throttle is above midstick (measuring max throttle):

While measuring

If throttle is above midstick for 3 seconds:



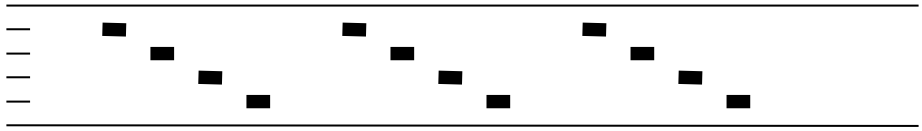
Once

This beep sequence indicates that max throttle has been stored

When throttle is below midstick (measuring min throttle):

While measuring

If throttle is below midstick for 3 seconds:



Once

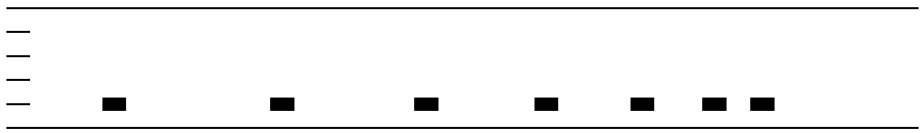
This beep sequence indicates that min throttle has been stored

At this point throttle calibration values are stored. You may remove power from the ESC, or just continue running your ESC.

Please note that for some ESCs, throttle calibration beeps are different from the above. If you are in doubt, consult the manual of your specific ESC.

Beeps - Not activated ESC:

All ESCs shall be activated during manufacturing.
If for some reason this is not done, the ESC will beep like this upon powerup, before the normal operation beep sequence starts:

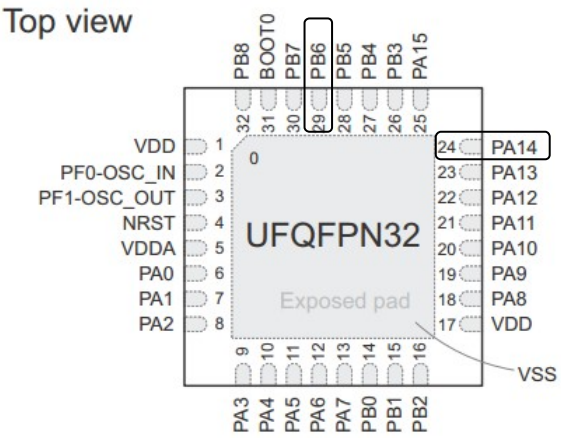


Beeps - Activation failed ESC:

All ESCs shall be activated during manufacturing.
If for some reason activation has failed and the ESC is not regarded as a valid BLHeli_32 unit, the ESC will beep like this upon powerup, before the normal operation beep sequence starts:

In this case the ESC will only accept 1-2ms pwm input signal.

Telemetry port:



Telemetry port versus firmware version is given below:

Airbot_Wraith32_Plus_ST	PB6
Airbot_Wraith32_ST	PB6
Airbot_Wraith32_Mini_ST	PB6
X_Racer_35A	PB6
X_Racer_25A	PB6
Betaflight_ESC_BL32_35A	PB6
Mantis_ST	PA14
BLARM_HK_6530	PA14
Siskin_ST	PA14
FVT_Littlebee_Summer_30A	PB6
FVT_Littlebee_Summer_35A	PA14
Emax_Formula_45A	PA14
Aikon_AK32_35A	PB6
KS_BLHeli_32_30A	PB6
Spedix_GS30A_V1_1	PB6
Spedix_GS35A_V1_1	PB6
Hobbywing_XRotor_BLHeli32	PB6
iFlight_Force32	PA14 (PB6 for Rev32.1)
DYS_Aria_32	PB6
Siskin_ST_32_Plus	PB6
Siskin_GD_32_Plus	PB6
FVT_Littlebee_Summer_25A	PB6
T_Motor_F30A_BLHeli32_V1_1	PB6
T_Motor_F35A_BLHeli32_V1_1	PB6
Typhoon32	PA14
Ori32	PB6
Spedix_GS35A_4IN1_4S_V1_1	PB6
Spedix_GS35A_4IN1_6S_V1_1	PB6
Flycolor_X_Cross_BL_32	PA14
Spedix_GS20A_4IN1_4S_V1_1	PB6
Gemfan_Maverick	PB6
AGF_BLHeli_32	PB6
TYPHEERX	PB6
FVT_CloudPhoenix_12AX4	PB6
Aikon_AK32_4S_V1_0	PB6
NOX_ESC	PB6
DALRC_ENGINE_40A	PB6
FVT_CloudPhoenix_35A	PB6
MARS32_35A	PB6
HGLRC_T-REX_60A_ST	PB6
HGLRC_DinoShot_40A_ST	PB6
Flycolor_X_Cross_BL_32_35A	PA14
Exuav_FishDrone	PB6
Aikon_AK32_4IN1_35A_6S_V1_0	PB6
YGRC_32	PB6
RobotDOG_ST	PB6
MARS32_40A	PB6
HAKRC_30A	PB6
HAKRC_35A	PB6
KS_BLHeli_32_40A	PB6
Airbot_Wraith32_Metal_ST	PB6
RF1.h	PA14
JHE_Aria_32	PB6
Siskin_Lite	PB6
Tmotor_32Bit	PA14
FrESC_80A	PB6
KS_40A_4IN1_4S	PB6

HGLRC_DinoShot_60A_ST	PB6
FrESC_35A_32	PB6
HGLRC_T-REX_35A_ST	PB6
KS_BLHeli_32_35A_V1	PB6
KS_BLHeli_32_50A_V1	PB6
FVT_CloudPhoenix_50A	PB6
Spedix_GS40A_4IN1_V1_1	PB6
GetFPV	PA14
Furling32	PA14
HAKRC_E45A	PB6
HAKRC_E50A	PB6
Airbot_Wraith32_Metal_Rev1_ST	PB6
GEPRC_BL32_PRO	PB6
GEPRC_BL32_4IN1	PB6
iFlight_BL32_PRO	PB6
iFlight_BL32_4IN1	PB6
Bardwell32	PA14
Makerfire_30A_BLHeli_32	PB6
Crazepony_30A_BLHeli_32	PB6
Dake_30A_BLHeli_32	PB6
Emax_6S_STSPIN32F0_V21	PB6
Atom32_35A	PA14
ALIENTEK_BLHeli32	PB6
FVT_MachineDog_20A	PB6
HTIRC_Hummingbird_32	PB6
Skyzone_BL32_40A	PB6
DALRC_Rocket_ESC	PB6
R_FlyFort	PA14
Furling32_Mini	PB6
Furling32_	PB6
Furling32_4in1	PB6
TEKKO32	PB6
TEKKO32_F3	PB6
TEKKO32_F3_Metal	PB6
Lumenier_Razor32	PB6
T_Motor_F_4IN1_F3	PB6
T_Motor_F55A_4IN1_PRO	PB6
RDQ_32_	PB6
HGLRC_Forward_ESC	PB6
TEKKO32_F3_Mini	PB6
AK32PIN_4IN1_6S_25A	PB6
AK32PIN_4IN1_6S_35A	PB6
Lumenier_Razor_Pro_F3	PB6
DALRC_ENGINE_PRO_ESC	PB6
TEKKO32_F3_4in1	PB6
TTTRC_BL32	PB6
ReadyToSky_32	PB6
BETAFPV-16A-BLHeli_32	PB6
MAMBA_25A	PB6
MAMBA_506	PB6
Crazepony_35A_BLHeli_32	PB6
Crazepony_45A_BLHeli_32	PB6
Dake_35A_BLHeli_32	PB6
Dake_45A_BLHeli_32	PB6
Makerfire_35A_BLHeli_32	PB6
Makerfire_45A_BLHeli_32	PB6
SMOOV_30A	PB6
Furling32_4in1_F0	PB6
HSKRC_35A_BLHeli_32	PB6
iFlight-35A-BLHeli_32	PB6
Neuron_40	PB6
Aikon_AK32_4IN1_55A_6S_V1_0	PB6

PYRO-32F3	PB6
VIVAFPV_45A_BL32	PA14
Aikon_AK32_4IN1_45A_6S_V1_0	PB6
Spedix_GS45A_4IN1_V1_1	PB6
Spedix_GS55A_4IN1_V1_1	PB6
T_Motor_F45A_4IN1_V2	PB6
T_Motor_F55A_4IN1_PRO_V2	PB6
Spedix_Mini_GS40_4IN1	PB6
XILO_ESC	PB6
TEKKO32_4in1	PA14
FLYWOO_INI32_PRO	PB6
BETAFPV-16A-32bit-V1_1	PB6
MAMBA_306	PB6
T_Motor_F55A_4IN1_PRO_V2_F3	PB6
MARS32_100A	PB6
Aikon_RD 32_45A_4IN1_6S_V1_0	PB6
Spedix_Mini_GS_40F_4IN1	PB6
BETAFPV-16A-32bit-V2_0	PB6
CLRACING_STSPINA	PB6
FL1_F3	PB6
LBEES_43	PB6
SkyStars-40A-Slim-F3	PB6
RUSH_MATRIX_30A	PB6
HGLRC_T-REX_32bit_ESC	PB6
DRL_4FL_ST	PB6
TRANSTECN_50A	PA14
HAKRC_603	PB6
MAMBA_F60_PRO	PB6
Infinity_40A	PB6
HellCat32-1_0	PB6
Forward_L4_ESC	PB6
Spedix_LS40A_Slim_V1_1	PB6
RCTimer_Ares_60A	PB6
WINGTRA	PB6
FL1	PB6
HIFIONRC_BLHeli32	PB6
Flycolor_X-Cross_HV	PB6
FLYWOO_GOKU406S_PRO	PB6
Racerstar_TaiChi_ESC	PB6
Lumenier_Razor_4in1	PB6
ZeeZ60A	PB6
MAMBA_F50_PRO	PB6
Emax_Formula_65A	PB6
FLYWOO_GOKU32_PRO	PB6
CRATER_ESC	PA14
Furling32_A	PB6
MAMBA_F50PRO	PB6
MAMBA_F60PRO	PB6
Forward_F3_ESC	PB6
Furling32_V2_C	PB6
Helsel_EKUAT	PA14
FL1_Megabolt	PB6
FL1_Afterburner	PB6
LANRC_BLHeli_32	PB6
LBEES_43C	PB6
FL1_Megabolt_C	PB6
T_Motor_F35A_BLHeli32_V1_1_S	PB6
TMOTOR_XCLASS_ESC_HV	PB6
MAMBA_F75_ARRAY_PRO	PB6
FL1_Megabolt_D	PB6
FL1_Megabolt_E	PB6
MARS32_25A	PB6

Razor4in1_B	PB6
Furling32_4in1_B	PB6
BETAFPV_35A_BLHeli_32	PB6
TEKKO32_F3_4in1_B	PB6
AK32PIN_4IN1_6S_35A_V2	PB6
T_Motor_F30A_4IN1	PB6
Pro-Tronic_BF32_18A	PB6
Pro-Tronic_BF32_23A	PB6
Aocoda	PB6
MAMBA_F40_MINI_PRO	PB6
TEKKO32_F3_Metal_B	PB6
IFLIGHT_8S_BLHeli32	PB6
BETAFPV_12A_BLHeli_32	PB6
SKYSTARS_KRAMAM32_60A_6s	PB6
Aikon_AK32_4IN1_55A_6S_V3_0.	PB6
T_Motor_F45A_4IN1_Mini	PB6
RUSH_BLADE_SPORT	PB6
RUSH_BLADE_SPEED	PB6
RUSH_BLADE_SUPER	PB6
DARWIN_F0_40	PB6
TEKKO32_F3_B	PB6
AIKON_AK32PRO_50A_4IN1_MINI_6S	PB6
Furling32_4in1_E	PB6
Aikon_AK32_4IN1_35A_6S_V3_0	PB6
Tekko32_F3_C	PB6
Furling32_4in1_C	PB6
Aocoda_32Bit_60A	PB6
Aocoda_32Bit_35A	PB6
Sunrise_G071_ST	PB6
ST_F0_03	PB6
ST_L4_02	PB6
TMOTOR_F0_01	PB6
TMOTOR_L4_01	PB6
SKYSTARS_KRAMAM32_35A_6s	PB6
SKYSTARS_KRAMAM32_40A_6s	PB6
AK32PIN_4IN1_6S_25A_V2	PB6
SKYSTARS_KRAMAM32_55A_6s	PB6
IFLIGHT_X-CLASS_12S	PB6
XILO_ESC_B	PB6
HGLRC_Zeus_ESC	PB6
Eachine_ESC	PB6
Eagle32	PB6
SKYSTARS_KRAMAM32_45A_6s	PB6
WUDI	PB6
CLRACING_STSPINA_V2	PB6
LAL5_BLHeli_32	PB6
RUSH_BLADE-SPORT	PB6

Programmable brake force and nondamped mode:

Programmable brake force and nondamped mode (from code revision 32.4) is supported by most ESCs. But some ESCs have deadtime controlled by the driver, and for these ESCs programmable brake force and nondamped mode is not supported:

BLARM_HK_6530
Gemfan_Maverick

Revision history:

- Rev32.0 Started
- Rev32.1 Added telemetry
  Tuned gain of current sensor
- Rev32.2 Withdrawn
- Rev32.3 Added support for Dshot programming commands
  Greatly improved reliability of bidirectional direction reversals
  Improved reliability of startup (lower probability of stuttering)
- Rev32.4 Added programmable current sensor calibration
  Added programmable nondamped mode
  Added support for ProShot input signal
  Added input signal status reporting
  Added programmable startup music
  Changed default throttle range to 1040-1960
  Some smaller fixes
- Rev32.5 Withdrawn
- Rev32.6 Added programmable sine modulation mode
  Added programmable soft bidirectional mode
  Added programmable autonomous periodic (every 32ms) telemetry mode
  Activated hardware noise filter on the signal input
  Fixed issue of corrupted telemetry CRC, and speeded up generation of CRC
  Fixed issue in bidirectional mode where motor would twitch before stopping
  Improved input signal detection, particularly for Proshot and also for Dshot at high input signal frequency
  Some smaller fixes
- Rev32.7 Added real time signal line telemetry (for Dshot and Proshot input)
  Added telemetry trig for PWM input (pulse <30us for 1-2ms PWM, OS125 and OS42)
  Added programmable stall protection
  Added capability of Dshot2400
  Some smaller fixes
- Rev32.8 Added support for S.BUS input signal and S.PORT telemetry
  Added support for variable motor pwm frequency
  Improved reliability of signal detection
  Removed support for Proshot
  Many smaller fixes
- Rev32.9 Added support for RPM controlled variable motor pwm frequency
  Added support for very high demag compensation
  Added support for adaptive low rpm power protection
  Modified relaxed stall protection mode to have no startup boost
  Greatly reduced noise level in the Dshot real time erpm data
  Improved rampup consistency, that helps make e.g. flip stops more precise
  Fixed a bug that caused MM32SPIN160 MCUs to occasionally hang
  Some smaller fixes

- Rev32.10 Added support for extended Dshot telemetry
  - Temp, volt, curr data now available over bidirectional Dshot signal

  Reduced ESC and motor stresses during stall conditions
  - Timeout before shutting off power is reduced

  Added protection against unintentional spoolups
  - Input signal is qualified before starting, and during the first 5s of running, signal timeout is only 30ms (vs generally 320ms)
  - Input signal loss does not lead to new detection of signal type (unless input line is held high more than 2 seconds so bootloader is entered)
  - So when changing input signal type, the ESC must be power cycled for it to take effect

  Some small changes to make thrust vs rpm more linear
  - Primarily for static or quasi-static load conditions

  Improved and now functional temperature protection for AT MCUs

  Some smaller fixes

Errata:

- Rev32.0:
Setting temperature protection to off causes maximum power to be very limited.
Direction reversals in bidirectional mode are not reliable.

- Rev32.1:
Setting temperature protection to off causes maximum power to be very limited.
Direction reversals in bidirectional mode are not reliable.
Stalled motor protection does not work as intended, starting is attempted indefinitely, even if motor is stalled.

- Rev32.2:
Withdrawn as in some cases it could beep very loud when disconnecting after a flash to this revision.

- Rev32.3:
Dshot/Proshot save settings command (command no 12) does not always work.
The telemetry reported erpm can have some percent error.

- Rev32.4:
Telemetry data has a high CRC error rate.
Motor can twitch when stopping in bidirectional mode.
Proshot sometimes does not detect input signal, particularly at high input signal frequencies.

- Rev32.5:
Withdrawn due to random failures during flashing and setting changes.

- Rev32.6:
LEDs do not work on GetFPV and Furling32 codes.

- Rev32.8:
Noise spikes in bidirectional Dshot erpm data.
Bidirectional mode does not work for SBUS input signal

- Rev32.8:
Temperature protection and telemetry does not work for AT MCUs without external NTC resistor