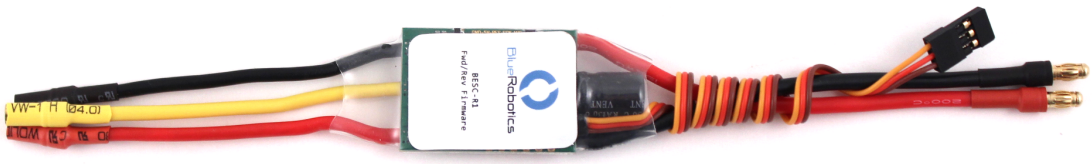
Basic ESC Documentation



Introduction

The Basic ESC is a simple speed controller. It is the AfroESC 30A pre-programmed with custom firmware that allows forward and backward operation. The firmware is open source and available in our branch of the tgy project (http://github.com/bluerobotics/tgy).

You are welcome to purchase the ESCs directly and reprogram them yourself. We offer them as a convenience to help you get started quickly. We also do quality control to ensure that every ESC operates correctly.

Note that the BasicESC no longer includes a BEC, and may only include a signal and ground wire.

Safety

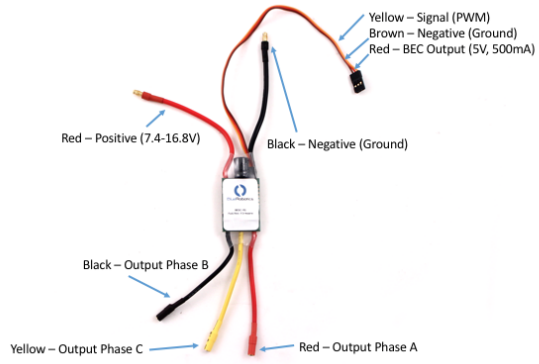
⚠ When working with electricity, especially in water, always practice caution. Always ensure that connections are secure and watertight. Keep your body away from spinning motors and propellers.

Quick Start

1. Connect the three motor wires to the motor. The order of connections does not matter, however, switching any two wires will change the direction of the motor. The output phases A, B, and C are completely interchangeable
2. Connect the red power wire and black ground wire to a power source like a battery. You will hear a few beeps from the ESC.
3. Connect the signal cable to your signal source like an RC radio receiver or microcontroller board. The yellow wire is the signal wire. The red wire is the battery eliminator circuit (BEC) output, which supplies 5V at 500mA to power a control system (no longer included). The brown wire is ground.
4. Send a stopped signal (1500 microseconds) for a few seconds to initialize the ESC. You will hear a long tone.

Specifications

Diagram



Specification Table

Electrical		
Voltage	6-16.8 volts	
Max Current	30 amps	
Physical		
Length	50 mm	2.0 in
Width	25 mm	1.0 in
Height	11 mm	0.45 in
Power Connectors	Male 3.5 mm bullet	
Motor Connectors	Female 3.5 mm bullet	

Electrical		
Signal Connector	3-pin servo connector (0.1” pitch) (ground, 5V, signal)	
Pulse Width Signal		
Signal Voltage	3.3-5 volts	
Max Update Rate	400 Hz	
Stopped	1500 microseconds	
Max forward	1900 microseconds	
Max reverse	1100 microseconds	
Signal Deadband	+/- 25 microseconds (centered around 1500 microseconds)	

3D Model

File Type	**Link**
SolidWorks Part (.sldprt)	BESC30-R1.sldprt (/besc/cad/BESC30-R1.sldprt)
STEP (.step)	BESC30-R1.step (/besc/cad/BESC30-R1.step)
IGES (.igs)	BESC30-R1.igs (/besc/cad/BESC30-R1.igs)
STL (.stl)	BESC30-R1.stl (/besc/cad/BESC30-R1.stl)

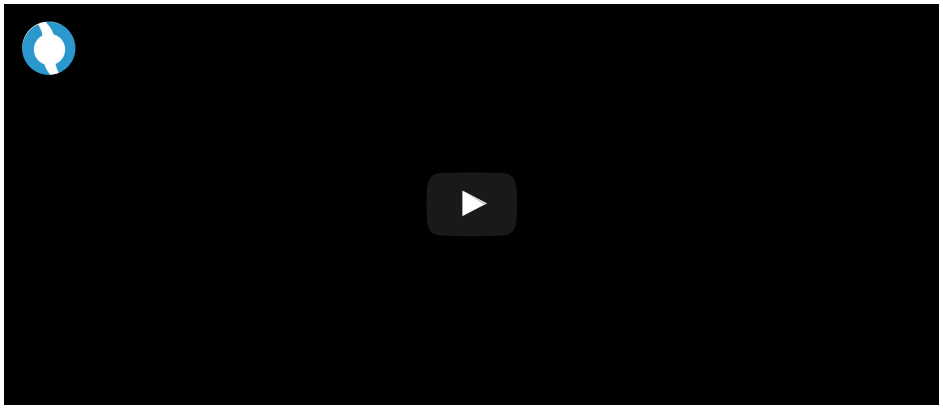
Installation

Thermal Considerations

Like all ESCs, the Basic ESC can generate a significant amount of heat when operated. It’s important to consider this when mounting and operating the ESC to ensure that it is not damaged by overheating. Most of the heat is generated in the MOSFETs (six black chips about 6mm x 6mm each) and the voltage regulator, which is on the same side of the ESC as the MOSFETs. Here are a few helpful tips:

1. If possible, make sure the side of the ESC with MOSFETs and voltage regulator is exposed to free air or a heat sink.
2. *Do not* use any adhesives that may insulate the heat generating components, such as silicone sealant.

Video Tutorial



Example Code

Arduino

This example uses the Arduino Servo library to control the speed controller. This provides an update rate of 50 Hz and can use any pin on the Arduino board as the “servoPin”.

If you’ve never used Arduino before, we suggest checking out some tutorials! (https://www.arduino.cc/en/Tutorial/HomePage)

Note: If you power the Arduino before powering the ESC, then the ESC will miss the initialization step and won’t start. Power them up at the same time, power the ESC first, or press “reset” on the Arduino after applying power to the ESC.

```
#include <Servo.h>

byte servoPin = 9;
Servo servo;

void setup() {
        servo.attach(servoPin);

        servo.writeMicroseconds(1500); // send "stop" signal to ESC.
        delay(1000); // delay to allow the ESC to recognize the stopped signal
}

void loop() {
        int signal = 1700; // Set signal value, which should be between 1100 and 1900

        servo.writeMicroseconds(signal); // Send signal to ESC.
}
```

Advanced

Firmware Files

The compiled firmware files can be downloaded here:

Basic ESC Firmware (BESC30-R1) (https://www.bluerobotics.com/downloads/besc30-r1/afro_nfet_besc30_r1.hex)

Firmware Update and Customization

The Basic ESC uses the tgy firmware (http://github.com/bluerobotics/tgy) which is open source and editable. There are many parameters that can be changed to change the performance of the speed controller.

Firmware Compilation

To compile the firmware, you'll need the avra AVR Assembler.

Mac: (Uses Homebrew)

```
brew update
brew install avra
make afro_nfet.hex
```

Firmware Flashing

The ESC includes a bootloader that allows flashing through the PWM signal wire using a programming like the Turnigy USB Linker (http://www.hobbyking.com/hobbyking/store/__10628__turnigy_usb_linker_for_aquastar_super_brain.html) or the AfroESC Programmer (http://www.hobbyking.com/hobbyking/store/__39437__afro_esc_usb_programming_tool.html).

```
avrdude -c stk500v2 -b 9600 -P [programmer port] -p m8 -U flash:w:afro_nfet_besc30_r1.hex:i
```