Dr.Edgar Huckert
Mail: edgar.huckert@huckert.com
V1.0 07-2019

Controlling a brushless motor via an ESC with Arduino

I am rather a beginner in the area of modelling. I had however some parts to start with. Controlling a normal DC motor is rather simple: you need an analog output (or PWM with simple additional components) and a current switch like a MosFet or a normal transistor or a special chip.

Things are a little bit more complicated for brushless motors. The have a minimum of three connection pins. The input current scheme is rather comparable to a 3-phase current connection. The ESCs (ESC = electronic speed control) on the market do the job for you while being controlled **via simple PWM** as in classical modelling.

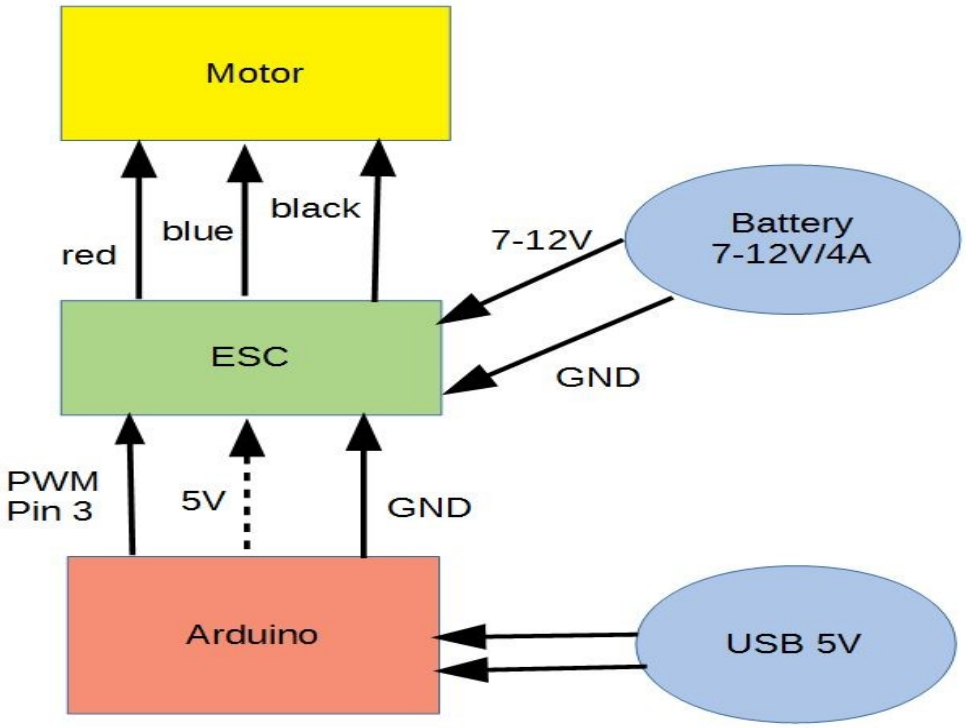
I just wanted to get the motor running and to test **speed variation**. The Arduino replaces in this configuration the remote control device. It is however not a suitable replacement for the usual remot control devices that also communicate with an ESC via PWM signals.

My **configuration**:

- **brushless motor**: Simprop Magic Torque 10-11
- **ESC** (electronic speed control): Robbe ROXXY BL-Control 810
- **Arduino**: PWM on Pin3, all classic Arduinos should work

While there a many proposals on the Internet how to connect a brushless motor directly to an Arduino using MosFets I found practically not description how an ESC works. Even the input wires were not marked in my data sheet for the ESC.This ESC accepts a PWM signal on the input and works as a motor driver on the output - no additional MosFets or normal transitors are needed.

Diagram for the configuration:



Note: the 5V connection from Arduino to ESC is not needed (dotted line). If you use other PWM pins on the Arduino be sure that they are speed-compatible with you ESC (490 or 980 Khz). The colours of the cables may be different if other components are used.

My motor started to work from a PWM value of ca. 140 onward. With a vPWM value of 200 it consumed nearly 4A under 10V (40 Watt). When I skipped the initialisation routine then the results were unpredictible. I cannot explain this as I found no clear documentation for this ESC.

If you need multiple motors or if you have to control more parameters than speed you may need multiple ESCs. A single Arduino has normally more than one PWM output pin - so the Arduino should not be the problem.

Conclusions:

- the initialisation (for the ESC) called in *setup()* seems to be necessary. If other ESCs are used the initilisation may be different.
- the motor consumes up to 4A with 10V. Take care: this is a rather small motor - the Ampère value is bigger if a bigger motor is used! Your ESC must support this.

PWM is a wide spread feature on nearly all current microprocessors (Raspberry, BeagleBone etc). The concept should thus work with nearly all of them if the sketch (the software) is adapted.

Here is the **Arduino sketch**:

```
// PWM output on Pin 3
// EH 07-2019
// The initialisation seems to be necessary
// Comsumes up to 4A with 10V
//
const int PwmPin = 3; // 3,5,6,10,11
int       outputValue = 255;
int       loopCount   = 0;
const int MAXIDX      = 12;
int       idx         = 0;
//
// table of values for initialisation
int       table1[MAXIDX] =
{ 0, 20, 40, 60, 80, 100, 120, 120, 120, 120, 120, 0 };
// table of values for test
int       table2[MAXIDX] =
{ 125, 145, 145, 150, 155, 160, 170, 180,
  190, 195, 140, 130 };

// ------------------------------------------------
// ESC initialisation: uses table1
void initMotor()
{
  Serial.println("Initializing...");
  for (int n=0; n < MAXIDX; n++)
  {
    outputValue = table1[n];
    analogWrite(PwmPin, outputValue);
    delay(300);
  }
  Serial.println("End Initialisation");
}   // end initMotor()

// ------------------------------------------------
void setup()
{
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
  idx         = 0;
  // initialise motor/ESC
  initMotor();
}   // end setup()
```

```
// ------------------------------------------------
// production" loop: uses table table2
void loop()
{
  outputValue = table2[idx++];
  if (idx >= MAXIDX) idx = 0;
  //
  Serial.print("value=");
  Serial.println(outputValue);
  Serial.print("count,value=");
  Serial.print(loopCount);
  Serial.print("  ");
  Serial.println(outputValue);
  analogWrite(PwmPin, outputValue);
  delay(800);
  loopCount++;
}   // end loop()
```