Literature Review of "VESC – Open Source ESC" project by Benjamin Vedder

Diogo Pestana

December 2023

Abstract

The VESC (Vedder Electronic Speed Controller) project is an open-source electronic speed controller for a brushless DC (BLDC) motor. The project's open-source hardware and software offer extensive customization possibilities and support various control modes, interfaces, and protection features. This versatility makes the VESC project applicable to many different contexts. In the context of a master thesis project focused on developing a BLDC motor controller, this document provides an extensive review of the VESC project, which serves as a foundational base for the future proposed solution, a modified version of the VESC project.

1 Introduction

This literature review document explores the details of the VESC project, an open-source electronic speed controller for a brushless DC motor. The purpose of this document is to provide a comprehensive overview of the VESC project, its features, and potential applications. This information will serve as an essential foundation for the master thesis project focused on developing an optimized open-source controller for BLDC motors intended for integration into robotic solutions used in agriculture.

The document is organized into several sections, each focusing on distinct aspects of the VESC project, investigating the project's hardware and software, detailing components such as the STM32F4 microcontroller, DRV8302 MOSFET driver, and the IRFS7530 MOEFETs. The document will also look into the project's various interfaces, protection features, and tools responsible for the close-loop control of the motor, its monitorization (GUI) and its adaptability for different applications.

The ultimate goal of this literature review is to provide an extensive understanding of the VESC project, its strengths, and, consequently, its potential areas of improvement. This understanding will be crucial for developing an open-source controller solution for a BLDC motor.

2 Motor Control Techniques and Algorithms

The VESC implements both sensored and sensorless FOC. Sensored FOC uses external sensors (like Hall effect sensors) for precise rotor position feedback, ideal for applications requiring high-precision control from a standstill. Sensorless FOC, while less precise at lower speeds, eliminates the need for external sensors, reducing complexity and cost. An essential feature of the VESC's FOC implementation is the auto-detection of motor parameters. This allows for a more straightforward setup process, automatically tuning the system for optimal performance for different motors.

Other Motor Control Techniques

- RPM-Based Phase Advance (Field Weakening): This technique increases the motor's top speed beyond its base speed. By advancing the current phase in relation to the motor's back EMF, higher speeds can be achieved, although at the cost of reduced efficiency and increased heat generation.

- Duty-Cycle Control: This method controls the motor by varying the duty cycle of the PWM signal.
- Speed Control and Current Control: The VESC can operate in different control modes, including speed control (controlling the motor to maintain a set speed) and current control (controlling the motor to maintain a set current), providing versatility for different use cases.

2.1 Adaptive PWM Frequency for Optimal ADC Measurements

VESC has Adaptive Pulse Width Modulation (PWM) frequency to enhance Analog-to-Digital Converter (ADC) measurements. PWM regulates power by switching the supply on/off at a specific frequency. VESC adjusts PWM based on operational conditions to minimize interference and noise in ADC measurements. Accurate ADC readings are vital for precise motor control and monitoring. Better data leads to more efficient system management, enhancing energy efficiency and reducing wear on components. This feature enables VESC to collect the best possible data from sensors, such as current and voltage measurements, which are critical for current control, voltage regulation, and system diagnostics.

3 Performance Characteristics and Limitations

3.1 Maximum Current and Voltage Ratings:

- Voltage Range: The VESC operates within a voltage range of 8V to 60V. This wide range makes it suitable for various applications, from low-voltage electronics to high-powered motors. The upper limit ensures compatibility with various battery types, including 3S to 12S LiPo batteries.
- Current Capacity: The VESC can handle up to 240A for brief periods, which is beneficial for applications requiring high start-up torque or sudden bursts of power. For continuous operation, it safely manages around 50A, depending on specific conditions like ambient temperature and system configuration.

3.2 Thermal Constraints

Thermal conditions significantly influence the VESC's performance. Continuous high-current operation generates heat, which can impact the efficiency and longevity of the system. Proper heat dissipation is crucial. Heat sinks and good air circulation around the PCB can manage heat and ensure consistent performance.

Impact of Air Circulation: Air circulation plays a critical role in cooling the system. Inadequate airflow can lead to overheating, triggering thermal protection mechanisms and reducing the continuous current capability. Additional cooling methods, such as fans or liquid cooling systems, might be necessary to maintain optimal performance in enclosed or compact spaces where air circulation is limited.

4 System Structure

The VESC project is a complex system with several integral parts that work together to control a BLDC motor. These parts can be categorized into two main sections: Hardware and Software.

4.1 Hardware

The VESC project is built around several key hardware components that work together to control a BLDC motor. Each component serves a specific function within the system and is crucial for its operation. This section overviews these components, their roles within the system, and how they interact.

The main components of the system are:

- **STM32F4 Microcontroller:** This is the central processing unit of the VESC project. It receives inputs from the user (like desired speed or direction) and processes these inputs to generate control signals. These control signals are sent to the DRV8302 MOSFET driver. The STM32F4 microcontroller is also responsible for communicating with the user interface via USB, sending data about the system's operation and receiving commands.
- **DRV8302 MOSFET Driver:** The DRV8302 is a gate driver IC that takes control signals from the STM32F4 microcontroller and uses these signals to control the operation of the IRFS7530 MOEFETs, producing gate dr. In addition to driving the MOEFETs, the DRV8302 also has built-in current shunt amplifiers that provide feedback to the microcontroller about the current flowing through the motor. This current feedback is crucial for controlling the accuracy of the motor.
- **IRFS7530 MOEFETs:** The IRFS7530 MOEFETs are the power switches of the system. They control the actual flow of power from the battery to the motor. The DRV8302 controls the operation of these MOEFETs. The gate drive signals switch the MOEFETs on and off, effectively controlling the power flow from the battery to the motor.
- **Power Supply:** The system is powered by an external source, like a battery. The high voltage from this source is connected to the DRV8302 MOSFET driver's three half-bridges. The DRV8302 includes a buck converter, which steps down this high voltage to a lower voltage needed for powering the STM32F4. This ensures that these components operate safely at their required voltage levels.

In a simplified manner, the following block diagram shows how the different components interact, focusing on the primary functions of each component.

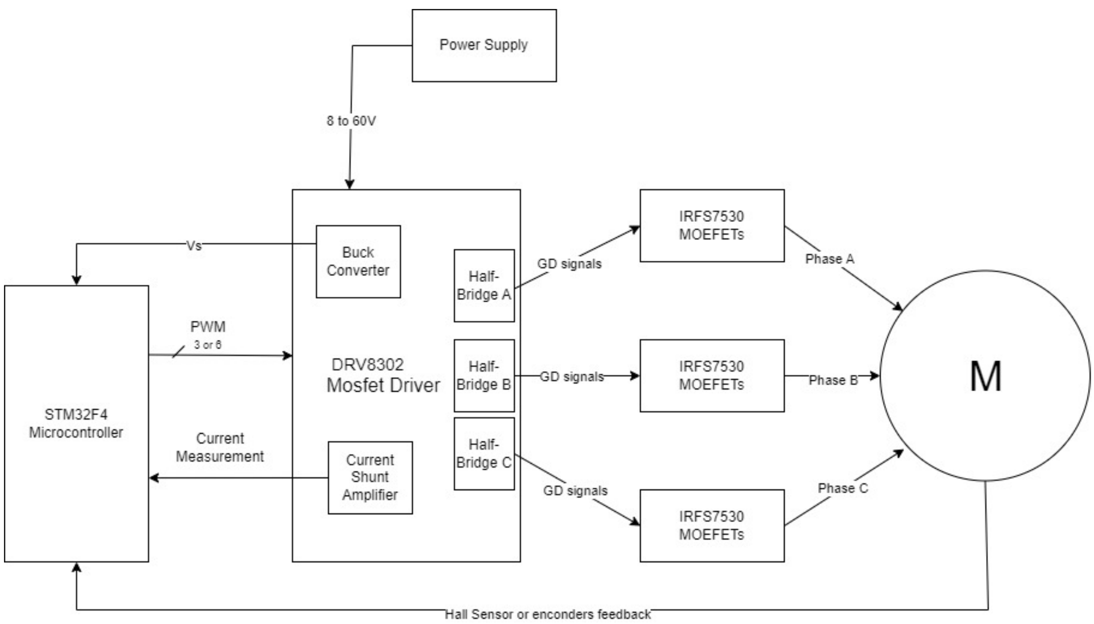

Figure 1: System's block diagram

4.1.1 STM32F40x LQFP64 Microcontroller

The STM32F4 is a high-performance microcontroller based on the ARM Cortex-M4 core. It is equipped with a Floating-Point Unit (FPU) and Digital Signal Processing (DSP) instructions, making it suitable for complex applications like motor control. The STM32F4 also offers a range of peripherals[1] including UART, SPI, I2C, GPIOs, ADCs, and timers used in the VESC project for interfacing with the DRV8302 driver, sensors, and user interface.

4.1.2 DRV8302 MOSFET Driver

The DRV8302 is a three-phase gate driver IC designed for motor drive applications. It provides three half-bridge drivers, each capable of driving two N-channel MOSFETs. It can operate off a single power supply with a wide range from 8V to 60V and supports up to 1.7A source and 2.3A peak current capability. It uses a bootstrap gate driver architecture with trickle charge circuitry to support 100% duty cycle.

The DRV8302 also includes two current shunt amplifiers for accurate current measurement. These amplifiers support bi-directional current sensing and provide an adjustable output offset up to 3V. In addition, it includes an integrated switching mode buck converter with adjustable output and switching frequency. The buck converter can provide up to 1.5A to support MCU or additional system power needs.

4.1.3 IRFS7530 MOEFETs

The IRFS7530 is a high voltage, high-speed power MOSFET designed for switch-mode power supplies and motor drive applications. It features a low gate charge and low Rds(on) to minimize conduction and switching losses. Furthermore, the IRFS7530 also offers robustness, with a fast intrinsic diode and voltage clamping.

4.1.4 Power Supply

In the VESC project, the power supply management is divided into two parts.

- **Battery or External Power Supply:** The main power source is an external battery or power supply, providing high voltage and current necessary for motor operation. This power supply is connected to the DRV8302 MOSFET driver's three half-bridges. The DRV8302 then controls the power flow to the IRFS7530 MOSFETs. Ensure the power supply voltage matches the voltage rating of the motor and VESC components, supporting 8V to 60V (suitable for 3S to 12S LiPo).

- **On-Board Power Supply:** The STM32F4 microcontroller require lower operating voltages (5V). The DRV8302 includes a buck converter that steps down the voltage from the main power supply to power the microcontroller, ensuring safe and efficient operation (5V 1A output for external electronics from the buck converter).

[1] Peripherals, in the context of microcontrollers, refer to the various interfaces through which the microcontroller can communicate with other devices or components. These may include digital input/output (GPIO), analogue-to-digital converters (ADC), digital-to-analogue converters (DAC), serial communication interfaces like UART, SPI, I2C, and timers, among others.

4.2 Software

4.2.1 Firmware and Operating System

The firmware of the VESC project is based on the ChibiOS/RT real-time operating system (RTOS). An RTOS is an operating system designed to process data as it comes in, typically within a very short time frame. RTOSs are used in environments where high reliability is required, and tasks must be processed under strict deadlines.

ChibiOS/RT is a compact and fast RTOS supporting multiple architectures. It is designed for embedded applications on 8-, 16-, and 32-bit microcontrollers. The kernel of ChibiOS/RT can achieve over 220,000 created/terminated threads per second and perform a context switch in 1.2 microseconds on an STM32 @ 72 MHz.

The firmware running on the VESC's STM32F4 microcontroller executes the control algorithms that drive the motor. These algorithms take input from the user interface and sensors to generate control signals for the MOSFET driver. The firmware supports several control modes, including duty cycle control, current control, and speed control, and it can be configured to use either sensorless control or sensor-based control with Hall-effect sensors or encoders. The firmware also includes features for protecting the system, such as under-voltage and over-current protection, and it provides diagnostic information like motor current, input voltage, and system status.

4.2.2 Writing custom applications

The VESC has several extra ports and much extra computational power, so it can be used to run custom user code in addition to controlling a motor. This is convenient when there are space constraints, and it is also the best way to implement real-time control applications where timing is critical. The VESC code is organised so it is easy to write and maintain custom applications while keeping the code up to date without having many conflicts when pulling updates using git. [https://vedder.se/2015/08/vesc-writing-custom-applications/]

The VESC project provides a tutorial on making a custom application to run a motor in a specific case: using speed control while a button is held with a speed proportional to the voltage on the ADC_EXT pin. To make a custom application, we can follow the steps provided by the project:

- **Preparation:** Ensure you can build and upload the firmware. We should be able to write and upload custom applications when we can successfully upload the firmware.
- **Connecting everything:** We must set everything up correctly depending on the application. The example used in the post is: When the pushbutton is held down, the VESC will run the connected motor in a speed control loop with a speed proportional to the position of the potentiometer between 0 and 10k ERPM.
- **Create the custom application:** In the VESC project post, there is all the steps necessary to implement a new application to the firmware.
- **Using your custom application:** Configure the VESC as usual. Select the "custom user application" in the app configuration section and write the configuration. The next time the VESC boots, our custom example application will be loaded.
- **Some tips:** In the VESC project post, a list of useful tips is provided when writing a custom application.

4.2.3 Graphical User Interface (GUI)

The VESC project includes a GUI known as the VESC Tool, which allows users to configure the VESC system, monitor its operation, and update the firmware. The VESC Tool is available for various platforms, including Windows, Linux, and Android, and it communicates with the VESC system via USB.

It allows users to select the control mode, set the motor parameters, configure the input sources (such as PPM, ADC, or UART), and adjust the system's protection settings. The VESC Tool also allows users to update the firmware on the VESC system. This involves downloading the appropriate firmware file, connecting the VESC system to the computer via USB, and using the VESC Tool to upload the firmware. This process allows users to upgrade to the latest firmware version easily or install custom firmware versions.

In addition to configuration, the VESC Tool provides real-time monitoring of the VESC system's operation. It displays information like motor speed, current, duty cycle, and input voltage, and it can plot these variables over time to help with troubleshooting and performance tuning.

Some screenshots of the configuration GUI:

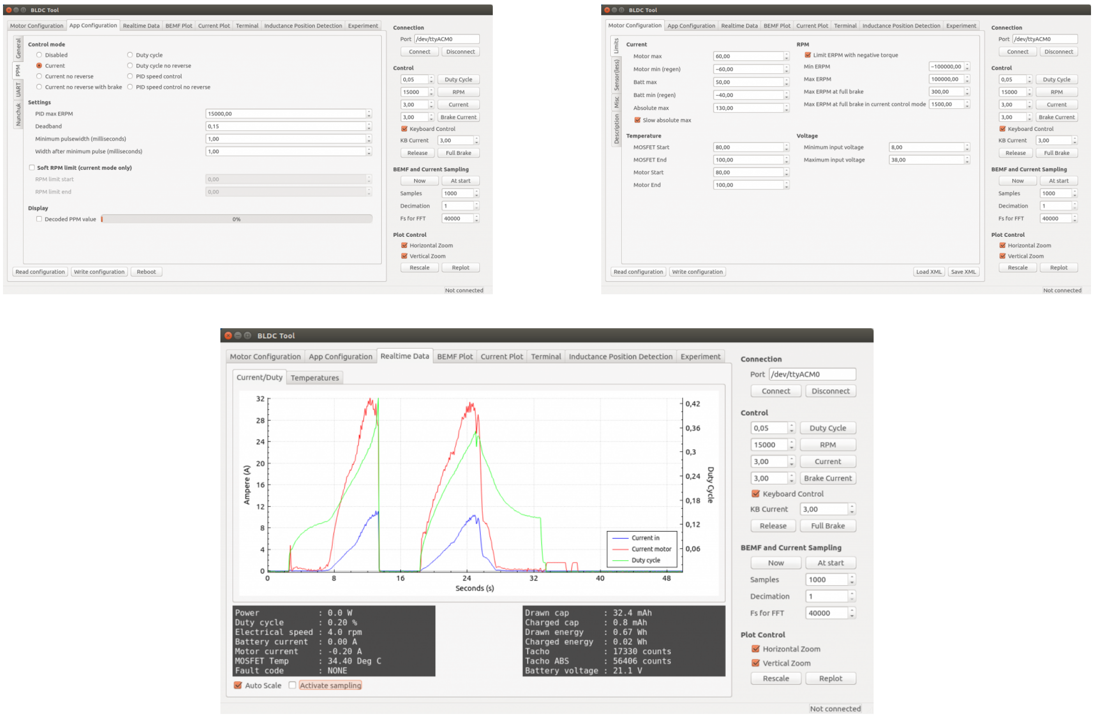

Figure 2: Some GUI screenshots

5 Interfaces

The VESC project provides multiple interfaces for controlling the motor and configuring the system. These interfaces allow the VESC to be controlled by various input sources and enable communication with external devices for monitoring and configuration.

5.1 PPM Signal

The PPM (Pulse Position Modulation) signal interface is commonly used in RC (Radio Control) applications. It is a type of analog signal where the position of each pulse corresponds to the control value. In the VESC project, a PPM signal can control the motor's speed or duty cycle, allowing the VESC to be used in RC vehicles like cars, boats, or drones.

5.2 Analog Input

The VESC supports analog inputs for controlling the motor. This can interface with various sensors or control devices with analog voltage outputs. Thanalogueog input can control different aspects of the motor's operation, such as speed or duty cycle.

5.3 UART

UART (Universal Asynchronous Receiver/Transmitter) is a serial communication protocol used for communication between the VESC and external devices. The VESC's UART interface allows it to send and receive data, enabling control and monitoring capabilities. For example, the VESC can send real-time data about the motor's operation, such as its speed, current and/or temperature, to an external device for monitoring or logging. The UART interface can also send commands to the VESC to control the motor or configure the system.

5.4 I^2C

The I^2C (Inter-Integrated Circuit) interface is a serial communication protocol for connecting low-speed devices. In the VESC project, the I^2C interface connects a wireless Wii Nunchuk controller for controlling, for example, an electric skateboard.

5.5 USB

The VESC includes a USB interface for connecting to a computer or Android device. This interface primarily configures the VESC system using the VESC Tool software. The USB interface allows users to easily set the system's parameters, update the firmware, and monitor the system's operation in real-time.

5.6 CAN-bus

The CAN (Controller Area Network) bus interface is a robust communication protocol used in automotive and industrial applications. The VESC supports CAN-bus communication, connecting multiple VESCs to the same network. This is particularly useful in applications that require multiple motors to be controlled independently but work together, such as in a four-wheel-drive electric vehicle.

6 Protection Mechanisms

The VESC project incorporates a range of protection mechanisms designed to ensure the safety, reliability, and longevity of the motor control system and its motors. These mechanisms are critical for preventing damage due to anomalous operating conditions, such as voltage spikes, excessive current draw, rapid changes in the duty cycle, and thermal overload.

- **Voltage Protection:**
  - Low Voltage Protection: Prevents the system from operating under a voltage that is too low, which could lead to inefficient motor performance or damage to the power electronics. This protection is essential for battery-powered applications, where low voltage may indicate a consumed battery.
  - High Voltage Protection: Protects the system against over-voltage conditions due to power surges. High voltage can cause excessive stress on electronic components, leading to potential failure.
- **Current Protection**
  - Motor Current Limiting: Monitors the current to the motor, ensuring it does not exceed a predefined threshold. Excessive current can cause overheating and damage the motor windings or the power MOSFETs.
  - Input Current Limiting: Similar to motor current limiting, but focuses on the current drawn from the power source (like a battery). This is crucial to prevent damage to the power source and the motor controller.
  - Regenerative Braking Current Limiting: Controls the current during regenerative braking, preventing excessive current flow back into the power source, which could be damaging.
- **Duty Cycle and RPM Protection**
  - Rapid Duty Cycle Changes (Ramping): Rapid changes in the duty cycle can lead to voltage spikes and mechanical stress on the motor. The VESC implements controlled ramping of the duty cycle to mitigate these risks.
  - High RPM Limits: Sets a maximum rotational speed for the motor to prevent mechanical and electrical overstressing. The limit can be set differently for each direction of motor rotation.
- **Thermal Protection**
  - Monitors the temperature of critical components and reduces output power or shuts down the system to prevent critical damage due to overheating.

Soft Back-Off Strategy: When any of the above limits are approached or reached, the VESC employs a soft back-off strategy. This strategy is designed to reduce the operational parameters (like current or duty cycle) in a controlled manner to bring the system back into a safe operating range. The approach is gradual to avoid sudden changes that could destabilize the motor control, offering a balance between maintaining performance and ensuring safety. If the parameters significantly exceed safe limits, the VESC may shut down the motor completely to prevent damage. This strategy is essential for applications where safety and reliability are paramount.

References