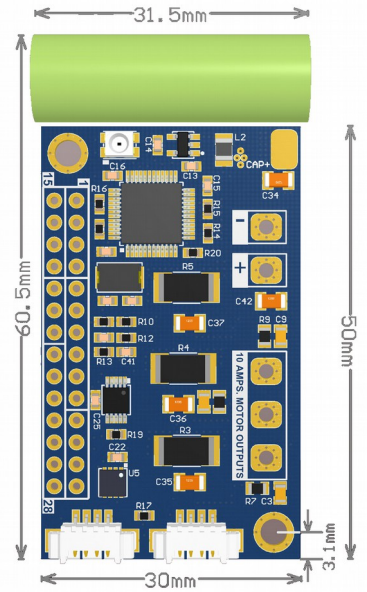
CAN_Drv module

3-phase BLDC motor driver to be used as a part of the SimpleBGC32 stabilizer system

https://www.basecamelectronics.com Version: 1.01 Edit: 15. Oct. 2020



Introduction

CAN_Drv is a DC brushless motor driver with the CAN-bus interface, intended to be used with the CAN-enabled SimpleBGC32 controller in a camera stabilization system, replacing the on-board motor drivers. System supports up to 3 CAN_Drv modules for the main stabilization axes, and up to 4 additional modules for the other tasks. Such modular scheme benefits by the optimized wiring between modules that is perfectly immune to EMI noise. It provides a better motor control algorithm compared to conventional SimpleBGC32 controllers with the integrated drivers.

This document provides a specifications and pin configuration of the CAN_Drv module designed by the Basecamelectronics company. Modules, developed by the partners of Basecamelectronics, may differ in specifications and pin-out – please refer to their manuals for details.

The information regarding a configuration and possible applications of this module can be found in the reference manuals for the CAN_Drv module and SimpleBGC32 main controller. The most actual version is published at the product's page: https://www.basecamelectronics.com/can_driver/ and in the "Manuals" section of our web site.

Features

- High-current and high-voltage output MOSFETs allows energy-efficient motor driving working with wide range of applications
- Fully-featured field-oriented control (FOC) of the brushless motor with various modes of operation (speed, torque, position and gyro-based feedbacks are supported)
- Built-in over-current, short-circuit, under-voltage, over-temperature protections make device immune to the most harmful working conditions
- Optional current limiting function and virtual temperature model saves battery lifetime and protect motors
- GUI provides easy tuning and calibrations (including fully automatic motor parameters estimation); a big number of adjustable parameters for maximum flexibility and efficiency

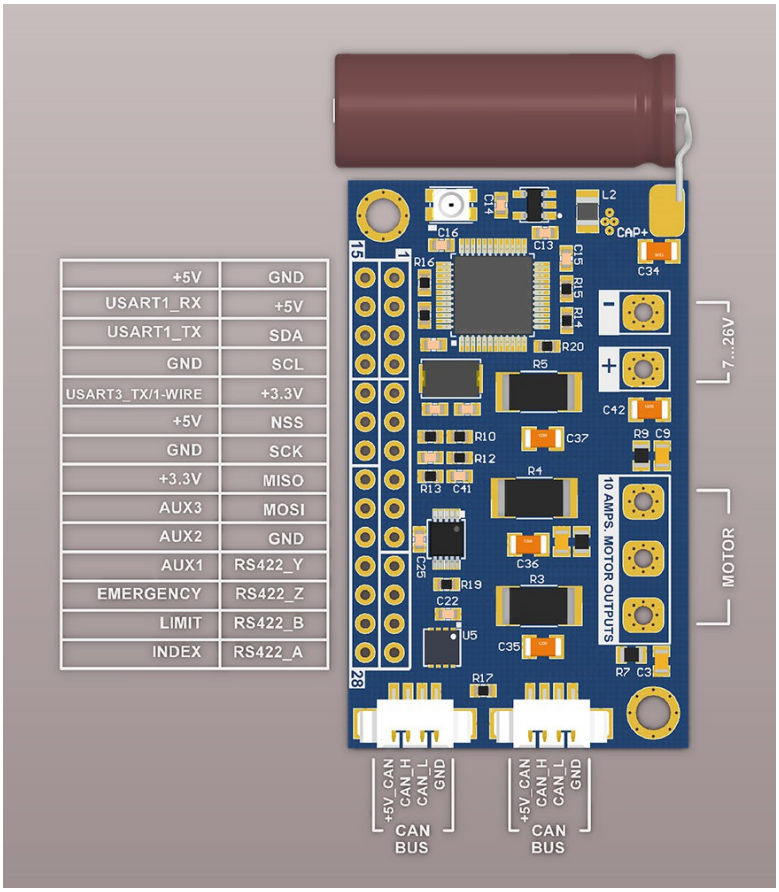
- Firmware upgradeable over CAN-bus interface simplifies a support of a system
- Build-in 5V and 3.3V switching regulators allow to connect an external load
- 2x can ports allows daisy-chain connection of modules
- Support of a big range of encoder interfaces and a big number of encoder models (new models could be add in firmware later)

Specifications

- Input voltage: 6-26V (2S – 6S Li-ion battery equivalent)
- Current: 10A continuous (15A with the heatsink installed)[(1)]; max. 40A impulse
- +5V line max. load: 800mA
- +3.3V line max. load: 100ma (150 mA for a short time)
- Working temperature[(1)]: -40 ... +85 °C
- Dimensions: 30mm x 50mm (31.5 x 61.5mm with capacitor)
- Weight: 11g (17g with pin headers and power terminal)
- Built-in protection systems:
  - Over-current protection with the configurable thresholds for an impulse and average current
  - Under-voltage protection with two thresholds: recoverable and critical
  - Output short-circuit protection [(2)]
  - Over-temperature protection with the on-board and external (motor) sensors or software-computed motor thermal model
- Control interface: 2x CAN-bus with proprietary protocol (specification can be provided upon request)
- Encoder interfaces:
  - SPI
  - PWM (not recommended due to big phase delay at high speeds of rotation)
  - I2C
  - A,B,Z
  - SSI, BiSS
- Other interfaces:
  - Limit switch
  - Z-Index switch
  - Emergency switch
  - 2x digital I/O pins
  - 1-wire for thermal sensor connection

(1) MOSFETs and shunt resistor may need proper thermal dissipation when working in a hard temperature environment

(2) Protected from short circuit phase-to-phase and battery-to-phase. **WARNING**: phase-to-GND is not protected and can damage MOSFET!



Pin configuration

Name	No.	Type[1]	Description	+5V tolerant
HEADER ROW #1				
+5V	15	P		
UART_RX*	16	I	Use only for firmware upgrade when motor is NOT CONNECTED!	•
UART_TX*	17	O		•
GND	18	P		
AUX3 / 1-Wire	19	OD	Pulled to +5V by the 4.7k resistor. Supported temperature sensor model: DS18B20 or compatible	•
+5V	20	P		
GND	21	P		
+3.3V	22	P		
AUX3 / 1-Wire	23	OD	connected to pin 19 internally	•
AUX2	24	I/O	depending on configuration	
AUX1	25	I/O	depending on configuration	•

EMERGENCY	26	I	Pulled HIGH internally, active 0	
LIMIT	27	I	Pulled LOW internally, active 1	
INDEX	28	I[2]	Pulled HIGH or LOW internally depending on a model of encoder	
HEADER ROW #2				
GND	1	P		
+5V	2	P		
I2C SDA	3	OD	Pulled HIGH by the 3.3k resistor	•
I2C SCL	4	I2C: OD PWM: I SPI: O	Pin mode depends on configuration: I2C: Pulled HIGH by the 3.3k resistor (connected to pin 6 internally) SPI, PWM: see below	•
+3.3V	5			
SPI CS / PWM IN	6	I2C: OD PWM: I SPI: O	Pin mode depends on configuration: SPI: “device select” output PWM: input (connected to pin 4 internally)	•
SPI SCK	7	O	SPI clock signal (CLK)	•
SPI MISO	8	I		•
SPI MOSI	9	O		•
GND	10			
RS422_Y	11	O		•
RS422_Z	12	O		•
RS422_B	13	I	Terminated by the 100 Ohm resistor internally	•
RS422_A	14	I		•
CAN PORT (top view)				
GND		P		
CAN_L		O	Can be terminated by the internal 120 Ohm resistor[3]	
CAN_H		O		
+5V		P	Is not connected to the +5V power line of the module. Used as pass-through only.	

(1) I = input, O = output, OD = open-drain output, P = power
(2) In the firmware 1.0 pin is configured as output, be careful on connection!
(3) Solder a jumper SW4, located on the back side of the board, to terminate the line by the 120 Ohm resistor. Should be done for the last device on a CAN bus in case of a "serial" connection. For a "star" connection, leave each device unterminated (but the length of unterminated line may affect the noise immunity in this case).

Logic level: HIGH = +3.3V, LOW = GND. +5V tolerant pins can operate with 5V logic levels.

Device address selection by jumpers

By soldering address selection jumpers, you can assign hardware address that the main controller will use to refer for a particular CAN_Drv module. Leave all jumpers open to assign address in the GUI - in this case, it will be stored in the EEPROM of main controller.

SW3	**SW2**	**SW1**	**Address**
0	0	0	software-assigned
0	0	1	drv#1
0	1	0	drv#2
0	1	1	drv#3
1	0	0	drv#4
1	0	1	drv#5
1	1	0	drv#6
1	1	1	drv#7