Rayon

April 2012

Address : Hamelacha 8 .st, industrial zone Netania. POB 916 Pardesia 42815 Israel

Tel: 972-9-8923125. Fax:972-9-8923126 Email: shaya@redler.co.il web: www.redler.co.il

**High performance motor controller driver**

**User manual**

**Revision 1.0**

**Redler Technologies**

**02.05.2019**

# About Redler

Thank you for choosing our product. Redler Computers Ltd since 1994, is an innovative & reputable Aerospace and defense advanced Control and Mechatronics Company.

Our company specializes in cost effective digital servo drivers and advanced multi and single axis motion control systems for Precise Guided Monition, Unmanned Avionic Vehicles, Unmanned Ground Vehicles, gimbaled stabilized system, heavy duty sensor less controller driver and many other applications that demands robustness and exceptional reliability.

The company's embedded systems, based DSP microcontrollers and unique motion control solutions, designed with special care for extreme environmental condition, high performance and low losses.

# Contents

[1. About Redler 2](#_Toc11086789)

[2. Contents 3](#_Toc11086790)

[3. Read this first 6](#_Toc11086791)

[3.1 Safety 6](#_Toc11086792)

[3.2 System considerations 6](#_Toc11086793)

[3.3 Cooling 6](#_Toc11086794)

[3.4 Load considerations 6](#_Toc11086795)

[3.5 Commands 6](#_Toc11086796)

[4. Motor controller overview 7](#_Toc11086797)

[4.1 Introduction 7](#_Toc11086798)

[4.2 Key features 7](#_Toc11086799)

[4.3 Motor feedback support. 8](#_Toc11086800)

[4.4 Single encoder Schematics 8](#_Toc11086801)

[4.5 Secondary feedback Schematics. 8](#_Toc11086802)

[5. Hardware installation 9](#_Toc11086803)

[5.1 Mounting motor controller. 9](#_Toc11086804)

[5.2 Power supply. 9](#_Toc11086805)

[5.2.1 Supply setup 9](#_Toc11086806)

[5.2.2 Supply current and voltage demands : 9](#_Toc11086807)

[5.2.3 Supply cable 9](#_Toc11086808)

[5.2.4 Regenaration 9](#_Toc11086809)

[5.3 Connecting Motor and communication cables. 9](#_Toc11086810)

[6. Using GUI. 10](#_Toc11086811)

[6.1 Preface 10](#_Toc11086812)

[6.2 GUI installation. 10](#_Toc11086813)

[6.3 Main window area 10](#_Toc11086814)

[6.4 Setting parameters. 10](#_Toc11086815)

[6.5 Motor calibrations. 10](#_Toc11086816)

[6.6 Storing parameters to file. 10](#_Toc11086817)

[6.7 Graph. 10](#_Toc11086818)

[6.8 Recording graph to file. 10](#_Toc11086819)

[6.9 Firmware update. 10](#_Toc11086820)

[7. Quick motor setup example 11](#_Toc11086821)

[7.1 Motor main parameters. 11](#_Toc11086822)

[7.2 Feedback main parameters. 11](#_Toc11086823)

[7.3 “Feedback source” and “Command source” select. 11](#_Toc11086824)

[7.4 Speed and position profiler (optional). 11](#_Toc11086825)

[7.5 Other motor options 11](#_Toc11086826)

[7.6 Calibrations 12](#_Toc11086827)

[8. Motion control examples 13](#_Toc11086828)

[8.1 Current control example. 13](#_Toc11086829)

[8.2 Speed control example. 14](#_Toc11086830)

[8.3 Position control example. 15](#_Toc11086831)

[9. Advance use 16](#_Toc11086832)

[9.1 Motion control overview 16](#_Toc11086833)

[9.2 The Current Drive controller 16](#_Toc11086834)

[9.3 The Speed Drive controller 19](#_Toc11086835)

[9.4 The position Drive controller 20](#_Toc11086836)

[9.5 Current Limits. 21](#_Toc11086837)

[9.6 Protections. 21](#_Toc11086838)

[9.7 Power considerations. 21](#_Toc11086839)

[9.8 Thermal considerations. 21](#_Toc11086840)

[9.9 Inertia considerations. 21](#_Toc11086841)

[10. Communication. 24](#_Toc11086842)

[10.1 Serial Protocol: 24](#_Toc11086843)

[10.1.1 Massage structure – Get command 24](#_Toc11086844)

[10.1.2 Massage structure – Data command 24](#_Toc11086845)

[10.1.3 Massage structure – Data response 24](#_Toc11086846)

[10.1.4 CRC calculations code: 25](#_Toc11086847)

[10.1.5 Commads presantation in this document 26](#_Toc11086848)

[10.1.6 Get example 27](#_Toc11086849)

[10.1.7 Set example “On/Off Command” 28](#_Toc11086850)

[10.1.8 Timing 28](#_Toc11086851)

[10.1.9 Requirements 28](#_Toc11086852)

[10.1.10 Error massages 29](#_Toc11086853)

[10.2 CanOpen protocol 29](#_Toc11086854)

[11. Commands 30](#_Toc11086855)

[11.1 Commands description 30](#_Toc11086856)

[11.1.1 Enable Drive 30](#_Toc11086857)

[11.1.2 Stop motion 30](#_Toc11086858)

[11.2 Commands table summery. 31](#_Toc11086859)

[12. Specifications. 37](#_Toc11086860)

[13. Electrical ICD. 38](#_Toc11086861)

[13.1 Electrical block diagram 38](#_Toc11086862)

[13.2 Power connector description 38](#_Toc11086863)

[13.2.1 Option 1 38](#_Toc11086864)

[13.2.2 Option 2 38](#_Toc11086865)

[13.2.3 power connector signals 38](#_Toc11086866)

[13.2.4 Capacitor board 38](#_Toc11086867)

[13.3 Communication connector description 38](#_Toc11086868)

[13.4 Digital Inputs and outputs 39](#_Toc11086869)

[14. Mechanical ICD. 40](#_Toc11086870)

# Read this first

## Safety

This product intended to use by professionals, user must take care when using this product, Redler does not take liability to loss or damage caused by this product, Skill needed to operate this controller, please read this manual carefully, while using this product, please apply safety precautions to protect human and machine, especially while developing and integrating system.

## System considerations

* Please verify that the motor is secured carefully to the system or bench.

If not secured, Motor may move during calibrations and accelerations.

* All motor controller screw must be tightened to specifications.
* Please secure all cables to bench, cables will should be arranged so that they will not be harmed or tingled.
* Driver, motor or cables should not be moved during operation.
* Verify that the driver environmental conditions are met.

## Cooling

Motor controller performance are related to temperature. Maximum current capability can be reduced if not cooled properly, see thermal consideration chapter.

## Load considerations

* Please verify load is secured to motor shaft or gear.
* Verify system rigidness and no excessive backlash.
* Stopping motor and deceleration with heavy load, can increase voltage on power supply rail, this can lead to motor controller failure or PSU failure. Special care should be taken to absorb this kinetic energy. See inertia consideration chapter.

## Short commands

For easy reading and short representation, commands referenced during this document will be with dedicated short representation “{ID[Idx]}”, for more information see “communication” chapter.

# Motor controller overview

## Introduction

Digital servo driver intended to use in current, speed or position control applications, when high accuracy, performance and efficiency is needed. sinusoidal phase currents. Advance motion support such: Homing, Feedforward, position and speed profilers. For application such as electric vehicles, robotic arms, CNC and gimbals.

## Key features

* Brushed or Brushless Motors (Sensor-less is optional)
* Sinusoidal phase currents. Field-oriented control, IQ and ID control, Space vector PWM.
* Motion support: current, speed or position control.
* Homing, Feedforward, position and speed trapezoid profilers.
* Most of common feedbacks can be supported: Hall, Qep Encoders, Absolute Sin/Cos, incremental Sin/Cos and SSI encoders. (Resolver is optional).
* Communication types: supports RS232 and RS422 up to 460800 baud and CanOpen up to1Mbit baud. (LAN, RS485 and USB are optional)
* Digital Inputs and Outputs: 3 digital isolated inputs, 2 digital isolated output, Additional 1 digital isolated input can be set as STO.
* Analog command input [-10v,10v] for current speed and position
* Power output – Power supply switch to drive DC external load such as break, 2A max.
* Protections: Under voltage, over voltage, Short protection, Temperature, out of position, Motor stuck and driver self HW tests.
* Comprehensive parameter list to control each aspect of the system.
* GUI: Advance GUI with 3KHz signal monitoring graph, support parameters command input and calibration.
* Automatic calibrations: Simple to use fast method to start working with you motor:
* High power: Current output 70A continues, 140A peak.
* Small dimensions

## Motor feedback support.

This motor controller can support both brushed and brushless motors, user must select the right feedback according application demand, See table below:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Feedbacks | Brushed | Brushless |
| 1 | **No feedback** | Current loop only | 1. Forced rotate commutation |
| 2 | **Incremental Enc** | All loops | Not supported |
| 3 | **Hall** | - | (2) All loops, Six step commutation |
| 4 | **Inc Enc + Hall** | - | (3) All loops, FOC, Sinusoidal commutation. |
| 5 | **Absolute Biss or SSI** | All loops | (4) All loops, FOC, Sinusoidal commutation. |
| 6 | **Absolute sin/cos** | All loops | All loops, FOC, Sinusoidal commutation. |

Notes:

1. Open loop Speed/Position by sending current command and force rotating electrical commutation angel (sinusoidal)
2. When high position accuracy and low speed is needed, Hall is not recommended.
3. Inc Encoder TTL or SIN/COS input is for increased resolution.
4. Please verify that your SSI/ Biss protocol is supported, see encoder computability table.
5. For best performance with brushless motor, it is recommended to use Sinusoidal commutation, FOC (options 4, 5 and 6).

## Single encoder Schematics

The motor encoder connected to driver, this encoder loops will be used as, commutation sensor, this encoder will also be used speed and position “Feedback source” for control loops.

Motor controller

PSU

15-80V

(-) (+)

Hall/Encoder

Phases

![](data:None;base64,)

RS232/RS422/CAN

Motor

## Secondary feedback Schematics.

In addition to motor feedback it is possible to add secondary encoder on main shaft, the user can select this as “position feedback source” this will eliminate any backlash errors and give absolute position if absolute encoder is used (or after homing).

Encoder 2

Enc2S

Main Shaft

PSU

15-80V

(-) (+)

Hall/Enc1

Phases

![](data:None;base64,)

Motor controller

RS232/RS422/CAN

Motor

Gear

# Hardware installation

## Mounting motor controller.

Mount driver to system or head-sink, apply thermal paste or thermal pads if needed, please tighten all screws to the specification. See “Mechanical ICD” and “Thermal considerations”

## Power supply.

### Supply setup

User can connect PSU and or Battery, do not turn On the power supply before verifying polarity securing all cable and motor connections and verifying polarity, For battery option add safety Off switch.

### Supply current and voltage demands :

**Input voltage:** from 15V to 80V with under voltage over voltage protection.

**Current:** Power supply should support motor peak power demands, Peak power will be:

Ppsu[W] = Ipsu\* Vpsu = Im^2\*Rm+w\*Nm == Im^2\*Rm+w\* Kt\*Im

Where:

Im – I max motor current [A]

Rm – R motor phase to phase resistance \*150% [Ohm]

w – Max angular velocity [Rad/sec]

Kt – Motor constant [Nm/A]

Nm – Max force needed for acceleration [Newton/meter]

### Supply cable

Supply cable should be short as possible to reduce voltage drop and power loss.

When using long cable user should add bus-capacitors close to Motor controller side.

### Regenaration

During motion brakes, the regenerative energy is injected into the motor power supply, please see “Inertia considerations”.

## Connecting Motor and communication cables.

Please connect Communication, Motor Feedbacks and power cables. tighten all connector screws to the specification, see “Electrical ICD” and “Mechanical ICD”

# Using GUI.

## Preface

This GUI is intended to use in laboratory for motor calibration, parameter setup, motion evaluation, debugging and monitoring. It is necessary to add safety power kill switch to overcome any communication loss in driver side or PC side.

## GUI installation.

### Requirements

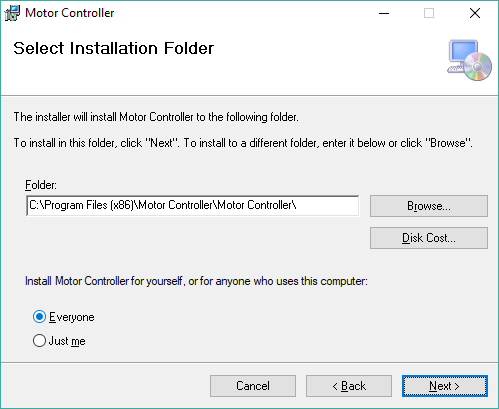
The installation is for Windows 7 64 Bit or higher computer.

To see graph records you will also need Microsoft Office Excel 2013.

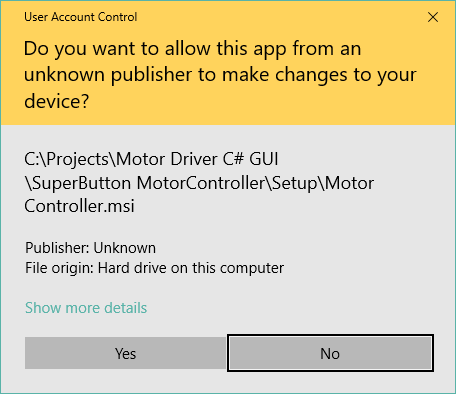
### Install “Motor Controller” on computer.

Open installation folder, Double click on: < Disk >\MotorController Setup\ “setup.exe”

Follow instructions, press next when asked and leave default installation folder.



If asked to allow app to make changes, press “Yes”:

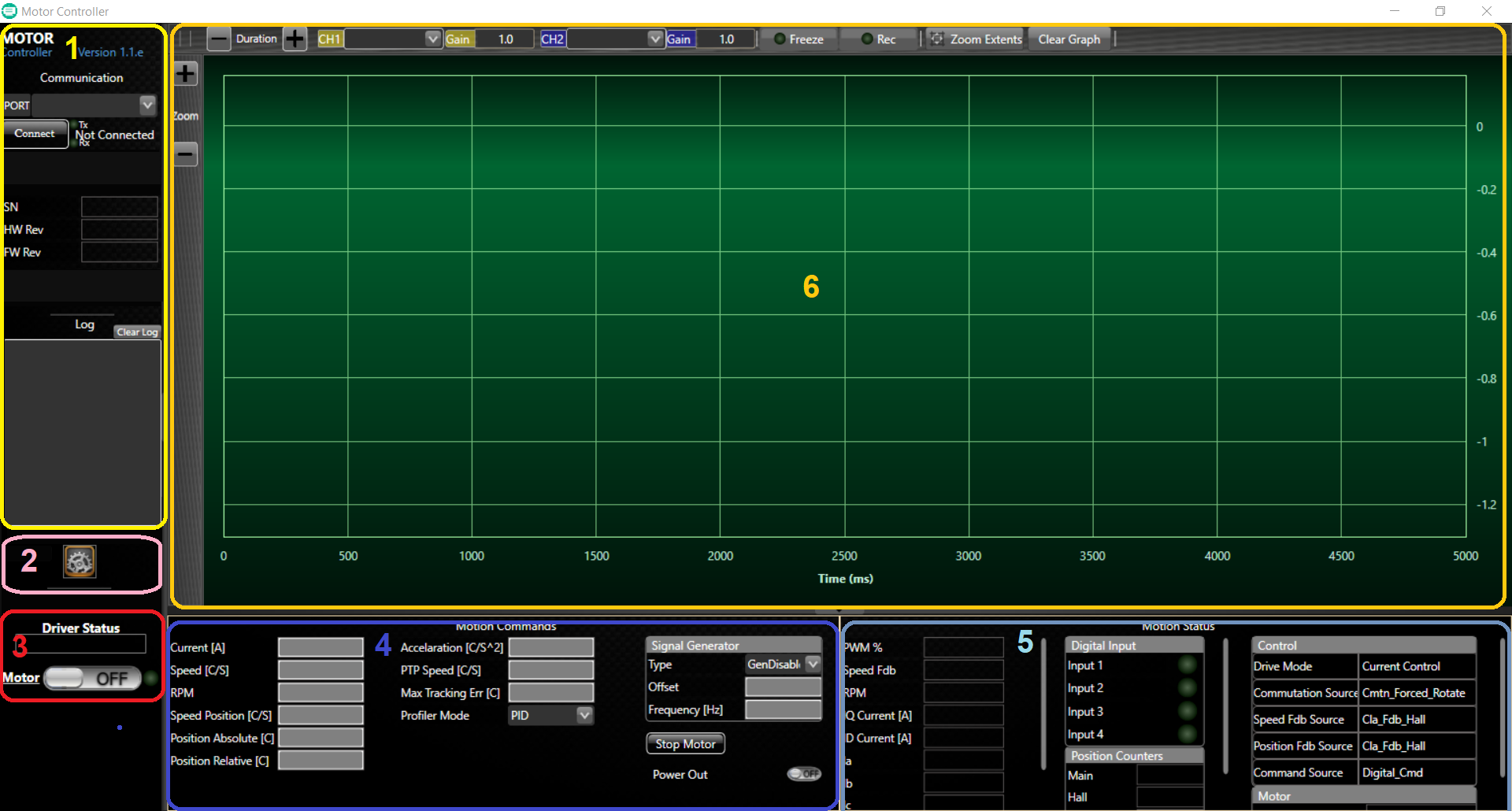


### Running aplication

The application shortcut is placed in the windows start menu and “Motor Controller:” shortcut will be placed on desktop

## Main window area

We will divide main GUI window to six areas and explain each part.



### Sending command and reading values

To send command click on the textbox, the textbox background color change to red, indicating the user can type a value to send. When you finished updating the value, you can press enter key to send the value or escape to cancel operation.

User do not need to read data, the displayed data in GUI is refreshed every few seconds by sending get command automatically.

### Comunication - area 1:

From top to bottom:

1. **Application version:** GUI version
2. **Communication:**

* COM port selection: if port detected at application starting time, the port will be selected by default, if more than one COM port are available you will have to choose from the combo box list.
* **Connect button:** allow to open the selected COM port to establish communication with the unit.
* **Connection status:** Tx and Rx led represents the received and transmitted data between the PC and the unit.

1. **Driver information:** updates at startup: SN, HW rev, FW rev
2. **Log:** Log section allow you to see some running processes of the application, Clear log allow you to erase the log data.

Note: The timestamp shows only minutes and seconds. e.g:

*34:07 - Failed*

*34:06 - Autobaud process...*

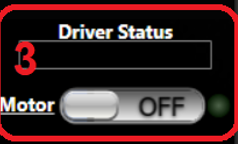
*34:06 - Success*

*34:06 - Connecting at COM5*

### Setting - area 2:

Open the unit parameters window.

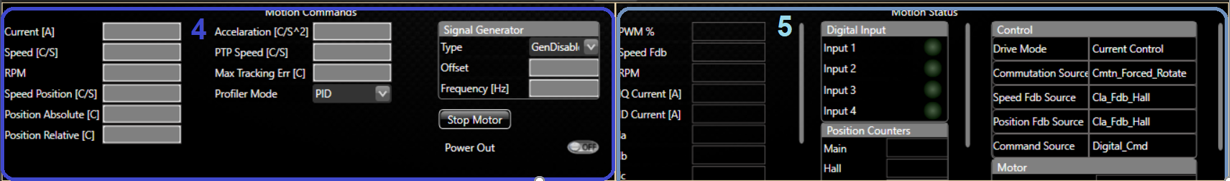
### Motor - area 3:



From top to bottom:

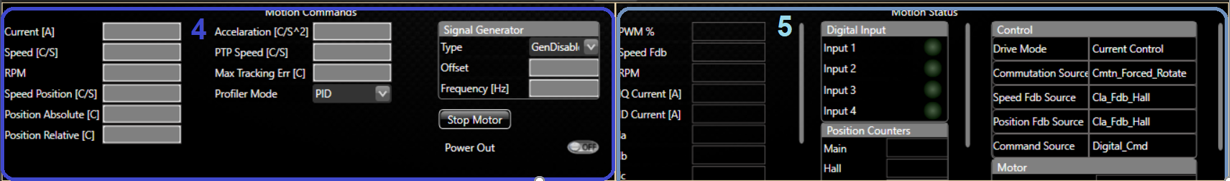
* **Driver Status:** display if the driver has no error or specify the error if there is one.
* **Motor ON/OFF:** Control the motor state, Green round led indicate motor is “ON” state.

### Motion Commands - area 4:



Motion commands allow you to send operation commands to the controller.

### Motion Status - area 5:



Motion status allow you to see the data of the controller.

### Graph - area 6:



1. **Zoom**: Allow you to increase or decrease amplitude per graph decade.

This operation can be done also with mouse wheel by scrolling in or out.

1. **Duration**: Allow you to increase or decrease time per graph decade.

This operation can be done also by moving mouse cursor right/left on Time axis.

1. **Ch1, Ch2**: Graph selection list. Allow you to plot driver signals such as Current or Speed at 3Khz.
2. **Gain**: Gain factor to scale two plots on the same graph.
3. **Freeze**: Stop updating the graph by performing “pause” operation until you release the button. A round red led blink during the freeze operation.
4. **Record**: Record the graph signals in an Excel document.

Directory: “C:\Users\<name>\Documents\MotorController\Charts”.

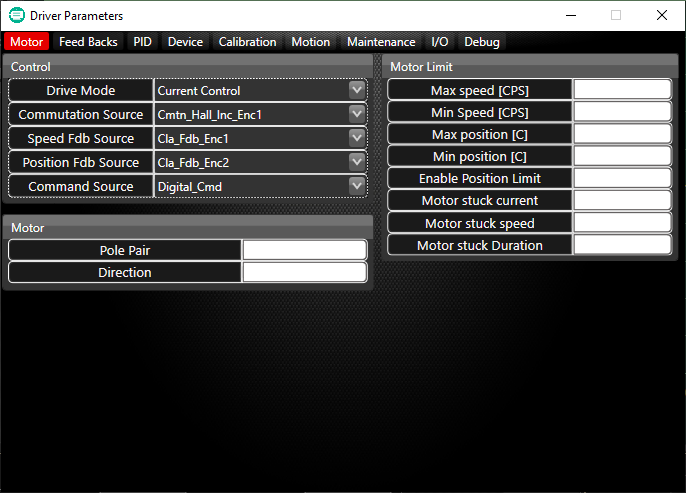
The file is created and saved after releasing the record button.

A round red led will blink during the record operation.

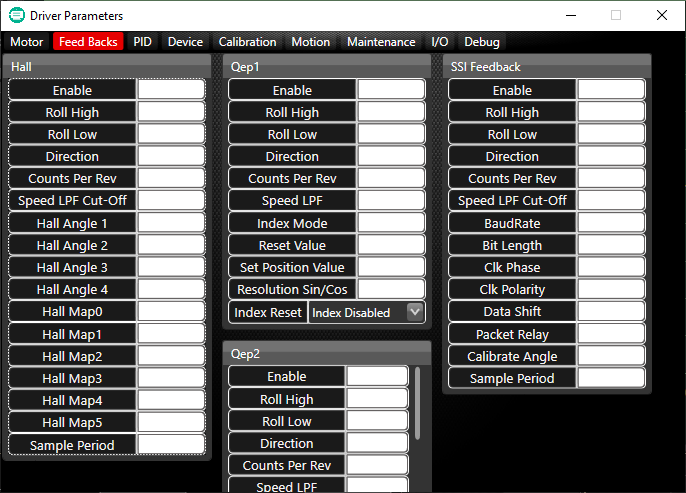
1. **Zoom Extent**: Fit the graph plot to the window area.
2. **Clear Graph**: Clear the graph area.
3. **Graph area**: X axis title is Time (ms), Y axis updated dynamically depend of the data graph selected and gain.

## Setting parameters.

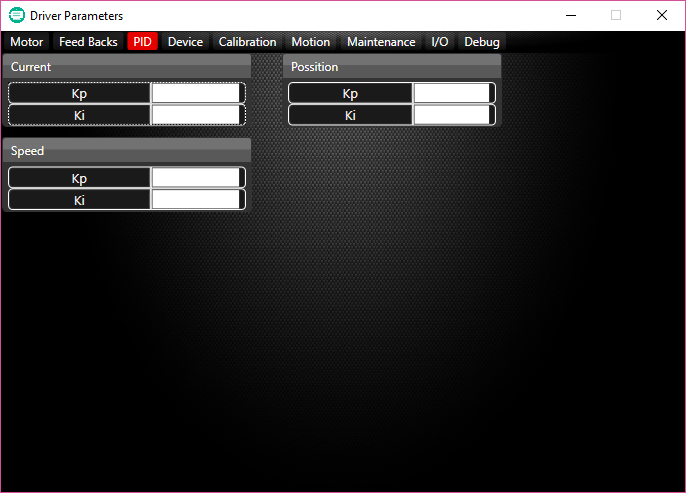
### Motor parameters tab



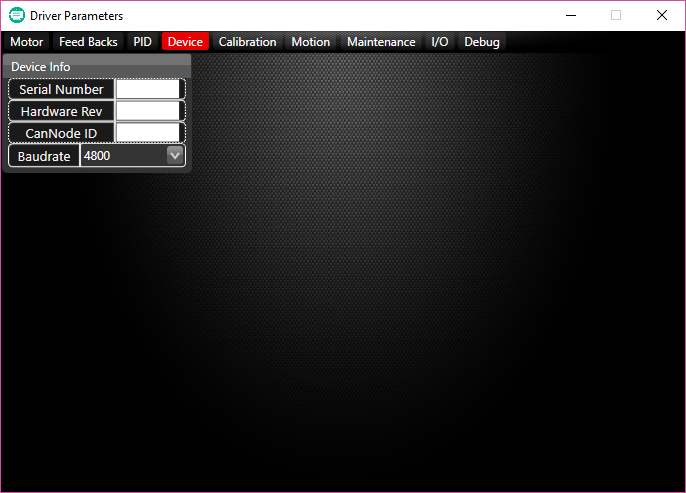
### Feedbacks parameters tab



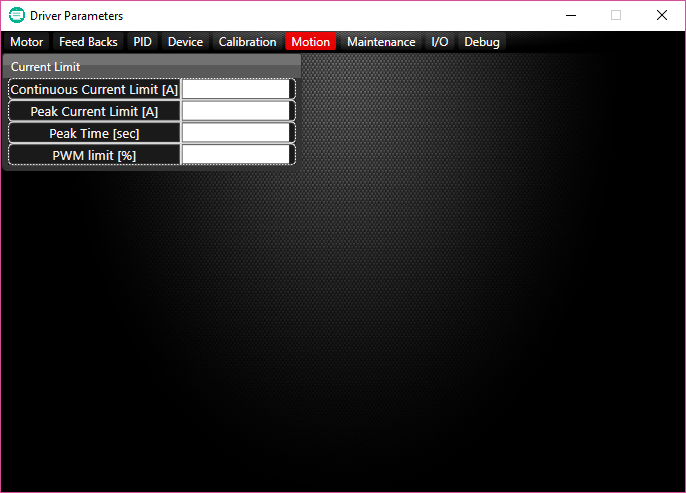
### PID parameters tab



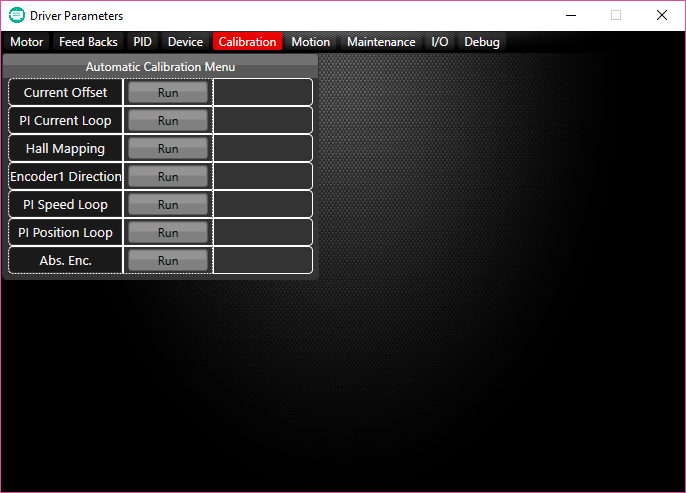
### Device parameters tab



### Motion parameters tab

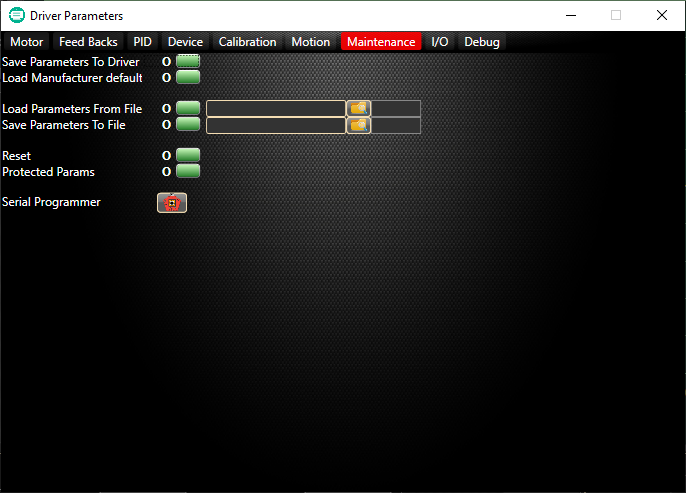


### Manual calibration Tab



For calibration steps please see user manual - Calibration paragraph. Do not calibrate without motor parameter setup and save.

### Maintenance tab



* **Save Parameters to Driver**: after any change performed in parameters, if you want to them to be saved to flash after a reset you must perform “Save Parameters to driver” operation. Otherwise the last parameters will be restored from flash after reset.
* **Load Manufacturer Default**: This operation allows you to restore manufacture defaults parameters of the unit.
* **Load Parameters from File**: you can load parameters to the unit from a file. You must click the “Folder Icon” and select the file before you press on “Save Button”.
* **Save Parameters to File**: allow you to save the unit parameters to a text file.

after downloading the data from the unit, you will be prompted to choose a directory and a file name to save the file.

directory.

* **Reset**: Send reset operation to the unit, useful to ensure that parameters were saved, after pressing “reset” in log window you will see “unit reset” massage.
* **Protected Params**: Not relevat, for advance use.
* **Serial Programmer:**  This feature will work only after installing LabVIEW Run Time Engine 2017 (32bit).

See next chapter (6.5) for further explanation.

## Firmware update

### Requirements

The installation is for Windows 7 32 Bit or higher computer.

### Install “LabVIEW Run Time Engine” on computer.

Open installation folder, Double click on: < Disk >\ “LVRTE2017std.exe” Follow instructions, press next when asked and leave default installation folder.

If asked to allow app to make changes, press “Yes”:



### Serial Programmer GUI.

After you install LabVIEW Run Time Engine 2017 you can laucnh the serial programmer from the chapter 6.4.7.

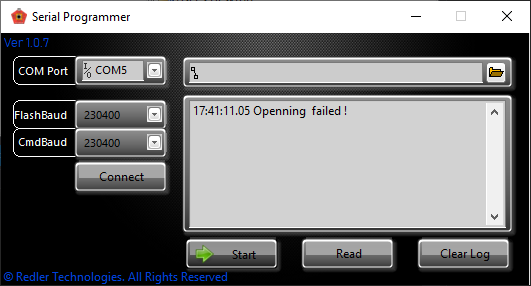
Choose a COM port from the “COM Port” combo box.

Choose a command baud rate and flash baud rate.

Select a folder to burn into the unit.

Press Start button when ready.

Wait until end of operation message in the logger window.



# Quick motor setup example

This chapter will review basic motor parameters options and calibration steps. Motor with Hall sensors and QEP incremental encoder will be used as example.

**Example** **motor spec:**

Mean current: 10A

Poles: 6

Halls sensors: yes

Encoder: incremental QEP 2048 lines.

**Electrical Setup:**

Connect encoder to driver QEP1 input.

Connect Hall sensor to hall driver input.

Connect rs232 communication cable to Driver and PC.

Turn on PSU.

(for more info see electrical ICD)

**GUI Setup:**

open GUI, press connect and go to Parameters window, change the following parameters:

## Motor main parameters.

* Set Mean current: 10A
* Set Peak current:10A
* Peak Time: 2 sec
* Set Pole pairs: 3 (poles/2)

## Feedback main parameters.

* Under Feedback, Enable motor Feedback: Hall, Qep1.
* Disable all non-active like Qep2, SSI etc.
* Set Qep1 “Counts Per Revolution” to 8192 (encoder lines\*4)

## “Feedback source” and “Command source” select.

* Set “Commutation source” to Hall and QEP1.
* Select “Speed feedback source” to “Encoder 1”
* Select “Position feedback source” to “Encoder 1”.
* Set “command source” to “Digital”

## Speed and position profiler (optional).

* Set “profiler mode” to “Trapezoid”
* Set Acceleration to 81920 [Counts/Sec^2]
* Set PTP Speed (max speed for trapezoid position motion) to 81920 [Counts/Sec]
* Set max speed limit to 100000 [Counts/Sec] (must be greater PTP Speed)

## Other motor options

For more information and full command list please review “motor calibrations”, “Motion commands” and “motor protections” in command description chapter.

## Calibrations

Caution! Please first set motor and feedback parameters 7.1 – 7.3

### Calibration by wizard

Will be updated

### Step by step

Go to “Calibration” tab press “Run” on each of the following commands by this exact order, each time wait for success and then continue:

* press Run “PI current loop” wait for success
* press Run “Hall mapping” wait for success.
* press Run “Encoder1 Direction” wait for success.
* press Run “PI speed loop” wait for success.

In case of calibrations failure, Stop the sequence and follow instructions in calibration troubleshoot chapter.

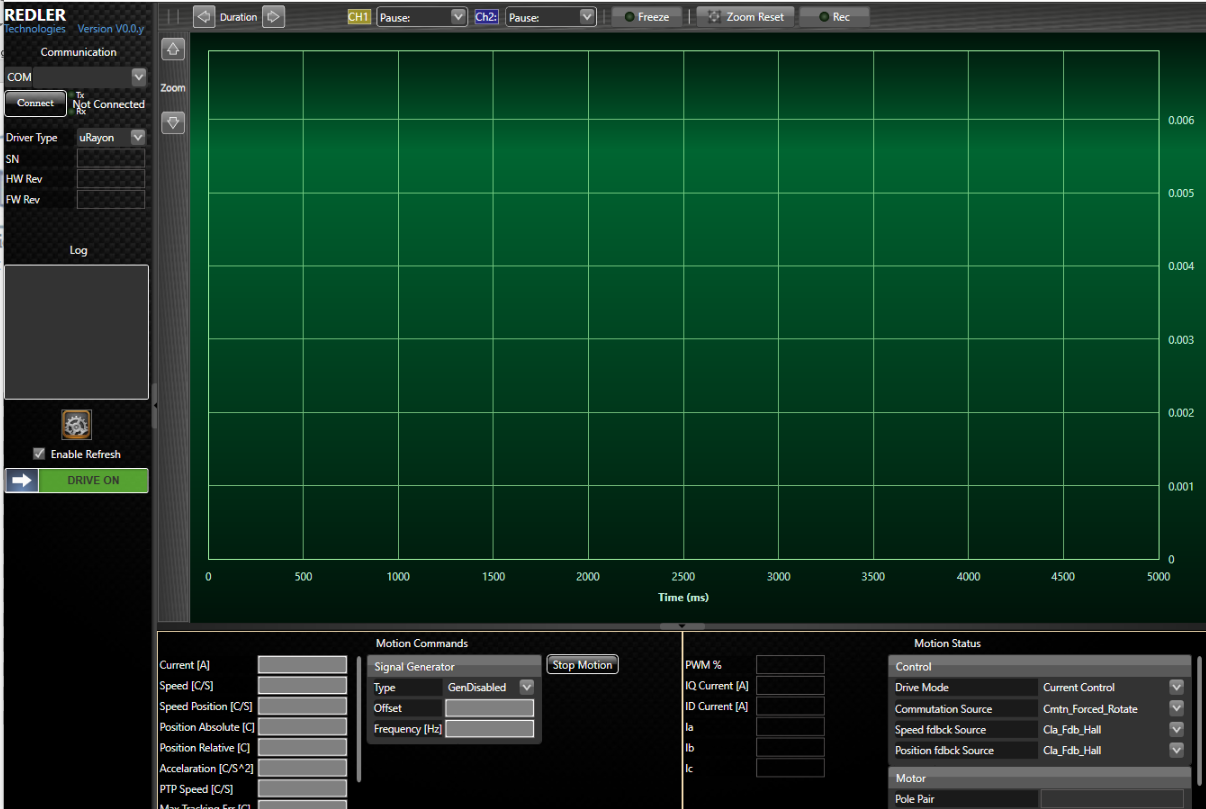
# Motion control examples

## Current control example.

Will be updated

Step 1. On main window Press Drive On.

Step 2. Under “Motion Commands” Set current: 1.0 [A] and press Enter



## Speed control example.

Step 1. Drive On.

Step 2. Send speed command: 1000 [counts/sec]

## Position control example.

Will be updated

# Advance use

## Motion control overview

The rayon contains 4 drive modes:

1. **Current mode:**

Closed loop control of motor current

Command in [Ampere]

PWM

Motor

Current controller

Command [A]

1. **Speed mode:**

Close loop control of motor speed.

Command in [counts/sec] or [rpm]

PWM

Motor

Current controller

[A]

Speed controller

Speed [c/s]

1. **Position mode:**

Close loop control of motor position.

Command in [counts].

PWM

Motor

Current controller

[A]

Speed controller

[c/s]

Position controller

Position [c]

1. **~~Stepping mode:~~**

~~Controls both amplitude [Ampere] and speed [Hz] of electrical stator field of 3 phase motor. No feedback needed.~~

PWM

Motor

Current controller

[A] + [Hz]

## The Current Drive controller

### Control algorithm

To control motor currents, the driver uses Field oriented control algorithm with space vector generator and PI closed loop. In this method the active and reactive current are controlled and the efficiency is rising, the ripple currents "torque ripple" are nonexistence so motor will have smooth, step less movement, especially in low speeds

All the functionality of the current controller is active in Speed, Position and Vector drive loops.

### Control blocks

Analog command

PWM command

Digital command

Upper level command

Simplified current controller

Current Limiter

PI

Space vector

3Phase PWM

Command

Motor

Motor Current Feedback

FOC

### Sinusoidal waveforms of current command

The current controller uses field-oriented vector control algorithm, inputs of this controller is IQ **command** Amplitude in[A], the driver calculate electricalAnglefrom encode or hall andmeasure phase currentsfor closed PI control loop.

Ia = command x Sin(angle)

Ib = command x Sin(angle+120)

Ic = command x Sin(angle+240)

Figure 1: IQ commands and phase currents

### Rotor electrical angle

The rotor electrical angle is calculated in real time from motor feedback and parameters.

Angle resolution and max current according commutation modes:

* **Sin-wave commutation:**

In sin-wave commutation we have the full encoder resolution for commutation.

Electrical angle resolution: (Encoder lines/pole pair).

Imax=command Amplitude

* **Six step commutation:**

In six-stem commutation we have 6 steps for every pole of the motor.

Electrical angle resolution: 60-degree steps [0, 60,120,180,240,300]

Imax=command x sin (120) = command x 0.866

* **DC motor:**

Current flow: Phase A to Phase B, alpha = {120}

Current flow: Phase B to Phase A, alpha = {240}

Imax=command x sin (120) = command x 0.866

### Motor Kt

**Kt[Nm/A]** usually defined with trapezoid drive, this means current is flowing through 2 phases only. In case of sinusoidal vector control this happens when angle = 60,120,180 etc. if the command is 1A then currents will be only 0.866: Ia=0.866[A], Ib=0.866[A], Ic=0.

Thus: **Kt sinusoidal = (Kt trapezoid) \*0.866**

The user should set higher current command and limits by about 15% (1.1547) to compensate this definition difference.

### PWM frequency:

PWM frequency of 20KHz (5Khz-100Khz) optional by request.

### PI loop:

Runs at 20 KHz.

The current PID is auto-calibrated from within the rayon driver, this releases a lot of time-consuming work from the end user, this calibration is done by applying PWM signal on motor phases and measuring results, driver calculate KP and Ki to give constant high-end performance result.

User have access to adjust Kp and Ki parameters to optimize this result manually.

### Current limit monitor:

The driver will limit continues current with Over-Drive capability.

With Over Drive user can set high peak motor current for a specific time to overcome accelerations and disturbances, while keeping all thermal conditions and limitation of motor / driver.

### Command source.

Available command input:

1. Digital command: RS232/CAN from PC.
2. Analog command: -10V to 10V.
3. PWM command: from external controller or rayon driver.
4. Lowlevel command: From speed or vector drive loop.

Only one of the above can be used at the same time.

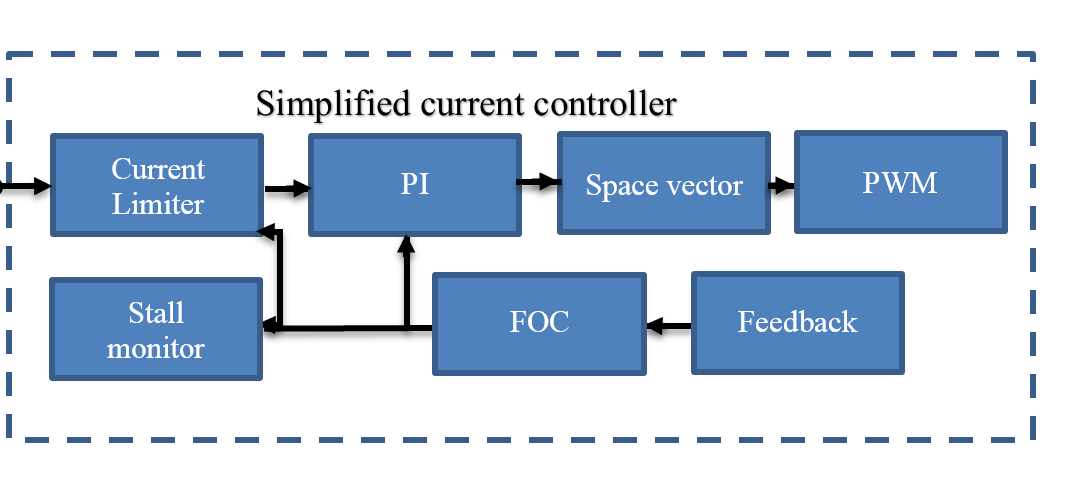
## The Speed Drive controller

The speed controller is built on the top of the current controller.

It offers speed control with Profiled acceleration, deceleration ramps.

Digital Speed command

Profiled motion



PID

Speed controller

Analog Speed

PWM

Speed

Position

Speed

**Command source:**

Analog, digital, PWM or output from Position loop.

**PID loop:**

Runs at 4 KHz.

The speed PID is auto-calibrated from within the rayon driver.

Over/under speedprotection, the drive will shut down reports failure in Error Register.

**Profiled Motion:**

Option to select PID Motion or Controlled Motion with Acceleration and deceleration.

**Stop manager.**

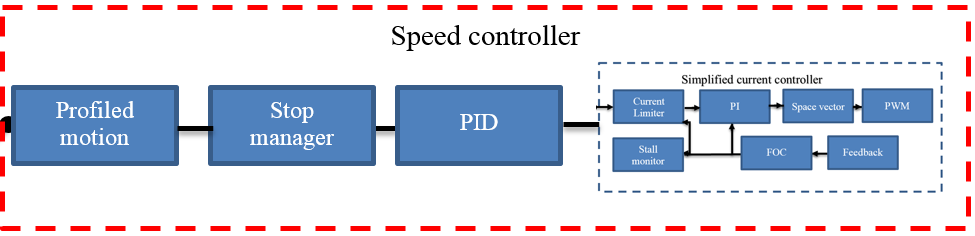
Stop the motor gently with deceleration according to Stop managerparameters**.**

## The position Drive controller

The position controller is built on the top of the speed controller.

It offers position tracking, Error monitor, Profiler and Feed Forward.

The controller also supports Dual Encoder for improved accuracy without backlash errors, the position range) count range) for each encoder, defined by modulo parameters, can be used for error monitor in case of encoders mismatch.

****

Digital position command

Profiler

PID

Feed Forward

Stepper position command

Motor

Position

External

Position

**PID loop:**

Runs at 2 KHz.

The position PID is auto-calibrated from within the rayon driver.

Over/under Position protection, the drive will shut down reports failure in Error Register.

**Profiled Position:**

Option to select PID constants and speed profiler according range to target.

**Position Feedback:**

Select Motor/external encoder**.**

**Command source:**

Digital or stepper incremental command via PWM command pin.

## Current Limits.

Will be updated

## Protections.

Will be updated

Startup tests:

1. Program flash checksum
2. Parameters checksum.
3. Analog to digital current sensors test.

Motor On tests:

1. Stall - motor stall.
2. Hall monitor - In event of Hall error.
3. Over Temperature – over temperature of motor or driver
4. Over/under voltage – bus voltage of motor
5. Speed tacking error (only with trapezoid profiler)
6. Position tracking error (only with trapezoid profiler)

The drive will shut down to protect the motor and driver, reports failure in Error Register,

## Power considerations.

Will be updated

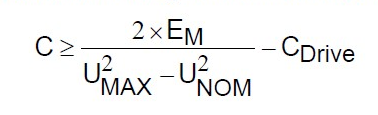
## Thermal considerations.

Will be updated

## Inertia considerations.

Under heavy decelerations such motor break or rapid direction change, kinetic energy of motor inertia and load inertia, will be converted and flow back to the power supply, depending on the power supply characteristic, the voltage my rise, over-voltage protection may be triggered, and the drive power stage will be disabled. This voltage rise may affect or even damage system or power supply

Several options to solve this issue:

1. Use battery as motor power supply, battery should accept the reverse current charge, mean charge 50% of max charge, it is possible to connect PSU with the **exact** battery voltage (constant voltage) to charge and supply the additional current when needed by heavy loads.
2. Add dedicated R-generation protection HW module parallel to power supply inputs, Watt ratings should be enough to absorb the overall energy flowing back to the supply.
3. Add capacitor- voltage rating equal or bigger than the maximum expected over-voltage and can be sized with the formula:

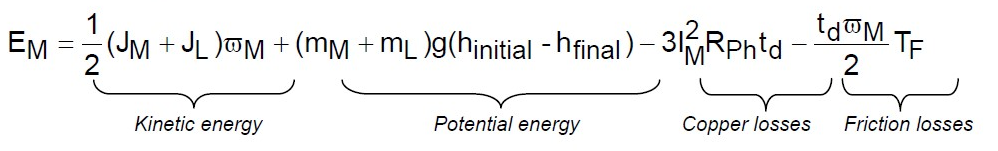
where:

Umax - is the over-voltage protection limit expressed in [V] (see spec).

CDrive - is the drive internal capacitance (see spec)

UNOM – nominal working supply voltage expressed in [V].

EM - the overall energy flowing back to the supply in Joules. In case of a rotary motor

and load, EM can be computed with the formula:

where:

JM – total rotor inertia [kgm2]

JL – total load inertia as seen at motor shaft after transmission [kgm2]

ϖM – motor angular speed before deceleration [rad/s]

mM – motor mass [kg] – when motor is moving in a non-horizontal plane

mL – load mass [kg] – when load is moving in a non-horizontal plane

g – gravitational acceleration i.e. 9.8 [m/s2]

h**initial** – initial system altitude [m]

h**final** – final system altitude [m]

IM – motor current during deceleration [ARMS/phase]

RPh – motor phase resistance [Ω]

t**d** – time to decelerate [s]

TF – total friction torque as seen at motor shaft [Nm] – includes load and transmission

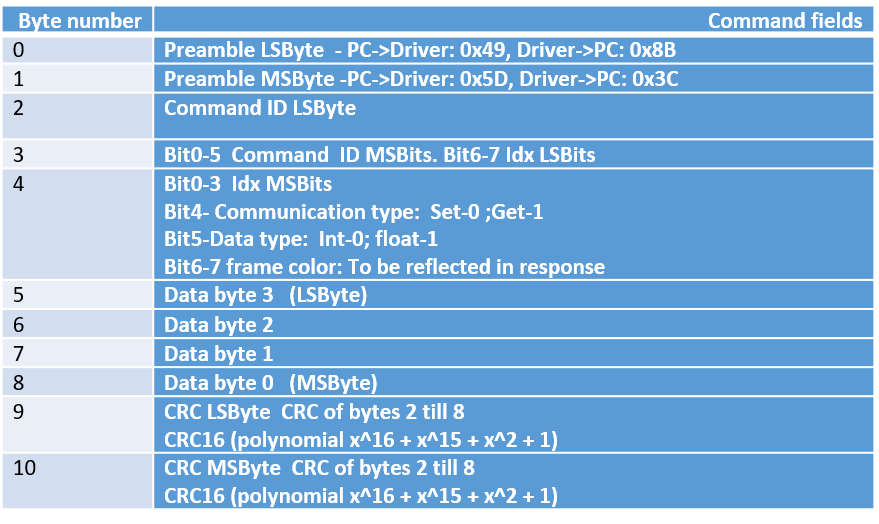
# Communication.

## Serial Protocol:

### Massage structure – Get command

7 Bytes, Request data from the driver.

### Massage structure – Data command

11 Bytes, Send command with 4 Byte of data to driver.

### Massage structure – Data response

Data response to “Get Command” Structure is the same as Data command, Response will be reflection of command (bytes 2-4 will stay the same) Data and CRC bytes (Bytes 5-10) will be updated with new values.

### CRC calculations code:

// Table for cached 4-bit CRC-16 lookups

**static** **const** **unsigned** **short** crctable[16] = {

0x0000, 0xCC01, 0xD801, 0x1400, 0xF001, 0x3C00, 0x2800, 0xE401,

0xA001, 0x6C00, 0x7800, 0xB401, 0x5000, 0x9C01, 0x8801, 0x4400 };

**Function:** crc16

**Arguments:**

Uint16 crc - Previous CRC value

Uint8 const \*buffer - Data pointer

size\_t len - Number of bytes in the buffer

**Returns:**

Uint16 - Updated CRC value

**Description:**

This function calculates CRC16 (polynomial x^16 + x^15 + x^2 + 1)

cyclic redundancy check of a data buffer.

**unsigned** **short** **crc16** (**unsigned** **char** **const** \*buffer, size\_t len)

{

**unsigned** **short** crc=0;

**while** (len--)

{

// CRC the lower 4 bits

crc = (crc >> 4) ^ crctable[((crc ^ (\*buffer & 0xF)) & 0xF)];

// CRC the upper 4 bits

crc = (crc >> 4) ^ crctable[((crc ^ (\*buffer >> 4)) & 0xF)];

// Move on to the next element

buffer++;

}

**return** crc; // Return the cumulative CRC-16 value

}

* Notes CRC calculation without Preamble Byte 0 and Byte 1

For 11 Byte Data structure => CRC size\_t len = 7

For 7 Byte get structure => CRC size\_t len = 3

### Short presantation in this document

For easy reading and short representation, each command will be described with short text structure.

**Text structures:**

1. **Command presentation:** {ID[Idx],type}

Example “Motor On/Off” Command: {1[0],int}

1. **Get status:** {Get, ID[Idx], type}

Example get “Motor On/Off”: {Get,1[0],int}

1. **Set value:** { Set, ID[Idx[=”Data”, type}

Example set “Motor On”: {Set, 1[0] =1, int}.

### Get example

Example of “GET\_VER\_ CMD” message sent from external PC:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Command | ID | IDX |  | Data | Type | Set/Get |
| “Get ver” | 1 | 0 |  | 201188 | Integer | get |

* Get Command from pc: Get 62[3] int.

|  |  |  |
| --- | --- | --- |
| Command fields | Data Hex | Byte number |
| Preamble LSByte | 49 | 0 |
| Preamble MSByte | 5D | 1 |
| Command LSByte (Cmd-ID) | 3E | 2 |
| Bit0-5 Command(Cmd-ID)MSBits. Bit6-7 sub command(Idx) LSBits | C0 | 3 |
| Bit0-3 sub command(Idx) MSBits  Bit4- Communication type Set-0 ;Get-1  Bit5-Data type Int-0; float-1  Bit6-Error No error -0 Error -1  Bit7- Response. Command-0 Response -1 | 10 | 4 |
| CRC LSByte CRC of bytes 2 till 8  CRC16 (polynomial x^16 + x^15 + x^2 + 1) | 30 | 5 |
| CRC MSByte CRC of bytes 2 till 8  CRC16 (polynomial x^16 + x^15 + x^2 + 1) | 0 | 6 |

* Response Data from Motor controller: 62[3]= 20118 int.

|  |  |  |
| --- | --- | --- |
| Command fields | Data | Byte number |
| Preamble LSByte | 8B | 0 |
| Preamble MSByte | 3C | 1 |
| Command LSByte (Cmd-ID) | 3E | 2 |
| Bit0-5 Command(Cmd-ID) MSBits. Bit6-7 sub command(Idx) LSBits | C0 | 3 |
| Bit0-3 sub command(Idx) MSBits  Bit4- Communication type Set-0 ;Get-1  Bit5-Data type Int-0; float-1  Bit6-Error No error -0 Error -1  Bit7- Response. Command-0 Response -1 | 10 | 4 |
| Data byte 3 (LSByte) | 96 | 5 |
| Data byte 2 | 4E | 6 |
| Data byte 1 | 00 | 7 |
| Data byte 0 (MSByte) | 00 | 8 |
| CRC LSByte CRC of bytes 2 till 8  CRC16 (polynomial x^16 + x^15 + x^2 + 1) | 42 | 9 |
| CRC MSByte CRC of bytes 2 till 8  CRC16 (polynomial x^16 + x^15 + x^2 + 1) | 9F | 10 |

### Set example “On/Off Command”

Example of Motor Controller=“ON” message sent from external PC to Motor Controller:

* Set Command from pc: set, 1[0]=1, Int

|  |  |  |
| --- | --- | --- |
| Command fields | Data Hex | Byte number |
| Preamble LSByte | 49 | 0 |
| Preamble MSByte | 5D | 1 |
| Command LSByte (Cmd-ID) | 01 | 2 |
| Bit0-5 Command(Cmd-ID)MSBits. Bit6-7 sub command(Idx) LSBits | 00 | 3 |
| Bit0-3 sub command(Idx) MSBits  Bit4- Communication type Set-0 ;Get-1  Bit5-Data type Int-0; float-1  Bit6-Error No error -0 Error -1  Bit7- Response. Command-0 Response -1 | 00 | 4 |
| Data byte 3 (LSByte) | 01 | 5 |
| Data byte 2 | 00 | 6 |
| Data byte 1 | 00 | 7 |
| Data byte 0 (MSByte) | 00 | 8 |
| CRC LSByte CRC of bytes 2 till 8  CRC16 (polynomial x^16 + x^15 + x^2 + 1) | 11 | 9 |
| CRC MSByte CRC of bytes 2 till 8  CRC16 (polynomial x^16 + x^15 + x^2 + 1) | 3C | 10 |

### Timing

* Time from power up to response > 500ms.
* Inter byte timing limitation < 10ms (this is also the timeout for incomplete / corrupted message).
* Minimum time between two messages to the driver ≥ 1ms
* Reaction time from last byte of message received < 1ms

### Requirements

* When no response received – send a new request.
* Bad message (corrupted) is:
* A message with **Bad checksum** shall be ignored.
* A message with **Bad Header** shall be ignored.
* A message with **Undefined ID** shall be ignored.
* If from any reason (describes in Paragraph (2) ) the device received bad command, the external device (client side) should send again. Nothing need to be done except this reaction of message resend.
* Communication fail shall be set in the Requester side (external side) when three consecutive failures occurred (as described in #3 above “Bad message”).

### Error massages

If Command cannot be executed for some reason. drive send Error massage as response.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Command | ID | IDX | Data | Type |
| Error massage | 100 | 0 | See 9.5 “Error table” | Integer |

## CanOpen protocol

Will be updated

# Commands descriptions

This chapter will give detailed information about basic communication commands.

(will be updated “Rayon Command rev 1.0”)

## Control commands group

This section describes all motor control commands, to control actual movement of motor, all of them are active only when Driver is enabled (with the exception of Motor on command)

### Example command

**Description:**

**Command protocol:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Command name | ID | IDX | units | Direction | Data Range | Data type |
|  |  |  |  |  |  |  |

**Values:**

**Limitations:**

**Related commands:**

### Enable Drive

**Description:**

Enable the PWM to the Inverter power stage.

Before any motor command for movement, the user should “Enable Drive”.

**Command protocol:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Command name | ID | IDX | units | Direction | Data type |
| Enable Drive | 1 | 0 | - | Set/Get | integer |

**Values:**

1 to Enable Drive

0 to Disable Drive

**Sub commands:**

**1[1]** - stop motion command, stops the motor according to Acceleration parameter.

**Limitations:**

none

### Stop motion

**Description:**

The stop motion command stops the motor according to Acceleration parameter (same as sending zero command in Speed Drive).

**Command protocol:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Command name | ID | IDX | units | Direction | Data type |
| Stop Motion | 1 | 1 | - | Set | integer |

**Values:**

1 to enable

**Limitations:**

Driver must be enabled

**Related commands:**

Acceleration.

### Current command

**Description:**

Current IQ command in [A]. in “Speed/Position Drive” this command automatically switches to “Current Drive”.

**Command protocol:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Command name | ID | IDX | units | Direction | Data type |
| Current | 3 | 1 | Ampere | Set/Get | Float |

**Values:**

Accepts current IQ command in [A] within current limits.

**Limitations:**

Driver must be enabled.

**Sub commands:**

**3[4]** - Current command in ampere with Speed feedback 25[0] as response.

**3[5]** - Current command in ampere with Position feedback 26[0] as response.

**Related commands:**

Current limits, Peak limits.

### Speed command

**Description:**

Controls system speed according main feedback source: hall or encoder.

Top speed is limited by parameters velocity limit Hi/Low parameters, and by mechanical and electrical properties like motor Kv (Rpm/V), supply voltage, load and current limits.

if trapezoid profiler selected, motor will accelerate according acceleration profile.

**Command protocol:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Command name | ID | IDX | units | Direction | Data type |
| Speed command | 4 | 0 | [c/s] | Get/Set | Integer 32 |
| Speed command | 4 | 10 | rpm | Get/Set | Integer 32 |

**Values:**

Speed limit Low 53[2] to Speed limit High 53[1]

**Limitations:**

Motor must be enabled.

**Related commands:**

velocity limit Hi/Low, Profiler mode, Acceleration

### Position

**Description:**

Position absolute 5[0] - Set Position target value, the motor will move to short distance related to Roll High/Low parameters.

Position absolute 5[1] – Set target value relative to current main position feedback (26[0])

Position Speed 5[2] – Speed command to profiled motion only, position profile will be used to control speed, helpful for smooth low speeds with low encoder resolution, is also useful for its position track error monitor.

If profiled motion is selected, motion motor will move while keeping PTP speed and acceleration limits, it is also possible to send command as reference directly to PI position loop (PI profiler).

Maximum position will be limited to Max position and Min position. maximum speed will be limited by Max Speed parameter (speed PI loop).

**Command protocol:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Command name | ID | IDX | units | Direction | Data type |
| Position absolute | 5 | 0 | Counts | Get/Set | integer |
| Position relative | 5 | 1 | Counts | Get/Set | integer |
| Speed position | 5 | 2 | Counts/Sec | Get/Set | integer |

**Values:**

Accepts position target within main counter Roll High/Low range.

**Limitations:**

Driver must be on.

**Related commands:**

Roll High/Low, Max speed, PTP speed, Acceleration, max tracking error, Max/Min position.

# Commands table summery.

List of all commands sorted by ID number. Unless otherwise specified all the command returns echo as response.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Command | Description | ID | IDX | Type |
| Enable Drive | Enable/Disable Motor Controller | 1 | 0 | Integer |
| Stop Motion | Stop motor (keep Enable drive) | 1 | 1 | Integer |
| Current Command | current command [A] | 3 | 0 | float |
| Current Command Spd Fdb | response to set command will be main speed feedback 25[0] | 3 | 4 | float |
| Current Command Pos Fdb | response to set command will be main position feedback 26[0] | 3 | 5 | float |
| Speed command | Velocity command in [counts/sec] | 4 | 0 | Integer |
| Speed Ref Rpm | Velocity command in [Rpm] | 4 | 10 | Integer |
| Speed Ref Counts Spd Fdb | response to set command will be main speed feedback 25[0] | 4 | 4 | Integer |
| Speed Ref Rpm Pos Fdb | response to set command will be main position feedback 26[0] | 4 | 5 | Integer |
| Position Absolute | Main feedback position command [counts] | 5 | 0 | Integer |
| Position Relative | relative position command + Main feedback as reference | 5 | 1 | Integer |
| Speed Position | Speed with profiled motion | 5 | 2 | Integer |
| Position Absolute Spd Fdb | response to set command will be main speed feedback 25[0] | 5 | 4 | Integer |
| Position Absolute Pos Fdb | response to set command will be main position feedback 26[0] | 5 | 5 | Integer |
| Speed Position Spd Fdb | response to set command will be main speed feedback 25[0] | 5 | 6 | Integer |
| Speed Position Pos Fdb | response to set command will be main position feedback 26[0] | 5 | 7 | Integer |
| Current Offset calib cmd | Enable calibration | 6 | 1 | Integer |
| Current Offset calib status | Calibration status | 6 | 2 | Integer |
| PI current loop calib cmd | Enable calibration | 6 | 3 | Integer |
| PI current loop calib status | Calibration status | 6 | 4 | Integer |
| Hall Map Calib cmd | Enable calibration | 6 | 5 | Integer |
| Hall Map Calibr status | Calibration status | 6 | 6 | Integer |
| encoder Direction calib cmd | Enable calibration | 6 | 7 | Integer |
| encoder Direction calib status | Calibration status | 6 | 8 | Integer |
| PI Speed Calib cmd | Enable calibration | 6 | 9 | Integer |
| PI Speed Calib ststus | Calibration status | 6 | 10 | Integer |
| PI Position Calib cmd | Enable calibration | 6 | 11 | Integer |
| PI Position Calib status | Calibration status | 6 | 12 | Integer |
| abs electrical angle cmd | Enable calibration | 6 | 13 | Integer |
| abs electrical angle state | Calibration status | 6 | 14 | Integer |
| Signal gen Type | Signal generator type | 7 | 1 | Integer |
| Signal gen period | Signal generator internal parameter | 7 | 2 | Integer |
| Signal gen count | Signal generator internal parameter | 7 | 3 | Integer |
| Signal gen amplitude | Signal generator command amplitude | 7 | 4 | Integer |
| Signal gen Offset | Signal generator command Offset | 7 | 5 | float |
| Signal gen frequency | Signal generator command frequency | 7 | 6 | Integer |
| Power Out1 | Bus-voltage switched output for up to 2[A] load (break) | 12 | 1 | Integer |
| Main speed Fdb |  | 25 | 0 | Integer |
| Hall Lpf Count Sec |  | 25 | 1 | Integer |
| Enc1 Lpf Count Sec |  | 25 | 2 | Integer |
| Enc2 Lpf Count Sec |  | 25 | 3 | Integer |
| Main position Fdb |  | 26 | 0 | Integer |
| Hal Position Counter |  | 26 | 1 | Integer |
| Enc1 Position Counter |  | 26 | 2 | Integer |
| Enc2 Position Counter |  | 26 | 3 | Integer |
| IQ Current |  | 30 | 0 | float |
| ID Current |  | 30 | 1 | float |
| PWM |  | 30 | 2 | float |
| Ia |  | 30 | 10 | float |
| Ib |  | 30 | 11 | float |
| Ic |  | 30 | 12 | float |
| Temperatures |  | 32 | 1 | float |
| Drv status |  | 33 | 0 | Integer |
| Driver Failures |  | 33 | 1 | Integer |
| analog input |  | 33 | 5 | Integer |
| stat\_reg1 |  | 33 | 11 | float |
| stat\_reg2 |  | 33 | 12 | Integer |
| cntrl\_reg1 |  | 33 | 13 | Integer |
| cntrl\_reg2 |  | 33 | 14 | Integer |
| cntrl reg1 |  | 33 | 21 | Integer |
| cntrl reg2 |  | 33 | 22 | Integer |
| Read all inputs |  | 29 | 0 | Integer |
| Input 1 |  | 29 | 1 | Integer |
| Input 2 |  | 29 | 2 | Integer |
| Input 3 |  | 29 | 3 | Integer |
| Input 4 |  | 29 | 4 | Integer |
| Read plot table Length |  | 34 |  | Integer |
| read table param (name, type, unit) |  | 35 | 1-table Length | Integer |
| read full scale table |  | 36 | 1-table Length | Integer |
| startupEnable |  | 50 | 0 | Integer |
| Drive Mode |  | 50 | 1 | Integer |
| Commutation Source |  | 50 | 2 | Integer |
| Speed Fdb Source |  | 50 | 3 | Integer |
| Position Fdb Source |  | 50 | 4 | Integer |
| Command Ref Source |  | 50 | 5 | Integer |
| Pole Pair |  | 51 | 1 | Integer |
| Direction |  | 51 | 2 | Integer |
| resistance |  | 51 | 3 | Integer |
| Inductance |  | 51 | 4 | Integer |
| Kv |  | 51 | 5 | Integer |
| Peak current (friction) |  | 51 | 6 | Integer |
| Peak current (interia) |  | 51 | 7 | Integer |
| Continuous Current Limit |  | 52 | 1 | float |
| Peak Current Limit |  | 52 | 2 | float |
| Peak Time |  | 52 | 3 | float |
| PWM limit |  | 52 | 4 | float |
| Max speed |  | 53 | 1 | Integer |
| Min Speed |  | 53 | 2 | Integer |
| Max position [Counts] |  | 53 | 3 | Integer |
| Min Position [Counts] |  | 53 | 4 | Integer |
| Enable Position Limit |  | 53 | 5 | Integer |
| Motor stuck current | 0 disabled | 53 | 6 | float |
| Motor stuck speed |  | 53 | 7 | Integer |
| Motor stuck Duration |  | 53 | 8 | float |
| Profiler Mode |  | 54 | 1 | Integer |
| Max Velocity |  | 54 | 2 | Integer |
| Max Acceleration |  | 54 | 3 | Integer |
| Max Jerk |  | 54 | 4 | Integer |
| jerk Div |  | 54 | 5 | Integer |
| max tracking error |  | 54 | 6 | Integer |
| home Speed |  | 55 | 1 | Integer |
| Travel Max |  | 55 | 2 | Integer |
| TravelMin |  | 55 | 3 | Integer |
| stallCurrent |  | 55 | 4 | float |
| stallspeed |  | 55 | 5 | Integer |
| stallTime |  | 55 | 6 | float |
| startPosition |  | 55 | 7 | Integer |
| Channel 1 |  | 60 | 1 | Integer |
| Channel 2 |  | 60 | 1 | Integer |
| baud Rate | Rs232 baud rate: 19200=2, 38400=3, 57600=4, 115200=5, Rate\_230400=6, Rate\_460800=7, Rate\_921600=8 | 61 | 1 | Integer |
| buff Cmd Rate |  | 61 | 2 | Integer |
| Driver Type |  | 62 | 0 | Integer |
| Serial Num |  | 62 | 1 | Integer |
| Hw Rev |  | 62 | 2 | Integer |
| Firmware rev |  | 62 | 3 | Integer |
| loader desc version |  | 62 | 4 | Integer |
| system ID 1 |  | 62 | 5 | Integer |
| systemI ID2 |  | 62 | 6 | Integer |
| systemI ID3 |  | 62 | 7 | Integer |
| CanNodeID |  | 62 | 8 | Integer |
| flashChecksum |  | 62 | 10 | Integer |
| Save |  | 63 | 0 | Integer |
| Load Manucfture defualts |  | 63 | 1 | Integer |
| Reboot Driver |  | 63 | 2 | Integer |
| reset driver |  | 62 | 9 | Integer |
| Enable Protected Write |  | 63 | 10 | Integer |
| Flash Checsum |  | 63 | 11 | Integer |
| Cmd Uart Synch Device |  | 64 |  | Integer |
| Enable Loader |  | 65 | 0 | Integer |
| full Scale Ampre |  | 66 | 0 | Integer |
| full scale current voltage float |  | 66 | 1 | Integer |
| offset Store |  | 67 | 1 | Integer |
| Start/Stop \*from\* File |  | 67 | 2 | Integer |
| send data |  | 67 | 3 | Integer |
| Send checksum |  | 67 | 4 | Integer |
| start/Stop Store \*to\* File |  | 67 | 12 | Integer |
| Get data \*from\* file |  | 67 | 13 | Integer |
| every Idx represent different plot signal |  | 68 | 1-n | Integer |
| enable |  | 70 | 1 | Integer |
| Roll up |  | 70 | 2 | Integer |
| Sample Period |  | 70 | 3 | Integer |
| Direction |  | 70 | 4 | Integer |
| Counts Per Rev |  | 70 | 5 | Integer |
| Speed LPF Cut-Off |  | 70 | 6 | float |
| Hall Angle 1 |  | 70 | 7 | float |
| Hall Angle 2 |  | 70 | 8 | float |
| Hall Angle 3 |  | 70 | 9 | float |
| Hall Angle 4 |  | 70 | 10 | float |
| Hall Angle 5 |  | 70 | 11 | float |
| Hall Angle 6 |  | 70 | 12 | float |
| Enable |  | 71 | 1 | Integer |
| Roll High |  | 71 | 2 | Integer |
| Roll Low |  | 71 | 3 | Integer |
| Direction |  | 71 | 4 | Integer |
| Counts Per Rev |  | 71 | 5 | Integer |
| Speed LPF |  | 71 | 6 | float |
| index mode |  | 71 | 7 | Integer |
| Index Reset |  | 71 | 8 | Integer |
| Reset Value |  | 71 | 9 | Integer |
| Set Position value |  | 71 | 13 | Integer |
| Resolution Sin Cos |  | 71 | 14 | Integer |
| SamplePeriod |  | 71 | 15 | Integer |
| Enable |  | 72 | 1 | Integer |
| Roll High |  | 72 | 2 | Integer |
| Roll Low |  | 72 | 3 | Integer |
| Sample Period |  | 72 | 4 | Integer |
| Direction |  | 72 | 5 | Integer |
| Counts Per Rev |  | 72 | 6 | Integer |
| Speed LPF Cut-Off |  | 72 | 7 | float |
| Index Mode |  | 72 | 8 | Integer |
| Index Value |  | 72 | 9 | Integer |
| IEI |  | 72 | 10 | Integer |
| idx Value |  | 72 | 13 | Integer |
| Encoder Resoultion |  | 72 | 14 | Integer |
| Enable |  | 73 | 1 | Integer |
| Roll High |  | 73 | 2 | Integer |
| Sample Period |  | 73 | 3 | Integer |
| Direction |  | 73 | 4 | Integer |
| Counts Per Rev |  | 73 | 5 | Integer |
| Speed LPF Cut-Off |  | 73 | 6 | float |
| Baud Rate |  | 74 | 1 | Integer |
| Char Length |  | 74 | 2 | Integer |
| Polarity |  | 74 | 3 | Integer |
| Phase |  | 74 | 4 | Integer |
| Calibrated Angle |  | 74 | 5 | float |
| Enable External Encoder interpolation |  | 78 | 1 | Integer |
| Interpolation Gear IQ15 |  | 78 | 2 | float |
| Max Encoder-Hall Sync Error [deg] |  | 78 | 3 | float |
| Ia |  | 80 | 1 | float |
| Ib |  | 80 | 2 | float |
| Ic |  | 80 | 3 | float |
| Kp |  | 81 | 1 | float |
| Ki |  | 81 | 2 | float |
| Kc |  | 81 | 3 | float |
| Kp1 |  | 82 | 1 | float |
| Ki |  | 82 | 2 | float |
| Kc |  | 82 | 3 | float |
| Kd |  | 82 | 4 | float |
| Kp2 |  | 82 | 5 | float |
| Kp range |  | 82 | 6 | Integer |
| Kp |  | 83 | 1 | float |
| Ki |  | 83 | 2 | float |
| Kc |  | 83 | 3 | float |
| Kd |  | 83 | 4 | float |
| Kp |  | 83 | 5 | float |
| Kp range |  | 83 | 6 | Integer |
| H[0] |  | 84 | 1 | Integer |
| H[1] |  | 84 | 2 | Integer |
| H[2] |  | 84 | 3 | Integer |
| H[3] |  | 84 | 4 | Integer |
| H[4] |  | 84 | 5 | Integer |
| H[5] |  | 84 | 6 | Integer |
| analog gain |  | 110 | 0 | Integer |
| analog offset |  | 110 | 3 | Integer |
| CpuMsg.EnableDrive |  | 300 | 0 | Integer |
| CpuMsg.EnableDrive |  | 300 | 1 | Integer |
| Set Absolute Position |  | 301 | 0 | Integer |
| GetAbsolutePosition |  | 302 | 0 | Integer |
| Partial Bit |  | 310 | 0 | Integer |
| Full Bit |  | 311 | 0 | Integer |
| Get HW ID |  | 321 | 0 | Integer |
| Get FW\_ID |  | 321 | 0 | Integer |

# Specifications.

|  |  |  |
| --- | --- | --- |
| Remarks | Parameter | Function |
|  |  | **CPU and Memory** |
| 32bits DSP | TI- TMS320-28069 | CORE |
|  | 256 Kbytes | Flash |
|  | 100 Kbytes | RAM |
|  | 90Mhz | CORE Speed |
|  |  | **Motor Interface** |
| TOP drive PWM | 6 100V@180A | BLDC 3 phases Bridge Drivers |
| 1.4 A turn on 1.8 A turn off | IRS21834S | Bridge Drivers |
| Need suitable heat sink | 70A Continues | Motor Current |
| Open collector/Drain | 60˚ | Hall angel inputs |
| @200mA | 5VDC | Output hall devices power |
| sinusoidal output signals 1VPP | Sin/Cos | Encoder Interface-A |
| CLK A CLK B Index | Digital RS422 | Encoder Interface-B |
| 0-3V 12bits resolution | 2 Channels | Motor Current Sense measurement |
| 256 currents levels. | 2 Channels | Adjustable Motor Current limiter |
|  |  | **General Purpose Interface** |
| Optically isolated | 4 Input 1 Output | Digital Input / Output |
| ±10V or ±5V 12bits resolution | 1 Channel | Analog input (analog command) |
| 2A max | PSU voltage switch for break | Power output (analog command) |
| PWM + Low pass |  | **Communication Interface** |
| CAN 2.0B 1Mbit/Sec | Single CANopen | CAN BUS |
| RS232 or RS422 comm. | 1 Channels | UART |
|  |  | **Power Supply**/**Operating temperature** |
|  | 12V to 75V | PSU Voltage |
| Core @ 100Mhz. | 50mA +Motor current | Current |
| Optional +85˚c | -40˚c +71˚c | Operating Temperature |
|  | -65˚c - +150˚c | Storage Temperature |
|  |  | **I/O Connection** |
| 5W5 Pin straight Solder 40A | NorComp 681M5W5103L001 | Power Input and Phase output Connector |
| Dtype-HD44PIN | Harting 09563517512 | Signals Input and Digital I/O Connector |

# Electrical ICD.

## Electrical block diagram

Will be updated

## Power connector description

### Option 1

Positronics CBD5W5M37S600X 5W5 Male PCB Vertical mate with: Positronics CBD5W5F000E2S solder wire 8 Gauge.

### Option 2

Bare wires directly from PCB, 100mm, #8 or #10.

### power connector signals

|  |  |  |
| --- | --- | --- |
| Remarks | Function | PIN |
| Motor Power supply 12 to 75V | V Motor + | 1 |
| Motor Power Supply Return | V Motor Return | 2 |
| BLDC Motor Phase A | Phase A | 3 |
| BLDC Motor Phase B | Phase B | 4 |
| BLDC Motor Phase C | Phase C | 5 |
| Connect with Back shell to shielded wire. | Earth signal | Chassis |

### Capacitor board

Bus capacitors must be connected, Solderd directly to board or connected in parallel on V\_Motor+ and V\_Motor- wired, close as possible to motor controller, 100mm max total combined wire length from capacitors to motor controller power board.

Voltage and Capacity are depended on applications (22uF per Ampere), recommend range of: 1000uF-4700uF, 63v-100v.

## Communication and feedback connector description

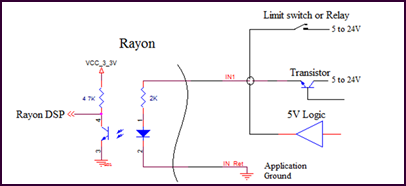
Harting 09563517512. 44HD FEMALE PCB Vertical mate with:

# Harting 09563005601. D-Sub High Density Connectors 44P MALE CRIMP.

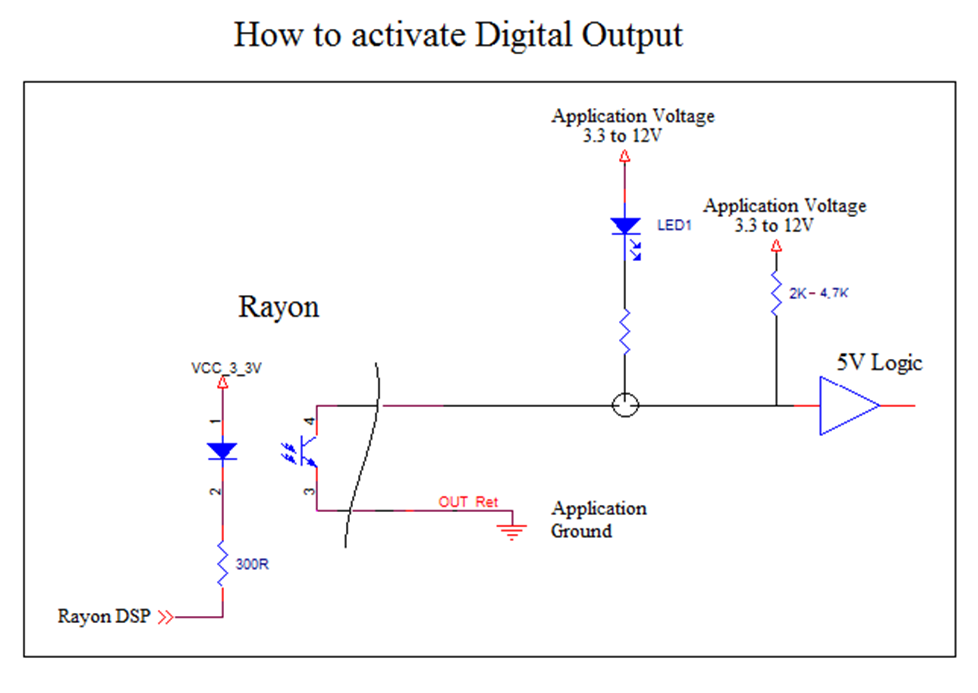
|  |  |  |
| --- | --- | --- |
| Digital and Logic Power Supply Voltage Input. | 12-75V Vin Control + | 1 |
| Digital and Logic Power Supply Voltage Return 0V | 0V Digital Ground | 2,7,16,27 |
| Resolver Cos+/Incremental encoder CLK B | CLK B | 3 |
| Incremental Encoder Index | IDX- | 4 |
| CANBUS H Signal | CAN\_H | 5 |
| ±10V input Analog command Negative side | Analog Command - | 6 |
| Digital and Logic Power Supply Voltage Return 0V | 0V Digital Ground | 7 |
| Open collector Hall 3 sensor input | Hall 3 | 8 |
| Digital absolute Encoder Clock Input - | SPI\_CLK\_422- | 9 |
| Communication Receive signal RS422+ | RX\_422+ | 10 |
| USB Full Speed Data+ / | USB DP / EXC | 11 |
| Master/Slave Direction command | PWM\_DIR\_IO | 12 |
| Optically isolated Discrete input | Input 1 | 13 |
| Optically isolated Discrete output (Open collector) | Output 1 | 14 |
| Short to Digital Ground to download firmware | Boot | 15 |
| Digital and Logic Power Supply Voltage Return 0V | 0V Digital Ground | 16 |
| Resolver sin+/ Incremental encoder CLK A+ (Positive signal) | CLK A | 17 |
| Resolver Cos- / Incremental encoder CLK B- (Negative signal) | CLK\_B- | 18 |
| Data to Rayon (from PC comm. DB9 pin 3) | RS232 RX | 19 |
| CANBUS L Signal | CAN\_L | 20 |
| Encoder2 Index (Single ended) | IDX2 | 21 |
| Open collector Hall 1 sensor input | Hall 1 | 22 |
| Do not use. | Spare | 23 |
| Communication Transmit signal RS422+ | TX 422+ | 24 |
| Communication Receive signal RS422- | RX 422- | 25 |
| USB Full Speed Data- | DM / EXCn | 26 |
| Digital and Logic Power Supply Voltage Return 0V | 0V Digital Ground | 27 |
| Optically isolated Discrete input | Input 2 | 28 |
| Optically isolated Discrete output (Open collector) | Output 2 | 29 |
| Motor Thermistor 5K Input | Motor Temperature | 30 |
| Hall and Encoders 5V supply | 5VDC @ 200mA | 31 |
| Resolver sin- /Incremental encoder CLK A- (Negative signal) | CLK An | 32 |
| Incremental Encoder Index + | IDX+ | 33 |
| Data from Rayon (to PC comm. DB9 pin 2) | RS232 TX | 34 |
| ±10V input Analog command positive side | Analog Command + | 35 |
| Hall and Encoders 5V supply | 5VDC @ 200mA | 36 |
| Open collector Hall 2 sensor input | Hall 2 | 37 |
| Digital absolute Encoder Clock Input + | SPI CLK 422+ | 38 |
| Communication Transmit signal RS422- | TX\_422- | 39 |
| USB Full Speed Voltage input (Sense only) | USB VBUS | 40 |
| Master/Slave PWM command | PWM IO | 41 |
| Optically isolated common for Discrete input/Output | IN Ret | 42 |
| Optically isolated motor halt Discrete input | Break | 43 |
| Do not use. | Spare | 44 |

## Digital Inputs and outputs

How to activate a digital input.



How to activate a digital output



# Mechanical ICD.

Will be updated