UART and USB User Manual

SOLO UNO
SOLO MINI
SOLO BETA

SOLO Motor Controllers

Firmware versions supported: 0x0006B00A or later

Revision History:

Revision	Date	Changes
V1.0.0	06/09/2021	- First Release
V1.0.1	30/09/2021	- SOLO MINI added -
V1.0.2	01/04/2022	- Current limit description updated - Analogue Speed Resolution Division Coefficient added - Incremental Encoder Index Counts added - USB and UART time-out added - Position Feedback (Incremental Encoder and Hall sensors) description updated - Position Reference description updated

Contents:

Introduction

Purpose of this user manual

This manual intends to discuss the data communication methods of UART and USB with SOLO UNO, SOLO MINI and SOLO BETA comprehensively.

We will initially start by introduction of major data types which are used for the communications and then explicitly discuss each of the mentioned protocols, eventually the goal of this Manual is to give a clear idea about how to setup and utilize SOLO communication networks powered by UART or USB and what is the purpose of each command existing in our command lists.

If after reading this manual or during your experimentations with our product you had any questions, you can use SOLO Motor Controllers Forum to share with us the questions and get back your answers promptly.

UART and USB Access points on SOLO UNO:

One of the ways of commanding and getting feedbacks from SOLO UNO is using the USB or UART communication protocols, the USB is accessible through a micro USB type-B connector mounted on SOLO UNO and UART is accessible through the “Communication port ” as well as the “UART/CAN bus Pinout” as can be seen below in Figure 1, by using any of these two terminals, you can fully control SOLO in a complete digital fashion with all the commands and feedbacks sent and received through data packets.

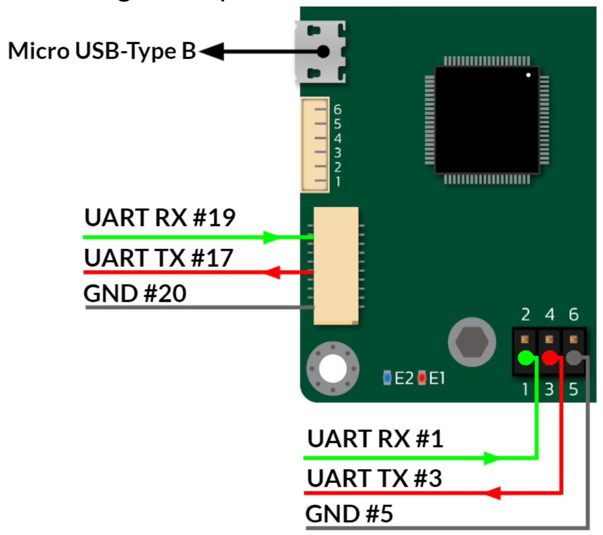

Figure 1- UART and USB access points on SOLO UNO

- <u>**UART TX is not +5V tolerant on SOLO UNO**</u>, as it’s an output pin with signal leveled at 3.3V which is compatible with all Arduino and Raspberry Pi modules, connection of this pin to +5V rated signals will damage the device.
- UART RX is +5V tolerant and it can be fed with signals both leveled at 3.3V or +5V.
- As can be seen in the communication port, Pins 17 and 19 are the TX and RX pins of UART respectively and they have the exact same functionality and ratings as UART_RX and UART_TX pins in “UART/CAN bus Pinout”

- On SOLO BETA models both of the UART_TX and UART_RX pins are 3.3V leveled and they are not +5V tolerant, to use them with +5V leveled systems proper circuitry (voltage dividers) must be utilized.

UART and USB Access points on SOLO MINI:

SOLO MINI, similar to SOLO UNO offers the exact same connectivity for UART and USB communications as shown below in Figure 2 both through its “I/O Port” as well as the “Communication Port”.

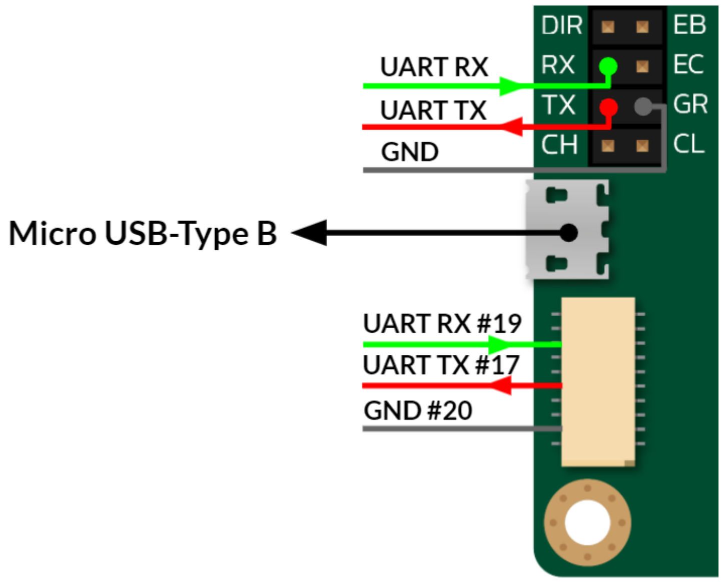

Figure 2- UART and USB access points on SOLO MINI

- **<u>UART TX is not +5V tolerant on SOLO MINI</u>**, as it’s an output pin with signal leveled at 3.3V which is compatible with all Arduino and Raspberry Pi modules, connection of this pin to +5V rated signals will damage the device.
- UART RX is +5V tolerant and it can be fed with signals both leveled at 3.3V or +5V.
- As can be seen in the communication port, Pins 17 and 19 are the TX and RX pins of UART respectively and they have the exact same functionality and ratings as UART_RX and UART_TX pins in “UART/CAN bus Pinout”

UART or USB Hardware Settings:

The proper hardware settings to communicate with SOLO are mentioned in table below:

Parameter	Value
Supported baudrates [bits/s]*	**937500 or 115200**
Number of stop bits	**1**
Character Length bits	**8**
Parity Mode	**None**

- *In Some systems, it might not be possible to use the exact baud-rate of "937500 bits/s", in such cases the user can set the baud-rate of the system at "921600 bits/s" and still the UART or USB communication will be functional and compatible.

- In case of using USB connection, it will be recognized as a **"Virtual COM Port (VCP)"**, and the notion of "Baud-rate" will not be relevant as SOLO has a native USB-2 communication and the data-rate is defined by USB-2 ratings, however you can select any desired baud-rate for this port in VCP mode.

- For Legacy devices it's possible to put the UART baud-rate on 115200 bits/s, this baud-rate will cause SOLO to have slightly reduced performance and it's not recommended if the best performance of SOLO is required specially for fast Brushless Motors.

UART/USB Packets formation _ Commanding and Feedbacks:

To send/receive a command to/from SOLO using UART or USB, you need to form a data packet combined out of **10 bytes**, based on the following format:

INITIATOR	DEVICE ADDRESS	COMMAND	DATA	CRC	ENDING
2 bytes (fixed)	1 byte	1 byte [read/write]	4 bytes [32 bits]	1 byte	1 byte(fixed)
0xFFFF	Variable[0-0xFF]	variable[0-0xFF]	Variable	Variable	0xFE

A complete data packet to be sent/received to/from SOLO is divided into 6 different sections and each of these sections are as below:

- **INITIATOR:** This is a constant valued two byte which indicates the start of a packet, the value is fixed at "0xFFFF '' in Hex format or "65535" in decimal format.
- **DEVICE ADDRESS:** This is a single byte which stands for the address of the device, each SOLO can have an address from 0 to 254 and this address will reside in non-volatile memory and will be remembered after power recycle, this device addressing is useful when you want to put lots of SOLO's in a network, so each of them can be assigned to a unique address. The default value of the address is set at zero.
- **COMMAND:** This is a single byte, which is a fixed code that defines the type of the commands/feedbacks that are sent/received to/from SOLO.
- **DATA:** This is a 4 bytes data, and it's used for sending/receiving the data part of a command/feedback, each DATA can have a different type from Uint32, Int32 or Fixed-Point which are explained later in this manual.
  CRC: This is the CRC byte to control the integrity of the data sent through UART, this functionality is inactive at the moment (it's filled with zero)
- **ENDING:** Similar to the packet initiator, this is again a constant single byte valued at "0xFE" in HEX format, which stands for the end of a packet sent or received.

After sending a full packet to SOLO, three scenarios will happen:

1. If the packet is correctly formed and acceptable, SOLO will echo-back the exact same packet that it received as a sign of acknowledgement of a correct receipt and settings.

2. If the packet is correctly formed, but the value inside the "DATA" part is out of range or the "COMMAND" is not existing or valid , SOLO will send back the following packets for each condition which is an indication of an Error in the packet sent.

Packet Error Format for DATA out of range:

0xFFFF	Device Address	Command sent	0xEEEEEEEE	Variable	0xFE

Packet Error Format for non existing Command code:

0xFFFF	Device Address	Command sent	0xAEEEEEEE	Variable	0xFE

3. If the packet is wrongly formed in terms of bytes, initiators and endings, there will be no response coming back from SOLO, until you send a correct packet.

To communicate with SOLO, there are two types of commands:

1. Commands to Write or Set something, these commands as their name suggests, will allow the user to tune a parameter inside of SOLO or set a value for the controllers like the desired speed or torque and so on, beside putting the right COMMAND code, the DATA section is used to write the desired value in desired register in SOLO.

2. Commands to Read a value from SOLO, using these types of commands, you'll be able to read various types of parameters and feedback from SOLO, in real-time or in Monitoring Mode which will be explained later in this chapter, in these types of commands the DATA section is mostly filled with ZEROs unless specified.

Memory Assignment for Write Commands:

Once a parameter is written into SOLO, depending on the nature of the command two scenarios for storing the value will occur, either it will be stored on volatile memory of SOLO and resets to default after power recycling or it will be saved on non-volatile memory of SOLO and remembered after even power recycling and forever until it's being overwritten or changed.

1. **Volatile parameters**: They are shown with **"V"** inside the Storage section of the writing table, and they are the parameters that you should write in them whenever you want to change them or after power recycling (turn ON/OFF), they will not be stored in a long-term memory and after a power reset they will be forgotten and set back to their default value.

2. **Memory stored Parameters:** These parameters are shown with **"M"** inside their storage section of the writing table, after writing in them, their values will be stored in a non-volatile memory and they will be remembered after power recycling, The memory used to store these values is a precious resource and the number of Writings are limited to a couple of Milion times (1,000,000 times guaranteed). So for these types of values which are basically one-time settings for a long period (as long as the Motor is the same), the users should avoid writing in them everytime if their value has not been changed with respect to the past.

Packet Time-out for UART or USB:

If a packet is sent incompletely and the within the timeout period the packet doesn't get completed or stays distorted SOLO will generate an internal timeout event and will reset its pointer back to zero waiting for new packets to arrive, in below table the timeout for packets sent through UART or USB is shown:

Protocol	Packet time-out
UART	20 milli-seconds
USB	1000 cycles (synched with PWM frequency)

DATA Types:

The data types in SOLO are categorized into three main types as shown in table below:

Name	Value	Size[bits]	Range	Resolution
UINT32	Unsigned Integer	32	[0 to 4,294,967,295]	+/- 1
INT32	Signed Integer	32	[-2,147,483,647 to 2,147,483,647]	+/- 1
Sfxt(32-17)	Fixed Point	32	[-16,384.000 to 16,384.000]	+/- 0.00000762

UINT32:

This data type is used for Unsigned Integer values and it occupies 32 bits.

INT32:

This data type is used for Signed Integer values and it occupies 32 bits.

Sfxt(32-17):

This data type is used to represent the variables with floating point and it occupies 32 bits.

To send and receive commands or feedback properly during communication with SOLO, you need to convert your data into one of these forms based on the command or feedback that you are using, as each command has a specific data type.

In all of these formats, the data section occupies 32 bits or 4 bytes, below you can see how one can convert these data types to tangible numbers from Hexadecimal format that is the default way of sending or receiving data with SOLO, in this manual the Hexadecimal numbers are either shown with “0x” in the beginning of the number or “h” at the end of the number.

DATA Types Conversions:

Converting Sfxt(32-17) data type to floating point data type:

If the DATA section of a packet received from SOLO, contains a number in Fixed-Point format, you can use the following two methods to convert it back to a floating point data type depending on the sign of the received data and whether if it's a positive value or a negative value:

Condition1) If the data read from SOLO is less than or equal to 0x7FFE0000 (Hex) or 2,147,352,576 (decimal), This means the data is positive, so follow the following steps

1) Convert the hex data read from SOLO into Decimal format
2) The float number = data read from SOLO (in Decimal format) / 131072

Condition2) If the data read from SOLO is greater than 0x7FFE0000 (Hex) or 2147352576 (decimal), This means the data is negative, so the conversion will be as :

1) Subtract the data from 0xFFFFFFFF and then add a 0x1 to it (add 1 to it)
2) Convert the result of step "1" into Decimal format
3) The float number = (data in Decimal format taken from step "2" /131072) * -1

Example1 _ positive Numbers:
Data read from SOLO is "0x00030000"
So it's a Positive Numbers Conversion since the data read from SOLO is less than or equal to 0x7FFE0000 so the conversion will be:
1) Decimal (0x00030000) = 196608
2) 196608 / (131072) = 1.5
Example2 _ negative Numbers:
Data read from SOLO is "0xFFFCCDD2"
So it's a Negative Numbers Conversion since the data read from SOLO is greater than 0x7FFE0000, so the conversion will be:

1) (0xFFFFFFFF – 0xFFFCCDD2) + 0x1 = 0x0003322E
2) Decimal (0x0003322E) = 209454
3) (209454 / (131072))* -1 = -1.598

Converting float data type to Sfxt(32-17) data type :

If the DATA section of a packet to be sent to SOLO, contains a number in Fixed-Point format, you can use the following two methods to convert your real world float number into a Sfxt(32-17) data type depending on the sign of the float data and whether if it's a positive value or a negative value:

Condition1) If the float number is Positive:

1- Multiply the float number into 131072
2- Round down the result into the nearest integer value
3- convert this value to HEX

Condition2) If the float number is Negative:

1- Multiply the float number into 131072
2- Round down the result into the nearest integer value
3- Ignore the sign of the integer and convert this value to HEX (compute the Absolute value)
4- Subtract data from "0xFFFFFFFF"

Example 1_ positive float number:
Data to be sent : 4.2
Conversion:

1) 4.2 * 2^17 = 550,502.4
2) Round(550502.4) = 550502
3) Hex (550502) = 0x0086666

Example 2_ negative float number:
Data to be sent : -14.36
Conversion:

1) -14.36 * 2^17 = -1,882,193.92
2) Round(-1,882,193.92) = -1,882,193
3) Hex (abs (-1,882,193))= 0x 001CB851
4) 0xFFFFFFFF - 0x001CB851 = 0xFFE3 47AE

Converting 32 bits Hex data to signed INT32 format:

If you want to convert a DATA part of a packet read from SOLO into the real world Int32 format, based on the sign of the DATA you will fact the following two conditions for the conversion:

Condition1) If the data read from SOLO is less than or equal to 0x7FFFFFFF(Hex) or 2147483647 (decimal), This means the data is positive

When the data is positive, we can treat it like a normal unsigned value, so you can just directly convert the Hex value to decimal with known methods.

Condition2) If the data read from SOLO is greater than 0x7FFFFFFF(Hex) or 2147483647(decimal), This means the data is negative, so the conversion will be as :

1) Subtract the data from 0xFFFFFFFF and then add a 0x1 to it (add 1 to it)
2) Convert the result of step "1" into Decimal format
3) Multiply the result of step 3 to "-1"

Example 1_ positive Numbers:

Data read from SOLO is "0x0003F393"

- The data is smaller than 0x7FFFFFFF so it becomes : Dec (0x0003F393) = +258963

Example 2_ negative Numbers:

Data read from SOLO is "0xFFFCA8AD"

- The data is bigger than 0x7FFFFFFF so we will have:

1. (0xFFFFFFFF - 0xFFFCA8AD) + 1 = 0x00035753
2. Dec(0x00035753) = 218963
3. 218963 * -1 = -218963

Converting signed INT32 to 32bits Hex format:

If you want to convert an INT32 value to a Hex formatted number to place in the DATA part of a packet to send to SOLO, based on the sign of the data you want to send, you will face the following two conditions for the conversion:

Condition1) If the data is positive:

If the number you want to send is positive, the only thing you need to do is converting the integer into HEX like an unsigned value

Condition1) If the data is Negative:

For sending negative Integers with Hex format to SOLO, you need to follow the following steps:

1. Subtract the absolute value of your number from 4294967295(Decimal) or 0xFFFFFFFF(Hex)
2. Add 1 to the result of step 1
3. Convert the result of 2nd step to Hex

Example 1_ positive Numbers:

Data to be sent: 1536

- Hex (1536) = 0x00000600

Example 2_ negative Numbers:

Data to be sent: -56329

- Hex (4294967295 - Abs (-56329) + 1) = 0xFFFF23F7

WRITE Commands:

Below all the existing Write commands to set a value in SOLO with their description for UART or USB communication for are listed.

0x01 : Set Device Address

Code: 0x01	Set Device Address			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0-254]	N/A	M	0
Description: This command sets the desired device address for a SOLO unit; the address can be used to network multiple SOLO's in a single network if the address assigned to each unit is unique.				

0x02 : Commanding Mode

Code: 0x02	Commanding Mode			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	0 or 1	N/A	V	0

Description:
This command sets the mode of the operation of SOLO in terms of operating in Analogue mode or Digital Mode based on the value of DATA in the packet with command code of 0x02 as below:

DATA	Actions
0 (0x00000000)	Puts SOLO in Analogue Mode
1 (0x00000001)	Puts SOLO in Digital Mode

Once in Analogue Mode some configuration can be done only at hardware level, as the table below suggests, everything else outside of the below table can be set only through sending data packets to SOLO through UART, USB or CAN bus.

Action	In Analogue Mode	In Digital Mode
Open-Loop or Closed-Loop Operation	Through PIN 5 of Piano Switch in SOLO UNO Through Control Mode Switch on SOLO MINI	Through PIN 5 of Piano Switch in SOLO UNO Through Control Mode Switch on SOLO MINI
Motor Type selection	Through PIN 1 and 2 of Piano Switch in SOLO UNO Through M1 and M2 Pins on SOLO MINI	Set with command code 0x15
Control Mode selection (Torque, Speed, Position)	Through PIN 4 of Piano Switch (only Torque and Speed) in SOLO UNO Through FN Pin on SOLO MINI	Set with command code 0x16 (Torque, Speed, Position)
DFU mode	Through PIN 3 of Piano Switch in SOLO UNO Through DF Pin on SOLO MINI	Through PIN3 of Piano Switchin SOLO UNO Through DF Pin on SOLO MINI
Current (Torque) controller Kp and Ki Gains in closed-loop mode	Auto-tuned after Motor Identification	Set with Command codes of 0x17 and 0x18 (Auto-tuned after Motor Identification)
Speed controller Kp and Ki Gains in closed-loop Speed mode	Through two physical potentiometers of Kp and Ki on the board	Set with command codes of 0x0A and 0x0B respectively
Speed Reference	Sent by PWM or Analogue voltages through S/T input on Analogue Input port	Set with command code of 0x05
Torque Reference	Sent by PWM or Analogue voltages through S/T input on Analogue Input port	Set with command code of 0x04
Power/ Current Limit / Magnetizing Current	Sent by PWM or Analogue voltages through P/F input on Analogue Input port	Set with command codes of 0x06, 0x03 and 0x1A respectively

0x03 : Current Limit

Code: 0x03	Current Limit			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0 - 32.0]	Amps	M	32

Description:
This command defines the maximum allowed current into the motor in terms of Amps, this command will be effective only once SOLO is in closed-loop digital mode as in analogue mode the current limit is set through "P/F" input pin, In Regeneration Mode, SOLO will limit the current fed back into supply to this value.
By setting the current limit value at zero (both at Analogue or Digital Modes) SOLO will stop the switching at its output and will allow for free-wheeling of the Motor without injection of any current.

0x04 : Torque Reference (Iq/IM)

Code: 0x04	Torque Reference (Iq/IM)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0 - 32.0]	Amps	V	0

Description:
This command sets the amount of desired current that acts in torque generation. In 3-phase motors this is called Iq or quadrature current as SOLO operates in FOC mode, however for DC brushed motors this value sets the reference for IM in DC brushed motors.
This command will be effective only once SOLO is in closed-loop digital Torque mode as in analogue mode the Torque reference is set through the "S/T" input pin once SOLO is in Torque Mode.
For all the motors including DC, BLDC, PMSM and ACIM the value of the torque reference can relate to Torque on the shaft of the motor based on following relation:
Requested Torque [N.m] = Torque Reference [A] x Motor's Torque constant [N.m/A]

0x05 : Speed Reference

Code: 0x05	Speed Reference			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0 - 30,000]	RPM*	V	0

Description:
This command defines the speed reference for SOLO once it's in Digital Speed Mode, as in analogue mode the reference is set through the "S/T" input pin with analogue voltages or PWM pulses once SOLO is in Speed Mode.
*The Speed Unit depends on the type of the Motor as well as the type of operation, and if SOLO is in Open-loop or Closed-loop it can take different meanings as shown below:

Motor Type (Analogue or Digital Mode)	**Closed-loop Sensor-less**	**Closed-loop Sensor-based**	**Open-loop**
BLDC - PMSM	RPM	RPM	RPM
BLDC - PMSM Ultrafast	RPM	RPM	RPM
DC Brushed	Motor Dependant**	RPM	Duty Cycle***
AC Induction Motor	RPM	RPM	RPM

Basically the major consideration will be for DC brushed motors while they are in Sensor-less or Open-loop Mode with the following conditions:

***Speed Unit for DC brushed motor in Closed-loop Sensor-less mode:** This is a qualitative value based on the observer gain defined for the sensorless speed estimator for DC brushed motors, the value can be ranged from 0 to 30,000 and the final speed of the motor for each value depends on the characteristics of the Motor itself like BEMF constant, the increase in the Motor's Speed with respect to the reference will be linear up until the nominal speed of the motor.

*****Speed Unit for DC brushed motor in Open-loop mode:** in This case the speed reference will act as the duty cycle percentage at the output on the Motor, the value can be between 0 to 30,000 which will be mapped into 0% duty cycle to 100% duty cycle, going from no speed to max speed.

0x06 : Power Reference

Code: 0x06	Power Reference			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0 - 100.0]	N/A	V	0

Description:
This command defines the amount of power percentage during only Open-loop mode for 3-phase motors. The value is from 0.0 to 100.0 standing for 0% to 100% output power on the shaft of the Motor.

0x07 : Motor's Parameters Identification

Code: 0x07	Motor Parameters Identification			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	0 or 1	N/A	V	0

Description:
By putting 1 in the DATA section of a packet sent with this command, SOLO will start identifying the electrical parameters of the Motor connected, The identification will take 1 second to be done and after that the Motor Inductance, Resistance and some internal parameters are Identified (or re-identified).
Identification process depends on the type of the motor selected, so before running the Identification the user has to make sure they have properly selected their motor type both in Analogue or Digital Mode.

0x08 : Emergency Stop

Code: 0x08	Emergency Stop			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint 32	0	N/A	V	1

Description:
This command if the DATA is set at zero will stop the whole power and switching system connected to the motor and it will cut the current floating into the Motor from SOLO, by sending this command SOLO should be power recycled to get back into normal operation externally.

0x09 : Output PWM Frequency (switching frequency)

Code: 0x09	Output PWM Frequency (switching frequency)			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[8 - 80]	kHz	M	20 / 80

Description:
This command sets the output switching frequency of the whole power unit on the Motor, SOLO supports switching frequencies from 8 to 80kHz and the user can set their desired frequency with steps of 1kHz.
As a rule of thumb, Higher switching frequencies are necessary for Motors with Low inductance (below 200uH), but the increase or decrease of this value should be done carefully, as increasing the switching frequency is not always good and it can cause saturation and excessive loss for both the magnetic cores of the Motor and the SOLO unit itself if the Motor under control can not support such high frequencies.
The whole internal algorithms including samplings in SOLO are synchronized to the switching frequency.

0x0A : Speed Controller Kp Gain

Code: 0x0A	Speed Controller Kp gain			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0 - 300.0]	N/A	M	0

Description:
This command sets the Speed controller Kp Gain, and it will be functional only in Digital Closed-loop mode, since in Analogue mode this gain is set using the Potentiometer named with “Kp” locally on the board.
This is the proportional gain of the PI controller that SOLO uses to control to stabilize the speed of a motor on a given reference point in close-loop mode, This gain is normally for most motors takes a value in between 0.001 to 0.5 but there might be some cases that you need to go even higher, the user has to increase/decrease these gains with care as they can cause instability for the device under test if not selected properly.

0x0B : Speed Controller Ki Gain

Code: 0x0B	Speed Controller Ki gain			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0 - 300.0]	N/A	M	0

Description:
This command sets the Speed controller Ki gain, and it will be functional only in Digital Closed-loop mode, since in Analogue mode this gain is set using the Potentiometer named with “Ki” locally on the board.
This is the integral gain of the PI controller that SOLO uses to control to stabilize the speed of a motor on a given reference point, This gain is normally for most motors takes a value in between 0.0001 to 0.01 but there might be some cases that you need to go even higher, the user has to increase/decrease these gains with care as they can cause instability for the device under test if not selected properly. In theory, this gain helps the system to have zero steady-state error.

0x0C : Motor’s Direction of Rotation

Code: 0x0C	Motor Direction of Rotation			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	0/1	N/A	V	0

Description:
This commands sets the direction of the rotation of the motor either to ClockWise rotation or to Counter Clockwise Rotation based on the table below,(In sensorless Modes, the order of the wirings of the motor can invert the direction of rotation with respect to table below)

DATA	Desired Rotational Direction
0 (0x00000000)	Counter ClockWise
1 (0x00000001)	ClockWise

0x0D : Motor’s Phase or Armature Resistance

Code: 0x0D	Motor’s Phase or Armature Resistance			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.001 - 50.0]	Ohm	M	0
Description: This command sets the amount of the Phase or Armature resistance for 3-phase or DC Brushed motors respectively. This value is automatically identified by SOLO after Motor Identification but in any case it’s possible for the user to alter the values as they like. After changing these values manually, for them to take effect a power recycle is required.				

0x0E :Motor’s Phase or Armature Inductance

Code: 0x0E	Motor’s Phase or Armature Inductance			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.00005 - 0.2]	Henry	M	0
Description: This command sets the amount of the Phase or Armature Inductance for 3-phase or DC Brushed motors respectively. This value is automatically identified by SOLO after Motor Identification but in any case it’s possible for the user to alter the values as they like. After changing these values manually, for them to take effect a power recycle is required.				

0x0F : Motor’s Number of Poles

Code: 0x0F	Motor’s Number of Poles			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[1-254]	N/A	M	8
Description: This command sets the number of the Poles of a 3-phase motor commissioned with SOLO, in case of BLDC or PMSM motors, the Number of poles is equal to the number of magnets on the rotor of the Motor, for ACIM motors, the user has to refer to the technical datasheet of their motor to find this parameter.				

0x10 : Incremental Encoder’s Lines

Code: 0x10	Incremental Encoder’s Lines			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[1-200,000]	pre-quad	M	1000

Description:
This command sets the pre-quad number of physical lines of an incremental encoder engraved on its disk, in another form it can be seen as the number of pulses generated on 1 line of the Encoder output once it is rotated for 1 exact round.

0x11 : Speed Limit

Code: 0x11	Speed Limit			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0-30,000]	RPM	M	30,000

Description:
This command sets the allowed speed during trajectory following in closed-loop position controlling mode, it can be used to change the speed of a position follower during motion, or it can be fixed on a desired value.

0x12 : Reset the device address to zero

Code: 0x12	Reset the device address to zero			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	N/A	N/A	V	0

Description:
This command resets the device address of any connected SOLO to zero, to properly form this command the user has to put “0xFF” as the Device Address inside the packet being sent to SOLO with DATA filled with zeros as below:
Example packet to reset the device address to zero:
FFFFFF120000000000 FE

0x13 : Feedback Control Mode

Code: 0x13	Feedback Control Mode			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0-2]	N/A	M	0

Description:
This command sets the type of the feedback control SOLO has to operate with based on table below both in Analogue or Digital Mode:

DATA	Operation
0 (0x00000000)	Operates in Sensor-less feedback Mode
1 (0x00000001)	Operates in Incremental Encoder feedback Mode (calibration required)
2 (0x00000002)	Operates in HALL Sensors feedback Mode (calibration required)

0x14 : Reset Factory

Code: 0x14	Reset Factory			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	1	N/A	V	0

Description:
This command resets SOLO to its factory setting to all the default parameters, after this command a power recycle is required so that the default values take effect, the DATA sent with this command to reset the device will be “0x0000001”.

0x15 : Motor Type

Code: 0x15	Motor Type			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0-3]	N/A	M	0

Description:
This command sets the Motor type that is connected to SOLO in Digital Mode based on the following table if the requested DATA is placed in a packet sent with a command address of 0x15, the user has to note that in Analogue Mode the Motor Type is only selected by Piano Switch in SOLO UNO using PIN 1 and 2 and on SOLO MINI it's selected by M1 and M2 pins on the "I/O Port" (refer to respective SOLO user manual to know more)

DATA	Motor Type
0 (0x00000000)	Selects DC brushed Motor in Digital Mode
1 (0x00000001)	Selects Normal BLDC-PMSM Motor in Digital Mode
2 (0x00000002)	Selects ACIM Motor in Digital Mode
3 (0x00000003)	Selects Ultra fast BLDC-PMSM Motor in Digital Mode

0x16 : Control Mode Type

Code: 0x16	Control Mode Type			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0-2]	N/A	M	1

Description:
This command sets the Control Mode in terms of Torque, Speed or Position only in Digital Mode based on the following table, in Analogue Mode this functionality is selected by using Piano switch PIN 4 in SOLO UNO and using FN pin on SOLO MINI for only Torque and Speed controlling functionalities. (refer to respective SOLO user manual to know more)

DATA	Control Mode Type
0 (0x00000000)	Operates in Speed Mode
1 (0x00000001)	Operates in Torque Mode
2 (0x00000002)	Operates in Position Mode

0x17 : Current Controller Kp Gain (Torque controller)

Code: 0x17	Current Controller Kp (Torque controller)			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0 - 16000.0]	N/A	M	0

Description:
This command sets the value for Current Controller Kp or proportional gain, which will be used in the PI controller that controls the current (torque) inside of the motor both in Analogue or Digital Mode, this value is automatically identified by SOLO after Motor Identification but in any case it's possible for the user to alter this value as they like. After changing this value manually, for them to take effect a power recycle is required.

0x18 : Current Controller Ki Gain (Torque controller)

Code: 0x18	Current Controller Ki Gain (Torque controller)			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0 - 16000.0]	N/A	M	0

Description:
This command sets the value for Current Controller Ki or integral gain, which will be used in the PI controller that controls the current (torque) inside of the motor both in Analogue or Digital Mode, this value is automatically identified by SOLO after Motor Identification but in any case it's possible for the user to alter this value as they like. After changing this value manually, for them to take effect a power recycle is required.

0x19 : Monitoring Modes Enable/Disable

Code: 0x19	Monitoring Modes Enable/Disable			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0-2]	N/A	V	0

Description:
This command enables the continuous Monitoring Mode which will send a flow of data based on table below if the requested DATA is sent using the command code of 0x19 in a data packet to SOLO, this functionality works both in Digital and Analogue Control Modes.
This Mode is recommended to be used only for inspections and during system evaluations, as its activation puts a heavy processing burden on the main DSP on SOLO.

DATA	Action
0 (0x00000000)	Disables the Monitoring Mode
1 (0x00000001)	Activates Normal Monitoring mode with 200 Hz sampling rate for each feedback, the following feedbacks will be sent in order:

Command code	Feedback
0x82	**VA:** 3 phase Motors phase A voltage (BLDC, PMSM, ACIM)*
0x83	**VB:** 3 phase Motors phase B voltage (BLDC, PMSM, ACIM)*
0x84	**IA:** 3 phase Motors phase A Current (BLDC, PMSM, ACIM)**
0x85	**IB:** 3 phase Motors phase B Current (BLDC, PMSM, ACIM)**
0x86	**VBUS :** The power supply voltage (BUS / Battery voltage)

	<table><tr><td>0x87</td><td>**IM:** The DC motor Current</td></tr><tr><td>0x88</td><td>**VM**: The DC motor Voltage</td></tr><tr><td>0x8D</td><td>**Iq:** The quadrature Current in 3-phase motors</td></tr><tr><td>0x8E</td><td>**Id:** The direct/Magnetizing Current in 3-phase motors</td></tr><tr><td>0x96</td><td>**Speed:** The speed measured or estimated by SOLO</td></tr></table> *The value of IC for 3-phase motors can be derived from IC = IA + IB **The value of VC for 3-phase motors can be derived from VC = VA + VB
2 (0x00000002)	Activates Performance Monitoring mode with 500 Hz sampling rate for each feedback, the following feedbacks will be sent in order: <table><tr><th>Command code</th><th>Feedback</th></tr><tr><td>0xA0</td><td>**Position:** the number of quadrature pulses counted from the incremental encoder (Motor's real position)</td></tr><tr><td>0x8D / 0x87</td><td>**Iq / IM:** the current in Amps acting in Torque generation depending on the Motor type (3-phase or DC)</td></tr><tr><td>0xB0</td><td>**Rotor Angle:** per-unit value of measured or estimated rotor angle in 3-phase motors</td></tr><tr><td>0x96</td><td>**Speed:** The speed measured or estimated by SOLO</td></tr></table>

0x1A : Magnetizing Current Reference (Id)

Code: 0x1A	Magnetizing Current Reference (Id)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0 - 32.0]	Amps	V	0

Description:
This command sets the desired magnetizing current (Id) required for controlling ACIM motors in FOC in Amps, it can be used only in Closed-loop digital mode, since in Analogue mode the Magnetizing current reference is set throug Pin "P/F" (refer to SOLO user manual to know more)

0x1B : Position Reference

Code: 0x1B	Position Reference			
Data Type	Data Range	Units	Memory Storage	Default Value
Int32	[-2,147,483,647 to 2,147,483,647]	Quad-Pulse	V	0

Description:
This command sets the desired Position reference in terms of quadrature pulses while SOLO operates with the Incremental Encoders or in terms of pulses while while SOLO operates with Hall sensors, this functionality will be available only once SOLO is in Close-loop Digital Position mode both for Incremental encoders as well as Hall sensors.

0x1C : Position Controller Kp Gain

Code: 0x1C	Position Controller Kp Gain			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0 - 16000.0]	N/A	M	0

Description:
This command sets the value for Position Controller Kp or proportional gain, which will be used in the PI controller that controls the position.

0x1D : Position Controller Ki Gain

Code: 0x1D	Position Controller Ki Gain			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0 - 16000.0]	N/A	M	0

Description:
This command sets the value for Position Controller Ki or integrator gain, which will be used in the PI controller that controls the position.

0x1F : Reset Position to Zero (Home)

Code: 0x1F	Reset Position to Zero (Home)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	1	N/A	V	0

Description:
This command resets the position counter back to zero if in the DATA part of the packet the value of “0x00000001” is placed.

0x20 : Overwrite the Errors

Code: 0x20	Overwrite the Errors			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	N/A	N/A	V	0

Description:
This command overwrites the reported errors in Error Register reported with command code of “0xA1”, The purpose of error overwriting is to allow the user to see the problems transparently and decide what to do, but SOLO will always keep track of catastrophic errors and it will not ignore them. The following table is the bit arrangement of the error register, if any of the mentioned errors occurs, you will receive “1” in the position of the bit corresponding to the error in a 32 bits register shown below, so if the value of the error register is anything other than Zero, it means there is a problem somewhere,which can be found exactly by checking the following bits. By putting zero in the place of each bit and sending a command with command code of “0x20” the error bits can be overwritten, similarly by sending a DATA part with only zeros inside, the whole error register will be overwritten. Once the error is over-written if it’s cause of existence is removed, SOLO will return back to normal operation, otherwise it will stay in fault mode without any switching at the output.

Bit Number	Error Description
0	Over-current error, while the current in the motor rises above 62A for more than 0.1ms
1	Over-voltage on BUS error, when the BUS voltage rises above 57V for more than 35us
2	Over Temperature Error, when the board temperature rises above 85 degrees
3	Encoder Calibration timeout - Index pulse is missing
4	Hall Sensors Calibration timeout
5	CAN Communication Lost (lifeTime expired)
6	N/A

0x21 : Sensorless Observer Gain for Normal BLDC-PMSM Motors

Code: 0x21	Sensorless Observer Gain for Normal Brushless Motor			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 1000]	N/A	M	0.9

Description:
This command sets the observer gain for the Non-linear observer that estimates the speed and angle of a BLDC or PMSM once the motor type is selected as normal BLDC-PMSM. This Observer Gain basically deals with the Motor Back EMF Estimation, and normally it has a value from 0.01 to 1.0 for regular operations. In General, the Motors with higher Electrical time constant need less values for this observer gain, Electrical Time constant can be driven from division of Inductance over Resistance of the Motor’s phases. This observer gain will be used both in Analogue and Digital Mode for Sensor-less operations and the gain can be modified dynamically in run-time.

0x22 : Sensorless Observer Gain for Ultra-fast BLDC-PMSM Motors

Code: 0x22	Sensorless Observer Gain for Ultra-fast BLDC-PMSM Motors			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 1000]	N/A	M	0.99

Description:
This command sets the observer gain for the Non-linear observer that estimates the speed and angle of a BLDC or PMSM once the motor type is selected as ultra-fast BLDC-PMSM. This Observer Gain basically deals with the Motor Back EMF Estimation, and normally it has a value from 0.01 to 1.0 for regular operations. In General, the Motors with higher Electrical time constant need less values for this observer gain, Electrical Time constant can be driven from division of Inductance over Resistance of the Motor’s phases. This observer gain will be used both in Analogue and Digital Mode for Sensor-less operations and the gain can be modified dynamically in run-time.

0x23 : Sensorless Observer Gain for DC Brushed Motors

Code: 0x23	Sensorless Observer Gain for DC Brushed Motors			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 1000]	N/A	M	50

Description:
This command sets the observer gain for the Non-linear observer that estimates the speed of a DC brushed once the motor type is selected as DC brushed. This Observer Gain basically deals with the Motor Back EMF Estimation, and normally it has a value from 10 to 100 for regular operations. This observer gain will be used both in Analogue and Digital Mode for Sensor-less operations and the gain can be modified dynamically in run-time.

0x24 : Sensorless Observer Filter Gain for Normal BLDC-PMSM Motors

Code: 0x24	Sensorless Observer Filter Gain for Normal BLDC-PMSM Motors			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 16000]	N/A	M	50

Description:
This command sets how fast the observer should operate once SOLO is in sensorless mode with normal BLDC-PMSM selected as the Motor type. Generally this gain can have values from 1 to 1000 and the lower the gain, the better will be the performance in Higher speeds. For tuning purposes, the user needs to check the behaviour of the motor for different values other than default value and try to find at what gain the performance is at best. The best method is to start around the default gain and try to reduce or increase the gain with steps of 2X, 3X, 4X, … and so on to be able to see the significance of any change on the final result. The reason for this is if this gain is changed with very small steps, the user might not be able to see any difference, so once a good region is found, the user can start tuning the gain around that value with more accuracy.

0x25 : Sensorless Observer Filter Gain for Ultra Fast BLDC-PMSM Motors

Code: 0x25	Sensorless Observer Filter Gain for Ultra Fast BLDC-PMSM Motors			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.01 - 16000]	N/A	M	10

Description:
This command sets how fast the observer should operate once SOLO is in sensorless mode with ultra-fast BLDC-PMSM selected as the Motor type. Generally this gain can have values from 1 to 1000 and the lower the gain, the better will be the performance in Higher speeds. For tuning purposes, the user needs to check the behaviour of the motor for different values other than default value and try to find at what gain the performance is at best. The best method is to start around the default gain and try to reduce or increase the gain with steps of 2X, 3X, 4X, … and so on to be able to see the significance of any change on the final result. The reason for this is if this gain is changed with very small steps, the user might not be able to see any difference, so once a good region is found, the user can start tuning the gain around that value with more accuracy.

0x26 : UART Baud-Rate

Code: 0x26	Set UART Baud-Rate			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	0/1	Bits/s	M	937500

Description:
This command sets the baud-rate of the UART line, currently there are two baudrates supported by SOLO, and they can be selected by sending either 0 or 1 in DATA section of the packet with command code of “0x26”, if the value of the baud rate is changed, to make it effective a power recycle is necessary.

DATA	Baud-Rate [bits/s]
0 (0x00000000)	937500 (compatible with 921600)
1 (0x00000001)	115200

0x27 : Encoder or Hall Sensors Calibration Start/Stop

Code: 0x27	Encoder or Hall Sensors Calibration Start/Stop			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0-2]	N/A	V	0

Description:
This command starts or stops the process of sensor calibration based on the table below if the DATA is sent as below with command code of "0x27" in the packet. The sensor calibration is done in full torque mode limited to Current Limit value. The Encoder or Hall sensor calibration is only necessary for 3-phase motors and in case of incremental encoders the presence of index pulse is mandatory to find the correct mechanical offset of the shaft of the motor with respect to the control unit.

DATA	Action
0 (0x00000000)	Stop the calibration process
1 (0x00000001)	Start Incremental Encoder Calibration
2 (0x00000002)	Start Hall Sensors Calibration

0x28 : Per-Unit Encoder or Hall sensor Counter Clockwise offset

Code: 0x28	Per-Unit Encoder or Hall sensor Counter Clockwise offset			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0-1.0]	N/A	M	0

Description:
This command sets the per-unit offset identified after sensor calibration for Encoder or Hall sensors in C.C.W direction, the value must be between 0 to 1 and it gets automatically updated each time after sensor calibration, however the users can insert their desired value dynamically.

0x29 : Per-Unit Encoder or Hall sensor Clockwise offset

Code: 0x29	Per-Unit Encoder or Hall sensor Clockwise offset			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0-1.0]	N/A	M	0
Description: This command sets the per-unit offset identified after sensor calibration for Encoder or Hall sensors in C.W direction, the value must be between 0 to 1 and it gets automatically updated each time after sensor calibration, however the users can insert their desired value dynamically.				

0x2A : Speed Acceleration Value

Code: 0x2A	Speed Acceleration Value			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0-1600.0]	Rev/S^2	M	0
Description: This command defines the acceleration value of the Speed for speed controller both in Analogue and Digital modes in Revolution per square seconds, this value only has effect once SOLO is in Speed control mode and it will be ignored once in Torque or Position Mode.				

0x2B : Speed Deceleration Value

Code: 0x2B	Speed Deceleration Value			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0-1600.0]	Rev/S^2	M	0

Description:
This command defines the deceleration value of the Speed for speed controller both in Analogue and Digital modes in Revolution per square seconds, this value only has effect once SOLO is in Speed control mode and it will be ignored once in Torque or Position Mode. Defining deceleration can reduce the regenerative power going back to supply as the speed reduces slowly. The deceleration will be by passed (fast stop) if the user sends the following speed references based on following table:

Mode	Speed Reference for fast stop with regeneration
Digital Control	0
Analogue Control PWM input	Duty cycle of 0.5% or less
Analogue Control Voltage input	Voltage of 5mV or less

0x2C : CAN Bus Baud-rate

Code: 0x2C	CAN Bus Baud-rate			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0-4]	kbits/s	M	1000

Description:
This command sets the baud rate of CAN bus in CANOpen network based on the following table if the DATA in the packet with command code of 0x2C is used, after changing the baud rate a power cycle is required for SOLO so the new baud rate can take effect.

Data	CAN bus Baud-rate [kbits/s]
0 (0x00000000)	1000
1 (0x00000001)	500
2 (0x00000002)	250
3 (0x00000003)	125
4 (0x00000004)	100

0x2D : Analogue Speed Resolution Division Coefficient (ASRDC)

Code: 0x2D	Analogue Speed Resolution Division Coefficient (ASRDC)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0.0001-10000]	NA	M	1

Description:
This command defines the resolution of the speed at S/T input while SOLO operates in Analogue mode based on the following formula:
The max Speed on S/T by applying 5V = Max default speed defined by the motor type / ASRDC

The default Max speed for each motor type can be found in the following table:

Motor Type	Closed-loop Sensor-less	Closed-loop Sensor-based
BLDC - PMSM	8000 [RPM]	8000 [RPM]
BLDC - PMSM Ultrafast	30,000 RPM	30,000 [RPM]
DC Brushed	30,000 [N/A]**	8000 [RPM]
AC Induction Motor	4000 [RPM]	4000 [RPM]

** This is a qualitative value based on the observer gain defined for the sensorless speed estimator for DC brushed motors, the value can be ranged from 0 to 30,000 and the final speed of the motor for each value depends on the characteristics of the Motor itself like BEMF constant.

Read Commands:

Below all the existing command codes to read a value from SOLO with their description for UART or USB communication for SOLO are listed, to read a value from SOLO, the proper command code should be used with DATA section filled with Zeros (unless specified) for the request of the feedback.

0x81 : Device Address

Code: 0x81	Device Address			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uin32	[0-254]	N/A	M	0
Description: This command reads the device address connected on the line.				

0x82 : Phase-A voltage

Code: 0x82	Phase-A voltage			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[-60 - 60]	Volts	V	0
Description: This command reads the phase-A voltage of the motor connected to the “A” pin output of SOLO for 3-phase Motors.				

0x83 : Phase-B voltage

Code: 0x83	Phase-B voltage			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[-60 - 60]	Volts	V	0
Description: This command reads the phase-B voltage of the motor connected to the “B” pin output of SOLO for 3-phase Motors.				

0x84 : Phase-A Current

Code: 0x84	Phase-A Current			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[-32 - 32]	Amps	V	0

Description:
This command reads the phase-A current of the motor connected to the "A" pin output of SOLO for 3-phase Motors.

0x85 : Phase-B Current

Code: 0x85	Phase-B Current			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[-32 - 32]	Amps	V	0

Description:
This command reads the phase-B current of the motor connected to the "B" pin output of SOLO for 3-phase Motors.

0x86 : BUS Voltage (Input Supply / Battery)

Code: 0x86	BUS Voltage (Input Supply / Battery)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0 - 60]	Volts	V	0

Description:
This command reads the input BUS voltage.

0x87 : DC Motor Current (IM)

Code: 0x87	DC Motor Current (IM)			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[-32 - 32]	Amps	V	0
Description: This command reads the current inside the DC brushed motor connected to “B” and “C” outputs of SOLO.				

0x88 : DC Motor Voltage (VM)

Code: 0x88	DC Motor Voltage (VM)			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[-60 - 60]	Volts	V	0
Description: This command reads the voltage of the DC brushed motor connected to “B” and “C” outputs of SOLO.				

0x89 : Speed Controller Kp Gain

Code: 0x89	Speed Controller Kp Gain			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0 - 16000.0]	N/A	M	0
Description: This command reads the value of the Speed controller Kp gain, set for Digital mode operations.				

0x8A : Speed Controller Ki Gain

Code: 0x89	Speed Controller Kp Gain			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0 - 16000.0]	N/A	M	0
Description: This command reads the value of the Speed controller Ki gain, set for Digital mode operations.				

0x8B : Output PWM Frequency (Switching Frequency)

Code: 0x--	Set Device Address			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[8000-80000]	Hz	M	20000/80000
Description: This command reads the output switching frequency of SOLO in Hertz.				

0x8C : Current Limit

Code: 0x8C	Current Limit			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0 - 32.0]	Amps	M	0
Description: This command reads the value of the current limit set for SOLO in closed-loop digital operation mode.				

0x8D : Quadrature Current (Iq)

Code: 0x8D	Quadrature Current (Iq)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[-64.0 - 64.0]	Amps	V	0

Description:
This command reads the actual monetary value of "Iq" that is the current acts in torque generation in FOC mode for 3-phase motors, the amount of Torque on the shaft of the motor can be calculated using the following formula:
Actual Torque [N.m] = Iq [A] * Motor's Torque constant [N.m/A]

0x8E : Direct Current / Magnetizing Current (Id)

Code: 0x8E	Direct Current / Magnetizing Current (Id)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[-64.0 - 64.0]	Amps	V	0

Description:
This command reads the actual monetary value of Id that is the direct current acting in FOC, this current for ACIM motors is known as Magnetizing current as well.

0x8F : Motor's Number of Poles

Code: 0x8F	Motor's Number of Poles			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[1 - 254]	N/A	M	8

Description:
This command reads the number of Poles set for 3-phase motors.

0x90 : Incremental Encoder’s Lines

Code: 0x90	Incremental Encoder’s Lines			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[1 - 200,000]	N/A	M	1000

Description:
This command reads the number of physical Incremental encoder lines set on SOLO.

0x91 : Current Controller Kp Gain

Code: 0x91	Current Controller Kp Gain			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0 - 16000.0]	N/A	M	0

Description:
This command reads the amount of value set for Current controller Kp or proportional gain.

0x92 : Current Controller Ki Gain

Code: 0x92	Current Controller Ki Gain			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0 - 16000.0]	N/A	M	0

Description:
This command reads the amount of value set for Current controller Ki or integrator gain.

0x93 : Board Temperature

Code: 0x93	Board Temperature			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[-30.0 - 150.0]	°C	V	0

Description:
This command reads the momentary temperature of the board in centigrade.

0x94 : Motor's Resistance value

Code: 0x94	Motor's Resistance value			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0 - 100.0]	Ohms	M	0

Description:
This command reads the Phase or Armature resistance of the 3-phase or DC brushed motor connected to SOLO respectively.

0x95 : Motor's Inductance value

Code: 0x95	Motor's Inductance value			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0 - 0.2]	Henry	M	0

Description:
This command reads the Phase or Armature Inductance of the 3-phase or DC brushed motor connected to SOLO respectively.

0x96 : Speed Feedback

Code: 0x96	Speed Feedback			
Data Type	Data Range	Units	Memory Storage	Default Value
Int32	[-30,000 - 30,000]	RPM	V	0

Description:
This command reads the actual speed of the motor measured or estimated by SOLO in sensorless or sensor-based modes respectively.

0x97 : Motor Type

Code: 0x97	Motor Type			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0 - 3]	N/A	M	0

Description:
This command reads the Motor type selected for Digital or Analogue mode operations, the value in DATA section corresponds to the Motor type shown in table below, if the system is in Analogue mode the motor type shown is the Motor type selected by Piano switch in SOLO UNO ro by the M1 and M2 pins in SOLO MINI in Analogue mode (refer to respective SOLO user manual to know more), to see the Motor type selected in Digital Mode, first the user has to put the system in Digital Mode using command code of “0x02”.

DATA	Motor Type
0 (0x00000000)	DC brushed Motor Selected
1 (0x00000001)	Normal BLDC-PMSM Motor Selected
2 (0x00000002)	ACIM Motor Selected
3 (0x00000003)	Ultra fast BLDC-PMSM Motor Selected

0x99: Feedback Control Mode

Code: 0x99	Feedback Control Mode			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0 - 2]	N/A	M	0

Description:
This command reads the feedback control mode selected on SOLO both for Analogue and Digital operations as below:

DATA	Operation
0 (0x00000000)	Operates in Sensor-less feedback Mode
1 (0x00000001)	Operates in Incremental Encoder feedback Mode
2 (0x00000002)	Operates in HALL Sensors feedback Mode

0x9A : Commanding Mode

Code: 0x9A	Commanding Mode			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	0 or 1	N/A	M	0

Description:
This command reads the actual commanding mode that SOLO is operating , based on table below :

DATA	Operating Mode
0 (0x00000000)	SOLO is in Analogue Mode
1 (0x00000001)	SOLO is in Digital Mode

0x9B : Control Mode Type

Code: 0x9B	Control Mode Type			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0-2]	N/A	M	1

Description:
This command reads the Control Mode type in terms of Torque, Speed or Position in both Digital and Analogue modes.

DATA	Control Mode Type
0 (0x00000000)	SOLO is in Speed Mode
1 (0x00000001)	SOLO is in Torque Mode
2 (0x00000002)	SOLO is in Position Mode

0x9C : Speed Limit

Code: 0x9C	Speed Limit			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0-30,000]	RPM	M	30,000

Description:
This command reads the value of the speed limit set on SOLO.

0x9D : Position Controller Kp Gain

Code: 0x9D	PositionController Kp Gain			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0 - 16000.0]	N/A	M	0

Description:
This command reads the amount of value set for Position controller Kp or proportional gain.

0x9E: Position Controller Ki Gain

Code: 0x9E	Position Controller Ki Gain			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0 - 16000.0]	N/A	M	0

Description:
This command reads the amount of value set for Position controller Ki or integrator gain.

0xA0 : Position Feedback (Incremental Encoder and Hall sensors)

Code: 0xA0	Position Feedback (Incremental Encoder and Hall sensors)			
Data Type	Data Range	Units	Memory Storage	Default Value
Int32	[-2,147,483,647 to 2,147,483,647]	Quad-Pulses	V	0

Description:
This command reads the number of counted pulses from the Incremental Encoder or Hall sensors. The counted pulses for Incremental Encoders will be in Quadrature format, however the counted pulses for the Hall sensors will be based on the following :
Hall sensor counted pulses in 1 revolution = Motor's number of poles x 3

0xA1 : Error Register

Code: 0xA1	Error Register			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	N/A	N/A	V	0

Description:
This command reads the error register which is a 32 bit register with each bit corresponding to following errors in the table below , the error has occurred if the respective bit is set to 1, The errors can be over-written using command code 0x20 , if an error occurs, until it's not over-written, SOLO will send out the Error register packet every 300 Milliseconds.

Bit Number	Error Description
0	Over-current error, while the current in the motor rises above 62A for more than 0.1ms
1	Over-voltage on BUS error, when the BUS voltage rises above 57V for more than 35us
2	Over Temperature Error, when the board temperature rises above 85 degrees
3	Encoder Calibration timeout - Index pulse is missing
4	Hall Sensors Calibration timeout
5	CAN Communication Lost (lifeTime expired)
6	N/A

0xA2 : Firmware Version

Code: 0xA2	Firmware Version			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	N/A	N/A	M	N/A

Description:
This command reads the Firmware version existing currently on the SOLO unit, the DATA section of the received command will contain a Number like “0x0000B009” which will indicate the firmware version.

0xA3 : Hardware Version

Code: 0xA3	Hardware Version			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	N/A	N/A	M	N/A

Description:
This command reads the Hardware version of the SOLO unit connected.

0xA4: Torque Reference (Iq/IM)

Code: 0xA4	Torque Reference (Iq/IM)			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0 - 32.0]	Amps	V	0

Description:
This command reads the amount of desired Torque reference (Iq or IM) already set for the Motor to follow in Digital Closed-loop Torque control mode , in case of 3-phase motors, this value will be the “Iq” reference, and in case of DC-Brushed Motors, the value indicates the “IM” reference.

0xA5 : Speed Reference

Code: 0xA5	Speed Reference			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0 - 30,000]	RPM	V	0

Description:
This command reads the amount of desired Speed reference already set for the Motor to follow in Digital Closed-loop Speed control mode.

0xA6 : Magnetizing Current / Id Reference

Code: 0xA6	Magnetizing Current / Id Reference			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0 - 32.0]	Amps	V	0

Description:
This command reads the amount of desired Id (direct current) or Magnetizing current reference already set for the Motor to follow in Digital Closed-loop Speed control mode for ACIM motors.

0xA7 : Position Reference

Code: 0xA7	Position Reference			
Data Type	Data Range	Units	Memory Storage	Default Value
Int32	[-2,147,483,647 to 2,147,483,647]	Quad-Pulses	V	0

Description:
This command reads the desired position reference set for the Motor to follow in Digital Closed-loop Position mode in terms of quadrature pulses.

0xA8 : Power Reference

Code: 0xA8	Power Reference			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0 - 100]	%	V	0

Description:
This command reads the desired Power reference for SOLO to apply in Digital Open-loop speed control mode for 3-phase motors in terms of percentage.

0xA9 : Desired Direction of Rotation

Code: 0xA9	Desired Direction of Rotation			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	0 or 1	N/A	V	0

Description:
This commands reads the desired direction of rotation set for the Motor to follow based on table below:

DATA	Requested Rotational Direction
0 (0x00000000)	Counter ClockWise
1 (0x00000001)	ClockWise

0xAA : Sensorless Observer Gain for Normal BLDC-PMSM Motors

Code: 0xAA	Sensorless Observer Gain for Normal BLDC-PMSM Motors			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.01 - 1000]	N/A	M	0.9

Description:
This command reads the value of Sensorless Observer Gain for Normal BLDC-PMSM Motors.

0xAB : Sensorless Observer Gain for Ultra-fast BLDC-PMSM Motors

Code: 0xAB	Sensorless Observer Gain for Ultra-fast BLDC-PMSM Motors			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 1000]	N/A	M	0.99
Description: This command reads the value of Sensorless Observer Gain for Normal BLDC-PMSM Motors.				

0xAC : Sensorless Observer Gain for DC Motor

Code: 0xAC	Sensorless Observer Gain for DC Motor			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 1000]	N/A	M	50
Description: This command reads the value of Sensorless Observer Gain for DC Motor.				

0xAD : Sensorless Observer Filter Gain for Normal BLDC-PMSM Motors

Code: 0xAD	Sensorless Observer Filter Gain for Normal BLDC-PMSM Motors			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 16000]	N/A	M	50
Description: This command reads the value of Sensorless Observer Filter Gain for Normal BLDC-PMSM Motors.				

0xAE : Sensorless Observer Filter Gain for Ultra Fast BLDC-PMSM Motors

Code: 0xAE	Sensorless Observer Filter Gain for Ultra Fast BLDC-PMSM Motors			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.01 - 16000]	N/A	M	10
Description: This command reads the value of Sensorless Observer Filter Gain for Ultra Fast BLDC-PMSM Motors.				

0xB0 : Motor’s Angle

Code: 0xB0	Motor’s Angle			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[-2 - 2]	Per Unit	V	10
Description: This command reads the measured or estimated per-unit angle of the 3-phase motors.				

0xB1 : Per-Unit Encoder or Hall sensor Counter Clockwise offset

Code: 0xB1	Per-Unit Encoder or Hall sensor Counter Clockwise offset			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0 - 1.0]	Per Unit	M	0
Description: This command reads the per-unit Encoder or Hall sensor offset in C.C.W direction.				

0xB2 : Per-Unit Encoder or Hall sensor Clockwise offset

Code: 0xB2	Per-Unit Encoder or Hall sensor Clockwise offset			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0 - 1.0]	Per Unit	M	0
Description: This command reads the per-unit Encoder or Hall sensor offset in C.C.W direction.				

0xB3 : UART Baud Rate

Code: 0xB3	UART Baud Rate			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	0/1	Bits/s	M	937500
Description: This command reads Baud Rate selected on SOLO unit to communicate through UART line as shown in table below:				

DATA	Baud-Rate [bits/s]
0 (0x00000000)	937500 (compatible with 921600)
1 (0x00000001)	115200

0xB4 : Speed Acceleration Value

Code: 0xB4	Speed Acceleration Value			
Data Type	Data Range	Units	Memory Storage	Default Value
Sfxt(32-17)	[0.0-1600.0]	Rev/S^2	M	0
Description: This command reads the acceleration value of the Speed for speed controller both in Analogue and Digital modes in Revolution per square seconds.				

0xB5 : Speed Deceleration Value

Code: 0xB5	Speed Deceleration Value			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Sfxt(32-17)	[0.0-1600.0]	Rev/S^2	M	0

Description:
This command reads the deceleration value of the Speed for speed controller both in Analogue and Digital modes in Revolution per square seconds.

0xB6 : CAN Bus Baud Rate

Code: 0xB6	CAN Bus Baud Rate			
Data Type	**Data Range**	**Units**	**Memory Storage**	**Default Value**
Uint32	[0-4]	kbits/s	M	1000

Description:
This command Reads the baud rate of CAN bus in CANOpen network based on the following table.

DATA	CAN bus Baud Rate [kbits/s]
0 (0x00000000)	1000
1 (0x00000001)	500
2 (0x00000002)	250
3 (0x00000003)	125
4 (0x00000004)	100

0xB7 : Analogue Speed Resolution Division Coefficient (ASRDC)

Code: 0xB7	Analogue Speed Resolution Division Coefficient (ASRDC)			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0.0001-10000]	NA	M	1

Description:
This command reads the Analogue Speed Resolution Division Coefficient (ASRDC) that defines resolution of the speed at S/T input while SOLO operates in Analogue mode based on the following formula:
The max Speed on S/T by applying 5V = Max default speed defined by the motor type / ASRDC
The default Max speed for each motor type can be found in the following table:

Motor Type	Closed-loop Sensor-less	Closed-loop Sensor-based
BLDC - PMSM	8000 [RPM]	8000 [RPM]
BLDC - PMSM Ultrafast	30,000 RPM	30,000 [RPM]
DC Brushed	30,000 [N/A]**	8000 [RPM]
AC Induction Motor	4000 [RPM]	4000 [RPM]

** This is a qualitative value based on the observer gain defined for the sensorless speed estimator for DC brushed motors, the value can be ranged from 0 to 30,000 and the final speed of the motor for each value depends on the characteristics of the Motor itself like BEMF constant.

0xB8 : Incremental Encoder Index Counts

Code: 0xB8	Incremental Encoder Index Counts			
Data Type	Data Range	Units	Memory Storage	Default Value
Uint32	[0 - 2,147,483,647]	Pulses	V	0

Description:
This Command reads the number of counted index pulses seen on the Incremental Encoder's output, the Index pulse should occur always at a fixed and certain mechanical position with respect to the stator of the Motor. While the Motor is turning the assumption is the index pulse occurs only once per revolution.

UART/USB Packet Formation Examples:

Below, you can find a number of examples on how to form packets for different purposes for sending to SOLO.

_ **NOTE :** The SPACE character in between the bytes below, is just for readability purposes and ASCII code of SPACE shouldn’t be sent to SOLO, all the bytes should be sent to SOLO one after another with nothing in between.

-NOTE: The device address in all the packets below is considered as “0”, since this is the default device address. If you change the device address you must also change it alle in these packets as well.

Change the Direction of Rotation:

Change the direction of rotation to CCW
FF FF 00 0C 00 00 00 01 00 FE

Change the direction of rotation to CW
FF FF 00 0C 00 00 00 00 00 FE

Stop the Motor [Emergency]

Emergency stopping the Motor, After sending this packet the power recycle is required
FF FF 00 08 00 00 00 00 00 FE

Set the Motor's Number of Poles

- **Note:**This is a parameter useful for 3 phase Motors and their accurate speed calculation. Notice that **<u>this is different with the number of pole-pairs</u>** and the relation is as below:

  Motor Number of Poles = Number of Pole-pairs * 2

Set the Number of Poles at 8:
FF FF 00 0F 00 00 00 08 00 FE

To check the written value:

Read the number of Poles:
FF FF 00 8F 00 00 00 00 00 FE

Change or Set/Reset the Device address:

The default device address on each SOLO upon delivery is 0x00, but you can change this value to anything in between 0x00 to 0xFE (254). For example, below you can see an example of how to change the device address from 0 to 1, and then check the changes.

Set Device Address to 1
FF FF 00 01 00 00 00 01 00 FE

Read Device Address
FF FF 01 81 00 00 00 00 00 FE

By putting "0xFF" inside the device address using the command code "0x12" you can reset any device address back to ZERO whenever you want. You can also "Reset Factory" the device to go back to the default address of 0x00.

Reset the Device Address to ZERO

FF FF FF 12 00 00 00 00 00 FE

To check the correct reset:

Read the Device Address

FF FF 00 81 00 00 00 00 00 FE

Set the output switching Frequency (PWM frequency):

On SOLO you can have any switching frequency from 8kHz to 80kHz at the output depending on your need and the inductance of your motor, in general the Motors with low inductance need higher switching frequencies to be able to keep their current stabilized and controlled.

_NOTE: By default there are two types of PWM frequencies available on SOLO before you overwrite them, 20kHz and 80kHz (check the User Manual of the device to learn more), but if you overwrite the PWM frequency by using the following method, the switching frequency for all the motor types becomes the selected value as long as you haven't done the "Reset factory" which will set back everything to default mode.

Set switching frequency on 30Khz
FF FF 00 09 00 00 00 1E 00 FE

To check the written value:

Read the switching frequency:

FF FF 00 8B 00 00 00 00 00 FE

Reset the Device to Factory Mode:

This is a packet to reset the device into factory default mode; after this action, all the parameters saved in the memory will be reset into their default values. After this action, a power recycle is required so that the changes can take effect.

FF FF 00 14 00 00 00 01 00 FE

Digital sensorless Torque Control of a BLDC motor.

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

As long as the piano switch number 5 is DOWN in SOLO UNO or the Control Mode Switch on SOLO MINI is on C.L, you can send the following commands in order to control the torque of your brushless Motor. Also after power recycling you don't need to repeat this part again, just leave the configurations as mentioned in closed-loop mode as long as your motor and the system are the same.

1- Set the Motor Type (Normal Brushless, type 1)
FF FF 00 15 00 00 00 01 00 FE

2- Set control Mode on Torque
FF FF 00 16 00 00 00 01 00 FE

3- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE

4- Set the Torque ref at 1.8A [Sfxt(32-17)]
FF FF 00 04 00 03 99 99 00 FE

_Note : the steps 1 to 3 are needed to be done only once since you don't need to change them, so for varying the torque you can keep sending different values only by using step number 4 from now on. Remember, after power recycling, you need to send all the first 3 commands to set back the parameters which are Volatile and will be forgotten after power recycling or reset.

Digital sensorless Speed Control of a Brushless motor.

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

As long as the piano switch number 5 is DOWN in SOLO UNO or the Control Mode Switch on SOLO MINI is on C.L, you can send the following commands in order to control the torque of your brushless Motor. Also after power recycling you don't need to repeat this part again, just leave the configurations as mentioned in closed-loop mode as long as your motor and the system are the same.

1- Set Motor Type (Normal Brushless)
FF FF 00 15 00 00 00 01 00 FE

2- Set control Mode on SPEED
FF FF 00 16 00 00 00 00 00 FE

3- set the speed Kp gain on 0.003 [Sfxt(32-17)]
FF FF 00 0A 00 00 01 89 00 FE

4- Set the speed ki gain on 0.001 [Sfxt(32-17)]
FF FF 00 0B 00 00 00 83 00 FE

5- Set the Motor No of Poles at 8 (Uint32)
FF FF 00 0F 00 00 00 08 00 FE

6- Set the Speed Control Mode on Sensor-less
FF FF 00 13 00 00 00 00 00 FE

7- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE

8- Set the speed ref at 1000 rpm (Uint32)
FF FF 00 05 00 00 03 E8 00 FE

- Using the command set in step 8, you can define any arbitrary speed for your motor to reach and follow in real-time.
- The commands stored in Memory will be remembered after power recycle, and there is no need to set them everytime.

Digital Speed Control of a Brushless motor using Encoders

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

As long as the piano switch number 5 is DOWN in SOLO UNO or the Control Mode Switch on SOLO MINI is on C.L, you can send the following commands in order to control the torque of your brushless Motor. Also after power recycling you don't need to repeat this part again, just leave the configurations as mentioned in closed-loop mode as long as your motor and the system are the same.

1- Set Motor Type (Normal Brushless, Type 1)
FF FF 00 15 00 00 00 01 00 FE
2- Set control Mode on SPEED
FF FF 00 16 00 00 00 00 00 FE
3- set the speed Kp gain on 0.001 [Sfxt(32-17)]
FF FF 00 0A 00 00 00 83 00 FE
4- Set the speed ki gain on 0.0008 [Sfxt(32-17)]
FF FF 00 0B 00 00 00 68 00 FE
5- Set the Motor No of Poles at 8 (Uint32)
FF FF 00 0F 00 00 00 08 00 FE
6- Set the Speed Control Mode on Encoder
FF FF 00 13 00 00 00 01 00 FE
7- Set the Number of Encoder Lines on 1000 lines (Uint32)
FF FF 00 10 00 00 03 E8 00 FE
8- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE
9- Set the speed ref at 1000 rpm (Uint32)
FF FF 00 05 00 00 03 E8 00 FE

Notes:

- Using the command set in step 9, you can define any arbitrary speed for your motor to reach and follow in real-time.
- The commands stored in Memory will be remembered after power recycle, and there is no need to set them everytime.

Digital sensorless Speed Control of a DC Brushed motor

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

As long as the piano switch number 5 is DOWN in SOLO UNO or the Control Mode Switch on SOLO MINI is on C.L, you can send the following commands in order to control the torque of your brushless Motor. Also after power recycling you don't need to repeat this part again, just leave the configurations as mentioned in closed-loop mode as long as your motor and the system are the same.

1- Set Motor Type (Brushed DC)
FF FF 00 15 00 00 00 00 00 FE
2- Set control Mode on SPEED
FF FF 00 16 00 00 00 00 00 FE
3- Set the speed Kp gain on 0.01 [Sfxt(32-17)]
FF FF 00 0A 00 00 05 1E 00 FE
4- Set the speed ki gain on 0.04 [Sfxt(32-17)]
FF FF 00 0B 00 00 14 7A 00 FE
5- Set the Speed Control Mode on sensor-less
FF FF 00 13 00 00 00 00 00 FE
6- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE
7- Set the speed ref at 200 * (read below)
FF FF 00 05 00 00 00 C8 00 FE

Notes:

- * The speed reference in Sensorless DC control here is not in RPM, but it's a quantitative value depending on the BEMF constant of your DC motor, you need to send different values and find out the best numbers for you specific motor, in future we will publish more information on how to exactly calculate the speed of your brushed DC motor using this method. In General values from 0 to 500 for most of the motors are the range from 0 speed to max speed.
- Using the command set in step 7, you can define any arbitrary speed for your motor to reach and follow in real-time.
- The commands stored in Memory will be remembered after power recycle, and there is no need to set them everytime.

Digital Speed Control of a DC Brushed motor using Encoder

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

As long as the piano switch number 5 is DOWN in SOLO UNO or the Control Mode Switch on SOLO MINI is on C.L, you can send the following commands in order to control the torque of your brushless Motor. Also after power recycling you don't need to repeat this part again, just leave the configurations as mentioned in closed-loop mode as long as your motor and the system are the same.

1- Set Motor Type (Brushed DC)
FF FF 00 15 00 00 00 00 00 FE
2- Set control Mode on SPEED
FF FF 00 16 00 00 00 00 00 FE
3- set the speed Kp gain on 0.003 [Sfxt(32-17)]
FF FF 00 0A 00 00 01 89 00 FE
4- Set the speed ki gain on 0.001 [Sfxt(32-17)]
FF FF 00 0B 00 00 00 83 00 FE
5- Set the Number of Encoder Lines on 500 lines(Uint32)
FF FF 00 10 00 00 01 F4 00 FE
6- Set the Speed Control Mode on Sensored using Encoder
FF FF 00 13 00 00 00 01 00 FE
7- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE
8- Set the speed ref at 1000 rpm (Uint32)
FF FF 00 05 00 00 03 E8 00 FE

Notes:

- Using the command set in step 8, you can define any arbitrary speed for your motor to reach and follow in real-time.
- The commands stored in Memory will be remembered after power recycle, and there is no need to set them everytime.

Digital Open-loop Control of a Brushless motor

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in open-loop by pushing or leaving Piano switch number 5 in UP position in SOLO UNO or by putting the Control Mode Switch on O.L on SOLO MINI

As long as the piano switch number 5 is UP in SOLO UNO or the Control Switch Mode is on O.L in SOLO MINI, you can send the following commands in order to control the speed of your brushless motor in open-loop mode.

1- Set Motor Type (Normal Brushless)
FF FF 00 15 00 00 00 01 00 FE
2- Set the Motor No of Poles at 8 (Uint32)
FF FF 00 0F 00 00 00 08 00 FE
3- Set the Speed Control Mode on sensor-less
FF FF 00 13 00 00 00 00 00 FE
4- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE
5- Set the power ref at 30.5 (similar to analog PWM duty cycle percentage) [Sfxt(32-17)]
FF FF 00 06 00 3D 00 00 00 FE
6- Set the speed ref at 1000 rpm (Uint32)
FF FF 00 05 00 00 03 E8 00 FE

Notes:

- By manipulating the values used in steps 5 and 6, you can reach any arbitrary required speed in real-time.
- The commands stored in Memory will be remembered after power recycle, and there is no need to set them everytime.

Digital sensorless closed-loop speed Control of an AC Induction Motor

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

As long as the piano switch number 5 is DOWN in SOLO UNO or the Control Mode Switch on SOLO MINI is on C.L, you can send the following commands in order to control the torque of your brushless Motor. Also after power recycling you don't need to repeat this part again, just leave the configurations as mentioned in closed-loop mode as long as your motor and the system are the same.

1- Set Motor Type (ACIM)
FF FF 00 15 00 00 00 02 00 FE
2- Set the Motor No of Poles at 4
FF FF 00 0F 00 00 00 04 00 FE
3- set the speed Kp gain on 0.008 [Sfxt(32-17)]
FF FF 00 0A 00 00 04 18 00 FE
4- Set the speed ki gain on 0.005 [Sfxt(32-17)]
FF FF 00 0B 00 00 04 18 00 FE
5- Set the Speed Control Mode on sensor-less
FF FF 00 13 00 00 00 00 00 FE
6- Set control Mode on SPEED
FF FF 00 16 00 00 00 00 00 FE
7- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE
8- Set the Magnetizing current reference (Id) at 1.8A [Sfxt(32-17)]
FF FF 00 1A 00 03 99 99 00 FE
9- Set the speed ref at 1000 rpm (Uint32)
FF FF 00 05 00 00 03 E8 00 FE

- Using the command set in step 9, you can define any arbitrary speed for your motor to reach and follow in real-time

- The magnetizing Current for AC induction motors set at step 8, is in charge of generating flux on the stator, so the higher this value the more will be the flux but you need to find a proper balance based on your Motor.

Digital Position Control using Quadrature Encoders Examples:

In this section we will show two different examples of controlling position of two different types of motors, the users before proceeding must make sure they have correctly connected their Quadrature Encoder wires to SOLO (check out this Article to learn more).

Note_ The pulses coming to SOLO are counted in quadrature mode, meaning that for each mechanical encoder line, there will be 4 counts.

Note_ The resolution of the position controlling highly depends on the number of the Encoder lines / slots and can be derived as:

$$\text{Position Control Resolution [deg]} = \frac{360}{Encoder\ Number\ of\ physical\ Lines * 4}$$

The position references sent to SOLO are from the signed Int32 data type and they can be anything from **-2,147,483,647** to **+2,147,483,647** in decimal form which are indicating the desired position to be reached in terms of quadrature pulses. The sign of the position reference will define the direction of the rotation for SOLO.

Note: To use Position Control feature, first you need to tune the speed controller, and make sure the Kp and Ki gains for speed controller are well defined, when the motor started to work well in speed control mode, you need to set the Kp and Ki gains for the position controller and find the best combination for your purpose. You can also define a “Speed Limit” value, which will limit the position tracking speed to that value at maximum.

Digital Position Control of a Brushless Motor using Encoders:

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

Now you can proceed to send the commands with following orders, notice that commands that are saved in memory will be remembered after power recycle, and you don't need to send them every time.

1- Set Motor Type (fast Brushless)
FF FF 00 15 00 00 00 03 00 FE
2- Set control Mode on Position
FF FF 00 16 00 00 00 02 00 FE
3- set the speed Kp gain on 0.0001 [Sfxt(32-17)]
FF FF 00 0A 00 00 00 0D 00 FE
4- Set the speed ki gain on 0.0005 [Sfxt(32-17)]
FF FF 00 0B 00 00 00 41 00 FE
5- Set the Motor No of Poles at 8 (Uint32)
FF FF 00 0F 00 00 00 08 00 FE
6- Set the Number of Encoder Lines on 1000 lines (Uint32)
FF FF 00 10 00 00 03 E8 00 FE
7- Set the position controller Kp gain at 0.15 [Sfxt(32-17)]
FF FF 00 1C 00 00 4C CC 00 FE
8- Set the position controller Ki gain at 0.16 [Sfxt(32-17)]
FF FF 00 1D 00 00 51 EB 00 FE
9- Reset the position to zero
FF FF 00 1F 00 00 00 01 00 FE
10- set the motor current Limit at 7.8 A [Sfxt(32-17)]
FF FF 00 03 00 0F 99 99 00 FE
11- set the speed limit at 6000 RPM (Uint32)
FF FF 00 11 00 00 17 70 00 FE
12- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE
Now after this step you can start sending Position commands like below:

13- set the desired position to go at +2560 pulses (Quad pulses) (Int32)
FF FF 00 1B 00 00 0A 00 00 FE

After a while when the Motor reached to the goal:

14- set the desired position to go at -45000 pulses (Quad pulses) (Int32)
FF FF 00 1B FF FF 50 38 00 FE

Digital Position Control of a DC Brushed Motor using Encoders:

Before sending the following commands you must do the following steps at least once:

1. Turn ON SOLO
2. Put SOLO in closed-loop (using Piano switch on SOLO UNO or Control Mode Switch on SOLO MINI)
3. Make sure the motor Identification is done.

Now you can proceed to send the commands with following orders, notice that commands that are saved in memory will be remembered after power recycle, and you don't need to send them every time.

1- Set Motor Type (DC brushed)
FF FF 00 15 00 00 00 00 00 FE
2- Set control Mode on Position
FF FF 00 16 00 00 00 02 00 FE
3- set the speed Kp gain on 0.03 [Sfxt(32-17)]
FF FF 00 0A 00 00 01 89 00 FE
4- Set the speed ki gain on 0.001 [Sfxt(32-17)]
FF FF 00 0B 00 00 00 83 00 FE
5- Set the Number of Encoder Lines on 500 lines (Uint32)
FF FF 00 10 00 00 01 F4 00 FE
6- Set the position controller Kp gain at 0.15 [Sfxt(32-17)]
FF FF 00 1C 00 00 4C CC 00 FE
7- Set the position controller Ki gain at 0.16 [Sfxt(32-17)]
FF FF 00 1D 00 00 51 EB 00 FE
8- Reset the position to zero
FF FF 00 1F 00 00 00 01 00 FE
9- set the motor current Limit at 18.5 A [Sfxt(32-17)]
FF FF 00 03 00 25 00 00 00 FE
10- set the speed limit at 4000 RPM (Uint32)
FF FF 00 11 00 00 0F A0 00 FE
11- Go to digital Mode
FF FF 00 02 00 00 00 01 00 FE

Now after this step you can start sending Position commands like below:

12- set the desired position to go at +2560 pulses (Quad pulses) (Int32)
FF FF 00 1B 00 00 0A 00 00 FE

After a while when the Motor reached to the goal:

13- set the desired position to go at -45000 pulses (Quad pulses) (Int32)

FF FF 00 1B FF FF 50 38 00 FE