BLDR-5010A Driver Specifications:

1. Voltage: 24 – 50VDC (absolute value 20 – 60VDC)
2. Max. Rated current: 10A
3. Max. peak current: 30A
4. Closed loop speed control with encoder and hall sensor feedback
5. Speed control PID parameters adjustable
6. Internal encoder counter (quadrature decode) with resettable resolution
7. RS485 communication
8. Speed input via RS485, analog 0-5V / direction or on-board POT
9. Two dip switches for speed input and hall sensor spacing setting
10. Over current protection
11. Over temperature protection for Motor (thermistor inside the motor)
12. the watchdog's timeout: 0.25 seconds

RS485 communication:

1. Drivers can be daisy trained up to 10 units and each driver will have an ID# (set via RS485)
2. Send: speed & direction command (has the highest priority to take over the control)
3. Read: encoder counter data, motor temperature data, driver status data

Preset data:

Driver ID number: 1	PID control parameters: 3
Encoder resolution: 1	Thermistor reference data: 1
Max. speed data/current: 2	Max。Motor Temp。：1

The serial port (RS485) setting:

38400 baud rate; 8 data bits; no parity; 1 stop bit

A maximum of 10 controllers can be connected to one single RS-485 port. All the commands sent to the controller must be in ASCII character string format and followed by a CR (carriage return - 0D). All the SPACE and TAB are ignored. After receiving the complete command line, the controller will be executed the coming command.

For detail command list refer to **RS485 Communication Command and Instruction section**

Dip switch setting:

DIP 1: Hall sensor spacing angle, ON=120°, OFF = 60°

DIP 2: ON=on board RV speed control; OFF=0-5V input or RS485 control

1. Connector PIN assignment

C-1 Power

PIN1 : 24-48VDC

PIN2: GND

C-2 Control signal

PIN1: GND

PIN2: Speed (0-5v)

PIN3: Error out

PIN4: Direction

PIN5: Enable (5V)

C-3 Motor

PIN1: Motor A

PIN2: Motor B

PIN3: Motor C

C-4 Sensors

PIN1: +5V out

PIN2: Hall-a

PIN3: Hall-b

PIN4: Hall-c

PIN5: Encoder-A

PIN6: Encoder-B

PIN7: Thermistor

PIN8: Thermistor

PIN9: GND

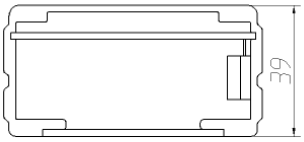
C-5 RS485

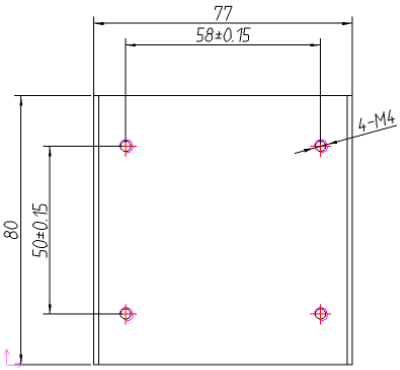
PIN1: RS485-A

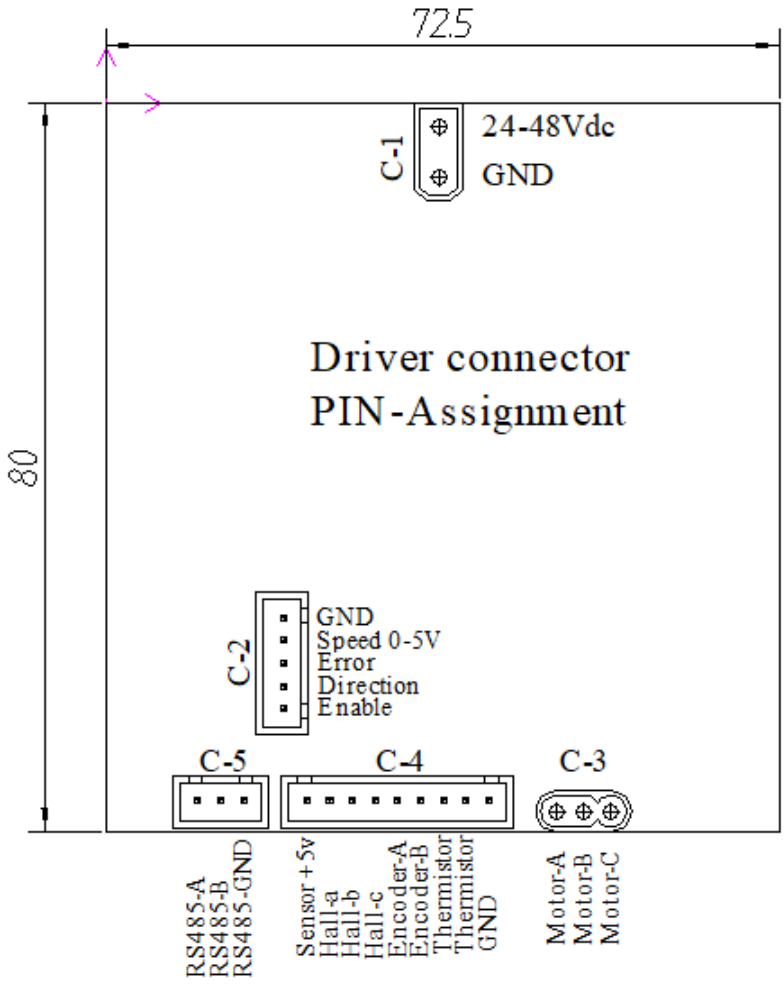
PIN2: RS485-B

PIN3: GND

2. Driver mechanical dimensions







1 RS485 Communication Command and Instruction

The communication start from the host and the slave respond。The maximum length of the command is 80ASCII code。Host send command start with'@'(40H)，follows with command code，and end with CR or LF(0DH, 0AH)。Slave response start with'&'(26H)，follows with data，and end with CR or LF(0DH, 0AH)。Data has to be “ascii code”。LRC checksums for ascii data strings is used. LRC checksums is used for all the character except the start and end code。

LRC checksums calculation： add all bytes in the message, excluding the starting and ending code, subtract the final field value from FF hex (all 1's), to produce the ones complement, add 1 to produce the twos complement.

2 command protocol

2.1 Read one byte

Host send:

Start code	Slave #	command	reg. address	checksum	end code	
@	Xx	01	xx	xx	CR	LF

Slave response:

Start code	Slave #	command	reg. address	Reg. value	checksum	end code	
&	Xx	01	xx	xx	xx	CR	LF

Ex: Read value of register 0x02 at slave #0, and if the value is 0x05 in this register :

Host send: ascii: @000102FD[CR][LF]

hex: 40H, 30H, 30H, 30H, 31H, 30H, 32H, 46H, 44H, 0DH, 0AH

slave send: ascii: &00010205F8[CR][LF]

hex: 26H, 30H, 30H, 30H, 31H, 30H, 32H, 30H, 35H, 46H, 38H, 0DH, 0AH

2.2 Read multiple bytes

Host send:

Start code	machine #	command	reg. address	# of byte	checksum	end code	
@	xx	02	xx	xx	xx	CR	LF

Slave response:

Start code	slave #	command	reg. address	# of byte	value of 1	***	value of N	checksum	end code	
&	xx	02	xx	xx	xx	***	xx	xx	CR	LF

Ex: Read 5 register value start at address 0x04 at Slave #0， and if the value is 0x01, 0x02, 0x03, 0x04, 0x05 in these 5 registers,

host : ascii: @00020405F5[CR][LF]

hex: 40H, 30H, 30H, 30H, 32H, 30H, 34H, 30H, 35H, 46H, 35H, 0DH, 0AH

slave: ascii: &000204050102030405E6[CR][LF]

hex: 26H, 30H, 30H, 30H, 32H, 30H, 34H, 30H, 35H, 30H, 31H, 30H, 32H, 30H, 33H, 30H, 34H, 30H, 35H, 45H, 36H, 0DH, 0AH

2.3 write one byte

Host send:

Start code	Slave #	command	reg. address	value	checksum	end code	
@	xx	03	xx	xx	xx	CR	LF

Slave response:

Start code	Slave #	command	reg. address	Status	checksum	end code	
&	xx	03	xx	xx	xx	CR	LF

Status of the writing（refer to this table in the following sections）：

Table 2.3.1

Value (Hex)	Description
00	Writing success
01	Writing failed
02	System busy，refuse to write
03	Do not allow to write

Ex: slave #0 at register 0x52, write 0x01, and assume the operation is successful.

host : ascii: @00035201AA[CR][LF]

hex: 40H, 30H, 30H, 30H, 33H, 35H, 32H, 30H, 31H, 41H, 41H, 0DH, 0AH

slave: ascii: &00035200AB[CR][LF]

hex: 26H, 30H, 30H, 30H, 33H, 35H, 32H, 30H, 30H, 41H, 42H, 0DH, 0AH

2.4 write multiple bytes

Host send:

Start code	slave #	command	reg. address	# of byte	value of 1	***	value of N	checksum	end code	
&	xx	04	xx	xx	xx	***	xx	xx	CR	LF

Slave response:

Start code	Slave #	Command	Reg. addr.	# of bytes	Status	Checksum	End code	
&	xx	**04**	xx	xx	xx	xx	CR	LF

Ex: slave #0, start register at 0x50write the following 4 bytes 0x01,0x02,0x03,0x04, successfully

host :

ascii: @00045004010203049E[CR][LF]

hex: 40H, 30H, 30H, 30H, 34H, 35H, 30H, 30H, 34H, 30H, 31H, 30H, 32H, 30H, 33H, 30H, 34H, 39H, 45H, 0DH, 0AH

slave:

ascii: &0004500400A8[CR][LF]

hex: 26H, 30H, 30H, 30H, 34H, 35H, 30H, 30H, 34H, 30H, 30H, 41H, 38H, 0DH, 0AH

2.5 write the slave (machine)

Host send:

Start code	Slave #	command	New #	checksum	End code	
@	xx	**7F**	xx	xx	CR	LF

Here the slave # can be any value (00~99)，the new # should be 00~09

Slave response:

Start code	Slave #	command	status	checksum	End code	
&	xx	**7F**	xx	xx	CR	LF

If the writing is successful，here the slave # should be the new #。

Ex: write slave # 0x01 successfully

host : ascii: @007F0180 [CR][LF]

hex: 40H, 30H, 30H, 37H, 46H, 30H, 31H, 38H, 30H, 0DH, 0AH

slave: ascii: &017F0080[CR][LF]

hex: 26H, 30H, 31H, 37H, 46H, 30H, 30H, 38H, 30H, 0DH, 0AH

2.7 Check if the slave is on line active

Host send:

Start code	Slave #	command	checksum	End code	
@	Xx	**80**	xx	CR	LF

Slave response:

Start code	Slave #	command	checksum	End code	
&	xx	**80**	xx	CR	LF

Ex: Check if slave #1is on line, and assume the slave #1 is on line

host : ascii: @01807F [CR][LF]

hex: 40H, 30H, 31H, 38H, 30H, 37H, 46H, 0DH, 0AH

slave: ascii: &01807F[CR][LF]

hex: 26H, 30H, 31H, 38H, 30H, 37H, 46H, 0DH, 0AH

2.8 Read slave system information

Host send:

Start code	Slave #	command	checksum	End code	
@	xx	**81**	xx	CR	LF

Slave response:

Start code	Slave #	command	# of bytes	Value 1	...	Value N	checksum	End code	
&	Xx	**81**	**17**	xx	...	xx	xx	CR	LF

System information total 17-byte，including hardware revision、software revision, and machine (ID)。

Software and hardware revision # is a two byte number，Ex. 100 = RV1.00。

Machine ID total 12-byte。

Table 2.8.1

Data	explanation
Data 1	Low byte of the hardware revision
Data 2	High byte of the hardware revision
Data 3	Low byte of the software revision
Data 4	Low byte of the software revision
Data 5	ID byte 0
Data 6	ID byte 1
Data 7	ID byte 2
Data 8	ID byte 3
Data 9	ID byte 4
Data 10	ID byte 5
Data 11	ID byte 6
Data 12	ID byte 7
Data 13	ID byte 8
Data 14	ID byte 9
Data 15	ID byte 10
Data 16	ID byte 11
Data 17	Reserved

Ex: read Slave #0 systeminformation，if hardware revision is v1.20，software revision is v1.00，ID is 6706291048486672066BFF52

host :

ascii: @01807F [CR][LF]

hex: 40H, 30H, 31H, 38H, 30H, 37H, 46H, 0DH, 0AH

slave:

ascii: &0081117800640052FF6B0672664848102906670 0C2[CR][LF]

hex: 26H, 30H, 30H, 38H, 31H, 31H, 31H, 37H, 38H, 30H, 30H, 36H, 34H, 30H, 30H, 35H, 32H, 46H, 46H, 36H, 42H, 30H, 36H, 37H, 32H, 36H, 36H, 34H, 38H, 34H, 38H, 31H, 30H, 32H, 39H, 30H, 36H, 36H, 37H, 30H, 30H, 43H, 32H, 0DH, 0AH

3 Registers

3.1 Register mapping

There are tree type of registers：read only（ro），read/write（rw），saved (nvm)。Saved registers are used to save settings，it will not lost even power off。

Table 3.1.1 Register address map

Name	Description	Address		Type	Reset Value
		Hex	Decimal		
MOT_TEMP_L	Current temperature LOW byte	00	0	ro	-
MOT_TEMP_H	Current temperature HIGH byte	01	1	ro	-
SPD_RPM_L	Current speed LOW byte	02	2	ro	-
SPD_RPM_H	Current speed HIGH byte	03	3	ro	-
DC_IK_L	Supply current LOW byte	04	4	ro	-
DC_IK_H	Supply current HGH byte	05	5	ro	-
DC_VOLT_L	Supply voltage LOW byte	06	6	ro	-
DC_VOLT_H	Supply voltage LOW byte	07	7	ro	-
STAT	System status	08	8	ro	-
ERR_COD	Error code	09	9	ro	-
HALL_VAL	Motor hall value	0A	10	ro	-
Reserved	Reserved(read only)	0B~4F		ro	
SPD_RPM_SET_L	Desired speed LOW-byte	50	80	rw	00
SPD_RPM_SET_H	Desired speed HIGH-byte	51	81	rw	00
CTRL_EN	Enable	52	82	rw	00
CTRL	Control byte	53	83	rw	00
Reserved	Do not modify	54~57		rw	-
ENC_TOT_CNT_0	32 encoder value byte 0	58	88	rw	00
ENC_TOT_CNT_1	32 encoder value byte 1	59	89	rw	00
ENC_TOT_CNT_2	32 encoder value byte 2	5A	90	rw	00

Name	Description	Address		Type	Reset Value
		Hex	Decimal		
ENC_TOT_CNT_3	32 encoder value byte 3	5B	91	rw	00
Reserved	Do not modify	5C~9F		rw	-
DEV_ADDR	Slave #	A0	160	nvm	-
ENC_EN	Encoder enable	A1	161	nvm	-
ENC_CPR_L	Encoder CPR LOW-byte	A2	162	nvm	-
ENC_CPR_H	Encoder CPR LOW-byte	A3	163	nvm	-
MOT_TS_EN	Thermistor enable	A4	164	nvm	-
MOT_TS_PARA_L	Thermistor parameter LOW-byte	A5	165	nvm	-
MOT_TS_PARA_H	Thermistor parameter LOW-byte	A6	166	nvm	-
MOT_TEMP_MAX	Motor Max. limit Temperature	A7	167	nvm	-
MOT_TEMP_TIM_L	Over temperature duration LOW-byte	A8	168	nvm	-
MOT_TEMP_TIM_H	Over temperature duration LOW-byte	A9	169	nvm	-
SPD_RPM_MAX_L	Max. speed setting LOW byte	AA	170	nvm	-
SPD_RPM_MAX_H	Max. speed setting HIGH byte	AB	171	nvm	-
IK_CHOKE	Stall current value	AC	172	nvm	-
IK_CHOKE_TM_L	Stall max. duration LOW-byte	AD	173	nvm	-
IK_CHOKE_TM_H	Stall max. duration LOW-byte	AE	174	nvm	-
IK_MAX	Max. current value	AF	175	nvm	-
IK_CONT_MAX	Max. continuous current value	B0	176	nvm	-
IK_CONT_TM_L	Max. continuous current duration LOW	B1	177	nvm	-
IK_CONT_TM_H	Max. continuous current duration High	B2	178	nvm	-
PID_KP_0	PID => P byte 0	B3	179	nvm	-
PID_KP_1	PID => P byte 1	B4	180	nvm	-
PID_KP_2	PID => P byte 2	B5	181	nvm	-
PID_KP_3	PID => P byte 3	B6	182	nvm	-
PID_KI_0	PID => I byte 0	B7	183	nvm	-
PID_KI_1	PID => I byte 1	B8	184	nvm	-
PID_KI_2	PID => I byte 2	B9	185	nvm	-
PID_KI_3	PID => I byte 3	BA	186	nvm	-
PID_KD_0	PID => D byte 0	BB	187	nvm	-
PID_KD_1	PID => D byte 1	BC	188	nvm	-
PID_KD_2	PID => D byte 2	BD	189	nvm	-
PID_KD_3	PID => D byte 3	BE	190	nvm	-
Reserved	Do not modify	BF~FF		nvm	

3.2 MOT_TEMP_L, MOT_TEMP_H

Motor temperature，Unit is ℃，saved in 16-bit data。

3.3 SPD_RPM_L, SPD_RPM_H

Motor speed，unit is RPM，saved in 16-bit data。

3.4 DC_IK_L, DC_IK_H

Supply current，unit is 0.1A。

3.5 DC_VOLT_L, DC_VOLT_H

Supply voltage，unit is 0.1V。

3.6 STAT

System status：

Table 3.6.1

value	Status
0x00	Initializing
0x01	Waiting
0x02	Running
0x03	Error
0x04	Brake

3.7 ERR_COD

Table 3.7.1

Value	Explanation
0x00	Normal
0x01	Hall sensor error
0x03	Enable signal error
0x04	Stall
0x06	Over temperature
0x08	Supply voltage low
0x09	Supply voltage low
0x0b	Encoder error

3.8 HALL_VAL

Motor hall sensor value。

3.9 SPD_RPM_SET_L, SPD_RPM_SET_H

Desired speed， unit is RPM，16-bit data。

3.10 CTRL_EN

Enable : 0 = voltage control；1 = RS485 command control。

3.11 CTRL

Control data，only when **CTRL_EN =** 1，this will mean：

value	Explanation
0x00	Stop，clear desired speed value
0x01	RUN，run at desired speed

3.12 ENC_TOT_CNT_0, ENC_TOT_CNT_1, ENC_TOT_CNT_2, ENC_TOT_CNT_3

Encoder value，32-bit data。

3.13 DEV_ADDR

This machine (slave) #。

3.14 ENC_EN

Enable the encoder: 0 = no encoder，1 = use encoder。

3.15 ENC_CPR_L, ENC_CPR_H

Encoder CPR value, 16-bti data。

3.16 MOT_TS_EN

Motor thermistor enable: 0 = no thermistor，1 = use thermistor。

3.17 MOT_TS_PARA_L, MOT_TS_PARA_H

Thermistor parameter?

3.18 MOT_TEMP_MAX

Max. allowed temperature，Unit is ℃。

3.19 MOT_TEMP_TIM_L, MOT_TEMP_TIM_H

How long period time is allowed to be over temperature, 16-bit data, unit is sec.。

3.20 SPD_RPM_MAX_L, SPD_RPM_MAX_H

Max. speed value，16-bit data, Unit is RPM。

3.21 IK_CHOKE

Stall current，8-bit data, unit is A。

3.22 IK_CHOKE_TM_L, IK_CHOKE_TM_H

Motor stall trigger time，unit is 0.1 秒。

3.23 IK_MAX

Max. current value，8-bit data, unit is A。

3.24 IK_CONT_MAX

Max. continuous current，8-bit data, unit is A。

3.25 IK_CONT_TM_L, IK_CONT_TM_H

Max. current allowed time，unit is sec.。??

3.26 PID_KP_0, PID_KP_1, PID_KP_2, PID_KP_3

Control parameter PID =>P，4-byte, PID_KP_0 is low byte，PID_KP_3 is highest byte。

3.27 PID_KI_0, PID_KI_1, PID_KI_2, PID_KI_3

Control parameter PID =>I，4-byte, PID_KI_0 is low byte，PID_KI_3 is highest byte。

3.28 PID_KD_0, PID_KD_1, PID_KD_2, PID_KD_3

Control parameter PID =>D，4-byte, PID_KD_0 is low byte，PID_KD_3 is highest byte。

4 Revision history

Table 4.1 Document revision history

Date	Revision	Changes	Author
2017-05-29	1	Initial release.	Paul Sk.

Special Communication commands for RS485 real time control

Host send “+250[CR]” 5 ASCII code, the motor1 will response by executing the forward speed 250rpm and send the current encoder “4 digit encoder data + [CR] “ 5 ASCII code back to host. Therefore, total 10 ASCII codes are used to make one control cycle for one motor.

Special command style 1：

Host send	Start	3 digit speed code			end
Motor 1 +speed	+	low	Middle	high	CR
Motor 1 response	4 digit encoder data (speed)				end
Motor 1 encoder	low	xx	Xx	high	CR
Host send	Start	3 digit speed code			end
Motor 2 +speed	>	low	middle	high	CR
Motor 2 response	4 digit encoder data (speed)				end
Motor 2 encoder	low	xx	xx	high	CR
Host send	Start code	3 digit speed code			end
Motor 1 -speed	-	low	middle	high	CR
Motor 1 response	4 digit encoder data (speed)				end
Motor 1 encoder	low	xx	xx	high	CR
Host send	Start code	3 digit speed code			end
Motor 2 -speed	<	low	middle	high	CR
Motor 2 response	4 digit encoder data (speed)				end
Motor 2 encoder	low	xx	xx	high	CR

Total 10 ASCII code each time / unit (driver)

Special command style 2:

			16 bit desired speed			
	start	**slave #**	**set_rpm_low**	**set_rpm_high**	**checksum**	**end**
Host Send	*	XX	XX	XX	XX	CR

Slave Response	32 bit encoder count				8 bit supply current (unit 0.1 A)	err_code	checksum	end
	enc_cnt_0	**enc_cnt_1**	**enc_cnt_2**	**enc_cnt_3**	**current_low**			
	XX	XX	XX	XX	XX	XX	XX	CR

EX:

Send command to #0 driver, speed 300(0X012C), assume at this time the encoder value is 1234(0X000004D2), current is 5.6A(0X[00]38), [status code 0X02], Error code 0X00.

Host :

ASCII:"*00 2C 01 2D[CR]"

Hex: 2AH 30H, 30H, 32H, 43H, 30H, 31H, 32H, 44H, 0DH

(NOTES: checksum= 0x00+0x2c+0x01 = 0x2D)

Slave:

ASCII：D2040000, 38[00], [02], 00, 10[CR]

Hex:
44H,32H,30H,34H,30H.30H,30H,30H,33H,38H,30H,30H,30H,32H,30H,30H,31H,30H,0DH

(NOTES: checksum= 0XD2+OX04+0X00+0X00+0X38+[0X00+0X02+]0X00 = 0x10)