CANcoder Howto

rev 1.12

Introduction:

The CANcoder is a circuit board which counts incremental quadrature encoder pulses and sends values to a CAN master. The circuit is suitable for mounting directly on an encoder to digitize the device. The device accepts TTL/CMOS single-ended quadrature signals, and can also accepts differential if half the signal wiring is used.

The CANcoder communicates on the CAN bus with a minimal implementation of the CANopen CiA-406 protocol. It supports 32 bit quadrature decoding, pulse counting, index pulse, and value preset functions. LEDs indicate node ID and communications status. Device configuration is achieved via an RS232 terminal port or CANopen LSS communication.

This documentation pertains to version 1B of the CANcoder.

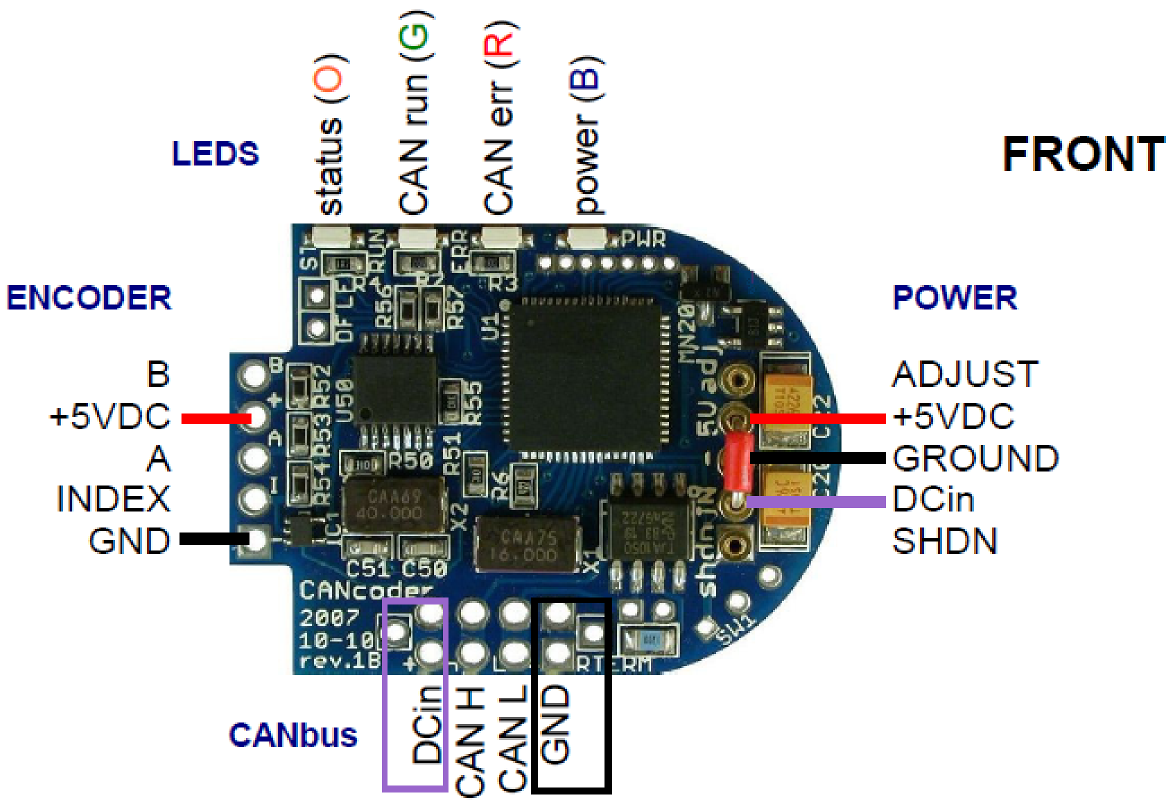


Figure 1

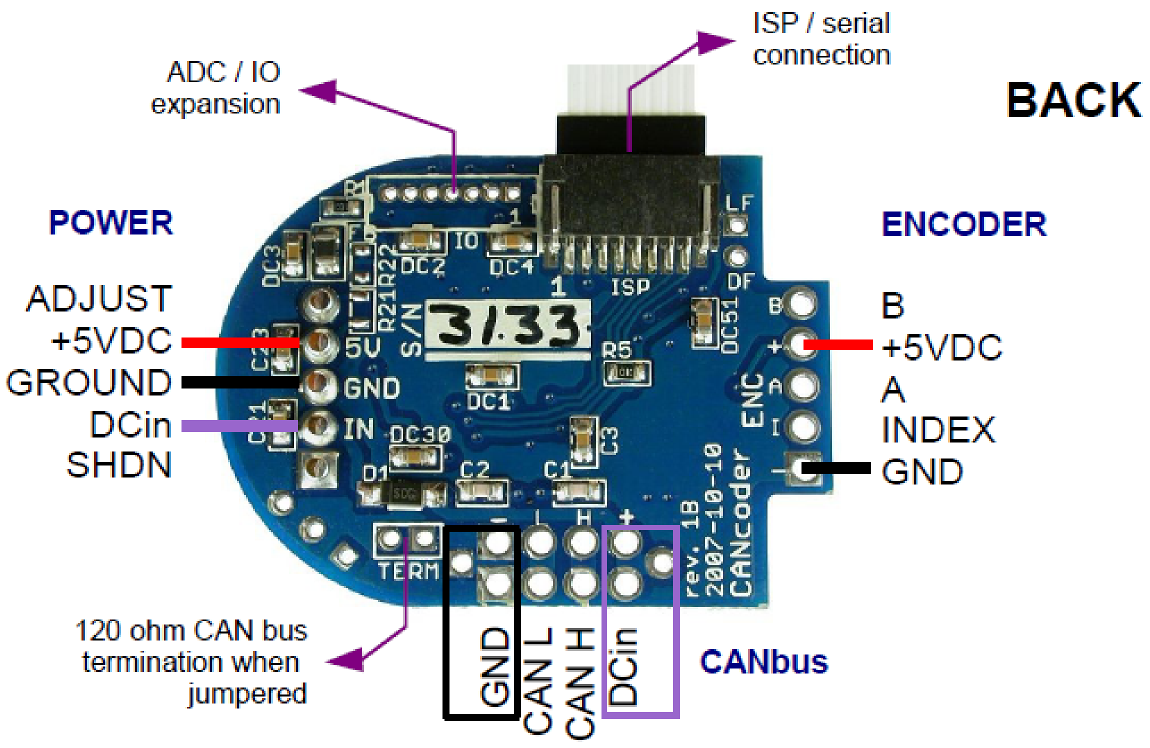


Figure 2

Mechanical Information:

PCB dimensions: 1.46 x 1.10 x .210" (37 x 28 x 5.334mm)
The Cancoder is designed to be the same size as a small encoder and can be mounted on models such as the US digital S1-1024-NTHS.

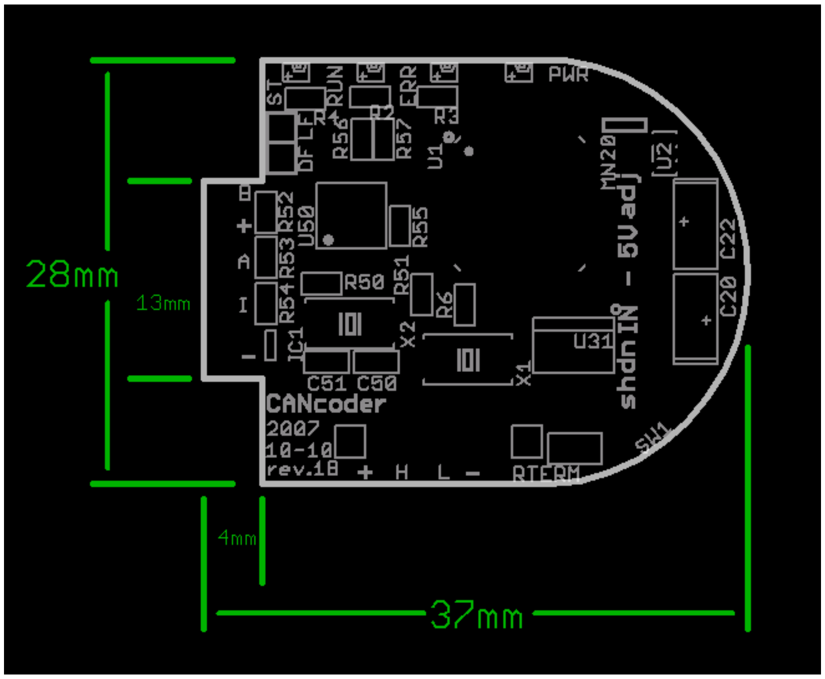


Figure 3

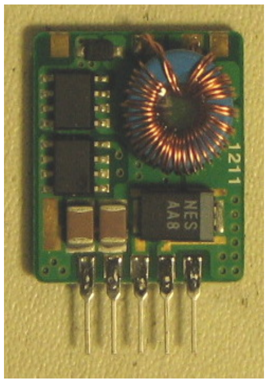


Figure 4

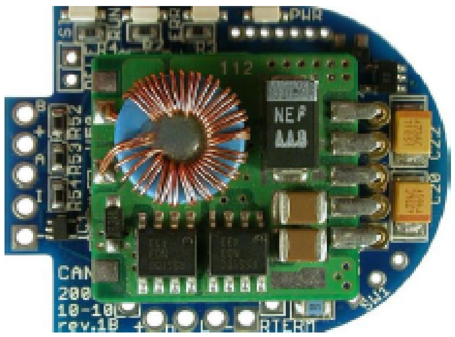


Figure 5

<u>Power & Electrical Specifications:</u>

The device is powered from either 5 VDC power or an optional 6-16.5V DCDC mezzanine, shown above in figure 4 & 5. The CANcoder draws approximately 47mA of power from a five volt supply.

5V without Mezzanine:
Five volts is applied at the “DCin” pads with silkscreen designations “+” and “-”, seen in figures 1 and 2. Additionally, the 5v and the IN pins on the Power socket must be jumpered, as shown with a red wire in Figure 1.

With Power Mezzanine:
The power mezzanine plugs into the the “power socket” area on the top side of the board. When the power mezzanine is installed the board can be powered with 6v-16.5V through “DCin” pads seen in figures 1 and 2. A mezzanine is shown in figure 4. Figure 5 shows the power mezzanine installed on the CANcoder.

The on-board microcontroller is industrial temp rated at -40 to +85 C.
The quadrature chip has an operating temperature of -25 to +80 C.
If the IDS112 DC/DC supply is used its rated temperature range is -25 to +70 C.

The CANcoder contains overvoltage protection circuitry.

<u>Other Features:</u>

- 32-bit quadrature decoder counter
- x1, x2 or x4 mode of quadrature counting
- index pulse support
- reset or preset load on index supported
- digital filtering
- up to 10MHz quadrature clock rate
- up/down pulse counting (non-quadrature)
- up to 40MHz count frequency
- CAN bus rate up to 1Mbps

Connectors:

On the back of the board there is a TTL serial port which is used for configuration of the device, such as its CAN node ID and bit-rate. A USB serial dongle is available for connection to a PC running a terminal emulator program.

1. The pads in the “Encoder” area are where quadrature signals are input.
Pads as seen in Figure 1 and Figure 2 :

    Signal B input
    5v power output
    Signal A input
    Index pulse input
    Ground

2. The pads in the CANbus area of Figure 1,2 are for CAN Low, CAN High, and DC Power input.

3. The ADC IO connector of Figure 1,2 is for digital input. See section “IO Port” below.

Encoder and CANcoder ground must be common.

LED indicators

There are 4 LEDs on the CANcoder (see Figure 1).

The blue LED indicates that the board has power.

The green LED means that a CAN packet has been transmitted. When the CANcoder connects to a CAN master the blinking will become noticeably faster and may look at if it is on constantly.

The orange LED indicates status. If you have the digital IO port enabled the orange LED blinks (for more info on IO, see programming section). The Orange LED will also blink when an index pulse arrives from the Encoder.

The red LED may blink when the board is powered up, but it is currently not used.

IO Port

The CANcoder implements four digital inputs (analog inputs not supported at this time).

Pinout of connector “ADC/IO expansion”, seen in Figure 2:

1. Ground
2. AVCC out
3. Digital input 1 / ADC 4
4. Digital input 2 / ADC 5
5. Digital input 3 / ADC 6
6. Digital input 4 / ADC 7
7. Reset

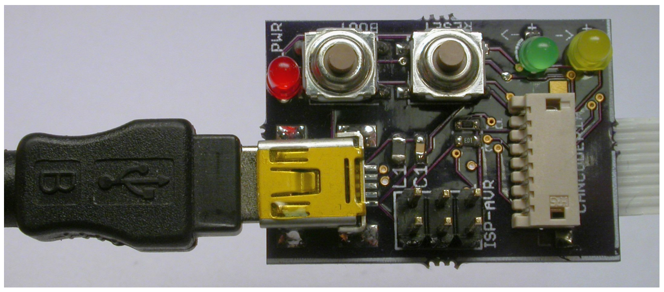
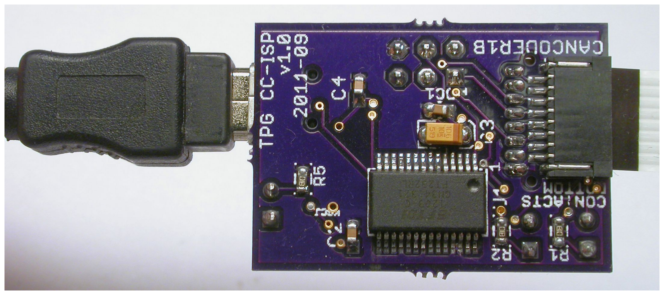
<u>ISP/Serial Programming Port</u>

The 8-pin flex cable has 8-pins:

1 GND
2 VCC
3 SCK
4 PDI (input for ISP) RXD receive of microcontroller
5 PDO (output for ISP) TXD transmit of microcontroller
6 RESET\
7 SDA (for I2C)
8 SCL (for I2C) -- also BOOT_BUTTON

(This interface is used for configuration via a serial terminal)

Programming

USB-serial Programming Dongle

For most Windows machines, the driver will be automatically installed for the dongle. If a driver is not found automatically, find the appropriate driver for OS at:
http://www.ftdichip.com/FTDrivers.htm (follow “VPC Drivers” link)
Once the driver is successfully installed, determine to which COM port the USB-serial dongle is connected. Go to Control Panel->System->Hardware->Device Manager (or equivalent). Under 'Ports (COM & LPT)', search for “FT232R USB UART” and note the COM port associated with it (COM1, COM2, etc).

Configuring CANcoders

1. Connect a serial programming cable to the ISP port of the CANcoder, and connect the other end to the serial port of a PC.
2. Launch a terminal emulator program such as Hyperterm or Minicom. The serial settings should be configured as : 115200 8-N-1, no flow control.
3. Power on CANcoder, banner information from the CANcoder should display.
4. See the table below for the complete list of available commands.
5. The following parameters are commonly changed :

- menu 0 parameter 2 – CAN node ID
- menu 5 parameter 3 – CAN data rate

6. To enable digital inputs, see Configuration for Digital Inputs section.

Command Summary :

command	description	notes
help	Help	
?	Help	
ls	List modules	
lp	List presets	
rp	Read preset into SRAM	
wp	Write preset: <module> <source> <dest>	
set	Set parameter value	
load_presets	Load all presets from EEPROM	
restore_presets	Restore default presets from PROGMEM	
save_presets	Save all configs to EEPROM preset	
setn	Set parameter value numerically	
getn	Get parameter value numerically	
reset	reset board	
boot	Jump to bootloader	

Configuration Example:

1. Consult the next table to determine which menu the target parameter is. For example, to change the node ID of the CANcoder, the menu entry is: 0 2.
2. Type :
   ls 0
   then press Enter to show the current settings of menu 0.
3. The current value of a menu entry is indicated on the right side, enclosed by square brackets.
4. To change the node ID of a CANcoder to 6 for example, type :
   setn 0 2 6
   then press Enter.
5. Type :
   save
   then press Enter to save any changes.
6. Restart the CANcoder.
7. Type:
   ls 0
   press Enter, and verify that the value of menu entry 0 2 is 6.

Internal Menus (accessible via serial terminal):

0 [SYS] optical encoder interface

param	#ch	type	name	desc	use	min	max	val
0	1	0	debug	debug mode - serial output	0	0	1	0
1	1	1	nodes	number of enabled CAN nodes	2	0	1	1
2	1	1	startID	CAN starting node ID	2	0	127	2
3	1	1	NMTclk	NMTboot interval (16.384ms incr)	2	0	255	7
4	1	1	ODto	Overdrive timeout (16.384ms incr)	2	0	255	76
5	1	0	CANinit	initialize CAN bus	0	0	1	1
6	1	0	ENCinit	initialize ENCODER chips	0	0	1	1
7	1	4	systime	system time	1	0	4294967295	0
8	1	8	sysname	System name	0	1	16	CANcoder default

1 [NODE] CAN NODE setup

param	#ch	type	name	desc	use	min	max	val
0	4	0	connect	connected flag	1	0	1	0
1	4	1	state	current NMT state	1	0	255	0
2	4	4	timeout	timeout time	1	0	4294967295	7

2 [IO] IO NODE setup

param	#ch	type	name	desc	use	min	max	val
0	1	0	enable	enable IO CAN node output	0	0	1	0
1	1	1	node_id	CAN node ID for IO	0	0	64	2
2	1	1	num	number of button bits to check	0	0	2	2
3	1	1	pullups	1=pullup 0=none - per axis	0	0	255	0
4	1	1	polarity	1=invert 0=normal - per axis	0	0	255	0
5	1	1	max_checks	Maximum # of checks for pin debouncing	0	1	16	10
6	1	1	PDOtype	1=TX after SYNC, 255=on-change	0	0	255	1
7	1	1	timer	event timer (milliseconds) for auto TX	0	0	255	0
8	1	1	state	debounced button state	1	0	255	0
9	1	0	debug	debug mode - serial output of IO	0	0	1	0

3 [ENC] encoder chip interface

param	#ch	type	name	desc	use	min	max	val
0	1	1	MDR0	Mode Register 0 (C3h disable idx, D3h pos idx, 93h neg idx)	0	0	255	195
1	1	1	MDR1	Mode Register 1 (10h=32bit mode w/flag on INDEX)	0	0	255	16
2	1	1	LEDidx	LED on index pulse (0=none 1=enabled)	0	0	1	1
3	1	0	loadPre	Load preset at startup (1=enabled)	0	0	1	1
4	1	2	readENCtime	uS to read ENC when polled	1	0	65535	0
5	1	4	CANrx	CAN packets received	1	0	4294967295	0
6	1	4	CANtx	CAN packets transmitted	1	0	4294967295	0
7	1	0	invert	Invert index (1=enabled)	0	0	1	1

4 [ENCaxis] ENC axis setup

param	#ch	type	name	desc	use	min	max	val
0	4	4	preset	Preset reload value	0	0	16777215	8388607
1	4	4	min	Minimum	0	0	16777215	0
2	4	4	max	Maximum	0	0	16777215	16777215

5 [CAN] CAN bus interface

param	#ch	type	name	desc	use	min	max	val
0	1	0	en	enable CAN interface	0	0	1	1
1	1	1	debug	debug level	0	0	255	0
2	1	1	node	starting node ID	0	1	127	1
3	1	1	baud	Baudrate 1=1000, 2=500, 3=250, 4=125 (Kbps)	0	1	3	1
4	1	1	TQ	time quantum 0=8 1=16	2	0	1	1
5	1	4	CANrx	CAN packets received	1	0	65535	0
6	1	4	CANtx	CAN packets transmittted	1	0	65535	0
7	1	2	delay	delay after each ASCII cmdline pkt	2	0	10000	500
8	1	2	TXerr	transmit fail error count	1	0	65535	0

CANOpen information:

When the CANcoder is powered on, the CANcoder will continuously send an NMT Bootup Message at 9 Hz. This message is intended to signal the CAN master that a CAN device is present on the CAN bus. The period between NMT Bootup Messages can be configured for a delay of up to 4 seconds.

The NMT bootup message is formatted as follows:

COB-ID	Length	Data
0x700 + CAN ID	1	0x00

Upon receiving a NMT bootup message from, the CAN master will send a Start Remote Node message to the CAN device. The Start Remote Node message 'activates' the CAN device and allows the device to send PDOs.

The Start Remote Node message is formatted as follows:

COB-ID	Length	Data	
0x000	2	0x01	CAN ID

The CAN ID in the Start Remote Node message refers to the CAN ID of the CAN device that the CAN master wants to activate.
At this point, the CANcoder is ready to send PDOs. By default, CANcoders are configured to send PDOs only in response to a SYNC packet.

The CAN master sends a SYNC packet with the following format:

COB-ID	Length
0x080	0

CANcoders are able to send up to 2 types of PDOs.

Encoder count data is sent as TPDO2, and is formatted as follows:

COB-ID	Length	Data
0x280 + CAN ID	4	Encoder count (32 bits)

CAN data is formatted little endian, so the first byte is the least significant byte, and the last byte is the most significant byte.

Digital input data, if enabled, is sent as TPDO1, and is formatted as follows:

COB-ID	Length	Data	
0x180 + CAN ID	2	8	digital input data

The digital input data byte is composed of:

msb							lsb
x	x	x	x	ADC7	ADC6	ADC5	ADC4

Note1: CANcoders can have different IDs for the encoder node versus the digital input node.
Note2: CANcoders can be configured via serial terminal to send a PDO only when the value of the PDO data changes.

The state of the red LED on the CANcoder board is settable through the least significant bit of Object Dictionary Index 2000h. By default, the red LED starts out cleared.

To set the red LED, send the following message on the CAN bus

COB-ID	Length	Data							
0x600 + CAN ID	8	0x2F	0x00	0x20	0x00	0x01	0x00	0x00	0x00

To clear the red LED, send the following message on the CAN bus

COB-ID	Length	Data							
0x600 + CAN ID	8	0x2F	0x00	0x20	0x00	0x00	0x00	0x00	0x00

Programming with CANopen LSS

The following LSS operations change the node ID of a CANcoder. These instructions utilize a CAN terminal application.

1. Connect the CANcoder to the CAN bus such that it is the only device on the
bus. Be sure that the CAN bus is properly terminated with a 120 Ohm resistor at
each end.

2. Launch a CAN terminal program and apply power to the CANcoder. In the
received packets window of the CAN packet terminal you will likely see a stream
of information of the form:

```
   0x07NN 1 0x00
```

These packets will continue to stream as you communicate with the device, but
they are harmless and may be ignored.

3. Enter each of the
following packets into the transmit packet window as shown.

a. Enter the following packet in the transmit window and press 'Send':

```
    0x07e5 8 0x04 0x01 0x00 0x00 0x00 0x00 0x00 0x00
```

Note that sending the above packet will not result in a response from the CANcoder.

b. Send the following packet to query the current node ID:

```
    0x07e5 8 0x5e 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

The following response should be received:

```
    0x07e4 8 0x5e 0xNN 0x00 0x00 0x00 0x00 0x00 0x00
```

(where NN is the current node ID)

c. Send the following packet to set the new node ID to 7, for example

```
    0x07e5 8 0x11 0x07 0x00 0x00 0x00 0x00 0x00 0x00
```

The node ID is in the fourth word above. node ID is in hexadecimal.

The following response packet indicates a success:

```
    0x07e4 8 0x11 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

The following response packet indicates an inadmissible node ID:

```
    0x07e4 8 0x11 0x01 0x00 0x00 0x00 0x00 0x00 0x00
```

d. Send the following packet to permanently save the new configuration:

```
    0x07e5 8 0x17 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

the following response packet indicates a successful save:

```
    0x07e5 8 0x17 0x00 0x00 0x00 0x00 0x00 0x00 0x00
```

4. Cycle the power on the CANcoder to test the new configuration.

<u>Troubleshooting serial configuration with Overdrive control systems*:*</u>

Normally the Overdrive control panel will set up serial ports and programs automatically, but if manual configuration is required:.

1. Using an Overdrive system, make sure The Real Time OS is not using the serial port. To verify, do the following:
    1. Launch a terminal.
    2. Su- to root.
    3. Type 'lm', press Enter. Check if mocon is running.
    4. If mocon is running, type 'unload_mocon.sh', press Enter.
    5. On FC1 systems: Type 'rmmod rt_com',
       On FC7 systems: rmmod rtl_serial_16550,
       press Enter .
    6. Type 'restore_serial.sh on', press Enter.
2. Check if minicom is connecting to the proper serial port. You may need to run minicom as root to change the serial port.

Configuration for Digital Inputs:

There are four digital inputs are available on the CANcoder, which can be found on the JTAG/IO port.

The following parameters are relevant for using digital inputs with the CANcoder:

Menu	Parameter	Name	Value
2	0	enable	0 – disables digital input TPDO 1 – enables digital input TPDO
2	1	node_id	ID – should match value set in menu 0 parameter 2
2	2	num	0 – digital inputs disabled 8 – digital inputs enabled
2	3	pullups	0 – internal pullup disabled 1 – internal pullup enabled
2	4	polarity	0 – logic follows voltage 1 – logic is inverted
2	6	PDOtype	1 – send TPDO every SYNC (recommended) 255 – send TPDO on value change

Configuration as Step-Direction Counter:

The CANcoder can also be configured to count Step and Direction signals. In this mode, the Step signal is expected on pin A, and the Direction signal is expected on pin B.

The following parameters are relevant for using the CANcoder as a step and direction counter:

Menu	Parameter	Name	Value
3	0	MDR0	0 – step and direction 195 – 4x quadrature, no index

This device made with RoHS compliant materials.