Revision: 4.5
Date: 2017-03-23

CrossLink BTC CANopen

User manual

maximatecc

www.maximatecc.com

Contents

Revision history

Rev	Date	Author	Comments
1.0	2009-11-03	Adam Eklund	First revision
2.0	2009-11-20	Adam Eklund	Added LED feedback information and some minor changes.
2.1	2010-03-03	Adam Eklund	Minor change to OD index 0x2004
3.0	2011-01-20	Adam Eklund	- Added index 0x2006 and 0x2010 to OD. - Changed inquiry scan result read example according to new software.
4.0	2011-02-01	Adam Eklund	- Changed manual graphical profile - Added “Principles of operation” - Added response times for SDO responses
4.5	2013-05-29	Fredrik Wahlström	- Added Extended Scanning - Filter out and removed “UNKOWN” units - Added possibility to set CAN ID > 10

1. Functionality

The following functions are supported by CrossLink BTC CANopen:

- Start inquiry scan
- Read inquiry scan result
- Connect to a specified node
- Read connection status
- Break an active connection
- Set CAN ID with hardware switch
- Set CAN ID using SDO
- Set Bluetooth ID using SDO
- Set CAN baud rate with hardware switch
- Automatically read stored inquiry information at startup
- Remove nodes from inquiry scan result
- Read software version via CAN

1.1. Rotary selectors

CrossLink BTC CANopen has two rotary selectors with 10 positions each. They are used to set CAN ID and CAN baud rate. The left selector (seen from the front) sets CrossLink BTC CANopen CAN ID and the right set CAN baud rate. The table below describes the different rotary selector settings.

Rotary selector position	CAN ID
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	CAN ID controlled from Software, (default 10)

Table 1. CAN ID rotary selection

Rotary selector position	CAN baud rate (Kbit/s)
0	1000
1	800
2	500
3	250
4	125
5	100
6	50
7	20
8	Reserved
9	Reserved

Table 2. CAN baud rate rotary selection

1.2. LED indicators

LED Description	Condition	Meaning
Bluetooth	Blinking Blue at 20 Hz	Initializing
	Blinking Blue at 1 Hz	Preoperational mode - CrossLink is initialized and waiting for command
	Blinking Blue at 5 Hz	Scanning - CrossLink is performing an inquiry scan
	Blinking Blue at 10 Hz	Connecting - CrossLink is trying to connect
	Solid Blue	Operational - CrossLink is connected
	Flashing Blue	Communication error
CAN	Solid Green	Communication OK
	Solid Red	Communication error

Table 3. LED indicators

1.3. Bluetooth naming convention

Bluetooth names of CrossLink BTC CANopen nodes are based on the Bluetooth ID written to the object dictionary. Each CrossLink BTC CANopen node has the Bluetooth name “CLBTCnn” where nn = Bluetooth ID. For example CrossLink BTC with Bluetooth ID 2 would have Bluetooth name “CLBTC2”.

After a successful inquiry scan, the Bluetooth names of the found nodes can be read from the object dictionary.

1.4. Principles of operation

This chapter gives describes operation states, parameters and functionality of CrossLink BTC CANopen.

1.4.1. Operational states

CrossLink BTC CANopen operates in three defined states, *initialization, preoperational* and *operational.* Each state offers different functionality where the current module state is indicated by the Bluetooth status LED. In the chapters below the different states and their purpose and functionality is described.

1.4.1.1. Initialization state

In the initialization state all hardware is initialized, the object dictionary is set up and, if necessary, the Bluetooth chip is reconfigured. If the Bluetooth ID has been changed since last restart, the Bluetooth chip needs to be reconfigured which increases the node initialization time. During the initialization phase SDO communication is not possible.

1.4.1.2. Preoperational state

When the initialization phase is complete or when a connected node is disconnected, it enters preoperational state. A transition from initialization to preoperational is signaled by a CANopen boot-up message. In this state all SDO functionality is available.

1.4.1.3. Operational state

The transition from preoperational to operational automatically occurs when a node has an active Bluetooth connection. If the node’s connection status is “Connected”, it is in operational state and only a subset of the SDO functionality is available. The possible functions are shown in Table 4. Object dictionary.

1.4.2. Functionality

CrossLink BTC CANopen offers functionality to find, connect and communicate with other nodes using Bluetooth. To control the node, a CAN protocol based on CANopen is used. In this chapter, node functionality and behavior which can be useful for application development is described.

1.4.2.1. Configuring

It is possible to configure CAN baud rate, identifier and Bluetooth identifier for each node. The CAN baud rate and identifier is set by the rotary selectors as described in Rotary selectors. Changes to baud rate and identifier for CAN are activated the next time the node is started. The CAN identifier is used as a base for all SDO communication and the current CAN ID can be determined by the boot-up message which is 0x700 + CAN ID. For a walkthrough of the SDO protocol, see SDO communication protocol.

The Bluetooth ID is changed using SDO communication. The new ID is not activated until the node is restarted and changing the Bluetooth ID triggers a chip reconfiguration the next reboot which prolongs the initialization time.

1.4.2.2. Scanning

An inquiry scan is triggered by and SDO write request. The scan status is indicated by an entry in the object dictionary and the Bluetooth LED also indicates an ongoing scan. The scan status entry could be read continuously by an application to determine when the scan is completed and this entry is reset every time the node is reset. There are two types of scans, normal inquiry scan and extended inquiry scan. The differences between these two are the initial scanning time, normal inquiry scan is scanning in 10s and extended inquiry scan is scanning in 30s. However the scan time varies depending on the number of new nodes found, this because each new node has to have its name resolved. Nodes are identified by their unique Bluetooth address and nodes already in the inquiry scan list will not have their names resolved, hence saving a lot of scan time. When new nodes are found, they are added to the inquiry scan list which is accessible from the object dictionary. Each node in the list is identified by its name which is composed according to Bluetooth naming convention. The inquiry scan result list is stored in non-volatile memory and is restored each time the node is started which makes it possible to connect to a node without first having to perform a time consuming scan. A full scan is only necessary when a node is added to the network or if a node changes Bluetooth ID and the inquiry scan list needs to be updated. To update the entire list there are functions to remove one or all of the found nodes from the list. This function can be used when for example a node in a network needs to be replaced where you remove the replaced node from the inquiry scan list, connect the new node and perform a new scan to add the new node to the list.

1.4.2.3. Connecting

Connecting to a node is done by writing its sub index in the inquiry scan list to the connection entry in the object dictionary. The connection status can be read from the object dictionary and is also signaled by the Bluetooth LED. When connected the node enters operational state and only a subset of the SDO functionality is permitted. By writing to the object dictionary disconnect entry the nodes are disconnected and returns to preoperational state.

1.4.2.4. Communicating

In the operational state all non-SDO CAN messages are transmitted wireless between the nodes. The CAN baud rates can be different between two communicating nodes, but the busload must not exceed the maximum limit for the lowest baud rate. If the communication busload is too high this is indicated by the CAN LED.

2. CANopen protocol description

This section describes the CrossLink BTC CAN protocol. The CrossLink BTC CAN protocol implements a subset of the CANopen standard for command and parameter transmission called SDO upload/download. Using the SDO protocol makes it possible to read and write data from/to a node. All functionality is defined as indexes and sub indexes in the object dictionary which can be read or written using CAN. In chapter SDO communication protocol there are details of how to read and write each object dictionary entry.

2.1. Object Dictionary (OD)

Index	Sub index	Type	Default	Response time (ms)	Description
0x100A	0x00	str, ro	"a.b.c.d"	0	**Software version** a,b,c and d is major, minor, release and build numbers.
0x2000	0x00	u8, rw	0x00	0	**Inquiry scan**** 0x0x = Start inquiry scan where x are a multiple of 10s. Valid values of x are 1-6, which implicates scanning in 10s-60s.
0x2001	0x00	u8, ro	0x00	0	**Inquiry scan status** 0x00 = No scan performed 0x01 = Scanning 0x02 = Inquiry scan performed
0x2002	0x00	u8, ro	0x00	0	**Inquiry scan result** Number of CrossLink BTC CANopen nodes found
	0x01	str, ro	""	0	Name of first node
	n	str, ro	""	0	Name of n:th node
0x2003	0x00	u8, rw	0x00	0	**Connect to node**** Use sub index from inquiry scan result (0x2002) to select node to connect to.
0x2004	0x00	u8, ro	0x00	0	**Connection status*** 0x00 = Not connected 0x01 = Trying to connect 0x02 = Connected 0x03 = Connection failed
0x2005	0x00	u8, rw	0x00	2200	**Disconnect*/****

					0x01 = break active connection
0x2006	0x00	u8, rw	0x00	1600	**Remove nodes from inquiry scan result**** 0 – remove all nodes 1:n – Use sub index from inquiry scan result (0x2002) to remove specific node. Inquiry scan result list will automatically be adjusted so nodes with higher indexes than the one removed are shifted to lower indexes.
0x2010	0x00	u8, rw	0x01	1600	**Bluetooth ID** The default Bluetooth ID is 1. Valid values are 1 – 99. If the Bluetooth ID is changed with an SDO, then the new Bluetooth ID will be activated when the unit is restarted.
0x2011	0x00	u8, rw	0x0a	1600	**CANOpen ID** When CANOpen ID rotary switch is set in position 9 the CANOpen ID is controlled via the OD. Valid values are 1 – 127. If the CANOpen ID is changed with an SDO, then the new CANOpen ID will be activated when the unit is restarted.

Table 4. Object dictionary

* = Function is also available when the node has an active connection

** = Value is automatically reset when action is performed

2.2. SDO communication protocol

A simple description of how the CANopen SDO communication protocol if used to read and write entries in the object dictionary. For the complete specification, see CANopen DS-301 specification.

2.2.1. Write 1 - 4 bytes

Write request

```
CAN ID:     0x600 + CAN ID
Byte 0:     0x2F (write 1 byte, u8)
            0x2B (two bytes, u16)
            0x27 (three bytes)
            0x23 (four bytes, u32)
```

```
Byte 1:     Low byte of OD index
Byte 2:     High byte of OD index
Byte 3:     OD sub index
Byte 4-7:   Data. Size depends on the number of bytes in OD entry
```

Write response

```
CAN ID:     0x580 + CAN ID
Byte 0:     0x60 (indicates success)
Byte 1:     Low byte of OD index
Byte 2:     High byte of OD index
Byte 3:     OD sub index
Byte 4-7:   Set to 0
```

Example

Start inquiry scan = write 1 to OD index 0x2000 sub index 0x00 to node with ID = 5.

Send the following message:

```
CANID:      0x605
Byte 0:     0x2F
Byte 1:     0x00
Byte 2:     0x20
Byte 3:     0x00
Byte 4:     0x01
```

If everything works ok you get the following response message:

```
CAN ID:     0x585
Byte 0:     0x60
Byte 1:     0x00
Byte 2:     0x20
Byte 3:     0x00
Byte 4-7:   0x00
```

2.2.2. Read 1 - 4 bytes

Read request

```
CAN ID:     0x600 + CAN ID
Byte 0:     0x40
Byte 1:     Low byte of OD index
Byte 2:     High byte of OD index
Byte 3:     OD sub index
```

Read response

```
CAN ID:     0x580 + CAN ID
Byte 0:     0x4F (if one byte in response, u8)
            0x4B (two bytes, u16)
            0x47 (three bytes)
            0x43   (four bytes, u32)
Byte 1:     Low byte of OD index
Byte 2:     High byte of OD index
Byte 3:     OD sub index
Byte 4-7:   Read data. Data in little endian format (LSB first)
```

Example

Read inquiry scan status, index 0x2001 sub index 0x00 u8, from node with ID = 5.

Send the following message:

```
CAN ID:     0x605
Byte 0:     0x40
Byte 1:     0x01
Byte 2:     0x20
Byte 3:     0x00
```

If everything works ok you get the following response message:

```
CAN ID:     0x585
Byte 0:     0x4F
Byte 1:     0x01
Byte 2:     0x20
Byte 3:     0x00
Byte 4:     0x02   (Inquiry scan performed)
```

2.2.3. Read more than 4 bytes

When reading OD entries containing more than 4 bytes of data (e.g. strings) segmented data transfer needs to be used. When reading more than 4 bytes the data block is divided into several segments. Each segment is requested with a request message and the response message contains up to 7 data bytes. For CrossLink BTC reading more than 4 bytes will be necessary when reading node names from inquiry scan result. When reading more than 4 bytes the segment request message must be sent within 5 seconds or the SDO read operation is cancelled. Sending any other message than segment request also cancels read operation.

Read request

```
CAN ID:     0x600 + CAN ID
Byte 0:     0x40
Byte 1:     Low byte of OD index
Byte 2:     High byte of OD index
Byte 3:     OD subindex
```

Response

```
CAN ID:     0x580 + CAN ID
Byte 0:     0x41
Byte 1:     Low byte of OD index
Byte 2:     High byte of OD index
Byte 3:     OD sub index
Byte 4 - 7: Number of bytes to be transferred in little endian
            format (LSB first)
```

Segment request 1

```
CAN ID:     0x600 + CAN ID
Byte 0:     0x60 (for odd requests)
```

Segment response 1

```
CAN ID:     0x580 + CAN ID
Byte 0:     0xXX. XX varies depending on data size
Byte 1 - 7: Data in little endian format (LSB first)
```

Segment request 2

```
CAN ID:     0x600 + CAN ID
Byte 0:     0x70 (for even requests)
```

Segment response 2

```
CAN ID:     0x580 + CAN ID
Byte 0:     0xXX. XX varies depending on data size
Byte 1 - 7: Data in little endian format (LSB first)
```