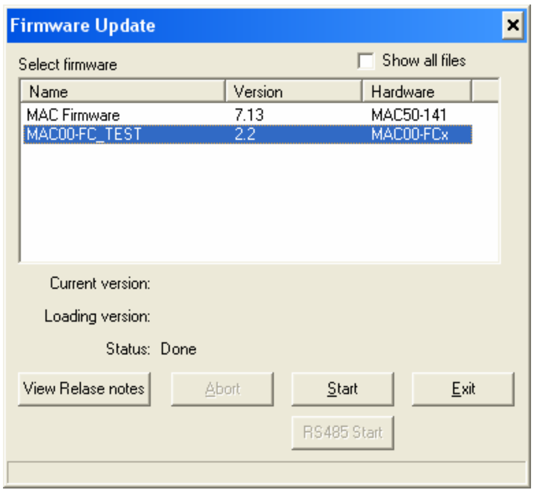
Using user-defined Receive PDOs with the MAC00-FCx modules

From firmware version 2.2 Beta 1, the Receive PDO24 can be configured for different data formats. This allows one 16-bit motor register and one 32-bit motor register to be received via PDOs. Make sure to upgrade to the V2.2 firmware in MacTalk as shown below. The firmware version of the MAC motor is less important – any version released after the summer 2006 should work well for CANopen.



Firmware Update

Select firmware

□ Show all files

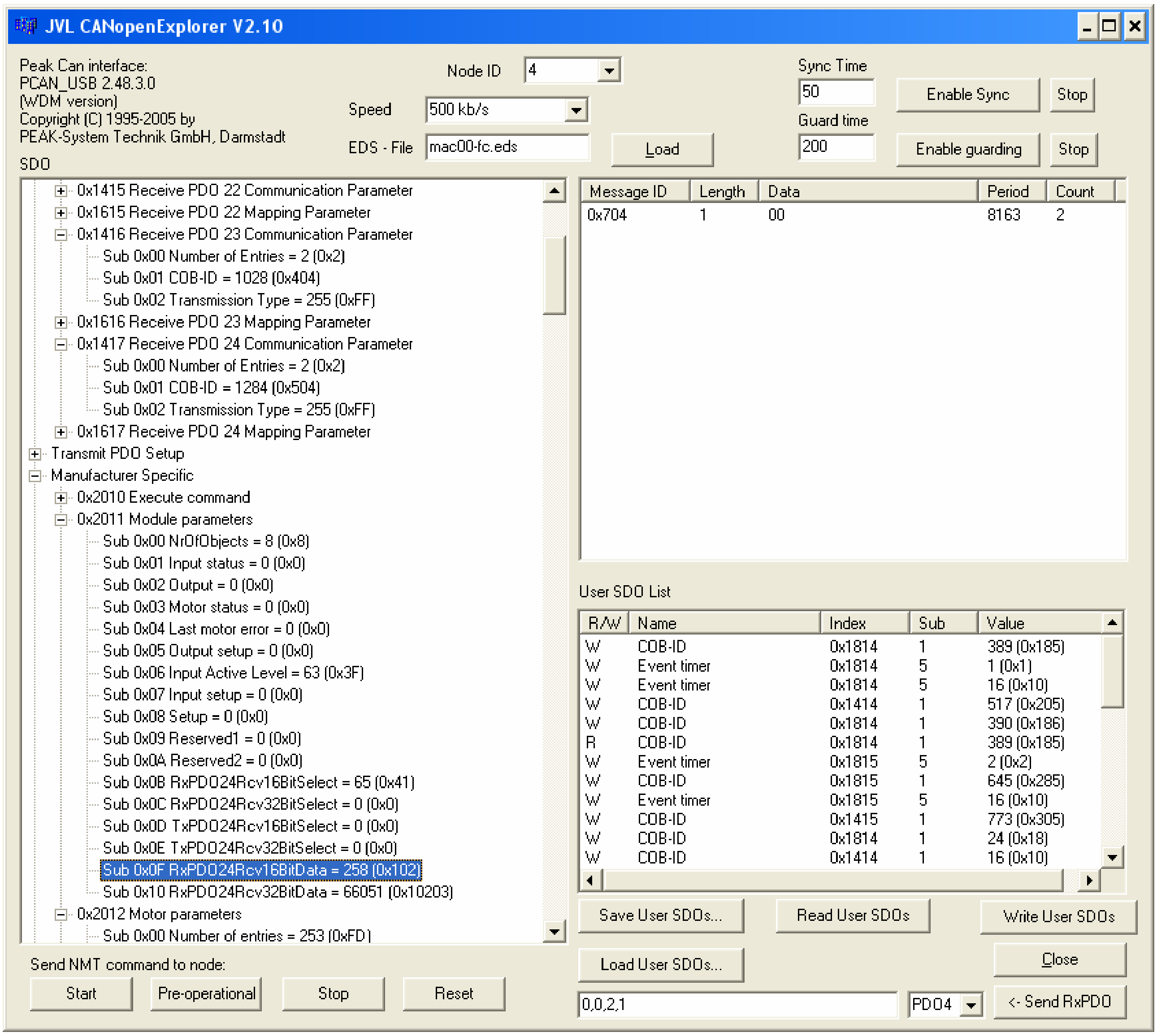
Name	Version	Hardware
MAC Firmware	7.13	MAC50-141
MAC00-FC_TEST	2.2	MAC00-FCx

Current version:

Loading version:

Status: Done

The setup is illustrated with the help of a screenshot from the JVL utility CANopen Explorer:



JVL CANopenExplorer V2.10

Peak Can interface:
PCAN_USB 2.48.3.0
(WDM version)
Copyright (C) 1995-2005 by
PEAK-System Technik GmbH, Darmstadt

Node ID: 4

Speed: 500 kb/s

EDS - File: mac00-fc.eds

Sync Time: 50

Guard time: 200

SDO

```
⊞ 0x1415 Receive PDO 22 Communication Parameter
⊞ 0x1615 Receive PDO 22 Mapping Parameter
⊟ 0x1416 Receive PDO 23 Communication Parameter
    Sub 0x00 Number of Entries = 2 (0x2)
    Sub 0x01 COB-ID = 1028 (0x404)
    Sub 0x02 Transmission Type = 255 (0xFF)
⊞ 0x1616 Receive PDO 23 Mapping Parameter
⊟ 0x1417 Receive PDO 24 Communication Parameter
    Sub 0x00 Number of Entries = 2 (0x2)
    Sub 0x01 COB-ID = 1284 (0x504)
    Sub 0x02 Transmission Type = 255 (0xFF)
⊞ 0x1617 Receive PDO 24 Mapping Parameter
⊞ Transmit PDO Setup
⊟ Manufacturer Specific
  ⊞ 0x2010 Execute command
  ⊟ 0x2011 Module parameters
    Sub 0x00 NrOfObjects = 8 (0x8)
    Sub 0x01 Input status = 0 (0x0)
    Sub 0x02 Output = 0 (0x0)
    Sub 0x03 Motor status = 0 (0x0)
    Sub 0x04 Last motor error = 0 (0x0)
    Sub 0x05 Output setup = 0 (0x0)
    Sub 0x06 Input Active Level = 63 (0x3F)
    Sub 0x07 Input setup = 0 (0x0)
    Sub 0x08 Setup = 0 (0x0)
    Sub 0x09 Reserved1 = 0 (0x0)
    Sub 0x0A Reserved2 = 0 (0x0)
    Sub 0x0B RxPDO24Rcv16BitSelect = 65 (0x41)
    Sub 0x0C RxPDO24Rcv32BitSelect = 0 (0x0)
    Sub 0x0D TxPDO24Rcv16BitSelect = 0 (0x0)
    Sub 0x0E TxPDO24Rcv32BitSelect = 0 (0x0)
    Sub 0x0F RxPDO24Rcv16BitData = 258 (0x102)
    Sub 0x10 RxPDO24Rcv32BitData = 66051 (0x10203)
  ⊟ 0x2012 Motor parameters
    Sub 0x00 Number of entries = 253 (0xFD)
```

Message ID	Length	Data	Period	Count
0x704	1	00	8163	2

User SDO List

R/W	Name	Index	Sub	Value
W	COB-ID	0x1814	1	389 (0x185)
W	Event timer	0x1814	5	1 (0x1)
W	Event timer	0x1814	5	16 (0x10)
W	COB-ID	0x1414	1	517 (0x205)
W	COB-ID	0x1814	1	390 (0x186)
R	COB-ID	0x1814	1	389 (0x185)
W	Event timer	0x1815	5	2 (0x2)
W	COB-ID	0x1815	1	645 (0x285)
W	Event timer	0x1815	5	16 (0x10)
W	COB-ID	0x1415	1	773 (0x305)
W	COB-ID	0x1814	1	24 (0x18)
W	COB-ID	0x1414	1	16 (0x10)

Send NMT command to node:

0,0,2,1 PDO4

Object 0x2011, SubIndex 0x0B and SubIndex 0x0C are used to select which motor register will receive the values written with RxPDO24.

Three basic formats are possible.

When both sub-indices are zero (default after power-up), RxPDO24 will work in backwards compatible way, with a length of one byte, that sets the modules digital outputs.

When SubIndex 0x0B is non-zero, but SubIndex 0x0C is zero, the length of the PDO must be 4 bytes, and the first byte is used to set up the module outputs. The second byte is not used. The third and fourth byte hold the 16-bit value (with the low byte first), that will be written to the motor register number 2..250, selected by SubIndex 0x0B. For example, writing the value 65 (decimal) to Object 0x2011, SubIndex 0x0B will cause data written with Receive PDO24 to be saved to motor register 65, V1.
When SubIndex 0x0C is non-zero, the length of PDO24 must be 8 bytes. The first four bytes work as described above, except that the 16-bit data will not be written to any motor register is SubIndex 0x0B is zero. Bytes 4, 5, 6 and 7 (lowest bytes first) are written as a 32-bit value to the motor register number stored in SubIndex 0x0C. For example a value in SubIndex 0x0C of 49 (decimal) will cause the last four bytes in Receive PDO 24 to be written to motor register 49, P1.

It is not necessary to change the PDO mapping objects for this to work. Also the firmware does not modify the mapping objects at index 0x16xx.

The new SubIndices 0x0E and 0x0F are generally not used in production, but can be used for development. These hold the latest 16-bit and 32-bit values received on RxPDO24 – before they are written to the basic motor.

Note that the 32-bit value is written to the basic motor before the 16-bit value, if both are used in the same PDO. This allows operations like updating a 32-bit Position register, and then execute a FastMac command immediately after that by writing the FastMac command value to the Command register Reg211.

<place description of user-defined TxPDOs here>

/This document was last updated on 3rd April 2009, NJG.