How to capture and use EtherCAT trace data with WireShark

1. Objective

There is a general miss understanding about EtherCAT. EtherCAT is a transport tool only. We use it to carry data from a CNC (PLC, EtherCAT master) to the drive. CanOpen over EtherCAT (CoE), CanOpen is the actual protocol that commands the drive.

In my experience, the EtherCAT part of the fieldbus works 99% of the time. Most Customers' issues are with the setup of CanOpen. What data is needed for Interpolated Position or Cyclic synchronous position? How do I setup to read the motor torque?

And the most frustrating issue is troubleshooting why it doesn't work? Is the problem in the CNC or in the drive? Who can tell me what is wrong? How do I collect the needed data so someone can help me?

So, my objective is to outline a simple way to collect the CoE traffic, between the CNC and the drive, so it can be emailed to Kollmorgen technical support.

2. Hardware and Software

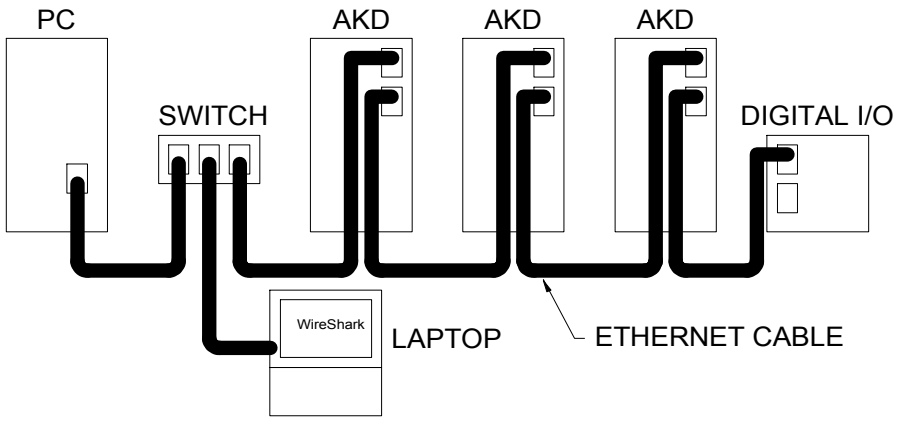
Unmanaged switch: You need a device that can be connected between the CNC and the I/O (drives, digital I/O, anything on the EtherCAT network). I have tested Netgear FS105 "Unmanaged Supper fast Ethernet Port" and it worked perfectly. No setup or "mirroring" needed. (Mirroring is needed for Modbus, EIP, anything TCP/IP based) At the time of writing this note, the FS105 was listed for $18 online.

NOTE: EtherCAT is not a standard office TCP/IP protocol. You just need a low cost, unmanaged, dumb switch. The reason the FS105 works is that it doesn't try to read or manage the message. "What comes in goes out to all". A more complicated switch requires setup and/or could even be damaged.

Copy of **WireShark software**: It can be found (Free) at https://www.wireshark.org/ Make sure you get it from the WireShark.org website. During the installation, it will ask you if it can install a couple of windows support programs. You should let it. If your unsure, get help! WireShark is well known with the IT profession and your IT people can answer your questions

3. Setup

Install the Unmanaged Switch between the CNC and the first node on the network. In my case, I had a PC with EtherCAT master software installed, three AKD drives, and a digital I/O module. The switch has to be in the middle and never at the end of the network.



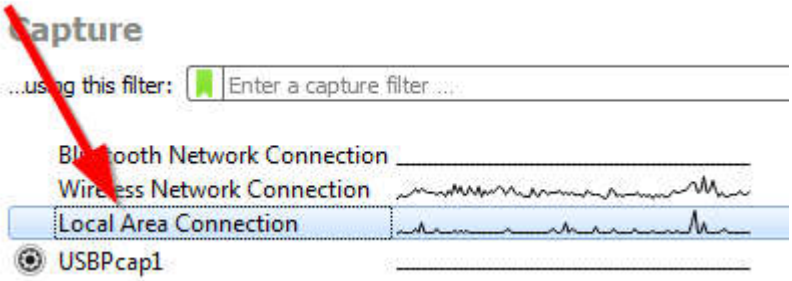
Plug in the laptop, with WireShark software, into the switch as well.

NOTE: Laptop is better than a Desktop PC. Desktop PC has an earth ground that tie into the Ethernet port. You could create a ground loop between the CNC and your PC.

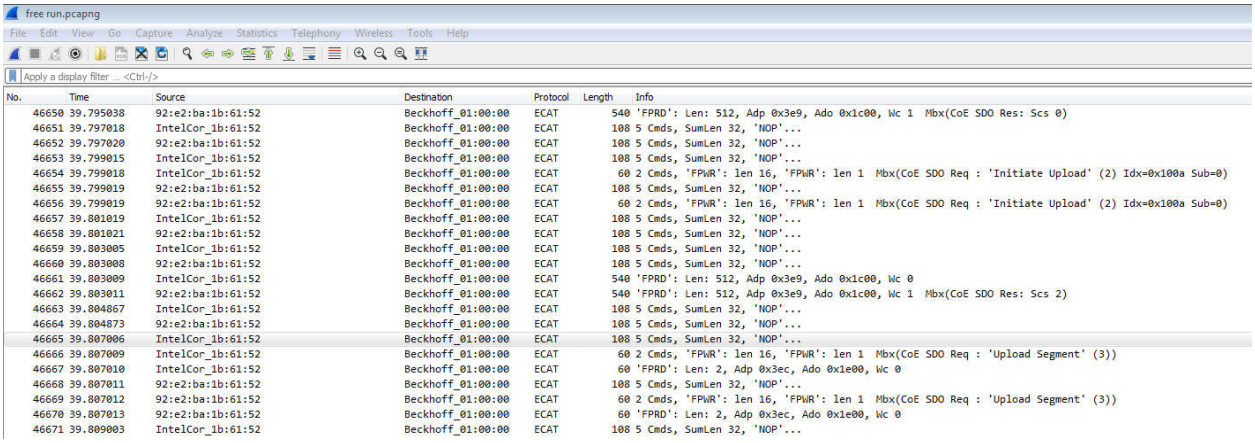
NOTE: Never try and mix EtherCAT with Ethernet. Damage can happen. There should be only three cables in the switch. The laptop should be direct to the switch.

4. Making and saving the trace

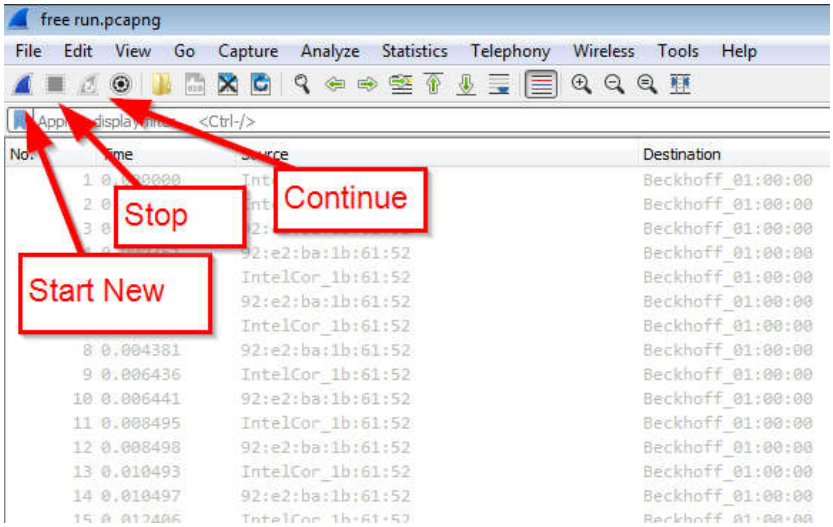
When you open the WireShark software, you should see a screen asking you which port you want to capture. Double click on the “Local connection”. If you have more than one “local connection”, pick the simple one that is linked to the port you installed the cable to the switch. (Otherwise, try them all until you find one that works)



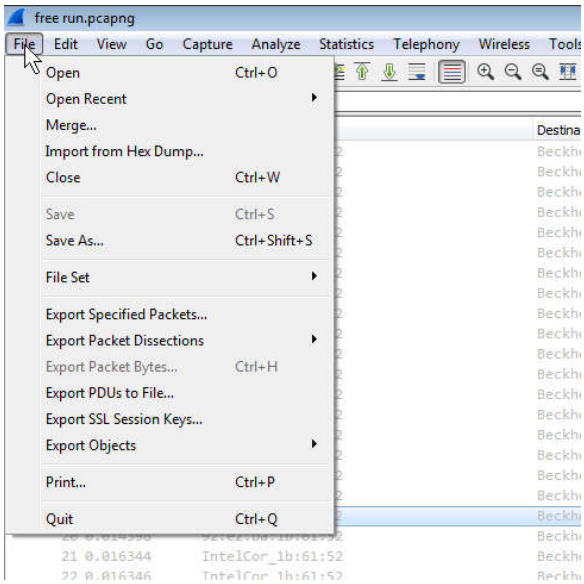
If all is well, you should see a screen that is full of EtherCAT messages



Typical EtherCAT network makes 1000 ~ 2000 messages a second. That is a huge amount of data to weed through if your in a hurry. There is a Stop/Start button at the top of the WireShark screen so you can do a better job of sequencing the capture with the event you want to capture.



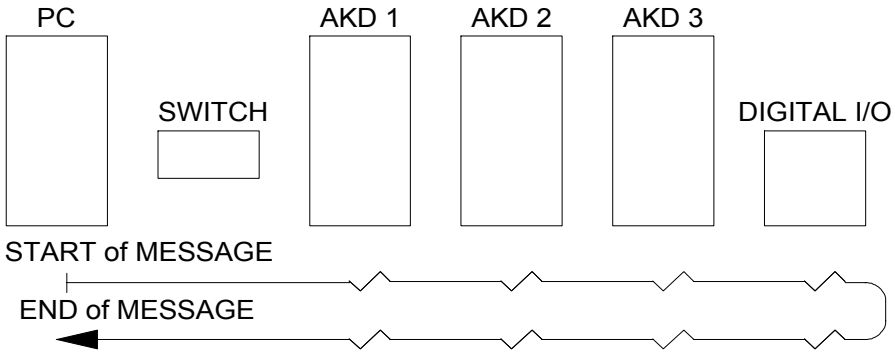
And, just like most Windows programs, there is a “FILE” on the pull down menu that will let you “save as” the file.



If you want to read the CoE data yourself

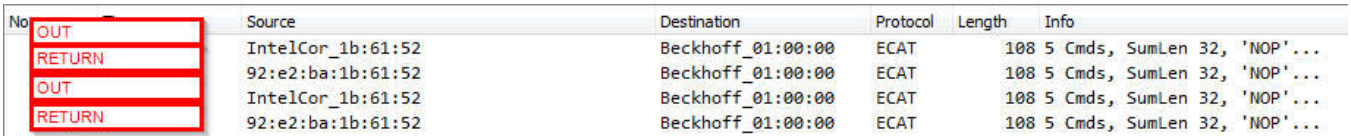
5. EtherCAT Operating Principle and Topology

The “secret sauce” of EtherCAT is that there is only one message, each node having a data region in that message, and that each node process the message “on the fly”. The flow of this “one message” is out to the last node and back. The “step” in my drawing shows that, even though the message passes through the node, there is a slight delay of the data entering the node to when that same data exits the node. (nS but it exists)



In this topology, the front of the message could be passing through the 3ed AKD while the tail of the message is passing through the 1st AKD.

The WireShark trace, that you captured, will show the message twice. Once on the way out and once on the way back.

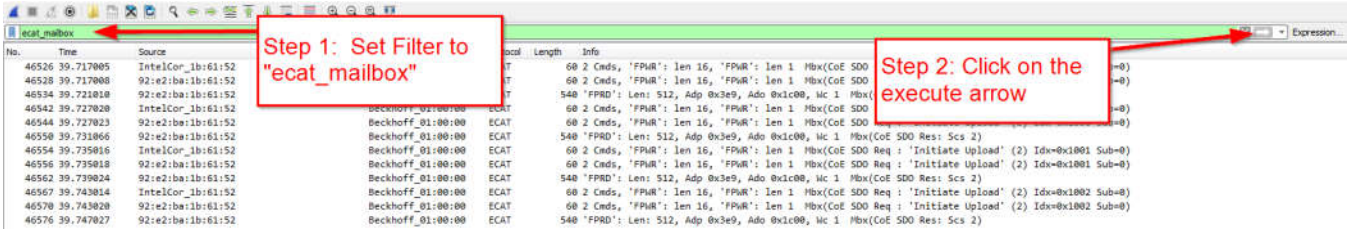


6. Reading the CoE SDO (Mailbox) data from a WireShark trace (Expert)

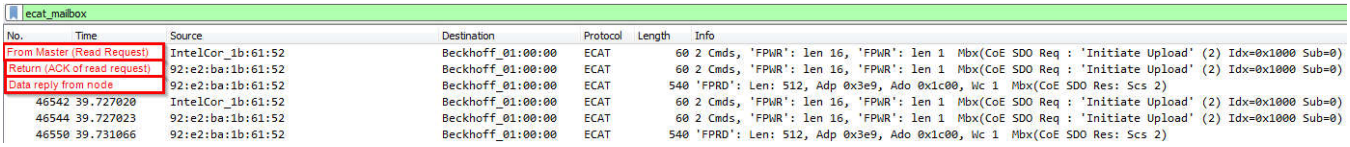
In CoE, SDO are delivered and returned using a Data Link type called “MailBox” or “MailBox protocol”. There can be multiple messages in one MailBox. The EtherCAT message, with the MailBox, is sent separate from the Process data message or PDO.

You will need to capture a trace with SDO activity in it. SDO messages don’t happen very often in CoE. In a typical application, there is a lot of SDO activity at the startup of the system and then only if your program specifically does a SDO read/write.

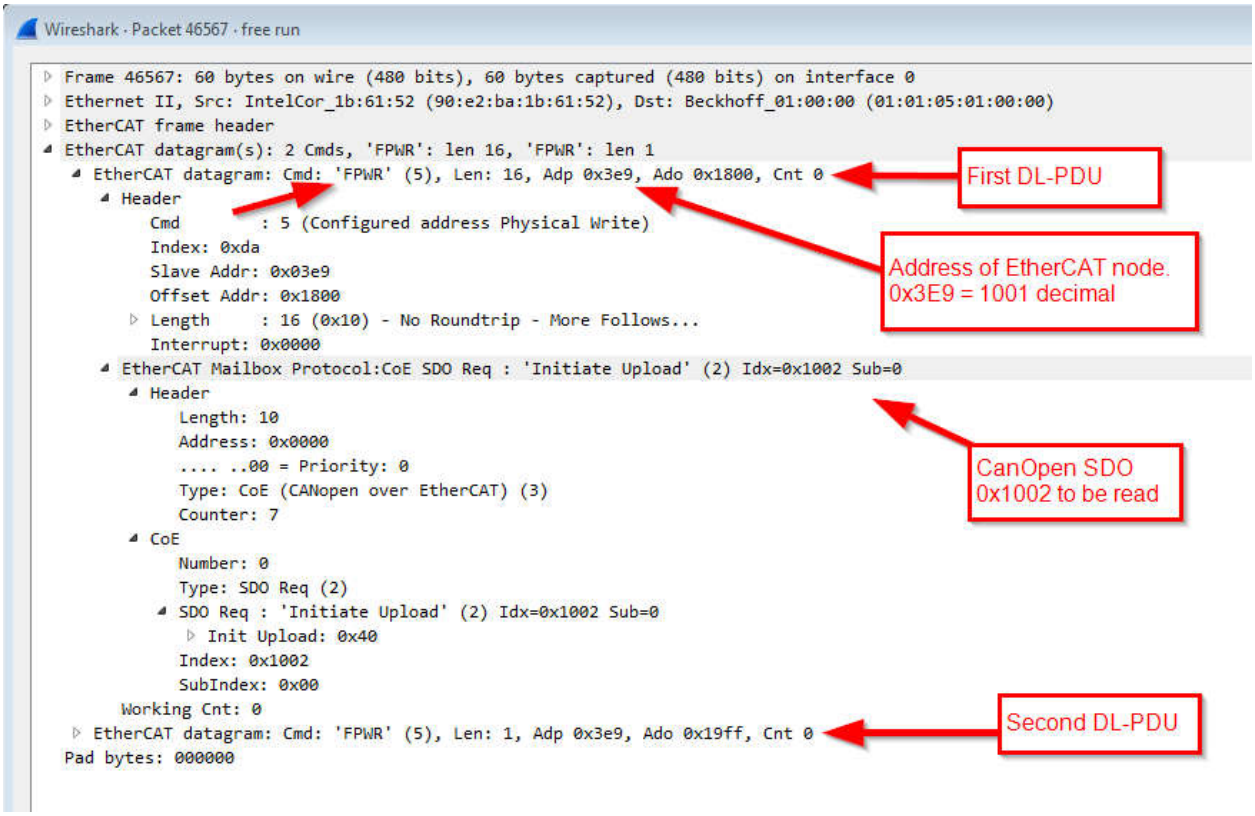
Once you have EtherCAT data being captured, apply the filter command “ecat_mailbox”



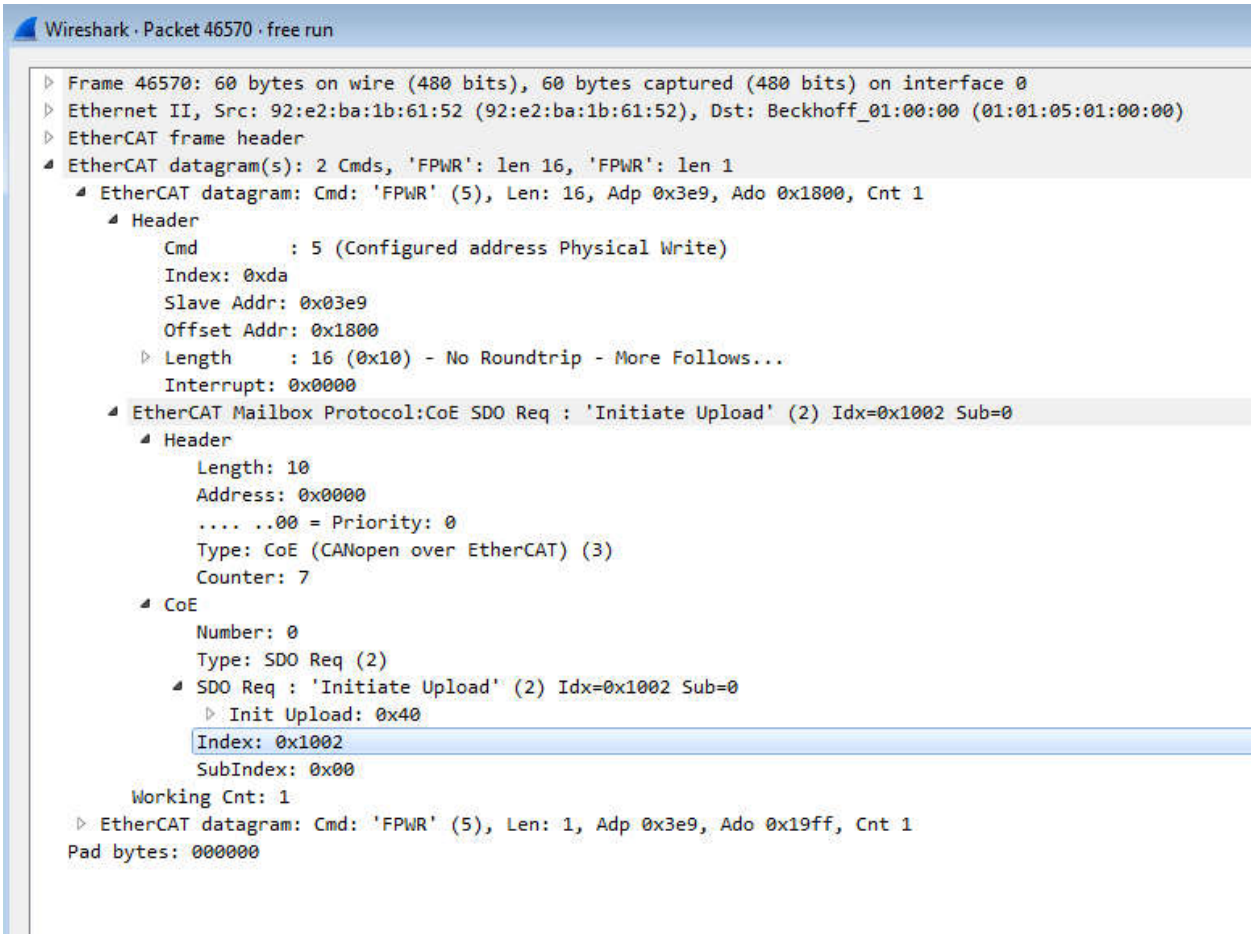
The read the information, you first need to identify



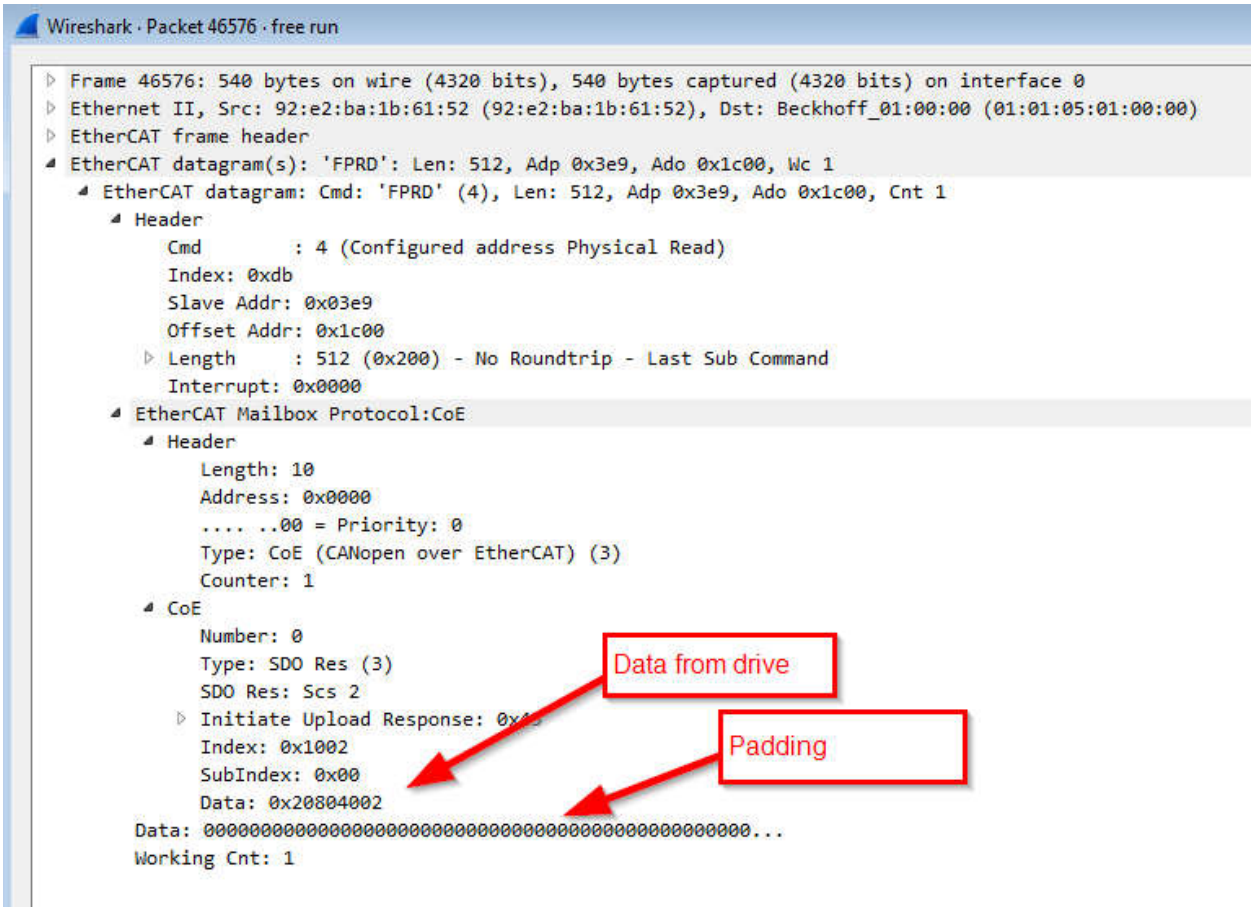
Double click on the message from the EtherCAT master. The read request uses the “Data Link Protocal Data Unit”, or DL-PUD for short, type “FPWR”. There can be more than one SDO message in the one EtherCAT message. Here, I have expanded just one of two.



And there is a response from the node (slave) that it received the message. Becaue it takes the node time to process the request, the data is send in a separate message.



And the data response follows. The AKD's default MailBoax size is 512 bytes. The drive needs to pad or fill in the unused space in the message with "00".



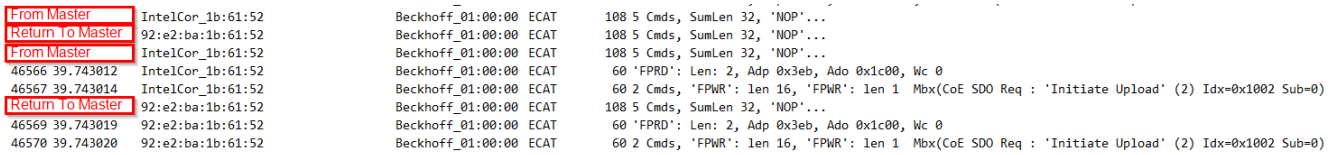
CanOpen uses “Octet” date in “Little Endian” sequence. The data (above) is grouped in sets of two hexadecimal numbers and then reversed. So, data 0x20804002 is read as 02 40 80 20. Plug that into your Windows Calculator (set calculator’s view for programmer) and you get 37781536 decimal or 10010000001000000000100000 binary. CanOpen object 0x1002 is a bit word and you can find the decode table in the AKD CanOpen manual.

7. Reading CoE PDO data from a WireShark trace (Expert)

You will need to remove the “ecat_mailbox” filer (if you set it reading SDOs).

PDO communication means the producer and the consumer already knows what the data means and how to truncate it into useful parameters. In the case of the AKD used in this example: The drive receives 6 bytes of data (0x6040 and 0x60C1sub1) and the drive sends 6 bytes of data (0x6041 and 0x6064)

Your looking for EtherCAT datagram “NOP” messages.



Double clicking on the “From Master” message, there is a lot of different DL-PDUs being sent to the drives. The PDO data is located in the “LRW”.

```
▷ Frame 46563: 108 bytes on wire (864 bits), 108 bytes captured (864 bits) on interface 0
▷ Ethernet II, Src: IntelCor_1b:61:52 (90:e2:ba:1b:61:52), Dst: Beckhoff_01:00:00 (01:01:05:01:00:00)
▷ EtherCAT frame header
◢ EtherCAT datagram(s): 5 Cmds, SumLen 32, 'NOP'...
  ▷ EtherCAT datagram: Cmd: 'NOP' (0), Len: 4, Adp 0x0, Ado 0x900, Cnt 0
  ▷ EtherCAT datagram: Cmd: 'ARMW' (13), Len: 4, Adp 0x0, Ado 0x910, Cnt 0
  ▷ EtherCAT datagram: Cmd: 'LRW' (12), Len: 4, Addr 0x1000000, Cnt 0
  ◢ EtherCAT datagram: Cmd: 'LRW' (12), Len: 18, Addr 0x1000800, Cnt 0
    ◢ Header
        Cmd        : 12 (Logical ReadWrite)
        Index: 0x00
        Log Addr: 0x01000800
      ▷ Length     : 18 (0x12) - No Roundtrip - More Follows...
        Interrupt: 0x0000
      Data: a89511011f00a1c303001f00489efaff1f00
      Working Cnt: 0
  ▷ EtherCAT datagram: Cmd: 'BRD' (7), Len: 2, Adp 0x0, Ado 0x130, Cnt 0
```

What your looking at is all the data to all the nodes. (the nodes know how to truncate the data) So, to make this data meaningful, you need to divide it into groups of two hexadecimal numbers and then reversed. So 0xA89511011F00A1C303001F00489EFAFF1F00 is read as 00 1F FF FA 9E 48 00 1F 01 30 30 1C 0A F0 11 10 51 95 A8. My AKD drive's PDO mapping is 6 bytes each. This long string of hexadecimal will need to be divided again in to groups of 6 octets.

Drive 1: 00 1F FF FA 9E 48 (0x6040 = 0x001F and 0x60C1sub1 = 0xFFFA9E48)
Drive 2: 00 1F 01 30 30 1C
Drive 3: 0A F0 11 10 51 95
Digital I/O: A8

And the return message is read the same way.

```
▷ Frame 46564: 108 bytes on wire (864 bits), 108 bytes captured (864 bits) on interface 0
▷ Ethernet II, Src: 92:e2:ba:1b:61:52 (92:e2:ba:1b:61:52), Dst: Beckhoff_01:00:00 (01:01:05:01:00:00)
▷ EtherCAT frame header
◢ EtherCAT datagram(s): 5 Cmds, SumLen 32, 'NOP'...
  ▷ EtherCAT datagram: Cmd: 'NOP' (0), Len: 4, Adp 0x0, Ado 0x900, Cnt 0
  ▷ EtherCAT datagram: Cmd: 'ARMW' (13), Len: 4, Adp 0x4, Ado 0x910, Cnt 3
  ▷ EtherCAT datagram: Cmd: 'LRW' (12), Len: 4, Addr 0x1000000, Cnt 3
  ◢ EtherCAT datagram: Cmd: 'LRW' (12), Len: 18, Addr 0x1000800, Cnt 9
    ◢ Header
        Cmd        : 12 (Logical ReadWrite)
        Index: 0x00
        Log Addr: 0x01000800
      ▷ Length     : 18 (0x12) - No Roundtrip - More Follows...
        Interrupt: 0x0000
      Data: a19511013712a5c3030037124d9efaff3712
      Working Cnt: 9
  ▷ EtherCAT datagram: Cmd: 'BRD' (7), Len: 2, Adp 0x4, Ado 0x130, Cnt 4
```

Raw Data	a1	95	11	01	37	12	a5	c3	03	00	37	12	4d	9e	fa	ff	37	12
Reversed and truncated	12	37	ff	fa	9e	4d	12	37	00	03	c3	a5	12	37	01	11	95	a1
CanOpen Object	0x6041		0x6064				0x6041		0x6064				0x6041		0x6064			
Node	Drive 1						Drive 2						Drive 3					

(The I/O module was only setup as "outputs" and is not mapped for inputs)

8. References

WireShark: https://www.wireshark.org/
"ecat" filter commands: https://www.wireshark.org/docs/dfref/e/ecat.html

EtherCAT: https://www.ethercat.org
CanOpen: https://www.can-cia.org/