Beckhoff EK9000 Notes

IP Address: 192.168.1.6

IP Address set on pc: 192.168.1.10

Subnet Mask: 255.255.252.0

Links to Beckhoff website:

EK900: https://www.beckhoff.com/en-us/products/i-o/ethercat-terminals/ekxxxx-bus-coupler/ek9000.html

EL5042: https://www.beckhoff.com/en-us/products/i-o/ethercat-terminals/el5xxx-position-measurement/el5042.html

- Took a while, but connecting worked. web address is: 192.168.1.6/config (works with or without connecting to wifi)
- If connecting from dev network, namrata used http://ek-602e17/config , on DHCP
- DIP switched 9 and 10 set to 0 (to get IP address and not DHCP) , DIP switched 2 and 3 are at 1 to get 00000110 in binary, this equates to 6 in decimal.

Modbus:

- Based on a master slave. Master sends request and slave responds with a response.
- Modbus data can be carried on two channels - over serial communication, or over a network connection. This flexibility is part of the lasting appeal of Modbus.
- There are two types of Modbus serial protocols, RS-232 and RS-485. Modbus RS-232 allows concurrent, full-duplex flow of data. Modbus RS-485 is half-duplex, and indicates values using differences in voltage.
- Modbus messages can also be sent over Ethernet or TCP/IP. These Modbus messages are packed as a single bit, or 16-bit word packets.

Using ModBusPoll (app):

- Connect on Modbus TCP/IP:
  - Address 192.168.1.6
- (3) Read holding registers

Using Modpoll (cmd lines):

- cd into directory that has modpoll (modpoll-3.10/win). `./modpoll -h` to give you info
- Cmd line that gave me data, that data changes with encoder changes, but is signed, and switches between -ve and +ve:
  - First thing that worked: `./modpoll -m tcp -t 3:int 192.168.1.6` (this gives some encoder readback, since the address for data is 0x0000 (or 0x0001 bc modpoll is weird))
- Another Cmd line:
  - `./modpoll -m tcp -t 3 -r 0x1120 -c 20 -1 192.168.1.6`
  - 16 bit response, hence `-t 3` is used
  - `-c 20` gives the 20 data after the address inclusive
  - `-0` starts reference from 0 instead of 1 **(when I do this, the output starts from the second register (which is hidden because it jumps between registers 024 etc) this gives better encoder readings because it's linear, but still not accurate)**
  - `./modpoll -m tcp -t 3:int -0 -r 0x0001 -c 4 192.168.1.6` (cmd line that worked, we care about the first register, if you only want that data, get rid of the `-c 4`

Switching encoder to the second terminal:

- I believe register 6 is the one we care about, which is consistent with register 1 from previous terminal (1+5=6)
- For some reason it is almost exactly half the encoder counts of the previous one.

Lines of Codes that work:

- To get the encoder reading of the first encoder:
  - `./modpoll -m tcp -t 3:int -0 -r 0x0001 192.168.1.6`
- To get the encoder reading of the second encoder:
  - `./modpoll -m tcp -t 3:int -0 -r 0x0006 192.168.1.6`