The SharkTapSFP exposes a standard UART (CDC) interface over USB. The interface should be natively supported by Windows and Linux – COMxx or ttyACMx (for example). The UART interface has two modes – a human readable interface and an API mode intended to be accessed by applications. But you can use all the API commands manually too. We recommend https://www.putty.org if you don't already have a favorite terminal emulator application. You will select 'serial' for the type of connection, but the serial parameters (baud rate) don't matter. Select 8 bits/character, no parity, no flow control, and ignore any other parameters.

For the human readable terminal interface:

- Hit <Enter> to redisplay list of modules (<Enter> = '\n' = 0x0a)
- To set time/date, type T, followed by 2 digit hex month, day, year, hour, minute, second. The interface does not care about the separator between each datum. It can be '/', ':', or space.
  Example:
  T 07/03/24 11:39:00<enter>
  The clock calendar clock is very inaccurate and date/time is not maintained across a SharkTap power cycle. The time/date is only intended to give you an approximate idea of when a module was installed or removed.
- Hit 'A <enter>' to start the serial API. When you enter the serial API, the SharkTap will no longer poll for insertion/removal of modules. See below for serial API commands
- Hit 'I <enter>' to stop the serial API.

All of the internal features of the SharkTap are accessed through I2C commands. Each [standard] module contains an identifying ROM at i2c address 0x50. Modules may also have other custom features accessed through its i2c interface, such as accessing the module's PHY. You can access any such i2c address, other than 0x20-0x23 or 0x70 (which are used by the SharkTap.) Each module also has general purpose I/O (gpio) signals defined by INF-8074. The SharkTap accesses these signals through i2c expander chips, as detailed below.

After entering 'A<enter>' the terminal interface will display some human readable information, which an application can discard. Here's the information displayed:

```
Serial API started. Enter I<enter> to exit API
Write I2C: W 00 01 02<enter> where 00 is hex address, 01 etc are hex data
Read I2C:  R 00 01<enter> where 00 is hex address, 01 is hex number of bytes
Up to 16 bytes per read or write
```

An application exits the API mode (and returns to autoscanning of port status) by entering:

```
I<enter>
```

If any i2c command is formatted incorrectly or the i2c command itself fails, the API will display "!!", or more specifically, printf("\n\r!!\n\r");
If a write command completes successfully, the API will printf("\n\rOK\n\r");
If a read command completes successfully, the API will printf("\n\rOK: ") followed by the requested bytes in lower case hex ("%02x " in printf-speak), followed by "\n\r"

Here’s the i2c addresses on the SharkTap motherboard

i2c address: 0x70, R/W
Description: A PCA9546 crossbar switch, which routes the i2c buss to the four modules
Values:
0x01 – NETWORK A
0x02 – NETWORK B
0x04 – Carbon Copy B
0x08 – Carbon Copy A
Example:
`W 70 02<enter>` - routes I2C buss to NETWORK B module
`OK`

I2C address: 0x20-0x23, R/W
Description: The gpio for each module are accessed through a PCA9554 chip. The addresses map as follows:

0x20,	NETWORK A
0x21,	NETWORK B
0x22,	Carbon Copy A
0x23,	Carbon Copy B

Values: (ignore bits not detailed below)
Input port: (write address, 0x00, then read data byte)
```
+-+-+-+-+-+-+-+-+
|7|6|5|4|3|2|1|0|
+-+-+-+-+-+-+-+-+
         | | +------- Tx_FAULT (transmit fault), input, (pin 2)
         | +--------- Rx_LOS (nLink), input, 0 = link (pin 8)
         +----------- MOD_ABS (nPresent), input, 0 = present (pin 6)
```
Output port: (write address, 0x01, data byte)
```
+-+-+-+-+-+-+-+-+
|7|6|5|4|3|2|1|0|
+-+-+-+-+-+-+-+-+
     | |       +----- Tx_DIS (transmit disable), out, 1 = disable (pin 3)
     | +------------- RATE_SEL0, output (pin 7)
     +--------------- RATE_SEL1, output (pin 9)
```

Examples:
`W 23 00` – select input port, Carbon Copies B
`OK`
`R 23 01`
`OK: ce`
Insert a module…
`R 23 01`
`OK: c4` – the MOD_ABS input has gone to zero