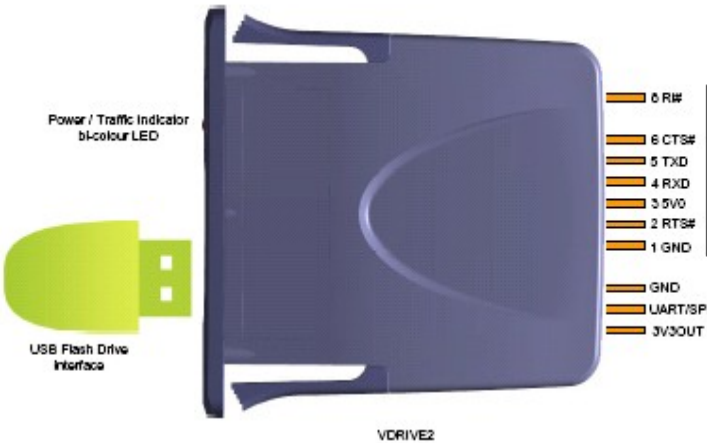
VINCULUM – Adapter für USB-Sticks
Getting Started

Leider gibt es von der Firma selbst keine Schnelleinführung. Hier das Ergebnis meiner Experimente. Vielleicht hilft es dem ein oder anderen, sein Vdrive schneller in Betrieb zu nehmen.

http://www.vinculum.com

Wichtige Dokumente:
http://www.vinculum.com/documents/datasheets/DS_VDRIVE2.pdf an
http://www.vinculum.com/documents/fwspecs/Vinculum%20Firmware%20User%20Manual%20V2.3%20Rev%202.pdf



Inbetriebnahme

Beschaltung für Steuerung über die serielle Schnittstelle:

+5V, GND
CTS auf GND
Jumper verbindet UART/SPI mit 3V3OUT (untere beiden Pins)
Kommunikation über TxD und RxD (9600 Baud, 8N1)

Ein erster Test erfolgt am einfachsten mit PC und zwischengeschaltetem MAX232, mit Terminalprogramm.
Als Terminalprogramm eignet sich gut das kostenlose wxterminal:
https://iftools.com/download.en.html

Nach dem Einschalten meldet sich das Modul mit

```
Ver 03.62VDAPF On-Line:
```

Wenn ein USB-Stick vorgefunden wird, blinkt die LED kurz auf und leuchtet dann gün, es erscheint die Meldung

```
Device detected P2
No Upgrade
D:\>
```

Nach dem Einschalten ist das Modul im Modus ECS (Enhanced Command Set) und IPH (Binary Command Mode).

Firmware-Update

Es ist empfehlenswert, die neueste Firmware aus dem Internet herunterzuladen, sonst sind eventuell verschiedene Kommandos nicht verfügbar.
Die Firmware-Datei wird umbenannt in „FTRFB.FTD“ und in das Root-Verzeichnis eines USB-Sticks kopiert. Bei jedem Start sucht das Vinculum-Modul nach dieser Datei und macht gegebenenfalls ein automatisches Update.

ECS und SCS

ECS (Enhanced Command Set, Default nach dem Einschalten) ist besser gegeignet beim Testen am Terminal, da alle Zeichen des Kommandos und der Antwort druckbare Zeichen sind.

SCS (Short Command Set) eignet sich zum Betrieb mit Mikrocontroller, da die Kommandos und Antworten möglichst kurz sind.

Den Unterschied sieht man, wenn man vom Terminal aus ein einfaches <CR> (Carriage Return) schickt (Darstellung hex!).

```
ECS-Mode:      44 3A 5C 3E 0D      D:\>       (5 Byte)
SCS-Mode:      3E 0D               >          (2 Byte)
```

Die Antwort entspricht einem „Successful Command Prompt“, welches die normale Antwort auf einen Befehl darstellt.
Jede Antwort wird mit einem 0D = 13 = <CR> – Zeichen abgeschlossen.
Dies gilt auch für die Kommandos.

IPA und IPH

Der IPA-Mode erwartet Zahleneingaben im Klartext, nicht in Binärform, er ist also für Experimente mit dem Terminal einfacher zu benutzen.

Alle Kommandos müssen mit <CR> ($0D, Carriage Return) abgeschlossen werden.

Beim IPH-Mode werden Zahlenwerte binär eingegeben.

Einfache Kommandos im ECS-Mode (Extended Command Set)

(Gross oder Kleinschreibung erlaubt)
Genaue Beschreibung siehe Vinculum-Dokumentation

<CR>	Disk present?
FWV	Display Firmware-Version
IDDE	Display information about disk
IDD	Display information about disk

DIR	List files in directory (keine langen Dateinamen!)
DIR *file*	List file and size (Dateigrösse wird binär hintendrangehängt)
RD *file*	Read file. Die Datei wird über die Schnittstelle ausgegeben.

CD *dir* Change Directory
Wechsel in den Folder *dir*
Der Wechsel macht sich im Prompt nicht bemerkbar!

Experimente im IPA-Mode (ASCII-Mode)

Der IPA-Mode erwartet Zahleneingaben im Klartext, nicht in Binärform, er ist also für Experimente mit dem Terminal einfacher zu benutzen.

Achtung: der Defaultmodus ist IPH (binär), es muss also erst mit dem Kommando „IPA“ umgeschaltet werden!

Lesen einer Datei

Für erste Experimente wurde mit Notepad eine Datei *TEST.TXT* im Root-Directory des Stick erzeugt.

„RD TEST.TXT“ liest die Datei und schickt den Inhalt an das Terminalprogramm.
Achtung: am Ende wird das PROMPT 44 3A 5C 3E 0D (D:\>) zusätzlich geschickt!
Will man nur den Inhalt der Datei, so muss diese Ausgabe weggefiltert werden!

Lesen von Bytes aus einer Datei

Achtung: für dieses Experiment in den ASCII-Modus mit IPA!

IPA	ASCII-Modus
OPR TEST.TXT	Datei zum Lesen öffnen
RDF 3	3 Bytes lesen
RDF 5	nächste 5 Bytes lesen
CLF TEST.TXT	Datei schliessen

*S*chreiben von Bytes in eine Datei

Achtung: für dieses Experiment in den ASCII-Modus mit IPA!

IPA	ASCII-Modus
OPW TEST.TXT	Datei zum LesenSchreiben öffnen
WRF 5	5 Bytes schreiben
Hello	5 Bytes zum Schreiben eingeben
CLF TEST.TXT	Datei schliessen
RD TEST.TXT	Ganze Datei lesen

Die Bytes „Hello“ wurden am Ende der Datei hinzugefügt.

Experimente im SCS-Mode (Short Command Set)

Hier werden die Werte binär geschickt, beim Betrieb am Terminal muss die Möglichkeit bestehen, auch nichtdruckbare Zeichen zu schicken.

Beim wxterminal kann man dies erreichen, indem man die Optionen auf „cooked hex“ umstellt:

Jetzt werden alle Zeichen normal gesendet, ausser Zahlen mit vorgesetztem „$“ .
Diese werden als Hex-Zahlen interpretiert.
Die Ausgabe erfolgt erst mit der <Return>-Taste.

Umschalten in den SCS-Mode:
SCS$0D

Als Antwort erhält man „> “ (3E 0D, 2 Bytes!)
Dies ist im SCS-Mode die allgemeine Antwort auf ein gültiges Kommando.

Das $0D muss mitgeschickt werden, da im ECS-Mode jedes Kommando mit <CR> = 0D abgeschlossen werden muss.

Um das <CR> automatisch zu schicken, kann das Terminal auch auf EOS (end of string) = <CR> eingestellt werden:

<u>Einfache Kommandos:</u>

$0D Disk present? (Antwort 3E 0D = ja, 4E 44 0D = nein)
$01 $0D DIR, List files in directory (keine langen Dateinamen!)

Suche nach einer Datei (DIR *file)*:
$01$20test.txt$0D
(das $20 ist ein Leerzeichen, dies kann auch mit der Tastatur eingegeben werden)

Datei lesen (RD *file)*:
$04$20test.txt$0D

Bytes in Datei schreiben:

$09$20test.txt$0D	OPW = Datei zum Schreiben öffnen
$08$20$00$00$00$03$0D	WRF 3: 3 Bytes schreiben (Zahl der Bytes als DWORD, MSB zuerst)
abc	3 Bytes werden eingegeben
$0A$20test.txt	CLF = Datei schliessen

Mit $04test.txt (RD) kann die Datei gelesen und überprüft werden.
Die 3 eingegebenen Bytes sind am Dateiende angehängt worden.

Bytes aus Datei lesen:

$0E$20test.txt$0D	OPR = Datei zum Schreiben öffnen
$0B$20$00$00$00$03$0D	RDF 3: 3 Bytes lesen (Zahl der Bytes als DWORD, MSB zuerst)
$0A$20test.txt$0D	CLF = Datei schliessen

Nach diesen Experimenten sollte man jetzt gerüstet sein, das Vinculum seinem eigentlichen Zweck zuzuführen, der Anbindung an einen Mikrocontroller.
Mehr dazu im nächsten Teil.