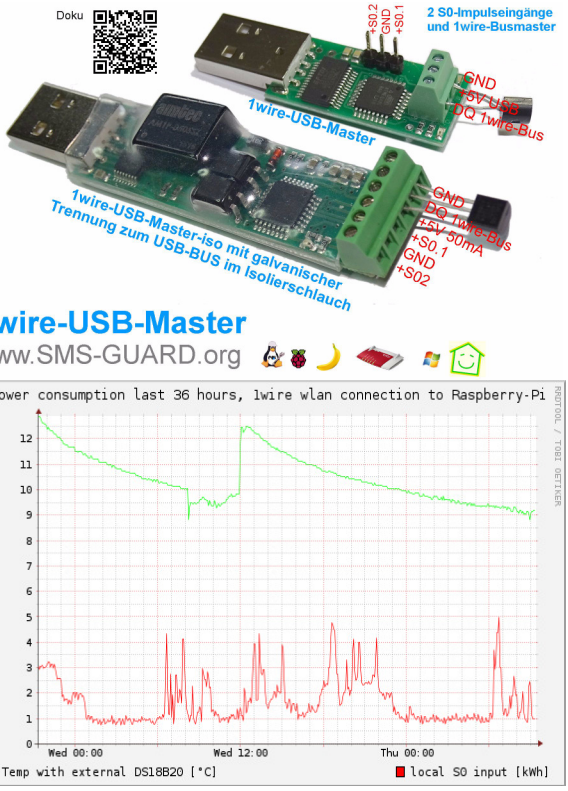
1wire-USB-Master und 2xS0-Eingänge



Der 1wire-USB-Master verfügt über ein 1 wire-Master-Interface mit Schraubklemmen und den Signalen „+5V“, „GND“ und „DQ“ als Datenleitung für den 1wire-Bus. Der 1-wire-USB-Master erfasst bis zu 64 Stück 1-wire-Sensoren, wie DS18B20, DS18S20, sowie intelligente 1wire-Sensorbus-Module, wie SB-S0-Module (S0-Zähler), SB-SS-S0-Module (Stromsensor als Klappwandler), SB-D0-Module (Mehrrichtungs-Stromzähler) etc. und sendet die Daten der Busteilnehmer über seine USB-serielle Schnittstelle (Standard-Treiber FT232) an einen PC, Raspberry-Pi, Fritzbox oder WIN. Außerdem verfügt der 1-wire-USB-Master über zwei vom 1 wire Bus unabhängige S0-Eingänge zur direkten und kostengünstigen Impuls-Erfassung von Stromzähler, Wasserzähler oder Gaszähler. Die Ausführung 1wire-USB-Master-iso ist vom USB-BUS galvanisch getrennt. Die Daten der 1wire-Teilnehmer und S0-Eingänge werden im Filesystem des PC als Text-Dateien abgespeichert und sind für PHP, Perl, Python, Ruby, usw. ohne speziellen Treiberaufwand einfach einlesbar, ohne weitere Tools wie owfs. Die Speicherung und Visualisierung der Daten erfolgt über die einfache Bash „1wire-USB-rrd-sh“ und dem kostenlosen „rrdtool“. In der Bash können eigene Signale hinzugefügt und die Diagramme beliebig angepasst werden.

Inbetriebnahme unter LINUX Raspbian

1) für Linux-PC-Systeme ist die Datei http://www.sms-guard.org/downloads/1wire-USB.tar im Userverzeichnis zu entpacken, die http://www.sms-guard.org/downloads/1wire-USB-pi.tar für Raspberry-Pi und Rb-Pi 2 oder die 1wire-USB-ba-pi.tar für den Banana-Pi oder die 1wire-USB-fritz.zip für die Fritzbox. Für die grafische Darstellung der Daten ist mit „sudo apt-get install rrdtool“ dieses zu installieren. Bei Raspberry ist in der /boot/config.txt enable_uart=1 anzufügen und ein halt auszuführen, den 1wire-USB-Master an den USB-Port stecken und den Raspberry einschalten.
2) Mit „lsusb“ die aktuellen USB-Teilnehmer ansehen.
3) den 1-wire-USB-Master in eine USB-Buchse stecken und die rote LED geht kurz an, mit „lsusb“ sollte der FT232-Treiber zu sehen sein und mit „dmesg | grep usb“ das dazugehörige Device, z.B. „/dev/ttyUSB0“ oder „/dev/ttyUSB5“, usw.
4) der User benötigt Berechtigungen zum Lesen- und Schreiben auf die serielle Schnittstelle (googeln).
5) die 1wire-Daten werden später laufend in einen Datenpfad geschrieben. Bei SD-Karten muss der Datenpfad als RAM-Disk eingerichtet werden, die „/etc/fstab“ (googeln) wäre zu erweitern um die Zeile:
„tmpfs /tmp tmpfs defaults,size=10M 0 0“
und dies durch Schreiben und Lesen eines Files in /tmp überprüfen
6) mit „/home/knoppix/Desktop/1wire-USB /dev/ttyUSB0 /tmp/ -l“ (beim 1wire-USB-Master-iso das -l erweitern auf -lb9600 und ab 09.04.20 auf -lb300) den Adapter manuell ansteuern und ttyUSB0 prüfen, siehe 3), dann blinkt die rote LED laufend. Mit „ctrl C“ beenden und in die 1wire-USB-rrd-sh eintragen:

```
########################## nur hier sind die eigenen Daten einzutragen
PROG_PATH="/home/knoppix/Desktop“    # Pfad in dem die 1wire-USB läuft
DATA_PATH="/tmp“                     # Pad zu den Daten in einer RAM-Disk tmpfs (/etc/fstab)
SERIAL="/dev/ttyUSB0“                # serielles Interface des 1 wire-USB-Master-Adapters
#############################################################################
```

7) die „1wire-USB-rrd-sh“ ist alle 5Minuten mit dem cron zu starten (/etc/crontab), prüfen mit ps -A . Die shell startet die „1wire-USB“ und legt die Datenbank „1wire-USB.rrd“ an und pflegt dann alle 5min die Daten nach rrd ein.

8) die 1wire-Daten werden nun alle paar Sekunden in den Datenpfad (/tmp o.ä. muss vorhanden sein) geschrieben, z.B. 1080974B020800BA.txt, diese 1wire-ID ist in der „1wire-USB-config.txt“ einzutragen:
1080974B020800BA;sb0;
10A673530208005F;sb1;SB-S0-Modul
S01;L1;1000
S02;L2;800
und folgend wird der Sensor mit der 1wire-ID „1080974B020800BA“ unter dem Dateinamen „sb0.txt“ geführt, mit dem Inhalt der empfangenen ***8 Datenbytes*** + CRC, z.B.: „AA;00;4B;46;FF;FF;0C;10;87;;“ (***siehe dazu Datenblatt des verwendeten 1-wire-Sensors***), außerdem wird bei bekannten 1wire-Teilnehmern der Sensorwert in die Datei „sb0v.txt“ abgelegt („v“ für „value“), hier „21.0“. Bei den S0-Eingängen folgt nach dem Namen die Impulsauflösung des Stromzählers, hier „1000“ Impulse/kWh, der Wert in der L1v.txt entspricht dann dem S0-Zählstand in [Wh], und kann mit „cat /tmp/L?v.txt“ gelistet werden. ***Bei unbekannten 1wire-Teilnehmern, wie z.B. ADCs oder Counter, werden immer die 8 Datenbytes + CRC in einem Textfile ID.txt abgelegt und können so weiter verarbeitet werden.***
9) die Dateien können einfach unter PHP eingelesen werden. Wollte man beispielsweise an einem SB-S0-Modul eine LED oder Ausgang einschalten, wäre in der Datei sb1o.txt eine 1 einzutragen, bei 0 geht die LED wieder aus, siehe für diese Funktion auch die Anleitungen der entsprechenden SB-Module. Wird ein „r“ eingetragen, so resetet sich der 1wire-USB-Master und alle S0-Zählerstände werden zurückgesetzt (siehe auch Fehlersuche g). Wird bei einem Sensorbus-Modul in die sb1o.txt eine 0;00;00 geschrieben, so wird der EEProm-Bereich von Adresse 0x01-0x08 in die sb1e.txt geschrieben und nach Ausführung des Befehls wieder eine 0 in die sb1o.txt geschrieben. Bei 0;00;09 wird die Adresse 0x09-0x010 gelesen und mit 0;01;55 wird im EEProm die Adresse 0x01 mit dem Wert 0x55 beschrieben, so lassen sich die Parameter in den SB-Modulen komfortabel und schnell einstellen.

Die 1wire-USB kennt als dritten Parameter folgendes:
-m (für multiple), die „1wire-USB“ kann mehrfach gestartet werden für verschiedene Schnittstellen, um z.B. zwei 1wire-USB-Master am Raspberry-Pi zu betreiben für 4 lokale S0-Eingänge (siehe auch Produktpflege).
-l LED-Ausgabe aktiv, ansonsten bleibt die LED ausgeschaltet und blinkt nur beim power-ON 2x
-d Debug-Ausgabe
-f für Betrieb an Fritz!box
mehrere Parameter müssen hintereinander übergeben werden als „-mld“

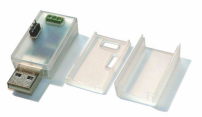
Technische Daten:

Messbereich S0-Eingänge:	0 - 60kW bei 1000 Impulse/kWh, min. S0-Impulsbreite 30ms, 32Bit Zähler, bei Gas- und Wasseruhren mit Reed-Kontakt als S0-Ausgang ist zur Entprellung ein Kondensator mit 100nF an S0+ und GND zu legen (entfällt beim -iso)) und bei EMV-Impulsen einen 330ohm pullup-Widerstand zwischen S0+ und +5V.
max. Werte an den beiden S0-Eingängen:	für potentialfreie S0-Ausgänge ist die Kontaktbelastung < 0.25mA und max. +5VDC
1wire-Bus:	für bis zu 64 Sensoren (auch im parasitären Modus) und SB-Module, Busmaster-IC: ATmega
automatisierte Werterfassung für:	DS18B20, DS18S20, SB-S0, SB-SS-S0, SB-UPM, SB-M (http://www.sms-guard.org/dupdates.htm#SB), nicht unterstützt werden DS2438, DS2423, usw.
Schraubklemmen für 1wire-Bus:	für Leiterquerschnitte von 0.14 - 0.5mm^2 und einen Schraubendreher bis 1.8mm Klingenbreite
Signale für 1wire-Bus:	Datenleitung „DQ“, „GND“ und „+5V“, bis zu 100m bei Verkabelung <10nF/100m
Versorgungsspannung:	+5V ±3% aus USB-Typ-A-Stecker
Stromaufnahme:	typ. 15mA, 1wire-USB-Master-iso typ. 100mA
Betriebstemperatur:	0°C bis +50°C
max. Luftfeuchtigkeit:	85% ohne Betauung
Gehäuseschutz:	trockener Innenbereich, kein Gehäuse
Abmessungen Modul:	16 x 12 x 49mm (BxHxT), -iso 16 x 12 x 76mm, eingesteckt in USB-A-Buchse 41mm Überstand,-iso 68mm
Gewicht:	ca. 5g, 1wire-USB-Master-iso ca. 10g

Lieferumfang:
- „1wire-USB-Master“ USB-Adapter (ohne externe 1wire-Sensoren)
- „1wire-USB“ ausführbares Programm als Download unter www.SMS-GUARD.org für 32Bit-PC-Linux, Raspberry-Pi, Banana-Pi, Fritz!box freetz

Zubehör:

- 2 Gehäuseschalen zum Verkleben, inkl. abziehbaren S0-Stecker (entfällt bei -iso da im Isolierschlauch)



Die Software (32 Bit) wurde getestet unter Ubuntu10 (ab Ubuntu 13.10 bei 64Bit Systeme ia32-libs installieren), Knoppix7, Raspberry Debian wheezy-raspbian vom 7.1.2014 und 16.02.2015, Bananian vom 11.01.2015 und Fritz!box 7390 freetz. Bitte haben Sie Verständnis, wir können für diese Plattformen und Systeme weder Haftung noch Support übernehmen und Linux-Systeme erfordern mehr Kenntnisse und Erfahrungen als WINDOWS. Für die Inbetriebnahme des 1wire-USB-Master wird Erfahrung im Umgang mit den benötigten Befehlen des Betriebssystems vorausgesetzt.

Eine Beschreibung für den Betrieb des 1wire-USB-Adapters unter WINDOWS finden Sie auf der Homepage von Frank Carius - Danke: http://www.msxfaq.de/verschiedenes/bastelbude/eac1wires0stick.htm
und das rrdtool dazu : http://www.msxfaq.de/tools/rrdtool.htm

Bitte beachten:
a) mit Umstecken des 1wire-USB-Adapters an der gleichen USB-Schnittstelle oder an einer anderen USB-Schnittstelle kann sich die Device-Nummer ändern, z.B. von „/dev/ttyUSB0“ nach /dev/ttyUSB1“, usw . In diesem Falle ist die richtige Schnittstelle zu ermitteln nach Punkt 2) und 3) der Inbetriebnahme und der richtige Parameter in die 1wire-USB-rrd-sh einzutragen.
b) bei Änderungen in der 1wire-USB-rrd-sh muss zu deren Wirksamkeit für die Parameterübernahme die 1wire-USB gekillt werden, durch den cron wird diese immer wieder neu gestartet.
c) werden in der 1wire-USB-config.txt Namen geändert und/oder zugefügt, muss auch die Übernahme der Sensorwerte aus den Dateien in der 1wire-USB-rrd-sh angepasst werden. Bei SB-Modulen darf am Zeilenende kein ; stehen in der *config.txt.
d) zählt der 1wire-USB-Master manchmal zu viele Impulse, z.B. beim Schalten von Licht, dann sollte ein pullup-Widerstand mit 330ohm zwischen S0+ und +5V gesetzt werden und ein Kondensator mit 100nF zwischen S0+ und GND, die Kondensatoren sind beim -iso bereits bestückt.

Fehlersuche:
der 1wire-USB-Master kann recht einfach mit einem seriellen Terminal minicom/HyperTerminal geprüft werden:
a) serielle Schnittstelle einstellen auf 115200 (ab 25.8.15 u1-01c 38400, -iso 9600Baud, -iso 300Baud ab 09.04.20) 8-N-1 kein Handshake RTS/CTS, kein On/Off Protokoll und darauf achten, dass der Adapter auch wirklich an „/dev/ttyUSB2“ o.ä. hängt und kein anderer Prozess diese Schnittstelle nutzt.
LED-Befehl eingeben, damit die rote LED jeden gültigen Befehl anzeigt: $L+<CR>
b) Startbefehl eingeben, damit wird auch die Wandlung im Sensor ausgelöst: $?<CR>
c) mit angeschlossenen 1wire-Sensoren werden die gefundenen IDs gelistet: $0;o;1080974B020800BA;
das „o“ steht für „ok“ und die Checksumme der ID wurde geprüft und ist ok
d) danach gibt der 1wire-USB-Master die beiden S0-Zählerstände zurück: $S0;0;0;
e) die Werte der 1-wire Sensoren können nach 1s abgefragt werden mit: $0<CR> ... $63<CR>
f) danach gibt der 1wire-USB-Master die Daten des Sensors zurück: $0;o;31;00;4B;46;FF;FF;07;10;8D;64;
das „o“ steht für „ok“ und die Checksumme (9.Byte) der ***8 Datenbytes*** wurde geprüft und ist ok („n“ wäre „nicht ok“)
die Beschreibung der ***8 Datenbytes in Hex ist dem Sensordatenblatt zu entnehmen***
das 10.Byte ist eine Checksumme für die serielle Übertragung (Byte1-9 aufaddiert).
g) die S0-Zählerstände können auf 0 gesetzt werden mit, danach erfolgt ein restart: $rez<CR>
h) wird das Terminalprogramm beendet und die Schnittstelle frei gegeben, so kann mit der „1wire-USB-rrd-sh“ (cron) das automatische Einlesen des Adapters vollzogen werden und die Sensordaten werden in Textfiles geschrieben.

Einbinden in FHEM oder IP-Symcon:
die 1wire-USB läuft auf der verwendeten Hardware und schreibt die Daten vom 1wire-USB-Master als Textfile in das lokale Filesystem. FHEM beruht auf Perl und somit können Daten aus Textfiles als aktuelle Sensorwerte übernommen werden, siehe auch http://www.sms-guard.org/downloads/1wire-USB-Master-fhem.pdf . IP-Symcon basiert auf PHP, auch hier können Textfiles sehr einfach eingelesen werden. Weitere spezielle Treiber dazu sind nicht notwendig. Neben dem 1wire-USB-Master kann ein anderer 1wire-Busmaster wie beispielsweise DS9490R unter owfs parallel im System an einem 2. 1wire-Bus betrieben werden. Keinesfalls ist unser 1wire-USB-Master mit einem Treiber einzulesen, der ein Busmaster-IC wie DS2482 erwartet, wie beispielsweise owfs und dann diese Sensordaten in Textfiles ablegt, was aber ja bereits die 1wire-USB mit dem 1wire-USB-Master leistet.

Zufügen weiterer Sensoren:
64 1w-Sensoren/SB-Module sind am Busmaster möglich und das Zufügen eines neuen Clients ist am Adapter denkbar einfach mit der 1wire-config.txt. Komplizierter ist es da mit dem vielseitigen rrdtool. Die rrd Datenbank muss neu angelegt werden (*-sh) mit den alten und dem neuen Wert (evt. für die Zukunft ein paar Reservewerte mit aufnehmen) und in der *-sh müssen alle Werte mit eingepflegt und die gewünschten Werte als png ausgegeben werden. Eventuell schafft man es mit Geduld und Probieren die *-sh durch kopieren und anpassen der vorhandenen Zeilen zu erweitern, andernfalls wird man sich näher mit der Beschreibung des rrdtool befassen müssen. Gerne würden wir hierzu Ihre Codebeispiele und Diagramme mit veröffentlichen.

Produktpflege „Treibersoftware“ 1wire-USB:
v1.00 1.Version
v1.01 17.11.14 der Kommandozeilenparameter -m wurde erweitert um eine folgende Zahl -m2. In diesem Falle werden die Parameter in der 1wire-config-2.txt erwartet und L1 und L2 kann umbenannt werden in L3 und L4. Es können bis zu 64 Adapter an einem PC betrieben werden.
v1.02 18.05.15 schaltet automatisch zwischen 38400 und 115200 Baud
v1.03 29.08.15 -e exit nach einem Durchlauf
v1.04 15.12.16 -b9600 für 1wire-USB-Master-iso mit 9600 Baud, sowie -b38400 und -b115200
v1.05 27.02.17 Sensorauswertung für unbekannte family-Codes überarbeitet und Anbindung von SB-Modulen
v1.06 09.04.20 -b300 für 1wire-USB-Master-iso u1-02f

Produktpflege Firmware auf dem 1wire-USB-Master:
u1-01a 1. Version
u1-01b 16.05.14 die lokalen S0-Impulsstände/1000 werden im EEPROM gespeichert, somit bleiben diese gekürzten S0-Zählstände auch nach einem Stromausfall erhalten. Mit „$rez“ können diese Zählerstände auf 0 gesetzt werden. Der 1wire-Daten-PIN am Prozessor wird kontinuierlich getestet, ist dieser defekt, leuchtet die rote LED dauerhaft.
u1-01c 25.08.15 Baudrate geändert auf 38400. Wenn Brücke gesetzt auf Lötseite, dann Baudrate wieder 115200.
u1-01d 16.11.16 die S0-Eingänge haben nun eine Glitchunterdrückung.
u1-02d 15.12.16 1wire-USB-Master-iso mit 9600 Baud
u1-02f 09.04.20 1wire-USB-Master-iso umgestellt auf 300 Baud

Anmerkungen:
durch die eigene CPU auf dem 1wire-USB-Master wird die CPU des PCs kaum belastet und beträgt bei einem Atom-D525-PC gerade mal 1-2%. Ebenfalls entlastet der 1wire-USB-Master den PC von Echtzeitaufgaben, was bei bisherigen 1wire-Anbindungen nicht immer gegeben ist. Soll der PC andere Echtzeitaufgaben ausschließlich erledigen, kann die „1wire-USB“ gestoppt und später wieder neu gestartet werden. Wird die USB-Buchse vom PC (BIOS-Einstellung USB) auch im ausgeschalteten Zustand mit Spannung versorgt, läuft der 1wire-USB-Master weiter und erfasst zuverlässig alle lokalen S0-Impulse. Der Zählerstand ist dann bei eingeschaltetem PC aktuell, lediglich der Verbrauchsverlauf wurde nicht erfasst.

Eine große Arbeitserleichterung gegenüber herkömmlichen 1wire-Anbindungen ist die Möglichkeit der Verwendung symbolischer Dateinamen anstelle der IDs. Gerade bei größeren Systemen ist es bei bisherigen 1-wire-Anbindungen mühsam, die ID-Dateinamen den installierten Sensoren und deren Funktion und Ort auf einen Blick im Kopf zuzuordnen.

Ein Vorteil von „rrdtool“ ist die gleichbleibende Größe der Datenbank ab Initialisierung über die gesamte Lebenszeit. Im beiliegenden Beispiel werden die Daten alle 5min eingelesen. Nach einem Monat extrahiert „rrdtool“ die Daten auf einen Mittelwert von 15min, danach auf 1Stunde und nach eine Jahr auf 3 Stunden. Es wird der Energieverbrauch über einen Zeitraum von 10 Jahren abrufbar gehalten. Die Zeiträume können Sie nach Ihren Bedürfnissen auch anpassen.