Python-Debugger

- TELIT Debug-Adapter SSC --> RS-232
- automatic Synchronisation
- stand alone mode

Beschreibung
07.04.2007

Roundsolutions GmbH & Co. KG
www.roundsolutions.com
info@roundsolutions.com

Allgemeines

Der *Python-Debugger* ermöglicht die Ausgabe von Debug-Informationen aus Python-Scripts, die auf TELIT-GSM-Modulen ausgeführt werden.

Bitte lesen Sie diese Beschreibung vor der ersten Inbetriebnahme komplett und sorgfältig. Sie beschreibt den bestimmungsgemäßen Gebrauch und enthält wichtige Hinweise zur Installation/Inbetriebnahme des *Python-Debuggers*. Für die Folgen nicht bestimmungsgemäßen Gebrauchs übernimmt der Hersteller keine Haftung. Sämtliche Garantieansprüche entfallen.

Anwendung des *Python-Debuggers*

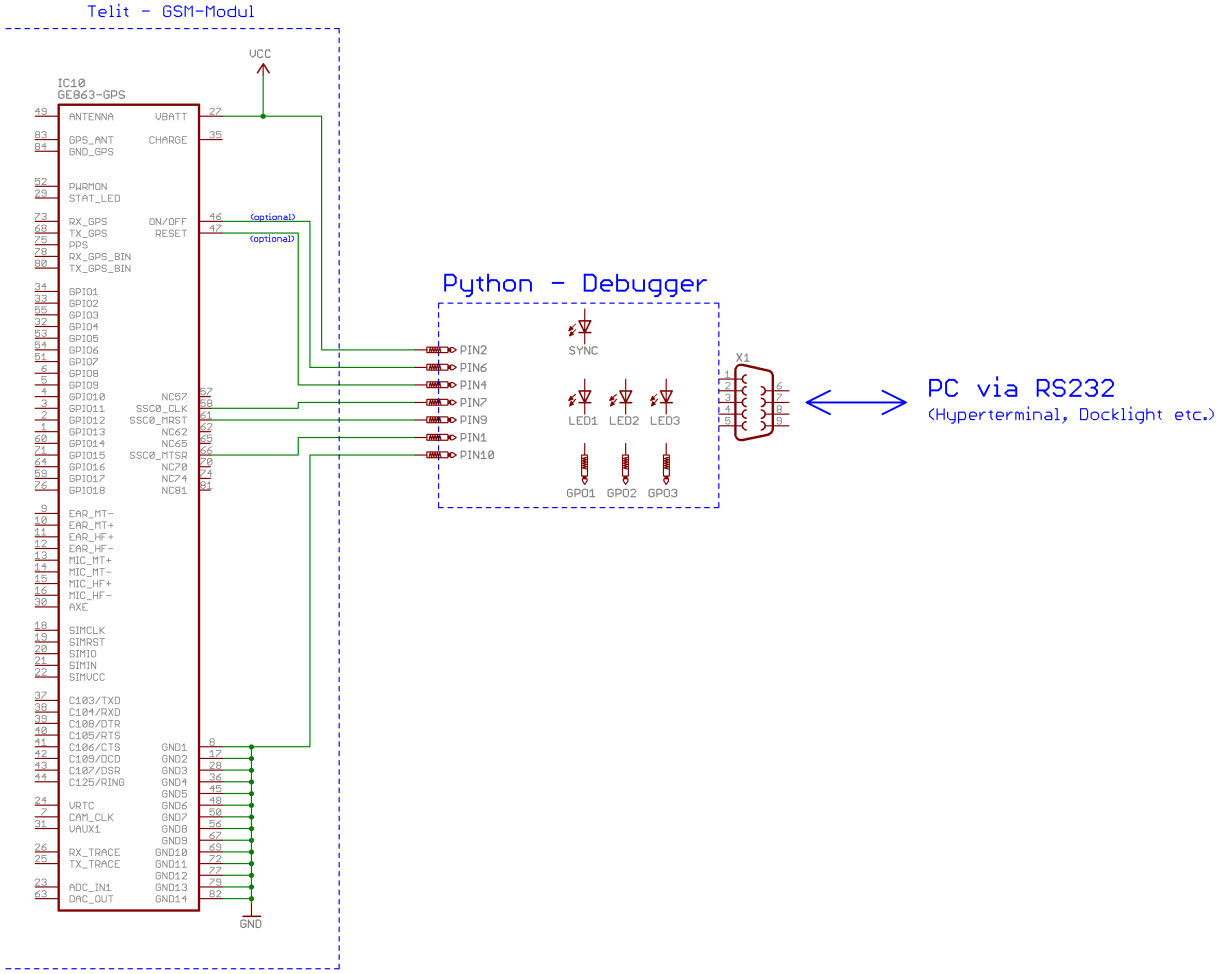
Der *Python-Debugger* kann bei folgenden TELIT-Modulen eingesetzt werden

- GM862-GPS
- GE863-GPS

und ermöglicht die Ausgabe von Debug-Informationen über eine serielle Schnittstelle (RS-232).

Inbetriebnahme

Anschluss des *Python-Debuggers*:



Das Schaltbild zeigt die Anschaltung des *Python-Debuggers* an ein TELIT-Modul vom Typ „GE863-GPS“. Bei anderen Modul-Typen sind die entsprechenden SSC0-Pins zu verwenden:

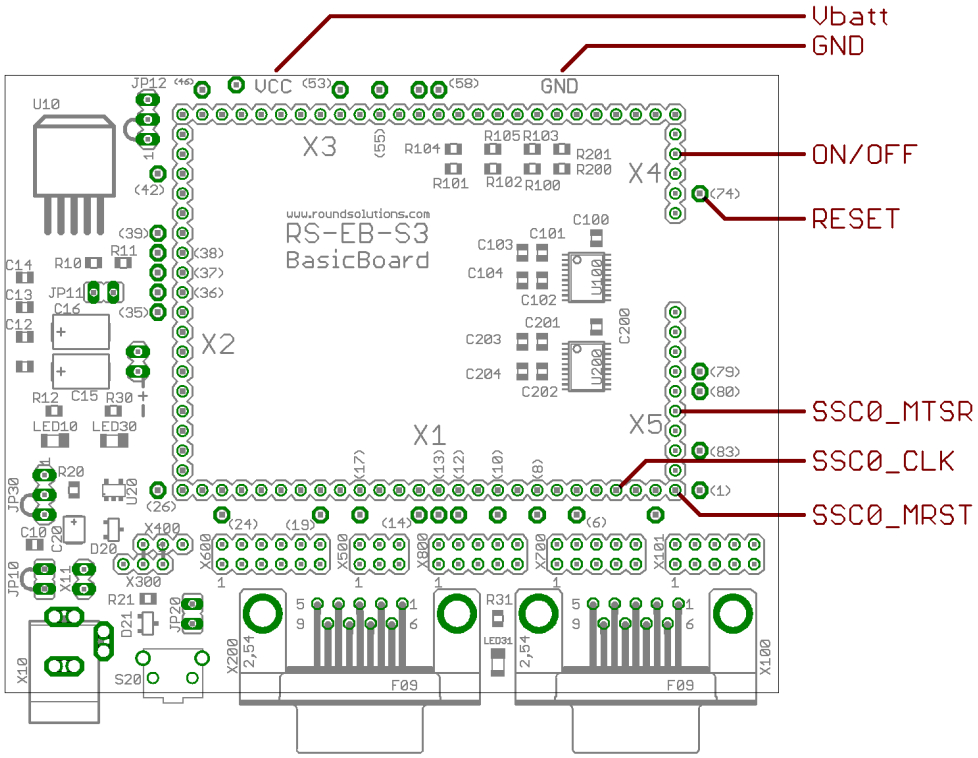
Python-Debugger ---> RS-EB-S3 / GM862-GPS - Adapterboard:

	Vbatt	SSC0_CLK	SSC0_MRST	SSC0_MTSR	ON/OFF	RESET	GND
		Debug-Schnittstelle			optional		
am Modul	Pin 1	Pin 38	Pin 44	Pin 48	Pin 17	Pin 23	z.B. Pin 2
am RS-EB-S3	Vcc-Pad	Pin 4	Pin 1	Pin 81	Pin 72	Pin 74	GND-Pad
	↕	↕	↕	↕	↕	↕	↕
Python-Debugger	Pin 2	Pin 7	Pin 9	Pin 1	Pin 6	Pin 4	Pin 10

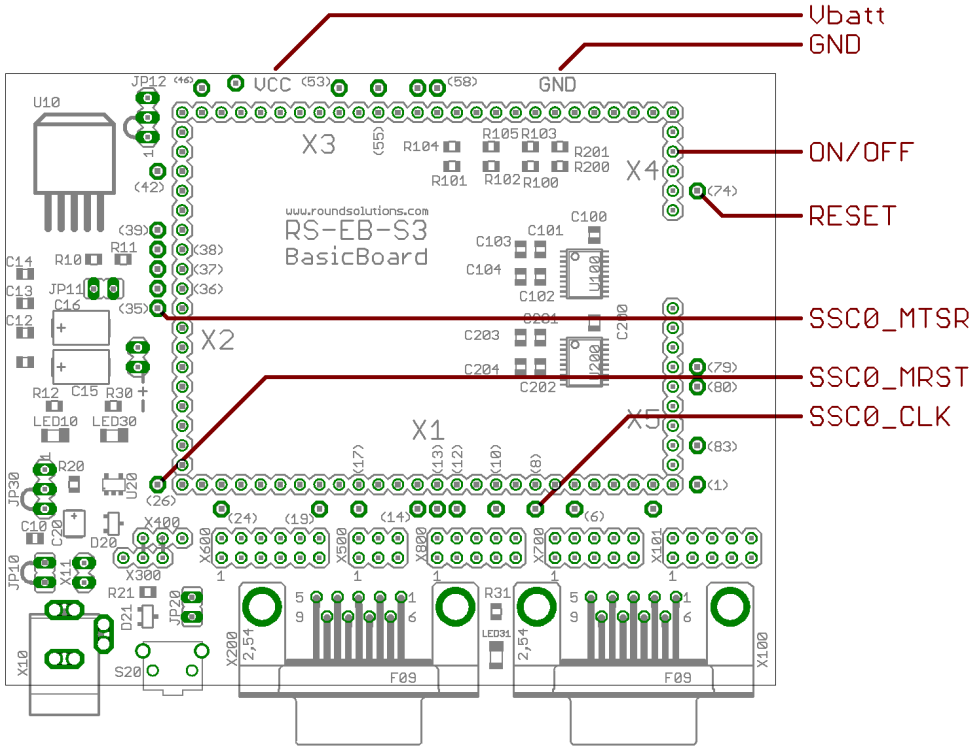
Python-Debugger ---> RS-EB-S3 / GE863-GPS - Adapterboard:

	Vbatt	SSC0_CLK	SSC0_MRST	SSC0_MTSR	ON/OFF	RESET	GND
		Debug-Schnittstelle			optional		
am Modul	Pin 27	Pin 58	Pin 61	Pin 66	Pin 46	Pin 47	z.B. Pin 8
am RS-EB-S3	Vcc-Pad	Pin 8	Pin 26	Pin 35	Pin 72	Pin 74	GND-Pad
	↕	↕	↕	↕	↕	↕	↕
Python-Debugger	Pin 2	Pin 7	Pin 9	Pin 1	Pin 6	Pin 4	Pin 10

Python-Debugger ---> RS-EB-S3 / GM862-GPS - Adapterboard:



Python-Debugger ---> RS-EB-S3 / GE863-GPS - Adapterboard:



Beispiel - Python-Script:

Die Texte der PRINT-Anweisungen werden über den *Python-Debugger* direkt an das Terminalprogramm des PC gesendet. Es werden keine weiteren Treiber (USB, FTDI etc.) oder Client-Software auf dem PC („TelitSerialPortMux.exe", „PythonDebug.exe" etc.) benötigt.

Enthält die PRINT-Anweisung einen Steuerbefehl, wird dieser zusätzlich vom *Python-Debugger* ausgeführt (LED ein-/ausschalten, GPO-Pin schalten):

```
""" ***************************************************
Python-Script  „pd_test.py"

Project : Python-Debugger - DEMO
Version : 1.1
Date    : 07.04.2007

Modul   : GM862-GPS, GE863-GPS
*************************************************** """

import MOD
import GPIO


print 'Program start:'                          # start message
counter = 0

while (1):
    print 'LED1-ON'                             # LEDs ON/OFF
    print 'LED2-OFF'
    print 'LED3-ON'

    counter = counter + 1
    print 'counter = ', counter                 # debug message

    res = GPIO.setIOdir(16,1,1)                 # Test-LED ON
    MOD.sleep(5)                                # wait 0,5s

    print 'LED1-OFF'                            # LEDs OFF/ON
    print 'LED2-ON'
    print 'LED3-OFF'

    print 'GPO2-OFF'                            # short trigger
    print 'GPO2-ON'                             #   on GPO2

    res = GPIO.setIOdir(16,0,1)                 # Test-LED OFF
    MOD.sleep(5)                                # wait 0,5s
```

Ausgabe im PC-Terminalprogramm

Ausgabe im PC-Terminalprogramm + Reaktion auf dem Debugger-Board

weitere Vorgehensweise / Modul-Einstellungen:

Auf dem TELIT-Modul muss die neueste Firmware geladen sein (GM862-GPS: 07.02.402 oder neuer, GE863-GPS : 07.02.702 oder neuer). Die Version der aktuell geladenen Firmware kann mit AT+CGMR ermittelt werden.

Das oben aufgeführte Python-Script wird auf dem TELIT-Modul installiert und gestartet:

1. oben aufgeführtes Python-Script „pd_test.py" auf das TELIT-Modul übertragen
   (bei installierter Python-Suite mit einem Rechts-Klick auf die zu übertragende Datei und „Download" oder der Software „TelitTest.exe")
2. danach Kommando AT#ESCRIPT="pd_test.py" eingeben
3. danach Kommando AT#SSCTRACE=1 eingeben
4. danach kann das Python-Script mit AT#EXECSCR oder einem Power-OFF/Power-ON + Modul ON gestartet werden
   (die Leitung DTR muss auf HIGH liegen)

Nach dem Startbefehl bzw. einem Power-ON wird der Übersetzungsvorgang des Python-Scripts auf dem TELIT-Modul gestartet. Schon dabei erfolgen etliche Ausgaben über die Debug-Schnittselle, die über die RS-232-Schnittstelle an den angeschlossenen PC übertragen werden.

Das Python-Script selbst fängt nach einigen Sekunden an zu laufen und überträgt die markierten Ausgaben zum angeschlossenen PC.

Die oben gelb markierten Ausgaben werden nur auf dem PC dargestellt, die grün markierten Texte werden vom *Python-Debugger* als Steuerbefehle erkannt und zusätzlich auf dem Board ausgeführt. Mit Hilfe dieser Befehle ist z. B. ein stand-alone Betrieb des Gerätes ohne angeschlossenen PC möglich.

Befehlsübersicht

Befehl	**Bedeutung**	**TELIT-Module ---> *Python-Debugger* (Steuerbefehle)**	**PC ---> *Python-Debugger* (PC-Befehle)**
CLEAR	alle Leuchtdiode ausschalten (um z. B. direkt nach dem Programmstart alte Zustände zu entfernen)	X	X
LEDx-ON	LEDx (1, 2 oder 3) einschalten (x = 1..3)	X	X
LEDx-OFF	LEDx (1, 2 oder 3) ausschalten (x = 1..3)	X	X
GPOx-ON	Testpin x einschalten (x = 1..3)	X	X
GPOx-OFF	Testpin x ausschalten (x = 1..3)	X	X
ON	TELIT-Modul über den Eingang ON/OFF einschalten		X *
OFF	TELIT-Modul über den Eingang ON/OFF ausschalten (incl. „richtiger" Abmeldung vom GSM-Netz)		X *
RESET	TELIT-Modul über den Eingang RESET resetten (sofortiger Reset ohne Netz-Abmeldung)		X *
RESTART	TELIT-Modul resetten (über RESET) und danach einschalten (über ON) (wird auch über den optionalen Taster eingeleitet)		X *
GET	Systemzustände und eingestellte Parameter abfragen		X
H	Hardwareversion des *Python-Debuggers* abfragen		X
F	Softwareversion der *Python-Debugger*-Firmware abfragen		X
Bx	Datenübertragungsrate der SPI-Schnittstelle einstellen (x = 1..4)		X
ADDCR-ON ADDCR-OFF	vom TELIT-Modul übertragene Zeilen um ein Carriage Return erweitern (ON/OFF) („Hyperterm"-friendly)		X
SSCDELAY	Siehe Troubleshooting		X

* = nur wenn die Signalleitungen RESET und ON/OFF angeschlossen sind

PC-Befehle:

Bei Befehlen, die vom PC zum *Python-Debugger* gesenden werden ist zu beachten:

- die Befehle werden über das PC-Terminalprogramm im Klartext über die serielle Schnittstelle an den *Python-Debugger* geschickt
- im Befehl enthaltene Leerzeichen werden ignoriert.
- Groß- und Kleinbuchstaben werden nicht unterschieden.
- jeder einzelne Befehl muss mindestens mit CR (Zeichen 13) beendet werden.
- nicht erkannte Befehle werden mit CRLF + '?' + CRLF quittiert.

Steuerbefehle:

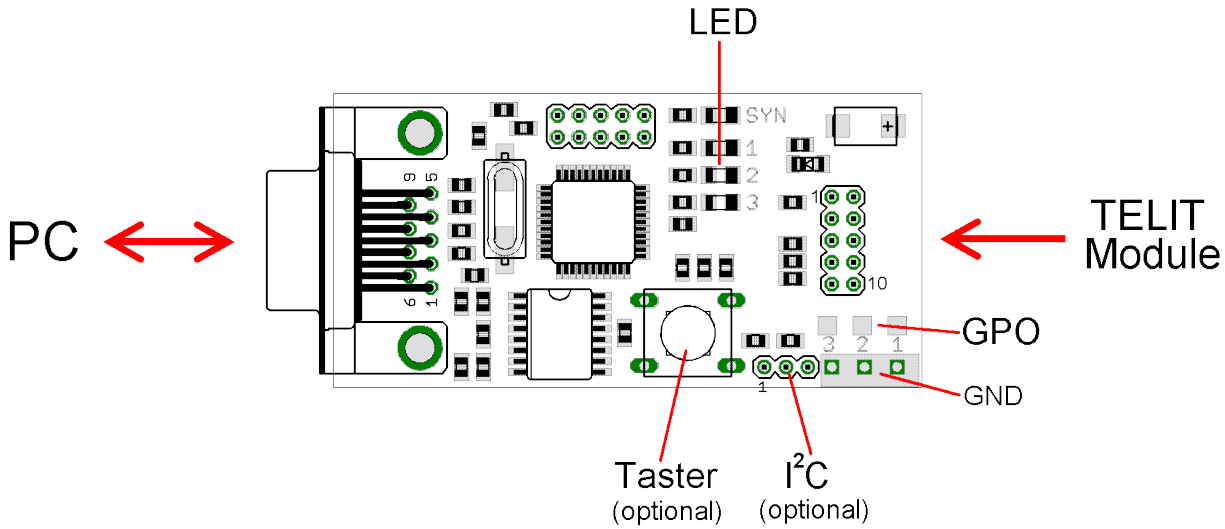
Bei Debug-Ausgaben, die vom TELIT-Modul zum *Python-Debugger* (= Debug-Ausgabe) gesenden werden ist zu beachten:

- sämtliche, in einem Python-Script enthaltene PRINT-Ausgaben werden über die serielle Schnittstelle an den angeschlossenen PC weitergeleitet (115200/8/N/1) und dort mit Hilfe eines Terminalprogrammes angezeigt
- enthält der Ausgabetext einen der oben angegeben Steuerbefehle (z. B. „LED2-ON"), wird dieser Befehl vom *Python-Debugger* zusätzlich „auf dem Board" ausgeführt
- diese Befehle müssen genau so geschrieben werden, wie oben angegeben (Der Controller macht aus Geschwindigkeitsgründen einen 1:1-Stringvergleich)!

Technische Merkmale:

- Schutzklasse: IP20, nur für trockene Räume
- RS-232-Schnittstelle fest eingestellt auf: 112500 / 8 / N / 1, kein Handshake
- Versorgungsspannung 3.4V ... 6.5V, verpolungssicher
- 3 einzeln schaltbare Signal-LED
- 3 einzeln schaltbare Ausgangssignale (kurzschlusssichere Push-Pull-Ausgänge)
- SYNC-Anzeige über eine LED (automatische Synchronisierung auf den proprietäte SSC0-Datenstrom)
- optionaler Taster: Funktion „RESTART"

Aufbau


LED
PC
TELIT
Module
GPO
GND
Taster
(optional)
I²C
(optional)
SYN
1
2
3
10
3 2 1

Troubleshooting

Keine Ausgabe auf dem PC	● SSC0 – Signale nicht richtig angeschlossen ● Vcc und GND nicht richtig angeschlossen ● falsche Firmware
Keine Ausgabe auf dem PC	● Geschwindigkeit des Terminalprogrammes nicht richtig eingestellt (115200 / 8 / N / 1, kein Handshake) ● Terminalprogramm nicht online
Keine Ausgabe auf dem PC	● Python-Script nicht gestartet (DTR auf High + Power ON oder AT#EXECSCR) ● „Select Active Script" (z. B. mit AT#ESCRIPT="pd_test.py") nicht zugewiesen
LEDs werden nicht / nicht richtig angesteuert	Steuerbefehle in den PRINT-Ausgaben nicht richtig geschrieben. Die Befehle müssen exakt so geschrieben sein wie oben angegeben (Gross-/Kleinschreibung beachten, keine Leerzeichen)
Hin und wieder werden falsche Zeichen übertragen	Dies ist ein bekanntes Problem der TELIT-Module und hat mit der Priorisierung der einzelnen Task unter Python zu tun (Python-Scripts haben die niedrigste Priorität und werden nur dann abgearbeitet, wenn die GSM-Engine gerade nichts zu tun hat). Dabei wird aus zeitlichen Gründen sporadisch das Bit 0 eines über den PRINT-Befehl gesendeten Bytes verfälscht, so wird z. B. aus dem Zeichen „C" (01000011) ein „B" (01000010). Da beide Zeichen richtig bzw. gewollt sein könnten, hat der *Python-Debugger* leider keine Chance, diesen TELIT-Modul-Fehler direkt zu korrigieren. Ein Besserung kann allerdings durch ein Timedelay bei der Übertragung erreicht werden. Der *Python-Debugger* fügt dabei während der Zeichenübertragung kurze Pausen ein. Der entsprechende Parameter heißt „SSCDELAY=xx" (xx in Millisekunden) und kann vom PC-Terminal-Programm im *Python-Debugger* eingestellt werden: SSCDELAY=xx (xx = 0 ... 100 ms) Default ist 0 ms (= keine Verzögerung). **ACHTUNG:** Die Übertragung verlangsamt sich bei Einsatz dieses Parameters wohlmöglich so lange, dass es auch dadurch zu Datenverlusten (dann allerdings durch Overruns) kommen kann. Wenn über die Debug-Schnittstelle wichtige Informationen übertragen werden sollen, muss eine entsprechende Datensicherung vorgenommen werden (Checksumme, Sendungswiederholungen etc.). **Nochmals:** Dieser Fehler wird nicht vom *Python-Debugger* verursacht und kann von diesem auch nicht völlig korrigiert werden!