

## Steuerung von Meßgeräten mit einer LAN-Schnittstelle über TCP/IP

Mit HTBasic ist es prinzipiell möglich, auch Meßgeräten über LAN mittels TCP/IP zu steuern. Dazu wird die Agilent SICL I/O Bibliothek benötigt. Die SICL-Bibliothek bietet über eine DLL (sicl32.dll) die Möglichkeit, mit Geräten über TCP/IP zu kommunizieren. Diese DLL bindet man einfach über das DLL Toolkit ein (LOAD DLL..).

Folgende grundsätzliche Funktionen der SICL-DLL werden benötigt:

```
DLL LOAD "SICL32"  
DLL GET "SHORT Sicl32:iopen" AS "Iopen"  
DLL GET "SHORT Sicl32:iwrite" AS "Iwrite"  
DLL GET "SHORT Sicl32:iread" AS "Iread"  
DLL GET "SHORT Sicl32:iclose" AS "Iclose"
```

Ein Beispiel für die Kommunikation zwischen HTBasic und einem Agilent 16000 Logic Analyzer finden Sie auf den nächsten Seiten.

--

**Tech Soft GmbH**, Test & Measurement Division  
HTB@TechSoft.de  
phone +49-30-40608 - 0 fax +49-30-40608-220  
<http://www.HTBasic.de>

```

10      ! DirectSiclSocket for Agilent 16700 Logic Analyzer
20      !
30      ! BOURBAKY Dll Toolkit Example for Real Instrument Remote Programming
40      ! -----
50      ! How easily work with 16700 from HTBasic
60      ! Only using existing SICL functions
70      !
80      PRINT "Loading DLL"
90      DLL LOAD "SICL32"
100     !
110     PRINT "Function Declaration"
120     DLL GET "SHORT Sicl32:iopen" AS "Iopen"
130     DLL GET "SHORT Sicl32:iwrite" AS "Iwrite"
140     DLL GET "SHORT Sicl32:iclose" AS "Iclose"
150     DLL GET "SHORT Sicl32:itimeout" AS "Itimeout"
160     DLL GET "SHORT Sicl32:iread" AS "Iread"
170     !
180     LONG Intf_hdl      ! interface session handler
190     LONG Inst_hdl     ! instrument session handler
200     !
220     INTEGER I
230     REAL T0
240     COM REAL Send,Receive
250     LONG Sent
260     !
270     INTEGER Answer
280     DIM Response$(256)
290     DIM Filename$(50)
300     Filename$="/hplogic/bky/myconf.____"      ! <= User must store the wanted conf in this file
310                                           !      using 16700 frontpanel
320     !
330     DIM Intf_symname$(5)
340     Intf_symname$="lan"
350     !
360     DIM Inst_symname$(80)
370     Inst_symname$="lan,6500[192.168.1.120]"    ! <= 16700 IP Address (User must check and change if
needed)                                           !      6500 is the socket number
380     !
390     !
400     DIM Comand$(80)
410     Sent=LEN(Comand$)
420     !
430     ON ERROR RECOVER Error_trap
440     T0=TIMEDATE
450     Intf_hdl=FNiopen(Intf_symname$)      ! Open interface session
460     IF Intf_hdl=0 THEN
470         PRINT "CAN'T OPEN INTERFACE SESSION"
480         GOTO Eop3
490     ELSE
500         PRINT "INTERFACE SESSION OPENed"
510     END IF
520     !
530     Inst_hdl=FNiopen(Inst_symname$)      ! Open instrument session
540     IF Inst_hdl=0 THEN
550         PRINT "CAN'T OPEN INSTRUMENT SESSION"
560         GOTO Eop2
570     ELSE
580         PRINT "INSTRUMENT SESSION OPENed"
590     END IF
600     !
610     ! Program will LOOP to check the ability to stay synchronous with 16700 execution
620     ! -----
630     !
640     LOOP
650         Loops=Loops+1
660         PRINT "==== Loop #";Loops;" ====="
670         !
680         !=== clear instrument =====
690         PRINT "clear"
700         Answer=FNSocket_send(Inst_hdl,"clear")
710     EXIT IF Answer<>0
720         Answer=FNSocket_rcv(Inst_hdl,Response$)
730     EXIT IF Answer<>0
740         !
750         !=== LOCK User interface =====
760         PRINT "lock"
770         Answer=FNSocket_send(Inst_hdl,"lock")
780     EXIT IF Answer<>0
790         Answer=FNSocket_rcv(Inst_hdl,Response$)
800     EXIT IF Answer<>0
810         !
820         !=== Ask for #version =====
830         PRINT "version : ";
840         Answer=FNSocket_send(Inst_hdl,"version")
850     EXIT IF Answer<>0
860         Answer=FNSocket_rcv(Inst_hdl,Response$)
870     EXIT IF Answer<>0
880         PRINT Response$
890     !

```

```

900      !=== Plugged Modules ? =====
910      PRINT "Modules List"
920      FOR I=1 TO 10
930          Answer=FNSocket_send(Inst_hdl,"modules -slot "&CHR$(64+I))
940          Answer=Answer+FNSocket_rcv(Inst_hdl,Response$)
950          IF POS(Response$,"No module")=0 THEN PRINT Response$
960      NEXT I
970  EXIT IF Answer<>0
980      !
990      !=== Loading analyzer configuration =====
1000     PRINT "loading configuration : ";Filename$
1010     Answer=FNSocket_send(Inst_hdl,"config -l "&Filename$)
1020  EXIT IF Answer<>0
1030     Answer=FNSocket_rcv(Inst_hdl,Response$)
1040  EXIT IF Answer<>0
1050      !
1060      !=== RUN analyzer =====
1070      PRINT "starting acquisition"
1080     Answer=FNSocket_send(Inst_hdl,"start")
1090  EXIT IF Answer<>0
1100      !
1110      !=== WAIT finished =====
1120     PRINT "wait for measurement complete"
1130     Answer=FNSocket_send(Inst_hdl,"wait -complete")
1140  EXIT IF Answer<>0
1150     Answer=FNSocket_rcv(Inst_hdl,Response$)
1160  EXIT IF Answer<>0
1170      !
1180      !=== Check Status =====
1190     PRINT "status : ";
1200     LOOP
1210         Answer=FNSocket_send(Inst_hdl,"status")
1220     EXIT IF Answer<>0
1230         Answer=FNSocket_rcv(Inst_hdl,Response$)
1240     EXIT IF Answer<>0
1250     EXIT IF POS(UPC$(Response$),"STOPPED")
1260     END LOOP
1270  EXIT IF Answer<>0
1280     PRINT Response$
1290      !
1300      !=== UNLOCK User interface =====
1310     PRINT "unlock"
1320     Answer=FNSocket_send(Inst_hdl,"unlock")
1330  EXIT IF Answer<>0
1340     Answer=FNSocket_rcv(Inst_hdl,Response$)
1350  EXIT IF Answer<>0
1360      !
1370     DISP "Started : ";DATE$(T0);" ";TIME$(T0);" / ";Loops;" Loops / ";Send;" Comands / ";Receive;
" Responses"
1380     END LOOP
1390     !
1400     !
1410     PRINT
Eop1:     !
1430     PRINT "CLOSE INSTRUMENT SESSION"
1440     Iclose(Inst_hdl)
Eop2:     !
1460     PRINT "CLOSE INTERFACE SESSION"
1470     Iclose(Intf_hdl)
Eop3:     !
1490     DLL UNLOAD ALL
1500     STOP
1510     !
Error_trap: !
1530     PRINT ERRM$
1540     GOTO Eop1
1550     END
1560     !
1570     DEF FNEnter_socket(LONG Hdl,Enter_str$,OPTIONAL S_timeout)
1580         ! Enter string, getting char by char
1590         INTEGER Answer
1600         INTEGER First_car=1
1610         LONG Bufsize=1
1620         LONG Actualcnt
1630         DIM C$(2)
1640         LONG Inst_timeout1=1000
1650         LONG Inst_timeout2=50
1660         IF NPAR=3 THEN Inst_timeout1=S_timeout
1670         Itimeout((Hdl),(Inst_timeout1))
1680         Enter_str$=""
1690         LOOP
1700             Answer=FNIread((Hdl),C$,(Bufsize),Actualcnt)
1710         EXIT IF Answer<>0
1720         EXIT IF Actualcnt=0
1730             Enter_str$=Enter_str$&C$
1740             IF First_car THEN
1750                 First_car=0
1760                 Itimeout((Hdl),(Inst_timeout2))
1770             END IF

```

```

1780         END LOOP
1790         RETURN Answer
1800     FNEND
1810     !
1820     DEF FNSocket_send(LONG Inst_hdl,Comand_$)
1830         COM REAL Send,Receive
1840         ALLOCATE Comand$[256]
1850         Comand$=Comand_$
1860         LONG Sent,Answer
1870         INTEGER Endi=0
1880         Comand$=Comand$&CHR$(10)
1890         Sent=LEN(Comand$)
1900         Answer=FNiwrite((Inst_hdl),Comand$,(Sent),Endi,Sent)
1910         Send=Send+1
1920         IF Answer<>0 THEN
1930             PRINT "Code Answer : ";Answer
1940             PRINT "ERROR WHILE SENDING COMAND"
1950             RETURN -1
1960         ELSE
1970             RETURN 0
1980         END IF
1990     FNEND
2000     !
2010     DEF FNSocket_rcv(LONG Inst_hdl,Response$)
2020         LONG Answer
2030         COM REAL Send,Receive
2040         Answer=FNEnter_socket(Inst_hdl,Response$)
2050         Receive=Receive+1
2060         ! Remove unwanted prompt and termination (if any)
2070         IF POS(Response$,"->") THEN Response$=Response$[1;POS(Response$,"->")-1]
2080         IF POS(Response$,CHR$(10))=LEN(Response$) AND LEN(Response$)<>0 THEN Response$=Response$[1;LEN
(Response$)-1]
2090         IF Answer=0 THEN
2100             PRINT "Code Answer : ";Answer
2110             PRINT "ERROR GETTING ANSWER"
2120             RETURN -1
2130         ELSE
2140             RETURN 0
2150         END IF
2160     FNEND

```