L&G Fugga PRV

L&G Fugga PRV communication protocol

Supported device types and versions

This protocol provides data reading/writing into Landis&Gyr devices, type PRV 1.xx, and 2.xx.

Communication line configuration

- Communication line category: Serial.
- Baud rate and transmission parameters are defined according to the parameters of the communication software in PRV.

Communication station configuration

- Communication protocol: **L&G FUGGA PRV**
- The station address is in the range of 0 up to 255. It is in decimal format according to the station number in the communication software in PRV (see below).

Station protocol parameters

You can define the following parameters:

Table 1

Keyword	Full name	Meaning	Unit	Default value
MR	Retry Count	The number of repeated requests if the communication error occurs.	-	3
MWR	Max Wait Retry	Maximum retries of reading the response until it is completed.	-	14
RT	Retry Timeout	The delay between request repetitions in case of a communication error.	s.ms	1.000 s
WT	Wait Timeout	The delay between reading the response until it is completed.	ms	300 millisec.
WFT	Wait First Timeout	The first wait after the request has been sent before reading the response.	ms	1200 millisec.
RD	Request Delay	Delay before sending the request.	ms	1000 millisec.
MBS	Maximum Buffer Size	Maximum size of request in bytes.	byte	60
DOFF	Date Offset	The correction of the date offset in PRV and D2000 System. D2000 System interprets the date as a number of days from 1-1-1972. To test PRV 1.64 (SW ver. 4.04) we had to set the value 20089 because PRV implements the date from 1-1-1917.	-	4806

A string containing the protocol parameters is defined as follows:

```
Key_word=value;Key_word=value; ...
```

Example:

```
MR=4;RD=500;
```

If there is used a keyword with an invalid value in the initialization string, there will be used corresponding default value according to the table 1.

I/O tag configuration

I/O tags: **AI, AO, CI, CO, DI, DO, TIR, TOR, TIA, TOA**

The address must be in text format for configuration. In comparison with the implementation in D2000 ver.3.xx, the range of types is not limited. The table shows the writing format of some types:

Type in D2000 v.3.xx	Write in D2000 v.4.x	Example
I/O	$xxx.name	$120.HW
PLT	PLTx.name	PLT5.PAR1
ZON	ZONx.name	
RGB	RGBx.name	
VIP	VIPx.name	
CVP	CVPx.name	
@MGR	@MGRx.name	
@MGG	@MGGx.name	
@SEZ	@SEZx.name	
@AI	@AIx.name	
@AO	@AOx.name	
@DI	@DIx.name	
@DO	@DOx.name	
@CI	@CIx.name	
@PHON	@PHONx.name	
@RING	@RINGx.name	
TTY	TTYx.name	
@PBUS	@PBUSx.name	
@MBUS	@MBUSx.name	
@OS	@OSx.name	
@BLN	@BLNx.name	
@BPS	@BPSx.name	
TSK	[tsknr]name	[10]S2
SYS	name	DATE

PRV (Fugga) protocol

Format of the message:

```
<STX><ADDRESS><FUNCTION><SEQ><CONTENT OF MESSAGE><ETX/ETB><CRC>
```

STX - Start of text, character 0x02

ADDRESS - double-digits in ASCII format - the range from 01 - 99.

FUNCTION - double-digits in ASCII format - the function number. The number 50 is added to the function number in response.

SEQ – a consecutive number of message - one byte - the values 0x41 – 0x5A

CONTENT OF MESSAGE - a message in ASCII format - the parts of the message are separated by “;”.

ETX - end of message

ETB - end of the block

CRC - checksum CRC-CCITT Cyclic Redundancy Check according to IBM

- to PRV - 6 bytes ASCII
- from PRV - 2 bytes binary

Acknowledgment:

ACK - O.K. - the message contains the ACK character

NAK - error - the message contains the NAK character

Functions:

Function	Number	Content of message	Response
Config Check	01	Config ID	ACK/NAK
New Point Def	02	ID:NAME;0:ConfigId;	ACK/NAK(memory)
Delete Point	03	ID;ConfigId;	ACK
Read All	04	---	ID:VALUE;...
ReadAllContinue	05	INDEX	ID:VALUE;...
Set Output	06	ID=Value;	ACK
Reset PRV	07	---	ACK
Clear Config	08	---	ACK

Solution of problematic situations

On the D2000 System side:

1. PRV does not respond in a defined limit.

2. Wrong checksum.

The message is repeated N-times - all necessary parameters can be set by the station protocol parameters.

On the PRV side:

Wrong checksum - PRV does not respond.

Changes in PRV (Fugga) protocol ver. 1.02

Usage: The communication with PRV1/PRV2 via the serial lines and a radio modem with OnLine configuration of transmitted values.

New features in comparison with ver. 1.01:

- time I/O tags were implemented
- optimization of communication - only the value changes are transmitted

The number of transmitted tags from one station is max. 255.

Communication parameters on the PRV side:

Task 3:

Communication port number in PRV: PRT variable <1..3>

Station address: variable AD$ - the address always consists of two digits, e.g. “01” - address 1.

Communication task number - 3 is the default. If it needs to be changed, modify tasks 254 and 253 - a restart after power failure and WatchDog Error. Task 4 must be modified - the references to the local variables of task 3.

WatchDog number - 7 is the default. If it needs to be changed, modify the task 253 and set a new WatchDog number in the calling of WADO function on line 305.

The maximum size of the transmitted message to PC - BLK variable - the limit is 160.

Baud rate: The parameter TTYPRT.BD on line 60.

Task number for data acquisition: Task 4 is used. If it needs to be changed, the task number on line 48 must be modified - the start of Task 4.

Task 4:

Parameter TSKFRM$ - defines an output format of the task. Actually, it defines an accuracy of transmitted float numbers.

Parameter WAIT DURING 5 on line 70 defines a value acquisition period in seconds.

Task listing for ver. 1.02

```
.TSK 3
```

```
1 -- FUGGA PROTOCOL (C) IPESOFT 1997 V 1.02
```

```
5 ad$ := "21"; -- station address
```

```
10 er := 1 ; cf$ ="";RT$ := "";mx:=100
```

```
15 sl := 2; bl = 60; cn := 1; wx := 5;
```

```
20 DIM N$(mx); DIM V$(mx); DIM NW(mx);
```

```
25 FOR I := 1 TO mx
```

```
26 V$(I) := ""; NW(I) := 0;
```

```
27 NEXT I;
```

```
48 STOP 4; RUN 4;
```

```
40 TSKPRP:=3
```

```
50 TSKTTY:=1
```

```
55 TSKPRIO:=15
```

```
56 SYSDIAG:=0;
```

```
58 ON ERROR GOTO 7000
```

```
60 TTY1.BD=19200
```

```
61 -- TTY1.PARCHK = 0
```

```
62 TTY1.SSB = 0
```

```
70 TTY1.NOEC=1
```

```
72 TTY1.NCON=1
```

```
74 TTY1.MOD=0
```

```
76 TTY1.NWCR=1
```

```
78 TTY1.NFCR=1
```

```
79 TTY1.NOX=1;
```

```
80 INITTY1
```

```
90 BEGIN_INKEY
```

```
100 cm$ := "" ;C$ := ""; ax$ := "";fn$ := "";CR$ := "";WT:=0
```

```
200 ST := 1; fc := 0;
```

```
300 INKEY A$
```

```
305 WADO(7,60) ; -- 7 is Watch Dog number
```

```
310 IF A$="" THEN WT := WT + 1 ELSE WT:= 0;
```

```
320 IF WT > wx THEN GOTO 100
```

```
330 IF WT > 0 THEN WAIT DURING 1; GOTO 300
```

```
400 FOR RI:= 1 TO LEN(A$)
```

```
450 C$ := A$[RI..RI]
```

```
500 ON ST GOSUB 600,800,1000,1200,1300,950,3000
```

```
540 cm$ := cm$ + C$
```

```
550 NEXT RI
```

```
560 GOTO 300
```

```
600 IF C$ = CHR$(2) THEN ST:= 2; ax$:="" ELSE ST:= 1
```

```
610 IF C$ = CHR$(24) THEN ST:=7;
```

```
620 cm$ := ""
```

```
700 RETURN
```

```
800 ax$ := ax$ + C$
```

```
820 IF LEN(ax$) < 2 THEN RETURN;
```

```
830 IF ax$ = ad$ THEN fn$:= "";ST := 3 ELSE ST := 1;
```

```
900 RETURN
```

```
950 RT$ := C$; ST := 4;
```

```
960 RETURN
```

```
1000 fn$ := fn$ + C$
```

```
1020 IF LEN(fn$) < 2 THEN RETURN
```

```
1025 fc := VAL(fn$); ST := 6;
```

```
1030 fn$ := STR$(fc+50);
```

```
1040 RETURN
```

```
1200 IF (C$ = CHR$(3)) OR (C$ = CHR$(23)) THEN ST := 5; CR$ := ""
```

```
1210 RETURN
```

```
1300 CR$ := CR$ + C$ ; C$ := "";C4$ := "";
```

```
1320 IF LEN(CR$) = 6 THEN GOTO 1400;
```

```
1340 RETURN
```

```
1400 C1 := VAL(CR$[1..3]); C2 := VAL(CR$[4..6]);C1$ := "";
```

```
1410 C1$ := CHR$(C1) + CHR$(C2);
```

```
1415 C4$ := CRC$(4,cm$)
```

```
1420 IF C1$ = C4$ THEN GOTO 1500 ELSE ST := 1;
```

```
1430 RETURN
```

```
1500 ON fc GOSUB 2100,2200,2300,2400,2500,2600,2700,2800
```

```
1505 ST := 1;cm$ := ""
```

```
1510 RETURN
```

```
2100 ET := INSTR(7,cm$,";");
```

```
2110 FOR I := 1 TO mx
```

```
2115 IF N$(I) <> "" THEN NW(I) := 1;
```

```
2117 NEXT I;
```

```
2120 IF cm$[7..ET-1] = cf$ THEN GOTO 2125 ELSE 2130
```

```
2125 SB(CHR$(6),3,ad$,fn$,RT$); RETURN
```

```
2130 SB(CHR$(21),3,ad$,fn$,RT$); RETURN
```

```
2200 S := 7
```

```
2210 FM := FREE
```

```
2220 IF FM < 1000 THEN SB(CHR$(21),3,ad$,fn$,RT$); RETURN
```

```
2230 E := INSTR(S,cm$,":")
```

```
2240 ix := VAL(cm$[S..E-1]);
```

```
2245 IF ix > mx THEN SB(CHR$(21),3,ad$,fn$,RT$); RETURN;
```

```
2250 ET := INSTR(E+1,cm$,";");
```

```
2255 IF ix = 0 THEN cf$ := cm$[E+1..ET-1]; GOTO 2270;
```

```
2260 N$(ix) := cm$[E+1..ET-1];
```

```
2270 S := ET+1;
```

```
2280 IF cm$[S..S] = CHR$(3) THEN GOTO 2290 ELSE GOTO 2210
```

```
2290 SB(CHR$(6),3,ad$,fn$,RT$); RETURN;
```

```
2300 E := INSTR(cm$,";")
```

```
2340 ix := VAL(cm$[7..E-1]);
```

```
2350 N$(ix) := ""
```

```
2360 ET := INSTR(E+1,cm$,';')
```

```
2370 cf$ := cm$[E+1..ET-1]
```

```
2380 SB(CHR$(6),3,ad$,fn$,RT$)
```

```
2390 RETURN
```

```
2400 cn := 1;
```

```
2402 SB$ := "" ;VL$ := ""
```

```
2405 FOR I:= cn TO mx
```

```
2407 IF NW(I) = 0 THEN GOTO 2420
```

```
2408 IF N$(I) = "" THEN GOTO 2420
```

```
2410 SB$:=SB$+STR$(I)+":"+V$(I)+";"; NW(I):=0; IF LEN(SB$)>bl THEN GOTO 2445
```

```
2420 NEXT I
```

```
2430 SB(SB$,3,ad$,fn$,RT$)
```

```
2440 RETURN
```

```
2445 SB(SB$,23,ad$,fn$,RT$);RETURN;
```

```
2500 cn := VAL(cm$[7..9]); GOTO 2402
```

```
2600 E := INSTR(cm$,"=")
```

```
2640 ix := VAL(cm$[7..E-1]);
```

```
2645 ET := INSTR(E+1,cm$,";");
```

```
2650 LVAL(N$(ix)) := VAL(cm$[E+1..ET-1])
```

```
2660 SB(CHR$(6),3,ad$,fn$,RT$)
```

```
2665 V$(ix) := STR$(VAL(N$(ix)));
```

```
2667 NW(ix) := 1;
```

```
2670 RETURN
```

```
2700 SB(CHR$(21),3,ad$,fn$,RT$)
```

```
2701 RETURN
```

```
2800 FOR I:= 1 TO mx
```

```
2810 N$(I) := ""; V$(I) := ""; NW(I) := 0;
```

```
2820 NEXT I
```

```
2825 cf$ := ""
```

```
2830 SB(CHR$(6),3,ad$,fn$,RT$)
```

```
2840 RETURN
```

```
3000 fn$ := fn$ + C$
```

```
3010 IF LEN(fn$) = 5 THEN GOTO 3100
```

```
3020 RETURN
```

```
3100 IF fn$ = "FUGGA" THEN GOTO 3200
```

```
3150 ST := 1; fn$ := ""; RETURN;
```

```
3200 WADO(7,-1);SDTTY1;SYSDIAG := 1; STOP;
```

```
7000 ON er GOTO 8100,8200
```

```
7100 RESUME 100
```

```
8100 RESUME 100;
```

```
8200 er := 1;
```

```
8300 VL$ := "???";
```

```
8400 RESUME 2410;
```

```
9999 END
```

```
.TSK 4
```

```
10 ON ERROR GOTO 200;
```

```
15 TSKFRM$ :=:(S:2) -- output format - transmission of float values
```

```
20 VL$ := ""
```

```
30 FOR I:= 1 TO [3]mx
```

```
40 IF [3]N$(I) = "" THEN GOTO 60 ELSE VL$ := STR$(VAL([3]N$(I)));
```

```
50 IF VL$ <> [3]V$(I) THEN [3]V$(I) := VL$; [3]NW(I) := 1;
```

```
60 NEXT I
```

```
65 WADO(6,20); -- watch dog for task 4
```

```
70 WAIT DURING 5; -- value acquisition period
```

```
80 GOTO 20
```

```
200 VL$ := "???";
```

```
210 RESUME 50;
```

```
500 END
```

```
.TSK 253
```

```
5 EA := EVADR
```

```
10 IF EA=$340.ADR AND $340.ERR = 20 AND $340.ER1=7 THEN GOTO 30 ELSE GOTO 100
```

```
20 ; -- 7 is Watch Dog number, 3 is the communication task number
```

```
30 WADO(7,-1); STOP 3; RUN 3,40; GOTO 200
```

```
100 IF EA=$340.ADR AND $340.ERR = 20 AND $340.ER1=6 THEN GOTO 130 ELSE GOTO 200
```

```
130 WADO(6,-1); STOP 4; RUN 4; GOTO 200
```

```
200 END
```

```
EXIT
```

```
TSK 254
```

```
5555 RUN 3,40
```

```
EXIT
```

\

```
.PROC SB
```

```
ENTRY (IN:B$,Ex,A$,F$,R$)
```

```
1050 BF$:=CHR$(2)+A$+F$+R$+B$+CHR$(Ex)
```

```
1060 CR$:=CRC$(4,BF$);
```

```
1070 BF$:=BF$+CR$
```

```
1076 FOR I:=1 TO 15 ; PRINT CHR$(255); NEXT I
```

```
1100 PRINT BF$
```

```
1200 WAIT DURING 1
```

```
1500 CLITTY1
```

```
7000 ENDPROC
```

```
EXIT
```

Literature

-

Changes and modifications

- May 2000 - Transferring of protocol into D2000 ver. 4.10

Document revisions

- Ver. 1.0 – May 9, 2000 – Creation of document

Related pages:

Communication protocols