The SG Driver

The SG driver provides a simple text-based API to the North device, called Simple Gateway, allowing you to request and adjust any object values within it. Available for Commander and ObSys.

This document relates to SG driver version 1.0

Please read the *Commander Manual* or *ObSys Manual* alongside this document, available from *www.northbt.com*

Contents

Purpose of SG Driver

The SG driver provides a simple text-based API to the North device, called Simple Gateway, allowing you to request and adjust any object values within it.

Use this driver to integrate data available using North interface technology directly into your own application or product.

The Simple Gateway API provides a request-reply interface using an RS232 interface. Your application makes a single request to the API, and then receives a reply some time later.

If your application requires a faster response time, then consider using the DataSync driver that provides an API to the North device's Essential Data and Extra Data.

A range of other APIs are available: the Telnet driver provides a TCP query-response interface; the DataSync driver provides a client-server interface; JSONData and JSONNotify drivers provide JSON web services; and many other standards-based drivers are available.

Values

You can connect to the Simple Gateway API using simple text-based line commands. The API presents all object values from the North device and connected systems. Refer to the North device and driver manuals for a list of objects available.

The functions currently available from the API are:

- Read function – read an object value from the North device or system
- Write function – set an object value within the North device or system.

Detailed Operation

The Simple Gateway API provides a simple text-based protocol for an application to request and adjust any object values within the North device or connected system.

The Simple Gateway protocol uses a client-server model. Typically, your client application would make a single request for an object value and then await a reply from the Simple Gateway server. As the request is routed to the connected system, a reply may take several seconds. Once your application has received a reply, it can make the next request.

Connection

Establish a connection from your application by connecting to an RS232 COM port on the North device.

The connection is a full-duplex bi-directional link, and supports a range of baud rate, byte format and flow control options. The driver's default configuration is 19200 baud, no parity, 8 data bits, 1 stop bit, and no flow control. Refer to *Making the Cable* for the details of the RS232 cable.

Message Format

The client sends a request and the server a reply message, as documented in each API method below.

Messages must be formatted as a line of text with ASCII character encoding. Each line of text, or command, must end with a termination character (default value: carriage return, ASCII 13) – represented in this manual using the symbol: ↵.

Request Format

Enter the request command at the ready prompt. Once a request has been made, another cannot be made until the server replies.

The default ready prompt is '>', which can be changed within the driver setup.

Here is a sample ready prompt and request:

```
>O.PL↵
```

Reply Format

If the request is successful, the server will respond with the reply prompt along with any value requested.

The default reply prompt is 'Rep', which can be changed within the driver setup.

Here is a sample reply, followed by the ready prompt:

```
Rep:BSDC Energy Management↵
>
```

Error Reply

If the request is unsuccessful, the server will respond with the error prompt along with an error reason.

The default error prompt is ‘Err’, which can be changed within the driver setup.

Error Reasons

Error Reason	Meaning
Invalid Object	Object reference is not valid
Invalid Action	The requested read/write action cannot be performed on this object reference
Invalid Value	The value set is not valid
Device Delivery Fault	The destination system has not replied. Check system and try again.
Retry…	Retry your request again
Unknown Error	Other error reason

Here is a sample error reply, followed by the ready prompt:

```
Err:Device Delivery Fault↵
>
```

API Methods

Read a Value

This method requests an object value.

On sending this message, the server will route the request to the object. Once the object responds, which may take up to four seconds, the server will then reply to the client.

Refer to the North device's manual or the relevant driver manual, for details of the object references available.

Message Format

Request

```
object↵
```

Reply

```
reply:value↵
ready
```

Parameters

Parameter	Description
object	Object reference to read
value	Object value
reply	Reply prompt, set within the driver setup
ready	Ready prompt, set within the driver setup
↵	Message end character, set within the driver setup

Example

In this example, the client request the number of licence units in a North device, using object O.O.IL.T. The server responds with the value '1'.

Query

```
O.O.IL.T↵
Rep:1↵
>
```

Write a Value

This method sets the value of an object.

On sending this message, the server will route the request to the object. Once the object acknowledges the write, which may take up to four seconds, the server will then reply to the client.

Refer to the North device's manual or the relevant driver manual, for details of the object references available.

Message Format

Request

```
object=value↵
```

Reply

```
reply:Ok↵
ready
```

Parameters

Parameter	Description
object	Object reference to write
value	Object value to write
reply	Reply prompt, set within the driver setup
ready	Ready prompt, set within the driver setup
↵	Message end character, set within the driver setup

Example

In this example, the client sets an Essential Data value in a North device, using object UD.P1.O6, to the value '21'. The server responds with a write acknowledgement.

Query

```
UD.P1.O6=21↵
Rep:Ok↵
>
```

Using the Driver

On ObSys and Commander, the SG driver is pre-installed. Once started, you will need to set up the driver before you can make requests to it.

The SG driver uses zero licence units.

Making the Cable

Using the RS232 cable specification (Fig. 1), connect the North device COM port to the third-party device's COM port using a null-modem cable. Connector types at each end of the cable are shown.

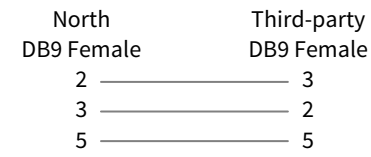

Fig 1 North to Third-party cable

If **Hardware Flow** is enabled in the driver setup, then the cable should also include RTS and CTS signals (North device pins 7 and 8).

The maximum RS232 cable length is 15m.

Starting the Interface

To start an interface using the SG driver, follow these steps:

- → **Start Engineering** your North device using ObSys
- → Navigate to **Configuration, Interfaces,** and set an unused **Interface** to 'SG' to start the particular interface
- → Navigate to the top-level of your North device and re-scan it

The driver setup object (Mc), labelled **SG Setup**, should now be available.

Setting up the Driver

To set up the driver, follow these steps:

- → Navigate to the **SG setup** object (Mc). For example, if you started interface 1 with the driver earlier, then the object reference will be 'M1'
- → Set **RS232 Com Port** object (RS.COM) to select which serial port on the North device the third-party is connected to
- → Set **Baud Rate** (RS.BR), **Byte Format** (RS.BF), and **Hardware Flow** (RS.HF) to match your device

Checking Communications

Connect to the Simple Gateway API using a terminal emulation application. The default connection is 19200 baud, no parity, 8 data bits, 1 stop bit.

Once connected, send a carriage return (ENTER) to the API and it will respond with the ready prompt '>'.

Read an object by typing 'O.T' then ENTER. The API will respond with the North device's current date and time.

Object Specifications

Once an interface is started, one or more extra objects become available within the top-level object of the device. As with all North objects, each of these extra objects may contain sub-objects, (and each of these may contain sub-objects, and so on) – the whole object structure being a multi-layer hierarchy. It is possible to navigate around the objects using the ObSys Engineering Software.

Each object is specified below, along with its sub-objects.

Example Object Reference

An example of a reference to an object in the same device: the SG Setup (M1) contains a Ready Prompt (IF.P) – therefore the complete object reference is 'M1.IF.P'.

An example of a reference to an object in a different device: the IP network object (IP) contains the Default Commander object (CDIP), which contains the object above (M1.IF.P) – therefore the complete object reference is 'IP.CDIP.M1.IF.P'.

Device Top-Level Objects

When an interface is started using the SG driver, the objects below become available within the top-level object of the device. For example, if Interface 1 is started, then the object with a reference 'M1' becomes available.

Description	Reference	Type
SG Setup Set up the SG driver, started on interface *c* (*c* is the interface number)	M*c*	Fixed Container: On the Commander platform this will be *[CDM v20\SG v10]* On the ObSys platform this will be *[OSM v20\SG v10]*

SG Setup

Object Type: *[OSM v20\SG v10]*
Object Type: *[CDM v20\SG v10]*

The SG Setup contains the following objects:

Description	Reference	Type
RS232 COM Port	RS.COM	Obj\Num; Range: 0…8; Adjustable
Baud Rate	RS.BR	Obj\Num; Adjustable Values: 1200,2400,4800, 9600, 19200, or 38400
Byte Format Sets the RS232 parity, data bits, and stop bits	RS.BF	Obj\ENum: 0..11; Adjustable See note 1
Hardware Flow Sets the RS232 hardware flow control	RS.HF	Obj\ENum; Adjustable Values: 0=None, 1=Rx/Tx Flow control, 2=Tx enable
Message End (ASCII char) Sets the ASCII character code that the driver will look for as the message termination character	O*x*	Obj\Num; Adjustable; Range 0..255 See note 2
Ready Prompt Optionally set this object to make the driver return this string as the interface's prompt text	IF.P	Obj\Text: 20 chars; Adjustable
Reply Prompt Optionally set this object to make the driver return this string as the interface's reply text, when a valid message has been received from the third-party device	IF.R	Obj\Text: 20 chars; Adjustable
Error Prompt Optionally set this object to make the driver return this string as the interface's error text, when an invalid message has been received from the third-party device	IF.E	Obj\Text: 20 chars; Adjustable
Echo Enable If set, causes the driver to echo back each character sent to it. This may be required for certain kinds of third-party device	IF.M.E	Obj\NoYes; Adjustable

Notes

1 Byte format can have the following values:

Value	Parity	Data bits	Stop bits
0	None	8	1
1	None	8	2
2	None	7	1
3	None	7	2
4	Odd	8	1
5	Odd	8	2
6	Odd	7	1
7	Odd	7	2
8	Even	8	1
9	Even	8	2
10	Even	7	1
11	Even	7	2

2 ASCII codes have a value range 0..255:

Char	Code	Char	Code	Char	Code	Char	Code	Char	Code	Char	Code	Char	Code	Char	Code
NUL	0	DLE	16	SPC	32	0	48	@	64	P	80	`	96	p	112
SO	1	XON	17	!	33	1	49	A	65	Q	81	a	97	q	113
STX	2		18	“	34	2	50	B	66	R	82	b	98	r	114
ETX	3	XOF	19	#	35	3	51	C	67	S	83	c	99	s	115
EOT	4		20	$	36	4	52	D	68	T	84	d	100	t	116
ENQ	5	NAK	21	%	37	5	53	E	69	U	85	e	101	u	117
ACK	6	SYN	22	&	38	6	54	F	70	V	86	f	102	v	118
BEL	7	ETB	23	‘	39	7	55	G	71	W	87	g	103	w	119
BS	8	CAN	24	(	40	8	56	H	72	X	88	h	104	x	120
HT	9	EM	25	)	41	9	57	I	73	Y	89	i	105	y	121
LF	10	SUB	26	*	42	:	58	J	74	Z	90	j	106	z	122
VT	11	ESC	27	+	43	;	59	K	75	[	91	k	107	{	123
FF	12	FS	28	,	44	<	60	L	76	\	92	l	108	|	124
CR	13	GS	29	-	45	=	61	M	77	]	93	m	109	}	125
SO	14	RS	30	.	46	>	62	N	78	^	94	n	110	~	126
SI	15	US	31	/	47	?	63	O	79	_	95	o	111	DEL	127

Driver Versions

Version	Build Date	Details
1.0	25/4/1997	Driver recompiled for RS485/422
1.0	20/2/2002	Removed EXMSG_ENABLE flag and reworked driver
1.0	16/6/2016	Add default baud rate and other default parameters

Next Steps…

If you require help, contact support on 01273 694422 or visit *www.northbt.com/support*

North Building Technologies Ltd
+44 (0) 1273 694422
support@northbt.com
www.northbt.com

Author: BS
Checked by: JF

Document issued 17/06/2016.