Semiconductors

UM_1SPP

BGB203 BT 2.0 Serial Port Profile Module User's Guide

Rev. 1.0.3 — 21 December 2005

Document information

Info	Content
Keywords	BGB203 ; Serial Port Profile ; Bluetooth
Abstract	User's Guide for the BGB203 Bluetooth 2.0 Serial Port Profile Module.

PHILIPS

Revision history

Rev	Date	Description
1.0.0	2005 Aug 15	Initial Revision
1.0.1	2005 Sep 30	Added Tuning Information
1.0.2	2005 Oct 20	Added Sleep Mode Power Handling Description
1.0.3	2005 Dec 21	Added Dynamic Settings Information

Contact information

For additional information, please visit: http://www.semiconductors.philips.com

For sales office addresses, please send an email to: sales.addresses@www.semiconductors.philips.com

1. Introduction

The 1SPP embedded firmware uses the Philips' BGB203 Bluetooth SIP to provide the Bluetooth Serial Port Profile (SPP) to embedded application designs. This firmware exposes a command set that follows the V.25 and GSM conventions to the application developer that can be accessed through a physical Universal Asynchronous Receive Transmit (UART) interface. These commands are translated by the 1SPP firmware into Bluetooth features. More specifically, the Bluetooth protocol stack features that are included in the 1SPP module are the Bluetooth RF, Baseband/Link Controller (LC), Link Manager (LM), Logical Link and Adaptation Protocol (L2CAP), Service Discovery Protocol (SDP), RFCOMM Protocol, Serial Port Profile (SPP), and the Generic Access Profile (GAP). By including the Bluetooth protocol stack in the firmware and providing an easy to use command set which applications can call externally (see Figure 1), the host processor's requirements are minimized.

In addition to providing the Bluetooth protocol stack functionality listed above, the firmware also provides several features to further support embedded applications that use the RS-232 protocol. These features include full support for DTE and DCE devices, including RTS/CTS flow control, and full support of all modem signals (CD, RI, DTR, and DSR). The 1SPP firmware is highly configurable and can be configured to support only the features that are required. For very simple applications, full functionality can be achieved without using any of the RS-232 signals.

Date of release:21 Dec 2005

Published in The Netherlands

2. Features

This section provides an overview of the features provided by the SPP Module firmware. More detail on the features is provided in later sections.

2.1 Command Interface

- Subset of V.25
- Extensions following the GSM command convention
- Configurable via command interface
- Configurable escape sequence

2.2 Physical Interface

- Universal Asynchronous Receive Transmit (UART)
- Configurable UART
    - Baud rate
    - Word length
    - Parity
    - Stop bits
    - RTS/CTS flow control
    - DTR/DSR flow control
- RS-232 signal support
    - Support for CD, RI, DTR/DSR (including pass-through)
    - Support for DCE or DTE device
- Active connection output

2.3 Bluetooth Features

- Supports Bluetooth 1.2 features of BGB203
    - Fast connection
    - Adaptive Frequency Hopping (AFH)
- Entirely embedded Bluetooth protocol stack
    - Link Controller (LC)
    - Link Manager (LM)
    - Logical Link and Adaptation Protocol (L2CAP)
    - Service Discovery Protocol (SDP)
    - RFCOMM Protocol (RFCOMM)
    - Serial Port Profile (SPP)
    - Generic Access Profile (GAP)

2.4 Bluetooth Functionality

- Inquiry
- Service Discovery with filter
- Remote Name discovery
- Link Key management
- Pairing (active and passive)
- Serial Port Profile server
- Configurable service name
- Configurable access control

Date of release:21 Dec 2005

Published in The Netherlands

- Serial Port Profile client
- Configurable RFCOMM port
- Configurable connection attempts
- Configurable local device discoverability
- Configurable local device name
- Configurable local class of device
- Configurable link supervision timeout
- Configurable PIN codes
- Configurable Security and/or Encryption
- Configurable sniff low power mode
- Master/slave role switch supported

2.5 Test Modes

- Bluetooth test mode
- FCC/Bluetooth transmission test modes
- Calibration mode (RXTUN)

2.6 Miscellaneous Features

- Configuration storable to Flash
- Automatic re-connect mode
- Low power sleep mode

Date of release:21 Dec 2005

Published in The Netherlands

3. Functional Overview

The 1SPP firmware builds on the strengths of the Philips' BGB203 SIP and adds the Serial Port Profile to the module. This provides a low-cost, low-power, highly configurable, serial cable replacement option. The 1SPP module follows the Bluetooth standard and is qualified by the Bluetooth SIG.
The 1SPP firmware is controlled by input received on the UART interface. To make development quicker and easier, the format of the input follows the typical format that a modem accepts. All commands are prefixed by the **AT** prefix. Figure 1 illustrates the overall architecture of the 1SPP firmware/BGB203 combination.

To design the 1SPP module into a product, the designer should follow the same board layout and connections as for the BGB203 SIP in any other design that uses the UART interface to communicate with the Bluetooth controller. This provides the basic communication to the 1SPP firmware. If the design also calls for the more advanced input and output features of the firmware, additional GPIO lines will also need to be connected. The advanced features that require additional connections include the modem signals (CD, RI, DTR, and DSR) and the active connection output. The following table lists the GPIOs that are used by the 1SPP firmware for the advanced options:

Table 1: Additional GPIO List

Feature	GPIO	Direction	Description
EXT_CLK	2	Output	Used for testing and to calibrate RXTUN
CD	11	Input/Output	Carrier Detect signal (direction depends if the device is configured to be a DCE or DTE device)
RI	12	Input/output	Ring Indicator signal (direction depends if device is configured to be DCE or DTE device)
DTR	13	Input/output	Data Terminal Ready signal (direction depends if device is configured to be DCE or DTE device)
DSR	14	Input/output	Data Set Ready signal (direction depends if device is configured to be DCE or DTE device)
Active Connection.	17	Output	Active high when there is a Bluetooth connection. Low when there is no Bluetooth connection.

The remaining connections that are used by the 1SPP firmware are the UART transmit, UART receive, and UART flow control signals (RTS and CTS). Please see the BGB203 datasheet for more information on the above listed input and output features. Note that all signals (except for UART transmit and UART receive) are not required, and are only required if the device is configured to support the required functionality.

Date of release:21 Dec 2005

Published in The Netherlands

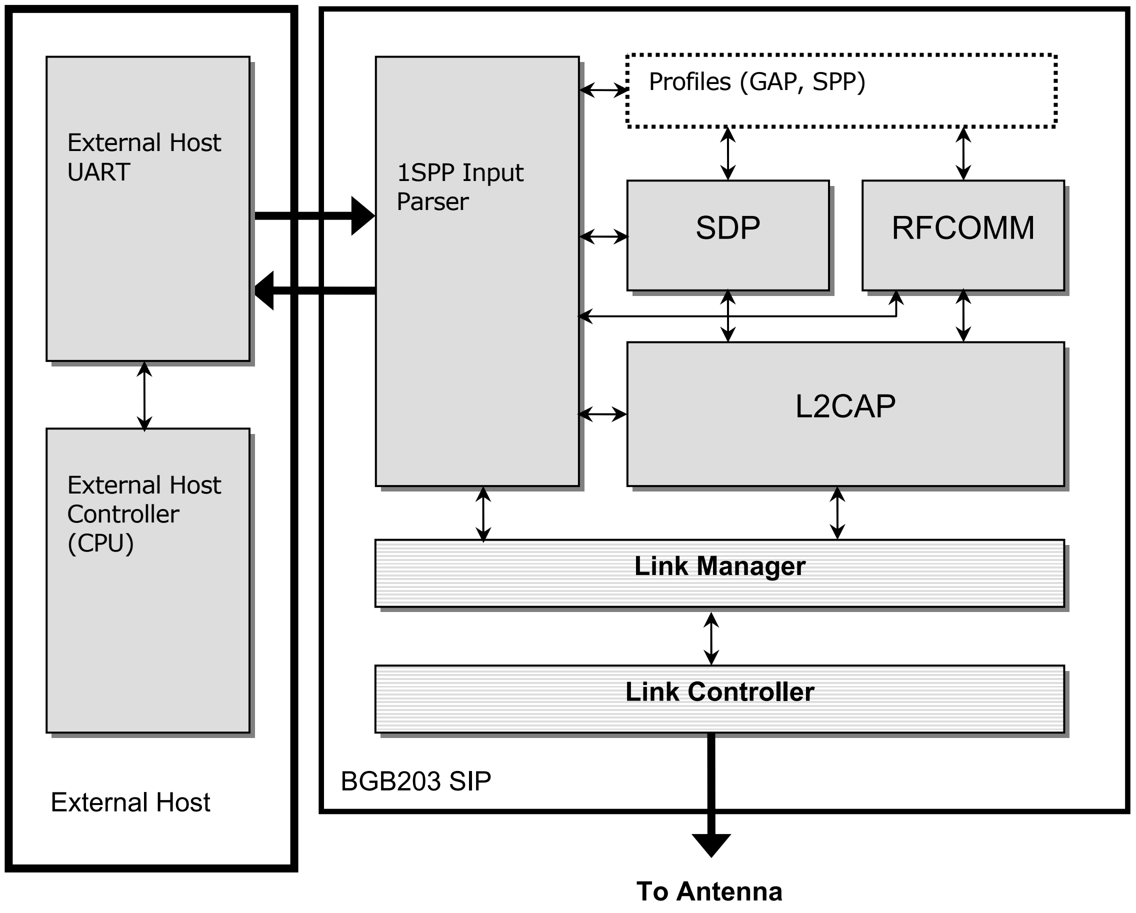

Figure 1:

The 1SPP firmware operates in one of two states. The states are called Command Mode and Data Mode. In Command Mode, the 1SPP firmware assumes that data coming from the UART are commands. The firmware parses the commands and processes them. In this mode, feedback is given (provided it is not disabled via configuration) about the status of the command.
Data Mode is entered when the device is attempting to make an SPP connection or has been configured to accept SPP connections. The device is always in Data Mode when there is an active SPP connection. There are two ways to enter Data Mode. One is by entering the Bluetooth SPP Server command (**AT+BTSRV**). The other is by entering the Bluetooth SPP client command (**AT+BTCLT**). While in Data Mode, everything that is received on the UART from the host processor will be transmitted over the Bluetooth SPP connection and all data that is received from the Bluetooth SPP connection will by transmitted to the host processor. Thera are two ways to enter Command Mode from Data Mode. The method used depends on how the device is configured).
In the first method of entering Command Mode from Data Mode, the host processor instructs the device to go into Command Mode. This can be done by the host processor issuing the escape sequence, or by the host processor lowering the DTR or DSR signal. The signal to lower depends on whether the device is configured to be a DTE or DCE

device. Whichever line is an input to the BGB203 chip should be lowered. The choice of using the escape sequence or the DTR/DSR signal is configurable by the host processor. The escape sequence to use is also configurable. The escape sequence can be configured to be any set of characters received over the UART (from 1 to 4 characters in length). Note that either the escape sequence method or the DTR/DSR method can be configured, but not both. This allows the host processor multiple options for different scenarios. Also note that when Command Mode is activated any connected Bluetooth SPP connection is disconnected.

The second method to enter Command Mode from Data Mode happens when a Bluetooth SPP connection is dropped. This can happen either due to a disconnected connection or a failure to connect at all. This method can be overridden by configuring the device in use a mode called Automatic Connection Mode. In this mode, the firmware always stays in Data Mode when the connection is complete. In the case of a client, it will automatically attempt to reconnect to the remote Bluetooth device. In the case of a server, the device will simply await another incoming connection. The host processor can monitor either Carrier Detect (CD) or the active connection output to determine when the Bluetooth connection is actually active.

If Automatic Connection Mode is not used, the device will inform the host processor of a connection by sending a connection string or a disconnection string. The host processor also can monitor either Carrier Detect (CD) or the active connection output to determine when the Bluetooth connection is active. The connection string is in the following format:

```
CONNECT xxxxxxxxxxxx
```

xxxxxxxxxxxx is the ASCII coded hexadecimal Bluetooth device address of the remote Bluetooth device that is connected. This address will always be 12 digits in length. For example, if a connection was made to the Bluetooth device with Bluetooth device address 012345678ABC, the connection string will be:

```
CONNECT 0123456789ABC
```

When the device is disconnected, the string that sent to the host processor will be:

```
NO CARRIER
```

Note that a carriage return/line feed pair will also be output after the above text. Please read the 1SPP AT command set information later in this document for more information on the format of commands and responses. Also note that the above connection and disconnection strings will not be output if Automatic Connection Mode is configured.

Automatic Connection Mode is useful when the host processor does not want to process any data that wouldn't normally be in the input UART stream. This means that the 1SPP firmware will not output any data or accept any data except the escape sequence (if configured) in this mode. The device can be configured to boot into Automatic Connection Mode, thus freeing the host processor from performing any 1SPP specific actions. (Note that at some point the 1SPP firmware has to be configured, either via the UART or by the external configuration program.) When using Automatic Connection Mode for client connections, further parameters can be configured to help minimize the radio activity, and thus lower power consumption. These parameters are the number of connection attempts and the connection period. These parameters control how often and how many connection attempts are to be performed. See the client command more information (**AT+BTCLT**).

In addition to the ability to support Automatic Connection Mode, the device can be configured to use different Bluetooth security modes. These modes are no security, authentication (pairing) required, and encryption required. No security means that the 1SPP firmware will not force any security procedures during connection. Authentication required means that the 1SPP firmware will force Bluetooth authentication when a connection is either accepted or made. Bluetooth authentication requires the use of Bluetooth PIN codes and/or Bluetooth link keys. The final security mode, encryption, is a superset of authentication. This mode encrypts all data that is sent over the Bluetooth connection. Encryption requires a Bluetooth link key that is generated during authentication. Therefore, encryption can be thought of as a superset of authentication since authentication must be performed prior to the actual encryption.

While in Data Mode, the 1SPP firmware supports the ability to map the state of the RS-232 input signals; RI, CD, DTR, and DSR, over the Bluetooth link. Likewise, the output RS-232 signals will be mapped to the signal state that is being reported by the remote device. This functionality is not the default functionality. By default, DTR and DSR will be used for local flow information (similar to how a modem behaves). When the signals are not mapped across the Bluetooth connection, the CD signal, if it is an output, will reflect the Bluetooth connection state (asserted if there is a connection). If the CD signal is an input signal, then it will be ignored. In this mode, the RI signal is mapped directly through.

The configuration of input and output RS-232 signals depends on whether the device is configured as a DTE or DCE device and can be seen in the following tables. The RTS/CTS lines are never mapped over the Bluetooth connection and are only used for flow control between the host processor and the BGB203 device. Note that the device does not have to be configured to use all of the signals discussed in this section.

Table 2: DCE Signals

Signal	**Direction**	**Description**
CD	Output	Carrier Detect
RI	Output	Ring Indicator signal
DTR	Input	Data Terminal Ready signal
DSR	Output	Data Set Ready signal
RTS	Output	Ready to Send signal
CTS	Input	Clear to Send signal

Table 3: DTE Signals

Signal	**Direction**	**Description**
CD	Input	Carrier Detect
RI	Input	Ring Indicator signal
DTR	Output	Data Terminal Ready signal
DSR	Input	Data Set Ready signal
RTS	Output	Ready to Send signal
CTS	Input	Clear to Send signal

The 1SPP firmware provides the ability to control Bluetooth low power settings (while in data mode). These settings, if enabled, utilize the Bluetooth sniff mode, and can drastically reduce the power consumption used for active Bluetooth connections. The trade-off for this reduced power is that the latency is increased. This is a facet of the Bluetooth sniff mode, since the link is only being serviced at regular intervals. Any data that needs to be sent or received during the sniff interval will have to wait until the interval expires before it can be set. The 1SPP firmware allows the configuration of this interval so that the host processor can determine the best latency/power consumption tradeoff (see the **AT+BTPWR** command more information).

In addition to Bluetooth low power settings (while in data mode), the 1SPP firmware allows the ability to reduce power consumption while in command mode. This mode is known as deep sleep mode and can be entered from command mode by using the **AT+BTSLP** command. Deep sleep mode allows the 1SPP firmware to reduce the power consumption of the chip to micro-Amp (uA) levels when no functionality is required of the device. The device can be woken from deep sleep mode by utilizing either DTR/DSR or using the RESET_N line of the BGB203. When the device is woken from deep sleep mode it retains all prior configuration information (unless the chip is woken by the RESET_N line in which case the chip is reset completely). This functionality allows the host to reduce the power consumption of the 1SPP device during periods of inactivity (i.e. the host does not require any services of the 1SPP device).

When the 1SPP device is operating in command mode (i.e. it is accepting commands) other Bluetooth functionality is available. This functionality allows the local device the ability to configure local settings as well as the ability to interactively discover other Bluetooth devices and the services that those devices support. See the 1SPP command set documentation for a detailed description of the features that are provided in this category.

In addition to the above device configuration and Bluetooth functionality, the 1SPP firmware provides features to aid in device qualification and testing (both Bluetooth and FCC). The 1SPP command set documentation contains a detailed description of these commands. The commands that are useful for this purpose are the **AT+BTTST**, **AT+BTTX**, and **AT+BTRXT** commands.

The 1SPP firmware provides the ability to save any currently configured settings to Flash so that those settings will become the default settings when the device is reset. This ability also allows the device to automatically boot into automatic connect mode (if so configured) which forces the device to boot into data mode (instead of the default mode which is command mode).

Readers that are familiar with other Philips Semiconductor Bluetooth products may be aware that there are two primary mechanisms with which to configure the radio settings of the device:

- Factory settings
- Dynamic settings

The configuration settings supported by the above allow options such as Bluetooth Address and RF tuning parameters to be set. The 1SPP firmware supports both configuration methods. Similar to other Philips Semiconductor Bluetooth products, the Dynamic settings configuration always overrides the Factory settings configuration. The Dynamic settings are the preferred mechanism for setting parameters because no external applications are required to configure the device (i.e. no further tools are required to build the Factory settings configuration and no further tools are required to program this information on the 1SPP device). The configuration of dynamic settings is done via the **AT+BTSET** command. The options that can be modified are listed below (with a reference to the Factory Settings option that is the equivalent setting). See the **AT+BTSET** command for more information.

Table 4: Dynamic Settings

Setting	**Factory Setting Name**	**Description**
Bluetooth Address	BD Address	Bluetooth Device Address
RXTUN	RXTUN	Radio RXTUN value
Reference Voltage	Reference Voltage	Reference Voltage
Class 2 Trimming	Fine Tuning	Used to ensure trim Class 2 power such that output does not exceed +4dBm
Course Tuning	Course Tuning	Course Tuning

4. Command Set Overview

The 1SPP firmware follows the AT command set convention of the V.25 command set and provides extensions that follow the conventions of the GSM command set. The following subset of V.25 commands is supported:

Table 5: V.25 Commands

Command	Description
I	Information- returns product information
E	Echo – changes echoing of characters when in command mode
Z	Reset to default – changes configuration back to the stored Flash settings
&F	Reset to factory defaults – changes configuration to factory defaults (set with Factory Settings Tool)

The following commands are command extensions (following GSM command conventions) that have been added to support the product functionality required:

Table 6: Extension Commands

Command	Description
+BTFLS	Store settings to Flash – writes current configuration to Flash
+BTRNM	Remote Name – query a remote Bluetooth Device's name
+BTSDP	SDP Query – query a remote Bluetooth Device's services
+BTTST	Test Mode – configure the local device for Bluetooth test mode
+BTRXT	RXTUN – calibrate RXTUN value (and output clock)
+BTTX	Transmission Test – perform FCC/Bluetooth transmission test
+BTCOD	Class of Device – read/write local Bluetooth Class of Device
+BTLNM	Local Name – read/write local Bluetooth device name
+BTBDA	Bluetooth Address – read local Bluetooth device address
+BTINQ	Inquiry – perform Bluetooth Inquiry (discover devices in range)
+BTAUT	Automatic Connection Mode – configure automatic connection settings
+BTCFG	Config – read/write local configuration options
+BTESC	Escape Sequence – read/write escape sequence used to enter command mode
+BTLSV	Link Supervision – read/write Bluetooth link supervision timeout that is used for active connections
+BTPWR	Power – configure Bluetooth low power handling
+BTSLP	Sleep – request device to enter deep sleep mode (very low power)
+BTURT	UART – read/write current UART configuration settings
+BTSRV	Server – configure device to accept SPP connections (incoming)
+BTCLT	Client – configure device to attempt SPP connections (outgoing)
+BTSEC	Security – read/write current security settings (authentication and encryption) required for connections
+BTPIN	PIN Code – read/write PIN code that is used for authentication and pairing
+BTPAR	Pairing – manage pairing operations (Link Key maintenance and accept and initiate pairing procedures)
+BTCAN	Cancel – cancels any currently outstanding command (e.g. Remote Name, SDP Query, Test Mode, Inquiry, or Pairing operation)

+BTSET	Dynamic Settings – override Factory setting with specified Dynamic setting (for example Bluetooth device address)

Each command will further be explained (and demonstrated) below in this document.

5. General Command Syntax

All commands listed in this document must be preceded with the string "**AT**". All commands listed in this document end when a carriage return character (ASCII character value 13 - hexadecimal 0x0D) is received. An entire command sequence will have the following format:

<Prefix><Command><CR>

where,

<Prefix> is the case-insensitive string "**AT**" (without the quotation marks),
<Command> is a command listed in the command section of this document, and
<CR> is the carriage return character (ASCII character value 13 - hexadecimal 0x0D)

Once a command is received by the device it will be processed. Once a command is processed, a response will be generated. A response will have the following format:

<CR><LF><Response><CR><LF>

where,

<CR> is the carriage return character (ASCII character value 13 - hexadecimal 0x0D),
<LF> is the line feed character (ASCII character 10 - hexadecimal 0x0A),
<Response> is the command specific response, or one of the following strings (without the quotation marks):
"**OK**"
"**ERROR**"
<CR> is the carriage return character (ASCII character value 13 – hexadecimal 0x0D),
<LF> is the line feed character (ASCII character 10 - hexadecimal 0x0A)

If the command is an extended command, it will return each line of the result in the following format:

<CR><LF><Command><Delimiter><Response Data><CR><LF>

where,

<CR> is the carriage return character (ASCII character value 13 - hexadecimal 0x0D),
<LF> is the line feed character (ASCII character 10 - hexadecimal 0x0A),
<Command> is the command itself (without the "**AT**" prefix)
<Delimiter> is the string "**:** "(without the quotation marks). Note the single space character in the above string.
<Response Data> is the command specific response data
<CR> is the carriage return character (ASCII character value 13 - hexadecimal 0x0D),
,
<LF> is the line feed character (ASCII character 10 - hexadecimal 0x0A),

Date of release:21 Dec 2005

Published in The Netherlands

In all cases, a command will end in one of the following two strings (to signify that the command has been fully processed):

<CR><LF>**OK**<CR><LF>
<CR><LF>**ERROR**<CR><LF>

The string "**OK**" signifies that the command processing has completed successfully, while the string "**ERROR**" indicates that there was an error processing the command.

To further illustrate the above, some examples will be given:

Example 1:

```
ATI
1SPP - Ver: 1.1
OK
```

The actual input would be:

ATI<CR>

The actual output would be:

<CR><LF>1SPP – Ver: 1.1<CR><LF>
OK<CR><LF>

Example 2:

```
ATZ
OK
```

The actual input would be:

ATZ<CR>

The actual output would be:

<CR><LF>OK<CR><LF>

Example 3:

```
AT+BTCFG
+BTCFG: 0
OK
```

The actual input would be:

AT+BTCFG<CR>

The actual output would be:

<CR><LF>+BTCFG: 0<CR><LF>
<CR><LF>OK<CR><LF>

Date of release:21 Dec 2005

Published in The Netherlands

6. V.25 Commands

The following section describes the V.25 commands that are supported. Each command will be described in detail below. This detail will include the command format, command parameters and command responses. Each command also will include examples to show how the command can be used.

I – Information

Description:

This command returns the information identification string about the current product.

Format:

ATI

Results:

Result will be one (or more) lines of product information, followed by the result code (either **OK** or **ERROR**).

Example:

```
ATI
1SPP - Ver: 1.1
OK
```

E – Echo

Description:

This command configures the device to either echo received characters in command mode (or not to echo received characters in command mode).

Format:

ATE<X>

where,

<X> is a single ASCII Digit:

0 – disable echo

1 – enable echo

Results:

Result will be the result code

OK

ERROR

It should be noted that command responses will still be generated when commands are processed (when echo is disabled).

Example:

```
ATE0
OK
```

Z – Reset configuration to default

Description:

This command is used to restore the current configuration settings back to internal default values (not values that are stored into Flash and/or Factory Settings).

Format:

ATZ

Results:

Result will be the result code (either **OK** or **ERROR**).

Example:

```
ATZ
OK
```

&F – Restore configuration settings set by Factory settings tool or stored to Flash

Description:

This command is used to restore the current configuration settings back to the settings that were stored by the Factory Settings tool (or settings that were stored to Flash). If no settings have been stored to Flash, then the settings that were written with the Factory Settings tool will be used.

Format:

AT&F

Results:

Result will be the result code (either **OK** or **ERROR**).

Example:

```
AT&F
OK
```

7. Extension Commands

The following section will describe the extended commands that are supported by the product. These commands will follow the GSM command set format conventions (as previously described). Each command will be described in detail below. This detail will include the command format, command parameters and command responses. Each command also will include examples to show how the command can be used.

+BTFLS – store current configuration to Flash

Description:

This command is used to store the current configuration settings to Flash memory. This forces the current settings to become the settings that are used whenever the device is reset.

Format:

AT+BTFLS

Results:

Result will be the result code (either **OK** or **ERROR**).

Example:

```
AT+BTFLS
OK
```

BTRNM – perform remote name discovery

Description:

This command is used to query a remote Bluetooth device's local name (not the local Bluetooth device's local name).

Format:

AT+BTRNM=<BDADDR>

where,

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device to discover the remote name. This value is required to be 12 digits (note that this value is not preceded with the "**\x**" or the "**\X**" prefix).

Results:

Result will be a single line of one of the following formats:

+BTRNM: <BDADDR>, "<Remote Name>"

where,

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device. This value will be 12 digits (note that this value is not preceded with the "**\x**" or the "**\X**" prefix).

<Remote Name> is the remote Bluetooth device name that was returned from the remote device. The name will be enclosed within quotation marks.

if a remote name was retrieved (note that the returned remote device name will be enclosed in quotation marks), or

+BTRNM: <BDADDR>, NO RESPONSE

Date of release:21 Dec 2005

Published in The Netherlands

if no response was retrieved from the remote device. Following the above output, the response will be the result code (either **OK** or **ERROR**).

Example:

```
  AT+BTRNM=0123456789AB
  +BTRNM: 0123456789AB, "BGB203 - 1SPP"
  OK

  AT+BTRNM=123456789ABC
  +BTRNM: 0123456789ABC, NO RESPONSE
OK
```

+BTSDP – perform remote service discovery

Description:

This command is used to query a remote Bluetooth device's local services (not the local Bluetooth device's local services) using the Service Discovery Protocol (SDP). This function only returns services that utilize the RFCOMM protocol (i.e. it does not return services that use L2CAP as the transport protocol).

Format:

AT+BTSDP=<BDADDR>, <Profile Filter>
where,

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device to discover services. This value is required to be 12 digits (note that this value is not preceded with the "**\x**" or the "**\X**" prefix).

<Profile Filter> is an optional integer value that represents the profiles that should be searched for. If this parameter is not present then all services will be returned. The values for the profile filter are given in the following table (note that multiple profiles can be searched for by logically OR'ing the values together (or by integer adding the values together)).

Table 7: Profile Filter

Value	Profile
0x00000001	Serial Port profile
0x00000002	Headset profile
0x00000004	Dial-up Networking profile
0x00000008	Fax profile
0x00000010	LAN Access profile
0x00000020	OBEX Object Push profile
0x00000040	OBEX File Transfer profile
0x00000080	OBEX Object Synchronization profile
0x00000100	Hands Free profile
0x00000200	SIM Access profile
0x00000400	Basic Printing profile
0x00000800	Basic Imaging profile
0x00001000	Video Distribution profile
0x00002000	Phonebook Access profile

The command will accept hexadecimal input (as listed above) by pre-appending "**\x**" or "**\X**" (without the quotation marks) to the value. If this prefix is not present, then the value is assumed to be an integer value (base 10).

Results:

Result will be one (or more lines) of one of the following formats:

+BTSDP: <Service Type>, <Profile Version>, "<Service Name>", <Port>

where,

<Service Type> represents the type of service. It will be one of the following values listed in the table below (note that these values do not match the profile filter values).

Table 8: Service Type

Value	Service
0	Unknown
1	Serial Port
2	Headset Audio Gateway
3	Headset
4	Dial-up Networking
5	Fax profile
6	LAN Access
7	OBEX Object Push
8	OBEX File Transfer
9	OBEX Object Synchronization
10	Hands Free Audio Gateway
11	Hands Free
12	SIM Access
13	Basic Printing Referenced Objects
14	Basic Printing Reflected User Interface
15	Basic Printing Status
16	Basic Imaging Responder
17	Basic Imaging Referenced Objects
18	Basic Imaging Archive
19	Video Distribution Sink
20	Video Distribution Source
21	Phonebook Access Server
22	Phonebook Access Client

<Profile Version> is the version of the profile that was reported. It will have the form of <Major Version>.<Minor Version> (e.g. "**1.0**" without the quotation marks).

<Service Name> is the name of the service that was returned. This value will be enclosed in quotation marks.

<Port> is the actual RFCOMM server port that the service is

located. This value will need to be specified if a connection is to be made to the specified service.

When the SDP search is complete, the result will have the following format:

+BTSDP: COMPLETE

if there was an error retrieving the SDP information, then the result will have the following format:

+BTSDP: ERROR

if no response was retrieved from the remote device (or an error occurred retrieving the service from the remote device).

It should be noted that the last result that will be output from the SDP search command will be one of the following:

+BTSDP: COMPLETE
+BTSDP: ERROR

Following any of the above output, the response will be the result code (either **OK** or **ERROR**).

Example:

```
AT+BTSDP=0123456789AB
+BTSDP: 1, 1.0, "Serial Port", 2
+BTSDP: 6, 1.0, "Lan Access Using PPP", 4
+BTSDP: 7, 1.0, "OBEX Object Push", 3
+BTSDP: 8, 1.0, "OBEX File Transfer", 3
+BTSDP: COMPLETE
OK

AT+BTSDP=0123456789AB, \x1
+BTSDP: 1, 1.0, "Serial Port", 2
+BTSDP: COMPLETE
OK

AT+BTSDP=0123456789AB, 8
+BTSDP: COMPLETE
OK

AT+BTSDP=123456789ABC
+BTSDP: ERROR
OK
```

+BTTST – configure the local device for Bluetooth test mode

Description:

This command is controls whether or not the local Bluetooth device enables device under test mode. When this mode is enabled, the local Bluetooth device responds to LMP requests (over the air) for Bluetooth testing. Once test mode is entered, most other commands will return error until the mode is exited.

Format:

AT+BTTST=<X>
where,

<X> is a single ASCII Digit:

0 – disable Bluetooth test mode
1 – enable Bluetooth test mode

or one of the following to query the current Bluetooth test mode value:

AT+BTTST?
AT+BTTST

Results:

If a query is performed, the output will be the following format:
+BTTST: <X>

where,

<X> is a single ASCII Digit:

0 – disable Bluetooth test mode
1 – enable Bluetooth test mode

Following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the test mode was set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTTST?
BTTST: 0
OK

AT+BTTST=1
OK
```

+BTRXT – calibrate RXTUN value (and output EXT_CLK signal)

Description:

This command is used to for debugging purposes to calibrate the RXTUN value. This command outputs the EXT_CLK signal on GPIO2 and optionally allows the user to specify a new RXTUN value to use. This command does not store the RXTUN value into Flash. When the system is reset, it will read the value that was set with the Factory

Settings tool. Once the output of EXT_CLK signal is enabled, most other commands will return error until the output of the EXT_CLK signal is disabled.

It should also be noted that UART Flow Control (RTS/CTS) cannot be enabled when enabling the output of EXT_CLK due to the multiplexing of the GPIO2 pin. If UART Flow Control (RTS/CTS) is enabled and this mode is enabled, an error will be returned.

Finally, once the correct RXTUN value is determined, it needs to be set programmed into the Factory settings information, or programmed dynamically with the Dynamic settings command (**AT+BTSET**). Until the RXTUN value is programmed into the Factory or Dynamic settings it will not be used by the 1SPP firmware for Bluetooth operation (once this command is complete).

Format:

AT+BTRXT=<X>, <Y>

where,

<X> is a single ASCII digit:

0 – disable output EXT_CLK signal on GPIO2
1 – enable disable output EXT_CLK signal on GPIO2

<Y> is either an integer value (0 -255) or a hexadecimal value (0x00 – 0xFF) which represents the RXTUN value to use. This parameter is optional when enabling the output of the EXT_CLK and is ignored when disabling the output of the EXT_CLK

or one of the following to query the current RXTUN calibration mode value:

AT+BTRXT?
AT+BTRXT

Results:

If a query is performed, the output will be the following format:

+BTRXT: <X>

where,

<X> is a single ASCII Digit:

0 – output of EXT_CLK signal on GPIO2 is disabled
1 – output of EXT_CLK signal on GPIO2 is enabled

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the test mode was set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTRXT?
+BTRXT: 0
OK

AT+BTRXT=1, \xA0
OK
```

+BTTX – perform FCC/Bluetooth transmission test

Description:

This command is used to for testing purposes to force the radio to continuously transmit in order to measure the TX spectrum. After enabling this mode the Bluetooth radio will begin to transmit packets (without whitening) according to the specified parameters. Once this mode is enabled, most other commands will return error until this mode is disabled. While in this mode, however, the command can be issued again to change the transmission parameters.

Format:

AT+BTTX=<X>, <RX On Start>, <Synt On Start>,<TX On Start>, <Phd Off Start>, <Test Scenario>, <Hopping Mode>, <TX Frequency>, <RX Frequency>, <TX Test Interval>, <Test Packet Type>, <Length of Test Data>

where,

<X> is a single ASCII digit:

0 – disable transmission test
1 – enable transmission test

<RX On Start> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF) of the following values (represented as binary bits below):

1xxxxxxx – No change
0xxxxxxx – Use timing as indicated in bits 0-6. Units are in 2us.

<Synt On Start> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF) of the following values (represented as binary bits below):

1xxxxxxx – No change
0xxxxxxx – Use timing as indicated in bits 0-6. Units are in 2us.

<TX On Start> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00

– 0xFF) of the following values (represented as binary bits below):

1xxxxxxx – No change
0xxxxxxx – Use timing as indicated in bits 0-6. Units are in 2us.

<Phd Off Start> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0x5D) of the following values (represented as binary bits below):

1xxxxxxx – No change
0xxxxxxx – Use timing as indicated in bits 0-6. Units are in 2us.

<Test Scenario> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF) that specifies the test scenario (see the Bluetooth Test Mode chapter of the Bluetooth specification for more information on this parameter).

<Hopping Mode> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF) that specifies the hopping mode (see the Bluetooth Test Mode chapter of the Bluetooth specification for more information on this parameter).

<TX Frequency> is an integer value (0-93) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00-0x5D) that specifies the transmission frequency. This value is specified in MHz (based on the offset of 2402 MHz – e.g. 0 represents 2402 MHz).

<RX Frequency> is an integer value (0-93) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0x5D) that specifies the receive frequency. This value is specified in MHz (based on the offset of 2402 MHz – e.g. 0 represents 2402 MHz).

<TX Test Interval> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF) that specifies the number of empty frames between subsequent transmissions.

<Test Packet Type> is an integer value (0-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF) that specifies the packet type that will be used (see the Bluetooth Test Mode chapter of the Bluetooth specification for more information on this parameter). Note that only DH1, DH3,

and DH5 values are allowed.

<Length of Test Data> is an integer value (0-65535) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x0000 – 0xFFFF) that specifies the length of the test data that will be used (see the Bluetooth Test Mode chapter of the Bluetooth specification for more information on this parameter).

The reader should note that none of the above parameters are optional (when enabling the transmission test), and if the test is being disabled, then all of the other parameters are ignored.

or one of the following to query the current Transmission test value:

AT+BTTX?
AT+BTTX

Results:

If a query is performed, the output will be the following format:

+BTTX: <X>

where,

<X> is a single ASCII Digit:

0 – transmission test is not currently in progress
1 – transmission test is currently in progress

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the test mode was set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTTX?
+BTTX: 0
OK

AT+BTTX=1,128,128,128,128,4,0,1,2,3,4,10
OK

AT+BTTX=0
OK
```

+BTCOD – read/write local Bluetooth Class of Device

Description:

This command is used to query (or set) the Bluetooth Class of Device value for the local Bluetooth device. Note that if the Class of Device is set that it will not be saved to Flash (unless the save to Flash command is used afterward).

Format:

AT+BTCOD=<COD>

where,

<COD> is an integer value (0 – 16777215) of a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x000000 – 0xFFFFFF) which represents the current Bluetooth Class of Device value to write to the device.

or one of the following to query the current Class of Device value:

AT+BTCOD?
AT+BTCOD

Results:

If a query is performed, the output will be the following format:

+BTCOD: <COD>

where,

<COD> is a 6 digit ASCII coded hexadecimal value representing the Class of Device. Note that this value is not preceded with the "**\x**" or "**\X**" prefix.

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the Class of Device was set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTCOD?
+BTCOD: 001F00
OK

AT+BTCOD=\x123456
OK
```

+BTLNM – read/write local Bluetooth device name

Description:

This command is used to query (or set) the Bluetooth device name value for the local Bluetooth device. Note that if the device name is set that it will not be saved to Flash (unless the save to Flash command is used afterward).

Format:

AT+BTLNM="<Local Name>"

where,

<Local Name> represents the ASCII string to use for the local device name. This value cannot be longer than 16 bytes. This string must be enclosed in quotation marks.

or one of the following to query the current local device name:

AT+BTLNM?
AT+BTLNM

Results:

If a query is performed, the output will be the following format:

+BTLNM: "<Local Name>"

where,

<Local Name> is the ASCII string that represents the name of the local Bluetooth device. This value will be enclosed in quotation marks.

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the local name was set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTLNM
+BTLNM: "BGB203 - 1SPP"
OK

AT+BTLNM="New Name"
OK
```

+BTBDA – read local Bluetooth device address

Description:

This command is used to query the Bluetooth device address for the local Bluetooth device. The device address can be set via the Factory settings of the Dynamic settings. See the **AT+BTSET** command for more information on changing the Bluetooth device address.

Format:

AT+BTBDA?

or

AT+BTBDA

Results:

The result will be a single line of the following format:

+BTBDA: <BDADDR>

where,

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the local Bluetooth device (note that this value is not preceded with the "**\x**" or the "**\X**" prefix).

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

Example:

```
AT+BTBDA
+BTBDA: BDB203002231
OK
```

+BTINQ – perform Bluetooth Inquiry (discover devices in range)

Description:

This command is used to discover Bluetooth devices that are currently within radio range (that are discoverable).

Format:

AT+BTINQ=<X>

where,

<X> represents the time (in seconds) that the local Bluetooth device is to perform the inquiry procedure. This duration is specified in seconds and has a range from 2 seconds to 61 seconds. This parameter is required and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x02 – 0x3D).

Results:

Result will be one (or more lines) of one of the following formats:

+BTINQ: <BDADDR>, <COD>

where,

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device that was discovered. Note that this value is not preceded with the "**\x**" or the "**\X**" prefix.

<COD> is a 6 digit ASCII coded hexadecimal value representing the Class of Device. Note that this value is not preceded with the "**\x**" or "**\X**" prefix.

When the Inquiry is complete, the result will have the following format:

+BTINQ: COMPLETE

It should be noted that the last result that will be output from the Inquiry command will be the following:

+BTSDP: COMPLETE

Following any of the above output, the response will be the result code (either **OK** or **ERROR**).

Example:

```
AT+BTINQ=5
+BTINQ: COMPLETE
OK

AT+BTINQ=3
+BTINQ: 000123456789, 200000
+BTINQ: 00123456789A, 920300
+BTINQ: 0123456789AB, 140680
+BTINQ: 123456789ABC, 020300
+BTINQ: COMPLETE
OK
```

+BTAUT – configure automatic connection settings

Description:

This command is used to query (or set) the settings that are used to control automatic connection mode. Automatic connection mode is a more where the device does not return to command mode when a connection is complete (either client or server). When this mode is enabled, no command input will be accepted. The only way to enter command mode is either via escape sequence (if configured) or lowering DTR (if configured to enter command mode). This command also provides the ability to have the device write the setting to Flash so that when the device is reset it will automatically boot into auto connection mode (or boot into non-auto connect mode).

It should be noted that when the auto connect mode setting is requested to be stored to Flash, it will not be physically stored until the user exits command mode by either connecting to a server or becoming a server (see the **AT+BTCLT** and **AT+BTSRV** commands). The deferred save is because the device does not know what type of auto connect is required (either client or server) and also does not know all of the parameters (until they are specified). It should also be noted that the store to Flash parameter is optional and does not need to specified (if the value is not to be stored to Flash).

Format:

AT+BTAUT=<X>, <Y>

where,

<X> is a single ASCII Digit:

0 – disable auto connect mode
1 – enable auto connect mode

<Y> is a single ASCII Digit:

0 – do not store auto connect mode setting to Flash
1 – store auto connect mode setting to Flash

This parameter is optional and is not required (unless it is explicitly required that the setting be stored to Flash).

or one of the following to query the current auto connect value:

AT+BTAUT?
AT+BTAUT

Results:

If a query is performed, the output will be the following format:

+BTAUT: <X>, <Y>

where,

<X> is a single ASCII Digit:

0 – auto connect mode is disabled
1 – auto connect mode is enabled

<Y> is a single ASCII Digit:

0 – auto connect mode setting will not be stored to Flash
1 – auto connect mode setting will be stored to Flash

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the auto connect parameters were set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTAUT?
+BTAUT: 0, 0
OK

AT+BTAUT=1
OK
```

+BTCFG – read/write local configuration options

Description:

This command is used to query (or change) miscellaneous device configuration options. The options that are supported with this command are:

Pass-through CD, RI, DTR, DSR signals

This option specifies whether the modem controls signals (CD, RI, DTR, and DSR) are directly passed through from PIN to Bluetooth. This differs quite a bit from ordinary use where the signals are local signals (and their value doesn't depend on the remote device) and are managed by the local device. An example would be the CD output signal (for a DCE device). This value is asserted when a Bluetooth connection is present, and de-asserted when no connection is present. If the pass-through mode is enabled, the value of this signal depends on what the remote device has set the value of CD to be. This feature is primarily useful when the state of the modem signals of the remote device are needed. Note that DTR/DSR flow control must be enabled in the UART flags to correctly process the DTR/DSR signals.

DTC/DCE emulation

This option specifies whether or not the device is acting as a DCE or DTE device. This option controls how the modem control signals (CD, RI, DTR, DSR) are utilized. The aforementioned signals are processed as listed in the following tables:

Table 9: DCE Device

Signal	Description	I/O Direction
RI	Ring Indicator	Output
CD	Carrier Detect	Output
DTR	Data Terminal Ready	Input
DSR	Data Set Ready	Output

Table 10: DTE Device

Signal	Description	I/O Direction
RI	Ring Indicator	Input
CD	Carrier Detect	Input
DTR	Data Terminal Ready	Output
DSR	Data Set Ready	Input

Note that in all cases, the remaining UART signals (RTS, CTS, TX, and RX) are utilized the same regardless of the type of emulation that is configured (i.e. RX is always an input, TX is always an output, etc.). Also note that DTR/DSR flow control must be enabled in the UART flags to correctly process the DTR/DSR signals (if DTR/DSR is used).

Ignore/process escape sequence
This option enables/disables the ability to utilize a configurable escape sequence to exit from data mode to command mode. If the escape sequence processing is disabled then there is no combination of data bytes that can force a change from data mode to command mode.

Suppress command responses
This option enables/disables command responses from accepted commands in command mode. If command responses are suppressed, then there will be no data received from the device that was not received over the Bluetooth link (this means no OK/ERROR responses as well as no command results).

Configure DTR/DSR to enter command mode
This option can be configured to allow the modem signal DTR/DSR (depending upon if the device is configured as a DCE or DTE) to be used as a transition from data mode to command. This option can be used to enter command if the escape sequence processing is disabled. Note that for this to work correctly, the UART must have DTR/DSR enabled in the Flow control flags.

Enable/disable a GPIO output to signify Bluetooth connection
This option can be used to force the device to output a signal reflecting the current state of the Bluetooth connection (actively connected or not connected). This signal can be tied to an LED or can be monitored externally. This feature is useful in the case where RI, CD, DTR, DSR is configured in the pass-through mode and the connection state is required (since CD cannot be monitored).

Note that if configuration options are set that they will not be saved to Flash (unless the save to Flash command is used afterward).

Format:
AT+BTCFG=<CFG>

where,

<CFG> is an integer value that represents the configuration

options that should be enabled. The values for the configuration flags are given in the following table (note that multiple flags can be enabled by logically OR'ing the values together (or by integer adding the values together)). This parameter is required and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x0000 – 0xFFFF).

Table 11: Configuration Flags

Value	Option
0x0001	Pass-through RI, CD, DTR, DSR
0x0002	DTE device emulation
0x0004	Ignore escape sequence
0x0008	Suppress responses
0x0010	DTR/DSR enter command mode
0x0020	Use connection active GPIO (output)

or one of the following to query the current configuration options

AT+BTCFG?
AT+BTCFG

Results:

If a query is performed, the output will be the following format:

+BTCFG: <CFG>

where,

<CFG> is an integer value that represents the current configuration. See table above for the values that are currently configured.

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the configuration parameters were set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTCFG
+BTCFG: 0
OK

AT+BTCFG=32
```

```
OK
```

+BTESC – read/write escape sequence used to enter command mode

Description:

This command is used to query (or set) the byte sequence that is to be used for the escape sequence. This sequence of bytes represents the byte sequence that will be used to signal to the device that it should leave data mode and enter command mode. Note that if the escape sequence is changed it will not be saved to Flash (unless the save to Flash command is used afterward).

Format:

AT+BTESC=<Length>, <Byte 1>, <Byte 2>, <Byte 3>, <Byte 4>

where,

<Length> is an integer value (1-4) representing the number of bytes that are to be used for the new escape sequence. This parameter is required and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x01 – 0x04). This number represents the number of individual arguments that follow that specify the bytes used for the command sequence (e.g. if this value is 2, then there will be two arguments that follow this, each representing the respective byte of the escape sequence).

<Byte 1> is an integer value (0-255) representing the byte value that is to be the first byte of the escape sequence that is being set. This parameter is always required (since the minimum length of an escape sequence is 1) and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF).

<Byte 2> is an integer value (0-255) representing the byte value that is to be the second byte of the escape sequence that is being set. This parameter is required if the <Length> parameter of this command is 2 (or greater) and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF).

<Byte 3> is an integer value (0-255) representing the byte value that is to be the third byte of the escape sequence that is being set. This parameter is required if the <Length> parameter of this command is 3 (or greater) and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF).

<Byte 4> is an integer value (0-255) representing the byte value that is to be the fourth byte of the escape sequence that is being set. This parameter is required if the <Length> parameter of this

command is 4 and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF).

or one of the following to query the current escape sequence:

AT+BTESC?
AT+BTESC

Results:

If a query is performed, the output will be the following format:

+BTESC: <Length>, <Byte 1>, <Byte 2>, <Byte 3>, <Byte 4>

where,

<Length> is an integer value (base 10) that specifies the current escape sequence length (1-4). This value represents the number of arguments that will follow.

<Byte 1> is an integer value (base 10) that specifies the first byte of the escape sequence. This value will be in the range of 0 – 255 and this parameter will always be present (since the minimum escape sequence length is 1).

<Byte 3> is an integer value (base 10) that specifies the second byte of the escape sequence. This value will be in the range of 0 – 255 and this parameter will only be present if the <Length> parameter is 2 (or greater).

<Byte 3> is an integer value (base 10) that specifies the third byte of the escape sequence. This value will be in the range of 0 – 255 and this parameter will only be present if the <Length> parameter is 3 (or greater).

<Byte 4> is an integer value (base 10) that specifies the fourth byte of the escape sequence. This value will be in the range of 0 – 255 and this parameter will only be present if the <Length> parameter is 4.

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the escape sequence was not changed) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTESC
+BTESC: 3, 43, 43, 43
```

```
OK

AT+BTESC=2, \x2B, \x2B
OK
```

+BTLSV – read/write Bluetooth link supervision timeout

Description:

This command is used to query (or set) the Bluetooth link supervision timeout that is used for an active serial port connection. The link supervision timeout specifies the amount of time that the Bluetooth radio will maintain a connected link when no radio communication is detected. If the Bluetooth radio does not receive a response to Bluetooth link manager protocol messages in the allotted time, the link will be disconnected. This parameter is particularly useful when the local device needs wants to be notified when the remote Bluetooth device (which is connected) leaves radio range.

It should be noted that this timeout is not related to actual serial port data transmitted across the link, but rather internal Bluetooth link manager protocol. This means that the device does NOT have to transmit serial port data to keep the link supervision timeout from expiring. The default link supervision timeout is the Bluetooth default 20 seconds. Note that if the link supervision timeout is changed it will not be saved to Flash (unless the save to Flash command is used afterward).

Format:

AT+BTLSV=<X>

where,

<X> is an integer value that represents the new link supervision timeout. This value is specified in seconds and must be between the range of 2 seconds and 40 seconds (inclusive). This value can either be specified as an integer value (base 10) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x02 – 0x28).

or one of the following to query the current link supervision timeout value:

AT+BTLSV?
AT+BTLSV

Results:

If a query is performed, the output will be the following format:

+BTLSV: <X>

where,

<X> is an integer value that represents the currently configured

Date of release:21 Dec 2005

Published in The Netherlands

link supervision timeout value.

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the link supervision timeout was set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTLSV?
+BTLSV: 20
OK

AT+BTLSV=2
OK
```

+BTPWR – configure Bluetooth low power handling

Description:

This command is used to query (or set) the Bluetooth low power handling for Bluetooth connections. This command allows the enabling/disabling of Bluetooth sniff mode, as well as (optionally) the ability to configure the serial port inactivity timeout as well as the parameters that are used to enter Bluetooth sniff mode. Note that if the power parameters are changed it will not be saved to Flash (unless the save to Flash command is used afterward).

Low power handling is useful to conserve power consumption during an active Bluetooth connection. This mode is implemented by utilizing an inactivity timeout. This timeout specifies the time that must elapse (with no serial port data being transmitted or received) before the lower power mode is entered. Once low power mode is entered, it will exit when data the first data byte is either transmitted or received. The Bluetooth power mode used to implement this power savings is Bluetooth sniff mode. The larger the sniff interval, the larger the power savings, however, as the sniff interval increases, so does the latency of the first data byte transmitted or received (as there is no Bluetooth activity during the sniff interval and the device has to wait until the sniff interval has passed before sending the data). The default values used for the power handling parameters are given in the following table (if not overridden by the command).

Table 12: Default Power Parameters

Parameter	Value	Description
Inactivity Timeout	30000 ms (30000 ms)	Inactivity timeout (in ms) before sniff mode is entered
Minimum Sniff Interval	320 slots (200 ms)	See Bluetooth specification for more information
Maximum Sniff Interval	480 slots (300 ms)	See Bluetooth specification for more information

Sniff Attempt	16 slots (20 ms)	See Bluetooth specification for more information
Sniff Timeout	8 slots (10 ms)	See Bluetooth specification for more information

Format:

AT+BTPWR=<X>, <Inactivity>, <Min Sniff Interval>, <Max Sniff Interval>, <Sniff Attempt>, <Sniff Timeout>

where,

<X> is a single ASCII Digit:

0 – low power handling is disabled
1 – low power handling is enabled

<Inactivity> is an integer value (1000-65535) representing the serial port inactivity timeout (in milliseconds) that must expire before lower power mode (sniff mode) is entered. This parameter is optional and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x03E8 – 0xFFFF). This number represents the number of milliseconds that must elapse with no serial port (either transmitted or received) activity before sniff mode is entered (e.g. if this value is 1000, then if no serial port data is transmitted for received in 1 second, sniff mode will be entered).

<Min Sniff Interval> is an integer value (1-65535) representing the minimum acceptable sniff window interval. This parameter is optional and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x0001 – 0xFFFF). This number represents the minimum number of baseband slots between each sniff period. If this value is specified then the next parameter, the maximum sniff interval must be specified as well. See the Bluetooth specification for more detailed information on this parameter.

<Max Sniff Interval> is an integer value (1-65535) representing the maximum acceptable sniff window interval. This parameter is required if the minimum sniff interval is specified and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x0001 – 0xFFFF). This number represents the maximum number of baseband slots between each sniff period. This parameter must be greater than or equal to the minimum sniff interval. See the Bluetooth specification for more detailed information on this parameter.

<Sniff Attempt> is an integer value (1-32767) representing the number of sniff attempts that are to be attempted (to check for data). This parameter is optional and can either be an integer

value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x0001 – 0x7FFF). This number represents the number of baseband slots that the device will check for received data. See the Bluetooth specification for more detailed information on this parameter.

<Sniff Timeout> is an integer value (0-32767) representing the number of slots to continue to receive data (if present). This parameter is optional and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x0000 – 0x7FFF). This number represents the number of baseband slots that the device will wait to timeout when receiving data. Note that since sniff mode is automatically exited when data is received (or transmitted) the importance of this parameter is minimized. See the Bluetooth specification for more detailed information on this parameter.

or one of the following to query the current power configuration values:

```
AT+BTPWR?
AT+BTPWR
```

Results:

If a query is performed, the output will be the following format:

```
+BTPWR: <X>, <Inactivity>, <Min Sniff Interval>,
<Max Sniff Interval>, <Sniff Attempt>, <Sniff Timeout>
```

where,

<X> is a single ASCII Digit:

0 – low power handling is disabled
1 – low power handling is enabled

Note that if low power handling is disabled then there will not be any further parameters in the result.

<Inactivity> is an integer value (1000-65535) representing the serial port inactivity timeout (in milliseconds) that must expire before lower power mode (sniff mode) is entered. This parameter will be an integer value (base 10) This number represents the number of milliseconds that must elapse with no serial port (either transmitted or received) activity before sniff mode is entered (e.g. if this value is 1000, then if no serial port data is transmitted for received in 1 second, sniff mode will be entered).

<Min Sniff Interval> is an integer value (1-65535) representing the minimum acceptable sniff window interval. This parameter is

Date of release:21 Dec 2005

Published in The Netherlands

will be an integer value (base 10). This number represents the minimum number of baseband slots between each sniff period. See the Bluetooth specification for more detailed information on this parameter.

<Max Sniff Interval> is an integer value (1-65535) representing the maximum acceptable sniff window interval. This parameter will be an integer value (base 10). This number represents the maximum number of baseband slots between each sniff period. This parameter will be greater than or equal to the minimum sniff interval. See the Bluetooth specification for more detailed information on this parameter.

<Sniff Attempt> is an integer value (1-32767) representing the number of sniff attempts that are to be attempted (to check for data). This parameter will be an integer value (base 10). This number represents the number of baseband slots that the device will check for received data. See the Bluetooth specification for more detailed information on this parameter.

<Sniff Timeout> is an integer value (0-32767) representing the number of slots to continue to receive data (if present). This parameter will be an integer value (base 10). This number represents the number of baseband slots that the device will wait to timeout when receiving data. Note that since sniff mode is automatically exited when data is received (or transmitted) the importance of this parameter is minimized. See the Bluetooth specification for more detailed information on this parameter.

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the power configuration parameters were set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTPWR?
+BTPWR: 0
OK

AT+BTPWR=1, 1000, 320, 480, 16, 8
OK

AT+BTPWR
+BTPWR: 1, 1000, 320, 480, 16, 8
OK

     Xxx
```

+BTSLP – request device to enter deep sleep mode (very low power)

Description:

This command is used to force the device to enter a very low power mode (deep sleep). When the device enters deep sleep mode all device activities are suspended. Activities remain suspended until the device is instructed to exit low power mode. Currently the only mechanisms that exist to wake the device involve the assertion of either the DTR or DSR input (depending upon the device configuration - DCE or DTE) or a device reset. When the device exits deep sleep mode all previously configured parameters are maintained (e.g. configuration, UART, etc.).

Low power handling is useful to conserve power consumption during inactive periods (i.e. no 1SPP device services are required). The current consumption during deep sleep mode is on the order of micro-Amps (uA) as opposed to milli-Amps (mA) when in normal operation.

As previously stated, the device can be instructed to exit low power deep sleep mode by one of two methods:

DSR/DTR Assertion

This exit mode is selected by asserting DTR/DSR (whichever is configured for input – based upon DCE or DTE device configuration). These signals are considered asserted when they are low (active low). It should be noted that these input signals (either DTR or DSR) are monitored regardless if hardware flow control has enabled using DTR/DSR (see the **AT+BTURT** command for more information). It is also very important to note that if the correct signal is already asserted when this command is processed, the BGB203 will wait for the signal to de-assert (high) and then re-assert (low) before exiting deep sleep mode. This final note signifies that if the command is issued to the BGB203 and the correct signal is already asserted, it will not be prevented from going into deep sleep mode (or immediately exit deep sleep mode). If the correct signal is de-asserted when the deep sleep mode command is received then the BGB203 simply exits deep sleep mode when the correct signal is asserted (there does not have to be a complete cycle).

Device Reset

This exit mode is selected by either using the RESET_N line (reset active low) or by physically cycling the power to the BGB203.

This command will output either the success response or an error response when it has been processed. The 1SPP firmware will not allow deep sleep mode to be entered if there are any asynchronous Bluetooth actions outstanding (e.g. Inquiry, SDP search, remote name discovery, pairing, etc.) or if any test or configuration modes have been entered (e.g. radio calibration, Bluetooth test mode, transmission test mode, etc.). In all cases either an **OK** or **ERROR** response will be returned when this command is received by the BGB203. If the command returns a successful response (**OK**), deep sleep mode will be entered immediately after the response is output from the UART.

Format:

AT+BTSLP

Results:

The result will be the result code (either **OK** or **ERROR**). If the result is success, immediately after the response is output the device will enter deep sleep mode.

Example:

```
AT+BTSLP
OK
```

+BTURT – read/write current UART configuration settings

Description:

This command is used to query (or set) the UART parameters that are used to communicate with the local host. This parameter allows the ability to set the baud rate, data bits, parity, stop bits, and flow control parameters. Note that if the UART parameters are changed it will not be saved to Flash (unless the save to Flash command is used afterward).

It should be noted that if the UART configuration is changed, the result response (**OK** or **ERROR**) will be returned with the original UART configuration parameters. This means that if the baud rate is originally 115,200 and this command is issued to change the baud rate to 57,600 (and no other changes are requested), and the change is successful, the **OK** response will be returned at the original baud rate of 115,200.

Format:

AT+BTURT=<Baud>, <Data Bits>, <Parity>, <Stop Bits>, <Flow Control>

where,

<Baud> is an integer value that represents the baud rate that is to be used. This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00000000 – 0xFFFFFFFF). This value is simply the numerical baud rate (e.g. not a mapping). The supported baud rates are listed in the following table.

Table 13: UART Baud Rates

Value	Description
9600	9,600 bps
14400	14,400 bps
19200	19,200 bps
38400	38,400 bps
57600	57,600 bps
115200	115,200 bps
230400	230,400 bps
460800	460,800 bps
921600	921,600 bps
1000000	1,000,000 bps

<Data Bits> is an integer value that represents the number of data bits that are to be used (word length). This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x05 – 0x08). This value is simply the numerical word length (e.g. not a mapping). The supported word lengths are listed in the following table.

Table 14: UART Word Lengths

Value	Description
5	5 data bits
6	6 data bits
7	7 data bits
8	8 data bits

<Parity> is an integer value that represents the UART parity that should be used. The values for the UART parity are given in the following table. This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0x05).

Date of release:21 Dec 2005

Published in The Netherlands

Table 15: UART Parity	
Value	**Description**
0x0000	No parity
0x0001	Odd parity
0x0002	Even parity
0x0003	Mark parity
0x0004	Space parity

<Stop Bits> is an integer value that represents the number of UART stop bits that are to be used. This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x01 – 0x02). This value is simply the numerical number of stop bits (e.g. not a mapping). The supported stop bits are listed in the following table.

Table 16: UART Stop Bits	
Value	**Description**
1	1 stop bit
2	2 stop bits

<Flow Control> is an integer value that represents the flow control options that should be enabled. The values for the flow control flags are given in the following table (note that multiple flags can be enabled by logically OR'ing the values together (or by integer adding the values together)). This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF).

Table 17: UART Flow Control	
Value	**Description**
0x0001	RTS/CTS flow control
0x0002	DTR/DSR flow control

or one of the following to query the current UART configuration values:

AT+BTURT?
AT+BTURT

<u>**Results:**</u>

If a query is performed, the output will be the following format:

+BTURT: <Baud>, <Data Bits>, <Parity>, <Stop Bits>,<Flow

Control>

where,

<Baud> is an integer value that represents the baud rate that is to be used. This parameter will be an integer value (base 10). This value is simply the numerical baud rate (e.g. not a mapping). The supported baud rates are listed in the following table.

Table 18: UART Baud Rates

Value	Description
9600	9,600 bps
14400	14,400 bps
19200	19,200 bps
38400	38,400 bps
57600	57,600 bps
115200	115,200 bps
230400	230,400 bps
460800	460,800 bps
921600	921,600 bps
1000000	1,000,000 bps

<Data Bits> is an integer value that represents the number of data bits that are to be used (word length). This parameter will be an integer value (base 10). This value is simply the numerical word length (e.g. not a mapping). The supported word lengths are listed in the following table.

Table 19: UART Word Lengths

Value	Description
5	5 data bits
6	6 data bits
7	7 data bits
8	8 data bits

<Parity> is an integer value that represents the UART parity that should be used. The values for the UART parity are given in the following table. This parameter will be an integer value (base

10).

Table 20: UART Parity	
Value	**Description**
0x0000	No parity
0x0001	Odd parity
0x0002	Even parity
0x0003	Mark parity
0x0004	Space parity

<Stop Bits> is an integer value that represents the number of UART stop bits that are to be used. This parameter will be an integer value (base 10). This value is simply the numerical number of stop bits (e.g. not a mapping). The supported stop bits are listed in the following table.

Table 21: UART Stop Bits	
Value	**Description**
1	1 stop bit
2	2 stop bits

<Flow Control> is an integer value that represents the flow control options that should be enabled. The values for the flow control flags are given in the following table (note that multiple flags can be enabled by logically OR'ing the values together (or by integer adding the values together)). This parameter will be an integer value (base 10).

Table 22: UART Flow Control	
Value	**Description**
0x0001	RTS/CTS flow control
0x0002	DTR/DSR flow control

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the UART configuration parameters were set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

Date of release:21 Dec 2005

Published in The Netherlands

```
AT+BTURT?
+BTURT: 115200, 8, 0, 1, 0
OK

AT+BTURT=14400,8,0,1,2
OK
```

+BTSRV – configure device to accept SPP connections (incoming)

Description:

This command is used to configure the local device as a Bluetooth Serial Port Profile (SPP) server. If this command is successful, a SPP server is created, a Service Discovery Protocol (SDP) record is added to the SDP database, the device is set into connectable mode (other devices can connect to the device) and (optionally) discoverable mode. Finally, the device transitions from command mode to data mode and awaits a remote Bluetooth connection to occur. At any time, the device can be made to transition back to command mode by either receiving the configured escape sequence or having DTR/DSR de-asserted (if configured). If automatic mode is not configured, the device will automatically return to command mode when the Bluetooth device is disconnected (after it has been connected of course). If the device is configured to be in automatic mode, then the device stays in data mode and awaits another incoming Bluetooth connection.

It should be noted that if device responses have not been suppressed (see the **AT+BTCFG** command) AND the device is not in automatic mode then the device will inform the local host when a connection has been made (and what Bluetooth device made the connection) as well as when the device disconnects. If automatic mode is being used, then there will be no feedback via the UART that a device is connected or disconnected. If feedback is required then either the modem CD signal or the active connection GPIO should be monitored.

If automatic mode has been enabled prior to the issuance of this command AND the store to Flash setting has been enabled then special action is taken (in addition to the actions explained above). The special action involves saving the currently configured device information to Flash. This will allow the possibility for the device to automatically boot into this mode when restarted. This is useful if the host does not want to reconfigure the device or further interact with the device. It should be noted that when this happens, the device will boot directly into data mode, and the host will have to perform one of the methods listed above to re-enter command mode. If the store to Flash setting has not been enabled for the automatic mode setting then this special processing does not occur.

Format:

AT+BTSRV=<Port>, "<Service Name>", <ACL BDADDR>, <Flags>,

where,

<Port> is an integer value that represents the Bluetooth RFCOMM port number that is to be used (1-30). This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x01 – 0x1E). This value is required. Unless there is a specific reason not to do so, using the port number value 1 will be sufficient for almost all cases. The ability to change the port number is only given for legacy remote devices that have hard-coded remote RFCOMM server port connection information. Utilizing this feature allows the port number to be changed to match whatever port that might be hard-coded. It should be noted that whatever value is chosen for the server port, it will be populated in the Service Discovery Protocol (SDP) server record that will be added to the device when the server is created. Thus, remote devices can query the services of the local device and determine the correct RFCOMM port in which to connect.

<Service Name> is an optional name that can specified for the Service Discovery Protocol (SDP) record that is populated in the local devices SDP database. This service name will be viewable to other devices that perform a SDP search on the local device. If this parameter is not specified, it will default to the string, "Serial Port" (without the quotation marks). This parameter is optional and cannot be longer than 16 bytes. Note that the service name must be enclosed in quotation marks (even though the quotation marks are not stored in the SDP record).

<ACL BDADDR> is an optional parameter that allows the ability to specify only a single, specific, Bluetooth device that can connect to the local device. This parameter is optional and is a 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device (note that this value is not preceded with the "**\x**" or the "**\X**" prefix). This provides the ability for the local device to only serve the serial port service to a specific device, implementing very simplistic Access Control List (ACL) ability. A special value of all zero's (12 ASCII digits of '0') signifies that any Bluetooth device is allowed to connect to the service.

<Flags> is an integer value that represents additional options that should that should be enabled. The values for the flags parameter are given in the following table (note that multiple flags can be enabled by logically OR'ing the values together (or by integer adding the values together)). This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF).

Table 23: Server Flags

Value	Description
0x00	No flags specified
0x01	Device is non-discoverable (device is only connectable)

Results:

Result will be the result code (either **OK** or **ERROR**). It should be noted that if the command is successful the device will enter data mode, otherwise the device remains in command mode.

Example:

```
AT+BTSRV=1
OK

AT+BTSRV=1, "My Service", 000000000000, 1
OK
```

+BTCLT – configure device to attempt SPP connections (outgoing)

Description:

This command is used to configure the local device as a Bluetooth Serial Port Profile (SPP) client. If this command is successful, a connection to the specified remote Bluetooth device's serial port server is attempted. Immediately, the device transitions from command mode to data mode and awaits the Bluetooth connection to complete. At any time, the device can be made to transition back to command mode by either receiving the configured escape sequence or having DTR/DSR de-asserted (if configured). If automatic mode is not configured, the device will automatically return to command mode when the Bluetooth device is disconnected (after it has been connected of course). If the device is configured to be in automatic mode, then the device stays in data mode and awaits attempts another connection. This command allows the ability to specify some additional connection parameters such as the number of attempts, and a period value which can be used to enable period connection attempts.

It should be noted that if device responses have not been suppressed (see the **AT+BTCFG** command) AND the device is not in automatic mode then the device will inform the local host when a connection has been made (and the Bluetooth device that the connection was made with) as well as when the device disconnects. If automatic mode is being used, then there will be no feedback via the UART that a device is connected or disconnected. If feedback is required then either the modem CD signal or the active connection GPIO should be monitored.

If automatic mode has been enabled prior to the issuance of this command AND the store to Flash setting has been enabled then special action is taken (in addition to the actions explained above). The special action involves saving the currently configured device information to Flash. This will allow the possibility for the device to automatically boot into this mode when restarted. This is useful if the host does not want to reconfigure the device or further interact with the device. It should be noted that when this happens, the device will boot directly into data mode, and the host will have to perform one of the methods listed above to re-enter command mode. If the store to Flash setting has not been enabled for the automatic mode setting then this special processing does not occur.

The final (optional) parameter, the connection period, allows a mechanism for the device to conserve power. Power is conserved because the device can be told to delay (sleep) for a specified duration when attempting to connect to the remote device (if the prior attempt failed). This is only useful in automatic mode where the device will not return to command mode and will always attempt to reconnect to a device when it is disconnected. By specifying a non-zero period (in seconds) the device can wait for the specified duration and re-attempt a connection. If this period is zero, the device will continually attempt to re-connect when a connection attempt fails. When power consumption should be as low possible, the continuous connection attempt scenario is not very optimal. When the device is not in automatic mode, the connection period is ignored (only the connection attempt parameter is used).

Format:

AT+BTCLT=<BDADDR>, <Port>, <Attempts>, <Period>

where,

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device to connect with. This value is required to be 12 digits (note that this value is not preceded with the "**\x**" or the "**\X**" prefix).

<Port> is an integer value that represents the Bluetooth RFCOMM port number that is to be connected (1-30). This parameter can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x01 – 0x1E). This value is required. This value can be determined (if not known in advance) by performing a Service Discovery Protocol (SDP) search. See the **AT+BTSDP** command for more information.

<Attempts> is an optional integer parameter that specifies the number of connection attempts that are to be attempted before either returning to command mode or sleeping for the specified period. This value can be either an integer value (1-255) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation

marks 0x01 – 0xFF). The default number of attempts (if not specified) is 1.

<Period> is an optional parameter that allows the ability to specify a timeout period (in seconds) for the device to wait before attempting to connect to the remote device. This timeout is only used if the prior attempt was not successful. If this parameter is zero then no timeout period occurs and the device will automatically attempt to connect if automatic mode is enabled, or will return to command mode if automatic mode is not enabled. This value can be either an integer value (0-65535) or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x0000 – 0xFFFF). The default period (if not specified) is 0. This parameter is only applicable if automatic mode is enabled. If automatic mode is disabled, then this parameter is ignored.

Results:

Result will be the result code (either **OK** or **ERROR**). It should be noted that if the command is successful the device will enter data mode, otherwise the device remains in command mode.

Example:

```
AT+BTCLT=0123456789ABCDEF, 1
OK

AT+BTCLT=0123456789ABCDEF, 1, 5, 120
OK
```

+BTSEC – read/write current security settings (authentication/encryption)

Description:

This command is used to query (or change) the authentication and encryption settings that are required for Bluetooth connections (either incoming or outgoing). The values that can be configured are as follows:

Security Not Required
This option specifies that the local device will not attempt to perform any Bluetooth authentication or encryption when connecting to a remote device or being connected to by a remote device. This does not mean that security will not be supported (i.e. the remote device could request it), it only means that security procedures will not be initiated by the local device.

Security Required
This option specifies that the local device will force Bluetooth authentication when either connecting to a remote device or being connected to by a remote device.

Encryption Required
This option specifies that the local device will force Bluetooth

authentication (and then encryption) when either connecting to a remote device or being connected to by a remote device. Authentication is performed first because Bluetooth requires this. Choosing this value forces all data that is sent over the physical Bluetooth ACL link to be encrypted.

This command is not responsible for configuring link keys and/or PIN codes. Please see the **AT+BTPAR** and **AT+BTPIN** commands more information about link key and PIN code maintenance. Note that if configuration options are set that they will not be saved to Flash (unless the save to Flash command is used afterward).

Format:

AT+BTSEC=<SEC>

where,

<SEC> is an integer value that represents the security options that should be enabled. The values for the configuration flags are given in the following table. This parameter is required and can either be an integer value or a hexadecimal value (preceded by "**\x**" or "**\X**" without the quotation marks 0x00 – 0xFF).

Table 24: Security Options

Value	**Description**
0x00	No security required (Security None)
0x01	Authentication required (Security Required)
0x02	Encryption required (Encryption Required)

or one of the following to query the current security options

AT+BTSEC?
AT+BTSEC

Results:

If a query is performed, the output will be the following format:

+BTSEC: <SEC>

where,

<SEC> is an integer value that represents the current configuration. See table above for the values that are currently configured.

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the security parameters were set) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTSEC
+BTSEC: 0
OK

AT+BTSEC=2
OK
```

+BTPIN – read/write PIN code that is used for authentication and pairing

Description:

This command is used to query (or change) the Bluetooth PIN code that is used for authentication (security) procedures. Note that if the PIN code is set that it will not be saved to Flash (unless the save to Flash command is used afterward).

Format:

AT+BTPIN="<PIN>"

where,

<PIN> represents the ASCII string to use for the Bluetooth PIN code. This value must be at least a single character and cannot be longer than 16 characters. This string must be enclosed in quotation marks. It should be noted that the PIN code is currently restricted to printable ASCII characters (it does not have to be restricted to numeric ASCII characters).

or one of the following to query the current PIN code

AT+BTPIN?
AT+BTPIN

Results:

If a query is performed, the output will be the following format:

+BTPIN: <PIN>

where,

<PIN> represents the ASCII string that is configured for the Bluetooth PIN code. This value will not be longer than 16 characters. This string will also be enclosed in quotation marks. It should be noted that the PIN code is currently restricted to printable ASCII characters (it does not have to be restricted to numeric ASCII characters).

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If a query was not performed (i.e. the PIN code was changed) then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTPIN
+BTPIN: "0000"
OK

AT+BTPIN="12345678"
OK
```

+BTPAR – manage pairing operations

Description:

This command is used to manage operations relating to Bluetooth link key maintenance and pairing. More specifically, this command allows the ability to initiate pairing with a remote device, allow a remote device to discover local device and pair, allow the deletion of a currently stored Bluetooth link key, and to view the current paired device (Bluetooth address and link key). This function relies on the configured PIN code when it requires a PIN code to complete the pairing operation (see the **AT+BTPIN** command). It should be noted that this function does not rely on the configured security properties (see the **AT+BTSEC** command). If the option to either pair remotely or wait for a remote device to pair is selected, the action can be cancelled at any time by issuing the cancel command (**AT+BTCAN**).

Note that whenever a new link key is generated it is stored directly to Flash. This means that when the device is reset it will maintain the known Bluetooth device address and link key settings.

Format:

AT+BTPAR=<X>, <BDADDR>,<LINKKEY>

where,

<X> is an integer that specifies the operation to be performed. This parameter is required and must be one of the operations listed in the following table:

Table 25: Pairing Operation

Value	Description
0x00	Delete any stored link key
0x01	Pair with remote device (initiate)
0x02	Allow another device to pair (wait for pair)
0x03	Configure Bluetooth address/Link key pair

The delete stored link key option, simply deletes any stored link key that has been stored.

The pair with remote device, attempts to connect to the specified remote Bluetooth device (next parameter – required) and perform an authentication to generate a new link key. This command will complete when the remote pairing operation is complete (or is cancelled locally).

The third option, forces the device to become discoverable and connectable, and simply waits for a remote device to connect and pair. This command will complete when a remote device pairs successfully with the local device (or is cancelled locally).

The final option, allows the ability to stipulate pairing information manually. This entails supplying a Bluetooth device address and Link key. The Link key supplied will be used when a connection is accepted or created to the corresponding Bluetooth device.

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device to pair with. This parameter is only required if the pairing operation is to pair with a remote device (by initiating the pairing locally) or if a Bluetooth address/Link key pair is being configured. This value is required to be 12 digits (note that this value is not proceded with the "**\x**" or the "**\X**" prefix).

<LINKKEY> is the 32 digit ASCII coded hexadecimal Bluetooth Link Key associated with the preceding Bluetooth device address. This parameter is only used with the configure Bluetooth address/Link key sub command and must be present when using this option (the Bluetooth address must be present as well).

or one of the following to query the currently paired device information:

AT+BTPAR?
AT+BTPAR

Results:

If any action other than the deletion of the local link key information, the output will be as follows:

+BTPAR: <BDADDR>, <Link Key>

where,

<BDADDR> is the 12 digit ASCII coded hexadecimal Bluetooth Device address of the remote Bluetooth device for which the link key is associated. This value will be 12 digits (note that this value is not preceded with the "**\x**" or the "**\X**" prefix). If there is no link key currently stored, then this value will consist of a Bluetooth address comprised of all zeros ("000000000000" – 12 zeros without the quotation marks).

<Link Key> is the 32 digit ASCII coded hexadecimal Bluetooth link key that is associated with the specified Bluetooth device address (prior parameter). This value will be 32 digits (note that this value is not preceded with the "**\x**" or the "**\X**" prefix). If no link key was assigned (or is present) then this value will consist of a link key comprised of all zeros ("00000000000000000000000000000000" – 32 zeros without the quotation marks).

Immediately following the above output, the response will be the result code (either **OK** or **ERROR**).

If the deletion of a link key was performed then the result will simply consist of the result code (either **OK** or **ERROR**).

Example:

```
AT+BTPAR?
+BTPAR: 000000000000, 00000000000000000000000000000000
OK

AT+BTPAR=0
OK

AT+BTPAR=1, 123456789012
+BTPAR: 123456789012, 00000000000000000000000000000000
OK

AT+BTPAR=2
+BTPAR: 000123456789, 52FC670498F0E217A4DB83C86ACB9D9F
OK
```

+BTCAN – cancels any currently outstanding command

Description:

This command is cancels any outstanding command that was issued (in command mode – i.e. it doesn't cancel the **AT+BTSRV** or **AT+BTCLT** commands). This command can be used to cancel the following commands (while they are outstanding):

Table 26: Outstanding Commands

Command	**Description**
AT+BTRNM	Discover remote device name
AT+BTSDP	Discover remote services
AT+BTTST	Bluetooth test mode
AT+BTRXT	Calibrate RXTUN
AT+BTTX	FCC/Bluetooth transmission test
AT+BTINQ	Discover remote devices
AT+BTPAR	Pair with remote devices

This command can be issued at any time, however, unless one of the above commands has been issued (and either hasn't completed or been exited), the cancel command will not perform any action.

Format:

AT+BTCAN

Results:

Result will be the result code (either **OK** or **ERROR**).

Example:

```
AT+BTTST=1
OK
AT+BTCAN
OK
```

+BTSET – override Factory setting with specified Dynamic setting

Description:

This command is used to override a Factory setting value. The mechanism by which a Factory settings is overridden is via the Dynamic settings. This command can be used to configure the following Dynamic settings (note that Dynamic settings ALWAYS override any configured Factory setting value):

Table 27: Dynamic Setting Supported

Setting	**Description**
Bluetooth Address	Bluetooth Device Address
RXTUN	Radio RXTUN value

Reference Voltage	Reference Voltage
Class 2 Trimming	Used to ensure trim Class 2 power such that output does not exceed +4dBm
Course Tuning	Course Tuning

This command can be issued at any time, however, once one of the above settings has been changed it will take effect immediately.

Format:

AT+BTSET=<X>,<Value>

where,

<X> is an integer that specifies the option to be configured. This parameter is required and must be one of the values listed in the following table:

Table 28: Dynamic Settings

Value	**Description**
0x01	Bluetooth device address
0x02	RXTUN value (tuning)
0x03	Reference Voltage
0x04	Class 2 Trimming
0x05	Course Tuning

The Bluetooth device address setting is used to configure the Bluetooth device address of the Bluetooth device. Note that when this value is set, any currently stored Link Key (pairing) information is deleted.

The RXTUN setting is used to specify the correct RXTUN value for the specified hardware. Please see the **Radio Calibration** section for more information.

The third, fourth, and fifth settings allow additional tuning information to be specified. Under normal circumstances these values do not have to be modified from their default settings.

<Value> is the value that is to be specified for the Dynamic setting. This parameter is required and is dependent upon the first parameter. When setting the Bluetooth device address, this value is the 12 digit ASCII coded hexadecimal Bluetooth Device address that is to be used. The remaining settings treat this value as either an integer value (0 -255) or a

hexadecimal value (0x00 – 0xFF) which represents the value to use for the corresponding setting (the first parameter).

Results:

Result will be the result code (either **OK** or **ERROR**).

Example:

```
AT+BTBDA
+BTBDA: BDB203005511
OK AT+BTSET=1,112233445566
OK
AT+BTBDA
+BTBDA: 112233445566
OK
```

8. Default Settings

This section details the default values that are used if no values are configured or changed. These are the values that each parameter will have unless they are overridden. These values can be overridden by one of two mechanisms:

- Factory settings
- Command configuration

The factory settings configuration is overridden by using the Factory settings configuration tool. The usage of this tool is not covered in this document, however, it allows all configurable settings to be set and downloaded to the device. The command configuration options are set via the commands that are documented in this document. Currently all options can be configured via the Factory settings configuration tool or via the command interface. This section high-lights the settings that are used if the device has not been configured with one of the above mechanisms. The table below lists all the configurable parameters, their defaults, and the command that can be used to change the parameter.

Note that physical parameters (for example, the crystal settings) can only be changed via the Factory settings tool. In practice this is not a limitation because changing these values to the wrong values will result in the incorrect operation of the 1SPP firmware.

Table 29: Default Configuration Parameters

Parameter	Value	Command
Bluetooth Address	BDB203005511	AT+BTSET
RXTUN	0	AT+BTSET
Reference Voltage	0xAC	AT+BTSET[1]
Class 2 Trimming	0 (Disabled)	AT+BTSET[1]
Course Tuning	0	AT+BTSET[1]
Device type	DCE	AT+BTCFG
RI/CD/DTR/DSR pass-through	Disabled	AT+BTCFG[2]
Ignore escape sequence	False	AT+BTCFG
Suppress command responses	False	AT+BTCFG
DTR/DSR enter command mode	False	AT+BTCFG[2]
Connection active GPIO output	False	AT+BTCFG
Device name	BGB203 – 1SPP	AT+BTLNM
Class of device	001F00	AT+BTCOD
Link supervision timeout	20 seconds	AT+BTLSV
Echo characters in command mode	True	ATE
Escape Sequence	+++	AT+BTESC
PIN code	0000	AT+BTPIN
Security	No Security	AT+BTSEC
Enable low power mode use	False	AT+BTPWR
Low power mode inactivity timeout	30000 (30 sec)	AT+BTPWR
Low power minimum sniff interval	320 (200 ms)	AT+BTPWR
Low power maximum sniff interval	480 (300 ms)	AT+BTPWR
Low power sniff attempt	16 (20 ms)	AT+BTPWR
Low power sniff timeout	8 (10 ms)	AT+BTPWR
UART baud rate	115200	AT+BTURT
UART word length	8	AT+BTURT
UART parity	0 (None)	AT+BTURT
UART stop bits	1	AT+BTURT
UART RTS/CTS flow control	False	AT+BTURT

UART DTR/DSR flow control	False	AT+BTURT[2]
SDP search profile filter	0xFFFFFFFF (All)	AT+BTSDP
SPP server SDP service name	Serial Port	AT+BTSRV
SPP server flags	0 (Discoverable)	AT+BTSRV
SPP client number of connection attempts	1	AT+BTCLT
SPP client connection period	0	AT+BTCLT
Automatic connection mode	Disabled	AT+BTAUT

[1]= Rarely required to be changed from the default value
[2]= DTR/DSR flow control must be enabled to support DTR/DSR functions

9. Tutorial

This section serves a very simplistic tutorial to show how a pair of 1SPP devices might be utilized to make a connection. This tutorial assumes that there are two 1SPP devices (one with Bluetooth device address 00123456789A, the other with Bluetooth device address 0123456789AB) and that neither device has been configured with any settings other than the default settings (listed in the default settings section). This tutorial will have the description in the current typeface, and will contain actual commands and responses from the devices themselves. The commands and responses will be represented in a font like the following: `AT+BTBDA`.

First, using device B (0123456789AB), we will change the device name as follows:

```
AT+BTLNM=”Device B”
OK
```

To verify that the above operation was successful, we will query the local device name on device B:

```
AT+BTLNM
+BTLNM: “Device B”
OK
```

Next, we will simply create a Serial Port Profile (SPP) server on device B. This causes the following events to happen:

- Serial Port Profile (SPP) server is created (using the port number specified)
- Serial Port Profile (SPP) service record is registered with the local Service Discovery Protocol (SDP) server (using either default name or the name specified)
- Local device is configured to allow devices to connect and discover it
- Device enters data mode (i.e. no further UART data will be processed as 1SPP commands until command mode is entered

All of the above is taken care of with the following command (issued on device B):

```
AT+BTSRV=1
OK
```

Since no options were specified, the service name will be the default and the device will be discoverable by other Bluetooth devices.

Now that device B is connectable and discoverable, we will use device A (00123456789A) to discover the device and the service it is publishing. First we will perform an inquiry to determine all devices in the local proximity to device A (that are discoverable) for five seconds:

```
AT+BTINQ=5
+BTINQ: 0123456789AB, 001F00
+BTINQ: COMPLETE
```

```
        OK
```

Looking at the above dump we see that device A has indeed found device B.

Next, we will attempt to discover the services that device B is publishing (note that since we already knew device B's Bluetooth address we didn't really have to perform an inquiry and could have simply proceeded to the service discovery step below:

```
        AT+BTSDP=0123456789AB
        +BTSDP: 1, 1.0, "Serial Port", 1
        OK
```

Looking at the above, we can see that there is one service being published, it is of type Serial Port Profile (first parameter), it has Bluetooth protocol version 1.0 (second parameter), the service name is "Serial Port" (third parameter), and the RFCOMM server port is port 1 (final parameter).

Now that we know the remote device and remote device server port (as with the inquiry procedure since we already knew the port (and service) of the remote device we could have skipped the service discovery phase and immediately tried to connect) we will attempt a connection from device A to device B. On device A we will issue the following;

```
        AT+BTCLT=0123456789AB, 1
        OK

        CONNECT 0123456789AB
```

Device A has shown that we have made a connection to device B. If we look at the output of device B we will see the following:

```
    CONNECT 00123456789A
```

At this point both devices are in data mode and anything that is entered on device A or B (through the UART) will be sent to the remote device (and should be displayed).

Next, using device A (either device could be used) we will disconnect the link. This is done by simply entering the escape sequence. Since we are using the default values, the escape sequence will be three ASCII "+" characters (minus the quotation marks). On device A we enter:

```
        +++

        NO CARRIER
        OK
```

The Bluetooth connection is now terminated and device A is in command mode again ready to accept commands. If we look at the output of device B we see:

```
        +++
NO CARRIER
```

It should be that there is no "OK" response printed because this was output on device A in response to the processing of the escape sequence. Since device B did not process the escape sequence (it only processed the Bluetooth link being terminated) it did not output an "OK" response.

Hopefully the above tutorial serves as enough of a starting point to try some of the more advanced features. It should be noted that even if advanced features are used the process will be very similar. Also, it should be noted that the 1SPP firmware is not required to connect with another 1SPP device. Any device that supports the Bluetooth RFCOMM protocol can be connected with.

Date of release:21 Dec 2005

Published in The Netherlands

10. Radio Calibration

This section serves to describe the procedure for calibrating the internal Bluetooth radio. This procedure is required because of the differing tolerances of external components (e.g. crystals, resistors, capacitors, board layout, etc.). The outcome of a successful calibration will yield an integer value (in the range of 0 through 255) that will be used as the RXTUN value in the **AT+BTSET** command.

The following tools will be needed to perform the calibration:

- Host to send commands to the BGB203 running the 1SPP firmware
- BGB203 running the 1SPP firmware (in the circuit to be calibrated)
- High precision frequency counter

The procedure begins by making sure that the host can communicate with the BGB203/1SPP using the AT-command set described elsewhere in this document. It should be noted that calibration needs to occur on the actual design that is to be used by the BGB203/1SPP solution. It should also be noted that if the same design is produced with different production runs ('lots') of chips then the different production runs should be calibrated separately for the best performance (note that this could mean calibrating a few boards from each of the lots and determining an average value that can be used to calibrate all the boards).

Connect the frequency counter to the CTS/EXT_CLK output on the BGB203 (pin 42). All hardware has been configured at this point.

To perform the actual calibration, the **AT+BTRXT** command will be used. This command takes two parameters. The first, an integer, specifies whether or not calibration is to be performed. A value of zero means that calibration is not to be performed; a value of one means that calibration is to be performed. The final parameter is the RXTUN value to use for the calibration. This value will be an integer in the range of 0 to 255.

The procedure for determining the RXTUN value begins by issuing the **AT+BTRXT** command, enabling calibration, and choosing an initial value. For example:

```
AT+BTRXT=1,128
OK
```

After the command is issued, a 12 MHz clock should be output on the CTS/EXT_CLK output of the BGB203 (pin 42). The frequency of this clock can be measured with the high precision frequency counter. The actual calibration involves changing the value of the RXTUN parameter (the last parameter) until the clock that is output is as close to 12 MHz as possible (using the calibration process). Once the 12 MHz clock is observed the value that was used for the RXTUN parameter should be noted. This is the value that will be used as the RXTUN value in the **AT+BTSET** command.

To increase the speed at which the correct RXTUN value is found a binary search methodology can be employed to more quickly locate the correct value. For this approach to work, it must be determined if larger values of RXTUN increase or decrease the frequency (and consequently smaller values will achieve the opposite result). After this is determined, values should be picked that either increase or decrease the frequency (based on the frequency counter) that are half-way between the current value and the current acceptable range. For example (only the values are listed not the commands):

128 (determined the frequency is too low and higher values increase frequency)
192 (half way between 128 and 255 – too high)
160 (half way between 128 and 192 – too low)
176 (half way between 160 and 192 – too high)
168 (half way between 160 and 176 – too low)
172 (half way between 168 and 176 – too low)
174 (half way between 172 and 176 – too high)
173 (half way between 172 and 174 – correct value)

Note, depending on the precision, it is possible that two values are determined such that one value is too low and the other value is two high. In this case, the value that is closest to the 12 MHz frequency should be chosen.

11. Pinout Information

As previously mentioned, the basic board layout and connections are almost identical to the layout required to mount the BGB203 SIP in any other design that uses the UART interface to communicate with the Bluetooth controller (for example the basic Host Controller Interface (HCI) firmware). The 1SPP firmware does however utilize some specific pins of the BGB203 SIP for some of the features. This section will highlight all of the pins that are explicitly used by the 1SPP firmware (note that this section does not reference all of the pins on the BGB203, such as those used by the Bluetooth radio, power, etc).

Table 30: Pinout List

Feature	Pin Number	Pin Name	Direction	Internal Pull up
TX	41	GPIO[4] TXD_UART	Output	None
RX	40	GPIO[5] RXD_UART	Input	None
RTS	39	GPIO[3] RTS_UART	Output	None
CTS EXT_CLK	42	GPIO[2] CTS EXT_CLK	Input/Output	None
CD	22	GPIO[11]	Input/output	10K Ohm
RI	12	GPIO[12]	Input/output	10K Ohm
DTR	21	GPIO[13]	Input/output	None
DSR	11	GPIO[14]	Input/output	None
Active Connection	14	GPIO[17]	Output	None

Date of release:21 Dec 2005

Published in The Netherlands

12. Contents

Date of release:21 Dec 2005

Published in The Netherlands