User Guide

SHIELD BLUETOOTH - AT Interface

These are text commands starting with the character sequence AT and terminated by a \r character (ASCII code 0x0D). Commands are not case sensitive and have zero or more parameters. Multiple parameters are separated by the comma (,) character and some commands tolerate empty fields (two consecutive commas) and provide a default value. If more parameters are supplied than those specified, then the extra parameters are either silently ignored or result in a syntax error response.

	Used to add an AD element with tag and payload specified to the scan report cache variables for adverts that are used when operating in non-vSP mode. To add to the advert report, use AT+AARA This does not affect the adverts that are already committed to the radio and can be called multiple times to add more AD elements. To commit to the radio, use the AT+ACMT command.
Possible Responses	OK ERROR

AT+BNDD

Command	**AT+BNDD address**
Description	Use this command to delete a device from the trusted device database.
Possible Responses	OK ERROR

AT+BNDP

Command	**AT+BNDP address**
Description	When a pairing is successful, the pairing keys and address are stored in the trusted device database and are marked as a rolling type. If the database is full, to guarantee storage of the newest pairing, the oldest rolling record is automatically deleted to make space. User this to change the type of record to persistent so it can only be deleted if explicitly done using the AT+BNDD command.
Possible Responses	OK ERROR

AT+BNDT

Command	**AT+BNDT address**

Description	Checks if a device identified by *address* (a 14-digit hex string) is present in the trusted device database (a result of a successful pairing).
	The following response is sent before the OK if it is not trusted:
	`\n0\r`
	If trusted, the response is:
	`\n1,t,14digithexaddr\r`
	…where *t* is 0 if the pairing is persistent and !0 if rolling.
	Note: *Rolling* means that, at some point, it could be automatically deleted on a new pairing if the database is full.
	14digithexaddr is the actual MAC address of the device if the *address* passed to this command is a resolvable address.
	At any time, the command ATI2009 returns the number of devices in the trusted device database.

Possible Responses	OK ERROR

AT+BNDX /Erase trusted device database/

Command	**AT+BNDX**
Description	Use this command to delete all devices from the trusted device database, both rolling and persistent types.
Possible Responses	OK ERROR

AT+GCTM

Command	**AT+GCTM hIdx**
Description	This is a GATT client-related command. Use it to obtain the GATT table schema (such as the structure) of the peer connected on the handle identified by hIDx. This results in many responses starting with either *TM: S* or *TM: C* and *TM: D*. For example, the following from a device contains three services: ▪ First service – Contains four characteristics ▪ Second service – Contains one characteristic ▪ Third service – Contains four characteristics In addition, the characteristic in the second service has a descriptor. In total, there are three descriptors in the entire GATT table.

```
AT+GCTM1
TM:S:1 ,(9) ,FE011800
TM: C:3 ,00000002 ,FE012A00 ,0
TM: C:5 ,00000002 ,FE012A01 ,0
TM: C:7 ,00000002 ,FE012A04 ,0
TM: C:9 ,00000002 ,FE012AA6 ,0
TM:S:10 ,(13) ,FE011801
TM: C:12 ,00000020 ,FE012A05 ,0
TM:  D:13 ,FE012902
TM:S:14 ,(65535) ,FD021101
TM: C:16 ,00000010 ,FD022000 ,0
TM:  D:17 ,FE012902
TM: C:19 ,0000000C ,FD022001 ,0
TM: C:21 ,00000010 ,FD022002 ,0
TM:  D:22 ,FE012902
TM: C:24 ,0000000C ,FD022003 ,0
OK
```

Where:

TM:S	Indicates the start of a BLE Service whose starting attribute handle is the integer value after the second ':' in that line.
The next integer parameter (in brackets)	The last attribute handle in that service.
Last eight-digit hex number	The UUID handle supplied by the firmware ***Note:*** *This is not the index mentioned in the AT+UUID command description.*

TM: C	Indicates the start of a BLE Characteristic
The integer after the second ':'	The handle for the value attribute
The next integer	Eight-digit hex value that denotes the characteristic properties (see command AT+GSCB for details)
The next eight-digit hex number	The UUID handle supplied by the firmware
The final decimal number	Is always 0. Intended as a place holder for the *Included Service UUID Handle*. ***Note:*** *We have not yet encountered an Included Service. We will add this functionality as needed.*

TM: D	Indicates the start of a BLE Descriptor that belongs to a Characteristic (such as CCCD)
The integer after the second colon (:)	Its attribute handle

Next hex number	The UUID handle supplied by the firmware. The last four digits of the UUID are the 16-bit adopted UUID if the first four digits are FE01. For example, if the last four digits are 2902, it is a CCCD. This means that you can use the attribute handle with the AT+GCWC command to write an enable/disable notify/indicates for the characteristic to which it belongs.
The host processing the TM responses know there are no more to come when it receives either an OK or ERROR message.	

Possible Responses

```
OK
ERROR
TM:S
TM: C
TM:  D
```

AT+GFCA

Command **AT+GCFA hIdx, uS, x, uC, y <,uD,z>**

Description

This is a GATT client-related command.

Use this command to search for the handle of the value attibute of a Characteristic or the attribute handle of a descriptor attached to a characteristic in the peer connected on the handle identified by hIDx.

<uD,z>	(Optional) When this is absent, it implies that the search is for the value handle of a characteristic. When present, it implies that the search is for the descriptor. OK or ERROR terminates this command. If a characteristic or descriptor is found, the FC or FD responses have been received respectively.
uS uC uD	These are the UUID index that were used to pre-create a UUID handle using the command AT+UUID.
x y z	The 0-based instance index of the appropriate entity in the remote GATT table. For example, if x=1, y=2, and z=0, it means search for the second instance of a service with the UUID *uS*. In that service, search for the third instance of the characteristic with UUID *uC*; and in that characteristic, look for the first instance of the descriptor with UUID *uD*. **Note:** Typically, GATT tables do not have multiple instance services.

The main use of such a command is to locate a characteristic or descriptor in a server device to obtain the attribute handle so that it can be subsequently used in read/write requests using commands AT+GCRD, AT+GDWA, AT+GCWC.

This command immediately responds with OK or ERROR and, at some time subsequent, the asynchronous response FC or FD is received

When the attribute handle specified in the FC or FD is 0, it implies that the object was not found in the remote GATT table.

Possible Responses

OK
ERROR
FC
FD

AT+GCRD

Command

AT+GCRD hIdx, hAttr, nOffset

Description

This is a GATT client-related command.

It is used to read the content of a remote attribute starting at offset specified within that attribute. For example, if the attribute contains *Hello World,* setting nOffset to 6 results in *World* being read.

hIdx	The connection handle of the server from which it reads
hAttr	The attribute of the handle that was extracted using either AT+GCTM or AT+GCFA commands

This command immediately responds with OK or ERROR and at some time subsequent, the asynchronous response AR is received.

If the read was successful then an AR response is received which contains the data. If the read failed (for example, if the attribute does not exist or it requires the connection to be authenticated), then the AS response is received. In rare occasions, an AB could also be received if, for example, the module is low in memory.

Possible Responses

OK
ERROR
AR
AS
AB

AT+GCWA

Command

AT+GCWA hIdx, hAttr, hexdatastring

Description

This is a GATT client-related command.

It is used to write data to an attribute in a remote GATT table and expects an acknowledgement which will be received as an asynchronous response “AW” after the terminating “OK” response.

hIdx	The connection handle of the server from which it reads
hAttr	The attribute of the handle that was extracted using either AT+GCTM or AT+GCFA commands
hexdatastring	A string consisting of only hexadecimal characters which must be an even number in length. It is converted to binary before writing to the peer.

It always writes to offset 0 in the destination attribute.

If the attribute rejects the write because say the connection is not encrypted, then the AW will have the appropriate status value.

Possible Responses

OK
ERROR
AW

AT+GCWC

Command

AT+GCWC hIdx, hAttr, hexdatastring

Description

This is a GATT client-related command.

It is used to write data to an attribute in a remote GATT table; it does not expect an acknowledgement after the terminating OK response. If the command fails to write the value then there will eventually be a disconnection because the link supervision timer will timeout.

hIdx	The connection handle of the server from which it reads
hAttr	The attribute of the handle that was extracted using either AT+GCTM or AT+GCFA commands
hexdatastring	A string consisting of only hexadecimal characters which must be an even number in length. It is converted to binary before writing to the peer.

It always writes to offset 0 in the destination attribute.

If the attribute rejects the write because say the connection is not encrypted, then the AW will have the appropriate status value.

Possible Responses

OK
ERROR

AT+GSMD, AT+GSCB, AT+GSCE, AT+GSSB, AT+GSSE

Command

AT+GSMD m, rdRights, wrRights, len
AT+GSCB uC, prop, mVal <,mCccd<,mSccd>>
AT+GSCE hexdatastring
AT+GSSB uS
AT+GSSE

Description

These are GATT server-related commands used to populate the local GATT server table with services, characteristics, and descriptors.

A characteristic can have properties like read/write and CCCD and/or SCCD descriptors which may or may not require authentication.

When adding a characteristic, those attributes must be specified. You can achieve this by using a metadata object which must be pre-created using the AT+GSMD command. Just like UUID handles management, this app provides for an array of metadata objects that are referenced using the index *m* in the range 0 to 3.

AT+GSMD is used to create a metadata object in array index *m* and creates an opaque integer value that contains the read and write which can be any one of these values:

0	No access
1	Open
2	Encrypted with no man-in-the-middle (MITM) protection

AT+SIOW

Command	**AT+SIOW sionum,val**
Description	Once a signal pin is configured using the AT+SIOC command, if it was configured as a digital_out, this command sets the current value which will be 0 or 1.
Possible Responses	OK ERROR

AT+UUID

Command	**AT+UUID u,16bitUuid** **AT+UUID u,32HexDigitNumber** **AT+UUID u,16bitUuid,v**
Description	*BLE makes wide use of UUIDs (universally unique identifiers) which are 128-bit (16-byte) random values. These values can be cumbersome to manage as string objects and so the module firmware exposes a concept of a 32-bit integer value which is a handle to an internal 16 byte buffer that contains the actual value.* *The smartBASIC application exposing the AT interface functionality extends that concept by using an array of integer variables to store those handles provided by the firmware. Those firmware handles are never exposed, but instead an index value* 'u' *is.* The 'u' in these three variants of the command is the index into that integer array. Think of there being a bunch of mailboxes numbered 0 to N (see MAX_UUID_HANDLES in the source code) which are your scratchpads to load UUID handles into (using these commands) as and when you need to supply a UUID into any of the AT commands that require a UUID. For example the command AT+GSSB takes a parameter which is one of these 0 to N indices. The value for 'u' shall always be in the range 0 to N, where N is 15 at the time of writing and can be modified by changing the #define for MAX_UUID_HANDLES. The command variant "AT+UUID u,16bitUuid" is used to create a handle from a Bluetooth SIG adopted 16 bit UUID and store it in the array index 'u'. The value 16bitUuid shall be in the range 0 to 0xFFFF. The command variant "AT+UUID u,32HexDigitNumber" takes the 32 character hexadecimal string and converts that into a handle and stores it in the array index 'u'. The command variant "AT+UUIS u,16bitUuid,v" takes the '16bitUuid' which is a value in the range 0 to 0xFFFF and creates a sibling of the handle stored in array index v and stores in array index 'u'. By sibling, it is meant that the base UUID of the handle stored in array index 'v' is used to create the new UUID.
Possible Responses	OK ERROR