Class 1 Bluetooth v2.0 Module

FIRMWARE USER'S GUIDE
VERSION 1.3

Part # BT730-SA, BT730-SC

Americas: +1-800-492-2320 Option 2
Europe: +44-1628-858-940
Hong Kong: +852-2923-0610
www.lairdtech.com/bluetooth

REVISION HISTORY

Revision	Revision Date	Description	Approved By
0.7	3 July 13	Prelim for KP	Jonathan Kaye
0.8	04 Oct 13	Updated Mechanical Drawing – Pad Definitions	Jonathan Kaye
0.9	6 November 13	Clean Up and DW Regulatory Updates Separated HIG and Firmware Manual CA version and JK review	Jonathan Kaye
1.0	24 June 2013	First release	Jonathan Kaye
1.1	30 Aug 2013	Minor updates.	Jonathan Kaye
1.2	27 Feb 2014	Removed reference to S522 and Go Blue.	Jonathan Kaye
1.3	04 April 2014	Updates to S534.	Jonathan Kaye

TABLE OF CONTENTS

1. AT Command Set Reference

1.1 Introduction

This document describes the protocol used to control and configure the following Laird Technologies Bluetooth devices:

- BT730-SA
- BT730-SC

The protocol is similar to the industry standard Hayes AT protocol used in telephony modems which is appropriate for cable replacement scenarios, as both types of devices are connection oriented. The telephony commands have been extended to make the Laird device perform the two core actions of a Bluetooth device, which is make/break a connection and Inquiry. Other AT commands are also provided to perform ancillary functions, such as pairing, trusted device database management, and S register maintenance.

Similar to telephony modems, the Laird device powers up in an unconnected state and only responds via the serial interface. In this state, the Laird device does not respond to Bluetooth inquiries. Then, just like controlling a modem, the host can issue AT commands which map to various Bluetooth activities. The command set is extensive enough to allow a host to make connections which are authenticated and/or encrypted or not authenticated and/or encrypted or any combination of these. Commands can be saved, so that on a subsequent power-up the device is discoverable or automatically connects.

The device has a serial interface which can be configured for baud rates from 1200 up to 921600 and an RF communications end point. The latter has a concept of connected and unconnected modes and the former has a concept of command and data modes. This leads to the matrix of states shown in Table 1-1.

Table 1-1: Matrix of mode states

	RF Unconnected	RF Connected
Local Command Mode	OK	OK
Remote Command Mode	ILLEGAL	OK
Data Mode	ILLEGAL	OK

The following combinations do not make sense and are ignored:

- Data and RF Unconnected Mode
- Remote Command and RF Unconnected Mode

Navigation between these states occurs using the AT commands which are detailed in subsequent sections.

1.2 Assumptions

The CSR (Cambridge Silicon Radio) BC04 chipset in Laird devices is memory resource limited. Therefore it is **not** proposed that there be full implementation of the AT protocol as seen in modems. The claim made for this device is that it has a protocol *similar* to an AT modem. In fact, the protocol is so similar that existing source code written for modems can be used with very little modification with a Laird device.

The following assumptions are made:

- All commands are terminated by the carriage return character 0x0D, which is represented by the string <cr> in descriptions below and this cannot be changed.

- All responses from the Laird device have carriage return and linefeed characters preceding and appending the response. These dual character sequences have the values 0x0D and 0x0A respectively and shall be represented by the string <cr,lf>.
- All Bluetooth addresses are represented by a fixed 12 digit hexadecimal string, case insensitive.
- All Bluetooth Device Class codes are represented by a fixed six digit hexadecimal string, case insensitive.
- All new Bluetooth specific commands are identified by the string +BT**x**, where **x** is generally a mnemonic of the intended functionality.

1.3 Commands

This section describes all available AT commands. Many commands require mandatory parameters and some take optional parameters. These parameters are integer values, strings, Bluetooth addresses, or device classes. The following convention is used when describing the various AT commands.

<bd_addr>	A 12 character Bluetooth address consisting of ASCII characters '0' to '9', 'A' to 'F' and 'a' to 'f'.
<devclass>	A 6 character Bluetooth device class consisting of ASCII characters '0' to '9', 'A' to 'F' and 'a' to 'f'.
n	A positive integer value.
m	An integer value (positive or negative) which can be entered as a decimal value or in hexadecimal if preceded by the '$' character. E.g. the value 1234 can also be entered as $4D2
<string>	A string delimited by double quotes. E.g. "Hello World". The " character MUST be supplied as delimiters.
<uuid>	A 4 character UUID number consisting of ASCII characters '0' to '9', 'A' to 'F' and 'a' to 'f'.

1.3.1 ^^^{Enter Local Command Mode}

When in data and connected mode, the host can force the device into a command and connected mode so that AT commands can be issued to the device. The character in this escape sequence is specified in the S2 register, therefore it can be changed. In addition, the escape sequence guard time is specified by S Register 12. By default the guard time is set to 100 milliseconds. Refer to Section 1.6: Dropping Connections for more information.

In modems this escape sequence is usually "+++". "^^^" is specified to avoid confusion when the module is providing access to a modem.

Response:<cr,lf>OK<cr,lf>

1.3.2 !!!{Enter Remote Command Mode}

When in data and connected mode, the host can force the remote device into a command and connected mode so that AT commands can be issued to the device remotely. The escape sequence guard time is specified by S Register 12 and is the same as per the ^^^ escape sequence. By default the guard time is set to 100 milliseconds. The remote device issues ATO as normal to return to data mode.

For this command to be effective, S Register 536 must be set to 1.

Response:<cr,lf>OK<cr,lf>

1.3.3 AT

Used to check the module is available.

Response:<cr,lf>**OK**<cr,lf>

1.3.4 ATA{Answer Call}

Accept an incoming connection, which is indicated by the unsolicited string

<cr,lf>**RING 123456789012**<cr,lf> every second. **123456789012** is the Bluetooth address of the connecting device.

Response: <cr,lf>**CONNECT 123456789012**<cr,lf>

1.3.5 ATD<U><Y><bd_addr>,<uuid> {Make Outgoing Connection}

Make a connection to a device with Bluetooth address <bd_addr> and profile <uuid>. The <uuid> is an optional parameter which specifies the UUID of the profile server to attach to; if not supplied, then the default UUID from S Register 101 is used. Because this is a Laird device which utilises the RFCOMM layer as described in the Bluetooth specification, it implies that only profiles based on RFCOMM can be accessed.

If <U> is not specified, then authentication is as per register 500, otherwise the connection will be authenticated.

If <Y> is not specified, then encryption is as per register 501, otherwise the connection will have encryption enabled.

The timeout is specified by S register 505.

Response: <cr,lf>**CONNECT 123456789012**<cr,lf>

Or <cr,lf>**NO CARRIER**<cr,lf>

Due to a known issue in the Bluetooth RFCOMM stack, it is not possible to make more than 65525 outgoing connections. Therefore if that number is exceeded, then the connection attempt fails with the following response:

Response: <cr,lf>**CALL LIMIT**<cr,lf>

Or <cr,lf>**NO CARRIER**<cr,lf>

In that case, issuing an ATZ to reset the device resets the count to zero and allows more connections.

The following RFCOMM based UUIDs are defined in the Bluetooth specification:

Profile Name	UUID
Serial Port	1101
LAN Access Using PPP	1102
Dialup Networking	1103
IrMC Sync	1104
OBEX Object Push	1105
OBEX File Transfer	1106
IrMC Sync Command	1107
Headset	1108
Cordless Telephony	1109
Intercom	1110
Fax	1111

Profile Name	UUID
Audio Gateway	1112
WAP	1113
WAP_CLIENT	1114

1.3.6 ATD<U><Y><bd_addr>,<ServiceName>{Make Connection}

Make a connection to device with Bluetooth address <bd_addr> and profile specified via S Reg 101 AND which has a service name starting with the string <ServiceName>. The service name parameter is a string delimited by ".

If <U> is not specified, then authentication is as per register 500, otherwise the connection is authenticated.

If <Y> is not specified, then encryption is as per register 501, otherwise the connection has encryption enabled.

The timeout is specified by S register 505.

Response:	<cr,lf>CONNECT 123456789012<cr,lf>
Or	<cr,lf>NO CARRIER<cr,lf>

1.3.7 ATD<U><Y>L{Remake Connection}

Make a connection with the same device and service as that specified in the most recent ATD command. The <UY> modifiers are optional. An error is returned if the 'L' modifier is specified as well as a Bluetooth address.

If both 'L' and 'R' modifiers (see section 13.8 below)are specified then an error is returned.

Response:	<cr,lf>CONNECT 123456789012 AE<cr,lf>
Or	<cr,lf>NO CARRIER<cr,lf>

1.3.8 ATD<U><Y>R{Make Connection to peer specified in AT+BTR}

Make a connection with the device address specified in the most recent AT+BTR command. The service is as specified in S Register 101. The <UY> modifiers are optional. An error is returned if the 'R' modifier is specified as well as a Bluetooth address.

If both 'R' and 'L' modifiers (see section 13.7 above)are specified then an error is returned.

Response:	<cr,lf>CONNECT 123456789012 AE<cr,lf>
Or	<cr,lf>NO CARRIER<cr,lf>

1.3.9 ATEn{Enable/Disable Echo}

This command enables or disables the echo of characters to the screen. A valid parameter value is written to S Register 506.

E0	Disable echo.
E1	Enable echo.

All other values of n generate an error.

Response:	<cr,lf>OK<cr,lf>
Or	<cr,lf>ERROR nn<cr,lf>

1.3.10 ATH{Drop Connection}

Drop an existing connection or reject an incoming connection indicated by unsolicited RING messages.

Response: <cr,lf>NO CARRIER<cr,lf>

1.3.11 ATIn{Information}

This returns the following information about the Laird device.

I0	The product name/variant.
I1	The CSR firmware build number.
I2	The Laird firmware build number. For internal use only.
I3	The Laird firmware revision.
I4	A 12 digit hexadecimal number corresponding to the Bluetooth address of the Laird device.
I5	The manufacturer of this device.
I6	The maximum size of trusted device database.
I7	The manufacturer of the Bluetooth chipset.
I8	The chipset format.
I9	0 if not in a connect state and 1 if in a connect state.
I11	The reason why a "NO CARRIER" resulted in the most recent attempt at making an outgoing connection. Where the response values are as follows: ▪ 0 = No prior connection ▪ 1 = Connection timeout ▪ 2 = Connection attempt cancelled ▪ 3 = Normal disconnection ▪ 4 = Peer device has refused connection ▪ 5 = Service profile <uuid> requested not available on remote device ▪ 6 = Connection has failed ▪ 32 = ATH was entered ▪ 33 = Incoming connection aborted because too many rings ▪ 34 = Unexpected incoming connection ▪ 35 = Invalid address ▪ 36 = DSR is not asserted ▪ 37 = Call limit of 65531 connections has been reached ▪ 38 = Pairing in progress ▪ 39 = No link key ▪ 40 = Invalid link key ▪ 255 = Unknown reason
I12	The last ERROR response number.
I13	The Sniff status is returned as follows: **Response:**<cr,lf>**a:b,c,d,e**<cr,lf>**OK**<cr,lf> Where 'a' = 0 when not online and 1 when online and Sniff has been enabled, 'b' is the Sniff Attempt parameter, 'c' is the Sniff timeout parameter, 'd' is the minimum sniff interval and 'e' is the maximum sniff interval. All parameters 'b', 'c', 'd' and 'e' are given as Bluetooth slots which are 625 microseconds long. Taken from S Registers 561, 562, 563, and 564 respectively.
I14	The current boot mode (only for firmware 1.18.0 and newer)

I15	The maximum length of an AT command, including the terminating carriage return (only for firmware 1.6.10 and newer).
I16	The size of AT command input buffer.
I20	Returns the number of bytes pending to be sent in the RF buffer when a connection is up.
I33	Version number of Multipoint application **Note:** ATI is provided for compatibility in multipoint mode; other AT commands are not available.
I42	State information. Where the response values are as follows: 13 = Not Open 14 = Open Idle 15 = Ringing 16 = Online Command 172 to 177 = waiting for connectable and/or discoverable where the lowest significant digit equates to the value stored in S Register 512 or 555. **Note**: When n=16, ATI9 returns 1.
I101	The RSSI value in dBm. If a connection does NOT exist then a value of -32786 is returned. A value of 0 means the RSSI is within the golden range; because this is a very large band, RSSI is not always a useful indicator. Use ATI111 instead which returns the bit error rate.
I111	Returns LinkQual which in the CSR chipset is defined as BER (bit error rate). This returns a value which is the number of bits in error out of 1 million. Hence a value of 0 is best, and larger values are worse. A value approaching 1000 (BER = 0.1%) is an indication that the link is bad and a large number of Bluetooth packets are being lost.
I333	Returns extended firmware version number.

For recognised values of n. All other values of n generate an error.

Response: <cr,lf>**As Appropriate**<cr,lf>OK<cr,lf>

Or <cr,lf>**ERROR**nn<cr,lf>

1.3.12 ATO{Enter Data Mode}(letter 'o')

Return to data mode. Assume that the module is in data mode after OK is received. Responds with an error if there is no Bluetooth connection.

Response: <cr,lf>**CONNECT 123456789012**<cr,lf>

Or <cr,lf>**ERROR**nn<cr,lf>

1.3.13 ATSn=m{Set S Register}

As with modems, the Laird Bluetooth module employs a concept of registers which are used to store parameters (such as escape sequence character and inquiry delay time) as listed in detail below.

The value part 'm' can be entered as decimal or hexadecimal. A hexadecimal value is specified via a '$' leading character. For example, $1234 is a hexadecimal number.

When S register values are changed, the changes are **not** stored in non-volatile memory *until* the AT&W command is used.

Note: AT&W is not required for S registers 520 to 525 or 1000 to 1010 as they are updated in non-volatile memory when the command is received.

Table 1-2: S Registers

Reg. #	Default	Range	Comment
S0	1	-1..15	Number of RING indications before automatically answering an incoming connection. A value of 0 disables autoanswer. If -1, then autoanswer on one RING and do NOT send RING/CONNECT response to the host. This emulates a serial cable replacement situation. Setting values >= 0, resets S Register 504 to 0 and <0 forces 504 to 1. If S0 <> 0 and S100 <> 0 then S0 must be < S100. If a value is entered which violates this rule, then ERROR 29 is sent in response. If S504 =1 then this register will return -1, regardless of the actual value stored in non-volatile memory.
S2	0x5E	0x20..0x7E	Escape sequence character. It is not '+' by default as a Bluetooth serial link can be used to connect to a mobile phone which exposes an AT command set, which in turn uses '+' as default. So if both used '+' there is confusion. 0x5e is the character '^'.
S12	100	40..5000	Escape sequence guard time in milliseconds, with a granularity of 20 ms. New values are rounded down to the nearest multiple of 20 ms.
S100	15	0..15	Number of RING indications before an auto disconnection is initiated. A value of 0 disables this feature. If S0 <> 0 and S100 <> 0 then S0 must be < S100. If a value is entered which violates this rule, then ERROR 29 is sent in response.
S101	$1101	0..$ffff	UUID of default SPP based profile when not specified explicitly in the ATD command.
S102	1	1..$7F	Defines a set of bits masks for enabling profile servers. Values can be ORed. ▪ 1 is Serial Port Profile ▪ 2 is Headset (S Reg 580 allows remote volume control bit to be adjusted) ▪ 4 is DUN ▪ 8 is Audio Gateway (Headset) ▪ 16 is Handsfree (S Reg 581 allows supported feature field to be adjusted) ▪ 32 is OBEX FTP ▪ 64 is Audio Gateway (Handsfree) It is recommended that, due to memory resource issues, no more than two profiles are activated at the same time.
S103	1	1..7	Boot mode on cold boot.
S126	?	0 .. 0xFFFF	Primer for changing to Multipoint mode
S127	?	0 .. 0xFFFF	0x100 for AT mode 0x200 for Multipoint mode Other values are reserved
S400	0	0..1	Piodaemon. 1 = Hostless Audio gateway Operation
S401	1000	100..5000	In Hostless Audio Gateway Operation – GPIO4 flash period while inquiring
S402	0	0..100	In Hostless Audio Gateway Operation – GPIO4 flash duty cycle while inquiring

Reg. #	Default	Range	Comment
S403	1000	100..5000	In Hostless Audio Gateway Operation – GPIO4 flash period when there is an ACL connection only to the headset
S404	0	0..100	In Hostless Audio Gateway Operation – GPIO4 flash duty cycle when there is an ACL connection only to the headset
S405	1000	100..5000	In Hostless Audio Gateway Operation – GPIO4 flash period when there is an ACL and SCO connection to the headset
S406	0	0..100	In Hostless Audio Gateway Operation – GPIO4 flash duty cycle when there is an ACL and SCO connection to the headset
S407	0	0..1	In Hostless Audio Gateway Operation – 'Lift-Hook' output follows SCO state
S408	0	0..1	In Hostless Audio Gateway Operation – if set to 1 then delete trusted device database when inquiry is initiated to look for headsets
S409	0	0..1	In Hostless Audio Gateway Operation – when inquiring and pairing, use the device class code of the response to classify which UUID to connect to the headset when initiating a Bluetooth connection from the gateway
S410	0	0..1	In AudioGatewayHostless mode, if set to 1, AG"" async responses will be forced out from the UART – good for debugging
S411	500	4000	In AudioGatewayHostless mode, Short press duration in milliseconds. 500 msec granularity
S412	500	4000	In AudioGatewayHostless mode, component of medium press duration in milliseconds. 500 msec granularity. Actual duration is this value plus S411
S413	500	4000	In AudioGatewayHostless mode, component of long press duration in milliseconds. 500 msec granularity. Actual duration is this value plus S412 plus S411
S414	30	240	In AudioGatewayHostless mode, the inquiry to search for headsets is aborted after this amount of time, in seconds. The granularity is 30 seconds.
S420	0	1	If this is set, then the module maintains a seconds counter. Use ATI420 to read the count value. It is basically the time the module has been powered up in seconds.
S500	0	0..1	Authentication for outgoing connections. Set to 1 to enable authentication.
S501	0	0..1	Encryption for outgoing connections. Set to 1 to enable encryption.
S502	0	0..1	Authentication for incoming connections. Set to 1 to enable authentication.
S503	0	0..1	Encryption for incoming connections. Set to 1 to enable encryption.
S504	0	0..1	Setting to 1 forces S0 to -1 and suppresses messages arising from connections or pairing. E.g. CONNECT, NO CARRIER, RING, PAIR etc. Suppressing connection-based messaged allows the Laird device to be configured in cable replacement mode.

Reg. #	Default	Range	Comment
S505	10	2..120	Minimum delay before abandoning connection attempt as a master. Referenced by ATD. In units of seconds. See S Registers 530 and 543. Please note that as disconnection time can vary, this register only guarantees the minimum delay. Note that for invalid addresses specified in the ATD command, the "NO CARRIER" response is immediate. See S register 560 for specifying disconnect max timeout.
S506	1	0..1	Enable/Disable echoes. The ATEn command also affects this.
S507	0	0..2	When set to 0, a connection can be dropped using ^^^ escape sequence only and the state of DSR line is ignored. When set to 1, a connection can be dropped using EITHER the ^^^ escape sequence OR the DSR handshaking line. When set to 2, a connection can only dropped using a deassertion of DSR. Mode 2 provides for the highest data transfer rate. If the status of the DSR line is to be conveyed to the remote device as a low bandwidth signal then this register MUST be set to 0, otherwise a deassertion of DSR will be seen as a request to drop the Bluetooth connection. This register affects S Register 536 – see details of 536.
S508	640	10..2550	Page Scan Interval in milliseconds. Minimum is 11.25 ms so 10/11 ms gives 11.25ms.
S509	320	10..2550	Page Scan Window in milliseconds. Minimum is 11.25 ms so 10/11 ms gives 11.25 ms.
S510	640	10..2550	Inquiry Scan Interval in milliseconds. Minimum is 11.25 ms so 10/11 ms gives 11.25ms.
S511	320	10..2550	Inquiry Scan Window in milliseconds. Minimum is 11.25 ms so 10/11 ms gives 11.25 ms.

<table>
<tr><th>Reg. #</th><th>Default</th><th>Range</th><th>Comment</th></tr>
<tr><td>S512</td><td>1</td><td>0..7</td><td>Specify power up state.
When set to 0, AT+BTO is required to open the device for Bluetooth activity.
<table>
<tr><th>Value</th><th>Effect</th></tr>
<tr><td>1</td><td>Proceeds to a state as if AT+BTO was entered.</td></tr>
<tr><td>2</td><td>Discoverable only (like AT+BTQ)</td></tr>
<tr><td>3</td><td>Connectable but not discoverable (like AT+BTG)</td></tr>
<tr><td>4</td><td>Connectable and discoverable (like AT+BTP)</td></tr>
<tr><td>5</td><td>Like 2, but all UART RX traffic is discarded in absence of a connection while DSR is asserted. If DSR is not asserted, then it behaves exactly as per mode 2.</td></tr>
<tr><td>6</td><td>Like 3, but all UART RX traffic is discarded in absence of a connection while DSR is asserted. If DSR is not asserted, then it behaves exactly as per mode 3.</td></tr>
<tr><td>7</td><td>Like 4, but all UART RX traffic is discarded in absence of a connection while DSR is asserted. If DSR is not asserted, then it behaves exactly as per mode 4.</td></tr>
</table>
<b>Note:</b> By implication, a change to this can only be seen after a power cycle AND if AT&W is actioned prior to the power cycle.
If S Reg 554 is non-zero and this register is between 2 and 7 inclusive, then the value of S554 specifies the time in seconds that the device will remain in the specified mode after power up. On timeout, the device falls back to the mode specified in S Register 555.
In modes 5, 6, and 7, when all RX activity is ignored, only the special command (capitalised) AT+BT&BISM& terminated by a <cr> forces the module temporarily back into modes 2, 3, and 4 respectively.
In some firmware builds, S Registers 565 to 569 inclusive are visible, which allows the start-up mode to depend on the state of RI line (Setting S Reg 565 forces the RI pin to be configured as an input). For this feature to be active, S Reg 565 should be set to 1. In that case, on start-up, if RI is asserted, then the start-up mode is defined by S Reg 566 and if deasserted then S Reg 567.</td></tr>
<tr><td>S513</td><td>1</td><td>0..1</td><td>Pairing Authentication, 1 = Enable</td></tr>
<tr><td>S514</td><td>10</td><td>1..60</td><td>Pairing Timeout in seconds. This includes the time a host takes to supply the PIN number when PIN? messages are indicated.</td></tr>
<tr><td>S515</td><td>0x001F00</td><td>0..
0xFFFFFF</td><td>Default Device Class Code to be used with AT+BTO when it is not explicitly specified. When queried, the value is always printed as a hexadecimal number.
To change the device class of the module, after AT+BTO, use the command AT+BTC.</td></tr>
</table>

Reg. #	Default	Range	Comment
S516	0x000000	0..0x2FFFFF	Default Device Class filter to be used with AT+BTI when it is not explicitly specified. When queried the value is always printed as a hex number. The seventh most significant digit, can be 0, 1, or 2, and is used to specify the type of device class filter. When 0, it specifies no filtering. When 1, it specifies an AND mask and all 24 bits are relevant When 2, it specifies a filter to look for devices with matching major device class which occupies a 5 bit field from bits 8 to 12 inclusive (assuming numbering starts at bit 0). All other 19 bits MUST be set to 0.
S517	20	2..61	Inquiry Length in units of seconds. This parameter is referenced by the AT+BTI command
S518	8	0..255	Maximum number of responses from an inquiry request. This parameter is reference by the AT+BTI command. If this number is set too high, then AT+BTI will return ERROR 27. For a particular firmware revision, determine the effective maximum value by trial and error. That is, set to a high value, send AT+BTI and if ERROR 27 is returned, then retry with a smaller value. This effective max value remains unchanged for that particular firmware build.
S519	500	100..3000	When S507>0, and in a connection, DSR can be used to change from data to command state by deasserting the DSR line for less than the time specified in this register. This value is rounded down to the nearest 100 ms.
S520	9600	1200..115200	Change to a standard baud rate. The effect is immediate and the OK is sent at the new baud rate. Only one of the following baud rates are accepted: 1200, 2400, 4800, 9600, 19200, 28800, 38400, 57600, 115200. If S register 525=1, then the maximum baud rate is limited to 115200. See S Register 526 for further information.
S521	9521	1200..921600	Change baud rate to non-standard value. Laird modules support any baud rate. The only limitation is the integer arithmetic involved, which may adjust the applied rate slightly. If the internally computed baud rate is more than 2% offset from the desired input value, then an ERROR is returned and the old baud rate prevails. To inspect the actual baud rate, do ATS521? S521 should only be used for non-standard baud rates. For standard baud rates use S520. The effect is immediate and the OK is sent at the new baud rate. If S Register 525=1, then the max baud rate is limited to 115200 In the event that a non-standard baud rate is requested, it is entirely possible that the host is not capable of generating such a baud rate. In this case the Laird Technologies device cannot be communicated with. If this happens, there is a procedure to recover from this situation which is described in section titled *Factory Default Mode*. The default is 9600 for the Laird module and 115200 for other Laird devices. See S Register 526 for further information.

Reg. #	Default	Range	Comment
S523	1	1..2	Number of Stop bits See S Register 526 for further information.
S524	0	0..2	Parity. 0=None, 1=Odd, 2=Even See S Register 526 for further information.
S525	0	0..1	Apply multiplier of 8 to baud rate internally. This is set to 0 (disabled) by default for the Laird Technologies Module/RS-232 Adaptor/Universal RS-232 Adaptor, and set to 1 (enabled) by default for the Laird Technologies PC Card. It is required in the PC Card because the UART chip on the PC Card is driven by a 14.7456MHZ crystal instead of 1.8432MHz. This means that when a host asks for a baud rate, in reality it gets a baud rate which is 8 times faster. If S Register 521 > 115200 then this register cannot be set to 1. See S Register 526 for further information.
S526	3	1..3	This register specifies a two bit mask used to qualify how S Registers 520 to 525 are actioned. When bit 0 is 1, the new comms parameter affects the UART immediately. When bit 1 is 1, the new comms parameter is stored in non-volatile memory. For example, to change comms parameters but have them come into effect only after subsequent power cycles, set this register to 2. Likewise, for an immediate effect that does not persist over a power cycle, set the value to 1. This must be set before the baud rate change.
S530	1000	100..15000	Reconnect delay when configured as master in pure-cable-replacement mode. This value is rounded down to the nearest 100 ms. See S Register 505 and 543.
S531	0	0..5	Specifies the mode on connection establishment. 0 = Normal. Data is exchanged between UART and RF 1 = LOCAL_COMMAND. UART input is parsed by the AT interpreter and RF data is discarded 2 = REMOTE_COMMAND. RF input is parsed by the AT interpreter and UART data is discarded. If S Reg 536 is not 1 then this register cannot be set to 2 and an ERROR is returned 3=LOCAL_COMMAND. UART input is parsed by the AT interpreter and incoming RF data is sent to the host using the RX<string> asynchronous response. 4=LOCAL_COMMAND and on the RF side, the GPIO is automatically sent when there is a change in input. See Section 5.7 for more details. 5=DEAMON mode
S532	0	0..7	If non zero, then on every connection, a SCO channel (audio) is initiated. Bit 0 for HV1, Bit1 for HV2, and Bit2 for HV3. When the connection is lost, the SCO channel also disappears.
S533	1	0..2	If set to 1, then GPIO5 follows RI state; if set to 2, it follows the state of DSR and if 0 it is not driven and GPIO5 is available as a user I/O. This register will not be effective immediately after changing the value. It must be saved to non-volatile memory using AT&W and will operate as expected after an ATZ or a power cycle.

Reg. #	Default	Range	Comment
S534	1	0..2	When set to 0, GPIO4 is available as user I/O It must be saved to non-volatile store using AT&W and will operate as expected after an ATZ or a power cycle.
S535	20	0..41	Link Supervision Timeout. If units go out of range, then a NO CARRIER message is sent to the host after the time specified here
S536	0	0..1	When set to 1, a remote device can capture the AT parser of this unit by it sending this module an escape "!!!" sequence. The inter character timing is set via S Register 12. If S Register 507 is >/= 2, then reading this register always returns 0 and writing 1 results in ERROR 33.
S537	X	X..X	This register is no longer available – See 551, 552, and 553 instead. It only exists in firmware version 1.1.12 to 1.1.47 The functionality it controlled is now defined by registers 551,552, and 553.
S538	0	0..1	If 1, then when a successful pairing occurs, it is automatically saved in the trusted device database, if the database has room to store it.
S539	0	0..1	When set to 1, in idle mode (S512=1), UART Rx characters are discarded if DSR is deasserted.
S540	0	0 48..127	Sets the MTU in L2CAP configuration negotiations. The value of 0 is a special value which means that the current value should remain.
S541	20	-43...20	This sets the power level in dBm when inquiring or paging. Reading this register returns the value stored in non-volatile memory.
S542	4	-43...20	As per S541, however reading this register returns the current power level as set in the base band. The read can be different from S541because the actual power is set using a lookup table and the base band rounds down to the nearest value in the table.
S543	0	0..1	If this is set to 1, then incoming pairing attempts are accepted (if a pin code has been pre-entered using AT+BTK) while in the wait phase of auto connect cycle initiated by the AT+BTR command. In addition to accepting pairing attempts, if the pairing is successful, then the new device is automatically set as the peer address for automatic connections (as if an explicit AT+BTR command was entered). See S Register 505 and 530.
S544	1	0..1	Configure the UART for either low latency or maximum throughput. A setting of 1 gives maximum throughput.

Reg. #	Default	Range	Comment
S551	0x3211	0xFFFF	This register specifies in each 4 bit nibble, how the outgoing modem status bits to the remote peer gets its value. Bluetooth allows for RTR, RTC, DV, and IC bits to be exchanged over an RFCOMM connection. Nibble 0..3 specifies the source for RTC 4..7 specifies the source for RTR 8..11 specifies the source for DV (i.e. DCD) 12..15 specifies the source for IC (i.e. RI) Each nibble can take the following value: ▪ 0 – Always set to 0 ▪ 1 – Always set to 1 ▪ 2 – If DCD (pin 8 on module connector) is output then always 1 If DCD is input then 1 if DCD is asserted otherwise 0 ▪ 3 – If RI (pin 6) is output then always 0 If RI is input then 1 if RI is asserted otherwise 0 If DSR (pin 10) is asserted then 1 otherwise 0 In the event that a nibble specifies DSR as the source of its state, be aware that if S Register 507 is anything other than 0, a de-assertion of DSR causes the Bluetooth connection to be dropped. If bits 0..3 and 4..7 are set to 0, then some Bluetooth devices will use that as a signal to stop sending any data back. For example, Nokia 6310 stops responding. If this register is changed while in command and connected mode, then on going back online using the ATO command, a fresh signal is sent to the peer to update the bits.
S552	0x0122	0x0FFF	This register specifies in each 4 bit nibble, how the DTR, DCD, RI output pins are controlled when in a Bluetooth connection Nibble 0..3 specifies the source for DTR 4..7 specifies the source for DCD 8..11 specifies the source for RI Each nibble can take the following value: ▪ 0 – Do NOT touch the I/O ▪ 1 – Always deassert ▪ 2 – Always assert ▪ 3 – If RTC bit in CONTROL_IND is 1 then assert otherwise deassert ▪ 4 – If RTR bit in CONTROL_IND is 1 then assert otherwise deassert ▪ 5 – If DV bit in CONTROL_IND is 1 then assert otherwise deassert ▪ 6 – If IC bit in CONTROL_IND is 1 then assert otherwise deassert If this register is changed while in command and connected mode, then on going back online using the ATO command, the modem output lines are refreshed.

Reg. #	Default	Range	Comment
S553	0x0201	0x0FFF	This register specifies in each 4 bit nibble how the DTR,DCD, and RI output pins are controlled when NOT in a Bluetooth connection Nibble 0..3 specifies the source for DTR 4..7 specifies the source for DCD 8..11 specifies the source for RI In addition it also refers to S Register 552 to see if the relevant pin is an input or not to be touched. If the nibble in 552 is 0, then the relevant pin is an input. Each nibble can take the following value: ▪ 0 – Always deassert ▪ 1 – Always assert ▪ 2 – Assert if RING is being sent to the host The default for the Universal RS-232 Adaptor is $0200.
S554	0	0..900	If S Register 512 >=2 and <=7 then this register specifies a time in seconds for which the device stays in the S512 mode after power up or reset. On timeout, it aborts the discoverable and/or connectable and falls back into S512=1 mode, when it is deaf and dumb. **Note:** If AT+BTR has been used to specify a peer device, then on reverting to mode 1, it attempts to make a connection to that peer device. A power cycle (reset via BREAK or ATZ) is required to see the change.
S555	1	1..7	If S Register 554 is nonzero, then after the post reset window expires, the mode reverts to the mode specified in this register. This allows, for example, the device to be discoverable and connectable on power up (mode 4 or 7) and on window timer expiry to revert to connectable only (mode 3 or 6). You must power cycle (reset via BREAK or ATZ) to see the change. In some firmware builds, S Registers 565 to 569 (inclusive) are visible, allowing the start-up mode to depend on the state of RI line (Setting S Reg 565 forces the RI pin to be configured as an input). For this feature, S Reg 565 should be set to 1. In that case, on start-up, if RI is asserted, the start-up mode is defined by S Reg 568. If deasserted then S Reg 569.
S556	0	0..3	Allows GPIO or ADC values to be read via the minor class field in an inquiry response. When this value is non-zero, bits 2 to 7 contain information as follows: ▪ 1 – ADC1 ▪ 2 – ADC2 ▪ 3 – GPIO1 to GPIO6 Set to 0 to disable this feature. This allows I/O information to be conveyed without a connection.
S557	32	4..900	Specified in seconds, the update interval for the feature enabled via S Reg 556

Reg. #	Default	Range	Comment
S558	0	0..1	When 1, the responses **RING**, **NO CARRIER**, and **CONNECT** are replaced by **BTIN**, **BTDOWN**, and **BTUP** respectively. This eliminates ambiguity when the module has a Bluetooth connection to an AT modem which also gives these responses.
S559	0	0..3	This specifies a mask. When Bit 0 is 1, the response word **ERROR** is replaced by **BTERR** and **OK** is replaced by **ok**. When Bit 1 is 1, then error responses do not include the error number; instead the error number can be retrieved using ATI12.
S560	15	15..120	Disconnect timeout in seconds. This timer specifies how long to wait for confirmation (from the peer device and/or the underlying stack) that the connection has been dropped. In some cases a confirmation does not arrive; in this case this timer is used to abort the procedure and put the state machine in a proper mode for new operations. Time is specified with 15 seconds intervals.
S561	0	0..1000	Sniff Attempt Time in units of milliseconds. 0 means disable. See "Power Consumption and Reset" in the User's Manual for details.
S562	0	0..1000	Sniff timeout Time in units of milliseconds. 0 means disable. See "Power Consumption and Reset" in the User's Manual for details.
S563	0	0..1000	Sniff Minimum Interval in units of milliseconds. 0 means disable. See "Power Consumption and Reset" in the User's Manual for details.
S564	0	0..1000	Sniff Maximum Interval in units of milliseconds. See "Power Consumption and Reset" in the User's Manual for details.
S565	0	1	If set to 1, RI (Ring Indicate) line is configured as an input and forces the start-up mode (S Reg 512)and post-timeout on Start-up mode (S Reg 555) to be dependent on the state of RI. The RI conditional modes are defined by S Regs 566 to 569 inclusive.
S566	1	7	If S565=1 and RI is asserted, this is the start mode for the device.
S567	1	7	If S565=1 and RI is deasserted, this is the start mode for the device.
S568	1	7	If S565=1 and RI is asserted, this is the mode the device assumes after the post-start-up timeout defined in S Reg 554 instead of the mode defined in S Reg 555.
S569	1	7	If S565=1 and RI is deasserted, this is the mode the device assumes after the post-start-up timeout defined in S Reg 554 instead of the mode defined in S Reg 555.
S580	0	0..1	Remote volume control feature for Headset profile when ATS102 enables headset profile.
S581	0	0..63	Lowest 6 bits of the supported features field for Handsfree profile when ATS102 enables handsfree profile. See S Reg 594 which allows you to select the HandsFree profile version.
S582	0	0..1	FTP Related: 0 = BodyLen in PUT obex packet = 0 1 = BodyLen in PUT obex packet = 1

Reg. #	Default	Range	Comment
S583	0xB	0 .. 0x1F	This specifies the initial state of the following modem control lines sent to the peer. Bit 0 – RTC (DTR/DSR) Bit 1 – RTR (RTS/CTS) Bit 2 – IC (Ring Indicate RI) Bit 3 – DV (DCD) Bit 4 – FC (Reserved)
S584	0	0..1	Enable/Disable eSCO When changing the unit returns ERROR 14, it implies the device is either in a connection or waiting for a connection and the new value cannot be accepted. For the former, drop the connection, issue the command AT+BTX, and then set the new value and issue (for the latter) the command AT+BTX prior to setting the register.
S585	0	0..9	GPIO pin set to 0 to disable the feature.
S586	1000	100..5000	Pulse period in milliseconds (rounded down to nearest multiple of 50).
S587	0	0..100	Duty cycle in percentage (rounded to the nearest multiple of four).
S588	0	0..1	After a disconnection, there is a cold reset.
S589	8	0..F	Codec output gain.
S590	1	0..3	Codec input gain.
S591	0	0..1FF	Default GPIO output states when not in a connection. This is used when virtual digital I/O cable replacement mode is in operation.
S592	0	0..1	Set this to 1 to reduce the trusted device database to one record when autosaving of pairing is enabled via S Reg 538.
S593	0	0..1	Automatically append last six digits of local Bluetooth address to the friendly name which was set via AT+BTN or AT+BTF.
S594	0	0..1	Set handsfree profile version in SDP record. Set to 0 for 1.1 and to 1 for 1.5.
S595	1	0..1	Set handsfree gateway profile version in SDP record. Set to 0 for 1.1 and to 1 for 1.5.
S596	0	1..1FF	Audio Gateway features to be advertised in SDP record. See handsfree profile specification for exact bit mapping.
S597	0	0..2	Audio Gateway mode: ▪ 0 – SDP record advert only ▪ 1 – Hosted operation ▪ 3 – Hostless operation See Audio Gateway specific documentation for more details.
S598	0	0..1	In hostless audio gateway serviced mode, if this is 1, incoming voice calls are reflected to bonded headset.
S599	0	0..2	SCO control for hostless gateway operation. ▪ 0 – Normal ▪ 1 – As early as possible ▪ 2 – Leave SCO to be controlled by headset

Reg. #	Default	Range	Comment
S600	?	0..65535	Number of times this module has gone through a reset cycle. This feature is enabled by S Reg 601. Writing any value to this register initialises it to a certain value.
S601	0	0..1	If this is 1, then on reset S Reg 600 value is incremented.
S610	0	0..7FFF	Set direction of digital I/O lines. This is a mask made up of 5 bits. Setting a bit to 1 makes that I/O line an output. GPIO1 is bit 0, GPIO2 is bit 1, up to bit 8 for GPIO9.
S611	0	1	Set to 1 to invert the logic of GPIO outputs. For example, ATS621=1 sets the output pin to low and vice versa.
S620	n/a	0..31	Read/Write to all eight digital lines in one atomic step. The value is returned as a four digit hexadecimal value with trailing 0s.
S621	n/a	0..1	Read/Write to GPIO1.
S622	n/a	0..1	Read/Write to GPIO2.
S623	n/a	0..1	Read/Write to GPIO3.
S624	n/a	0..1	Read/Write to GPIO4.
S625	n/a	0..1	Read/Write to GPIO5.
S626	n/a	0..1	Read/Write to GPIO6.
S627	n/a	0..1	Read/Write to GPIO7.
S628	n/a	0..1	Read/Write to GPIO8.
S629	n/a	0..1	Read/Write to GPIO9.
S631	n/a	0..65535	When GPIO1 is configured as an input, low to high transitions are counted. There is no software debouncing. External RC circuit may be required. The counter wraps to 0 when it overflows beyond 65535.
S632	n/a	0..65535	When GPIO2 is configured as an input, low to high transitions are counted. There is no software debouncing. External RC circuit may be required. The counter wraps to 0 when it overflows beyond 65535.
S641	n/a	0..65535	As per 631, but the action of reading the value resets the count to 0.
S642	n/a	0..65535	As per 632, but the action of reading the value resets the count to 0.
S701	n/a	0..65535	Read Analogue Line 0. Value is returned in decimal .
S702	n/a	0..65535	Read Analogue Line 1. Value is returned in decimal
S711	n/a	0000..FFFF	Read Analogue Line 0. Value is returned in hexadecimal
S712	n/a	0000..FFFF	Read Analogue Line 1. Value is returned in hexadecimal
S721	0	0	Set direction of Analogue Line 0.
S722	0	0	Set direction of Analogue Line 1.
S1001 to S1010		0.. 2^32	Ten General Purpose 32 bit Registers for use by host. These are stored in non-volatile memory.

1.3.14 ATSn?{Read S Register Value}

This returns the current value of register n.

For recognised values of n:

Response: <cr,lf>As Appropriate<cr,lf>OK<cr,lf>

For unrecognised values of n:

Response: <cr,lf>ERROR nn<cr,lf>

1.3.15 ATSn=?{Read S Register – Valid Range}

This returns the valid range of values for register n.

For recognised values of n:

Response: <cr,lf>Sn:(nnnn..mmmm)<cr,lf>OK<cr,lf>

For unrecognised values of n:

Response: <cr,lf>ERRORnn<cr,lf>

1.3.16 ATX<string>{Send Data in Local Command and Connected Mode}

This command is used to send data to the remote device when in local command and connected mode.

The parameter <string> is any string not more than 24 characters long. If a non-visual character is to be sent, then insert the escape sequence \hh where hh are two hexadecimal digits. The 3 character sequence \hh is converted into a single byte before transmission to the peer.

Response: <cr,lf>OK<cr,lf>

1.3.17 ATY<string>{Send Data in Local Command and Connected Mode}

This command is similar to ATX in syntax and functionality, except that the string is only copied to the output RF buffer. Only when an empty string is presented are all pending data in the output RF buffer flushed out.

The parameter <string> is any string not more than 24 characters long. If a non-visual character is to be sent then insert the escape sequence \hh where hh are two hexadecimal digits. The three character sequence \hh is converted into a single byte before transmission to the peer.

Response: <cr,lf>OK<cr,lf>

1.3.18 ATZ<n>{Hardware Reset and emerge into mode 'n'}

Forces the device through a hardware reset which means it eventually comes alive in the local command and unconnected mode. This allows changes to the PS store to take effect. Prior to version 2.7.0, allow for approximately two seconds for the device to once again start responding to AT commands. The best way to determine if the device is alive is to keep sending it AT<cr> until it responds with an OK response. After version 2.7.0, it is safe to communicate after receiving an OK.

The optional parameter <n> is only available for firmware 2.7.0 and newer and is a value in the range 0 to 7 (up to version 7.18.0). After version 9.18.6, valid values are 0 to 4 (inclusive).

ATZ and ATZ0 signify reset and emerge into the current mode (see command ATI14). ATZ1 to ATZ4 instructs the module to reset and then emerge into the appropriate boot mode.

Note: S Reg 103 specifies the boot mode from cold.

For firmware prior to v2.7.0:

Response: <cr,lf>OK<cr,lf>

Note: OK is returned before the RESET.

For firmware v2.7.0 and newer:

Response: <cr,lf>OK<cr,lf>

Note: OK is returned after the RESET.

1.3.19 AT&Fn{Set S Register Defaults}

This command only works when the device is in local command and unconnected mode. Depending on the value of 'n', it installs S Register values appropriate for various power modes, ranging from minimum power consumption to maximum.

See Table 1-3 for the legal values of 'n'. All other values of n generate a syntax error response. If 'n' is not specified, then a default value of 0 is assumed where the BAUD rate is NOT changed.

Table 1-3: Legal values of 'n"

&F0 (Default)	Medium power consumption, UART baud rate unchanged
&F1	Minimum power consumption, UART baud rate set to 9600
&F2	Minimum power consumption, UART baud rate set to 38400
&F3	Minimum power consumption, UART baud rate set to 115200
&F4	Medium power consumption, UART baud rate set to 115200
&F5	Maximum power consumption, UART baud set to 115200
&F6	Maximum power consumption, UART baud set to 115200 Explicitly set higher baud rates using ATS521=n

Refer to the "Power Consumption" chapter in the relevant Laird User's Manual for more detailed information of power usage.

The new values are NOT updated in non-volatile memory until the AT&W command is sent to the Laird device.

Response: <cr,lf>**OK**<cr,lf>

Or

Response: <cr,lf>**ERROR nn**<cr,lf>

1.3.20 AT&F*{Clear Non-volatile Memory}

The AT&F* variant of the command installs values in S registers as per command AT&F4 and then all other user parameters in non-volatile memory are erased. This means that the trusted device database is cleared, as well as parameters related to the following commands: AT+BTR, AT+BTN, AT+BTS.

Response: <cr,lf>**OK**<cr,lf>

Or

Response: <cr,lf>**ERROR nn**<cr,lf>

1.3.21 AT&F+{Clear Non-volatile Memory}

This command erases all user parameters in non-volatile memory except S Registers 520 to 525. This means that the trusted device database is cleared, and so are parameters related to the following commands:- AT+BTR, AT+BTN, AT+BTS.

Response: <cr,lf>**OK**<cr,lf>

Or

Response: <cr,lf>**ERROR nn**<cr,lf>

1.3.22 AT&W{Write S Registers to Non-volatile Memory}

Writes current S Register values to non-volatile memory so that they are retained over a power cycle.

Response: <cr,lf>**OK**<cr,lf>

Or

Response: <cr,lf>**ERROR nn**<cr,lf>

1.3.23 AT+BTAn{Control Audio Channel}

Once a Bluetooth connection is active, *and assuming the peer device is a Laird device,* this command is used to start/stop a SCO channel which connects the PCM interfaces of the two peer devices. If a codec is attached to the PCM pins, then 2-way audio can be established.

+BTA0	Switch off the channel.
+BTA1	Switch on the channel.

On receipt of the command, the following response immediately follows.

Response: <cr,lf>**OK**<cr,lf>

The lower layers then go through the process of setting up the SCO channel. Once a SCO link is established, the following response is asynchronously sent to the host.

Response: <cr,lf>**AUDIO ON**<cr,lf>

Or if the SCO failed to be established.

Response: <cr,lf>**AUDIO FAIL**<cr,lf>

On the peer device, the host will asynchronously receive the following response.

Response: <cr,lf>**AUDIO ON**<cr,lf>

1.3.24 AT+BTC<devclass>{Set Device Class Code}

This command is used to set the device class code which is sent in subsequent inquiry responses. It can be read back using the AT+BTC? command as described below.

<devclass> is a six digit hexadecimal number derived as per section "1.2 The Class of Device/Service Field" of the Bluetooth specification "Bluetooth Assigned Numbers".

The 24 bits are made of the following four fields (bit 0 corresponds to the least significant bit):

Bits 0-1:	Format Type. This field currently only has a value of 00 (i.e. format type 1).
Bits 2-7:	These 6 bits define the Minor Device Class and the value is interpreted differently based on the Major Device class stored in the next 5 bits.
Bits 8-12:	These 5 bits define the Major Device Class as per Table 1.3 in "Bluetooth Assigned Numbers".
Bits 13-23:	This is an 11 bit field used as a mask to define the Major Service Class, as per Table 1.2 in "Bluetooth Assigned Number".

Laird devices do not map to any predefined Major Service Class or Major Device Class. The default devclass as shipped is 001F00 which means no Major Service Class and "Unclassified" Major Device class.

Table 1-4 shows examples of device class codes.

Table 1-4: Device class codes

Code (Hex)	Name	Major Service	Major Device	Minor Device
001F00	Unclassified	None	Unclassified	n/a
200404	Headset	Audio	Audio	Headset

Response: <cr,lf>OK<cr,lf>

Or for an invalid <devclass> value (usually a value which is not six hexadecimal characters long).

Response: <cr,lf>ERROR 08<cr,lf>

1.3.25 AT+BTC?{Read Device Class Code}

This command is used to read the current device class code.

Response: <cr,lf>123456<cr,lf>OK<cr,lf>

1.3.26 AT+BTD<bd_addr>{Remove Trusted Device}

This command is used to remove the specified device from the list of trusted devices in the non-volatile database. If the device is not in the database, the response is still OK.

Response: <cr,lf>OK<cr,lf>

1.3.27 AT+BTD*{Remove All Trusted Devices}

WARNING: This command is used to remove all devices from the list of trusted devices in the non-volatile database. The software does not ask for confirmation.

WARNING: If you make an authenticated connection, the link key gets cached in the underlying stack. If you subsequently delete the key using AT+BTD* and immediately request an authenticated connection to the same device, then the connection will be established. To ensure this does not happen, either send ATZ after the AT+BTD* OR send AT+BTD<bd_addr> for each item in the trusted device database.

Response: <cr,lf>OK<cr,lf>

1.3.28 AT+BTF=<string>{Set Friendly Name}

This sets the friendly name of this device as seen by other devices.

Response: <cr,lf>OK<cr,lf>

1.3.29 AT+BTF<bd_addr>{Get Remote Friendly Name}

This command gets the remote friendly name of the specified peer.

Response: <cr,lf><bd_addr>,"Friendly Name"
<cr,lf>OK<cr,lf>

1.3.30 AT+BTG<bd_addr>{Enable Cautious Page Scanning ONLY}

Enable page scanning and wait for a connection from device with Bluetooth address <bd_addr>. If the specified address is 000000000000 then incoming connections are accepted from any device (like AT+BTP without an address). Inquiry Scans are disabled.

This command also has variants which allow authentication and encryption to be explicitly specified. For example:

AT+BTGU123456789012
AT+BTGY123456789012
AT+BTGUY123456789012
AT+BTGYU123456789012

Response: <cr,lf>**OK**<cr,lf>

1.3.31 AT+BTG{Enable Promiscuous Page Scanning ONLY}

Enable page scanning only and wait for a connection from any device. Inquiry scans are disabled. Authentication and encryption is as per S registers 502 and 503.

Response: <cr,lf>**OK**<cr,lf>

1.3.32 AT+BTGU{Enable Promiscuous Page Scanning ONLY}

Enable page scanning only and wait for a connection from any device. Inquiry scans are disabled. Authentication is enabled and encryption is disabled.

Response: <cr,lf>**OK**<cr,lf>

1.3.33 AT+BTGY{Enable Promiscuous Page Scanning ONLY}

Enable page scanning only and wait for a connection from any device. Inquiry scans disabled. Authentication is disabled and encryption is enabled.

Response: <cr,lf>**OK**<cr,lf>

1.3.34 AT+BTGUY{Enable Promiscuous Page Scanning ONLY}

Enable page scanning only and wait for a connection from any device. Inquiry scans are disabled. Authentication and encryption are both enabled. The order of U and Y is not significant.

Response: <cr,lf>**OK**<cr,lf>

1.3.35 AT+BTI<devclass>{Inquire}

This makes the device perform an inquiry for device class code for **delay** milliseconds and **max** number of unique responses, where **delay** is specified by S register 517 and **max** is specified by S register 518.

The <devclass> is an optional parameter where the value specifies either a six digit device class code or a two digit major device class. If it is not specified, the value is taken from S register 516.

When <devclass> is six hexadecimal characters long, it specifies an AND mask which is used to filter inquiry responses. When <devclass> is two hexadecimal characters long, it forces the inquiry to filter responses to devices that match their major device class code to this value (can only be in the range 00 to 1F).

Response: <cr,lf>12346789012

<cr,lf>12345678914

<cr,lf>OK<cr,lf>

If the module is waiting for an incoming connection, (entered via AT+BTP, AT+BTG, AT+BTQ), then it responds with ERROR 14. To perform the inquiry, send AT+BTX to put the module back into idle mode.

Response: <cr,lf>ERROR 14<cr,lf>

ERROR RESPONSE

For a single inquiry request with the Bluetooth inquiry process, a device could respond many times. To ensure that an address is sent to the host only once for a particular AT+BTI, an array of addresses is created in dynamic memory at the start of each AT+BTI and is filled as responses come in. So in a single AT+BTI the same Bluetooth address will not be returned more than once, however it may be returned on subsequent inquiries.

If the memory allocation fails, the inquiry procedure is aborted. If that occurs, an error response is sent to the host.

Response: <cr,lf>ERROR 27<cr,lf>

1.3.36 AT+BTIV<devclass>{Inquire}

As per AT+BTI but the response includes the device class code for all inquiry responses. Refer to the 'ERROR RESPONSE' note in the description for AT+BTI<devclass>.

Response: <cr,lf>12346789012,123456

<cr,lf>12345678914,123456

<cr,lf>OK<cr,lf>

1.3.37 AT+BTIN<devclass>{Inquire}

As per AT+BTI but the response includes the device class code and friendly name for all inquiry responses. Please refer to the 'ERROR ESPONSE' note in the description for AT+BTI<devclass>. The friendly name strings are in UTF-8 format as per the Bluetooth specification.

Response: <cr,lf>12346789012,123456,"TDK SYSTEMS AT DONGLE 1"

<cr,lf>12345678914,123456, "TDK SYSTEMS RS232"

<cr,lf>OK<cr,lf>

Note: Many releases of firmware return the product name as LAIRD TECHNOLOGIES.

Response: <cr,lf>12346789012,123456,"TDK SYSTEMS AT DONGLE 1"

<cr,lf>12345678914,123456, "TDK SYSTEMS RS232"

<cr,lf>OK<cr,lf>

Note: We strongly recommend that any software implementation that uses this command should check for any of Laird, EZURIO, and TDK SYSTEMS to ensure backward and forward compatibility.

1.3.38 AT+BTK=<string>{Set Passkey}

This command is used to provide a passkey when **PIN? 12345678** indications are received asynchronously. If a pairing is not in progress then the pin is written to non-volatile memory for future use. Specifying an empty string deletes the key from the non-volatile memory.

The string length must be in the range 0 to 8, otherwise an error is returned.

Response: <cr,lf>**OK**<cr,lf>

1.3.39 AT+BTM<bd_addr>{Set Incoming Peer Address}

This command is used to store a peer address for incoming connections in non-volatile memory. A value of 000000000000 has the special meaning of invalid peer address.

When S register 512 = 3, 4, 6, or 7 it waits for an incoming connection from the peer address specified. If the peer address is not 000000000000, then it waits for a connection from the specified master, otherwise it connects to anyone.

Response: <cr,lf>**OK**<cr,lf>

1.3.40 AT+BTM{Delete Incoming Peer Address}

This command is used to delete the peer address previously stored using AT+BTMbd_addr>.

Response: <cr,lf>**OK**<cr,lf>

1.3.41 AT+BTM?{Read Incoming Peer Address}

This command is used to display the peer address stored in non-volatile memory, used to put the module in pure cable replacement mode.

Response: <cr,lf>**12346789012**
<cr,lf>**OK**<cr,lf>

If the location is empty the response is as follows.

Response: <cr,lf>**00000000000**
<cr,lf>**OK**<cr,lf>

1.3.42 AT+BTN=<string>{Set Friendly Name in Non-volatile Memory}

This sets the default friendly name of this device as seen by other devices. It is stored in non-volatile memory. Use AT+BTF to make the name visible to other devices. Use AT+BTN? To read it back. An empty string (" ") deletes the string from non-volatile memory which forces use of the default name.

Response: <cr,lf>**OK**<cr,lf>

1.3.43 AT+BTN?{Read Friendly Name from Non-volatile Memory}

Read the default friendly name from non-volatile memory.

Response: <cr,lf>**"My FriendlyName"**<cr,lf>
<cr,lf>**OK**<cr,lf>

1.3.44 AT+BTO<devclass>{Open and make Unit Detectable}

After power up and ATZ, this command is sent to ensure that the RFCOMM is initialised and opened and the service name as specified in AT+BTN is exposed via the SDP registry.

The <devclass> value specifies an optional fixed length hexadecimal device class code. If it is not specified, then the device class code is taken from S Register 515.

For this command to be effective, S Register 512 must be set to zero.

Response: <cr,lf>OK<cr,lf>

1.3.45 AT+BTP<bd_addr>{Enable Cautious Page/Inquiry Scanning}

Enable page scanning and wait for a connection from device with Bluetooth address <bd_addr>. If the specified address is 000000000000 then incoming connections are accepted from any device, is as per AT+BTP without an address. Inquiry scanning is also enabled.

This command has variants which allow authentication and encryption to be explicitly specified. For example:

- AT+BTPU123456789012
- AT+BTPY123456789012
- AT+BTPUY123456789012
- AT+BTPYU123456789012

Response: <cr,lf>OK<cr,lf>

1.3.46 AT+BTP{Enable Promiscuous Page/Inquiry Scanning}

Enable page scanning and wait for a connection from any device. Inquiry scanning is also enabled. Authentication and encryption is as per S registers 502 and 503.

Response: <cr,lf>OK<cr,lf>

1.3.47 AT+BTPU{Enable Promiscuous Page/Inquiry Scanning}

Enable page scanning and wait for a connection from any device. Inquiry scanning is also enabled. Authentication is enabled and encryption is disabled.

Response: <cr,lf>OK<cr,lf>

1.3.48 AT+BTPY{Enable Promiscuous Page/Inquiry Scanning}

Enable page scanning and wait for a connection from any device. Inquiry scanning is also enabled. Authentication is disabled and encryption is enabled.

Response: <cr,lf>OK<cr,lf>

1.3.49 AT+BTPUY{Enable Promiscuous Page/Inquiry Scanning}

Enable page scanning and wait for a connection from any device. Inquiry scanning is also enabled. Authentication and encryption are both enabled. The order of U and Y is not significant.

Response: <cr,lf>OK<cr,lf>

1.3.50 AT+BTQ{Enable Inquiry Scans ONLY}

When inquiry scan is enabled, it implies that this device responds to inquiries from other devices. Use AT+BTX to disable inquiries.

Response: <cr,lf>**OK**<cr,lf>

1.3.51 AT+BTR<bd_addr>{Set Outgoing Peer Address}

This command is used to store a peer address for outbound connections in non-volatile memory. A value of 000000000000 has the special meaning of invalid peer address.

This command is used to set up a module in pure cable replacement mode.

If S register 512 = 1 and the peer address is not 000000000000, then it periodically (time specified via S register 505) attempts to connect to the peer address specified. In this circumstance, all commands from the host are buffered in the receive buffer until a Bluetooth connection is established with the peer device and it then sends the buffered commands across. This means that if the peer device is not in the vicinity and will never be there, the device effectively becomes useless, as in this circumstance a host would want to get attention of the AT parser to send it new commands – probably one to delete the peer device.

In this circumstance, a recovery is possible by one of two methods. The first method assumes that the DTR from the host is connected to the DSR line of the module and the second method assumes that this connection is absent. In the first method it is enough to deassert the DTR line from the host and that will abort the autoconnect cycle. The second method is initiated by resetting the device and then ensuring that the text string "AT+BT&BISM&<cr>" is sent (where <cr> is the carriage return character). There is special code which looks out for this magic command and terminates the autoconnect cycle if it sees it and confirms to the host of that fact by sending an "OK" response.

Response: <cr,lf>**OK**<cr,lf>

1.3.52 AT+BTR{Delete Outgoing Peer Address}

This command is used to delete the peer address previously stored using AT+BTR<bd_addr>.

Response: <cr,lf>**OK**<cr,lf>

1.3.53 AT+BTR?{Read Outgoing Peer Address}

This command is used to display the peer address stored in non-volatile memory, used to put the Laird device in pure cable replacement mode.

Response: <cr,lf>**12346789012**
<cr,lf>**OK**<cr,lf>

If the location is empty the response is as follows.

Response: <cr,lf>**00000000000**
<cr,lf>**OK**<cr,lf>

1.3.54 AT+BTS=<string>{Set Service Name}

This writes the name to non-volatile memory. It is used after ATZ, power cycle, or AT+BTO if it has not yet been issued. Use **AT+BTS?** to read it back from non-volatile memory. An empty string ("") deletes the string from non-volatile memory which forces the default service to be used.

Response: <cr,lf>**OK**<cr,lf>

If the service name cannot be set for any reason then an error response **ERROR 11** is returned.

1.3.55 AT+BTS?{Read Service Name from Non-volatile Memory}

Reads the default service name from non-volatile memory.

Response: <cr,lf>**"My ServiceName"**<cr,lf>
<cr,lf>**OK**<cr,lf>

1.3.56 AT+BTT{Add Trusted Device}

This command is used to store the cached link key in the non-volatile database. If the database is full it responds with an ERROR. If the device is already in the database, then the key is replaced.

If the link key cache is empty (a pairing has not been performed since the device was powered) then the response is an ERROR.

Response: <cr,lf>**OK**<cr,lf>

Or

Response: <cr,lf>**ERROR**<cr,lf>

1.3.57 AT+BTT?{List Trusted Device}

This command is used to list the contents of the trusted device database. The link key is not displayed so the response is as shown below. If the list is empty then just the OK response is sent otherwise an OK is used to terminate the list. Use the command ATI6 to read the maximum size of the trusted device database.

Response: <cr,lf>**12346789012**
<cr,lf>**12345678913**
<cr,lf>**12345678914**
<cr,lf>**OK**<cr,lf>

1.3.58 AT+BTV<U><Y><bd_addr>,<uuid>{SDP Query for Service }

This command is used to interrogate the SDP database of the peer device <bd_addr> for the service <uuid>. It results in an ACL connection and then a SDP transaction.

If the <uuid> service is present then

Response: <cr,lf>**0**
<cr,lf>**OK**<cr,lf>

If the <uuid> service is not present then

Response: <cr,lf>**1**
<cr,lf>**OK**<cr,lf>

If the device <bd_addr> cannot be reached, or is in non-connectable mode then

Response: <cr,lf>**2**
<cr,lf>**OK**<cr,lf>

If the SDP database is corrupt or invalid then

Response: <cr,lf>**3**
<cr,lf>**OK**<cr,lf>

If the device is not in idle mode then

Response: <cr,lf>**4**

<cr,lf>**OK**<cr,lf>

and in this case, the command AT+BTX may put the device into the correct idle mode.

1.3.59 AT+BTW<bd_addr>{Initiate Pairing}

This initiates pairing with a device whose Bluetooth address is <bd_addr>. An OK response is sent and when the PIN is required, asynchronous indications are sent to the host in the form **PIN? <bd_addr>** where the address confirms the device with which the pairing is to be performed. To supply a PIN, use AT+BTK.

For a successful pairing, the link key is stored in a volatile cache which is overwritten each time a new pairing is initiated using this command. The link key can be stored in a non-volatile database within the device. The list of trusted devices is managed using commands AT+BTT?, AT+BTT, and AT+BTD. The AT+BTT? command produces a list of trusted Bluetooth addresses (link key is *never* displayed) and AT+BTT is used to store the cached link key. The command AT+BTD123456789012 is used to remove the specified device from the database.

The OK response is sent immediately upon receipt of the AT+BTW command. On pairing completion, an unsolicited message is sent to the host which is in the form PAIR n <bd_addr>.

If AT+BTI, AT+BTP, AT+BTG, AT+BTQ, or ATD is issued between the AT+BTW command and the subsequence PAIR asynchronous response, then an ERROR response is sent to those commands as the device is not in a mode from where such commands can be actioned.

Response: <cr,lf>**OK**<cr,lf>

1.3.60 AT+BTW?{List Cached Trusted Device}

This command is used to list the cached trusted device.

Response: <cr,lf>**12346789012**

<cr,lf>**OK**<cr,lf>

If the cache is empty the response is as follows.

Response: <cr,lf>**OK**<cr,lf>

1.3.61 AT+BTX{Disable Page/Inquiry Scanning}

Disable page/inquiry scanning. This means it is accept incoming connections or inquiry requests. In fact, this negates the effect of AT+BTQ, AT+BTG and AT+BTP commands.

Response: <cr,lf>**OK**<cr,lf>

1.3.62 AT+AG<command><parm>{Audio gateway Control}

See audio gateway specific specification for more details.

1.4 Unsolicited Responses

The 'AT' Protocol is a command/response type of protocol. This means that the Laird device will normally only respond to AT commands.

Under special circumstances, unsolicited responses are sent to the host. They are described in the following subsections.

1.4.1 RING

This string is sent to the host when a remote device is initiating a serial port connection. The fully qualified string is in the form RING 012345678901 where 012345678901 is a 12 digit hexadecimal number which corresponds to the remote device's Bluetooth address. This response is sent to the host every two seconds until the host either accepts the connection using the ATA command or rejects it using the ATH command.

1.4.2 PIN?

This response is sent to the host during a pairing negotiation.

The fully qualified string is PIN? 012345678901 where 012345678901 is the Bluetooth address of the peer device. In response, the host must supply a pin code which is entered using the AT+BTK command.

If the peer address does not supply the address in the message exchange, then the address is specified as 000000000000 – and the paring will proceed as normal.

1.4.3 AUDIO ON

This response is sent to the host when a SCO channel has been established.

1.4.4 AUDIO OFF

This response is sent to the host when an existing SCO channel has been closed.

1.4.5 AUDIO FAIL

This response is sent to the host when a SCO channel setup fails.

1.4.6 ERROR 27

This response is sent to the host on power up if the firmware is unlicensed.

1.4.7 PAIR n <bd_addr>

This response is sent to the host on termination of a pairing process. If pairing was successful then 'n' = 0, if a timeout occurred then 'n'=1 and for all other unsuccessful outcomes the value is 2.
The parameter <bd_addr> is the address of the peer device if available.

1.4.8 PAIR 0 <bd_addr> MM

This response is sent to the host on termination of a successful pairing process. The optional MM is sent only if S Register 538 is set to 1 to automatically save the link key. The value MM indicates the result of the save operation and a value of 00 implies success, otherwise the value corresponds to an error code.

1.4.9 RX<string>

This response is sent to the host when the unit is in online-command mode and S Register 531 is set to 3 and data arrives from a peer.

If the data from the string contains non-visual characters (for example ASCII 0 to 31 and ASCII 128 to 255), then those characters are translated into a three character escape sequence starting with '\'. For example the embedded <cr><lf> sequence would be sent as the six character string \0D\0A.

If the data contains the character '"' then it is sent as \22.

If the data contains the character '\' then it is sent as \5C.

1.4.10 AG<string>

This response is sent to the host when a serviced audio gateway connection is in progress and the profile requires some action from the host.

1.5 Incoming Connections

The Laird device can be configured using the AT+BTP or AT+BTG command so that it scans for incoming connections from other Bluetooth devices. It can also be configured via S Register 512 to be in this mode by default on power up.

When the lower layers detect an incoming call, a RING 123456789012 string is sent to the host every second. The command ATA is used to accept the connection and ATH to reject it.

On connection, if the S0 Register is >=0 then confirmation to the host is in the form:

- CONNECT 123456789012
- CONNECT 123456789012 A
- CONNECT 123456789012 E
- CONNECT 123456789012 AE

(A = Authenticated connection; E = Encryption enabled)

When S0 register is -1, neither RING nor CONNECT is sent to the host and the connection is silently accepted.

If the S 100 register is non-zero, then after the ring indications specified by this register have been sent to the host, and the host has failed to accept or reject the incoming connection, then an automatic 'hangup' is initiated.

1.6 Dropping Connections

In a conventional telephony modem, a call is normally terminated by first sending a +++ character sequence enveloped by an escape sequence guard time (of the order of 100 to 1000 milliseconds) to enter local command and connected mode and then the ATH command.

Laird Bluetooth modules provide a variety of ways of dropping a connection. One method is similar to the above, but instead a ^^^ character sequence is used; this eliminates ambiguity when a data call is in progress via a mobile phone which was established using the mobile phone's Bluetooth AT modem. The second method involves the host dropping the DTR (DSR from the module's viewpoint) handshaking line.

Being able to drop a connection using the escape sequence ^^^ has a severe penalty on data throughput. In fact, the data rate is of the order of 85 kbps instead of about 200 kbps. To cater for this performance hit, the device's connection drop capability is configurable to be in one of two modes.

One mode allows for a connection to be dropped using either method; the other mode allows for a connection drop using the DTR method only. By default, the device is in the former mode. This mode is selected using the S507 register.

To reiterate, the escape sequence is as follows:

<Guard time><Esc Chr><Guard time><Esc Chr><Guard time><Esc Chr><Guard time>

Even when a file transfer is occurring and it happens to be full of <Esc Chr> characters, it is not going to drop into command mode because, when transferring a file, it is going to occur as fast as possible; this means that the inter character gap is going to be significantly shorter than the <Guard time>.

The <Esc Chr> character can be changed via the S2 register and the <Guard time> interval can be specified via the S12 register.

1.7 Pairing and Trusted Devices

When authentication is enabled via S register 500 or when using the 'u' modifier in the ATD and AT+BTP commands, a connection attempt requires a link key for the peer device. The link key can be obtained prior to connection by invoking the AT+BTW and AT+BTK commands. A new link key can be obtained as often as required and is stored in a volatile cache. At any time, this cached link key can be added to the trusted devices database using the AT+BTT command. A trusted device can be deleted using the AT+BTD command. To view a list of trusted device, issue the command AT+BTT?.

In addition, if S Register 538 is set to 1, then on a successful pairing, the link key is automatically saved to the trusted device database. In that case, the asynchronous message PAIR 0 <bd_addr> has an error code appended at the end to convey the result of the save operation.

When a connection attempt requires a link key, the trusted device database is searched automatically and if one exists, it is provided without host interaction. If the link key is not present, then the connection attempt is terminated and a NO CARRIER response is given to the ATD command.

A typical session to pair an Ericsson T68i (for example) to a serial module would be:

1. Make the T68i discoverable and send AT+BTI to the serial module. This results in inquiry responses from all devices. Make a note of the Bluetooth address of the phone e.g. 123456789012.
2. On the T68i start pairing procedure by selecting "Phone accepts" in the relevant Bluetooth menu.
3. Send command AT+BTW123456789012 to the serial module.
4. Confirm that you get an OK response and then PIN? responds on a two second interval.
5. Enter a pin code on the phone (such as 12345768).
6. Enter the command AT+BTK="12345678". The phone confirms success and likewise the serial module responds with OK.
7. On success, the serial module sends an unsolicited message in the form of PAIR 0 <bd_addr>.
8. Send AT+BTT to the serial module so that the pairing information is stored in the non-volatile database.
9. Confirm that the link key has been stored by sending the command AT+BTT?. This results in a list of all devices paired with the module.

   If two Laird devices need to be paired, then it can be accomplished as follows:

   - To device 1 send ATI4, it responds with the local Bluetooth address (e.g., 123456789001).
   - To device 1 send AT+BTP. It becomes discoverable and connectable.
   - To device 2 send AT+BTW123456789001 and it responds with OK.
     Both devices display PIN? asynchronous responses.
   - To both modules. send AT+BTK="12345678".
   - On success, the serial module sends an unsolicited message in the form of PAIR 0 <bd_addr>
   - The pairing link key is now in volatile memory; send AT+BTT to both.
   - The two units now have pairing information which will survive a power cycle.

1.8 Error Responses

All error responses from the Laird device are in the form <cr,lf>**ERROR nn**<cr,lf>, where nn is a number in the range 00 to 99.

Error	Description
01	Register not recognised
02	Value for register is out of range
03	Incoming call *not* pending
04	No call to connect to. This error code has meaning for ATO only
05	Syntax error

Error	Description
06	Empty string
06	Device class could not be stored
08	Invalid Device Class code
09	Invalid Bluetooth address
10	Could not set Service or Friendly name
11	PS Store Write
12	PS Store Read
13	Not Idle
14	Incorrect mode
15	Already scanning
16	Pairing is already in progress
17	Not USED
18	Not USED
19	Not USED
20	Not safe to write to Non-volatile Store - Ongoing Bluetooth Connection
21	Link Key Cache is empty
22	Link Key Database is full
23	Malloc returned NULL - Resource Issue
24	Remote Address same as Local Address
25	Connection Setup Fail, DSR Not asserted
26	Unauthenticated licence
27	Max Responses (See S Register 518) too high. Memory allocation error
28	The length of Pin in AT+BTK is too long
29	Invalid Ring count specified for S Register 0 or 100. If S0<>0 and S100<>0 then S0 must be < S100
30	ADC Error
31	Analogue Value cannot be read as it is set for output
32	Analogue Value cannot be written as it is set for input
33	S Register value is invalid
34	Both L and R modifier cannot be specified in ATD command
35	Invalid Major Device class – valid value in range 0x00 to 0x1F inclusive
36	Pairing in progress – Command cannot be actioned – try again later
37	Invalid Sniff parameter specified. E.g. new Attempt value greater than MinInterval. Solution is to first increase MinInterval and re-enter the Attempt value.
38	Get Remote Friendly name Failed
39	Failed to change mode to Multipoint
40	7 Bit mode requires parity to be even or odd
41	Stream error
42	Stream pending

Error	Description
43	Unknown AG command
44	Busy try later
45	Not allowed – call waiting has not been enabled by peer or in hostless mode
46	GPIO line can't be set as it has not been configured as an output

1.9 Factory Default Mode

Laird devices are capable of operating at a wide range of baud rates. S Registers 520 and 521 allow the baud rate to be easily set. The baud rate clock generator in the Laird device is more versatile than one available in a standard 16550 UART commonly available in PCs.

In fact, as long as the equation BAUDRATE * 0.004096 produces an integer value, there is 0% error in clocking for that baud rate.

Because of this, it is possible to set a baud rate that a PC cannot cope with and, in that circumstance, it is virtually impossible to communicate with it.

For this type of circumstance, the Laird device comes out of reset using 9600,N,8,1 comms settings for exactly 750 milliseconds and then reverts to the comms parameters as per the S Registers.

If the host sends the string **!<BISM>!<cr>** where <cr> is the carriage return character within that 750 ms period, then the module remains at 9600,N,8,1 and configures itself using factory default S Register values.

1.10 Miscellaneous Features

This chapter describes various features which cannot be categorized appropriately.

1.10.1 RI dependent Start-up Mode

The UART_RI line can be configured as an input and on power its state can be used to force the device into one of two modes. See description for S Registers 565 to 569 inclusive for more details.

For example, the feature could allow a device to make an outgoing connection if RI is in one state, and be ready for an incoming connection in the other.

1.10.2 Pulse a GPIO pin

To flash a GPIO pin, set it as an output using S reg 610 and then use S reg 585 to 587 inclusive to set the pin, period, and duty cycle respectively.

1.10.3 Flash LED on Connectable Mode

S reg 534 now takes a value up to two. A value of two configures it so that it blinks when the module is in connectable mode.

1.10.4 Reset via BREAK

The module can be reset by sending a BREAK signal. A BREAK signal exists when the module's UART_RX input is in a non-idle state (0v) for more than 125 milliseconds.

1.10.5 Digital I/O Cable Replacement

The module has a number of general purpose digital I/O pins. The direction of these is specified via S 610.

When S Reg 531 is set to four at both ends of the connection, on connection, any changes in the states of the inputs at one end are transmitted to the peer, which then reflects those states on the appropriate I/O pins if they have been configured as outputs.

It is recommended that the value of S Reg 610 at one end be the complement of the other end. That way, inputs at one end are mirrored at the other end and vice versa.

In addition S Reg 506 *must* be set to zero, which disables echoes.

Note: Due to inherent latency of Bluetooth transmission, expect the change of state to be delayed. This value is typically 100 ms and can be much more if the quality of the link is bad which results in many retries.

It is assumed that an audio channel is not active at any time.

1.10.6 Append Bluetooth Address to Friendly name

If S Reg 593 is set to one, then the last six hex digits of the Bluetooth address are automatically appended to the friendly name. This allows multiple devices with the same name in a neighbourhood to be differentiated.

1.11 Disclaimers

LAIRD WIRELESS PRODUCTS ARE NOT AUTHORISED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE MANAGING DIRECTOR OF LAIRD TECHNOLOGIES.

The definitions used herein are:

a) Life support devices or systems are devices which (1) are intended for surgical implant into the body, or (2) support or sustain life and whose failure to perform when properly used in accordance with the instructions for use provided in the labelling can reasonably be expected to result in a significant injury to the user.

b) A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Laird Technologies does not assume responsibility for use of any of the circuitry described, no circuit patent licenses are implied and Laird Technologies reserves the right at any time to change without notice said circuitry and specifications.

1.12 Data Sheet Status

Laird reserves the right to change the specification without prior notice in order to improve the design and supply the best possible product.

1.13 Changes Between Release

Although every effort is made to ensure compatibility, the functionality of some features has changed due to the evolution of the Bluetooth chips and stack implementations. Users migrating between firmware variants should check the following differences:

- ATZ
- AT+BTIN

Please check with Laird Technologies for the most recent data before initiating or completing a design.

2. Multipoint Protocol

2.1 Introduction

This document describes a packet based messaging interface which is used by a host to send commands, receive responses and exchange multiplexed data with the **blu2i Multipoint Module**, henceforth described as the Module.

The Module consists of the same hardware as that used for the single point 'AT' blu2i module described elsewhere. Whereas the latter only allows one-to-one connection, the module described here allows simultaneous connections to a minimum of 3 slaves, and depending on hardware build, up to 7 slaves. It also allows connections to multiple profiles to one or more slaves. Hence this document adopts a concept of channels instead of slave connections.

The term 'host' in this document is taken to mean any entity which is a source of command messages, sink for response/event messages and both source and sink for multiplexed data packets.

To further eliminate any confusion, when the term 'command message' is used, it implies a message from the host to the module and likewise 'response message' is used to imply a message from the module to the host.

This document does NOT describe how the packets are physically exchanged between the host and the module. The transport medium will be either UART or USB. It also does NOT describe the format of any envelope that may be required to reliably and quickly transfer the message packet between the host and module.

This implies that when the packets proposed in this document are processed, they are assumed not to contain any errors.

2.1.1 Flow control & Data Integrity

It must be recognised that the transport mechanism will be streaming in nature. If the transport medium is USB then flow control and data integrity is inherently provided by the USB protocol.

If UART is the medium, then it shall be assumed that there will be a 5 wire interface, RX,TX,CTS,RTS and GND. Any host attached to the UART of the module shall **strictly** observe CTS/RTS hardware handshaking. Packet data integrity may or may not be provided depending on the build. It is expected that for a UART transport media, guaranteeing data integrity will be at the **severe** expense of data throughput.

2.1.2 Maximum Packet Size Consideration

Laird acknowledges that hosts attaching to the module may have limited RAM (random access memory) resource and so the module makes provision for this by guaranteeing not to send packets larger than a certain negotiated size (See S Reg 2 in the S Registers section). However, it is assumed that there will be a certain minimum packet size (16) that the host SHALL be capable of coping with. The host must guarantee that there is space for at least 2 packet in the receive buffer.

2.2 Packet General Format

This section describes the general format of incoming and outgoing packets.

The term 'incoming' will henceforth imply packets sent by the host to the module and 'outgoing' in the reverse direction. That is, the direction terminology is module centric.

All packets have octet granularity. When an octet is described as containing bit fields, it shall be taken that bit 0 is the least significant bit and bit 7 is the most significant bit.

Subfields in the packet which require multiple octets, shall be ordered so that the lowest significant octet is transmitted **LAST** over the transport media, **unless specifically described otherwise**. For example, a 16 bit word value will require 2 octets within the packet and the first octet will correspond to the upper byte. Similarly, a 6 byte Bluetooth address shall be transported most significant byte first. If the order is reversed then it will be specifically highlighted in the description of appropriate packets.

Subfields which are data arrays shall be described with the '[ ]' operator in descriptions which come in subsequent chapters.

Apart from data packets, all command, respond and event packets are of fixed size. If there is enough data to fill a packet, then the packet is filled with zeroes.

The protocol and packet format is optimised to ensure maximum data throughput via the virtual machine which processes these packets, and as such shall allow a maximum of 7 channels. It is possible to have all 7 channels in a single slave connection (to multiple profiles).

In recognition of the fact that a host may have limited RAM, the command/response will be designed to minimise the size of response packets. This means the number of commands may seem unduly large where multiple command packets will be required to get a set of data, where it would logically have been prudent to send all the data in one large response packet.

The design philosophy is to optimise data throughput over the air.

Subsequent sections describe the packets in detail.

2.2.1 Host to Module Packets

These are packets used to convey commands to the module or raw data to be sent over an open Bluetooth RFCOMM connection.

Command Packets

The format for command packets is as per the table below.

Octet	Field	Description
0	LENGTH	Total length of this packet, including this octet
1	CHANNEL	Always 0
2	COMMAND	Described in the next chapter
3	FLOW_IN	Bit 0 to 6 specify a mask. A clear bit means the module should NOT send any more packets to that corresponding data channel. Bit 7 is always 0 and will be used as an extension bit in the future. It is assumed that the host will always be able to receive a response or status packet.
4..N	DATA[]	Data as required and has meaning specific to COMMAND. For example, if the command is to make a connection to a peer device, then it will be at least a 6 octet array specifying the Bluetooth address of the peer.

The value of COMMAND shall be in the range 0 to 127.

Unknown command values result in an EVT_UNKNOWN_COMMAND event, with the command value reflected in the data field. If the octet value is specified in the range 128 to 255 (0x80 to 0xFF), then reflecting that value in the data field of an EVT_UNKNOWN_COMMAND instead of the COMMAND field of a response packet guarantees that the packet will NOT be mistakenly processed as an event.

Data Packets

The format for data packets is as per the table below and can arrive at any time. The only method by which the host can be stopped from sending this message is by sending a 0 value in the FLOW_OUT field of a response or status message. The module will be prepared to receive at least one data packet after deasserting the appropriate flow control bit.

If CHANNEL is 0xFF, then the data packet is broadcast to all open channels. (Planned for future release).

Octet	Field	Description
0	LENGTH	Total length of this packet, including this octet
1	CHANNEL	Bits 0 to 4 contain the value as per the channel id value returned in a successful connection message. It will be a value in the range 1 to 31. Bits 5,6,7 are reserved for future use and should always be set to 0.
2..N	DATA[]	This data array is unconditionally sent over the air

2.2.2 Module to Host Packets

These are packets used to convey responses or events from the module and raw data received over an open Bluetooth RFCOMM connection. Response packets **shall** always be as a result of a command packet and event packets are **asynchronously** sent to the host as and when required. *The host shall ensure that it is always ready to accept response and event packets, especially event packets as they can be sent at any time.*

Response Packets

The format for response packets is as per the table below.

Octet	Field	Description
0	LENGTH	Total length of this packet, including this octet
1	CHANNEL	Always 0
2	COMMAND	Echoed from the command packet (Shall be > 0)
3	FLOW_OUT	Bit 0 to 6 specify a mask. A clear bit means the host should NOT send any more packets to that corresponding data channel. Bit 7 is always 0 and will be used as an extension bit in the future.
4	STATUS	0 means success
N..M	DATA[]	Data as required and has meaning specific to the response for COMMAND.

Event Packets

The format for status packets is as per the table below.

Octet	Field	Description
0	LENGTH	Total length of this packet, including this octet
1	CHANNEL	Always 0
2	EVENT	Described later
3	FLOW_OUT	Bit 0 to 6 specify a mask. A clear bit means the host should NOT send any more packets to that corresponding data channel. Bit 7 is always 0 and will be used as an extension bit in the future.
N..M	DATA[]	Data as required and has meaning specific to the response for COMMAND

The only difference between a response and an event packet is that in the latter, octet 2 is defined as COMMAND in the former and EVENT in the latter. Also in the latter, the STATUS field is missing.

The value of COMMAND shall be in the range 0 to 0x7F and EVENT shall take values in the range 0x80 to 0xFF. This allows bit 7 of that octet to be used to decode whether the packet is a response packet or an event packet.

The value of STATUS is in the range 0 to 255. A value of 0 means SUCCESS and any other value is a failure, where the value gives more details of the failure type. The values of STATUS are defined in a 'C' header file which can be obtained on request from Laird.

Data Packets

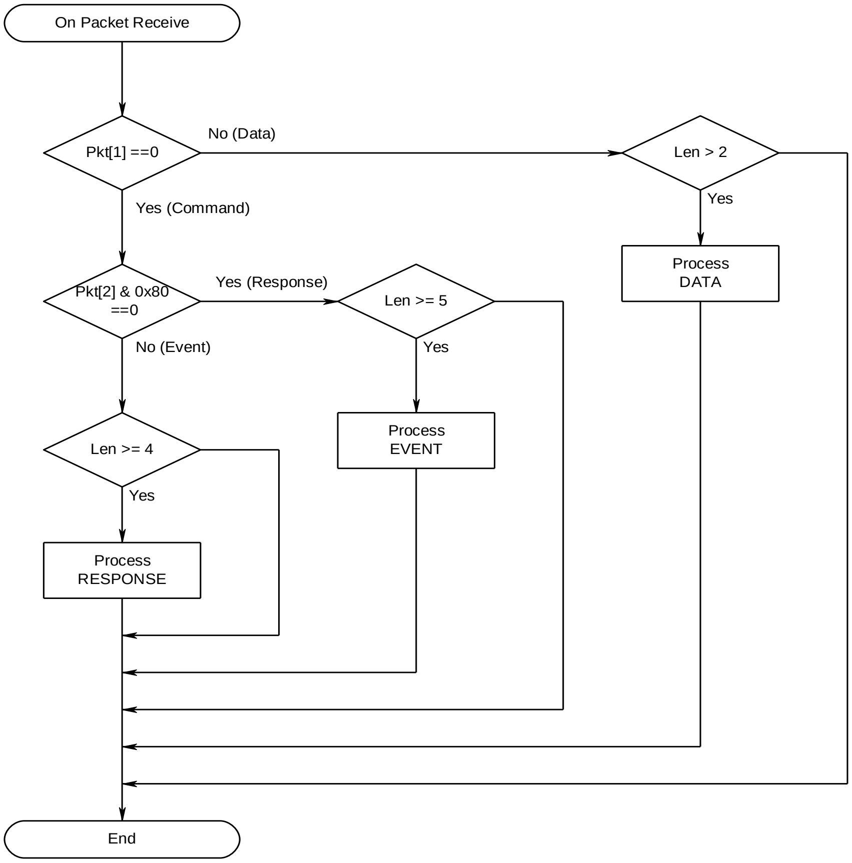
The format for data packets is as per the table below. The only method by which the host can stop the module from sending this message is by sending a 0 value in the FLOW_IN field of command message.

Octet	Field	Description
0	LENGTH	Total length of this packet, including this octet
1	CHANNEL	Bits 0 to 4 contain the value as per the channel id value returned in a successful connection message. It will be a value in the range 1 to 31. Bits 5,6,7 are reserved for future use and should always be set to 0.
2..N	DATA[]	Data received over the air for the channel

Data packets are symmetrical in both directions.

2.3.1 Host Packet Receive Flowchart

As maximum data throughput is the design goal, the format and detail of packets have been optimised. Implement the following flowchart, in the host, for rapid servicing and flow control of the packets.



2.3.2 Host Packet Receive Flowchart

As maximum data throughput is the design goal, the format and detail of packets have been optimised. Implement the following flowchart, in the host, for rapid servicing and flow control of the packets.

2.4 Host Command/Responses

This section describes all host commands in detail and is specified via the COMMAND field of all command packets.

The description for each command below is in the form of a command packet table and a corresponding response packet table.

Each command has a unique COMMAND value in the range 1 to 127 (0x01 to 0x7F). 0 is reserved.

The actual value of COMMAND in the Value column is described as [Descriptive_Name] where "Descriptive_Name" can be found in a 'C' header file which can be obtained on request from Laird.

The value of STATUS is similarly defined in a header file which can be obtained from Laird by request.

2.4.1 Information Commands

This group of commands are used to obtain information about the module.

No Operation

This command results in no action other than to convey new FLOW_IN status to the module and get a response packet with the latest status for the FLOW_OUT bits.

It is expected that a host will use this packet to poll for a change in the flow bits.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	4	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_NO_OPERATION]	
3	FLOW_IN	??	Runtime value

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	
1	CHANNEL	0	
2	COMMAND	[CMD_NO_OPERATION]	
3	FLOW_OUT	??	Runtime value
4	STATUS	[OK]	Or [INVALID_LICENSE]

Get Connectable, Discoverable, Security Modes

This command is used to get the current mo connectable, discoverable and security modes.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	4	Fixed

1	CHANNEL	0	Fixed
2	COMMAND	[CMD_GET_MODES]	
3	FLOW_IN	??	Runtime value

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	8	
1	CHANNEL	0	
2	COMMAND	[CMD_GET_MODES]	
3	FLOW_OUT	??	Runtime value
4	STATUS	OK or INVALID_LICENSE	
5	DISCMODE	0..1	1 for discoverable mode
6	CONNMODE	0..7	Bit 0: 1 for connectable mode Bit 1: 1 for Auto Accept Channe1 Bit 2: 1 for Auto Accept Mux
7	SECMODE	0..2	0 = No Auth, No Encryption 1 = Auth, No Encryption 2 = Auth + Encryption

Read Local Bluetooth Address

This command is used to read the Bluetooth address of the module.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	4	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_READ_BLUETOOTH_ADDRESS]	
3	FLOW_IN	??	Runtime value

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_READ_BLUETOOTH_ADDRESS]	
3	FLOW_OUT	??	Runtime value
4	STATUS	[OK]	
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address

Information

This command is used to extract information from the module, for example version number.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed

2	COMMAND	[CMD_INFORMATION]	
3	FLOW_IN	??	Runtime value
4	INFOTYPE	0..255	

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	14	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_INFORMATION]	
3	FLOW_OUT	??	Runtime value
4	STATUS	[OK]	
5	INFOTYPE	0..255	Echoed from command
6.13	DATA[8]	As per the table below	

The type of information requested is specified by the INFOTYPE parameter, as per the table below.

INFOTYPE = GET_VERSION (0)			
Offset	Field Name	Range	Comments
0	VERSION_SKEW	0..255	Firmware skew. Custom firmware is differentiated with this field
1	VERSION_MAJOR	0..255	Major version number
2	VERSION_MINOR	0..255	Minor version number
3	VERSION_ENG	0..255	Will be 0 for production releases otherwise Engineering release.
4	MAX_INFO_TYPE	N	Max value of INFOTYPE field
5..6	CSR STACK BUILD No	0..65535	Offset 5 = MSB
7	Reserved	Any value	Laird private Use

INFOTYPE = GET_MANUFACTURER (1)			
Offset	Field Name	Range	Comments
0..7	Manufacturer Name	E.g. "CSR"	Chip manufacturer, null terminated string

INFOTYPE = GET_CHIP_INFO (2)			
Offset	Field Name	Range	Comments
0..7	Chip Designation	E.g. "BC2-EXT"	Chip designation, null terminated string

INFOTYPE = GET_PHYSICAL_MEDIUM (3)			
Offset	Field Name	Range	Comments
0	Physical Medium	0	0=Bluetooth
1..7	Reserved	0	

INFOTYPE = GET_ RSSI_RANGE_LIMITS (4)			
Offset	Field Name	Range	Comments
0..1	GOLDEN_RECEIVE_RANGE	0..65535	Golden Receive Range
2..3	RECEIVE_LOWER_LIMIT	-32768..32767	Golden Receive Lower limit
4..5	RECEIVE_UPPER_LIMIT	-32768..32767	Golden Receive Upper limit
6..7	Reserved	Any value	Laird private Use

INFOTYPE = GET_ ACTUAL_TX_POWER (5)			
Offset	Field Name	Range	Comments
0..1	TX_POWER_IN_DBM	-32768..32767	In dBm. This is a signed 16 bit value. Offset 0 is msb and offset 1 is lsb

INFOTYPE = GET_ MAX_CODEC_OUTPUT_GAIN (6)			
Offset	Field Name	Range	Comments
0..1	Max Output Gain	0..65535	

2.4.2 Configuration Commands

This group of commands are used to configure the module.

Read 'S' Register

The module is configured using integer values which can be stored in non-volatile memory, similar to the S registers provided in the single point 'AT' module.

Valid register numbers are in the range 0 to 255. Registers 0 to 127 generally specify parameters which can be stored in a 16 bit storage location and registers 128 to 255 can store 32 bit values.

See the S Registers section for a full list of all registers.

The following command is used to read the current value of the S register REGNO.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_READ_SREG]	
3	FLOW_IN	??	Runtime value
4	REGNO	0 to 255	

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_READ_SREG]	
3	FLOW_OUT	??	Runtime value
4	STATUS	[OK]	
5	REGNO	0 to 255	Echoed from Command
6..9	REGVAL[]	Register Value	REGVAL[0] is the most significant octet.

Write 'S' Register

This command is used to write a new value to the S register REGNO.

See section 'S Registers' in chapter "Miscellaneous" for a full list of all registers.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	9	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_WRITE_SREG]	
3	FLOW_IN	??	Runtime value
4	REGNO	0 to 255	Registers 250 to 255 inclusive are used for manufacturing/firmware upgrade purpose only. DO NOT WRITE TO THESE REGISTERS
5..8	REGVAL[]	New Register Value	REGVAL[0] is the most significant octet.

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	
1	CHANNEL	0	
2	COMMAND	[CMD_WRITE_SREG]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	REGNO	0 to 255	Echoed from Command
6..9	REGVAL[]	Register Value	REGVAL[0] is the most significant octet.

Store 'S' Registers

This command is used to save the current 'S' register value in non-volatile memory so that they survive a power cycle.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	4	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_STORE_SREG]	
3	FLOW_IN	??	Runtime value

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_STORE_SREG_]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

Default 'S' Registers

This command is used to force all S register values to factory default.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	4	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ DEFAULT_SREG]	
3	FLOW_IN	??	Runtime value

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ DEFAULT_SREG_]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

Read Communications Parameters

This command is used to read the communications settings as stored in non-volatile storage.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	4	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ READ_COMMSPARM]	
3	FLOW_IN	??	Runtime value

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH		Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ READ_COMMSPARM]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5..8	BAUD[]	1200..921600	Offset 5 is the MSB and offset 9 the LSB
9	STOPBITS	1..2	Stop bits
10	PARITY	0..2	0=None, 1=Odd, 2=Even
11	FLOW	1	1= CTS/RTS handshaking

At the time of writing, only CTS/RTS handshaking is available. It cannot be disabled.

Write Communications Parameters

This command is used to write new communications settings to non-volatile storage.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_WRITE_COMMSPARM]	
3	FLOW_IN	??	Runtime value
4..7	BAUD[]	1200..921600	Offset 5 is the MSB and offset 9 the LSB
8	STOPBITS	1..2	Stop bits
9	PARITY	0..2	0=None, 1=Odd, 2=Even
10	FLOW	1	1= CTS/RTS handshaking

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_WRITE_COMMSPARM]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

Note: The communication parameters come into effect after a reset/power cycle.

2.4.3 Connection Commands

This group of commands are used to establish and break connections.

Set Connectable Mode

This command is used to enable/disable connectable mode and to specify auto accept parameters for channels and muxs.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONNECTABLE_MODE]	
3	FLOW_IN	??	Runtime value
4	ENABLE	0..1, 0xFF	0 = Disable, 1=Enable 0xFF = Read current mode
5	ACCEPT	Bit mask	Bit 0: Set to auto accept channel setup Bit 1: Set to auto accept mux setup (CURRENTLY IGNORE – FORCED TO 1 INTERNALLY)

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONNECTABLE_MODE]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	CURMODE	0..1	0 = Not connectable 1 = Connectable

In future releases of the firmware, the ability to accept/reject incoming RFCOMM mux setup will be provided. Until then, mux setups will be automatically accepted.

The module will use the parameters stored in 'S' Registers 9 and 10 to set the inquiry scan interval and window

Service Incoming Connection

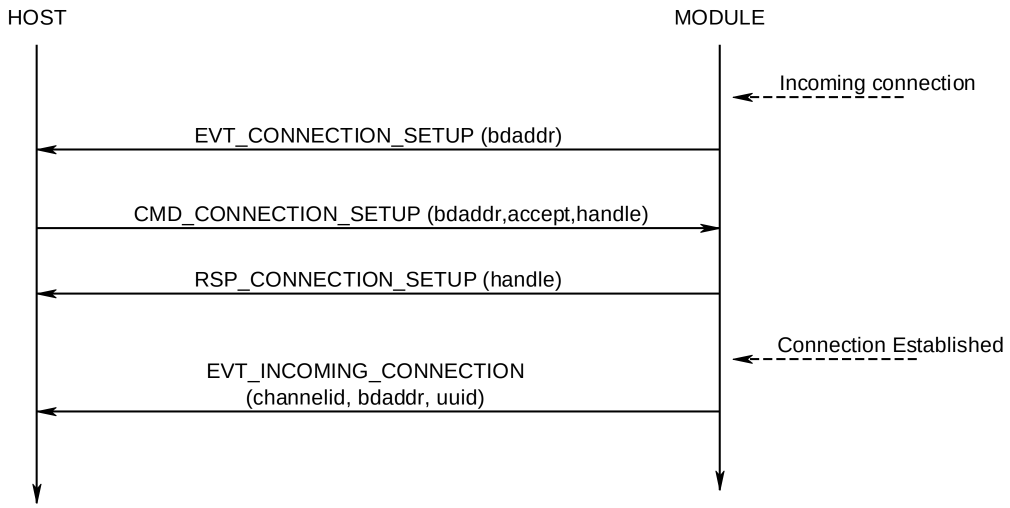
When the module is in connectable mode, incoming connection requests are passed up to the host via an EVT_CONNECTION_SETUP message. The host accepts or rejects the remote connection request using this message.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	12	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONNECTION_SETUP]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr

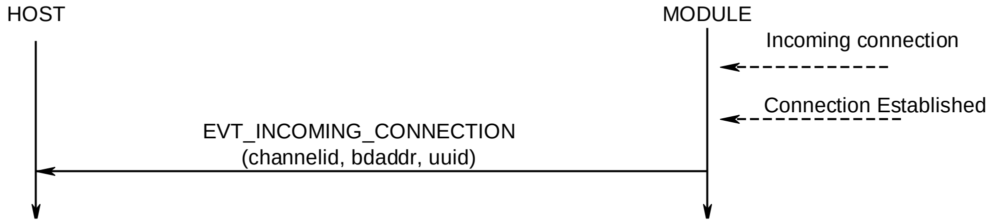
10	HANDLE	0..255 Can be any value the host wants to set.	This value is echoed by the module in the response.
11	ACCEPT	0..1	0 = reject, 1 = accept

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONNECTION_SETUP]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	HANDLE	0..255	Echoed from the command

Receipt of the response does **not** indicate the connection is established. If the connection will be accepted, the module sends EVT_INCOMING_CONNECTION when fully established, as in the message sequence chart.


HOST
MODULE
Incoming connection
EVT_CONNECTION_SETUP (bdaddr)
CMD_CONNECTION_SETUP (bdaddr,accept,handle)
RSP_CONNECTION_SETUP (handle)
Connection Established
EVT_INCOMING_CONNECTION
(channelid, bdaddr, uuid)

Note: If auto accept was specified when the module was put into connectable mode, then for incoming connections there will only be an EVT_INCOMING_CONNECTION message.



Make Outgoing Connection

This command is used to make an outgoing connection to a RFCOMM based profile in the remote peer.

Some peers have more than one instance of a given profile. For example, some mobile phones advertise two serial port profiles. The 'make connection' message has an instance field which is used to specify which instance of the remote profile the connection should be made to.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	13	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ MAKE_CONNECTION]	
3	FLOW_IN	??	Runtime value
4	HANDLE	0..255. Can be any value the host wants to set.	Value is echoed by the module in the response.
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address
11..12	UUID[]	0x1101 .. 0x11xx	UUID of the desired profile. Offset 11 is the MSB.
13	INSTANCE	0..255	0 is the first instance of the profile, 1 is the second etc.

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	7	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ MAKE_CONNECTION]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	HANDLE	0..255	Echoed from the command
6	CHANNEL	1..7	Channel Id to be used for subsequent data packets, and also when dropping the connection

If the STATUS field in the response is MPSTATUS_OK, then a connection was successfully established. Any other value is a failure.

The timeout specified in 'S' register 16 is used to abort the connection attempt if it takes too long. Likewise, once the connection is established, the value of 'S' register 12 is used to specify the link supervision timeout.

Finally the content of 'S' register 11 is used to specify the max frame size to be used by the lower layers – 0 means use default value.

Drop Connection

This command is used to destroy an existing channel.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ DROP_CONNECTION]	
3	FLOW_IN	??	Runtime value
4	HANDLE	0..255. Can be any value the host wants to set.	This value is echoed by the module in the response.
5	CHANNEL	1..7	As was specified in either RSP_MAKE_CONNECTION or EVT_INCOMMING_CONNECTION

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH		Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ DROP_CONNECTION]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	HANDLE	0..255	Echoed from the command

If the STATUS field in the response is MPSTATUS_OK, then the request to drop the channel was successfully submitted to the lower layers of the stack. When the channel is dropped, an EVT_DISCONNECT event will be sent to the host.

Note: Prior to v0.1.14, the meaning of this message was:
"If the STATUS field in the response is MPSTATUS_OK, then the channel was successfully destroyed. Any other value is a failure."

The meaning of this message was changed post 0.1.14 to make it easier for the host as it now needs to cater for only EVT_DISCONNECT to mean that a connection has been dropped.

Set Modem Lines

Bluetooth Serial Port Profile is capable of exchanging modem signals DTR, DSR, RTS, CTS, DCD and RI over air.

From a host's perspective, it can have DTR, RTS, DCD and RI as output lines. (Note DCD and RI are outputs for modems and 'host' in this context can mean either a PC or a peripheral like a modem).

Additionally UARTs are capable of sending BREAK signals. BREAK output signals are defined as a non-idle state TX pin for a period much greater than the character width at the current baud rate setting.

This command is used to send DTR, RTS, DCD and RI states to the peer device and also to specify a BREAK.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	7	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONTROLMODEMLINES]	
3	FLOW_IN	??	Runtime value
4	CHANNEL	1..7	Channel ID of an open channel
5	MODEM	Bit Mask	Bit 0: DTR state Bit 1: RTS state Bit 2: DCD state Bit 3: RI state
6	BREAK	0	For future implementation

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONTROLMODEMLINES]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	CHANNEL	1..7	Echoed from command

The STATUS value will be MPSTATUS_OK if the message was successful.

Modem signals sent by the peer device are presented to the host in the message EVT_MODEM_STATUS defined in subsequent chapters.

Note: BREAK signal capability is currently not provided by the lower stack, and so it is mentioned in the context of this command message for future implementation.

SCO Connect

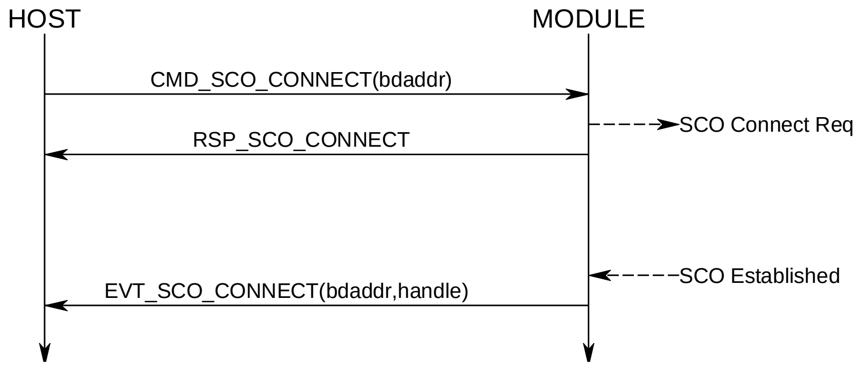
This command establishes a SCO channel alongside a pre-existing connection. SCO channels transfer audio between peers and are bi-directional.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SCO_CONNECT]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address
10	PKTTYPE	Bit Mask	Bit 0: HV1/EV3 Bit 1: HV2/EV4 Bit 2: HV3/EV5 Bit 3: Enhanced Sco

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SCO_CONNECT]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

The STATUS value will be MPSTATUS_OK if the request was successfully submitted to the stack.

When the SCO channel is established an EVENT called 'SCO Connect' will be sent to the host which will contain a handle to be used to disconnection requests.



SCO Disconnect

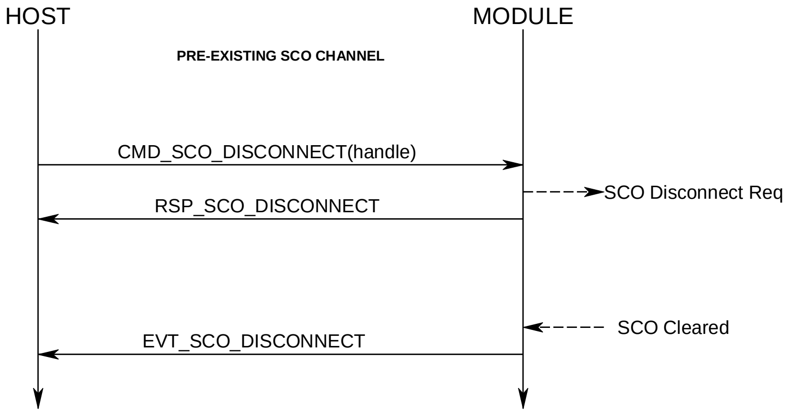
This command is used to disconnect an existing SCO channel identified by a handle which is provided in the SCO establishment event – see Event SCO Connect.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SCO_DISCONNECT]	
3	FLOW_IN	??	Runtime value
4..5	HANDLE[]	2 byte SCO handle	Handle

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SCO_DISCONNECT]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

The STATUS value will be MPSTATUS_OK if the message was submitted to the lower stack.

When the SCO channel is cleared an EVENT called 'SCO Disconnect' will be sent to the host.



SCO Incoming Setup

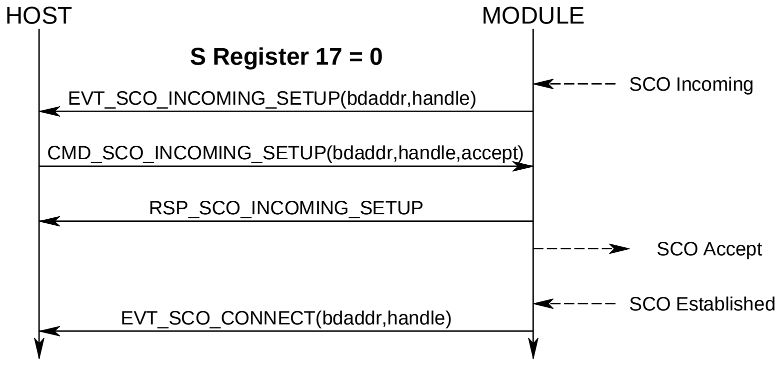
This command is used to accept or reject an incoming SCO channel and is used in response to a 'SCO Incoming Setup' event.

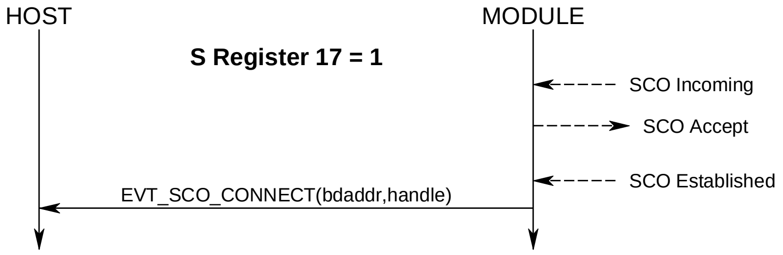
The 'SCO Incoming Setup' event in turn will only be sent to the host if S Reg 17 is 0. By default S Reg 17 is 1 and so incoming SCO channel setup will be automatically accepted.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	13	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONTROLMODEMLINES]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address, echoed from EVT_SCO_INCOMING_SETUP
10..11	HANDLE[]	SCO handle	2 byte handle, echoed from EVT_SCO_INCOMING_SETUP
12	ACCEPT	0, 1 or 2	0 to reject, 1 to accept SCO and 2 to accept eSCO

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ CONTROLMODEMLINES]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

The BDADDR and HANDLE fields **MUST** be copied from the EVT_SCO_INCOMING_SETUP packet.


HOST
MODULE
S Register 17 = 0
SCO Incoming
EVT_SCO_INCOMING_SETUP(bdaddr,handle)
CMD_SCO_INCOMING_SETUP(bdaddr,handle,accept)
RSP_SCO_INCOMING_SETUP
SCO Accept
SCO Established
EVT_SCO_CONNECT(bdaddr,handle)



Sniff Request

This command is used to enter or exit Sniff mode for a given open connection. If the ATTEMPT value is set to 0, then the device will take that to mean exit sniff mode.

The response packet will be received immediately. The host shall wait for an EVT_LOWPOWER_MODE event for actual notification of success or failure.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	18	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SNIFF_REQUEST]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address
10..11	ATTEMPT[]	Sniff Attempt	Millisecs. Offset 10 is MSB
12..13	TIMEOUT[]	Sniff Timeout	Millisecs. Offset 12 is MSB
14..15	MIN_INT[]	Sniff Minimum Interval	Millisecs. Offset 14 is MSB
16..17	MAX_INT[]	Sniff Maximum Interval	Millisecs. Offset 16 is MSB

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SNIFF_REQUEST]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

The STATUS value will be MPSTATUS_OK if the request was successfully submitted to the stack.

Minimum value of ATTEMPT is MIN_SNIFF_ATTEMPT_MSEC as defined in the header file BmHostProtocol.H. (12ms at the time of writing this manual).

Minimum value of TIMEOUT is MIN_SNIFF_TIMEOUT_MSEC as defined in the header file BmHostProtocol.H. (12ms at the time of writing this manual).

Minimum value of MIN_INT is MIN_SNIFF_MININTERVAL_MSEC as defined in the header file BmHostProtocol.H. (100ms at the time of writing this manual).

Minimum value of MAX_INT is MIN_SNIFF_MAXINTERVAL_MSEC as defined in the header file BmHostProtocol.H. (100ms at the time of writing this manual).

Maximum value of MAX_INT is MAX_SNIFF_MAXINTERVAL_MSEC as defined in the header file BmHostProtocol.H. (2000ms at the time of writing this manual).

Maximum value of MIN_INT cannot exceed the value specified in MAX_INT.

Maximum values of ATTEMPT and TIMEOUT cannot exceed value of MIN_INT.

Park Request

This command is used to enter or exit PARK mode for a given open connection. If either MIN_INT or MAX_INT value is set to 0, then the device will take that to mean exit park mode.

The response packet will be received immediately. The host shall wait for an EVT_LOWPOWER_MODE event for actual notification of success or failure.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	14	Fixed

1	CHANNEL	0	Fixed
2	COMMAND	[CMD_PARK_REQUEST]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address
10..11	MIN_INT[]	Park Minimum Interval	Millisecs. Offset 10 is MSB
12..13	MAX_INT[]	Park Maximum Interval	Millisecs. Offset 12 is MSB

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_PARK_REQUEST]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

The STATUS value will be MPSTATUS_OK if the request was successfully submitted to the stack.
Minimum value of MIN_INT is MIN_PARK_MININTERVAL_MSEC as defined in the header file BmHostProtocol.H. (10ms at the time of writing this manual).
Minimum value of MAX_INT is MIN_PARK_MAXINTERVAL_MSEC as defined in the header file BmHostProtocol.H. (10ms at the time of writing this manual).
Maximum value of MAX_INT is MAX_PARK_MAXINTERVAL_MSEC as defined in the header file BmHostProtocol.H. (40000ms at the time of writing this manual).
Maximum value of MIN_INT cannot exceed the value specified in MAX_INT.

RSSI and Link Quality

This command is used to obtain the RSSI and Link Quality values for a given open connection. This is a parameter associated with the ACL connection to a peer device and has no meaning with channel IDs.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_RSSI_LINKQUAL]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH		Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_RSSI_LINKQUAL]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr
11	RSSI	-128 to 127	RSSI value. Zero if the signal is in "golden range."
12	LINKQUAL	0 – 255	

The definitions of RSSI and LINKQUAL are paraphrased from the Bluetooth specification as follows:-

- RSSI - This value is the difference between the measured Received Signal Strength Indication (RSSI) and the limits of the Golden Receive Power Range (see below for definition). Any positive RSSI value returned by the Host Controller indicates how many dB the RSSI is above the upper limit, any negative value indicates how many dB the RSSI is below the lower limit. A value of zero indicates that the RSSI is inside the Golden Receive Power Range. Note: how accurate the dB values will be depends on the Bluetooth hardware. The only requirements for the hardware are that the Bluetooth device is able to tell whether the RSSI is inside, above or below the Golden Device Power Range.
- GOLDEN RECEIVE POWER RANGE - The lower threshold level of the golden receive power range corresponds to a received power between -56 dBm and 6 dB above the actual sensitivity of the receiver. The upper threshold level is 20 dB above the lower threshold level to an accuracy of +/- 6 dB
- LINK QUAL - Link_Quality is a value from 0-255, which represents the quality of the link between two Bluetooth devices. The higher the value, the better the link quality is. Each Bluetooth module vendor determines how to measure link quality. In the case of CSR, it is a measure of BER (Bit Error Rate).

Stream Connect

Conceptually, connections can be seen as pipes of data, where each pipe is a sink and a source. For example, when an SCO channel is established, it can be a source or sink of data. Likewise, the Bluetooth module has a PCM interface which has 3 slots and each slot can be viewed as a pipe too.

This command is used to connect a 'source' of data to a 'sink' which consumes that data.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_STREAM_CONNECT]	
3	FLOW_IN	??	Runtime value
4	TYPE	See note 1 below	Source Stream Type
5..6	HANDLE[]	2 byte handle id	Source handle
7	TYPE	See note 1 below	Sink Stream Type
8..9	HANDLE[]	2 byte handle id	Sink handle

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_STREAM_CONNECT]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	TYPE	See note 1 below	Source Stream Type
6..7	HANDLE[]	2 byte handle id	Source handle
8	TYPE	See note 1 below	Sink Stream Type
9..10	HANDLE[]	2 byte handle id	Sink handle

The STATUS value will be MPSTATUS_OK if the Source was successfully connected to the Sink.

Note: Source or Sink Stream Type values are:
01 - SCO channel and the handle value is as returned in the EVT_SCO_CONNECT message.
02 - PCM channel / valid handle values are 0 to 2 inclusive and map to slots on the PCM highway.
03 - Reserved for future use.

Stream Disconnect

There is a concept of pipes of data where each pipe is a sink and a source.

This command is used to disconnect a 'sink' which is a consumer of data.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_STREAM_DISCONNECT]	
3	FLOW_IN	??	Runtime value
4	TYPE	See note 1 below	Source Stream Type
5..6	HANDLE[]	2 byte handle id	Source handle
7	TYPE	See note 1 below	Sink Stream Type
8..9	HANDLE[]	2 byte handle id	Sink handle

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	8	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_STREAM_DISCONNECT]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	TYPE	See note 1 below	Sink Stream Type
6..7	HANDLE[]	2 byte handle id	Sink handle

The STATUS value will be MPSTATUS_OK if the Source was successfully disconnected from the Sink.

2.4.4 Inquiry Commands

This group of commands are used to performing inquiries and putting the module into discoverable mode.

Inquiry Request

This command is used to perform a Bluetooth inquiry.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_INQUIRY_REQ]	
3	FLOW_IN	??	Runtime value
4	MAXRESP	1..255	Maximum number of responses before aborting the inquiry procedure
5	TIMEOUT	1..120	Time in seconds, before aborting the inquiry procedure.
6	FLAGS	0	Future use

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	7	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_INQUIRY_REQ]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	TOTAL	??	The total number of inquiry responses that were received from peers.
6	DUMP	??	The total number of inquiry result events that were NOT sent because the transmit buffer of the module was full. This will be as a result of the host deasserting its RTS line.

As a result of this command, as and when peer devices respond with inquiry responses, for each inquiry response, an event EVT_INQUIRY_RESULT is sent to the host.

When the number of inquiry responses specified in the command are received OR the specified time has elapsed, the response will be sent to indicate to the host that the inquiry procedure is complete.

If the DUMP field in the response is non-zero it is indicating that the host is not reading it's receive buffer fast enough and is resulting in RTS being deasserted towards the module.

Set Discoverable Mode

This command is used to enable/disable discoverable mode.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ DISCOVERABLE_MODE]	
3	FLOW_IN	??	Runtime value
4	ENABLE	0..1, 0xFF	0 = Disable, 1=Enable 0xFF = Read current mode

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ DISCOVERABLE_MODE]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	CURMODE	0..1	0 = Not discoverable, 1 = discoverable

The module will use the parameters stored in 'S' Registers 7 and 8 to set the inquiry scan interval and window

2.4.5 Pairing Commands

This group of commands are used to perform either incoming or outgoing pairings and to manage the trusted device database which resides in the module.

The trusted device database is database with a two tables, each with records of two fields. One field is the Bluetooth address of a paired device and the other is used to store the link key. For security reasons, it is not possible to put the link key on the UART, hence treat it as an invisible field.

One database is classed a ROLLING database and is used to store new pairing information as they happen. If the database is full, then the oldest is discarded to make space for the latest one.

The other database is classed as a PERSISTANT database which stores pairing information which cannot ONLY be deleted when a new pairing is initiated to that device OR on request from the host.

The host protocol provides for a command to transfer a record between these two databases. In addition there is a command for the host to determine if a device is trusted.

Pairing initiation or accept procedures can only be entered if there are no existing connections.

Pair Initiate

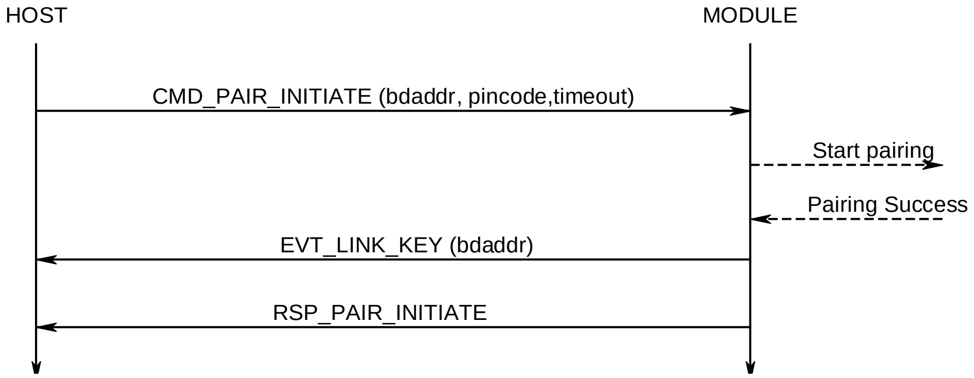
This command initiates a pairing with a peer device which is assumed to be ready and waiting for a pairing.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	28	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ PAIR_INITIATE]	
3	FLOW_IN	??	Runtime value
4	TIMEOUT	5..120	Pairing timeout in seconds
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr of device to be paired
11..27	PIN[]	17 byte string array	Null Terminated Pin code String. Max pin code length is 16.

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ PAIR_INITIATE]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

If pairing is successful the event message EVT_LINK_KEY will be sent to the host before the response, to close the procedure, as shown in the message sequence chart below.

Pair Initiate will fail if there are any existing open connections. The status byte in the response will have an appropriate value.



Pair Accept

This command is used to put the module into a mode such that it accepts a peer initiated pairing. While in this mode, it is not possible to initiate pairing, inquiry or make connections.

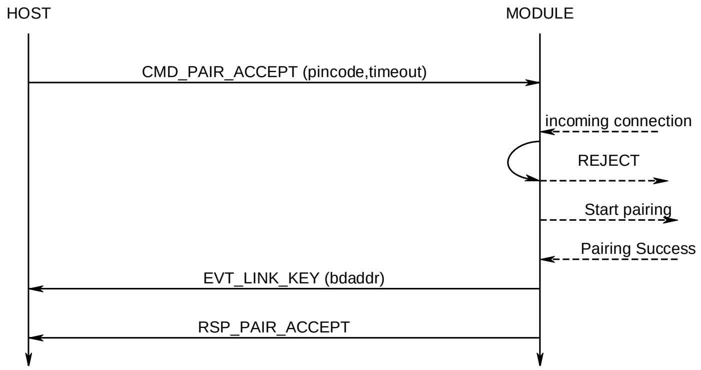
In addition, any incoming connections will be silently rejected.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	22	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_PAIR_ACCEPT]	
3	FLOW_IN	??	Runtime value
4	TIMEOUT	5..120	Pairing timeout in seconds
5..21	PIN[]	17 byte string array	Null Terminated Pin code String. Max pin code length is 16.

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_PAIR_ACCEPT]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

If pairing is successful the event message EVT_LINK_KEY will be sent to the host before the response, to close the procedure, as shown in the message sequence chart below.

Pair Accept will fail if there are any existing open connections. The status byte in the response will have an appropriate value.



PinCode

This command is used to send a pincode in response to an EVT_PINCODE_REQUEST message.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	28	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_PINCODE]	
3	FLOW_IN	??	Runtime value
4	TIMEOUT	5..120	Pairing timeout in seconds
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address of device requesting the pairing
11..27	PIN[]	17 byte string array	Null Terminated Pin code String. Max pin code length is 16.

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_PINCODE]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

If pairing is successful the event message EVT_LINK_KEY will be sent to the host after the response.

Trusted Database Record Count

This command is used to obtain the number of trusted devices in the database specified.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ TRUSTED_DB_COUNT]	
3	FLOW_IN	??	Runtime value
4	DBTYPE	0..1	0 = ROLLING DATABASE 1 = PERSISTANT DATABASE

RESPONSE PACKET Offset	 Field	 Value	 Comments
0	LENGTH	8	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ TRUSTED_DB_COUNT]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	DBTYPE	0..1	Echoed from command
6	COUNT	0..N	Number of trusted devices in this database
7	MAXCOUNT	0..N	Maximum number of devices that can be stored in this database

Note: ROLLING database is used to store all new pairings. It is up to the host to transfer a record from ROLLING to the PERSISTANT database.

Trusted Database Read Record

This command reads the Bluetooth address of the ITEMNO pairing in the database specified, counted from the top. ITEMNO is indexed from 1.

COMMAND PACKET Offset	 Field	 Value	 Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_TRUSTED_DB_READ]	
3	FLOW_IN	??	Runtime value
4	DBTYPE	0..1	0 = ROLLING DATABASE 1 = PERSISTANT DATABASE
5	ITEMNO	1..N	Item number to read

RESPONSE PACKET Offset	 Field	 Value	 Comments
0	LENGTH	13	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_TRUSTED_DB_READ]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	DBTYPE	0..1	Echoed from command
6	ITEMNO	1..N	Echoed from command
7..12	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr. All 0's if the item does not exist

Trusted Database Delete Record

This command is used to delete a pairing from the databases. It is not necessary to specify the database type, as any type of instance is deleted.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ TRUSTED_DB_DELETE]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr of device to be unpaired

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ TRUSTED_DB_DELETE]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr, echoed from command

Trusted Database Change Type

This command is used to transfer a record to the database specified.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_TRUSTED_DB_CHANGETYPE]	
3	FLOW_IN	??	Runtime value
4	DBTYPE	0..1	
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	12	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_TRUSTED_DB_CHANGETYPE]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	DBTYPE	0..1	Echoed from command
6..11	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr, echoed from command

Trusted Database Is Peer Trusted

This command is used to check if a device is trusted.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ TRUSTED_DB_ISTRUSTED]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_ TRUSTED_DB_ISTRUSTED]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr, echoed from the command

The STATUS value will be MPSTATUS_OK if the device is trusted, any other value means not trusted.

2.4.6 Miscellaneous Commands

Set/Get Security Mode

This command is used to set or get the current security mode of the module. The module can be in one of three modes. No authentication (0), Authentication Only (1), Authentication and Encryption (2). These modes are applied to all connection attempts, whether inbound or outbound.

Specifying a value of 0xFF means leave the mode as it is, but inform the host with regards to current mode.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	5	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SECURITY_MODE]	
3	FLOW_IN	??	Runtime value
4	SECMODE	0..2, 0xFF	0 = No Auth, No Encryption 1 = Auth, No Encryption 2 = Auth + Encryption 0xFF = Get current mode

RESPONSE PACKET Offset	 Field	 Value	 Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SECURITY_MODE]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	SECMODE	0..2	Current mode

Get Remote Friendly Name

This command is used to get the friendly name of the specified peer device.

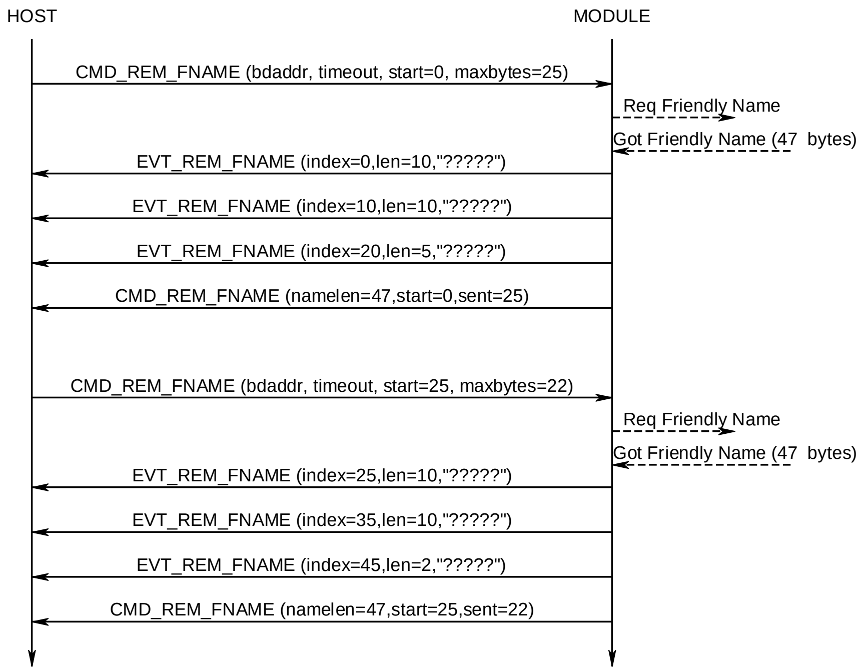
According to the Bluetooth specification a friendly name can be up to 248 bytes long. Sending this name in one go to the host could violate the max packet length capability of the host due to memory restrictions in the host OR transmit buffers in the module may not be able to cope. Therefore, the mechanism for getting the name to the host is via event packets EVT_REM_FNAME. The host decides how many bytes of the name is to be passed up to it via these events from the offset it also specifies. This implies that in a memory constraint environment, it will be possible to relay the name to the host using multiple commands.

For example, if the host has space for only 10 bytes and a peer happens to have a very long name, the host can ask for 10 byte fragments of the name over multiple get name requests.

COMMAND PACKET Offset	 Field	 Value	 Comments
0	LENGTH	13	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_GET_REM_FNAME]	
3	FLOW_IN	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth addr
10	TIMEOUT	1..120	Timeout in seconds
11	START	n	Offset into the friendly name string
12	MAXBYTES	m	Max characters to read

RESPONSE PACKET Offset	 Field	 Value	 Comments
0	LENGTH	8	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_GET_REM_FNAME]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	NAMELEN	0..248	Actual size of the friendly name
6	START	n	Echoed from the command
7	SENTLEN	s	Total number of bytes sent

Note: SENTLEN may be less than MAXBYTES if there is no space in the module TX buffer to send events.


HOST
MODULE
CMD_REM_FNAME (bdaddr, timeout, start=0, maxbytes=25)
Req Friendly Name
Got Friendly Name (47 bytes)
EVT_REM_FNAME (index=0,len=10,"?????")
EVT_REM_FNAME (index=10,len=10,"?????")
EVT_REM_FNAME (index=20,len=5,"?????")
CMD_REM_FNAME (namelen=47,start=0,sent=25)
CMD_REM_FNAME (bdaddr, timeout, start=25, maxbytes=22)
Req Friendly Name
Got Friendly Name (47 bytes)
EVT_REM_FNAME (index=25,len=10,"?????")
EVT_REM_FNAME (index=35,len=10,"?????")
EVT_REM_FNAME (index=45,len=2,"?????")
CMD_REM_FNAME (namelen=47,start=25,sent=22)

Get Local Friendly Name

This command is used read the local friendly name which is stored in non-volatile memory. Unlike the remote friendly name with no maximum length, the local friendly name is limited to 31 characters.

This length still may too big to send to the host in one packet. Therefore the name is sent in a similar fashion to 'get friendly name'. However in this case the event EVT_LCL_FNAME is used to get the name to the host.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_GET_LCL_FNAME]	
3	FLOW_IN	??	Runtime value
4	START	n	Offset into the friendly name string
5	MAXBYTES	m	Max characters to read

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH		Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_GET_LCL_FNAME]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	NAMELEN	0..31	Actual size of the friendly name
6	START	n	Echoed from the command
7	SENTLEN	s	Total number of bytes sent in preceding events.

The name string is sent to the host in EVT_LCL_FNAME packets. See description of Get Remote Friendly name

Note: SENTLEN could be less than MAXBYTES. It can happen if there is no space in the module's TX buffer to send events.

Set Local Friendly Name

This command sets the local friendly name in non-volatile memory so it is used after a power up/reset cycle.

Since the module can cope with large packets, the name sent to the module in a single command packet as a null terminated string.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	36	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SET_LCL_FNAME]	
3	FLOW_IN	??	Runtime value
4	FLAGS	1 MUST ALWAYS BE SET TO 1	Bit 0: Set to store in nonvol memory Bits 1..7: Future use
5	NAMELEN	1..30	
6..36	NAME[31]	Null terminated string	Not more than 30 characters

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH		Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SET_LCL_FNAME]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

Note: Future releases will allow setting the friendly name immediately in the baseband and/or storage in non-volatile memory. The mechanism for this will be via bit flags in field FLAGS.

Get Digital/Analog I/O

This command is used read the states of up to 16 digital input lines and optionally request an analogue input reading.

This response packet contains 2 octets containing the digital input states. If an analogue input reading is requested then the ADC reading in the range 0..1800 will be supplied in an EVENT_ADC event.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_GET_IO]	
3	FLOW_IN	??	Runtime value
4	digId	0	0 = Onboard Digital I/o 1 = I/O Mapped as GPIO pins
5	analogId	0..2	0 = No ADC access 1 = ANAL_1 (MAIO_0) 2 = ANAL_2 (MAIO_1)

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH		Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_GET_IO]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5	digId	0	Echoed from command packet
6..7	digIn[2]	??	Digital inputs 0 to 15

Note: If analogId field in the command is 1 or 2 and EVENT_ADC will be generated when the ADC is read and available.

Set Digital I/O

This command is used to control the states of up to 16 digital output lines. The appropriate GPIO pin will have been set to an output pin via S Register 23.

COMMAND PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SET_IO]	
3	FLOW_IN	??	Runtime value
4	ioId	1	1 = I/O Mapped as GPIO pins
5..6	ioVal[2]	0000..FFFF	

RESPONSE PACKET			
Offset	Field	Value	Comments
0	LENGTH		Fixed
1	CHANNEL	0	Fixed
2	COMMAND	[CMD_SET_IO]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	

2.5 Module Events

This section describes all module originated asynchronous events in detail and is specified via the EVENT field of all event packets.

The description for each event below is in the form of an event packet tables.

Each event has a unique EVENT value in the range 129 to 255 (0x81 to 0xFF), 0x80 is reserved.

The actual value of EVENT in the Value column is described as [Descriptive_Name] where "Descriptive_Name" can be found in a 'C' header file which can be obtained on request from Laird.

2.5.1 Inquiry Events

This group of events are inquiry related.

Inquiry Result

This event is used to send the inquiry response from a peer as a result of an inquiry request.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	13	
1	CHANNEL	0	
2	EVENT	[EVT_ INQUIRY_RESULT]	
3	FLOW_OUT	??	Runtime value
4..9	BDADDR[6]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address of responding device
10..12	DEVCLASS[3]	0x000000 .. 0xFFFFFF	Device class of responding device

2.5.2 Information Events

This group of events are used to convey information about the module, for example to status.

Unknown Command

This event is used to inform the host that a command was received with an unknown COMMAND value. The COMMAND value is echoed in offset 4.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH		
1	CHANNEL		
2	EVENT	[EVT_UNKNOWN_COMMAND]	
3	FLOW_OUT	??	Runtime value
4	command	xx	COMMAND value echoed from command

Status

This event is used to asynchronously send current status to the host. This event is sent to the host after power up to inform the host that the module is ready and operational. The information contained in this message can also be obtained by sending the CMD_GET_MODES command.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	8	
1	CHANNEL	0	
2	EVENT	[EVT_ STATUS]	
3	FLOW_OUT	??	Runtime value
4	STATUS	OK or INVALID_LICENSE	
5	DISCMODE	0..1	1 for discoverable mode
6	CONNMODE	0..7	Bit 0: 1 for connectable mode Bit 1: 1 for Auto Accept Channe1 Bit 2: 1 for Auto Accept Mux
7	SECMODE	0..2	0 = No Auth, No Encryption 1 = Auth, No Encryption 2 = Auth + Encryption

Invalid Packet Size

This event is used to inform the host that a command packet has been received whose length does not .

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	7	
1	CHANNEL	0	
2	EVENT	[EVT_INVALID_PKTSIZE]	
3	FLOW_OUT	??	Runtime value
4	COMMAND	1..127	Echoed from the command
5	ACTSIZE	A	Actual size of the packet
6	DESSIZE	D	Desired size of the packet

2.5.3 Connection Events

This group of events are connection related.

Connection Setup

This event is used to inform the host that a remote device is requesting a connection.

The host shall respond with a CMD_CONNECTION_SETUP with an accept or reject flag.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	12	
1	CHANNEL	0	
2	EVENT	[EVT_ CONNECTION_SETUP]	

3	FLOW_OUT	??	Runtime value
4..9	BDADDR[6]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address of device requesting connection
10..11	UUID[]	Server profile uuid the peer wishes to connect to	0x1101 = SPP 0x1108 = HEADSET 0x1112 = AUDIO GATEWAY 0x1103 = DUN

The UUID field tells the host to which server profile the peer wishes to connect to.

Incoming Connection

This event is used to inform the host that an incoming connection has been established.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	13	
1	CHANNEL	0	
2	EVENT	[EVT_ CONNECTION_SETUP]	
3	FLOW_OUT	??	Runtime value
4	CHANNEL	1..7	Channel ID to be used to send/receive data
5..10	BDADDR[6]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address of device requesting connection
11..12	UUID[]	Server profile UUID the peer wishes to connect to	0x1101 = SPP 0x1108 = HEADSET 0x1112 = AUDIO GATEWAY 0x1103 = DUN

The UUID field tells the host to which server profile the peer has connected to.

Disconnect

This event is used to inform the host that a connection has been dropped by the remote device.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	
1	CHANNEL	0	
2	EVENT	[EVT_ DISCONNECT]	
3	FLOW_OUT	??	Runtime value
4	CHANNEL	1..7	Channel number
5	REASON	0..255	As per the table below

The reason value is specified in the Bluetooth specification and are reproduced here for your convenience as follows, please note that values in the range 0xF0 to 0xFF are custom values defined for this implementation and do not appear in the Bluetooth specification.

0x01	Unknown HCI Command.
0x02	No Connection.
0x03	Hardware Failure.
0x04	Page Timeout.

0x05	Authentication Failure.
0x06	Key Missing.
0x07	Memory Full.
0x08	Connection Timeout.
0x09	Max Number Of Connections.
0x0A	Max Number Of SCO Connections To A Device.
0x0B	ACL connection already exists.
0x0C	Command Disallowed.
0x0D	Host Rejected due to limited resources.
0x0E	Host Rejected due to security reasons.
0x0F	Host Rejected due to remote device is only a personal device.
0x10	Host Timeout.
0x11	Unsupported Feature or Parameter Value.
0x12	Invalid HCI Command Parameters.
0x13	Other End Terminated Connection: User Ended Connection.
0x14	Other End Terminated Connection: Low Resources.
0x15	Other End Terminated Connection: About to Power Off.
0x16	Connection Terminated by Local Host.
0x17	Repeated Attempts.
0xFF	High probability that the remote device went out range for longer than the link supervision timeout or was powered down.

Modem Status

This event is used to convey modem status signals originating from the peer device.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	6	
1	CHANNEL	0	
2	EVENT	[EVT_ MODEM_STATUS]	
3	FLOW_OUT	??	Runtime value
4	CHANNEL	1..7	Channel ID of an open channel
5	MODEMSIG	Bit Mask	Bit 0: DSR state Bit 1: CTS state Bit 2: DCD state Bit 3: RI state
6	BREAKSIG	0	For future implementation

Low Power Mode

This event is used to inform the host that the connection related to the Bluetooth address specified has changed its low power mode to one of ACTIVE, HOLD, SNIFF or PARK. The new mode shall assumed to be the case ONLY if the 'status' field has the value MPSTATUS_OK (== 0).

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	12	
1	CHANNEL	0	
2	EVENT	[EVT_LOW_POWER_MODE]	
3	FLOW_OUT	??	Runtime value

4	STATUS	0 = ok	
5..10	BDADDR[6]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address
11	MODE	New Mode	0 = ACTIVE 1 = HOLD 2 = SNIFF 3 = PARK

SCO Connect

This event is used to inform the host that a SCO connection has been established.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	15	
1	CHANNEL	0	
2	EVENT	[EVT_ SCO_CONNECT]	
3	FLOW_OUT	??	Runtime value
4	STATUS	As appropriate	
5..10	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	
11	COUNT	1..3	Total SCO channels open
12..13	HANDLE[]	2 byte SCO Handle	
14	INCOMING	0..1	1 if SCO was established by peer

SCO Disconnect

This event is used to inform the host that a SCO connection has been cleared.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	15	
1	CHANNEL	0	
2	EVENT	[EVT_SCO_DISCONNECT]	
3	FLOW_OUT	??	Runtime value
12..13	HANDLE[]	2 byte SCO Handle	
14	REASON	XX	

SCO Incoming Setup

This event is used to inform the host that a peer is attempting to establish a SCO connection, and the host needs to accept or reject using the CMD_SCO_INCOMING_SETUP command packet.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	13	
1	CHANNEL	0	
2	EVENT	[EVT_ SCO_CONNECT]	
3	FLOW_OUT	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	
10	COUNT	1..3	Total SCO channels open
11..12	HANDLE[]	2 byte SCO Handle	

When the host sends the CM_SCO_INCOMING_SETUP command packet in response to this event is MUST echo the BDADDR[] and HANDLE[] fields from this event packet.

Remote Features

This event is used to inform the host that the peer just about to connect has the feature list as provided in the message.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	18	
1	CHANNEL	0	
2	EVENT	[EVT_REMOTE_FEATURES]	
3	FLOW_OUT	??	Runtime value
4..9	BDADDR[]	Nap[0,1]:Uap[2]:Lap[3,4,5]	
10..17	FEATURE[]	8 bytes	The feature list as described belo

The feature list is as described below:-

0.0 : 3 Slot packets
0.1 : 5 Slot packets
0.2 : Encryption
0.3 : Slot Offset
0.4 : Timing accuracy
0.5 : Role switch
0.6 : Hold mode
0.7 : Sniff Mode
1.0 : Park State
1.1 : Power control requests
1.2 : Channel quality driven data rate (CQDDR)
1.3 : SCO link
1.4 : HV2 packet
1.5 : HV3 packet
1.6 : u-law log synchronous data
1.7 : A-law log synchronous data
2.0 : CVSD synchronous data
2.1 : Paging parameter negotiation
2.2 : Power control
2.3 : Transparent synchronous data
2.4 : Flow control lag (least significant bit)
2.5 : Flow control lag (middle bit)
2.6 : Flow control lag (most significant bit)
2.7 : Broadcast encryption
3.0 : Reserved
3.1 : Enhanced Data Rate ACL 2 Mbps mode
3.2 : Enhanced Data Rate ACL 3 Mbps mode
3.3 : Enhanced Inquiry Scan
3.4 : Interlaced Inquiry Scan
3.5 : Interlaced Page Scan
3.6 : RSSI with inquiry results
3.7 : Extended SCO lin (EV3 packets)
4.0 : EV4 packets
4.1 : EV5 packets
4.2 : Reserved
4.3 : AFH capable slave
4.4 : AFH classification slave
4.5 : Reserved
4.6 : Reserved
4.7 : 3-slot Enhanced Data Rate ACL packets

5.0 : 5-slot Enhanced Data Rate ACL packets
5.1 : Reserved
5.2 : Reserved
5.3 : AFH capable master
5.4 : AFH classification master
5.5 : Enhanced Data Rate eSCO 2 Mbps mode
5.6 : Enhanced Data Rate eSCO 3 Mbps mode
5.7 : Reserved
6.0 : Reserved
6.1 : Reserved
6.2 : Reserved
6.3 : Reserved
6.4 : Reserved
6.5 : Reserved
6.6 : Reserved
6.7 : Reserved
7.0 : Reserved
7.1 : Reserved
7.2 : Reserved
7.3 : Reserved
7.4 : Reserved
7.5 : Reserved
7.6 : Reserved
7.7 : Extended feature

2.5.4 Miscellaneous Events

Link Key

This event is used to inform the host that a new link key has been created for the device indicated and the result of writing to the ROLLING database.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	11	
1	CHANNEL	0	
2	EVENT	[EVT_ LINK_KEY]	
3	FLOW_OUT	??	Runtime value
4..9	BDADDR[6]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address of paired device
10	DBRESULT	0: Success	Any other value is a failure and the reason is a STATUS value as per MPSTATUS.H

Pin Code Request

This event is used to inform the host that a remote device has requested a pairing and a pin code is required to complete the procedure.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	10	
1	CHANNEL	0	
2	EVENT	[EVT_ PINCODE_REQUEST]	
3	FLOW_OUT	??	Runtime value
4..9	BDADDR[6]	Nap[0,1]:Uap[2]:Lap[3,4,5]	Bluetooth address of pairing device

The host shall send a CMD_PINCODE in response to this event.

This event is only received if 'accept pairing while in connectable mode' is enabled via S Register 15.

Local Friendly Name

This event is used to send a fragment of the local friendly name to the host. The maximum length of the fragment is 10, so at least 3 of these events are required to convey a local friendly name, if it has the maximum length of 30.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	16	
1	CHANNEL	0	
2	EVENT	[EVT_LCL_FNAME]	
3	FLOW_OUT	??	Runtime value
4	INDEX	0..29	Start index into the string
5	LEN	1..10	Number of valid characters in the NAME[] field that follows
6..15	NAME[10]	Xx xx xx xx	The name fragment

Remote Friendly Name

This event is used to send a fragment of the remote friendly name to the host. The maximum length of the fragment is 10, so at least 25 of these events are required to convey a remote friendly name, if it has the maximum length of 248.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	16	
1	CHANNEL	0	
2	EVENT	[EVT_REM_FNAME]	
3	FLOW_OUT	??	Runtime value
4	INDEX	0..247	Start index into the string
5	LEN	1..10	Number of valid characters in the NAME[] field that follows
6..15	NAME[10]	Xx xx xx xx	The name fragment

ADC value

This event is used to send the ADC input reading of the ADC channel requested in the most recent CMD_GET_IO command.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	7	
1	CHANNEL	0	
2	EVENT	[EVT_ADC]	
3	FLOW_OUT	??	Runtime value
4	adcId	1..2	ADC channel Id
5	adcValMsb	??	Runtime value
6	adcValMsb	??	Runtime value

2.5.5 Debug Events

Debug Packet

This event is used to convey debugging information to the host, and will be available in engineering/beta builds only.

EVENT PACKET			
Offset	Field	Value	Comments
0	LENGTH	16	
1	CHANNEL	0	
2	EVENT	[EVT_DEBUG_PACKET]	
3	FLOW_OUT	??	Runtime value
4	TYPE_FLAG	XX	Bit 0: First Packet Bit 1: Last Packet Bit 2..5: Reserved Bit 6..7: Message Type
5..15	DATA[]	Contains Ascii data	String conveying message

2.6 S Registers

This section details all the configuration 'S' registers. Min and Max values are given in decimal, unless the value is prefixed by 0x, in that case the value is in hexadecimal.

RegNo Dec (Hex)	Min	Max	Description
1 (01)	0	0	Host Interface
2 (02)	16	255	Maximum data packet size from module to host
3 (03)	1	0x1F	Server Profile record Mask Bit 0 = SPP Bit 1 = HEADSET Bit 2 = DUN Bit 3 = AUDIO GATEWAY Bit 4 = HANDSFREE
4 (04)	0	1	Default Connectable Mode on power up/reset
5 (05)	0	1	Default Discoverable Mode on power up/reset
6 (06)	0	2	Default Security Mode on power up/reset 0 = No Authentication/No Encryption 1 = Authentication/No Encryption 2 = Authentication/Encryption
7 (07)	10	2550	Inquiry Scan Interval in units of msec, rounded to the nearest 10ms. (10ms will give 11.25)
8 (08)	10	2550	Inquiry Scan Window in units of msec, rounded to the nearest 10ms. (10ms will give 11.25)
9 (09)	10	2550	Page Scan Interval in units of msec, rounded to the nearest 10ms. (10ms will give 11.25)
10 (0A)	10	2550	Page Scan Window in units of msec, rounded to the nearest 10ms. (10ms will give 11.25)
11 (0B)	0	127	Max frame size for lower Bluetooth stack. Recommend this not be changed unless you are absolutely sure what you are doing.
12 (0C)	0	30	Link supervision timeout for connections
13 (0D)	0	1	Auto Accept Mux setup. If this is 1, then incoming RFCOMM channels are automatically accepted.
14 (0E)	0	1	Auto Accept Channel Setup. If this is 1, incoming connections will be automatically accepted. If this is 1, EVT_CONNECTION_SETUP events are not sent to the host when an incoming connection arrives.
15 (0F)	0	1	If this is 1, then pairing requests while in connectable mode are automatically rejected. To accept pairing, the host will need to be in accept pairing mode, which is initiated by the CMD_PAIR_ACCEPT host command
16 (10)	2	120	For outgoing connection attempts, this is the timeout before the attempt is abandoned.
17 (11)	0	2	Auto Accept incoming SCO channel. If this is 1, then incoming SCO channels are auto accepted, if set to 2, then eSCO is auto accepted. Set to 0 to get the host to accept/reject the connection.
18 (12)	0	1	Auto Route 1st SCO channel to PCM Slot 0 **See Note 1 below**

RegNo Dec (Hex)	Min	Max	Description
19 (13)	1	4	BootMode. Default is 1 which sets the unit up for a Motorola codec in master mode. A value of 3 sets it up for the OKI7702/5 in master mode
20 (14)	-27	6	Maximum RF transmit power. To read actual power at any moment, read it back via CMD_INFORMATION, subcommand 5
21 (15)	0	X (<=15)	Output gain of Codec A. X is dependent on csr stack build and use CMD_INFORMATION, sub command 06 to get actual maximum value
22 (16)	0	X (<=15)	Output gain of Codec B. X is dependent on csr stack build and use CMD_INFORMATION, sub command 06 to get actual maximum value
23 (17)	0	0x01FF	GPIO Direction register. Setting a bit to 1 will configure the GPIO pin as output. CMD_SET_IO command packet can be used to control the state.
24 (18)	0	1	Option field for Headset profile service record
25 (19)	0	63	Option field for Handsfree profile service record
26 (1A)	0	1	If set to 1, master/slave role switch requests from a peer will be denied.
27 (1B)	0	1	Some events sent to the host have to be enabled by setting appropriate bits in the register. Bit: 0 - Enable REMOTE_FEATURES event
126 (7E)	0	0xFFFF	Primer for changing to AT mode.
127 (7F)	0	0xFFFF	Change to AT mode using value 0x0100.
128 (80)	0	0xFFFFFF	Module's Class of Device

Note: SReg 18 is read by the firmware at reset. Hence after setting this register a reset is required for it to be effective. This means the S Register set MUST be committed to non-volatile memory before initiating a reset. The S Registers are stored to non-volatile memory using the command [CMD_STORE_SREG].

2.6.1 COMMAND & EVENT Values

The following is a listing of a snapshot of the file BMHOSTPROTOCOL.H at the time of writing this document. Laird does NOT guarantee that this listing will be kept up to date.

For development purposes, please send a request to wireless.support@lairdtech.com for the latest version of the 'C' header file.

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
//The following are COMMAND (octet 2) values in command/response packets
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#define CMD_NO_OPERATION                                  0x01
#define CMD_READ_BDADDR                                   0x02
#define CMD_READ_SREG                                     0x03
#define CMD_WRITE_SREG                                    0x04
. . . .
. . . .
. . . .
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
//The following are EVENT (octet 2) values in event packets
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#define EVT_STATUS                                        0x81
#define EVT_INVALID_PKTSIZE                               0x82
#define EVT_UNKNOWN_COMMAND                               0x83
```

```
#define EVT_INQUIRY_RESULT                              0x84
#define EVT_MODEM_STATUS                                0x85
. . . .
. . . .
. . . .
```

SEND REQUEST TO wireless.support@lairdtech.com FOR LATEST VERSION OF THIS HEADER FILE

2.6.2 STATUS Values

The following is a listing of a snapshot of the file MPSTATUS.H at the time of writing this document. Laird does NOT guarantee that this listing will be kept up to date.

For development purposes, please send a request to wireless.support@lairdtech.com for the latest version of the 'C' header file.

```
#define MPSTATUS_OK                                     0x00

#define MPSTATUS_ILLEGAL_COMMAND                        0x01
#define MPSTATUS_NO_CONNECTION                          0x02
#define MPSTATUS_HARDWARE_FAIL                          0x03
#define MPSTATUS_PAGE_TIMEOUT                           0x04
. . . .
. . . .
. . . .
```

SEND REQUEST TO wireless.support@lairdtech.com FOR LATEST VERSION OF THIS HEADER FILE

3 PCM CODEC INTERFACE

PCM_OUT, PCM_IN, PCM_CLK, and PCM_SYNC carry up to three bi-directional channels of voice data, each at 8ksamples/s. The format of the PCM samples can be 8-bit A-law, 8-bit µ-law, 13-bit linear, or 16-bit linear. The PCM_CLK and PCM_SYNC terminals can be configured as inputs or outputs, depending on whether the module is the master or slave of the PCM interface.

Contact a Laird FAE for further details.

The module is compatible with the Motorola SSI TM interface and interfaces directly to PCM audio devices including the following:

3.3 Compatible Codec Chips

- Winbond W61360 13-bit linear CODEC (Motorola MC145483 compatible)
- OKI MSM7702 single channel A-law and µ-law CODEC

The default codec support is for the Winbond W61360.

4 FTP Client Profile Commands

The FTP profile provides a capability allowing a host to act as an 'FTP client' to a peer device providing an 'FTP server' profile as shown in the diagram below.

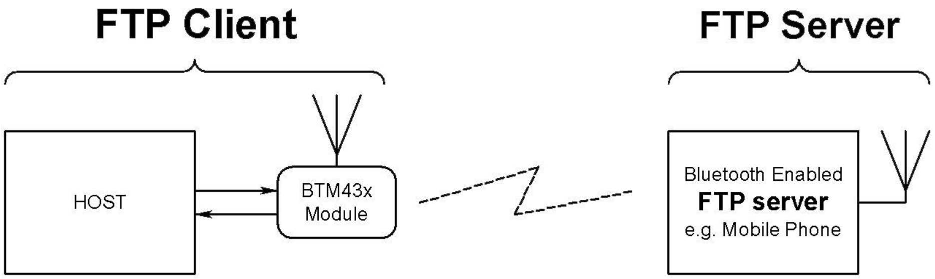


Figure 1: FTP client and server

An FTP client capability implies the ability to send and receive files and also to manipulate file objects in the remote device. The Bluetooth FTP server profile specification describes the profile as one built on Obex Exchange (OBEX) which is in turn built on SPP.

The FTP implementation allows a host attached to the module to send and receive files and in addition to manipulate files and folders.

The format used for describing this protocol is a series of message sequence charts with accompanying notes as appropriate, which unambiguously convey how a host and the module shall interact to perform the task.

4.3 Generic Notes and Guidance

In the message sequence charts the following abbreviations apply:

- <filename> shall mean a string delimited by the " character. For example, "hello.txt".
- <foldername> shall mean a string delimited by the " character.
- nnn shall be a decimal number with at least one digit.
- The backspace character is not supported.
- All FTP commands are case sensitive.
- FTP commands shall not exceed 32 characters in total.
- While an FTP session is open, the host shall not deassert the modules UART_CTS line and conversely the host MUST always be ready to accept data.
- When Unicode data is transmitted, it shall be assumed that the most significant byte is transmitted first.
- If an FTP command is expecting a Unicode string as a parameter and the host has an ASCII string, the string shall be expanded with a 0 byte in the most significant position.

4.4 FTP Related AT Commands

4.4.1 AT+FTP<bd_addr>

This command is used to establish a connection to an FTP server profile in a peer device with Bluetooth address <bd_addr>.

When a connection is successfully established, the host assumes that the current folder is the root folder. This root folder is always relative to the host. It is *not* necessarily the absolute root folder of the host machine.

4.4.2 AT+OFT<bd_addr>

This command is used to establish a connection to an FTP server profile in a peer device with Bluetooth address <bd_addr> and functions similarly to AT+FTP, but instead of the response being "\r\nNNN FTP\r\n" it is "\r\nNNN OBX\r\n", where NNN is a decimal number.

This capability of choosing the response type is to allow a host to cater generically a connection which is either FTP or ObexPush.

4.4.3 ATSn=m

The following values of n are relevant to FTP operation: 582. A description of these values is given in S582.

4.5 FTP Related Subcommands

This section describes FTP-related subcommands that the host can use to control the FTP session.

4.5.1 PUT <filename><length> (Send file)

This FTP subcommand is used to send a file to the FTP server.

The length of <filename> shall not exceed 24 characters.

The optional <length> value is inserted into the OBEX length header field. This is optional for FTP.

There is some ambiguity as to how the first OBEX PUT packet is formed with respect to the 'Body' header. The OBEX specification does not prohibit the first 'Body' header to be empty; neither does it say that it must *not* be empty. If the first body is not empty when sending a file to a Nokia 6820 phone then it seems to confuse it. Because of this, Laird sends out an empty 'Body' header by default which is also what the Windows PC based Widcomm Bluetooth stack sends.

To cater for future devices which require the first 'Body' header to be non-empty, a new S Register 582 has been added to allow a host to have control over how the first body header is constructed.

The new S register 582 takes values in the range 0 to 1. The default value is 0 which implies that the first 'Body' header in the PUT OBEX packet is empty. A value of 1 forces that 'Body' header to have one byte of data – and in this case, when the module prompts the host for a length value it shall respond accordingly.

Laird hopes that the default value of zero suffices for all occasions, but provides the control to modify the packet as required.

4.5.2 PUT –nnn<length>(Send file)

This FTP subcommand is used to send a file to the FTP server where the filename is in UNICODE text and the filename is –nnn bytes long.

The optional <length> value is inserted into the OBEX length header field. This is optional for FTP.

See comment above in PUT <filename><length> (Send file) with regards to S Reg 582.

4.5.3 GET <filename>(Retrieve a file)

This FTP subcommand is used to retrieve a file from the FTP server. The length of <filename> shall not exceed 24 characters.

4.5.4 GET -nnn (Retrieve a file)

This FTP subcommand is used to retrieve a file from the FTP server where the filename is in UNICODE and the filename is –nnn bytes long.

4.5.5 DIR (Get folder listing)

This FTP subcommand is used to retrieve the folder listing.

The Bluetooth FTP specification requires a server to transmit a folder listing as a properly formed XML document. In a properly formed document, the character '&' is supposed to be escaped into a five character string &amp;. Testing shows that the Widcomm Bluetooth stack and the Microsoft Windows CE Bluetooth stack do not comply with that requirement; they send the '&' unescaped.

This means that if a folder contains filenames with '&' characters, the XML document fails parsing and this command fails with the response "090 FTP".

In this circumstance, the only way for the host to extract the folder listing from the server is to request the folder listing in raw XML. This is expedited using the DIR -RAW (Get folder listing, XML document) command.

4.5.6 DIR -RAW (Get folder listing, XML document)

This FTP subcommand is used to retrieve the folder listing. In this variant, the OBEX response packet, which is in ASCII XML format, is sent to the host verbatim.

WARNING: The Widcomm stack seems to append two null characters at the end of the XML document. This means extra care if the host stores data as null terminated strings, because the final "200 FTP" prompt appears corrupted when it is not.

4.5.7 MD <foldername> (Create a folder)

This FTP subcommand is used to create a subfolder. The length of <foldername> shall not exceed 24 characters.

4.5.8 MD -nnn (Create a folder)

This FTP subcommand is used to create a subfolder which is specified in Unicode.

4.5.9 CD <foldername> (Change folder)

This FTP subcommand is used to navigate to the subfolder specified. The length of <foldername> shall not exceed 24 characters.

4.5.10 CD -nnn (Change folder)

This FTP subcommand is used to navigate to the subfolder specified in Unicode.

4.5.11 CD \ (Change folder to root)

This FTP subcommand is used to navigate to the root folder.

4.5.12 CD .. (Change folder to parent)

This FTP subcommand is used to navigate to the parent folder.

4.5.13 RD <foldername> (Delete a folder)

This FTP subcommand is used to delete the folder specified. Some FTP servers do not allow non-empty folders to be deleted. In that case, an appropriate error response shall be returned to the host. The length of <foldername> shall not exceed 24 characters.

4.5.14 RD -nnn (Delete a folder)

This FTP subcommand is used to delete the folder specified where the folder name is specified in Unicode.

4.5.15 DEL <filename> (Delete a file)

This FTP subcommand is used to delete the file specified.

4.5.16 DEL -nnn (Delete a file)

This FTP subcommand is used to delete the file specified where the filename is specified in Unicode.

4.5.17 ABORT (Abort current ftp command)

This FTP subcommand is used to abort a file transfer where appropriate. To abort a PUT session, send 0 when the module prompts for a length value.

4.5.18 QUIT (Terminate the FTP session)

This FTP subcommand is used to terminate the FTP session and also results in the Bluetooth connection being terminated.

4.5.19 MAX (Max outgoing obex packet size)

This FTP subcommand is used to get the maximum OBEX packet size tolerated by server and can be used by the host to optimize the data throughput.

4.6 FTP-Related Subresponses

All FTP-related sub responses sent from the module to the host SHALL be 11 characters long in the format:

<cr><lf>nnn AAA<cr><lf>

The characters nnn shall be decimal digits '0' to '9', then there is a space character and finally a three character word followed by <cr><lf>.

The <cr><lf> envelope plus the fixed length will hopefully make the parsing task in the host much simpler.

Values for 'nnn' are as specified for HTTP status codes, and reproduced from the OBEX specification as follows:

OBEX Response Code	HTTP Status Code	Definition
0x00to 0x0F	None	Reserved
0x10(0x90)	100	Continue
0x20(0xA0)	200	OK, Success
0x21(0xA1)	201	Created
0x22(0xA2)	202	Accepted
0x23(0xA3)	203	Non-Authoritative Information
0x24(0xA4)	204	No Content
0x25(0xA5)	205	Reset Content

OBEX Response Code	HTTP Status Code	Definition
0x26(0xA6)	206	Partial Content
0x30(0xB0)	300	Multiple Choices
0x31(0xB1)	301	Moved Permanently
0x32(0xB2)	302	Moved temporarily
0x33(0xB3)	303	See Other
0x34(0xB4)	304	Not modified
0x35(0xB5)	305	Use Proxy
0x40(0xC0)	400	Bad Request - server couldn't understand request
0x41(0xC1)	401	Unauthorized
0x42(0xC2)	402	Payment required
0x43(0xC3)	403	Forbidden - operation is understood but refused
0x44(0xC4)	404	Not Found
0x45(0xC5)	405	Method not allowed
0x46(0xC6)	406	Not Acceptable
0x47(0xC7)	407	Proxy Authentication required
0x48(0xC8)	408	Request Time Out
0x49(0xC9)	409	Conflict
0x4A(0xCA)	410	Gone
0x4B(0xCB)	411	Length Required
0x4C(0xCC)	412	Precondition failed
0x4D(0xCD)	413	Requested entity too large
0x4E(0xCE)	414	Request URL too large
0x4F(0xCF)	415	Unsupported media type
0x50(0xD0)	500	Internal Server Error
0x51(0xD1)	501	Not Implemented
0x52(0xD2)	502	Bad Gateway
0x53(0xD3)	503	Service Unavailable
0x54(0xD4)	504	Gateway Timeout
0x55(0xD5)	505	HTTP version not supported
0x60 (0xE0)	- - -	Database Full
0x61 (0xE1)	- - -	Database Locked

For more details of these values, refer to the irDA specification which can be freely downloaded from www.irda.org.

In addition, values in the range 050 to 099 and 250 to 299 inclusive are specific to this Laird application and are defined in Table 4-1.

Table 4-1: nnn Codes

Response Code 'nnn'	(Laird Technologies Specific) Definition
050	Syntax Error / Command Unrecognized
051	Server sent unexpected information in obex packet
052	Obex connection fail, because it is unauthorized
053	Memory allocation failure (Please contact Laird Technologies with details)
055	Unicode File/Folder name length cannot be an odd value
056	Command not recognized
090	An XML parsing error occurred (while processing response to DIR command)
099	The Bluetooth connection has unexpected been dropped. (I.e., remote out of range etc.)
250	GET procedure was aborted

4.6.1 nnn FTP

This FTP response is used, when the connection was opened using AT+FTP, to convey the outcome of a subcommand to the host where 'nnn' is a fixed three digit decimal number as defined in the irDAspecification and map to HTTP status codes.

For example, a value of n=200 implies successful operation, any other value usually conveys an error as described in the irDA specification (except the range 050 to 099 inclusive and 250 to299 inclusive).

> Note: When a Bluetooth OBEX session is established, we will be specifying v1.0 in the header packets by default as that is what the Widcomm stack seems to be using and the FTP specifies.

4.6.2 nnn OBX

This has the same meaning as "nnn FTP" and is used when AT+OFT command was used to open an FTP connection.

4.6.3 nnn GET

This FTP response is used during a GET file operation. See appropriate message sequence charts for more details. See Table 4-1 for 'nnn' values.

4.7 FTP Line Multiplexing Commands

When transferring a file, the single serial interface between the host and the module is used to send and receive data and commands. In other words, a scheme is required to unambiguously determine when a byte on the line corresponds to a command or data belonging to a file.

The module uses negotiated multiplexing to achieve this and commands are used to toggle the line between command and data mode.

This scheme is symmetrical and the commands for toggling the state of the line are relevant for both direction. The only difference being that the terminator is <cr> in the host to module direction and<crlf> in the reverse direction.

The commands are described in the following sub sections.

4.7.1

This command is used to ask the other end how many bytes of a filename or foldername it will send next.

4.7.2 >

This command is used to inform the other end that it is safe to send the number of bytes belonging to a filename or foldername as indicated in the most recent # command.

4.7.3 !

This command is used to ask the module/host how many bytes of data it will send next. If the module/host sends a length value that is too large to handle, it can be rejected by sending the ! command again. This is because accepting a value implies this end should send a ? prompt to trigger the data phase (See Section 4.7.4).

4.7.4 ?

This command is used to inform the other end that it is safe to send the number of bytes belonging to 'data' phase indicated in the most recent ! command.

4.8 Message Sequence Charts

In the following sections, the color scheme uses RED text as commands from a host to the module and BLUE text as responses and prompts from the module to the host. Command/responses shown in BLACK are associated with non-FTP related states.

Apart from FTP connection and disconnection scenarios, to avoid repetition, all message sequence charts assume that the module is in a FTP-connected state. It also assumes that where "nnn FTP" occurs, it can be read as "nnn OBX" if the command AT+OFT was used to initiate the command.

4.8.1 Usage: Make FTP Connection (No Authentication)

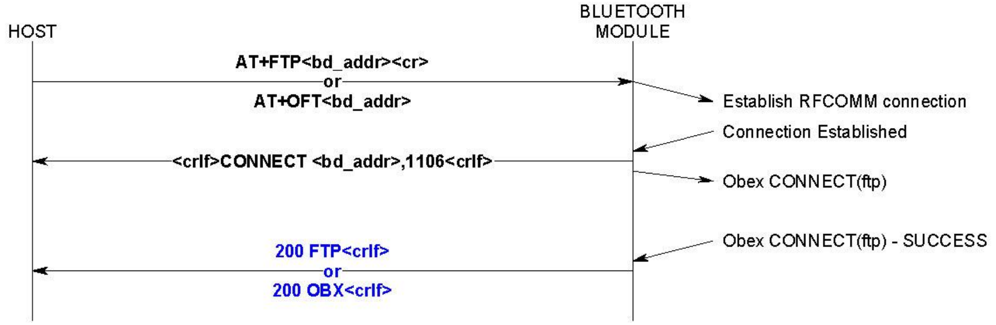


Figure 2: Make FTP connection (no authentication)

4.8.2 Usage: FTP Disconnection

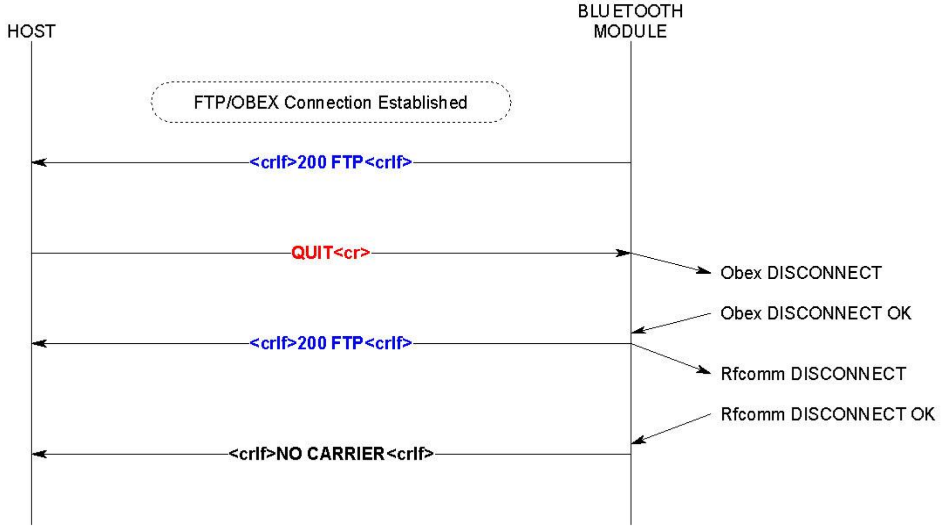
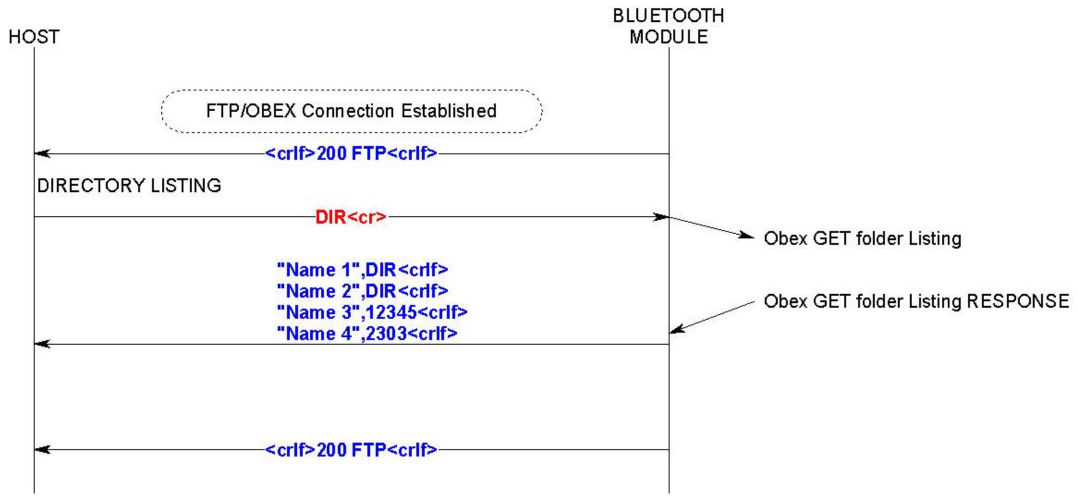


Figure 3: FTP disconnection

4.8.3 Usage: Folder Listing

Note: If a number follows a name then that implies a file.

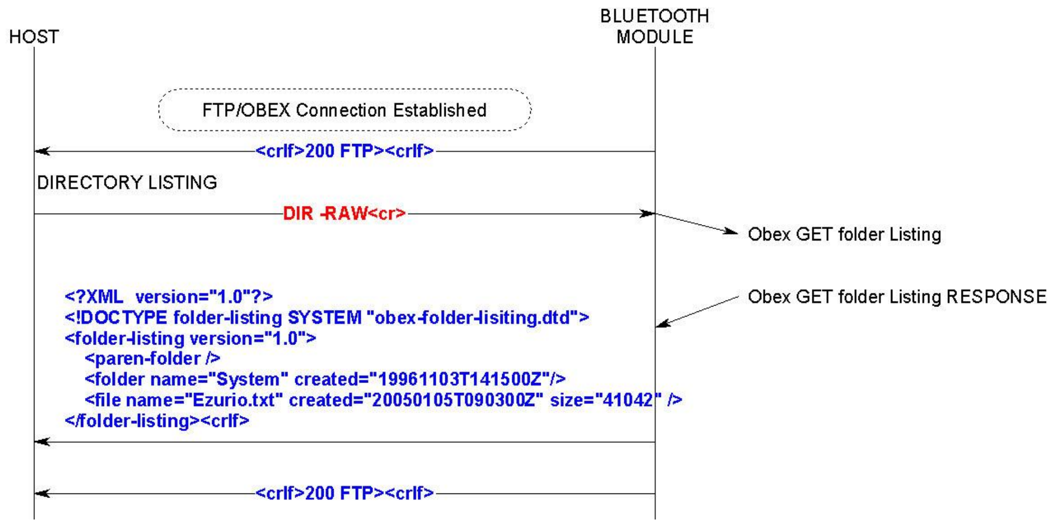


```
"Name 1",DIR<crlf>
"Name 2",DIR<crlf>
"Name 3",12345<crlf>
"Name 4",2303<crlf>
```

Figure 4: Folder listing

4.8.4 Usage: Folder Listing (Raw Output)

Note: The raw output is ASCII text and is in XML format.



```
<?XML  version="1.0"?>
<!DOCTYPE folder-listing SYSTEM "obex-folder-lisiting.dtd">
<folder-listing version="1.0">
   <paren-folder />
   <folder name="System" created="19961103T141500Z"/>
   <file name="Ezurio.txt" created="20050105T090300Z" size="41042" />
</folder-listing><crlf>
```

Figure 5: Folder listing (raw output)

4.8.5 Usage: Create Folder (Short Folder name)

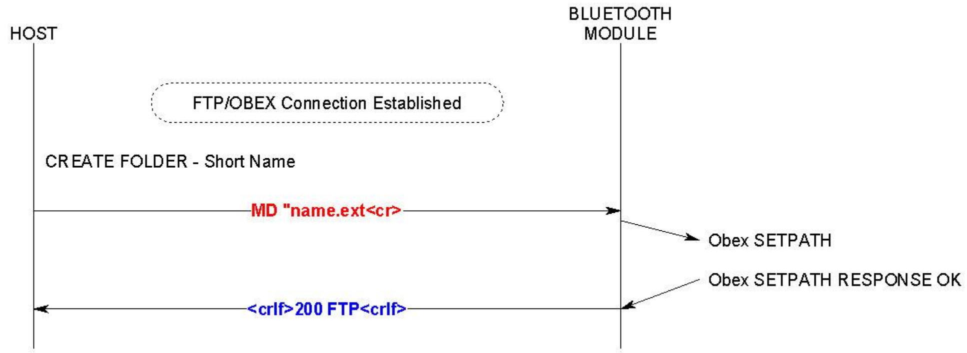


Figure 6: Create folder (short folder name)

4.8.6 Usage: Create Folder (Long Folder name - UNICODE)

Note: 'nn' is the size of folder name in bytes. The folder name is supplied in Unicode.

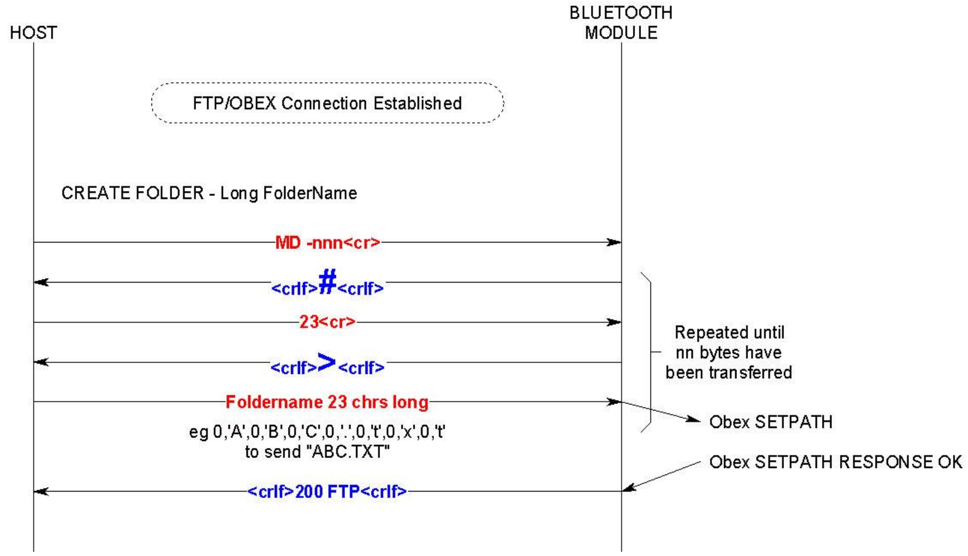


Figure 7: Create folder (long name - UNICODE)

4.8.7 Usage: Create Folder (Unsuccessful)

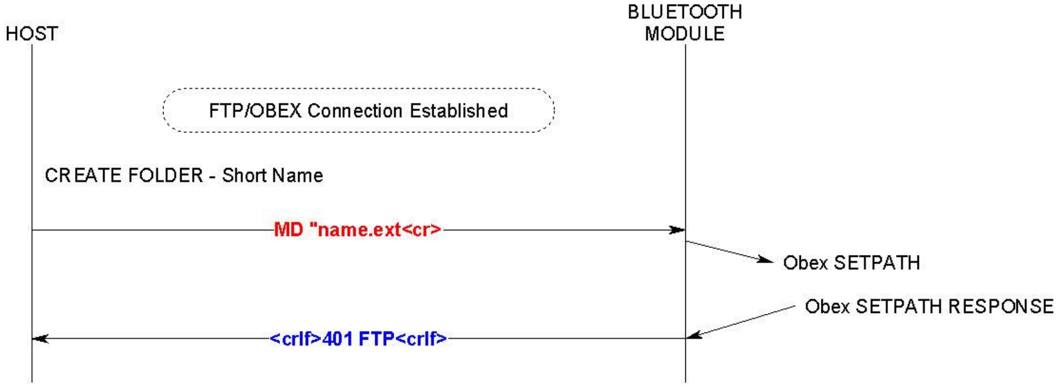


Figure 8: Create folder (unsuccessful)

4.8.8 Usage: Change Folder (Short Folder name)

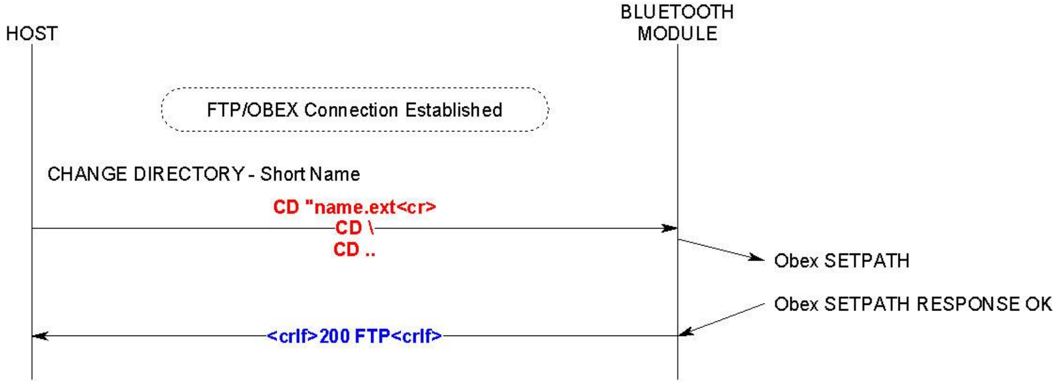


Figure 9: Change folder (short folder name)

4.8.9 Usage: Change Folder (Long Folder name - UNICODE)

Note: 'nn' is the size of folder name in bytes. The folder name is supplied in Unicode.

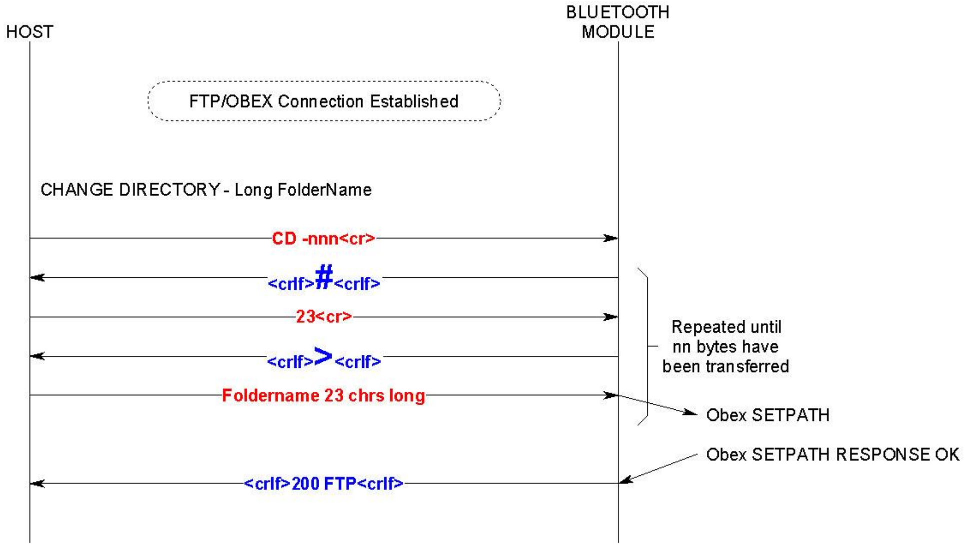


Figure 10: Change folder (long folder name - UNICODE)

4.8.10 Usage: Remove Folder (Short Folder name)

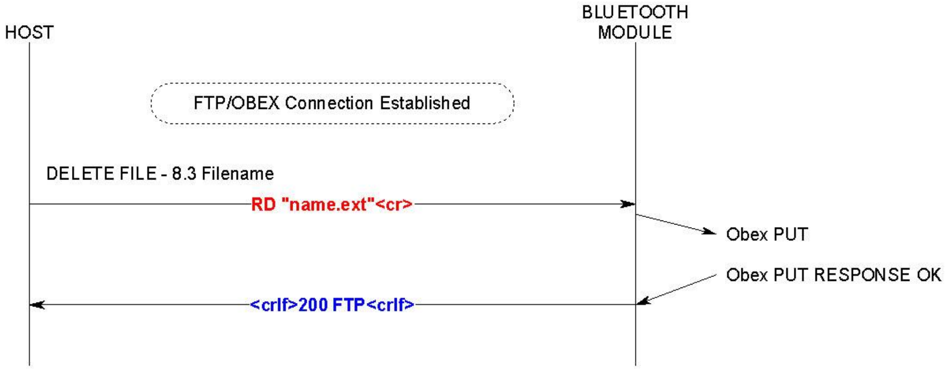


Figure 11: Remove folder (short folder name)

4.8.11 Usage: Remove Folder (Long Folder name – UNICODE)

Note: 'nn' is the size of folder name in bytes. The folder name is supplied in unicode.

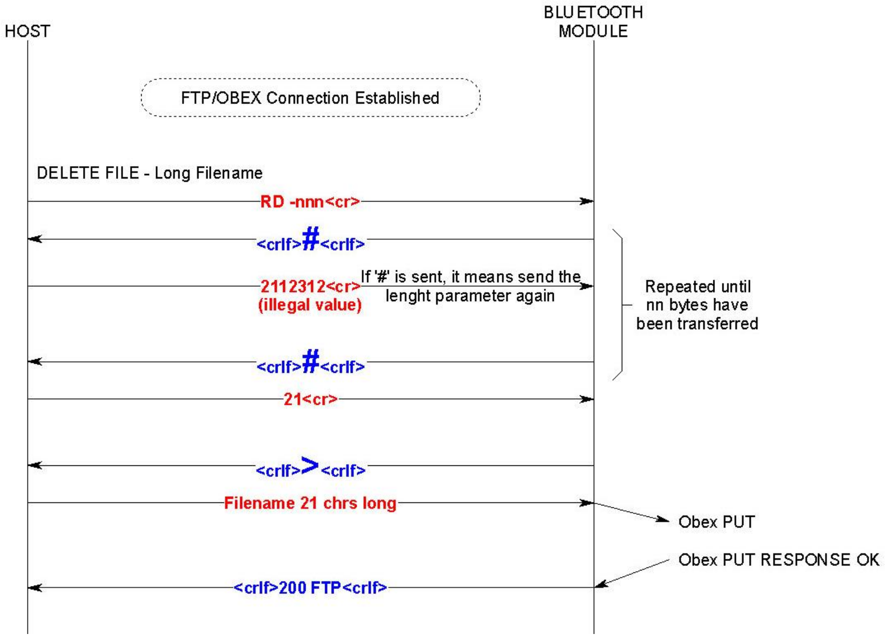


Figure 12: Remove folder (long folder name - UNICODE)

4.8.12 Usage: Delete File (Short Filename)

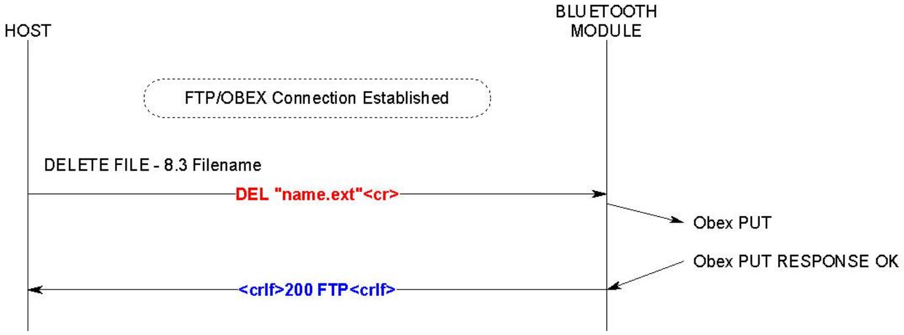


Figure 13: Delete file (short filename)

4.8.13 Usage: Delete File (Long Filename – UNICODE)

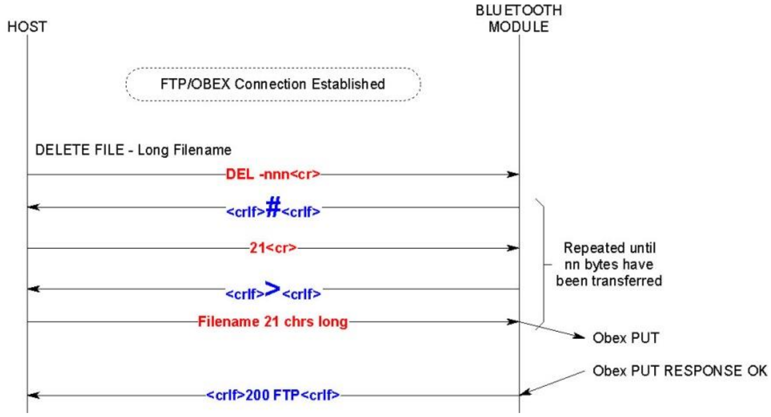


Figure 14: Delete file (long filename - UNICODE)

4.8.14 Usage: Put File (Short Filename)

Note: The first NN from the host shall specify a value of 1.
Subsequent NN values shall be less than the value returned to command MAX

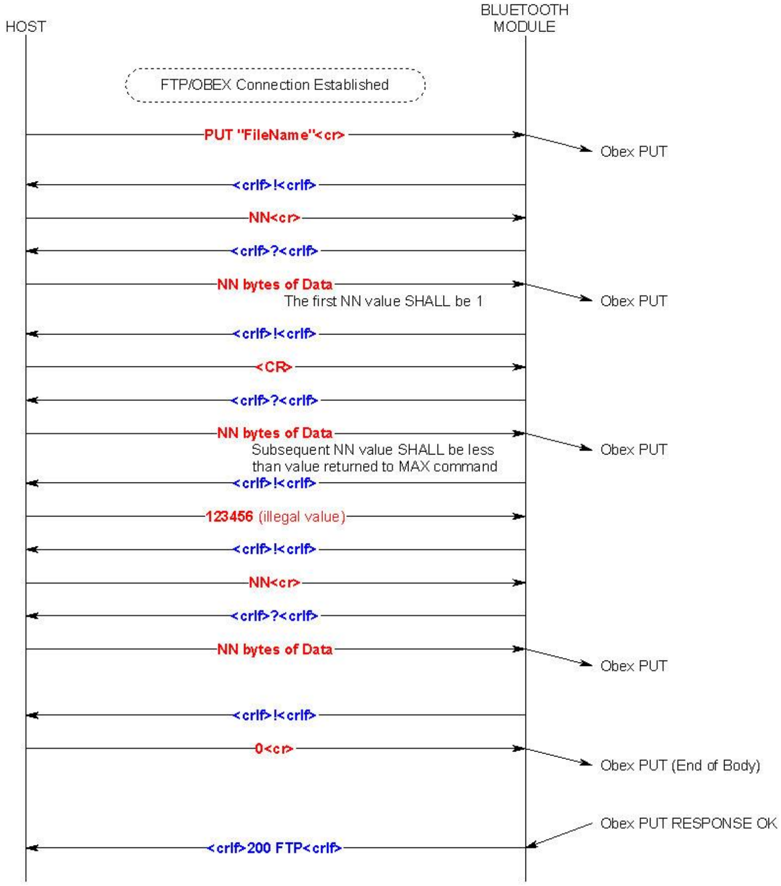


Figure 15: Put file (short filename)

4.8.15 Usage: Put File (Long Filename - UNICODE)

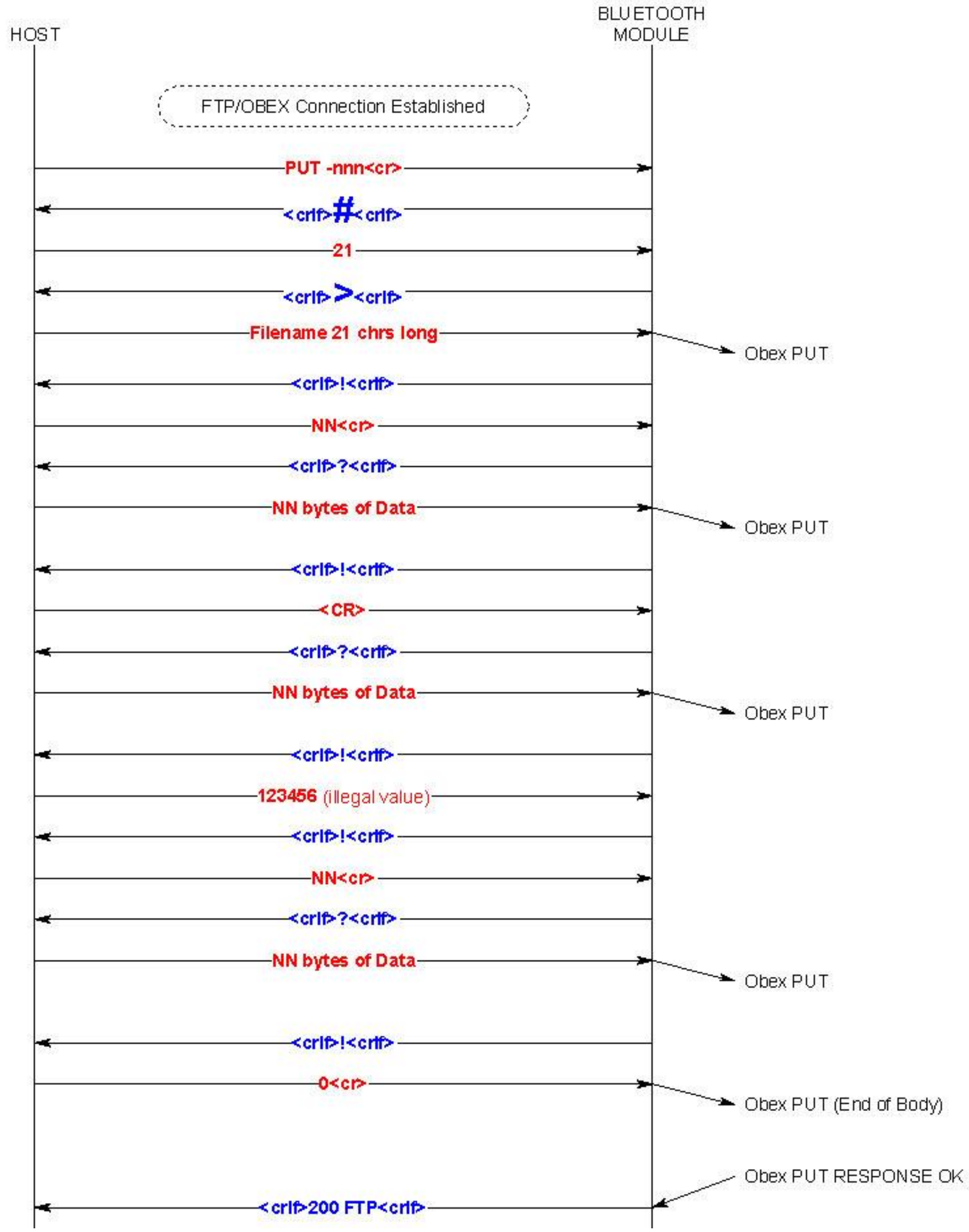


Figure 16: Put file (long filename - UNICODE)

4.8.16 Usage: Get File (Short Filename)

Note: After each !command, if a +NN response is not received after 2 seconds, the host can repeat that command.
The response to ! is "+NN" instead of just "NN" to make easier for the host to predict the command, since on completion the standard response is "200 FTP" which also happens to start with a number. The '+' shall be early warning to the host that the procedure is not complete.

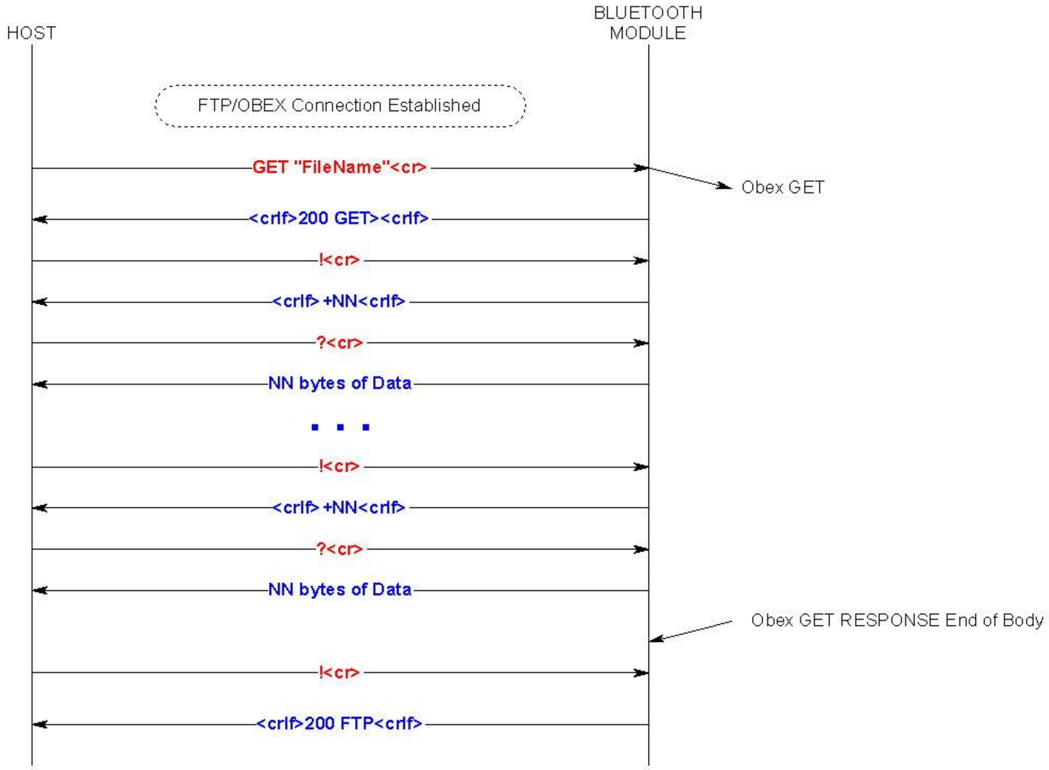


Figure 17: Get file (short filename)

4.8.17 Usage: Get File (Empty file in server)

Note: If the file size is 0, then the host shall receive "200 FTP" instead of "200 GET". The latter is invitation to the host that data needs to be marshalled across.

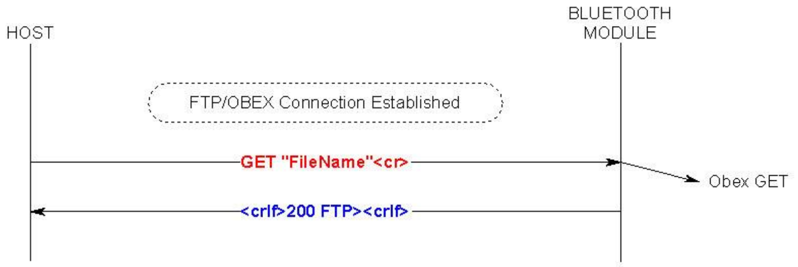


Figure 18: Get file (empty file in server)

4.8.18 Usage: Get File (Long Filename - UNICODE)

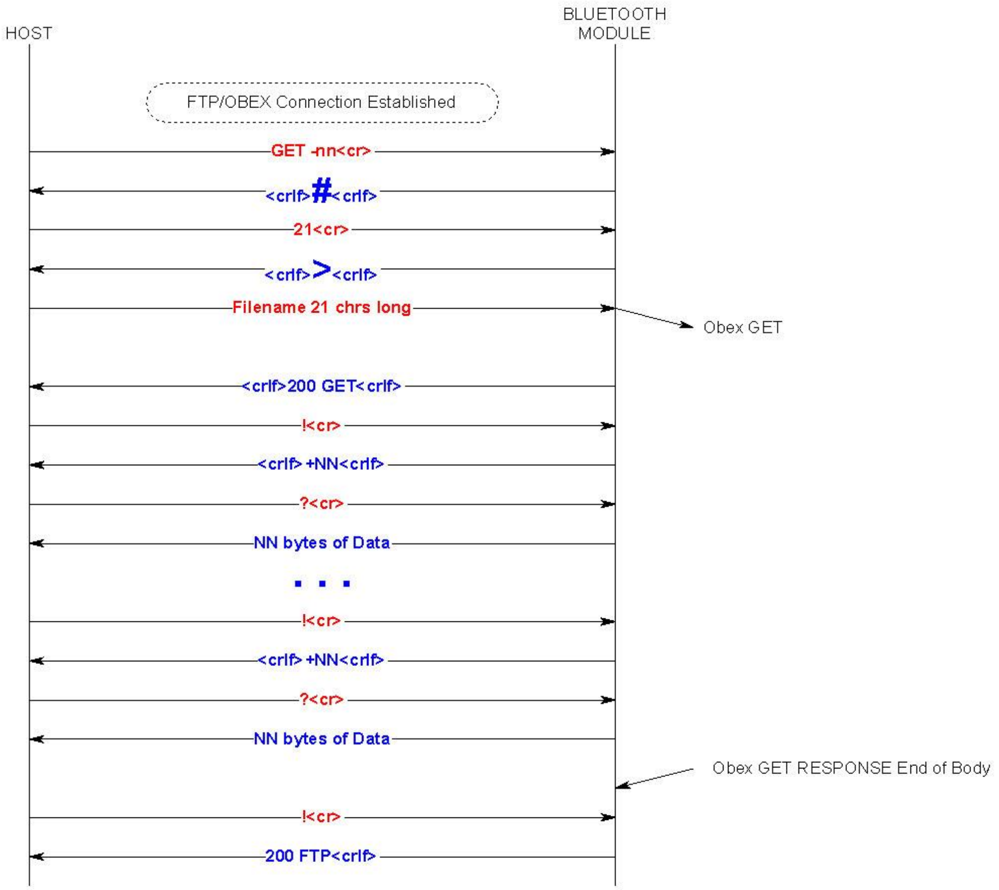


Figure 19: Get file (long filename - UNICODE)

4.8.19 Usage: Get File – ABORT

Note: If "200 FTP" is received after submitting an ABORT command then it implies that the entire file was transferred before the abort had been received.

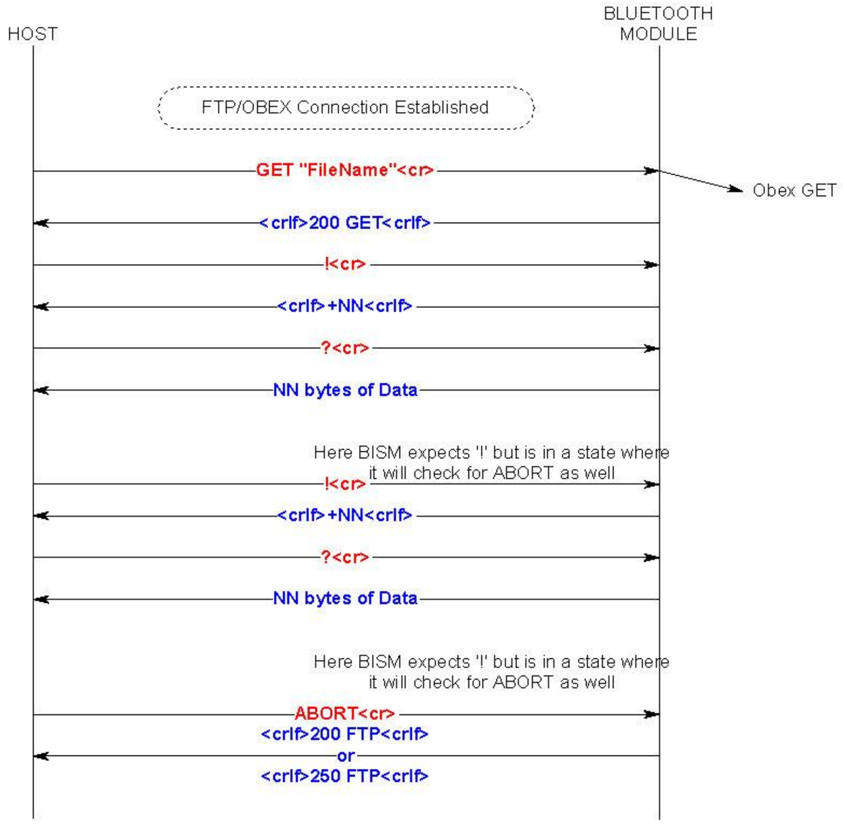


Figure 20: Get file - ABORT

4.8.20 Usage: Unsuccessful FTP connection

Note: Reason for connection failure could be:
Device is not in range
Device is not connectable
Device does not support FTP server profile.

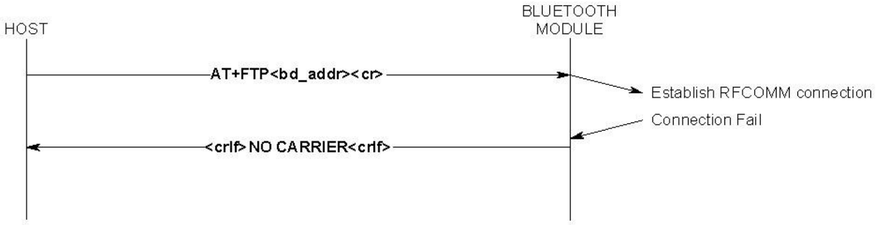


Figure 21: Unsuccessful FTP connection

4.8.21 Usage: ABORT a DIR request

Note: The host may get more file/folder names after submitting an ABORT request because the device could have received a folder data OBEX packet at the same time but slightly earlier so it may have started processing it.

Hence the host must look out for a "200 FTP<crlf>" to be sure that the DIR operation has terminated. Some ftp servers (like widcomm stack) will return a 500 response code.

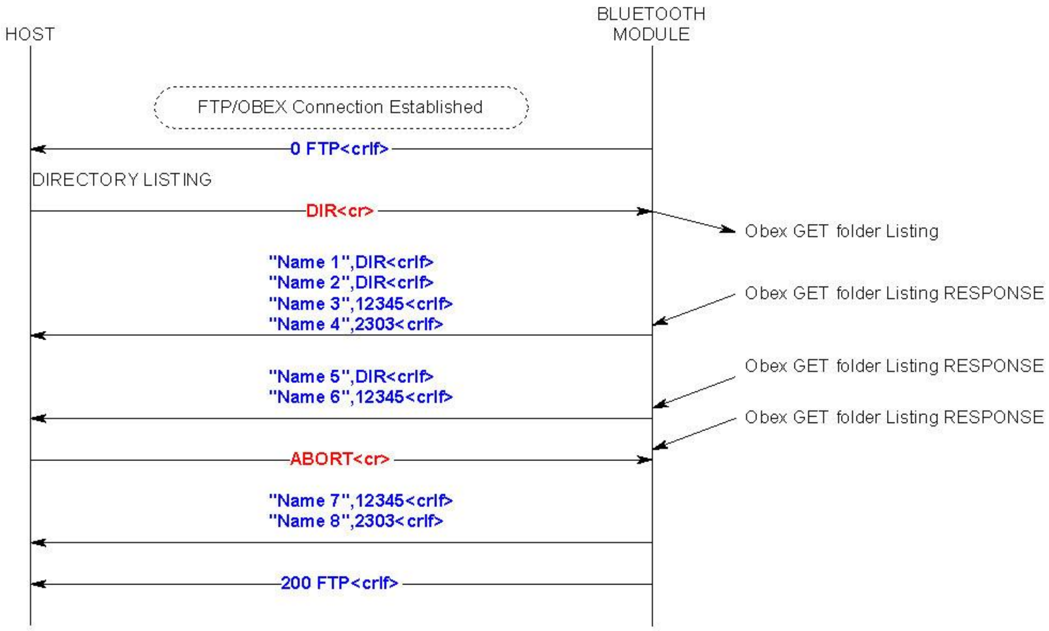


Figure 22: ABORT a DIR request

5 OBEX PROFILE COMMANDS

This section describes the OBEX implementation on BTM430/431 which allows performing the role of 'OBEX Push Client' as illustrated in the diagram below.

Note: OBEX push profile as implemented in this module is not BT SIG approved as it fails the OBEX Disconnect test case. The profile therefore cannot be used in end products, until it has been BT SIG certified. Please contact your local FAE for further information wireless.support@lairdtech.com

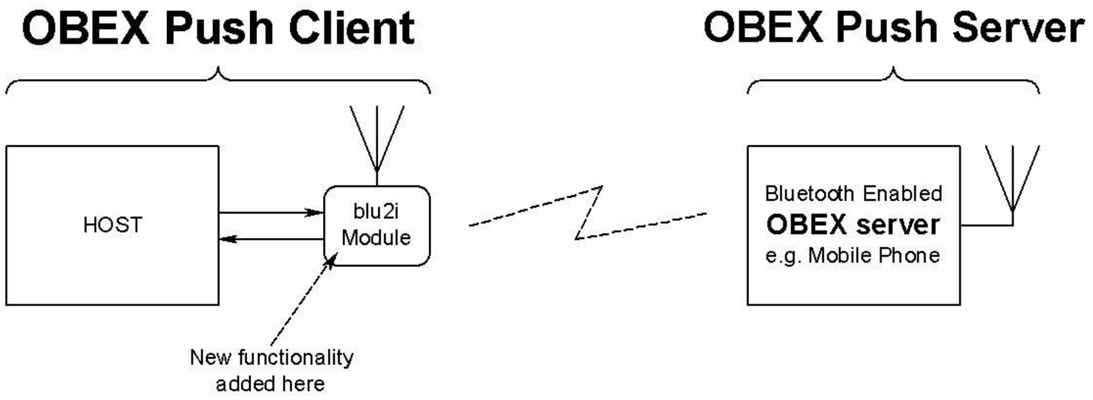


Figure 23: OBEX Push client and server

The OBEX Push profile implementation provides the capability to a host to act as an 'OBEX Push client' to a peer device providing an 'obex push server' profile.

An OBEX Push client capability implies the ability to send and optionally receive a default file. The Bluetooth OBEX Push server profile specification describes the profile as one built on OBEX Exchange (OBEX) which is in turn built on SPP.

The Object Push Profile [3] describes the profile as having the three following features:

- Object Push – Mandatory
- Business Card Pull – Optional
- Business Card Exchange – Optional

This implementation only offers the Mandatory Object Push feature.

The implementation on BTM430/431 allows a host attached to the module to send and receive files and to manipulate files and folders.

The format used in this document for describing this protocol is a series of message sequence charts with accompanying notes as appropriate, which unambiguously convey how a host and the module shall interact to perform the task.

5.3 Generic Notes and Guidance

In the message sequence charts the following abbreviations apply:

- <crlf> shall mean a two character sequence made up of the ASCII characters 0x0D (carriage return) and 0x0A (line feed).
- <cr> shall mean a one character sequence made up of the ASCII character 0x0D.

- <lf> shall mean a one character sequence made up of the ASCII character 0x0A.
- <bd_addr> shall mean a 12 digit string consisting of only hexadecimal digits 0-9, A-F, and a-f.
- <filename> shall mean a string delimited by the " character. For example, "hello.txt".
- <foldername> shall mean a string delimited by the " character.
- nnn shall be a decimal number with at least one digit.
- The backspace character is not supported.
- All OBEX commands are case sensitive.
- OBEX Commands shall not exceed 32 characters in total.
- While an OBEX session is open, the host shall not deassert the modules UART_CTS line and conversely the host MUST always be ready to accept data.
- When Unicode data is transmitted, it shall be assumed that the most significant byte is transmitted first.
- If an OBEX command is expecting a Unicode string as a parameter and the host has an ASCII string, the string shall be expanded with a 0 byte in the most significant position.

5.4 OBEX Push Related AT Commands

5.4.1 AT+OPS<bd_addr>

This command is used to establish a connection to an OBEX Push server profile in a peer device with Bluetooth address <bd_addr>.

5.4.2 ATSn=m

The following values of n are relevant to OBEX operation: 582. A description of these values is given in the S Register table (S582).

5.5 OBEX Push Related Subcommands

This section describes OBEX Push-related subcommands that the host can use to control the OBEX Push session.

5.5.1 PUT <filename> length(Send file)

This OBEX subcommand is used to send a file to the OBEX server.

The length of <filename> shall not exceed 24 characters.

The length value is inserted into the OBEX length header field.

There is some ambiguity as to how the first OBEX PUT packet is formed with respect to the 'Body'header. The OBEX specification does not prohibit the first 'Body' header to be empty, neither does it say that it must NOT be empty. If the first body is NOT empty when sending a file to a Nokia6820 phone then it seems to confuse it. Hence Laird sends out an empty 'Body' header by default as does the Windows PC based Widcomm Bluetoothstack.

To cater for future devices which NEED the first 'Body' header to be non-empty, a new S Register582 has been added to allow a host to have control over how the first body header is constructed.

The new S register 582 takes values in the range 0 to 1. The default value is 0 which implies that the first 'Body' header in the PUT OBEX packet is empty. A value of 1 forces that 'Body' header to have one byte of data – and in this case when the module prompts the host for a length value it shall respond accordingly.

Hopefully the default value of zero suffices for all occasions but, in case it does not, we provide the control to modify the packet as required.

5.5.2 PUT –nnnlength (Send file)

This OBEX subcommand is used to send a file to the OBEX push server where the filename is in UNICODE text and the filename is –nnn bytes long.

The length value is inserted into the OBEX length header field.

See comment above with regards to S Reg 582 (S582).

5.5.3 ABORT (Abort current OBEX command)

This OBEX subcommand is used to abort a file transfer where appropriate.

To abort a PUT session, please send 0 when the module prompts for a length value.

5.5.4 QUIT (Terminate the OBEX Push session)

This OBEX subcommand is used to terminate the OBEX push session and also results in the Bluetooth connection being terminated.

5.5.5 MAX (Max outgoing OBEX packet size)

This OBEX subcommand is used to get the maximum OBEX packet size tolerated by server and can be used by the host to optimize the data throughput.

5.5.6 WHO (Identify current profile)

This OBEX subcommand is used to identify the current profile. 0 means OBEX Push and 1 means FTP.

5.6 OBEX Push Related Subresponses

All OBEX Push-related sub responses sent from the module to the host are 11 characters long in the following format:

<cr><lf>nnn AAA<cr><lf>.

The characters nnn shall be decimal digits '0' to '9', then there is a space character and finally a 3 character word followed by <cr><lf>.

The <cr><lf> envelope plus the fixed length hopefully makes the parsing task in the host much simpler.

Values for 'nnn' are as specified for HTTP status codes and reproduced from the OBEX specification as indicated in Table 5-1.

Table 5-1: OBEX response and HTTP status codes

OBEX Response Code	HTTP Status Code	Definition
0x00to 0x0F	None	Reserved
0x10(0x90)	100	Continue
0x20(0xA0)	200	OK, Success
0x21(0xA1)	201	Created
0x22(0xA2)	202	Accepted
0x23(0xA3)	203	Non-Authoritative Information
0x24(0xA4)	204	No Content
0x25(0xA5)	205	Reset Content
0x26(0xA6)	206	Partial Content
0x30(0xB0)	300	Multiple Choices

OBEX Response Code	HTTP Status Code	Definition
0x31(0xB1)	301	Moved Permanently
0x32(0xB2)	302	Moved temporarily
0x33(0xB3)	303	See Other
0x34(0xB4)	304	Not modified
0x35(0xB5)	305	Use Proxy
0x40(0xC0)	400	Bad Request - server couldn't understand request
0x41(0xC1)	401	Unauthorized
0x42(0xC2)	402	Payment required
0x43(0xC3)	403	Forbidden - operation is understood but refused
0x44(0xC4)	404	Not Found
0x45(0xC5)	405	Method not allowed
0x46(0xC6)	406	Not Acceptable
0x47(0xC7)	407	Proxy Authentication required
0x48(0xC8)	408	Request Time Out
0x49(0xC9)	409	Conflict
0x4A(0xCA)	410	Gone
0x4B(0xCB)	411	Length Required
0x4C(0xCC)	412	Precondition failed
0x4D(0xCD)	413	Requested entity too large
0x4E(0xCE)	414	Request URL too large
0x4F(0xCF)	415	Unsupported media type
0x50(0xD0)	500	Internal Server Error
0x51(0xD1)	501	Not Implemented
0x52(0xD2)	502	Bad Gateway
0x53(0xD3)	503	Service Unavailable
0x54(0xD4)	504	Gateway Timeout
0x55(0xD5)	505	HTTP version not supported
0x60 (0xE0)	- - -	Database Full
0x61 (0xE1)	- - -	Database Locked

For more details of these values, Refer to the irDA specification which can be freely downloaded from www.irda.org.

In addition, values in the range 050 to 099 and 250 to 299 inclusive are specific to this Laird application and are defined as per the table below.

Response Code 'nnn'	(Laird Technologies Specific) Definition
050	Syntax Error / Command Unrecognized
051	Server sent unexpected information in OBEX packet
052	OBEX connection fail, because it is unauthorized
053	Memory allocation failure (Please contact Laird with details)
055	Unicode File/Folder name length cannot be an odd value
056	Command not recognized
090	An XML parsing error occurred (while processing response to DIR command)
099	The Bluetooth connection has unexpected been dropped (i.e., remote out of range etc.)
250	GET procedure was aborted

5.6.1 nnn OBX

This OBEX response is used to convey the outcome of a subcommand to the host where 'nnn' is a fixed three digit decimal number as defined in the irDA specification and map to HTTP status codes.

For example, a value of n=200 implies successful operation, any other value usually conveys an error as described in the irDA specification (except the range 050 to 099 inclusive and 250 to 299 inclusive).

Note: When a Bluetooth OBEX session is established, we will be specifying v1.0 in the header packets by default.

5.7 OBEX Push Line Multiplexing Commands

When transferring an object, the single serial interface between the host and the module is used to send and receive data and commands. This means a scheme is required to unambiguously determine when a byte on the line corresponds to a command or data belonging to a file.

The module uses negotiated multiplexing to achieve this, and commands are used to toggle the line between command and data mode.

This scheme is symmetrical and the commands for toggling the state of the line are relevant for both direction. The only difference being that the terminator is <cr> in the host to module direction and <crlf> in the reverse direction.

The commands are described in the following sub sections.

5.7.1

This command is used to ask the other end how many bytes of a filename or object it will send next.

5.7.2 >

This command is used to inform the other end that it is safe to send the number of bytes belonging to a filename or object as indicated in the most recent # command.

5.7.3 !

This command is used to ask the module/host how many bytes of data it will send next.

If the module/host sends a length value that is too large to handle, then it can be rejected by resending the !command. This is because accepting a value implies this end should send a ? prompt to trigger the data phase (see Section 5.7.4).

5.7.4 ?

This command is used to inform the other end that it is safe to send the number of bytes belonging to 'data' phase indicated in the most recent ! command.

5.8 Message Sequence Charts

The color scheme uses RED text as commands from a host to the module and BLUE text as responses and prompts from the module to the host. Command/responses shown in BLACK are associated with non-OBEX related states.

Apart from OBEX connection and disconnection scenarios, to avoid repetition, all message sequence charts shall assume that the module is in an OBEX connected state.

5.8.1 Usage: Make OBEX PUSH Connection (No Authentication)

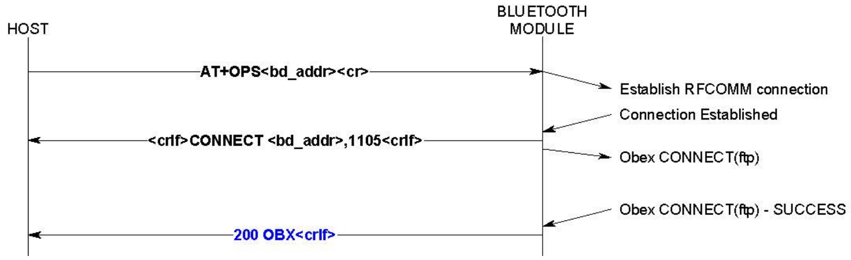


Figure 24: Make OBEX PUSH connection (no authentication)

5.8.2 Usage: OBEX Push Disconnection

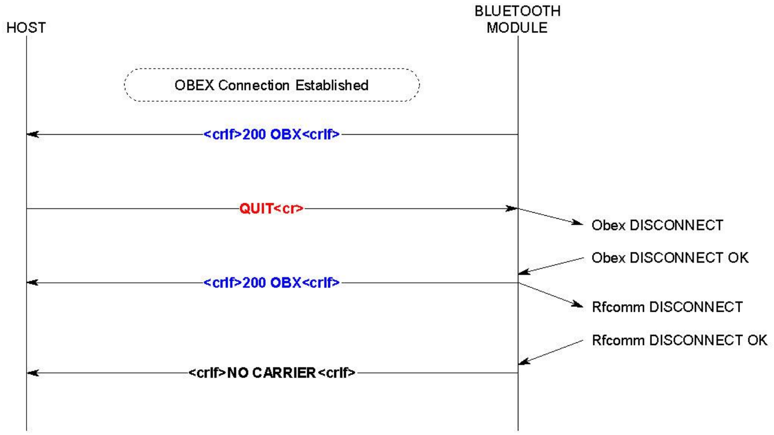


Figure 25: OBEX Push disconnection

5.8.3 Usage: Put File (Short Filename)

Note: The first NN from the host shall specify a value of 1.
Subsequent NN values shall be less than the value returned to command MAX

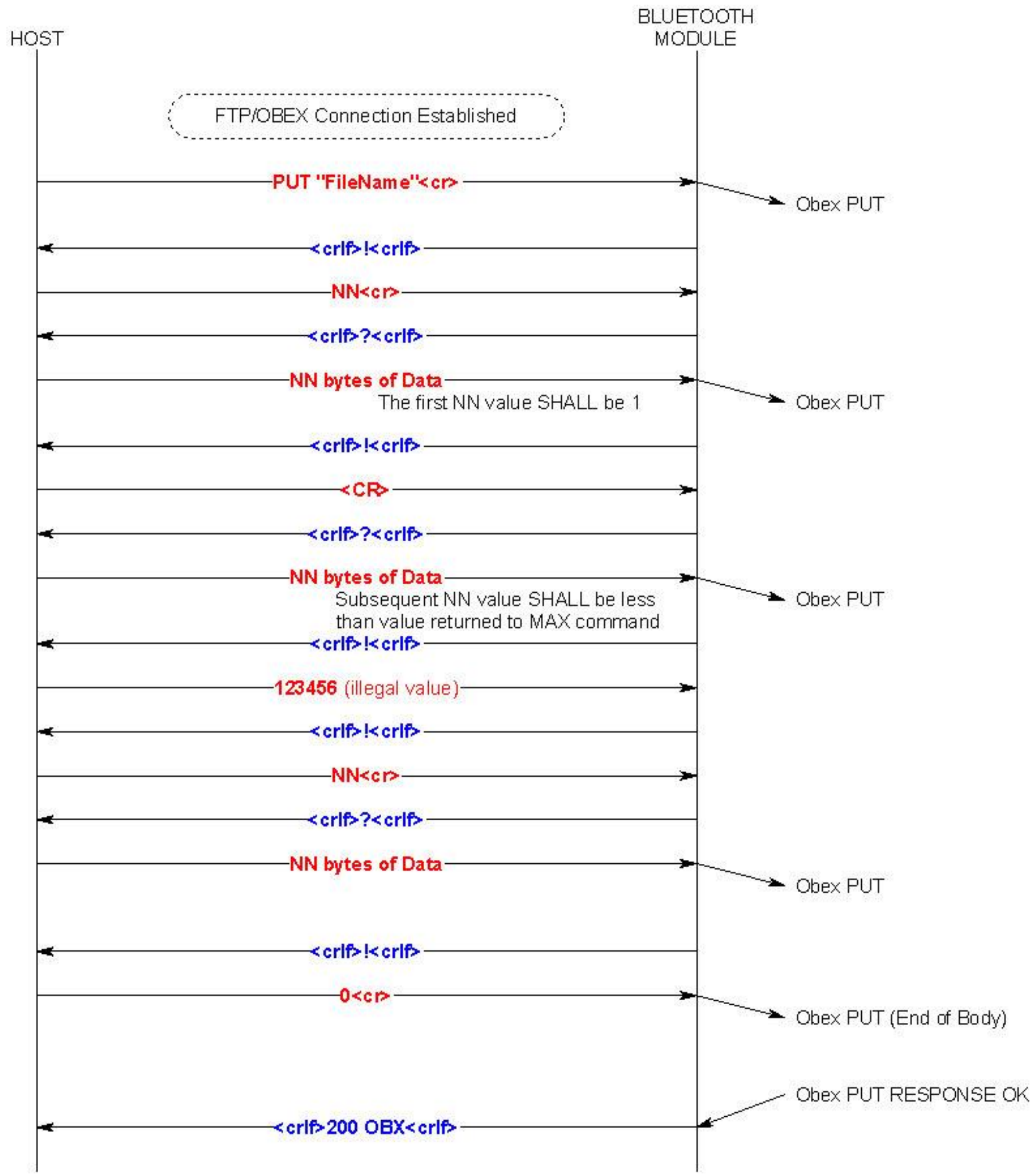


Figure 26: Put file (short filename)

5.8.4 Usage: Put File (Long Filename - UNICODE)

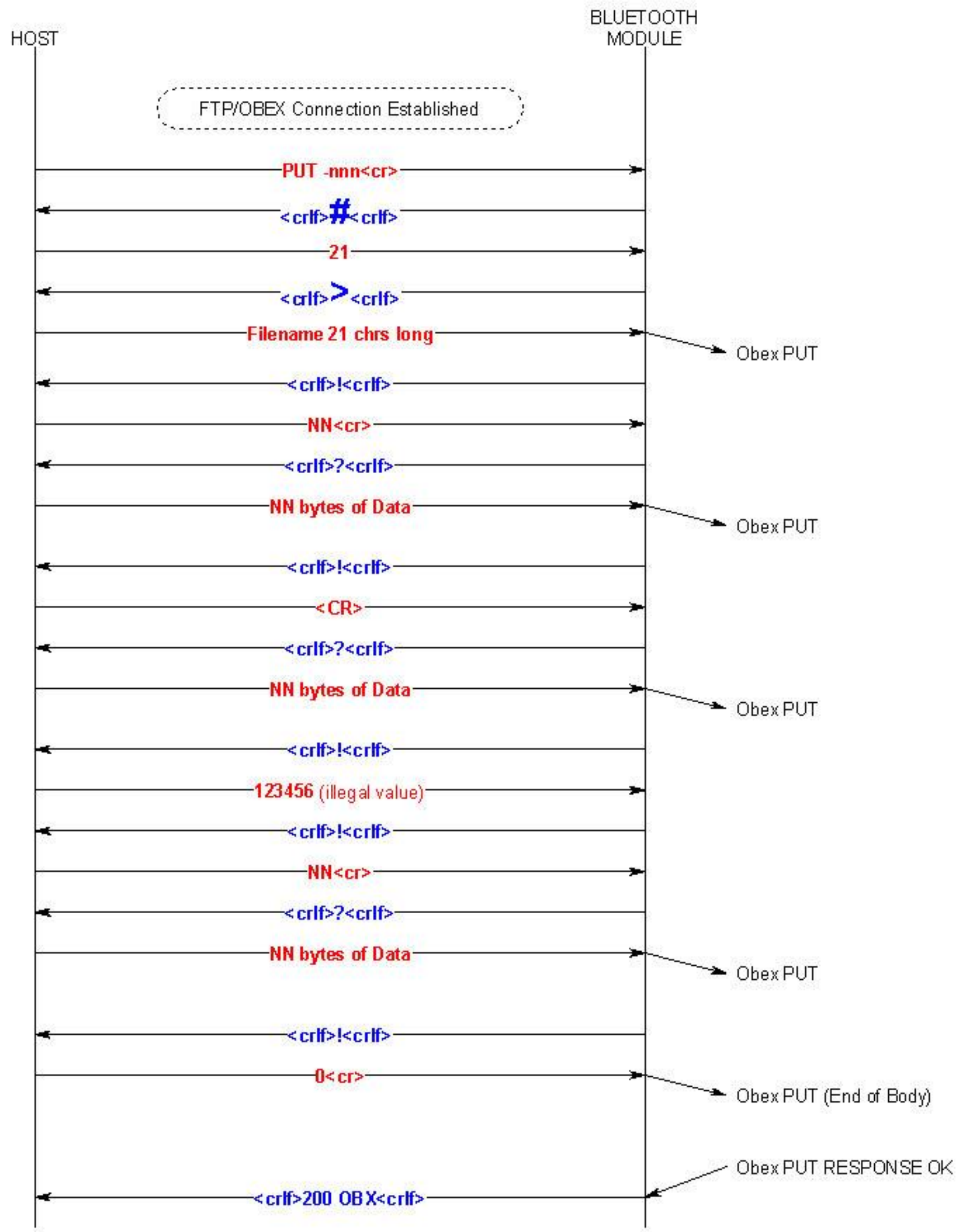


Figure 27: Put file (long filename - UNICODE)

5.8.5 Usage: Unsuccessful OBEX Push connection

Note: Reason for connection failure could be:
Device is not in range,
Device is not connectable,
Device does not support OBEX Push server profile.

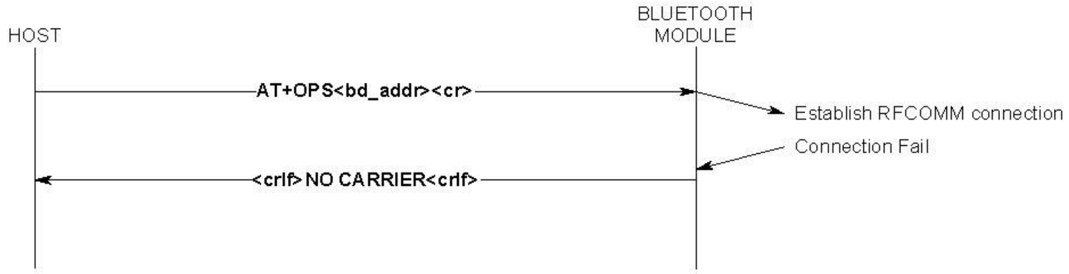


Figure 28: Unsuccessful OBEX Push connection

Smart Technology. Delivered.

Laird Technologies is the world leader in the design and manufacture of customized, performance-critical products for wireless and other advanced electronics applications.

Laird Technologies partners with its customers to find solutions for applications in various industries such as:

- Network Equipment
- Telecommunications
- Data Communications
- Automotive Electronics
- Computers
- Aerospace
- Military
- Medical Equipment
- Consumer Electronics

Laird Technologies offers its customers unique product solutions, dedication to research and development, as well as a seamless network of manufacturing and customer support facilities across the globe.

globalsolutions: **local** support™

USA: +1.800.492.2320
Europe: +44.1628.858.940
Asia: +852.2923-0610
wirelessinfo@lairdtech.com
www.lairdtech.com/wireless