SERIAL-TO-WIFI ADAPTER

APPLICATION PROGRAMMING GUIDE

Reference: GS-S2WF-APG

Version: SP-5.2

Date: *26-May-11*

Version	Date	Remarks
1.0	17 Nov 2009	Initial release. Applies to S2WiFi version 1.0.9 and above.
2.0	8 June 2010	▶ Section (3): add Raw Data mode. ▶ Section 4: change comma position to allow option of no BSSID but Ch provided. ▶ Section (4): add Raw Ethernet command.
2.1	9 July 2010	Reconciled drafts including new or updated sections: ▶ Unsolicited Data Handling ▶ WPS ▶ EAP-Configuration ▶ EAP ▶ Certificate Addition ▶ Certificate Deletion ▶ Sync Loss Interval ▶ External PA ▶ Association Keep Alive Timer ▶ Socket Options ▶ SSL Connections ▶ Closing SSL ▶ HTTP Client Config ▶ HTTP Client Connection ▶ HTTP Client GET/POST ▶ Closing HTTP Client ▶ Unsolicited Data Transmission ▶ Provisioning ▶ Web Provisioning ▶ Web Logo Provisioning ▶ Set System Time ▶ GPIO Out HIGH/LOW
2.2	12 August 2010	▶ Battery Warning Level Set ▶ RF Test

		▶ Asynchronous Frame Transmission ▶ Asynchronous Frame Reception ▶ Modulated and Un-Modulated Wave transmission ▶ Frame Transmission/Reception Stop ▶ External PA Auto Detection ▶ Regulatory Domain Configuration ▶ Regulatory Domain Information General improvements to document layout and language
2.3	3-Nov-10	▶ Added GET System Time ▶ Error Counts
2.4	8-Nov-10	▶ Fixed WPAPSK command language
3.0	9-Nov-10	GA
3.1	11-Nov-10	▶ Enhanced the RFT sections of this guide with additional information and examples.
4.0	18-Nov-10	▶ Factory Default MAC Settings ▶ SPI Interface Configuration and Handling Information
4.1	29-Nov-10	▶ Exception Messages ▶ Boot Messages
4.3	22-Dec-10	▶ Document is updated for S2W ver 2.2.4 ▶ Bulk Data Mode for Tx and Rx ▶ AT? No longer supported ▶ Added header GSN_HTTP_REQUEST_URL (23) to HTTP client list of accepted values ▶ Added "proxy" parameter to AT+HTTPOPEN=<host >, <Port Number>, [<SSL Flag>, <certificate name>,<proxy>] command. ▶ General corrections of language and layout
4.4	11-Jan-11	▶ Rev version for sync control
4.5	14-Jan-11	▶ Corrected Escape Sequence and Response Code table errors
4.6	17-Jan-11	▶ Removed <esc> Q and <esc> T

		▶ General corrections ▶ Added SSID and Passphrase Materials
4.7	18-Jan-11	▶ Updates to Unsolicited Data Transmission section
4.8	25-Jan-11	▶ General corrections made to Command Processing Mode ▶ Expanded SSID and Passphrase section ▶ General corrections
4.9 - 4.10	1-Feb-11	▶ General corrections and clarifications to SSID and PassPhrase ▶ Corrected command structure for Scanning
4.11	12-Feb-11	▶ Made clarifications to Transmit Power command information
4.12	21-Mar-11	▶ Document is updated for S2W ver 2.2.10 ▶ Asynchronous Messages ▶ Security Configuration ▶ Node Startup Handling ▶ Pin Connection for SPI ▶ Enhanced Asynchronous Notifications ▶ General improvements and coverage for how to compile for SPI
4.13	27-Mar-11	▶ Correction to Store Network Context
5.0	28-Mar-11	GA
5.1	22-Apr-11	▶ Added Appendix 6 Data handling using Esc Sequences on UART and SPI Interface. ▶ Updated section 3.2: Command Processing Mode ▶ Updated section 3.4.1: Bulk Data Mode
5.2	26-May-11	▶ Document is updated for S2W ver 2.3.1 ▶ Added DHCP Server Material ▶ Added DNS Server and Lookup Client Material ▶ Added Store Network Context Material ▶ Added Memory Trace Material
5.3	14-Jun-11	▶ Updated ASCII codes in 3.7.3 to decimal instead of Hex per S2W code

GainSpan Corporation
125 South Market Street, Suite 400
San Jose, CA 95113
U.S.A.

+1 (408) 673-2900

info@GainSpan.com
www.GainSpan.com

Table of Contents

Figures

Tables

1 System Overview

1.1 Purpose

This document describes the operation and serial command interface for the GainSpan System-On-Chip (SOC) ***Serial2WiFi Adapter.*** The Serial2WiFi Adapter enables embedded devices with a UART/SPI interface to gain access to an IP network over an 802.11-compliant (Wi-Fi®) wireless network connection, using only serial commands.

1.2 Scope

This document reviews the architecture of the Serial2WiFi software and provides the programmer with necessary command syntax required to manage the Wi-Fi interface and send and receive network messages. This document assumes that the reader is generally familiar with GainSpan SOC products, Internet Protocol (IP) networks and the operation and management of 802.11 wireless devices.

1.3 Overview

The Serial2WiFi stack is used to provide Wi-Fi Capability to any devices having a serial interface. This approach offloads WLAN, TCP/IP stack and network management overhead to the Wi-Fi chip, allowing a small embedded host, based on such low-cost microcontrollers as the 8051, PIC, MSP430, or AVR to communicate with other hosts on the network using a Wi-Fi wireless link. The host processor can use serial commands to configure the Serial2WiFi Adapter and to create wireless and network connections.

1.4 Terminology

Table 1: Glossary of Terms

Term	*Explanation*
AP	Access Point
API	Application Programmer's Interface
BSSID	Basic Service Set Identifier
CID	Connection Identifier
CPL	Clock Polarity
CPH	Clock Phase
DHCP	Dynamic Host Configuration Protocol
DIN	Data Input
DOUT	Data Output
IP	Internet Protocol
MSPI	Master SPI
MTU	Maximum Transfer Unit
PSK	Pre-shared key
RSSI	Received Signal Strength Indication
SSID	Service Set Identifier
SPI	Serial Peripheral Interface
SSPI	Slave SPI
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access

Term	Explanation
PA	Power Amplifier

1.5 Standards

The following standards and conventions are considered in this design:

- IEEE 802.11 a/b/g
- ITU V.25ter AT Command Set

2 Interface Architecture

The overall architecture of the Serial2WiFi interface is depicted in Figure 1. Tx and Rx Data Handlers pass messages to, and from, the TCP/IP network. Commands related to management of the Serial2WiFi interface and the network connections are intercepted by a Command Processor. A Serial Data Handler translates data to and from a UART/SPI-compatible format.

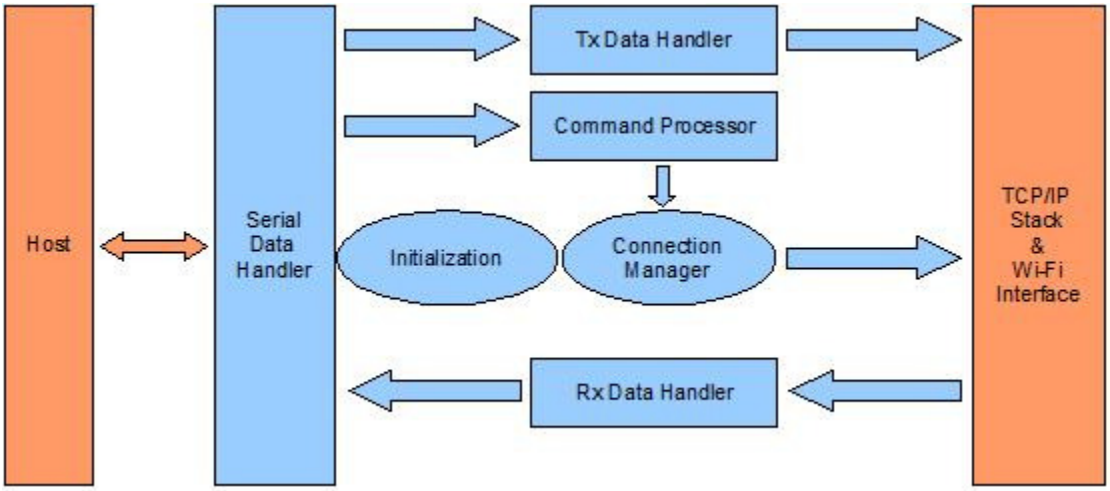

Figure 1: Overall Architecture of the Adapter

The system is composed of the following modules:

- System Initialization (section 3.1)
- Command Processor (section 3.2)
- Data Handlers (section 3.4)
- Serial Data Handler (section 3.5)
- Network Connection Manager (section 3.6)
- Wireless Connection Management (section 3.7)

The software for the Serial2WiFi Adapter is mainly driven using a state machine. Upon powering on, the required initialization of all the modules is performed and then the state machine is entered. This state machine is event-driven and processes the events received from either the serial port or from the Wi-Fi / Network interface as well as internal events from its own modules. The state machine calls the appropriate handler for a given event per the current state.

The Serial2WiFi Adapter has three distinct operating modes (Figure 2). In the default ***command processing operating*** mode, commands to configure and manage the interface are sent over the serial interface. In the default mode, the node accepts commands entered by the Host CPU and processes each

of the commands. All commands are available in this mode. The User can establish a data connection here and send data.

In ***auto connection*** mode, data sent over the serial interface is transparently sent over the IP network to a single, pre-configured IP address/port pair, where data from that address is transparently sent over the UART/SPI to the serial host. With Auto mode, the IP Layer connections are already established and the data is sent directly to the target destination. In this mode, the node does not accept all commands. To accept commands the node needs to be brought back to "Command Processing" mode by pressing an escape sequence.

In ***data processing*** mode, data can be sent to, or received from, any of 16 possible connections. Each connection consists of a TCP or UDP path to a destination IP address and port. Auto connection mode is entered using a serial command (section 4.13.4) and terminated using a special escape sequence (section 3.4).

For each mode, configuration parameters are stored in non-volatile memory. In addition to factory-default parameter values, two user-defined profiles (0 and 1) are available. The parameter set to be used is determined by a user command (section 4.6.3).

3 Adapter Description

3.1 System Initialization

Upon startup, the Serial2WiFi interface performs the following actions, depicted graphically in Figure 2.

- During the initialization process, the module will search for a saved configuration file. The configuration file include the auto connection settings, default profile and profile settings. If a saved configuration file is available, it is loaded from non-volatile memory. If no saved configuration file, the default settings will be applied. If there are no saved parameters, the factory-default configuration is loaded.
- The Serial2WiFi application is initialized based on the profile settings.
- If auto connection is enabled, the interface will attempt to associate with the specified network, previously set by the user (section 4.13.1). Once associated, it will establish a TCP or UDP connection within the specified parameters. If successful, the interface will enter the Auto Connect mode, where all data received on the serial port is transmitted to the network destination and vice versa.
- If auto-connection is disabled or fails, the interface enters the command processing state.

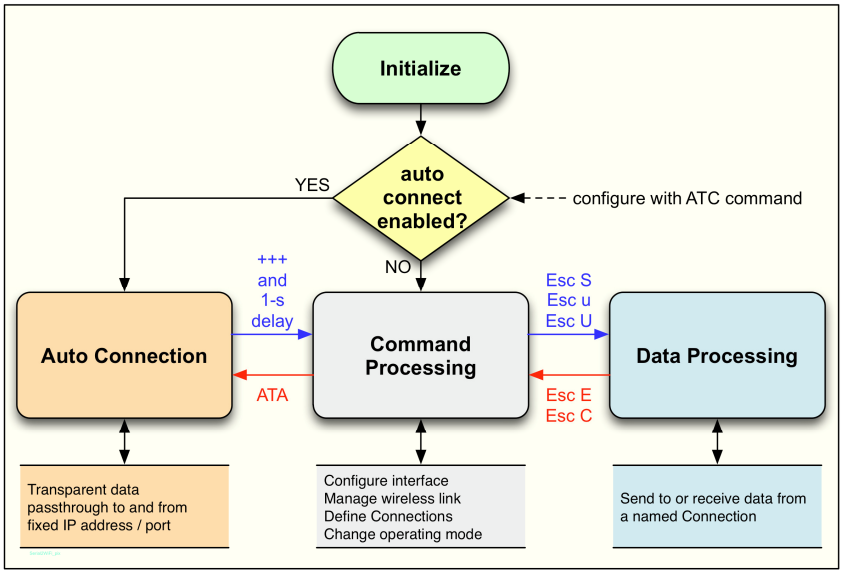

Figure 2: Operating Modes of the Adapter

Upon power-up, the UART interface defaults to 9600 baud, using 8 bit characters with no parity bits and one stop bit. Similarly SPI interface defaults to Mode#0 (CPL=0, CPH=0) Any changes to this configuration that were made in a previous session using the `ATB` command (section 4.2.1) will be lost when power is lost. To make changes in the UART/SPI parameters that will persist across power cycling, the relevant changes must be saved into the power-on profile using `AT&W` (section 4.6.1) and `AT&Y` (section 4.6.3).

3.1.1 External PA Auto Detection

Upon startup, the Serial2WiFi interface performs an auto detection of External PA. This detection is done through the GPIO pin 12. If this GPIO is "high" during startup, meaning the external PA is present; the adapter enables the external PA and forces the adapter to go into and out of standby mode for a moment just to make any changes effective for the external PA configuration.

3.1.2 Network Configurations

Once associated, the adapter supports instances of four types of network entities: TCP client, TCP server, UDP client and UDP server. Each client, or server, is associated with one or more of a possible 16 ***Connection Identifiers***, where the CID is a single hexadecimal number. More than one such entity can exist simultaneously; and a TCP server can have multiple connections, each with its own CID. When the adapter is in Auto Connect mode (section 3.3), the entity called for by the Profile is created automatically upon startup. In Command modes, servers and clients are created using specific serial commands (section 4.10).

A TCP client (Figure 3) is created with the serial command `AT+NCTCP` (section 4.10.1). The client attempts to create a TCP network connection with the destination IP address and port specified within the command. If successful, it issues a CONNECT response with the CID of the client. Data can then be sent to the remote server using the `<Esc>S` sequence (section3.4) with the appropriate CID. Data from the server is passed back to the Host, with the CID to identify its source.

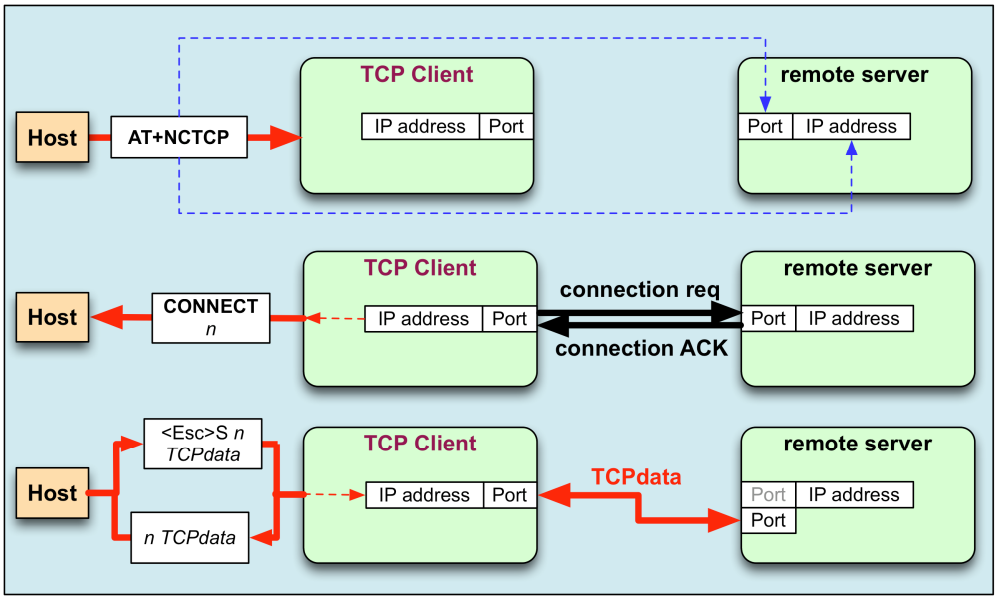

Figure 3: Creation and Use of a TCP Client

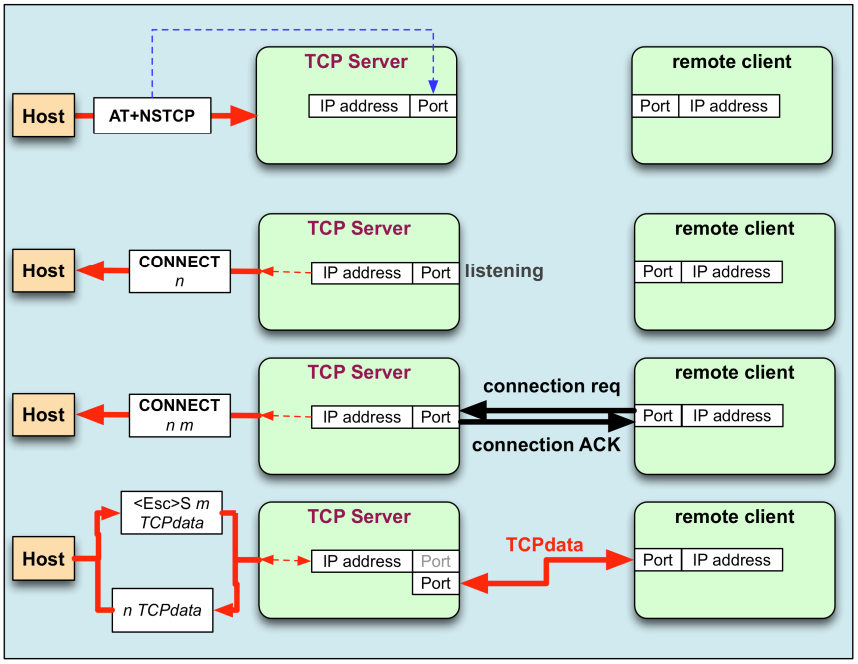

Figure 4: Creation and Use of a TCP Server

Figure 4 schematically depicts the corresponding sequence for a TCP server. A server is created with the serial command `AT+NSTCP`; it receives a CID, but listens passively until a remote client requests a connection. If that connection is successfully created, a second `CONNECT` message and a new CID are provided to the Host. It is this second CID that is used to send data to the remote client and identify received data from that client. A TCP server may support multiple clients, each with a unique CID.

A UDP client's life is depicted in Figure 5. The client is created with the serial command `AT+NCUDP` and receives a CID. The UDP client is associated with a specific destination port and address.

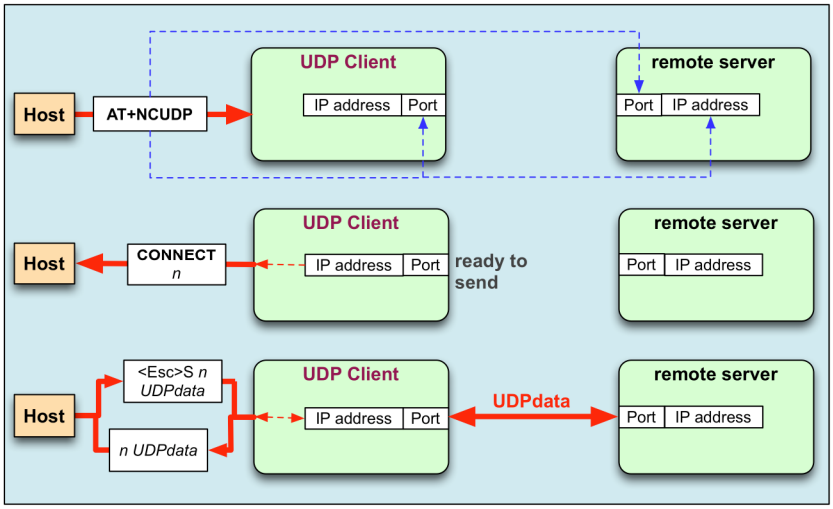

Figure 5: Creation and Use of a UDP Client

Finally, Figure 6 shows a UDP server. The server is created with `AT+NSUDP` and is assigned a CID. Individual clients do not receive unique CIDs; data sent using the UDP server must be accompanied with the destination IP address and port, and data received via the server is modified with the identifying source address and port number.

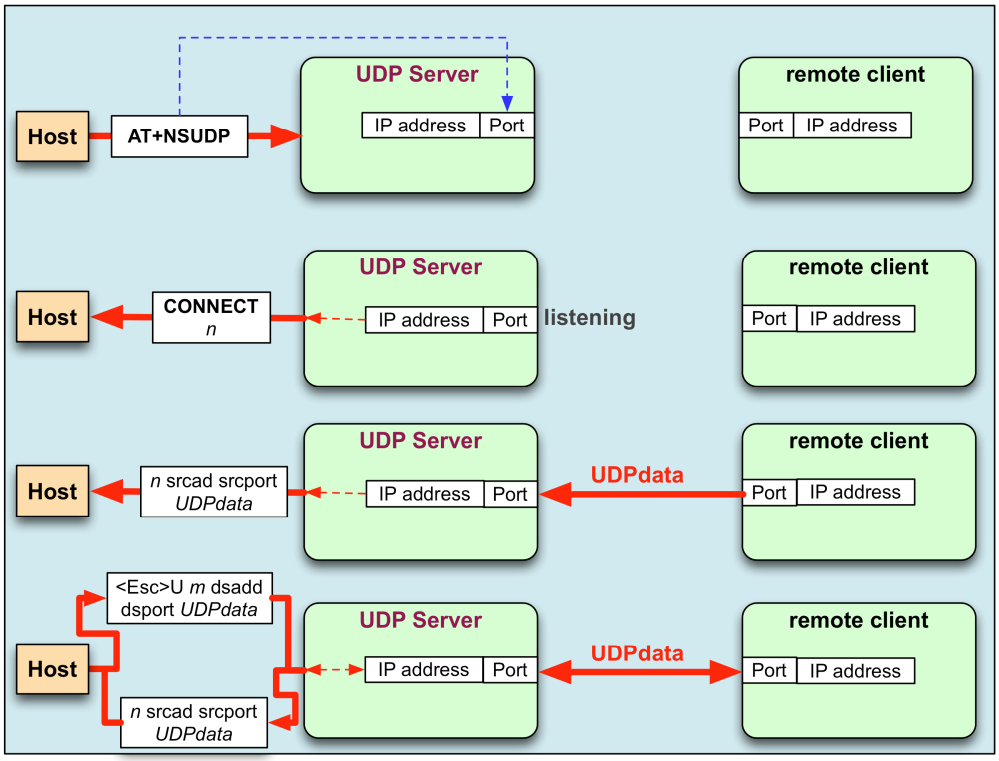

Figure 6: Creation and Use of a UDP Server

3.1.3 Profile Definition

The configuration parameter values that define the behavior of the Adapter are grouped into Profiles. These profiles are stored in non-volatile memory when not in use. The default configuration supports two Profiles. The contents of a profile are listed in Table 2.

Table 2: Profile Parameters

Parameter	*Values*	*Reference*
General Wireless Parameters		
802.11 Operating Mode	BSS, IBSS, Limited AP	4.7.6
Transmit Power Configuration		4.8.14
802.11 Transmit Retry Count		4.7.13
Power Save Mode	Enabled, Disabled	4.8.12
802.11 Radio Mode	Enabled, Disabled	4.8.11
Auto Connect Mode, Wireless Interface Settings		
802.11 Operating Mode	BSS, IBSS	4.13.1
Operating Channel	1 to 14	4.13.1
SSID Parameter	Any valid SSID	4.13.1
BSSID Parameter	Any valid BSSID	4.13.1
Maximum Scan Time		4.4
Auto Connect Mode, Network Interface Settings		
Mode	Server, Client	4.13.2
Protocol	TCP, UDP	4.13.2
Server Port Number	Any valid port	4.13.2
Server IP Address	Any valid IP address	4.13.2

Parameter	Values	Reference
Wireless Interface Security Configuration		
Authentication Mode	Open, Shared	4.8.1
PSK Valid	Valid, Invalid	4.8.5
PSK-SSID	Any valid SSID; used for PSK key computation.	4.8.5
WEP Key Configuration		4.8.3
WPA Passphrase		4.8.4
TCP/IP Configuration		
DHCP Mode	Enabled, Disabled	4.9.2
IP Address	Valid IP address	4.9.3
Net Mask Address	Valid mask	4.9.3
Default Gateway Address	Valid IP address	4.9.3
DNS1	Valid DNS1 IP address	4.9.7
DNS2	Valid DNS2 IP address	4.9.7
UART Configuration		
Echo Mode	Enabled, Disabled	4.1.2
Verbose Mode	Enabled, Disabled	4.1.3
Bits Per Character	5,6,7,8	4.2.1
Number of Stop Bits	1,2	4.2.1
Parity Type	No, Odd, Even	4.2.1
Software Flow Control Mode	Enabled, Disabled	4.2.2
Hardware Flow Control Mode	Enabled, Disabled	4.2.3
Baud Rate		4.2.1

Parameter	*Values*	*Reference*
Limits and Timeouts		
Network Connection Timeout	Units of 10 milliseconds	4.4
Auto Association Timeout	Units of 10 milliseconds	4.4
TCP Connection Timeout	Units of 10 milliseconds	4.4
Association Retry Count		4.4
Nagle Wait Time	Units of 10 milliseconds	4.4
SPI Configuration		
SPI clock polarity and clock phase	0,1	4.3.1

3.2 Command Processing Mode

In command mode, the application receives commands over the serial port. Commands are processed line by line. "Verbose Mode", when referring to commands being executing, refers to the displaying of status of any command executed in ASCII (human readable) format. When the verbose mode is disabled, the output will simply be in numeric digits, each digit indicating a particular status. Verbose Mode is enabled by default.

- If "echo" is enabled then each character is echoed back on the serial port
- Each command is terminated with a ***carriage return*** <CR> or ***line feed*** <LF>
- Each response is started with a carriage return <CR> and line feed<LF>, with the exception of the responses to the following commands:

  a) The response to the following group of commands starts with a line feed <LF> only:

  - AT+WA
  - AT+NSTAT
  - AT+WPAPSK=<SSID>,<Passphrase>
  - AT+NSET=<IP Address>,<Subnet Mask>,<Gateway IP Address>
  - AT+TRACEROUTE=<IP Address>
  - AT+PING=<IP Address>
  - ATA
  - AT+NDHCP after association

  b) The response to the following group of commands starts with a line feed and carriage return: <LF><CR>.

  - AT+SETTIME=<dd/mm/yyyy>,<hh:mm:ss>
  - AT+HTTPOPEN=<IP Address>

- Each response is terminated with a carriage return <CR> and line feed <LF>
- If the characters "A" and "/" are entered at the beginning of a line (after <CRLF>), then the previous command is executed
- Once a complete line (ending with <CR or LF>) is entered, then the command contained therein is processed and an appropriate response returned

Unless otherwise specified, if verbose mode is enabled, then the response to a successful command is the characters "OK". The response to an unsuccessful command is the word "ERROR", followed by a

detailed error message, if available. If verbose mode is disabled, command responses is numerical with OK having a value of 0 and error codes represented by positive integers.

The commands are described in Section 4. Possible response codes are described in 3.7.3

3.3 Auto Connection

If auto connection is enabled (section 0), then upon startup the Adapter will:

- Attempt to associate to or from the specified network, for a maximum time of *Auto Associate Timeout* (section 4.4)
- On successful association, attempt to establish a network connection based on the specified parameters
- On successful connection establishment, enter the pass-through auto connect mode
- On failure, enter the command processing state

In TCP client mode, the connection is considered established only when the client successfully connects to the server specified in the parameters. The client address may be fixed or obtained from a DHCP server. The client port is selected at random during creation of the client. The connection is attempted for a maximum time based on the *Network Connection Timeout,* specified in units of 10 milliseconds (section 4.4). Data is sent to, and received from, this server. If the connection is terminated, auto-connect mode also terminates and the command processing state is entered.

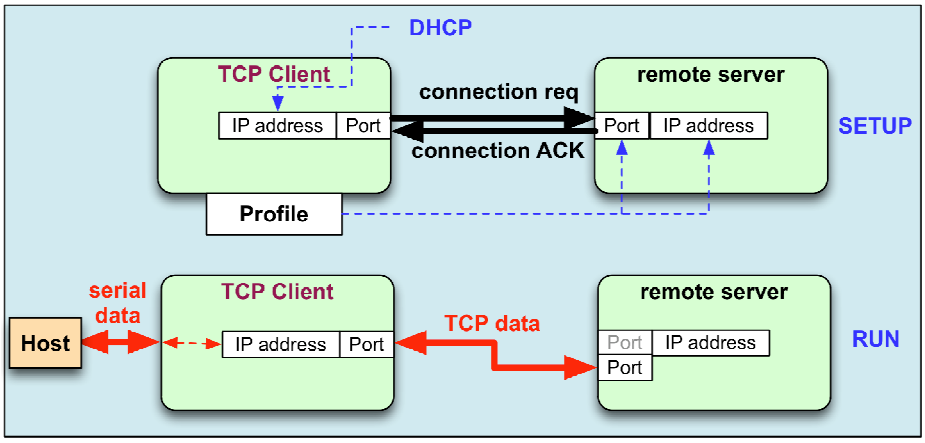

Figure 7: TCP Client Operation in Auto Connect Mode

The TCP server IP address may be fixed in the profile or obtained from DHCP. The port for connection attempts to be made is obtained from the profile. In TCP server mode, the connection is considered established when the first client connects to the server. Data is sent to, and received from, this client. If the client disconnects, the adapter waits for the next client to connect.

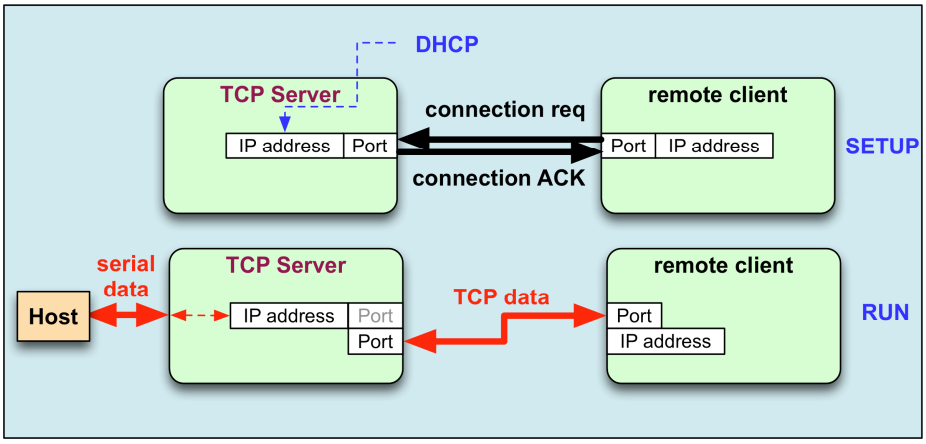

Figure 8: TCP Server Operation in Auto Connect Mode

In UDP client mode, the connection is considered established when the client is created. The client IP address may be fixed or obtained from DHCP. The client port number is set at random upon creation of the client. Data is sent to and received from the configured server.

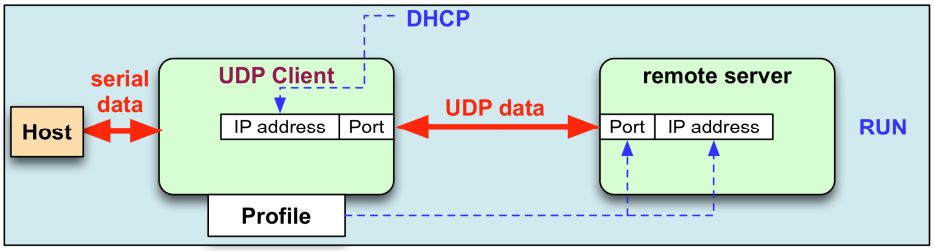

Figure 9: UDP Client Operation in Auto Connect Mode

In UDP server mode, the connection is considered established when data is received from any client. The UDP server IP address may be fixed or obtained by DHCP. The port is set by the profile. Data received from any client is output on the serial port and data received on the serial port is transmitted to the client based on the last packet was received.

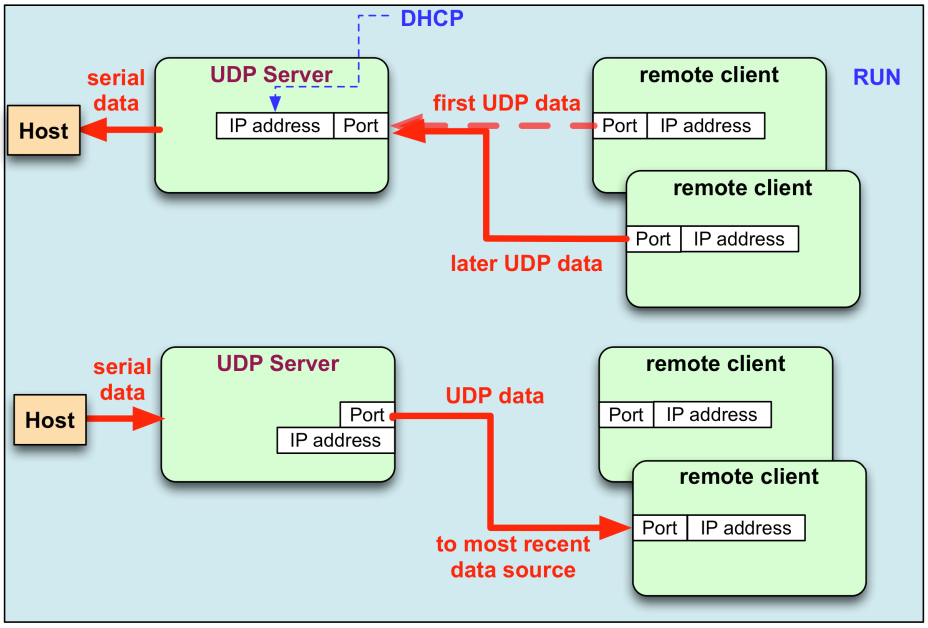

Figure 10: UDP Server Operation in Auto Connect Mode

In TCP and UDP server mode, even where no connection is established, the serial host may take control of the Serial2WiFi interface by issuing a specific escape sequence, described in section 3.3.1.

3.3.1 Auto Connection Operation

Auto Connect Mode acts as a cable replacement so that the interface acts like a serial interface and no commands or user intervention are required for connection management. The node automatically establishes the wireless and network connections by using parameter values from the current active Profile and transfers data transparently between the Host and Target in data mode. No status information is sent to the Host.

In auto connection mode the Adapter:

- Receives characters from the serial port and transmits them over the Wi-Fi connection
- Receives data from the Wi-Fi connection and transmits it on the serial port

The serial host may gain control of the interface by issuing the ***escape sequence*** "+++", followed by a one-second gap where no characters are received on the serial port. When this sequence is encountered, the Adapter suspends auto connection mode and resumes command processing. The Host then may make changes in the network configuration or other parameters as needed. However, the Adapter does not accept any new TCP/UDP client/server or auto connection requests since auto connection exists in the background. The ATO command (terminated by the ASCII character "O", not the number 0) is used to return to auto connection mode.

In auto connection mode, the Nagle Algorithm Wait Time (section 4.4) can be used to buffer any characters to be sent, in order to avoid sending a large number of packets with small payloads onto the network. The wait time is specified in units of 10 milliseconds. This functionality is available for both UDP and TCP connections.

3.4 Data Handling

In Data Processing Mode, data transfers are managed using various ***escape sequences***. Each escape sequence starts with the ASCII character 27 (0x1B); this is equivalent to the ESC key. The encoding of data and related commands are described in the following pages. This encoding is used for both transmitted and received data.

The network destination, or destination source, for a given data packet is established by means of a ***Connection Identifier***, and represented as a single hexadecimal number. Data is transferred on a per CID basis. Data is normally buffered until the end-of-data escape sequence is received. However, if the amount of data exceeds the size of the data buffer, the data received, thus far, is sent immediately. The data buffer size depends on the implementation, but is usually one MTU (1400 bytes).

The process of sending a data packet is depicted in Figure 11. The sequence Esc S or Esc U is sent to initiate the data transfer. This sequence is followed by a single-digit CID; if the CID is valid, the subsequent characters are assembled into a data stream, terminated by Esc E, Esc C, Esc S or Esc U. With a terminating sequence, the data is sent via the requested network connection and the system either returns to command processing or to further data processing.

Escape sequences like Esc S, Esc u and Esc U support only ASCII data handling while Esc Z, Esc Y and Esc y supports all types of data (ASCII, Binary etc.) handling.

Please refer to Appendix 6 for a complete description of all the Escape sequences used for data handling.

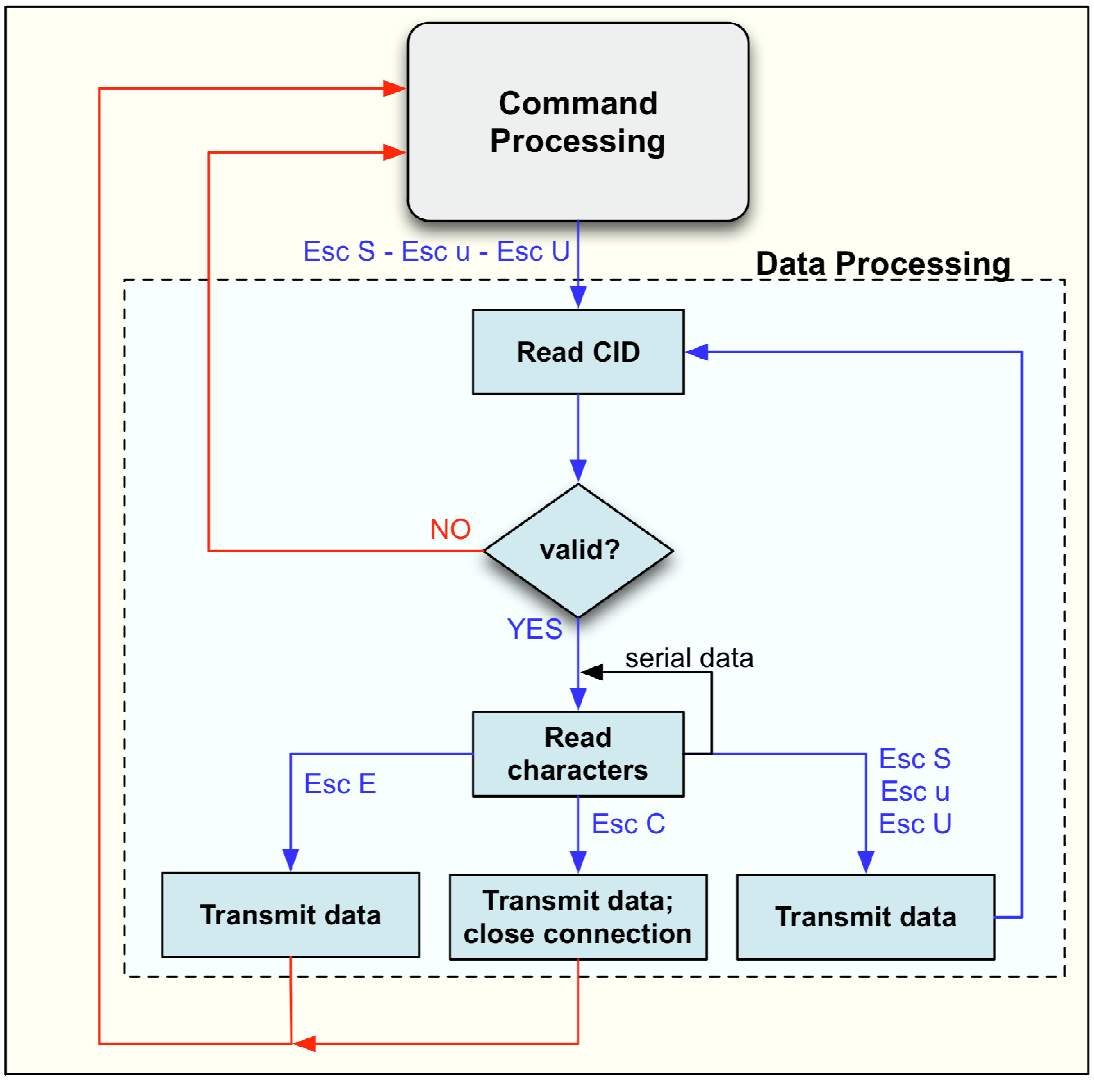

Figure 11: Data Processing Flow

Table 3: Data handling responses at completion

Operation	Escape Sequence	Description
Send and Return to Command Mode Sequence	<Esc>C	This sequence causes transmission of the data received on the serial interface on a TCP server/client or UDP client connection. After, the currently selected connection is closed and the interface returns to Command Mode. Any buffered data is sent before the connection is closed. This can be issue from the serial host once the data transmissions start on a socket using <Esc>S<CID> sequence.
Success Indication	<Esc>O	"OK": This sequence is sent to the serial host by the Serial2WiFi Adapter upon successful completion of the <Esc>S<CID>, <Esc>E, <Esc>U<CID> or <Esc>C commands.
Failure Indication	<Esc>F	"FAILURE": This sequence is sent to the host by the Serial2WiFi Adapter if an <Esc>S, <Esc>E, <Esc>U, or <Esc>C command failed.

The contents of < > are either a byte or byte stream, except for <Esc>; literals outside brackets are ASCII characters.

3.4.1 Bulk data Tx and Rx

In Bulk Data Mode, data transfers are managed using ***escape sequences*** (***Esc Z, Esc Y and Esc y***). Each escape sequence starts with the ESC key (ASCII character 27 (0x1B)). Encoding is used for both transmitted and received data. Enable bulk data by using command "AT+BDATA=" (1 is enable and 0 is disable).

The format of a bulk data frame for TCP client, TCP server, or UDP client is:

```
<Esc>Z<CID><Data Length xxxx 4 ascii char><data>
```

The contents of < > are a byte or byte stream.

- CID is connection identifier (UDP, TCP, etc.; as derived when TCP socket is created by issuing the command: AT+NCTCP, for example.)
- Data Length is 4 ascii char represents decimal value i.e. 1400 byte (0x31 0x34 0x30 0x30).
- The Data Length range should be 1 to 1400 bytes.
- User Data size **must match** the specified Data Length. Ignore all command or esc sequence in between data pay load. User should send the specified length of data to the adapter irrespective of

any asynchronous events happened on the adapter so that the adapter can start receiving next commands.

For example, if CID value is 3, then:

- To send a 5 byte user data (e.g. ABCDE) for a TCP client connection, the format will be:
  `<ESC>Z30005ABCDE`
- To send a 512 byte user data for a TCP client connection, the format will be:
  `<ESC>Z30512<512 bytes of user data>`

To send data on UDP server, the bulk data frame format is:

```
<Esc>Y<CID><Ip address>:<port>:<Data Length xxxx 4 ascii char><data>
```

When receiving data on UDP server, the format of a bulk data frame is:

```
<Esc>y<CID><IP address><space><port><horizontal tab><Data Length xxxx 4 ascii
char><data>
```

Table 4: Escape Sequences.

Operation	Escape Sequence	Description
Bulk Data transfer on TCP Server/Client and UDP Client connection	<Esc>Z<CID>Data Len 4 digit ascii<Data>	To improve data transfer speed, one can use this bulk data transfer. This escape sequence is used to send and receive data on a TCP Client/Server and UDP client connection. Example: <Esc>Z40005Hello where 4 is the CID, 0005 is the 5 byte data length and Hello is the data to be sent.
Bulk Data Send on UDP sever connection	<Esc>Y<CID> remote address: remote port:Data Len 4 digit ascii<Data>	This escape sequence is used when sending UDP data on a UDP server connection. When this command is used, the remote address and remote port is transmitted in ASCII text encoding and terminated with a ':' character. Example: <Esc>Y4192.168.1.1:52:0005Hello where 4 is the CID, 0005 is the 5 byte data length and Hello is the data to be sent.
Bulk Data Receive on UDP Server Connection	<Esc>y<CID> remoteaddress<space>remote port<horizontal tab>Data length in 4 digit ascii<Data>	This escape sequence is used when receiving UDP data on a UDP server connection. When this sequence is used, the remote address and remote port is transmitted in ASCII text encoding and separated be a space ( ) character. Example: <Esc>y4192.168.1.1<space>52<horizontal tab>0005Hello where 4 is the CID, 0005 is the 5 byte data length and Hello is the data received.

The contents of < > are either a byte or byte stream, except for <Esc>; literals outside brackets are ASCII characters.

3.4.2 Raw Data Handling (BACNET Support Only)

In Raw Data Mode, data transfers are managed using ***escape sequences***. Each escape sequence starts with the ASCII character 27 (0x1B), the equivalent to the ESC key. The encoding of data is described below. Encoding is used for both transmitted and received data. The Raw Ethernet Support Enable command (4.10.16) must be issued before sending or receiving raw data through the Adapter.

The format of a raw-data frame is:

<Esc>:R:<Length>:<DstAddr><SrcAddr><EtherType><Raw-Payload>

The contents of < > are a byte or byte stream.

- Length is the size of DstAddr, SrcAddr, EtherType and Raw-Payload

- `DstAddr` is the destination MAC address
- `SrcAddr` is the source MAC address
- `EtherType` is the type of the Ethernet packet. For example, for BACNET-over-Ethernet, `EtherType` is 0x0000.
- `Raw-Payload` is the raw data

3.4.3 Unsolicited Data Handling

In Unsolicited Data Mode (data transmission without association), data transfer is managed using ***escape sequences***. Each escape sequence starts with the ASCII character 27 (0x1B), equivalent to the ESC key. The encoding of data is described below. This encoding is used for transmitted data only. The unsolicited data transmission Enable command (4.10.16) must be issued before sending unsolicited data through the Adapter.

The format of an unsolicited data frame is:

<ESC>D/d<PayLoad>

The PayLoad contents are byte or byte stream.

3.4.4 Software Flow Control

Software flow control works only with ASCII data transfers and cannot be used for binary data.

If software flow control is enabled, and the interface receives an XOFF character from the serial host, it stops sending to the host until it receives an XON character. If the Adapter is receiving data over the wireless connection during the time that XOFF is enabled, it is possible for the wireless buffer to become full before XON is received. In such a case, data from the network will be lost.

If software flow control is enabled, then the interface sends an XOFF character to the host when it will be unable to service the serial port. The XON character is sent when the interface is once again able to accept data over the serial port.

Note: *With initialization, the Adapter treats the serial channel as clear with no restrictions on data transmission or reception; no explicit XON is transmitted by the Adapter or required from the Host, even if flow control is enabled.*

3.4.5 Hardware Flow Control

The Hardware Flow control is a handshake mechanism between the Serial host and S2W adapter on UART interface, using two additional CTS and RTS connections. This feature prevents the UART hardware FIFO overflow on S2W adapter due to high speed data transmission from/to the S2W adapter. If hardware flow control is enabled, an RTS/CTS handshake will occur between the serial host and the Adapter This is a hardware feature and available only for UART interface.

The S2W adapter uses both CTS and RTS signals as "low" to indicate the readiness to send or receive data from serial host.

3.5 Serial Data Handling

The Serial Data Handler receives and transmits data to and from the hardware serial controller. Data read from the serial port is passed to:

- The command processor in command mode
- The Tx data handler in data mode
- The auto connection mode processor for data transfer in auto connection mode

Then Data is transferred on the serial port from:

- The command processor in order to output responses to commands
- The Rx data handler in order to output incoming packets
- The auto connection handler in order to output incoming data
- The connection manager in order to output status indications
- The wireless connection manager in order to output status indications

When configured in Auto Connection Mode, the Adapter enters directly into Data Processing Mode after the completing the connection without sending any status information to the Host.

3.6 Connection Management

The connection management module is responsible for processing connection-related events. The interface provides UDP and TCP sockets (similar to the familiar BSD network sockets). Each socket may represent either a server or client connection. Each connection has a unique, single-digit hexadecimal value (0 to F), for the CID. The allowed maximum number of connections (up to 16) may be specified at compile time. Note that this single pool of CID's is used for TCP, UDP, Server and Client connections.

3.6.1 Packet Reception

When a packet is received on any open connection, and the application is not currently in auto-connect mode, the packet is transferred on the UART/SPI in the form described in Section 3.4 above. Received data payloads are encoded with the appropriate Escape sequence. The connection ID is used to inform the serial host of the origin of an IP data packet. The source IP address and port are provided along with the data when a UDP packet is received.

If auto-connect mode is enabled and a packet is received on the auto-connected CID, the packet data is sent without modification over the UART/SPI to the serial host.

3.6.2 Remote Close

If a TCP connection is terminated by disconnection from the remote end, an unsolicited ASCII-format response of the form `DISCONNECT Connection ID` is sent to the serial host, and the specified CID should be considered unavailable. If the connection ends because the remote server has shut down, the unsolicited response `ERROR: SOCKET FAILURE Connection ID` will be sent to the host. Note that a data packet from the remote client or server containing the same ASCII characters `CLOSE Connection ID` is treated as data rather than a command and forwarded to the serial host.

3.6.3 TCP Server Connections

Upon deployment of incoming TCP connections on a socket, the incoming connection is allowed if the limit on the maximum number of connections has not been reached. There is an unsolicited response of the form `CONNECT <server CID> <new CID> <ip> <port>`, where:

- Server CID is the CID of the server where the connection has arrived
- New CID is the CID allocated for this client connections
- IP and port of the client encoded in the binary encoding used for UDP server data packets described in section 3.4 above is sent to the serial host. The host can use the IP address to ascertain the source of the TCP connection request. The TCP server has no timeout limitation for an incoming connect request. It waits indefinitely, until a CLOSE command is received.

Note that if Verbose mode is disabled (section 4.1.3), the word CONNECT in the unsolicited response is replaced by the number 7.

3.7 Wireless Network Management

3.7.1 Scanning

The Serial2WiFi interface can instruct the Wi-Fi radio to scan for access points and ad hoc networks with a specified SSID, BSSID and/or channel for a specified scan time. Scanning can be performed to find networks with a specific SSID or BSSID, networks operating on a specific radio channel or a combination of these constraints.

3.7.2 Association

The Serial2WiFi interface performs all the actions required to join an infrastructure IP network:

- Scan for a specific AP (AT+WS, section 4.7.5)
- Authenticate the specified network using the configured authentication mode (AT+WAUTH, section 4.8)
- Associate to the AP (AT+WA, section 4.7.7)
- Perform security negotiation if required
- Change state to Wireless Connected
- Initialize the networking stack using the configured static IP address or via DHCP (AT+NDHCP, section 4.9.2)

In ad hoc mode, the interface can:

- Scan for a specified Ad-hoc Network
- Join the ad hoc network, if it exists
- If the ad hoc network does not exist, create a new ad hoc network to join
- Perform security negotiation, if required
- Change state to Wireless Connected
- Initialize the networking stack using the configured static IP address or via DHCP

3.7.3 Response Codes

The possible responses sent by the Adapter to the serial host are enumerated in Table . The table below reflects all characters including <CR> or <LF> that would be seen on the interface.

Table 5: Response Codes.

No	ASCII CHAR	Response	ASCII STRING	Meaning
1	0	S2W_SUCCESS	"\r\nOK\r\n"	Command Request Success.
2	1	S2W_FAILURE	"\r\nERROR\r\n"	Command Request Failed.
3	2	S2W_EINVAL	"\r\nERROR: INVALID INPUT\r\n"	Invalid Command or Option or Parameter.
4	3	S2W_SOCK_FAIL	"\r\nERROR: SOCKET FAILURE <CID>\r\n"	Socket Operation Failed.
5	4	S2W_ENOCID	"\r\nERROR: NO CID\r\n"	All allowed CID's in use, so there was no CID to assign to the new connection.
6	5	S2W_EBADCID	"\r\nERROR: INVALID CID\r\n"	Invalid Connection Identifier.
7	6	S2W_ENOTSUP	"\r\nERROR: NOT SUPPORTED\r\n"	Operation or Feature not supported.
8	7	S2W_CON_SUCCESS	"\r\nCONNECT <CID>\r\n\r\nOK\r\n"	TCP/IP connection successful. <CID> = the new CID in hexadecimal format. Followed by command request success
9	8	S2W_ECIDCLOSE	"\r\nDISCONNECT <CID>\r\n"	TCP/IP connection with the given CID is closed. This response is sent to the host when a connection is closed either by the remote device or by the serial host.
10	9	S2W_LINK_LOST	"\r\nDISASSOCIATED\r\n"	Not associated to a wireless network.

No	ASCII CHAR	Response	ASCII STRING	Meaning
11	10	S2W_DISASSO_EVT	“\r\n\r\nDisassociation Event\r\n\r\n”	Wireless network association lost.
12	11	S2W_STBY_TMR_EVT	"\r\nOut of StandBy-Timer\r\n"	Wake up from Standby due to RTC timer expiration.
13	12	S2W_STBY_ALM_EVT	"\r\n\n\rOut of StandBy-Alarm\r\n\r\n"	Wake up from Standby due to receipt of an Alarm signal.
14	13	S2W_DPSLEEP_EVT	"\r\n\r\nOut of Deep Sleep\r\n\r\n\r\nOK\r\n”	Wake from Deep Sleep followed by command request success
15	14	S2W_BOOT_UNEXPECTED_EVT	"\r\n\r\nUnExpected Warm Boot(Possibly Low Battery)\r\n\r\n"	Unexpected reset. Possible reasons: external reset or low battery
16	15	S2W_ENOIP	"\r\nERROR: IP CONFIG FAIL\r\n"	IP configuration has failed. This message also can come asynchronously when there is a DHCP renew fails.

3.7.4 Enhanced Asynchronous Messages

NO	Message	Subtype	Meaning
1	ERROR: SOCKET FAILURE <CID>	0	Socket Operation Failed
2	CONNECT <CID>	1	TCP/IP connection successful. <CID> = the new CID in hexadecimal format.
3	DISCONNECT <CID>	2	TCP/IP connection with the given CID is closed. This response is sent to the host when a connection is closed by the remote device.

4	Disassociation Event	3	Wireless network association lost.
5	Out of StandBy-Timer	4	Wake up from Standby due to RTC timer expiration.
6	Out of StandBy-Alarm	5	Wake up from Standby due to receipt of an Alarm signal.
7	Out of Deep Sleep	6	Wake from Deep Sleep.
8	UnExpected Warm Boot(Possibly Low Battery)	7	Unexpected reset. Possible reasons: external reset or low battery.
9	ERROR: IP CONFIG FAIL	8	IP configuration has failed. This message comes asynchronously when there is a DHCP renew fails.

3.7.5 Exception Messages

The possible exception messages sent by the Adapter to the serial host are enumerated in Table 5.

Table 6: Exception Messages.

No	*ASCII STRING*	*Meaning*
1	\n\rAPP Reset-Wlan SW Reset\r\n	Adapter reset due to WLAN processor software reset.
2	"\n\rAPP Reset-APP SW Reset\r\n"	Adapter reset due to app processor software reset...
3	\n\rAPP Reset-Wlan-Wd\r\n	Adapter reset due to WLAN processor watchdog.
4	\n\rAPP Reset-App-Wd\r\n	Adapter reset due to app processor watchdog

5	\n\rAPP Reset-Wlan Except\r\n	Adapter reset due to WLAN processor software abort or assert.
6	\n\rAPP Reset-FW-UP-FAILURE\r\n	Adapter reset due to firmware upgrade failure.
7	\n\rAPP Reset-FW-UP-SUCCESS\r\n	Adapter reset due to firmware upgrade success.
8	\n\rAPP Reset-FW-UP-RECOVERY\r\n	Adapter reset due to firmware upgrade failure with one of the flash image updated successfully.

If the exception is due to one of the WLAN wd/SW Reset/Except, then the adapter send memory dump information of its WLAN registers to the serial host starts with the message \r\n---MEM-DUMP-START:\r\n and end with the message \n\r---MEM-DUMP-END:\r\n.

3.7.6 Boot Messages

The possible boot messages sent by the Adapter to the serial host are enumerated in Table 6.

Table 7: Boot Messages.

NO	ASCII STRING	Meaning
1	\r\n Serial2WiFi APP\r\n	Normal Serila2WiFi adapter boot message with internal PA.
2	\r\nSerial2WiFi APP-Ext.PA\r\n	Normal Serila2WiFi adapter boot message with external PA.
3	\r\n Factory Default CheckSum Error\r\n	The factory default section contains invalid data. This comes along with either one of the above boot message.

3.7.7 SSID and PassPhrase

Rules:

1- The S2W adapter accepts the following ASCII characters for SSID and passphrase.

Category	Accepted Characters
Numerical	0-9
Alphabets	a-z and A-Z
Special characters	SP ! # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _ ` { | } ~ ”

Note: SP = space

2- The SSID or PassPhrase parameter may be captured within or without double quotation marks (“SSID”).

3- The quotation mark (“) may not be used as the first character of the SSID or passphrase.

4- If comma (,) is a part of the SSID, then SSID parameter needs to be framed with double quotation marks (“SS,ID”).

Expected SSID	Input SSID	Remarks
TEST	TEST	Valid (satisfies rule 2)
TEST	“TEST”	Valid (satisfies rule 2)
TE”ST	TE”ST	Valid (satisfies rule 3)
TE”ST	“TE”ST”	**Invalid** (*breaks rule 3*)
TE,ST	“TE,ST”	Valid (satisfies rule 4)
TE,ST	TES,T	**Invalid** (*breaks rule 4*)
TE,S”T	“TE,S”T”	**Invalid** (*breaks rule 3 and 4*)

4 Commands for Command Processing Mode

This section provides a list of Serial2WiFi commands and their effects. Formatting and processing of commands was described in section 3.2 above. Parameters are generally ASCII characters, e.g. ATEn with n=1 is the series of ASCII characters 'A', 'T', 'E', and '1'. Where some parameters are optional, mandatory parameters are denoted by < > and optional parameters by [ ]. If a parameter is mandatory, any associated sub-parameters are also mandatory; sub-parameters of an optional parameter are optional. Parameters must always be provided in the order given in the command description. When an optional parameter is not supplied, the comma delimiters must still be included in the command. Every command starts with the characters "AT"; any other initial characters will cause an error to be returned.

Command Response: In most cases, valid commands return the characters `OK` if verbose mode is enabled and `0` verbose mode is not enabled. Invalid inputs return `ERROR: INVALID INPUT` if verbose is enabled and `2` if it is not. Exceptions to this rule are noted explicitly below.

4.1 Command Interface

4.1.1 Interface Verification

The command AT can be issued to verify that the interface is operating correctly; it should return a successful response `OK` (or `0` if verbose mode is disabled).

4.1.2 Echo

The command to enable/disable echo is

`ATEn`

If *n* is 0, echo is disabled and if *n* is 1, echo is enabled.

If echo is enabled, every character received on the serial port is transmitted back on the serial port. This command returns the standard command response (section 4) to the serial interface. By default echo is enabled in s2w adapter.

4.1.3 Verbose

The command to enable/disable verbose responses is

`ATVn`

If *n* is 0, verbose responses is disabled and if *n* is 1, verbose responses is enabled.

If verbose mode is disabled, the status response is in the form of numerical response codes. If verbose mode is enabled, the status response is in the form of ASCII strings. Verbose Mode is enabled by default.

This command returns the standard command response (section 4) to the serial interface.

4.1.4 Help

The command to display help is

`AT?`

This command is no longer supported.

4.2 UART Interface Configuration

4.2.1 UART Parameters

The command to set the UART communication parameters is

`ATB=<baudrate>[[,<bitsperchar>][,<parity>][,<stopbits>]]`

All standard baud rates are supported.

Allowed baud rates include: 9600, 19200, 38400, 57600, 115200, 230400,460800 and 921600.

Parity is *n* for no parity, *e* for even parity and *o* for odd parity.

Allowed values are 5, 6, 7 or 8 bits/character, with 1 or 2 stop bits

The new UART parameters take effect immediately. However, they are stored in RAM and will be lost when power is lost unless they are saved to a profile using `AT&W` (section 4.6.1). The profile used in that command must also be set as the power-on profile using `AT&Y` (section 4.6.3).

This command returns the standard command response (section 4) to the serial interface with the new UART configuration.

4.2.2 Software Flow Control

The command to configure software flow control is

`AT&Kn`

If *n* is 0, software flow control is disabled. If *n* is 1, software flow control is enabled.

The use of software flow control is described in section 3.4.4 above.

This command returns the standard command response (section 4) to the serial interface.

4.2.3 Hardware Flow Control

The command to configure hardware flow control is

`AT&Rn`

If *n* is 0, hardware flow control is disabled. If *n* is 1, hardware flow control is enabled. This command returns the standard command response (section 4) to the serial interface.

The use of software flow control is described in section 3.4.5 above.

4.3 SPI Interface Configuration

4.3.1 SPI Parameters

The command to set the SPI clock phase and clock polarity parameter is as follows:

```
AT+SPICONF=<clockpolarity>, <clockphase>
```

If clock polarity is 0, then inactive state of serial clock is low.

If clock polarity is 1, then inactive state of serial clock is high.

If clock phase is 0, then data is captured on the first toggling edge of the serial clock (clock phase zero), after the falling edge of slave select signal.

If clock phase is 1, then data is captured on the second edge of the serial clock (clock phase 180), after the falling edge of slave select signal.

Default is clock polarity 0 and clock phase 0.

The new SPI parameters take effect after node reset/restart. However, they are stored in RAM and will be lost when power is lost unless they are saved to a profile using `AT&W` (section 4.6.1). The profile used in that command must also be set as the power-on profile using `AT&Y` (section 4.6.3).

This command returns the standard command response (section 4) to the serial interface with the new SPI configuration.

4.4 Serial to Wi-Fi Configuration

The command to configure network and connection parameters is

```
ATSn=p
```

n is the parameter id to set and *p* is the value to set the parameter to. The parameters available are described in Table .

Table 8: Configuration Parameters for Network Association.

Parameter ID	Name	Description	Citation
0	Network Connection Timeout	The maximum amount of time allowed establishing the network connection in Auto Connect Mode. Measured in units of 10 milliseconds. Allowed values: 1 to 65535 (but the TCP/IP stack limits the maximum timeout value). Default value: 1000 (10 seconds). If the connection attempt is a TCP client connection, and TCP Connection Timeout below is less than Network Connection Timeout, the value of Network Connection Timeout will be ignored.	4.4
1	Auto Associate Timeout	The maximum amount of time allowed associating to the desired wireless network in Auto Connect Mode, in units of 10 milliseconds. Allowed values: 0 to 65535. Default value: 500 (5 seconds).	4.4
2	TCP Connection Timeout	The maximum amount of time allowed establishing a TCP client connection, in units of 10 milliseconds. Allowed values: 0 to 65535 (but the TCP/IP stack limits the maximum timeout value). Default value: 500 (5 seconds). Note that 0 corresponds to the default TCP/IP stack timeout (75 seconds).	4.10.1
3	Association Retry Count	Not currently supported.	3.1.3
4	Nagle Algorithm Wait Time	The maximum time for serial data sent in Auto Connect Mode to be buffered, in units of 10 milliseconds. Allowed values: 0 to 65535 (but the amount of data is limited by available buffer size). Default value: 10 (100 ms).	3.3.1
5	Scan Time	The maximum time for scanning in one radio channel, in units of milliseconds. Allowed values: 0 to 65535 (but at the high limit a 14-channel scan will consume 2.6 hours!). Default value: 20 (20 ms).	3.7.1

This command returns the standard command response (section 4) to the serial interface.

4.5 Identification information

The command to obtain identification information from the application is

`ATIn`

n is the ID of the information to obtain. The responses are listed in Table . These responses are provided as ASCII strings in addition to the standard command response (section 4).

Table 9: Application Information.

Information ID	*Description*
0	OEM identification
1	Hardware version
2	Software version

4.6 Serial to Wi-Fi Configuration Profiles

Adapter configuration parameters can be stored and recalled as a Profile; see 3.1.3 for a detailed description of the profile parameters.

4.6.1 Save Profile

The command to save the current profile is

`AT&Wn`

n shall either be 0 for profile 0 or 1 for profile 1. (Higher values are allowed if more profiles are configured at compile time.)

Upon deployment of this command, the current configuration settings are stored in non-volatile memory under the specified profile. Note that, in order to ensure that these parameters are restored after power cycling the adapter, the command `AT&Y` (section 4.6.3) must also be issued, using the same profile number selected here.

This command returns the standard command response (section 4) or ERROR (1, if verbose disabled) ,if the operation failed.

4.6.2 Load Profile

The command to load a profile is

`ATZn`

n shall either be 0 for profile 0 or 1 for profile 1. (Higher values are allowed if more profiles are configured at compile time.)

Upon deployment of this command, the currently configured settings are set to those stored in non-volatile memory under the specified profile. This command returns the standard command response (section 4) to the serial interface. The s2w adapter uses profile 0 as the default profile.

4.6.3 Selection of Default Profile

The command to select the default profile is

`AT&Yn`

n shall either be 0 for profile 0 or 1 for profile 1. (Higher values are allowed if more profiles are configured at compile time.)

The settings from the profile that is chosen as the default profile are loaded from non-volatile memory when the device is started.

In addition to the standard status responses, this command returns `ERROR` or `1`, based on verbose settings, if a valid input cannot be executed.

4.6.4 Restore to Factory Defaults

The command to reset to factory defaults is

`AT&F`

Upon deployment of this command, the current configuration variables are reset to the factory defaults. These defaults are defined by macro values in the configuration header, and can be modified at compile time. Issuing this command resets essentially all configuration variables ***except*** the IEEE MAC address. Only the command `AT+NMAC` (section 4.7.1) changes the MAC address.

This command returns the standard command response (section 4) to the serial interface.

4.6.5 Output current configuration

The command to output the configuration is

`AT&V`

Upon deployment of this command, the current configuration and the configuration of the saved profiles are output on the serial port in ASCII format in addition to the standard command response (section 4). The details of the profile parameters are described in section 3.1.3.

4.7 Wi-Fi Interface Configuration

4.7.1 MAC Address Configuration

The command to set the configuration is

`AT+NMAC=<MAC ADDRESS>`

Upon deployment of this command, the Adapter sets the IEEE MAC address as specified. The format of the MAC address is an 8-byte colon-delimited hexadecimal number. An example is shown below:

`AT+NMAC=00:1d:c9:00:01:a2`

The MAC address is used in the 802.11 protocol to identify the various nodes communicating with an Access Point and to route messages within the local area (layer 2) network. Fixed MAC addresses issued to network interfaces are hierarchically structured and are intended to be globally unique. Before issuing a MAC address to a given Adapter, ensure that no other local device is using that address.

The MAC address supplied in the `AT+NMAC` command is saved to flash memory, and will be used on each subsequent cold boot (from power off) or warm boot (from Standby).

The alternative command

`AT+NMAC2=<MAC ADDRESS>`

stores the MAC address in RTC RAM. Each warm boot (from Standby) will use the MAC address stored in RTC RAM (from the most recent `AT+NMAC2=` command), but if power to the device is lost, the next

cold boot will use the MAC address stored in flash memory (from the most recent `AT+NMAC=` command). This command is particularly useful in cases where writing to flash memory is undesirable.

In addition to the standard command responses (section 4) , this command returns `ERROR` or `1`, based on verbose settings, if a valid input cannot be executed.

4.7.2 Output MAC Address

The command to output the configuration is

`AT+NMAC=?`

Upon deployment of the command, the Adapter outputs the current MAC address of the wireless interface to the serial port, in addition to the usual command responses (section 4) . The alternate command is

`AT+NMAC2=?`

may also be used, and returns the same value.

4.7.3 Regulatory Domain Configuration

The command to set the regulatory domain is

AT+WREGDOMAIN=<Regulatory Domain>

This command sets the regulatory domain as per the Regulatory Domain parameter passed. The supported regulatory domains are:

- FCC → supported Channel range is 1 to 11.
- ETSI → supported Channel range is 1 to 13.
- TELEC → supported Channel range is 1 to 14.

The corresponding values for this regulatory domain that needs to be passed as the parameter are:

- FCC : 0
- ETSI : 1
- TELEC : 2

The default regulatory domain is FCC. The Regulatory domain set is required only once since it is being updated in the flash. This command returns the standard command response (section 4) to the serial interface.

4.7.4 Regulatory Domain Information

The command to get the configured regulatory domain in the Serial2WiFi adapter is

AT+WREGDOMAIN=?

Upon reception of the command, the Adapter outputs the current Regulatory domain of the wireless interface to the serial port as the following format:

REG_DOMAIN=FCC or ETSI or TELEC, in addition to the standard command responses.

4.7.5 Scanning

The command to scan for access points or ad hoc networks is

AT+WS[=<SSID>[,<BSSID>][,<Channel>][,<Scan Time>]]

Upon deployment of the command, the Adapter scans for networks with the specified parameters, and displays the results. Scanning can be performed to find networks with specific SSID or in a particular operating channel, or a combination of these parameters. Scanning for a specific SSID employs active scanning, in which probe requests are transmitted with the SSID fields being filled appropriately.

The SSID is a string containing between 1 and 32 ASCII characters, Refer section 3.7.6 for details.

This command does not support scan based on the BSSID.

The Scan Time is in units of Milliseconds with a range of 0-65535.

Upon completion, the adapter reports the list of networks and information for each network along with the standard command response (section 4) one per line, in the following format to the serial interface

<space><BSSID>,<space><SSID>,<space><Channel>,<space><space><Type><space>,<space><RSSI><space>,<space><Security>

Also this sends out the total number of networks found as follows (after send out the above information to the serial interface).

"No. Of AP Found:<n><CR><LF>"

Where n is the total number of networks found during scan.

`Type` is `INFRA` for an infrastructure network and `ADHOC` for an ad hoc network.

4.7.6 Mode

The command to set the wireless mode:

`AT+WM=n`

If *n* is 0, the mode is set to *infrastructure*; if *n* is 1, the mode is set to *ad hoc.*

If n is 2, the mode is set to limited AP so that the adapter can act as a limited wireless Access Point. S2w Adapter uses infrastructure(0) as the default mode.

This command returns the standard command response (section 4) to the serial interface.

4.7.7 Associate with a Network, or Start an Ad Hoc or Infrastructure (AP) Network

The command to associate to an access point, to join an ad hoc network or to create an ad hoc/ infrastructure (AP)/ network is

```
AT+WA=<SSID>[,[<BSSID>][,<Ch>]]
```

In infrastructure mode (section 4.7.6, n is 0), the adapter will attempt to associate with the requested network. In ad hoc mode (section 4.7.6, n is 1), if a network with the desired SSID or channel or both is not found, then a new network is created. However, if the BSSID was specified in the request and the applicable BSSID is not found, the Adapter will report an error and will not create an ad hoc network.

In AP mode (section 4.7.6, n is 2), the adapter creates an infrastructure network (limited AP) with the SSID passed

The SSID is a string containing between 1 and 32 ASCII characters. Refer section 3.7.6 for details.

Upon completion, the adapter reports its IP address to the serial interface in the following format:

<LF><4 spaces>IP<14 spaces>SubNet<9 spaces>Gateway<3 spaces>

<space><IP address>:<space><SubNet address>:<space><Gateway address>

In addition to the usual status responses, this command will return `ERROR` or `1` (depending on verbose status) if a valid command was issued but association failed.

In adhoc and AP modes, the radio should be on in active mode (section 4.9.1)

4.7.8 Disassociation

The command to disassociate is

```
AT+WD
```

An equivalent command is

```
ATH
```

Upon deployment of this command, the interface disassociates from the current infrastructure or ad hoc network, if associated. This command returns the standard command response (section 4) to the serial interface.

4.7.9 WPS

The command to associate to an AP using WPS is

AT+WWPS=<METHOD>[,PIN]

- METHOD is push button (1) or pin (2).
- PIN is the pin for PIN method.

Upon execution of this command, the adapter uses either push button or pin method as per the METHOD parameter to associate to the WPS enabled AP. The PIN is optional and is valid for pin method only.

In addition to the usual status responses this command returns the following information to the serial host on success case:

- SSID=<ssid>
- CHANNEL=<channel>
- PASSPHRASE=<passphrase> for wpa/wpa2 security;
- WEP KEY=<wep key> for WEP security;
- WEPKEYINDEX=<key index> for WEP security

The above information is send to the serial interface with one information element per line.

This command returns `ERROR` or `1` (depending on verbose status) if a valid command was issued but WPS failed.

On success case the serial host should issue the AT+NDHCP=0/1 to establish the L3 connection.

4.7.10 Status

The command to retrieve information about the current network is

`AT+NSTAT=?`

Upon deployment of this command, the adapter reports the current network configuration to the serial host:

- MAC address;
- WLAN state;
- SSID;
- Mode;
- Security;
- Channel;
- BSSID;
- Network configuration: IP Address, Subnet mask, Gateway address, DNS1 address, DNS2 address;

- TX count;
- RX count.
- RSSI value

in addition to the usual status response.

The alternate command

`AT+WSTATUS`

may also be used, Upon deployment of this command, the adapter reports the current network configuration to the serial host:

- Mode;
- Channel;
- SSID;
- BSSID;
- Security;

if the adapter associated to an Access Point. If no association is present, the error message `NOT ASSOCIATED` is returned, in addition to the standard command response (section 4).

4.7.11 Get RSSI

The command obtains the current RSSI is

`AT+WRSSI=?`

Upon deployment of this command, the current RSSI value (in dBm) is output on the serial port in ASCII format, in addition to the standard command response.

4.7.12 Get Transmit Rate

The command obtains the current transmit rate is

`AT+WRATE=?`

Upon deployment of this command, the current transmit rate used is output on the serial port in ASCII format along with the standard command response

4.7.13 Set Retry count

The command to set the wireless retry count is

`AT+WRETRY=<retrycount>`

Upon deployment of this command, the current wireless retry count is set to the supplied value. The transmission retry count determines the maximum number of times a data packet is retransmitted, if an 802.11 ACK is not received. (Note that the count includes the initial transmission attempt.) The valid range is 4 to 7 with default value 5.

This command returns the standard command response (section 4) to the serial interface.

4.8 Wi-Fi Security Configuration

4.8.1 Authentication Mode

The command to choose the authentication mode to use is

`AT+WAUTH=n`

n is:

- 0- None
- 1 – Open
- 2 – Shared with WEP

Note that this command configures the authentication mode, but any required encryption keys must be set using the key commands described below. This authentication mode command is specific to WEP encryption; if WPA/WPA2 operation is employed, the authentication mode may be left at the default value " None". This command returns the standard command response (section 4) to the serial interface.

4.8.2 Security Configuration

The S2w adapter supports a strict security configuration. The command required to configure this feature is

AT+WSEC= n

Where *n* is:

- 0 – Auto security (All)
- 1 – Open security
- 2 – Wep security
- 4 – Wpa-psk security
- 8 – Wpa2-psk security
- 16 – Wpa Enterprise
- 32 – Wpa2 Enterprise

The s2w adapter supports either one of the above value with default security configuration as auto. This strict security compliance is not applicable for WPS feature. This command returns the standard command response (section 4) to the serial interface.

4.8.3 WEP Keys

The command to set WEP keys is

```
AT+WWEPn=<key>
```

n is the key index, between 1 and 4, and key are either 10 or 26 hexadecimal digits corresponding to a 40-bit or 104-bit key. Some examples:

```
AT+WWEP1=123456abdc
```

```
AT+WWEP3=abcdef12345678901234567890
```

Upon receiving a valid command, the relevant WEP key is set to the value provided. This command returns the standard command response (section 4) to the serial interface.

4.8.4 WPA-PSK and WPA2-PSK Passphrase

The command to set the WPA-PSK and WPA2-PSK passphrase is

```
AT+WWPA=<passphrase>
```

The passphrase is a string containing between 8 and 63 ASCII characters, used as a seed to create the WPA ***pre-shared key*** (PSK).

If the comma (,) is a part of the passphrase, then the passphrase parameter is to be framed in double quotation marks ("passphrase"). Refer section 3.7.6 for details.

Upon receiving the command, the PSK passphrase is reset to the value provided. This command returns the standard command response (section 4) to the serial interface.

4.8.5 WPA-PSK and WPA2-PSK KEY CALCULATION

Computation of the PSK from the passphrase is complex and consumes substantial amounts of time and energy. To avoid recalculating this quantity every time the adapter associates, the adapter provides the capability to compute the PSK once and store the resulting value. The key value is stored in the SRAM copy of the current profile; the profile needs to be saved in flash memory for this value to persist during a transition to Standby. The command to compute and store the value of the WPA/WPA2 PSK, derived from the passphrase and SSID value, is

```
AT+WPAPSK=<SSID>,<PASSPHRASE>
```

The passphrase is a string containing between 8 and 63 ASCII characters, used as a seed to create the PSK. The SSID is a string of between 1 and 32 ASCII characters. Refer section 3.7.7 for details

Each Parameter of the above command separated by comma (,). If the comma(,) is a part of the SSID or PASSPHRASE, then SSID and PASSPHRASE parameters is to be framed in double quotation marks ("SSID","PASSPHRASE").

When the command is issued, the adapter immediately responds with "<LF>`Computing PSK from SSID and PassPhrase`". Computation of the passphrase can be time-consuming! When it is complete, the adapter will issue the usual `OK` or `0`. Invalid inputs will result in `ERROR: INVALID INPUT` or `2`, as usual.

Upon receiving the command, the adapter computes the PSK from the SSID and passphrase provided, and stores those values in the current profile. The current profile parameters PSK Valid, PSK-SSID, and WPA Passphrase are updated, and can be queried with `AT&V` (4.6.5). The next time the adapter associates to the given SSID, the PSK value is used without being recalculated.

After the PSK has been computed, the commands `AT&W` (to save the relevant profile) and `AT&Y` (to ensure that the profile containing the new PSK is the default profile) should be issued. The PSK will then be available when the adapter awakens from Standby. Refer to sections 0 and 4.6.3 for more information on profile management.

4.8.6 WPA-PSK and WPA2-PSK KEY

The command to configure the WPA / WPA2 PSK key directly is

`AT+WPSK=<PSK>`

This command directly sets the pre-shared key as provided. The argument is a 32-byte key, formatted as an ASCII hexadecimal number; any other length or format is considered invalid. Example:

`AT+WPSK= 000102030405060708090001020304050607080900010203040506070809000 1`

This command returns the standard command response (section 4) to the serial interface.

After the PSK has been entered, the commands `AT&W` (to save the relevant profile) and `AT&Y` (to ensure that the profile containing the new PSK is the default profile) should be issued. The PSK will then be available when the adapter awakens from Standby. Refer to sections 0 and 4.6.3 for more information on profile management.

4.8.7 EAP-Configuration

The command to configure the EAP-security is

AT+ WEAPCONF=<Outer Authentication>,<Inner Authentication>,<user name>,<password>

Upon execution of this command, the adapter set the Outer authentication, Inner authentication, user name and password for EAP Security. This command returns the standard command responses (section 4).

The valid outer authentication values are:

Eap-FAST: 43

Eap-TLS: 13

Eap-TTLS: 21

Eap-PEAP: 25

The valid Inner Authentication values are:

Eap-MSCHAP: 26

Eap-GTC: 6

4.8.8 EAP

The command to configure certificate for EAP-TLS is

AT+ WEAP=< Type >,< Format >,< Size >,< Location >

<ESC>W <data of size above>

- Type: CA certificate(0)/ Client certificate(1)/ Private Key(2)
- Format: Binary(0)/Hex(1)
- Size: size of the file to be transferred.
- Location: Flash(0)/Ram(1)

This command enables the adapter to receive the certificate for EAP-TLS. This command stores the certificate in flash or RAM, depending on the parameter. Upon deployment of this command, the interface returns the standard command response (section 4) or ERROR, 1 (verbose disabled), if the operation failed.

4.8.9 Certificate Addition

The command to configure the certificate for SSL/HTTPS connection is

AT+ TCERTADD=<Name>,<Format>,<Size>,<Location>

<ESC>W <data of size above>

- Name: Name of the certificate
- Format: Binary(0)/Hex(1)
- Size: Size of the file to be transferred.
- Location : Flash (0)/Ram(1)

This command enables the adapter to receive the certificate for SSL/HTTPS connection. It stores the certificate in flash or ram depends on the parameter. Upon deployment of this command, the interface returns the standard command response (section 4) or ERROR, 1 (verbose disabled), if the operation failed.

4.8.10 Certificate Deletion

The command to delete a certificate from memory is

AT+TCERTDEL=<certificate name>

This command deletes the SSL/HTTPS/EAP-TLS certificate stored in flash/ram by name.

In the case of EAP-TLS certificate names are:

- TLS_CA
- TLS_CLIENT
- TLS_KEY

Upon deployment of this command, the interface returns the standard command response (section 4) or ERROR, 1 (verbose disabled), if the operation failed.

4.8.11 Enable/Disable 802.11 Radio

The command to enable or disable the radio is

`AT+WRXACTIVE=n`

If *n* is 0, the radio is disabled and if *n* is 1, the radio is enabled with default setting as disabled.

This command returns the standard command response (section 4) to the serial interface. If `WRXACTIVE` = 1, the 802.11 radio receiver is always on. This minimizes latency and ensures that packets are received at the cost of increased power consumption. The GainSpan SOC cannot enter Deep Sleep (section 4.12.1) even if it is enabled (`PSDPSLEEP=1`). Power Save mode (section 4.8.12) can be enabled but will not save power, since the receiver is left on. If `WRXACTIVE` = 0, the receiver is switched off after association is complete. If Power Save mode is not enabled (`WRXPS` not issued or `WRXPS`=0), the receiver will not be turned on again unless `WRXACTIVE` = 1 is received. Packets will not be received, and disassociation could occur. If Power Save mode is enabled (`WRXPS`=1) prior to issuing `WRXACTIVE` = 0, the receiver will be turned off, but will turn on again when it is time to listen for the next beacon from the Access Point. If Deep Sleep is also enabled, the receiver will turn off, and the SOC will enter Deep Sleep when all pending tasks are completed, but again the system will be awakened to listen to the next beacon. If a transition to Standby is requested and occurs (section 4.12.2), the SOC will remain in Standby for the requested period, and will ***not*** awaken to receive a beacon during that time.

4.8.12 Enable/Disable 802.11 Power Save Mode

The command to configure 802.11 Power Save Mode is

`AT+WRXPS=n`

If *n* is 0, Power Save is disabled and if *n* is 1, Power Save is enabled with default setting as enabled.

This command returns the standard command response (section 4) to the serial interface. In 802.11 Power Save Mode, the node (in this case, the Serial2WiFi Adapter) will inform the Access Point that it will become inactive, and the Access Point will buffer any packets addressed to that node. In this case, the GainSpan SOC radio receiver is turned off between beacons. The node will awaken to listen to

periodic beacons from the Access Point that contains a Traffic Indication Map (TIM) that will inform the Station if packets are waiting for it. Buffered packets can be retrieved at that time, using ***PSPoll*** commands sent by the node. In this fashion, power consumed by the radio is reduced (although the benefit obtained depends on traffic load and beacon timing), at the cost of some latency.

The latency encountered depends in part on the timing of beacons, set by the Access Point configuration. Many Access Points default to 100msec between beacons; in most cases this parameter can be adjusted.

4.8.13 Enable/Disable Multicast Reception

The command to configure multicast reception is

`AT+MCSTSET=n`

If n = 0, multicast reception is disabled; if n = 1, multicast reception is enabled. By default the multicast reception is enabled. This command returns the standard command response (section 4) to the serial interface.

4.8.14 Transmit power

The command to set the transmit power is

`AT+WP=<power>`

On reception of this command, the transmit power is set to the supplied value. The desired power level shall be specified in ASCII decimal format. The value of the parameter can range from 0 to 7 for internal PA GS101x, with a default value of 0 (for maximum RF output) and from 2 to 15 for external PA GS101x, with default value of 2 (for maximum RF output).

This command returns the standard command response (section 4) to the serial interface.

4.8.15 Sync Loss Interval

The command to configure the sync loss interval is

AT+WSYNCINTRL=<n>

n is the number of beacon interval.

On execution of this command the adapter set the sync loss interval for n times the beacon interval so that if the adapter does not receive the beacon for this time it informs the user this event as “Dissociation event”. The default value of sync loss interval is 30.This command accept the sync loss interval from 1 to 65535.

This command returns the standard command response (section 4) to the serial interface.

4.8.16 External PA

The command to enable the external PA is

AT+EXTPA=<n>

n=1 to enable the external PA

n=0 to disable external PA

If enabled ,this command forces the adapter to standby and comes back immediately and causing all configured parameters and network connection will be lost.

This command returns the standard command response (section 4) to the serial interface.

4.8.17 Association Keep Alive Timer

The command to configure the keep-alive timer interval is

AT+PSPOLLINTRL=<n>

On execution of this command, the adapter will set the keep-alive time interval for n seconds. This keep-alive timer will fire for every n seconds once the adapters associated. This timer will keep the adapter in associated state even there is no activity between AP and adapter. The default vale is 45 seconds. This command accepts keep-alive timer interval from 0 to 65535 seconds. The value 0 disables this timer.

This command returns the standard command response (section 4) to the serial interface.

4.9 Network Interface

4.9.1 Network Parameters

Note that IP addresses in the network commands are to be given in ASCII dotted-decimal format.

4.9.2 DHCP Support

The command to enable or disable DHCP is

`AT+NDHCP=n`

If *n* is 0, DHCP is disabled and if *n* is 1, DHCP is enabled.

If the interface is associated with a network, enabling DHCP will cause an attempt to obtain an IP address using DHCP from that network. Thus issuing this command with n=1 will cause the Adapter to attempt to refresh an existing DHCP address. If the Adapter is not associated when the command is received, future associations will attempt to employ DHCP. If the adapter fails to obtain an address via DHCP it will return an error response `ERROR: IP CONFIG FAIL` if verbose is enabled, or `F`(0x0F) if verbose is disabled

If the interface is not associated, this command returns the standard command response (section 4) else it returns the ip address information along with the standard command response (section 4) in the following format:

<LF><4 spaces>IP<14 spaces>SubNet<9 spaces>Gateway<3 spaces><CR><LF>

<space><IP address>:<space><SubNet address>:<space><Gateway address>

By default, DHSCP is disabled.

4.9.3 Static Configuration of Network Parameters

The command to statically configure the network parameters is

`AT+NSET=<Src Address>,<Net-mask>,<Gateway>`

Upon deployment of this command, any previously-specified network parameters are overridden, and the Adapter is configured to use the newly-specified network parameters for the current association, if associated, and for any future association. The use of DHCP is disabled if the network parameters are configured statically. The DNS address can be set using `AT+DNSSET` (4.9.7).

This command returns the standard command response (section 4) to the serial interface.

4.9.4 DHCP Server

The adapter support DHCP server and the command to start/stop the server is

AT+DHCPSRVR=1/0

1 is for start the server and 0 is for stop the server.

Prior to start the server, the adapter should be configured with a valid static ip address (using command described in section 4.9.3, both Src address and Gateway should be same) and created or configure to create a limited AP network.

This DHCP server can support maximum 8 client connections with server ip as the statically configured IP address and client ip address starts from the next ip address of the configured static IP address.

This command returns the standard command response (section 4) to the serial interface

4.9.5 DNS Server

The adapter support DNS server and the command to start/stop the server is

AT+DNS=1/0,<url>

1 is for start the server and 0 is for stop the server.

URL is the DNS name associated to the DNS IP address.

Prior to start the server, the DHCP server (section 4.9.4) should be started and created or configure to create a limited AP network. This DNS server use the same DHCP server ip address as it ip address.

This command returns the standard command response (section 4) to the serial interface.

4.9.6 DNS Lookup (Client)

The command to get an IP address from a host name is

`AT+DNSLOOKUP=<URL>,[<RETRY>,<TIMEOUT-S>]`

where `URL` is the hostname to be identified. Upon deployment of this command, the Adapter queries the DNS server to obtain the IP address corresponding to the hostname provided in `URL`, and returns the address if found. Retry and timeout are optional; if they are not given, or if 0 values are provided, the default value of 2 is used. Timeout is in seconds.

The retry range is 0 to 10 and timeout range is 0 to 20.

In addition to the standard command response , the interface returns ERROR (1 , if verbose disabled) if a valid command was issued but DNS lookup failed.

4.9.7 Static Configuration of DNS (Client)

The command to statically configure the DNS IP addresses is

`AT+DNSSET=<DNS1 IP>,[<DNS2 IP>]`

This command sets the values of the DNS server addresses to be used by the adapter. The second address, DNS2 IP, is optional but should not be same as DNS1 IP. This command returns the standard command response (section 4) to the serial interface.

This static configuration of DNS set will take effect only in the case of static IP address on the adapter.

4.9.8 Store Network Context

The command to store the network context and configuration prior to a transition to Standby is AT+STORENWCONN

This command will preserve network connection parameters (layer 2 and layer 3 information) in RTC memory when the GainSpan SOC is sent to Standby mode using the Request Standby command (4.12.2). Note that CID's are lost when the transition to Standby occurs. In addition to the standard response (section 4) this command returns " DISASSOCIATED " or 9 (based on verbose setting) if the interface is not associated state.

4.9.9 Restore Network Context

The command to recover a saved network context is

`AT+RESTORENWCONN`

This command reads the layer 3 (IP) network connection parameters saved by Store Network Context (4.9.3), and reestablishes the connection that existed before the transition to Standby. If needed, the node will re-associate and re-authenticate with the specified SSID. In addition to the usual status responses,

this command returns `ERROR` or `1` (based on verbose setting) is it is called prior to storing the network connection, or after storing the network connection but before a transition to Standby has occurred.

4.10 Connection Management Configuration

All connection commands, except for the transport of Raw Ethernet data (section 4.10.16), use the embedded TCP/IP Network Stack functions to perform the required actions. Connection identifiers, denoted as `<CID>` below, are to be sent as single hexadecimal characters in ASCII format.

4.10.1 TCP Clients

The command to open a TCP client connection will be

`AT+NCTCP=<Dest-Address>,<Port>`

Upon deployment of this command, the interface attempts to open a socket and connect to the specified address and port. The connection attempt shall timeout if a socket has not been opened after a delay equal to TCP Connection Timeout.

On successful connection, the interface sends `CONNECT<space><CID>` to the serial host along with the standard response, where CID is the newly allocated connection identifier. `ERROR` or `1` is returned if a timeout occurs.

Note:

By default the TCP keep alive option is disabled but the user can enable it using the command described in section 4.10.8.

The default TCP retransmission timeout is infinite, but the user can change it using the command described in section 4.10.8

To detect the abnormal disconnection in L3 Layer after establishing the TCP connection on the S2W adapter, the user should configure the proper values of the above two timeouts.

4.10.2 UDP Clients

The command to open a UDP client connection will be

`AT+NCUDP=<Dest-Address>,<Port>[<,Src.Port>]`

Dest-Address is the destination (server) ip address

Port is the destination (server) port

Upon deployment of this command, the interface opens a UDP socket capable of sending data to the specified destination address and port. If a source port is provided, the socket will bind to the specified port. On successful completion, the interface sends `CONNECT<space><CID>` to the serial host, followed by standard response. where CID is the newly allocated connection identifier.

The port range 0xBAC0 (47808) to 0xBACF (47823) may not be used for destination port

4.10.3 TCP Servers

The command to start a TCP server is

`AT+NSTCP=<Port>`

Upon deployment of this command, the interface opens a socket on the specified port and listens for connections. On successful creation of the server, `CONNECT<space><CID>` followed by standard command response (section 4) is sent to the serial host, where `CID` is the newly allocated connection identifier, followed by `OK` or 0. Up to 16 total CID's can be supported by the application, so a TCP server can support up to 15 distinct client connections, if no other entity has assigned CID's.

4.10.4 UDP Servers

The command to start a UDP server is

`AT+NSUDP=<Port>`

Upon deployment of this command, the interface:

- Allocates a CID for this connection. If no CID is available, the command fails.
- If a valid CID was allocated, a UDP socket is opened on the specified port.
- If the socket is successfully created, `CONNECT<space><CID>` is sent to the serial host, followed by standard command response. where CID is the allocated connection identifier.

The port range 0xBAC0 (47808) to 0xBACF (47823) may not be used.

4.10.5 Output Connections

The command to output the current CID configuration is:

`AT+CID=?`

This command returns the current CID configuration for all existing CID's:

- CID number, In decimal format.

- CID type;
- Protocol;
- Local port;
- Remote port;
- Remote IP address

followed by the usual status response. If no valid CID's are present, the message "<space>`No valid Cids`" is sent to serial interface, followed standard command response.

4.10.6 Closing a Connection

The command to close a connection is

`AT+NCLOSE=<CID>`

Upon deployment of this command, the connection associated with the specified CID is closed, if it is currently open. On completion of this command the CID is free for use in future connections. If an invalid CID is provided, the command returns `ERROR: INVALID CID` or `5`, depending on verbose status else it returns the standard command response (section 4)

4.10.7 Closing All Connections

The command to close all connections is

`AT+NCLOSEALL`

Upon execution of this command, all open connections are closed and returns the standard command response (section 4).

4.10.8 SOCKET Options Configuration

The command to configure a socket which is identified by a CID is

AT+SETSOCKOPT=<CID>,<Type>,<Parameter>,<Value>,<Length>

Upon execution of this command the adapter configure the socket identified by CID with the value passed.

CID: is the socket identifier received after opening a connection.

Type: is the type of the option to be set

- SOCKET: 65535
- IP : 0
- TCP: 6

Parameter: The Option name to be set. Accepts hex values.

- TCP_MAXRT : 10(Hex)
- TCP_KEEPALIVE: 4001(Hex)

- SO_KEEPALIVE: 8(Hex)
- TCP_KEEPALIVE_CNT: 4005(Hex)

Value: The value to be set. This in seconds (Ex: 30 → 30 seconds)

Length: The length of the value in bytes (Ex: in above case it is 4, basically it tells the type of the value is integer, Short or Char)

Integer →4

Short →2

Char →1

This command returns the standard command response (section 4) to the serial interface.

Ex:

Set the TCP retransmission timeout to 20 seconds is AT+SETSOCKOPT=0,6,10,20,4

Where 0 is the CID.

Similarly, to enable the TCP Keepalive is:

AT+SETSOCKOPT= 0,65535,8,1,4 → Enable SO_KEEPALIVE option at base socket level. Without enabling this TCP_KEEPALIVE will not work. AT+SETSOCKOPT= 0,6,4001,600,4 → Enable TCP_KEEPALIVE option at TCP level with timeout as 600 seconds.

Note: *The default keepalive count is 8 so the minimum keepalive timeout is 8*75=600 seconds. To reduce the keepalive timeout further, set the Keepalive count first to an appropriate value and set the keepalive timeout.*

Ex: To set the keep alive timeout to 75 seconds:

AT+SETSOCKOPT =0,6,4005,1,4 → Configure TCP Keep Alive Probe Sending count at just 1.

AT+SETSOCKOPT= 0,6,4001,75,4 → Enable TCP_KEEPALIVE option at TCP level with 75 seconds as Keep Alive timeout.

4.10.9 SSL Connection Open

The command to open an SSL connection is

AT+SSLOPEN=<CID>,[<certificate name>]

Upon execution of this command, the adapter opens an SSL connection over the TCP connection identified by the CID. For this SSL connection, the adapter uses the certificate stored in memory that is identified by the certificate name. Prior issuing this command, a valid TCP connection should exists with connection identifier as CID. This command returns the standard command response or ERROR if the operation fails..

4.10.10 Closing SSL connection

The command to close an SSL connection is

AT+SSLCLOSE=<CID>

Upon reception of this command, the adapter closes the existing SSL connection identified by CID. This command returns normal response codes or ERROR if the operation fails.

4.10.11 HTTP Client Configuration

The command to configure an HTTP client is

AT+HTTPCONF=<Param>,<Value>

Upon reception of this command the adapter configures the HTTP parameters. The 'param' is the HTTP header and is one of the following:

- GSN_HTTP_HEADER_AUTHORIZATION (2)
- GSN_HTTP_HEADER_CONNECTION (3)
- GSN_HTTP_HEADER_CONTENT_ENCODING (4)
- GSN_HTTP_HEADER_CONTENT_LENGTH (5)
- GSN_HTTP_HEADER_CONTENT_RANGE (6)
- GSN_HTTP_HEADER_CONTENT_TYPE (7)
- GSN_HTTP_HEADER_DATE (8)
- GSN_HTTP_HEADER_EXPIRES (9)
- GSN_HTTP_HEADER_FROM (10)
- GSN_HTTP_HEADER_HOST (11)
- GSN_HTTP_HEADER_IF_MODIFIED_SINCE (12)
- GSN_HTTP_HEADER_LAST_MODIFIED (13)
- GSN_HTTP_HEADER_LOCATION (14)
- GSN_HTTP_HEADER_PRAGMA (15)
- GSN_HTTP_HEADER_RANGE (16)
- GSN_HTTP_HEADER_REFERER (17)
- GSN_HTTP_HEADER_SERVER (18)
- GSN_HTTP_HEADER_TRANSFER_ENCODING (19)
- GSN_HTTP_HEADER_USER_AGENT (20)
- GSN_HTTP_HEADER_WWW_AUTHENTICATE (21)
- GSN_HTTP_REQUEST_URL (23)

The 'value' is a string that depends on the above parameters.

This command returns standard command response (section 4) or ERROR, if the operation fails.

4.10.12 HTTP Client Configuration Removal

The command to remove an http client configuration is

AT+HTTPCONFDEL=<Param>

Upon reception of this command the adapter removes the HTTP configuration specified by the param. The 'param' is the HTTP header and is one of the following:

- GSN_HTTP_HEADER_AUTHORIZATION (2)
- GSN_HTTP_HEADER_CONNECTION (3)
- GSN_HTTP_HEADER_CONTENT_ENCODING (4)
- GSN_HTTP_HEADER_CONTENT_LENGTH (5)
- GSN_HTTP_HEADER_CONTENT_RANGE (6)
- GSN_HTTP_HEADER_CONTENT_TYPE (7)
- GSN_HTTP_HEADER_DATE (8)
- GSN_HTTP_HEADER_EXPIRES (9)
- GSN_HTTP_HEADER_FROM (10)
- GSN_HTTP_HEADER_HOST (11)
- GSN_HTTP_HEADER_IF_MODIFIED_SINCE (12)
- GSN_HTTP_HEADER_LAST_MODIFIED (13)
- GSN_HTTP_HEADER_LOCATION (14)
- GSN_HTTP_HEADER_PRAGMA (15)
- GSN_HTTP_HEADER_RANGE (16)
- GSN_HTTP_HEADER_REFERER (17)
- GSN_HTTP_HEADER_SERVER (18)
- GSN_HTTP_HEADER_TRANSFER_ENCODING (19)
- GSN_HTTP_HEADER_USER_AGENT (20)
- GSN_HTTP_HEADER_WWW_AUTHENTICATE (21)
- GSN_HTTP_REQUEST_URL (23)

This command returns standard command response (section 4) or ERROR, if the operation fails.

4.10.13 HTTP Client Connection Open

The command to open an HTTP client connection is

AT+HTTPOPEN=<host >[, <Port Number>, <SSL Flag>, <certificate name>,<proxy>]

This command opens an HTTP client on the adapter and connects to the server specified by the host name or IP address.

- Host: Host is either the Fully Qualified Domain Name of the Server or the IP address of the server to which the HTTP client will open the connection e.g. www.gainspan.com or 74.208.130.221
- Port Number: Port number of the server to which the HTTP client will open the connection. The client can specify the port when the server is running on a non-standard port. Default is the standard port – 80 for HTTP and 443 for HTTPS.
- SSL Flag: 0 – SSL Disabled, 1 – SSL Enabled. Default is SSL Disabled
- Certificate Name: The name of the CA Certificate to be used for Server Certificate Authentication in case SSL is enabled. The CA Certificate must be provisioned before this.

It uses the certificate configured on the adapter identified by the certificate name.

- Proxy: This flag is used only during HTTPS connection through proxy 1 – The HTTPS connection is through proxy server.

It returns the normal response code and the CID of the HTTP client connection on success.

4.10.14 HTTP Client Get/Post

The command to get/post HTTP data on the HTTP client connection is

AT+HTTPSEND=<CID>,<Type>,<Timeout>,<Page>[,Size of the content]

ESC<H><Content of above size>

This command sends a get or post HTTP request to the server. The content can be transferred using the escape sequence mentioned previously.

- CID : HTTP client identifier.
- Type: GSN_HTTP_METHOD_GET (1) / GSN_HTTP_METHOD_POST (3)
- Page: The page/script being accessed e.g. /index.html
- Timeout: timeout value in seconds.
- Size: Actual Content size, Optional in case of GET

In case the HTTP connection is opened with SSL encryption enabled, this command encrypt the data based with encrypt key in SSL connection structure for the specific CID. This encryption happens before Network Layer and the Encrypted data will be sent through the network layer

Response: Receive is implicit in AT+HTTPSEND based on the HTTPS Server's response to the sent data. Received data is asynchronous and should be handled accordingly.

The response from the server is sent to the host in one or more chunks with max size of 1024 bytes. Each chunk is of the format:

<Esc>H<1 Byte - CID><4 bytes – Length of the data><data>

The data part of first chunk of the response will have the status line at the beginning . The status line contains the status code and the status phrase . This will be in the format:

<status code><space><status phrase>\r\n

After the last chunk, OK/ERROR is sent to the host.

4.10.15 Closing HTTP Client

The command to close the HTTP client connection is

AT+HTTPCLOSE=<CID>

Upon execution of this command the adapter closes the HTTP client connection identified by the CID and returns the standard command response (section 4).

4.10.16 Enable / Disable Raw Ethernet Support

The command to enable or disable support of Raw Ethernet data is:

`AT+NRAW=<0|1|2>`

The results of this command are summarized in Table .

Table 10: Raw Ethernet Support Options.

Information ID	*Description*
0	Disable Raw Ethernet frame transmission / reception.
1	Enable Raw Ethernet frames with NON-SNAP 802.2LLC headers.
2	Enable all Raw Ethernet frames.

When selection 1 is chosen, 802.3 frames are presumed to include an 802.2 header which is not a SNAP header. These frames are used, for example, for sending BACNET data over Ethernet. A frame of this type has the format:

`<ESC>R:<Length>:<DstAddr><SrcAddr>0x0000<Raw-Payload>`

On the receiving side, frames with 802.2 headers which are not a SNAP header, are sent directly to serial interface and DATA Frames with UDP port range 0xBAC0 to 0xBACF will be ignored.

When selection 2 is chosen, the 802.2 header (presumed to be a SNAP header) is removed, and a raw Ethernet II frame payload is expected, as per the format below:

`<ESC>R:<Length>:<DstAddr><SrcAddr><EtherType><Raw-Payload>`

On the receiving side, frames with 802.2 headers that are not SNAP headers and DATA Frames with UDP port ranges 0xBAC0 to 0xBACF are sent directly to serial interface.

This frame format is used for sending IP data over BACNET.

Length is size of DstAddr, SrcAddr, EtherType and Payload.

If the Adapter receives DATA Frames, where the 802.2 LLC headers' SSAP and DSAP are not both 0xAA, these frames are presumed to be 802.3 frames, and are sent to the Adapter's serial port as described above.

If the Adapter received DATA Frames with UDP port range 0xBAC0 to 0xBACF, they are presumed to be BACNET/IP frames, BacNet Ip frame, and are sent to the Adapter's serial port as described above.

This command returns standard command response (section 4).

4.10.17 Unsolicited Data Transmission

The adapter supports unsolicited data transmission (data transmission without association). The Command to enable this is:

AT+UNSOLICITEDTX=<Frame Control>,<Sequence Control>,<Channel>,<Rate>,<WmmInfo>,
<Receiver Mac>,<Bssid of AP>,<Frame Length>

This command enables the unsolicited data transmission with the parameters configured. After issuing this command, the user needs to send the payload data as following:

<ESC>D/d <PayLoad of the above Frame length>

- Frame Control: is the 802.11 frame control field. It should be limited to all data frames and management frames like beacons, association requests and probe responses.
- Sequence Control: is the sequence number of the frame. This field consists of 12 bits (LSB) fragment number and 4 bit (MSB) sequence number (0-65535).
- Channel: is the channel on which the data to be sent.
- Rate: is the rate at which the data to be send and the possible values are:

  RATE_1MBPS = 130,

  RATE_2MBPS = 132,

  RATE_5_5MBPS = 139,

  RATE_11MBPS = 150
- WmmInfo: is the wmm information to be sent.
- Receiver Mac: is the remote MAC address of the frame to be sent.
- Bssid: is bssid of the AP.
- Frame Length: is the length of the payload. The maximum size of the frame is limited to 1400 bytes.

This command returns standard command response (section 4).

4.11 BATTERY CHECK

4.11.1 Battery Check Start

The command to initiate battery checking is:

```
AT+BCHKSTRT=<Batt.chk.freq>
```

The unit of Batt.chk.freq is in number of packets send out from the Serial2WiFi adapter.

The valid range for the parameter Batt.chk.freq is between 1 and 100. Upon deployment of this command, the adapter performs a check of the battery voltage each Batt.chk.freq number of sent packets, and stores the resulting value in nonvolatile memory; only the most recent value is stored. Note that battery checks are performed during packet transmission to ensure that they reflect loaded conditions. Battery checks can be used to ensure that a battery-powered system is provided with sufficient voltage for normal operation. Low supply voltages can result in data corruption when profile data is written to flash memory.

This command returns standard command response (section 4) or ERROR, if the operation fails..

4.11.2 Battery Warning/Standby Level Set

The command to set the battery warning/standby level to enable the adapter's internal battery measuring logic:

AT+ BATTLVLSET=<Warning Level>,<Warning Freq>,<Standby Level>

Upon execution of this command the adapter's internal battery level monitoring logic starts. This command should be executed before the battery check start command (4.11.1).

Warning Level: The battery voltage, in millivolts. When the adapter battery voltage is less than this level, it sends the message "Battery Low" to the serial interface.

Warning Freq: is the frequency at which the adapter sends the "Battery Low" message to the serial interface once the adapter's battery check detected low battery.

Standby Level: The battery voltage, in millivolts, When the adapter battery voltage reaches this level, it sends the message "Battery Dead" to the serial interface and goes to long standby.

This command returns standard command response (section 4).

4.11.3 Battery Check Set

The command to set/reset the battery check period after battery check has been started is:

```
AT+BCHK=< Batt.chk.freq >
```

The valid range for the parameter Batt.chk.freq is between 1 and 100. Upon receipt, the adapter records the new value of the battery check frequency so that adapter performs the battery voltage check with the new value set. This command returns standard command response (section 4).

The same command can be used to get the current configured battery check period, the usage as follows

AT+BCHK=?

This command returns the battery check frequency along with standard command response (section 4).

4.11.4 Battery Check stop

The command to stop checking the battery state is:

AT+BCHKSTOP

Upon deployment of this command, battery check is halted. This command returns standard command response (section 4).

4.11.5 Battery Value Get

The command to retrieve the results of battery check operations is:

AT+BATTVALGET

This command should return a message with the latest value, e.g. `Battery Value: 3.4 V`, followed by the standard command response.

If this command is issued before issuing the command to start battery checks, it returns `ERROR` or `1`, depending on the current verbose setting.

4.12 Power State Management

4.12.1 Enable/Disable SOC Deep Sleep

The command to enable the GainSpan SOC's power-saving Deep Sleep processor mode is

AT+PSDPSLEEP

When enabled, the SOC will enter the power-saving Deep Sleep mode when no actions are pending. In Deep Sleep mode, the processor clock is turned off, and SOC power consumption is reduced to less than 1 mW (about 0.1 mA at 1.8 V). Note that other components external to the SOC may continue to dissipate power during this time, unless measures are taken to ensure that they are also off or disabled.

The processor can be awakened by sending data on the serial port. However, several milliseconds are required to stabilize the clock oscillator when the system awakens from Deep Sleep. Since the clock oscillator must stabilize before data can be read, the initial data will not be received; "dummy" (discardable) characters or commands should be sent until an indication is received from the application.

This command does not return any response code to the serial interface. The s2w adapter sends the message "Out of Deep Sleep" once it comes out from deep sleep.

4.12.2 Request Standby Mode

The command to request a transition to ultra-low-power Standby operation is

`AT+PSSTBY=x[,<DELAY TIME>,<ALARM1 POL>,<ALARM2 POL>]`

The parameters are:

- `x` is the Standby time in milliseconds. If a delay time (see below) is provided, the Standby count begins after the delay time has expired.
- `DELAY TIME` is the delay in milliseconds from the time the command is issued to the time when the SOC goes to Standby.
- `ALARM1 POL` is the polarity of the transition at pin 31 of the SOC which will trigger an alarm input and waken the GainSpan SOC from Standby. A value of 0 specifies a high-to-low transition as active; a value of 1 specifies low-to-high.
- `ALARM2 POL` is the polarity of the transition at pin 36 that triggers an alarm input, using the same convention used for Alarm1.

The parameters `DELAY TIME`, `ALARM1 POL`, and `ALARM2 POL` are optional. Specifying an alarm polarity also enables the corresponding alarm input.

This command does not return any response code to the serial interface. When this command is issued, the GainSpan SOC will enter the ultra-low-power Standby state (after the optional delay time if present), remaining there until *x* milliseconds have passed since the command was issued, or an enabled alarm input is received. Any current CID's are lost on transition to Standby. On wakeup, the adapter sends the message `Out of Standby-<reason of wakeup> or the corresponding error code (section 3.6.3), depending on verbose status.`

In Standby, only the low-power clock and some associated circuits are active. Serial messages sent to the UART port will not be received. The radio is off and packets cannot be sent or received. Therefore, before requesting a transition to Standby, the requesting application should ensure that no actions are needed from the interface until the requested time has passed, or provide an alarm input to awaken the SOC when needed. The alarm should trigger about 10 msec prior to issuance of any serial commands.

The Standby clock employs a 34-bit counter operating at 131,072 Hz, so the maximum possible Standby time is 131,072,000 milliseconds, or about 36.4 hours. Standby is not entered until all pending tasks are completed, and a few milliseconds are required to store any changes and enter the Standby state; a similar delay is encountered in awaking from Standby at the end of the requested time. Therefore, we do not recommend Standby times less than about 32 milliseconds.

4.13 Auto Connection

4.13.1 Wireless Parameters

The command to set the auto connection wireless parameters for the current profile is

`AT+WAUTO=<mode>,<SSID>,<BSSID>,[channel]`

- Mode is 0 for Infrastructure and 1 for Ad-hoc mode;
- SSID is the SSID of the AP or Ad-hoc Network to connect to;
- BSSID is the BSSID of the AP or Ad-hoc Network to connect to;
- Channel is the operating channel.

All other parameters required to configure the wireless connection are taken from the current Profile (3.1.3). This command returns standard command response (section 4).

4.13.2 Network Parameters

The command to set the network parameters for auto connection operation for the current profile is

`AT+NAUTO=<Type>,<Protocol>,<Destination IP>,<Destination Port>`

- Type is 0 for Client and 1 for Server;
- Protocol is 0 for UDP and 1 for TCP;
- Destination IP is the IP address of the remote system (optional if the Adapter is acting as a server);
- Destination Port is the port number to connect to on the remote system.

This command returns standard command response (section 4).

4.13.3 Enable Auto Connection

The command to enable auto connection is

`ATCn`

n is 0 to disable auto connection or 1 to enable auto connection.

Upon receipt of this command, the configuration setting in non-volatile memory is modified according to the parameter value in the command; the resulting change (if any) takes effect on the next reboot, or the next issuance of an ATA command.

This command returns standard command response (section 4).

4.13.4 Initiate Auto Connect

The command to initiate auto connection is

`ATA`

On reception of this command, the interface initiates the auto connection procedure as described in section 3.3 above, using the parameters specified by the `AT+WAUTO` and `AT+NAUTO` commands (4.13.1 and 4.13.2). The adapter responds with the IP address, subnet mask, and Gateway IP address, followed by CONNECT<space>CID and `OK` or `0` (per verbose status), if the connection is successful. If the connection attempt is unsuccessful the adapter returns `ERROR` or `1` (per verbose status). After the connection is established, the adapter enters the data transfer mode described in section 3.3 above.

If the adapter is already associated with a wireless network, the alternative command ATA2 below may be used.

4.13.5 Initiate Auto Connect – TCP/UDP Level

The command to initiate auto connection when the Adapter is already associated with an Access Point is

`ATA2`

This command requires a pre-existing wireless association. On reception of this command, the interface establishes a network connection to a TCP or UDP server with the parameters specified by the `AT+NAUTO` command (4.13.2). This command assumes a pre-existing association and should not be issued unless such exists. If the connection successful it returns CONNECT<space>CID followed by standard command response If a valid command input was received, but the connection cannot be established due to a socket bound failure, the message `ERROR: SOCKET FAILURE` or `3` (per verbose settings) is returned.

4.13.6 Return to Auto Connect Mode

The command to return to auto connect mode is

`ATO`

If the interface receives this command after it has exited the auto connect mode with +++, it shall return to auto connect mode. If the connection no longer exists, the interface attempts to reestablish the previous connection, and returns to data mode if the reconnection is successful. If the Adapter was not previously connected when this command is received, it returns an error.

This command returns standard command response (section 4) to the serial interface.

4.14 PROVISIONING

4.14.1 Web Provisioning

The adapter supports provisioning through web pages. The command to start web provisioning is

AT+WEBPROV=<user name>,<passwd>

Prior to issuing this command the adapter should be in an ad hoc or limited AP network with a valid ip address. Upon reception of this command the adapter starts a web server. It returns the normal response code OK or ERROR depends on the success or failure condition.

Once the adapter returns the success response ("OK"), the user can open a webpage on the PC (where the ad hoc network was created) with the IP address of the adapter and the HTTP client application (e.g. Internet Explorer).

If the adapter is configured as limited AP, the DHCP and DNS server should be started prior to issuing this command. Once the adapter returns the success response ("OK"), the user can open a webpage on the PC or smartphone that is connected to the limited AP.

User can configure both L2 and L3 level information on the provisioning web pages. Submit button stores all the configured information in the adapter and logout/boot button presents all provisioned information to the serial host and resets the adapter.

The size of the username and password is limited to 16 characters.

The provisioned information sends to serial host is:

- SSID=<ssid>
- CHNL=<channel>
- CONN_TYPE=<connType> /* either BSS or IBSS */
- MODE=<mode> /* 0 –> 802.11b */
- WEP_ID=<wep ID>
- WEP_KEY=<wep key>
- PSK_PASS_PHRASE=<psk PassPhrase>
- EAP_USER_NAME=<eap User name>
- EAP_PASS_WORD=<eap PassWord>
- PRIVATE_KEY_LEN=<private Key Length>
- PRIVATE_KEY=<private key file> /* private key file is stream of bytes of length= private Key Length
- CLIENT_CERT_LEN=<client Certificate Length>
- CLIENT_CERT =<client certificate> /* client certificate is stream of bytes of length= client Certificate
- CA_CERT_LEN=<CA certificate Length>
- CA_CERT =<CA certificate> /* CA certificate is stream of bytes of length= CA Certificate

- DHCP_ENBL=<0/1>
- STATIC_IP=<static IP address>
- SUBNT_MASK=<subnet Mask>
- GATEWAY_IP=<gateway>
- AUTO_DNS_ENBL=<0 /1>
- PRIMERY_DNS_IP=<primary DNS server IP>
- SECNDRY_DNS_IP<secondary DNS IP>
- NEW_USER_NAME<new User Name>
- NEW_PASS=<new Password>

This command returns standard command response (section 4) or ERROR, if the operation fails.

4.14.2 Web Provisioning (Logo)

The adapter supports adding the Logo that will appear on the web pages used for provisioning. The command to add the logo is

AT+WEBLOGOADD=<size>

<Esc>L<Actual File content>

<size> is measured in bytes and the maximum size is 1788 bytes. This command is typically done at the manufacturing line in the factory. This command can be done only once. There is no command to delete the Logo. This command returns standard command response (section 4) to the serial interface.

4.15 RF Tests

The adapter supports different types of frame transmission for RF capability measurement. It supports asynchronous data transmission/reception and modulated/un-modulated wave transmission.

4.15.1 Asynchronous Frame Transmission

The command to enable the asynchronous frame transmission is:

AT+RFFRAMETXSTART=<Channel>,<Power>,<Rate>,<No.Of.Times>,<Fr.Intrvel>,<FrameControl>,<DurationId>,<Sequence Control>,<frameLen>,<Preamble>,<Scrambler>[,<DstMac>,<Src Mac>]

This command enables the asynchronous data transmission with the parameter configured. After issuing this command the user needs to send the payload data as following,

<ESC>A/a <PayLoad of the above Frame length>

- Channel: the channel on which the data to be send.
- Power: the power in db at which the frame to be sent. The value of this parameter can range from 0 to 7 for internal PA and from 2 to 15 for external PA.
- Rate: the rate at which the data can be sent and the possible values are:

  RATE_1MBPS = 2,

  RATE_2MBPS = 4,

  RATE_5.5MBPS = 11,

  RATE_11MBPS = 22
- No. Times: the number of asynchronous frames to be sent. (1-65535)
- Fr. Interval: the interval between each frame, in microseconds. (1-65535)
- Frame Control: expects only the lower byte (B0...B7) of 802.11 frame control field, which includes protocol version, Type and Subtype. All the higher order bits (B8...B15) are made zero for this command.

  E.g. Frame control field of beacon frame is: 128

Higher Byte B15 – B8	Sub Type B7-B4	Type B3- B2	Protocol Version B1 – B0
00000000	1000	00	00

  *Note: This command is intended to transfer **only** data & a few Management frames like Beacon/Probe request/Probe response/Association request.*
- DurationId: duration id information to be sent. (0-65535)
- Sequence Control: the sequence number of the frame. This field consists of 12 bits(LSB) fragment number and 4 bit (MSB)sequence number. (0-65535)
- frameLen: the length of the payload. The maximum size of the frame is limited to 1400 bytes.
- Preamble: the short (1) or long (0) preamble.
- Scrambler: the ON(0) or OFF(1) scrambler field of the frame
- DstMac: the MAC address through which the frame to be send.
- Src Mac: MAC address for the WiFi Bridge.

Example: **AT+RFFRAMETXSTART=1,3,4,2,200,0,11,0,30,0,1,00:1d:c9:00:07:a2**

<ESC>A12345678901234567890123456789O

Please check the wireless sniffer to see the frame on air.

The AT+RFSTOP (section 4.16.4) command should be issued prior to successive frame transmission command.

CSMA/CA is not executed before transmitting this command; hence it could destroy the network.

This command returns standard command response (section 4) to the serial interface.

4.15.2 Asynchronous Frame Reception

The command to enable the asynchronous frame reception is:

AT+RFRXSTART=<Channel>[,<Sendtouser>]

- Channel: is the channel on which the data to be received.
- Sendtouser: is a flag (0/1) which instructs the adapter to send the received data to the serial interface.

The Frame Transmission/Reception Stop command (4.15.4) will send the status information of the received frames to the serial interface.

Example: **AT+RFRXSTART=1,1** → this will send the received data to the serial interface

AT+RFRXSTART=1,0 → this will not send the received data to the serial interface

In both case the received frame information is stored in SRAM and once issue the command AT+RFSTOP sends the received frame information to the user through serial. We recommend using the second option. This command returns standard command response (section 4) to the serial interface.

4.15.3 Modulated/Un-Modulated Wave Transmission

The command to enable the modulated/un-modulated wave transmission is:

AT+RFWAVETXSTART=<Modulated>,<Channel>,<Rate>,<PreambleLong>,<ScamblerOff>,<Cont.Tx>,<Power>,<Ssid>

- Modulated : is the flag to tell whether the wave transmission should be modulated(1) or un-modulated (0)
- Channel: is the channel on which the data to be received.
- Rate: the rate at which the wave transmission should happen.
  TX_RATE 1mbps = 0,
  TX_RATE 2 mbps = 1,
  TX_RATE 5.5 mbps = 2,
  TX_RATE 11 mbps = 3,
- PreambleLong: is long preamble (1) or short preamble (0).
- ScamblerOff: is the scrambler field OFF (0) or ON (1).
- Cont.Tx: is the wave transmission is continuous (1) or not (0).
- Power: is the power in db at which the wave transmission should happen. The value of this parameter can range from 0 to 7 for internal PA and from 2 to 15 for external PA.
- Ssid: is the ssid of the network created for the wave transmission.

Example: AT+RFWAVETXSTART=1,4,2,1,1,1,3,aaa -→(modulated)

AT+RFWAVETXSTART=0,4,3,0,1,1,3,bbb --→(un-modulated)

This command returns standard command response (section 4) or ERROR if it fails.

4.15.4 Frame Transmission/Reception Stop

The command to stop any of the RF tests transmission/reception is:

AT+RFSTOP

Upon reception of this command the adapter stops any of the frame transmission/reception RF tests started. This command sends the status information of the received asynchronous frames to the serial interface other than the normal command response if this command issued for the asynchronous frame reception stop.

Example:

AT+RFSTOP (if this command issued after AT+ RFRXSTART, then it sends the following information to the serial interface)

Total frames received =xxxx

Correct frames received =xxxx

Incorrect frames received =xxx

FCS Error frames received =xxx

4.16 Miscellaneous

4.16.1 Enhanced Asynchronous Notification

S2w Adapter supports an enhanced asynchronous notification method. The command to enable/disable this feature is

AT+ASYNCMSGFMT=n

n is

- 0 – Disable this feature
- 1 – Enable this feature

This command returns standard command response (section 4) to the serial interface. Default is disable

Enabling this feature results with all asynchronous messages going to the serial interface with a header. Also during these asynchronous message transfer s2w adapter make the gpio 19 high. The asynchronous message format is as shown below:

<ESC><TYPE><SUBTYPE><LENGTH><MESSAGE>

TYPE – Type of message and the length is one byte. For asynchronous message , it is 0x41 (Ascii value A)

SUBTYPE – Message subtype and the length of this field is one byte. Normally this field contains the ascii value of the subtype message. Refer section 3.7.4 for subtype values.

LENGTH – Length of the asynchronous message in hex. This field length is 2 bytes.

MESSAGE – Exact asynchronous message as string. Refer section 3.7.4 for all enhanced asynchronous messages.

4.16.2 Node Start Up Handling

For proper synchronization between host micro controller (MCU) and S2w node, the following steps must be followed:

- In case of UART interface, during boot up host MCU shall send dummy 'AT' command and wait for response from the S2w node. The host MCU must continuously send these dummy 'AT' commands till 'OK' response is received from S2w node.
- In case of SPI interface, during boot up host MCU must check the status of host wake-up signal (GPIO#28 of S2w node). Once host wake-up signal is HIGH, then host MCU can send the 'AT' commands.
- If for some reason host MCU getting reset, then S2w adapter must be explicitly reset using EXT_RESET pin and the MCU should wait for the wake-up signal(GPIO#28) become high in case of SPI interface. However if reset provision is not available, then host MCU must continuously send dummy 'AT' commands till 'OK' response is received from S2w adapter.

4.16.3 Firmware Upgrade

The command to upgrade the firmware is

`AT+FWUP= <SrvIp>,<SrvPort>,<SrcPort>,[<retry>]`

This command starts the firmware upgrade procedure over the wireless link.

- SrvIp is the IP address of the firmware upgrade server;
- SrvPort is the server port number to be used for firmware upgrade;
- SrcPort is the adapter port number to be used for firmware upgrade.
- Retry is the number of times the node will repeat the firmware upgrade attempt if failures are encountered. The default value is 10 and the retry count ranges from 0 to 0xffffffff.

When a valid command has been received, the adapter returns the message: `Firmware upgrade is going on, Please wait....` followed with the status message OK or 0, which applies only to the validity of the command. After attempting to upgrade the firmware, the node sends an additional message describing the result of the actual firmware upgrade attempt.

After a successful firmware upgrade, the Adapter will reset and boot up using the updated firmware; when startup is complete, it will issue the message `APP Reset-FW-UP-SUCCESS`.

If the firmware upgrade attempt failed, the Adapter will reset and boot up with the old firmware, and issue the message `APP Reset-FW-UP-FAILURE`.

If the firmware upgrade attempt failed after successful upgrade of one flash image (flash0), the adapter will reset and boot up, issue the message APP Reset-FW-UP-RECOVERY, associate back to the network with previous settings and try to upgrade the firmware again. The retry count decides how many times this can be done.

If the node is not associated, the adapter returns `ERROR` or `1`, based on verbose settings.

4.16.4 SPI Interface Handling

In the case of SPI interface, the GS101X node acts as slave and will communicate to master SPI controller. By default, SPI interface supports Motorola protocol with clock polarity 0 and clock phase 0. For more detailed specification of SPI frame format and timing characteristics refer GS1011 data sheet.

Since SPI data transfer works in full duplex mode, its required to make use of special octet to indicate idle data. Similarly if host MCU is sending data at higher rate flow control mechanism is required. In order differentiate these special control codes (such as idle pattern , flow control codes and other control octets) from user data, byte stuffing mechanism is incorporated.

SPI transmit data handling procedure:
The SPI data transfer layer makes use of an octet (or byte) stuffing procedure. The Control Escape octet is defined as binary 11111011 (hexadecimal **0xFB**), most significant bit first. Each special control pattern is replaced by a two octet sequences consisting of the Control Escape octet followed by the original octet exclusive-or'd (XOR) with hexadecimal **0x20**. Receiving implementations must correctly process all Control Escape sequences.

Escaped data is transmitted on the link as follows:

Pattern	Encoded as	Description
0xFD	0xFB 0xDD	***Flow control XON***
0xFA	0xFB 0xDA	***Flow control XOFF***
0x00	0xFB 0x20	Inactive link detection
0xFB	0xFB 0xDB	Control ESCAPE
0xF5	0xFB 0xD5	***IDLE character***
0xFF	0xFB 0xDF	Inactive link detection
0xF3	0xFB 0xD3	SPI link ready indication

One dedicated GPIO signal (***GS_SPI _HOST_WAKEUP***: ***GPIO#28)*** is available for data ready indications from Slave GS1011 node to Master Host controller. This ***GS_SPI _HOST_WAKEUP*** signal is asserted high during valid data transmission period, so that the host (master SPI) starts pulling out data by giving SPI clock and ***GS_SPI _HOST_WAKEUP*** signal is de-asserted once transmission is completed. Master host controller must provide clock as long as ***GS_SPI_HOST_WAKEUP*** signal is active.

Special character (***GS_SPI _IDLE***) will be transmitted during idle period (if there is no more data to transmit) and must be dropped at the receiving Host.

SPI receive data handling procedure:
Since byte stuffing is used, each Control Escape octet must be removed and the next immediate octet is exclusive-or'd (XOR) with hexadecimal **0x20**. If received buffer has reached the upper water mark, then ***XOFF*** character will be sent out informing the host to stop transmitting actual data. After receiving ***XOFF*** character host must stop transmitting actual data and can send IDLE bytes, until the XON is received. Once the host receives ***XON***, then it may resume the valid data transmissions.

Special control byte ***IDLE*** will be dropped at receiver.

4.16.5 Pin connection for SPI Interface

Host MCU	S2W Node	Remarks
MSPI_DOUT	SSPI_DIN	
MSPI_DIN	SSPI_DOUT	
MSPI_SS	SSPI_SS	
MSPI_CLK	SSPI_CLK	
GPIO	GPIO#28	Host wake-up signal
Ground	Ground	

4.16.6 Factory Defaults

The Serial2Wifi adapter stores the factory defaults to its flash; currently supporting only the MAC addresses as factory the default. If the factory default MAC address location contains a valid address, then the Serial2Wifi adapter reads and uses it as the MAC address, otherwise it use the default MAC as it MAC address.

The factory default location starts at 122Kbytes of second application flash and the Serial2Wifi stores the factory default in the following format:

Checksum(1 byte)	Length (1 byte)	Mac address (6 byte)

Checksum : the simple byte wise xor of both length and MAC address.

Length : the length in bytes of MAC address and length (here it is 7).

Mac Address : the MAC address. The user can override the factory default MAC address by using the AT commands mentioned in section 4.7.1.

4.16.7 Set System Time

The command to set the adapter system time is

AT+SETTIME=<dd/mm/yyyy>,<HH:MM:SS>

Upon execution of this command the adapter set its system time to the time specified as the parameters and returns the standard command response.

4.16.8 Get System Time

The command to get the current system is

AT+GETTIME=?

Upon reception of this command the adapter sends the current system time in milliseconds since epoch (1970) followed by the standard command response to the serial interface. The time format comes on the serial interface as follows:

"Current Time in msec since epoch=xxxxxxx"

4.16.9 GPIO Out HIGH/LOW

The command to set/reset (high/low) a gpio pin is

AT+DGPIO=<GPIO-NO>,<SET/RESET(0/1)>

This command sets the Gpio 'GPIO-NO' pin level to high or low as per the SET/RESET parameter and returns the standard command response (section 4)

Note: Only the Gpio Pins which are not mixed with the any used IOs like UART/SPI etc. that can be set high/low with this command.

The supported Gpios and the corresponding numbers are:

Gpio10 : 10

Gpio11 : 11

Gpio30 : 30

Gpio31 : 31

4.16.10 Error Counts

The command to get the error count statistics is

AT+ERRCOUNT=?

This command returns error count information to the interface followed by the standard command response (section 4).

The error counts include:

Watchdog reset counts

Software reset counts

Wlan abort/assert counts

4.16.11 Version

The command to output the current version information is

```
AT+VER=?
```

The command returns version information followed by the standard command response (section 4). to the serial host:

- Serial-to-Wi-Fi version;
- GainSpan Embedded Platform Software version;
- WLAN firmware version.

4.16.12 Ping

The command to initiate a network *ping* is:

```
AT+PING=<Ip>,[[<Trails>],[<Interval>],[<Len>],[<TOS>],[<TTL>],[<PAYLOAD>]]
```

Upon deployment of this command the device sends a *ping* to the remote machine specified by the IP address.

- Ip is the IP address of the server to which the command is directed;
- Trails indicate the number of *ping* requests to send. The default value is 0; in this case, *ping* will continue until terminated as described below.
- Interval is the interval in milliseconds between each *ping* request; the valid range is 1000-99000. The default value is 3000.
- Len is the length of the *ping* packet; the valid range is 0 to 1024. The default value is 56.
- TOS is the type of service; the valid range is 0-99. The default value is 0.
- TTL is the time to live; the valid range is 0-2047. The default value is 30.
- Payload is the data to be sent in each *ping* packet. The payload length should be in the range 0-16; the payload may contain valid alphanumeric characters (0-9, a-e).

To terminate a Ping sequence, issue `<Esc> C`.

4.16.13 Trace Route

The command to start a *trace route* operation is:

```
AT+TRACEROUTE=<Ip>,[[<Interval>],[<MaxHops>],[<MinHops>],[<TOS>]]
```

Parameters:

- Ip is the IP address of the remote server;
- Interval is the interval in milliseconds between each request; the valid range is 1000-99000. The default value is 1000.
- MaxHops is the maximum time-to-live; the valid range is 2-99. The default value is 30.

- MinHops is the minimum time-to-live; the value given should be greater than 1 and less than MaxHops. The default value is 1.
- TOS is the type of service; the valid range is 0-99. The default value is 0.

Upon reception of this command the adapter starts the trace route operation and returns the following information to serial interface along with the standard command response(section 4).

<LF>Tracing Route to<space><Ip address><space>over a max hops<space>< MaxHops><CR><LF>

During this trace route operation the adapter sends the ping delays and the next hop ip address information to the serial interface one at a line in the following format:

<CR><LF><current TTL ><2 space><1st RTT in 4 bytes>ms<2 space><2nd RTT in 4 bytes>ms<2 space><3rd RTT in 4 bytes>ms<2 space><ip address of hop>

Once the trace route operation complete, the adapter sends the message"<CR><LF><CR><LF> Trace Complete<CR><LF>" to the serial interface.

4.16.14 Memory Trace

The command to display the adapter memory trace information is

AT+MEMTRACE

Upon reception of this command the adapter sends the memory trace information to the serial interface along with standard command response.

The memory trace information contains the following :

- Number Of Allocation
- Number Of Free
- Current Used Memory in bytes
- Peak Memory Usage in bytes
- Memory Details of currently used allocations in the following format:

  <address>,<line number>,<size>,<module name>
- Number of Allocations to be freed

5 References

[1]	GS1011M or GS1011 Data Sheet, GS1011-DS, GainSpan Corporation
[2]	IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 802.11-2007, IEEE, www.ieee.org
[3]	V.250, Serial asynchronous automatic dialing and control, and V.251, Procedure for DTE-controlled call negotiation, International Telecommunications Union, www.itu.int
[4]	**Communications Networks,** A. Leon-Garcia and I. Widjaja, McGraw-Hill 2000, p. 582
[5]	**AN023 – Battery Level Check,** GainSpan Corporation

6 Appendix

6.1 Data handling using Esc Sequences on UART Interface

Flow Control	*Data Mode (Data Type)*	*Connection Type*	*Description and Escape Command Sequence*
SW or HW	Normal (ASCII Text)	TCP client TCP server	This escape sequence selects the specified Connection ID as the current connection. This switches the connection to be used without exiting from the Data mode of operation. Use this sequence to send data from a TCP server, TCP client or UDP client (must be done before data can be received by that client). GS1011 send and receive sequence: <Esc>S<CID><data><Esc>E Example: To send user data (e.g. Hello) on CID 1, the format will be: <Esc>S1Hello<Esc>E
SW or HW	Normal (ASCII Text)	UDP client	If UDP client is configured with unicast destination server IP address, then: GS1011 send and receive sequence: <Esc>S<CID><data><Esc>E If UDP client is configured with broadcast destination server IP address (i.e. 255.255.255.255), then: GS1011 expects to receive the following data sequence from Host: <Esc>S<CID><data><Esc>E GS1011 sends the following data sequence to Host: <Esc>u<CID><IPAddress><space><port><horizontal tab<data><Esc>E

Flow Control	Data Mode (Data Type)	Connection Type	Description and Escape Command Sequence
SW or HW	Normal (ASCII Text)	UDP server	This escape sequence is used when sending and receiving UDP data on a UDP server connection. When this command is used, the remote address and remote port is transmitted. GS1011 expects to receive the following data sequence from Host: <Esc>U<CID><IP Address>:<port>:<data><Esc>E GS1011 sends the following data sequence to Host: <Esc>u<CID><IPAddress><space><port><horizontal tab<data><Esc>E Example: When GS1011 sends data (e.g. Hello) on CID 0, the format will be: <Esc>u0192.168.0.101<space>1001<horizontal tab>Hello<Esc>E
SW or HW	Normal (Binary)	NA	Binary data transfer with software or hardware flow control are not supported with ESC sequence.
SW or HW	Bulk (ASCII Text)	TCP client TCP server	To improve data transfer speed , one can use this bulk data transfer. This sequence is used to send and receive data on TCP client, TCP server, or UDP client connection. GS1011 send and receive sequence: <Esc>Z<CID><data length><data> Example: To send a 5 byte user data (e.g. Hello) on CID 1, the format will be: <Esc>Z10005Hello

Flow Control	Data Mode (Data Type)	Connection Type	Description and Escape Command Sequence
SW	Bulk (ASCII Text or Binary)	UDP client	If UDP client is configured with an unicast destination server IP address, then GS1011 send and receive sequence: <Esc>Z<CID><Data Length><data> If UDP client is configured with a broadcast destination server IP address (i.e. 255.255.255.255), then: GS1011 expects to receive the following data sequence from Host: <Esc>Z<CID><Data Length><data> GS1011 sends the following data sequence to Host: <Esc>y<CID><IPAddress><Space><Port><horizontal tab><data length><data>
SW or HW	Bulk (ASCII Text)	UDP server	This escape sequence is used when sending and receiving UDP bulk data on a UDP server connection. When this command is used, the remote address and remote port is transmitted. GS1011 expects to receive the following data sequence from Host: <Esc>Y<CID><IP address>:<port>:<data length><data> GS1011 sends the following data sequence to Host: <Esc>y<CID><IPAddress><Space><Port><horizontal tab><data length><data> Example: When receiving a 5 byte user data (e.g. Hello) on CID 1, the format will be: <Esc>y0192.168.0.101<space>1001<horizontal tab>0005Hello

Flow Control	Data Mode (Data Type)	Connection Type	Description and Escape Command Sequence
HW	Bulk (Binary)	TCP client TCP server UDP client	To improve data transfer speed , one can use this bulk data transfer. This sequence is used to send and receive data on TCP client, TCP server, or UDP client connection. GS1011 send and receive sequence: <Esc>Z<CID><data length><data> Example: To send a 5 byte user data (e.g. Hello) on CID 1, the format will be: <Esc>Z10005Hello
SW	Bulk (Binary)	NA	Binary data transfer with software flow control not supported.

6.2 Data handling using Esc Sequences on SPI Interface

Data Mode (Data Type)	*Connection Type*	*Description and Escape Command Sequence*
Normal (ASCII Text)	TCP client TCP server	1. Data transfer is transparent due to byte stuffing at SPI driver level. 2. Byte stuffing must be incorporated in Host controller as per the Adaptor guide. GS1011 send and receive sequence: <Esc>S<CID><data><Esc>E or Auto mode
Normal (ASCII Text)	UDP client	If UDP client is configured with an unicast destination server IP address, then: GS1011 send and receive sequence: <Esc>S<CID><data><Esc>E If UDP client is configured with a broadcast destination server IP address (i.e. 255.255.255.255), then: GS1011 expects to receive the following data sequence from MCU: <Esc>S<CID><data><Esc>E GS1011 sends the following data sequence to MCU: <Esc>u<CID><IP Address><space><port><horizontal tab<data><Esc>E
Normal (ASCII Text)	UDP server	This escape sequence is used when sending and receiving UDP data on a UDP server connection. When this command is used, the remote address and remote port is transmitted.

Data Mode (Data Type)	Connection Type	Description and Escape Command Sequence
		GS1011 expects to receive the following data sequence from Host: <Esc>U<CID><IP Address>:<port>:<data><Esc>E GS1011 send the following data sequence to Host: <Esc>u<CID><IP Address><space><port><horizontal tab<data><Esc>E Example: When receiving user data (e.g. Hello) on CID 0, the format will be: <Esc>u0192.168.0.101<space>1001<horizontal tab>Hello<Esc>E
Normal (Binary)	NA	Binary data transfer with software flow control is not supported with ESC sequence.
Normal (ASCII Text or Binary)	NA	Hardware flow control is not supported.
Bulk (ASCII Text or Binary)	TCP client TCP server	1. Data transfer is transparent due to byte stuffing at SPI driver level. 2. Byte stuffing must be incorporated in Host controller as per the Adaptor guide. GS1011 send and receive sequence: <Esc>Z<CID><Data Length><data> Example: To send a 5 byte user data (e.g. Hello) on CID 1, the format will be: <Esc>Z10005Hello

Data Mode (Data Type)	Connection Type	Description and Escape Command Sequence
Bulk (ASCII Text or Binary)	UDP client	If UDP client is configured with an unicast destination server IP address, then: GS1011 sends and receives the following data sequence: <Esc>Z<CID><Data Length><data> If UDP client is configured with a broadcast destination server IP address (i.e. 255.255.255.255), then: GS1011 expects to receive the following data sequence from Host: <Esc>Z<CID><Data Length><data> GS1011 sends the following data sequence to Host: <Esc>y<CID><IP Address><Space><Port><horizontal tab><data length><data>
Bulk (ASCII Text or Binary)	UDP server	This escape sequence is used when sending and receiving UDP bulk data on a UDP server connection. When this command is used, the remote address and remote port is transmitted. GS1011 receives from Host the following data sequence: <Esc>Y<CID><IP address>:<port>:<data length><data> GS1011 sends the following data sequence to Host: <Esc>y<CID><IP Address><Space><Port><horizontal tab><data length><data> Example: When receiving a 5 byte user data (e.g. Hello) on CID 1, the format will be: <Esc>y0192.168.0.101<space>1001<horizontal tab>0005Hello