## DEFINITIONS

In this document, the following naming conventions are used:

* DCE (Data Communications Equipment): the Wi-Fi module.
* DTE (Data Terminal Equipment): the terminal that issues commands to the Wi-Fi module.

## NOTE

* In the command description below or while using the AT commands with the module, there are some parameters which are placeholder for future releases. Please refer to this document to find the list of parameters which are supported. If a parameter is not mentioned in this document but gets printed while reading the configuration, it is just a placeholder for a future release.

# Command Reference: Scanning

## Configure Scan Behaviour

### Description

This command is used to modify or query the configuration of the scanning function.

### Command Syntax

|  |  |
| --- | --- |
| Command | Description |
| AT+WSCNC[=<param\_id>] | Read configuration |
| AT+WSCNC=<param\_id>,<param\_val> | Set configuration |

### Supported Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name & Default Value | Type | Description |
| 1 | <SCAN\_TIME> | Integer | The time in milliseconds to wait for probe responses |
| 2 | <PASV\_LISTEN> | Integer | The time in milliseconds to wait for beacons |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +WSCNC:<param\_id>,<param\_val> | Read response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### Examples:

|  |  |
| --- | --- |
| AT+WSCNC | Query the current active scan configuration settings |
| +WSCNC:1,30 | 30ms wait |
| +WSCNC:2,300 | 300ms duration for passive scans |
| OK | Command completed |

## Active Scanning Command

### Description

This command is used to actively scan for infrastructure networks in range of the DCE.

This form of scanning causes the device to transmit energy on each channel being scanned.

### Command Syntax

AT+WSCNA=<CHANNEL>[,<SSID>]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <CHANNEL> | Integer | | The channel to scan, a value of 255 scans all available channels | |
| <SSID> | String | | Scan for a specific SSID, confirms presence of a cloaked network | |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax (scan indications)

:<RSSI>,<SEC\_TYPE>,<CHANNEL>,<BSSID>,<SSID>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <RSSI> | Integer | Received Signal Strength (higher is better) |
| <SEC\_TYPE> | Integer | Security Type   |  |  | | --- | --- | | 0 | No security (open network) | | 1 | WEP | | 2 | WPA Personal (pre-shared key) | | 3 | WPA2 Personal (pre-shared key) | |
| <CHANNEL> | Integer | Channel number of detected device |
| <BSSID> | String | BSSID of detected device |
| <SSID> | String | SSID of detected device |

### AEC Syntax (scan completed)

### AEC Syntax (scan failed)

### Example:

|  |  |
| --- | --- |
| AT+WSCNA=255 | Scan all channels |
| OK |  |
| +WSCNIND:-57,0,1,"98:FC:11:41:E7:7C","test-network" | Scan indication: open network, channel 1, SSID test-network |
| +WSCNIND:-62,0,6,"24:F2:7F:A5:8B:81",[466F6F20C2A920626172] | Open network, channel 6, with UTF-8 SSID Foo © Bar |
| +WSCNDONE | Indication of completion of scanning activities |

## Passive Scanning Command

### Description

This command is used to passively scan for infrastructure networks in range of the DCE.

This form of scanning causes the device to tune to the required channel(s), turn on its receiver, and listen for beacons from nearby devices. No energy is transmitted on the scanned channel(s) during this operation.

### Command Syntax

AT+WSCNP=<CHANNEL>[,<SSID>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <CHANNEL> | Integer | The channel to scan, a value of 255 scans all available channels |
| <SSID> | Integer | Scan for a specific SSID |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax (scan indications)

:<RSSI>,<SEC\_TYPE>,<CHANNEL>,<BSSID>,<SSID>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <RSSI> | Integer | Received Signal Strength (higher is better) |
| <SEC\_TYPE> | Integer | Security Type   |  |  | | --- | --- | | 0 | No security (open network) | | 1 | WEP | | 2 | WPA Personal (pre-shared key) | | 3 | WPA2 Personal (pre-shared key) | | 4 | Enterprise | |
| <CHANNEL> | Integer | Channel number of detected device |
| <BSSID> | String | BSSID of detected device |
| <SSID> | String | SSID of detected device |

### AEC Syntax (scan completed)

### AEC Syntax (scan failed)

### Examples:

|  |  |
| --- | --- |
| AT+WSCNP=255 | Scan all channels |
| OK |  |
| +WSCNIND:-57,0,1,"98:FC:11:41:E7:7C","test-network" | Scan indication: open network, channel 1, SSID test-network |
| +WSCNIND:-62,4,1,"24:F2:7F:A5:8B:80","enterprise-secure" | Scan indication: Enterprise, channel 1, SSID enterprise-secure |
| +WSCNIND:-62,0,6,"24:F2:7F:A5:8B:81",[466F6F20C2A920626172] | Open network, channel 6, with UTF-8 SSID Foo © Bar |
| +WSCNIND:-70,1,11,"24:F2:7F:A5:8B:81",[] | Open network, channel 11, with hidden SSID |
| +WSCNDONE | Scanning completed |

# Command Reference: WLAN Configuration

## Wi-Fi Station Configuration

### Description

This command is used to read / set the DCE’s Wi-Fi station mode configuration.

### Command Syntax

|  |  |
| --- | --- |
| Command | Description |
| AT+WSTAC[=<param\_id>] | Read configuration |
| AT+WSTAC=<param\_id>,<param\_val> | Set configuration |

### Supported Parameters

When not connected to an AP these parameters are available for configuration.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name & Default Value | Type | Description |
| 1 | <SSID> | String | Network Name (Mandatory parameter) |
| 2 | <SEC\_TYPE>: 0 | Integer | Security Type   |  |  | | --- | --- | | 0 | No security (open network) | | 1 | WEP | | 2 | WPA Personal (pre-shared key) | | 3 | WPA2 Personal (pre-shared key) | | 4 | Enterprise | |
| 3 | <CREDENTIALS>: “” | String | Credentials for connecting to the network:   |  |  | | --- | --- | | SEC\_TYPE | Credentials format (ASCII or hex string) | | 0 | “” (empty string) | | 1 | Any ASCII or hex string | | 2 | Of format “KEYINDEX\*PASSWORD” | | 3 | Of format “USERNAME\*PASSWORD” | |
| 4 | <CHANNEL>: 255 (any channel) | Integer | The channel the network must reside on. |
| 12 | <NTP\_SVR>: “” | String | The address/name of NTP server |
| 13 | <NTP\_STATIC>: 0 | Integer | NTP configuration mode   |  |  | | --- | --- | | 0 | DHCP – can be set via DHCP  **(Placeholder for future release)** | | 1 | Static – cannot be set by DHCP | |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +WSTAC:<param\_id>,<param\_val> | Read response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

## Wi-Fi Station, Start and Stop , & Link Notifications

### Description

This command is used to enable the DCE’s station mode functionality.

### Command Syntax

|  |  |
| --- | --- |
| Command | Description |
| AT+WSTA | Read State of STA function |
| AT+WSTA=<STATE> | Set State of STA function |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <STATE> | Integer | State of the Wi-Fi station feature   |  |  | | --- | --- | | 0 | Disable | | 1 | Use configuration from +WSTAC command | |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +WSTA:<ASSOC\_ID>,<STATE>[,<SSID>,<RSSI>,<SEC\_TYPE>] | Read response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ASSOC\_ID> | Integer | Association ID |
| <SSID> | String | Network Name |
| <RSSI> | Integer | Received Signal Strength (higher is better)  **Placeholder for future release (always reported as 0 in this version)** |
| <SEC\_TYPE> | Integer | Security Type   |  |  | | --- | --- | | 0 | No security (open network) | | 1 | WEP (**Not Supported in PIC32MZW1)** | | 2 | WPA Personal (pre-shared key) | | 3 | WPA2 Personal (pre-shared key) | | 4 | Enterprise | |

### AEC Syntax (Link established)

: <ASSOC\_ID>,<BSSID>,<CHANNEL>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ASSOC\_ID> | Integer | Association ID |
| <BSSID> | String | The BSSID of the Access Point the DCE has connected to |
| <CHANNEL> | Integer | The channel number of network |

### AEC Syntax (Link lost)

: <ASSOC\_ID>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ASSOC\_ID> | Integer | Association ID |

### AEC Syntax (Indication of STA Automatic Address Assignment)

: <ASSOC\_ID>,<IP\_ADDRESS>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ASSOC\_ID> | Integer | Association ID |
| <IP\_ADDRESS> | String | IP address assigned |

### AEC Syntax (Connection Error)

: <ERROR\_CODE>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ERROR\_CODE> | Integer | Error Code |

### Examples:

|  |  |
| --- | --- |
| AT+WSTAC=1,"MyAP" | Configure AP connection parameters |
| OK |  |
| AT+WSTAC=2,1 |  |
| OK |  |
| AT+WSTAC=3,"MyAPPSK" |  |
| OK |  |
| AT+WSTA=1 | Connect to AP |
| OK |  |
|  |  |
| +WSTALU:1,"00:01:02:03:04:05",6 | Link up: connected on channel 6 |
| +WSTAAIP:1,"19.168.0.20" | IP address received via auto configuration (DHCP/SLAAC) |
|  |  |
| AT+WSTA |  |
| +WSTA:1,1,"MyAP",-50,0 |  |
| OK |  |
|  |  |
| +WSTALD | Link down |

## Wi-Fi Hotspot (AP) Configuration +WAPC

### Description

This command is used to read or set the DCE’s hotspot access point configuration.

### Command Syntax

|  |  |
| --- | --- |
| Command | Description |
| AT+WAPC[=<param\_id>] | Read configuration |
| AT+WAPC=<param\_id>,<param\_val> | Set configuration |

### Supported Parameters

When not configured as an AP these parameters are available for configuration.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name & Default Value | Type | Description |
| 1 | <SSID>: DEMO\_AP | String | Network Name |
| 2 | <SEC\_TYPE>: 0 | Integer | Security Type   |  |  | | --- | --- | | 0 | No security (open network) | | 1 | WEP | | 2 | WPA Personal (pre-shared key) | | 3 | WPA2 Personal (pre-shared key) | | 4 | Enterprise | |
| 3 | <CREDENTIALS>: “” | String | Credentials required for connecting to the network of the security type specified:   |  |  | | --- | --- | | SEC\_TYPE | Credentials format | | 0 | “” (empty string) | | 1 | Any ASCII or hexadecimal string | | 2 | Any ASCII or hexadecimal string of format  “KEYINDEX\*PASSWORD” | | 3 | Any ASCII or hexadecimal string of format “USERNAME\*PASSWORD” | |
| 4 | <CHANNEL>: 6 | Integer | The channel of the network to connect to |
| 5 | <HIDDEN>: 0 | Integer | Visibility of the network   |  |  | | --- | --- | | 0 | Not hidden, SSID is broadcast in beacons | | 1 | Hidden, SSID is not broadcast | |

When configured as an AP the parameter entries are ‘Read Only’ and cannot be modified.

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +WAPC:<param\_id>,<param\_val> | Read response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |
| --- | --- |
|  |  |

## Wi-Fi Hotspot (Soft AP) Start/Stop/Status , & Link Indications

### Description

This command is used to enable the DCE’s hotspot access point functionality.

### Command Syntax

|  |  |
| --- | --- |
| Command | Description |
| AT+WAP | Read Status of AP function |
| AT+WAP=<STATE> | Set State of AP function |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <STATE> | Integer | State of the hotspot feature   |  |  | | --- | --- | | 0 | Disable | | 1 | Enable | |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +WAP:<STATE>[,<BSSID>,<RSSI>] | Query state response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <BSSID> | String | The BSSID of the STA connecting to the AP |
| <RSSI> | Integer | Received Signal Strength (higher is better) |

### AEC Syntax (Station Connected)

: <ASSOC\_ID>,<BSSID>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ASSOC\_ID> | Integer | Association ID |

### AEC Syntax (Station Disconnected)

: <ASSOC\_ID>,<BSSID>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ASSOC\_ID> | Integer | Association ID |

### AEC Syntax (DHCP Server Indication of Connected STA Address Assignment)

: <ASSOC\_ID>,<IP\_ADDRESS>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ASSOC\_ID> | Integer | Association ID |

Examples:

|  |  |
| --- | --- |
| AT+WAPC=1,"MyAPSSID" | AP SSID |
| OK |  |
| AT+WAPC=2,2 | WEP security |
| OK |  |
| AT+WAPC=3,"MyAPPassword" | WEP key |
| OK |  |
| AT+WAPC=4,11 | Channel |
| OK |  |
| AT+WAP=1 |  |
| OK |  |
|  |  |
| +WAPSC:2,"00:01:02:03:04:05" | STA connected |
| +WAPAIP:2,"192.168.0.100" | STA assigned IP address |
|  |  |
| +WAPSD:2,"00:01:02:03:04:05" | STA disconnected |

# Command Reference: MQTT

## MQTT Configuration +MQTTC

### Description

This command is used to read or set the MQTT configuration.

### Command Syntax

|  |  |
| --- | --- |
| Command | Description |
| AT+MQTTC[=<param\_id>] | Read configuration |
| AT+MQTTC=<param\_id>,<param\_val> | Set configuration |

### Supported Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name & Default Value | Type | Description |
| 1 | <BROKER\_ADDR>: “” | String | Broker domain name or IPv4 address |
| 2 | <BROKER\_PORT>: 8883 | Integer | Broker listening TCP port |
| 3 | <CLIENT\_ID >: “<Device ID>” | String | MQTT Client ID |
| 4 | <USERNAME>: “” | String | Username |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +MQTTC:<param\_id>,<param\_val> | Read response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

## MQTT Connection +MQTTCONN

### Description

This is used to connect to an MQTT broker or request current connection status.

### Command Syntax

AT+MQTTCONN[=<CLEAN>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <CLEAN> | Integer | **Placeholder for future release. Ignored in this release.**  Clean session flag:  0 Reuse existing session if possible  **(Placeholder for future release)** 1 Request clean session |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax

+MQTTCONNACK:<CONNACK\_FLAGS>,<RETURN\_CODE>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <CONNACK\_FLAGS> | Integer | Connect Acknowledge Flags |
| <RETURN\_CODE> | Integer | Connect Return code |

### AEC Syntax

+MQTTCONN:<CONN\_STATE>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <CONN\_STATE> | Integer | MQTT connected state:  0 Not connected 1 Connected |

### Examples:

|  |  |
| --- | --- |
| AT+MQTTCONN=1 | Connect to broker |
| OK | Command completed |
| +MQTTCONNACK:0,0 | Connection acknowledgement |
| +MQTTCONN:1 | Connected |

## MQTT Subscribe +MQTTSUB

### Description

This is used to subscribe to an MQTT topic.

### Command Syntax

AT+MQTTSUB=<TOPIC\_NAME>,<MAX\_QOS>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <TOPIC\_NAME> | String | Name of topic to subscribe to |
| <MAX\_QOS> | Integer | Max QoS (Valid values are 0, 1 and 2) |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax

+MQTTSUB:<REASON\_CODE>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <REASON\_CODE> | Integer | Result of unsubscribe request:   1. Request succeeded 2. Request failed |

### Examples:

|  |  |
| --- | --- |
| AT+MQTTSUB="Topic/name" | Subscribe to a topic |
| OK | Command completed |
| +MQTTSUB:0 | Subscription successful |

## MQTT Unsubscribe +MQTTUNSUB

### Description

This is used to unsubscribe from an MQTT topic.

### Command Syntax

AT+MQTTUNSUB=<TOPIC\_NAME>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <TOPIC\_NAME> | String | Name of topic to unsubscribe from |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax

+MQTTUNSUB:<RESULT>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <RESULT> | Integer | Result of unsubscribe request:   1. Request failed 2. Request succeeded |

### Examples:

|  |  |
| --- | --- |
| AT+MQTTUNSUB="Topic/name" | Unsubscribe from topic |
| OK | Command completed |
| +MQTTUNSUB:1 | Unsubscribe was successful |

## MQTT Publish +MQTTPUB

### Description

This is used to publish a message.

### Command Syntax

AT+MQTTPUB=<DUP>,<QOS>,<RETAIN>,<TOPIC\_NAME>,<TOPIC\_PAYLOAD>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <DUP> | Integer | **Placeholder for future release. Always set to 0 in this release.**  Duplicate flag:   1. First attempt to send message 2. Subsequent attempt to send message |
| <QOS> | Integer | QoS:   1. At most once 2. At least once 3. Exactly once |
| <RETAIN> | Integer | Retain flag:   1. Do not retain message on server 2. Retain message on server |
| <TOPIC\_NAME> | String | Name of topic to send message to |
| <TOPIC\_PAYLOAD> | String | Content of message to send |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax (QoS=1)

+MQTTPUBACC

### AEC Syntax (QoS=2)

+MQTTPUBCOMP

### AEC Syntax (Error)

+MQTTPUBERR

### Examples:

|  |  |
| --- | --- |
| AT+MQTTPUB=0,1,0,"Topic/name","123456" | Publish to a topic, QoS is 1 |
| OK | Command completed |
| +MQTTPUBACC | Publish acknowledged |

## MQTT Disconnect +MQTTDISCONN

### Description

This is used to disconnect from a broker.

<REASON\_CODE> parameters is for MQTT V5 only.

### Command Syntax

AT+MQTTDISCONN[=<REASON\_CODE>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <REASON\_CODE> | Integer | Reason code describing reason for disconnecting. |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax

+MQTTCONN:<CONN\_STATE>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <CONN\_STATE> | Integer | MQTT connected state:  0 Not connected 1 Connected |

### Examples:

|  |  |
| --- | --- |
| AT+MQTTDISCONN | Disconnect |
| OK | Command completed |
| +MQTTCONN:0 | Not connected |

# Command Reference: Network Stack

## Create Socket

### Description

This command is used to open a new socket.

### Command Syntax

AT+SOCKO=<PROTOCOL>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <PROTOCOL> | Integer | The protocol to use   |  |  | | --- | --- | | 1 | UDP | | 2 | TCP | |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +SOCKO:<SOCK\_ID> | Socket open response with new ID |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | ID of the socket opened |

### Examples:

(TCP):

|  |  |
| --- | --- |
| AT+SOCKO=2 | Open a new listening socket, TCP |
| +SOCKO:1 | Socket opened, ID is 1 |
| OK | Command completed |

(UDP):

|  |  |
| --- | --- |
| AT+SOCKO=1 | Open a new listening socket, UDP |
| +SOCKO:5 | Socket opened, ID is 5 |
| OK | Command completed |

## Close Socket

### Description

This command is used to close a socket.

The AEC signals that the socket was closed by the remote address or due to a failure.

Note: Receiving the AEC indicates a shutdown of the socket at the remote end, the socket will still exist locally, AT+SOCKCL must still be sent to remove the socket locally.

### Command Syntax

AT+SOCKCL=<SOCK\_ID>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | ID of the socket to close |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax

+SOCKCL:<SOCK\_ID>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | The socket ID which has closed |

### Examples:

|  |  |
| --- | --- |
| AT+SOCKCL=2 | Close socket ID 2 |
| OK | Command completed |
|  |  |
| +SOCKCL:4 | Socket ID 4 was closed remotely |
| AT+SOCKCL=4 | Close socket ID 2 locally |
| OK | Command completed |

## Bind a Listening Socket

### Description

This command is used to bind a socket to a local port.

A locally bound socket allows network services to be implemented. Any remote device may send data to this socket and the DCE will issue a data ready indication. For TCP sockets a connection must have been established prior to sending and receiving data.

### Command Syntax

AT+SOCKBL=<SOCK\_ID>,<LCL\_PORT>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | ID of the socket to bind |
| <LCL\_PORT> | Integer | The port number to use |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax (TCP Connection Established)

:<SOCK\_ID>,<LCL\_ADDR>,<LCL\_PORT>,<RMT\_ADDR>,<RMT\_PORT>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | Integer | | The ID of the passive listening socket which received connection | |
| <LCL\_ADDR> | String | | The address of the local end of the connection | |
| <LCL\_PORT> | Integer | | The port number of the local end of the connection | |
| <RMT\_ADDR> | String | | The address of the remote end of the connection | |
| <RMT\_PORT> | Integer | | The port number of the remote end of the connection | |

### Examples:

(TCP):

|  |  |
| --- | --- |
| AT+SOCKBL=2,6000 | Bind a listening socket, port 6000 |
| OK | Command completed |
| : |  |
| +SOCKIND:4,"192.168.0.100",6000,"192.168.0.1",12345 | New incoming TCP connection creating socket ID is 4 |

(UDP):

|  |  |
| --- | --- |
| AT+SOCKBL=1,6000 | Bind a listening socket, port 6000 |
| OK | Command completed |

## Bind a Remote Socket

### Description

This command is used to bind a socket to a remote address.

### Command Syntax

AT+SOCKBR=<SOCK\_ID>,<RMT\_ADDR>,<RMT\_PORT>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | ID of the socket to bind |
| <RMT\_ADDR> | String | The address of the remote device |
| <RMT\_PORT> | Integer | The port number on the remote device |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax (TCP Bind Success)

:<SOCK\_ID>,<LCL\_ADDR>,<LCL\_PORT>,<RMT\_ADDR>,<RMT\_PORT>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | Integer | | The ID of the bound socket | |
| <LCL\_ADDR> | String | | The address of the local end of the connection | |
| <LCL\_PORT> | Integer | | The port number of the local end of the connection | |
| <RMT\_ADDR> | String | | The address of the remote end of the connection | |
| <RMT\_PORT> | Integer | | The port number of the remote end of the connection | |

### AEC Syntax (TCP Bind Failed)

:<SOCK\_ID>,<STATUS\_CODE>[,<STATUS\_MSG>]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | Integer | | The ID of the bound socket | |
| <STATUS\_CODE> | Integer | | Numeric status code, see [Status Response Codes](#_Status_Response_Codes) | |
| <STATUS\_MSG> | String | | Descriptive text detailing the error | |

### Examples:

(UDP):

|  |  |
| --- | --- |
| AT+SOCKBR=1,"192.168.0.1",6000 | Bind a connection to remote host, port 6000 |
| OK | Command completed |
|  |  |

(TCP):

|  |  |
| --- | --- |
| AT+SOCKBR=4,"192.168.0.1",9000 | Bind a connection to remote host, port 9000 |
| OK | Command completed |
| +SOCKIND:4,"192.168.0.100",12345,"192.168.0.1",9000 | Connection established |

## Bind a Multicast Socket ()

### Description

These commands are used to bind a UDP socket to a multicast group.

### Command Syntax

AT+SOCKBM=<SOCK\_ID>,<MCAST\_ADDR>,<MCAST\_PORT>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | The socket ID to change |
| <MCAST\_ADDR> | String | The address of the multicast group |
| <MCAST\_PORT> | Integer | The port of the multicast group |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### Examples:

|  |  |
| --- | --- |
| AT+SOCKBM=7,"224.0.0.1",6000 | Add socket to multicast group |
| OK | Command completed |

## Upgrade Socket to TLS

## Description

This command is used to enable TLS on a socket.

### Command Syntax

AT+SOCKTLS=<SOCK\_ID>,<TLS\_CONF\_IDX>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | ID of the socket |
| <TLS\_CONF\_IDX> | Integer | TLS certificate configuration index to use (see +TLSC) |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +SOCKTLS:<SOCK\_ID> | Socket information |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | Socket ID |

### AEC Syntax (TLS Succeed)

:<SOCK\_ID>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | Integer | | The ID of the bound socket | |

### AEC Syntax (TLS Failed)

:<SOCK\_ID>,<STATUS\_CODE>[,<STATUS\_MSG>]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | Integer | | The ID of the bound socket | |
| <STATUS\_CODE> | Integer | | Numeric status code, see [Status Response Codes](#_Status_Response_Codes) | |
| <STATUS\_MSG> | String | | Descriptive text detailing the error | |

### Examples:

|  |  |
| --- | --- |
| AT+SOCKTLS=1 | Command: list all open sockets |
| OK | Command completed successfully |
| +SOCKTLS:1 | TCP socket completed TLS negotiation |
|  |  |

## List Current Sockets

### Description

This command is used to present a list of the DCE’s open sockets/connections.

### Command Syntax

AT+SOCKLST[=<SOCK\_ID>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | ID of the socket |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +SOCKLST:<SOCK\_ID>,<PROTOCOL>,<RMT\_ADDR>,<RMT\_PORT>,<LCL\_PORT> | Socket information |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | | Integer | | Socket ID |
| <PROTOCOL> | Integer | | The protocol in use on the socket   |  |  | | --- | --- | | 1 | UDP | | 2 | TCP | | |
| <RMT\_ADDR> | String | | The destination address associated with the socket (if any) | |
| <RMT\_PORT> | Integer | | The destination port associated with the socket (if any) | |
| <LCL\_PORT> | | Integer | | The local port associated with this socket |

### Examples:

|  |  |
| --- | --- |
| AT+SOCKLST | Command: list all open sockets |
| +SOCKLST:1,2,"",0,6000 | TCP listening socket on port 6000 |
| +SOCKLST:2,2,"192,168.0.1",4000,12345 | TCP connection with 192.168.0.1:4000 local port 12345 |
| +SOCKLST:3,1,"192.168.0.2",4000,12345 | UDP connection with 192.168.0.1:4000 local port 12345 |
| OK | Command completed successfully |
|  |  |

## Send Data to Network

### Description

This command is used to send data over a socket.

For TCP sockets the socket must be connected to a remote address/port.

### Command Syntax

AT+SOCKWR=<SOCK\_ID>,<LENGTH>[,<DATA>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | The socket ID to transmit the data through |
| <LENGTH> | Integer | The length of the data to send (1 – 1500 bytes) |
| <DATA> | String | The data to send in either ASCII or hexadecimal string format. If omitted the DCE will enter raw binary mode and will remain in that mode until the specified length of binary data has been received from the DTE. |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### Examples:

Terminal mode:

|  |  |
| --- | --- |
| AT+SOCKWR=1,7,"hello2u" | Send hello2u to 192.168.0.1:6000 via socket ID 1 |
| OK |  |
| AT+SOCKWR=1,7,[68656c6c6f3275] | Send data given in hexadecimal string format |
| OK | 7 bytes successfully queued for transmission |

Non-Terminal mode, length known:

|  |  |
| --- | --- |
| AT+SOCKWR=1,7 | Prepare to send 7 bytes given in raw binary format |
| #<7\_bytes\_of\_binary\_data\_from\_DTE> | ‘#’ followed by 7 bytes of data |
| OK | Command completed successfully |

Non-Terminal mode, length unknown:

|  |  |
| --- | --- |
| AT+SOCKWR=1,0 | Prepare to send an unknown number of bytes (non-terminal mode) |
| #One fine day+++ | ‘#’ followed by 12 bytes of data |
| OK | Command completed successfully |

## Send Data to Specific Address (UDP only)

### Description

This command is used to send data to an arbitrary destination using the connectionless UDP protocol.

### Command Syntax

AT+SOCKWRTO=<SOCK\_ID>,<RMT\_ADDR>,<RMT\_PORT>,<LENGTH>[,<DATA>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | The socket ID to transmit the data through |
| <RMT\_ADDR> | String | The destination address |
| <RMT\_PORT> | Integer | The destination port |
| <LENGTH> | Integer | The length of the data to send (1 – 1500 bytes) |
| <DATA> | String | The data to send in either ASCII or hexadecimal string format. If omitted the DCE will enter raw binary mode and will remain in that mode until the specified length of binary data has been input. |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### Examples:

Terminal mode:

|  |  |
| --- | --- |
| AT+SOCKWRTO=1,"192.168.0.1",6000,7,"hello2u" | Send hello2u to 192.168.0.1:6000 via socket ID 1 |
| OK |  |
| AT+SOCKWRTO=1,"192.167.0.1",6000,7,[68656c6c6f3275] | Send data given in hexadecimal string format |
| OK | 7 bytes successfully queued for transmission |

Non-Terminal mode, length known:

|  |  |
| --- | --- |
| AT+SOCKWRTO=1,"192.168.0.1",6000,7 | Prepare to send 7 bytes given in raw binary format |
| #<7\_bytes\_of\_binary\_data\_from\_DTE> | ‘#’ followed by 7 bytes of data |
| OK | Command completed successfully |

Non-Terminal mode, length unknown:

|  |  |
| --- | --- |
| AT+SOCKWRTO=1,"192.168.0.1",6000,0 | Prepare to send an unknown number of bytes (non-terminal mode) |
| #One fine day+++ | ‘#’ followed by 12 bytes of data |
| OK | Command completed successfully |

## Receive Data from Network , and

### Description

Two AECs present notification of data received by the DCE:

* indicates UDP data has been received.
* indicates TCP data has been received.

The DTE is responsible for retrieving the datagram/stream data via the command.

For TCP sockets the DCE will indicate, via , the number of bytes of data which are currently available for reading via the command. The DCE may issue multiple AECs as data is received. When requesting data via the command the DTE may receive less data than request, the number of bytes provided by the DCE will be declared in the response before the data is presented. The DTE may request less data than that declared by the AEC as being available, the remaining data will be available for subsequent reading.

For UDP sockets the DCE will indicate, via , the number of bytes of data which were received in the oldest datagram received by the DCE. Only a single will be issued by the DCE even if subsequent UDP datagrams are received. Reading data from the UDP socket via the command will read and discard the current datagram, if less data is requested than was indicated by the AEC the remaining unread data in the datagram will be discarded.

### AEC Syntax (TCP)

:<SOCK\_ID>,<LENGTH>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <SOCK\_ID> | Integer | The socket ID on which there is data to read |
| <LENGTH> | Integer | The number of bytes received |

### AEC Syntax (UDP)

:<SOCK\_ID>,<RMT\_ADDR>,<RMT\_PORT>,<LENGTH>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | | Integer | | The socket ID on which there is data to read |
| <RMT\_ADDR> | String | | The sender’s IP address | |
| <RMT\_PORT> | Integer | | The sender’s port number | |
| <LENGTH> | | Integer | | The number of bytes received |

### Command Syntax ( - read socket data)

AT=<SOCK\_ID>,<OUTPUT\_MODE>,<LENGTH>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | | Integer | | The socket number to read data from |
| <OUTPUT\_MODE > | Integer | | The format the DTE wishes to receive the data:   |  |  | | --- | --- | | 1 | ASCII or hex String | | 2 | Binary | | |
| <LENGTH> | | Integer | | The number of bytes the DTE wishes to read |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +SOCKRD:<SOCK\_ID>,<LENGTH>,<DATA> | Socket read |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter Name | | Type | | Description |
| <SOCK\_ID> | | Integer | | The socket ID the data is being retrieved from |
| <LENGTH> | Integer | | The number of bytes being retrieved | |
| <DATA> | | Variable | | The data in the format requested, either ASCII/hex string or binary |

### Examples:

|  |  |
| --- | --- |
| +SOCKRXT:1,7 | AEC indicating TCP data available on socket ID 1, length 7 bytes |
| AT+SOCKRD=1,1,999 | Command – read 999 bytes, ASCII/hex string format, socket ID 1 |
| +SOCKRD:1,7,[466F6FC2A9626172] | Response – 7 bytes. Data is Foo©Bar |
| OK | Command completed |
|  |  |
| +SOCKRXU:4,"1.2.3.4",6000,7 | AEC: UDP data from 1.2.3.4:6000 available on socket 4, 7 bytes |
| AT+SOCKRD=4,2,999 | Command – read 999 bytes, binary format, socket ID 4 |
| +SOCKRD:1,7,  #<7\_bytes\_binary> | Response – 7 bytes from socket 4 to be presented in binary  Response – 8 bytes binary data (includes leading ‘#’ marker) |
| OK | Command completed |

## TLS Configuration +TLSC

### Description

This command is used to read or set the TLS configuration. Multiple TLS configurations are possible, each configuration has an index starting with 1 for the first configuration. The index must be specified when using this command.

### Command Syntax

|  |  |
| --- | --- |
| Command | Description |
| AT+TLSC=<conf\_idx>[,<param\_id>] | Read configuration |
| AT+TLSC=<conf\_idx>,<param\_id>,<param\_val> | Set configuration |

### Supported Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name & Default Value | Type | Description |
| 1 | <CA\_CERT\_NAME>: “” | String | CA certificate name |
| 2 | <CERT\_NAME>: “” | String | Certificate name |
| 3 | <PRI\_KEY\_NAME>: “” | String | Private key name |
| 4 | <PRI\_KEY\_PASSWORD>: “” | String | Private key password |
| 5 | <SERVER\_NAME>: “” | String | Server name |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +TLSC:<param\_id>,<param\_val> | Read response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

Note: The user needs to ensure that the certificates are a part of the image. Please check “Appendix A: Establishing a secured connection using AT Commands” for more details.

## DNS Name Resolution

### Description

This command is used to resolve domain names via DNS.

### Command Syntax

AT=<TYPE><DOMAIN\_NAME>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <TYPE> | Integer | Type of record:   1. A 2. AAAA |
| <DOMAIN\_NAME> | String | Domain name to resolve |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax (Success)

:<TYPE>,<DOMAIN\_NAME>[,<QUERY\_RESP>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <TYPE> | Integer | Type of record:   1. A Record 2. AAAA Record |
| <DOMAIN\_NAME> | String | Original domain name requested |
| <QUERY\_RESP> | String | Query response received |

### AEC Syntax (Failure)

:<ERROR\_CODE>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <ERROR\_CODE> | Integer | Error response |

### Examples:

|  |  |
| --- | --- |
| AT+DNSRESOLV=1,"www.example.com" | Request A record for www.example.com |
| OK | Command completed |
| +DNSRESOLV:1,"www.example.com","192.168.0.1" | A record response for www.example.com |

## Ping Link Check

### Description

This command sends a ping (ICMP Echo Request) to the target address.

### Command Syntax

AT=<TARGET\_ADDR>,[<IP\_PROTOCOL\_VERSION>]

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <TARGET\_ADDR> | String | IP address or domain name of target |
| <IP\_PROTOCOL\_VERSION> | Integer | IP protocol version:   1. IPv4 2. IPv6 |

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

### AEC Syntax

:<IP\_ADDRESS>,<RTT>

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <IP\_ADDRESS> | String | IP address of the target responding |
| <RTT> | Integer | Round trip time (in milliseconds) |

### Examples:

|  |  |
| --- | --- |
| AT+PING="www.example.com" | Ping the host www.example.com |
| OK | Command completed |

## Request Manufacturer Identification

### Description

This command retrieves the manufacturers information from the DCE.

The format of the returned data is platform dependent, the information returned for a WINC-based platform is as given below:

### Command Syntax

AT+GMI

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +GMI:<MAN\_ID> | Information response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <MAN\_ID> | String | Manufacturers ID/Name |

### Examples:

|  |  |
| --- | --- |
| AT+GMI | Query the manufacturers ID/name |
| +GMI:"Microchip" | Manufacturers information |
| OK | Command completed |

## Request Model Identification

### Description

This command retrieves the model information from the DCE.

The format of the returned data is platform dependent, the information returned for a WINC-based platform is as given below:

### Command Syntax

AT+GMM

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +GMM:<MODEL\_ID> | Information response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <MODEL\_ID> | String | Model Information |

### Examples:

|  |  |
| --- | --- |
| AT+GMM | Query the model information |
| +GMM:"WINC1500" | Model information |
| OK | Command completed |

## Request Revision Identification +GMR

### Description

This command retrieves version information from the DCE.

The format of the returned data is platform dependent, the information returned for a WINC-based platform is as given below:

### Command Syntax

AT+GMR

### Response Syntax

|  |  |
| --- | --- |
| Response | Description |
| +VER:<CHIP\_ID>,<FW\_VER>,<DRV\_VER>,<RB\_FW> | Information response |
| OK | Successful response |
| ERROR:<ERROR\_CODE> | Error response |

|  |  |  |
| --- | --- | --- |
| Parameter Name | Type | Description |
| <CHIP\_ID> | String | ID / version of the Wi-Fi chip |
| <FW\_VER> | String | The version of the firmware (software running on Wi-Fi IC) |
| <DRV\_VER> | String | Any additional versioning info (e.g. of driver to application layer) |
| <RB\_FW> | String | Rollback firmware version |

### Examples:

|  |  |
| --- | --- |
| AT+GMR | Query the revision information |
| +GMR:"15x0","19.6.1","19.6.1","19.5.3" | Version information |
| OK | Command completed |

## Appendix A: Establishing a secured connection using AT Commands

In case the user wants to establish a connection with a Server, one needs to ensure that the certificate needed for verification of the server during the TLS negotiations is present in the *PIC32MZW1\_AnyCloud\firmware\src\cert\_header.h* file in a C Array format. The certificate needs to be in DER format.

Also, one needs to ensure that there is an entry for each such certificate that the user is adding in the below format in the file *PIC32MZW1\_AnyCloud\firmware\src\cert.h:*

*INSERT\_CERT\_DER\_DATA(app\_client\_cert\_der\_mosquitto\_org, sizeof\_app\_client\_cert\_der\_mosquitto\_org)*

When establishing a secured TCP connection:

1. Once the Wifi connection has been established, create an entry for the TLS Certificates:
   1. *AT+TLSC=1,2,"app\_client\_cert\_der\_mosquitto\_org"*
2. Establish the normal TCP connection with the server first
   1. *AT+SOCKO=2*
   2. *AT+SOCKBR=1,"5.196.95.208",8883*
3. Once the connection has been established, to start the secured connection
   1. *AT+SOCKTLS=1,1*
4. Check more details in the section +TLSC and +SOCKTLS on the usage of respective AT Commands.

***Note: In case the user is adding a new certificate, the code will need to be recompiled. In a future release, the addition of these certificates would be done dynamically using AT Commands, eliminating any need for recompilation of the code.***