WISM SLIP - User Guide

Contents

1. Introduction

1.1 Overview

This document provides a User Guide for the Ezurio WISM module incorporating the SLIP interface. The module provides a UART interface to the host device which is used to configure the module operation and then transfer data between the host and the communication end-point via the wireless interface. The Wireless Lan module is the first in a family of Universal Wireless (UW) modules from Ezurio which aim to provide the same control and data interface regardless of the wireless interface being used (Bluetooth, Wireless Lan etc). For this reason the module must contain all of the necessary elements to allow the transport of data over the selected wireless interface. In the case of Wireless Lan this means that the module contains the TCP/IP stack and associated protocols in addition to the Wireless Lan driver. The module takes responsibility for the packetisation of data from the host prior to transmission and the serialisation of received packets.

In addition to the above features, Ezurio have implemented a SLIP interface to the module which allows the transfer of 802.3 packets via the UART. In this mode the internal TCP/IP stack of the module is disabled and the 802.3 packets are directly interfaced to the top of the Wireless Lan driver. In addition, because the interface to the module is now packet based, a packet framing protocol is applied to the data as it is transferred across the UART. To simplify the interface, the same packet framing protocol is applied to both control and data information when operating in the SLIP mode. Also, to simplify the interface, the Universal Wireless scripting language method of control has been replaced by a fixed set of Wireless Lan specific commands that will be used to configure the module in SLIP mode.

1.2 References

1.3 Glossary of Terms

AP	Access Point
MSDU	MAC Service Data Unit
RAT	Radio Access Technology
SLIP	Serial Line Internet Protocol
UW	Universal Wireless

2. SLIP Mode Operation

The WISM SLIP module provides an interface that allows 802.3 packets to be passed to and from the module. In this mode the TCP/IP stack is bypassed and the 802.3 packets are interfaced directly to the Wireless Lan device driver. Due to the packet nature of the data flowing across the UART a framing protocol is needed to allow packet start and end to be delimited. For simplicity, the SLIP protocol has been selected for this purpose and is described below.

Given that the data flowing across the interface uses the SLIP protocol, it makes sense for the commands and responses to use the same protocol.

2.1 SLIP Mode Command and Data Framing

2.1.1 SLIP Overview

SLIP is a simple framing standard for the transmission of IP datagrams over a serial interface (RFC 1055). SLIP defines an END character (0xC0) and an ESC character (0xDB). The END character is prepended and appended to the packet to be transferred. Pre-pending the packet with the END character allows the interface to re-synchronise on the next frame if noise corrupts a packet.

Given that the END character has special meaning, it is necessary to ensure that the END character does not appear within the packet payload. To avoid this, any occurrence of the END character within the payload is escaped (replaced) by two characters, ESC and 0xDC. If the ESC character is encountered within the packet, then it too is escaped (replaced) with two characters ESC and 0xDD.

2.1.2 Frame Format

Data and command information transferred across the UART interface is encapsulated in frames using the SLIP frame structure shown in Figure 1.

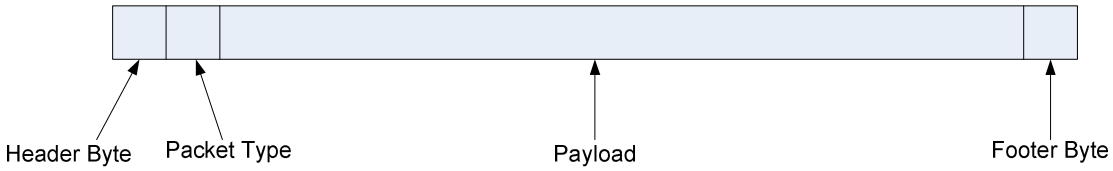


Figure 1 UART Frame Format

The fields are as follows:

Header Byte: SLIP frame END byte, set to '0xC0'.

Packet Type: Describes the contents of the payload. This byte is coded as follows:

Byte Value	Payload Contents
0x01	Command frame. Frame contains a command / response string.
0x02	Data frame. Frame contains an 802.3 frame.

Payload: The format of the payload depends on the Packet Type description. The payload must have all occurrences of the END and ESC character removed as described in Section 2.1.1.

Footer Byte: SLIP frame END byte, set to '0xC0'

2.1.3 Example Frame

The following example is a control frame containing a UW command:

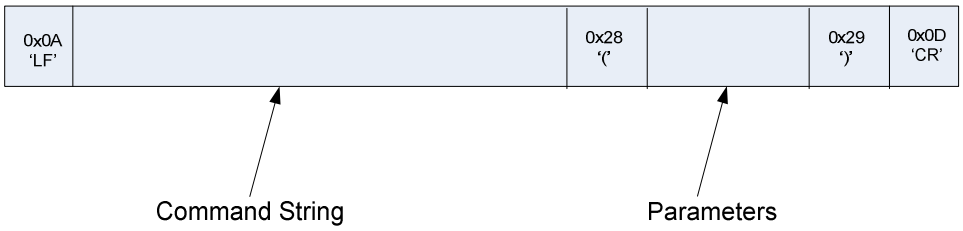
0xC0	0x01	0x43 'C'	0x48 'H'	0x41 'A'	0x4E 'N'	0x4E 'N'	0x45 'E'	0x4C 'L'	0x28 '('	0x36 '6'	0x29 ')'	0xC0

2.2 Command Transactions

Commands can be sent to the module at any time. All commands are acknowledged by a response from the module. The host must not send the next command until the response for the current command has been received, however, data frames can be sent to or from the module while a command response is awaited. Responses are sent when the requested command has been executed. In practice this means that some commands (SEARCH for example) may take a significant time before the response is returned.

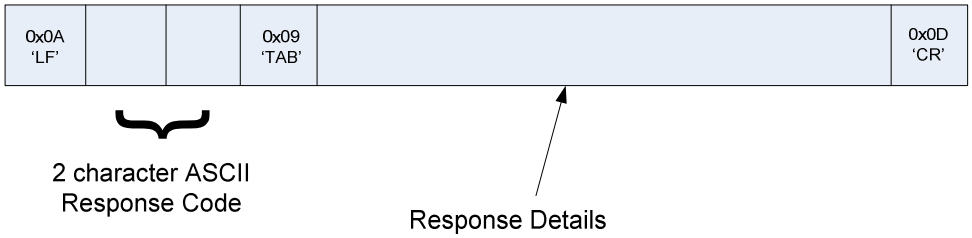
2.2.1 Command Format

Commands are ASCII strings and have the following generic format:



2.2.2 Response Format

Responses are ONLY sent in response to commands from the host. Responses are in ASCII and are sent as the payload in a SLIP frame. All responses have the following generic format:



The response codes and associated details are described in Section 4.

2.3 UART Signalling

UART control signals are used at the host interface to indicate the status of the module, as outlined below:

Signal Name	Module Input / Output	Function

CTS (Clear to Send)	Input	Flow control. Asserted when the host is able to accept data.
RTS (Request to Send)	Output	Flow control. Asserted when the module is able to accept data.
DCD (Data Carrier Detect)	Output	Link Status. Asserted when the Wireless Lan link has been established and the module is able to receive 802.3 frames. De-asserted prior to connection or when the link is lost. If the link is lost the module will automatically attempt to re-connect.
DTR	Output	Used to signal current power saving state in powersave(2) mode (see section 2.4.3)
DSR	Input	Used to control transitions into and out of powersave(2) mode (see section 2.4.3)
RI	Output	For future use

2.4 Power Management

The WISM SLIP module implements a number of power saving modes that significantly reduce the power consumption of the module when no data is being transferred. The majority of these modes rely on IEEE power save operation which is described below.

2.4.1 IEEE Power Save Mode

IEEE power save mode is a mode in which the WLAN chipset enters a low power sleep mode when there is no data to transfer in either direction. As the operation of this function requires the regular receipt of beacons from an access point, it only functions if the chipset is attached to an AP. In this mode, the chipset wakes on a regular basis to maintain contact with the access point and to request the transfer of data (if user data is ready to send) or receive data (if data has been queued at the access point). There are two parameters which govern the operation of this power save mode:

- DTIM period. This parameter sets the frequency at which the access point sends multi-point or broadcast frames. The DTIM counter is decremented and sent on each beacon transmission. When the DTIM counter reaches zero, all broadcast or multicast frames that have been queued in the AP since the last transmission are sent. If the module is required to receive transmissions of this type then it must be awake in time to receive the transmissions.
- Listen interval. The listen interval specifies the frequency (in multiples of the beacon period) at which the module will wake to receive or send stored messages. When the module wakes, it listens to the beacon transmission from the AP to find out if there are stored messages at the AP. If there are stored messages, then the module transmits a Poll request to the AP to trigger the transmission of the messages. If the listen interval is longer than the DTIM period, then the listen interval is ignored and the DTIM period is used to govern wakeup intervals.

If there is no data to be transferred when the chipset wakes up – it returns to the power save state immediately. If there is data in the queue in either direction, then the chipset remains awake until all of the data has been transferred.

2.4.2 Power Save Mode 1

Assuming that the module is attached to an AP, power save mode 1 sets the WLAN chipset into IEEE power save mode. The WISM CPU remains awake for the whole time and hence is able to receive commands and data via the UART. Data packets from the host are held in the module until the next transmission opportunity. This will normally be at the next regular DTIM wakeup. The module always listens to broadcast and multicast frames based on the DTIM period from the access point.

If the module is not attached to an AP, it remains in the fully powered state until a successful attach is completed at which point it automatically enters the power save mode.

The module power consumption in this mode should be less than 40mA.

2.4.3 Power Save Mode 2

In power save mode 2, the WLAN chipset operates in IEEE power save mode and the WISM CPU is put into standby mode (main oscillator disabled, ARM core disabled, peripheral clocks running from slow clock). The module power consumption in this mode should be less than 10mA. Due to its reliance on IEEE powersave mode, power save mode 2 can only be used when the WISM module is attached to an AP.

Once the powersave command has been sent to select this mode, the DSR / DTR signals are used to manage the transitions into and out of the power save mode. When DSR is de-asserted by the WISM host the module enters the power save mode and DTR is de-asserted. Once the WISM CPU is asleep, it will be configured to be woken by two events:

- DSR assertion from WISM host. Assertion of DSR will generate a WISM CPU interrupt that will trigger the wake-up process. When the WISM CPU is awake, and ready to receive data or commands from the host, it will assert DTR. In this state, the WLAN chipset continues to operate in IEEE power save mode, and any data sent by the WISM host will be buffered until the WLAN chipset next wakes up.
- PS awake event from the WLAN chipset. This event generates a WISM CPU interrupt when the appropriate WISM CPU wakeup conditions The default is that the WISM CPU wakeup will be generated on receipt of a unicast or multicast message.

Once awake the WISM CPU remains fully operational (which is now equivalent to power save mode 1) until the host either de-asserts DSR or issues a powersave command to change the power save mode.

3. Command Catalogue

3.1 UARTMODIFY (baud, length, parity, stop, persist)

Synopsis: Used to modify the current UART settings. The UART parameters are changed immediately. The response to the UARTMODIFY command is sent 1 second later using the new UART parameters, allowing the host processor time to modify the parameters of the host UART to match.

Parameters:

- **baud**: The following baud rates are allowed: 9600, 19200, 38400, 57600, 115200 and 230400.
- **length**: Sets the length of the serial characters. 8 bits is the only allowed character length.
- **parity**: Sets the parity of the transmission. Allowed options are:

Parity	Parity parameter setting
None	0
Even	1
Odd	2

The current version of the software supports only parity set to None.

- **stop**: Sets the number of stop bits. Allowed values are 1 or 2. The current version of the software supports only a stop bit value of 1.
- **persist**: For future implementation – set to 0.

Responses: OK, Error

Example: UARTMODIFY(115200, 8, 0, 1, 0)

Sets the UART to 115.2kbaud, 8 bit characters, no parity, 1 stop bit and no retention over power on.

Default after Reset: 115.2kbaud, 8 bit characters, no parity, 1 start bit and 1 stop bit.

3.2 AUTHENTICATE (flag)

Synopsis: Selects the use of shared key authentication over the Wireless Lan connection to the access point. If shared key authentication is selected, then this must be combined with setting the WEP key using the KEY() command.

Parameters:

- flag: 0 indicates that shared key authentication is disabled (open system authentication), 1 indicates that shared key authentication is enabled.

Responses: OK, Error

Example: AUTHENTICATE (1) – switches authentication on.

Default after Reset: Shared key authentication is disabled.

3.3 SECURITY (flag)

Synopsis: Selects the encryption method to be applied to the Wireless Lan link. If WEP encryption is selected then this must be combined with setting the WEP key using the KEY() command.

Parameters:

- flag: Currently allowed settings are:

Security option	Flag parameter setting
None	0
WEP	1
Reserved	2

Responses: OK, Error

Example: SECURITY (1) – switches WEP encryption on.

Default after Reset: No encryption is enabled.

3.4 BSSTYPE (flag)

Synopsis: This command is not currently supported. Selects whether an ad-hoc (point to point) connection is to be established or an access point connection.

Parameters:

- flag: Allowed settings are:

BSS option	Flag parameter setting
Access Point	0
Ad-Hoc	1

Responses: OK, Error

Example: BSSTYPE(0) – Selects an access point connection.

Default after Reset: Access point connection.

3.5 ATTACH("name")

Synopsis: Establishes a connection with the named Wireless Lan network. When this command is issued, the module searches for the named network and, if found, the module authenticates and associates with the network. After successful completion of this command, DCD is asserted to indicate that the module is ready to pass traffic data over the Wireless Lan link.

Parameters:

- **name**: The string name of the network to be attached to. Maximum length of *name* is 32 characters.

Responses: OK, Error

Example: ATTACH("Ezurio_Network") – searches for and, if present, attaches to the Ezurio_Network.

Default after Reset: N/A

3.6 KEY("keystring")

Synopsis: Sets the key to be used by the WEP algorithm for encryption.

Parameters:

- keystring: String representing 10 or 26 hex digits providing either a 64 or 128 bit encryption key respectively.

Responses: OK, Error

Example: KEY("6C4B5D9EDD") – sets a 64 bit key.

Default after Reset: N/A

3.7 SEARCH()

Synopsis: Performs a search for all available networks.

Parameters: None

Responses: OK, Error, SEARCH_RESULTS

Example: SEARCH() – performs search for available networks

Default after Reset: N/A

3.8 CHANNEL(number)

Synopsis: This command is not currently supported. Selects the channel to be used for an ad-hoc connection. This command has no meaning in an access point connection (the channel is set by the AP).

Parameters:

- number: Valid channel number. This has the range 1-13 for Europe, 1-11 for US and 1-14 for Japan. The module does not check the channel number with geographical region, any channel number in the range 1-14 will be accepted.

Responses: OK, Error

Example: CHANNEL(6) – Sets the operating channel for the ad-hoc network to be 6.

Default after Reset: Channel 6 is selected.

3.9 MACADDRESS()

Synopsis: Requests that the module return the MAC address of the Wireless Lan device.

Parameters: None

Returns: Error, MACADDRESS

Example: MACADDRESS()

Default after Reset: N/A

3.10 DETACH()

Synopsis: Requests that the current connection is terminated. DCD is de-asserted when the disconnection has been completed.

Parameters: None

Returns: OK, Error

Example: DETACH() – disconnects the current connection. De-asserts DCD when the disconnection is complete.

Default after Reset: N/A

3.11 POWERSAVE(flag)

Synopsis: Switches the power saving mode of the module. Power saving modes are described in detail in section 2.4. For power saving modes that require the use of DSR (power save mode 2 for example), DSR must be asserted BEFORE the powersave command is issued. Failure to do this will result in a error code being raised.

Parameters:

- flag: Allowed settings are:

Power Save Mode	Flag parameter setting
Disabled	0
IEEE Only	1
IEEE and Processor	2

Responses: OK, Error

Example: POWERSAVE(1): Switches on IEEE power saving mode.

Default after Reset: Power saving is disabled.

3.12 GETSTATS()

Synopsis: This command is not currently supported. Returns the current Wireless Lan link statistics.

Parameters: None

Responses: Error, STATISTICS

Example: GETSTATS()

Default after Reset: N/A

3.13 GETRSSI()

Synopsis: Returns various RSSI values that will provide information on the quality of the current link.

Parameters: None

Returns: Error, RSSI

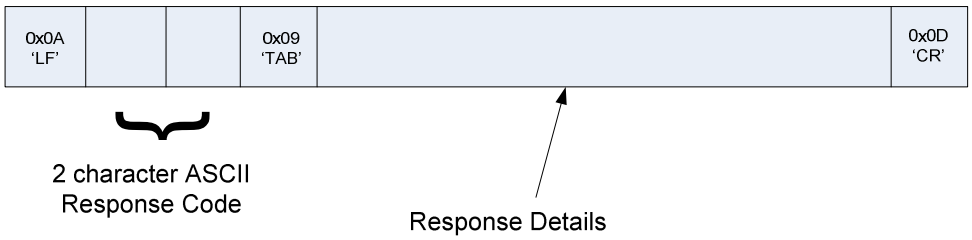
Example: GETRSSI()

Default after Reset: N/A

4. Response Catalogue

4.1 Response Format

Responses are ONLY sent in response to commands from the host. Responses are in ASCII and are sent as the payload in a SLIP frame. All responses have the following generic format:



For all responses other than ERROR and OK (for example the SEARCHRESULTS response), an OK response frame is always sent to indicate the completion of the response.

For example, a SEARCHRESULTS response where two access points have been discovered (NET-1 and NET-2) would consist of the following frames:

0x40A 'LF'	0x30 '0'	0x32 '2'	0x09 'TAB'	0x4E 'N'	0x45 'E'	0x54 'T'	0x2D '-'	0x31 '1'	0x2C ','	0x30 '0'	0x37 '7'	0x39 '9'	0x0D 'CR'

0x40A 'LF'	0x30 '0'	0x32 '2'	0x09 'TAB'	0x4E 'N'	0x45 'E'	0x54 'T'	0x2D '-'	0x32 '2'	0x2C ','	0x30 '0'	0x35 '5'	0x36 '6'	0x0D 'CR'

0x40A 'LF'	0x30 '0'	0x30 '0'	0x0D 'CR'

4.2 OK

Response Code: 00

Synopsis: Sent when no other response is required by the command. Indicates that the previous command has been executed successfully.

Details: None

Example:

0x40A 'LF'	0x30 '0'	0x30 '0'	0x0D 'CR'

4.3 ERROR code

Response Code: 01

Synopsis: Sent if the requested command has failed for some reason. The reason code indicates the precise details of the failure.

Details:

- code: Two digit error code. The following codes have been defined (more are likely to be defined over time):

Code	Meaning
1	A parameter passed in the command was out of range
2	An unexpected parameter was passed in the command
3	No networks have been found in the search
5	The command contained a syntax error
6	The WISM failed to attach to the supplied SSID
7	The command is unrecognized
10	An unspecified error has occurred
14	Incorrect mode
29	The WISM has attempted to access a NULL pointer while carrying out command
44	The UART configuration has failed
45	The DSR line status is incorrect for this command to be auctioned
46	Command option not currently implemented

Example:

No network found.

0x40A 'LF'	0x30 '0'	0x31 '1'	0x09 'TAB'	0x30 '0'	0x33 3'	0x0D 'CR'

4.4 SEARCHRESULTS results

Response Code: 02

Synopsis: Returns the list of BSSIDs and associated signal strengths for all Wireless Lan detected networks. The search results are sent when the search is completed. Given that this command may take a long time to complete, the module will return an empty SEARCHRESULTS response every 1 second indicating that the search is still in progress. If no access points are found, then the final empty SEARCHRESULTS response will be followed by the OK response and the command will be complete.

Details:

- results: The results are presented as comma separated tuples with the following format:

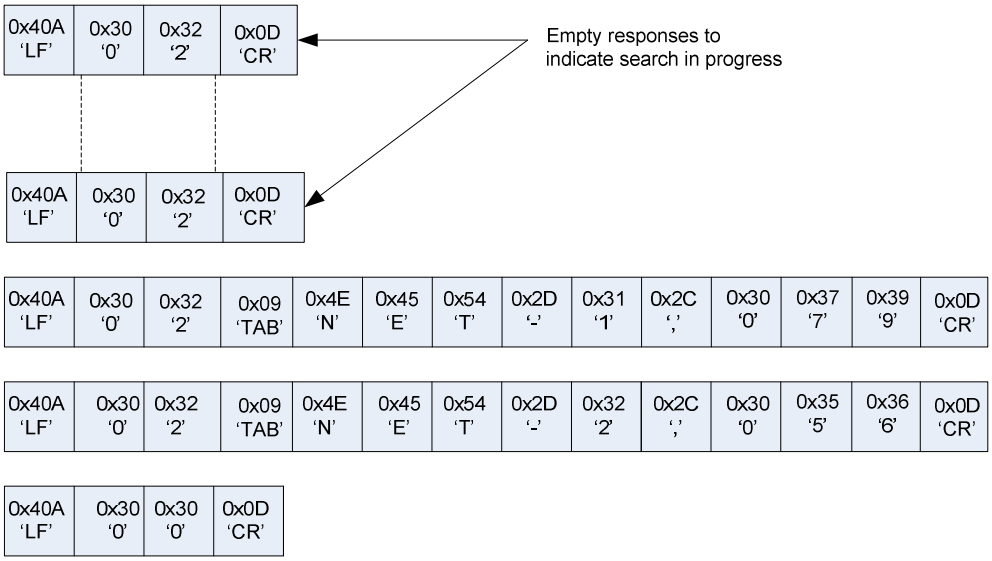
  SSIDname, RSSI

  SSIDname is the alphanumeric SSID name up to 32 characters in length.

  RSSI is the RSSI of the beacon transmission from that access point. RSSI is an 8 bit integer represented as 3 decimal digits (0 – 255). The value represents the absolute value of RSSI in dBm (the actual RSSI is always a negative number).

Example:

Two APs found with SSIDs of NET-1 and NET-2:



4.5 MACADDRESS address

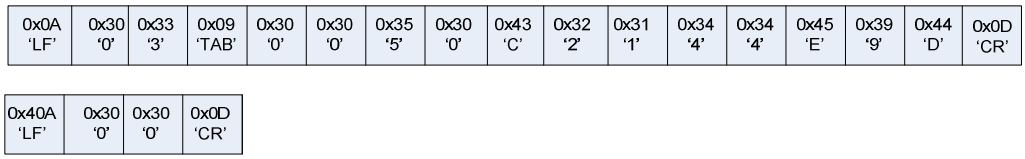
Response Code: 03

Synopsis: Returns the MAC address of the Wireless Lan module.

Details:

- address: MAC address sent as a 12 character string representing the 6 hex digit, MAC address e.g 0050C2144E9D.

Example:



4.6 STATISTICS stats

Response Code: 04

Synopsis: This command / response is not currently supported, the following is provided for information only. This response returns the current statistics from the Wireless Lan interface. The statistics indicate how many MSDUs (data packets presented at the MAC interface – not including any 802.11 management frames) succeeded, failed or re-tried.

Details:

- stats: The statistics are reported as a string representing a comma separated list of 32 bit numbers (e.g. 00000010, 04550203, 00AA0F5C, 10106DD5) showing the following parameters:

  Failed count – Number of MSDU transmission failures

  Retry count – Number of MSDUs transmitted after one or more re-transmissions

Multiple retry count – Number of MSDUs transmitted after at least one re-transmission.

Transmitted frame count – Number of successfully transmitted MSDUs

Example:

0x0A 'LF'	0x30 '0'	0x34 '4'	0x09 'TAB'	0x30 '0'	0x30 '0'	0x30 '0'	0x30 '0'	0x30 '0'	0x30 '0'	0x31 '1'	0x30 '0'	0x2C ','	0x30 '0'		0x44 'D'	0x35 '5'	0x0D 'CR'

0x40A 'LF'	0x30 '0'	0x30 '0'	0x0D 'CR'

4.7 RSSI values

Response code: 05

Synopsis: Returns various RSSI measurements characterising the current link. RSSI measurements reported through this command are made on AP beacon transmissions.

Details:

- values: The following RSSI values are reported as a string representing a comma separated list of 3 digit decimal numbers (e.g. 015, 110, 0016, 120) showing the following parameters:

  SNR – signal to noise ratio (dB) of most recent beacon received (always positive).

  Noise floor – Noise floor (absolute in dBm) of most recent beacon received (absolute value of an always negative number)

  AvgSNR – average SNR in received beacons (always positive)

  AvgNoiseFloor – average of the absolute noise floor (absolute value of an always negative number)

  The SNR value is the best statistic to use for signal strength display as it represents the quality of the signal presented to the demodulator. The SNR values range from ~70 close to the access point to 0 at the limit of range.

Example:

0x0A 'LF'	0x30 '0'	0x35 '5'	0x09 'TAB'	0x30 '0'	0x31 '1'	0x35 '5'	0x2C ','	0x31 '1'	0x31 '1'		0x32 '2'	0x30 '0'	0x0D 'CR'

0x40A 'LF'	0x30 '0'	0x30 '0'	0x0D 'CR'