AT+i ™

Programmer's Manual

Version 8.41

for iChip™ CO2128 and CO2144
with Firmware Version 807B21

August 2013

International:
Connect One Ltd.
20 Atir Yeda Street
Kfar Saba 44643, Israel
Tel: +972-9-766-0456
Fax: +972-9-766-0461
http://www.connectone.com

Pub. No. 20-5000-08_41, August 2013

Revision History 20-5000-08_41

Version	Date	Description
1.0	Nov. 1999	Original Release.
3.0	Nov. 2000	Updated with iChip LAN support.
4.0	January 2001	Updated with AT+i commands for listening sockets and email firmware update. Requires iChip/iChip LAN with Firmware IC602B01 and boot block BBIC0620 or higher.
5.0	February 2001	
6.0	March 2001	The sizes of the following iChip parameters have been increased: ISPn, USRN, PWD, SMTP, POP3, MBX, MPWD, FLS, TOA, REA, CCn, UMBX, UPOP, UMPW. New or enhanced AT+i commands: WWW, P#, WPWD, RPG. Applicable to Firmware revision IC603Bxx or higher.
6.1	June 2001	Miscellaneous Corrections.
6.3	July 2001	Additions for iChip Plus, Firmware revision IC604Bxx or higher.
6.4	Sep. 2001	Internal Reorganization.
6.5	Oct. 2001	Additions describing SerialNET mode, Firmware revision IC605Bxx or higher.
7.1	July 2002	Additions describing the full Web server, FTP and Telnet for Firmware revision Ix701B25 or higher.
7.2	Nov. 2002	Updates that cover additions and changes in Firmware revision 702P11 or higher for CO561AD-x and CO661AL-x iChip devices.
7.4	August 2003	Updates that cover additions and changes up to Firmware revision 704B12 for CO561AD-x and CO661AL-x iChip devices.
7.4a	October 2003	Updates that cover firmware revision up to 704B16.
7.4b	Dec. 2004	Updates that cover firmware revision up to 704P09.
7.6	September 2005	Updates that cover firmware revisions 705xxx and 706xxx or higher. Removes references to the CO561-x iChip devices, which are not supported beyond 704xxx.
7.6a	April 2006	Corrected a few typos
8.0a	June 2007	Updates that cover firmware revisions up to 800xxx.
8.14	September 2007	Extensive editing.
8.20	December 2007	Extensive editing.
8.21	January 2008	Miscellaneous corrections.
8.30	March 2008	Updates covering firmware additions up to 722B05
8.31	March 2008	iRouter mode modifications
8.32	December 2010	Updates covering firmware additions up to 708B21
8.40	July 2011	Updates covering firmware additions up to 807B21
8.41	August 2013	SPI section updated with diagrams

Contents

Figures

Tables

1 AT+i Command Set

Scope

This manual describes Connect One's AT+i™ interface standard, protocol, and syntax for the iChip CO2128 and CO2144.

AT+i Command Guidelines

AT+i commands are an extension to the basic AT command set. They are parsed and acted upon by iChip.

Paragraphs which are highlighted in GREY color refer to iChip in dial-up mode only.

iChip in dial-up mode only: When iChip is in COMMAND mode, basic AT commands and raw data (not prefixed by AT+i) are transparently transferred to the underlying modem Digital Communications Equipment (DCE), where they are serviced. When transferring data transparently to the DCE, the hardware flow control signals (CTS, RTS, DTR and DSR) are mirrored across the iChip, unless disabled by the FLW parameter. AT and AT+i commands may be issued intermittently. During an Internet session, when iChip is online, an AT command can be sent to the modem using the AT+iMCM command.

The ASCII ISO 646 character set (CCITT T.50 International Alphabet 5, American Standard Code for Information Interchange) is used for issuing commands and responses. Only the low-order 7 bits of each character are used for commands and parameters; the high-order bit is ignored. Uppercase characters are equivalent to lowercase ones.

AT+i Command Format

An AT+i command line is a string of characters sent from the host to the iChip while it is in command state. The command line has a prefix, body, and terminator. Each command must begin with the character sequence `AT+i` and terminated by a carriage return `<CR>`. Commands can be entered either in uppercase or lowercase.

iChip in dial-up mode only: Commands that do not begin with the `AT+i` prefix are transferred to the underlying DCE, where they are parsed and acted upon. DCE responses are transparently returned to the host.

The `AT+i` command body is restricted to printable ASCII characters (032–126). The command terminator is the ASCII `<CR>` character. The command line interpretation begins upon receipt of the carriage return character. An exception to this rule are the AT+iEMB, AT+iSSND, AT+iTBSN and AT+iFSND commands.

When ECHO is enabled, the `<CR>` character is echoed as a two-character sequence: `<CR><LF>` (Carriage Return+Line Feed).

Characters within the `AT+i` command line are parsed as commands with associated parameter values.

The iChip supports editing of command lines by recognizing a backspace character. When ECHO is enabled, the iChip responds to receipt of a backspace by echoing a backspace character, a space character, and another backspace. When ECHO is disabled, backspace characters are treated as data characters without any further processing.

If a syntax error is found anywhere in a command line, the remainder of the line is ignored and the I/ERROR result code returned.

An `AT+i` command is accepted by iChip once the previous command has been fully executed, which is normally indicated by the return of an appropriate result code.

Due to the fact that iChip is intended for Machine-to-Machine applications, only limited parsing is performed on `AT+i` commands it receives from the host. The following restrictions apply:

- When setting parameters to values larger than the 65535 limit, the values is accepted as mudulo 65535.
- The validity of input IP addresses is not checked.
- Illegal numbers, for example, 0.5 or 1.5 are not checked for validity.

Escape Code Sequence

While the iChip is in Internet mode attending to Internet communications, it is possible to break into the communications and abort the Internet mode in an orderly manner. This is achieved by sending the iChip a sequence of three (+) ASCII characters (+++) after a half second silence period. In response to this, the iChip:

- Shuts down Internet communications.
- Terminates data transmission to the host.
- Performs a software reset.
- Responds with an **I/ERROR (056)** message.
- If in SerialNET mode, responds with an **I/ONLINE** or **I/DONE** message.
- Returns to command mode.

A maximum delay of 10msec may elapse from the time the (+++) escape sequence is sent until iChip cuts off transmission to the host. The interrupted Internet activity is not completed. Nevertheless, this is considered to comprise a session. Thus, parameters set with the (~) character are restored to their permanent value.

Socket Command Abort

While the iChip is in Internet mode, during a TCP or UDP socket operation, it is possible to override iChip's normal timeout procedure and abort the current socket operation in an orderly manner. This is achieved by sending the iChip a sequence of three ASCII (-) characters (- - -) following a half second silence period. The socket commands to which this applies are: STCP, SUDP, SSND, and SFSH. When iChip detects the socket abort command, it aborts the last socket command and returns an **I/ERROR** following the STCP and SUDP commands, or **I/OK** during an SSND or SFSH command.

Flexible Host and Modem Interfaces

Users may select the interface through which iChip accepts AT+i commands from the host processor, as well as the interface through which AT commands are sent to a dial-up or cellular modem.

Available host interfaces are:

- USART0
- USART1
- USART2
- USB Device (identifies itself as a CDC device)
- USB Host (supports only USB Modem class)
- SPI
- RS-485

Available modem interfaces are:

- USART0
- USART1
- USART2
- USB Device
- USB Host (See list of supported modems)

As a USB host/device, iChip supports the Full-Speed USB standard (12Mbps).

Host-to-iChip interface is selected by setting the value of the Host Interface (HIF) parameter. Any value from 1 to 6 specifies a certain choice of interface, while a 0 value specifies automatic interface detection. In automatic interface detection mode, the first character sent from the host over one of the supported interfaces sets the host interface to be used throughout that session until the next iChip power cycle.

When automatic host interface detection mode is enabled, a host is connected to one of the interfaces, and the Host Fixed Baud Rate (BDRF) parameter is set to 'a' (automatic baud rate detection), the first character the host has to send to iChip in order to trigger detection must be an 'a' or 'A'. If BDRF is set to a fixed baud rate, *any* character sent from the host triggers automatic host interface detection.

In a similar fashion, an iChip-to-modem interface can be selected using the Modem Interface (MIF) parameter, except that automatic modem interface detection is not available.

Note that any changes to the HIF and MIF parameters take effect only after the following iChip power-up. Also note that iChip cannot be operated in SerialNET mode when the HIF parameter is set to automatic mode. Sending an SNMD command (activate SerialNET mode) with HIF set to automatic mode will result in an error message **I/ERROR (122)**. In addition, any feature that requires setting a fixed baud rate requires setting a fixed host interface, as well.

Hardware flow control is supported on USART0 and USART1 only. Hardware signal mirroring is enabled only if the host and modem interfaces are set to either USART0 or USART1. See description of Bit 2 of the FLW parameter.

Auto Baud Rate Detection

iChip supports auto baud rate detection on the host serial communications line. After power-up, iChip enters auto baud mode when the BDRF parameter is set to the value 'a'. The AT+iBDRA command forces iChip into auto baud mode while it is already in operation.

In auto baud mode, iChip expects an `'A'` or `'a'` character. This is usually the first character sent, since in command mode a meaningful command is always prefixed by `AT+i`.

The host may send an `'a'` or `'A'` to the iChip to allow it to determine the host's baud rate. It may also send a complete `AT+i` command. In any case, iChip detects the `'A'` or `'a'` character, determines the correct baud rate, and configures its serial channel during the stop bit. Thus, the next character is received by the serial port at the correct baud rate. The `A` itself is retained as well. iChip supports auto baud rate detection for the following baud rates: 2400, 4800, 9600, 19200, 38400, 57600, and 115200.

When the BDRF parameter contains a fixed baud rate, iChip initializes to the specified baud rate without entering auto baud rate mode. Commands issued by the host must be sent using that baud rate in order to be recognized. In this case, iChip can be forced into auto baud rate mode by holding the special input signal low for not more than five seconds following power-up.

iChip dial-up mode only: When the BDRM parameter is set to an `'a'` value, iChip assumes the attached modem has the auto baud rate feature. Once the host⇔iChip baud rate is determined, the iChip⇔modem baud rate is set to the same rate. Any other BDRM value is used as a fixed baud rate to the modem.

High Speed USART

Very high baud rates, up to 3Mbps, can be reached between host and iChip via one of iChip's USARTs. The BDRD parameter acts as baud rate divider. When set to '0', iChip sets its host USART baud rate according to the value of the BDRF parameter. When set to any value in the range 1-255, it divides the maximum supported baud rate – 3Mbps – by that value. The quotient of this division is set as the host baud rate, and the value of BDRF is ignored. For example, if BDRD is set to 2, then the host baud rate will be 3Mbps÷2=1.5Mbps.

If the iChip⇔modem interface is a USART, BDRD is set to any value other than '0', and the modem baud rate is set to Auto (BDRM='a'), then the modem baud rate will be set to a fixed value of 115,200bps.

In SerialNET mode, you can specify that host⇔iChip baud rate over USART be determined by the BDRD parameter. You do so by setting the first field of the SNSI parameter (<*baud*>) to '0'.

Reset via Serial Link

Issuing a BREAK signal on the host serial link effectively resets the iChip. A BREAK signal is issued by transmitting a LOW (zero value) for a period that is longer than 23 bits at the current baud rate. Considerably lowering the host baud rate (300 baud or less) and transmitting a binary zero generates a BREAK signal. After a BREAK signal is issued, iChip requires 4 seconds to complete the reset cycle before commands can be issued. When iChip is configured for auto baud rate, the BREAK method is especially useful to force iChip back into auto baud rate mode when iChip and the host lose synchronization.

Entering Rescue Mode during Runtime

The MSEL (Mode Select) input signal of the iChip (see the iChip CO2128 Datasheet), can be used for entering iChip into Rescue mode.

If MSEL is pulled low (logical 0) for more than 5 seconds during runtime, iChip waits until MSEL is pulled high (logical 1), performs a software reset and restarts in Rescue mode. In Rescue mode, iChip performs the following operations:

- If in SerialNET mode — iChip exits SerialNET mode (SNMD is permanently changed to 0).
- If serial baud rate (in BDRF or BDRD) is set to a fixed value — iChip forces auto baud rate detection. BDRF/BDRD retain their values and will be used again upon the next power-up.
- If Always Online mode is defined (TUP=2), or Automatic Router Start is enabled (ARS=1) — iChip bypasses this mode, which means that iChip does not attempt to go online until the next software or hardware reset.
- If LAN-to-WiFi Bridge mode is enabled and the BRM parameter contains a non-zero value — iChip disables Bridge mode and permanently assigns BRM=0.
- If the Host Interface parameter (HIF) is set to a fixed interface, it is forced to auto host interface detection mode (HIF=0).

Internet Session Hang-Up Procedure (Modem Only)

Upon completion of a dial-up Internet session, the iChip automatically executes a modem hang-up procedure:

- The DTR line is dropped.
- After a 1 second delay, iChip raises the DTR.
- If the modem responds to the DTR drop with a **No Carrier** then Done. Otherwise, iChip issues a (+++) to the modem followed by **ATH**.

Modem Startup

Following power-up and baud rate determination, iChip in dial-up mode issues the `AT<CR>` command to the modem to configure the modem's baud rate.

Analog-to-Digital Converter

iChip contains an Analog-to-Digital (A/D) 8-bit converter that receives analog input voltage through the ADC signal. This input voltage can be monitored: if it reaches a predefined upper threshold or goes below a certain lower threshold, an acknowledgement can be sent. This acknowledgement is sent to the host processor through one of iChip's general-purpose I/O pins (GPIO).

Input voltage can be polled every predefined number of milliseconds. In addition, a report can be obtained at any given time by issuing the AT+iRP19 command.

The following parameters determine the behavior of the A/D converter:

- ADCL and ADCD specify threshold and delta values, respectively. If the value read from the register of the A/D converter is greater than the sum of ADCL and ADCD, then the GPIO pin specified by the ADCP parameter is asserted High. If that value is less than ADCL minus ADCD, the GPIO pin is asserted Low.
- The ADCT parameter defines an interval, in milliseconds, between consecutive queries of the value of the A/D converter's register. iChip's response time to value changes is up to 40ms.

In order to enable the A/D converter polling mechanism, you must, at the very least, set the ADCL, ADCT, and ADCP parameters to a non-zero value.

The following table summarizes the behavior of the A/D converter.

ADC Register Value	*GPIO Pin State*
R > L+D	High
R < L-D	Low

Table 1.1 A/D Output vs. Input

Legend:

- R — ADC register value, which is a binary representation of the A/D converter's analog input voltage.
- L — Base level, or threshold, as defined by the ADCL parameter.
- D — Delta, as defined by the ADCD parameter.

iChip Readiness Indication

This iChip Readiness Indication feature provides an indication of iChip's readiness to accept AT+i commands following a hardware reset. Using this feature, iChip can also notify the host when it is ready for IP communication.

This functionality is based on two parameters – RRA and RRHW. The RRA parameter can be set to send a software message to the host, assert a dedicated hardware pin, or do both. The RRHW parameter specifies which of iChip's I/O pins will be asserted.

The hardware pin specified by the RRHW parameter is asserted High immediately after power up. It will be asserted Low when iChip is ready to receive AT+i commands, and asserted High again following iChip's response to any AT+i command.

Gratuitous ARP

The purpose of Gratuitous ARP is mainly to inform routers and stations of a new LAN client that has joined the network or a client whose MAC address or IP address has changed. This way other stations and routers / switches can update their ARP tables with the new data and save time and traffic later when they need to access that LAN client. Clients usually send this message when they are assigned an IP address, recover from a Link Lost condition or undergo any change in their MAC / IP address.
iChip shall send a Gratuitous ARP immediately upon going online and establishing a new IP on its LAN end. iChip shall also send a Gratuitous ARP after regaining the LAN link from a Link-Lost condition.
iChip sends Gratuitous ARP packets only when connected to a wired Ethernet or Wireless LAN.

2 General Format

AT+i<*cc*>[<*del*>[<*parameter*> | #UF*n*]…]<*CR*>

<*cc*> *(or* <*par*>*)*	2–4 letter command code (<*cc*>) or parameter name (<*par*>)
<*del*>	Delimiter: '=', '~', '?', ':', ','
<*parameter*>	Optional parameter or data. If <*parameter*> includes a <*del*>, as defined above, it must be enclosed in single (') or double (") quotes. The terminating <*CR*> is considered as a terminating quote as well.
#UF*n*	User-field macro substitution
<*CR*>	Carriage Return line terminator (ASCII 13)

AT+i Commands by Category

Command	Function	Parameters/Description
AT+i	Command prefix	Required to precede all commands
	Host Interface	
En	Echo Mode	n=0 Do not echo host characters n=1 Echo all host characters (default upon power-up) This command is equivalent to and interchangeable with ATEn.
	Parameter Database Maintenance	
<*par*>=*value* *-or-* <*par*>:*value*	Set parameter	*value* stored in parameter <*par*> in nonvolatile memory. <*par*> retains set value indefinitely after power down.
<*par*>~*value*	Assign single session parameter value	*value* is assigned to parameter <*par*> for the duration of a single Internet session. Following the session, the original value is restored.
<*par*>?	Read parameter	Parameter value is returned.
<*par*>=?	Parameter allowed values	Returns the allowed values for this parameter.
FD	Factory Defaults	Restores all parameters to factory defaults.
	Status Report	
RP<*i*>	Request status report	Returns a status report value based on <*i*>.
	Connection	
BDRA	Auto baud rate mode	Forces iChip into auto baud rate detection mode.
UP	Connect to Internet	Forces iChip to go online, establish an Internet session, and optionally register its IP address.
BDRI	Change baud rate	n=0..255 Sets baud rate divider from 3Mbps
TUP	Triggered Internet session mode	Enters a mode in which iChip goes online in response to triggers from external signals. It also supports a special Always Online mode.
DOWN	Perform a software reset	Performs a software reset. Forces iChip to terminate an Internet session and go offline.
PING	PING a remote system	Sends a PING message and waits for its echo response.

Command	Function	Parameters/Description
Send E-mail		
[!]EMA:<*text*>	Send textual e-mail	Defines the textual contents of the e-mail body. Following this command, several text lines can be sent in sequence.
[!]EMB:<*sz*>,<*data*>	Send binary e-mail	Prefixes a binary data stream. The data is encapsulated as a base 64 encoded MIME attachment. Following this prefix, exactly <*sz*> bytes are streamed to iChip.
[!]E*	Terminate binary e-mail	Terminates a binary (MIME attachment) e-mail.
Retrieve E-mail		
[!]RML	Retrieve mail list	Retrieves an indexed, short form list of all qualifying messages in mailbox.
[!]RMH[:<*i*>]	Retrieve header	Retrieves only the e-mail header part from the <*i*>'th e-mail in the mailbox, or the entire mailbox.
[!]RMM[:<*i*>]	Retrieve e-mail	Retrieves all e-mail contents of the <*i*>'th e-mail in the mailbox, or the entire mailbox.
HTTP Client		
[!]RLNK[:<*URL*>]	Retrieve link	Retrieves a file from a URL on a web server. If <*URL*> is not specified, uses the URL stored in the URL parameter.
[!]SLNK:<*text*>	Send POST request	Sends a file consisting lines of ASCII to a web server defined in the URL parameter.
HTTP Server		
WWW	Activate the web server	Activates iChip's internal web server. Once activated, remote browsers can surf iChip's website.
WNXT	Retrieve next changed web parameter	Returns the parameter tag name and new value of the next web parameter that has been changed as a result of a submit by a remote browser.
SerialNET		
[!|@]SNMD	Activate SerialNET mode	Activates iChip's dedicated serial-to-network SerialNET mode.
Telnet Client		
TOPN	Telnet open session	Opens a Telnet session to a remote Telnet server. If iChip is not online, it is connected.
TRCV	Telnet receive	Receives data from a remote Telnet server.
TSND	Telnet send line	Sends an ASCII data line to a remote Telnet server.
TBSN[%]	Telnet send binary stream	Sends a binary data stream to a remote Telnet server.
TFSH[%]	Telnet flush	Flushes a Telnet socket's outbound data.
TCLS	Telnet close	Closes a Telnet session.

Command	Function	Parameters/Description
File Transfer Protocol (FTP)		
FOPN	Open FTP link	Opens an FTP command socket to a remote FTP server. If iChip is not online, it is connected. Once an FTP link is established, it can be used to carry out operations on the server's file system.
FOPS	Open secure FTP link	Opens an FTP link and negotiates an SSL3/TLS1 connection on the control channel. All following FTP operations in this session are performed over an SSL3/TLS1 connection.
FDL	FTP directory listing	Retrieves the remote FTP server's file directory listing. The full server-dependent listing is returned.
FDNL	FTP directory name list	Retrieves the remote FTP server's file directory listing. Only file names are returned.
FMKD	FTP make directory	Creates a directory on a remote FTP server.
FCWD	FTP change directory	Changes a remote FTP server's current directory.
FSZ	FTP file size	Retrieves the size of a file stored on a remote FTP server.
FRCV	FTP file receive	Downloads a file from a remote FTP server.
FSTO	FTP file store	Opens a file for upload to a remote FTP server. If the file already exists, it is overwritten.
FAPN	FTP file append	Opens a file on a remote FTP server for appending. If the file does not already exist, it is created.
FSND	FTP file send	Sends data to a file on a remote FTP server. The file must be already open by a previous FSTO or FAPN command.
FCLF	FTP close file	Closes the currently open file on an FTP server. Any data uploaded to the file with the FSND command is retained on the server.
FDEL	FTP delete file	Deletes a file from a remote FTP server's file system.
FCLS	FTP close	Closes an FTP link.

Command	Function	Parameters/Description
Socket Interface		
STCP:<*host*>, <*port*>[,<*lport*>]	Socket TCP	Opens and connects a TCP socket. If iChip is not online, it is connected. The responding system is assumed to be a server listening on the specified socket. Returns a handle to the socket.
SUDP: <*host*>,<*rport*> [,<*lport*>]	Socket UDP	Opens, connects, and optionally binds a UDP socket. If iChip is not online, it is connected. Returns a handle to the socket.
LTCP: <*port*>,<*backlog*>	Listening socket	Opens a TCP listening socket on <*port*>. Allows a maximum of <*backlog*> concurrent connections. Returns a handle to the socket. Up to two listening sockets are supported.
LSST:<*hn*>	Listening socket status	Returns a list of active socket handles accepted for a listening socket identified by handle <*hn*>.
SST:<*hn*>	Single socket status	Returns status of a single socket identified by handle <*hn*>. A subset of RP4 report.
SCS:<*hn*>	Socket connection status	Returns status of a single socket identified by handle <*hn*>. A subset of RP4 report. Does not report number of buffered characters.
SSND[%]: <*hn*>,<*sz*>:<*stream*>	Socket send	Sends a byte stream of size <*sz*> to the socket identified by handle <*hn*>. The % flag indicates automatic socket flush.
SRCV:<*hn*> [,<*max*>]	Socket receive	Receives a byte stream from the socket identified by handle <*hn*>. Accepts up to <*max*> bytes. If <*max*> is not specified, all available bytes are retrieved.
GPNM:<*hn*>	Get peer name	Retrieves peer name (<*IP*>:<*port*>) of a remote connection to the TCP/UDP socket specified by socket handle <*hn*>.
SDMP:<*hn*>	Dump socket buffer	Dumps all buffered data currently accumulated in a socket's input buffer. The socket remains open.
SFSH[%]:<*hn*>	Flush socket's outbound data	Flushes (sends immediately) data accumulated in a socket's outbound buffer. If the flush-and-acknowledge flag (!) is specified, iChip waits for peer to acknowledge receipt of the TCP packet.
[!]SCLS:<*hn*>	Close socket	Closes a TCP/UDP socket. If that socket is the only socket open and the stay online flag (!) is not specified, iChip terminates the Internet session and goes offline.
SSL:<*hn*>	SSL3/TLS1 socket connection	Negotiates an SSL3/TLS1 connection over an active TCP socket.

Command	Function	Parameters/Description
Special Modem Command		
MCM	Interlaced modem command	Sends an interlaced AT command to the modem while it is online.
LAN Commands		
ETHD	Ethernet Down	Turn OFF the Ethernet PHY
ETHU	Etherent Up	Re-start the Ethernet PHY
Wireless LAN		
WLTR	WLAN transmisssion rate	Sets the maximum allowable WLAN transmission rate.
WLPW	WLAN Tx power	Sets the transmission power of the Marvell WLAN chipset.
WRFU	WLAN radio up	Turns on radio transmission of the Marvell WLAN chipset.
WRFD	WLAN radio down	Turns off radio transmission of the Marvell WLAN chipset.
WRST	Reset WLAN chipset	Performs a hardware reset of the Marvell WLAN chipset.
WLBM	WLAN b mode	Sets the Marvell WLAN chipset to 802.11/b mode.
WLGM	WLAN g mode	Sets the Marvell WLAN chipset to 802.11/g mode.
AWPS	Activate WPS	Activates a WPS configuration session.
Routing		
STRR	Start iRouter	Immediately start iRouter mode
STPR	Stop iRouter	Exit iRouter mode and go offline on the modem side
[!]SPPP:<mode>[,IP]	Start PPP session	Start PPP session with the host processor on the interface defined by [!], with mode <mode>. Assign IP address defined by optional [IP].
Remote Firmware Update		
RFU	Remote firmware update	Updates firmware from a remote HTTP or FTP server.
Parameter Profiles		
SPRF	Store a profile	Store all existing settings to profile number 1.
LPRF	Load a profile	Replace existing settings with profile number 1.
DPRF	Display a profile	Display the context of profile 0 or 1.

Table 2.1 AT+i Commands by Category

3 AT+i Result Code Summary

Response String		**Denotation**		
I/OK		Command was successfully executed.		
I/BUSY		iChip busy. Command discarded.		
I/DONE		iChip completed Internet activity; returned to command mode, or entered SerialNET mode.		
I/ONLINE		iChip completed Internet activity and returned to command mode, or entered SerialNET mode. iChip issues this response when it has remained online as a result of the stay online flag (!) or as a result of the web server being online.		
I/OFFLINE		iChip in LAN mode entered SerialNET Always Online mode but failed to detect a LAN link at time of entry.		
I/RCV		Marks beginning of e-mail retrieve mode, with XFH=1. iChip does not respond to any commands, except for (+++) (Break).		
I/PART		Marks beginning of MIME attachment part.		
I/EOP		Marks end of MIME attachment part.		
I/EOM		Marks end of e-mail message during retrieve.		
I/MBE		This flag is returned when attempting to retrieve mail from an empty mailbox.		
I/UPDATE		iChip is downloading a new firmware version. Allow up to 5 minutes to complete.		
I/ERROR (*nnn*)	*nnn*	Command error encountered. Command discarded.		
	41	*Illegal delimiter*	42	*Illegal value*
	43	*CR expected*	44	*Number expected*
	45	*CR or ‘,’ expected*	46	*DNS expected*
	47	*‘:’ or ‘~’ expected*	48	*String expected*
	49	*‘:’ or ‘=’ expected*	50	*Text expected*
	51	*Syntax error*	52	*‘,’ expected*
	53	*Illegal command code*	54	*Error when setting parameter*
	55	*Error when getting parameter value*	56	*User abort*
	57	*Error when trying to establish PPP*	58	*Error when trying to establish SMTP*
	59	*Error when trying to establish POP3*	60	*Single session body for MIME exceeds the maximum allowed*
	61	*Internal memory failure*	62	*User aborted the system*
	63	*~CTSH needs to be LOW to change to hardware flow control.*	64	*User aborted last command using ‘---’*
	65	*iChip unique ID already exists*	66	*Error when setting the MIF parameter*
	67	*Command ignored as irrelevant*	68	*iChip serial number already exists*
	69	*Timeout on host communication*	70	*Modem failed to respond*
	71	*No dial tone response*	72	*No carrier modem response*
	73	*Dial failed*	74	*Modem connection with ISP lost* *-or-* *LAN connection lost* *-or-* *WLAN connection lost*
	75	*Access denied to ISP server*	76	*Unable to locate POP3 server*
	77	*POP3 server timed out*	78	*Access denied to POP3 server*

	79	*POP3 failed*	80	*No suitable message in mailbox*
	81	*Unable to locate SMTP server*	82	*SMTP server timed out*
	83	*SMTP failed*	84	*RESERVED*
	85	*RESERVED*	86	*Writing to internal non-volatile parameters database failed*
	87	*Web server IP registration failed*	88	*Socket IP registration failed*
	89	*E-mail IP registration failed*	90	*IP registration failed for all methods specified*
	91	*RESERVED*	92	*RESERVED*
	93	*RESERVED*	94	*In Always Online mode, connection was lost and re-established*
			96	*A remote host, which had taken over iChip through the LATI port, was disconnected*
			98	*RESERVED*
	99	*RESERVED*	100	*Error restoring default parameters*
	101	*No ISP access numbers defined*	102	*No USRN defined*
	103	*No PWD entered*	104	*No DNS defined*
	105	*POP3 server not defined*	106	*MBX (mailbox) not defined*
	107	*MPWD (mailbox password) not defined*	108	*TOA (addressee) not defined*
	109	*REA (return e-mail address) not defined*	110	*SMTP server not defined*
	111	*Serial data overflow*	112	*Illegal command when modem online*
	113	*Remote firmware update attempted but not completed. The original firmware remained intact.*	114	*E-mail parameters update rejected*
	115	*SerialNET could not be started due to missing parameters*	116	*Error parsing a new trusted CA certificate*
	117	*Error parsing a new Private Key*	118	*Protocol specified in the USRV parameter does not exist or is unknown*
	119	*WPA passphrase too short - has to be 8-63 chars*	120	*RESERVED*
	121	*RESERVED*	122	*SerialNET error: Host Interface undefined (HIF=0)*
	123	*SerialNET mode error: Host baud rate cannot be determined*	124	*SerialNET over TELNET error: HIF parameter must be set to 1 or 2*
	125	*Invalid WEP key*	126	*Invalid parameters' profile number*
			128	*Product ID already exists*
	129	*HW pin can not be changed after Product-ID was set*		
			200	*Socket does not exist*
	201	*Socket empty on receive*	202	*Socket not in use*
	203	*Socket down*	204	*No available sockets*
			206	*PPP open failed for socket*
	207	*Error creating socket*	208	*Socket send error*
	209	*Socket receive error*	210	*PPP down for socket*
			212	*Socket flush error*
	215	*No carrier error on socket operation*	216	*General exception*

	217	*Out of memory*	218	*An STCP (Open Socket) command specified a local port number that is already in use*
	219	*SSL initialization/internal CA certificate loading error*	220	*SSL3 negotiation error*
	221	*Illegal SSL socket handle. Must be an open and active TCP socket.*	222	*Trusted CA certificate does not exist*
	223	*RESERVED*	224	*Decoding error on incoming SSL data*
	225	*No additional SSL sockets available*	226	*Maximum SSL packet size (2KB) exceeded*
	227	*AT+iSSND command failed because size of stream sent exceeded 2048 bytes*	228	*AT+iSSND command failed because checksum calculated does not match checksum sent by host*
	229	*SSL parameters are missing*	230	*Maximum packet size (4GB) exceeded*
			300	*HTTP server unknown*
	301	*HTTP server timeout*	302	*RESERVED*
	303	*No URL specified*	304	*Illegal HTTP host name*
	305	*Illegal HTTP port number*	306	*Illegal URL address*
	307	*URL address too long*	308	*The AT+iWWW command failed because iChip does not contain a home page*
	309	*WEB server is already active with a different backlog.*	400	*MAC address exists*
	401	*No IP address*	402	*Wireless LAN power set failed*
	403	*Wireless LAN radio control failed*	404	*Wireless LAN reset failed*
	405	*Wireless LAN hardware setup failed*	406	*Command failed because WiFi module is currently busy*
	407	*Illegal WiFi channel*	408	*Illegal SNR threshold*
	409	*WPA connection process has not yet completed*	410	*The network connection is offline (modem)*
	411	*Command is illegal when Bridge mode is active*		
			500	*RESERVED*
	501	*Communications platform already active*	502	*RESERVED*
	503	*RESERVED*	504	*RESERVED*
	505	*Cannot open additional FTP session – all FTP handles in use*	506	*Not an FTP session handle*
	507	*FTP server not found*	508	*Timeout when connecting to FTP server*
	509	*Failed to login to FTP server (bad username or password or account)*	510	*FTP command could not be completed*
	511	*FTP data socket could not be opened*	512	*Failed to send data on FTP data socket*
	513	*FTP shutdown by remote server*	514	*RESERVED*
			550	*Telnet server not found*
	551	*Timeout when connecting to Telnet server*	552	*Telnet command could not be completed*
	553	*Telnet session shutdown by remote server*	554	*A Telnet session is not currently active*
	555	*A Telnet session is already open*	556	*Telnet server refused to switch to BINARY mode*

	557	*Telnet server refused to switch to ASCII mode*	558	*RESERVED*
	559	*RESERVED*	560	*Client could not retrieve a ring response e-mail*
	561	*Remote peer closed the SerialNET socket*		
			570	*PING destination not found*
	571	*No reply to PING request*		
			600	*Port Forwarding Rule will create ambiguous NAT entry*

Table 3.1 AT+i Result Code Summary

Note: All iChip response strings are terminated with `<CR><LF>`.

4 Report Status

+i[!]RP*i* — Report Status

Syntax: AT+i[!]RP*i*

Returns a status report.

Parameters: *i*=0..22

Command Options:

i=0 Returns the iChip part number.

i=1 Returns the current firmware revision and date.

i=2 Returns the connection status.

i=3 Returns boot-block revision and date.

i=4 Returns iChip socket status.

i=5 Returns a unique serial number.

i=6 Returns current ARP table.

i=7 Returns socket buffers utilization bitmap. iChip's DATA_RDY signal can be used to signal socket buffer status changes in hardware. This signal is raised when new data in one or more sockets is available, or when a remote browser has changed a web parameter. It is lowered when ***any*** socket or web parameter is read.

i=8 Returns current time-of-day based on time retrieved from the Network Time Server and the GMT offset setting. Returns an all-zero response if a timestamp has not yet been retrieved from the network since the last power-up.

i=9 *Reserved*

i=10 and AT+i!RP10 Returns two different status reports about the current Wireless LAN connection.

i=11 Returns a list of all Access Points available in the surrounding area.

AT+i!RP11 Returns a list of all Ad-Hoc networks available in the surrounding area.

i=14 Returns a DHCP server table of MAC and IP addresses of all the stations connected to iChip.

i=19 Returns Analog-to-Digital Converter (ADC) pin status report.

i=20 Returns a list of all APs and Ad-Hoc networks available in the surrounding area.

i=22	Returns a list of all non-empty Port Forwarding rules (PFWn)
Default:	None
Result Code:	
i=0..22	Status message as detailed in the next section, followed by **I/OK**.
I/ERROR	Otherwise

Status Message Format

<table>
<tr><th>Report Option</th><th>Format</th></tr>
<tr><td>0</td><td>COnnnn-ii
nnn – Version number; ii – Interface code: S-Serial, L-LAN, D-Dual</td></tr>
<tr><td>1</td><td>I<i>iimmmTss</i> (<<i>version-date</i>>)
<i>Iii – Interface code; mmm</i> – Major Version; <i>T</i> – Version type code;
<i>ss</i> – Sub-version</td></tr>
<tr><td>2</td><td>Status string:
<i>"Modem data<CR/LF>"</i>
<i>"Command mode<CR/LF>"</i>
<i>"<CR/LF>Connecting to ISP<CR/LF>"</i>
<i>"<CR/LF>Connected to ISP<CR/LF>"</i>
<i>"<CR/LF>Connecting as RAS<CR/LF>"</i>
<i>"<CR/LF>RAS Connected<CR/LF>"</i>
<i>"<CR/LF>Closing PPP<CR/LF>"</i>
<i>"<CR/LF>Establishing SMTP<CR/LF>"</i>
<i>"<CR/LF>Sending Email<CR/LF>"</i>
<i>"<CR/LF>Establishing POP3<CR/LF>"</i>
<i>"<CR/LF>POP3 Open<CR/LF>"</i>
<i>"<CR/LF>Establishing HTTP<CR/LF>"</i>
<i>"<CR/LF>Receiving HTTP<CR/LF>"</i>

"<i><CR/LF></i>Carrier Lost<CR/LF>"<br>"<i><CR/LF></i>Link Lost<CR/LF>"

LAN-to-WiFi Bridge Mode when BRM>0:
"LAN/WIFI Bridge Mode,<<i>LAN Status</i>>,<WIFI status>"
<i>Where,</i>
<<i>LAN Status</i>>: 0 – No Link
1 – Link OK
<<i>WiFi Status</i>>: 1 – Not Connected
2 – Connecting
4 – Connected</td></tr>
<tr><td>3</td><td>nnmm – Boot block version number</td></tr>
<tr><td>4</td><td>I/(<<i>sock0sz</i>>, <<i>sock1sz</i>>, … ,<<i>sock9sz</i>>)
<i>sock<i>sz</i>   >=0 : Number of bytes pending in socket's input buffer
<0 : Negative value of socket's error code</td></tr>
<tr><td>5</td><td>nnnnnnnn – Hexadecimal representation of iChip serial number.</td></tr>
<tr><td>6</td><td>Current ARP table listing:
INTERNET ADDRESS PHYSICAL ADDRESS STATE TTL
nnn.nnn.nnn.nnn xxxxxxxxxxxx VALID nnn sec.
For debugging purposes.</td></tr>
<tr><td>7</td><td>I/xxxx
xxxx – 16 bit Hex Value Bitmap
A bit set to '1' indicates that the corresponding socket contains buffered data, which needs to be read by the host.
<table>
<tr><td>bit</td><td>15</td><td></td><td></td><td></td><td></td><td>10</td><td></td><td></td><td>7</td><td></td><td></td><td></td><td></td><td></td><td></td><td>0</td></tr>
<tr><td>socket</td><td></td><td></td><td></td><td></td><td></td><td>WEB</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
</table>

Bit 10 is set to '1', when the remote browser updates one or more application website parameter tags. It will be reset to '0' when the host reads any</td></tr>
</table>

Report Option	Format
	application website parameter, using AT+i<*Parameter Tag*>?
8	The current time-of-day is returned according to ISO 8601: <*YYYY-MM-DD*>T<*HH:MM:SS*> <*TZD*> *YYYY-MM-DD -- Year-Month-Day* ; 'T' – *Fixed Separator* ; *HH:MM:SS - Hrs:Mins:Secs* ; TZD - *Time Zone Designator: +hh:mm or –hh:mm* All-zeros response: 0000-00-00T00:00:00 <*TZD*>.
9	*Reserved*
10	I/(<port stat>, <xfer rate>, <sig level>, <lnk qual>) port stat -- Port Status: 0: Wireless LAN adapter not present 1: Wireless LAN adapter disabled 2: Searching for initial connection 4: Connected 5: Out of range xfer rate -- Transfer rate in the range 1..54 sig level -- Signal level [%], in the range 0..100 lnk qual -- Link quality [%], in the range 0..100

AT+i!RP10	Returns a report of the current WLAN connection.
	`<SSID>,<BSSID>,<security type>,<WPA status>,<channel>,<SNR>`
	
	▪ `<security type>=NONE|WEP64|WEP128|WPA|WPA2`
	▪ `<WPA status>=Completed|Not Completed`
	This indication whether WPA negotiation completed or not, appears only when WPA/WPA2 security is specified.
	
	Note: For Ad-Hoc networks, `SSID` starts with (!).
	
	For example:
	Jetta,06:14:6C:69:4A:7C,WPA,Completed,1,68
	I/OK
	
	If +iWIAP is defined to enable Auto Connection to an IP-Enabled IP, the following message may be returned instead of the above:
	`"Scanning for IP-Enabled AP"`

11	iChip scans all available Access Points (APs) in the surrounding area and returns a list of APs. The AP having the strongest signal appears first. `<SSID>,<security_scheme>,<signal_strength><CR><LF>` `<SSID>,<security_scheme>,<signal_strength><CR><LF>` . . `<SSID>,<security_scheme>,<signal_strength><CR><LF>` `SSID` – Up to 32 alphanumeric characters `security_scheme` – None | WEP | WPA `signal_strength` – 0 - low, 1 - good, 2 - excellent ***Note:*** If no APs are detected, only I/OK<CR><LF> is returned.

Report Option	Format
AT+i!RP11	Returns a list of all Ad-Hoc networks available in the surrounding area. The Ad-Hoc network having the strongest signal appears first. `<SSID>,<security_scheme>,<signal_strength><CR><LF>` `<SSID>,<security_scheme>,<signal_strength><CR><LF>` . . `<SSID>,<security_scheme>,<signal_strength><CR><LF>` *SSID* – Up to 32 alphanumeric characters *security_scheme* – None | WEP *signal_strength* – 0 - low, 1 - good, 2 - excellent ***Note*:** If no Ad-Hoc networks are detected, only **I/OK**<CR><LF> is returned. For example: **Free Public WiFi,NONE,1** **I/OK**

14	Returns a DHCP server table of MAC and IP addresses of all the stations connected to iChip.
	`MAC Address` `IP Address`
	`<MAC_Address_1>` `<IP_Address_1>`
	.
	`<MAC_Address_n>` `<IP_Address_n>`
	
	For example:
	MAC Address **IP Address**
	00039406068C **192.168.0.2**
	000394094D1B **192.168.0.3**
	I/OK

19	Returns Analog-to-Digital Converter (ADC) pin status report. If the ADCP parameter is set, the report returns the GPIO pin state. Otherwise, it returns the ADC value only. `ADC value=<level>, GPIO state=<state>` ▪ *level* is an integer in the range 0-255 representing the input voltage measured on the ADC pin, calculated as follows: (A/3.3V)*255=*level*, where A is the analog input voltage. ▪ *state* indicates the state of the output GPIO pin: 0 (High) | 1 (Low). GPIO state is reported only if the ADCL, ADCT and ADCP parameters are set. For example, if the ADCP parameter is set: **ADC value = 255, GPIO state = 0** **I/OK** If the ADCP parameter is not set: **ADC value = 255** **I/OK**

Report Option	Format
20	Returns a list of up to 16 APs and Ad-Hoc networks available in the surrounding area. Each line contains the following comma-separated fields: *<SSID>*,ADHOC|AP,*<BSSID>*,*<security type>*,*<channel>*,*<RSSI>* ▪ *<security type>*=NONE|WEP64|WEP128|WPA|WPA2 ▪ *<RSSI>*= Value between 0-255 which represents (SNR+NoiseFloor). Higher RSSI values indicate weaker signal strength. For example: Jetta,AP,06:14:6C:69:4A:7C,WPA,1,25 RTL8186-default,AP,00:E0:4C:81:86:86,NONE,1,77 dlink_test,AP,00:1C:F0:9A:63:7A,NONE,1,68 Guest,AP,00:15:E9:0C:38:F2,WPA2,6,69 ABC,AP,00:1C:F0:40:CC:60,NONE,6,65 Yuval,AP,00:0E:2E:C6:B6:E1,NONE,6,62 GANG_TEST,AP,00:17:3F:9F:89:6E,NONE,7,67 Bora,AP,00:14:78:F7:11:BA,NONE,7,26 3com_test,AP,00:0F:CB:FF:27:8F,NONE,7,81 INET,AP,00:0F:CB:FF:7E:5D,WPA,7,82 Blue-I The Lab,AP,00:1B:2F:57:65:62,WEP,7,45 Mistral,AP,00:11:6B:3B:55:E2,WEP,9,27 Sirocco,AP,00:18:4D:DE:D7:DF,WPA2,11,44 Free Public WiFi,ADHOC,D2:B3:5B:06:CA:04,NONE,11,69 BlueI,AP,00:0E:2E:55:39:A6,WEP,11,57 private,AP,00:0E:2E:FD:F0:69,WPA,11,74 I/OK
22	Returns a list of all non-empty Port Forwarding rules (PFWn). The report displays each Port-Forwarding rule in a separate line, prefixed with the rule's index number. The report line syntax is: *#* - [L | M]*<w-port>*,*<l-IP:l-port>*[,*<type>*]<CR/LF> For example, **0 - 8000,192.168.0.1 :80,0** **3 - 8800,192.168.0.5 :80** **I/OK**

Table 4.1 Report Status Message Format

5 Connection

+iBDRA — Forces iChip into Auto Baud Rate Mode

Syntax: AT+iBDRA

Forces the iChip into auto baud rate mode. The following `A`, `AT` or `AT+i` command (in any combination of upper or lowercase) from the host will synchronize on the host's baud rate. iChip supports auto baud rate detection for the following baud rates: 2400, 4800, 9600, 19200, 38400, 57600, and 115200.

Result code:

I/OK This result code is sent using the previous baud rate.

+iBDRI — Change Baud Rate for Current Session

Syntax: AT+iBDRI=<*n*>

Temporarily change the baud rate. When set to '0', iChip sets its host USART baud rate according to the value of the BDRF parameter. When set to any value in the range 1-255, it divides the maximum supported baud rate of 3Mbps, by that value. The quotient of this division is set as the host baud rate, and the value of BDRF is ignored.

It is recommended to wait a minimum period of 10mSec before issuing the first command at the new baud rate to allow iChip to complete the baud rate change.

The previous baud rate settings will be applied in the next power cycle.

Parameters:

n=0 Host baud rate is determined by the BDRF parameter.

n=1-255 Host baud rate is set by dividing 3Mbps by *n*.

For example, if *n*=2, the host baud rate will be set to 3Mbps÷2=1.5Mbps.

Result Code:

I/OK If *n* is within limits. Returned in the original baud rate.

I/ERROR Otherwise

AT+iBDRI=? Returns the message **"0-255"** followed by **I/OK**.

+iUP — Initiate Internet Session

Syntax: AT+iUP[:*n*]

Initiates an Internet session by going online. In a dialup/cellular environment, a PPP Internet connection is established. Once online, optionally goes through an IP registration process, as determined by *n*.

Parameters: *n*=0..1

Default: *n*=0

Command Options:

n=0 Go online.

n=1 Go online and carry out the IP registration process according to the relevant registration option parameters.

Result Code:

I/ONLINE After successfully establishing an Internet session and completing the IP registration (if requested).

I/ERROR If iChip cannot go online and establish an Internet session or cannot complete the requested IP registration.

+iTUP — Triggered Internet Session Initiation

Syntax: AT+iTUP:<*n*>

Enter triggered Internet session initiation mode.

This command is relevant in a modem environment only.

Parameters: *n*=0..2

Command Options:

n=0 Disable triggered Internet session initiation mode.

n=1 Enter triggered Internet session initiation mode. Upon receiving a hardware signal trigger (Modem RING or MDSEL signal pulled low), establish a PPP Internet connection and carry out the IP registration process according to the relevant registration option parameters.

If any characters are received on the host port prior to receiving a hardware signal, iChip exits this mode and functions normally. In this case, to reinstate this mode, issue AT+iTUP=1 again; reset iChip by issuing the AT+iDOWN command, or recycle power.

n=2 Always Online mode. Whenever iChip is offline, it automatically attempts to establish a PPP Internet connection and possibly carry out the IP registration process according to the relevant registration option parameters.

iChip disregards this mode and remains offline until the next SW or HW reset if:

- The MSEL (Mode Select) signal was pulled low (logical 0) for more than 5 seconds during runtime.

-or-

- The host issues the (`+++`) escape sequence.

Power must be recycled or the `AT+iDOWN` command issued for this command to take effect.

If iChip is in Auto Baud Rate mode (BDRF=a) and/or Auto Host mode (HIF=0), iChip waits for the a character on the host serial port to resolve the baud rate after rebooting and before activating the iRouter and going online, or before activating the DHCP server. Therefore, it is recommended to set a fixed host interface and a fixed baud rate in this case.

n=3 Always Online mode with keep-alive tests. iChip automatically attempts to establish Internet connection as described in mode TUP=2 above. iChip maintains its online status by issuing PING requests to one or two servers that are defined in the +iPDSn parameters. iChip will attempt to go offline and then online again if the PING requests fail. iChip considers the PING as failed only if previously the PING has succeeded at least once. In a sense, iChip is qualifying the PING server(s) address defined in the +iPDSn parameters before activating corrective measures.

Result Code:

I/OK If *n* is within limits

I/ERROR Otherwise

Notes:

1. When going online in one of these modes, iChip activates its web server if the AWS parameter is set (AWS>0).

2. In this mode, iChip does not go offline after a completion of any successful or unsuccessful Internet session started by the host, even if the stay online flag is not used.

3. When a Carrier Lost event is detected, iChip automatically retries to establish a connection (without performing a software reset), with the following exception: If, at the time of the detection, the host was waiting for a reply from iChip or was in the process of sending binary data (SSND, FSND, EMB), iChip reports error code 094 as soon as it can and only then tries to re-establish the connection. In all other cases, iChip gives the host no indication of losing the carrier. In the event of Carrier Lost, iChip closes any open TCP active sockets, but leaves UDP sockets and TCP passive (listening) sockets intact and updates their local IP if a new IP is assigned after establishing a new PPP connection. iChip does not close any open Internet sessions (FTP/Telnet sessions and so on), nor releases the handle of the active TCP sockets, thus giving the host a chance to read the session errors and get buffered incoming data from active TCP sockets.

4. When the PFR is larger than 0 and the PDSn parameters are configured, iChip verifies that it is online by sending PING messages to the PING destination servers defined in PDSn at a polling frequency defined by PFR. If both PING destination servers do not respond, iChip concludes that the Internet connection failed and tries to reestablish an Internet connection, as described above for the case of a lost carrier signal.

+iDOWN — Terminate Internet Session

Syntax: AT+iDOWN

Performs a software reset. Terminates an ongoing Internet session, goes offline and returns to Command mode.

This command is useful in a dialup environment following a command where the stay online flag (!) was specified.

All open sockets are closed and the web server deactivated.

Result Code:

I/OK

Followed by:

I/ERROR I/ERROR (056) is expected after terminating the current Internet session when the command caused iChip to abort an ongoing Internet activity or close an active socket. No further action is required following this notification.

-or-

I/DONE After terminating the current Internet session. Allow a 2.5 sec. delay for iChip
re-initialization following an Internet mode session. Relevant for iChip in dial-up mode only.

-or-

I/ONLINE After terminating the current Internet session.

+iPING — Send a PING Request to a Remote Server

Syntax: AT+iPING:<*host*>

Sends a two-byte ICMP PING request packet to the remote host defined by *host*.

Parameters: <*host*>=Logical name of the target host or a host IP address.

Command Options:

<*host*> The host name may be any legal Internet server name, which can be resolved by the iChip's DNS (Domain Name Server) settings. The host name may also be specified as an absolute IP address given in DOT form.

Result Code:

I/<*RTT*> Upon successfully receiving an ICMP PING reply from the *host*, the round trip time in milliseconds is returned (*RTT*). iChip allows up to <*PGT*> milliseconds for a PING reply. If a reply is not received within <*PGT*> milliseconds, iChip sends two more PING requests, allowing <*PGT*> milliseconds for a reply on each of the requests before reporting an error.

I/ERROR Otherwise

6 Special Modem Commands

+iMCM — Issue Intermediate Command to Modem

Syntax: AT+iMCM[:<*AT command*>]

Sends a single AT command to the modem during an internet session or enters Modem Command mode.

Parameters:

<*AT command*> Optional single AT command to be sent to modem.

Command Options:

<*AT command*> iChip puts the modem in command mode by issuing the (+++) escape sequence and then sends <*AT command*> to the modem, followed by a <CR>. <*AT command*> must include the AT prefix. After receiving the modem's response, iChip restores the modem to online operation mode by issuing the ATO command.

If <*AT command*> is not specified, iChip enters Modem Command mode. In this mode, all following commands are transferred as-is to the modem. Modem replies are relayed back to the host processor. iChip does not translate the commands. Modem Command mode is exited after the host issues the ATO command. iChip transfers the ATO command to the modem and relays the modem's response back to the host.

Returns: Modem's responses including command echo, if enabled.

Followed by:

I/OK When the modem successfully returns online.

I/ERROR If modem was unable to go back online.

7 MIME Encapsulated E-Mail Messages

iChip-Generated Binary Message Formats

Binary e-mail messages are sent via iChip using one or more AT+iEMB commands. The message format is limited to an optional body of text and a single attachment.

The following fields are added by iChip to the main message header:

```
X-Mailer: iChip <software version>
Message-ID: <Unique #>@iChip
Mime-Version: 1.0
Content-Type: multipart/mixed; boundary="CONE-iChip-<software version>"
```

The message's preface contains the following text:

```
"This MIME message was coded by iChip."
```

If the host application includes a text body for the message, it also contains the following lines in its header:

```
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-iCoverpage: Email
```

When no textual body contents are included – this section is omitted.

If different encoding is required, these fields may be changed. See the next section discussing encoding.

The binary attachment section follows, beginning with a MIME attachment header containing the following fields:

```
Content-Type: <User defined media type>/<User defined media subtype>;
      name=<User defined attachment filename>
      Content-Transfer-Encoding: base64
```

where,

- <*media type*> := "text" / "image" / "audio" / "video" / "application"
- <*media subtype*> := <A publicly-defined extension token.>
- <*filename*> := <User-defined name (including extension)> or <*unique filename*>
- <*media type*> defaults to "application" when otherwise not defined.
- <*media subtype*> defaults to "octet-stream" when otherwise not defined.

Following the header, a base 64-encoded data stream includes the entire binary data transferred to iChip from the host.

MIME-Related AT+i Commands and Parameters

Binary images are transferred to iChip for MIME message encapsulation via one or more AT+iEMB commands. An AT+iEMB command sequence must be terminated by the AT+iE* command, indicating the end of the binary e-mail message.

When several consecutive AT+iEMB commands are used, the host must issue the commands with an inter-command delay, which does not violate the SMTP server's timeout constraints. Otherwise, the SMTP server will timeout and abort the session. Average SMTP servers allow for delays in the range of 30 to 120 seconds. Additional AT+i commands may be interlaced within a sequence of AT+iEMB commands, except for the following AT+i commands: AT+iEMA, AT+iRML, AT+iRMH, AT+iRMM, AT+iRFU, AT+iRLNK, AT+iBDRA, and AT+iSNMD.

iChip does not limit the size of the binary attachment. However, ISPs do have limitations. An Internet connection is initiated immediately after the first AT+iEMB command, while the rest of the command is received. Once the connection to the SMTP server has been established, iChip acts as a pipeline, receiving binary info from the host, encoding it, and transmitting it to the Internet on-the-fly. Following the AT+iE* command, the e-mail is terminated and the Internet connection closed.

The escape sequence command (+++) is allowed within an AT+iEMB command, provided there is a half-second silence period before the (+++) is sent. Upon receiving the escape sequence, iChip aborts and orderly closes the Internet session. The partial mail message is not sent to the destination.

Binary Attachment Parameters

Parameter	Default	Description
MT	4 (application)	Media Type: 0 – Text; 1 – Image ; 2 – Audio ; 3 – Video ; 4 – Application
MST	octet-stream	Media Subtype String. For a list, see Appendix A.
FN	None	Attachment File Name (inc. extension). If a file name is not defined, iChip generates a unique filename without an extension.
BDY	None	ASCII text to be included in the e-mail's body in addition to the attachment. (Multiple lines allowed).

Table 7.1 Binary Attachment Parameters

Defining a Textual Body for Binary Messages

1. Permanent textual body contents:

```
AT+iBDY:<text lines> … <CR>.<CR>
```

The maximum fixed body size allowed is 96 characters (including embedded <CR><LF>). The text body is included in all future binary messages. In addition, the textual contents are committed to non-volatile memory on board the iChip.

2. Single session textual body contents:

```
AT+iBDY~<text lines> … <CR>.<CR>
```

The maximum temporary body size allowed is 1K characters (including embedded <CR><LF>). The text body is included in the next session binary message and then purged.

Email Encoding

Textual emails which are sent using the command +iEMA do not include encoding specifications. By default, such emails are interpreted by the receiving party as if encoded with US-ASCII, 7bit character set.

US-ASCII, 7bit character set is also applied to encode the textual body, saved in the parameter +iBDY, of binary e-mail messages which are sent via iChip using the AT+iEMB command.

The encoding may be changed to suit the application for both +iEMA and +iBDY, by changing the parameters +iCSTY and +iCTE. MIME encapsulation of the text is added if needed.

The parameter +iCSTY accepts a character set name (not case sensitive), which augments the Email header as defined in RFC2045. Character set names may be found on the web: http://www.iana.org/assignments/character-sets. If +iCSTY is empty "US-ASCII" is assumed.

The parameter +iCTE contains an alternative value for the Content-Transfer-Encoding header field, which can be used to specify both the encoding transformation that was applied to the body and the domain of the result. Encoding transformations other than the identity transformation are usually applied to data in order to allow it to pass through mail transport mechanisms which may have data or character set limitations. Examples for possible values are: "7bit" / "8bit" / "binary" / "quoted-printable" / "base64" / ietf-token / x-token. These values are not case sensitive.

MIME-Encapsulated E-Mail Message Format

Note: Bold lines are added by iChip.

```
Received: from JFK by FTGate SmartPop;
     Tue, 23 Nov 1999 09:26:21 +0200
Received: from mail.inter.net.il (hrz-153-147.access.net.il
[212.68.153.147])
      by mail.inter.net.il (8.9.3/8.8.6/PA) with SMTP id OAA11594;
      Mon, 22 Nov 1999 14:18:03 +0200 (IST)
Date: Mon, 22 Nov 1999 14:18:03 +0200 (IST)
From: lims@connectone.com
To: lims@connectone.com
To: connect1@inter.net.il
To: gadyl@netvision.net.il
X-Mailer: iChip ic401d05
X-Serial: 123456
Return-Receipt-To: lims@connectone.com
Message-ID: <15322@iChip>
Subject: iChip binary message via iModem
Mime-Version: 1.0
Content-Type: multipart/mixed; boundary="CONE-iChip-ic401d05"
X-UIDL: ad0c01ac458208bedea8b8522012e4b6

This MIME message was coded by iChip.
```

```
--CONE-iChip-ic401d05
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-Coverpage: Email
.
<Textual body, here>
.
.
--CONE-iChip-ic401d05
Content-Type: image/tiff; name="FaxImage.tif"
Content-Transfer-Encoding: base64

.
.
.
<Binary Base64-encoded data, here>
.
.
.
--CONE-iChip-ic401d05
```

8 E-mail Send Commands

+iEMA — Accept ASCII-Coded Lines for E-Mail Send

Syntax: AT+i[!]EMA:<*text lines*>

Defines a plain text e-mail body.

Parameters:

<*text lines*> Plain text e-mail body. The e-mail body contains <CR/LF> terminated ASCII character strings. <*text lines*> must be terminated by a dot character (.) in the 1st column of an otherwise empty line.

Command Options: <*text lines*>::={<ASCII *text line*><CRLF> …}<CRLF>.<CRLF>

Maximum size of <*text lines*> is limited to 18K, provided that no additional system resources are in use.

EMA uses the specified SMTP server to send the e-mail message. When iChip acquires TOD from a network timeserver, outgoing e-mail messages are time and date stamped.

! Stay online after completing the command

Result Code:

I/OK After all text lines are received and terminated by the (.) line.

I/ERROR If memory overflow occurred before all text lines are received.

Followed by:

I/DONE After successfully sending the e-mail. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.

-or-

I/ONLINE After successfully sending the e-mail, if the stay online flag (!) is specified.

-or-

I/ERROR If some error occurred during the send session.

+iEMB — Accept Binary Data for Immediate E-Mail Send

Syntax: AT+i[!]EMB[#]:<*sz*>,<*data*>

Defines and sends a MIME-encoded binary e-mail.

Parameters:

<*sz*> size of <*data*> in bytes

<*data*> <*sz*> bytes of binary data

Command Options:

<sz> 0..4GB

<*data*> 8 bit binary data. Must be exactly <*sz*> bytes long.

The binary data is encapsulated in a MIME-encoded e-mail message. The receiving end views the binary data as a standard e-mail attachment.

Several consecutive `+iEMB` commands can be issued in sequence to create a larger aggregate of data to be sent.

The e-mail contents are completed by issuing an AT+iE* (terminate binary e-mail) command. Following the first +iEMB command, iChip establishes an Internet connection while the data stream is being transmitted from the host. Once an SMTP session is established, iChip maintains a data transmit pipeline between the host and the SMTP server. iChip converts the binary data using BASE64 encoding on-the-fly. Following this command, the Internet session remains active to service additional +iEMB commands, until the +iE* terminating command.

EMB uses the specified SMTP server to send the e-mail message. When iChip acquires TOD from a network timeserver, outgoing e-mail messages are time and date stamped.

! Stay online after completing the command. This flag is redundant, as the iChip defaults to staying online until the AT+iE* command is issued.

Modem baud rate limit flag. When this character is included in the command, the iChip baud rate to the modem is limited by the baud rate from the host. This flag is relevant for serial modems only and is especially useful in GSM modem configurations. When this character is not present, the iChip attempts to lift the baud rate to the modem to its maximal value.

Result Code:

I/OK	If <*sz*> is within limits and after <*sz*> bytes have been received successfully.
I/ERROR	If <*sz*> is out of bounds, or if a communication error occurred during the Internet session.

Notes:

- If <*sz*> is larger than 256 bytes, iChip assumes host flow control. Depending on the setting of the FLW parameter, the flow control mode is either software or hardware. Under software flow control, the host processor must respond to iChip's flow control characters. The software flow control protocol is detailed in the Host ➔ iChip Software Flow Control section later in this document. When software flow control is active, it is recommended to set the iChip to Echo-Off mode. Under hardware flow control, the ~CTS/~RTS RS232 control signals must be connected and the host must respond to the iChip's ~CTS signal. The host may send data only when the ~CTS signal is asserted (active low). If a transmission error occurs while in hardware flow control, iChip continues receiving all remaining <*sz*> bytes before returning the I/ERROR response.
- Some SMTP servers limit e-mail message size to a value that is lower than iChip's limitations.
- If the BDY parameter is not empty, its contents are added to the outgoing e-mail as a textual body, in addition to the attachments.

+iE* — Terminate Binary E-Mail

Syntax:	AT+i[!]E*
	Terminates the current binary e-mail attachment.
Command Options:	
!	Stay online after completing the command
Result Code:	
I/OK	If a binary e-mail attachment is in the process of being defined. The e-mail message is terminated and the SMTP session is then completed and closed.
I/ERROR	Otherwise
Followed by:	
I/DONE	After successfully sending the e-mail. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.
-or-	
I/ONLINE	After successfully sending the e-mail, if the stay online flag (!) is specified.
-or-	
I/ERROR	If some error occurred during the send session.

9 E-Mail Retrieve

+iRML — Retrieve Mail List

Syntax:	AT+i[!]RML
	Retrieves pending e-mail list from current mailbox.
Command Options:	
!	Stay online after completing the command
Result Code:	
I/OK	To acknowledge successful receipt of the command.
I/ERROR	Otherwise
Returns:	
I/MBE	If the mailbox is empty.
Otherwise:	A list of qualifying e-mail message descriptors, separated by <CR/LF>. An e-mail message descriptor is composed of 5 <TAB> separated fields:
	<**i**><TAB><**sz**><TAB><**date**><TAB><**sbjct string**><TAB><**type/subtype**><CR/LF>
	where,
	<*i*> - E-mail message index in mailbox
	<*sz*> - E-mail message size in bytes
	<*date*> - E-mail message date (for the date field format refer to RFC822)
	<*sbjct string*> - E-mail message subject string (limited to 128 bytes)
	<*type/subtype*> - MIME content type. The literal NONE is used for non-MIME e-mail messages.
	E-mail messages that qualify the E-Mail Delete Filter (DELF) are not listed.
Followed by:	
I/DONE	After successfully retrieving the e-mail list. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.
-or-	
I/ONLINE	After successfully retrieving the e-mail list. When connected to a modem: After successfully retrieving the e-mail list, if the stay online flag (!) is specified.
I/ERROR	Otherwise

+iRMH — Retrieve Mail Header

Syntax: AT+i[!]RMH[:*i*]

Retrieves header of e-mail message <*i*> from current mailbox.

Parameters:

i Optional e-mail message index of a qualifying message. If no parameter is used, all e-mail headers are retrieved.

Command Options:

i Optional index of a qualifying message, as reported by AT+iRML.

! Stay online after completing the command

Default: Retrieves headers of all pending qualified mail messages.

Result Code:

I/OK When command is received and about to be processed.

I/ERROR Otherwise

Returns:

I/MBE If the mailbox is empty.

Otherwise: All header lines of all qualifying e-mail messages. Header lines are returned as-is. A line containing solely a (.) (period) in column 1 acts as a separator between the header lines of each e-mail. The HDL parameter limits the number of header lines per mail (HDL=0 specifies an unlimited number of lines per e-mail). Header field syntax is described in RFC822 and RFC2045.

Followed by:

I/DONE After successfully retrieving the e-mail headers. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.

-or-

I/ONLINE After successfully retrieving the e-mail headers.

When connected to a modem: After successfully retrieving the e-mail headers, if the stay online flag (!) is specified.

-or-

I/ERROR Otherwise

+iRMM — Retrieve Mail Message

Syntax: AT+i[!]RMM[:*i*]

Retrieves contents of e-mail message *i* from current mailbox.

Parameters:

i Optional e-mail message index of a qualifying message. If no parameter is used, all e-mails are retrieved.

Command Options:

i Optional index of a qualifying message, as reported by AT+iRML.

! Stay online after completing the command.

Default: Retrieves all pending qualified mail messages.

Result Code:

I/OK When command is received and about to be processed.

I/ERROR Otherwise

Returns:

I/MBE If the mailbox is empty.

Otherwise: For each e-mail part:

(For plain-text e-mails without MIME attachments)

I/PART – <*text*><TAB><*plain*><TAB><TAB> <*quoted-printable*><CR/LF>

-or- (For e-mails containing MIME attachments)

I/PART – <*media type*><TAB><*media subtype*><TAB> <*filename*><TAB> <*encoding method*><CR/LF>

-or- (When XFH – transfer e-mail headers – is set to YES)

I/RCV

-or-

Followed by: <*e-mail message contents*>

If the XFH parameter (transfer e-mail headers) is set to YES, all e-mail contents are returned as-is. The e-mail's headers followed by the e-mail's body are retrieved. MIME encapsulated e-mail messages are retrieved without BASE64 decoding. It is assumed that when the XFH parameter is set to YES, the host processor attends to all e-mail field parsing and contents decoding.

If the XFH parameter is set to NO, only the email's body (contents) is retrieved. If the email message contains a MIME-encapsulated attachment encoded in BASE64, iChip performs the decoding and transfers pure binary data to the host. Binary attachments encoded in a scheme other than BASE64 are returned as-is.

E-mails that qualify the Delete E-Mail Filter (DELF) are deleted from the mailbox without being downloaded.

Followed by:	
I/EOP	End of Part Message, if message is prefixed with an **I/PART** line. This repeats itself for all e-mail parts.
Followed by:	
I/EOM	End of Message This repeats itself for all qualifying e-mail messages.
When all messages have been retrieved:	
I/DONE	After successfully retrieving the e-mail. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.
-or-	
I/ONLINE	After successfully retrieving the e-mail. When connected to a modem: After successfully retrieving the e-mail, if the stay online flag (!) is specified.
-or-	
I/ERROR	Otherwise

E-Mail Receive (RMM) Flow Diagram

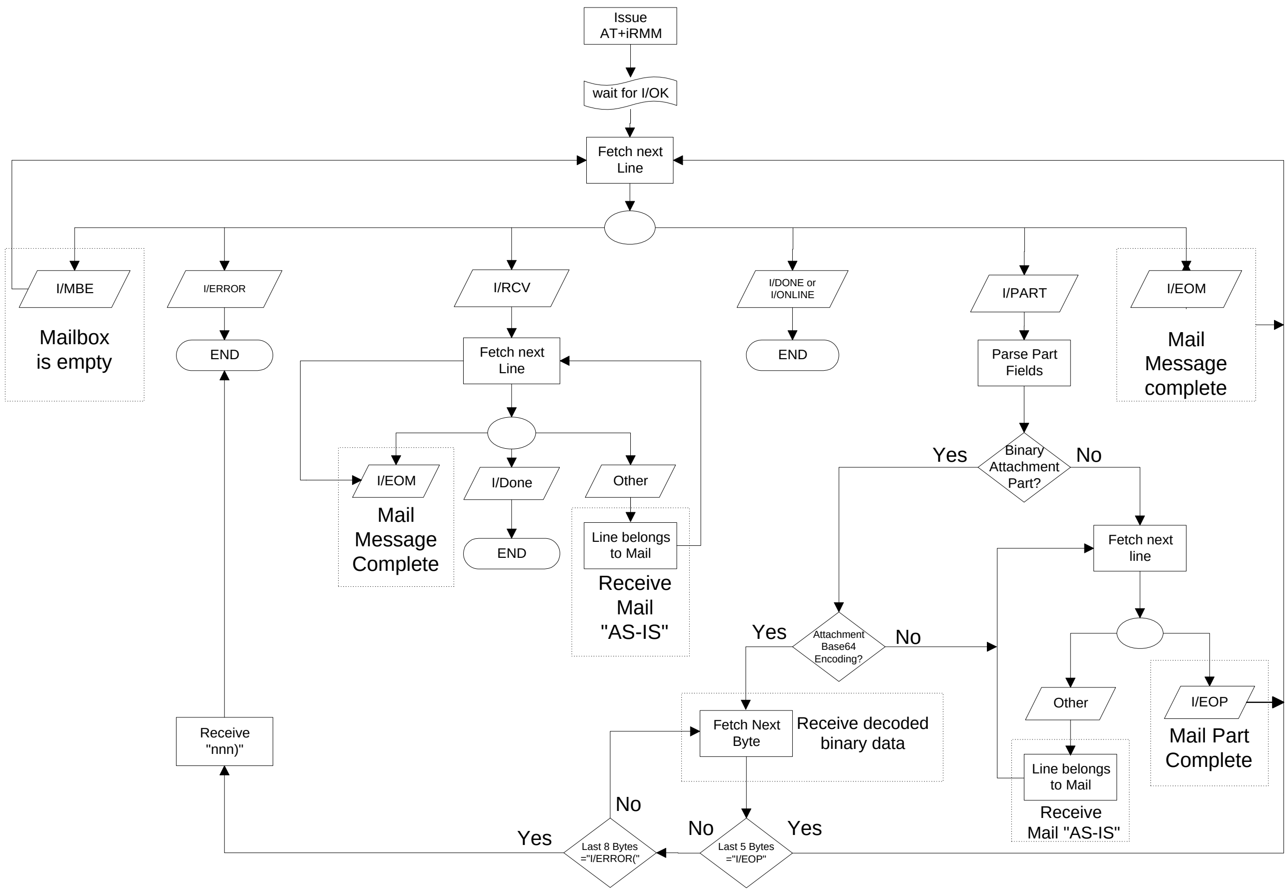


Figure 9-1 E-Mail Receive (RMM) Flow Diagram

10 HTTP Client Interface

+iRLNK — Retrieve Link

Syntax: AT+i[!]RLNK[:*URL*]

Retrieves a file from a URL.

Parameters: *URL* = Optional URL address, which specifies the host, path, and source file to be retrieved.

URL address syntax:

"<*protocol*>://<*host*>[:<*port*>]/[<*abs_link*>]"

Command Options:

<*protocol*> http or https

<*host*> Host name or IP address

<*port*> 0..65535

If not specified, defaults to 80 for http and 443 for https.

<*abs_link*> Path, filename, and file extension of the file to retrieve on the designated host.

! Stay online after completing the command.

Default: Uses the URL address stored in the URL parameter.

Result Code:

I/OK When command is received and about to be processed.

I/ERROR Otherwise

Returns: I/<*sz*><CR><LF>

Followed by: <*binary data stream*>

where,

<*sz*> is the exact size of the <*binary data stream*> to follow.

If <*sz*> is unknown, iChip returns **I/0** followed by the data stream. When this is the case, the host must monitor for a timeout condition of a few seconds without any data being transmitted before seeing one of the terminator lines described under 'Followed by'. The timeout is set in the LDLY parameter.

Followed by:

I/DONE After successfully retrieving the file. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.

-or-

I/ONLINE After successfully retrieving the file, if the stay online flag (!) is specified.

-or-

I/ERROR Otherwise. (Always preceded by a 5 seconds silence period.)

+iSLNK — Submit A POST Request to A Web Server

Syntax: AT+i[!]SLNK:<*text*>

Submits a plain text POST request to a web server defined in the URL parameter. The "Content-type:" field of the POST request is defined by the CTT parameter.

Parameters: <*text*> = Plain text POST request body containing <CR[LF]> terminated ASCII character strings. <*text*> must be terminated by a dot character (.) in the first column of an otherwise empty line.

Command Options:

<*text*> <*ASCII text line*><CRLF> …<CRLF>.<CRLF>

Maximum size of <*text*> depends on the amount of memory available in the specific iChip. SLNK uses the URL address stored in the URL parameter to send the POST request.

! Stay online after completing the command.

Result Code:

I/OK After all text lines are received from the host.

I/ERROR If a memory overflow occurred before all text lines are received.

Returns: I/<*sz*><CR><LF>

Followed by: <*binary data stream*>

where,

<*sz*> is the exact size of the <*binary data stream*> to follow.

If <*sz*> is unknown, iChip returns **I/0** followed by the data stream. When this is the case, the host must monitor for a timeout condition of a few seconds without any data being transmitted before seeing one of the terminator lines described under 'Followed by'. The timeout is set in the LDLY parameter.

Followed by:

I/ONLINE After successfully retrieving the updated web page, if the stay online flag (!) is specified.

-or-

I/ERROR Otherwise. (Always preceded by a 5 seconds silence period.)

11 iChip Embedded Web Server

Introduction

iChip includes a web server that handles HTTP 1.0/1.1 web interactions independently of its host processor. It allows system designers to build web-based products, which can be remotely monitored, configured, and managed via the Internet using a standard web browser interface.

iChip devices host two on-chip websites stored in non-volatile memory. One website is inherent to the iChip firmware and dedicated to iChip configuration and maintenance. The second site is uploaded to iChip for device application use. This website can include multiple linked HTML pages, links to external pages, images, graphics, Java applets, WAP pages, and more. A special facility allows the web pages to include references to the embedded application's variables.

iChip's embedded web server is designed to integrate with the existing iChip-to-host API methodology based on Connect One's AT+i command interface.

Features

- Responds to standard web browser GET and POST commands issued on port 80.
- Supports up to three concurrent remote browsers.
- Serves on-chip HTML pages stored in non-volatile memory.
- Can incorporate WAP pages to allow browsing iChip's website using an Internet-enabled cellular handset.
- The internal iChip configuration website supports remote iChip parameter configuration, remote iChip firmware upload, and remote application website upload. This is achieved using a standard web browser. Configuration access is protected by an SHA1-encrypted password mechanism.
- Supports monitoring and controlling the host device using a pre-defined set of parameters embedded within the application website (also SHA1 password protected).
- Allows OEMs to design their own embedded website using standard web authoring tools along with Connect One's windows-based website packing utility.
- Clear text HTTP server or SSL3 encrypted HTTPS server.
- Web server ports 80 and 443 can be changed.

Web Server Modes

Two web server modes are defined as (see figure below):

- iChip configuration mode
- Host interaction mode

Each of these modes is supported by a dedicated website and a parameter access password.

The iChip configuration mode allows remote iChip configuration. It encompasses web interactions between iChip and a remote browser to carry out iChip parameter maintenance and iChip firmware and application website uploads. The host processor does not take part in the interactions under this mode. Moreover, the host processor is not required at all for this mode to operate. Once an iChip is online and in possession of an IP address, any remote browser may surf to the iChip and update its non-volatile parameters without the host's involvement. The iChip configuration site is located at:

HTTP://<*iChip_IP_Address*>/ichip

In Host interaction mode, iChip is used to host, serve, and manage web interactions with a remote web browser on behalf of the embedded device's host processor. The host gains access to the web-based parameters via AT+i commands sent to iChip through the serial connection.

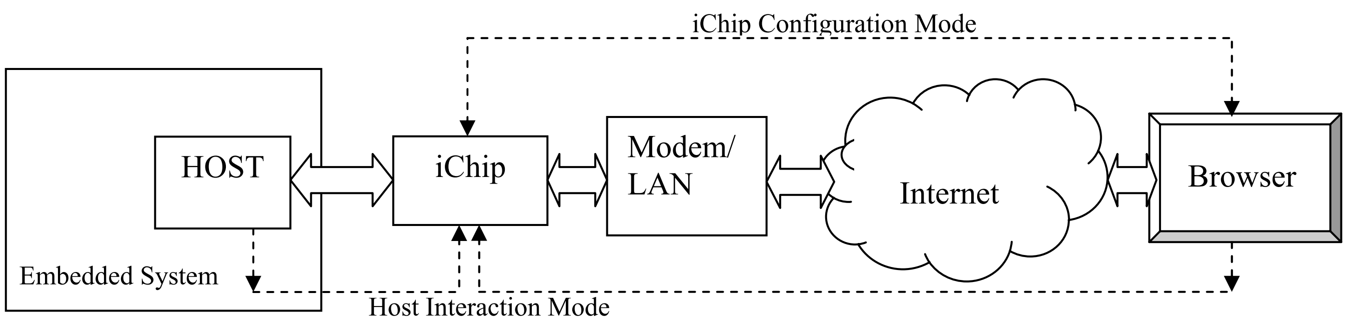


Figure 11-1: iChip Web Server Modes

The Application Website

The application website is stored in non-volatile memory. It consists HTML code, which can include links to local or remote web pages, graphic images, text files, Java applets, WAP pages, and more.

Device manufacturers can design their own embedded website using any web authoring tool. The iChip implementation supports a maximum website size of 256K. The site is uploaded to iChip through the serial connection, or through iChip's configuration website.

Parameter Tags

iChip and host real-time parameters can be referred to in the embedded websites through the use of Parameter Tags. When Parameter Tags are placed in an HTML web page, actual values are sent by iChip's web server component when the page is served out. Parameter Tags are also used to change corresponding parameter values from a remote web browser. Syntactically, Parameter Tags are parameter names enclosed between two (~) characters. If the (~) character needs to be included in a Web page, two consecutive (~) characters must be used (~~).

The iChip Internet configuration parameters defined in the AT+i API retain their name when used as Parameter Tags. For example, the value of the *TOA* AT+i parameter (Send to E-Mail Address) may be referenced in the website by ~*TOA*~.

Host Parameter Tags defined by the parameter name <*param*>, may be referenced in the website using ~<*param*>~. <*param*> can be any freeform parameter name consisting of a single word that does not include blanks or iChip delimiters. For example, a parameter reflecting a temperature reading can be called *temperature* and referenced in the website as ~*temperature*~.

iChip Configuration Mode

iChip configuration entails monitoring and updating iChip parameter values. By making use of iChip's inherent configuration website, an iChip device can be configured remotely using a standard web browser. The iChip RPG parameter is used to password-protect remote iChip parameter updates. See Security and Restrictions.

The configuration site includes web forms to monitor and update most iChip parameters and an upload page consisting of file upload forms. Note that, the following iChip parameters *cannot* be configured remotely and are therefore not displayed on iChip's configuration website:

- Fast USART parameter (BDRD)
- Analog-to-digital converter (ADC) parameters

The following iChip parameters can be viewed and configured remotely *only when* viewing the website securely via its HTTPS address:

- Website protection passwords: RPG and WPWD
- WiFi security parameters

Each upload form allows file uploading using the POST method for a single file. The forms support uploading the following files:

- Firmware update *.imz file
- Parameters update *.rpf file
- Packed application website *.img file

When new firmware (*.imz file) is uploaded to iChip, iChip submits an acknowledgment page to the browser, after receiving the complete *.imz file, and then goes offline and updates its firmware.

In some rare cases, iChip's internal configuration website may be accidentally corrupted. This happens when iChip fails to complete a remote firmware update process via web. To resolve this problem, iChip includes a recovery website. This website allows a user at the remote browser end to upload the .imz file again in order to restore iChip's internal website.

The LOGO image which is shown in the top frame of the configuration website can be changed, from the default Connect One LOGO, to a customized LOGO, using the command: AT+iLOGO.

iChip's configuration site is located at:

HTTP://<*iChip_IP_Address*>/ichip

Host Interaction Mode

Host Interaction mode allows OEMs to design and implement a product-related embedded website that is managed by iChip on behalf of the host. The host-defined embedded website supports live host parameter monitoring and updating by a remote browser. This is achieved by a dynamic AT+i layer implemented across the serial link between the host and iChip.

The application developer creates a website using conventional web authoring tools. The HTML or WAP files can then be edited to contain Parameter Tags. Parameter tags are regarded as placeholders in HTML or WAP files. They are replaced on-the-fly with real-time values as the page is served to the browser. Browsers may also change values of Parameter Tags in order to submit the value back to the host via iChip. This is done by defining the Parameter Tag in the NAME field in an HTML FORM (without the (~) characters). The iChip WPWD parameter is used to password-protect remote Parameter Tags update. See Security and Restrictions.

Once a website is created and Parameter Tags are edited in, the site is packed and uploaded to iChip. The website is linked into the iChip firmware, automatically expanding the existing AT+i command set to encompass the website Parameter Tags. This happens when the web server is activated using the +iWWW command or the AWS parameter.

Extended AT+i commands have the following syntax:

```
> AT+i<param>=<value>
> AT+i<param>?
```

for setting and querying Parameter Tag values, respectively.

For example, the ~temperature~ Parameter Tag referenced in a web page, can be set using:

```
> AT+itemperature='45 Deg.'
```

and queried using:

```
> AT+itemperature?
```

When the host issues a Set Parameter Tag Value command, iChip links the updated value to the Parameter Tag and stores it in its internal RAM. In response to a browser's GET request, the real value is substituted everywhere in the page where the Parameter Tag exists while the page is being served, on-the-fly.

Parameter Tag values are printable ASCII text. This convention allows implementing any part of an HTML or WAP page as a parameter tag: numeric values, links, file names, HTML code, etc. A Parameter Tag value is limited to 256 characters.

Parameter Tag values can be changed and submitted from the browser end using HTML forms. iChip stores the updated values and responds appropriately to host AT+i parameter query commands. Thus, the host can poll specific parameters for value changes. Status Report 7 (AT+iRP7) can be used to facilitate polling on all application web parameters. RP7 returns a bitmap result, where bit 10 is set to '1' if ***one or more*** application web parameters have been remotely changed. The iChip DATA_RDY signal is an associated hardware signal that can be used to generate an interrupt on the host CPU when new data has been buffered in iChip. The ISR can issue an RP7 to determine if the new data is a result of an application web parameter change.

The AT+iWNXT command can be issued to scan through the application web parameters that have been remotely updated and not yet retrieved by the application.

The iChip application site is located at:

HTTP://<*iChip_IP_Address*>/

Website Creation, Packing, and Uploading

Device manufacturers can design their own embedded website using any typical web authoring tool. A website can include one or more files residing in a dedicated file directory structure on the designer's PC. The topmost directory of this structure is referred to as the website *root* directory. The root directory must contain an HTML page named index.htm, which serves as the default home page.

Before downloading the website to an iChip device, the entire website needs to be packed. In order to pack the site into an uploadable image file, the designer must run Connect One's web packing utility and specify the root directory of the site. The utility packs all files in the root directory and its subfolders in a format suitable for iChip. If the site contains Parameter Tags, the user is prompted to enter a maximum value length for each Parameter Tag. Any Parameter Tag specified with a zero length value will not be included in the resulting packed file. After the user has entered all parameters' max value length, the user is prompted to specify a destination for the packed file.

The following restrictions apply when creating the packed website:

- The length of a single Parameter Tag must not exceed 256 characters.
- The sum of all Parameter Tags' value lengths must not exceed 8K.
- The total packed file must not exceed 256K.

To take effect, the packed website file needs to be uploaded to iChip. This is done through iChip's configuration website over the Internet.

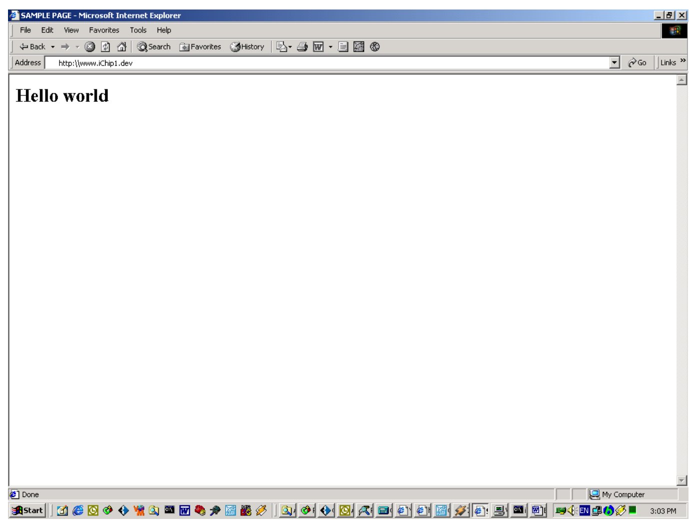
Manipulating Variables in the Application Website

The application website is composed of HTML or WAP files, which may contain links to internal or external websites, Java Scripts, VB scripts, graphic files, and more (See list of supported file types). Using Parameter Tags, the page can also be used to dynamically display and update values of iChip's configuration parameters and device-specific Parameter Tags in the manner described above.

For example, to display the current value of the *headline* web parameter, enter ~*headline*~ anywhere on the page, as in the following example:

```
<HTML>
<HEAD>
<TITLE>SAMPLE PAGE</TITLE>
</HEAD>
<BODY>
<h1>~headline~</h1>
</BODY>
</HTML>
```

When serving this home page, iChip's web server replaces the ~*headline*~ string in the served page with the current value of that parameter.



For example, if the host issues:

```
AT+iheadline="Hello world"
```

a browser pointing to iChip's URL address will display the image as seen on the right.

To update iChip configuration parameters via the web page, simply use iChip's parameter names (excluding the AT+i prefix) in an HTML form.

For example:

```
<HTML>
<HEAD>
<TITLE>SAMPLE PAGE</TITLE>
</HEAD>
<BODY>
<FORM METHOD='GET' ACTION=''>
Dial To:<INPUT type='text' name='ISP1' value='~ISP1~'>
<input type='submit' size='8' value='Submit'>
</FORM>
```

```
</BODY>
</HTML>
```

Note that the variable name is used in the NAME field, while ~<*parameter name*>~ is used to display the current value.

After activating SUBMIT, the browser issues a GET command to iChip's web server that includes the parameter's name and the new value entered in the form. The page is then served to the browser again with the updated values.

In addition to specifying iChip configuration parameters and Parameter Tags, it is also possible to display iChip reports and iChip's LAN MAC address. For example:

```
<table>
      <tr>
<td width=250><b>MAC Address: ~MACA~ <b></td>
     </tr>
      <tr>
<td width=250><b>Bootblock Version: BBIC~RP3~</b> </td>
<td width=400><b>Firmware Version: ~RP1~</b></td>
      </tr>
      <tr>
<td width=250><b>Serial Number: ~RP5~ </b></td>
<td width=400><b>Hardware Version: ~RP0~</b></td>
     </tr>
</table>
```

Security and Restrictions

The authorization to view and update iChip's configuration parameters, firmware, or application website via the web can be password-protected using the AT+iRPG parameter (Remote Parameter Group/Password).

When the RPG parameter in an iChip device contains a value, it is considered a password that restricts remote iChip parameter viewing/updates. By default, iChip's configuration site can be viewed (browsed) and the distant user is prompted for a password only when changing iChip parameter values. However, if the Security Disable Mode (SDM) bit 2 is set, the user is authenticated prior to viewing the website, by submitting the RPG value. If the Security Disable Mode (SDM) bit 5 is set, the configuration website is disabled. The iChip configuration site includes an authentication form that automatically pops up on the remote browser when parameter updates are attempted. The password submitted through this form must match the actual value of iChip's local RPG parameter. Otherwise, remote value updates are rejected.

iChip uses the industry standard SHA1 algorithm to authenticate the remote user. According to SHA1, the password typed into the authentication form is not literally communicated back to iChip. Rather, a SHA1-encrypted token is transferred. To achieve this, iChip's web server sends a JavaScript, which calculates SHA1 encryption at the browser end together with the authentication form. iChip also issues a different random number, used as part of the encryption key, each time authentication is required, to eliminate the possibility of impersonation based on eavesdropping to a legal authentication session.

If the RPG parameter is empty (AT+iRPG=''), remote iChip configuration parameter update is fully restricted. In other words, it is not possible to update configuration parameter values using a remote browser. Conversely, if the RPG parameter contains an (*) character (match any), the configuration parameters can be updated freely, without requiring authentication at all.

The Parameter Tags defined in the application website are secured from remote updates in the same manner as the iChip configuration parameters. In this case, the authentication password is stored in iChip's local parameter WPWD (Web Password). If the WPWD parameter contains a value, a remote user needs to issue this value as an authentication password in order to gain update access to the application level Parameter Tags. Like in the case of the RPG parameter, if WPWD is empty, application level Parameter Tags are fully restricted, whereas when WPWD contains an (*), updates are unrestricted and authentication is not required.

When authentication is required, iChip's web server automatically issues an authentication form to the remote browser in response to an attempt to update Parameter Tags. This procedure allows the application site to include HTML submit instances anywhere in the website without worrying about the authentication process. Authentication is automatically activated depending on the local value of the WPWD parameter.

Authentication needs to be submitted only once per session in order to enable browsing, Parameter Tags, or iChip configuration updates. In addition, authentication automatically expires after 10 minutes of inactivity.

Secure (HTTPS) Web Server

iChip's web server can be configured to be a Secure (SSL3 over HTTP 1.1) Web server, which listens on TCP port 443. During the SSL3 Handshake the SSL server sends a Server Hello message (similar to the client hello) and a certificate. That means that the server needs to hold a private key and a certificate signed by a CA. The server will request a client certificate (client authentication) only if the CA parameter holds a non empty value.

The CERT and PKEY parameters are used as server parameters. The implications are that client authentication on SSL3 connections (as SSL client) will use the same certificate and private key as the HTTPS server. Similarly, the same CA used for SSL3 client connections, will be used by the HTTPS server as well.

Some restrictions are imposed by memory limitations on the CO2128 and CO2144 iChips:
When an SSL client socket is Active – the HTTPS Web server may be enabled but an HTTPS client will not be able to connect, until the SSL3 socket is closed.

When the HTTPS server is enabled, an SSL3 client socket may be connected only if a remote client is not connected to the HTTPS server.
SSL3 connections should normally support up to a maximum of 16KB receive buffers. However, iChip's receive buffers have been allocated to 9KB. HTTPS GET/POST requests to the server should not be in excess of 9KB. If a record larger than 9KB is received, an SSL ALERT (SSL_ALERT_UNEXPECTED_MESSAGE) will be sent, and the SSL connection will be closed.

Parameter Update Error Handling

An attempt to assign an illegal value to a parameter will fail and a string containing the relevant error message will be stored in a special iChip Parameter Tag named WST (Web Server Status). This value can be displayed in the page as any other parameter value (using ~*WST*~). For Example:

```
<b>Update Error Message: ~WST~</b>
```

File Types Supported by iChip's Web Server

- The following files can include parameter tags:
  .HTM, .HTML, .JS, .VBS, .INC, .STM, .XML, .XSL, .HTC,.CSS, .WML, .WMLS, .XHTML
- The following files cannot include parameter tags:
  .CLASS, .GIF, .JPG, .PDF, .DOC, .PPT, .BMP, .XLS, .WMLC, .WMLSC, .WBMP

12 Web Server Interface

+iWWW — Activate Embedded Web Server

Syntax: AT+iWWW[:*n*]

Activates iChip's internal web server.

Parameters: <*n*>=Web browser backlog. *n* represents the number of browsers that can connect to iChip's internal web server simultaneously at any given time.

Command Options:

<*n*>=0 Deactivate iChip's internal web server. All HTTP/HTTPS connections shall be closed and the Listen task shall be removed.

<*n*>=1..3 1, 2 or 3 browser connections can connect simultaneously to iChip's TCP port 80, or as defined in the WEBP parameter, implementing HTTP protocol.

<*n*>=100 One browser connection can connect to iChip's TCP port 443, or as defined in the WEBP parameter, implementing HTTPS protocol.

Default: <*n*>=1

Returns: **I**/(<*Local IP addr*>)

where,

<*Local IP addr*> is the iChip local IP address.

Note: If the web server is already open, then **I**/(<*Local IP addr*>) is returned without any action taken.

In a dial-up environment, iChip goes online and the <*local IP addr*> is assigned dynamically by the ISP.

In an LAN environment, the IP address is assigned by a DHCP server or configured by the DIP parameter.

I/ERROR If connection to the Internet failed or if missing SSL parameters prevent launching of the secure web server.

+iWNXT — Retrieve Next Changed Web Parameter

Syntax: AT+iWNXT

Retrieves the Parameter Tag name and new value of the next changed application web parameter, which has not been retrieved since it has been changed by the remote browser.

Returns: <*Parameter Tag*>=<*New Value*> <CR><LF>

When there are no more remaining changed parameters, a blank <CR><LF> terminated line is returned.

Followed by:

I/OK

13 File Transfer Protocol (FTP) Theory of Operation

Introduction

The FTP client component in iChip extends iChip's general-purpose sockets to incorporate an additional, dedicated socket for FTP activities. From the host's perspective, the FTP capabilities are a logical extension of the capabilities of e-mail and direct socket manipulation.

As in all other iChip protocol implementations, host involvement in the specifics of FTP is minimal. iChip needs to deal with non-standard FTP issues, such as possible differences between FTP server responses, on its own. Multi-stage FTP protocol sequences are atomized under iChip control to minimize complexity and need for host processor intervention.

The FTP protocol is described in RFC 959.

iChip Family FTP Client Command Set

- Open FTP link to FTP Server
- Open SSL-Encrypted FTP link to FTP server
- Retrieve File List from Server
- Change Directory on Server
- Retrieve File Contents from Server
- Open a New File on Server
- Open an existing File on Server for Append
- Send Binary Data to an open File on Server
- Close a File on Server After Binary Data Send
- Delete File on Server
- Close FTP Session

iChip FTP Client Operation Mode

FTP specifies several operational modes. The RFC calls for a minimum implementation, which should be observed by all FTP servers. iChip restricts its operation mode to the minimum implementation to assure best intersystem compatibility.

Character Types:	ASCII Non-print
Structure:	File
Mode:	Stream

FTP Command Socket

The FTP command socket is normally on port 21 (decimal) of an FTP server. However, other ports can be specified to support special cases.

FTP Receive Flow

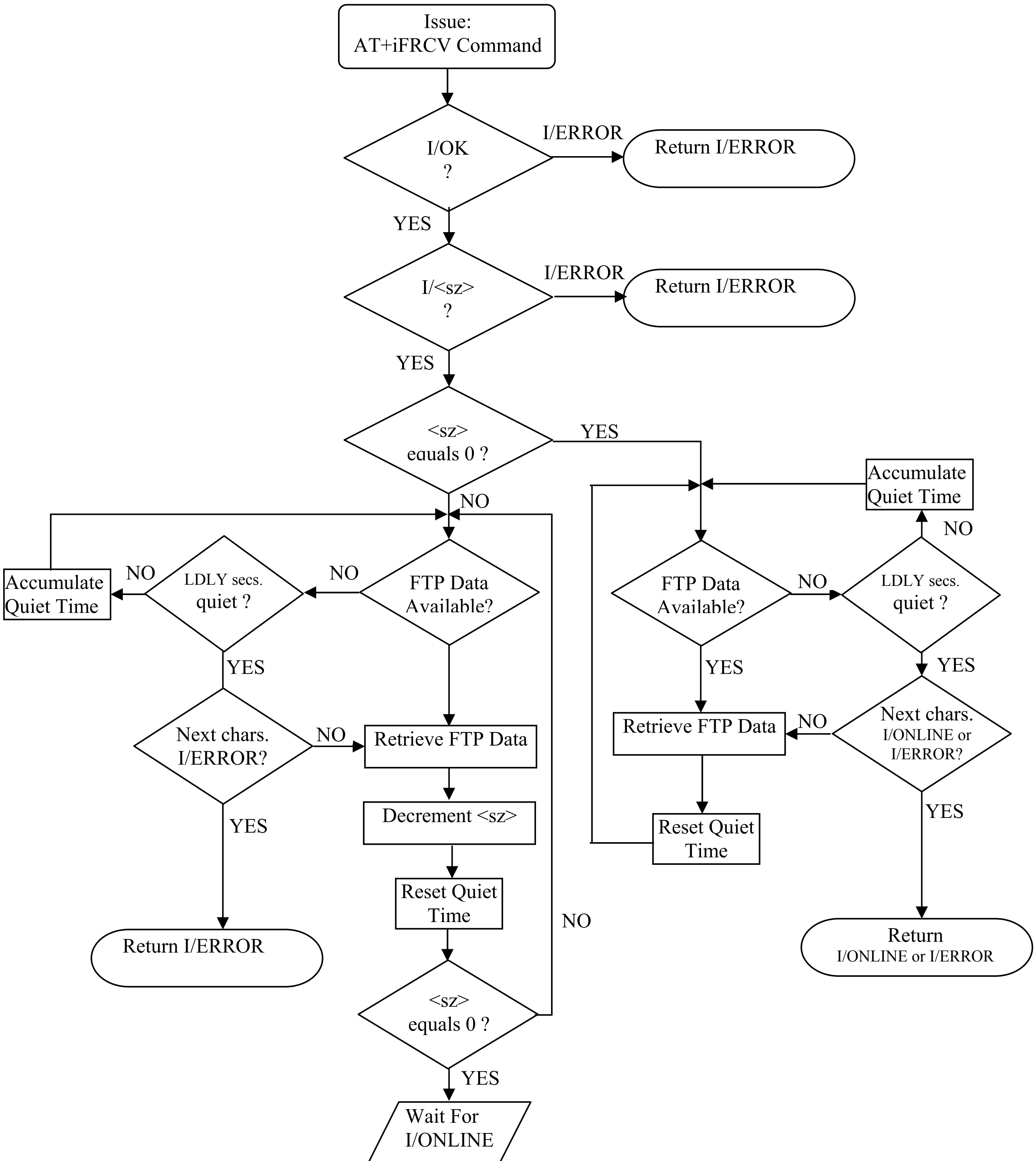


Figure 13-1 FTP Receive Flowchart

14 File Transfer Protocol (FTP)

+i[@]FOPN — FTP Open Session

Syntax:	AT+i[**@**]FOPN:<*server*>[,<*port*>]:<*user*>,<*pass*>[,<*accnt*>]
	Opens an FTP link to an FTP server.
Parameters:	
<*server*>	Logical name of the FTP or the server's IP address.
<*port*>	Optional FTP port in the range 0..65535.
<*user*>	FTP user's name
<*pass*>	FTP user's password
<*accnt*>	Optional FTP account
Command Options:	
<*server*>	The server name may be any legal Internet-server name, which can be resolved by the iChip's DNS (Domain Name Server) settings. The server name may also be specified as an absolute IP address given in DOT form.
<*port*>	Specifies the FTP server's listening port. If not specified, port 21 (decimal) is assumed.
<*user*>	User's name string. This must be a registered user on the FTP server. Some servers allow anonymous login, in which case *user*=anonymous.
<*pass*>	Password to authenticate user. If special characters are used, the password must be specified within quotes. It is customary that servers that allow anonymous login request an e-mail address as a password.
<*accnt*>	Some FTP servers require an account in order to allow a certain subset of the commands. In this case, the account name must be specified when opening the FTP link.
@	The optional @ is used to flag the Force PASV mode. When @ is specified, iChip only uses the PASV method when opening a data socket to *server* for FTP data transfer.
Result Code:	
I/<*FTP handle*>	Upon successfully connecting to the FTP Server and authenticating the user, a socket handle is returned. The handle <*FTP handle*> is used to reference the FTP session in all following FTP commands.
I/ERROR	Otherwise

+iFDL — FTP Directory Listing

Syntax: AT+iFDL:<*F_hn*>[,<*path*>]

Returns a full FTP directory listing.

Parameters:

<*F_hn*> An open FTP session handle

<*path*> Directory or filename wild card

Command Options:

<*F_hn*> Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.

<*path*> Optional directory name or filename wild card. If <*path*> is a directory, that directory's files are listed. If it is a filename wild card, only matching filenames in the current directory are listed.

If <*path*> is not specified, the current directory is listed in full.

Result Code:

I/OK To acknowledge successful receipt of the command.

I/ERROR If <*F_hn*> is not an open FTP session or otherwise some error has occurred.

Returns: A list of filenames with file attributes. Each file is listed on a separate line, terminated by <CR/LF>. The file data line syntax is FTP server-dependant.

Followed by:

I/ONLINE After successfully retrieving the directory list.

+iFDNL — FTP Directory Names Listing

Syntax: AT+iFDNL:<*F_hn*>[,<*path*>]

Returns the FTP directory name list.

Parameters:

<*F_hn*> An open FTP session handle

<*path*> Optional directory or filename wild card

Command Options:

<*F_hn*> Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.

<*path*> Optional directory name or filename wild card. If <*path*> is a directory, that directory's files are listed. If it is a filename wild card, only matching filenames in the current directory are listed.

If <*path*> is not specified, the current directory is listed in full.

Result Code:

I/OK To acknowledge successful receipt of the command.

I/ERROR If <*F_hn*> is not an open FTP session or otherwise some error has occurred.

Returns: A bare list of filenames. Each file name is listed on a separate line, terminated by <CR/LF>. No attributes are returned in addition to the filename.

Followed by:

I/ONLINE After successfully retrieving the directory list.

+iFMKD — FTP Make Directory

Syntax:	AT+iFMKD:<F_hn>,<path>
	Creates a new directory on the FTP server's file system.
Parameters:	
<F_hn>	An open FTP session handle
<path>	Directory pathname
Command Options:	
<F_hn>	Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.
<path>	Directory name. A new directory will be created under the current directory, as indicated by *path*. If path includes nonexistent subdirectories, some FTP servers will create them as well.
Result Code:	
I/OK	To acknowledge successful completion of the command.
I/ERROR	If *<F_hn>* is not an open FTP session or otherwise some error has occurred.

+iFCWD — FTP Change Working Directory

Syntax:	AT+iFCWD:<*F_hn*>,<*path*>
	Changes the current FTP working directory.
Parameters:	
<*F_hn*>	An open FTP session handle
<*path*>	New directory path name
Command Options:	
<*F_hn*>	Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.
<*path*>	Absolute or relative path name of the new directory. The special directory ".." signifies "one directory up".
Result Code:	
I/OK	After successfully changing the working directory.
I/ERROR	Otherwise

+iFSZ — FTP File Size

Syntax: AT+iFSZ:<*F_hn*>,<*path*>

Reports an FTP file size.

Parameters:

<*F_hn*> An open FTP session handle

<*path*> File pathname

Command Options:

<*F_hn*> Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.

<*path*> Absolute or relative path name of the remote file.

Result Code:

I/<*file size*> iChip reports *path*'s file size in bytes if the file exists and the FTP server supports the file size FTP command. Followed by: **I/OK**.

I/ERROR Otherwise

+iFRCV — FTP Receive File

Syntax:	AT+iFRCV:<*F_hn*>,<*path*>
	Downloads a file from an FTP server.
Parameters:	
<*F_hn*>	An open FTP session handle
<*path*>	File pathname
Command Options:	
<*F_hn*>	Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.
<*path*>	Absolute or relative path name of the remote file.
Result Code:	
I/OK	When command has been received and about to be processed.
I/ERROR	If <*F_hn*> is not an open FTP session or otherwise some error has occurred.
Followed by:	
I/ERROR	If the FTP RECV command could not be processed.
-or-	**I/**<*sz*><CR><LF>
Followed by:	<*data stream*>
	where,
	<*sz*> is the exact size (in bytes) of the <*data stream*> to follow. If <*sz*> cannot be determined, iChip returns **I/0** followed by the data stream. When this is the case, the host must monitor for a timeout condition of a few seconds without any data being transmitted before seeing the **I/ONLINE** to deduce that the data stream is complete. The timeout is set in the LDLY parameter.
	If <*sz*> was reported but a transmission error occurred, preventing the iChip from returning all <*sz*> data bytes — an **I/ERROR** command is issued after a 5 seconds non-transmission period. See FTP Receive Flow Diagram.
Followed by:	
I/ONLINE	After successfully retrieving file contents.

+iFSTO — FTP Open File for Storage

Syntax: AT+iFSTO:<*F_hn*>,<*path*>[,<*sz*>]

Opens a remote FTP server file for upload.

Parameters:

<*F_hn*> An open FTP session handle

<*path*> Destination file pathname

<*sz*> Optional size in bytes to reserve for the file on the remote FTP server

Command Options:

<*F_hn*> Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.

<*path*> Absolute or relative path name of the remote destination file.

Following this command data is transferred to the remote file using one or more +iFSND commands. The file transfer is complete by issuing a +iFCLF (FTP File Close) command.

Result Code:

I/OK If file <*path*> was successfully opened for writing on the FTP server.

I/ERROR Otherwise

+iFAPN — FTP Open File for Appending

Syntax:	AT+iFAPN:<*F_hn*>,<*path*>[,<*sz*>]
	Opens an existing remote FTP server file for Append.
Parameters:	
<*F_hn*>	An open FTP session handle
<*path*>	File pathname
<*sz*>	Size in bytes to reserve for the file on the server
Command Options:	
<*F_hn*>	Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.
<*path*>	Absolute or relative path name of the remote destination file.
	Following this command data is transferred to the remote file using one or more +iFSND commands. The file transfer is complete by issuing a +iFCLF (FTP File Close) command.
Result Code:	
I/OK	If file <*path*> was successfully opened for appending on the FTP server.
I/ERROR	Otherwise

+iFSND — FTP Send File Data

Syntax: AT+iFSND:<*F_hn*>,<*sz*>:<*stream…*>

Uploads data to a remote FTP server file. Valid only after a successful AT+iFSTO or AT+iFAPN command.

Parameters:

<*F_hn*> An open FTP session handle

<*sz*> The exact size of the data stream that follows

<*stream*> A byte stream of size <*sz*> composing the remote file contents

Command Options:

<*F_hn*> Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.

<*stream*> An 8-bit byte stream of exactly size <*sz*>. If <*sz*> is larger than 256 bytes, iChip assumes host flow control. Depending on the setting of the FLW parameter, the flow control mode is either software or hardware. Under software flow control mode, the host processor must respond to iChip's flow control characters. The flow control protocol is detailed in the "Host ➔ iChip Software Flow Control" section later in this document. When software flow control is active, it is recommended to set iChip to Echo-Off mode.

Under hardware flow control, the ~CTS/~RTS RS232 control signals must be connected and the host must respond to iChip's ~CTS signal. The host may send data only when the ~CTS signal is asserted (active low).

Several consecutive +iFSND commands may be issued in sequence to create a larger aggregate of data to be sent.

The file transfer is complete by issuing a +iFCLF (FTP Close File) command.

Result Code:

I/OK After <*sz*> bytes have been transferred successfully to the FTP data socket.

I/ERROR Otherwise

+iFCLF — FTP Close File

Syntax: AT+iFCLF:<*F_hn*>

Closes a file downloaded to a remote FTP server. Only valid after a successful AT+iFSTO or AT+iFAPN command and optional AT+iFSND commands.

Parameters:

<*F_hn*> An open FTP session handle

Command Options:

<*F_hn*> Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.

Result Code:

I/OK After successfully closing the file.

I/ERROR Otherwise

+iFDEL — FTP Delete File

Syntax: AT+iFDEL:<*F_hn*>,<*path*>

Deletes a remote FTP file.

Parameters:

<*F_hn*> An open FTP session handle

<*path*> File pathname

Command Options:

<*F_hn*> Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.

<*path*> Absolute or relative pathname of the remote destination file to delete.

Result Code:

I/OK After successfully deleting the remote file.

I/ERROR Otherwise

+iFCLS — FTP Close Session

Syntax:	AT+i[!]FCLS:<*F_hn*>
	Closes the FTP link.
Parameters:	
<*F_hn*>	An open FTP session handle
Command Options:	
<*F_hn*>	Must have been obtained by a previous execution of an AT+iFOPN command during the current Internet mode session.
!	Stay online after completing the command
Result Code:	
I/OK	When command has been received and about to be processed.
Followed by:	
I/DONE	When the FTP link was the last open socket and after successfully closing the FTP link. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.
-or-	
I/ONLINE	After successfully closing the FTP link.
	When connected to a modem: After successfully closing the FTP link, when additional sockets are still active or the stay online flag (!) is specified.
-or-	
I/ERROR	Otherwise

15 Telnet Client Operation

There are four operation modes for most Telnet applications, namely, half-duplex, character at a time, line at a time, and line mode.

iChip incorporates two methods to send data to the remote Telnet server: One line at a time, namely, an AT+i command (+iTSND) is used to send a single (CR/LF terminated line to the Telnet server); and Binary Transmission, where an AT+i command (+iTBSN) is used to send an arbitrary amount of binary data.

Data is retrieved from the remote Telnet server as it is made available. Embedded Telnet options in the server's response stream are stripped by iChip before being turned over to the host.

Telnet specifies many operational options. iChip restricts its operation mode to the minimum implementation to assure best intersystem compatibility.

Following are the Telnet options negotiated by iChip:

Option ID	Name	Val	RFC
1	Echo	OFF	857
3	Suppress go ahead	Suppress	858
24	Terminal type	VT100	1091
31	Window size	Whatever	1073

Any other options negotiated by the Telnet server are rejected by iChip.

16 Telnet Client

+iTOPN — Telnet Open Session

Syntax:	AT+iTOPN:<*server*>
	Opens a Telnet link (socket) to a Telnet server on port 23.
Parameters:	
<*server*>	Logical name of the Telnet server or the server's IP address.
Command Options:	
<*server*>	The server name can be any legal Internet Server name that can be resolved by iChip's DNS (Domain Name Server) settings. The server name may also be specified as an absolute IP address given in DOT form.
Result Code:	
I/OK	Upon successfully connecting to the remote Telnet server.
I/ERROR	Otherwise

+iTRCV — Telnet Receive Data

Syntax: AT+iTRCV[:<*max*>]

Receives data from the Telnet server.

Parameters:

<*max*> Optionally specifies the maximum number of bytes to transfer.

Result Code:

I/ERROR If no Telnet session is open or otherwise some error has occurred.

Returns: **I/**<*sz*>[:<*binary data stream*>]

where,

<*sz*> is the exact size of the binary data stream to follow.

If the socket input buffer is empty, iChip returns **I/O**. In this case the (:) and <*binary data stream*> are omitted.

<*sz*> is guaranteed to be equal or less than <*max*>, when specified.

+iTSND — Telnet Send Data Line

Syntax:	AT+iTSND:<*data line*>
	Sends data to the remote Telnet server.
Parameters:	
<*data line*>	A line of data bytes to be sent to the Telnet server. iChip terminates the <*data line*> with a <CR><LF> and sends it to the Telnet server.
Command Options:	
<*data line*>	If the line to be sent incorporates iChip delimiter characters (, ; : ; = ; ~), <*data line*> must be enclosed in single (‘) or double (“) quotes. AT+i command’s terminating <*CR*> is considered a terminating quote, as well.
Result Code:	
I/OK	After the <*data line*> has been successfully sent to the Telnet server.
I/ERROR	Otherwise

+iTBSN[%] — Telnet Send A Byte Stream

Syntax:	AT+iTBSN[%]:<*sz*>:<*stream*>
	Sends a byte stream of size <*sz*> to the Telnet server.
Parameters:	
<*sz*>	The exact size of the byte stream that follows.
<*stream*>	A byte stream of size <*sz*> to be sent to the Telnet server.
Command Options:	
<*sz*>	0..4GB
<*stream*>	An 8-bit byte stream of exactly size <*sz*>. If <*sz*> is larger than 256 bytes, iChip assumes host flow control. Depending on the setting of the FLW parameter, the flow control mode is either software or hardware. Under software flow control mode, the host processor must respond to iChip's flow control characters. The flow control protocol is detailed in the "Host ➔ iChip Software Flow Control" section later in this document. Under hardware flow control, the ~CTS/~RTS RS232 control signals must be connected and the host must respond to iChip's ~CTS signal. The host may send data only when the ~CTS signal is asserted (active low).
%	When the auto-flush ('%') flag is specified, the Telnet socket is automatically flushed immediately after receiving the <*stream*> from the host. Otherwise, data will be transmitted to the Internet only in integral quantities of the specified Maximum Transfer Unit (MTU) or when the AT+iTFSH command is issued.
Result Code:	
I/OK	After <*sz*> bytes have been transferred successfully to the Telnet socket's output buffer.
I/ERROR	Otherwise

+iTFSH[%] — Flush Telnet Socket's Outbound Data

Syntax:	AT+iTFSH[%]
	Flushes (immediately sends) all the data accumulated in a Telnet socket's outbound buffer.
Command Options:	
%	When the flush-and-acknowledge ('%') flag is specified, iChip flushes and waits for the Telnet server receipt acknowledgment of all outstanding outbound data.
Result Code:	
I/OK	If all outbound data has been received and acknowledged by the Telnet server.
I/ERROR	Otherwise

+iTCLS — Telnet Close Session

Syntax:	AT+i[!]TCLS
	Closes the Telnet link.
Command Options:	
!	Stay online after completing the command
Result Code:	
I/OK	If an active Telnet socket exists.
Followed by:	
I/DONE	When the Telnet link was the last open socket and after successfully closing the Telnet link. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.
-or-	
I/ONLINE	After successfully closing the socket.
	When connected to a modem: After successfully closing the Telnet link, when additional sockets are still active or the stay online flag (!) is specified.
-or-	
I/ERROR	Otherwise

17 Direct Socket Interface

+iSTCP — Open and Connect A TCP Socket

Syntax: AT+iSTCP:<*host*>,<*port*>[,<*lport*>]

Opens a Transmission Control Protocol (TCP) client socket and attempts to connect it to the specified <*port*> on a server defined by <*host*>.

Parameters:

<*host*> Logical name of the target server or a host IP address

<*port*> 0..65535, target port

<*lport*> Optional local port on iChip

Command Options:

<*host*> The server name may be any legal Internet server name that can be resolved by iChip's DNS (Domain Name Server) settings. The server name can also be specified as an absolute IP address given in DOT form.

<*port*> It is assumed that the server system is listening on the specified port.

<*lport*> Can be optionally specified to force iChip to use *lport* as the local port when opening the TCP socket. If unspecified, iChip allocates a port from its internal pool[1].

Result Code:

I/<*sock handle*> Upon successfully opening and connecting the TCP socket to the <*host*>:<*port*>, a socket handle is returned. The socket handle <*sock handle*> is in the range 0..9 and used to reference the socket in all following socket commands.

I/ERROR Otherwise

The Socket Command Abort (---) may be used to abort prematurely.

[1]***Note***: iChip uses the port range [1025 .. 2048] when assigning default local ports. The host should refrain from specifying local ports in this range to ensure that Error 218 is not generated as a result of requesting local ports that overlap internal assignments.

+iSUDP — Open A Connectionless UDP Socket

Syntax: AT+iSUDP:<*host*>,<*rport*>[,<*lport*>]

Opens a UDP (User Datagram Protocol) socket and sets the remote system's <*host*>:<*port*> address.

Parameters:

<*host*> Logical name of the target server or a host IP address, or 0.0.0.0 to open a non-connected socket.

<*rport*> Remote port number to send to, or 0 to open a non-connected socket.

<*lport*> Optional local UDP port to use.

Command Options:

<*host*> The remote system's name may be any legal Internet server name that can be resolved by iChip's DNS (Domain Name Server) settings. The server name may also be specified as an absolute IP address given in DOT form. When the <*host*> is defined, the resulting UDP socket is created and connected.

If <*host*>=0.0.0.0, the socket is created but remains unconnected. The first UDP packet to arrive automatically latches the sender's IP port, in effect connecting the socket.

<*host*> may be a subnet directed Broadcast address which allows to broadcast packets to the immediate subnet, not crossing gateways. For example, to broadcast to subnet 192.168.x.x on destination port 1234: AT+iUDP:192.168.255.255,1234

<*host*> may be a multicast IP address in the range 224.0.0.0 to 239.255.255.255. IP multicast datagrams will not cross gateways. Data is sent and received on <*rport*>. <*lport*> is ignored.

<*rport*> Specifies the remote system's port.

<*lport*> Specifies the local port to use. If unspecified, iChip allocates a port from its internal pool.

Result Code:

I/<*sock handle*> Upon successfully opening and connecting the UDP socket to <*host*>:<*port*>, a socket handle is returned. The socket handle <*sock handle*> is in the range 0..9 and used to reference the socket in all following socket commands.

I/ERROR Otherwise

The Socket Command Abort (---) may be used to abort prematurely.

+iLTCP — Open A TCP Listening Socket

Syntax: AT+iLTCP:<*port*>,<*backlog*>

Opens a TCP listening socket on the local IP address and the specified port <*port*>. The <*backlog*> parameter specifies the maximum number of remote concurrent connections allowed through the listening socket.

Parameters:

<*port*> 0..65535

<*backlog*> 1..10

Command Options:

<*port*> Listening port to be used by a remote system when connecting to iChip.

<*backlog*> Specifies the maximum number of active connections that may be concurrently established through the listening socket.

Once the listening socket is open, it automatically *accepts* remote *connect* requests up to the maximum allowed. When a remote system connects through the listening socket, a new TCP socket is spawned internally ready to send and receive data. See the AT+iLSST command for details on retrieving the handles of active sockets connected through a listening socket. When a connected socket is closed by the host using the AT+iSCLS command, the listening socket allows a new connection in its place.

Result Code:

I/<*sock handle*> Upon successfully opening a TCP listening socket, a socket handle is returned. The socket handle <*sock handle*> is in the range 10..11 and used to reference the socket in all following socket commands.

I/ERROR Otherwise

+iLSST — Get A Listening Socket's Active Connection Status

Syntax: AT+iLSST:<*hn*>

Retrieves handles of active socket connections established through the listening socket identified by <*hn*>.

Parameters:

<*hn*> A TCP listening socket handle of an open listening socket.

Command Options:

<*hn*> Must have been obtained by a previous AT+iLTCP command during the current Internet session.

Result Code:

I/($<hn_1>$,…,$<hn_{Backlog}>$) A list of active socket handles. The list contains <*backlog*> elements, where <*backlog*> was used when opening the listening socket identified by <*hn*>.

Where,

$<hn_i>$ >=0 : A handle to an active connected socket

=-1 : No connection has been established

I/ERROR If <*hn*> is not an open listening socket, or otherwise some error occurred.

+iSST — Get A Single Socket Status Report

Syntax: AT+iSST:<*hn*>

Retrieves a socket status report for a single socket. This is a subset of the general AT+iRP4 report command.

Parameters:

<*hn*> A TCP/UDP socket handle

Command Options:

<*hn*> Must have been obtained by a previous execution of an AT+iSTCP or AT+iSUDP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.

Result Code:

I/*(<sockstat>)* where,

sockstat >=0 – Number of bytes pending in socket <*hn*>'s input buffer

sockstat <0 – Socket error code

I/ERROR If some error occurred

+iSCS — Get A Socket Connection Status Report

Syntax: AT+iSCS:<*hn*>

Retrieves a socket's connection status report without reporting the number of buffered characters.

Parameters:

<*hn*> A TCP/UDP socket handle

Command Options:

<*hn*> Must have been obtained by a previous execution of an AT+iSTCP or AT+iSUDP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.

Result Code:

I/(<*sockstat*>) where,

sockstat=000 – Socket is connected without any associated errors.

sockstat<0 – Socket error code

I/ERROR If some error occurred.

+iSSND[%] — Send A Byte Stream to A Socket

Syntax: AT+iSSND[%]:<*hn*>,<*sz*>:<*stream*>[<*checksum*>]

Sends a byte stream of size *sz* to the socket specified by the socket handle *hn*.

Parameters:

<*hn*> A TCP/UDP socket handle of an open socket

<*sz*> The exact size of the byte stream that follows

<*stream*> A byte stream of size *sz* to be sent to the specified socket. When iChip is in checksum mode (CKSM set to 1), or when sending data over an SSL socket, *sz* is limited to 2048 bytes. ECHO is automatically disabled for these bytes.

<*checksum*> A two-byte checksum. Checksum is calculated by summing all the characters in *stream* modulo 65536 and taking two's complement of the result. Checksum is sent as big-endian. This parameter must be appended by the host application when iChip is in checksum mode.

Command Options:

<*hn*> Must have been obtained by a previous execution of an AT+iSTCP or AT+iSUDP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.

<*sz*> Regular TCP or UDP socket: 0..4GB

SSL Socket or Checksum mode: 0..2048

<*stream*> An 8-bit byte stream of exactly size *sz*. If *sz* is larger than 256 bytes, iChip assumes host flow control. Depending on the setting of the FLW parameter, the flow control mode is either software or hardware.

Under software flow control mode, the host processor must respond to iChip's flow control characters. The flow control protocol is detailed in the "Host ➔ iChip Software Flow Control" section.

Under hardware flow control, the ~CTS/~RTS RS232 control signals must be connected and the host must respond to iChip's ~CTS signal. The host may send data only when the ~CTS signal is asserted (active low).

% When the auto flush (%) flag is specified for a TCP socket, the socket is automatically flushed immediately after receiving the *stream*. Otherwise, data is transmitted to the Internet only in integral quantities of the specified Maximum Transfer Unit (MTU) or when the AT+iSFSH command is issued. When using a

UDP socket, every SSND command generates and flushes a packet.

Result Code:

I/OK After *sz* bytes have been transferred successfully to the socket's output buffer.

I/ERROR Otherwise

Note: When iChip is in checksum mode, it calculates the checksum of the data received from the Host and compares it with *checksum* sent by the Host. If the two match, the result code is **I/OK**. Otherwise, **I/ERROR (228)** is returned and the data is discarded. If Host attempts to send more than 2048 bytes, **I/ERROR (227)** is returned.

The Socket Command Abort (---) may be used to abort prematurely.

+iSRCV — Receive A Byte Stream from A Socket's Input Buffer

Syntax: AT+iSRCV:<*hn*>[,<*max*>]

Receives a byte stream from the TCP/UDP socket specified by the socket handle *hn*. Received data is valid only if it already resides in iChip's socket input buffer at the time this command is issued.

Parameters:

<*hn*> A TCP/UDP socket handle of an open socket

<*max*> Optionally specifies the maximum number of bytes to transfer. Additional bytes may remain in the socket input buffer following this command.

Command Options:

<*hn*> Must have been obtained by a previous execution of an AT+iSTCP or AT+iSUDP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.

<*max*> For TCP sockets, if <*max*> is not specified, all available bytes residing in the socket input buffer are returned.

For UDP sockets, if <*max*> is not specified, data from one UDP packet is returned. Additional packets may remain in the socket input buffer.

Returns:

I/<*sz*>[:<*stream*>][<*checksum*>] where,

sz is the exact size of the binary data stream to follow.

If the socket input buffer is empty, iChip returns **I/O<CR><LF>**. In this case, *stream* is omitted.

sz is guaranteed to be equal or less-than *max*, when specified.

checksum is a two-byte checksum. This parameter is calculated by iChip only when it is in checksum mode (CKSM set to '1'). *checksum* is calculated by summing all the characters in *stream* modulo 65536 and taking two's complement of the result. *checksum* is sent as big-endian. The host application is assumed to calculate its own checksum upon receipt of *stream* and compare it against the checksum bytes received from iChip. If the two checksums don't match, the host can issue an AT+i!SRCV command, which

	causes iChip to re-transmit the data. The next AT+iSRCV command that the host issues causes iChip to dump all data transmitted to host in the previous AT+iSRCV command.
I/ERROR	If <*hn*> is not an open socket, or otherwise some error occurred.

+iGPNM — Get Peer Name for A Specified Socket

Syntax: AT+iGPNM:<*hn*>

Retrieves peer name (<*IP*>:<*Port*>) of a remote connection to a TCP/UDP socket specified by the socket handle <*hn*>.

Parameters:

<*hn*> A TCP/UDP socket handle of an open socket

Command Options:

<*hn*> Must have been obtained by a previous execution of an AT+iSTCP or AT+iSUDP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.

Result Code:

I/(<*IP*>:<*Port*>) where,

<*IP*> is the remote peer's IP address, and <*Port*> is the remote peer's port for this connection.

I/ERROR If <*hn*> is not an open socket handle, or otherwise some error occurred.

+iSDMP — Dump Socket Buffer

Syntax:	AT+iSDMP:<*hn*>
	Dumps all buffered data currently accumulated in a TCP socket's inbound buffer. The socket remains open.
Parameters:	
<*hn*>	A TCP socket handle of an open socket
Command Options:	
<*hn*>	Must have been obtained by a previous execution of an AT+iSTCP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.
Result Code:	
I/OK	If <*hn*> is a handle to an open socket.
I/ERROR	Otherwise

+iSFSH[%] — Flush Socket's Outbound Data

Syntax: AT+iSFSH[%]:<*hn*>

Flushes (immediately sends) accumulated data in a TCP socket's outbound buffer.

Parameters:

<*hn*> A TCP socket handle of an open socket

Command Options:

<*hn*> Must have been obtained by a previous execution of an AT+iSTCP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.

% When the flush-and-acknowledge (%) flag is specified and <*hn*> is a TCP socket handle, iChip flushes and waits for the peer receipt acknowledgment of all outstanding outbound data.

Common errors associated with this flag are 215 (carrier lost) and 203 (socket closed by peer in an orderly manner or did not receive ACK after repeated attempts to retransmit unacknowledged data).

Result Code:

I/OK If <*hn*> is a handle to an open socket and, when <*hn*> is a TCP socket handle, all outbound data has been received (and when (%) flag specified also acknowledged) by peer.

I/ERROR Otherwise

The Socket Command Abort may be used to abort prematurely.

+iSCLS — Close Socket

Syntax:	AT+i[!]SCLS:<*hn*>
	Closes a TCP/UDP socket.
	If the socket is the only open socket and the stay online flag (!) is not specified, iChip terminates the Internet session and goes offline.
Parameters:	
<*hn*>	A TCP/UDP socket handle of an open socket
Command Options:	
<*hn*>	Must have been obtained by a previous execution of an AT+iLTCP, AT+iSTCP or AT+iSUDP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.
	A socket is always flushed before being closed. TCP sockets are disconnected from the remote host server in an orderly manner.
!	Stay online after completing the command.
Result Code:	
I/OK	If <*hn*> is a handle to an open socket
I/ERROR	Otherwise
Followed by:	
I/DONE	After successfully closing the last open socket. Allow a 2.5 seconds delay for iChip re-initialization following an Internet mode session.
-or-	
I/ONLINE	After successfully closing the socket.
	When connected to a modem: After successfully closing the socket, while additional sockets are still open or if the stay online flag (!) is specified.
-or-	
I/ERROR	Otherwise

18 Secure Socket Protocol Theory of Operation

Introduction

iChip implements an SSL3/TLS1 client socket connection. When connecting to an SSL3/TLS1 server, iChip negotiates an SSL3/TLS1 secure connection. During the negotiation process, the server identifies itself to the client (iChip) by sending a certificate. The certificate's main purpose is to allow iChip to determine that the server is indeed the server it claims to be.

To fulfill its purpose, the certificate contains the server's ID information (name, address, description, etc.) and its public key. It also contains a digital signature, signed by a third-party called a Certificate Authority (CA), which authenticates this information. The client must trust the CA in order to accept its signature on a certificate. Furthermore, the trust relationship between the client and the CA must be established prior to the communication session and preferably using alternate methods. iChip's CA parameter is used to store the CA's certificate. Once a trusted CA's certificate is stored on iChip, it will accept certificates signed by that CA from SSL3/TLS1 servers it connects to.

Generating Certificates for Use with Servers

The most common way to obtain a certificate is to buy one from a commercial certificate authority. This results in a public key that has been digitally signed by a trusted third-party. Any clients receiving this certificate can be sure they are communicating with an authentic entity. However, in a trusted environment, it is possible to create an in-house CA and to self-sign the certificate.

Commercial CA's are usually preferred when connecting to multiple unknown servers. However, in distributed system configurations where not more than a handful of secure servers are deployed; an in-house CA is probably more appropriate and just as secure.

Several free software packages are available for generating certificates. The following sections describe how to use the standard OpenSSL package to generate certificates. They contain instructions on how to obtain your own certificates suitable for use with servers to which iChip will connect. Furthermore, most FTP servers that support SSL3 include a certificate generation utility that may be used to generate self-signed certificates. The self-signed certificate is part of the FTP server's configuration and may also be loaded into iChip to allow it to connect to that FTP server using SSL3 secure sockets.

Using the OpenSSL Package to Create Certificates

OpenSSL is a widely used SSL toolkit available for free download at http://www.openssl.org. The SSL toolkit contains source code that can be compiled for Unix, Linux, or Windows. Pre-compiled binaries are also available for these platforms. OpenSSL comes with a command line utility for generating keys, creating CA's, and creating certificates.

The following instructions assume the OpenSSL package has been installed and configured properly on your machine. The instructions walk you through using OpenSSL to create an in-house Certificate Authority, sign your own certificates, and generate the proper requests in order to receive a signed certificate from a commercial CA. The signed certificates can then be installed on servers to which iChip will connect in a secure (SSL3/TLS1) manner.

Creating a Certificate Authority

The certificate generated using the following steps can be used in deployed systems, in which **you** are the trusted authority. Users of these certificates can be confident of your identity. For example, iChip devices communicating with servers that are setup and configured by the device vendor can secure their communications using certificates signed by the vendor-created Certificate Authority.

In order to store the files to be generated, create a new directory named *testCA.*

Open a command shell (on Windows, enter **cmd** in the Start > Run dialog box). Change the command shell's working directory to *testCA* and follow these instructions:

Creating the CA Environment

The creation of a CA produces several files that must be preserved throughout the lifecycle of the CA. You can sign an unlimited number of certificates using a single CA. These files are written to each time you sign a certificate.

1. Under the *testCA* directory create sub-directories *certs* and *private*.
2. Create a new file named *serial*. In this file enter the numerals '01' and save the file.
3. Create an empty file named *index.txt*.

Creating the Test CA Configuration File

Whereas you can enter all configuration information in a command line, creating a configuration file makes these steps easier to reproduce and allows you to save the options used to create a CA.

1. Create a new file named *CAcnf.ca* using a text editor of your choice.
2. Add the following basic CA configuration information:

```
[ ca ]
default_ca = CA_default

[ CA_default ]
dir = /testCA
certificate = $dir/cacert.pem
database = $dir/index.txt
new_certs_dir = $dir/certs
private_key = $dir/private/caprivkey.pem
serial = $dir/serial
default_crl_days = 7
default_days = 365
default_md = md5
policy = CA_default_policy
x509_extensions = certificate_extensions
```

```
[ CA_default_policy ]
commonName = supplied
stateOrProvinceName = supplied
countryName = supplied
emailAddress = supplied
organizationName = supplied
organizationalUnitName = optional

[ certificate_extensions ]
basicConstraints = CA:false

[ req ]
dir = /testCA
default_bits = 1024
default_keyfile = $dir/private/caprivkey.pem
default_md = md5
prompt = no
distinguished_name = root_ca_DN
x509_extensions = root_ca_extensions

[ root_ca_DN ]
commonName = Common Name                      # Server name or YOUR
name
stateOrProvinceName = My State
countryName = US                              # 2 Letter Code
emailAddress = myemail@mydomain.com           # Your Email Address
organizationName = My Organization
organizationalUnitName = Organization Unit    # Unit Name (ie, section)

[ root_ca_extensions ]
basicConstraints = CA:true
```

Note that both *dir* entries under [CA_default] and [req] must be set to the path to the *testCA* directory created earlier. The *root_ca_DN* section can be changed to enter information specific to your organization.

Creating a Self-Signed Root Certificate

A certificate authority is essentially a self-signed root certificate. This root certificate is used to respond to new certificate requests to create a signed certificate. In this case, iChip is both the CA and the originator of the certificate request, so no identity verification issues exist. In a more typical situation, however, a CA can only be trusted if it performs sufficient background checks into the originator of the certificate request to verify its identity.

1. Set the OPENSSL_CONF system environment variable to point to the newly created configuration file.

- On Linux\Unix, type the following:
  ```
  OPENSSL_CONF=/testCA/CAcnf.ca
  export OPENSSL_CONF
  ```
- On Windows, type the following:
  ```
  set OPENSSL_CONF=C:\testCA\CAcnf.ca
  ```

2. Enter the command for generating the self-signed root certificate (all text is a single command typed on one line):

```
openssl req –x509 –newkey rsa:1024 –out cacert.pem –outform PEM
```

3. You are prompted to enter a PEM pass phrase. This is your password to the CA private key. It is essential for the security of the system that both this password *and* the CA private key are kept secret.

An encrypted *caprivkey.pem* file, which is the private key for the CA is now stored under the *private* sub-directory. The self-signed *cacert.pem* file is stored under the top-level *testCA* directory.

The *cacert.pem* certificate can be used to sign new certificate requests as detailed in the following steps. Alternatively, the *cacert.pem* certificate can be used as-is in a server system if the single level hierarchy is considered sufficient.

The *cacert.pem* certificate has to be loaded into iChip's CA parameter to enable iChip to trust and communicate securely with servers whose certificate is *cacert.pem* or that use certificates **signed** with *cacert.pem* (see description on how to do that with the iChipConfig utility or using iChip's web server).

Signing a Certificate with a CA Certificate

Creating a Certificate Request

Now that the CA has been created, you can use it to sign new certificates. In this example, iChip plays the role of the CA, the certificate subject, and the end-user of the certificate, so no trust issues exist. A typical process, however, involves communication between the certificate subject (you) and a trusted CA. Usually someone wishing to issue certificates to end-users would generate a certificate request file and submit it to the administrators of a CA. Once the administrators of the CA have determined the request to be valid, a self-signed root certificate would be used to sign the certificate request and create a new certificate to be returned to the originator of the request, and eventually to the end-user.

1. Reset the OPENSSL_CONF environment variable to the default *openssl.cnf* file. Generating a request has nothing to do with a CA before it is actually submitted. It is safe to point OPENSSL_CONF to the default configuration file because it will force the request command to prompt the user for all information regarding the certificate request. Set the environment variable to the default file by typing the following:

   - On Linux\Unix:

   ```
   OPENSSL_CONF=/OpenSSL/apps/openssl.cnf
   export OPENSSL_CONF
   ```

   - On Windows:

   ```
   set OPENSSL_CONF=C:\OpenSSL\bin\openssl.cnf
   ```

2. Generate the request with the following single line command and answer all questions at the prompt:

```
openssl req -newkey rsa:1024 -keyout myprivkey.pem -keyform PEM -out
myreq.pem -outform PEM
```

If you do not want an encrypted private key, add *–nodes* to the above command. At the conclusion of this step two new files are created. The *myprivkey.pem* file contains the encrypted private key. This file must never be shared, not even with the CA. The other file is the certificate request file, *myreq.pem*, which will be used by the CA to create the final signed certificate.

Using the Test CA to Issue the Certificate

The final step of the process is to use the CA self-signed certificate to sign the certificate and return it to the originator of the request (subject).

1. Reset the OPENSSL_CONF system environment variable to reference the CA configuration file again.
   - On Linux\Unix type the following:

   ```
   OPENSSL_CONF=/testCA/CAcnf.cnf
   export OPENSSL_CONF
   ```

   - On Windows type the following:

   ```
   set OPENSSL_CONF=C:\testCA\CAcnf.cnf
   ```

Make sure that the request file is in the current directory and run the following command. The PEM password you are prompted to enter is the password for the CA private key file:

```
openssl ca -in myreq.pem
```

You will be requested to enter the pass phrase for the CA private key that was generated above. Enter the pass phrase to continue.

Answer ‘y’ at the next two prompts, then at the conclusion of this step several files are updated and a new certificate is created.

The new certificate can be found in the *certs* sub-directory. It is named as the serial number it is associated with by the CA. The file can be renamed, but the .pem extension must be preserved for clarity. The *serial* file itself increments its count for the next certificate request and the *index.txt* file shows a record of the creation. The new certificate file and the *myprivkey.pem* file are now suitable for use by an SSL server to which iChip needs to connect. As mentioned above, the iChip +iCA parameter must contain the CA certificate *cacert.pem* used to sign the server’s certificate.

19 Secure Socket Protocol

iChip supports the SSL3/TLS1 secure socket protocol, based on RFC2246. iChip supports the following Cipher suites:

- SSL_RSA_WITH_RC4_128_MD5
- SSL_RSA_WITH_RC4_128_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA

Establishing An SSL3/TLS1 Socket Connection

iChip supports a single SSL3/TLS1 TCP/IP active socket connection. Opening a secure socket on iChip involves two steps:

1. Open a standard TCP/IP socket to a secure server.
2. Initiate an SSL3/TLS1 handshake over the open socket to establish a secure session. SSL3/TLS1 handshake negotiations are initiated using the AT+iSSL command.

iChip negotiates the secure connection based on several security-related parameters. It authenticates the remote secure server by verifying that the server's certificate is signed by a trusted Certificate Authority (CA). The authentication may be disabled by setting the SDM parameter accordingly. The trusted CA's certificate is stored in iChip's CA parameter. Following a successful SSL3/TLS1 handshake, iChip encrypts all data sent across the socket according to the cipher suite and keys agreed upon during the handshake. Data received on the socket is decrypted by iChip prior to making it available to the host processor.

Sending and Receiving Data over An SSL3/TLS1 Socket

The AT+iSSND command is used to send data over an SSL3/TLS1 socket, using the same syntax as for non-secure sockets:

`AT+iSSND[%]:<hn>,<size>:<data>`

However, the *size* parameter is interpreted as the size of the data packet to encrypt. It is limited to 2K. Receiving data on an SSL3/TLS1 socket is carried out using the AT+iSRCV command. iChip automatically decrypts data that arrives on the secure socket. The data transferred to the host is always decrypted data.

SSL3/TLS1 Handshake and Session Example

Take for example an SSL3/TLS1 server at `secure.sslserver.com` running a secure application on port 1503. Using iChip, the following sequence opens a secure SSL3/TLS1 socket to that application and exchanges data securely. For clarity, commands sent to iChip appear in **bold** and iChip replies appear in *italics*.

AT+iSTCP:secure.sslserver.com,1503	Open a TCP/IP socket to a secure application.
I/000	iChip opens socket and returns handle 0.
AT+iSSL:0	iChip is instructed to negotiate an SSL3/TLS1 connection on socket handle 0.
I/OK	SSL3/TLS1 handshake was successful. SSL3/TLS1 connection established on socket handle 0.
AT+iSSND%:0,323:<…323 bytes of plain text data>	Host sends 323 bytes of plain text data via SSL3/TLS1 socket. iChip encrypts data and sends cipher text over the Internet. The '%' attribute indicates immediate flush.
I/OK	iChip encrypted and sent data.
AT+iRP4	Request socket status
I/(1267,-200,-200,-200,-200,-200,-200,-200, -200,-200)	Socket 0 has 1267 plain text bytes buffered. The data was originally sent encrypted by the server. iChip decrypted the cipher text in the background.
AT+iSRCV:0	Command to retrieve buffered plain text.
I/1267:<…1267 bytes of plaintext data…>	iChip transmits buffered data to host.
AT+iscls:0	Close socket handle 0
I/OK	SSL3/TLS1 socket is closed
I/DONE	iChip is offline

Secure FTP Session on iChip

iChip supports a secure FTP session using SSL3/TLS1 sockets for both the FTP command and FTP data channels. The command used for opening a secure FTP session is AT+iFOPS.

Secure FTP implementation in iChip is based on RFC 2228 (FTP security extensions) and the IETF Internet draft "Securing FTP with TLS" (draft-murray-auth-ftp-ssl-16.txt).

When the AT+iFOPS command is used to initiate a secure FTP session, iChip performs the following operations:

1. Opens an FTP control socket.
2. Sends `AUTH TLS`.
3. Performs the SSL3/TLS1 handshake.
4. Sends `USER` command.
5. Sends `PASS` command.
6. Sends `PBSZ 0`, followed by `PROT P`.

Once the data channel TCP socket is established, all subsequent data connections (send or retrieve files as well as directory listings) start with an SSL3/TLS1 handshake. When a data socket is re-opened for another FTP command, iChip attempts a quick re-negotiation using the previous SSL3/TLS1 session parameters.

+iSSL — Secure Socket Connection Handshake

Syntax: AT+iSSL:<*hn*>

Negotiates a secure SSL3/TLS1 connection over an open TCP/IP socket.

Parameters: <*hn*> = A previously open TCP/IP socket handle.

Command Options:

<*hn*> Must be obtained using the AT+iSTCP command during the current Internet mode session. Or a socket ***accepted*** by a listening socket.

When a Network Time Server is defined and NTOD is set to 1, iChip confirms the server's certificate date validity using the retrieved network time. If, for some reason, the network time is not retrieved successfully, iChip does not accept the certificate until the time is retrieved successfully.

Result Code:

I/OK If the SSL3/TLS1 negotiation is successful.

I/ERROR Otherwise

+i[@]FOPS — Secure FTP Open Session

Syntax:	AT+i[**@**]FOPS:<*server*>[,<*port*>]:<*user*>,<*pass*>[,<*accnt*>]
	Opens a secure FTP link to a secure FTP server.
Parameters:	
<*server*>	Logical name of the FTP server or the server's IP address.
<*port*>	Optional FTP port in the range 0-65535
<*user*>	FTP user's name
<*pass*>	FTP user's password
<*accnt*>	Optional FTP account
Command Options:	
<*server*>	The server name may be any legal Internet server name that can be resolved by iChip's Domain Name Server (DNS) settings. The server name may also be specified as an absolute IP address given in DOT form.
<*port*>	Specifies the FTP server's listening port. If not specified, port 21 (decimal) is assumed.
<*user*>	User's name string. This must be a registered user on the FTP server. Some servers allow anonymous login, in which case *user*=anonymous.
<*pass*>	Password for user authentication. If special characters are used, the password must be specified within quotes. It is customary that servers that allow anonymous login request an e-mail address as a password.
<*accnt*>	Some FTP servers require an account in order to allow a certain subset of the commands. In this case, the account name must be specified when opening the FTP link.
@	The optional @ is used to flag the Force PASV mode. When @ is specified, iChip uses only the PASV method when opening a data socket to *server* for FTP data transfer.
Result Code:	
I/<*FTP handle*>	Upon successfully connecting to the FTP server and authenticating the user, a socket handle is returned. The handle <*FTP handle*> is used to reference the FTP session in all subsequent FTP commands.
I/ERROR	Otherwise

20 SerialNET Theory of Operation

Introduction

iChip's SerialNET mode extends a local asynchronous serial link to a TCP or UDP socket across a LAN or Internet. Its main purpose is to allow simple devices, which normally interact over a serial line, to interact in a similar fashion across a network without requiring any changes in the device itself. In order to achieve this, SerialNET mode defines a set of associated operational parameters, which determine the nature of the desired network connection. When iChip is put in SerialNET mode, it acts as a router between the device's serial port and the network.

Devices that communicate with a terminal over a serial link fall into three major categories: Output only (i.e. printers), Input only (i.e. controllers) and interactive (bi-directional communications). The latter are subdivided further into **clients** and **servers**. Generally, clients initiate communications by sending service demands to a server, while servers respond to client demands.

SerialNET mode reacts differently to client or server devices. When a client device initiates communications, SerialNET mode must establish a network connection to a remote server before data may flow between the two systems. On the other hand, when a remote client needs to invoke a device, the remote client first contacts the iChip and SerialNET is invoked to create a communication flow to the local server device.

SerialNET mode includes components to handle both server and client local devices. The iChip under SerialNET mode routes full-duplex data between a networked terminal and both types of devices.

SerialNET Mode

SerialNET mode is established by first defining all related parameters using AT+i commands, followed by a special Enter SerialNET Mode command.

SerialNET can be entered by:
Issuing a special AT+i command from the local host: AT+iSNMD.
Issuing the command: AT+iSNMD from a Remote AT+i Service.
Enabling SerialNET mode from the embedded configuration website.

Once in SerialNET mode, no additional AT+i commands can be sent, as the host serial link will be dedicated to raw local-device data. In this mode, auto baud rate is also disabled, since it cannot be guaranteed that the device will issue an 'a' or 'A' as its first character. Thus, a predefined fixed baud rate must be specified before switching over to SerialNET mode. Similarly, the host interface cannot be determined automatically and therefore you must set iChip's Host Interface to USART0 (HIF=1) or USART1 (HIF=2).

SerialNET mode extends across power-down, since it is assumed that once acting in this mode, iChip is not necessarily connected to an AT+i aware host.

SerialNET mode can be terminated by:

- Pulling the MSEL signal low for more than 5 seconds.

- Issuing the ESC sequence, defined as a half second delay followed by +++ (three '+' characters), over the serial port.
- Sending BREAK signal over the serial interface.
- Termination from the embedded configuration website.
- Issuing the ESC sequence over a LATI connection.

The SDM parameter defines which termination methods can be used. When one of these occurs, iChip reboots after terminating SerialNET mode. At this point iChip reverts to its normal operational mode and again responds to AT+i commands.

Server Devices

Server devices linger until approached by a remote client. The remote client must know iChip's IP and listening port address in order to establish communications.

LAN-based devices and dial-up devices linger differently.

A LAN device is normally online and may thus have an associated listening (passive) socket ready to accept remote socket connections. While in SerialNET mode, iChip establishes a listening socket on the port defined in its LPRT parameter. A remote client terminal can connect to that port.

A dial-up device is normally offline and must be awakened to go online at a precise moment. Moreover, once it connects to the Internet, it usually receives a dynamic IP address. This address must be communicated in some way to the client device in order to establish a link across the Internet. iChip resolves these problems by supporting a wake-up call and automatically implementing one or more IP registration procedures. This allows a client to wake up an iChip in SerialNET mode and retrieve its dynamic IP address from a registration server.

The iChip, in dial-up mode, is offline by default, but waits for a RING signal on the modem to trigger it into activity. In this case, the remote client device dials directly to the iChip and hangs up after two rings. When contacted, iChip (under SerialNET mode) waits for the RING to subside and then dials into its ISP and connects to the Internet. If the RRMA parameter contains an e-mail address, iChip registers its IP address using the Email registration method. iChip then listens on the LPRT port for a socket connection. The recipient of the e-mail can use the registered IP address and port to create a link to iChip's SerialNET socket.

If the RRSV parameter contains a server name and port, iChip registers its IP address using the Socket registration method.

If the RRWS parameter contains a URL, iChip registers its IP address using the Web server registration method.

Once connected, iChip transfers all arriving data from the local device over the serial link. Device responses are routed back to the initiating client. Data flows freely between the two systems until a predefined activity termination event is triggered, upon which the remote connection is dropped.

In a LAN environment the iChip continues to listen on the port server listening socket, while in a dial-up environment, iChip goes offline and waits for another RING trigger.

The iChip MSEL signal (see iChip datasheet) can be lowered to GND to emulate the RING event. This is useful for testing and debugging purposes of the SerialNET connection procedure or as a means to cause iChip to activate the ring response procedure as a result of some TTL hardware signal.

A remote client may terminate a socket without notifying the iChip. The same client may recover and try initiating the connection again. If previous socket is still established on LPRT port and a second connection is made by the same client, the first socket will be closed and the new socket will be active. Any other connection from different clients will be denied by iChip.

Client Devices

Client devices initiate communications to a server. When a client device first sends data on its serial link, iChip (in SerialNET mode) buffers the incoming data bytes and attempts to establish a connection to a remote server. After going online, iChip performs an IP registration process according to the RRSV, RRWS, and RRMA parameters.

Once the socket connection is established, iChip transmits the buffered data collected during the connection period. The MBTB parameter dictates the maximum number of bytes to buffer. If additional bytes are received on the serial port before the connection is established, they are discarded.

iChip will dial-up the ISP to establish an Internet connection before attempting to open the server socket.

iChip closes its listening socket (if one is defined by the LPRT parameter) to avoid remote client devices from connecting during this session.

The remote server's IP and port are part of the SerialNET mode configuration parameters: HSRV, HSR1, HSR2. Once a data connection is established, data can flow freely between the local client device and the remote server. If a connection cannot be obtained, eventually the client device's data will be discarded (similar to the case of a device transmitting serial data without a serial cable connected). Data continues to flow until a predefined activity termination event is triggered, upon which the remote connection is dropped.

Secure SerialNET

When the parameter STYP is set to 2 (AT+iSTYP=2), iChip assumes that the server defined in the HSRV parameter is an SSL3 server and shall negotiate an SSL3 connection to that server. The secure connection is applicable for client devices and should be setup accordingly. If a non-zero value is define in LPRT and a remote system opens a TCP socket to the LPRT port, a regular TCP (non SSL3) socket shall be maintained for that SerialNET session.

SSL related parameters should be set prior to entering SerialNET mode: CS, CA, CERT, PKEY.

Automatic SerialNET Server Wake-Up Procedure

A SerialNET client may be configured to wake up a remote SerialNET server provided it has its phone number. The SPN parameter is used to store this wakeup number.

When SPN contains a phone number and no Host Server Name and/or IP are defined, the SerialNET client tries to retrieve them from the registration e-mail of a remote SerialNET server. When characters are received from the host port, the SerialNET client dials the SerialNET server and then hangs up, causing the server to connect to its ISP, send a registration e-mail containing its IP address and local port, and open a listening socket on that port.

The client, after waking up the server, connects to its ISP and starts polling the predefined mailbox for the server's registration e-mail. Once this e-mail arrives, the client opens a socket to the IP address and port defined in the e-mail. The SWT (SerialNET Wakeup Timeout) parameter defines how long iChip will wait for this procedure to conclude before stopping. Data then flows until a predefined activity termination event is triggered, upon which the remote connection is dropped.

Transmit Packets

Data originating in the local device is buffered, packetized, and transmitted to the remote system over the network. Packets are formed as a result of meeting at least one of the following criteria:

- A predetermined number of bytes has been received from the local link (MCBF).
- The TCP/IP connection MTU was met.
- A predetermined flush character has been received (FCHR).
- A predetermined inactivity timeout event was triggered (MTTF).

Until one of these events occurs, data is buffered in the iChip. When an event occurs, a packet is transmitted. The event parameters are configured by setting AT+i parameters prior to initiating SerialNET mode. When a UDP connection is used, data packets are atomic, maintaining their original size. When a TCP connection is used, packets can be combined before being actually transmitted. This follows from the stream nature of the TCP protocol. Data originating in the remote system is routed to the local device as it is made available. Flow control can be governed locally using hardware flow control only.

The PTD parameter can be used to define the number of packets to be cyclically discarded in a SerialNET mode session. When PTD>0, iChip first discards <*ptd*> packets before actually sending one to the SerialNET socket. This can be used to dilute repetitive information.

Completing a SerialNET Session

A socket is closed, and a SerialNET session is completed when one of the following occurs:

- The local device transmitted the disconnection string, as defined in the DSTR parameter.

- Following an inactivity timeout, as defined in the IATO parameter.

In a modem environment the iChip goes offline when the SerialNET session is terminated.

In a LAN environment, the iChip reopens the SerialNET listening socket defined in the LPRT parameter (if it is non-zero) to service future remote client connections.

SerialNET Failed Connection

If the iChip fails to establish a SerialNET connection, SerialNET mode is deactivated for a delay period defined in the SNRD parameter.

Local Serial Port Configuration

Prior to entering SerialNET mode, iChip's local serial port can be configured to comply with a wide range of devices by assigning a value to the SNSI parameter.

Serial port configuration entails settings to:

Baud rate:	600, 1200, 2400, 4800, 9600, 19200, 38400, 56K, 115K, 230K or a division of 3Mbps as determined by the BDRD parameter
Bits/byte:	7 or 8
Parity:	None, Even, or Odd
Stop Bit:	1, 1.5 or 2
Flow Control:	None (0) or Hardware (1)

Activation Command

The iChip is forced into SerialNET mode by issuing the following command:

AT+i[!|@]SNMD or AT+iSNMD=n, where n=1, 2, 3 or 4

If the minimal SerialNET parameters are defined, iChip replies with **I/OK** followed by **I/DONE** or **I/ONLINE** or **I/OFFLINE**. The IO signal designated by the SLED parameter is asserted LOW.

If the iChip is online at the time this command is issued, it closes the Internet session in an orderly manner. This includes closing all open sockets and disconnecting from the ISP in a modem environment.

When iChip boots up in SerialNET mode, it sets the host serial channel to the fixed baud rate and serial interface parameters defined in the SNSI parameter. iChip in LAN mode opens the SerialNET listening socket (if it is defined in the LPRT parameter) and, if defined, launches the web server. It carries out the IP Registration procedure, if defined.

In an iChip dial-up environment, the modem is polled for the RING string. If one or more ring-response registration methods, RRMA, RRSV or RRWS, contain values, iChip waits for the RING strings to subside and connects to the Internet. Once online, it registers via e-mail, TCP socket or Web server as defined. The transmission contains the dynamic IP address received from the ISP and the listening port, on which iChip has an open listening socket, ready to serve the remote client.

iChip terminates the socket connection if one of the following events occurs:

- The remote peer closes the SerialNET socket.
- The IATO parameter is defined and times out.
- The terminating string defined in the DSTR parameter is received.

When the optional (!) (Auto-Link mode) flag is specified, iChip immediately goes online in response to the AT+i!SNMD command, opens the SerialNET listening socket (if it is defined in the LPRT parameter) or attempts to establish a socket to an HSR*n* address (if any HSR*n* is defined and LPRT is not). In this case, if one of the terminating events occurs, iChip does not go offline. Rather, the SerialNET socket is closed while iChip stays online and opens the listening or active socket again, after waiting the SNRD delay.

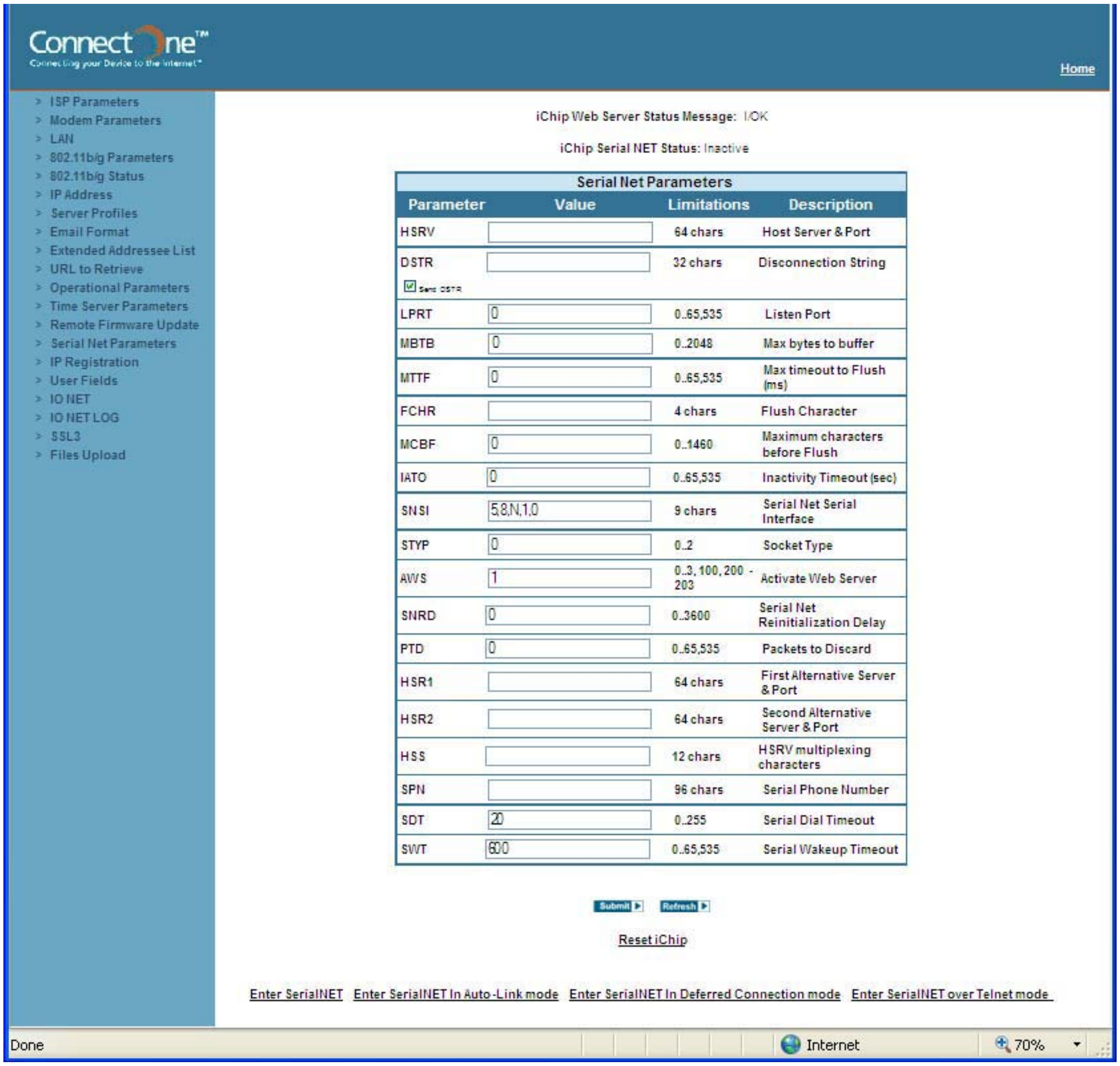
When the optional (@) (Deferred Connection mode) flag is specified, iChip immediately goes online in response to the AT+I@SNMD command. It opens the SerialNET listening socket (if it is defined in the LPRT parameter) but does not attempt to establish a socket to the HSRV address if it is defined. In this case, if one of the terminating events occurs, iChip does not go offline. Rather, the SerialNET socket is closed while iChip stays online and opens the listening socket again, after waiting the SNRD delay.

The optional AT+iSNMD=4 flag (SerialNET over TELNET) expands the Auto-Link mode (!SNMD). The data socket is opened as a TELNET socket, which allows negotiations of TELNET options over the same socket while the host is sending and receiving raw data only.

iChip exits SerialNET mode when one of the Escape procedures is activated.

Remote Initiation/Termination

iChip's SerialNET tab in the Configuration Web site includes links that may be used to activate or terminate SerialNET mode. When iChip is online and its Web server is enabled, a remote browser may surf to iChip's Configuration Web site, select the SerialNET tab and activate SerialNET mode (if iChip is not in SerialNET mode) or exit SerialNET mode (if iChip is currently in SerialNET mode).



When iChip is not in SerialNET mode, but online, the LATI socket may be used to remotely connect to iChip and issue the AT+i[!|@]SNMD command to initiate SerialNET mode. A remote user may connect to the LATI socket while iChip is in SerialNET mode. The connected socket may then be used to send the '+++' SerialNET termination string and cause the iChip to exit SerialNET mode. The host microcontroller cannot send and receive data on the local interface while the LATI socket is connected, but can still terminate SerialNET mode with any of the termination methods.

Note that changing the SerialNET mode status is always followed by a Soft-Reset, which will cause the remote browser to disconnect and the LATI socket to disconnect. When iChip reboots again it will obtain an IP address and go online according to its parameter settings and environment. If iChip is online and its +iLATI or +iAWS parameters are set to values larger than zero, the LATI socket will re-open or the iChip Web server shall be enabled. The remote socket or web connection may be re'established (assuming iChip's IP address is known).

SerialNET over TELNET

SerialNET over TELNET mode of operation opens a data socket as a TELNET socket, which allows negotiations of TELNET options over the same socket while the host is sending and receiving raw data only. This mode partially supports the RFC2217 standard.

SerialNET over TELNET mode is entered by sending the command AT+iSNMD=4 after setting iChip's Host Interface to USART0 (HIF=1) or USART1 (HIF=2). An error code – **I/ERROR (124)** – is returned upon setting the SNMD parameter to 4 while the HIF parameter is not set to either 1 or 2.

Mode of Operation

SerialNET over TELNET mode expands the Auto-Link mode (!SNMD). In this mode, iChip immediately goes online upon activating SerialNET, regardless of whether serial data has arrived or not.

If the LPRT (Listening Port) parameter is defined, iChip opens a listening port and awaits a connection, and so it acts as a TELNET server. If, on the other hand, LPRT is *not* defined, but HSRV (Host Server) is defined, iChip acts as a TELNET client and immediately opens a TELNET socket link to the TELNET server.

Note that, even when configured as a client, iChip still acts as a server in RFC2217. See the following section – "RFC2217 Implementation" – for a more detailed explanation.

The SerialNET over TELNET mode expands iChip's TELNET client in the following aspects:

- It allows iChip to operate both as a TELNET server and client.
- It partially supports RFC2217.

In this mode, data is retrieved from the remote side as it is made available. TELNET options embedded in the server/client response stream are stripped by iChip before being turned over to the host. TELNET specifies many operational options. iChip restricts its operation mode to the minimum implementation to assure best inter-system compatibility.

Following are the TELNET options negotiated by iChip. Any other options negotiated by the remote side are rejected by iChip.

Option ID	Name	Value	RFC
1	echo	OFF	857
3	suppress go ahead	suppress	858
24	terminal type	VT100	1091
31	window size	whatever	1073
44	com port	partial implementation	2217

Notes:

1. In SerialNET over TELNET mode, a BREAK signal that is detected on the host USART is relayed to the remote side and no reset is performed.
2. If the host interface is USART1, then DSR signal changes are not detected.

RFC2217 Implementation

The RFC2217 implementation in SerialNET over TELNET mode is designed to:

- Add the ability for a remote client that connects to iChip to send COM port configuration information to the host device connected to the Internet via iChip's TELNET server. The configuration changes take effect immediately, but are not preserved over software or hardware reset. The allowed configurations are the same ones available by the SNSI parameter.
- Add the ability for the host device to inform the remote side about signal changes in CTS and DSR.
- Add the ability for the remote side to change the value of the RTS and DTR signals of the host device.
- Add the ability to exchange BREAK signal indications between the host device and the remote side.

The table below lists the RFC2217 options and sub-options supported by iChip. Note that iChip does not send any replies to commands or command values not supported. For more information about RFC2217, refer to the RFC2217 protocol document.

When issuing any of the following commands, iChip plays the role of a server.

<table>
<tr><th>Option</th><th colspan="3">Allowed Values</th></tr>
<tr><td>Baud Rate</td><td colspan="3">300-115200 bps</td></tr>
<tr><td>Data Size</td><td colspan="3">7 or 8 bits</td></tr>
<tr><td>Parity</td><td colspan="3">None | Odd | Even</td></tr>
<tr><td>Stop Bit</td><td colspan="3">1</td></tr>
<tr><td>Flow Control</td><td colspan="3">BREAK ON | BREAK OFF | DTR ON | DTR OFF | RTS ON | RTS OFF</td></tr>
<tr><td>Notify Line State</td><td colspan="3">One octet (byte). The value is a bit-level composition made up from the value table that appears in the RFC2217 protocol document. Only bit 4 is supported, value 16, meaning BREAK-detect error.</td></tr>
<tr><td rowspan="6">Notify Mode State</td><td colspan="3">One octet (byte). The value is a bit-level composition made up from the value table that appears in the RFC2217 protocol document. Only the following bits are supported:</td></tr>
<tr><th>Bit Position</th><th>Value</th><th>Meaning</th></tr>
<tr><td>5</td><td>32</td><td>Data-Set-Ready Signal State</td></tr>
<tr><td>4</td><td>16</td><td>Clear-To-Send Signal State</td></tr>
<tr><td>1</td><td>2</td><td>Delta Data-Set-Ready</td></tr>
<tr><td>0</td><td>1</td><td>Delta Clear-To-Send</td></tr>
</table>

21 SerialNET Mode Initiation

+iSNMD — Activate SerialNET Mode

Syntax:	AT+i[! | @]SNMD
	Activates SerialNET mode. Instead of using the optional (!) and (@) flags, you can use the following syntax:
	AT+iSNMD=1 is equivalent to AT+iSNMD
	AT+iSNMD=2 is equivalent to AT+i!SNMD
	AT+iSNMD=3 is equivalent to AT+i@SNMD
	AT+iSNMD=4 causes iChip to enter SerialNET-over-TELNET mode
Command Options:	
AT+iSNMD -or- AT+iSNMD=1	Activates SerialNET mode. iChip does not immediately open a socket to the server defined in the HSRV parameter and waits for data to arrive on the local host interface. In modem environments iChip remains offline. In LAN environments it opens the listening port, the remote AT+i port and launches its web server, if defined by the LPRT, LATI and AWS parameters respectively.
AT+i!SNMD -or- AT+iSNMD=2	Optional Auto-Link mode. When this flag is specified, iChip immediately goes online when activating SerialNET mode (even when serial data has not yet arrived). If the LPRT (Listening Port) parameter is defined, iChip opens the listening port and awaits a connection. If LPRT is not defined, but HSRV (Host Server) is defined, iChip immediately opens a SerialNET socket link to the server.
AT+i@SNMD -or- AT+iSNMD=3	Optional Deferred Connection mode. When this flag is specified, iChip automatically goes online (as in the case of AT+i!SNMD). However, if the HSRV parameter is defined, a socket is not opened until data arrives on the local serial port. If the SerialNET mode listening port is defined (LPRT), iChip opens a listening socket and waits for a remote connection during the idle period before data arrives on the local serial port. When the SerialNET socket type (STYP) is TCP and serial data arrives, iChip buffers the data in the MBTB Buffer and tries to connect to HSR0. If HSR0 does not respond, iChip tries HSR1, then HSR2. If all three connection attempts fail, iChip retries them all. After three full retry cycles, iChip dumps the MBTB buffer and remains idle until new serial data arrives.

AT+iSNMD=4	Optional SerialNET over TELNET mode. In this mode, iChip opens a data socket as a TELNET socket, which allows negotiations of TELNET options over the same socket while the host is sending and receiving raw data only. This mode partially supports the RFC2217 standard. For more information about this mode, refer to the SerialNET over TELNET description.

Note: Before entering SerialNET mode, you must set iChip's Host Interface to USART0 (HIF=1) or USART1 (HIF=2).

Result Code:

I/OK	If all minimum required parameters for SerialNET mode operation are defined (HSRV or LPRT and HIF. In a modem environment, also ISP1, USRN, PWD)
I/ERROR	Otherwise

Followed by:

I/DONE	After successfully activating SerialNET mode in a modem environment. Allow a 2.5 seconds delay for iChip re-initialization.
-or-	
I/ONLINE	After successfully activating SerialNET mode. Allow a 2.5 seconds delay for iChip re-initialization.
-or-	
I/OFFLINE	After successfully activating SerialNET Auto-Link mode (!) or Deferred Connection mode (@) with LAN communications, and a LAN link is not detected at the time that iChip enters the new mode.

Note: To terminate SerialNET mode, issue the ESC sequence (+++) or issue a Serial BREAK signal, as defined in the SDM parameter. Alternatively, pull the MSEL signal low for more than 5 seconds. After exiting SerialNET mode, iChip returns to normal AT+i command mode.

22 DHCP Client

A DHCP client component in iChip supports IP and server name acquisition from a standard DHCP Server. iChip attempts to contact and acquire server names from a DHCP server if and when its DIP (Default IP) parameter contains the special value '0.0.0.0'. When the DHCP acquisition procedure is successful, iChip's IPA (IP Address) parameter contains the assigned IP address retrieved from the DHCP server. In addition, server names relevant to iChip parameters are retrieved from the DHCP server, if and only if they contain empty values at power-up. Parameters that contain non-empty values retain those values. In addition, DNS values retrieved from the DHCP server are retained as additional alternative DNS addresses when DNS*n* contain user-defined values.

Parameter Name	Function	Empty Value
IPG	Gateway	0.0.0.0
SNET	Subnet Mask	0.0.0.0
DNS1	Primary Domain Name Server	0.0.0.0
DNS2	Secondary Domain Name Server	0.0.0.0
SMTP	Email Send Server	'' (Empty String)
POP3	Email Receive Server	'' (Empty String)

Table 22.1: Server Names Acquired from DHCP Server

All values acquired from the DHCP server are not retained as nonvolatile values. New values are acquired during the next DHCP session, which is activated during the next iChip power-up, following a soft or hard reset or after the DHCP lease expires.

The DHCP client has two associated points in time when the DHCP server is contacted for additional negotiations. At T1 (usually after half the original lease period), iChip attempts to renew the lease period. If the renew procedure fails, at T2 (usually after 7/8 the original lease period) iChip attempts to renegotiate the lease. If the procedures at T1 and T2 fail and the lease expires, iChip reverts parameter values to their pre-DHCP state and continuously attempts to locate a DHCP server for re-negotiation. When this is the case, iChip stores 0.0.0.0 in the IPA parameter and cannot communicate on the LAN until a DHCP server is found and IP and server names are acquired.

Note that the AT+iFD command disables iChip's DHCP client. In order to re-activate the DHCP client process, you need to perform a HW or SW reset. This command also resets iChip's active IP address stored in the IPA parameter.

23 DHCP Server

iChip's DHCP server allows it to manage a network segment when no DHCP server is available. When iChip is configured to operate in iRouter mode, it provides access to the public internet via its modem connection. The DHCP server can handle up to 255 IP addresses concurrently.

Two parameters govern DHCP server functionality:

- DPSZ: The DHCP pool size parameter determines the range of IP addresses that iChip allocates for its clients.
- DSLT: The DHCP server lease time determines the lease time that iChip grants when assigning IP addresses.

The DHCP server is activated under the following conditions:

- An IP address is defined by the DIP parameter.
- The DPSZ parameter is set to a value greater than 0.
- Executing a software reset (AT+iDOWN) following the above settings.

When activated, iChip's DHCP server assigns IP addresses starting from DIP+1 up to DIP+DPSZ. In addition, the DHCP server offers the IP address stored in the IPG parameter as a gateway to clients, and the mask address stored in its SNET parameter as a Sub-Net. The assignment policy of iChip is as follows:

1. iChip attempts to assign the same IP for the same MAC address.
2. iChip starts re-using addresses only after using all the addresses in the pool.
3. iChip attempts to re-use the oldest expired address first.
4. iChip attempts to ping the address it is about to assign in order to avoid assigning an address already used.
5. iChip offers its SNET parameter as a Sub-Net. If SNET is 0.0.0.0, iChip calculates a new one according to address class.
6. iChip offers its IPG parameter as a gateway. If IPG is 0.0.0.0, iChip offers its IP address as a gateway.
7. iChip offers the primary IP address of the Domain Name Server stored in its DNS1 parameter to the client, provided it is not 0.0.0.0. If DNS1=0.0.0.0 and bit 6 of the SDM parameter is set to 1, iChip withholds responses to DHCP requests until DNS1 is resolved. This configuration is useful when iRouter feature is enabled.

24 IP Registration

When iChip goes online in a dial-up environment, it is normally assigned a dynamic IP address during PPP establishment. Since a different IP address is usually assigned every session, it is not practical to use iChip as a server, since the clients do not know what IP address to use. Furthermore, under these restrictions, there is no practical way to know whether a specific system is online or offline. A similar problem occurs when using the iChip LAN, which is configured to use a DHCP server. In this environment, a different IP address is usually assigned every time the iChip LAN boots and connects to the LAN.

To overcome this problem, iChip incorporates built-in procedures designed to register its IP address on a server system each time it goes online. Once registered, client systems may interrogate the servers in order to verify the online status of a specific system and retrieve its currently assigned IP address. The IP registration process is governed by several AT+i parameters. Once these parameters are configured, iChip registers its IP address accordingly when it goes online as a result of an explicit AT+i command (AT+iUP) or as a result of automated Internet session establishment procedures, such as a triggered Internet session or when going online as a SerialNET mode server.

In cases where iChip uses a NAT gateway to the Internet, it can be configured to register the NAT's IP address and a special port that is linked to iChip in the NAT's configuration. See details in the RRRL parameter description. When this is the case, the RRRL parameters (IP and port) are used instead of the local IP and port values that iChip is assigned, in all registration methods (RRMA, RRSV, and RRWS).

iChip includes several IP registration methods, as described below.

E-Mail Registration

iChip registers itself by sending an e-mail that contains its ID information and current IP address. When the RRMA parameter contains an e-mail address, iChip sends an e-mail containing its current IP address or its RRRL to the address defined in RRMA during the registration procedure. The syntax of the e-mail body is:

```
<BDY parameter contents>
```

```
iChip-<D/L/S> S/N:<RP5> Version:<RP1> HN:<HSTN> IP:<IPA or RRRL>
Port:<LPRT or 80 or 0> http:// <IPA or RRRL><CR><LF>
```

The subject line of the e-mail is:

```
"RING RESPONSE LINK From: iChip-<D/L/S> S/N:<RP5> Version:<F/W ver>
HN:<HSTN> IP:<IPA or RRRL> Port:<LPRT or 80 or 0>"
```

where,

Port is LPRT if in SerialNET mode; 80 if not in SerialNET mode and AWS is enabled, and 0 if not in SerialNET mode and AWS=0. The receiving end may refer to the contents of the subject line to filter out this e-mail message.

Socket Registration

iChip registers itself by opening a socket to a registration server and sending its ID information and current IP address. When iChip's RRSV parameter contains a value, iChip establishes a socket to the server defined in RRSV during the registration procedure. When a socket is established, iChip transmits its ID information and current IP address (or the RRRL) in the following format:

```
"iChip-<D/L/S> S/N:<RP5> version: <RP1> HN:<HSTN> IP:<IPA or RRRL>
Port:<LPRT or 80 or 0>"
```

The registration socket is then closed.

Web Server Registration

iChip registers itself by surfing to a web server with its ID information and current IP address as parameters.

If the RRWS parameter contains a URL (of a registration web server), iChip registers its ID information and IP using the URL by issuing a GET command along with a fixed format parameter line:

```
"<RRWS path>?SN=<RP5>&IP=<IPA or RRRL>&WPt=<0 or the port defined in
RRRL>&HN=<HSTN>".
```

The web server must contain a CGI, .asp page, exe, etc., which make use of these parameters to register the iChip.

If several registration parameters are configured, iChip goes through multiple registration processes. If more than one registration process fails, iChip returns an I/ERROR describing the first failure encountered. If all registrations fail, iChip returns I/ERROR(90).

25 Network Time Client

iChip incorporates a Simple Network Time Protocol (SNTP) client. With this protocol support, iChip can be configured to check SNTP servers for current time and date each time it goes online. iChip is configured to retrieve time data from a Network Time Server each time it goes online with the NTOD parameter. After updating its internal Time-Of-Day (TOD) registers at least once, iChip continues to keep track of time independently, even after it goes offline.

When iChip contains real TOD data, e-mails sent are automatically stamped with Time and Date of delivery, according to RFC (822) definition for the date header field. In addition, the AT+iRP8 report returns the current time and date.

iChip also contains parameters to configure local GMT offset and a DSTD (Daylight Saving Time) rule. These parameters allow iChip to determine the local TOD. When iChip is configured for TOD retrieval from a Network Time Server, iChip automatically retrieves an updated time reading every two hours while online. This configuration improves the long-term accuracy of its internal time management.

26 Parameter Profiles

Introduction

iChip contains an internal database of non-volatile environmental parameters, which stores a configuration that is used by iChip to determine its operation in the network environment it is configured for.

Each iChip parameter has a "Factory Default" value, which it is assigned to that parameter as a default, before any other value is assigned to it. The AT+iFD command is available to restore all iChip parameters to their Factory Default at any given time.

The Parameters Profiles feature allows OEM manufacturers or iChip users to define a different set of default values, described as a "Profile". The Profile may then be restored at any given time.

The command AT+iSPRF may be used to store the current "snap shot" of iChip parameter values as a Profile. Currently only one profile is supported, however in future versions support for additional profiles may be added.

The Profile is stored in a dedicated storage space, which may be referenced at a later time to be loaded into the active iChip parameters. The command AT+iLPRF may be used to load the saved profile.

The command AT+iDPRF may be used to display a report of the profile. The report includes one line for each iChip parameter with the syntax:

<*Parameter Name*>=<*Value*> where, *Parameter Name* is the AT+i name code used.

+iSPRF — Store Parameters Profile

Syntax:	AT+iSPRF[:*n*]
	Store the current system state (all iChip parameter values) as Parameter Profile *n*.
Parameters:	*n* – Optional Profile number
n=1	Parameter Profile #1
Default:	1
Result code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise

+iLPRF — Load Parameters Profile

Syntax:	AT+iLPRF[:*n*]
	Load Parameter Profile *n* and restore it as the system state. This operation is destructive – it will change the values of the system parameters without an option for rolling back.
	The loaded settings will take effect following a hardware or software reset.
Parameters:	*n* – Optional Profile number
n=1	Parameter Profile #1
Default:	1
Result code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise

+iDPRF — Display Parameters Profile

Syntax:	AT+iDPRF[:n]
	Display the context of Profile *n* by displaying pairs of name and value. Each pair will be displayed in a new line. Note that passwords are displayed with '*' signs instead of plaintext.
Parameters:	*n* – Optional Profile number
n=0	Current System state
n=1	Parameter Profile #1
Default:	1
Result code:	One line for each iChip parameter with the syntax: <*Parameter Name*>=<*Value*> Where, *Parameter Name* is the AT+i name code used.
Followed by:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise

27 Easy Network Configuration

A simple first-time configuration methodology is available for iChip devices in situations where:

- A Host port is not readily available to accept AT+i commands
- It is preferable to configure iChip over the network.

Introduction

iChip may be configured using a combination of the following methods:

1. Issue AT+i commands to the iChip Host port (Serial, USB or SPI). AT+i commands allow setting each individual parameter value. Since the iChip parameters are non-volatile, this setting may be a one-time procedure.
2. Connect the iChip USB or Serial port to a PC running Windows and activate the iChipConfig GUI application. This application provides a convenient Windows GUI with which to configure iChip. The actual configuration is taken care of in the background with AT+i commands.
3. Activate iChip's Web server and use a standard Browser to surf to iChip's configuration Web site, where most of iChip's parameters may be viewed and updated.
4. Upload an RPF file via iChip's Configuration Web site. The iChip Webserver must be active. An RPF file contains iChip parameter names and value settings. Loading an RPF has the equivalent affect of setting the iChip's parameters to the specified values.

Methods 1 and 2 require a physical connection to the iChip Host port (Serial, USB or SPI). Methods 3 and 4 require an active IP connection to the network in order to properly allow parameter configuration.

In many cases, neither of these connections is readily available **before** a preliminary configuration is made.

Therefore, an additional method has been added to provide for an easy preliminary configuration over a network, in what is called "**Easy-Configuration**" mode.

Preliminary Network Based Configuration

The Easy-Configuration mode supports a simple configuration methodology in cases where the physical iChip Host port is not available and iChip needs to be configured (in at least several parameters) in order to connect to the network. This methodology is based on the Parameter Profile feature.

It is assumed that in these cases, a Parameter Profile will be devised and stored in iChip to allow setting up a temporary IP connection, which may then be used for preliminary configuration purposes. For example, given a WPA secured WiFi network, it is required to configure an SSID and Pass-Phrase to connect to the Access point. In this case a temporary network connection may be devised to support an Ad-Hoc connection on a specific (pre-known) channel, SSID and a fixed IP address. The AP SSID and Pass-Phrase may then be configured through that network using a standard browser.

iChip Facilities to support Easy Configuration

Several iChip facilities support Easy-Configuration:

Auto Start Web Server

When the +iAWS (Auto Web Server) parameter: +iAWS>0 iChip enables its Web Server under all circumstances. Furthermore, when enabled, the Web server is accessible from any iChip network interface. For example, if iChip supports both a LAN network interface and a cellular network interface, when +iAWS>0 the iChip configuration site may be accessed from the LAN network in order to configure the cellular parameters required to create a connection on the cellular network.

Network Configuration Mode

Network configuration mode is governed by a special set of values that may be assigned to the +iAWS parameter:

- 200 -- Use SSL3 Secure Web Server for network configuration
- 201, 202 or 203 -- Use Web Server with 2, 4 or 6 sockets respectively for network configuration

When +iAWS≥200 the iChip Configuration Web site (which is part of the iChip firmware) displays a special Network Configuration page. This page is dedicated to configuring the required parameters that need to be defined in order to connect to a specific network.

It is envisioned that this configuration shall be a one-time configuration, under a temporary profile. It shall be used for a first time configuration to get iChip connected to a specific network, whose configuration is unknown at the time of manufacturing the product.

After the one-time configuration, it is assumed that the +iAWS parameter is assigned a value less than 200 and therefore iChip will connect to the required network and the regular iChip Configuration Web site shall be displayed and used to fully configure the remaining iChip parameters.

iChip Network Configuration Web Page (when +iAWS≥200)

The network configuration web site is divided into four configuration sections:

- WiFi
- LAN
- Dialup
- Miscellaneous

See Fig 1 below.

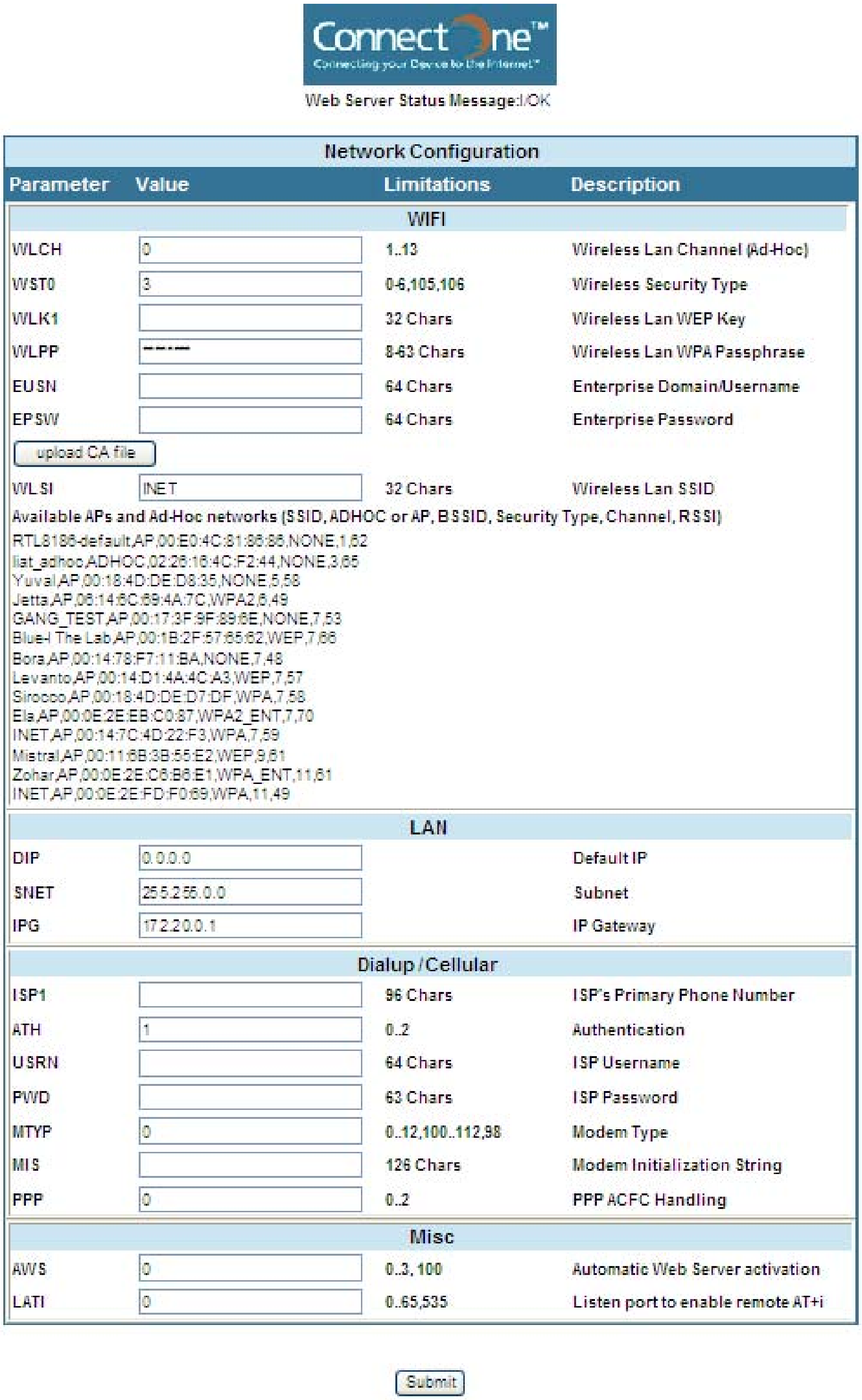
The WiFi section contains dialog boxes to configure an Ad-Hoc connection or an AP (infrastructure) connection, as well as security related modes, pass-phrases and certificates. It also includes a subsection which lists the available AP's and Ad-Hoc networks in the near vicinity.

The LAN section contains dialog boxes to configure the iChip's IP and gateway, or may be left clear to obtain IP addresses from a DHCP server.

The dialup section contains dialog boxes to configure a modem, dialup and PPP related parameters.

The last section contains several miscellaneous fields related to future configuration capability:

- AWS: The AWS parameter, which defines the availability of the Web server after power-up.
- LATI: The LATI parameter may contain a non-zero port number, which iChip will Listen on. When configured, a TCP socket connection to this port will allow an alternative route for iChip AT+I commands and replies (see AT+I Programmers Manual).



Web Server Status Message:I/OK

Parameter	Value	Limitations	Description
Network Configuration			
WIFI			
WLCH	0	1..13	Wireless Lan Channel (Ad-Hoc)
WST0	3	0-6,105,106	Wireless Security Type
WLK1		32 Chars	Wireless Lan WEP Key
WLPP	******	8-63 Chars	Wireless Lan WPA Passphrase
EUSN		64 Chars	Enterprise Domain/Username
EPSW		64 Chars	Enterprise Password
upload CA file			
WLSI	INET	32 Chars	Wireless Lan SSID

Available APs and Ad-Hoc networks (SSID, ADHOC or AP, BSSID, Security Type, Channel, RSSI)

RTL8186-default,AP,00:E0:4C:81:86:86,NONE,1,62
liat_adhoc,ADHOC,02:26:16:4C:F2:44,NONE,3,65
Yuval,AP,00:18:4D:DE:D8:35,NONE,5,58
Jetta,AP,00:14:6C:69:4A:7C,WPA2,6,49
GANG_TEST,AP,00:17:3F:9F:89:6E,NONE,7,53
Blue-I The Lab,AP,00:1B:2F:57:65:62,WEP,7,66
Bora,AP,00:14:78:F7:11:BA,NONE,7,48
Levanto,AP,00:14:D1:4A:4C:A3,WEP,7,57
Sirocco,AP,00:18:4D:DE:D7:DF,WPA,7,58
Ela,AP,00:0E:2E:EB:C0:87,WPA2_ENT,7,70
INET,AP,00:14:7C:4D:22:F3,WPA,7,59
Mistral,AP,00:11:6B:3B:55:E2,WEP,9,61
Zohar,AP,00:0E:2E:C6:B6:E1,WPA_ENT,11,61
INET,AP,00:0E:2E:FD:F0:69,WPA,11,49

Parameter	Value	Limitations	Description
LAN			
DIP	0.0.0.0		Default IP
SNET	255.255.0.0		Subnet
IPG	172.20.0.1		IP Gateway
Dialup / Cellular			
ISP1		96 Chars	ISP's Primary Phone Number
ATH	1	0..2	Authentication
USRN		64 Chars	ISP Username
PWD		63 Chars	ISP Password
MTYP	0	0..12,100..112,98	Modem Type
MIS		126 Chars	Modem Initialization String
PPP	0	0..2	PPP ACFC Handling
Misc			
AWS	0	0..3, 100	Automatic Web Server activation
LATI	0	0..65,535	Listen port to enable remote AT+i

Submit

Figure 27-1: Special Network Configuration Web Page

Summary of Network Configuration Methodology

The following guidelines summarize the Easy-Network-Configuration methodology:

- Create and store a temporary network configuration Parameter Profile in iChip. The parameter values configured, should reflect a standard preliminary state, which is a convenient starting point from which to manage the final configuration onsite. Make sure to assign +iAWS=20*x* in the Parameter Profile, so the special network-configuration page is displayed.
- When in the field pull the MSEL signal LOW for +30 seconds
- Use the temporary network settings (as defined in the stored profile) to browse to iChip's configuration Web site
- Because +iAWS≥200 you will receive the special Network-Configuration Web page discussed above
- Configure the relevant Network parameters for the specific environment
- Assign +iAWS<200 and SUBMIT the configuration
- iChip should reboot and connect to the specific network it was configured for
- Try accessing the iChip from the current network. Optionally, continue configuring iChip using its normal configuration Web site
- If iChip did not successfully connect to the specific Network, pull MSEL LOW for +30 seconds to reinforce the parameter profile and start over from the 3rd step.

Case Study Example

Assume iChip is embedded in a monitoring apparatus, which includes several sensors, an application MCU and connects over WiFi to an external AP. The iChip is interfaced to the application MCU on its serial port.

Since the monitoring apparatus does not include a keyboard and display – nor does it contain any external communication connectors such as serial or USB – configuring the device to connect to the AP presents a problem, since it necessitates specifying the required SSID, security type and security Pass-Phrases or keys.

This problem may be overcome with the Easy-Configuration method in iChip.

Starting from iChip's initial Factory-Defaults state, the following parameters are preconfigured and then stored in the Parameter Profile:

AT+iWLCH=11	<Define a WiFi channel>
AT+iSSID=!easyconf	<Configure an Ad-Hoc network called *easyconf*>
AT+iDIP=192.168.1.1	<Define a default IP address>
AT+iDPSZ=1	<Configure iChip as a DHCP server for a single client>
AT+iRPG=mypass	<Configure a Remote (Web) Update Password>
AT+iAWS=201	<Configure iChip for 1st-time Network-Configuration>
AT+iSPRF	<Store this configuration in a Parameter Profile>

When the monitoring apparatus arrives in the field, the technician collects the relevant configuration info for the current venue. After powering up the device, the technician may use a standard PC, Laptop or PDA to find and connect to the *easyconf* Ad-Hoc network, which iChip creates. If the technicians system is configured for obtaining its IP addresses from a DHCP server, it shall be allocated IP 192.168.1.2 as it connects to iChip's Ad-Hoc network.

The technician may then open a standard browser and surf to: http://192.168.1.1/iChip, which is iChip's standard configuration Web site.

The following dialog shall be displayed:

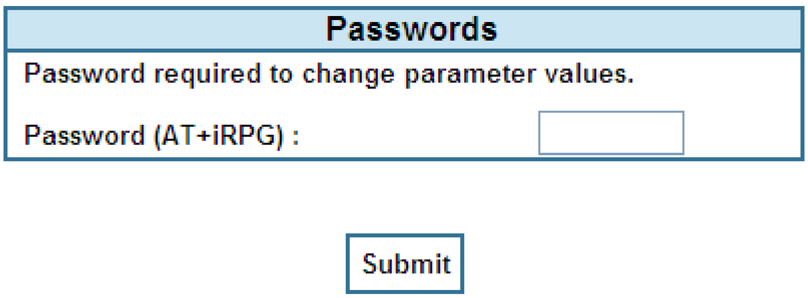


Figure 27-2: Website Password Protection

Since in this case the +iRPG password parameter has been set to *mypass* – enter that password and click on submit.

The network configuration page shall be displayed, as seen in Fig 1 above.

Using this dialog it is possible to configure the current venue's SSID and security parameters, as well as iChip's default IP (or 0.0.0.0 to use the venue's DHCP server).

Note that the +iAWS parameter is cleared to 0, but may be configured to any other value. For example, setting AT+iAWS=1 shall enable iChip's Web server after rebooting.

When the network configuration is complete, click on the submit button. iChip should now configure its parameters with the new settings and reboot. The changes made on the network configuration page should cause iChip to connect to the current venue's network. Assuming this was successful and that the +iAWS parameter was configured to enable the iChip Web server – iChip's configuration Web site can now be browsed over the current venue's network. Additional iChip parameters can then be configured.

If, however, the network configuration was not successful, the iChip MSEL signal may be pulled LOW for just above 30 seconds in order to re-instate the Parameter Profile, in which case iChip shall return to create the preconfigured Ad-Hoc network, through which it may be configured again.

28 Remote AT+i Service

Introduction

The LATI parameter allows configuring iChip to maintain a communication channel that supports interacting with iChip from a remote location using the AT+i command set as if the commands are administered through the local serial port. When LATI is set to a non-zero value, iChip opens a TCP listening socket on port *<LATI>*. This listening socket can be used to connect to iChip's remote AT+i service.

In a dial-up environment, iChip opens a TCP listening socket on port *<LATI>*, only after the PPP connection is established. However, connecting to the *<LATI>* port is allowed through either the LAN/WiFi or the dial-up communication platform regardless of the setting of the +iCPF parameter. As a result, iChip attempts to resolve an IP address on the LAN/WiFi end even when +iCPF is set to 0 (Modem) and supports setting up a TCP socket to the *<LATI>* port if it contains a port value (>0).

The LAR parameter restricts the remote client from residing on the LAN/WiFi side or on the modem side.

Remote AT+i Commands

When a remote client connects to iChip's LATI socket, iChip redirects the socket's data flow to the AT+i parser, in effect allowing the socket to take over the parser. Any data coming from the socket is processed by iChip as if it came from the host serial port and the replies are returned to the socket instead of being sent to the host serial port. iChip replies with an I/BUSY to commands coming from the host serial port, while the remote client is connected.

An exception to this is the (+++) escape sequence. On detection of (+++) from the host serial port, iChip closes the remote connection and reboots.

If iChip was in the process of performing some Internet activity initiated by the host at the time the remote client connected, iChip allows this activity to end and the final reply to reach the host before passing control over the parser to the remote client.

Closing a Remote AT+i Session

To close a remote AT+i session, the remote client can choose to issue AT+iDOWN via the socket. In response to this, iChip restarts. Only I/OK is returned over the socket before it is closed by iChip. Alternatively, the remote client can close the socket in order to disconnect, leaving iChip's Internet session as-is. In the latter case, iChip returns control over the parser to the local host port. The LATI listen port remains open, available to service additional remote connections. After a LATI session is closed, the LSR (last session error) web parameter contains the value 096 to indicate that a LATI session has been disconnected.

Note: (+++) sent over the LATI socket is not recognized as an escape sequence.

SerialNET Initiation/Termination Using a Remote AT+i Session

A remote client may connect to iChip's LATI socket and instruct initiation or termination of SerialNET mode. Please refer to the chapter discussing SerialNET Theory of Operation for a complete description.

Caveats and Restrictions

- When iChip in dial-up mode is in auto baud rate detection mode (after re-starting with BDRF=a or in response to the AT+iBDRA command), a remote AT+i session cannot be established, even if the LATI parameter contains a port value.
- In iChip LAN the remote AT+i service is available, even if iChip LAN is in auto baud rate detect mode. However, once the remote AT+i connection is established, iChip LAN will no longer be in auto baud rate mode and the host will be able to send the (+++) escape sequence only at 9600 baud, if it needs to close the remote session. iChip LAN will then return to auto baud rate detect mode when and if the local host or the remote client close the LATI session, in effect re-starting iChip LAN.
- During a remote AT+i session, the remote client taking over the parser cannot make use of iChip's mechanisms of Hardware or Software flow control, which exist for the local host port. The only mechanism iChip will use is this mode is TCP level flow control (using the TCP window).
- In iChip LAN, the command AT+iBDRA will return **I/OK** but will not initiate a baud rate detection process.

29 iChip Parameter Update

Introduction

The iChip remote parameter update file allows users to remotely modify various non-volatile parameters in iChip products. The file is an ASCII-formatted text file, edited by the user or created by a dedicated application. The file's size must not exceed 10k.

The remote parameter file (RPF) naming convention is <*filename*>.rpf. If a parameter is assigned a legal value within the file, that value replaces the current value in iChip's non-volatile parameter database. A parameter value that is not referred to in the file, or that is not defined using the correct syntax rules, specified below, does not affect the current parameter value.

Remote Parameter File (RPF) Structure

The RPF file must include the letters "RP_" as its first 3 characters, and can include additional header lines (defined below), as well as various parameter assignments. Assignments follow the rules defined for parameter settings, but excluding the AT+i prefix. For example, to assign the value `myname` to the POP3 mailbox name parameter, the correct assignment is `MBX=myname` This is equivalent to the host sending `AT+iMBX=myname` to iChip. Each line, terminated with <CR>/<LF>, can contain one assignment only. The order of assignments is not important, except for the RPF header parameters, which must be first and must follow the header definitions below. After the first non-RPF header parameter, additional header parameters are ignored.

Comment lines can appear anywhere in the file. Comment line syntax is defined as: `#<anything>CR/LF`

The first line in the file that is not a comment line is considered the authentication header line and must have the following syntax:

`RP_[GROUP=<string><space_character>][RP_DEST=<string>]CR/LF`

The remainder of the header must contain lines with the following syntax:

`<header_parameter_name>=<general_parameter_value>CR/LF`

Header Parameter Names and Values

Name	Value	Default
RP_DEST	Single string, no space characters	NONE
RP_GROUP		NONE
RP_START_FROM_FACTORY_DEFAULTS	YES/NO	NO

Table 29.1 Header Parameter Names and Values

- **RP_GROUP** — If the RPG Group/Password parameter contains a value, the RPF file must include an RP_GROUP definition and its value must be identical to the RPG value. Otherwise, the parameter update file will be rejected. Nevertheless, if the RPG parameter is set to the special value (*) (match any), the RPF file will be accepted with any value of RP_GROUP, as well as without any value at all. The RPG Group/Password parameter can be viewed and changed by sending an AT+iRPG? command to iChip.
- **RP_DEST** — If the RPF file contains this parameter, the parameter update file will be rejected unless the value given in this parameter is identical to the unique ID of the iChip it was sent to. The unique ID can be viewed by sending an AT+iRP5 command to iChip, but cannot be changed. This feature facilitates sending a parameter update to a specific iChip controller only.
- **RP_START_FROM_FACTORY_DEFAULTS** — This flag defines the initial value of parameters. A YES value will initially restore all iChip parameters to their factory default values before processing the new RPF file values.

Uploading a Parameters Update File to iChip

By default, the RPG parameter does not contain a value, thus disabling the option to receive and process a parameters update file in the iChip. To enable this option, the RPG parameter must be set to some value. If a value other than (*) is set, the value must match the parameters update file RP_GROUP value. This feature facilitates group updates, and can be used as a password to secure parameter updates.

A remote parameters update file can be uploaded to iChip using iChip's internal configuration site.

Note: See Appendix B for a sample RPF file.

30 Remote Firmware Update

Introduction

iChip accepts remote firmware updates from an HTTP or FTP server. The firmware update is stored as an .imz file on the host server and downloaded by iChip acting as a client. iChip replaces its existing firmware with the new one through a special application that is part of the .imz file. This method is especially convenient when managing firmware updates in a globally distributed install base of internet-enabled devices.

Updating Firmware from a Remote Server

This method involves placing the firmware update .imz file on an HTTP or FTP server. iChip has the provisions to use its respective HTTP or FTP client to download the firmware update file and perform the update process.

Before the actual remote firmware update command can be issued, the following parameters must be set:

- USRV — Defines the protocol to be used (HTTP or FTP), and the name of the host on which one or more .imz files are stored.
- UUSR — Defines FTP user name (FTP only).
- UPWD — Defines FTP user password (FTP only).
- UEN — This flag indicates whether iChip updates to a firmware version that is newer than the currently installed one only, or to any firmware version it finds.

In addition, an appropriate .imz firmware update file must be placed on the remote server at the location specified by the USRV parameter.

Once the above parameters are defined, the firmware update process can be initiated by sending the following command to iChip:

AT+iRFU

iChip returns **I/OK** to acknowledge receipt of the command. As the update process may take up to 4 minutes to complete, iChip issues an **I/UPDATE** message to notify the host that it is in the process of updating its firmware. The host must allow for an extended delay period until iChip completes the process. Once completed, iChip re-boots the new firmware and issues an **I/DONE** message when in dialup mode, or an **I/ONLINE** in LAN mode.

Several safeguards have been instated to ensure a successful firmware update. The firmware update file is structured by Connect One in a specific format, which allows iChip to authenticate its origin as a legal firmware image. iChip also verifies that the firmware update is the correct version for its hardware environment. iChip rejects an update file if it contains an image that is identical to the one already installed.

The remote firmware update procedure is detailed below:

1. iChip downloads the new firmware imz file.

2. If the download fails, iChip returns an error message and continues to work as before.
3. If during the download iChip is going over a reset cycle (SW or HW), iChip re-boots and executes the old firmware.
4. If the download is successful, iChip authenticates the firmware image file.
5. iChip replaces the old image with the new image.
6. If the replacement process fails, for example due to power failure, iChip re-boots from boot loader in the flash memory and re-tries the replacement process until successful.
7. If the replacement process is successful, iChip re-boots and executes the new firmware.

+iRFU — Remote Firmware Update

Syntax:	AT+iRFU
	Downloads and updates iChip firmware from a remote HTTP or FTP server. The value of the USRV parameter is used to determine the remote server from which to download the firmware. The value of the UEN flag is used to determine whether to update any firmware version or only a version that is newer than the one already installed. In addition, if an FTP server is specified for download, the UUSR and UPWD parameter values are used to determine FTP user name and password.
Result Code:	
I/OK	To acknowledge successful receipt of the command
I/ERROR	Otherwise
Followed by:	
I/UPDATE	If a qualifying firmware update .imz file is found
I/ERROR	Otherwise
Followed by:	
I/DONE	After successfully updating new firmware in dialup mode
I/ONLINE	After successfully updating new firmware in LAN mode
I/ERROR	Otherwise

31 iChip RAS Server

Introduction

iChip features an internal Remote Access Server (RAS) that allows a remote dialer to dial into iChip using an active modem platform. When configured as RAS, iChip answers the incoming call and negotiates a PPP connection.

iChip's RAS supports acknowledging an IP address request from the remote dialer side, as well as assigning a default IP address. Once the connection is established, the client can browse iChip's website. (If the AWS parameter is set to a non-zero value.) All other iChip IP protocol functionality is also enabled, allowing the host to issue Internet protocol AT+i commands based on the PPP connection. Note, however, that since iChip is not connected to an actual ISP in this mode, iChip does not have access to the public Internet and thus only direct connections between iChip and the connected PPP client are possible.

RAS Parameters

Three parameters govern the use of iChip's RAS server:

RAU	RAS Login User Name The RAU parameter defines the allowable user name for login purposes when iChip answers an incoming call as a RAS. The remote dialer must specify the correct user name and matching password in order to successfully complete the PPP connection. This parameter must have a non-empty value for the RAS feature to be enabled. Otherwise, when RAU is empty, iChip's RAS is effectively disabled. When RAU contains the special character (*), RAS is enabled but no authentication is required.
RAP	RAS Login Password The remote dialer must provide the correct password in order to successfully complete the PPP connection. When the RAP parameter is empty or contains a (*), any password string is accepted, in effect nullifying the authentication process.
RAR	Number of RINGs before picking up the line. When the RAS feature is enabled, the RAR parameter defines the number of RINGs that must arrive before iChip picks up the line and transfers control to its RAS.

RAS Theory of Operation

When a remote client dials into iChip, the modem RING strings are transferred by iChip (which defaults to transparent mode) to the host. When the RAS feature is enabled (RAU contains a value), iChip picks up the line and negotiates a PPP connection by issuing the ATA (modem) command after RAR RING strings have been received.

If the host chooses to manage a direct (modem-to-modem) data connection, it can pick up the line before RAR RING strings have arrived by issuing the ATA modem command.

During RAS PPP negotiations, iChip will replies only to (+++) (escape sequence) and AT+iRP*n* commands. Specifically, iChip replies “Connecting as RAS” to the AT+iRP2 (iChip status) command. The escape sequence can be used to abort the RAS session at any time. The AT+iRP2 command is the only means for the host processor to determine that a PPP session is in progress. iChip manages the RAS protocol internally and does not transfer any information to the host. Any other commands received from the host are disregarded by iChip.

Once the PPP connection has been fully negotiated and established, iChip responds to all AT+i commands as when it is online. Specifically, iChip replies “RAS Connected” to the AT+iRP2 command.

After a RAS PPP connection is established and IP addresses are assigned, iChip automatically activates the internal web server, if the AWS parameter is set to a non-zero value. Thus, the remote client can browse iChip’s website.

RAS IP Configuration

As part of the PPP negotiation, iChip assigns itself the default IP 192.168.0.1 and allocates 192.168.0.2 as the client IP. When the DIP parameter contains a fixed IP value (other than 0.0.0.0), iChip will assign the remote RAS Client with an IP of DIP while iChip will assume an IP of client IP – 1 (equals DIP-1). However, if the client requests a specific IP, iChip always grants the client’s request and uses the client’s IP minus 1 as its own IP.

The following restriction apply: If the DIP value causes the client address to have an LSB of 000 or 001, iChip will force the LSB to 254. For example, DIP=10.0.0.1 will result in a Client assignment of 10.0.0.254.

If the DIP value causes the client IP to have an LSB of 255, iChip decrease the IP by one and forces it to have an LSB of 254. For example, DIP=10.0.0.0 will result in a Client assignment of 10.0.0.254.

If the IP requested by the client minus 1 is an IP address that ends with 000 or 255 as the last nibble, iChip assigns itself with the client’s IP *plus* 1 instead of minus 1. This is done to assure that the IP that iChip assigns itself never violates the rule that defines that a network or host IP segment may not be all binary 1’s, nor all binary 0’s.

Auto PPP RAS Mode

iChip allows combining RAS and direct modem-to-modem communication sessions. A special mode, named Auto PPP RAS, supports dialing into the iChip with a PPP dialer or a regular modem.

Auto PPP RAS mode is enabled by enabling RAS mode *and* adding a +100 offset to the RAR parameter, where [<*RAR*>-100] determines the number of RINGS after which iChip automatically picks up the line and negotiates a PPP connection. The host processor can instruct the modem to pick up the line beforehand by issuing the ATA (modem) command or by setting the modem to auto-answer after less than [<*RAR*>-100] RING strings. This is normally done in order to manage a direct modem-to-modem (non-PPP) communication session.

When iChip is in the Auto PPP RAS mode, it monitors the data stream following the modem CONNECT line. If the first character transmitted by the remote end is (~) (0x7E), iChip defers to PPP negotiation. The (~) is the last character transmitted to the host end to signal that iChip has taken over the negotiations. Upon this event, iChip continues to negotiate a PPP connection internally in a manner similar to the procedure that occurs when iChip picks up the line after receiving <*RAR*> RING strings. If, however, the first character received from the calling end after the CONNECT line is not a (~) (0x7E), iChip remains in Transparent mode, and a regular modem-to-modem data session takes place.

SerialNET Mode

The RAS can also be enabled while iChip is in SerialNet mode. In this case, however, the modem RING strings are not forwarded to the host serial port. Once the PPP connection is established, iChip proceeds to act as it would after receiving a RING event and creating a PPP connection to a remote RAS server. That is, a listening socket is established on the LPRT socket, available for a SerialNET connection. This provides an alternative means to wake-up a SerialNET server device.

Lost Carrier

When iChip is online as a result of a RAS connection and the carrier signal is lost (due to an error or due to the PPP client closing the connection), iChip checks if the host used the PPP connection (tried to open an Internet session) during the connection. If the host did not use the connection, or iChip was in SerialNET mode, iChip silently performs a software reset and no indication of the disconnection is given to the host. Otherwise, if the host did use the connection, iChip acts as if this is a regular session created by the host that was terminated with a lost carrier signal. The error code is returned to the host on the next command that requires the use of the connection and only then will a software reset be performed.

Restrictions

Modem RING strings are not detected while the baud rate between iChip and the host is not yet established. This means that in order to use the RAS feature, one of the following must apply:

- BDRF is set to a fixed value (3-9 or h).
- iChip is in SerialNET mode with its baud rate defined by the SNSI parameter.
- An `a` or `A` was previously received from the host serial port and iChip has determined the host's baud rate.

In addition, Modem RING strings are not detected when iChip is in Modem Command (MCM) mode.

32 iRouter Mode

Introduction

iChip's iRouter mode is used to provide a gateway to a multitude of LAN or WiFi devices through a single dialup or cellular link. In this configuration, iChip's DHCP server may be used to assign IP addresses to the local hosts on the LAN/WiFi side. The DHCP server can be configured to withhold its responses until DNS settings are assigned by the ISP on the modem side. iChip also uses a Network Address Translator (NAT) to translate between local and public IP addresses.

While routing IP packets, iChip also accepts AT+i commands, as during normal operation. The CPF (Communication Platform) parameter selects which interface to use for Internet-related AT+i commands.

The following parameters and commands are used to configure iRouter mode behavior:

- Automatic Router Start (ARS) parameter — When set to 1, this parameter causes iChip to go online in iRouter mode upon power-up and start routing packets.
- Inactivity Timeout (IATO) parameter — When in iRouter mode, if no routing activity is detected for the period of time specified by this parameter, iChip disconnects its modem/cellular side and goes offline. After going offline and if ARS=1, iChip will go online and continue routing when the next packet that requires routing arrives.
- Start Router (STRR) command — Causes iChip to enter iRouter mode, go online on the dialup/cellular side, and start routing packets.
- Stop Router (STPR) command — Causes iChip to exit iRouter mode, go offline on the dialup/cellular side, and stop routing packets.

Establishing iRouter mode

iChip can be entered into iRouter mode using one of two possible methods:

- When the ARS parameter is set to 1, automatically and immediately after power-up and after every soft reset induced by AT+iDOWN.
- By issuing the AT+iSTRR (Start Routing) command.

Upon entering iRouter mode, iChip immediately goes online on the dialup/cellular side. Packets are not buffered during dialup/cellular connection establishment. After establishing the connection, iChip starts the routing service.

Basic Routing

When iChip is in iRouter mode, it routes packets between its two communication platforms utilizing a Network Address Translator (NAT) to translate between the internal IP address space used on the LAN/WiFi side and the real IP address used on the dialup/cellular side.

The NAT translates internal IP addresses of outgoing packets to the real IP address space and makes the reciprocal translation of packets received in response.

Note: When using an FTP client to connect to an external FTP server through the iRouter, you must use the FTP client in passive mode. For example, if the FTP client is an iChip, you must open the FTP session using AT+i@FTP.

Terminating iRouter Mode

iRouter mode is terminated by any of the following occurrences:

- By issuing the AT+iSTPR (Stop Routing) command. When iChip receives this command, routing services are stopped and iChip goes offline on the dialup/cellular side. If ARS=1 (Auto Routing), iChip automatically goes online and restores routing services when the next packet arrives.
- Automatically after an idle time period (with no routing activity) has passed. The idle time period is defined in the IATO (Inactivity Timeout) parameter. Idle time terminates routing only if IATO has a positive value larger than 0. When IATO=0, idle time termination is effectively disabled. If ARS=1 (Auto Routing), iChip automatically goes online and restores routing services when the next packet arrives.
- By issuing the (+++) escape string. iChip terminates iRouter mode and goes offline on the dialup/cellular side. Following an ESC sequence termination, iChip does not restore routing services even if ARS=1. To restore routing, either issue the AT+iSTRR command or, alternatively, if ARS=1– issue AT+iDOWN.

Configuring iChip when in iRouter Mode

While in iRouter mode, iChip can be configured using the same methods for iChip in general:

- Assuming iChip's website is enabled on the LAN/WiFi end, iChip's internal configuration website can be accessed by any browser that is connected to the same LAN/WiFi network.
- Assuming iChip's website is enabled on the dialup/cellular side, iChip's internal configuration website can be accessed by any remote browser connecting to iChip's port 80 over its public IP address.
- If the LATI parameter is set to a non-zero value, a remote host may open a socket to the LATI TCP port and send AT+i commands to that port. The LAR parameter restricts the remote host to residing on the LAN or the modem side.
- AT+i commands coming from the host application.

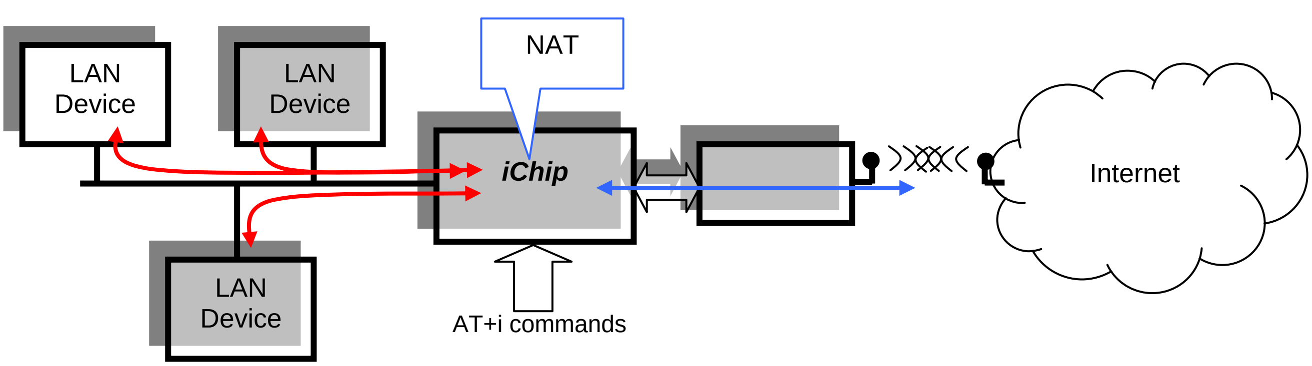


Figure 32-1: TCP/IP in iRouter Mode

AT+i Interface to iChip

In addition to configuring the iChip, AT+i commands can also be used to perform operations on either the LAN/WiFi or dialup/cellular communication platform.

Using the CPF (Communication Platform) parameter, you can select either one of the communication platforms. When CPF=0, AT+i commands are directed towards the dialup/cellular side; when CPF=1, they are directed towards the LAN/WiFi side. While processing AT+i commands, iChip continues to route packets seamlessly between the two platforms.

iChip's responses to AT+i commands depend on the CPF value, as well. For example, the IP returned by AT+iIPA? command while CPF=1 is the LAN-side IP.

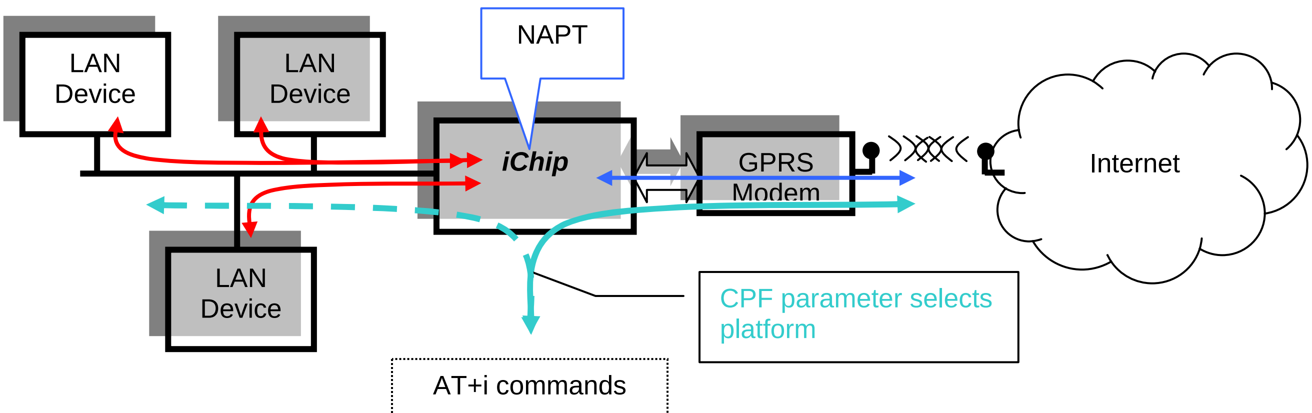


Figure 32-2: AT+i commands in iRouter Mode

Baud Rate Settings and Auto Baud Rate

iRouter mode supports all host and modem baud rates supported by iChip. However, when auto routing is set (ARS=1), iChip does not support Auto Baud Rate. This is due to the fact that in iRouter mode, iChip starts routing packets immediately after power-up, and skips auto baud rate determination.

Therefore, when configuring iChip for auto routing (ARS=1), you must set a fixed baud rate in the BDRF parameter.

iRouter and Power Save Mode

iChip can be configured for Power Save mode while acting as a router. Note, however, that there is no buffering of packets in iRouter mode. The first packet arriving to iChip while in Power Save mode triggers iChip to wake up and go online on the cellular or dialup modem. Only after establishing a connection, does iChip start routing packets. The packets received during connection establishment are lost.

Port Forwarding

The use of the NAT table provides the ability for each IP node on the local subnet to create a connection to an IP on the WAN. A device on the WAN can reply to messages originating from the LAN, however, it cannot create a connection into the LAN. This problem can be remedied by defining Port-Forward rules, which set aside designated ports on the WAN IP address of the iChip and automatically forward them to specific IP nodes on the local subnet.

Besides WAN (Modem) and LAN (Ethernet/WiFi) the iChip also supports a PPP connection to the host, when executing AT+iSPPP. Therefore there are three potential routing paths which should be provided with Port Forwarding:

1. LAN ⇔ Modem
2. LAN ⇔ Host-PPP
3. Host-PPP ⇔ Modem

The Port Forwarding rules can be defined to forward packets from the Modem port to the LAN or host-PPP ports and also from the LAN port to the host-PPP port. When specifying a Port-forward rule on the Modem platform, the destination (referred to as the local port) is dependant on the local IP. When iChip enters its routing function (in response to AT+iSTRR or after power-up when +iARS=1) it scans the Port-forward rules and compares the local-IP component with the actual LAN-side IP. Rules with a matching LAN subnet will be subsequently routed to the LAN side, otherwise routing shall default to the host end and the packets forwarded to the host over PPP.

The iChip can maintain up to 10 Port-Forwarding rules. Port Forwarding rules may be restricted to either TCP or UDP or non-restricted and forward either.

The syntax of the AT+i command to configure and manage Port-Forwarding rules is:
AT+iPFW*n*="[L | M]<*w-port*>,<*l-IP:l-port*>[,<*type*>]"

Where,

n	- Index in the range 0..9
L or M	- Optional indication whether *w-port* is on the LAN or the Modem.
w-port	- The WAN port. to be forwarded to an internal port.
l-IP	- An internal IP on the local LAN (or WiFi) or PPP connection.
l-port	- A port on l-IP.
type	- Optional: 0-TCP, 1-UDP restriction. Not specified: Both.

For example, given an iRouter system with WAN IP 10.0.0.100 and a Port-Forward rule:
AT+iPFW0="8000,192.168.0.1:80"
A packet (either TCP or UDP) to 10.0.0.100:8000 arriving at iChip from the Modem (WAN) end shall cause iChip to route the packet to 192.168.0.1:80 on the LAN and then route the response packet back to the originating system on the WAN.
The same Port Forwarding rule, restricted to TCP shall be:
AT+iPFW0="8000,192.168.0.1:80,0"
The same Port Forwarding rule, specifying the connection originates from the Modem:
AT+iPFW0="M8000,192.168.0.1:80,0"

Notes and Restrictions

- The Port-Forwarding rule must be enclosed in double-quotes.
- Newly assigned Port Forwarding rules take effect only after recycling power to the iChip or executing a soft-reset (AT+iDOWN).
- The Port Forwarding rules should not include two or more rules with the same local IP:Port.
- The Port Forwarding rules should not include two or more rules with the same Network and Map-port ([L | M]<*w-port*>).
- If these restrictions are not met, I/ERROR (600) is returned when trying to create the offending rule.

All existing Port Forwarding rules can be viewed with a dedicated report: AT+iRP22. The report line syntax is:

- [L|M]<*w-port*>,<*l-IP:l_port*>[,<*type*>]<CR/LF>

For example,

AT+iRP22
0- 8000,192.168.0.1 :80,0
3- 8800,192.168.0.5 :80
I/OK

+iSTRR — Start Router

Syntax:	AT+iSTRR
	Causes iChip to immediately enter iRouter mode.
	Upon entering iRouter mode, iChip immediately goes online on the dialup/cellular side. Packets are not buffered during dialup/cellular connection establishment. After establishing the connection, iChip starts the routing service.
Result Code:	
I/OK	When command is received and about to be processed.
Followed by:	
I/ONLINE	After successfully going online on the dialup/cellular side.
I/ERROR	Otherwise

+iSTPR — Stop Router

Syntax:	AT+iSTPR
	Causes iChip to exit iRouter mode, go offline on the dialup/cellular side, and stop routing packets.
	If ARS=1 (Auto Routing), iChip automatically goes online and restores routing services when the next packet arrives.
Result Code:	
I/OK	When command is received and about to be processed.
Followed by:	
I/ONLINE	After terminating the connection on the dialup/cellular side when CPF=1.
-or-	
I/DONE	After terminating the connection on the dialup/cellular side when CPF=0.
-or-	
I/ERROR	Otherwise

33 PPP Host Interface & Routing

Introduction

iChip's PPP Host Interface & Routing adds an API above the PPP protocol. In this mode it is assumed that a basic PPP stack exists in the host processor and that it is capable of negotiating a PPP connection. The host processor would be the PPP client in this connection. Once a PPP connection is established between the host and the iChip, the host processor may send PPP packets with destination being either the iChip itself or remote IP addresses, in which case the iChip will route those packets through its Modem or LAN interface. AT+i commands may still be addressed to the iChip through use of the LATI socket. If the LATI parameter is defined, the host may open a socket to the LATI TCP port (over PPP) and send its AT+i commands to that port.
The host processor may be connected to the iChip over UART or USB when enabling the PPP connection between them.

Connectivity Paths

The iChip may be connected to a dialup device, such as a cellular modem and/or to a LAN device, such as Ethernet or WLAN. The host processor establishes the PPP session using the command: AT+iSPPP. Command parameters allow the host to determine if iChip should establish an additional PPP connection over its modem link (assuming a modem exists). This feature supports up to three separate IP domains:
a. iChip to host over PPP.
b. iChip to LAN over Ethernet or WiFi.
c. iChip to modem network.

IP Addresses

When the iChip negotiates a PPP host connection, it shall assign itself the default private IP address: **192.168.0.1** and allocate the succeeding address: **192.168.0.2** to the client (the host processor). However, if the host requests a specific IP address during the PPP negotiation process, iChip shall grant the host's request and assign the host's address minus 1 as its own IP. For example, if the host requests the IP address 192.168.0.50, the iChip shall assign its own IP address to 192.168.0.49.
If the host's PPP stack is not setup to request an IP address, this may be done through a second, optional, parameter in the +iSPPP command. If this parameter is present, the iChip shall assign this IP address to the PPP client and assume this address minus 1 as its own. This optional parameter defines the IP address used in this PPP connection, regardless of other IP address settings.
The following restriction will apply to the "minus 1" rule: if the IP requested by the host minus 1 is an IP address that ends with 0x00 or 0xFF (decimals 0 or 255) as the last octet, iChip shall assign itself with the host's IP plus 1 instead of minus 1. This is done to assure that the IP which iChip assigns itself is legal and does violate the rule that defines that a network or host IP segment may not be all binary 1's, nor all binary 0's. For example, if the host requests 192.168.0.1 as its IP address, the iChip will assign itself: 192.168.0.2.

The IP address assignment rules in the PPP connection between the iChip and the host are summarized in the following table:

IP requested in +iSPPP command	IP requested by host in PPP negotiation	iChip PPP IP	Host PPP IP
IP	any	IP-1	IP
none	IP	IP-1	IP
none	none	192.168.0.1	192.168.0.2

Routing in PPP Mode

The iChip determines which connections are allowed, when to use NAPT and how to maintain the routing relationship. Port Forwarding entries and Routing Rules are factored into this calculation.

IP packets can originate in any of the three IP domains described in the chapter "Connectivity Paths". The iChip applies a NAPT algorithm where needed, in order to facilitate correct routing of data packets between the host processor and the modem and/or the LAN connections. The algorithm requires changing the source IP address and port number of the packets before transferring them to the specified destination networks. Responses which arrive back to the iChip are then distributed back to the original senders. Port Forwarding Rules can be manually added to the NAPT table as described in the "iRouter" chapter and the description of the PFW parameter.

Table 1 depicts the possible routing connections between networks and determines when a connection can be originated.

Table 33.1: PPP Routing Paths

Note 1.1: iChip Gateway Mode Configuration

Connection Originator	**Source IP**	**Destination IP**	**Source MAC**	**Destination MAC**	**Comments**
LAN Client	LAN Client	Internet	LAN Client	iChip MAC	See Note 1.2
LAN Client	LAN Client	Host IP	LAN Client	iChip MAC	See Note 1.2
LAN Client	LAN Client	iChip IP-LAN	LAN Client	iChip MAC	See Note 1.3
Host	Host IP	LAN Client	N/A	N/A	See Note 1.4
Host	Host IP	Internet	N/A	N/A	See Note 1.5
Host	Host IP	iChip Internal	N/A	N/A	
Modem	Modem IP	LAN Client	N/A	N/A	See Note 1.6
Modem	Modem IP	Host IP	N/A	N/A	See Note 1.6
Modem	Modem IP	iChip Internal	N/A	N/A	See Note 1.7

The iChip compares the setting of the IPG parameter with that of the IPA. If these values are equal, it indicates that the iChip is the LAN gateway. iChip's configuration as the LAN gateway has bearing on its routing behavior.

Note 1.2: Packets originated in the LAN

The iChip must be set as the gateway of the stations on the LAN. Routing to the Internet is achieved through iChip's modem connection. Routing to the Host IP is achieved through the PPP Host Interface.

Note 1.3: Packets with destination of iChip IP

If a NAPT entry matches the packet it will be routed accordingly otherwise it will be routed to iChip's internal stack.

Note 1.4: Packets sent from the Host to the LAN

When the host initiates a connection to a LAN station and the iChip is configured as the LAN gateway, NAPT is not being applied. When the iChip is not the LAN gateway then the CPF parameter must be directed towards the LAN (CPF=1). NAPT will automatically be applied. See Note 1.5.

Note 1.5: Packets sent from the Host to the Internet

If the iChip is NOT the LAN gateway, packets will be routed according to the CPF parameter, unless a matching NAPT entry is found in the NAPT table. If so they will be routed according to the relevant NAPT entry. This will allow a connection created on one platform (LAN/modem) to be persistent even when the CPF is changed after the connection was created.

If the iChip is the LAN stations' gateway: Packets with destination IP in the LAN Subnet will always be routed to the LAN. All other IP addresses will be routed to the Modem regardless of the CPF parameter.

Note 1.6: Packets arriving from the Internet via the modem

This situation is not possible, since the outside world is not aware of the Private IP assignments, so it is not likely that a connection to a private IP will be originated from the Modem side (or internet).

Note 1.7: Packets arriving from the Internet to the iChip

If a matching entry is set in the Port Forwarding NAPT table, the packets will be routed accordingly. Otherwise, these packets are handled by the internal TCP/IP stack.

Servers on the Internet can reply to packets which originated from the Host or LAN side by the nature of iChip's NAPT and there is no need for a manual Port Forwarding rule.

Note 1.8: Broadcast messages

Broadcast messages are handled by the internal TCP/IP stack of the iChip and are not routed to other networks.

Terminating the PPP Connection

An active PPP connection may be terminated by sending a Termination Request using the PPP Link Control Protocol (LCP). iChip will take the following actions:

1. If the Remote AT+i socket (LATI) was connected, iChip shall close it in an orderly manner.
2. iChip shall close all open sockets in an orderly manner.
3. The PPP connection shall be terminated.
4. iChip shall perform software reset and return to AT+i Command Mode on the Host Interface.

+iSPPP — Start PPP session

Syntax:	AT+i[!]SPPP:<mode>[,IP]
Parameters:	
Mode	0..2
IP	IP address to assign to the host in PPP negotiation
Command Options:	
n=0	Open PPP connection over the Host Interface (HIF) only.
n=1	Open PPP connection over the Host Interface and a PPP connection over the Modem Interface (MIF) using the dial string in ISP1.
n=2	Open PPP connection over the Host Interface and a PPP connection over Modem Interface using the dial string in ISP2.
!	Restart and open PPP connection on the Modem Interface only (previously determined the Host Interface is ignored)
Result code:	
I/OK	If mode and IP are legal values. If n>0 and '!' was not used this reply will only be given after a successful PPP session was opened over the Modem Interface, indicating to the host it can start PPP negotiation.
-or-	
I/ERROR	Otherwise

34 LAN to WiFi Bridge Mode

Introduction

LAN to WiFi bridge mode is a special mode in which iChip acts as layer 2 bridge between a LAN Ethernet network on one side and a WiFi network on the other side. This mode enables WiFi onto any device which accommodates a wired Ethernet connection. The iChip is responsible for the WiFi connectivity and security. Two modes of LAN to WiFi bridge are supported:

- Cable replacement with Ad-Hoc mode, between two iChips
- Cable replacement AP mode, between an iChip and an AP (Access Point)

Note that when iChip is configured for Bridge mode, it enters this mode immediately after power-up and automatic Host interface and baud rate detection are not supported. Therefore, the +iHIF and +iBDRF parameters must be defined. See description of +iBRM for more details.

In this mode iChip supports two types of LAN connections:

- PHY connection to iChip
- MII/RMII connection to iChip

When there is no PHY, iChip assumes the Ethernet is specified as a 100 Mbps, full-duplex connection.

Cable Replacement Ad-Hoc Mode

In this mode iChip acts as a cable replacement and placed on both sides of the WiFi connection. The connection between the two sides is done using WiFi Ad-Hoc mode. iChip's LAN connection may be either through PHY or directly through MII/RMII. The iChips on both sides of the line may be configured differently based on design requirements.

In Cable Replacement Ad-Hoc mode, iChip supports two optional security layers. The first layer is Ad-Hoc WEP encryption and the second layer is MAC forwarding. The +iMACF parameter is an optional setting which defines a single MAC address to which all packets on the WiFi will be transmitted. Without MAC forwarding, iChip will need to broadcast outgoing packets. Broadcasting is slower than Unicasting and also has the disadvantage of being received by all systems on the Ad-Hoc network.

In this mode, all traffic from the LAN infrastructure is transmitted to the user application over the WiFi Ad-Hoc connection and all traffic sent from the user application is transmitted back to the LAN infrastructure.

The diagrams below outline the iChip “Cable Replacement Ad-Hoc Mode”:

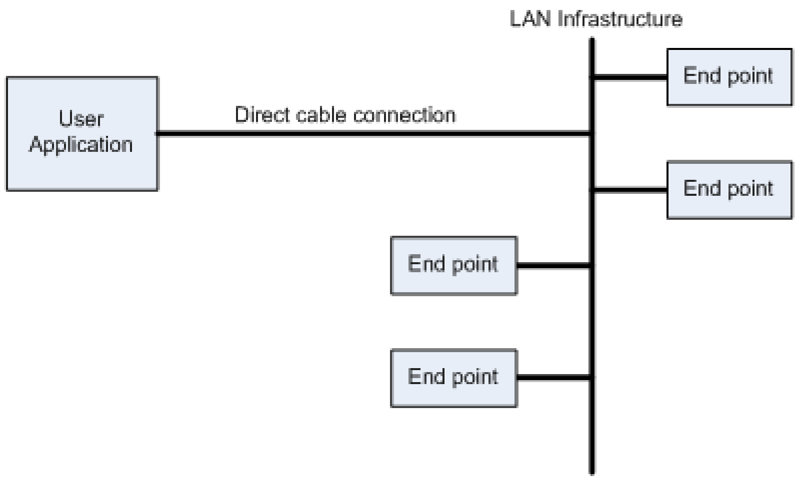


Figure 34-1: Direct Cable connection (Original state)

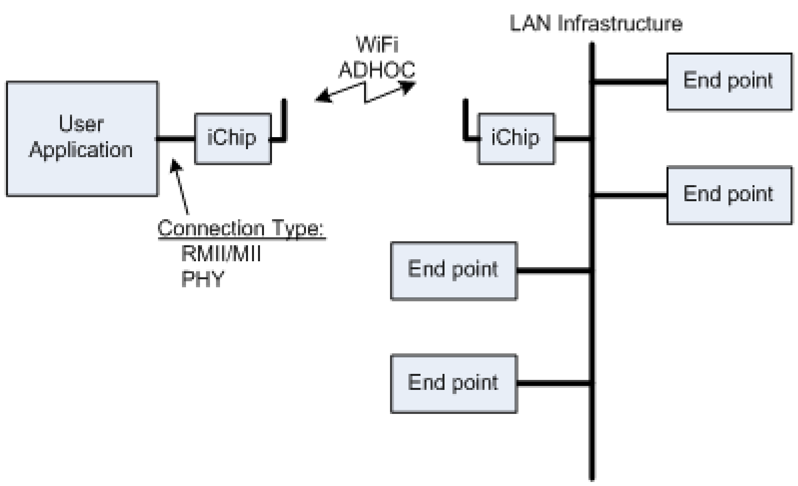


Figure 34-2: Cable Replacement ADHOC mode

To enter "Cable Replacement Ad-Hoc Mode", the following parameters should be set:

+iWLCH	-	Ad-Hoc wireless channel	
+iWLSI	-	Ad-Hoc network SSID (prefix with ‘!’)	
+iWST0	-	WEP security type (64 or 128)	- *optional*
+iWKY0	-	WEP key (10 or 26 HEX digits)	- *optional*
+iBRM	-	LAN interface: MII/RMII or PHY	- *1 or 3*
+iMACF	-	MAC Forward on both sides	- *optional*
+iHIF	-	Host Interface (non zero)	
+iBDRF	-	Fixed UART Baud Rate, if applicable	

Cable Replacement AP Mode

In this mode iChip replaces a direct cable connection of the user application to the LAN infrastructure, by connecting the user application to an Access Point on the LAN infrastructure.
The diagrams below outline the iChip "Cable Replacement AP Mode":

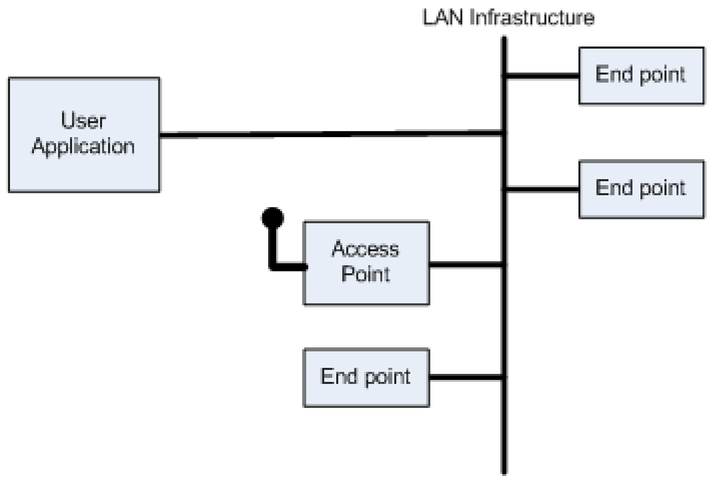


Figure 34-3: Direct cable connection to LAN infrastructure (Original state)

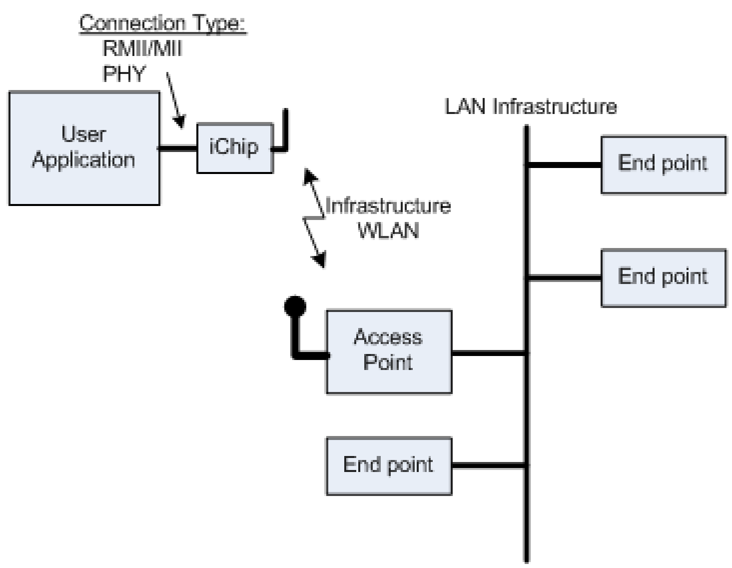


Figure 34-4: Cable replacement AP Mode

In "Cable Replacement AP Mode", iChip enables the user application to connect to an existing AP on the LAN infrastructure. Alternatively, the connection can be made using Ad-Hoc connection to a non-iChip device, such as a laptop or a smartphone. The iChip connects to the AP using the same MAC address of the user application which enables

the LAN infrastructure to maintain the same connection rules set for the user application, such as IP address. The +iMACF parameter is an optional setting which defines the user application's MAC address.
To enter "Cable Replacement AP mode", for example with WPA-PSK encryption, the following parameters should be set:

+iWLSI	-	Network SSID	
+iWST0	-	Security type (3 or 4)	- *optional*
+iWPP0	-	WPA/WPA2 security key	- *optional*
+iBRM	-	LAN interface: MII/RMII or PHY	- *2 or 4*
+iMACF	-	MAC address of the user application	- *optional*
+iHIF	-	Host Interface (non zero)	
+iBDRF	-	Fixed UART Baud Rate, if applicable	

35 Wireless LAN Mode

The iChip includes a Wireless LAN driver for the Marvell 88W8686 802.11b/g WiFi chipset. In addition, the iChip firmware contains WEP, WPA and WPA2 encryption methods for this chipset. Authentication to RADIUS on Enterprise networks is supported using EAP-TLS, EAP-MD5 and PEAP-MSCHAPv2 standards.

Multiple SSIDs (Service Set Identifiers) can be defined in an ordered list of Access Points (APs) or Ad-Hoc networks. Each SSID can have any of the supported security types. The following parameters allow definition of multiple SSIDs:

- WSIn - Defines an ordered list of allowable SSIDs.
- WPPn - Sets the Wireless LAN PSK pass-phrase for WPA and WPA2 encryptions for each individual SSID on the list .
- WKYn - Sets the Wireless LAN WEP key for each individual SSID on the list.
- WSTn - Sets the Wireless LAN security type for each individual SSID on the list.

The setting for the SSID in index n=0, which is attempted to be associated-with with the highest priority, may be assigned through the parameters WSI0, WPP0, WKY0 and WST0. They may also be assigned through an alternative set of parameters: WLSI, WLWM, WLK1, WLPP and WSEC. Changes to one set of parameters automatically change the alternative parameters as well.

WPA security requires a parameter that contains the Personal Shared Key (PSK), sometimes referred to as the *passphrase*. The Wireless LAN Passphrase (WLPP) parameter is used to set the *passphrase*. When *passphrase* contains a value, iChip uses WPA security when connecting to an Access Point (AP). Note, however, that for WPA-PSK to be active, an *SSID* (+iWLSI parameter) must also be defined. This parameter has precedence over WEP parameters. In other words, when WLPP contains a value (and WLSI is defined) WPA is used – even if WEP parameters are defined.

The type of WPA protocol to be used is determined by the value of the WSEC parameter: a ‘0’ value means the WPA-TKIP protocol will be used, whereas a ‘1’ value specifies the WPA2 protocol with TKIP or AES.

Enterprise Mode WLAN security approach focuses on a framework for providing centralized authentication and dynamic key distribution for encryption. To support enterprise security modes, iChip includes parameters for username (EUSN), password (EPSW), public certificates (CA, CERT) and a private key (PKEY).

Several commands, listed below, enable iChip to control the operation of the Marvell WiFi chipset.

+iWLTR — Wireless LAN Transmission Rate

Syntax:	AT+iWLTR=<*tr*>
	Sets the maximum allowable wireless LAN transmission rate.
	After a HW or SW reset, WLTR returns to its default value (54 Mbps).
Parameters:	*tr*=0..13
Command Options:	
tr=0	Maximum supported transmission rate (54 MBps)
tr=1	Limited to 1 Mbps
tr=2	Limited to 2 Mbps
tr=3	Limited to 5.5 Mbps
tr=4	Limited to 11 Mbps
tr=5	*Reserved*
tr=6	Limited to 6 Mbps
tr=7	Limited to 9 Mbps
tr=8	Limited to 12 Mbps
tr=9	Limited to 18 Mbps
tr=10	Limited to 24 Mbps
tr=11	Limited to 36 Mbps
tr=12	Limited to 48 Mbps
tr=13	Limited to 54 Mbps
Default:	0 (Maximum transmission rate)
Result Code:	
I/OK	If *tr*=0..13
I/ERROR	Otherwise

+iWLPW — Set WLAN Tx Power

Syntax:	AT+iWLPW=<*n*>
	Sets the transmission power of the Marvell WLAN chipset. After a HW or SW reset, WLPW returns to its default value.
Parameters:	*n*=0-14
n=0	Use Marvell's automatic power level adaptation scheme.
n=1-14	Set a fixed transmission power level.
Default:	*n*=0
Result Code:	
I/OK	If power set succeeded
I/ERROR (042)	If *n* is an illegal value
-or-	
I/ERROR (402)	If power set failed
AT+iWLPW=?	Returns the message '**0-14**' followed by **I/OK**.

+iWRFU — WLAN Radio Up

Syntax:	AT+iWRFU
	Turns on radio transmission of the Marvell WLAN chipset.
Parameters:	None
Default:	The radio automatically turns on after hardware or software reset.
Result Code:	
I/OK	If operation succeeded
I/ERROR (403)	Otherwise

+iWRFD — WLAN Radio Down

Syntax:	AT+iWRFD
	Turns off radio transmission of the Marvell WLAN chipset.
Parameters:	None
Result Code:	
I/OK	If operation succeeded
I/ERROR (403)	Otherwise

+iWRST — Reset WLAN Chipset

Syntax:	AT+iWRST
	Performs a hardware reset of the Marvell WLAN chipset.
Parameters:	None
Result Code:	
I/OK	If operation succeeded
I/ERROR (404)	Otherwise

+iWLBM — WLAN B Mode

Syntax: AT+iWLBM

Sets the Marvell WLAN chipset to 802.11/b mode. Allowable Tx transmission rates for this mode are: 1, 2, 5.5 and 11 Mbps.

After a HW or SW reset, the Marvell WLAN chipset returns to its default 802.11b/g mode.

Parameters: None

Result Code:

I/OK Always

+iWLGM — WLAN G Mode

Syntax: AT+iWLGM

Sets the Marvell WLAN chipset to 802.11/g mode. Allowable Tx transmission rates for this mode are: 6, 9, 12, 18, 24, 36, 48 and 54 Mbps.

After a HW or SW reset, the Marvell WLAN chipset returns to its default 802.11b/g mode.

Parameters: None

Result Code:

I/OK Always

Roaming Mode

When set to operate in Roaming mode, iChip can roam seamlessly among Access Points (APs) sharing the same SSID and the same security configuration without interrupting its IP connectivity. iChip also has a monitoring mechanism that is sensitive to drops in AP signal strength. When iChip detects such a drop, it automatically starts searching for APs in its vicinity that have a stronger signal, while remaining connected to the current AP.

The following parameters are required to set iChip to Roaming mode:

- WROM — Enables Roaming mode.
- WPSI — Sets the time interval between consecutive scans that iChip performs for APs in its vicinity.
- WSRL — Sets a low SNR threshold for iChip in Roaming mode.
- WSRH — Sets a high SNR threshold for iChip in Roaming mode.

In addition, two reports provide useful information pertaining to the Roaming feature:

- AT+i!RP10 — Returns a report of the current WLAN connection.
- AT+iRP20 — Returns a list of all APs and Ad-Hoc networks available in the vicinity**.**

iChip Behavior Following a Hardware or Software Reset

After power-up, hardware or software reset, iChip starts scanning for APs in its vicinity at intervals set by the WPSI parameter. iChip reads the value set in the WLSI parameter and acts accordingly:

- If WLSI refers to an AP, iChip scans for all APs in its vicinity. iChip attempts to connect to an AP whose SSID is listed first in the WSI*n* parameter. If several APs having that same SSID exist, iChip attempts to connect to the one having the strongest signal. If association succeeds, iChip stops scanning and activates its DHCP client. It then monitors the SNR level of the AP it is associated with.
- If WLSI refers to an Ad-Hoc network, iChip scans for all Ad-Hoc networks in its vicinity. iChip attempts to join an Ad-Hoc network whose SSID is listed first in the WSI*n* parameter. If no such network is found, iChip creates its own network and stops scanning.
- If WLSI is set to (*), iChip stops scanning and remains disconnected.

iChip Behavior when AP Signal Becomes Weak

When the beacon signal of the AP with which iChip is associated becomes weak (SNR drops below the level set by the WSRL parameter), iChip starts its periodic scan for APs having SNR above the threshold set by the WSRH parameter.

iChip attempts to connect to the AP that appears first on the list of SSIDs specified in the WSI*n* parameter, while remaining connected to the current AP. If association with the new AP fails, iChip continues scanning until it succeeds connecting to an AP with a stronger signal.

When in Roaming mode, iChip does not restart its DHCP client process for new connections.

When iChip is *not* in Roaming mode, iChip remains connected to an AP as long as it has an open active socket, or until triggered by a Link Lost event. When not in Roaming mode, iChip ignores any decrease in AP signal strength while having open active sockets.

When iChip is *not* in Roaming mode and *no active sockets are open*, iChip starts periodic scanning for APs having an SNR level above the WSRH threshold. iChip attempts to connect to the AP that has the highest priority. After associating with an AP, iChip starts its DHCP client and monitors the SNR level of the AP it is associated with.

iChip Behavior in the Event of a Lost Link

If the connection is *not* active, iChip starts periodic scanning for APs and attempts to connect to an AP having the highest priority. After associating to an AP, iChip starts its DHCP client and monitors the SNR level of the AP it is associated with.

If the connection *is* active, iChip waits for an IP activity command from the host. When such a command is sent, iChip performs a software reset and starts scanning for APs. iChip responds with **ERROR (074)** to indicate that the current connection has been lost.

Multiple SSIDs

The Multiple SSIDs feature allows you to define an ordered list of SSIDs of Access Points (APs) or Ad-Hoc networks with which iChip attempts to connect upon power-up. Each SSID listed can have one of the following security types:

- WEP-64
- WEP-128
- WPA-TKIP
- WPA2-AES
- No security

The following parameters allow you to define multiple SSIDs:

- WSI*n* — Defines an ordered list of allowable SSIDs.
- WPP*n* — Sets the Wireless LAN PSK passphrase for WPA and WPA2 encryption for each individual SSID on the list.
- WKY*n* — Sets the Wireless LAN WEP key for each individual SSID on the list.
- WST*n* — Sets the Wireless LAN security type for each individual SSID on the list.

WPS (WiFi Protected Setup)

WPS (WiFi Protected Setup) is a standard endorsed by the WiFi alliance for easy and secure establishment of a wireless network. Traditionally, users need to manually create a wireless network name (**SSID**), and then manually enter a creative, yet sometimes predictable, security key on both the AP and clients to prevent unwanted access to their wireless network. This process requires the users to have the background knowledge related to WiFi device configuration.

With WPS, users can automatically configure a wireless network with a network name **(SSID)** and strong WPA data encryption and authentication.

Several methodologies are available to carry out the WPS negotiation. iChip implements the PBS method which specifies that a physical or virtual push-button is depressed in both the AP and the client, allowing for a 2-minute window within which the WPS negotiation must be completed. The WPS negotiation is managed entirely automatically over the wireless medium and results in the WiFi station fully configured for communications through the AP.

The parameter, +iWPSP (WPS Pin) allows configuring a HW pin that is referenced as the WPS physical Push-Button, to activate a WPS session. Furthermore, the command, AT+iAWPS (Activate WPS) activates a WPS session from the host processor.

After activating a WPS session, iChip de-authenticates from current AP and starts scanning for WPS-enabled APs in the surroundings. iChip maintains this scan for a maximum of two minutes. If within this period time, iChip does not find a WPS-enabled AP, it will exit WPS mode and reconnect to the last AP.
According to WPS specifications, the Scan results should contain only **ONE** valid AP. Otherwise, iChip must exit WPS mode and reconnect to the last AP.
When iChip finds a single valid AP, it starts the WPS session negotiation. The WPS session will run on the wireless medium as a series of EAP request/response messages, which is successfully concluded with iChip updating its parameters in flash with the new AP credentials. iChip will then connect to the AP using the new credentials.

IMPORTANT NOTE: If the WPS AP credentials contain a WPA pass-phrase, iChip shall automatically calculate the PSK according to the new SSID and pass-phrase. Hence, iChip will not respond to AT+i commands for about 25-30 seconds (as is the case when changing +iWLSI or +iWLPP parameters).

+iAWPS — Activate WPS from Host

Syntax:	AT+iAWPS
	Activates a WPS configuration session regardless of the +iWPSP setting. iChip's communication platform must be WiFi for this command to perform correctly.
Parameters:	None
Result Code:	
I/OK	If iChip's Communication platform is WiFi (+iCPF is 0 or 1).
I/ERROR	Otherwise

iChip Power Save Mode

iChip has a Power Save mode for achieving energy savings. You enable Power Save mode by setting the PSE parameter to any value *n* between 1 and 255 seconds. When *n* seconds have elapsed without any activity on the host or modem serial ports, iChip shuts down most of its circuits. Renewed activity on the serial ports, or incoming data from the LAN, restores iChip to full operational mode.

If, in addition, the WLPS parameter is set to any value *m* between 1 and 5, iChip can force the Marvell WiFi chipset into either Power Save or Deep Sleep mode:

- If iChip is currently associated with an AP, or is configured to operate in Ad-Hoc mode, iChip will force the Marvell chipset into Power Save mode. In Power Save mode, the Marvell chipset will go to sleep for *m* beacon periods when no communication has taken place (command, Tx, or Rx activity) for one full beacon period.
- If iChip is *not* associated with an AP, iChip will force the Marvell chipset into Deep Sleep mode. iChip will perform a periodic scan every *p* seconds, as set by the WPSI parameter, for APs in its vicinity. If it fails to locate and associate with an AP, it will wait for *n* seconds, as set by the PSE parameter, before forcing the Marvell chipset back to Deep Sleep.
- If WLSI=* or WSI0=* then iChip will not associate with any AP. If, in addition, Power Save mode is enabled, iChip will not wakeup the Marvell chipset to perform periodic scans.

Auto Connection to IP-Enabled AP

iChip contains an elaborate scheme according to which it selects the most appropriate AP to connect with, given an SSID setting and relative transmission strengths. An additional criterion can be applied to decide on an appropriate AP. The criterion is that the AP is Internet enabled. In other words, an AP that can be used to access the public internet.

The modified selection procedure contains two phases. The first phase includes locating an AP in the normal manner. The second phase includes attempting a PING to a well-defined system on the Internet. If the PING command succeeds, iChip shall remain connected to that AP, otherwise it will continue searching for another AP. Two existing parameters are utilized to contain the PING destination addresses to use. These parameters are +iPDSn (n=1..2). The +iPDSn parameters are also used for a similar purpose, when verifying that the modem is ONLINE.

The parameter +iWIAP (Wireless Internet-Enabled AP) enables this type of AP search. Setting the SSID to the special value $$$$, in one of the parameters WLSI or WSIn, also enables this mode.

The default value is 0 (disabled). When assigned with a value in the range 1-255, the Internet-Enabled AP seek mode shall be enabled and the +iWIAP value shall indicate the maximum time in seconds to wait for an IP assignment from a DHCP server. When in Internet-enabled AP seek mode, while seeking for an Internet-Enabled AP, the AT+i!RP10 shall report:

"Scanning for IP-Enabled AP"

Since the +iPDSn parameters are used to verify that the AP tested is Internet-Enabled, if both are empty the test cannot be performed. Therefore, if +iWIAP>0 and both +iPDSn parameters are empty, iChip will not connect to any AP.

Changes to the +iWIAP parameter require a SW reset to take affect.

36 Ad-Hoc Networks

An Ad-Hoc network is a Wireless Local Area Network (WLAN) in which some of the stations are part of the network only for the duration of a communications session or, in the case of mobile or portable devices, while in some close proximity to the rest of the network.

Ad-hoc networks do not require an Access Point (AP) to enable communication among stations. Each station can create a new Ad-Hoc network or join an existing one. Networks can freely merge into a single network or split into smaller ones, thus adapting to changing conditions such as topology, signal strength, and proximity to nearby Ad-Hoc networks. Combined with an iChip configured as an iRouter, an Ad-Hoc network can connect to the Internet through a dial-up or GPRS modem.

Configuration

Configuring the iChip to operate as a station in an Ad-Hoc network requires setting the following parameters:

- WLSI must be set to either '!' or '!<*SSID*>'. When it is set to '!', iChip continuously searches for existing Ad-Hoc networks in its vicinity and joins the one having the strongest signal. When it is set to '!<*SSID*>', iChip searches for an Ad-Hoc network having the specified Service Set IDentifier (SSID). If it finds one it joins it, otherwise it creates a new network with this SSID.
- WLCH must be set to a default value. This value indicates the communication channel (1-13) to be used for beacon transmission in the Ad-Hoc network. When iChip joins an already existing network, it adopts the channel used by that network. If WLSI=!<*SSID*> and WLCH=0, iChip will only join an already existing network.

iChip Behavior in Ad-Hoc Mode

Automatic Scanning for Existing Ad-Hoc Networks

If the WLSI parameter contains an SSID string preceded by (!) or set to (!), iChip scans for Ad-Hoc networks only. If the WLSI parameter is set to an empty string or to an SSID not preceded by (!), iChip will not attempt to connect to an Ad-Hoc network. Finally, if the WLSI parameter is set to (*), iChip will not attempt to connect to any type of network.

The WLAS parameter defines the number of times that iChip scans for existing Ad-Hoc networks with the defined SSID and channel, before it establishes such a network.

Creating a New Ad-Hoc Network

If iChip does not detect any Ad-Hoc networks in its vicinity, and the WLSI parameter contains an SSID, iChip creates a new Ad-Hoc network with its own BSSID.

Joining an Existing Ad-Hoc Network

If iChip detects Ad-Hoc networks in its vicinity and the WLSI parameter is set to (!), iChip joins the network having the strongest signal. Otherwise, iChip joins the network whose SSID is set by the WLSI parameter.

Merging Ad-Hoc Networks

When iChip is configured to operate in Ad-Hoc mode it performs a periodic scan for other Ad-Hoc networks in the vicinity having the same SSID but a different BSSID. If a scan indicates the existence of such an Ad-Hoc network, it initiates a procedure for merging the networks. Networks will merge into one, provided they operate on the same channel.
There is an option to prohibit iChip from merging with other Ad-Hoc networks. To do this the +iWLCH (Wireless LAN Channel number) must contain a +100 offest. For example, instead of setting to channel 6, set +iWLCH to channel 106.
With this setting iChip shall setup an Ad-Hoc network (Assuming +iWLSI is preceded with an '!') on channel 6 and network merging is disabled.

37 LAN Commands

Introduction

The commands +iETHD and +iETHU facilitate saving power in an Ethernet LAN environment. These commands turn the Ethernet PHY OFF or ON. When turning the PHY OFF the existing communication status is not saved, therefore, the system performs a soft-reset when the PHY is turned ON again. When the Ethernet PHY is down, 80-90 mAmp are conserved.
Additional power saving can be enabled on the iChip using the PSE parameter.

+iETHD – Ethernet PHY Shut Down

Syntax:	AT+iETHD
	Turns OFF the Ethernet PHY.
Parameters:	None
Command Options:	None
Result code:	
I/OK	If the Ethernet PHY was turned OFF.
I/ERROR	Otherwise

+iETHU – Etherent PHY Re-Start

Syntax:	AT+iETHU
	Turns ON the Ethernet PHY.
Parameters:	None
Command Options:	None
Result code:	
I/OK	If the Ethernet PHY was turned ON.
I/ERROR	Otherwise

38 Flow Control

Host ➔ iChip Software Flow Control

When issuing an AT+iEMB command to generate a binary e-mail, an AT+iSSND command to transfer data to a socket, an AT+iTBSN to send a binary stream to a Telnet server, or an AT+iFSND command to transfer a file, the host transfers a binary data stream to iChip. At times, this stream may be very large.

Once iChip establishes a connection, it acts as a pipeline, transferring data received from the host to the Internet. However, the data rates at the host and Internet ends are not always balanced. This happens for several reasons:

- While iChip logs onto the Internet and establishes a connection, the host proceeds to send its data stream to iChip. During this time iChip receives data from the host, but cannot send it out.
- When sending MIME attachments, iChip encodes the binary data using base 64. This roughly inflates binary data by 30%. Thus, more data needs to be transmitted than is received from the host.
- When using a TCP/IP socket, iChip might need to re-transmit packets.

The amount of buffer space available in the iChip to accommodate for this imbalance is limited. Therefore, a flow control scheme is required to regulate host⇔iChip communications. The FLW parameter is set to reflect the preferred flow control mode.

The software-driven flow control protocol is defined as follows:

1. While the host is transferring the binary stream, following the +iEMB, +iSSND, or +FSND prefixes, iChip issues a 'WAIT' control character when it needs to pause the host. The host application is required to monitor its serial receive line and pause the transmission when a 'WAIT' control character is received.
2. To resume the host transmission, iChip issues a 'CONTINUE' control character. The host is required to monitor its receive line after being paused in anticipation of this control character. Once received, the host might continue to transfer the data stream.
3. If an error occurs during the Internet session while the host is transferring the data stream (or while paused), iChip issues an 'ERROR' control character if some error occurred. Immediately after issuing this control character, iChip aborts the Internet session and issues an 'I/ERROR (error number)' string. The host must cease transmitting the data stream when the 'ERROR' control character is received.

The control characters are defined as:

Control	ASCII Dec	ASCII Hex	Mnemonic
WAIT	22	0x16	SYN
CONTINUE	24	0x18	CAN
ERROR	5	0x5	ENQ

Table 38.1 Software Flow Control Characters

Software Flow Control Diagram in Binary E-Mail Send

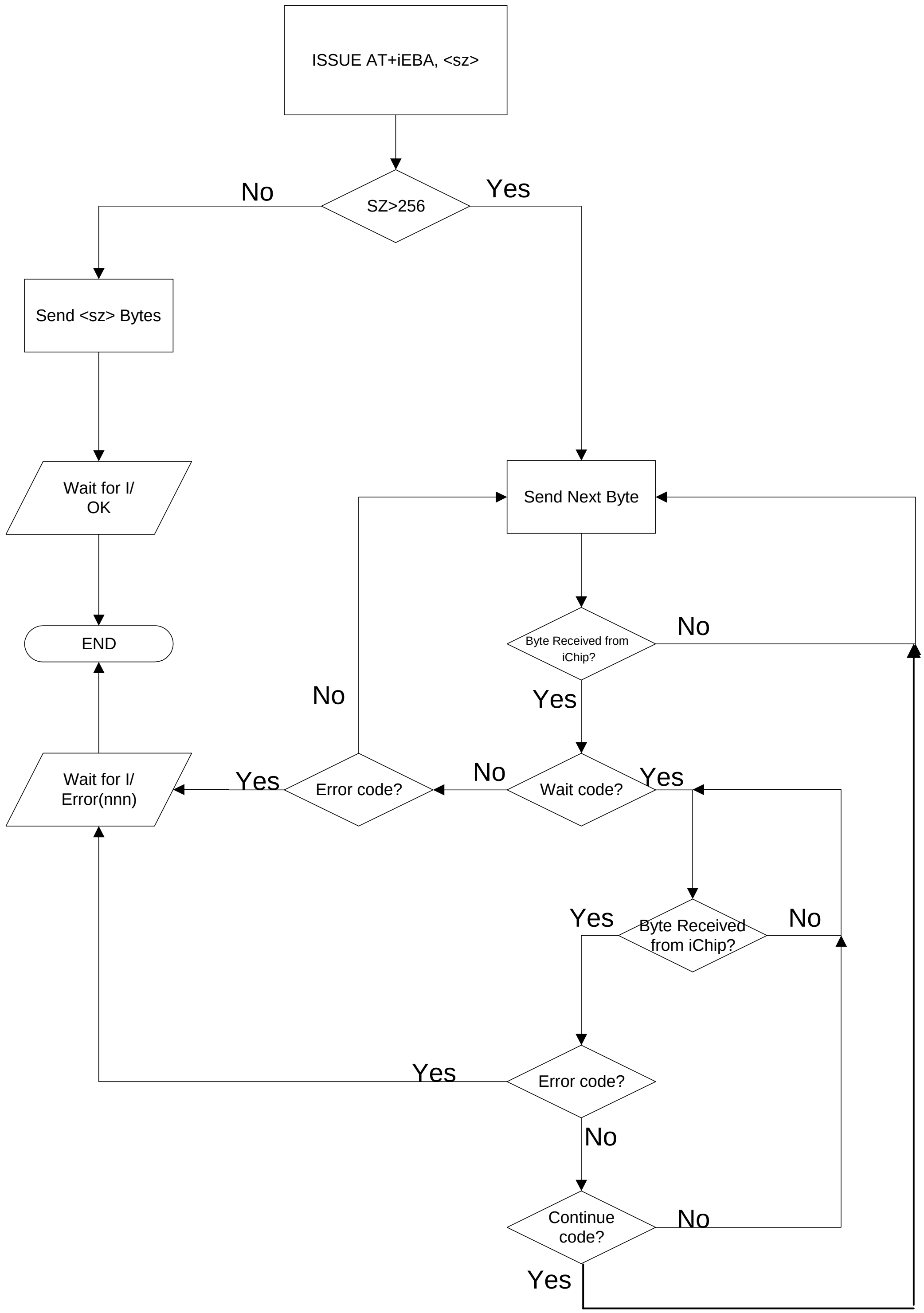


Figure 38-1 Software Flow Control in Binary E-Mail Send

Software Flow Control During A Socket Send

When a WAIT control is sent to the host during a socket send (AT+iSSND) command, it is automatically followed by an RP4 socket status report in the following syntax:

`I/(<sock0sz>, <sock1sz>, … ,<sock9sz>)<CR/LF>`

See the AT+iRP command for a full description.

While the host is waiting for the CONTINUE control, it may analyze the sockets' input buffer status. If the host detects a need to execute a socket receive command to empty one or more socket input buffers, it may escape the current SSND command by issuing a '*Pause*' sequence immediately after receiving the 'CONTINUE' control.

The '*Pause*' sequence is defined as: half a second of silence followed by (---) (three consecutive minus sign characters). iChip responds by prematurely terminating the SSND command, including flushing the current socket if the (%) flag is specified. Following this, the I/OK message is issued and the host may issue the required SRCV command in addition to any other operations it needs to execute. The host may return to the pre-empted socket at any time and issue a new SSND command to send out the balance of data.

Software Flow Control Diagram in Socket Send

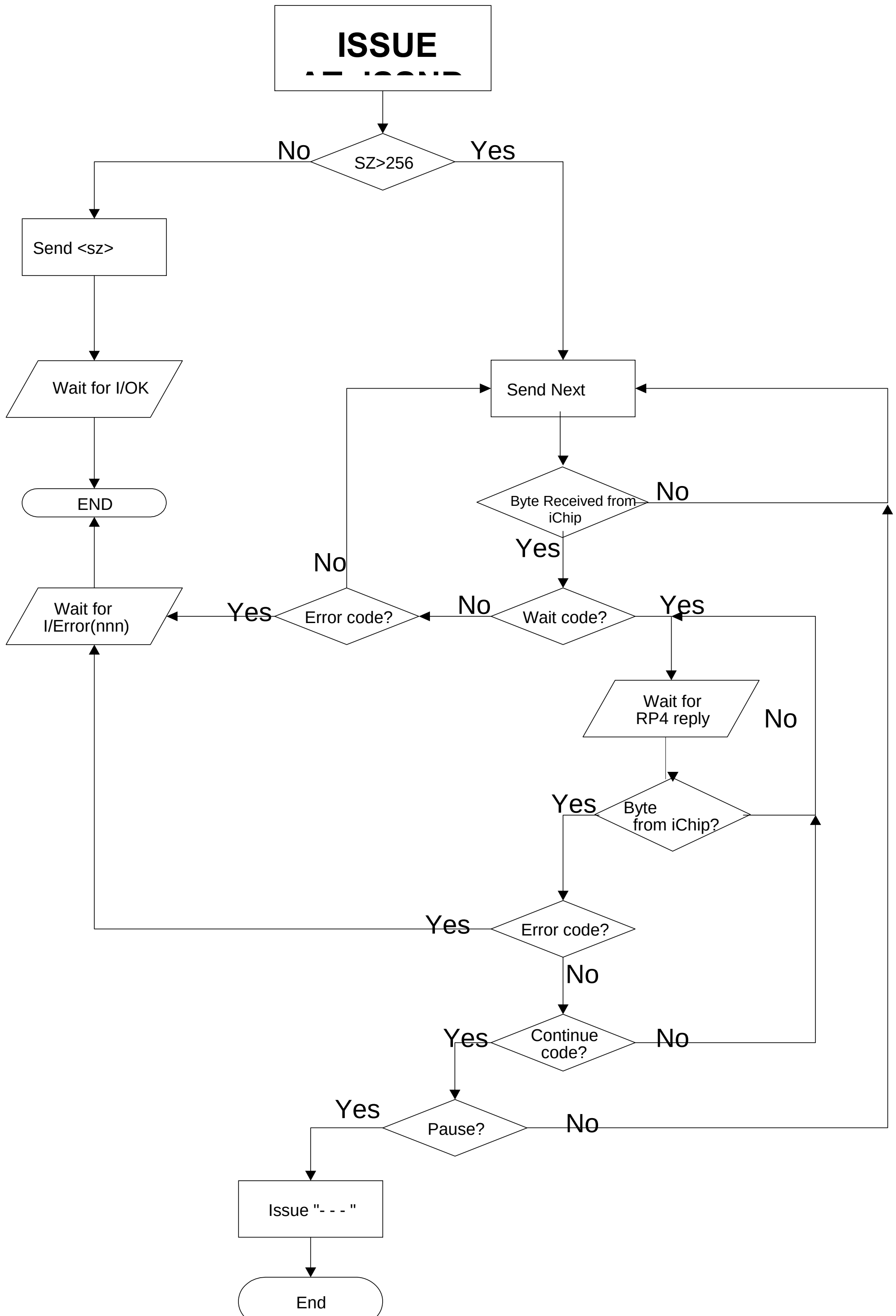


Figure 38-2 Software Flow Control in Socket Send

Host ➔ iChip Hardware Flow Control

As an alternative to the software flow control method, which requires some software attention on behalf of the host, iChip offers a hardware flow control mode.

This mode is selected by setting iChip's FLW parameter Bit 0, using the AT+iFLW command. Note that to set FLW Bit 0, the ~CTSH signal needs to be LOW (enabled), otherwise iChip returns I/ERROR (063). This convention safeguards iChip from lockup, which may arise if FLW Bit 0 is set while the ~CTSH signal is constantly HIGH.

For hardware flow control to operate properly, the ~CTS and ~RTS signals between the host and iChip UARTS must be interconnected.

The iChip ~CTSH and ~RTSH signals can be shorted to circumvent hardware flow control.

Under this mode, iChip assumes that the host transmission might be paused by de-asserting the ~CTS signal. The host must adhere to this convention. Most UARTs support hardware flow control. However, if this is not the case, iChip's ~CTS signal must be monitored by the host software on a general purpose I/O.

The host can also pause iChip by de-asserting its ~CTS signal.

If a transmission error occurs during processing of a send command (EMB, SSND, TBSN, FSND), iChip accepts all remaining characters pertaining to the current command (as specified by the <*sz*> parameter) before returning the relevant I/ERROR response.

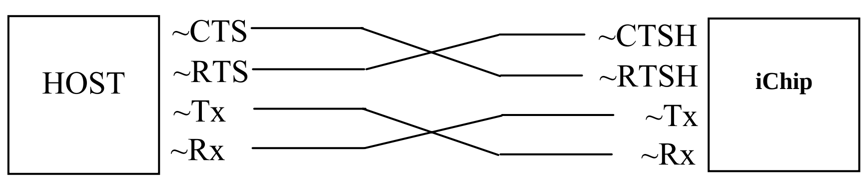


Figure 38-3 Minimum Hardware Flow Control Connections

39 Nonvolatile Parameter Database

Parameter Descriptions

Parameter	Type	Range	Default	Description
Operational				
XRC	Byte	0..4	4	Extended Return Code. Same as ATXn
DMD	Byte	0..2	0	Modem Dial Mode: ATD<*m*> *m*: Tone (0); Pulse (1); None (2)
MIS	String	126 chars	"AT&FE0V1X4Q0&D2M1L3\r"	Modem initialization string. May contain several consecutive AT commands.
MTYP	Byte	0..11	0	Modem Type Designator
WTC	Byte	0..255	45	Wait Time Constant. Initialization constant for modem's S7 register. Defines a timeout constant for a variety of modem activities.
TTO	INT	0..3600	0	TCP Timeout. Number of seconds to wait before returning a timeout error on a TCP transaction.
PGT	Unsigned INT	0, 50-65535	0	Timeout in [mSec] to resend a PING request. Default timeout is 2 sec.
MPS	Byte	0..3	0 (1500)	Max PPP Packet Size.
TTR	INT	1000.. 65535	3000 [mSec]	Timeout to resend an unacknowledged TCP packet over PPP, in milliseconds.
MSS	Byte	0..3	0	Set Maximum TCP Segment Size: 536/1460 (Bit 0) and also use of Nagle's Algorithm (Bit 1).
BDRF	Byte	3..9|'a'|'h'	'a' (Auto)	Sets the iChip⇔Host to a fixed baud rate.
BDRM	Byte	3..9|'a'|'h'	'a' (Auto)	Sets the iChip⇔modem baud rate to a fixed baud.
BDRD	Byte	0..255	0 (Use BDRF)	Sets High speed baud rate, division of 3 Mbps
AWS	Byte	0..3	0	Sets flag to define web server activation. 0 (web server disabled), 1 | 2 | 3(web server enabled).
WEBP	INT	0-65535	0	Port number of the embedded Web server.
LATI	INT	0-65535	0 (Disabled)	Remote AT+i Service, port number.
LAR	Byte	0..7	0 (Disabled)	Restrict access to the LATI port.
FLW	Byte	0..7	0 (S/W)	Flow Control Mode
CPF	Byte	0..1	1 (LAN)	Sets Communication Platform: Modem (0); LAN (1).
LTYP	Byte	0..3	0 (Auto)	Sets the active LAN interface to be WiFi or LAN or automatic scanned.

Parameter	Type	Range	Default	Description
Operational				
PSE	Byte	0..255	0 (Disabled)	Sets Power Save Mode: Disabled(0); idle time in seconds before activating Power Save mode (1..255)
SDM	Byte	0..127	0	Service Disable Bitmap
DF	Byte	0..1	0	IP Protocol Don't Fragment Bit
CKSM	Byte	0..1	0 (Disabled)	Sets checksum mode
HIF	Byte	0..6,101, 102	0 (Auto detect)	Sets host-to-iChip interface
MIF	Byte	1..5	2 (USART1)	Sets iChip-to-modem interface
ADCL	Byte	0-255	0	A/D Converter base level
ADCD	Byte	0-255	0	A/D Converter delta
ADCT	INT	0-65535	0	Time interval between queries of the A/D Converter's register
ADCP	INT	0-6	0	iChip's PIOC pin to be asserted by the A/D Converter's polling mechanism
RRA	Byte	0-6	0	iChip readiness indication
RRHW	INT	0-96	0	iChip readiness HW pin
SPIP	Byte	0..6	0	Pin number of the SPI Control Signal
ISP Connection				
ISPn	Phone #	96 chars	NULL	ISP's access phone number. <*n*>: 1..2
ATH	Byte	0..2	1 (PAP)	Use CHAP (2), PAP (1) or Script (0) authentication
USRN	String	64 chars	NULL	ISP Connection User Name
PWD	String	64 chars	NULL	ISP Connection Password
RDL	Byte	0..20	5	Number of Redial tries
RTO	Byte	0..3600	180	Timeout before redialing [seconds]
Server Profiles				
LVS	Byte	0..1	1 (YES)	Leave on Server: 1(YES), 0 (NO)
DNSn*[p]*	IP address		0.0.0.0	Domain Name Server IP address <*n*>:1..2
SMTP*[p]*	String	64 chars	NULL	SMTP Server Name
SP	INT	0-65535	25	SMTP Server port number
SMA	Byte	0..1	0 (None)	Define SMTP Authenticated Method: 0 (None) 1(Login authentication)
SMU	String	64 chars	NULL	SMTP Authentication User Name
SMP	String	64 chars	NULL	SMTP Authentication Password
POP3*[p]*	String	64 chars	NULL	POP3 Server Name
MBX	String	64 chars	NULL	Mailbox User Name
MPWD	String	64 chars	NULL	Mailbox Password
NTS*n*	String	64 chars	NULL	Network Time Server name <*n*>: 1..2
NTOD	Byte	0..1	0 (Disabled)	Network time-of-day retrieval flag
GMTO	Byte	-12..12	0	iChip location's GMT Offset
DSTD	String	64 chars	NULL	Sets iChip's Daylight Savings transition rule

Parameter	Type	Range	Default	Description
Server Profiles				
PDS*n*	String	64 chars	NULL	Sets iChip's PING Destination servers, used for online status verification.
PFR	INT	0-65535	0 (Disabled)	Sets PING destination server polling frequency.
PRXY	String	64 chars	NULL	Sets the IP address and port of the Proxy for all socket communications.
User Fields				
UF*n*	String	256 chars	NULL	User Storage field and Macro Substitution <*n*>: 01..12
E-Mail Format				
XFH	Byte	0..1	1	Transfers e-mail headers. 1 (Enable) 0 (Disable)
HDL	Byte	0..255	0 (no limit)	Limits number of header lines retrieved.
FLS	String	64 chars	NULL (no filter)	Filter string must exist in message header to Qualify for Retrieve.
DELF	String	64 chars	None	E-mail Delete Filter
SBJ	String	96 chars	NULL	Contents of the e-mail subject field
TOA*[n]*	String	64 chars	NULL	E-mail Addressee
TO	String	96 chars	NULL	Addressee Description/Name in e-mail header
REA	String	64 chars	NULL	Returns e-mail address.
FRM	String	96 chars	NULL	Sender Description/Name in e-mail header
CCn	String	64 chars	NULL	Alternate Addressee (CC: field) <*n*>: 1..4
BDY	Text lines	96 chars	NULL	Textual body contents for MIME-encapsulated e-mail messages
MT	Byte	0..4	4 (app.)	Media Type: 0: Text; 1: Image ; 2: Audio ; 3: Video ; 4: application
MST	String	64 chars	octet-stream	Media Subtype String. For a list see Appendix A.
FN	String	64 chars	None	Attachment File Name (inc. extension). If a file name is not defined, iChip generates a unique filename.
CSTY	String	64 chars	None	Character set for encoding e-mails
CTE	String	64 chars	None	Context encoding type for e-mails
IP Registration				
RRMA	String	64 chars	NULL	Sets the e-mail address to use for dynamic IP address registration after going online.
RRSV	String	64 chars	NULL	Sets the server name/IP and port to contact for dynamic IP address registration after going online.
RRWS	String	128 chars	NULL	Sets the web server URL used for dynamic registration after going online.

Parameter	Type	Range	Default	Description
IP Registration				
RRRL	String	64 chars	NULL	Sets the Return Link IP address to use when performing an IP address registration behind a NAT.
HSTN	String	64 chars	NULL	iChip's Network Host Name, included in all IP registration methods. iChip LAN will be registered in DNS through DHCP Server.
HTTP				
URL	String	256 chars	None	URL string used for subsequent +iRLNK and +iSLNK commands.
CTT	String	64 chars	NULL	Defines the "Content-type" field sent in the POST request by the +iSLNK command.
WPWD	String	64 chars	NULL	Password for restricting host parameter updates via a web browser.
LOGO	String	128 chars	"iChipimages/ConnectOne.gif"	Path to a file that contains the LOGO picture that will be shown in the top frame of the configuration website that is embedded in iChip.
LDLY	Byte	0..5	5	Limit the delay for download from HTTP and FTP servers
RAS Server				
RAR	Byte	2..20	4	Number of RINGs after which iChip will activate its internal RAS Server.
RAU	String	64 chars	NULL	RAS Login User Name
RAP	String	64 chars	NULL	RAS Login Password
Unique Identifiers				
SNUM	Hex String	8 chars	FFFFFFFF	Factory-assigned serial number. Cannot be modified. Can be read using +iRP5
UID	String	8 chars	" Empty	Unique ID for use by the application. Cannot be modified once assigned.
LAN				
MACA	String	12 chars	MAC address assigned by Connect One	MAC address assigned to iChip
DIP	Default IP address		0.0.0.0	Default IP address stored in iChip's nonvolatile memory.
IPA	IP address		0.0.0.0	IP address assigned to iChip
IPG	IP address		0.0.0.0	IP gateway address assigned to iChip
SNET	IP address		0.0.0.0	Subnet address assigned to iChip

Parameter	Type	Range	Default	Description
802.11b/g Wireless LAN				
WLCH	Byte	0..13, 101-113	0	Wireless LAN Communication Channel in Ad-Hoc mode
WLSI	String	32 chars	NULL	Wireless LAN System Set ID
WLWM	Byte	0..2	0 (Disabled)	Wireless LAN WEP Mode
WLKI	Byte	1..4	1	Wireless LAN Transmission WEP Key Index
WLKn	Hex Key String	26 chars	NULL	Wireless LAN WEP Key Array
WLPS	Byte	0..5	0	Marvell WiFi chipset Power Save mode dose time.
WLPP	String	8-63 chars	NULL	Wireless LAN WPA- PSK pass phrase
WSEC	Byte	0..1	0 (WPA security)	Wireless LAN WPA security option
WROM	Byte	0..1	0	Enable Roaming mode
WPSI	INT	1-3600	5	Periodic scan for APs interval
WSRL	Byte	0-254	10	Roaming mode SNR low threshold
WSRH	Byte	10-255	30	Roaming mode SNR high threshold
WSI*n*	String	32 chars	“ (Empty)	WLAN SSID for multiple SSIDs
WPP*n*	String	8-63 chars	“ (Empty)	Pre-shared key passphrase for multiple SSIDs
WKY*n*	String	26 chars	“ (Empty)	WLAN WEP key for multiple SSIDs
WST*n*	Byte	0..8,105, 106	0	WLAN security type for multiple SSIDs
EUSN	String	64 chars	“ (Empty)	WLAN Enterprise Domain\Username
EPSW	String	64 chars	“ (Empty)	WLAN Enterprise Password
BSID	String	12 chars	000000000000	WLAN BSSID of an AP to connect to
WLAS	Byte	0-255	3	WLAN number of scans before creating an Ad-Hoc network
WPSP	INT	0..6	0 (None)	Define a GPIO as the push-button for activating WPS
WIAP	Byte	0-255	0	Sets iChip to Internet-Enabled AP seek mode.
LAN-to-WiFi Bridge Mode				
BRM	Byte	0..4	0 (disabled)	LAN to WiFi Bridge Mode.
MACF	String	12 chars	“ (Empty)	MAC address forwarding in LAN-to-WiFi Bridge Mode
SerialNET Mode				
HSRV or HSR*n*	String	64 chars	NULL	Set the remote host server name/IP and port.
HSS	String	3 chars	NULL	Switches among three possible HSRV parameters.
DSTR	String	8 chars	NULL	Set the disconnection string template.
LPRT	Unsigned INT	0-65535	0	Set the SerialNET mode listen socket.

Parameter	Type	Range	Default	Description
SerialNET Mode				
MBTB	INT	0-2048	0	Max bytes to buffer while iChip is establishing a connection.
MTTF	Unsigned INT	0-65535	0 (None)	Max inactivity timeout in milliseconds before flushing the SerialNET socket.
FCHR	Byte	1 char	0 (None)	Flush character. When received, SerialNET socket will be flushed.
MCBF	INT	0-1460	0 (None)	Max. characters before flushing the SerialNET socket.
IATO	INT	0-65535	0 (None)	Inactivity timeout in seconds before closing the SerialNET connection.
SNSI	String	up to 13 chars	"5,8,N,1,0"	SerialNET mode Serial interface configuration. Defines baud, bits, parity, stop-bits and flow control.
STYP	Byte	0..2	0 (TCP)	Set the SerialNET mode socket type. 0 (TCP) or 1 (UDP) or 2 (SSL).
SNRD	INT	0..3600	0 (No Delay)	Delay time in seconds before re-enabling SerialNET mode after a failed connection.
SPN	String	96 chars	NULL	SerialNET Phone Number to wake-up SerialNET Server.
SDT	Byte	0..255	20	SerialNET Dial Timeout. When waking up a SerialNET server, iChip will hangup after SDT seconds have elapsed.
SWT	INT	0..65535	600	SerialNET Wake-up Timeout. Number of seconds to allow for the SerialNET server wake-up procedure.
PTD	INT	0..65535	0 (No Filter)	Specifies the number of Packets to Drop during a SerialNET session.
SLED	INT	0..6	0 (None)	Define a GPIO as the SerialNET signal
Remote Firmware Update				
UEN	Byte	0..1	0	Remote Firmware Update flag
USRV	String	128 chars	" (Empty)	HTTP or FTP server which contains firmware files
UUSR	String	64 chars	" (Empty)	Login user name for FTP
UPWD	String	64 chars	" (Empty)	Login password for FTP
RPG	String	64 chars	NULL	Remote Parameter Update Group/Password
Secure Socket Protocol (SSL3/TLS1)				
CS	INT	0, 4, 5, 10, 47, 53 (and +1000)	0 (propose all)	Set the cipher suite to be used during SSL3/TLS negotiations.
CAn	String	1500 characters	NULL	Set iChip's array of SSL3/TLS trusted Certificate Authority (CA). n=1..4
CERT	String	1500 characters	NULL	Set iChip's SSL3/TLS certificate.
PKEY	String	1500 chrs	NULL	Set iChip's private key.

Parameter	Type	Range	Default	Description
DHCP Server				
DPSZ	Byte	0-255	0 (DHCP server off)	Set number of addresses in iChip's IP pool.
DSLT	INT	0-65535	0 (No limit)	Define lease time, in minutes, granted when assigning IP addresses to clients.
iRouter Mode				
ARS	Byte	0..1	0	Causes iChip to automatically enter iRouter mode upon power-up or soft reset.
PFWn	String		NULL	Port Forwarding Rules Array

Table 34-39.1 Nonvolatile Parameter Database

+iFD — Restore All Parameters to Factory Defaults

Syntax: AT+iFD

Restore iChip's non-volatile parameter database values to factory defaults.

Each of iChip's nonvolatile parameters, described in the following section, has an associated default value. This command restores all parameters to their factory default values.

This command disables iChip's DHCP client. In order to re-activate the DHCP client process, you need to perform a HW or SW reset.

This command also resets iChip's active IP address stored in the IPA parameter.

An exception to the above are the MIS (Modem Init String), MTYP (Modem Type), RPG (Remote Parameter Group/Password) , CPF (Communications Platform) and LOGO (on the website) parameters, which will always retain the last set value.

Another exception includes parameters which affect the utilization of physical pins, and would normally reflect the actual connections of these signals. The AT+iFD command will not alter the last settings of:
RRHW (iChip Readiness signal), SPIP (SPI Control signal), SLED (SerialNET indicator pin), WPSP (WiFi WPS button) and ADCP (A/D polling indicator signal).

Result Code:

I/OK After restoring parameters to factory default values.

Operational Parameters

+iXRC — Extended Result Code

Syntax:	AT+iXRC=*n*
	Extended Result Code. Same as ATXn. This command selects which subset of the result messages will be used by the modem to inform the Host of the results of commands.
Parameters:	*n*=0..4
Command Options:	For a detailed description of the command options see the ATX*n* command in the AT command set manual for the modem in use.
Default:	4
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iXRC~*n*	Temporarily sets the Extended Result Code for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iXRC?	Report the current Extended Result Code used when dialing the ISP. The reply is followed by **I/OK**.
AT+iXRC=?	Returns the message “0-4” followed by **I/OK**.

+iDMD — Modem Dial Mode

Syntax:	AT+iDMD=*n*
	Permanently sets the modem dial mode to Tone, Pulse or none. This parameter defines the dial character ***m*** used when issuing the ATD***m*** dial command to the modem.
Parameters:	*n*=0..2
Command Options:	
n=0	Use Tone dialing (*m*=T)
n=1	Use Pulse dialing (*m*=P)
n=2	Use modem's default dialing (*m*='')
Default:	0 (Tone Dialing)
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iDMD~*n*	Temporarily sets the modem dial mode for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iDMD?	Reports the current modem dial mode used when dialing the ISP. The reply is followed by **I/OK**.
AT+iDMD=?	Returns the message "0-2". The reply is followed by **I/OK**.

+iMIS — Modem Initialization String

Syntax:	AT+iMIS=*str[;str…]*
	Sets the Modem Initialization String.
Parameters:	*str*=Modem initialization string
Command Options:	
str=”	Empty: No modem initialization string defined.
str=<string>	*string* will be used as the modem initialization string. If *string* contains special characters, such as quotation marks (‘ or “), these may be included in *string* by prefixing each special character with a backslash (‘\’). For example: “AT+CGDCONT=1,\”IP\”,\”INTERNET\””. *string* must include the AT prefix and the modem reply is expected to include ‘OK’. MIS may include several consecutive modem commands separated by a semicolon. Each command must begin with ‘AT’ and its modem reply must include ‘OK’. iChip will send each ‘AT’ command separately, followed by <CR> and wait for the ‘OK’ before proceeding.
Default:	‘AT&FE0V1X4Q0&D2M1L3’
	***Note**: This default value is shipped from the factory. The AT+iFD command does not restore MIS to this value.*
Result Code:	
I/OK	If *str* is an empty or a legal string
I/ERROR	Otherwise
AT+iMIS~ *str[;str…]*	Temporarily sets the modem initialization string to *str[;str…]*. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iMIS?	Reports the current modem initialization string. If the modem initialization string is empty, only <CRLF> will be returned. The reply is followed by **I/OK**.
AT+iMIS=?	Returns the message ‘String’ followed by **I/OK**.

+iMTYP — Set Type of Modem Connected to iChip

Syntax:	AT+iMTYP=*n*
	Sets the modem type. This parameter takes effect only after power-up or AT+iDOWN.
Parameters:	*n*=0..12, 100..112
Command Options:	
n=0	Standard, Hayes compatible, dialup modem
n=1	Silicon Laboratories Si2400 modem. See note below
n=2	Standard GSM modem
n=3	AMPS CM900 modem
n=4	Falcom GSM modem
n=5	Silicon Laboratories high-speed modems Si2414/33/56
n=6	Standard 2400 baud modem (increased timeout)
n=7	GSM 536 modem (packets limited to 536 bytes)
n=8	CDPD cellular modem
n=9	Wavecom Fastrack cellular modem
n=10	SiLABs World modem
n=11	Telit GE862-PY cellular modem
n=12	Reserved
n=13	TC65i GSM cellular modem which requires a prolonged startup delay.
+100	Add 100 to any modem type to prohibit iChip from issuing an ATZ to the modem before dialing the ISP when an Internet session is activated. This is useful if the modem needs to be initialized manually before an Internet session. Note that an ATZ will be issued when the session is terminated.
Default:	*n*=0 Standard, Hayes compatible, dialup modem
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iMTYP?	Returns current modem type designator followed by **I/OK**.

AT+iMTYP=? Returns the message "0-12,100-112" followed by **I/OK**.

Note 1	Configuring the iChip to work with Silicon Laboratories Si2400:
1.	AT+iMTYP=1
2.	AT+iMIS="
3.	AT+iBDRF=3
4.	AT+iBDRM=3
Note 2	Configuring the iChip to work with GPRS modems:
1.	AT+iMTYP=2 – GSM/GPRS modem type
2.	AT+iXRC=0 – blind dialing
3.	AT+iISP1=<*ISP/Provider dial number*> (usually ***99****1#)
4.	AT+iMIS="AT+CGDCONT=1,IP,<*Proxy*>"
Note 3	Changing from modem type 4 (Falcom GSM):
	When iChip is configured with MTYP=4, the MTYP parameter must first be changed to the special value 99 before it can be changed to some other value.
Note 4	Working with SiLABS World modems:
	With modem type 10 selected, iChip waits 300msec after issuing ATZ at the end of a session before issuing additional commands to the modem.

+iWTC — Wait Time Constant

Syntax:	AT+iWTC=*n*
	This parameter is used to set the modem register S7 to the required value (using the "ATS7=*n*" command).
Parameters:	*n*=0..255
Command Options:	The WTC parameter defines a timeout constant for a variety of modem activities. For a detailed description of this parameter, see the ATS7=*n* command in the AT command set manual for the modem in use.
Default:	45 seconds
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iWTC~*n*	Temporarily sets the Wait Time Constant value for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iWTC?	Reports the current Wait Time Constant used. The reply is followed by **I/OK**.
AT+iWTC=?	Returns the message "0-255". The reply is followed by **I/OK**.

+iTTO — TCP Timeout

Syntax:	AT+iTTO=*n*
	Sets the number of seconds iChip allots an Internet transaction to complete before returning the timeout error.
Parameters:	*n*=0..3600 seconds
Command Options:	The TTO parameter defines the timeout constant for Internet transactions. iChip will return with a timeout error for any TCP/UDP/IP transaction that didn't complete properly within *n*±10%. Timeout measurement is defined between receipt of an AT+i command and an iChip response to the host.
	In dial-up environments, timeout measurement begins only after establishing a PPP connection. Furthermore, an additional 10-15 seconds may be required to allow the iChip to disconnect the modem.
	n=0 is a special case where internal timeout constants will be used.
Default:	0 (use iChip's factory default timeout values)
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iTTO~*n*	Temporarily sets the Internet transaction timeout value for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iTTO?	Reports the current Internet transaction timeout used. The reply is followed by **I/OK**.
AT+iTTO=?	Returns the message "0-3600" followed by **I/OK**.

+iPGT — PING Timeout

Syntax:	AT+iPGT=*n*
	Sets the timeout in milliseconds, after which iChip will reissue a PING request that has not been replied to.
Parameters:	*n*=0, 50..65535 milliseconds
Command Options:	After issuing a PING request, in response to the AT+iPING command, iChip will wait up to *n* milliseconds for a reply. If a reply is not received, iChip will reissue the PING request.
	n=0 is a special case where a timeout of 2 seconds is used.
Default:	0 (use iChip's factory default 2 seconds timeout)
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iPGT~*n*	Temporarily sets the PING transaction timeout value for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iPGT?	Reports the current PING transaction timeout used. The reply is followed by **I/OK**.
AT+iPGT=?	Returns the message "0, 50-65535" is followed by **I/OK**.

+iMPS — Max PPP Packet Size

Syntax:	AT+iMPS=*n*
	Limits the size of an outgoing PPP packet in dial-up environments. In effect, the MPS parameter limits the iChip's MTU (Maximum Transfer Unit).
Parameters:	*n*=0..3
Command Options:	
n=0	1500 bytes
n=1	256 bytes
n=2	536 bytes
n=3	1024 bytes
Default:	*n*=0
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iMPS?	Returns current value followed by **I/OK**.
AT+iMPS=?	Returns the message "0-3" followed by **I/OK**.

+iTTR — TCP Retransmit Timeout

Syntax:	AT+iTTR=*n*
	Sets the timeout, in milliseconds, after which an unacknowledged TCP packet will be retransmitted over a PPP connection by iChip.
Parameters:	*n*=1000..65535
Default:	3000 milliseconds
Result Code:	
I/OK	if *n* is within limits
I/ERROR	Otherwise
AT+iTTR?	Reports the current value followed by **I/OK**.
AT+iTTR=?	Returns the message "1000-65535" followed by **I/OK**.

+iMSS — Maximum Segment Size

Syntax:	AT+iMSS=*b*
	Permanently sets the maximum TCP segment size that iChip will negotiate with the Peer (number of data bytes it is willing to accept in each packet). In addition, this parameter defines the use of the Nagle algorithm (See Note).
Parameters:	*b* is a decimal value of bitmapped flags
Command Options:	
Bit 0	0 – Use MSS=536
	1 – Use MSS=1460
Bit 1	0 – Use Nagle Algorithm
	1 – Disable Nagle Algorithm
Default:	0 (MSS=536 with Nagle Algorithm)
Result Code:	
I/OK	If *b* is within limits
I/ERROR	Otherwise
AT+iMSS~*b*	Temporarily sets the MSS bitmapped value for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iMSS?	Reports the current MSS bitmapped value. The reply is followed by **I/OK**.
AT+iMSS=?	Returns the message “0-3”. The reply is followed by **I/OK**.
Example:	AT+iMSS=1 defines MSS=1460 with Nagle Algorithm. Only bit 0 is set, because 1 (DECIMAL) equals 01 (BINARY).

Note: When exchanging information over TCP/IP, iChip makes use of the Nagle algorithm (http://en.wikipedia.org/wiki/Nagle's_algorithm), which defines that outgoing data should be coalesced until receiving an ACK from the remote peer. A new packet of the accumulated data is sent after the ACK is received or when a full sized packet, as defined by *Bit 0* of this parameter, is ready to be sent.

+iBDRF — Define A Fixed Baud Rate on the Host Connection

Syntax:	AT+iBDRF=<*n*>
	Sets the baud rate on host serial connection. This parameter is saved to nonvolatile memory and activated only after power-up.
Parameters:	n=3..9|'a'|'h'
Command Options:	
n=a	set baud rate to Auto Baud
n=3	set baud rate to 2400
n=4	set baud rate to 4800
n=5	set baud rate to 9600
n=6	set baud rate to 19200
n=7	set baud rate to 38400
n=8	set baud rate to 57600
n=9	set baud rate to 115200
n=h	set baud rate to 230400
	When BDRF is set to a, iChip boots in auto baud rate mode. In this mode, iChip synchronizes on the first a or A character sent (normally as part of an AT or AT+i command) and detect its baud rate. The detected baud rate remains in effect until the iChip is power-cycled or issued the AT+iBDRA command.
	If BDRF is set to a fixed value and the MSEL signal is pulled low for more than 5 seconds during runtime, iChip enters Rescue mode and forces auto baud rate detection. BDRF value will be used again upon the next power-up.
Default:	'a' (Auto Baud)
Result Code:	
I/OK	If *n* is within limits. iChip will continue operating in the current baud rate setting. Further power-ups will initialize the baud rate to the new selected value, until a different AT+iBDRF command is issued.
I/ERROR	Otherwise
AT+iBDRF?	Returns the code for the specified fixed baud rate followed by **I/OK**.

AT+iBDRF=? Returns the message “3-9, ‘a’ or ‘h’” followed by **I/OK**.

Note: Setting of additional baud rates is provided via the BDRD parameter.

+iBDRM — Define A Fixed Baud Rate on the Modem Connection

Syntax:	AT+iBDRM=<*n*> Sets the baud rate on modem connection. This parameter is saved to nonvolatile memory and activated after every power-up.
Parameters:	3..9|'a'|'h'
Command Options:	
n=a	set baud rate to Auto Baud
n=3	set baud rate to 2400
n=4	set baud rate to 4800
n=5	set baud rate to 9600
n=6	set baud rate to 19200
n=7	set baud rate to 38400
n=8	set baud rate to 57600
n=9	set baud rate to 115200
n=h	set baud rate to 230400
Default:	'a' (auto baud) The iChip⇔ modem connection will be set to the same baud rate as that detected on the host⇔ iChip connection.
Result Code:	
I/OK	If n is within limits. The iChip will continue operating in the current baud rate setting. Further power-up will initialize the baud rate to the new selected value, until a different AT+iBDRM command is issued.
I/ERROR	Otherwise
AT+iBDRM?	Returns the code for the specified fixed modem baud rate followed by **I/OK**.
AT+iBDRM=?	Returns the message "3-9, 'a' or 'h'" followed by **I/OK**.

+iBDRD — Baud Rate Divider

Syntax:	AT+iBDRD=<*n*>
	When set to '0', iChip sets its host USART baud rate according to the value of the BDRF parameter. When set to any value in the range 1-255, it divides the maximum supported baud rate – 3Mbps – by that value. The quotient of this division is set as the host baud rate, and the value of BDRF is ignored.
Parameters:	
n=0	Host baud rate is determined by the BDRF parameter.
n=1-255	Host baud rate is set by dividing 3Mbps by *n*.
	For example, if *n*=2, the host baud rate will be set to 3Mbps÷2=1.5Mbps.
Default:	0 (host baud rate taken from BDRF parameter)
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iBDRD?	Reports the current value followed by **I/OK**.
AT+iBDRD=?	Returns the message **"0-255"** followed by **I/OK**.

+iAWS — Activate WEB Server Automatically

Syntax:	AT+iAWS=*v* Sets Activate Web Server flag to *v*.
Parameters:	*v*=0 | 1 | 2 | 3 | 100 | 200 | 201 | 202 | 203
Command Options:	
v=0	Automatic web server activation disabled
v=1 | 2 | 3	HTTP Web server will be activated automatically. Maximum number of concurrent browser connections is 2, 4 or 6 respectively.
v=100	HTTPS Secure Web server will be activated automatically.
v=200	HTTPS Secure Web server will be activated automatically. A special Network Configuration Page is served instead of the default Configuration Website.
v=201 | 202 | 203	HTTP Web server will be activated automatically. A special Network Configuration Page is served instead of the default Configuration Website. Maximum number of concurrent browser connections is 2, 4 or 6 respectively.
Default:	0 (Automatic web server activation disabled)
Result Code:	
I/OK	if *v* is a legal value.
I/ERROR	Otherwise
AT+iAWS?	Reports the current value of the Activate WEB Server flag followed by **I/OK**.
AT+iAWS=?	Returns the message “0-3,100,200-203” followed by **I/OK**.

+iWEBP – Web Port

Syntax:	AT+iWEBP=<*port*>
	Sets the port number for iChip's Web services. Remote browsers will need to open sockets to this port for Web services.
Parameters:	*port*=0..65535
Command Options:	
port=0	Use the well known port numbers: (80) for HTTP service or (443) for HTTPS service.
port>0	Use port number <*port*> for HTTP or HTTPS service.
Default:	0 (use well known ports)
Result Code:	
I/OK	If *port* is a legal value
I/ERROR	Otherwise
AT+iWEBP?	Returns current port number followed by **I/OK**.
AT+iWEBP=?	Returns the message "0-65535" followed by **I/OK**.

+iLATI — TCP Listening Socket to Service Remote AT+i Commands

Syntax: AT+iLATI=<*port*>

Sets the Remote AT+i service listening port number. When connected to the Internet, opens a TCP/IP listen socket on the local IP address and the specified *port*.

Parameters: *port*=0..65535

Command Options:

port=0 Remote AT+i service disabled

port=<*portnum*> Listening port to be used by a remote system when connecting to the iChip Family in order to send AT+i commands over the Internet.

The listening socket will *accept* one remote *connect* request. When a remote system connects through the listen socket, iChip will disable its local host serial port and spawn a new TCP/IP socket, ready to receive AT+i commands. AT+i response strings will be transmitted back to the same socket.

When the connected socket is closed, the local host serial port will be re-enabled and the listen socket will be ready to *accept* a new connection. The remote end may also issue the AT+iDOWN command or the +++ escape sequence to force iChip to disconnect and reboot.

Default: 0 (Disabled)

Result Code:

I/OK Upon successfully opening the remote AT+i service TCP/IP listening socket.

I/ERROR Otherwise

AT+iLATI~n Temporarily set the remote AT+i service Listen port. The permanent value will be restored after completing the next session, both if the session was successful or not.

AT+iLATI? Returns current AT+i service listening port number followed by **I/OK**.

AT+iLATI=? Returns the message “0-65535” followed by **I/OK**.

+iLAR — LATI Restrictions

Syntax:	AT+iLAR=*n*
	Restricts access to the LATI port from different network interfaces: LAN, WiFi and modem.
Parameters:	*n*=0 .. 7
Command Options:	
n=	Decimal value of bitmapped flags:
Bit 0	0 = Do not block access to the LATI port from the LAN/WiFi side.
	1 = Block access to the LATI port from the LAN/WiFi side.
Bit 1	0 = Do not block access to the LATI port from the modem side.
	1 = Block access to the LATI port from the modem side.
Bit 2	Reserved. Must be '0'.
Default:	0 (LATI port is accessible from any connected network)
Result Code:	
I/OK	If *n* is within limits.
I/ERROR	Otherwise
AT+iLAR~n	Temporarily set the LAR for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iLAR?	Returns current restrictions followed by **I/OK**.
AT+iLAR=?	Returns the message "0-7" followed by **I/OK**.

+iFLW — Set Flow Control Mode

Syntax:	AT+iFLW=*n*
	Sets the flow control mode.
Parameters:	*n*=0 .. 7
Command Options:	
n=	Decimal value of bitmapped flags:
Bit 0	0 = Host S/W flow control, using Wait/Continue control characters.
	1 = Host hardware flow control based on ~CTS/~RTS hardware signals.
Bit 1	0 = No Modem flow control.
	1 = Modem hardware flow control based on ~CTS/~RTS hardware signals.
Bit 2	0 = All hardware control signals: ~CTS, ~RTS, DTR and DSR are mirrored across iChip when transferring data transparently to the DCE.
	1 = Hardware signal mirroring is disabled.
Default:	‘0’ (Host software flow control, no modem hardware flow control)
Result Code:	
I/OK	If *n* is within limits. See Note.
I/ERROR	Otherwise
AT+iFLW~n	Temporarily sets the flow control mode for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iFLW?	Returns current flow control mode followed by **I/OK**.
AT+iFLW=?	Returns the message “0-7” followed by **I/OK**.

Note: When setting Bit 0 (Host hardware flow control), the ~CTSH signal must be LOW (enabled), otherwise iChip will return **I/ERROR (063)**.

+iCPF — Active Communications Platform

Syntax: AT+iCPF=*n*

Sets the active communications platform to either modem or LAN.

Parameters: *n*=0..1

Command Options:

n=0 Sets active communications platform to dial-up or cellular modem. When the modem is online, any character, including `<CR>`, sent from the host that is not part of an `AT+i` command is transferred directly to the modem.

n=1 Sets active communications platform to LAN.

Default: *n*=1 (LAN)

Note: This default value is shipped from the factory. The AT+iFD command does not restore CPF to this value.

Result Code:

I/OK If *n* is within limits and the communications platform was actually changed.

I/ERROR Otherwise

Followed by:

I/DONE After changing the current platform to modem. Allow a 2.5 sec. delay for iChip re-initialization.

-or-

I/ONLINE After changing the current platform to LAN.

AT+iCPF~*n* Temporarily sets the active communications platform to *n* for one session. The permanent value will be restored after completing the next session, both if the session was successful or not. Note that **I/ONLINE** or **I/DONE** will be returned according to the new permanent communications platform.

AT+iCPF? Reports the currently active communications platform followed by **I/OK**.

AT+iCPF=? Returns the message “0-1” followed by **I/OK**.

+iLTYP — Select the Active LAN Interface

Syntax: AT+iLTYP=<*n*>
Sets the active LAN interface to be WLAN, LAN or automatic scanning.

Parameters:

n=0 Scanning, iChip will try to initiate all possible LAN interfaces one after the other until the first successful initialization. The interfaces' order is predefined: first try to initialize the Marvell WiFi chipset and then try to initialize the Ethernet LAN interface.

n=1 Marvell 88w8686 WiFi chipset

n=2 Ethernet LAN

n=3 Ethernet LAN using PLC PHY[1]

Default: 0 (Scanning)

Result code:

I/OK If *n* is a legal value.

I/ERROR Otherwise

AT+iLTYP? Returns the current LTYP value followed by **I/OK**.

AT+iLTYP=? Returns the message **"0-3"** followed by **I/OK**.

Note 1: HomePlug 1.1 PLC communications based on the Intellon 5500 PHY

+iPSE — Set Power Save Mode

Syntax:	AT+iPSE=*n*
	Enables or disables iChip's Power Save Mode.
Parameters:	n=0..255
Command Options:	
n=0	Disable Power Save mode.
n=1..255	Enable Power Save mode. When Power Save mode is enabled, iChip automatically shuts down most of its circuits after a period of *n* seconds without any activity on the host or modem ports. Renewed activity on these ports restores iChip to full operational mode.
Default:	0 (Disabled)
Result Code:	
I/OK	If n is within limits
I/ERROR	Otherwise
AT+iPSE?	Reports the current Power Save mode setting followed by **I/OK**.
AT+iPSE=?	Returns the message "0-255" followed by **I/OK**.

+iSDM — Service Disabling Mode

Syntax: AT+iSDM=*n*

Sets the service disabling mode bits.

Parameters: *n*=0 .. 255

Command Options:

n= Decimal value of bitmapped flags:

Bit 0: Disable iChip's response to ICMP ECHO (PING) requests. When this bit is set, iChip will not respond to any PING requests, thereby eliminating the possibility of a PING attack on iChip.

Bit 1: Disable iChip's remote debug daemon. When this bit is set, iChip will not enable its internal (UDP) debug port, which is normally activated for administering remote support.

Bit 2: Disable unauthenticated viewing of the iChip's internal website. When this bit is set, the internal Web site may be browsed only if the remote browser provides the RPG parameter (password). In this case, when the RPG parameter contains a password value, iChip's Configuration Web site will first display a password entry form. The remote end must submit the correct RPG value in order to continue to the Configuration site's home page. iChip uses the SHA1 hash algorithm throughout the authentication process, so actual password values are never transmitted. When this bit is set, but the RPG parameter is empty, the Configuration Web site is effectively disabled, as all password values will be rejected. However, if the RPG parameter contains the special '*' wildcard value, authentication is bypassed and the authentication form will be skipped altogether. In this case, the Configuration website's home page will be displayed immediately.

Bit 3: Disable the functionality of the "+++" sequence to exit SerialNET mode. When this bit is set the Escape Code Sequence will not terminate SerialNET mode.

Bit 4: Enable the functionality of the "BREAK" signal to exit SerialNET mode, when this bit is set. By default, when this bit is not set, Break signal invokes a reset cycle but does not exit SerialNET mode.

Bit 5: Disable iChip's configuration site from showing on address: HTTP://<iChip_IP_Address>/ichip/. If the embedded web site has been enabled, and this bit is set, only

the application website will be enabled.
The WPWD parameter shall be used as the password reference for changing parameters in the application website, both when these parameters are application parameters or iChip AT+i parameters.

Bit 6: If DNS1=0.0.0.0 and bit 6 of the SDM parameter is set to 1, iChip withholds responses to DHCP requests until DNS1 is resolved. This configuration is useful when iRouter feature is enabled and iChip is set as DHCP server.

Bit 7: Disable CA Verification in SSL sessions, Secure FTP and HTTPS. When this bit is set, iChip receives the Server Certificate but does not compare it with the certificates saved in CA, CA2, CA3 and CA4 parameters. iChip uses the public key in the Server Certificate for the SSL session. In this option, CA parameter can be empty and ERROR 222 is not returned.

Default: 0 (All services enabled)

Result Code:

I/OK If *n* is within limits

I/ERROR Otherwise

AT+iSDM? Returns current Service Disabling mode followed by **I/OK**.

AT+iSDM=? Returns the message “0-255” followed by **I/OK**.

Example: AT+iSDM=56

Only bits 3, 4 and 5 are set, because 56 (DECIMAL) equals 111000 (BINARY).

+iDF — IP Protocol 'Don't Fragment' Bit Value

Syntax: AT+iDF=*n*

Sets the value of the Don't Fragment bit used in all subsequent IP packets.

Parameters: *n*=0..1

Command Options:

n=0 IP packets transmitted may be fragmented by routers.

n=1 IP packets transmitted may not be fragmented by routers.

Default: 0

Result Code:

I/OK If *n* is within limits

I/ERROR Otherwise

AT+iDF~*n* Temporarily sets the IP protocol Don't Fragment bit to *n* for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.

AT+iDF? Reports the current IP protocol Don't Fragment bit setting followed by **I/OK**.

AT+iDF=? Returns the message "0-1" followed by **I/OK**.

+iCKSM — Checksum Mode

Syntax:	AT+iCKSM=<*n*>
	Sets iChip's checksum mode. With this mode enabled, iChip calculates the checksum of data it returns to host upon receiving the AT+iSRCV command. At the same time, iChip expects the host to append checksum to the data it sends with the AT+iSSND command. iChip compares the checksum it calculates with the one calculated by the host to verify that data was not corrupted during transmission between host and iChip.
Parameters:	*n*=0 | 1
Command Options:	
n=0	Checksum mode disabled
n=1	Checksum mode enabled
Default:	*n*=0 (checksum mode disabled)
Result code:	
I/OK	If *n* is either '0' or '1'.
I/ERROR	Otherwise
AT+iCKSM?	Reports the current Checksum Mode followed by **I/OK**.
AT+iCKSM=?	Returns the message "0-1" followed by **I/OK**.

+iHIF — Host Interface

Syntax: AT+iHIF=*n*

Specifies the interface to be used for communication between the host processor and iChip in subsequent sessions. This parameter takes effect only after power-up.

Parameters:

n=0 Automatic host interface detection. In this mode, the first character sent from the host over one of the supported interfaces sets the host interface to be used throughout that session until the next iChip power cycle.

If HIF is set to a fixed interface (*n*=1-6) and the MSEL signal is pulled low for more than 5 seconds during runtime, iChip switches to auto host interface detection mode (HIF=0).

n=1 USART0

n=2 USART1

n=3 USART2

n=4 USB Device

n=5 USB Host

n=6 SPI1

n=101 Half-Duplex, 2-wire, RS485 on USART0

n=102 Half-Duplex, 2-wire, RS485 on USART1

Default: 0 (Automatic host interface detection)

Result Code:

I/OK If *n* is within limits

I/ERROR Otherwise

AT+iHIF? Reports the current value followed by **I/OK**.

AT+iHIF=? Returns the message **“0-6,101,102”** followed by **I/OK**.

+iMIF — Modem Interface

Syntax:	AT+iMIF=*n*
	Specifies the interface to be used for communication between iChip and a dialup or cellular modem in subsequent sessions. This parameter takes effect only after power-up.
Parameters:	
n=1	USART0
n=2	USART1
n=3	USART2
n=4	USB Device
n=5	USB Host (Some modems may not be compatible)
Default:	2 (USART1)
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iMIF?	Reports the current value followed by **I/OK**.
AT+iMIF=?	Returns the message "1-5" followed by **I/OK**.

+iADCL — ADC Level

Syntax: AT+iADCL=<*level*>

Specifies an ADC base level, or threshold, in the range 0-255 that corresponds to an analog voltage measured on the input pin of iChip's A/D converter.

Together with ADCD, these two parameters determine when the A/D converter asserts the GPIO pin specified by the ADCP parameter. ADCL must be greater than ADCD.

Parameters:

level=0 A/D converter polling is disabled

level=1-255 ADC threshold level

Default: 0 (polling disabled)

Result Code:

I/OK If level is within limits

I/ERROR Otherwise

AT+iADCL? Reports the current value followed by **I/OK**.

AT+iADCL=? Returns the message **"0-255"** followed by **I/OK**.

+iADCD — ADC Delta

Syntax:	AT+iADCD=<*delta*>
	Specifies an ADC delta. Together with ADCL, these two parameters determine when the A/D converter asserts the GPIO pin specified by the ADCP parameter. ADCD must be less than ADCL.
Parameters:	
delta=0-255	ADC delta
Default:	0 (zero delta)
Result Code:	
I/OK	If *delta* is within limits
I/ERROR	Otherwise
AT+iADCD?	Reports the current value followed by **I/OK**.
AT+iADCD=?	Returns the message **"0-255"** followed by **I/OK**.

+iADCT — ADC Polling Time

Syntax:	AT+iADCT=<*interval*>
	Specifies the time interval between consecutive queries of the value of the A/D converter's register. iChip's response time to value changes is up to 40ms.
Parameters:	
interval=0	A/D converter polling is disabled.
interval=1-65535	Time interval, in milliseconds, between queries.
Default:	0 (polling disabled)
Result Code:	
I/OK	If *interval* is within limits
I/ERROR	Otherwise
AT+iADCT?	Reports the current value followed by **I/OK**.
AT+iADCT=?	Returns the message **"0-65535"** followed by **I/OK**.

+iADCP — ADC GPIO Pin

Syntax:	AT+iADCP=<*n*>
	Defines which of iChip's general-purpose I/O pins (GPIO) is the output signal which is asserted by the A/D converter's polling mechanism.
	This parameter takes effect only after a SW or HW reset.
Parameters:	
n=0	A/D converter polling is disabled.
n=1..6	Use PIOC [<*n*>-1] as the ADC output signal.
Default:	0 (polling disabled)
Result Code:	
I/OK	If *pin* is within limits
I/ERROR	Otherwise
AT+iADCP?	Reports the current value followed by **I/OK**.
AT+iADCP=?	Returns the message **"0-6"** followed by **I/OK**.

+iRRA — iChip Readiness Report Activation

Syntax: AT+iRRA=<*n*>

Sets the type of iChip readiness indication sent to the host following a hardware reset.

Command Options:

n=0 No indication is sent.

n=1 An **I/ATI** message is sent, indicating iChip is ready to accept AT+i commands.

n=2 An **I/<*IP Address*>** message is sent, indicating iChip has an IP address and is ready for IP communication.

In a wireless LAN environment, this message indicates the following:

- iChip has established a connection with an AP.
- iChip has completed WPA negotiations. (In case the WPA protocol is used, which means that the WLSI and WLPP parameters are not empty.)
- iChip has been set to a static IP (DIP parameter is set to a value other than 0.0.0.0), or an IP address has been acquired from a DHCP server.

In a LAN environment, this message indicates that iChip has been set to a static IP (DIP parameter is set to a value other than 0.0.0.0), or an IP address has been acquired from a DHCP server.

In a dialup/cellular environment, this message indicates that a PPP connection has been successfully established with a PPP server.

n=3 The I/O pin specified by the RRHW parameter is asserted Low, indicating iChip is ready to accept AT+i commands.

n=4 The I/O pin specified by the RRHW parameter is asserted Low, indicating iChip has an IP address and is ready for IP communication.

n=5 An **I/ATI** message is sent, *and* the I/O pin specified by the RRHW parameter is asserted Low, indicating iChip is ready to accept AT+i commands.

n=6 An **I/<*IP Address*>** message is sent, *and* the I/O pin specified by the RRHW parameter is asserted Low, indicating iChip has an IP address and is ready for IP communication.

Default:	0 (No Indication)
Result code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iRRA?	Returns the current RRA value followed by **I/OK**.
AT+iRRA=?	Returns the message **“0-6”** followed by **I/OK**.

Notes:

1. The **I/ATI** and **I/<*IP Address*>** messages are sent only if:
   - iChip is set to communicate with the host over a fixed interface (HIF≠0).
   - Either the host interface is not a USART, or host⇔iChip baud rate is set to a fixed value (BDRF≠a).
   - iChip is not configured to operate in SerialNET mode.

4. In a dialup/cellular environment, the **I/<*IP Address*>** message is sent only if iChip is configured to operate in Always Online mode (TUP=2).

+iRRHW — iChip Readiness Hardware Pin

Syntax:	AT+iRRHW=<*pin*>
	Defines which of iChip's general-purpose I/O pins (GPIO) will be asserted Low to indicate iChip readiness to the host. iChip readiness indication is specified by the RRA parameter.
Parameters:	
pin=0	No hardware indication is given.
pin=1..6	Use PIOC [<*n*>-1]
Default:	0 (no hardware indication is given)
Result Code:	
I/OK	If *pin* is within limits
I/ERROR	Otherwise
AT+iRRHW?	Reports the current value followed by **I/OK**.
AT+iRRHW=?	Returns the message **"0-6"** followed by **I/OK**.

Note: Before specifying the I/O pin for this parameter, it is recommended that you consult the pin-out section of the iChip datasheet. Incorrect selection of pin might cause unexpected iChip behavior.

+iSPIP — SPI GPIO Pin

Syntax:	AT+iSPIP=<*n*>
	Defines which of iChip's general-purpose I/O pins (GPIO) will output the SPI Control Signal. Also defines the SPI data signal polarity and SPI clock phase.
	This parameter takes effect only after a SW or HW reset.
Parameter:	*n* is a decimal value of bitmapped flags
<p>=Bits 0..3	Select which pin to use as the SPI Control Signal *p = 0*: SPI Control Signal is disabled *p* = 1..6: Set PIOC [<p>-1] as the SPI Control signal *p* = 7: Reserved
Bit 4	Selects the inactive state of the SPI clock
	Bit 4 = 0: The inactive state of the SPI clock is LOW
	Bit 4 =1: The inactive state of the SPI clock is HIGH
Bit 5	Select when to capture SPI data
	Bit 5 = 0: Data is changed on the leading edge of the SPI clock and captured on the following edge.
	Bit 5 = 1: Data is captured on the leading edge of the SPI clock and changed on the following edge.
Default:	*n*=0 (SPI Control Signal is disabled)
Result code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iSPIP?	Returns the current SPIP value followed by **I/OK**.
AT+iSPIP=?	Returns the message **"0-6, 17-22, 35-38, 49-54"** followed by **I/OK**.

Note: Before specifying the I/O pin for this parameter, it is recommended that you consult the pin-out section of the iChip datasheet. Incorrect selection of pin might cause unexpected iChip behavior. WiFi and LAN modules are usually preset to SPIP=2.

ISP Connection Parameters

+iISP*n* — Set ISP Phone Number

Syntax:	AT+iISP*n*=*dial-s*
	Sets the ISP's access phone numbers.
	Use *n*=1 to set the ISP's primary access phone number.
	Use *n*=2 to set the ISP's alternate number. The alternate number is dialed after exhausting all redial attempts of the primary number.
Parameters:	*n*=1..2
dial-s=	Telephone number string, composed of digits, ',', '-', 'W', 'w', '*', '#', '!' or ' '. See description of the standard ATD command.
	Note: If a character that is defined as a delimiter is used within the dial string, the string must be entered between apostrophes.
Command Options:	
dial-s=' '	Empty access number
dial-s=<*number*>	*number* will be set as ISP access number
Default:	Empty. No permanent ISP access number defined.
Result Code:	
I/OK	If *dial-s* is a legal phone number string.
I/ERROR	Otherwise
AT+iISP*n*~*dial-s*	Temporarily sets the ISP's primary/alternate access number. The permanent value will be restored after completing the session, whether the session was successful or not.
AT+iISP*n*?	Reports the current value of the ISP's primary/alternate access numbers. If the number does not exist, only <CRLF> is returned. The reply is followed by **I/OK**.
AT+iISP*n*=?	Returns the message "Phone #" followed by **I/OK**.

+iATH — Set PPP Authentication Method

Syntax:	AT+iATH=*v*
	Sets authentication method to *v*.
Parameters:	*v*=0 .. 2
Command Options:	
v=1	Use PAP authentication
v=2	Use CHAP authentication
Default:	1 (PAP)
Result Code:	
I/OK	If v is within limits
I/ERROR	Otherwise
AT+iATH~*v*	Temporarily sets the authentication method to *v* for the duration of the next session. The permanent value will be restored after completing the session, whether the session was successful or not.
AT+iATH?	Reports the current setting of the authentication method followed by **I/OK**.
AT+iATH=?	Returns the message “0-2” followed by **I/OK**.

+iUSRN — Define Connection User Name

Syntax:	AT+iUSRN=*user*
	Sets connection user name.
Parameters:	*user*=nser name to be used when logging onto the ISP.
Command Options:	
user=“ ”	Empty: No user name defined.
user=<*user-name*>	*user-name* is used to login to the ISP.
Default:	
user=“ ”	Empty. No user name defined. The login user name can be defined Ad-Hoc.
Result Code:	
I/OK	If *user* is an empty or legal ISP login name.
I/ERROR	Otherwise
AT+iUSRN~*user*	Temporarily sets the login user name to *user*. The permanent value will be restored after completing the next session, whether the session was successful or not.
AT+iUSRN?	Reports the current login user name. If the user name does not exist, only <CRLF> is returned. The reply is followed by **I/OK**.
AT+iUSRN=?	Returns the message ‘String’ followed by **I/OK**.

+iPWD — Define Connection Password

Syntax:	AT+iPWD=*pass*
	Sets connection password.
Parameters:	*pass*=Password to be used when logging onto the ISP.
Command Options:	
pass=''	Empty — no password defined.
pass=<*password*>	*password* is used to login to the ISP.
Default:	Empty — no password defined. The login password can be defined Ad-Hoc.
Result Code:	
I/OK	If *password* is an empty or a legal ISP login password.
I/ERROR	Otherwise
AT+iPWD~*pass*	Temporarily sets the login password to *pass*. The permanent value will be restored after completing the next session, whether the session was successful or not.
AT+iPWD?	Reports the current login password. The reported value will consist of '*' characters. The number of '*' characters shall reflect the number of characters in the actual password. If a password does not exist only <CRLF> will be returned. The reply is followed by **I/OK**.
AT+iPWD=?	Returns the message ‘String’ followed by **I/OK**.

+iRDL — Number of Times to Redial ISP

Syntax:	AT+iRDL=*n*
	Sets the number of times to redial ISP.
Parameters:	*n*= Number of redial attempts to the ISP. If the ISP number is busy or the ISP does not pick up the line, the system will attempt to redial the ISP after a delay period as defined in the RTO parameter. If all redial attempts are exhausted, an attempt to dial the alternate ISP number will be made, if an alternate number exists. In the event that the number is busy or the ISP does not respond, the system will attempt to redial up to *n* times, as with the primary ISP number. If all redial attempts are exhausted, the system will quit with the error message: "All Redial Attempts Failed."
	If the ISP does not pick-up the line, the iChip will timeout and determine a redial situation after the number of seconds stored in the WTC iChip parameter.
Command Options:	*n*=0 .. 20
Default:	*n*=5
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iRDL~*n*	Temporarily sets the number of times to redial the ISP. The permanent number of redial attempts will be restored after completing the next session, whether the session was successful or not.
AT+iRDL?	Reports the current value of the number of times to redial ISP followed by **I/OK**.
AT+iRDL=?	Returns the message "0-20" followed by **I/OK**.

+iRTO — Delay Period between Redials to ISP

Syntax:	AT+iRTO=*n*
	Sets delay period, in seconds, between redials to ISP.
Parameters:	*n*= Number of seconds to delay before redialing the ISP, after a busy signal or in the event that the ISP did not answer the call. iChip will enforce a minimal 5 second delay for values of *n* less than 5 seconds.
Command Options:	*n*=0 .. 3600 [seconds]
Default:	*n*=180 [seconds]
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iRTO~*n*	Temporarily sets the number of seconds to delay before redialing the ISP. The permanent value will be restored after completing the next session, whether the session was successful or not.
AT+iRTO?	Reports the current number of seconds to delay before redialing the ISP. The reply is followed by **I/OK**.
AT+iRTO=?	Returns the message “0-3600” followed by **I/OK**.

Server Profile Parameters

+iLVS — 'Leave on Server' Flag

Syntax:	AT+iLVS=*v*
	Sets the 'Leave on Server' flag to *v*.
Parameters:	*v* = 0 | 1
Command Options:	
v=0	After successful retrieval, messages will be deleted from server.
v=1	All messages will remain on server.
Default:	1
Result Code:	
I/OK	If *v* = 0 or 1
I/ERROR	Otherwise
AT+iLVS~*v*	Temporarily sets the Leave on Server flag to *v* for the duration of the next session. The permanent value will be restored after completing the session, whether the session was successful or not.
AT+iLVS?	Reports the current value of the Leave on Server flag followed by **I/OK**.
AT+iLVS=?	Returns the message "0-1" followed by **I/OK**.

+iDNSn — Define Domain Name Server IP Address

Syntax:	AT+iDNS*n[p]=IP*
	Sets the Domain Name Server IP Address.
	Use *n*=1 to define the Primary IP address of the Domain Name Server associated with the ISP.
	Use *n*=2 to define the alternate IP address.
	IP::=<nnn>.<nnn>.<nnn>.<nnn>
	where,
	<nnn>: [000..255]
Parameters:	
n=1..2	
p=	Optional communication platform modifier. Where, *p*='S' to force the (serial) dial-up platform and *p*='L' to force the LAN platform. *p* may be used to select any platform. If *p* is omitted, the active platform will be used.
Command Options:	
IP=0.0.0.0	Empty: No DNS defined.
IP=<*IP add*>	*IP add.* will be used to communicate to the Domain Name Server on the Internet.
Default:	Empty. No DNS defined. The DNS must be defined Ad-Hoc.
	In a LAN environment, an empty DNS (0.0.0.0) will acquire a value from the DHCP server (if DIP is 0.0.0.0).
	In a dial-up environment, the ISP will assign a DNS IP to an empty DNS, if the ISP supports RFC 1877 (PPP Extensions for Name Server Addresses).
Result Code:	
I/OK	If *IP* is an empty or legal IP address.
I/ERROR	Otherwise
AT+iDNS*n[p]~IP*	Temporarily sets the DNS IP addresses. The permanent values will be restored after completing the next session, whether the session was successful or not.
AT+iDNS*n[p]*?	Reports the current main/alternate DNS address. If no DNS address exists, 0.0.0.0 will be returned. The reply is followed by **I/OK**.
AT+iDNS*n[p]*=?	Returns the message 'IP Addr.' followed by **I/OK**.

Note: This parameter may be omitted when the target server is defined with an IP addresses rather than a symbolic name.

+iSMTP — Define SMTP Server Name

Syntax:	AT+ iSMTP*[p]=server*
	Sets the SMTP Server Name or IP.
Parameters:	
server	An SMTP server name or IP address. Server names must be resolvable by the primary or alternate DNS.
p	Optional communication platform modifier. Where, *p*='S' to force the (serial) dial-up platform and *p*='L' to force the LAN platform. *p* may be used to select any platform. If *p* is omitted, the active platform will be used.
Command Options:	
server = ''	Empty: No server is defined.
server = <*SMTP_SRVR*>	*SMTP_SRVR* will be used to locate and establish an SMTP connection when sending Email messages. If *SMTP_SRVR* is a symbolic name, a DNS server will be used to resolve the IP address.
Default:	Empty. No SMTP server defined. To send Email messages, the SMTP server name must be defined Ad-Hoc.
	In a LAN environment, an empty SMTP server will acquire a value from the DHCP server (if DIP is 0.0.0.0).
Result Code:	
I/OK	If *server* is an empty or legal IP address or server name.
I/ERROR	Otherwise
AT+iSMTP*[p]~ server*	Temporarily sets the SMTP server name to *server*. The permanent server name will be restored after completing the next session, whether the session was successful or not.
AT+iSMTP*[p]*?	Reports the current SMTP server name. If a server name does not exist, only <CRLF> will be returned. The reply is followed by I/OK.
AT+iSMTP*[p]*=?	Returns the message 'String/IP'. The reply is followed by I/OK.

+iSP — SMTP Server Port

Syntax: AT+ iSP=*port* Permanently sets SMTP server's port.

Parameters: *port* = SMTP service port number to use when sending Email.

Command Options:

port = 0..65535

Default: 25

Result code:

I/OK If *port* is in range

I/ERROR Otherwise

AT+iSP~*port* Temporarily sets the SMTP port to *port*. The permanent value will be restored after completing the session, whether the session was successful or not.

AT+iSP? Reports the current SMTP port. The reply is followed by I/OK.

AT+iSP=? Returns the message "0-65535". The reply is followed by I/OK.

+iSMA — SMTP Authentication Method

Syntax: AT+iSMA=*v* Permanently sets SMTP authentication method.

Parameters: *v* = 0 or 1

Command Options:

v=0 SMTP authentication will be disabled.

v=1 iChip will support the “AUTH LOGIN” SMTP authentication method, if forced by SMTP server.

Default: 0 (SMTP authentication disabled)

Result code:

I/OK if *v* = 0 or 1.

I/ERROR Otherwise.

AT+iSP? Reports the current method of SMTP authentication.
The reply is followed by I/OK.

AT+iSMA=? Returns the message "0-1".

+iSMU — Define SMTP Login User Name

Syntax: AT+iSMU=*user* Permanently sets Authenticated SMTP login User Name.

Parameters: *user* = User Name to be used when logging on to an SMTP server that requires authentication (if SMA is set to a non zero value).

Command Options:

user='' Empty: No SMTP authentication User Name defined.

user=<*user-name*> *user-name* will be used to login to an authenticated SMTP server.

Default: Empty. No User Name defined.

Result code:

I/OK If *user* is an empty or a legal SMTP login name.

I/ERROR Otherwise

AT+iSMU~*user* Temporarily sets the SMTP login User Name to *user*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iSMU? Reports the current SMTP login User Name. If the User Name does not exist, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iSMU=? Returns the message 'String'.
The reply is followed by I/OK.

+iSMP — Define SMTP Login Password

Syntax: AT+iSMP=*pass* Permanently sets authenticated SMTP login password.

Parameters: *pass* = Password to be used when logging on to an SMTP server that requires authentication.

Command Options:

pass='' Empty: No SMTP authentication password defined.

pass=<*password*> *password* will be used to login to an authenticated SMTP server.

Default: Empty. No password defined.

Result code:

I/OK If *password* is an empty or a legal SMTP login password.

I/ERROR Otherwise.

AT+iSMP~*pass* Temporarily sets the SMTP login password to *pass*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iSMP? Reports the current SMTP login password. The reported value will consist of '*' characters. The number of '*' characters shall reflect the number of characters in the actual password. If a password does not exist, only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iSMP=? Returns the message 'String'.
The reply is followed by I/OK.

+iPOP3 — Define POP3 Server Name

Syntax: AT+iPOP3*[p]=server*

Permanently sets the POP3 Server Name or IP.

Parameters: *server* = a POP3 Server Name or IP address. The Server Name must be resolvable by the primary or alternate DNS.
p = optional communication platform modifier for iChip Plus. Where, *p*='S' to force the (serial) dial-up platform and *p*='L' to force the LAN platform. *p* may be used to select any platform. If *p* is omitted, the active platform will be used.

Command Options:

server = '' Empty: No Server Name defined.

server = <*POP3_SRVR*> *POP3_SRVR* will be used to locate and establish a POP3 connection when recieving Email messages. If *POP3_SRVR* is a symbolic name, a DNS server will be used to resolve the IP address.

Default: Empty. No POP3 server defined. To retrieve Email messages, a POP3 Server Name must be defined Ad-Hoc. In a LAN environment, an empty POP3 server will acquire a value from the DHCP server (if DIP is 0.0.0.0).

Result code:

I/OK If *server* is empty or a legal IP address or POP3 server name.

I/ERROR Otherwise

AT+iPOP3*[p]~ server* Temporarily sets the POP3 server name to *server*. The permanent server name will be restored after completing the next session, whether the session was successful or not.

AT+iPOP3*[p]*? Reports the current POP3 server name. If a server name does not exist, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iPOP3*[p]*=? Returns the message 'String/IP'.
The reply is followed by I/OK.

+iMBX — Define POP3 Mailbox Name

Syntax: AT+iMBX=*mailbox* Permanently sets mailbox name.

Parameters: *mailbox* = Mailbox name to be used for Email retrieve.

Command Options:

mailbox ='' Empty: No mailbox name defined.

mailbox =<*mbox-name*> *mbox-name* will be used to retrieve Email messages.

Default: Empty. No mailbox defined. To retrieve Email messages, a mailbox name must be defined Ad-Hoc.

Result code:

I/OK If *mailbox* is an empty or legal mailbox name.

I/ERROR Otherwise.

AT+iMBX~*mailbox* Temporarily sets the mailbox name to *mailbox*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iMBX? Reports the current mailbox name. If a mailbox name does not exist, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iMBX=? Returns the message 'String'. The reply is followed by I/OK.

+iMPWD — Define POP3 Mailbox Password

Syntax: AT+iMPWD=*MBxPass* Permanently sets POP3 mailbox password.

Parameters: *MBxPass* = Mailbox password to be used for authentication, when retrieving Email messages from the mailbox.

Command Options:

MBxPass ='' Empty: No mailbox password defined.

MBxPass =<*mbox-pass*> *mbox-pass* will be used to authenticate receiver, when retrieving Email messages from the mailbox.

Default: Empty. No mailbox password defined. To retrieve Email messages, the mailbox password must be defined Ad-Hoc.

Result code:

I/OK If *mbox-pass* is an empty or legal mailbox password.

I/ERROR Otherwise.

AT+iMPWD~ *MbxPass* Temporarily sets the mailbox password to *MBxPass.* The permanent password will be restored after completing the next session, whether the session was successful or not.

AT+iMPWD? Reports the current mailbox password. The reported value will consist of '*' characters. The number of '*' characters shall reflect the number of characters in the actual password. If a mailbox password does not exist, only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iMPWD=? Returns the message ‘String’.
The reply is followed by I/OK.

+iNTS*n* — Define Network Time Server

Syntax: AT+iNTS*n*=<*server*>

Sets the network time server name or IP.

Use *n*=1 to define the primary time server.
Use *n*=2 to define an alternate time server.

Parameters: *n* = 1..2
server = A network timeserver name or IP address. See Appendix C for a list of NIST Time servers.

Command Options:

Server='' Empty. No Network Time Server defined.

Server=<*nts*> The server name or IP address, *nts,* will be used to retrieve the current time-of-day – if the NTOD parameter is set to enable time-of-day retrieval. Current Time-of-Day will be returned in response to the RP8 command. Outgoing Email messages will be Time and Date stamped.

Default: Empty. No Network Time Servers defined.

Result code:

I/OK If *server* is an empty or legal IP address or server name.

I/ERROR Otherwise.

AT+iNTS*n*~*server* Temporarily sets the Network Time Server to value *server*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iNTS*n*? Reports the current value of NTS*n*. If NTS*n* is empty, an empty line containing only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iNTS*n*=? Returns the message 'String / IP Addr.'.
The reply is followed by I/OK.

+NTOD — Define Network Time-of-Day Activation Flag

Syntax:	AT+iNTOD=*n*	Sets the network time-of-day activation flag to *n*. If this flag is enabled, iChip will retrieve an updated time reading the next time it goes online.
		Note: In a LAN environment, since iChip is *always* online, time retrieval will take place following a hardware or software (AT+iDOWN) reset *only*.

Parameters: *n*=0 or 1

Command Options:

n = 0: Network time retrieval from timeserver is disabled.

n = 1: Network time retrieval is enabled – iChip will connect to the time server and retrieve an updated time reading each time it connects to the network. From that point on, iChip will maintain time internally. While iChip is online, network time will be refreshed every two hours.

Current time-of-day will be returned in response to the RP8 command. Outgoing e-mail messages will be time and date stamped.

The expiry data of an incoming server certificate in secure SSL communication will also be checked. If iChip cannot read the time from the time server, an SSL session cannot be established.

Default: 0 (time server retrieval disabled)

Result code:

I/OK If *n* is 0 or 1.

I/ERROR Otherwise

AT+iNTOD~*n* Temporarily sets the network time-of-day activation flag to value *n*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iNTOD? Reports the current value of the network time-of-day activation flag followed by **I/OK**.

AT+iNTOD=? Returns the message ‘0-1’.

The reply is followed by I/OK.

+iGMTO — Define Greenwich Mean Time Offset

Syntax:	AT+iGMTO=*n*	Permanently sets iChip location's Greenwich mean time offset, in hours.
	Parameters:	*n* = -12..12
	Default:	0
	Result code:	
	I/OK	If *n* is 0 or 1.
	I/ERROR	Otherwise
	AT+iGMTO~*n*	Temporarily sets the Greenwich Mean Time Offset to value *n*. The permanent values will be restored after completing the next session, whether the session was successful or not.
	AT+iGMTO?	Reports the current value of GMTO. The reply is followed by I/OK.
	AT+iGMTO=?	Returns the message '-12-+12'. The reply is followed by I/OK.

+iDSTD — Define Daylight Savings Transition Rule

Syntax:	AT+iDSTD=DST_rule
	Permanently sets the daylight savings time transition rule, which will be applied to the time retrieved from the Time Server when reporting the current time.
Parameters:	DST_rule ::= “<HH1.DD1.MM1>;<HH2.DD2.MM2>” Where, <HH1.DD1.MM1> indicates the date when Daylight Saving Time starts and <HH2.DD2.MM2> indicates the date when Daylight Saving Time ends.
HH1, HH2	Hour in 24 hour clock format (two digits).
DD1, DD2	Either specific day in the range: 01..31, or <F/L/1/2/3/4/5><Day of Week>. <F/L> represents First/Last occurrence in a month. <1/2/3/4/5> represents the 1st, 2nd etc. occurrence in a month. <Day of Week> ::= {“Sun”, “Mon”, “Tue”, “Wed”, “Thu”, “Fri”, “Sat”}. For example, FSun is First Sunday, while 3Sun is Third Sunday.
MM1, MM2	Month (two digits)
Command Options:	
DST_rule=’’	Empty – no Daylight Saving Time definition is applied.
DST_rule=<dst>	Daylight Savings rule is defined.
Default:	Empty.
Result code:	
I/OK	If DST_rule is an empty or legal rule
I/ERROR	Otherwise
AT+iDSTD~*DST_rule*	Temporarily sets the Daylight Saving Time Definition to DST_rule . The permanent value will be restored after completing the next session, whether the session was successful or not.
AT+iDSTD?	Reports the current value of the Daylight Saving Time Definition. The reply is followed by I/OK.
AT+iDSTD=?	Returns the message ‘String’. The reply is followed by I/OK.

+iPDS*n* — Define PING Destination Server

Syntax: AT+iPDS*n*=*Server* Permanently sets the PING destination server name or IP.

Use *n*=1 to define the p*rimary d*estination server.
Use *n*=2 to define the secondary destination server.

Parameters: *n* = 1..2
Server = A network server name or IP address.

Command Options:

Server='' Empty. No PING destination Server defined.

Server=<*nps*> The server name or IP address, *nps,* will be PING'ed in order to verify iChip's online status, when iChip is in "Always Online" mode. If the primary server does not respond, iChip will try the secondary server (if it exists). When both servers do not respond to PING requests, iChip will retry to establish the connection by going offline and then online again.

Default: Empty. No PING destination Servers defined.

Result code:

I/OK If *Server* is an empty or legal IP address or server name.

I/ERROR Otherwise.

AT+iPDS*n*~*Server* Temporarily set the PING destination server to value *Server*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iPDS*n*? Report the current value of PDS*n*. If PDS*n* is empty, an empty line containing only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iPDS*n*=? Returns the message 'String / IP Addr.'.
The reply is followed by I/OK.

+iPFR — PING Destination Server Polling Frequency

Syntax:	AT+iPFR=*n*	Permanently sets the time interval, in seconds, upon which iChip will issue a PING request to one of the PING destination servers.
	Parameters:	*n* = 0..65535 [seconds]
	Command Options:	
	Default:	0 (Disabled PING polling)
	Result code:	
	I/OK	If *n* is within limits
	I/ERROR	Otherwise
	AT+iPFR~*n*	Temporarily sets the PING polling interval value for one session. The permanent value will be restored after completing the next session, whether the session was successful or not.
	AT+iPFR?	Reports the current PING polling interval used. The reply is followed by I/OK.
	AT+iPFR=?	Returns the message "0-65535". The reply is followed by I/OK.

+iPRXY — Define Proxy Server

Syntax: AT+iPRXY="<IP>:<Port>"

Sets the IP address and port of the Proxy to use for all subsequent socket communications. iChip uses the CONNECT method, supported by all Proxy servers that relay HTTPS connections.

Parameters:

<IP>:<Port> *IP* has the format x.x.x.x

Port is in the range 0..65535

Default: Empty (Do not use a Proxy server)

Result Code:

I/OK If the parameter value is a single string

I/ERROR Otherwise

AT+iPRXY? Returns the current PRXY value followed by **I/OK**.

AT+iPRXY=? Returns the message "**string"** followed by **I/OK**.

Note**:** In version 807, +iPRXY is an alias to +iUF12. Therefore, values set to +iPRXY or +iUF12 shall be stored in the same memory element. This situation shall be ramified in future FW versions (808 and onward), where +iUF12 and +iPRXY shall be individual, mutually exclusive, parameters.

+iUF*n* — User Fields and Macro Substitution

Syntax: AT+iUF*n*=<*String*> Permanently sets user field *n*.

Parameters: *n* = 01..12
String = Parameter string-value.

Command Options:
String='' Empty User Field.
String=<*Str*> *Str* is stored in the specified User Field. Maximum *Str* length is 256 characters.

A User Field may be used for general-purpose storage.
The backslash character ('\') may be used to include the quote and double-quote characters as part of a User Field contents. The backslash itself is not stored, therefore does not appear when retrieving the User Field contents. For example:
AT+iUF01="This String includes \"double\" quotes"
In addition, a User Field may be used as a macro replacement wherever an AT+i Command <*parameter*> is allowed:

The '#' character is used to prefix the UF*n* parameter to define indirection. When used, the value of the User Field will be substituted in the command before the command is processed. #UF01 -- #UF12 are allowed.

For example:

Given:	AT+iUF01=ftp.domain.com
Issuing:	AT+iFOPN:#UF01:anonymous,myemail@domain.com
Is equivalent to:	AT+iFOPN:ftp.domain.com:anonymous,myemail@domain.com

The advantage of this is that the FTP server may be specified dynamically by changing the UF01 parameter without requiring a change in the AT+iFOPN command.

Default: Empty. No User Field value defined.

Result code:
I/OK

AT+iUF*n*~<*String*> Temporarily sets User Field *n* to value *String*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iUF*n*? Reports the current value of UF*n*. If the User Field is empty, an empty line containing only <CR/LF> will be returned. The reply is followed by I/OK.

AT+iUF*n*=? Returns the message 'String', followed by I/OK.

Email Format Parameters

+iXFH — Transfer Headers Flag

Syntax:	AT+iXFH=*v*	Permanently sets 'Transfer Headers' flag to *v*.
	Parameters:	*v* = 0 or 1
	Command Options:	*v*=0 Retrieve only Email body - No headers. BASE64 MIME attachments will be decoded by iChip, on-the-fly. *v*=1 Retrieve Email headers with Email body. Attachments shall not be decoded.
	Default:	1
	Result code: I/OK I/ERROR	 If *v* = 0 or 1 Otherwise.
	AT+iXFH~*v*	Temporarily set the 'Transfer Headers Flag' to *v* for the duration of the next session. The permanent value will be restored after completing the next session, whether the session was successful or not.
	AT+iXFH?	Report the current value of the 'Transfer Headers Flag'. The reply is followed by I/OK.
	AT+iXFH=?	Returns the message "0-1". The reply is followed by I/OK.

+iHDL — Limit Number of Header Lines

Syntax:	AT+iHDL=*n*	Sets maximum number of header lines to retrieve.
	Parameters:	*n* = 0 – 255
	Default:	0 (no limit)
	Result code:	
	I/OK	If *n* is within limits
	I/ERROR	Otherwise
	AT+iHDL~*n*	Temporarily set the maximum limit of header lines for the duration of the next session. The permanent value will be restored after completing the next session, whether the session was successful or not.
	AT+iHDL?	Report the current value of the header line limit. The reply is followed by I/OK.
	AT+iHDL=?	Returns the message "0-255". The reply is followed by I/OK.

+iFLS — Define Filter String

Syntax:	AT+iFLS=*str*	Permanently sets a filter string.
	Parameters:	*str* = ASCII string which qualifies an Email message to be listed or retrieved by the iChip. This string must exist in the Email header for the message to qualify. If the string does not exist, the message will be ignored.
	Command Options:	
	str=''	Empty string: Filter disabled. All messages shall be qualified for retrieval.
	str=<*f/string*>	Set *f/string* to be the qualifying filter.
	Default:	Empty. Filter disabled.
	Result code:	
	I/OK	If *str* is an empty or legal filter string.
	I/ERROR	Otherwise
	AT+iFLS~*f/string*	Temporarily set the filter string to *f/string* The permanent value will be restored after completing the next session, whether the session was successful or not.
	AT+iFLS?	Report the current value of the filter string. If no filter is defined, only <CRLF> will be returned. The reply is followed by I/OK.
	AT+iFLS=?	Returns the message 'String'. The reply is followed by I/OK.

+iDELF — Email Delete Filter String

Syntax: AT+iDELF=[#]*str* Permanently sets the Email delete filter string.

Parameters: *str* = ASCII string which qualifies an Email message to be deleted from the mailbox. This string must exist in the Email header for the message to qualify. If the string exists in at least one header field, the message will be deleted from the mailbox during the next Email retrieve session (AT+iRMM).

Command Options:

str='' Empty string: delete filter disabled. No messages shall be deleted.

str=<*f/string*> Set *f/string* to be the qualifying Email delete filter.

flag When the optional '#' (NOT) flag precedes the filter string, iChip will reverse the deletion criterion. In other words, iChip will delete all but Emails that qualify the filter.

Default: Empty. Delete filter disabled.

Result code:

I/OK If *str* is an empty or legal filter string.

I/ERROR Otherwise.

AT+iDELF~[#]*f/string* Temporarily set the Email delete filter string to *f/string*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iDELF? Report the current value of the Email delete filter string. If no filter is defined, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iDELF=? Returns the message 'String'. The reply is followed by I/OK.

+iSBJ — Email Subject Field

Syntax: AT+iSBJ:*subject* Permanently sets Email header's Subject field.

Parameters: *subject* = Contents of subject field.

Command Options:

subject='' Empty string. 'Subject:' Field in Email header will be left empty.

subject=<*subject string*> The 'Subject:' field in the Email header will contain *subject string*

Default: Empty.

Result code:

I/OK If *subject* is an empty or legal string.

I/ERROR Otherwise.

AT+iSBJ~*subject* Temporarily set the contents of the 'Subject:' field of the next Email to be sent. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iSBJ? Report the current contents of the 'Subject:' parameter. If no subject is defined, only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iSBJ=? Returns the message 'String'.
The reply is followed by I/OK.

+iTOA — Define Primary Addressee

Syntax: AT+iTOA*[n]=Email@* Permanently sets Email addressee.

Parameters: *Email@* = Email addressee. This is the default Email addressee, which will be used to direct Email messages sent by iChip.
n = optional index of addressee. When *n* is not specified, TOA*00* (primary addressee) is used.

Command Options:

Email@='' Empty address: No addressee defined.

Email@=<*addr*> *addr* will be used as a destination address for future Email SEND commands (+iEMA, +iEMB).

n = 01..50

Default: Empty. No addressee defined.

Result code:

I/OK

AT+iTOA*[n]*~<*add*> Temporarily set the Email addressee to *add.* The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iTOA*[n]*? Report the current value of the Email addressee. If the addressee does not exist, an empty line containing only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iTOA*[n]*=? Returns the message 'String'.
The reply is followed by I/OK.

+iTO — Email ‘To’ Description/Name

Syntax: AT+iTO:*to* — Permanently sets Email header’s ‘To:’ description.

Parameters: *to* = Contents of 'To:' description/name field.

Command Options:

to=''	Empty string.
to=<*to_str*>	The 'To:' description field in the Email header will contain *to_str*.

Default: Empty

Result code:

I/OK	If *to* is an empty or legal string.
I/ERROR	Otherwise.

AT+iTO~*to* — Temporarily set the contents of the 'To:' description field of the next Email to be sent. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iTO? — Report the current contents of the *to* parameter. If the *to* parameter is empty, only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iTO=? — Returns the message ‘String’.
The reply is followed by I/OK.

+iREA — Return Email Address

Syntax: AT+iREA=*Email@* — Permanently sets the Return Email Address. This is the Email address that will be used when replying to this Email.

Parameters: *Email@* = Email addressee.

Command Options:

Email@='' — Empty address: No return address defined.

Email@=<*addr*> — *addr* will be used as the return Email address.

Default: Empty. No return Email address defined. The return Email address will be defined Ad-Hoc.

Result code:

I/OK

AT+iREA~<*addr*> — Temporarily set the return Email address to *addr*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iREA? — Report the current value of the return Email address. If the return Email address does not exist an empty line containing only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iREA=? — Returns the message 'String'.
The reply is followed by I/OK.

+iFRM — Email ‘From’ Description/Name

Syntax: AT+iFRM:*from* — Permanently sets Email header ‘From:’ description.

Parameters: *from* = Contents of 'From:' description field.

Command Options:

from='' — Empty string.

from=<*from string*> — The 'From:' description field in the Email header will contain *from string*.

Default: Empty

Result code:

I/OK — If *from* is an empty or legal string.

I/ERROR — Otherwise.

AT+iFRM~*from* — Temporarily set the contents of the 'From:' description field of the next Email to be sent. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iFRM? — Report the current contents of the *from* parameter. If the *from* parameter is empty, only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iFRM=? — Returns the message ‘String’.
The reply is followed by I/OK.

+iCCn — Define Alternate Addressee <n>

Syntax: AT+iCC*n*=*Email@* Permanently sets alternative addressee.

Parameters: *n* = 1..4
Email@ = Email addressee. This is the Email address, which will be used to copy Email messages sent by the iChip to the primary addressee list.

Command Options:

Email@='' Empty address: Alternate addressee *n* not defined.

Email@=<*addr*> *addr* will be used as alternate Email addressee *n*.

Default: Empty. No alternate addressees defined.

Result code:
I/OK

AT+iCCn~<*addr*> Temporarily set alternate addressee *n* to *addr*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iCCn? Report the current value of alternate addressee *n*. If the alternate addressee does not exist, only <CRLF> will be returned.
The reply is followed by I/OK.

AT+iCCn=? Returns the message ‘String’.
The reply is followed by I/OK.

+iBDY — Body for E-Mail Messages with Attachments

Syntax: AT+iBDY=<*text lines*> Permanently sets the contents of a plain text body to be added to outgoing e-mail messages which contain attachments, sent using AT+iEMB.

Parameters: <*text lines*> = Plain text e-mail body.

Command Options:

<*text lines*>=CRLF>.<CRLF>

Empty.

<*text lines*>={<ASCII *text line*><CRLF> …}<CRLF>.<CRLF>

The maximum fixed body size allowed is 96 characters (including embedded <CR><LF>). The e-mail body contains multi-line <CR><LF> terminated ASCII character strings. <*text lines*> must be terminated by a dot character (.) in the 1st column of an otherwise empty line.

Default: Empty

Result code:

I/OK	After all text lines are received and terminated by the (.) line.
I/ERROR	Otherwise

AT+ iBDY~<*text lines*> Temporarily sets the contents of a plain text body to be added to outgoing e-mail messages. The maximum temporary body size allowed is 1K characters (including embedded <CR><LF>). The text body is included in the next session binary message and then purged.

AT+iBDY? Reports the current contents of BDY. The reply is followed by I/OK.

AT+iBDY=? Returns the message "String". The reply is followed by I/OK.

+iMT — Media Type Value

Syntax:	AT+iMT=*type*	Permanently sets the media type used for generating Email messages with a MIME encapsulated attachment.
	Parameters:	*type* = Media type.
	Command Options:	
	type=0..4	*type* will be used as the media type: *0 – text* *1 – image* *2 – audio* *3 – video* *4 -- application*
	Default:	4 (application)
	Result code:	
	I/OK	If *type* is in the range: 0..4
	I/ERROR	Otherwise
	AT+iMT~*type*	Temporarily set the media type. The permanent value will be restored after completing the next session, whether the session was successful or not.
	AT+iMT?	Report the current media type value. The reply is followed by I/OK.
	AT+iMT=?	Returns the message "0-4". The reply is followed by I/OK.

+iMST — Media Subtype String

Syntax:	AT+iMST=*str*	Permanently sets the media subtype string used for generating Email messages with a MIME encapsulated attachment.
	Parameters:	*str* = Media subtype string.
	Command Options:	
	str=''	Empty: No media subtype string defined, the default will be used.
	str=<*string*>	*string* will be used as the media subtype string. A list of subtype strings is detailed in appendix A.
	Default:	'octet-stream'
	Result code:	
	I/OK	If *str* is an empty or a legal media subtype string.
	I/ERROR	Otherwise.
	AT+iMST~*str*	Temporarily set the media subtype string to *str*. The permanent value will be restored after completing the next session, whether the session was successful or not.
	AT+iMST?	Report the current media subtype string. If the string is empty, only <CRLF> will be returned. The reply is followed by I/OK.
	AT+iMST=?	Returns the message 'String'. The reply is followed by I/OK.

+iFN — Attachment File Name

Syntax:	AT+iFN=*fname*	Permanently sets the attachment file name string used for generating Email messages with a MIME encapsulated attachment.
	Parameters:	*fname* = Attachment file name.
	Command Options:	
	fname =''	Empty: No file name string defined, the default will be used.
	fname=<*str*>	*str* will be used as the file name string when constructing a MIME attachment. The file name should be complete with an explicit extension.
	Default:	iChip generated unique filename, without an extension.
	Result code:	
	I/OK	If *fname* is an empty or a legal file name string.
	I/ERROR	Otherwise
	AT+iFN~*fname*	Temporarily set the file name string to *fname*. The permanent value will be restored after completing the next session, whether the session was successful or not.
	AT+iFN?	Report the current file name string. If the filename is empty, only <CRLF> will be returned. The reply is followed by I/OK.
	AT+iFN=?	Returns the message 'String'. The reply is followed by I/OK.

+iCSTY – Character Set Type

Syntax:	AT+iCSTY=*charset*
	Permanently sets a character set name (not case sensitive) that iChip will use as the charset field in the header of MIME encapsulated outgoing emails, as defined in RFC2045.
Command Options:	
charset	Name of a character set chosen from the character sets registered with IANA (Internet Assigned Numbers Authority).
Default:	Empty. “us-ascii” charset will be used.
Result Code:	
I/OK	If *charset* is a legal string value
I/ERROR	Otherwise
AT+iCSTY~*charset*	Temporarily sets the charset field for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iCSTY?	Reports the current charset value. The reply is followed by **I/OK**.
AT+iCSTY=?	Returns the message “String”. The reply is followed by **I/OK**.
Example:	AT+iMSS=1 defines MSS=1460 with Nagle Algorithm. Only bit 0 is set, because 1 (DECIMAL) equals 01 (BINARY).

+iCTE – Context Encoding Type

Syntax:	AT+iCTE=*encoding*
	Permanently sets the string that iChip will use as the Content-Transfer-Encoding field in the header of MIME encapsulated outgoing emails.
Command Options:	
encoding	Name of the encoding chosen from the character sets registered with IANA (Internet Assigned Numbers Authority).
Default:	Empty.
	"7bit" encoding will be used.
Result Code:	
I/OK	If *encoding* is a legal string value
I/ERROR	Otherwise
AT+iCTE~*charset*	Temporarily sets the charset field for one session. The permanent value will be restored after completing the next session, both if the session was successful or not.
AT+iCTE?	Reports the current charset value. The reply is followed by **I/OK**.
AT+iCTE=?	Returns the message "String". The reply is followed by **I/OK**.
Example:	AT+iMSS=1 defines MSS=1460 with Nagle Algorithm. Only bit 0 is set, because 1 (DECIMAL) equals 01 (BINARY).

Note: In firmware version 807 the CTE parameter is an alias to the CTT parameter. If the host uses both the EMB and SLNK commands, it must set the proper value to the CTE/CTT parameter prior to using each command. In firmware version 809 and onwards the CTE and CTT parameters will be separated.

IP Registration Parameters

+iRRMA — IP Registration Mail Address

Syntax: AT+iRRMA= *Email@* Permanently sets the IP registration addressee.

Parameters: *Email@* = Email addressee. This addressee will receive a registration Email message after iChip establishes an Internet session connection as a result of an explicit AT+i command or as a result of automated Internet session establishment procedures. The Email will contain the iChip's ID and dynamically assigned IP address, in ASCII form. See Email IP Registration.

Command Options:

Email@='' Empty address: No Email will be sent after iChip goes online.

Email@=<*addr*> *addr* will be used as the IP registration Email addressee.

Default: Empty.

Result code:

I/OK

AT+iRRMA? Report the current value of the IP registration addressee. If the IP registration addressee does not exist, an empty line containing only <CR/LF> will be returned.
The reply is followed by I/OK.

AT+iRRMA=? Returns the message 'String'.
The reply is followed by I/OK.

+iRRSV — IP Registration Host Server Name

Syntax: AT+iRRSV=*server_name:port*

Permanently sets the IP registration server name or IP and port number to be used in an IP registration procedure.

Parameters: *server_name* = A server name or IP address. Server names must be resolvable by the primary or alternate DNS.
port = 0..65535

Command Options:

server_name='' Empty: No IP registration server name defined.

server_name=<*ip_registration_server*>

ip_registration_server will be used to locate and establish a connection after iChip establishes an Internet session connection as a result of an explicit AT+i command or as a result of automated Internet session establishment procedures. The dynamically assigned IP address will be sent to the server in ASCII form, after which the socket will be closed. See Socket IP Registration.

port=<port number>

It is assumed that the host server is "listening" on *port number*.

Default: Empty. No server defined.

Result code:

I/OK If *ip_registration_server* is an empty or legal server name and *port* is within limits.

I/ERROR Otherwise.

AT+iRRSV? Report the current IP registration server name and port number. If a server name does not exist, only <CR/LF> will be returned.
The reply is followed by I/OK.

AT+iRRSV=? Returns the message ‘Name/IP:Port’.
The reply is followed by I/OK.

+iRRWS — IP Registration Web Server

Syntax: AT+iRRWS=*url* — Permanently sets the IP registration web server URL.

Parameters: *url* = The web server URL to use for registration after going online.

Command Options:

url = ‘’ — Empty: No IP registration URL defined.

url = <*Reg_URL*> — *Reg_URL* will be used to dynamically register iChip’s IP and Port after going online as a result of an explicit AT+i command or as a result of automated Internet session establishment procedures. See Web Server IP Registration.

Default: Empty. No Registration Web server defined.

Result code:

I/OK — If *Reg_URL* is an empty or legal URL string.

I/ERROR — Otherwise.

AT+iRRWS? Report the current IP registration Web server URL. If a URL does not exist only <CR/LF> will be returned. The reply is followed by I/OK.

AT+iRRWS=? — Returns the message “String”. The reply is followed by I/OK.

+iRRRL — IP Registration Return Link

Syntax: AT+iRRRL=*IP[:Port]* — Permanently sets the IP registration Return Link IP and Web Port.

Parameters: *IP* = IP address to use for registration after going online.
Port = Port number to assign to iChip's Web server.

See description of RRRL when registering IP.

Command Options:

IP = 0.0.0.0 — Empty: No Return Link defined.

IP = <*IP_add*> — *IP_addr* will be used when registering after establishing an Internet session, rather than the iChip's actual local IP address. This is useful when the iChip receives an internal IP address behind a NAT. Assigning the NAT's IP address to *IP_addr* will allow reaching the iChip from the Internet. In SerialNET, the LPRT parameter may be pre-configured in the NAT to connect to the iChip device. See SerialNET Server Devices.

Port = *Web_port* — Optional port to map iChip's Web server in order to allow surfing iChip across a NAT in association with *IP_addr*.

Default: Empty. No return link IP and Port defined.

Result code:

I/OK — If *IP* is a legal IP address and *Port* is a legal IP port number.

I/ERROR — Otherwise.

AT+iRRRL? — Report the current return link IP and port. The reply is followed by I/OK.

AT+iRRRL=? — Returns the message "Name/IP[:Port]". The reply is followed by I/OK.

+iHSTN — iChip LAN Host Name

Syntax: AT+iHSTN=*host* Permanently sets iChip's Network Host Name.

Parameters: *host* = Symbolic Host Name string.

Command Options:

host = '' Empty: Do not attempt to register a symbolic host name. If the iChip LAN is already registered in the DNS, the symbolic name will typically be cleared only after the last DHCP lease assigned to this iChip has expired.

host = <*NAME*> *NAME* will be used to negotiate the registration of the iChip LAN on the LAN's DNS server via the DHCP server. Host name negotiation will be implemented only during the **next** DHCP session. Typically this session will occur after a hardware reset or by issuing the AT+iDOWN command. Note that in order to achieve a successful host name registration, the iChip LAN must utilize a DHCP (DIP = 0.0.0.0) and the DHCP server must both exists and be configured to dynamically add entries to the local DNS server.
NAME will also be included in all IP registration method formats.

Default: Empty. No network host name defined.

Result code:

I/OK If *host* is empty or a string.
I/ERROR Otherwise.

AT+iHSTN? Report the current host name.
The reply is followed by I/OK.

AT+iHSTN=? Returns the message 'string'.
The reply is followed by I/OK.

HTTP Parameters

+iURL — Default URL Address

Syntax: AT+iURL=*URLadd* Sets the URL address string used for downloading web pages and files and uploading files to web servers.

Parameters: *URLadd* = URL address string.

Command Options:

URLadd ='' Empty: No URL address string defined.

URLadd =<*str*> *str* will be used as the URL address string when downloading a Web page or file.

The URL address format is:

"<Protocol>://<host>[:<port>]/[<absolute_link>]"

Where,

<protocol> -- HTTP or HTTPS

<host> -- Web Server Name: IP address or server name resolved by DNS.

<port> -- Port number on server. Default: 80 for HTTP, 443 for HTTPS.

<absolute link> -- Absolute path name of Web page or file on server.

Default: None

Result code:

I/OK If *URLadd* is an empty or a legal URL address string surrounded by quotation marks.

I/ERROR Otherwise

AT+iURL~*URLadd* Temporarily set the URL address string to *URLadd*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iURL? Reports the current URL address string. If the URL address is empty, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iURL=? Returns the message 'String'. The reply is followed by I/OK.

+iCTT — Define Content Type Field in POST Request

Syntax:	AT+iCTT=<*string*>
	Defines the contents of the "Content-type:" field that is sent in the POST request by the AT+iSLNK command. This field specifies the type of file being sent.
Parameters:	*string*=max length 64 bytes
Command Options:	
string="	Empty. A default value of "`application/x-www-form-urlencoded`" will be used, and the server will expect the data to be the data sent in a "Submit" of a form.
string=<*Content-type*>	type of file being sent by the AT+iSLNK command.
Default:	Empty
Result Code:	
I/OK	If *string* is empty or a legal string.
I/ERROR	Otherwise
AT+iCTT~<string>	Temporarily set the CTT parameter. The permanent value will be restored after completing the next session, whether the session was successful or not.
AT+iCTT?	Reports the current CTT value. If empty, only <CRLF> will be returned. The reply is followed by I/OK.
AT+iCTT=?	Returns the message 'String'. The reply is followed by I/OK.

+iWPWD — Password for Application Website Authentication

Syntax:	AT+iWPWD=*Pass*	Permanently sets the application website's remote parameter update Password.

Parameters: *Pass* = Password to be used for authentication, when accepting application Web site parameter updates from a remote Web browser.

Command Options:

Pass =''	Empty: Remote application Web site parameter updates over the Web are effectively disabled.
Pass =<*password*>	*password* will be used to restrict application Web site parameter updates via a remote Web browser.
Pass ="*"	A *password* will not be required to authenticate application Web site parameter updates via the Web, effectively unrestricting remote parameter updates.

Default: Empty. No Password defined. Application Web site parameter updates via a remote browser are fully restricted.

Result code:

I/OK	If *pass* is an empty or legal Password.
I/ERROR	Otherwise

AT+iWPWD~*pass*	Temporarily set the application Web site parameter Update Password to *pass*. The permanent Password will be restored after completing the next session, whether the session was successful or not.
AT+iWPWD?	Report the current Password. If a Password does not exist, only <CRLF> will be returned. The reply is followed by I/OK.
AT+iWPWD=?	Returns the message 'String'. The reply is followed by I/OK.

+iLOGO — Configuration Website LOGO

Syntax: AT+iLOGO=<*path*>

Sets path to a file that contains the LOGO picture that will be shown in the top frame of the configuration website that is embedded in iChip.

Parameters: *path* = Path to a file that contains the LOGO picture.
Note: The AT+iFD command does **NOT** restore *path* to its default value.

Command Options:

Path=<string> *path* can be set to:

- A relative reference to an image file that is included in the application's website, which was loaded into iChip. For example, if the main directory of the host website contains an image file called "Logo.gif", then the parameter can be set this way:
  AT+iLOGO="Logo.gif"
- A full path to an image in any website on the Web. For example, if the main directory of the customer's company website (www.customer.com) contains an image file called "Logo.gif", then the parameter can be set this way:
  AT+iLOGO="http://www.customer.com/Logo.gif"

The image's recommended properties are:
width: 193 pixels, height: 70 pixels, horizontal and vertical resolution: 96 dpi. A larger image will be resized to fit a width this size (which may result in an un-proportional picture). It is recommended to use a file that is as small as possible, of type GIF.

path=" Empty – will be automatically set to the default value.

Default: `"iChipimages/ConnectOne.gif"`
Points to the Connect One logo picture that is a part of the embedded configuration website.

Result Code:

I/OK If *path* is empty or a legal string.

I/ERROR Otherwise

AT+iWPWD~*path* Temporarily set the *path*. The permanent path will be restored after completing the next session, whether the session was successful or not.

AT+iWPWD?	Report the current path. The reply is followed by I/OK.
AT+iWPWD=?	Returns the message ‘String’. The reply is followed by I/OK.

+iLDLY – Limit the Delay for Download from HTTP and FTP Servers

Syntax: AT+iLDLY=*n*

Sets a time in seconds for iChip to wait for completion of a received stream of data from an HTTP or an FTP server. When the server does not indicate the size of the received stream, iChip shall allow *n* seconds in which data is no longer received, and then determine completion of the session.

Parameters: *n* = 0..5

n = 0 No delay.

n =1..5 Delay in seconds.

Default: 5 seconds

Result Code:

I/OK If *n* is a legal value.

I/ERROR Otherwise

AT+iLDLY~*n* Temporarily set value to the *LDLY*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iLDLY? Reports the current value.
The reply is followed by I/OK.

AT+iLDLY=? Returns the message '0-5'.
The reply is followed by I/OK.

RAS Server Parameters

+iRAR — RAS RINGs

Syntax:	AT+iRAR=*n*	Sets the number of RINGs that will activate iChip's internal RAS if RAU is not empty.
	Parameters:	*n* = number of RINGs iChip will detect before answering an incoming call and activating its internal RAS. If *n* is set to a value greater than 100 and an incoming call is picked up by the host or the modem after less than *n*-100 RINGs, iChip will activate its internal RAS. The RAS server will negotiate a PPP connection if a '~' is received as the first character from the modem after the CONNECT line to indicate a PPP packet. Otherwise, iChip will revert to transparent mode communications, allowing the host to conduct direct modem to modem data transfer.
	Command Options:	
	n =	2 .. 20
	+100	Add 100 to any RAR value to force iChip to activate its internal RAS even if the call was picked up by the host or the modem (if a '~' is received as the first character from the modem after the CONNECT line to indicate a PPP packet).
	Default:	*n* = 4
	Result code:	
	I/OK	If *n* is within limits.
	I/ERROR	Otherwise.
	AT+iRAR?	Returns RAR's current value. The reply is followed by I/OK.
	AT+iRAR=?	Returns the message "2-20". The reply is followed by I/OK.

+iRAU — Define RAS Login User Name

Syntax: AT+iRAU=*user* Permanently sets RAS login user name.

Parameters: *user* = User Name to be used for authentication when accepting a call from a PPP client connecting to iChip's internal RAS.

Command Options:

user='' Empty: iChip's internal RAS is effectively disabled.

user =<*user-name*> *user-name* will be used to establish login rights of a remote PPP client connection to iChip's internal RAS.

user ="*" A user-name will not be required to authenticate a remote PPP client connection to iChip's internal RAS, effectively unrestricting remote access.

Default: Empty. iChip's internal RAS is effectively disabled.

Result code:

I/OK If *user* is an empty or a legal login User Name.

I/ERROR Otherwise.

AT+iRAU~*user* Temporarily set the RAS login User Name to *user*. The permanent value will be restored after completing the next session, whether the session was successful or not.

AT+iRAU? Reports the current RAS login User Name. If the User Name does not exist, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iRAU=? Returns the message 'String'. The reply is followed by I/OK.

+iRAP — Password for RAS Authentication

Syntax: AT+iRAP=*Pass* Sets the RAS Password.

Parameters: *Pass* = Password to be used for login authentication when accepting a call from a PPP client connecting to iChip's internal RAS.

Command Options:

Pass ='' or *Pass*=''*''	A *password* will not be required to authenticate a remote PPP client connection to iChip's internal RAS.
Pass =<*password*>	*password* will be used to restrict access of a remote PPP client connection to iChip's internal RAS.

Default: Empty. No Password defined.

Result code:

I/OK	If *pass* is an empty or legal Password.
I/ERROR	Otherwise.

AT+iRAP~*pass* Temporarily set the RAS password to *pass*. The permanent Password will be restored after completing the next session, whether the session was successful or not.

AT+iRAP? Reports the current RAS Password. If a Password does not exist, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iRAP=? Returns the message 'String'. The reply is followed by I/OK.

Unique Identifiers

+iSNUM — iChip Serial Number

Syntax: AT+iSNUM=<*serial_number*>

Programs iChip's serial number into flash memory.

Parameters:

<*serial_number*> iChip's serial number consisting 8 hexadecimal characters. The serial number can be assigned only once, while the current serial number is still FFFFFFFF. Once a serial number is assigned, it cannot be modified. To find out the current serial number, use the AT+iRP5 command.

Default: The serial number assigned at the factory.

Result Code:

I/OK If *serial_number* is a legal hexadecimal string and is being set for the first time.

I/ERROR (068) Serial number already exists.

AT+iSNUM=? Returns the message "**String**" followed by **I/OK**.

+ iUID — Unique ID

Syntax: AT+iUID=<*ID*>

Programs a unique ID number into the Flash memory that iChip is connected to. This value is provided empty and may be utilized by OEM manufacturers.

Parameters:

<*ID*> A unique string consisting of 8 characters or less. This string can be assigned only once, while the current ID is still empty. Once an ID is assigned it cannot be modified.

Default: Empty.

Result Code:

I/OK If *ID* is 8 characters or less in length and is being set for the first time.

I/ERROR (065) A unique ID already exists.

AT+iUID? Returns the current UID value followed by **I/OK**.

AT+iUID=? Returns the message "**String**" followed by **I/OK**.

LAN Parameters

+iMACA — MAC Address of iChip

Syntax: AT+iMACA=*mac* Permanently sets iChip's MAC address.

Parameters: *mac* = MAC address. The MAC address may only be assigned **once** in the lifetime of the device, i.e., while the current MAC address is still FFFFFFFFFFFF. After a MAC address is assigned it cannot be changed or overwritten.

Command Options:

mac=<*mac@*> *mac@* must consist of 12 hexadecimal characters. If the current MAC is FFFFFFFFFF then *mac@* will become the permanent MAC address.

Default: MAC address assigned by Connect One at the factory .[1]

Result code:

I/OK If *mac* is a legal hexadecimal string, and the MAC address is being set for the first time.

I/ERROR Otherwise

AT+iMACA? Report the current MAC address. If no MAC address has been defined, the reply will be "FFFFFFFFFFFF".
The reply is followed by I/OK.

AT+iMACA=? Returns the message 'String'.
The reply is followed by I/OK.

Note: Connect One owns a registered IEEE MAC address range. MAC addresses are normally set by Connect One in the factory in that address range. However, Users may request to purchase iChip devices without MAC address assignments in order to assign addresses in their own address range.

+iDIP — iChip Default IP Address

Syntax: AT+iDIP=*IP address* — Permanently sets iChip's default IP address to *IP address*.

Parameters: *IP address* = IP address

Command Options:

IP address = 0.0.0.0	Empty: iChip will attempt to resolve an IP address via a DHCP server (LAN) or PPP (Modem). The assigned address will be stored in the IPA (active IP address) parameter.
IP address = 255.255.255.255	*Reserved*
IP address =<*IP ADDR.*>	*IP ADDR.* will be assigned to iChip. The address will be stored in the DIP parameter. The DIP parameter's value is copied into the IPA parameter after power-up and after the AT+iDOWN command.

Default: Empty (0.0.0.0). No static IP address is defined. IP address will be resolved through a DHCP server, if one is available (LAN) or PPP (Modem).

Result code:

I/OK	If *IP address* is empty or a legal IP address.
I/ERROR	Otherwise

AT+iDIP? — Report the current Default IP address. The reply is followed by I/OK.

AT+iDIP=? — Returns the message 'IP addr.'. The reply is followed by I/OK.

+iIPA — Active IP Address

Syntax: AT+iIPA= *IP address* Changes the active IP to *IP address*.

Parameters: *IP address* = IP address.

Command Options:

IP address =<*IP ADDR.*> *IP ADDR.* will be assigned as the active iChip IP address. Also changes the permanent Default IP address in nonvolatile memory. See description of the DIP parameter.

Default: Contents of the DIP parameter at power up.

Result code:

I/OK If *IP address* is empty or a legal IP address.

I/ERROR Otherwise.

AT+iIPA? Report the current IP address.

AT+iIPA~*IP address* Temporarily set the current IP address. The permanent IP address (stored in the DIP parameter) will be restored/resolved after completing the next session, whether the session was successful or not.

AT+iIPA=? Returns the message ‘IP addr.’. The reply is followed by I/OK.

Note: The IP address is always 0.0.0.0 when iChip is connected to a modem and is offline.

+iIPG — IP Address of the Gateway

Syntax:	AT+iIPG=*IP addres*	Permanently sets the IP address of the gateway to be used by iChip.

Parameters: *IP address* = Gateway IP address.

Command Options:

IP address = 0.0.0.0	Empty: iChip will try to resolve the gateway IP address via DHCP, but **ONLY** if the DIP parameter value has been set to empty (0.0.0.0).
IP address = <*IP ADDR*>	*IP ADDR* will be used as the gateway IP address.

Default: Empty. No Gateway IP defined.

Result code:

I/OK	If *IP address* is empty or a legal IP.
I/ERROR	Otherwise

AT+iIPG~*IP address*	Temporarily set the gateway IP address. The permanent IP address will be restored/resolved after completing the next session, whether the session was successful or not.
AT+iIPG?	Report the current gateway IP. The reply is followed by I/OK.
AT+iIPG=?	Returns the message 'IP addr.'. The reply is followed by I/OK.

+iSNET — Subnet Address

Syntax: AT+iSNET=*IP mask* Sets the Sub Net to *IP mask*.

Parameters: *IP mask* = Subnet mask address.

Command Options:

IP mask =0.0.0.0 Empty: iChip will try to resolve the subnet address via DHCP, but **ONLY** if the DIP parameter value has been set to empty.
If the DIP parameter value has been set to a static IP address, then *IP mask* is automatically calculated following the next hardware or software RESET.

IP mask =<*MASK*> *MASK* will be used by iChip as the subnet mask.

Default: Empty. No subnet mask address defined.

Result code:

I/OK If *IP mask* is empty or a legal IP mask.
I/ERROR Otherwise

AT+iSNET~*IP mask* Temporarily set the subnet mask to *IP mask*. The permanent subnet mask will be restored/resolved after completing the next session, whether the session was successful or not.

AT+iSNET? Report the current subnet mask.
The reply is followed by I/OK.

AT+iSNET=? Returns the message 'IP addr.'.
The reply is followed by I/OK.

Wireless LAN Parameters

+iWLCH — Wireless LAN Communication Channel

Syntax: AT+iWLCH=<*channel*>

Sets the default WiFi communication channel.

When iChip is configured to operate in Ad-Hoc mode, this parameter must be given a value between 1 and 13 that defines the channel to be used for beacon transmission. When iChip joins an already existing Ad-Hoc network, it adopts that network's channel.

Parameters: *channel* = 0-13 (Europe) or 0-11 (USA) and 101-113 or 101-111 respectively.

Command Options:

0 The channel will be selected automatically according to the chosen SSID. Recommended when connecting to an AP.

1-13 or 1-11 When iChip is configured to operate in Ad-Hoc mode, the defined *channel* is used for beacon transmission.

+100 Add 100 to any channel selection between 1 and 13 to prohibit iChip from merging Ad-Hoc networks. Assuming +iWLSI is preceded with an '!'. The network merging procedure occurs when 2 individual Ad-Hoc networks with the same SSID come dynamically into range and merge into one large Ad-Hoc network.

Default: 0 (connecting to an Access Point)

Result Code:

I/OK If *channel* is within limits.

I/ERROR Otherwise

AT+iWLCH? Reports the currently configured WiFi communication channel followed by **I/OK**.

AT+iWLCH=? Returns the message '**0-13,101-113**' followed by **I/OK**.

+iWLSI — Wireless LAN Service Set Identifier

Syntax:	AT+iWLSI=<*ssid*>
	Sets the destination Wireless LAN Service Set IDentifier (SSID) string.
Parameters:	*ssid* = SSID required for communications with a specific WLAN Access Point (AP) or Ad-Hoc. The AP must be configured with the same SSID.
Command Options:	
ssid=''	Empty. No SSID defined. iChip will communicate with any AP.
ssid=<*ID*>	*ID* will be used as the destination SSID. *ID* must be configured in the AP for iChip to successfully communicate with that AP.
ssid=*	Prevents iChip from automatically attempting to connect to an AP or Ad-Hoc network immediately after power-up. If the *ssid* parameter value is changed to (*) while iChip is already connected to an AP, the current connection will not be affected. In power save mode, iChip will not wakeup the WiFi module to scan for available APs.
ssid=!	Optional flag indicating Ad-Hoc mode. Upon power-up, iChip will continuously search for existing Ad-Hoc networks in its vicinity and join the one having the strongest signal.
ssid=!<*ID*>	iChip will search for an Ad-Hoc network with the specified *ID* on the channel specified by +iWLCH. If it finds one it will join it, otherwise it will create a new network with this *ID*.
Default:	Empty. No SSID defined.
Result Code:	
I/OK	If *ssid* is an empty or legal SSID string.
I/ERROR	Otherwise
AT+iWLSI?	Reports the current *ssid* value followed by **I/OK**.
AT+iWLSI=?	Returns the message '**String**' followed by **I/OK**.

+iWLWM — Wireless LAN WEP Mode

Syntax:	AT+iWLWM=*md*	Sets the Wireless LAN WEP operation mode.

Parameters: *md* = 0..2

Command Options:

md=0	WEP Disabled.
md=1	WEP Enabled, using 64-bit keys.
md=2	WEP Enabled, using 128-bit keys.

Default: 0 - WEP disabled

Result code:

I/OK	if *md* is within limits.
I/ERROR	Otherwise

AT+iWLWM?	Reports the current WEP mode used. The reply is followed by I/OK.
AT+iWLWM=?	Returns the message “0-2”. The reply is followed by I/OK.

+iWLKI — Wireless LAN Transmission WEP Key Index

Syntax:	AT+iWLKI=*ki*	Sets the Wireless LAN transmission WEP-Key index.
	Parameters:	*ki* = 1 .. 4
	Command Options:	
	ki=<*key_indx*>	When transmitting WiFi packets, the WEP key at position *key_indx* in the 4 key array will be used for packet encryption.
	Default:	1
	Result code:	
	I/OK	if *ki* = 1 .. 4
	I/ERROR	otherwise
	AT+iWLKI?	Reports the current Wireless LAN transmission WEP key index. The reply is followed by I/OK.
	AT+iWLKI=?	Returns the message "1-4".

+iWLK*n* — Wireless LAN WEP Key Array

Syntax: AT+iWLK*n*=*keyString* Permanently sets the Wireless LAN WEP keys in the 4-slot WEP key array.

Parameters: *n* = 1..4.
keyString = WEP key string represented by a Hexadecimal ASCII string.

Command Options:
keyString='' Empty: No WEP key defined in position *n*.
keyString=<*key*> *key* will be used as the key string value in position *n*. The identical value must be configured in the same position in the AP router.

key must be a Hexadecimal representation string, where each byte is described by 2 ASCII characters in the range [‘0’..’9’], [‘A’..’F’] or [‘a’..’f’].

When using 64-bit WEP (WLWM=1), *key* may contain up to 10 characters (defining 5 bytes). When using 128-bit WEP (WLWM=2), *key* may contain up to 26 characters (defining 13 bytes).

Default: Empty. No WEP key defined.

Result code:
I/OK if *keyString* is an empty or legal WEP key string.
I/ERROR otherwise

AT+iWLK*n*? Reports the current WEP key value in position *n*. The reported value will consist of '*' characters. The number of '*' characters shall reflect the number of characters in the actual key string. If the key string is empty, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iWLK*n*=? Returns the message ‘String’. The reply is followed by I/OK.

+iWLPS — Wireless LAN Power Save

Syntax: AT+iWLPS=*n*

Sets a time interval during which the Marvell WiFi chipset connected to iChip remains in Power Save mode. Value changes take effect only after a SW or HW reset.

Parameters:

n=0 WiFi chipset Power Save mode is disabled.

n=1-5 The number of beacon periods during which the WiFi chipset remains in Power Save mode. The beacon period is set by the Access Point (AP) and is typically 100ms. In Ad-Hoc mode, the beacon period is set by the creator of the Ad-Hoc network. In iChip the beacon period is 100ms. In Ad-Hoc mode the WLPS time interval is always one beacon period, even if *n*>1.

Default: *n*=0 (Power Save mode disabled)

Result Code:

I/OK If *n* is within limits

I/ERROR Otherwise

AT+iWLPS? Returns the current value stored in WLPS followed by **I/OK**.

AT+iWLPS=? Returns the message **"0-5"** followed by **I/OK**.

note: iChip will not wake up periodically when it is in WiFi power-save mode and the +iWLSI parameter is set to '*'.

+iWLPP — Personal Shared Key Pass-Phrase

Syntax:	AT+iWLPP=<*passphrase*> Sets the wireless LAN WPA-PSK pass-phrase. For WPA2, the WSEC parameter must be set as well.
Parameters:	<*passphrase*> = Pass-phrase to be used in generating the WPA-PSK encryption key.
Command Options:	
passphrase =”	Empty — WPA security is disabled.
passphrase =<*pass*>	If WLSI (SSID) is not empty, WPA-PSK security is enabled for WiFi connections and *pass* is used in generating the WPA-PSK encryption key. The allowed value for *pass* is an ASCII string containing 8-63 characters.
Default:	Empty
Result code:	
I/OK	If *pass* is an empty or legal pass-phrase.
I/ERROR	Otherwise
AT+iWLPP?	Reports the current pass-phrase. The reported value consists of ‘*’ characters. The number of ‘*’ characters reflects the number of characters in the pass-phrase. If a pass-phrase is not defined, only <CRLF> are returned. The reply is followed by **I/OK**.
AT+iWLPP=?	Returns the message ‘**String**’ followed by **I/OK**.

+iWSEC — Wireless LAN WPA Security

Syntax:	AT+iWSEC=*n*
	Sets the WPA protocol type to be used for wireless LAN security. This parameter takes effect following either a hardware or software reset (AT+iDOWN) only. A change to this parameter during iChip operation does not affect the current connection.
Parameters:	
n=0	WPA-TKIP protocol
n=1	WPA2-AES protocol
Default:	0
Result Code:	
I/OK	if *n* is within limits
I/ERROR	Otherwise
AT+iWSEC?	Reports the current value followed by I/OK.
AT+iWSEC=?	Returns the message “0, 1” followed by I/OK.

+iWROM — Enable Roaming in WiFi

Syntax:	AT+iWROM=<*n*>
	Sets iChip to Roaming mode.
Parameters:	*n*=0 | 1
n=0	Disable Roaming mode.
n=1	Enable Roaming mode.
Default:	*n*=0
Result Code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iWROM?	Returns the current WROM value followed by **I/OK**.
AT+iWROM=?	Returns the message "**0-1**" followed by **I/OK**.

+iWPSI — Periodic WiFi Scan Interval

Syntax:	AT+iWPSI=*n*
	Sets the time interval – *n* – between consecutive scans that iChip performs for APs in its vicinity.
Parameters:	*n*=1-3600 seconds
Default:	*n*=5 seconds
Result Code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iWPSI?	Returns the current WPSI value followed by **I/OK**.
AT+iWPSI=?	Returns the message **"1-3600"** followed by **I/OK**.

+iWSRL — SNR Low Threshold

Syntax: AT+iWSRL=<*n*>

Sets a low SNR threshold for iChip in Roaming mode. If the SNR value of the signal from the AP that iChip is currently associated with drops below *n*, iChip is triggered by the SNR low event.

Parameters: *n*=0-254 dB

Default: *n*=10 dB

Result Code:

I/OK If *n* is a legal value.

I/ERROR Otherwise

AT+iWSRL? Returns the current WSRL value followed by **I/OK**.

AT+iWSRL=? Returns the message **"0-254"** followed by **I/OK**.

+iWSRH — SNR High Threshold

Syntax:	AT+iWSRH=<*n*>
	Sets a high SNR threshold for iChip in Roaming mode. iChip will re-associate only with APs having SNR that is better than *n*.
Parameters:	*n*=10-255 dB
Default:	30 dB
Result Code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iWSRH?	Returns the current WSRH value followed by **I/OK**.
AT+iWSRH=?	Returns the message **“10-255”** followed by **I/OK**.

+iWSI*n* — Wireless LAN Service Set Identifier Array

Syntax: AT+iWSI<*n*>=<*ssid*>

Sets the destination Wireless LAN Service Set Identifier (SSID) string into position *n* in the array. This array defines the order in which iChip attempts to connect to an AP or Ad-Hoc network.

Parameters:

n=0-9 — *n*=0 is equivalent to the WLSI parameter and defines the default SSID. The default SSID (WSI0 or WLSI) determines the type of scanning that iChip performs. If the default SSID refers to an AP, all SSIDs on the list must be configured for APs as well. If the default SSID refers to an Ad-Hoc network (starts with the (!) character), all SSIDs on the list must be configured for Ad-Hoc networks as well (start with the (!) character).

The location of an SSID within the list defines its priority, where the first SSID has the top priority. The SSIDs must be configured consecutively. For example, if WSI0 and WSI2 are set but WSI1 is not, iChip ignores WSI2.

If, for example, iChip is connected to an AP having an SSID value defined by WSI3, and that SSID is set to a different value using the AT+iWSI3=*new SSID* command, the change will take effect immediately and iChip will attempt to associate with an AP having the new SSID. If, on the other hand, iChip is not currently connected to an AP with SSID defined by WSI3 and the value of WSI3 is changed, the change will take effect only upon the next connection attempt.

<*ssid*>=<*ID*> — *ID* will be used as the destination SSID. *ID* must be configured in the AP for iChip to successfully communicate with that AP.

Command Options: *The options below apply only to WSI0.*

ssid=” — Empty. No SSID defined. iChip will communicate with the strongest AP in its vicinity.

ssid=* — Prevents iChip from automatically attempting to connect to an AP or Ad-Hoc network immediately after power-up. If the *ssid* parameter value is changed to (*) while iChip is already connected to an AP, the current connection will not be affected.
In power save mode, iChip will not wakeup the WiFi module to scan for available APs.

ssid=$$$$	A special value which enables connecting to any Internet-Enabled AP in addition to the SSIDs in the list.
ssid=!	Optional flag indicating Ad-Hoc mode. Upon power-up, iChip will continuously search for existing Ad-Hoc networks in its vicinity and join the one having the strongest signal.
Default:	Empty. No SSID defined.
Result Code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iWSI*n*?	Reports the current SSID value in position *n*.
AT+iWSI*n*=?	Returns the message **"String"** followed by **I/OK**.

+iWPP*n* — Pre-Shared Key Passphrase Array

Syntax:	AT+iWPP*n*=<*passphrase*>
	Sets the Wireless LAN PSK passphrase for WPA and WPA2 encryption for each individual SSID in the array.
Parameters:	
n=0-9	10 WPA passphrases, one for each SSID, respectively.
	Setting WPP0=<*passphrase*> is equivalent to setting the WLPP parameter, and vice versa.
<*passphrase*>=<*pass*>	*pass* is the passphrase to be used in generating the PSK encryption key for WPA and WPA2. The allowed value for *pass* is an ASCII string containing 8-63 characters.
Command Options:	
<*passphrase*>="	Empty – WPA security is disabled.
<*passphrase*>=<*pass*>	If WSI*n* is not empty, *pass* is used in generating the PSK encryption key for WSI*n*.
Default:	Empty
Result Code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iWPP*n*?	Reports the current passphrase value in position *n*. The reported value consists of (*) characters.
	The number of (*) characters reflects the number of characters in the passphrase. If a passphrase is not defined, only <CRLF> is returned. The reply is followed by **I/OK**.
AT+iWPP*n*=?	Returns the message **"String"** followed by **I/OK**.

+iWKY*n* — Wireless LAN WEP Key Array

Syntax:	AT+iWKY*n*=<*KeyString*>
	Sets the Wireless LAN WEP key for each individual SSID in the array.
Parameters:	
n=0-9	10 WEP keys, one for each SSID, respectively.
	Setting *KeyString* with *n*=0 is equivalent to setting the WLK1 parameter.
<*KeyString*>	WEP key string represented by a hexadecimal ASCII string.
Command Options:	
KeyString=”	Empty
KeyString=<*key*>	*key* will be used as the *KeyString* value in position *n*.
	key must be a hexadecimal representation ASCII string, where each byte is described by two ASCII characters in the range [0.. 9], [A.. F] or [a.. f].
	When using 64-bit WEP encryption (WST0=1), *key* can contain up to 10 characters (defining 5 bytes). When using 128-bit WEP encryption (WST0=2), *key* can contain up to 26 characters (defining 13 bytes).
Default:	Empty
Result Code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iWKY*n*?	Reports the current WEP key value in position *n*. The reported value consists of (*) characters. The number of (*) characters reflects the number of characters in the actual key string. If the key string is empty, only <CRLF> is returned. The reply is followed by **I/OK**.
AT+iWKY*n*=?	Returns the message **“String”** followed by **I/OK**.

+iWST*n* — Wireless LAN Security Type Array

Syntax:	AT+iWSTn=<sec>
	Sets the Wireless LAN security type for each individual SSID in the array. This parameter takes effect following either a hardware or software reset (AT+iDOWN) only. A change to this parameter during iChip operation does not affect the current connection.
Parameters:	
n=0-9	Index of SSID
sec=0	no security.
sec=1	WEP 64.
sec=2	WEP 128.
sec=3	WPA-PSK with TKIP encryption.
sec=4	WPA2-PSK with TKIP or AES encryption.
sec=5	WPA-TKIP Enterprise with EAP-TLS or PEAP-MSCHAPv2. If CA parameter is empty, returns ERROR 222.
sec=6	WPA2-AES Enterprise with EAP-TLS or PEAP-MSCHAPv2. If CA parameter is empty, returns ERROR 222.
sec=7	EAP-MD5 and static WEP64
sec=8	EAP-MD5 and static WEP128
sec=105	WPA-TKIP Enterprise with EAP-TLS or PEAP-MSCHAPv2. RADIUS Certification Verification will be skipped.
sec=106	WPA2-AES Enterprise with EAP-TLS or PEAP-MSCHAPv2. RADIUS Certification Verification will be skipped.
Default:	0
Result code:	
I/OK	If sec is within limits provided that when sec equals 5 or 6 the CA parameter is defined.
I/ERROR	Otherwise
AT+iWST0?	Reports the current value followed by I/OK.
AT+iWST0=?	Returns the message "0-8,105,106", followed by I/OK.

***Note 1*:** In order to nullify CA, all WSTn parameters need to be set to values other than 5 or 6.

***Note 2*:** Configuring WSTn to 105 or 106 eliminates the verification of the RADIUS server's certificate. The user may decide to skip verification in favor of simplicity.

***Note 3*:** RADIUS authentication using EAP-TLS requires valid content in the CERT and PKEY parameters.

EAP-MD5 in Enterprise Mode

When configured for Enterprise mode security, iChip's WiFi driver supports the EAP-MD5 protocol for authentication in addition to PEAP-MSCHAP.

EAP-MD5-Challenge, which is described in RFC 2284, enables a RADIUS server to authenticate a connection request by verifying an MD5 hash of a user's password. The server sends a random challenge value to the client, and the client proves its identity by hashing the challenge and its password with MD5.

EAP-MD5 is considered as the 'lowest common denominator' EAP type for wireless LAN. It does not support use of per session WEP keys. Therefore, when using EAP-MD5 authentication protocol, the AP and the client should use static WEP keys.

Note that the WEP keys (WKY*n*) need to be configured accordingly.

+iEUSN — Domain and User name for WiFi Enterprise mode

Syntax: AT+iEUSN=*user*.

Sets the login user name to be used for WiFi Enterprise. This parameter takes effect following either a hardware or software reset (AT+iDOWN) only. A change to this parameter during iChip operation does not affect the current connection.

Parameters: *user* = login user name in the format: **Domain\Username**.

The Domain\Username shall be used in authentication process vs. the RADIUS server.

Domain indicates the RADIUS Domain name.

Username indicates a user that has permission to enter the RADIUS domain.

Length of *user* is limited to 64 characters.

Default: "" (empty)

Result code:

I/OK If *user* is an empty or a legal login User Name.

I/ERROR Otherwise

AT+iEUSN? Report the current login User Name. If the User Name does not exist, only <CRLF> will be returned. The reply is followed by I/OK.

AT+iEUSN=? Returns the message 'String'. The reply is followed by I/OK.

+iEPSW — Password for WiFi Enterprise mode

Syntax:	AT+iEPSW=*pass*
	Sets the password to be used for WiFi Enterprise. This parameter takes effect following either a hardware or software reset (AT+iDOWN) only. A change to this parameter during iChip operation does not affect the current connection.
Parameters:	*pass* = login password as defined in RADIUS server.
Default:	“” (empty)
Result code:	
I/OK	If pass is an empty or a legal password.
I/ERROR	Otherwise
AT+iEPSW?	Reports the current password. The reported value consists of ‘*‘ characters. The number of ‘*‘ characters reflects the number of characters in the password. If a password is not defined, only <CRLF> will be returned. The reply is followed by I/OK.
AT+iEPSW=?	Returns the message 'String‘. The reply is followed by I/OK.

+iBSID — Wireless LAN Basic Service Set Identifier

Syntax: AT+iBSID=<bssid>

Sets the BSSID value. In Infrastructure BSS networks, the BSSID is the MAC address of the AP and in Ad-Hoc network, the BSSID is generated randomly and is unique to a network.

Parameters: bssid = BSSID value in MAC address format.

- The BSID value should be set according to AT+iRP20 scan results.
- If BSID parameter is zero, iChip will associate to AP according to WLSI, Security type and signal strength.
- If BSID is non zero, regardless of WLSI parameter, iChip will scan only for AP with the specific BSSID. If the specific BSSID does not exist, iChip will scan forever.
- BSID is not multiple and correlates only to WLSI.
- If BSID parameter contains BSSID of an AP with WPA or WPA2 security type, WLSI and WLPP parameters MUST be set to appropriate values in order to compute correct PSK.
- The only way to zero BSID parameter (except Factory Default) is: At+iBSID=000000000000 (12 zeroes).

Default: 000000000000

Result code:

I/OK If *bsid* contains 12 hexadecimal characters.

I/ERROR Otherwise

AT+iBSID? Report the current BSSID value. If no BSSID value has been defined, the reply will be "000000000000".

The reply is followed by **I/OK.**

AT+iBSID=? Returns the message **'String'** followed by **I/OK**.

+iWPSP — Wireless LAN "Push-Button" Pin

Syntax:	AT+iWPSP=<*pin*>
	Defines which of iChip's general-purpose I/O pins (GPIO) will be used as a push-button for activating WPS.
Parameters:	*pin* = 0-6
Pin = 0	WPS mode is disabled.
pin = 1-6	Pins 0-5 of PIOC (general-purpose I/O pins group C)
Default:	0
Result code:	
I/OK	If *pin* is within limits.
I/ERROR	Otherwise
AT+iWPSP?	Reports the current value, followed by **I/OK.**
AT+iWPSP=?	Returns the message **'0-6'**, followed by **I/OK**.

Note: The command AT+iFD does **NOT** restore WPSP to its default value.

Note: The setting will take effect only after a SW or HW reset.

+iWLAS — Wireless LAN Ad-Hoc Scan Time

Syntax:	AT+iWLAS=<*n*> Sets the number of scans before creating an Ad-Hoc network. If WLSI = "!" or WLSI = "<AP name>", the +iWLAS parameter is ignored. .
Parameters:	*n* = 0-255 When *n*=0, iChip immediately creates its own Ad-Hoc network. If such a network already exists, it will eventually merge.
Default:	3
Result code:	
I/OK	If *n* is between 0-255.
I/ERROR	Otherwise
AT+iWLAS?	Reports the current value, followed by **I/OK.**
AT+iWLAS=?	Returns the message **'0-255'**, followed by **I/OK**.

+iWIAP — Enable Seeking Internet-Enabled AP

Syntax:	AT+iWIAP=<*n*>
	Sets iChip to seek for an Internet-Enabled AP. Requires SW reset to take affect
Parameters:	*n*=0 - 255
n=0	Disables seek mode.
n>0	Enables seek mode. *n* indicates the maximum time in seconds to wait for an IP assignment from a DHCP server.
Default:	*n*=0
Result Code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iWIAP?	Returns the current WIAP value followed by **I/OK**.
AT+iWIAP=?	Returns the message "**0-255**" followed by **I/OK**.

LAN to WiFi Bridge Mode Parameters

+iBRM — Bridge Mode

Syntax:	AT+iBRM=<n>
	Sets iChip LAN to WiFi bridge mode
Parameters:	n=0..4
Command Options:	
n=0	Bridge mode is disabled.
n=1	Cable replacement Ad-Hoc mode, Ethernet PHY to WiFi.
n=2	Cable replacement AP mode, Ethernet PHY to WiFi.
n=3	Cable replacement Ad-Hoc mode, MII/RMII to WiFi.
n=4	Cable replacement AP mode, MII/RMII to WiFi.
Result code:	
I/OK	If n is a legal value.
I/ERROR	Otherwise.

Note 1: To enter "Cable Replacement Ad-Hoc Mode", the following parameters should be set:

+iWLCH	-	Ad-Hoc wireless channel	
+iWLSI	-	Ad-Hoc network SSID (prefix with '!')	
+iWST0	-	WEP security type (64 or 128)	*- optional*
+iWKY0	-	WEP key (10 or 26 HEX characters)	*- optional*
+iBRM	-	LAN interface: MII/RMII or PHY	*- 1 or 3*
+iMACF	-	MAC Forward on both sides	*- optional*
+iHIF	-	Host Interface (non zero)	
+iBDRF	-	Fixed UART Baud Rate, if applicable	

Note 2: To enter "Cable Replacement AP Mode", for example with WPA-PSK encryption, the following parameters should be set:

+iWLSI	-	Network SSID	
+iWST0	-	Security type (3 or 4)	*- optional*
+iWPP0	-	WPA/WPA2 security key	*- optional*
+iBRM	-	LAN interface: MII/RMII or PHY	*- 2 or 4*
+iMACF	-	MAC address of the user application	*- optional*
+iHIF	-	Host Interface (non zero)	
+iBDRF	-	Fixed UART Baud Rate, if applicable	

Note 3: When BRM=3 or 4, the EMAC is fixed to 100Mbps full-duplex connection.

+iMACF — MAC Address Forwarding

Syntax:	AT+iMACF=<*mac*>
	In Ad-Hoc bridge mode, when BRM=1 or 3: Sets MAC forwarding in Ad-Hoc bridge mode. *mac* will be used as the destination MAC address of the packets transmitted over WiFi. If left empty Broadcast is used.
	In AP bridge mode, when BRM=2 or 4: Sets MAC filtering in AP bridge mode. Bridging of packets on the LAN will be restricted to/from this address only. Other packets shall be discarded.
Parameters:	*mac* = "hhhhhhhhhhhh" 12 HEX digits of MAC address
Command Options:	
mac=""	Empty. No MAC forwarding. In Ad-Hoc bridge mode, when BRM=1 or 3: Packets will be Broadcast. In AP bridge mode, when BRM=2 or 4: iChip will automatically save the source MAC address of the first packet on the LAN and use it as the MACF value. This MACF value will be erased following hardware reset or power-cycle.
mac=MAC-Address	A valid 12 HEX digit representation of a MAC address. If the MAC address is invalid, MAC forwarding will not work.
Default:	Empty
Result Code:	
I/OK	If *mac* is empty or not more than 12 characters long.
I/ERROR	Otherwise
AT+iMACF?	Reports the current MACF value followed by **I/OK**.
AT+iLVS=?	Returns the message "String" followed by **I/OK**.

SerialNET Mode Parameters

+iHSRV | +iHSR*n* — Host Server Name/IP

Syntax:	AT+i{HSRV | HSR*n}* = *server_name:port*
	Sets the host server-name or IP and port number to be used in SerialNET mode. Use *n*=0 or HSRV to define the primary server. Use *n*=1 or 2 to define secondary servers.
Parameters:	*n = 0 .. 2*
	server_name = A server name or IP address. Server names must be resolvable by the primary or alternate DNS.
	port = 0..65535
Command Options:	
server_name=''	Empty: No server name defined. Serial data transmitted from device in SerialNET mode will be ignored until a remote client accesses iChip.
server_name=<*server*>	*server* will be used in SerialNET mode to locate and establish a connection when serial data is transmitted from the device or when “Auto Link” SerialNET modes are defined. The server name may be any legal Internet server name, which can be resolved by the iChip’s DNS (Domain Name Server) settings. The server name may also be specified as an absolute IP address given in DOT form. If the primary server does not respond, iChip will try the secondary servers (if they are defined).
port=<*port number*>	It is assumed that the host server is "listening" on *port number*.
Default:	Empty
Result code:	
I/OK	If *server_name* is an empty or legal server name and *port* is within limits.
I/ERROR	Otherwise
AT+i{HSRV | HSRn}?	Reports the current host server and port as: <*server*>:<*port*>. If a server name does not exist, only <CRLF> will be returned.
	The reply is followed by I/OK.
AT+i{HSRV | }HSRn=?	Returns the message ‘Name/IP:Port’.
	The reply is followed by I/OK.

+iHSS — Assign Special Characters to Hosts

Syntax: AT+iHSS= <*control_characters*>

When iChip is connected to HSR*n* (where *n*=0..2) in SerialNet mode, and character <*Cm*> (where HSS=<*C0*><*C1*><*C2*>) arrives from the host, iChip will flush all characters received from host prior to <*Cm*>, close the socket to remote server HSR*n*, and open a socket to remote server HSR*m*. In the special case when *n*=*m*, iChip doesn't do anything. In any case, the control character will not be sent to remote server over the socket. iChip doesn't perform software reset, and stores all characters received from the host in MBTB (if defined). In addition, the SNRD parameter doesn't have any affect.

Parameters: *control_characters* = A string containing three control characters.

Command Options:

control_characters=" No control characters are defined. iChip will not respond to control characters to switch among HSRVs.

control_characters=<*string*> *string* is <C0><C1><C2>, where <C*m*> is an ASCII character or a binary escape sequence (or an empty character). A binary escape sequence is represented as \xhh (4 characters) where h is a hexadecimal digit 0..9 or A..F. For example: AT+iHSS="abc"

Default: Empty

Result code:

I/OK If *control_characters* is an empty or legal string.

I/ERROR Otherwise

AT+iHSS? Reports the current contents of the *control_characters*. If no *control_characters are defined*, only <CRLF> will be returned
The reply is followed by I/OK.

AT+iHSS=? Returns the message '**String**' followed by I/OK.

Example: **at+ihss=\x23\x24\x00**

When a number sign character '#' is received from host (ASCII 023 in hexadecimal notation), switch to primary remote server (HSR0). When a dollar sign '$' arrives, switch to HSR1. When a Null character arrives, switch to HSR2.

+iDSTR — Define Disconnection String for SerialNET Mode

Syntax: AT+i[!]DSTR:<*disconnect_string*>

Permanently sets SerialNET device disconnection string.

In a modem environment, iChip also goes offline following this event.

Parameters: *disconnect_string* = The string expected on a serial link to signal socket disconnection.

Command Options:

disconnect_string= '' Empty string – the connection will never be terminated due to a string arriving on serial link.

disconnect_string=<*string*> *string* received on serial link signals socket disconnection.

string consists any combination of printable ASCII characters and characters represented by two hexadecimal digits, such as: \xhh, where h is a hexadecimal digit 0..9 or A..F. Hexadecimal representation allows specifying non-printable characters.

! iChip will not send a DSTR to the socket upon detection. When this flag is not specified, iChip will send a DSTR each time it detects it.

Default: Empty

Result code:

I/OK If *disconnect_string* is an empty or legal string.

I/ERROR Otherwise

AT+iDSTR? Reports the current contents of the *disconnect_string* parameter. If the *disconnect_string* parameter is empty, only <CRLF> are returned. If the '!' flag is specified, the " *" string is appended to the report.

For example, the reply to a AT+iDSTR? command will be "&&& *" in case AT+i!DSTR=&&& was previously specified.

The reply is followed by I/OK.

AT+iDSTR=? Returns the message '**String**' followed by I/OK.

+iLPRT — SerialNET Device Listening Port

Syntax:	AT+iLPRT=*n*	Permanently sets the port number on which iChip will listen for client connections in SerialNET mode.
	Parameters:	*n* = 0-65535
	Default:	0 (no port).
	Result code:	
	I/OK	If *n* is within limits.
	I/ERROR	Otherwise
	AT+iLPRT?	Reports the current value of the SerialNET device listen port. The reply is followed by I/OK.
	AT+iLPRT=?	Returns the message "0-65535". The reply is followed by I/OK.

+iMBTB — Max Bytes To Buffer

Syntax: AT+iMBTB=*n* — Permanently sets max bytes to buffer while the iChip is establishing an Internet connection.

Parameters: *n* = number of bytes to buffer while establishing the connection in SerialNET mode.

Command Options:
n = 0 .. 2048

Default: 0 – No Buffering.

Result code:

I/OK — If *n* is within limits.
I/ERROR — Otherwise

AT+iMBTB? — Reports the current setting of max bytes to buffer. The reply is followed by I/OK.

AT+iMBTB=? — Returns the message "0-2048". The reply is followed by I/OK.

+iMTTF — Max Timeout to Socket Flush

Syntax: AT+iMTTF=*n* Sets max inactivity timeout before flushing the SerialNET socket.

Parameters: *n* = number of milliseconds of inactivity on serial link to signal socket flush in SerialNET mode.

Command Options:

n = 0 .. 65535

Default: 0 – No timeout.

Result code:

I/OK If *n* is within limits.

I/ERROR Otherwise.

AT+iMTTF? Reports the current timeout before SerialNET socket flush in milliseconds.
The reply is followed by I/OK.

AT+iMTTF=? Returns the message "0-65535".
The reply is followed by I/OK.

+iFCHR — Flush Character

Syntax: AT+iFCHR=*flush_chr* Permanently sets flush character in SerialNET mode.

Parameters: *flush_chr* = character received on serial link to signal socket flush in SerialNET mode.

Command Options:

flush_chr = ‘’ Empty: No Flush character defined. The SerialNET socket will not be flushed as a result of receiving a special flush character.

flush_chr = ‘a’ – ‘z’ | ‘A’ – ‘Z’ | ‘0’ – ‘9’ | <hex_char>

where,

<hex_char> = \x<hh>

<hh> = 00-FF

Default: Empty. No flush character defined.

Result code:

I/OK If *flush_chr* is empty or a legal character representation.

I/ERROR Otherwise.

AT+iFCHR? Reports the current flush character. The reply is followed by I/OK.

AT+iFCHR=? Returns the message ‘String’. The reply is followed by I/OK.

+iMCBF — Maximum Characters before Socket Flush

Syntax:	AT+iMCBF=*n*	Permanently sets max number of characters before flushing the SerialNET socket.

Parameters: *n* = maximum number of characters received on the serial link before flushing the SerialNET socket.

Command Options:
n = 0 .. 1460

Default: 0 – No specific limit. Flushing governed by Network.

Result code:

I/OK	If *n* is within limits.
I/ERROR	Otherwise.

AT+iMCBF?	Reports the current maximum number of characters before flushing the SerialNET socket. The reply is followed by I/OK.
AT+iMCBF=?	Returns the message "0-1460". The reply is followed by I/OK.

+iIATO — Inactivity Timeout

Syntax: AT+iIATO=*n*

Permanently sets maximum inactivity timeout in seconds to signal socket disconnection in SerialNET mode. When signaled, iChip will close the connected SerialNET communication socket. In a modem environment, the iChip will also go offline following this event.

When iChip is in iRouter mode and TUP< >2, if no activity is detected for the specified period, iChip will disconnect its modem side and go offline.

Parameters: *n* = number of seconds of inactivity, on a connected SerialNET socket, to signal socket disconnection. In iRouter mode, this number specifies a period of no activity on either the LAN/WiFi or modem/cellular side.

Command Options:

n = 0 .. 65535

When iChip is in Server SerialNET mode (LPRT defined) and it goes online in response to a triggering event: RING signal, MSEL signal pulled low or AT+I!SNMD -- timeout calculation commences only after the iChip opens the Listen port. When the Web server is activated (using AWS=1), an external reference to the Web server will restart the IATO timeout calculation.

Default: 0 – No timeout limit.

Result code:

I/OK If *n* is within limits.

I/ERROR Otherwise.

AT+iIATO? Reports the current inactivity timeout in seconds to signal socket disconnection in SerialNET mode. The reply is followed by I/OK.

AT+iIATO=? Returns the message "0-65535". The reply is followed by I/OK.

+iSNSI — SerialNET Device Serial Interface

Syntax: AT+iSNSI=*settings_str* Sets serial interface settings for SerialNET mode.

Parameters: *settings_str* = Serial link settings in SerialNET mode.

Command Options:

settings_str=”<*baud*>,<*data_bits*>,<*parity*>,<*stop_bits*>,<*flow*>”
where,
<*baud*> = 0..9 or h
<*data_bits*> = 7 | 8
<*parity*> = N | E | O
<*stop_bits*> = 1 | 1.5 | 2
<*flow*> = 0 | 1
-or-
settings_str= <*baud*>
where,
<*baud*> = 0..9 or h

The following table summarizes supported baud rates:

Baud Code	Baud Rate	Baud Code	Baud Rate
0	*See note, below*	6	19,200
1	600	7	38400
2	1200	8	57600
3	2400	9	115200
4	4800	h	230,400
5	9600		

Note: Baud Code ‘0’ means that host⇔iChip baud rate in SerialNET mode is determined according to the value of the BDRD parameter.

Default: “5,8,N,1,0” – baud rate 9600bps, 8 bits, No parity, 1 stop bit, no flow control.

Result code:
I/OK If *settings_str* is a valid serial link setting string.
I/ERROR Otherwise

AT+iSNSI? Reports the current serial settings string followed by **I/OK**.

AT+iSNSI=? Returns the message “**String**” followed by **I/OK**.

+iSTYP — SerialNET Device Socket Type

Syntax:	AT+iSTYP=*v*	Sets SerialNET socket type to *v*.
	Parameters:	*v* = 0, 1 or 2
	Command Options:	
		v=0 TCP
		v=1 UDP
		v=2 SSL3/TLS1
	Default:	0 (TCP)
	Result Code:	
	I/OK	if *v* = 0, 1 or 2
	I/ERROR	Otherwise
	AT+iSTYP?	Reports the current value of the SerialNET socket type followed by **I/OK**.
	AT+iSTYP=?	Returns the message **"0-2"** followed by **I/OK**.

Note: Setting STYP=2 for creating an SSL3/TLS1 TCP socket is applicable only for the server defined in the HSRV, HSR1 and HSR2 parameters. SSL related parameters should be set prior to entering SerialNET mode: CS, CA, CERT, PKEY. If a remote system opens a TCP socket to the LPRT port, a regular TCP (non SSL3) socket shall be maintained for that SerialNET session.

+iSNRD — SerialNET Device Re-Initialization Delay

Syntax: AT+iSNRD=*n* Sets SerialNET mode re-initialization delay in seconds.

Parameters: *n* = number of seconds iChip will pause before re-initializing SerialNET mode after a failed attempt to establish a socket connection to the peer or a connection related fatal error. A new SerialNET connection will only be attempted after SerialNET re-initializes. The SNRD delay will not be in effect as a result of an Escape Sequence ('+++').

Command Options:

n = 0 .. 3600

Default: 0 – No delay.

Result code:

I/OK If *n* is within limits.
I/ERROR Otherwise.

AT+iSNRD? Reports the current SerialNET re-initialization delay in seconds.
The reply is followed by I/OK.

AT+iSNRD=? Returns the message "0-3600".
The reply is followed by I/OK.

+iSPN — SerialNET Server Phone Number

Syntax: AT+i SPN=*number* — Permanently sets the SerialNET phone number to use to wake up a remote SerialNET server.

Parameters: *number* = Telephone number to use to dial up a remote SerialNET server in order to wake it up and activate its preprogrammed Ring-Response procedures. The SerialNET client will attempt RDL redials. During each dial-up attempt it will wait for SDT seconds before hanging up.

Command Options:

number = Telephone number string, composed of digits, ‘,’, '-', 'W', 'w', '*', '#', '!' or ' '. See description of the standard ATD command[1].

Default: Empty. Do not attempt to wake up a remote SerialNET server.

Result code:

I/OK If *number* is a legal phone number string.

I/ERROR Otherwise

AT+iSPN? — Reports the current SerialNET wakeup telephone number.
The reply is followed by I/OK.

AT+iSPN=? — Returns the message “Phone #".
The reply is followed by I/OK.

Note: If a character that is defined as a Delimiter is used within the dial string, the string must be entered between apostrophes.

+iSDT — SerialNET Dialup Timeout

Syntax: AT+iSDT=*n*

Permanently sets the SerialNET Dial timeout when waking up a remote SerialNET server.

Parameters: *n* = Number of seconds to allow after dialing up the remote SerialNET server, before hanging up.

Command Options:

n = 0..255 [seconds].

Default: 20 [seconds]

Result code:

I/OK If *n* is within limits.

I/ERROR Otherwise

AT+iSDT? Reports the current SerialNET dial timeout. The reply is followed by I/OK.

AT+iSDT=? Returns the message "0-255". The reply is followed by I/OK.

+iSWT — SerialNET Wake-Up Timeout

Syntax: AT+iSWT=*n* — Sets the SerialNET wake-up timeout when waking up a remote SerialNET server.

Parameters: *n* = Number of seconds to allow the entire SerialNET server wakeup procedure before hanging up and retrying.

Command Options:

n = 0..65535 [seconds]

Default: 600 [seconds].

Result code:

I/OK If *n* is within limits.

I/ERROR Otherwise

AT+iSWT? Reports the current SerialNET Wake-up timeout. The reply is followed by I/OK.

AT+iSWT=? Returns the message "0-65535". The reply is followed by I/OK.

+iSLED — SerialNET Indicator Signal

Syntax:	AT+iSLED=<*n*>
	Define a GPIO as the SerialNET signal.
	Active LOW when iChip is in SerialNET mode.
Parameters:	
n=0	No signal defined
n=1..6	Use PIOC [<*n*>-1] as the SerialNET signal
Default:	0 – no SerialNET signal used
Result code:	
I/OK	If *n* is a legal value.
I/ERROR	Otherwise
AT+iSLED?	Returns the current SLED value followed by **I/OK**.
AT+iSLED=?	Returns the message **“0-6”** followed by **I/OK**.

Note: The command AT+iFD does **NOT** restore SLED to its default value.

+iPTD — SerialNET Packets to Discard

Syntax: AT+iPTD=*n*

Sets the number of packets to be cyclically discarded in a SerialNET mode session. A packet is defined as the group of characters received on the serial link, meeting one (or more) of the socket flush conditions defined (+iFCHR, +iMTTF, +iMCBF).

Parameters: *n* = 0 – 65535

Default: 0 – No packet filtering. All data is transferred.

Result code:

I/OK If *n* is within limits.

I/ERROR Otherwise.

AT+iPTD? Reports the current value. The reply is followed by I/OK.

AT+iPTD=? Returns the message "1-65535". The reply is followed by I/OK.

Remote Firmware Update Parameters

+iUEN — Remote Firmware Update Flag

Syntax:	AT+iUEN=<*v*>
	Sets the remote firmware update flag.
Parameters:	*v* = 0 or 1
Command Options:	
v=0	Update only to a firmware version that is newer than the currently installed one.
v=1	Update to any firmware version available.
Default:	0
Result Code:	
I/OK	If *v* = 0 or 1
I/ERROR	Otherwise
AT+iUEN~*v*	Temporarily set the remote firmware update flag to *v* for the duration of the current session. The permanent value will be restored after completing the current session.
AT+iUEN?	Reports the current value of the remote firmware update flag followed by I/OK.
AT+iUEN=?	Returns the message "**0-1**" followed by I/OK.

+iUSRV — Remote Firmware Update Server Name

Syntax:	AT+iUSRV="<*protocol*>://<*host*>[:<*port*>]/[<*relative_path*>/]"
	Sets name of server to be used for updating iChip firmware remotely. This server must contain one or more firmware .imz files. The actual update process is initiated using the AT+iRFU command.
Parameters:	<*protocol*> = http or ftp
	<*host*> = Host name or IP address
	<*port*> = 1..65535 Optional port number. The well known ports 80 (http) or 21 (ftp) will be used in case *port* is not defined.
	<*relative_path*> = Path to a directory which contains one or more .imz files on the host or a path to a text file containing a list of one or more <CRLF>-separated .imz filenames. *relative_path* must be relative to the FTP home directory. If *relative_path* contains sub-directories, they can be divided using either '\' or '/'. *absolute_path* must end with '\' or '/'.
Command Options:	
AT+iUSRV="	Empty. No server name defined.
Default:	Empty. No dedicated remote firmware update server defined.
Result Code:	
I/OK	If *host* is an empty or legal host name.
I/ERROR	Otherwise
AT+iUSRV~ "<*protocol*>://<*host*>"	Temporarily set the firmware update server name to *host.* The permanent value will be restored after completing the next session.
AT+iUSRV?	Report the current firmware update server name. If a server name is not defined, only <CRLF> will be returned. The reply is followed by I/OK.
AT+iUSRV=?	Returns the message '**String / IP Addr**' followed by I/OK.
Example:	**`AT+iUSRV="ftp://172.20.101.5:21/RFU_CO2128/"`**

+iUUSR — Remote Firmware Update FTP User Name

Syntax:	AT+iUUSR=<*username*>
	Sets name of user to logon to the FTP server which is defined in the AT+iUSRV parameter.
Parameters:	<*username*> = Name of user to logon to the FTP server. This must be a registered user on the FTP server. Some servers allow anonymous login, in which case *username*=anonymous.
Command Options:	
AT+iUUSR="	Empty. No user name defined.
Default:	Empty. No user name defined.
Result Code:	
I/OK	If *username* is an empty or legal user name.
I/ERROR	Otherwise
AT+iUUSR~<*username*>	Temporarily set the user name to *username*. The permanent value will be restored after completing the next session.
AT+iUUSR?	Report the current user name. If a user name is not defined, only <CRLF> will be returned. The reply is followed by I/OK.
AT+iUUSR=?	Returns the message '**String**' followed by I/OK.

+iUPWD — Remote Firmware Update FTP User Password

Syntax:	AT+iUPWD=<*password*>
	Sets user password to logon to the FTP server defined in the AT+iUSRV parameter.
Parameters:	<*password*> = User password to logon to the FTP server. If special characters are used, the password should be specified within quotes. Servers that allow anonymous login usually request an Email address as a password.
Command Options:	
AT+iUPWD=“	Empty. No user password defined.
Default:	Empty. No user password defined.
Result Code:	
I/OK	If *password* is an empty or legal user password.
I/ERROR	Otherwise
AT+iUPWD~<*password*>	Temporarily set the user password to *password.* The permanent value will be restored after completing the next session.
AT+iUPWD?	Returns a string of asterisk (*) characters indicating the number of characters in the password. If a password is not defined, only <CRLF> will be returned. The reply is followed by I/OK.
AT+iUPWD=?	Returns the message ‘**String**’ followed by I/OK.

+iRPG — Remote Parameter Update

Syntax:	AT+iRPG=*GroupPass*
	Sets the remote parameter update group/password. Also used to authenticate a remote technician connecting for remote debug purposes..
Parameters:	*GroupPass* = Group/Password to be used for authentication when accepting iChip parameter updates from a remote web browser.
Command Options:	
GroupPass =''	No password. Remote update via the configuration website is effectively disabled.[1]
GroupPass =<grp-pass>	*grp-pass* will be used to authenticate the RPF file retrieved and restrict iChip parameter updates via a remote Web browser. [1]
GroupPass =""*	A password will not be used to authenticate the RPF file retrieved or remote updates via the Web. Effectively unrestricting any remote iChip parameter updates.
Default:	Empty. No Group/Password defined. When retrieving Email Parameter Update mails shall be skipped. iChip parameter updates via a remote browser are restricted. [2]
Result Code:	
I/OK	If *Group-pass* is an empty or legal Group/Password
I/ERROR	Otherwise
AT+iRPG~*GroupPass*	
AT+iRPG?	Reports the current Group/Password. If a Group/Password does not exist only <CRLF> will be returned. The reply is followed by I/OK.
AT+iRPG=?	Returns the message 'String'. The reply is followed by I/OK.

Note[1]: The remote update features which are protected by the RPG parameter are: update of parameter values, upload of new firmware and upload of a custom application website. When bit2 of the SDM parameter is set, viewing of the configuration website is also protected.
Note[2]: This default value is shipped from the factory. The AT+iFD command does *not* restore RPG to this value.

Secure Socket Protocol Parameters

+iCS — Define the SSL3/TLS Cipher Suite

Syntax:	AT+iCS=*n* Sets the cipher suite to be used in SSL3/TLS negotiations with a secure server. The default value '0' allows the server to choose from all supported cipher suites. When a specific value is specified, iChip requires the server to use that specific cipher.
Parameters:	*n* = A supported cipher suite code, as defined in RFC2246.
Command Options:	
n = 0	Set cipher suite to 'propose all'. When CS is set to 'propose all', iChip offers all supported cipher suites for SSL3/TLS negotiations. The server selects the most appropriate cipher suite during the handshake procedure.
n = 4	Set cipher suite to SSL_RSA_WITH_RC4_128_MD5
n = 5	Set cipher suite to SSL_RSA_WITH_RC4_128_SHA
n = 10	Set cipher suite to SSL_RSA_WITH_3DES_EDE_CBC_SHA
n = 47	Set cipher suite to TLS_RSA_WITH_AES_128_CBC_SHA
n = 53	Set cipher suite to TLS_RSA_WITH_AES_256_CBC_SHA
+1000	Add 1000 to any cipher suite to prohibit updating this parameter from the internal configuration website
Default:	0 (Propose All)
Result code:	
I/OK	If *n* is a supported cipher suite code
I/ERROR	Otherwise
AT+iCS?	Returns the current cipher suite value. The reply is followed by I/OK
AT+iCS=?	Returns the message "0,4,5,10,47,53". The reply is followed by I/OK

+iCA — Define SSL3/TLS Certificate Authority

Syntax:	AT+iCA[*n*]=*tca* Sets the certificates of the trusted certificate authorities. iChip accepts a server's identity only if its certificate is signed by one of these authorities.
Parameters:	*tca* = PEM format X509 certificate (DER format, Base-64 encoded with header and footer lines)
Command Options:	
Optional *n*	*n* is optional and may be 2, 3 or 4. Use *n* to specify alternative CA certificates (CA2, CA3 and CA4). When more than one CA certificate exists, iChip will check all its CAs when verifying a server's certificate.
tca =<CR><CR>	Empty: No trusted certificate authority.
tca =<*cert*>	*cert* is referenced as the trusted certificate authority's certificate during SSL3/TLS1 socket connection establishment (handshake). iChip establishes an SSL3/TLS1 socket connection only to servers having a certificate authenticated by this certificate authority. iChip expects *cert* to be multiple lines separated by <CR>, beginning with:`-----BEGIN CERTIFICATE-----` and terminating with: `-----END CERTIFICATE-----`. *cert* should include an RSA encryption public key of 1024 or 2048 bit. The signature algorithm may be MD2, MD5 or SHA1. Maximum size of *cert* is 1500 characters.
Default:	Empty. No trusted Certificate Authority defined.
Result code:	
I/OK	If *tca* is an empty or legal certificate.
I/ERROR	Otherwise
AT+iCA?	Reports the current trusted certificate's contents. The reported value displays the Certificate Authority's name, certificate validity date range, and the entire PEM contents. If the trusted certificate is empty, only <CRLF> is returned. The reply is followed by I/OK.
AT+iCA=?	Returns the message '**String**' followed by I/OK. Sample PEM format DER-encoded X509 certificate:


```
-----BEGIN CERTIFICATE-----
MIICPDCCAaUCEHC65B0Q2Sk0tjjKewPMur8wDQYJKoZIhvcNAQECBQAwXzELMAkG
A1UEBhMCVVMxFzAVBgNVBAoTDlZlcmlTaWduLCBJbmMuMTcwNQYDVQQLEy5DbGFz
cyAzIFB1YmxpYyBQcmltYXJ5IENlcnRpZmljYXRpb24gQXV0aG9yaXR5MB4XDTk2
MDEyOTAwMDAwMFoXDTI4MDgwMTIzNTk1OVowXzELMAkGA1UEBhMCVVMxFzAVBgNV
BAoTDlZlcmlTaWduLCBJbmMuMTcwNQYDVQQLEy5DbGFzcyAzIFB1YmxpYyBQcmlt
YXJ5IENlcnRpZmljYXRpb24gQXV0aG9yaXR5MIGfMA0GCSqGSIb3DQEBAQUAA4GN
ADCBiQKBgQDJXFme8huKARS0EN8EQNvjV69qRUCPhAwL0TPZ2RHP7gJYHyX3KqhE
BarsAx94f56TuZoAqiN91qyFomNFx3InzPRMxnVx0jnvT0Lwdd8KkMaOIG+YD/is
I19wKTakyYbnsZogy1Olhec9vn2a/iRFM9x2Fe0PonFkTGUugWhFpwIDAQABMA0G
CSqGSIb3DQEBAgUAA4GBALtMEivPLCYATxQT3ab7/AoRhIzzKBxnki98tsX63/Do
lbwdj2wsqFHMc9ikwFPwTtYmwHYBV4GSXiHx0bH/59AhWM1pF+NEHJwZRDmJXNyc
AA9WjQKZ7aKQRUzkuxCkPfAyAw7xzvjoyVGM5mKf5p/AfbdynMk2OmufTqj/ZA1k
-----END CERTIFICATE-----
```

+iCERT — Define SSL3/TLS1 Certificate

Syntax:	AT+iCERT=*ct* Set iChip's SSL3/TLS1 certificate. Some SSL3/TLS1 servers require the client side to authenticate its identity by requesting the client to provide a certificate during the SSL socket negotiation phase. This is called "client side authentication". If the CERT parameter contains a certificate, iChip provides it to the server upon request. iChip also needs a private key (see PKEY parameter) in order to encrypt its certificate before sending it to the server. In addition, the certificate should be signed by a certificate authority accepted by the server for the client side authentication to succeed.
Parameters:	*ct* = PEM format (DER format, Base-64 encoded with header and footer lines)
Command Options:	
ct = <CR><CR>	Empty. No trusted certificate authority.

ct =<*cert*>	*cert* is used as iChip's certificate during client side authentication. The certificate must be signed by a certificate authority acceptable by the server.
	iChip expects *cert* to be multiple lines separated by <CR>, beginning with
	`-----BEGIN CERTIFICATE-----`
	and terminating with
	`-----END CERTIFICATE-----.`

Default:	Empty. No trusted certificate authority defined.
Result code:	
I/OK	If *ct* is an empty or legal certificate.
I/ERROR	Otherwise
AT+iCERT?	Displays current certificate contents. If the trusted certificate is empty, only <CRLF> is returned, followed by I/OK.
AT+iCERT=?	Returns the message '**String**' followed by I/OK.

+iPKEY — Define iChip's Private Key

Syntax: AT+iPKEY=*pky*
Set iChip's private key.
The private key is required to perform an RSA encryption of its certificate (see CERT parameter) when performing client side authentication. Special care should be taken to protect private key contents from unauthorized parties. For this reason, once the private key is stored on iChip, it cannot be read – only erased or overwritten.

Parameters: *pky* = PEM format

Command Options:

pky =<CR><CR> Empty. Any existing private key is erased.

pky =<*pkey*> *pkey* is used as iChip's private key to RSA encrypt its certificate during client side authentication.
iChip expects *pkey* to be in PKCS#1 format, which appears as multiple lines separated by <CR>, beginning with
`-----BEGIN RSA PRIVATE KEY-----`
and terminating with
`-----END RSA PRIVATE KEY-----`

Default: Empty. No private key defined.

Result code:

I/OK If *pky* is an empty or legal private key.

I/ERROR Otherwise

AT+iPKEY? Reports the current private key's strength (number of bits in key). If the key is empty, only <CRLF> is returned. There is no way to retrieve *pkey* contents. The reply is followed by I/OK.

AT+iPKEY=? Returns the message '**String**' followed by I/OK.

Example:


```
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQCoMGVcZ3HNFB/cRfWP7vdZrRK+YB+lez07mAN6Zcd4C19Xi6M6
dmewb6qQ6TRYC1gBhJ+KtMopGoqQ3v1VSu0Ve/ZrjWNxLN9UAtRMubtkGz2j6OCt
lx4WsFUWebF8QEEm9+3coMnRqtAdluYEU2F2PTeWUsQfjRQQmBjus/y0wwIDAQAB
AoGBAKWaKWOHk1zbENfhpn1XTQNmT4tVuDNHGi6gaeRNbM79W54mpsy8ozHtcWOH
y3tZiAjOngyEIH3CXWdxuL0PrkmdSk39+V0EIuA0sRxyUTb3/LlDU9DpxlYXBYK5
Kclq2qH5GBv28QJChG6/dfvuO8a1JyPwD61iOvBvBye/C7QRAkEA1uU7pT8ejcxf
ZLwaBwUift9Y1kpzrdHYnqJggrhGeZq4bIb8ioOFEgB+JKXSxaQZgxUsIkDVzkO/
+J/H8KZKywJBAMhcGEftwPqtZMWyqis7rSUpsewaxg79QYDZVSRwi5ynLqtqui4d
GVSftbXvtZHRs8uyp3plTFUVFnvPRsUJpukCQEZyJzdola+OS8dOEooymLhWp1y4
U2ur2wNF37V6iz/aBJMvPSJ7MuhP2QpSgeHghax/CFTCRFS1yPzMBFNTcDkCQEHq
ko5veNK/4uxruDJbAr68Ne3gbRKXXUp/tdQ0NqpGEkOQ7EmphyDhHk4J2+lqXUWB
tDm/Q9qmAmyfJ8BBSakCQAaOl0MGdUnyFuanp19jRfLB29oOqMQqyV90r25AxOcN
HD8Jsmn5vBYm4wdtR8x84Gh7128RfuBS8J0hFb90yRY=
-----END RSA PRIVATE KEY------
```

DHCP Server Parameters

+iDPSZ — DHCP Server Pool Size

Syntax:	AT+iDPSZ=<*range*>
	Sets number of addresses to be allocated in the IP pool of iChip's DHCP server.
Parameters:	*range* = number of IP addresses in pool
Command Options:	
range=0-255	When *range*=0 the pool is empty and the DHCP server is inactive. When *range* is set to any number between 1 and 255, *and* the DIP parameter is defined – the DHCP server becomes active.
Default:	0 — DHCP server is inactive
Result Code:	
I/OK	If *range* is an integer between 0 and 255.
I/ERROR	Otherwise
AT+iDPSZ?	Reports the current *range* value followed by **I/OK**.
AT+iDPSZ=?	Returns the message '**0-255**' followed by **I/OK**.

+iDSLT — DHCP Server Lease Time

Syntax:	AT+iDSLT=<*time*>
	Defines lease time, in minutes, to be granted by iChip's DHCP server when assigning IP addresses to clients.
Parameters:	*time* = lease time in minutes
Scope:	
Command Options:	
time=0-65535	When *time*=0 lease time is indefinite. Any other value sets a limit on the lease time.
Default:	0 — Indefinite lease time
Result Code:	
I/OK	If *time* is an integer between 0 and 65535.
I/ERROR	Otherwise
AT+iDSLT?	Reports the current *time* value followed by **I/OK**.
AT+iDSLT=?	Returns the message '**0-65535**' followed by **I/OK**.

iRouter Parameters

+iARS — Automatic Router Start

Syntax:	AT+iARS=*n*
	Causes iChip to automatically enter iRouter mode upon power-up or soft reset.
	Upon entering iRouter mode, iChip immediately goes online on the dial-up/cellular side. Packets are not buffered during establishment of the dial-up/cellular connection. After establishing the connection, iChip starts the routing service.
Parameters:	
n=0	Do not start iRouter mode upon power-up or soft reset.
n=1	Enter iRouter mode upon power-up or soft reset.
Default:	0
Result Code:	
I/OK	If *n* is within limits
I/ERROR	Otherwise
AT+iARS?	Reports the current value followed by **I/OK**.
AT+iARS=?	Returns the message "0, 1" followed by **I/OK**.

+iPFW — Port Forwarding Rules

Syntax: AT+iPFW*n*=”[L | M]<*w-port*>,<*l-IP:l-port*>[,<*type*>]”
Set Port Forwarding rule number n.
The Port-Forward rule must be enclosed in double-quotes.

Parameters:

n — Index in the range 0..9

[L | M]*w-port* — *w-port* = 0..65535
The port on the Public-IP (WAN), which will be forwarded to an internal IP-Port.
The optional modifiers ‘L’ or ‘M’ may be added to imply whether *w-port* resides on the Modem or on the LAN/WiFi network side. When not specified, the Modem port will be chosen as default.

l-IP: l-port — The local IP and Port to which packets are forwarded from *w-port*. The IP may be on the LAN/WiFi or on a PPP connection on the Host interface, as defined by the HIF parameter.
l-port = 0..65535

type — Optional socket type modifier:
0 – TCP port
1 - UDP port
Not specified – Both socket types

Default: “” (empty)

Result code:

I/OK — If *PFWn* contains a legal value.

I/ERROR — Otherwise

AT+iPFWn? — Reports the current PFW value at index n in the format:
w-port,l-IP:l-port[,type] or
[L | M] *w-port,l-IP:l-port[,type]*
If no value has been defined, only <CRLF> is returned.
The reply is followed by **I/OK**.

AT+iPFWn=? — Returns the message **'String'** followed by **I/OK**.

Note 1: Newly assigned Port Forwarding rules take effect only after recycling power to the iChip or executing a soft-reset (AT+iDOWN).
Note 2: The Port Forwarding rules should not include two or more rules with the same local IP:Port.
Note 3: The Port Forwarding rules should not include two or more rules with the same Network and Map-port ([L | M]<*w-port*>).

40 Appendix A

MIME content types and subtypes

Type	Subtype
text	plain
	richtext
	enriched
	tab-seperated-values
	html
	sgml
	vnd.latex-z
	vnd.fmi.flexstor
multipart	mixed
	alternative
	digest
	parallel
	appledouble
	header-set
	Form-data
	related
	report
	voice-message
	signed
	encrypted
message	rfc822
	partial
	external-body
	news
	http

Type	Subtype	Subtype
application	octet-stream	vnd.ms-works
	postscript	vnd.music-niff
	oda	vnd.ms-artgalry
	atomicmail	vnd.truedoc
	andrew-inset	vnd.koan
	slate	vnd.street-stream
	wita	vnd.fdf
	dec-dx	set-payment-initiation
	dca-rft	set_payment
	activemessage	set-registration-initiation
	rtf	set-registration
	applefile	vnd.seemail
	mac-binhex40	vnd.businessobjects
	news-message-id	vnd.meridian-slingshot
	news-transmission	vnd.xara
	wordperfect5.1	sgml-open-catalog
	pdf	vnd.rapid
	zip	vnd.enliven
	macwriteii	vnd.japannet-registration-wakeup
	msword	vnd.japannet-verification-wakeup
	remote-printing	vnd.japannet-payment-wakeup
	mathematica	vnd.japannet-directory-service
	cybercash	vnd.intertrust.digibox
	commonground	vnd.intertrust.nncp
	iges	vnd.ms-tnef
	riscos	vnd.svd
	eshop	
	x400-bp	
	sgml	
	cals-1840	
	pgp-encrypted	
	pgp-signature	
	pgp-keys	
	vnd.framemaker	
	vnd.mif	
	vnd.ms-excel	
	vnd.ms-powerpoint	
	vnd.ms-project	

Type	Subtype
image	jpeg
	gif
	ief
	g3fax
	tiff
	cgm
	naplps
	vnd.dwg
	vnd.svf
	vnd.dxf
	png
	vnd.fpx
	vnd.net-fpx
audio	basic
	32kadpcm
	vnd.qcelp
video	mpeg
	quicktime
	vnd.vivo
	vnd.motorola.video
	vnd.motorola.videop

Table 40.1 MIME Content Types and Subtypes

41 Appendix B

Sample Parameter Update File

RP_DEST="00010001" RP_GROUP="111"
RP_START_FROM_FACTORY_DEFAULTS=YES

 # MODEM PARAMETERS:
MIS="ATX4E1&C1&D2M2L2"
XRC="1"
BDRM="8"

 # CONNECTION PARAMETERS:
ISP1="7777555"
ISP2="036666555"
USRN="name"
PWD="pass"
DNS1="192.115.106.10"
DNS2="192.115.106.11"
ATH="1"
SMTP="smtp.com"

 # POP3 PARAMETERS:
MBX="pop_name"
MPWD="pop_pass"
POP3="pop3.com"
LVS="0"
FLS="mymail"

 # EMAIL STRUCTURE_PARAMETERS:
TOA="someone@hisServer.com"
TO="name"
CC1=" cc1@address.com"
CC2="cc2@address.com"
CC3="cc3@address.com"
CC4="cc4@address.com"
REA="myEmail@myServer.com"
FRM="me"
SBJ="MySubject"
BDY=This is my Email
.

MT="0"
MST="text-plain"
FN="myfile.txt"

 # CONNECTION TIMEOUT/RETRIES PARAMETERS:
RDL="2"
RTO="180"
WTC="100"

 # OTHER PARAMETERS:
HDL="5"
URL="http://www.connectone.com/

42 Appendix C

NIST Time Servers

Server	IP	Address Location
nist1.aol-ca.truetime.com	207.200.81.113	TrueTime, AOL facility, Sunnyvale, California
nist1.aol-va.truetime.com	205.188.185.33	TrueTime, AOL facility, Virginia
nist1.datum.com	66.243.43.21	Datum, San Jose, California
nist1.datum.com	209.0.72.7	Datum, San Jose, California
nist1.dc.certifiedtime.com	216.200.93.8	Abovnet, Virginia
nist1.nyc.certifiedtime.com	208.184.49.9	Abovnet, New York City
nist1.sjc.certifiedtime.com	208.185.146.41	Abovnet, San Jose, California
nist1-dc.glassey.com	216.200.93.8	Abovenet, Virginia
nist1-ny.glassey.com	208.184.49.9	Abovenet, New York City
nist1-sj.glassey.com	207.126.98.204	Abovenet, San Jose, California
time.nist.gov	192.43.244.18	NCAR, Boulder, Colorado
time-a.nist.gov	129.6.15.28	NIST, Gaithersburg, Maryland
time-a.timefreq.bldrdoc.gov	132.163.4.101	NIST, Boulder, Colorado
time-b.nist.gov	129.6.15.29	NIST, Gaithersburg, Maryland
time-b.timefreq.bldrdoc.gov	132.163.4.102	NIST, Boulder, Colorado
time-c.timefreq.bldrdoc.gov	132.163.4.103	NIST, Boulder, Colorado
time-nw.nist.gov	131.107.1.10	Microsoft, Redmond, Washington
utcnist.colorado.edu	128.138.140.44	University of Colorado, Boulder

Table 42.1: List of NIST Time Servers

Note: Check http://tf.nist.gov/tf-cgi/servers.cgi# for updates

43 Appendix D: SPI Host Interface

Introduction

The iChip CO2128 \ CO2144 contains an SPI slave port, which allows a Host processor to interface the iChip using an SPI Master port. The +iHIF parameter defines whether iChip monitors its SPI for commands from the Host.

The SPI data transfer is based on the 'Command-Response' principle (Half Duplex). Meaning, until the Host receives an answer to a command, it must not send a new one.

Differences in the AT+i protocol from UART and USB:

- No echo from the iChip to Host (i.e. when iChip's host interface is set to SPI, the command AT+iEn is meaningless).
- When iChip's host interface is set to SPI, iChip doesn't support SerialNET mode since this mode is not Half Duplex compatible.
- When iChip's host interface is set to SPI, iChip doesn't support the "+++" Escape sequence.

SPI Protocol

SPI on the iChip implements the following behavior:

- SPI number of bits per transfer is: 8.
- Fixed peripheral select.
- The CS (Chip Select) is directly connected to the SPI Master device.
- Mode fault detection is enabled.
- The inactive state value of the serial clock is logic level zero (LOW).
- Data is changed on the leading edge of the serial clock and captured on the following edge of the serial clock.
- Polarity and Phase are defined as follows: CPOL=0 and CPHA=1, and may be changed by setting the parameter +iSPIP.
- The CS line is deactivated as soon as the last transfer is achieved.

An iChip GPIO Output signal is dedicated as the SPI Control Signal. The SPI Control Signal pin is defined by the +iSPIP parameter. This signal is also referred to as SPI1_INT. After processing a command from the Host, iChip asserts this signal HIGH to indicate that the response is ready to be read. This signal is also utilized as a flow-control signal when the Host transmits data to the iChip.

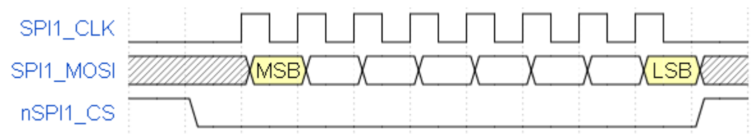


Figure 43-1: SPI write transaction

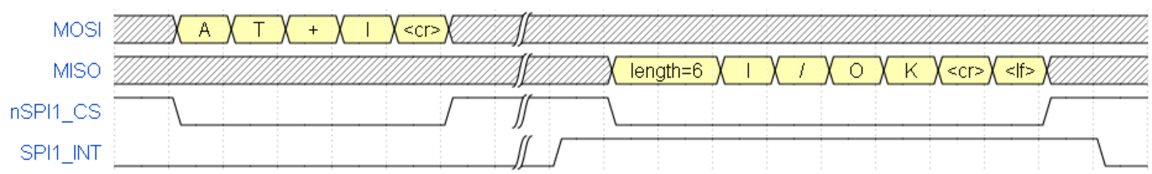


Figure 43-2: Zoom out view on SPI command and response

Reading from iChip

After processing a command from the Host, iChip prepares a response according to the AT+i protocol. The Host should follow these steps:

1. Wait for the SPI1_INT to be asserted HIGH by the iChip.
2. Read the 2-byte header. It will be constructed as follows:
   - Bit 15 is set
   - Bits 14-12 are zero
   - Lower 12 bits contain length of the payload in bytes.
3. Read the payload of <length> bytes from iChip.
4. Wait for the SPI1_INT to be asserted LOW.
5. iChip's response may be returned in more than one fragment. If the payload does not include a complete response, ending with CR character (ASCII code 0x0D), then the Host should follow steps 1 through 4 again to continue reading.

The Host should not attempt to send the next command before it has read the complete response from iChip, which is pre-defined in the AT+i protocol per each command and can include multiple lines.

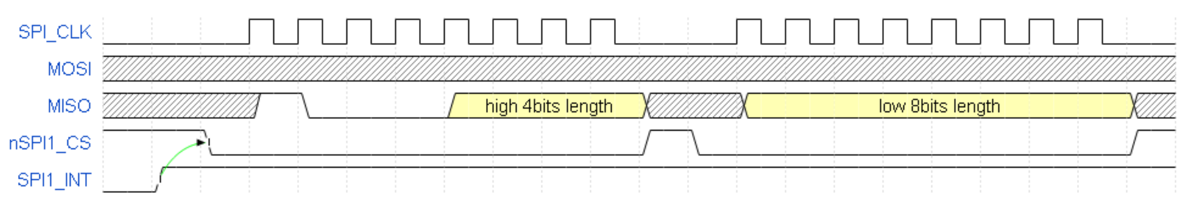


Figure 43-3: Header Prefix when reading from SPI

Once the length of the response is known, the host CPU should read the amount of bytes specified in the header, following this, the iChip will negate the SPI1_INT signal indicating the readiness to accept a new AT+I command.
Please note: as stated above, iChip can send it's response in chunks, the host CPU should check that the last received character is CR, if not - the host CPU should expect

additional chunks in response. Also please note that some commands return multiple lines in response, see example below:

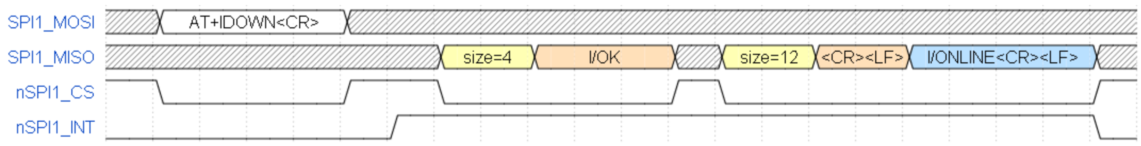


Figure 43-4: multiple chunks response example

Flow Control when writing to iChip

When the Host transfers data to iChip, that contain streams of more than 256 bytes at a time, flow control from iChip to the Host is managed using the SPI Control Signal. iChip asserts SPI1_INT to indicate flow control WAIT. As soon as the Host identifies that the SPI1_INT was asserted it should stop transmitting and disable the CS and the SPI clock. At SPI clock of 12MHz the Host's response time should be less than 1.3 msec in order to avoid buffer overflow in iChip. When iChip can resume reception, it will de-assert the SPI Control Signal, indicating flow control CONTINUE. At the end of a data transfer, when the host should wait for the response from iChip, iChip might actually raise the SPI_INT as a flow control and not as a response indicator. As a result the host might get confused and try to read an invalid data from iChip. At these situations, the host is advised to sample the SPI_INT for 5 – 10 ms. If it goes low, then it was a flow control, if it stays HIGH then it is DATA_READY signal and the host should initiate a read sequence.

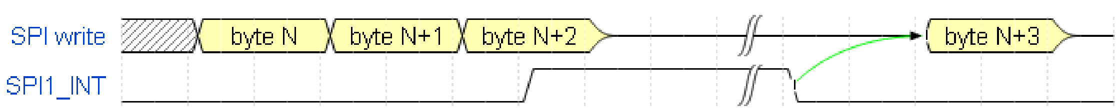


Figure 43-5: SPI HW flow control during write

Initialization of the SPI related Signals

- On power up, the SPI Control Signal is internally pulled-up with resistance of 15K Ohm. After 1 second it is initialized and asserted LOW, signaling readiness of iChip for AT+i commands.
- The MISO line is in TRI-STATE when the CS is HIGH (not selected). This allows connecting additional slave devices to the same SPI.
- The SPI clock must be LOW when the CS is asserted LOW (activated).

44 Appendix E: RS-485 Host Interface

Introduction

The iChip CO2128 \ CO2144 supports full-duplex and half-duplex, RS-485 connections in point-to-point configuration. Multi-drop configuration is not supported.

RS-485 Half Duplex

Half-duplex, 2-Wire RS-485 is based on using the same 2 wires for transmit and receive. The potential interference between transmit and receive modes is resolved by toggling the wire use in the time-space. The RTS H/W flow control signal is used by the UART to signal the wire direction and can thus be used to gate the send/receive RS485 buffers.

The following block diagram depicts this:

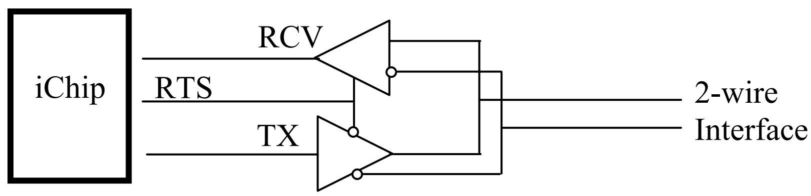


Figure 44-1: RS-485 Half-Duplex Diagram

Note that the 2-wire interface is connected to both the receive and transmit buffers. The RTS signal is used to gate the buffers so only one path (TX or RCV) is active at one time.

When iChip is configured for half-duplex RS-485, the iChip ECHO is automatically turned off, regardless of the AT+iEn command.

Half-duplex, 2-wire, RS-485 setting is also available in SerialNET, however, if full-duplex exchange shall be attempted, data shall be lost.

The iChip is configured for half-duplex, 2-wire, RS-485 mode via the +iHIF parameter. A value of 101 defines the RS-485 interface on iChip's USART0 while a value of 102 defines this interface on iChip's USART 1.

RS-485 Full Duplex

Full-duplex, 4-Wire RS-485 (similar to RS-422) can be connected to a UART port of the iChip via a conversion circuit. The +HIF parameter should be set to 1, 2 or 3 according to the UART on the iChip. The conversion circuit is described in the following diagram:

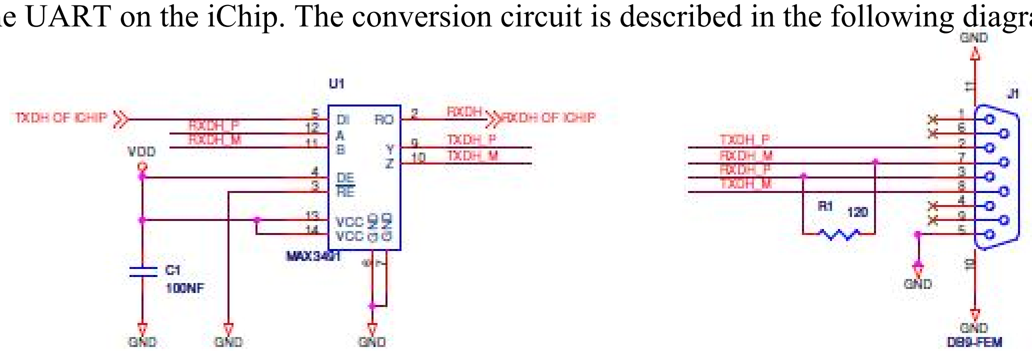


Figure 44-2: RS-485 Full-Duplex Diagram

Index