iControl Incorporated

iCHIME Command List

Revision 2.2

Date: January 2015

Command List Index

iTAG Command List

iTAG Operation Description.

The iTAG is configured through the 802.15.4 radio interface for network management, trip set-up, and checkout. The factory default radio channel is 802.15.4 channel 21. All commands for controlling the radio are identified in the iTAG Command List.
Once a User as completed the initial iTAG configuration, complete control and operation of the iTAG can be accomplished through the radio link interface using commands available in the iTAG Command List.
The radio in the iTAG is nominally operated in a power saving mode with the receiver periodically enabled and "listening" for commands or data requests. A key element to the power saving mode is the principle of "tag-talks-last". In this mode, battery power is conserved by only responding to messages that are sent directly to the specific iTAG MAC address or mutual broadcast address. For a majority of iTAG operation, the iTAG is in a deep power saving mode (~98% of the time). The iTAG will only transmit when it receives a properly encrypted message during its receiver active period. The encrypted message directed to the iTAG may include information about which channel the iTAG should use for further transmissions. Once, the communication parameters are exchanged, there will typically be a period a several seconds of active radio transmission as the iTAG relays data. The data transmitted may either be security data destined for government servers, and/or commercial data destined for the end customer. This data may originate from internal flash storage or it may be received and relayed from another compatible device.

Command Formats:

To enhance speed, minimize memory, and reduce network traffic, iTAG's use binary codes (command ID's) to represent iTAG functions. Command codes eliminate the need to perform string comparisons to determine a particular operation. When encrypted, command codes are also harder to break since "named" commands are identifiable patterns.

iTAG User interfaces, (iControl's iView and other applications) convert iTAG command ID's to "named" strings to aid in interpreting iTAG commands. Each command listed in this document includes the iVIEW command name (ASCII string), the hexadecimal command ID, and the decimal equivalent command used by PC terminal emulators. These command descriptions are the minimum information required to configure and operate an iTAG using the terminal emulator available on most PC's (HyperTerminal).

iTAG's may be commanded in either binary or ASCII mode. User interfaces with **direct serial** interfaces to the iTAG use ASCII Decimal commanding. All commands and telemetry transmitted over an iTAG network are commanded in binary.

The iTAG OS provides multiple levels of security. The User is responsible for determining the appropriate level protection when setting up an iTAG network.

Command Types:

destination **cmd**

ASCII Decimal: Z *0 0 0 0* 1 2 3 4 **97 1 1**(ENTER)

ASCII decimal commands are the simplest to use when getting started. ASCII decimal commands may be sent from any terminal emulator. No synchronization, time code, or CRC is required.

Alternate ASCII: r **97 1 1**(ENTER)

A command may also be entered using the r identifier. A leading 'r' or 'R' indicates that the command is for the connected iTAG which is most often the case when connected to the serial port of that iTAG.

Binary:

Binary commands require synchronization patterns, length, time code pads, and a CRC. Binary commands may be encrypted.

ASCII Name: **Run_Macro**<parameter1><parameter2> etc

iControl's iView provides command code translation to ASCII names. Any translation from command code to ASCII name may be programmed using a customer provided application. iView provides the necessary encryption of the final command format.

ASCII Commands

The ASCII decimal format is the easiest to implement. ASCII commands do not require (or support) a Cyclical Redundancy Code (CRC), Time Code pad, or encryption. Users may deploy an iTAG using only ASCII commanding.

The only security feature available with ASCII commanding is password protection.

To send an ASCII command, a command file may be generated from any text editor, and transferred to an iTAG using the "Transfer Text File" option in HyperTerminal. Each ASCII command from a PC to an iTAG must use the "Z" or "R" command format. The "Z" prefix signals the iTAG to send an ASCII formatted command to the indicated address. All commands included in this list may be sent in ASCII mode with an "Z" or "R" at the start of the command, "Space" as character delimiter, and a carriage return at the end.

Examples:

The following commands may be sent (using HyperTerminal) to an iTAG whose 8 byte ID is Destination". The Destination Address is the first 8 bytes, then command, and a carriage return. The Destination Address must match the **iTAG_ID** for which the command is intended. An iGATE will automatically forward commands it receives with different destination iTAG_ID's.

Z 0 0 0 0 0 1 2 255 1 5 5 Destination(0.0.0.0.0.1.2.255), Command (1 5 5)

Same command for the iTAG that we are directly connected to

R 1 5 5

Invalid commands,

1 2 3 4 5 5 6 (need 'Z' and command is too short)

Z 0 0 0 0 1 2 3 (too few characters, need full destination and command)

Z12345678123 (need spaces between each character)

Z 0 255 1000 1 1 2 3 4 1 2 (All numbers represent 1 byte, no number larger than 255)

ASCII Command Files:

iTAG ASCII commands may be grouped in a file and loaded all at once with a terminal emulator or other PC application. The command files may be built using common text editors such as Notepad, Word, etc.

iTAG command files may include comment fields and up to 1k bytes of commands to be sent. All characters following a "//" are considered comments and are ignored by the iTAG processing functions.

The only iTAG serial ports which are enabled for ASCII commanding are User Port, Modem Port, or DataPort. If password protection is enabled for ASCII commanding, include a **Password** command at the beginning of the command file.

ASCII Command File Examples:

The following text may be copied and pasted into a text (.txt) file. If HyperTerminal (or other application) sends this file to an iTAG (via modem, User Port etc) the iTAG will process all commands and commit them to flash.

Example1.txt

```
//  This File is an iTAG command file.
//  To load, send file using any terminal emulator with a "Transfer Text" Option
//  Send file once. The iTAG will begin running its schedule as long as schedule
//  checking is enabled.
//
        R 253 0 255 255 255 255 23 30 0 97 6 1      // Run Macro 6 at 11:30:00 PM
R 253 1 255 255 255 255 4 15 0 97 0 1       // Run Macro 0 at 4:15:00 AM
R 253 2 255 255 255 255 19 15 97 0 1        // Run Macro 0 at 7:15:00 PM
R 99 0 16 2 18 0 2 17 1 4 100 0 2 120 2 17 0 1 20    // Load Macro 0
R 99 6 5 2 18 1 1 29                        // Load Macro 6
R 240                                       // Commit to flash
```

Example2.txt

```
// This File is an iTAG command file.
// To load , send file using any terminal emulator with a "Transfer Text" Option
//  Send file once.
//  File loads Macro with Relay cycling Macro. Turns all relays off, delays 0.100 seconds,
//  Turns all relays on
//
        R 99 0 15 4 8 2 0 0 4 101 0 0 1 4 8 2 255 255// Load Macro 0 w/Relay cmds
        R 240                                       // Commit to flash
```

To run Example2, send the command; R 97 0 1 (Run_Macro[0] once)

Binary Commands

Binary commands are sent between iTAG's and between an iGATE and iTAG using the radio interface. As such, this document does not cover binary commanding. This document covers only user initiated commands.

Power Considerations

As a remote asset, an iTAG will generally be powered by a battery with no means or very minor means of being recharged. As such, operating in power savings mode is an important concept.

The present iTAG firmware implements a power savings mode in which the iTAG will sleep for most of each second (see commands 12, 15, and 16). During the remaining fraction, the iTAG may come out of sleep by one of three actions: 1) It is time for a scheduled command to be executed, 2) A radio packet addressed for this iTAG is received, or 3) There is user port activity on the serial port.

When a scheduled command is executed, it may or may not cause the iTAG to come out of power savings mode.

Each command is that will affect the sleep mode is noted in the section for each command.

In general, the iTAG will not sleep when a macro is running.

When the iTAG wakes up from sleep, it will remain awake for the number of seconds indicated by command 16.

iControl Products

There are a variety of iControl products. Each of these products implements the same basic command set.

Due to functionality and processor based resource constraints, some products may or may not implement certain functions.

Each command ID is identified as being implemented by each product by indicating one or more of the following abbreviations.

Product Name	Abbreviation
iTAG	**GEN2**
3rd Generation iTAG	**GEN3**
iGATE	**GATE**
mLOCK	**LOCK**
miKEY	**KEY**
iCHIME	**CHIME**
iGATE Repeater	**RPT**

iTAG_Mode: CMD_ID=0x01 (decimal 1)

Format: r 1 <iTAG_Mode> [Param]

iTAG_Mode sets the data reporting function on the iTAG. Additional functionality may also be accomplished by appending **Run_Macro** commands to the end of the **iTAG_Mode** command. If a User wishes to execute additional commands while the iTAG is in a broadcast mode, other iTAG functions may be linked to the mode by pointing to a Macro at the end of the iTAG mode command. While the iTAG mode is 0x00 (0) the iTAG is not broadcasting data over the Local Area Network (LAN).

Definitions:

			<Param>
<iTAG_Mode>	0x00	Silent	N/A
	0x01	One Sample	N/A
	0x02	Stream (1Hz):	N/A
	0x05	Stream "N" Samples (Once per second)	(0<"N"<255)
	0x06	Stream Samples (For "N" minutes)	(0<"N"<255)
	0x0a	Send record to default destination	P1: IO MSB Address P2: IO LSB Address P3: Xmit Port
	0x0b	Send record to destination	P1: IO MSB Address P2: IO LSB Address P3: Xmit Port P4-P7: Destination Address
	0x0c	Send record using repeater channel **(RPT)**	P1: IO MSB Address P2: IO LSB Address P3: don't care P5: 0, use repeater channel Otherwise, indicates channel

Example:

Append the **Run_Macro** command at the end of the **iTAG_Mode** to cause the iTAG to execute Macro_ID (3) once per second while transmitting data over the LAN.

r **1 2 97 3 1**

See Also: Run_Macro, Load_Macro, IO_Address

Last updated 11-Nov-08

iTAG_ID: CMD_ID=0x03 (decimal 3)

Format: r 3 <D3> <D2> <D1> <D0> [DeviceType]

iTAG_ID sets the four byte iTAG identification number.

This ID may also be programmed from the Setup Menu. (**GEN2**, **GATE**, **RPT**)

Each iTAG should have a unique four byte address. Group calls for multiple iTAG's may be achieved by setting Group Call ID's via the **Broadcast_Address** command.

This command also updates the first four bytes of the **MAC_Addr** with this entered **iTAG_ID** value.

If DeviceType is included and it is 'I', 'g', or 'G', then the system Device Type variable is also updated.

Description:

<D3>	Most Significant Byte of iTAG address
<D2>	
<D1>	
<D0>	Least Significant Byte of iTAG address
[DeviceType]	If 'I', 'g', or 'G', then system device type is updated.

Example:

Program new **iTAG_ID** (0.1.2.3)

r 3 0 1 2 3

See Also: Broadcast_Address, Default_Dest, Server_Address, Commit_Flash

Last updated 11-Nov-08

No_OP: CMD_ID=0x04 (decimal 4)

Format: r 4 [D3] [D2] [D1] [Dn]

An iTAG **No_Op** command is used to test the Network connection to an iTAG. Sending a **No_OP** command from one iTAG to another causes the receiving iTAG to respond to the Source ITAG with an ACK packet containing the iTAG_ID. A **No_OP** causes the iTAG to perform no other functions. Data appended to the end of the **No_Op** command can be used to perform a communication test. The user may send up to 236 bytes of data using the **No_OP** command. The returned ACK will append the user data to the end of the ACK. A failed CRC error will be written to the iTAG display if a communication error occurs.

Examples:

Send **No_Op** packet to iGate 0.0.0.0

This command is useful to establish whether a recently installed iTAG is connected to the iTAG LAN. When an installer sets up an iTAG, the No_OP command can be sent to the network iGATE while at the iTAG location. If networked, the iGATE will respond with an ACK packet to the iTAG. The installed iTAG will print to the screen that an ACK has been received from the iGATE. If available, the Radio signal strength will be displayed when the ACK is received.

R 5 1 0 0 0 0 **4** (standard No_OP command)

R 5 1 0 0 0 0 **4** *1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16* (No_OP with test data)

See Also: Forward_Packet

Last updated 11-Nov-08

***Forward_Packet*: CMD_ID=0x05 (decimal 5)**

Format: r 5 <Forward_Type> <Dest> <Forward Packet>

iTAG's may forward data packets and commands to other iTAG's or iGATEs using the **Forward_Packet** command. The **Forward_Packet** provides all of the necessary routing information for the packet to make multiple hops, command an iGATE, or send data to a server through a complex network topology.

Decimal:

<Forward_Type>:	'C':	Forward to an iCHIME (**GEN2**)
	0:	Forward packet once. Do not wait for an ACK. Use encryption configured on iTAG (see **Net_Type**)
	1:	Forward packet using transmit buffer configured for timeout and retries (see **Command_Retry**). Use encryption configured on iTAG
	's':	Forward packet once. Do not wait for an ACK. Do not encrypt. (**GEN2**)
	243:	Forward packet once. Do not wait for an ACK. Use encryption. Note 243 == 128 | 's' (**GEN2**)
	'S':	Forward packet using transmit buffer configured for timeout and retries. Do not encrypt (**GEN2**)
	211:	Forward packet using transmit buffer configured for timeout and retries. Use encryption. Note 211 == 128 | 'S' (**GEN2**)
<Dest>:		Four byte address which is the desired destination ID for the packet.
< Forward Packet>:		Data to be sent to other iTAG's

Example:

Send **No_Op** packet to iGate 0.0.0.0

r 5 1 0 0 0 0 **4**

Send **iTAG_Mode** command to (4.5.6.7)

r 5 1 4 5 6 7 **1 2**

See Also: Command_Retry, Net_Type

Last updated 11-Nov-08

Load_Config: CMD_ID=0x06 (decimal 6)

Format: R 7 <Lat> <Lon> <Alt>

iGATE location information may be sent using the **Load_Config** command. The **Load_Config** command sets the iGATE position in memory.

The iGATE location is used as the default location when GPS has not been acquired. It is also used to calculate distance relative to an iGATE.

Definitions:

<Lat>	Latitude of the iGATE in degrees (ASCII formatted float)
<Lon>	Longitude of the iGATE in degrees (ASCII formatted float)
<Alt>	Altitude of the iGATE in degrees (ASCII formatted float)

Examples:

R 6 "30.5 -121.7 10." Set the iGATE position to 30.5 degrees latitude, -121.7 degrees longitude, and altitude of 10

R 6 51 48 46 53 32 45 49 50 49 46 55 32 49 48 46 Same command using ASCII codes

See Also: Appendix C, Packet ID 0x00 0xfa

Last updated 11-Nov-08

Send_Config: CMD_ID=0x07 (decimal 7)

Format: r 7 <port> <param>

iTAG configuration data may be requested (or sent) using the **Send_Config** command. The **Send_Config** command causes the iTAG to collect the specified data (defined by Param) and send the data out the specified Port. Data format is automatically determined by the function define for the specified port.

Any packet id (as defined in Appendix C) with the first byte of 0 can be sent using this command.

Definitions:

<port>	0-2	Serial port used for configuration data transmission
<param>	6	Standard iTAG configuration data (Packet ID 0x00 0x06)
	23	Broadcast ID's (up to 20 four byte addresses) (ID 0x00 0x17)
	27	Programmed Phone Number for Modem dial outs (ID 0x00 0x1b)

Examples:

r 7 1 6 Send the "standard" iTAG configuration parameters out serial port 1

r 7 2 23 Send all programmed Broadcast ID's out serial port 2

See Also: Broadcast_ID, Phone_Number, Appendix C

Last updated 11-Nov-08

Network: CMD_ID=0x08 (decimal 8)

Format: r 8 <Type> [Params]

iTAG's use an 802.15.4 radio for LAN communications. **Network** commands are used to configure the network.

Definitions:

		Params
<Type>	'A'	Force association to specified state ('l', 'L', 'N', 'S', 'W') P0: association state P1: Comm (Association) timer value – stay associated P1 seconds
	'C'	Set Commissioned state P0: 'Y' or 'N': Commissioned state is Yes or No. See **Test_Condition**
	'D'	Action when beacon received from specified address P0: 'E'-enable macro, 'D'-disable, 'C'-command enabled P1-P4: 4 byte address to check against beacon sender P5: Macro number for P0='E', Command length for P0='C' P6: Command number for P0='C' P7-Pn: Command parameters for P0='C'
	'I'	Setup Internetworking for meshing
	'J'	Insert a unit into ACL (Join) P0-P7: 8 byte address to insert into the ACL P8: 0x80 or 0 indicates whether to use encryption or not when communicating with this unit P9: Non zero indicates that this unit has power saving enabled. Send wakeup packets to this unit before trying to communicate.
	'L'	Delete unit form the ACL (Leaving) P0-P7: 8 byte address of unit to remove from the ACL
	'R'	If address is different from destination ID, send an association request P0-P3: 4 byte address to send the association request to
	'T'	Set ACL Timer to specified value. Unit in ACL stay in ACL for this specified number of seconds P0: MSB of ACL timer (unsigned int) P1: LSB of ACL timer (unsigned int)

Examples:

r 8 'C' 'Y' Set commissioned state to Yes

r 8 'DC' 14 0 0 1 3 249 10 11 Set DataByte[10]=11 when a beacon from 14 0 0 1 is received

r 8 ‘J’ 0 0 0 0 1 2 3 4 128 5	Add unit 0.0.0.0.1.2.3.4 to the ACL, unit uses encryption (128), unit uses power save (5)
r 8 ‘T’ 1 5	Set units ACL Timer to 261 seconds

Last updated 11-Nov-08

Port_CFG: CMD_ID=0x09 (decimal 9)

Format: r 9 <port> <baud> <cfg> <function> [Param]

Port_CFG remotely (or via a Macro) change a serial port configuration identified by <port>. Once the **Port_CFG** is executed, the specified port is re-initialized.

The iTAG serial ports can be configured via the iTAG Setup Menu. (**GEN2, GATE, RPT**)

Definitions:

		Param
<port>	Serial port to be changed (0-2)	
<baud>	New baud rate, ranges from 1.2-115 Kbps (use baud/1200)	
<cfg>	For Radio (port 2) specify radio parameters	
<function>	2:	LANPort[0]
	3:	LANPort[1]
	4:	ModemPort
	5:	DataPort
		P0: 'T' or 'F', enables checking of this serial port as a user interface
	6:	LCD port
	'D':	Disable echo port (Not implemented)
		P0: Port number to be used as an echo port
	'E':	Set Echo Port (Not implemented)

Example:

Set Serial Port 1, to 57.6KBps, use for LANPort[1]

R 9 1 48 0 3

See Also:

Last updated 11-Nov-08

Set_Time: CMD_ID=0x0a (decimal 10)

Format: r 10 <D0> <D1> <D2> <D3> <D4> <D5> <D6>

Set_Time programs the iTAG real-time clock using seconds since 1980. This command is not implemented.

Example:

Set timer to 1(month) 2(day) 3(year) 4(week day) 5(hour) 6(minute) 7(seconds): January 2, 1983, 05:06:07. Week Day is ignored.

r 10 1 2 3 4 5 6 7

See Also:

Last updated 12-Nov-08

Net_Type: CMD_ID=0x0b (decimal 11)

Format: r 11 <cmd> <Params>

Net_Type defines alternative addressing schemes available on iTAG's along with an encryption key. In most cases, **Net_Type** is not a command sent over the LAN. Typically, **Net_Type** is one of many commands in an iTAG configuration file which programs iTAG's prior to deployment. It is possible to use **Net_Type** to periodically change an encryption key to increase the security of a Network. Extreme caution should be used when sending **Net_Type** over an established LAN network. Changing an iTAG address scheme or encryption may render an iTAG unreachable until the Network iGATE has an updated configuration also.

Definitions:

		Params
<cmd>	'I'|'Z'|'F':	Set Network type to iControl, Zigbee, or Fulton respectively
	'K'	Set encryption Key P0: ASCII formatted index of encryption key to set P1-P16: Encryption key
	'E'	Enable or disable encryption for the radio network P0: 0 disables encryption, any non-zero enables encryption

Examples:

Create an iTAG configuration file which defines the network type (wireless LAN) and sets encryption key 0 to "myencryptionkey0". The configuration file should also define the **iTAG_ID**, and **Radio_Type.** Make sure that the configuration file commits the changes to iTAG flash.

Note: The Default Broadcast_Address is (0.0.0.0).

The Configuration file would contain.

r 3 0 0 0 1 // Set the ITAG_ID to (0.0.0.1)
r 11 'I' // Use iControl wireless LAN
r 11 'K0' "myencryptionkey0" // load encryption key 0
r 25 "Z" // Use Zigbee Radio
r 240 // Commit changes to flash

See Also: Configuration Files, Broadcast Address, Network Security

Last updated 11-Nov-08

Power_Cfg: CMD_ID=0x0c (decimal 12)

Format: r 12 <Normal> <Sleep> <Macro>

Power_Cfg sets the normal and Sleep mode power configuration. During normal operation, the Radio, GPS, and ASIC are enabled. In this configuration the iTAG draws approximately 100 mamps. To reduce power, the iTAG may be placed in a sleep mode using a command or programmable timeout. During Sleep, the iTAG either wakes up once per second to check for radio traffic, a scheduled command, or serial port data. While in Sleep, an iTAG uses approximately 10 mamps. Both the normal and sleep configurations may be set by the **Power_Cfg** command.

Definitions:

	Power mode setting for normal operation	
<Normal>	0x01	Enable radio
	0x02	Enable GPS
	0x04	Enable ASIC
	Power mode sitting during sleep operation	
<Sleep>	0x01	Enable radio
	0x02	Enable GPS
	0x04	Enable ASIC
	0x08	Enable deep sleep (Deprecated)
	0x10	Enable sleep
<Macro>	Not implemented	

Example:

While awake, use a completely powered iTAG, while in Sleep, power down everything.

Note: This command only needs to be sent once to maintain this sleep/wake configuration.

r 12 7 16

See Also: Power_Save

Last updated 11-Nov-08

IO_Addr: CMD_ID=0x0d (decimal 13)

Format: r 13 <BoardAddress> <ExtAddress>

Legacy hardware allowed a unit interface to a variety of Input and Output (IO) devices. Each IO interface board had a unique address. An IO address would include a Board address and an Extended address; thus the naming convention.

IO_Addr is typically used to set the default address of the iTAG IO. When the **iTAG_Mode** is set to send data, the current value of **IO_Addr** is used to build the packet sent over the LAN.

Definition:

<BoardAddress> and <ExtAddress> are equivalent to the two byte Packet ID shown in Appendix C.

Example:

Set Default IO address to reflect an Analog-to-Digital status packet (56)(12).

r 13 56 12

See Also: Appendix C, iTAG_Mode, Save_Data, Upload_Record, DataByte

Last updated 11-Nov-08

Default_Dest: CMD_ID=0x0e (decimal 14)

Format: r 14 <D3> <D2> <D1> <D0>

When the iTAG must send an un-requested data packet (like alarms) to the iGate or Server, the iTAG uses the **Default_Dest** as the sending address. Typically, the **Default_Dest** should be the same four byte address as the network's iGate ID.

Definition:

<D0>	Least significant Byte of Destination address
<D1>	
<D2>	
<D3>	Most significant Byte of Destination address

Example:

Change the iTAG default destination ID to (0.0.0.0)

r 14 0 0 0 0

See Also: Check_Alrms, Run_Macros

Last updated 11-Nov-08

Power_Override: CMD_ID=0x0f (decimal 15)

Format: r 15 <PowerCfg>

Power_Override sets the current power configuration to that specified by the PowerCfg bitmask parameter.

Definitions:

<PowerCfg>	0x01	Enable radio
	0x02	Enable GPS
	0x04	Enable ASIC
	0x08	Enable deep sleep (Deprecated)
	0x10	Enable sleep

Example:

Enable only GPS and ASIC.

r 15 6

See Also: Power_Cfg

Last updated 11-Nov-08

***Power_Save*: CMD_ID=0x10 (decimal 16)**

Format: Z(Source) (Destination) (16) (Power_Save) (seconds)

The iTAG will enter Sleep mode after the sleep counter decrements from the **Power_Save** count to zero. Setting **Power_Save** to zero disables the iTAG from transitioning into Sleep mode. Any **Power_Save** value above zero, represents the number of seconds before the iTAG will transition to Sleep.

While in Sleep mode, the iTAG checks to see if any Scheduled commands need to be processed. Also, all serial ports are monitored to see if any activity indicates a transition to normal operation may be required. When an iTAG is wakened, the Power configuration on wake up is controlled by the **Power_Cfg**. The initial **Power_Save** count remains the same value set by the **Power_Save** command. The power down count re-starts from the **Power_Save** count. If there is no serial port activity, or executing macros, the iTAG will return to sleep mode after the **Power_Save** counter decrements to zero.

Definitions:

<Power_Save>	0x00	Disable Power_Save
	1-255	Transition to Power Save mode after (seconds)

Example:

Enable Sleep mode with a **Power_Save** count down of 10 seconds (0x0a).

S *0 0 0 0* 1 2 3 4 **16 10**

Last updated 11-Nov-08

AT_Dial: CMD_ID=0x11 (decimal 17)

Format: r 17 <Enable | Disable>

An iTAG that is configured with a modem uses the **AT_Dial** command to enable (or disable) an attempt to communicate with iVIEW using the modem. If a modem is not enabled, the **AT_Dial** command will be ignored. The number the modem uses to dial is set with the **Phone_Number** command.

When **AT_Dial** is enabled, **AT_Answer** is automatically disabled.

Definitions:

<Enable/Disable>: 0x01= Enable modem dial out
0x00= Disable modem dial out

Example:

Command the modem to dial the stored **Phone_Number**

r 17 1

See Also: Phone_Number, AT_Answer, Set_Modem

Last updated 11-Nov-08

AT_Answer: CMD_ID=0x12 (decimal 18)

Format: r 18 <Enable | Disable>

An iTAG that is configured with a modem utilizes the **AT_Answer** command to enable (or disable) checking for connections on the modem. If a modem is not enabled, the **AT_Answer** command will be ignored. An iTAG may leave the parameter **AT_Answer** equal to 0x01 (Answer enable) even if the modem is not receiving a call. The parameter **AT_Answer** simple permits the iTAG to answer if a remote computer is attempting to communicate via the modem.

Definition:

<Enable/Disable>: 0x01= Enable modem Answer
0x00= Disable modem Answer

Example:

r 18 1

See Also: Phone_Number, AT_Dial, Set_Modem

Last updated 11-Nov-08

Modem_Type: CMD_ID=0x13 (decimal 19)

Format: r 19 <Modem_Type>

To enable the iTAG to use a modem, the **Modem_Type** command is sent to enable the appropriate iTAG drivers. Before enabling the iTAG **Modem_Type**, the modem serial port must be enabled for modem use and the corresponding modem baud rate should be selected.

Check the Set Up Menu to verify the modem serial port properties. (**GEN2, GATE, RPT**)

Definition:

< Modem_Type >	'N':	No Modem
	'P':	PSTN
	'C'	Cellular connecting to remote PSTN
	's':	Cellular using SMS
	'S':	Satellite
	'e':	Ethernet with DNS and DHCP enabled
	'E':	Ethernet
	'g':	GSM using GPRS
	'i':	Iridium SBD modem
	'I':	Iridium modem

Example:

Enable the iTAG to use the cellular modem using SMS.

r 19 "s"

See Also: AT_Answer**,** AT_Dial, Phone_Number, Hangup_Modem

Last updated 11-Nov-08

Hangup_Modem CMD_ID=0x14 (decimal 20)

Format: r 20 [ModemState]

For general usage, the **Hangup_Modem** command ends the present modem connection. In most cases, the ModemState is set to 'S' which will (shutdown) power down and disable the modem.

For a PSTN modem, if the modem is connected, the iTAG may hang-up the modem and return to command mode using the **Hangup_Modem** command. The **Hangup_Modem** command is equivalent to the standard (AT modem command) ATH0. If the user wants to override the modem state (in the iTAG software), an optional parameter can be provided which controls the software state. This feature is useful for writing custom modem drivers using macros, or adapting the iTAG to unsupported modem types.

Please Note: Overriding the modem state parameter does not control the external modem, only the iTAG firmware state machine is affected by the ModemState parameter i.e. (commanding ModemState=> Connect, does not mean the modem is connected. The iTAG with "think" it is connected)

Definition:

< ModemState >	**'**I':	Initialize modem
	'C':	Modem State is connected
	'G':	Modem command mode, while online
	'R':	Modem state = ready
	'O':	Modem is offline
	'D':	Modem is dialing a number
	'A':	Modem is answering a call
	'X':	Modem is waiting for a connection
	'S':	Shutdown modem

Example:

Send command to hang up modem, then reconfigure modem to Answer any incoming call.

r 20 // Hangup command
r 18 1 // AT_Answer = 1; Enable Answer

See Also: AT_Answer**,** AT_Dial, Phone_Number

Last updated 11-Nov-08

***Enable_Schedule*: CMD_ID=0x15 (decimal 21)**

Format: r 21 [Enable | Disable]

The iTAG Operating System supports stored commands which are executed at User specified times. The stored commands utilize the clock/calendar function on the iTAG to determine when a scheduled command should be executed. **Enable_Schedule** is used to enable or disable the execution of stored commands. A User may wish to disable schedule checking to prevent an iTAG from "Hanging up" during a commanded connection.

Definition:

<Enable/Disable>: 0x01= Enable Schedule Checking
0x00= Disable Schedule Checking

Example:

r 21 1 Enable Schedule checking

See Also: Load_Schedule, Dump_Schedule

Last updated 11-Nov-08

Radio_Parameters: CMD_ID=0x16 (decimal 22)

Format: r 22 <TXpreamble> <TXtail> <bSpacing>

An iTAG configured with a LAN radio uses parameters to control the timing for transmit, receipt and RX mode. **TXpreamble** is deprecated and no longer used, but is reserved for future use. **TXtail** is the duration of time (milliseconds) that the iTAG waits before sending an ACK. **bSpacing** is used in the **Delay_mSec** command.

Definitions:

<TXpreamble>:	Deprecated
<TXtail>:	0-255 msec
<bSpacing>:	0-255 msec * DataByte[0] to delay in **Delay_mSec** command

Example:

Set Radio parameters

TXpreamble don't care
TXtail to 150 msec
bSpacing 10 msec

r 22 0 150 10

See also: Delay_mSec

Last updated 11-Nov-08

Broadcast_Address: CMD_ID=0x17 (decimal 23)

Format: r 23 <N_Addrs> <B03> <B02> <B01> <B00> … <Bn3> <Bn2> <Bn1> <Bn0>

An iTAG may be programmed with "Broadcast" addresses that are used as alternative **iTAG_ID's**. The iTAG will not respond with an ACK when commanded using a broadcast address. Multiple iTAG's may have the same broadcast address allowing groups of iTAG's to be commanded with a single command. Typical uses for broadcast addresses are commanded time updates or Macro loads which all units in a group need. Up to 20 broadcast addresses may be used per iTAG.

The factory default configuration defines one **Broadcast_Address** (0.0.0.0.). This allows multiple iTAG's to be programmed with the same Configuration File via the user interface. iControl recommends retaining the first **Broadcast_Address** as a generic **iTAG_ID** for configuration loading.

Definitions:

N_Addrs:	1-20 possible broadcast addresses (0 disables)
B00-B03:	Broadcast address <0> (Bytes 1-4)
Bn0-Bn3:	Broadcast address <n> (Bytes 1-4)

Example:

Program the following three broadcast addresses:

0.0.0.0
1.2.3.4
5.6.7.8

r 23 3 0 0 0 0 1 2 3 4 5 6 7 8

See Also: Configuration File

Last updated 11-Nov-08

LAN_Device: CMD_ID=0x19 (decimal 25)

Format: r 25 <Device>

The iTAG LAN network device is an 802.15.4 compliant radio.

LAN device types are selected during iTAG initialization with the setup menu (**GEN2, GATE, RPT**) or via a configuration command file.

Definitions:

<Device>:	'N':	No Device (disables Devices)
	'Z':	802.15.4 compliant device

Example:

Set **LAN_Device** for 802.15.4 compliant radio.

R 25 "Z"

See Also: iModem_Param

Last updated 11-Nov-08

CMD_Retry: CMD_ID=0x1a (decimal 26)

Format: r 26 <TX_Attempts> <TX Retry Time> <MaxUploadFail>

Command transmissions are placed in the iTAG transmit queue. Commands are transmitted immediately following the placement in the queue. After "TX_Attempts", the packet is discarded and a message is posted to the user interface. Each transmission attempt is separated by "Retry Time" seconds. All commands forwarded from an iGate places commands in the transmit queue. iTAG ACK's are not placed in transmit queues.

Definitions:

< TX_Attempts >	Number of times a packet may be sent before it is discarded
<TX Retry Time>	Number of seconds to wait before attempting to send packet again
<MaxUploadFail>	During a data upload attempt, the upload fails if the transmit fails TX_Attemtps times. If this failure occurs MaxUploadFail times, the iTAG is automatically disassociated

Example:

Set the transmit queue to discard messages after 4 attempts. After each transmission failure, wait 8 seconds before attempting the next transmission. Allow four failed attempts at uploading before disassociating.

R 26 4 8 4

See Also:

Last updated 11-Nov-08

Phone_Number: CMD_ID=0x1b (decimal 27)

Format: r 27 ”(c0)(c1)(c2)(c3)….(cn)”

An iGATE or modem enabled iTAG can connect to either an ISP, modem bank , directly to a customer phone line, connect using GPRS, or send an SMS message. All these services use the **Phone_Number** command to set the default dial out number for connections. **Phone_Number** is up to 20 ASCII characters long, and includes the numbers 0-9 and “,”. Commas insert a 2 second delay between numbers while dialing. Multiple commas may be inserted to allow switch board timing from internal phone system to transition to external phone lines. Alternate numbers may be stored in Macros which may be used in the event of a connection failure.

Examples:

Set the default dial out number to be 1 800 555 5555.

r 27 “18005555555”

Set the default dial out number to be, 9 wait 4 seconds then dial 555 1234.

r 27 “9,, 5551234”

See Also: Modem_Type, Appendix C

Last updated 11-Nov-08

Password: CMD_ID=0x1c (decimal 28)

Format: r 28 <p0><p1><p2><p3>…<pn>

iTAG Set-Up and ASCII command functions may be password protected to prevent accidental or deliberate commands from being executed. **Password** can be sent remotely through the LAN or via the User Interface. The **Password** command is functionally the same as entering the Set-Up Menu using the iTAG login via the User Interface. The password is an ASCII alpha-numeric string with no more than 9 characters. Spaces and control characters are not allowed in the password. When the **Password** command is sent, both the Set Up menu and iTAG command processing functions are enabled. The string which accompanies the **Password** command is the iTAG password. Sending a zero length string disables both Set-Up Menu access and the command process functions.

The Factory default password settings are:

Password:	"icontrol".
SET-Up Menu:	Password Required
ASCII Commanding:	Access Permitted

Examples:

An iTAG has its password set to "yourword". For security purposes, a User programmed Macro[38] sends the password command "0" effectively disabling Set-Up menu access and command processing. Send a command to the iTAG to re-enable ASCII commanding.

r 28 "yourword"

After the iTAG receives this command, access to the Set Up menu and command processing is enabled. When the user is ready to disable access, send a zero length **Password** command.

R 28

Password may be used remotely to enable/disable iTAG access.

See Also: Set_Password, Run_Macro,Load_Macro

Last updated 11-Nov-08

***Upload_ACL*: CMD_ID=0x1d (decimal 29)**
***Upload_Data*: CMD_ID=0x1f (decimal 31)**
***Upload_Macros*: CMD_ID=0x62 (decimal 98)**
***Upload_TextMsg*: CMD_ID=0xc9 (decimal 201)**
***Upload_Schedules*: CMD_ID=0xfe (decimal 254)**

Format: r <cmd> <port> <n_packet>

To upload data to iVIEW (or other application), **Upload_ACL/Upload_Data/Upload_Macros/Upload_TextMsg/Upload_Schedules** is used to start the data transfer process. This command defines which serial port is used for the upload. The user is responsible for knowing which port is required for the data transfer. The port selected may be the UserPort, the LANPort, or the modemPort. The second parameter is deprecated.

During the upload process, the MSB bit of the Board Address is set 'high' to indicate the data is stored (not currently streaming).

Definitions:

<port>:	0:	UserPort
	1:	(modemPort, dataPort)
	2:	LANPort
<n_packet>:	deprecated	

Examples:

Command the iTAG to upload the ACL using port1.

r 29 1 4

Command the iTAG to upload the Macros using port2.

r 98 2 0

See Also: Network, Save_Data, Load_Macro, Text_Msg, Load_Schedule

Last updated 11-Nov-08

Save_Data: CMD_ID=0x1e (decimal 30)

Format: r 30 <Board_Addr> <Ext_Addr>

Both iTAG's and iGATE's provide a flexible approach to save data for later upload. The **Save_Data** command instructs the iTAG to collect data using the specified packet IDs. See Appendix C for packet ID definitions. The **Save_Data** command may be sent in real-time, by a stored command, or as a result of an event. All saved data is placed in a single data buffer that places the data with a time tag (most recent first). The saved data buffer is a circular buffer. When the data buffer is full, oldest data is over written first. When data is uploaded from the stored data buffer, the MSB of the Board address is set "high" to signal the packet is a stored data packet. Users are responsible for performing periodic uploads of the stored data to prevent loss of mission critical information.

Description

See Appendix C for packet ID definitions.

Examples:

Load an iTAG program to save analog data from the motherboard every 15 minutes.

r 253 0 255 255 255 255 255 0 0 **30 56 12** Top of hour
r 253 1 255 255 255 255 255 15 0 **30 56 12** 15 minutes
r 253 2 255 255 255 255 255 30 0 **30 56 12** 30 minutes
r 253 3 255 255 255 255 255 45 0 **30 56 12** 45 minutes

See Also: IO_Addr, Appendix C

Last updated 11-Nov-08

Modem_Parms: CMD_ID=0x22 (decimal 34)

Format: r 34 <cmd> <Params>

The iTAG 802.15.4 radio may be configured using the **Modem_Parms** command. The iTAG radio supports channels ranging from1 to 26. The iTAG will listen on a Default Channel when it is awake. It will listen on the Sleep Channel when asleep. When configured as a repeater, the iTAG will communicate with an iGATE on the Modem Channel and operate as above when communicating with other iTAGs.

There is also the concept of a current channel. This value holds the channel that the radio is currently operating on.

Some modem parameters may also be set using this command.

Definitions:

		Params
<cmd>:	'P':	Copy 802.15.4 radio initialization parameters. This command does not actually initialize the radio with these parameters. It only copies the parameters that may be used to initialize the radio using the **Radio_Config** command
	's':	Save the current channel for later restoring
	'r':	restore current channel to that saved with the 's' command
	'S':	if Association is not 'L' change channel as in 'C'
	'C':	change indicated channel P0: 'D' – change default channel 'S' – change sleep channel 'M' – change modem channel (**RPT**) P1: (1-26) Channel to change to
	'c':	Change the current channel P0: (1-26) Channel to change to
	'D':	Reinitialize radio with default values
	'T':	Modem timeout – used for maximum time to wait for a valid modem signal strength and registration on a network P0: MSB of timeout (unsigned int) P1: LSB of timeout (unsigned int)
	'R':	Flag indicating that an iGATE should store any upload data from a remote tag in the iGATEs local flash. This data may be uploaded through the iGATE modem at a later time. P0: 0 indicates do not save data, otherwise do so
	'Q':	Query iVIEW for any commands through the modem

Examples:

Set the default radio to channel 11.

R 34 ‘CD’ 11

See Also: Radio_Config

Last updated 11-Nov-08

iModem: CMD_ID=0x23 (decimal 35)

Format: r 35 <mode> <len> <Params>

Definitions:

<mode>

Examples:

See Also:

Serial_R/W: CMD_ID=0x27 (decimal 39)

Format: r 39 <R/W> <port> <data>

iTAG's may send or receive data via the serial ports to communicate with smart sensors. For most applications this operation is performed using the serial port configured as the DataPort. With this command, users define the parameters necessary to prompt MODBUS devices or PLC's to return data for the iTAG to store or place in telemetry.

Definitions:

<R/W>	(0)	Sets message sync characters used for receiving data. There are 3 sync characters. Two start characters and a terminating character.
	(1)	Write <serial message> bytes out specified port
	(2)	Write contents of DataBytes out specified port
	(3)	Reset specified serial port to flush receive buffer
<port>	0-2	Serial port for data read write operations

Examples:

Prompt a GPS receiver to return data. Send appropriate prompt, wait 20 milliseconds, read data into a stored data packet.

Assume:

a) GPS receiver is hooked to serial port 2
b) "#<ENTER>" Generates data prompt
c) Receiver needs 20 milliseconds to respond with data
d) Start of returned data packet is a "&=".
e) The terminating character is <ENTER>

Load Macro 0 with the following commands, (in a text file they are decimal)

r 39 0 2 38 61 13	Sets the message sync and terminating characters
r 39 1 2 35 13	Send the Prompt (ASCII decimal #,enter)
r 102 20	Delay Macro[0] 20 milliseconds
r 30 0 32	Read/save data string from serial port 2 (0x20=32)

Each time the user wishes to save data from the GPS receiver, execute Macro 0 with a **Run_Macro** command.

See Also: MS_Delay, Save_Data, Run_Macro

Last updated 11-Nov-08

General_R/W: CMD_ID=0x28 (decimal 40)

Format: r 40 <R/W> <Addr> <data>

iTAG's may read or write data to internal or external IO address space. At present, **General_R/W** only writes to general purpose I/O pins.

Although implemented in hardware which can change with each revision, the GPIO 3 pin is usually used to control the power to an installed modem.

Definitions:

<R/W>	(4) Write to a general purpose I/O pin
<Addr>	(3,4,10,15) GPIO Pin to write to
<data>	(0 or 1) Set specified GPIO pin to 0 or 1

Examples:

Set GPIO pin 3 to high (1).

r 40 3 1

See Also: Save_Data, Load_Macro,Run_Macro

Last updated 11-Nov-08

Beacon: CMD_ID=0x3c (decimal 60)

Format: r 60 <broadcast addr> <BeaconTimerMSB> <BeaconTimerLSB> <type> <Channel>

iTAGs join the iGATE 802.15.4 network by sending a beacon request. The only valid parameter when sending a beacon request is the broadcast address.

An iGATE sends a Beacon signal to inform iTAGs of the channel that the iGATE is listening on.

Definitions:

<broadcast addr>	Index of broadcast address to use when sending beacon or beacon request
<BeaconTimerMSB>	Most significant byte of an unsigned int value indicating how many seconds that the remote iTAG should remain associated
<BeaconTimerLSB>	Least significant byte of an unsigned int value indicating how many seconds that the remote iTAG should remain associated
<type>	Beacon type: 'W' or 'B'. W – send beacon using the Sleep Channel. B – send beacon using the Default Channel
<Channel>	Radio channel that remote iTAGs use to communicate with this iGATE

Example:

Request a beacon using the first broadcast address.

r 60 0

Send a beacon on the Sleep Channel using broadcast address 0. Send a beacon timer of 900 seconds. Tell remote iTAGs to use Channel 14 to communicate with this iGATE.

r 60 0 3 132 'W' 14

See Also: Broadcast_Address, Modem_Parms

Last updated 11-Nov-08

Track: CMD_ID=0x3e (decimal 62)

Format: r 62 <Command> [params]

Program Track mode and motion sense parameters.

The motion sense parameters are used when the iTAG is equipped with an accelerometer. The accelerometer is checked once per second. If the magnitude of the accelerometer axis has changed more than MotionDetect, then increment a counter. If that counter is greater than MotionTimer, indicate that motion has occurred. This counter is reset with each 63 'M' command.

The track parameters are used to check for changes related to GPS. The 63 'T' 'P' or the 62 't' 'P' command uses the TrackRange value to determine if the iTAG has moved greater than TrackRange meters from Waypoint[0]. Waypoint[0] is overwritten with the present location if the iTAG is greater than TrackRange meters from the previous Waypoint[0] value.

The algorithm that uses the TrackVelocity, MovingFix, and StationaryFix is not implemented.

The waypoints are used to calculate distance from the present position. These distances can then be formatted into a data packet and used in the Check_Alarm command.

All parameters can be saved to non-volatile ram using the 240 command. (**GEN2**)

Definitions:

		(Params)
<Command>	C	Set Velocity counters (Not presently implemented)
		P0: Moving Fix MSB of unsigned int
		P1: Moving Fix LSB of unsigned int
		P2: Stationary Fix MSB of unsigned int
		P3: Stationary Fix LSB of unsigned int
	D	Debug/Display parameters – only valid in DN display
	M	Set motion sense parameters
		P0: Motion Detect MSB of unsigned int
		P1: Motion Detect LSB of unsigned int
		P2: Motion Timer MSB of unsigned int
		P3: Motion Timer LSB of unsigned int
	t	Run track algorithm
		P0: P | V | T
	T	Set Tracking parameters
		P0: Track Range (string formatted float)
		P1: Track Velocity (string formatted float)
	W	Program waypoints
		P0: 'C' – clear all waypoints

P0: Index of waypoint to program
P1: Waypoint Lat (string formatted float)
P2: Waypoint Long (string formatted float)

Example:

Set Track Range to 500 meters and Track Velocity to 10.3 meters per second.

r 62 'T' "500. 10.3"
r 63 84 53 48 48 46 32 49 48 46 51

Set waypoint 2 to 35.12 and -90.01 degrees

r 62 'W' 2 "35.12 -90.01"
r 62 87 2 51 53 46 49 50 32 45 57 48 46 48 49

Set motion detect to 60 and motion timer to 301.

r 62 'M' 0 60 1 45
r 62 77 0 60 1 45

If DataByte[5]==8 and iTAG is associated and GPS is valid, execute macro 20, else do nothing. Note that the 'x' value in "G=x" is a don't care as GPS is either valid or not and this state is not compared to any value that may be set in the firmware.

r '?d58?A=L?G=xrM' 20 99
r 63 100 5 8 63 65 61 76 63 71 61 120 114 77 20 99

See Also: Test_Condition, Check_Alarm

Last updated 11-Nov-08

Test_Condition: CMD_ID=0x3f (decimal 63)

Format: r 63 "<Object>=<Test>rM" <true Macro> <false Macro>
Alternate: r 63 "<Object>=<Test>rC" <Command>
Alternate: r 63 "d<Index><Value>rM" <true Macro> <false Macro>
Alternate: r 63 "d<Index><Value>rC" <Command>

iTAG's have many conditions that may be tested for a certain condition. The Test_Condition command allows logic that will execute a certain macro if a condition is true and a different macro if the condition is false. Alternately, a command may be executed if the condition is true.

Tests may be combined, where each test case will be checked in order. The true case is executed if and only if all test cases are true.

Definitions:

<Object>: iTAG state to be tested
<Test>: State of Object for comparison
<true Macro>: Macro to run if above comparison is TRUE
<false Macro>: Macro to run if above comparison is FALSE
<Command>: Command to run if above comparison is TRUE

Object	Description	Possible States for Test
A	**Is iTAG is associated in an 802.15.4 network**	**l: transitory state, iTAG is about to become associated** **L: iTAG is associated with an iGATE** **n: transitory state, iTAG is about to become disassociated** **N: iTAG is not associated** **S: iTAG is scanning** **T: iCHIME is associated** **W: iGATE is associated in a Wide Area Network**
C	**Is iTAG commissioned**	**N: iTAG has not been commissioned** **Any other 8-bit value can be commanded and therefore tested**
d	**Test if DataByte[Index]==value**	**Any 8-bit value can be commanded and therefore tested for**
D	**Is there stored data**	**None**
G	**Is GPS valid**	**None**
M	**Has there been motion sensed**	**N: no motion** **Y: motion has occurred**

N	**Check for a change in the number of members in the ACL**	**None**
t	**Check Track status, save data if true**	**None**
T	**Check Track status**	**None**

Power Considerations:

If either true or false Macro is a valid macro and that macro will be executed, the iTAG is reset from power savings mode.

If a command is to be executed, the power savings mode will be reset according to that specific command.

Example:

If iTAG is associated, execute macro 5, execute macro 6 otherwise.

r '?A=LrM' 5 6
r 63 65 61 76 114 77 5 6

If iTAG is associated, execute command to change mode to flow 5 samples.

r '?A=LrC' 1 5 5
r 63 65 61 76 114 67 1 5 5

If DataByte[5]==8, execute macro 20, else do nothing

r '?d58rM' 20 99
r 63 100 5 8 114 77 20 99

If DataByte[5]==8 and iTAG is associated and GPS is valid, execute macro 20, else do nothing. Note that the 'x' value in "G=x" is a don't care as GPS is either valid or not and this state is not compared to any value that may be set in the firmware.

r '?d58?A=L?G=xrM' 20 99
r 63 100 5 8 63 65 61 76 63 71 61 120 114 77 20 99

See Also: Commissioned, Track, DataByte

Last updated 11-Nov-08

MAC_Addr: CMD_ID=0x4d (decimal 77)

Format: r 77 <8 byte MAC>

All units in an 802.15.4 network have a unique 8 byte MAC address. The **MAC_Addr** command is used to program the 8 byte MAC for the iTAG.

Definitions:

<8 byte MAC> MAC address of this unit. This value is entered in reverse byte order.

Example:

Set the MAC address of this unit to 63.0.1.2.10.11.12.13.

r 77 13 12 11 10 2 1 0 63

See Also:

Last updated 11-Nov-08

***Stop_Macro*: CMD_ID=0x60 (decimal 96)**

Format: r 96 <Macro_ID>

When a user (or the iTAG) wants to stop a running a Macro, the **Stop_Macro** command is used. **Stop_Macro** will stop the Macro specified by Macro_ID. If the user wants to stop all running Macros, a Macro_ID of 0xff will stop all non-Background Macros. To stop a Background Macro configured for infinite operation, a **Stop_Macro** command is sent using the specific ID of the Background Macro. There is no problem stopping an already stopped Macro.

Definitions:

<Macro_ID>	(0-39)	Stops the Macro specified by the Macro_ID
	(255)	Stops all non-Background Macros

Example:

Stop Macro 3

r 96 3

See Also: Run_Macro, Load_Macro

Last updated 11-Nov-08

***Run_Macro*: CMD_ID=0x61 (decimal 97)**

Format: r 97 <Macro_ID> <Loop>

A User, remote command, Alarm condition, or another Macro may be used to start a loaded Macro. Macro_ID is used to identify the desired Macro to run. Macros may be run multiple times using the Loop parameter at the end of the **Run_Macro** Command. A loop value of (255,<0xff>) causes the Macro to run in infinite Background mode.

There are two special purpose Macros. Macro[38] is run each time the iTAG reboots. Background Macros and initialization parameters may be placed in Macro[38] to start each time an iTAG reboots. Macro[39] may be used as a general purpose Macro. If the iTAG is configured with a modem, Macro[39] is run when the modem establishes a connection. Any command the user wishes to execute when an iTAG modem connects should be placed in Macro[39]. Typical uses for Macro[39] are triggered data uploads, LAN network commands, or backup timers which limit connection times. If an empty Macro is commanded to run, an error message will be written to the screen.

Definitions:

<Macro_ID>	(0-39) Runs the specified Macro
<Loop>	(0-254) Runs the specified Macro up to 254 times before stopping
	255 Places Macro in an infinite loop (Background)

Examples:

Command Macro(0) to run 20 times.

r 97 0 20

Command Macro(37) to run in Background mode.

r 97 37 255

See Also:

Last updated 11-Nov-08

Load_Macro: CMD_ID=0x63 (decimal 99)

Format: r 99 <Macro_ID> <M_length> <Cmd0_len> <C00> <Cmd1_len> <C10> <C11>

Load_Macro is a simple load format for grouping iTAG commands into a single string. The string may include any iTAG commands. When loaded, the Macro may be run with a single "**Run_Macro**" command. Macros's are stored in iTAG flash memory and will not be erased when an iTAG is power down or reset. iTAG's parse the commands using the total Macro length (M_length), and each individual command length (Cmdn_len). All commands are executed when encountered in the list. Delays and branching to other Macros are accomplished using specific iTAG commands.

There are two special purpose Macros. Macro 38 is run each time the iTAG reboots. Background Macros, and initialization parameters may be placed in Macro 38 to be started each time an iTAG reboots. Macro 39 may be used as a general purpose Macro. If the iTAG is configured with a modem, Macro 39 is run when the modem establishes a connection. A command the user wishes to perform when the iTAG connects should be placed in Macro 39. Typical uses for Macro 39 are triggered data uploads, LAN network commands, or backup timers which limit connection times. If an empty Macro is commanded to run, a "Tried to Run Empty Macro" message will be written to the iTAG terminal display. If Macro 38 is empty, this message will appear with every reboot.

Definitions:

<Macro_ID>	(0-39) Macro to be loaded, Use this ID to reference when run
<M_length>	(1-128) Total Macro length, Maximum length is 128 bytes
<Cmd0_len>	(1-32) Command length for individual iTAG command 0
<C00-C0n>	iTAG command string
<Cmd1_len>	(1-32) Command length for individual iTAG command 1
<C10-C1n>	iTAG command string
<Cmdm_len>	(1-32) Command length for individual iTAG command m
<Cm0-Cmn>	Last command in iTAG command string

Example:
Load a Macro that on reboot, powers down all relays, delays 5 seconds, then starts a background Macro loaded in Macro 37. Since Macro 38 is the only Macro run on a reboot,

Load_Macro Delay_Sec
|Total Length | Run_Macro 37
| | | | |
r 99 38 13 3 8 1 0 4 100 38 0 5 3 97 37 255
| | | \
Macro_ID \Relay Cmd cmd len Macro 37->Infinite loop

Last updated 11-Nov-08

***Delay_Sec*: CMD_ID=0x64 (decimal 100)**

Format: r 100 <Macro_ID> <MSB_Delay> <LSB_Delay>

Delay_Sec inserts a delay between commands that are executing in a Macro string. Each **Delay_Sec** command is tied to a particular executing Macro with the Macro_ID parameter. When a Macro is executing a string of iTAG commands, all commands are executed until the Macro is empty or a Delay command is encountered. Typically **Delay_Sec** commands are only sent with a **Load_Macro** command, it is possible to send a **Delay_Sec** command individually to override a running delay. **Delay_Sec** is a "non-blocking" command. Other commands, serial port functions, and Macros may execute while waiting for a delay to complete.

Definitions:

<Macro_ID>	Macro which is to be delayed (0-39)
<MSB_Delay>	Delay in seconds= MSB_Delay*256 + LSB_Delay
<LSB_Delay>	(Maximum Delay is 6553 seconds)

Examples:

Load Macro 0 to delay 20 seconds, select a modem, delay 20 seconds, then enable dial-out.

```
                          Modem
                           cmd
                            |                 Enable Dial-out
                            |                       |
r 99 0 16 4 100 0 0 20 2 19 3 4 100 0 0 20 2 17 1
        |     |
        |    20 seconds
        |  delay Macro(0)
 total length
```

See Also: Delay_tSec, Delay_mSec, Run_Macro, Load_Macro

Last updated 11-Nov-08

Delay_tSec: CMD_ID=0x65 (decimal 101)

Format: r 101 <Macro_ID> <MSB_Delay> <LSB_Delay>

Delay_tSec inserts tenth of a second delays between commands that are executing in a Macro string. Each **Delay_tSec** command is tied to a particular executing Macro with the Macro_ID parameter. When a Macro is executing a string of iTAG commands, all commands are executed until the Macro is empty or a Delay command is encountered. Typically **Delay_tSec** commands are only sent with a **Load_Macro** command. It is possible to send a **Delay_tSec** command individually to override a running delay. Other commands, serial port functions, and Macros may execute while waiting for a delay to complete.

Definitions:

<Macro_ID>	Macro which is to be delayed (0-39)
<MSB_Delay>	Delay in tenth-seconds= MSB_Delay*256+LSB_Delay
<LSB_Delay>	(Maximum Delay is 655.3 seconds)

Examples:

Load Macro 1 to turn (board 1 / relay 1) "on", delay 0.5 seconds, then turn (board 1 / relay 1) "off"

```
            Relay
load        Cmd                  Relay
Macro         |                   Cmd
   |          |                    |
r 99 1 13 3 8 1 1 4 101 1 0 5 3 8 1 1 0
          |                |
          |                |         \ 5 x 0.1 seconds
          |            delay Macro(1)
    total length
```

See Also: Delay_Sec, Delay_mSec, Run_Macro, Load_Macro

Last updated 11-Nov-08

***Delay_mSec*: CMD_ID=0x66 (decimal 102)**

Format: r 102 <MS_Delay>

Delay_mSec inserts milli-second delays between commands that are executing in a Macro string. **Delay_mSec** commands are not tied to a particular Macro. When a Macro is executing a string of iTAG commands, all commands are executed until the Macro is empty or a Delay command is encountered. **Delay_mSec** commands are only sent with a Load_Macro command. **Delay_mSec** is a blocking command, other commands and Macros are suspended while Delay_mSec completes.

Definitions:

<MS_Delay>	(1-50) Delay in milli-seconds (Maximum Delay is 50 milli-seconds) (>50) Delay is DataByte[0]*Bspacing seconds

Examples:

Load Macro 1 to turn (board 1/relay 1) "on", delay 0.010 seconds, then turn (board 1/relay 1) "off"

```
            Relay
 Load       Cmd                Relay
 Macro        |                Cmd
   |          |                  |
r 99 1 11 3 8 1 1 2 102 10 3 8 1 0
          |               |      \
          |               |       10 x 0.001 seconds
          |           delay Macro
   total length
```

See Also: Delay_tSec, Delay_Sec, Run_Macro, Load_Macro, Radio_Parameters, DataByte

Last updated 11-Nov-08

Wait_While: CMD_ID=0x67 (decimal 103)

Format: r 103 <Delay Macro "n"> <until Macro "m"> <timeout>

Wait_While is a non-blocking delay command which holds execution of **Macro_ID** "n" until **Macro_ID** "m" completes. When a Macro is executing a string of iTAG commands, all commands are executed until the Macro is empty or a Delay command is encountered. **Wait_While** commands are only sent with a Load_Macro command. Serial port functions, commands, and Macros all execute while waiting for a delay to complete. A **Wait_While** command is placed in the Macro which is required to delay. Command processing will resume after the Macro_ID[m] is finished.

The **Wait_While** command initiates a 65535 count 100 mSec delay for macro n (similar to a 102 n 255 255 command). Only one macro complete value is saved for each macro. The delay is only reset when macro m completes. A **Stop_Macro** command deletes the completion flag, but does not signal the waiting macro to continue.

To illustrate these issues, say macro 5 is to wait for macro 6. If another **Wait_While** command is executed for macro 10 to wait for macro 6 before 6 completes, then macro 5 will delay until 655.35 seconds have expired because each macro can only signal one other macro after it has completed. Macro 10 will delay until macro 6 completes.

If macro 6 is stopped, then both macro 5 and macro 10 will delay until each has waited for 655.35 seconds since their respective 103 commands was executed.

There is no interaction when different macros are used for macro m, so macro 5 may wait for macro 6 and macro 10 may wait for macro 20. In each case, macro 5 and macro 10 will resume execution as expected when their respective macros complete execution.

Alternately, m may be replace with 'A', 'G', or 'U'. In each case, macro n will wait until the iTAG is Associated, GPS is acquired, or a data upload has completed. Only one macro is saved for each flag. That is, a 103 A 5 command will delay macro 5 until Associated or for timeout seconds whichever comes first. A subsequent 103 A 10 command that is executed while macro 5 is still waiting for the iTAG to become associated will cause macro 10 to delay until associated or for timeout seconds whichever comes first and macro 5 will only be waiting for the timeout second timeout and will not continue when the iTAG becomes associated. Furthermore, the associated and GPS delay states are checked every second. The upload complete delay is only terminated when an upload has ended.

Definitions:

< Delay Macro "n">	(0-39) Macro which will wait until Macro[m] completes
< until Macro "m">	(0-39) Macro which is to be waited for completion

Example:

Load Macro[4] which will start a second Macro(Macro[24]) . Macro[4] should wait until Macro[24] is complete, then save the resulting data.

r 99 4 12 3 97 24 25 3 103 4 24 **3 30 4 255 (waits until M24 completes)**
r 99 24 20 4 8 2 255 0 4 101 24 0 2 4 8 2 0 0 4 101 24 0 2

Load Macro [4] which will initiate a data upload and delay until that upload is complete or 60 seconds whichever comes first.

r 103 4 'U' 60 (waits up to 60 seconds until upload completes)

See Also: Delay_Sec, Delay_tSec, Delay_mSec, Run_Macro, Load_Macro, Upload_Data

Last updated 11-Nov-08

LCD_CMD: CMD_ID=0x8c (decimal 140)

Format: r 140 <cmd> <Params>

Definition:

Example:

See Also:

Scale_Factors: CMD_ID=0x95 (decimal 149)

Format: r 149 <Index> <Scale> <Offset>

Definition:

Example:

See Also:

Labels: CMD_ID=0x96 (decimal 150)

Format: r 150 <Index> <displayString>

Definition:

Example:

See Also:

LCD_Strings: CMD_ID=0x97 (decimal 151)

Format: r 151 <Index> <displayString>

Definition:

Example:

See Also:

Text_Msg: CMD_ID=0xc8 (decimal 200)

Format: r 200 [‘#’] <“Text String to be Sent”>

Text_Msg is a command which places received text messages into battery backed RAM. The text message buffer is 20 bytes x 40 bytes. Each message is placed on a new line when a carriage return is encountered in the text message. The User at the receiving iTAG may review the text message by using the Text Message Display option.

If the first byte after command 200 is a ‘#’ character, then program the unit ID instead of the Text Message.

Definition:

“Text String”	Any ACSII character string not exceeding 40 characters per line String should terminate with a carriage return.

Example:

Send a text message.

r 200 “ET Phone Home!”

See Also:

Last updated 11-Nov-08

Reset_Buffer: CMD_ID=0xca (decimal 202)

Format: r 202 <Buffer_ID>

iTAGs' utilize numerous storage buffers for commanding, network traffic, and status changes. These buffers reside in flash or RAM. If a User wishes to clear a particular buffer, the **Reset_Buffer** command may be used.

Definitions:

<Buffer_ID>	0x00	Transmit Buffer
	0x04	Clear stored data in Flash (if stored data saved in flash)
	0x05	Clear Macros
	0x06	Clear Stored Commands
	0x08	Clear stored data in RAM (if stored data saved in RAM)
	0x09	Stored Text Messages

Example:

A remote iTAG has its Cmd_Attempt count set too high (255). Since the number of retries is so high, packets in it's transmit buffer could take many minutes to clear. Send a command to clear the iTAG's transmit buffer, then reset the Cmd_Attempt count.

r 202 0

See Also: Load_Macros

Last updated 11-Nov-08

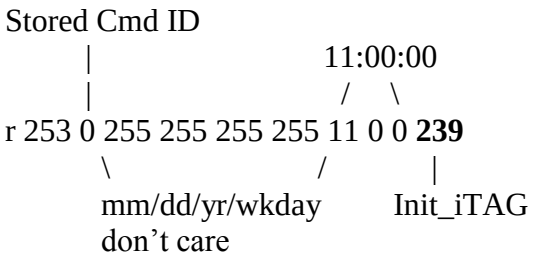
Init_iTAG: CMD_ID=0xef (decimal 239)

Format: r 239

iTAGs' may be commanded to restart without performing a hardware reset or commanding a restart from the Setup Menu. Commanding an **Init_iTAG** forces the iTAG to restart using all initialization parameters and data stored in flash memory. If **Init_iTAG** is commanded, all currently running Macros will be stopped, serial ports will be re-initialized, and Macro 38 will be run as if a hard reboot occurred. Caution should be observed when commanding **Init_iTAG**. If an iGATE is commanded to re-initialize while connected via a modem, the modem will be commanded to hang-up. At this point communication with the iGATE would be lost until a re-connection is commanded.

Example:

Load a scheduled command to re-initialize the iTAG once per day at 11:00:00 AM.



See Also: Load_Defaults, Commit_Flash, Super_Reset

Last updated 11-Nov-08

Commit_Flash: CMD_ID=0xf0 (decimal 240)

Format: r 240

A majority of iTAG configure parameters are accessed from RAM but are stored in flash. Once an iTAG is configured by a User, parameters that are set during configuration should be saved to flash. The Setup Menu provides an approach to commit to flash. If a configuration load file is used, the last command in the file should be a **Commit_Flash.** If parameters are changed via network commands, a **Commit_Flash** should be sent after all parameters have been changed. Execution of the **Commit_Flash** command may take up to 2 seconds. Commands sent immediately following the **Commit_Flash** should be delayed by a few seconds, otherwise they may not be properly processed.

Example:

Build a configuration load file which sets several iTAG parameters. Commit the changes to flash using **Commit_Flash.**

Load File:

```
r 25 'Z'                // Select Radio
r 19 'P'                // Select  PSTN bus modem
r 27 "18005551234"      // Set Dial-out number
r 240                   // Commit to flash
```

See Also: Load_Defaults, Init_Device, Super_Reset

Last updated 11-Nov-08

Load_Defaults: CMD_ID=0xf1 (decimal 241)

Format: r 241

To ensure minimal functionality during an iTAG reboot, default configurations are programmed into the iTAG flash. Many of these parameters are over-written during the Set-up process. If a User wishes to restore the factory defaults for an iTAG configuration, the **Load_Defaults** command is sent. The User should exercise caution in using **Load_Defaults**. Depending on the iTAG configuration, the User may lose communication capability if this command is sent over the Network. If a User resets the iTAG configuration over the Network, the User can restore Network operation parameters only through the Setup Menu on the iTAG.

Example:

A User has corrupted an iTAG configuration when performing a firmware upload. Many of the parameters in flash may be corrupted. To restore the iTAG parameters to the factory defaults send the **Load _Defaults** command.

r 241

See Also: Commit_Flash, Super_Reset

Last updated 11-Nov-08

Load_Flash: CMD_ID=0xf2 (decimal 242)

Format: r 242

During start up, an iTAG copies configuration data from flash into a global array in RAM. While the iTAG is executing, the configuration data is accessed from the RAM parameters. If the RAM variables are corrupted via a command or internal RAM errors, **Load_Flash** will reload the global configuration data with data in Flash.

Load_Flash is also performed by **Init_Device** except the iTAG is not re-booted, serial ports are not initialized, and Macros are not stopped.

Example:

A near-by lighting strike has effected an iTAG configuration. Many of the parameters in RAM may be corrupted resulting in a loss of communication with the iGATE. The following command (if residing) in the Stored Command buffer will execute once per hour (at the top of the hour) to restore the Flash configuration to RAM.

r 253 10 255 255 255 255 255 0 0 **242**

See Also: Commit_Flash, Load_Defaults, Init_Device

Last updated 11-Nov-08

Set_Password: CMD_ID=0x1c (decimal 248)

Format: r 28 “<p0><p1><p2><p3>….<pn>

The iTAG Set-Up Menu and ASCII command functions are password protected to prevent accidental or deliberate commands from being executed. **Set_Password** changes the default access password to a User configurable string. The maximum password length is nine (9) characters. Spaces and control characters are not valid password characters. Use caution when changing the password string. If a boot macro enables password checking, sending a command to change the password will not be possible if a User forgets the updated value. Make sure to use **Commit_Flash** to save the new password in flash**.**

The Factory default password settings are:

Password:	“icontrol”.
SET-Up Menu:	Password Required
ASCII Commanding:	No Password Required

See Also: ASCII Commands, Password, User Menu

Last updated 11-Nov-08

DataByte: CMD_ID=0xf9 (decimal 249)

Format: r 249 <Index> <Value> [Params]

iTAG's provide a global data structure which may be used to build user specified data packets**.** **DataByte** is a 40 byte structure which may be loaded with data which is sent, received, or alarmed. The **DataByte** command is used to set the value of a particular **Databyte**, or if requested, increment or decrement the current **DataByte** value. A User should make sure that other Macro's or commands are not affecting the use of a particular **DataByte**.

Definitions:

<Index>	(0-39)	Index identifying which DataByte[] to modify
<Value>	'+'	Current value of DataByte[index] is incremented by one
	'-'	Current value of DataByte[index] is decremented by one
	's'	Copy command Params starting at DataByte[Index] to DataByte[Index + number of Params]
	'c'	copy Packet data into DataByte[Index] P0-P1: Packet ID bytes (see Appendix C) P2: start byte in above data packet to start copy from P3: number of bytes to copy
	(0-255)	If an integer is sent, DataByte[index] will be assigned the new value)

Example:

Use **DataByte**[3] as an internal counter that is incremented every time Macro [4] is run.

Load into Macro[4]:

r 99 4 "Macro Commands……" 249 3+

See Also: Load_Macros, Run_Macros

Last updated 11-Nov-08

Check_Alarm: CMD_ID=0xfa (decimal 250)
Check_Alarm_Save: CMD_ID=0xfb (decimal 251)

Format: r 250 <Board Add> <Extended Add> <Index> <Op> <value> [value2] <Macro_ID>

Scheduled or background commands may be used to check iTAG input data or system configuration for alarm conditions (or desired events). **Check_Alarms** retrieves data using the specified Board Address and Extended Address, performs a comparison using the Op character against the value parameter. If the Op condition is met, Macro[Macro_ID] will be run once.

The **Check_Alarm_Save** command saves the retrieved data if the Op condition is met.

Definitions:

<Board Add>	IO card address (See **IO_Add**)	
<Extended Add>	IO card address (See **IO_Add**) 5, 250, 251: copy native floating point for test value 83, 254: copy one byte into floating point test value Default: format floating point test value using sFactors[Index] (See Scale_Factors)	
<Index>	0-12	Data[index] is value to be checked
<Op>	>	Runs Macro[ID] if Data[index]>value
	<	Runs Macro[ID] if Data[index]<value
	=	Runs Macro[ID] if Data[index]=value
	!	Runs Macro[ID] if Data[index]!=value
	"A"	Run Macro[ID] if Data[index] < value && Data[index] > value2
	"O"	Run Macro[ID] if Data[index] & value || Data[index] > value2
	"&"	Run Macro[ID] if Data[index] & value
	"|"	Run Macro[ID] if Data[index] | value
<value>	"Threshold" value which triggers Macro[ID]	
<value2>	Second Threshold value	
<Macro_ID>	ID of Macro which is run on an alarm condition	

Examples:

Load Macro[37] with a **Check_Alarm** command that calls Macro[36]. Macro[37] will be a Background Macro which checks data once per second. The **Check_Alarm** command executes when motherboard analog[1] is greater than value "128".

Default scale factors are used which indicate to use counts for comparison.

r 99 36 7 3 96 37 2 17 1 Stop Macro[37], Enable Dialout
r 99 37 13 4 100 37 0 1 7 **250 56 12 1> 128 36** **Check_Alarm**
r 99 38 3 97 37 255 Runs Macro[37] on boot up.
Loop counter 255 places Macro[37] in an infinite loop

See Also: IO_Addr, Stop_Macro, Delay_Sec, Load_Macro, Scale_Factors

Last updated 11-Nov-08

Server_Addr: CMD_ID=0xfc (decimal 252)

Format: r 252 <D3> <D2> <D1> <D1>

Server_Addr programs the four byte Server identification number used by an iGATE for Internet connections. This ID may also be programmed from the Setup Menu. **Server_ID is used only if the iTAG is configured as an iGATE.** If the iTAG is not configured as an iGATE, the Server address is ignored.

Description:

<D3> Most Significant Byte of Server address
<D2>
<D1>
<D0> Least Significant Byte of Server address

Example:

Program the **Server_Addr** to 3.4.5.6

r 252 3 4 5 6

See Also: Broadcast_Address, Default_Dest, iTAG_ID

Last updated 11-Nov-08

Load_Schedule: CMD_ID=0xfd (decimal 253)

Format: r 253 <Index> <M> <D> <Y> <WkD> <Hr> <Min> <Sec> <cmd……..>

Load_Schedule places iTAG commands into an iTAG command buffer used for scheduled execution. The buffer holds up to 40 commands which are continually scanned for a valid execution time. The index into the stored command is used for loading only. Execution of the commands is determined by matching time tags with the current iTAG clock. If multiple commands are to be called at the same time, use a **Run_Macro** command to link the commands into a single call from the check schedule function.

Description:

<Index>	(0-39)	Index into storage buffer	
<M>	(1-12,255)	Desired month for scheduled command,	255= don't check
<D>	(1-31,255)	Desired day for scheduled command,	255= don't check
<Y>	(0-99,255)	Desired year for scheduled command,	255= don't check
<WkD>	(0-6,255)	Desired week day for scheduled command,	255= don't check
<Hr>	(0-23,255)	Desired hour for scheduled command,	255= don't check
<Min>	(0-59,255)	Desired minute for scheduled command,	255= don't check
<Sec>	(0-59,255)	Desired second for scheduled command,	255= don't check

Example:

Load a command into the Schedule buffer location (0) to perform a **Save_Data** command once per minute (top of each minute).

```
      Stored Index                  seconds
         |                             |
r 253 0 255 255 255 255 255 255 0 30 56 12
                                       |
                                   Save Data
```

This command loads "don't cares" in all time fields except the (seconds) field. The iTAG will ignore this command until the iTAG clock (seconds) matches the "seconds" time field. Every time the iTAG seconds equals "0" the command **Save_Data** (30)(56)(12) will be executed.

See Also: Load_Macro, Dump_Schedule

Last updated 11-Nov-08

***Super_Reset*: CMD_ID=0xff (decimal 255)**

Format: r 255

In the event that an iTAG configuration has become corrupted to the point of non-operation, (continual reboots), or loss of communication, a unit may be commanded to perform a **Super_Reset**. A **Super_Reset** restores all protected variables, performs a **Load_Defaults**, and reboots the system immediately after the restoration of the default configuration. Users should exercise caution when sending the **Super_Reset**. Loss of communication across the iTAG LAN may result if a **Super_Reset** is performed using a LAN command.

Example:

A User has performed a series of configuration changes which results in an iTAG not being able to successfully complete a reboot. Typical errors which may cause this are corrupted protected variables, or boot-up macros that run in a background mode (high speed infinite loop)

While the iTAG attempts to reboot, load and send a **Super_Reset** command to clear all buffers and restore the protected variable values. Once sent, the iTAG should be able to boot correctly and allow the User to correct the boot problems.

r 255

See Also: Load_Defaults

Last updated 11-Nov-08

Appendix A

Security:

Password Protection:

The only level of protection available for ASCII commanding is password protection. The factory default protection allows password access for the Set-Up Menu and open access to ASCII commanding. To enable password access for commanding, the User should load and enable a "Null" Password command in the boot up Macro (Macro[38]). Once the iTAG is rebooted, password protection will be enabled. When password protection is set, a **Password** command must be sent prior to sending any ASCII command. The **Password** command enables commanding until a "Null" **Password** command is sent, the iTAG reboots, or until a Stored Command executes a "Null" **Password**.

Encryption:

A majority of the wireless devices iControl provides include built in encryption. These devices enable a DES encryption algorithm independent of the iTAG OS setting. Setting the radio encryption key is covered by the Radio Manufacturers User Manual included with your purchase.

In addition to the radio encryption, iControl offers both LAN and iGATE encryption algorithms that may be used over cellular and satellite links. iControl ships each iTAG unit with encryption disabled (a zero encryption key). Use the **Net_Type** command to set the iTAG OS encryption key. In addition to the encryption key, Users should be aware of **Time_Window**. **Time_Window** specifies the valid duration of time for an iTAG command. **Time_Window** reduces the possibility of recorded (even if encrypted) commands being recorded and played back by un-authorized parties.

See Also:

Password, Set_Password, Time_Window, Net_Type

Appendix B

Cyclical Redundancy Code (CRC):

The following 'C' source code may be utilized by Users to develop binary protocols for interfacing to iTAG's and iGATE's. Two bytes are returned which represent the 16 bit CRC for any input string. iTAG's apply CRC's to all binary communication packets.

```
void getCRC(char *buf, int len, char *value)
{
        //CCITT CRC
        // buf: pointer to string which is to have CRC computed
        // len: Number of bytes in buf which are used for CRC computation
        // value: Two bytes representing 16 bit CRC
        //
unsigned int crcval, temp;
int i;
         crcval = 0xffff;
         for (i=0; i<len; ++i)
                {
                temp = crcval ^ (unsigned int)buf[i];
                temp = (temp ^ (temp << 4)) & 0xff;
                crcval = (crcval >> 8) ^ (temp << 8) ^ (temp << 3) ^ (temp >> 4);
                }
        crcval = ~crcval;
        value[0]=       (char)(crcval  &  0x00ff);              // return LSB (sent first)
        value[1]=    (char)((crcval & 0xff00)>>8);              // return MSB (sent second)
}
```

Appendix C

I/O Packet Definitions:

Many iTAG commands use I/O data packets to make decisions or to format the information that the user is interested in seeing. Below is a table showing each data packet and the information that is in each one.

Data packets are defined by a two byte address. Each data packets unique data starts at byte 12 and they all have the following bytes inserted at the beginning.

Byte	0	1,2	3-9	10	11
	Length	Packet ID	Time	TCW	Mode – see iTAG_Mode

Packet ID 0x00 0x0c
Network Beacon status from an iGATE to iVIEW
Length: 32

Byte	12,13	14,15	16-27	28	29	30	31	32
	Beacon Timer	Comm Timer	Native floating point Latitude, Longitude, Altitude	Radio Sleep Channel	Present Radio Channel	Radio LQI	Number of units in ACL	Difference in the number of unit in ACL since last beacon sent

Packet ID 0x00 0x10
Packet ID 0x00 0x20
Data read from serial port 1 (0x00 0x10) or serial port 2 (0x00 0x20)
Length: variable
The data in this packet depends on the fsync parameters (see Serial_R/W). There are three fsync values. If either of the first two fsync values is non-zero then wait for those two bytes in succession before saving data. Once those two bytes are read, start saving data until either the input UART is empty or until the maximum stored data buffer size is reached or until the third fsync byte value is found (if the third fsync byte value is non-zero.
Note: You must allow enough time for the expected data to fill the UART as this read process does not allow for any timing. It simply reads as fast as it can, terminating when no data is retrieved. You must also insure that you do not overflow the UART buffer.

Byte	12	Last byte
	First byte after two byte sync or first byte read if	Last valid byte read from UART if third fsync value is zero or the last byte

	first two bytes of fsync are zero	read from UART which matches the third fsync byte

Packet ID 0x00 0x04
Tag Tracking and Status packet
Length: 23

Byte	12,13	14,15	16.17	18	19	20,21	22,23
	Nav mode (native unsigned int format). Only first byte is valid. Second byte is always zero	LastFix variable (native unsigned int format). Reserved for future use	Comm. Timer (native unsigned int format). Seconds to remain associated.	Count of visible satellites in the GPS constellation	Reserved for future use.	Magnitude of the GPS velocity vector. (native unsigned int format)	Moved counter variable (native unsigned int format). Used in motion sensing.

Packet ID 0x00 0x05
Distance to each waypoint in meters. There are 5 available waypoints in the present Gen 2 iTAG. This value will change depending on need and memory constraints. This algorithm outputs the difference between the last recorded position and the waypoints saved using the Track_Mode command. The last recorded position is set by executing any command that retrieves a GPS packet – Packet IDs 0x00 0xfa, 0x00 0xfb, 0x00 0xfc.
Length: 11 + (4*(<num defined waypoints>-1))

Byte	12-15	…	12+(4*(<num defined waypoints> -1)) - 12+(4*(<num defined waypoints> -1))+3
	Floating point Number representing magnitude of the distance between last recorded position (present position) and waypoint 0		Floating point Number representing magnitude of the distance between last recorded position (present position) and waypoint N

Packet ID 0x00 0x06
iTAG configuration data
Length: 78

Byte	12-24	25-28	29-32	33-36	37	38	39	40	41
	Version String	**iTAG_ID**	**Default_Dest**	**Server_Addr**	Device Type (**iTAG_ID**)	Pri LAN Port (**Port_Cfg**)	Alt LAN Port (**Port_Cfg**)	Modem Port (**Port_Cfg**)	Data Port (**Port_Cfg**)

Byte	42-44	45	46	47	48	49
	Baud rate of ports 0,1,2. Value=Baud/12	Reserved for future use	Tpreamble/4 Not Used	TXTail (**Radio_Parameters**)	Present radio channel (**Modem_Parms**)	Nav Mode. See Chip doc
Byte	50	51	52	53	54	55
	Power_Save	Wake Power Config (**Power_Config**)	Sleep Power Config (**Power_Config**)	Present Power Config (**Power_Config**, **Power_Override**)	Radio Status Byte. See chip doc	Lower 8 bits of comm. Timer
Byte	56	57	58	59	60	61
	Reboot counter – incremented with each soft reboot	TX_Attempts (**CMD_Retry**)	TX Retry Time (**CMD_Retry**)	Default Channel (**Modem_Parms**)	Sleep Channel (**Modem_Parms**)	Primary Radio (**LAN_Device**)
Byte	62	63	64	65	66	67
	Alt Radio	**Enable_Schedule**	**Modem_Type**	Modem State	**AT_Answer**	**AT_Dial**
Byte	68	69	70	71	72-76	77,78
	Data Port User Interface flag (**Port_Cfg**)	Flag indicating flash has been initialized	**Network**	'E': Encryption enabled 'N': Disabled	Bit mask indicating Macro is running or not	ACL Timer (native unsigned int format)

Packet ID 0x00 0x17
Broadcast IDs set with command 23
Length: 12 + (4*<number of broadcast groups>)

Byte	12	13-16	[13+(n*4)] – [16+(n*4)]
	Number of valid Broadcast groups (n)	Broadcast Group 1	Broadcast Group n

Packet ID 0x00 0x22
Radio initialization parameters set with command 34
Length: 29

Packet ID 0x00 0x1b
Telephone number used for modem communication. Number starts at byte 12. Length does not include null terminator.
Length: 11 + length of telephone number

Packet ID 0x00 0xfa

Difference between present GPS position and last recorded iGATE position. GPS solution must be valid, otherwise length of data packet is zero.
Length: 42

Byte	12-15	16-19	20-23	24-27	28-31	32-35	36-39	40,41	42
	Difference in meters between iGATE latitude and latitude of present location	Difference in meters between iGATE longitude and longitude of present location	Altitude – indeterminate value	Northern Velocity as measured by present GPS solution	Eastern Velocity as measured by present GPS solution	Downward Velocity as measured by present GPS solution	Magnitude of distance between iGATE position and present location in meters. Ignores Altitude.	GPS Nav Mode. See Chip documentation for description.	Radio LQI

Packet ID 0x00 0xfb
Packet ID 0x00 0xfc
GPS position. If GPS solution is invalid and associated with an iGATE, report the iGATE position. If GPS solution is invalid and not associated with an iGATE, length of data is zero.
Length: 42

Byte	12-15	16-19	20-23	24-27	28-31	32-35	36-39	40,41	42
	Lattitude in degrees	Longitude in degrees	Altitude	Northern Velocity as measured by present GPS solution	Eastern Velocity as measured by present GPS solution	Downward Velocity as measured by present GPS solution	Magnitude of the present velocities	GPS Nav Mode. See Chip documentation for description.	Radio LQI

Packet ID 0x00 0xfc (Deprecated)
Floating point State Variables
Length: 42

Packet ID 0x00 0xfe
DataByte variables. . The value of DataByte[0] defines the number of DataByte values to output in this packet.
Length: 11 + DataByte[0]

Byte	12	…	12+DataByte[0]
	DataByte[1]		DataByte[DataByte[0]]

Packet ID 0x20 0xXX (Deprecated)
Address of parallel processor connected via serial port

Packet ID 0x01 0xNN
Packet ID 0x30 0xNN (Deprecated)
Packet ID 0x38 0xNN
Status information including Analog-to-Digital converter data. NN defines the number of bits in each 2-byte A/D value. This NN value is only used when decoding the A/D values as it records the maximum counts for the A/D values, it is not used by internal firmware. Although the connections to the individual A/D channels is totally dependent on hardware connections, A8 (A0 being the first channel) is connected to a voltage divider indicating the voltage powering the unit.
Length: 42

Byte	12-35	36	37	37	37	38	39	40	41	42
	12 channels of A/D data (2-bytes each)	Reboot counter – incremented with each soft reboot	Lower 5 bits of comm. timer	Bit 6 – flag indicating in range of an 802.15.4 radio network	Bit 7 – flag indicating that there is stored data	Clock source 0: SW 1: RTC 2: GPS	Last DataByte value	Radio LQI	Last RSSI indicated from an attached modem	'N' - not commissioned 'Y' – commissioned

Last updated 12-Nov-08