Listypes and Idents

Request data types

Mon, Feb 23, 1998

The data request protocol used in local stations/IRMs includes a field that is used to indicate the type of data requested, referred to as a "listype." This same specification is also used in the data setting message format. This note is a summary of the meanings of the various listype numbers. Given a listype, the "ident" is used to identify which device is targeted. Since each listype implies a specific ident type, the following description of listypes in use is organized around the ident types that groups of listypes imply. The listype number is shown, followed by the data size, followed by the maximum data bytes allowed in a setting. The data size used for a data *request* has no limit, although the use of values larger than the data size shown may place undue dependence on the exact layout of internal data structures. The number of bytes of data used for a *setting* may not exceed the maximum value shown, or the setting message will be ignored. Where the max #bytes is zero, no setting is valid for that listype. Where the data size is zero, the listype is not suitable for data requests, but is meant for use with settings only.

Analog channel number

The first ident type to be described is the analog channel, by far the most common. In IRMs, its values are usually in the range `0000`–`03FF`. The ident format is two words. The first word is the node number; the second is the channel number within that node.

Listype	*Data size*	*Max #bytes*	*Meaning*
0	2	0	Reading
1	2	2	Setting
2	2–8	6	Nominal value for alarm scanning
3	2	2	Tolerance value for alarm scanning
4	4	2	Alarm flags, counter
5	1–4	0	Associated digital status
6	2	0	Motor countdown word
7	2	2	Relative setting (#steps for motor)
8	4–62	62	Analog descriptor spec
9	4	4	Associated digital status spec
10	6	6	Associated digital control spec
11	2	2	Conversion flags
12	16	16	Scale factors
13	18	18	Descriptive text
14	6	6	Associated digital status/control text
15	6	0	Name text
16	4	4	Units text
17	2	2	Family word
18	2	0	Date descriptor last updated
22	0	2	Associated digital control
27	2	0	Captured reading
28	2	2	Spare analog data word (n.u.)
39	0	2	Delta setting
40	4	0	Reading (engineering units)
41	4	4	Setting (engineering units)

42	4	4	Nominal (engineering units)
43	4	4	Tolerance (engineering units)
44	4	4	Delta setting (engineering units)
49	n	0	Family of channels
57	10	10	D0 analog alarm control
61	30	30	D0 Analog Alarm Device Info Block
62	16	0	D0 Analog 16-character name
63	2	0	D0 Date of last change in AADIB
83	2	2	Set/clear good/bad alarm flag bit

Listype 2 specifies a data size of up to 8 bytes. This allows access to all analog channel alarm information together: the nominal, tolerance, flag word, and alarm count, as they are in sequential fields in the analog data table entry structure.

For the case of listype 4 that accesses the analog channel alarm flags, one may read up to 4 bytes, in which the first word is the alarm flags word and the second the alarm count. But only the alarm flags word can be set.

Listype 6 accesses associated digital status. One may request up to 4 bytes, which would yield the following structure:

associated status
active flags
bad alarm status
inhibit alarm status

For each byte returned, the relevant data bits occupy only the most significant two bits of the byte. The most significant bit is used for associated status bit #1, and the next most significant bit is used for associated status bit #2. The meaning of associated status depends upon information stored in the associated digital status field of the descriptor. The general purpose is to associate one or two bits of digital status with an analog channel. As an common example, one can support on/off status with one status bit, and interlocks tripped status with a second bit. With associated digital control enabled, one can include on/off control (via a pair of pulsed control lines) and reset interlocks control. For more on associated status and control bits, see the document Digital Control Pulse Delays.

For listype 7, the support for "relative setting" was designed to work with a knob. The amount of adjustment to a D/A for a minimum one-step knob change depends upon the type of analog control specified for that channel. If it is a 12-bit D/A, the knob counts are shifted up 4 bits before being added to the current setting value, since setting (and reading) values are internally maintained as a left-adjusted fraction of full scale. When this logic seems unsuitable, use the listype 39 instead, in which the setting data is the exact raw change desired, and no shifting is done before adding into the current setting value. (Listype 44 supports the engineering units equivalent of listype 39.)

Listype 8 allows access to the analog control field of the analog descriptor, but it can also be used to access the entire descriptor, as the analog control field occurs first. Request 64 bytes to read the entire structure, or set up to 62 bytes to change it. (The last

2 bytes are automatically updated to the date-of-last-change when a setting is made to any of the descriptor structure; they are not independently settable.)

Listypes 8–18 merely access the various fields of an analog descriptor entry. Listype 12 accesses the linear scale factors used for engineering units conversion. The four IEEE floating point values are as follows:

Reading full scale
Reading offset
Setting full scale
Setting offset

All values are in units of the engineering units used for the device. The scaling formula is as follows:

```
eng = (v/10)*fScale + offset;
```

Here, `eng` is the engineering units result, `v` is the voltage value in the ±10 volt range, `fScale` is the full scale value, `offset` is the offset value. Since raw values are stored as 16-bit word that is a left-adjusted fraction of full scale, from –10 volts (`8000`) to nearly +10 volts(`7FFF`), and the raw value is taken as an integer, the voltage value above is

```
v = raw*10/32768;
```

or

```
v = raw/3276.8;
```

To derive a raw setting value, given a desired engineering units value, merely work the formulas backwards. Note that the engineering units formula for the tolerance value does not use the offset scale factor.

Listypes 40–44 support engineering units access to analog data. The scale factors in the analog descriptor are used to perform the above-mentioned linear scaling.

Digital bit number

The binary bit number is the digital analog to the analog channel. In IRMs, its values are usually in the range `0000–03FF`. The ident format is two words. The first word is the node number; the second is the bit number within that node.

Listype	*Data size*	*Max #bytes*	*Meaning*
21	2	2	Digital bit I/O
23	16	16	Digital bit text
24	4	2	Digital alarm flags, count
58	2	2	D0 Binary alarm control
64	30	30	D0 Binary Alarm Device Info Block
65	16	16	D0 Binary 16-character device name
66	2	0	D0 Date-of-last-change in BADIB entry

Use listype 21 to perform digital control at the bit level. The data that is read in reply to a data request is one byte, whose value is either `00` or `01`. If two bytes are requested, the first byte is of this same form, and the second is the value of the digital byte from which the single bit was sampled—probably not very useful. When sending a setting with this listype, use a 2-byte data value as follows:

Data	*Meaning*
0000	(no control)
0100	Toggle bit to opposite state
0200	Set bit = 1.
0300	Set bit = 0.
04xx	Pulse bit high, where xx=pulse width in cycles.
05xx	Pulse bit low.
06xx	Pulse one of a pair of bits high.
07xx	Pulse one of a pair of bits low.
0Cxx	Same as 04xx, but pulse terminated by hardware.
0Dxx	Same as 05xx, but pulse terminated by hardware. 0Exx Same as 06xx, but pulse terminated by hardware. 0Fxx Same as 07xx, but pulse terminated by hardware.

Where a pulse width is specified in cycles, a value of 00 means a short pulse, intended to be 20 μs. A value of 01 may last less than a full cycle, depending upon when the setting was received. Use 02 to guarantee at least one full cycle of pulse width. For more details, see the document Digital Control Pulse Delays.

Memory address

Listypes that use the memory address ident provide access to memory space. The 6-byte format is a node number word followed by a 32-bit memory address.

Listype	*Data size*	*Max #bytes*	*Meaning*
20	n	n	Memory access by bytes
29	n*2	n*2	Memory access by words
38	n*2	n*2	FIFO memory access by words
46	n*2	n*2	D0 Memory access w/ MMAPS (words)
72	n*4	n*4	Memory access by longwords
73	n*4	n*4	D0 Memory access by floating point
77	n*2	n*2	Memory access by words w/verify

Analog 6-character name

Only a single listype uses the 6-character name ident type, which is unique in having no node number at the beginning of its format. The name consists only of six ascii characters, blank filled for a name with less than 6 characters.

Listype	*Data size*	*Max #bytes*	*Meaning*
19	4	0	Look up name, return channel# ident.

The format of the returned data is the usual 4-byte channel number ident, with the node number followed by the channel# index. Note that only a single ident of this form may be included in one data request. (If many names need to be looked up, send a separate request, each with a distinct list number, so that the list number included in the reply will indicate to which name the reply refers.) A request for name look up would likely be multicast/broadcast to the network so every node has a chance to search for a match in its internal analog descriptor table. Only a node that finds a match will return a reply to such a request.

Digital byte number

The digital byte number ident also parallels the analog channel number. The data referenced by a byte number is the same as is referenced by 8 consecutive digital bit numbers; for example, byte number `001C` includes bit numbers `00E0`–`00E7`, with bit number `00E7` the most significant bit of the data byte.

Listype	*Data size*	*Max #bytes*	*Meaning*
25	n	2	Digital byte I/O via byte number
34	4	4	Digital byte address table entry
71	n*2	8	Digital word I/O via byte number

Listype 34 allows access to the table of digital byte addresses that is used when updating the digital data pool.

Global system variables

The IRM system software uses a global data structure to hold many pieces of general useful parameters. This listype accesses this structure via a 4-byte ident whose first two word is the node number and the second word is an offset applied to the base global variable pointer. The offset value is often negative.

Listype	*Data size*	*Max #bytes*	*Meaning*
26	n	0	Global system variable access

While it is beyond the scope of this note to describe all relevant offset values, a few that may be of interest are listed here:

Offset	*Name*	*Meaning*
–120	`cycleCnt`	Longword cycle counter
–112	`myNode`	Local node number word
–110	`cpu`	CPU type byte: 20=68020, 40=68040
–109	`cpuBoard`	CPU board number byte: 133, 162
–104	`udpNode`	Node number word used by Acnet for UDP/IP
–75	`cycLength`	Length of cycle (unsigned byte)
–44	`maxContM`	Maximum contiguous free memory longword
–26	`options`	Option switches reading (byte)
–8	`uLites`	Units lights byte for little console
–7	`mLites`	Mode lights byte for little console
–6	`uMask`	Mask of enabled units lights
–5	`mMask`	Mask of enabled mode lights
–4	`cursCol`	Cursor column position for little console
–2	`cursRow`	Cursor row position for little console

Data access table entry

The data access table entry number ident is of the typical 4-byte form, with the node number in the first word and the table entry number in the second word. The data access table is a simple array of 16-byte entries, the number of which can be obtained via a table directory access for entry #4. IRMs normally have 128 entries in the data access table, giving a total size of 2K bytes. One can read or write the entire 2K bytes in one access.

Listype	*Data size*	*Max #bytes*	*Meaning*
30	n*16	n*16	Digital byte I/O via byte number

Page number

The page number ident refers to the little console interface that can be accessed over the network via "Page-G" support. The little console display consists of a 16-line by 32-character display, which adds up to only 512 characters in all. This makes it easy to monitor and emulate the display for such remote access. A page application executes in the IRM and maintains a display in memory, reacting to the values of certain global variables. This support allows invocation of a page application from an index page, along with a simple user interface based upon character-based cursor position.

Listype	*Data size*	*Max #bytes*	*Meaning*
31	4	20	Page name, title
32	16	16	Page 16-character title
33	128	120	Page-private memory
37	8	8	Auto-page parameters
48	0	2	Demand auto-page invocation

Serial port I/O

The serial port 4-byte ident format consists of a node number and a port number. The current implementation only supports a single port that is referred to as port zero.

Listype	*Data size*	*Max #bytes*	*Meaning*
36	n	132	Serial port I/O w/ editing
84	n	n	Serial port I/O w/o editing

Listype 36 setting support includes editing, which means that CR and LF character is appended to the text in the setting data buffer before it is sent to the serial port. This means that a single setting should include the text for one line. Listype 84, in contrast, includes no editing of this sort, so that all characters in the setting buffer are delivered to the serial port with nothing extra added. Note also that all serial output is buffered; the setting is considered complete even though not all characters have yet been delivered to the port.

In either case, serial input ignores NUL and LF characters. A line of serial input is limited to 250 characters, by which point a CR should be received, or else one is assumed.

Data stream number

A data stream is a formalized circular buffer that permits multi-user access without interference. The 4-byte ident form consists of the node number followed by the data stream number, which is an index into the data stream table.

Listype	*Data size*	*Max #bytes*	*Meaning*
50	4+m	n	Data stream records (future)
51	4+m	n	Data stream records (most recent)
52	n	0	Data stream queue header
53	n	32	Data stream table entry
54	8	8	Data stream name
78	4+m	n	Data stream records (from oldest)

Data stream records may be accessed in three different ways. In each case, a requester should specify a buffer size that includes room for a two-word header in addition to the space for the maximum number of records to be returned in one reply. The first word in the reply is the count of the number of records to be found in the buffer. The second word indicates the size of each record, in which zero is given if the records are of variable size. Listype 50 would likely be used with a periodic request. Only records are returned to the user that are written into the queue after the request has been initialized. Listype 51 would likely be used in a one-shot data request. It fills the user's buffer with as many records as will fit, ending with the most recent record. Listype 78 would be used in a periodic request and is designed to read out all the records in a queue, even if they were too large for a one-shot request. The first reply fills the buffer with as many records as can fit starting with the oldest in the queue. Subsequent replies continue where the first reply left off. In all three cases, a setting writes into the queue as many records as are indicated by the queue record size specification. If the queue is configured for variable size records, the setting data buffer is considered to be a single record. Additional details can be found in the document Data Streams.

D0 Analog 16-character name

For use with the D0 protocol, one can form a data request to look up the name. This is very similar to the case for the 6-character analog channel name. But this ident format is 18 bytes long, and it includes a node number as the first word. The rest of the ident consists of 16 ascii characters, blank filled for a name whose length is less than 16 characters.

Listype	*Data size*	*Max #bytes*	*Meaning*
55	4	0	Look up name, return channel# ident.

The format of the returned data is the usual 4-byte channel number ident, with the node number followed by the channel# index. Only a single ident of this form may be included in one data request. (If many names need to be looked up, send a separate request, each with a distinct list number, so that the list number included in the reply will indicate to which name the reply refers.) A request for name look-up would likely be multicast/broadcast to the network so every node has a chance to search for a match in its internal analog descriptor table, in which case the node number word would be zero. Only a node that finds a match returns a reply to such a request.

Comment alarm number

The comment alarm number is used for alarm messages without a good/bad state. They simply indicate that some event transpired of importance. The ident format is two words. The first word is the node number; the second is the comment number index.

Listype	*Data size*	*Max #bytes*	*Meaning*
56	32	2	Comment data
59	2	2	D0 Comment alarm control
67	30	30	D0 Comment Alarm Device Info Block
68	16	16	D0 Comment 16-character device name
69	2	0	D0 Date-of-last-change in CADIB entry

There are only two comment alarms traditionally defined. Comment #0 is the system reset message that is emitted soon after a system has been reset. Comment #1 is the alarm reset message that is emitted in response to performing an alarms reset action, which "forgets" about all the outstanding bad alarm conditions, allowing the next alarm scan to produce new messages about bad analog and digital devices.

System table directory

As part of a system configuration, certain system tables are defined according to the system table directory. The 4-byte ident consists of the node number and the system table index.

Listype	*Data size*	*Max #bytes*	*Meaning*
70	8	8	System table directory entry

The system table directory, as well as most of the system tables defined therein, are maintained in nonvolatile memory. There are 32 eight-byte entries in the system table directory that define system tables 0–31. The tables are as follows:

Table#	*Name*	*Meaning*
0	ADATA	Analog data
1	ADESC	Analog descriptor
2	BALRM	Binary alarm flags, count
3	BDESC	Binary bit title
4	RDATA	Data access table
5	BBYTE	Binary bytes
6	PAGEP	Page name, title
7	PAGEM	Page-private memory
8	LISTP	Data request pointers
9	CODES	Non-volatile memory file directory
10	CDATA	Comment alarm flags, count, text
11	BADDR	Digital byte addresses
12	OUTPQ	Network message output pointer queue
13	PRNTQ	Print queue for serial port output
14	LATBL	Local applications
15	CPROQ	Co-processor queues
16	MMAPS	Memory-mapped board access for D0
17	Q1553	Queue pointers for 1553 controllers
18	DSTRM	Data stream directory
19	SERIQ	Serial port input queue
20	TRUNK	IP addresses for nodes using Acnet
21	AADIB	Analog alarm device info block
22	BADIB	Binary alarm device info block
23	CADIB	Comment alarm device info block
24	CSTAT	Combined binary status specs
25	CINFO	Analog channel-related extra info
26	—	(spare)
27	IPNAT	Node-number/IP-address table (from DNS)
28	IPARP	IP-ARP table

29	—	(spare)
30	TRING	Token ring network table
31	DLOAD	Non-volatile memory file directory

Each system table is defined in the table directory as an array of structs. Each table directory entry is 8 bytes long and describes the number of entries in the table, the size of each entry, and the address of the start of the table.

```
typedef struct {
            short nEnt;  /* number of entries */
            short eSiz;  /* size of each entry */
            int32 tPtr;  /* base address of table */
            } TDIR_ENTRY;
```

There is a checksum word included in the last entry that insures that the table directory has not been corrupted at system initialization, in which the table directory is copied into low memory for use during system operation, allowing the nonvolatile version to be modified while the system is active. When a setting is made using this listype 70, the checksum is automatically adjusted to conform to the change made.

Non-volatile memory file

The system supports a read-only memory file system for keeping a copy of local and page applications needed by the node. The ident type that permits access to such a memory file is 14 bytes long, consisting of a node number word, an 8-character file name, and a byte offset longword.

Listype	*Data size*	*Max #bytes*	*Meaning*
76	n	n	Non-volatile memory file

This listype can be used to copy a file from one node to another in pieces, preserving a "version date," even though the location of a given file will of course quite likely be different in each node. There is no interference with a previous version of a program that may be running, because programs only execute in a dynamic memory copy.

Combined binary status definitions

Status words can be constructed from arbitrary status bits sampled from the BBYTE table, according to specifications in the CSTAT system table. An ident type that access this table's entries is of the usual 4-byte format, with the node number followed by a table index number.

Listype	*Data size*	*Max #bytes*	*Meaning*
79	32*n	32	Non-volatile memory file

Each table entry is 32 bytes long, including 8 four-byte sub-entries. Each sub-entry consists of a byte number word, a shift count byte, and a mask. The sub-entries define how to sample bits from the digital data pool to construct a 16-bit status word that can participate in the digital alarm scan as a unit.

IP node address table

The IP node address table is used to relate node numbers to IP addresses and is a

cache for Domain Name Server query results. The 4-byte ident format that accesses this table is the node number word and a table index.

Listype	*Data size*	*Max #bytes*	*Meaning*
81	8*n	8	IP node address table

The first 6 entries (48 bytes) are used as a header that contains a Domain Name Server IP address and a suffix (such as `fnal.gov`) to be used in constructing DNS queries. The 8-byte entries consist of a node number, a countdown timer, and the node's IP address as returned from a DNS query. Each entry is updated from the DNS every 8 hours so that the IP addresses do not become stale. One can also enter static entries that do not participate in DNS queries, in order to facilitate access to foreign nodes, analogous to a Unix "hosts" file.

Fast sampled data

An IRM includes support for 64 A/D channels that are sampled at 1000 Hz. A 6-byte ident format that accesses such data consists of a node number word, a channel number word, and a clock event number word.

Listype	*Data size*	*Max #bytes*	*Meaning*
83	8+n	0	Fast sampled data access

The format of data received in response to such a request includes two longwords at the start of the buffer. The first longword specifies the time (in 10 μs units) of the first data point in the buffer. The second longword is used to support the case when a clock event occurs during the time the data points were acquired that are in the buffer. Each data point is a 16-bit data word from the A/D, followed by a 16-bit data word that is the time (in 10 μs units) relative to the time of the first data point. For more details, see the document Moderately Fast Data Collection, or Fast Time Data Collection.

Raw 1553 command

This listype permits support of generic I/O access via the MIL-STD-1553 bus. The 6-byte ident format is a node number word followed by the 1553 controller memory address followed by the 1553 command to be used. (The actual 1553 controller base memory address is always of the form `xxxy0000`, where `y` is taken as the controller number, so that only the upper word must be specified.) The 1553 command includes a 5-bit word count, which may sometimes represent a mode code. Apart from the mode code case, the number of bytes of setting data determine what will be used in the 5-bit field.

Listype	*Data size*	*Max #bytes*	*Meaning*
85	n*2	n*2	Execute 1553 command

Support for 1553 data access has traditionally been hidden from the user. But the need for extensive 1553 downloading of detector parameters for use in the D0 upgrade required a low level access, analogous to generic memory access. The number of bytes of setting data may be very large, so that the software will have to divide up the data into chunks of 32 words or less—in order to conform to the inherent 32-word limit of the 1553 protocol.

IP security table

The IP security table is used to prevent unauthorized IP addresses from gaining *setting* access to the system. The ident format is the node number word and the index into the table.

Listype	*Data size*	*Max #bytes*	*Meaning*
80	64*n	n	IP security table

Following a 64-byte table header, each table entry is 64 bytes long, the first two longwords containing an IP address and mask. If the source IP address of a setting message matches the IP address within the mask, it is authorized; otherwise, setting access is denied. A record is made of such failed attempts.