

# SRM Message Protocols

*From Local Station over Arcnet*

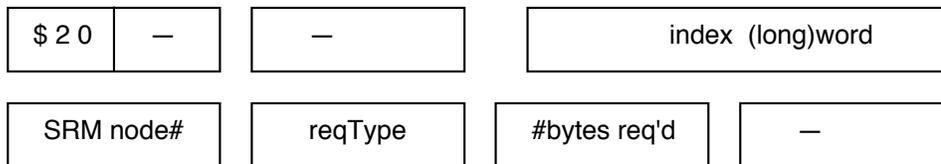
Jul 23, 1991

The new Linac control system local stations support an Arcnet interface to Smart Rack Monitors (SRMs). Each station connects to several SRMs using a private Arcnet network. The SRM acts as a data concentrator to make data access to the hardware more uniform. In the future it may support closed loop algorithms. To the local station, an SRM is treated partly as a data interface and partly as a network node. The simple message protocol is based upon the Arcnet header used for task-task communications in Fermilab accelerator systems.

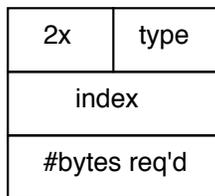
## *Data interface*

As a data interface, three types of entries in the Data Access Table (DAT) of the local station reference the SRM for collecting readings and settings. The advantage that the SRM provides is that a single network frame can deliver all the SRM data in response to a data acquisition request, thereby making access to the data more efficient. The following describes some details of this connection.

Every 15 Hz cycle, the DAT is interpreted by the local station. This first type of DAT entry for SRM data acquisition is used to send a request message to the SRMD destination task in the SRM. The entry format is as follows:



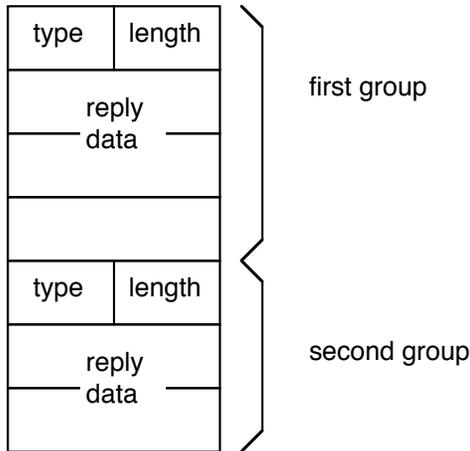
A one-shot data cycle request message is queued to be sent to an Arcnet node or broadcast to all Arcnet nodes. `NetXmit` is called to flush the network queue after this entry. Beyond the Arcnet header the format of the message is:



For this case, `x` is 2, indicating one index word value present. The maximum number of bytes requested is 490 bytes. The `type` byte indicates the usual 15 Hz data collection. The `index` word is not used in this case. In response to this message, the SRMD task collects all its data using its own version of a data access table, builds a frame buffer with its response data, and returns the response frame to the local station. `Type` byte values defined for SRM data requests are:

Type	Function
01	Request to read and return 15 Hz cycle data
02	Memory dump (index = 4-byte address)
03	Table access (index = table#)

The data response message from the SRM has the following format:



Several groups of variable lengths follow the acnet header. Each group represents data of a different type. The group types, which are SRM table #s, are listed below. The length byte is the number of words in the group, including the type/length word. Error status is given by the acnet header status word. The reply message remains in the circular receive buffer until needed by post-processing logic that maps these sections of data into local station analog channels and binary data bytes.

SRM table#	Data referenced	#entries
01	Linac mpx A/D via daughter-board	I/O 16*n words
02	Linac mpx D/A via daughter-board	I/O 16*n words
03	spare	—
04	A/D on SRM board	64 words
05	D/A on SRM board	16 words
06	Digital I/O on SRM board	8 bytes
07	Mpx digital input via daughter-board	16 bytes
08	Digital I/O on a daughter board	4 bytes
09	Actel timer board	8 words
0A	Opto-22 digital I/O	bytes 2-3 bytes
0B	Digital I/O board #1	9 bytes
0C	Digital I/O board #2	9 bytes

Returning to the DAT entries in the local station, when no more work can be done until the SRM response data has been collected, a second type of entry in the data access table is used to wait for the SRM data response.

\$ 2 1	—	—	—	—
SRM node#	deadline	—	—	—

This is done by waiting on the message queue containing the reference to that response message. When the message from the given SRM is received, or if a deadline time (in 0.5 msec units from the start of the current 15 Hz cycle) is exceeded, processing of the DAT continues. If messages are received from other SRMs besides the one specified in this entry, a record of it is kept so that the succeeding wait type entry that refers to it is immediately satisfied. To insure that old messages are not left over from the previous cycle, the message queue is emptied prior to sending the SRM data request message.

A third type of DAT entry is used to map the response data into analog channels and binary bytes of local station data.

\$ 2 2	tbl#	entry#	—	—
SRM node#	entry offset	SRM table#	offset	#entries

The `entry#` identifies a target analog channel, for example. (In the usual case, the `tbl#` would be 0, indicating the `ADATA` table. One can target the setting word of an analog channel, rather than the usual reading word, by using a value of 2 for the `entry offset`.) Also specified is an `SRM node#` and an `SRM table#`, given in the table above. A search is made through the structure in the cycle response message for a match on this group type, and data words are copied into successive entries in the `ADATA` table. If the `#entries` is not equal to the amount of data included of the given type, then only the smaller amount of data is copied. This allows modifications to the SRM's data access table without simultaneously having to change the local station DAT. For the case of digital byte data, the `#entries` refers to a `#bytes`; otherwise, it is a `#words`. Many such DAT entries may be required to map the different groups of data included in the cycle data response frame into the local station's data pool. It is assumed there is only one occurrence of a given `SRM table#` in the received message.

If an SRM does not respond to the cycle data request by the given deadline, the target channel or byte readings are written as zeros, in order to avoid reporting stale data readings. Also, settings are not accepted for such an SRM. A longword of status bits is maintained that can be assigned as digital data bytes and included in the alarm scan to provide alarms about SRMs which are down.

#### Examples of DAT entries for SRM data acquisition

Request cycle data from all SRMs:

```
2000 0000 0000 0000
7A00 2201 01F0 0000
```

Wait for cycle data from SRM A2

```
2100 0000 0000 0000
7AA2 0030 0000 0000
```

Map 64 channels of SRM A/D into analog channels 0080–00BF:

```
2200 0080 0000 0000
7AA2 0000 0400 0040
```

Map 16 channels of SRM D/A into settings of channels 0080–008F:

```
2200 0080 0000 0000
7AA2 0002 0500 0010
```

Map 6 bytes of SRM digital data into binary bytes 000C–0011:

```
2205 000C 0000 0000
7AA2 0000 0600 0006
```

#### Network interface

To support Arcnet in the local station, which has extensive support for the token ring network, a range of node numbers is used for Arcnet nodes. A node number is a word, so

there is plenty of addressing space available to do this. The Arcnet range is  $0x7Axx$ , where  $xx$  is the one-byte node# used by the Arcnet interface. Support for motors, described later in the settings section of this document, further restricts the range used for SRM Arcnet nodes to  $A1-BF$ , which still permits up to 31 SRM nodes to be installed on Arcnet.

Messages to be sent to an SRM on Arcnet pass through a separate Output Pointer Queue (OUTPQ) than that used for token ring transmissions. (This permits Arcnet activity even during the time that a station is opening onto the token ring network.) The `NetXmit` code combines messages into frames using the appropriate transmit buffer. There is a separate transmit parameter list chain, which is a queue of pointers to network frames ready to be transmitted on Arcnet. When the frame is ready, a check is made for current Arcnet transmit activity. If it is idle, then an Arcnet transmit interrupt is forced by enabling the TA interrupt. (If it is busy, nothing is done, as the pending Arcnet transmit interrupt will take care of it.) The interrupt routine does error checking on the last frame transmitted, if any, and copies the next frame in the queue to the hardware buffer. It then enables the buffer to hand it over to the Arcnet hardware. In this way, many Arcnet frames can be queued awaiting transmission.

When a frame is received from Arcnet, the interrupt routine copies the frame from the hardware buffer into a circular receive frame buffer. This buffer is separate from the one that receives token ring frames because the token ring chipset has DMA control over its frame buffer. A second hardware buffer is re-enabled to provide a place to receive the next frame. The software emulates the token ring format in the circular buffer. It also checks the Acnet header of each message within the frame and sends a pSOS message reference (consisting of four longwords that include a pointer to the message itself in the circular buffer) to the message queue used by the receiving task, according to an entry in the `NetConnect` table (`NETCT`). That entry# is given by the `srcTaskId` in the Acnet header for replies. An optional event can also be sent to a task in case the task waits on events instead of the message queue. When a task receives such a reference message about an Arcnet message, it looks exactly the same as if it came from the token ring network. All subsequent processing is unaffected.

To a user program, Arcnet nodes can be accessed in the same way as for token ring. The only difference is that the protocol of the messages supported is different. The SRMs do not support the Classic protocol, the D0 protocol or the accelerator protocol. They only support their own Acnet-header-based protocol.

### Settings

Settings are of several types, including memory, analog D/A or motors, and binary bit or byte control. Each results in a short message sent over Arcnet to the SRM. The message format (beyond the Acnet header) is:

3x	type
index	
#bytes data	
setting data	

The  $x$  indicates the size of the index field. It is 4 for the memory case since the index field is a 32-bit address. It is 2 for the other cases. The `type` byte denotes a control type# to the SRM using the following values:

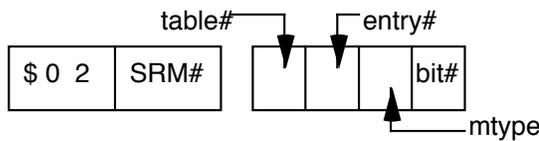
Type#	Hardware control	Index word	Data
05	Motor control	table,entry/mType,bit	#steps
06	Bit control	table,entry/0,bit	dcType/pulse
07	Byte control	table/entry	Data byte
08	D/A control	table/entry	Setting value
09	Memory write	32-bit memory address	Memory data

(The index field syntax above uses “,” to separate nibbles and “/” to separate bytes.) In the local station, the index parameter of the message originates in various tables. First consider the analog control case. For a D/A or motor, the analog control field of an analog channel descriptor is required to contain the needed information to effect the setting.



For a D/A analog control field entry, the SRM# is the Arcnet node# byte. The SRM table# and entry# make up the last two bytes.

For a motor, the Analog control type# is constrained to be 02, so there is less space available for the needed parameters.



The table# and entry# (byte#) is limited to 4 bits each. The bit# (range 0–7) is the bit# of the byte that contains the pair of control bits that drive the motor. The mType specifies the type of motor interface as follows:

mType	Motor interface	bit# indicates:
1	cw, ccw pulse pair	cw bit
2	pulse, direction	pulse bit

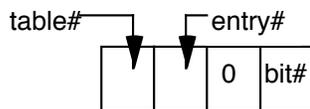
For digital control, the BADDR table in the local station contains longword entries for each byte of binary data. The entry is normally a pointer to the source of the byte of data, except for special cases. For the case of an entry in the range 0x80xxxxxx, the lower three bytes are a 24-bit pointer to a byte in a 1553 command block that accesses one word of digital data. For control, the corresponding command block is found located 16 bytes from the reading command block.

The range 0x81xxxxxx is used to indicate SRM digital data. The second byte is the SRM node#, and the last two bytes are parameters of the digital interface.



For this case, it is not necessary to contain a pointer to the data byte reading, since the post-processing that is done with the third type of DAT entries above takes care of updating these bytes that are stored in the BBYTE table of raw status data. The processing for the 0x04 DAT entry skips to the next BADDR entry if it finds an entry in the range 0x81xxxxxx.

For digital byte control, the index word in the message sent to the SRM is just the second word above. For digital bit control, however, the message passed to the SRM has a modified index word, in order to specify the bit# within the byte that is to be controlled. The modified format is:



The `table#` and `entry#` are squeezed into a nibble, and the `bit#` of the byte to be controlled is placed in the low 3 bits.

For analog and binary settings, the SRM keeps a copy of the last values that were sent to the hardware interfaces. When an SRM resets, it re-sends all of these values to the hardware, since it may be resetting from a power-off condition, and the hardware interfaces would have lost their settings.

The local station also re-sends settings for the same reason after a reset of the local station system. For the SRM devices, this means that the local station values are passed to the SRM via Arcnet. The local station values are expected to be correct, since they are updated when a setting message is queued for the SRM and also when the SRM returns setting values in the cycle data response message. These saved values are returned in response to a host's request for setting data.

As an example, when a setting is received for an analog channel which is attached to an SRM, the value is placed in a setting message that is queued for Arcnet. If the queuing is successful, including a check on whether the SRM is actively participating in the data cycle request activity, the setting word in the `ADATA` table is updated. When the SRM executes the setting, and there are no errors, then it updates its own setting value. If the SRM is unsuccessful in making the setting, this value is not updated. In any case, the following cycle's data response message contains the last value kept by the SRM. In the case that the setting was unsuccessful, this value will replace the optimistic value stored by the local station. In this way, the local station value is nearly always correct. By having the local station store the optimistic value, one can maintain sensible support for multiple user control of the same analog channel, immediate read-back of the setting value, and multiple setting commands that reference different bits in the same byte.