

Module Interface Board - MIB

Data Port

Interface Control Document

Version 1.2.0

Table of Contents

1	Introduction.....	1
2	Connection.....	1
3	Data Organization.....	2
4	Document Keys.....	2
4.1	Syntax Key.....	2
	Table 1. DP Syntax Key.....	2
4.2	Terminology Key.....	3
	Table 2. DP Terminology Key.....	3
5	Data Control.....	3
5.1	Archive and Screen.....	3
5.1.1	MIB Control Points used by the Data Port	3
5.1.2	Data Rate Control.....	4
5.2	Alert Data Control.....	4
5.2.1	Analog MP Alert Control.....	4
5.2.2	Digital MP Alert Control	5
6	Output Format from the MIB.....	5
6.1	Archive and Screen Data.....	5
6.1.1	Sample Archive and Screen Data Output	5
6.2	Alert Data.....	6
6.2.1	Analog MP alert.....	6
6.2.2	Digital MP Alert.....	6
7	Data Port Details	7
7.1	Terminology.....	7
	Table 3. Data Port Facility Terms	7
7.2	Processing setup	7
7.3	Data Port scan sequence.....	8
7.3.1	Alert Data Generation.....	8
7.3.2	Archive and Screen Data Generation.....	10
7.4	Data Port Common Routines	10
7.5	Data Port Tasks	10
7.5.1	Alert Task Details	11
7.5.2	Archive and Screen Task Details.....	11
8	MIB Device Points.....	11
9	Appendix.....	12
	Figure 1. MIB Software Diagram	12
	Table 4. Flash Memory Layout.....	13

Revision History

<i>Revision</i>	<i>Date</i>	<i>Author(s)</i>	<i>Description of Changes</i>
1.0.0	November 5, 2002	William Sahr	Original version using custom binary data formats.
1.1.0	October 9, 2003	?	Incomplete modification of 1.0.0
1.2.0	September 14, 2004	C. Frank Helvey	Completely rewritten to reflect actual implementation of the Data Port in MIB Framework revision 0.18. Implemented document revision number as document property "REVISIONLEVEL" which is accessed through the File->Property->Custom menu.

1 Introduction

This document describes the communication protocol over the Data Port (DP) on the Module Interface Board (MIB) and how the data is generated on the DP.

The MIB is physically connected to the EVLA LAN on the one hand and to one or more pieces of electronic equipment on the other. The MIB functions as a uniform LAN interface for this disparate equipment.

The DP is used primarily by the MIB to send out periodic data snapshots of the monitor points (MPs) defined in the MIB. Two data streams have been defined – an archive data stream and a screen data stream. The rates for the archive and screen data are configurable in the MIB's configuration XML file on a per-MP basis. Software processes located on the network can capture these messages and act on them; there should always be a process capturing the archive data and storing it into a database for later use, but the MIB has no knowledge of the existence of the consumer process(es); if there is no process receiving the data, it is lost.

The MIB also uses the DP to send out alert messages asynchronously with respect to the archive and screen data activity. These alert messages have information on a MP that has changed alert state, and contain data that is different from the data contained in the archive and screen data messages. In the same fashion as the other two data streams, software processes located on the network can capture these alert messages and take action as required; and again, the MIB does not know if there are any processes receiving these messages. If not, the data is lost.

For completeness, it should be mentioned there exists another LAN connection on the MIB called the Service Port (SP). It permits operators and/or software processes to acquire data immediately from the MIB's Control Point (CP) and MP database with the SP's "get" command. It also allows control of the MIB (including control of DP activities) and the hardware connected to the MIB via the SP "set" command. Details of the Service Port are described in the MIB Service Port Interface Control Document.

See *Figure 1. MIB Software Diagram* for a schematic depiction of the overall MIB software.

2 Connection

There is no inbound connection to the MIB for the DP. Outbound, the MIB multicasts UDP datagrams containing archive data to port 20010 using a defined multicast group IP address in the "Organization-Local Scope" site-specific range of multicast addresses. Alert data is multicast as UDP datagrams to port 20011 using another similarly allocated multicast group IP address. Currently, screen data is still implemented as a unicast, but will soon be converted to multicast UDP datagrams to port 2004 using yet another similarly allocated multicast group IP address. The network design currently calls for the address ranges 239.192.0.x to be used for testing in the AOC and 239.192.1.x to be used for the systems at the VLA site.

By using multicast, any number of consumer/client process(es) can receive these messages, but only one copy of the message will be present on the network. Furthermore, the network can easily route the multicast messages to the subnets that have a process looking for that particular type of data.

Note that at the time of the writing of revision 1.2.0 of this document, the exact allocation for the multicast IP addresses had not yet been determined. The following IP addresses are being used in the AOC until such a determination is made:

Archive and alert data are using the multicast address 239.192.0.1
Screen data is using the unicast address 146.88.3.214

According to the convention currently agreed to, the addresses at the VLA site should be in the range of 239.192.1.x. This means that the archive and alert data will use 239.192.1.1 for the initial implementation.

3 Data Organization

The MIB presents the electronics equipment to the DP as one or more logical Devices, each of which has one or more Monitor Points (MPs) or Control Points (CPs). Each MP and CP has a set of Attributes that includes a value field. The other Attributes expand upon certain auxiliary issues related to that value, such as allowed minimum and maximum; each Attribute is defined to be a read/write or read-only field. For the purposes of the DP, only an MP's name, type, value, and alert state attribute contents are ever sent.

A logical Device need not correspond to a physical device. The association between logical Devices on the Data Port and physical devices at the hardware level is made by the module-specific code and the MIB Framework code, which in turn depends upon the exact equipment connected to the MIB. The MIB itself will be one such logical Device for controlling and monitoring functionality wholly within the MIB. The DP does not care how the MPs are associated with logical devices, since each MP that is ready to have archive or screen data sent is processed individually. The XML output format does contain the device information for the MP(s) that are being sent; each device's list of MPs that are ready to be sent will be grouped together by device name in the message in the order in which they were read from the internal database table of the MIB.

4 Document Keys

4.1 Syntax Key

All DP outputs are in XML format. XML is a standard Internet format for data exchange and will not be discussed in this document. The following table shows some important features of the format. In this document, XML output will be shown in the font used in this table.

Table 1. DP Syntax Key

Syntax	Meaning
<	opens an XML statement
/>	closes an XML statement (one per opening <)
abc	literal name; could be outer level name, device, property, or attribute (see Table 2. DP Terminology Key below).
=	introduces attribute value
'123'	attribute value; the value is 123

4.2 Terminology Key

The following terms are used in the document.

Table 2. DP Terminology Key

Term	Meaning
device	The name of a device connected to the MIB, including the MIB itself; generally there is a 1:1 correspondence between a device and a single piece of hardware. The device name can be up to 7 characters long.
property	A particular CP or MP on a device. The property name can be up to 23 characters long.
attribute	The name of an individual piece of data that makes up a MP or CP entry. Up to 23 characters long. Each MP and CP has an attribute called <code>type</code> that further qualifies its attribute list; the <code>type</code> attribute's value can be either analog or digital . There are tables in the MIB Service Port ICD document that summarize the attributes available for each type of MP and CP.
value	The value (an attribute) of a CP or MP. The value can be up to 47 characters long.
element	An XML placeholder or term.
MIB Framework	The common software base for all MIB module software. It provides a common set of features and interfaces; the Data Port is an example of the features provided.
system scan cycle	Used to denote the point scan rate of the MIB Framework. The MIB scans points at 100ms intervals. Thus, the fastest rate at which a point can be scanned by the Framework is 100ms.

5 Data Control

5.1 Archive and Screen

5.1.1 MIB Control Points used by the Data Port

The MIB has two Control Points (CPs) that are used to control the Data Port's transmission of messages.

The “wantArchive” digital control point is used to determine if archive data messages will be sent. If this control point has a value of 1 (the default value) then archive data will be sent by the MIB; a value of 0 will cause the MIB to not send the data.

The “wantScreen” digital control point is used to determine if the screen data messages will be sent. If this control point has a value of 1 then screen data will be sent by the MIB; a value of 0 (the default value) will cause the MIB to not send the data.

If one of these control points is set to disable transmission of messages, the MIB will continue to track the time for the MPs for that feature (see 5.1.2 Data Rate Control below). Thus, if the control point is later set to allow the transmission of messages, the MIB will start sending messages associated with that point at any time thereafter.

5.1.2 Data Rate Control

Each MP has attributes to control the rate at which its archive and screen messages are sent. There is also a rate for archiving data when the point is in the alert state, primarily to allow a fast rate of capture during that time. If not desired, then that rate should be set to be the same as the standard archive rate.

The attribute names are `a_period` for the standard archive data rate, `s_period` for the screen data rate, and `aa_period` for the alert state archive data rate.

Each of these attributes will have a value that is the number of system scan cycles that must elapse before a data message is sent. A value of 0 indicates that no data capture of that type is desired. Once the count of elapsed system scan cycles has reached the appropriate level, the MP will have its data sent and the count will start anew. This is covered in more detail in section 7, Data Port Details.

5.2 Alert Data Control

Control of the generation of alert data is governed on a MP by MP basis by using alert enable attributes. The analog and digital MPs have slightly different attributes, so each is discussed separately below.

5.2.1 Analog MP Alert Control

There are two attributes in an analog MP that control the point's ability to enter the alert state and to send an alert data message. These attributes are called `hi_alert_arm` and `lo_alert_arm`. They are binary values; if set to 1, that MP will be able to enter that state.

There are two attributes of an analog MP that define the inclusive range of values that are valid. The names of these attributes are `min` and `max`. Values below the `min` limit are capable of placing the analog MP into the low alert state; values above the `max` limit are capable of placing the analog MP into the high alert state.

If an analog MP is enabled to enter the alert state (either as a high or low alert), it will not necessarily do so the first time the MIB detects an out of range value. Instead, there are two attributes that serve as a debounce or time delay on the entry to or exit from the alert state for an analog MP. These attributes are called `alert_in_count` and `alert_out_count`. The values of these attributes are in units of system scan cycles (not scan cycles for that MP) which must elapse with the analog MP's value in/out of range before the MP will exit/enter the appropriate alert state. A value of 0 or 1 will cause the MP to exit/enter the alert state the first time the MIB detects an in/out of range value.

The two count attributes are not available on the service port; they can only be modified in the configuration XML file for the MIB.

See section 7, Data Port Details, for more information on the two analog alert states.

5.2.2 Digital MP Alert Control

There are two attributes that control the entry of a digital MP into the alert state and the exit from it as well. They are called `alert_arm` and `alert_on1`. The `alert_arm` attribute takes a binary value; if set to 1, that digital MP will be able to enter the alert state. The `alert_on1` attribute's value is the binary value that the digital MP should have normally; divergence from that will cause the point to enter the alert state depending on the `alert_arm` attribute's value. Likewise, if the MP is in the alert state and the value equals the `alert_on1` attribute's value, the MP will exit the alert state. See section 7, Data Port Details for more information on the digital MPs and the alert state.

6 Output Format from the MIB

All data is sent in UDP datagrams as ASCII plain text in XML format. Each data message fits within one Ethernet packet so no fragmentation issues can cause the message to be partially lost in transit; several messages may be sent if the amount of data at any particular system scan cycle is greater than one Ethernet packet's capacity.

The outer XML element is `EVLAMessage` with three optional attributes: `status`, `location`, and `timestamp`. The DP uses all of these except the `status` attribute. The outer element is then followed by one or more `device` elements, each with a `name` attribute value. Each `device` element has 1 or more properties (MPs) listed, with the appropriate attributes for the MPs depending on the type of message being sent. There are no line breaks in the data.

The attributes of the properties (MPs) for each type of message are described below.

6.1 Archive and Screen Data

These messages will include the following attribute values for the MP being reported: `name`, `type`, and `value`. The attribute values are captured at the time of the generation of the message. Only MPs that are ready at the timestamp given will be reported in the message; if multiple messages are generated they could have different timestamps since their data is being captured at different times, but all should be very close together.

See the following sample output for an example.

6.1.1 Sample Archive and Screen Data Output

Note that this sample output has been modified for readability; normally, the data runs together without line breaks.

```
<EVLAMessage location='Antenna 13' timestamp='53264.9191010'>
  <device name='ACU'>
    <monitor name='AZCur2' type='analog' value='0.000000' />
    <monitor name='AZCur1' type='analog' value='0.000000' />
    <monitor name='Mode' type='analog' value='0x0' />
    <monitor name='ELPos' type='analog' value='0.000000' />
  </device>
  <device name='FRM'>
    <monitor name='P5V' type='analog' value='0.000000' />
    <monitor name='ApXRot_ROT_VEL' type='analog' value='0.000000' />
    <monitor name='ApXRot_ALG_GRND' type='analog' value='0.000000' />
    <monitor name='RotBrakeMon' type='digital' value='0' />
    <monitor name='IF_Parity_Errors' type='analog' value='0.000000' />
  </device>
</EVLAMessage>
```

```
<monitor name='IF_No_S_Codes' type='digital' value='0' />
</device>
</EVLMMessage>
```

6.2 Alert Data

These messages will include the following common attribute values for the MP being reported: `name`, `type`, `value`, and `alert`. Besides those common attributes, analog MPs will report specific attributes for the type of alert condition detected. The attribute values and the time are captured at the time of the detection of the alert condition and sent to the alert task. Since this is done on an MP-by-MP basis, only one MP will be in each alert message sent out of the DP.

See the sample output in the following section for an example.

6.2.1 Analog MP alert

The analog MP alert messages will include the attribute values for the attributes `hi_alert` and `lo_alert` in addition to the values of the common attributes given above.

6.2.1.1 Sample Analog Alerts

Note that these sample outputs have been modified for readability; normally, the data runs together without line breaks.

Alarm Active (low alert state):

```
<EVLMMessage location='Antenna 13' timestamp='53264.9191010'>
  <device name='ACU'>
    <monitor name='PhaseA' type='analog' value='-248.000000' alert='1'
hi_alert='0' lo_alert='1' />
  </device>
</EVLMMessage>
```

Alarm Clear:

```
<EVLMMessage location='Antenna 13' timestamp='53264.9191010'>
  <device name='ACU'>
    <monitor name='PhaseA' type='analog' value='160.000000' alert='0'
hi_alert='0' lo_alert='0' />
  </device>
</EVLMMessage>
```

6.2.2 Digital MP Alert

Digital alert messages have the common attributes only.

6.2.2.1 Sample Digital Alert

Note that these sample outputs have been modified for readability; normally, the data runs together without line breaks.

Alarm Active:

```
<EVLMMessage location='Antenna 13' timestamp='53264.9191015'>
  <device name='ACU'>
    <monitor name='IF_No_S_Codes' type='digital' value='1' alert='1' />
  </device>
</EVLMMessage>
```

Alarm Clear:

```
<EVLMMessage location='Antenna 13' timestamp='53264.9191015'>
  <device name='ACU'>
    <monitor name='IF_No_S_Codes' type='digital' value='0' alert='0' />
  </device>
</EVLMMessage>
```

7 Data Port Details

This section describes the internal workings of the Data Port facility of the MIB Framework in more detail.

7.1 Terminology

In addition to the terms used in the table 4.2, "Terminology Key", the following terms are used in this section of the document.

Table 3. Data Port Facility Terms

Term	Meaning
maximum Ethernet message	A data buffer of 1280 bytes in length. This is the maximum amount of user data that Ethernet can send, after deducting space for the UDP and IP header information in the message.
logical point	Either a Control Point (CP) or Monitor Point (MP).
flag	A boolean value in the logical points database of the MIB.

7.2 Processing setup

The DP's processing is split between the MIB Framework's point scanning task and three other tasks – one each for alert, archive, and screen data message generation.

Processing in the MIB Framework's point scanning task will track the states of the MPs and determine when they are ready for alert, archive, or screen data generation. A message is then sent to the appropriate task, which will generate and send the DP message.

7.3 Data Port scan sequence

The MIB Framework code will perform the following scans each system scan cycle, in the order given. During each scan, the entire logical points list (control points and monitor points) is examined, but only the monitor points are of interest – control points are ignored.

7.3.1 Alert Data Generation

Depending on the type of point being examined, different processing is done. In all cases, the messages generated for the alert task will contain a logical point index number for the point transitioning from or to the alert state, plus the value of the attributes `value`, `alert`, `hi_alert`, and `lo_alert` (depending on the type of MP being reported), and the time the alert state change was detected.

7.3.1.1 Analog Monitor Points

First, the point's `hi_alert_arm` attribute's value is checked. If set to 1, the software will compare the MP's `value` to the value of the `max` attribute.

If the MP's `value` exceeds the value of the `max` attribute, then the current value of the high alert state flag is checked. If set to 1 no further action is taken – the point is already in the high alert state. If the high alert state flag is set to 0, the analog MP's count of high alerts is incremented. If the high alert count is equal to 1 then the current time will be captured into the analog MP's alert timestamp field. The high alert count is then checked to see if it equals or exceeds the `alert_in_count` attribute's value. If it does, the high alert state flag is set to 1, and the counter governing how often the MP's `value` is sent to the archive while the MP is in the alert state is set equal to the time remaining before the MP's `value` would have been sent to the archive if the point had not entered the alert state. (If the MP remains in the alert state beyond the 1st transmission of its value to the archive, the value of `aa_period` (see section 5.1.2, Data Rate Control) governs how often the MP's value is sent to the archive while the MP remains in the alert state.) The count of high alerts is then reset to 0, and a message is sent to the alert task.

If the MP's `value` does not exceed the `max` attribute's value, then the current value of the high alert state flag is checked. If its value is 1, indicating that the point is in the high alert state, then the analog MP's count of high alerts is incremented (in this case the count of high alerts is being used to accumulate information to determine when to exit from the high alert state). If the high alert count is equal to 1 then the current time will be captured into the analog MP's alert timestamp field. The high alert count is then checked to see if equals or exceeds the `alert_out_count` attribute's value; if it does, the high alert state flag is set to 0, and the counter controlling when the MP's `value` will next be sent to the archive is set equal to the time remaining before the MP's `value` would have been sent to the archive if the MP had remained in the alert state. (After that transmission of the MP value, the value of `a_period` governs how often the MP's value is sent to the archive while the MP is not in the alert state.) The count of high alerts is reset to 0. Finally, a message is sent to the alert task.

The count of high alerts will also be set to 0 if the MP is not enabled for high alerts, or if the MP is not in the high alert state and its `value` is equal to or less than the `max` attribute's value. If the MP is in high alert state, the count of high alerts will be set to 0 if the MP's value is greater than the `max` attribute's value. (Please recall that if the MP is in the high alert state, the high alert counter is subverted to serve as a counter used to determine when to exit the alert state.)

The result of this logic is that an MP will enter the high alert state only if its value exceeds the `max` attribute's value an `alert_in_count` number of consecutive times. Likewise, an MP will

exit the high alert state only if its value is less than or equal to the `max` attribute's value an `alert_out_count` number of consecutive times.

Next, regardless of the result of the processing for high alerts, the point's `lo_alert_arm` attribute's value is checked. If set to 1, the software will compare the MP's `value` to the `min` attribute's value. From this point onward, the logic for low alerts is exactly analogous to the logic for the high alerts.

If the MP's `value` is less than the `min` attribute's value, then the current value of the low alert state flag is checked. If it is set to 1 no further action is taken – the point is already in low alert state. If the low alert state flag is set to 0, the analog MP's count of low alerts is incremented. If the low alert count is equal to 1 then the current time will be captured into the analog MP's alert timestamp field. The low alert count is then checked to see if equals or exceeds the `alert_in_count` attribute's value; if it does, the low alert state flag is set to 1, and the counter governing how often the MP's `value` is sent to the archive while the MP is in the alert state is set equal to the time remaining before the MP's `value` would have been sent to the archive if the point had not entered the alert state. The count of low alerts is reset to 0, and a message is sent to the alert task.

If the MP's `value` is not less than the `min` attribute's value, then the current value of the low alert state flag is checked. If its value is 1, indicating that the point is in the low alert state, then the MP's count of low alerts is incremented (preparing, in this case, for an exit from that state). If the low alert count is equal to 1 then the current time will be captured into the analog MP's alert timestamp field. The low alert count is then checked to see if equals or exceeds the `alert_out_count` attribute's value; if it does, the low alert state flag will be set to 0, and the counter controlling when the MP's `value` will next be sent to the archive while not in alert state is set equal to the time remaining before the MP's `value` would have been sent to the archive if the MP had remained in the alert state. The count of low alerts is then set to 0. Finally, a message is sent to the alert task.

The count of low alerts will also be set to 0 if the MP is not enabled for low alerts, or if the MP is not in the low alert state and the MP's `value` is equal to or greater than the `min` attribute's value. If the MP is in low alert state, the count of low alerts will be set to 0 if the MP's `value` is less than the `min` attribute's value, in order to enable use of the low alert counter as an indicator of when to exit the low alert state.

As with the high alert, the result of this logic is that an MP will not enter the low alert state unless its values are less than the `min` attribute's value an `alert_in_count` number of consecutive times. Likewise an MP will not exit the low alert state unless its value is greater than or equal to the `min` attributes's value an `alert_out_count` number of consecutive times.

The `alert` attribute's value is set to 1 if either the `hi_alert` or `lo_alert` attributes' values are set to 1, indicating the point is in an alert state. The `alert` attribute's value is set to 0 if both of the `hi_alert` and `lo_alert` attributes' values are set to 0.

7.3.1.2 Digital Monitor Points

A digital MP's alert enable flag is first checked to see if the MP has alerts enabled. If its value is 0, no further action is taken. If it's set to 1, then the value of the `alert` attribute is checked.

If the value of the `alert` attribute is 0, then the MP's `value` is compared to the `alert_on1`

attribute's value. If they are not equal, then the digital MP will enter the alert state and the `alert` attribute's value will be set to 1. The current time will also be captured and stored into the digital MP's alert timestamp field. The counter governing how often the MP's `value` is sent to the archive while the MP is in the alert state is set equal to the time remaining before the MP's `value` would have been sent to the archive if the point had not entered the alert state. Finally, a message is sent to the alert task so it can generate an alert DP message.

If the value of the `alert` attribute is 1, then the MP's `value` is compared to the `alert_on1` attribute's value. If they are equal, then the digital MP will exit the alert state and the `alert` attribute's value will be set to 0. The current time will also be captured and stored into the digital MP's alert timestamp field. The counter controlling when the MP's `value` will next be sent to the archive is set equal to the time remaining before the MP's `value` would have been sent to the archive if the MP had remained in the alert state. Finally, a message is sent to the alert task so it can generate an alert DP message indicating that the alert has cleared.

One change is planned to behavior described above. For both analog and digital MPs, when a MP first enters an alert state the `value` of the MP will always be sent to the archive immediately after the MP enters the alert state, and every `aa_period` thereafter while it is in the alert state.

7.3.2 Archive and Screen Data Generation

Both use the same type of processing, so the following discussion is applicable to both.

Each MP has three counters associated with it. These counters control how often an MP's `value` is 1) sent to the archive when the MP is in an alert state, 2) sent to the archive when the MP is not in an alert state, and 3) sent out as screen data. The unit for the counters is system scan cycles. During system initialization these counters are set equal to the values specified, respectively, in the MP rate attributes `aa_period`, `a_period`, and `s_period` (see 5.1.2 Data Rate Control above).

At each system scan interval, the archive and screen data checking process goes through the entire logical points table in the MIB. Each MP's counters are decremented and then checked. If a counter is 0 or less, that MP is ready to send that type of data. A count of all points that are ready is kept. Once the entire table has been checked, if the count of points ready to send data is greater than 0 a message is sent to the archive or screen task as appropriate.

7.4 Data Port Common Routines

There are several common Data Port XML formatting functions in the MIB Framework. All operate on a caller-provided output buffer that is sized to be the maximum Ethernet message. There are Framework functions to start and end an `EVLAMessage` and the `device` portion of the output. Each type of MP (analog or digital) has its own archive and alert output formatting function. The screen data output uses the archive output formatting function.

7.5 Data Port Tasks

There are alert, archive, and screen tasks as stated previously which are used to actually create and send the DP messages. These tasks utilize the common DP XML routines along with specific routines for their functions to create the messages to be sent.

7.5.1 Alert Task Details

The alert task will examine the message it receives from the processing step to get the logical point index value to use, that point's `value` attribute's value, the values of the `alert`, `hi_alert`, and `lo_alert` attributes for the point (depending on the MP's type), and the timestamp it should use for the alert DP message. It then examines the logical point in that position to determine the type of MP, and generates a message for that point.

Alert messages will always contain information for just one device and one MP (see the example messages provided in this document).

7.5.2 Archive and Screen Task Details

Upon receiving a message indicating their type of data needs to be produced, these tasks will scan the entire logical points database looking for all points that are ready to send data of that type.

The archive or the screen task will create archive or screen data messages from the MPs that are enabled for archive or screen data reporting and have 0 or less in their counters for that type of data. For each type of data, a counter that is 0 or less is reset to the value of the `rate` attribute for that type of data.

As these tasks build the output message, they will track the amount of data written into the message buffer; if the amount of remaining space is not sufficient for the data the task is trying to write into the buffer, then that message is closed out (common XML routines are called to close the `device` and `EVLAMessage` elements), and the message is sent. The buffer is then cleared to all 0s and a new `EVLAMessage` and `device` element are started, with the `device` element's `name` attribute's value set to the device name of the current MP. Then the current MP's data is written to the buffer.

Each of these tasks will track the current device name for which it is generating data, and if the device name changes it will close out the previous `device` element and open a new XML `device` element for the new device name.

Once the entire table is scanned, any unsent messages are closed out, and the messages are sent.

8 MIB Device Points

The logical MIB Device supports a set of CPs and MPs, but none of them are configured for archive, alert, or screen output.

There are two CPs that are used by the DP; those were described above.

9 Appendix

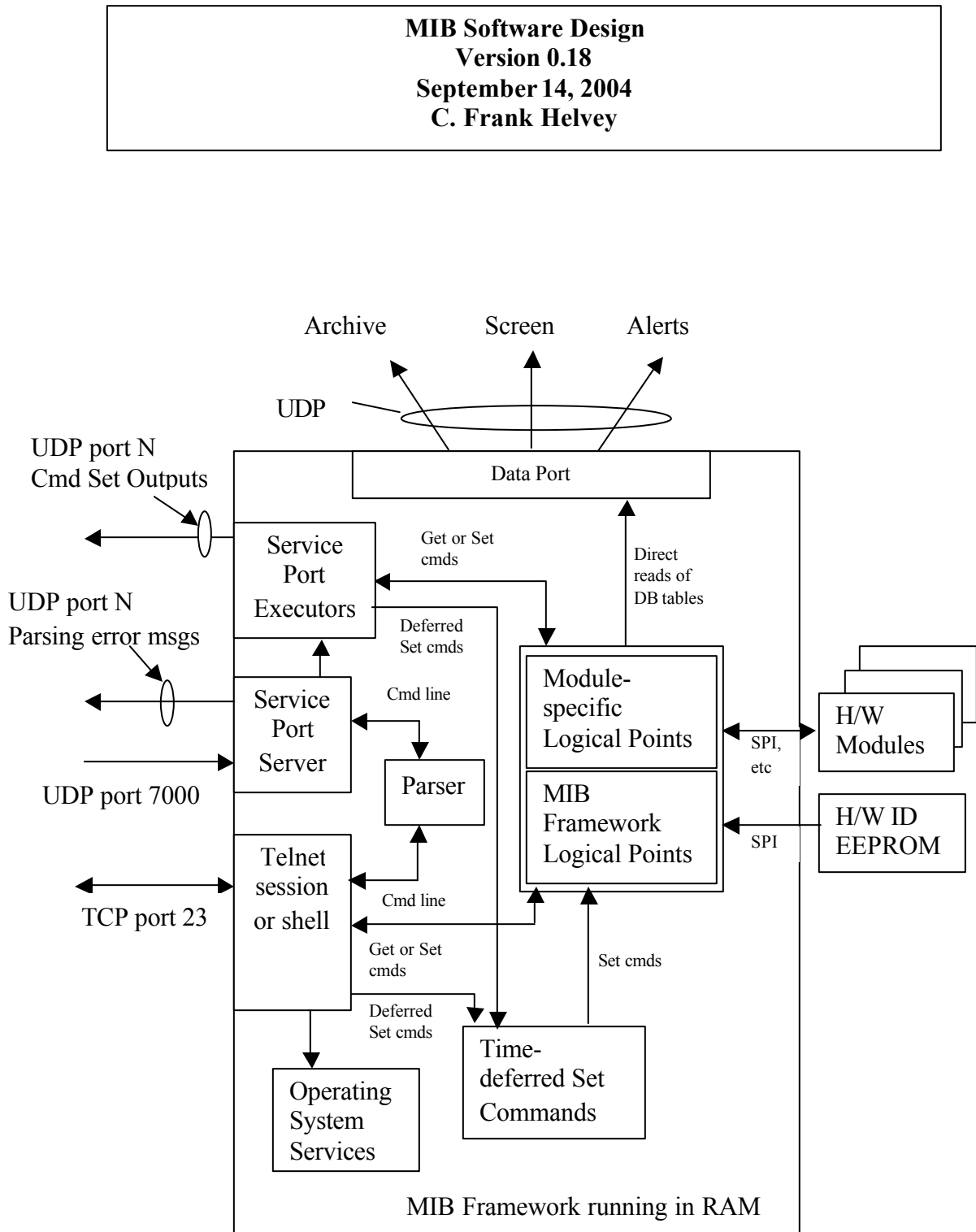


Figure 1. MIB Software Diagram

**MIB Software Design
Flash Memory Layout
Revision 0.18
September 14, 2004
C. Frank Helvey**

Table 4. Flash Memory Layout

Address Range	Description
0xA0000000 to 0xA0070000	Boot loader program (boot_bootloader 23-Apr-2004 or later)
0xA0080000 to 0xA00F0000	Current image, module software + MIB Framework software.
0xA0100000 to 0xA0140000	Spare
0xA0150000 to 0xA0170000	Module specific configuration information in XML format
0xA0180000 to 0xA01F0000	Spare
0xA0200000 to 0xA020FFFF	MAC address