3.4 ADM Functional Blocks

3.4.1 Database

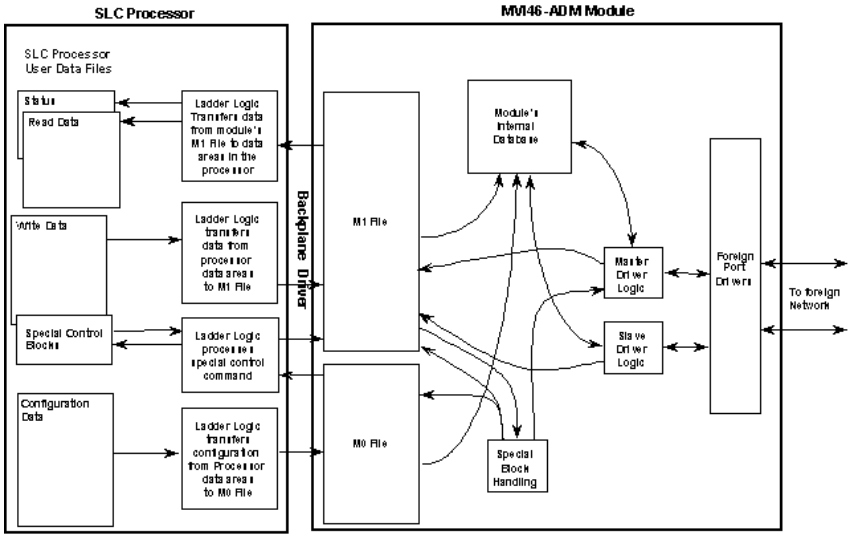
The database functions of the ADM API allow the creation of a database in memory to store data to be accessed via the backplane interface and the application ports. The database consists of word registers that can be accessed as bits, bytes, words, longs, floats or doubles. Functions are provided for reading and writing the data in the various data types. The database serves as a holding area for exchanging data with the processor on the backplane, and with a foreign device attached to the application port. Data transferred into the module from the processor can be requested via the serial port. Conversely, data written into the module database by the foreign device can be transferred to the processor over the backplane.

3.4.2 Backplane Communications

MVI46 Backplane Data Transfer

The MVI46-ADM module communicates directly over the backplane. All data for the module is contained in the module's M1 file. Data is moved between the module and the SLC processor across the backplane using the module's M-files. The SLC scan rate and the communication load on the module determine the update frequency of the M-files. The COP instruction can be used to move data between user data files and the module's M1 file.

The following illustration shows the data transfer method used to move data between the SLC processor, the MVI46-ADM module and the foreign network.



All data transferred between the module and the processor over the backplane is through the M0 and M1 files. Ladder logic must be written in the SLC processor to interface the M-file data with data defined in the user-defined data files in the SLC.

5.1 ROM Disk Configuration

User programs are stored in the MVI-ADM module's ROM disk. This disk is actually a portion of Flash ROM that appears as Drive A:.

The ROM disk size is:

Module Type	Disk Size
MVI46	896K bytes
MVI56	896K bytes
MVI69	896K bytes
MVI71	896K bytes
MVI94	384K bytes

This section describes the contents of the ROM disk.

Along with the user application, the ROM disk image must also contain, at a minimum, a CONFIG.SYS file and the backplane device driver file.

Module Type	File Name
MVI46	MVI46BP.EXE
MVI56	MVI56BP.EXE & MVI56DD.EXE
MVI69	MVI69BP.EXE
MVI71	MVI71BP.EXE
MVI94	MVI94BP.EXE

If a command interpreter is needed, it should also be included.

5.1.1 CONFIG.SYS File

The following lines should always be present in your CONFIG.SYS file:

MVI46

```
IRQPRIORITY=1
INSTALL=A:\MVI46bp.exe -iomix=0 -class=4 -m0size=3000 -m1size=10000
```

Note: The MVI46 driver file is called **MVI46BP.EXE**, and may be loaded from the **CONFIG.SYS** or **AUTOEXEC.BAT** files. The driver must be loaded before executing an application which uses the MVI API.

The SLC platform supports several classes of modules. The MVI46 can be configured as a Class 1 or Class 4 module. Also, the I/O image sizes are configurable. If the MVI46 is configured as Class 4, M0 and M1 files are supported and their sizes are configurable.

Note: Messaging is only supported when the MVI46 is Class 4.

To configure the class of the MVI46, use the command line options shown below when loading the MVI driver MVI46BP.EXE. If no options are given, the MVI46 MVI driver defaults to Class 4, 32 words of I/O, and M0 and M1 sizes of 1024 words (module ID = 13635).

```
[C:\]MVI46bp -?
MVI46 MVI Driver V1.00
Copyright (c) 2000 Online Development, Inc.
Usage:
C:\MVI46bp.EXE [-iomix=n] [-class=n] [-m0size=n] [-m1size=n]
where:
- iomix=n sets the I/O image sizes. Valid values for n are:
0 => 2 words of IO 5 => 12 words of IO
1 => 4 words of IO 6 => 16 words of IO
2 => 6 words of IO 7 => 24 words of IO
3 => 8 words of IO 8 => 32 words of IO (default)
4 => 10 words of IO
- class=n sets the module class. Valid values for n are:
1 => Class 1 (Messaging disabled)
4 => Class 4 (Messaging enabled, default)
- m0size=n sets the number of words for the Messaging
receive buffer, default m0size=1024
- m1size=n sets the number of words for the Messaging
send buffer, default m1size=1024
```

NOTE: m0size + m1size must be less than 16320 words.

When configuring the Host Controller for the MVI46, the programming software requires the Module ID for each module in the system. The Module ID for the MVI46 depends upon the configuration set by the driver. When the driver is loaded, it prints to the console the Module ID value that can be entered into the programming software for the Host Controller. For example, the default configuration prints the following information:

```
[C:\]MVI46bp
MVI46 MVI Driver V1.00
Copyright (c) 2000 Online Development, Inc.
1746 MVI Configuration
----------------------
Class 4
IO mix 8 = 32 words of IO
M0 File size = 1024 words
M1 File size = 1024 words
SLC Module ID = 13635
```

The first line, IRQPRIORITY=1, assigns the highest interrupt priority to the I/O backplane interrupt. The next line loads the backplane device driver. In this example, the backplane device driver file (MVI46BP.EXE) must be located in the root directory of the ROM disk. In the case of the MVI46, the module I/O is set when the backplane driver is loaded. The module is set to class 4 with a 3000 word M0 file and a 10000 word M1 file. The Module ID for installing and configuring the module in the ladder program will be printed to the console when the backplane driver is loaded.

If a command interpreter is needed, a line like the following should be included in CONFIG.SYS:

6.1 MVI46 Ladder Logic

6.1.1 Main Routine



ADM_SetConsolePort

Syntax

```
void ADM_SetConsolePort(int Port);
```

Parameters

Port	Com port to use as the console (COM1=0, COM2=1, COM3=2)

Description

ADM_SetConsolePort sets the specified communication port as the console. This allows the console to be disabled in the BIOS setup and the application can still configure the console for use.

MVI46 Note: The MVI46 should have the console disabled in the BIOS setup in order for the module to avoid faulting the processor on power-on boot. The console can still be used if the application uses ADM_SetConsolePort to enable console services and ADM_SetConsoleSpeed to set the baud rate.

Return Value

None

Example

```
/* enable console on COM1 */
ADM_SetConsolePort(COM1);
```

See Also

ADM_SetConsoleSpeed (page 187)

ADM_SetConsoleSpeed

Syntax

```
void ADM_SetConsoleSpeed(int Port, long Speed);
```

Parameters

Port	Com port to use as the console (COM1=0, COM2=1, COM3=2)
Speed	Baud rate for console port. Available settings are: 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600 and 115200.

Description

ADM_SetConsoleSpeed sets the specified communication port to the baud rate specified.

MVI46 Note: The MVI46 should have the console disabled in the BIOS setup in order for the module to avoid faulting the processor on power-on boot. The console can still be used if the application uses ADM_SetConsolePort to enable console services and ADM_SetConsoleSpeed to set the baud rate.

Return Value

None

Example

```
/* set console to 115200 baud */
ADM_SetConsoleSpeed (COM1, 115200L);
```

See Also

ADM_SetConsolePort (page 186)