ADI Interface Issues – Comments
T10/02-007r0

Rod Wideman
ADIC
31-Oct-01

These comments are based upon the initial draft from Paul Suhler, and follow the same section headings. Thanks, Paul, for kicking this off!

2 General technical requirements

Additional features and capabilities that we would suggest are:

- Support for a standard primitive command, such as0x00 (all drives would also start with the same communication parameters, 9600, 8N1)
- Ability of the library to set what the drive reports for serial number and ID (subset of Inquiry page 0x83 data)
- Ability of the library to retrieve exactly what the drive reports as Inquiry page 0x83
- Primary (data path) enable/disable capability
- Ability of the library to retrieve all information displayed on the front panel of the drive, including LED usage
- Ability of the library to emulate or control the drive front panel operations (pushbutton support)
- Ability of the library to reset the drive

Additional issues and concerns that we have are:

- Front panel information and serial port information must match (we have seen inconsistencies).
- If we use a SCSI based protocol for the serial port, the drive implementation must avoid this disrupting the primary data path operations (and vice versa). The library side should not have restrictions, nor require special knowledge of when SCSI based commands can or cannot be issued when the drive is processing data path commands.
- We would like to see consistent terminology established for “loaded” and “unloaded” (more below).

3 Application layer (commands)

3.2 Report Status

In general, we would like to see a separation of drive status from cartridge status. We would prefer that these be independently obtainable, and as concise and condensed as possible for performance reasons. We would rather retrieve a standard set of values (bits) that we can interpret, since the meaning of terms such as “Ready,” “Not Ready,” “Loaded” and the like can vary. With specific bits indicating specific states, the library can then draw its own conclusion.

For drive status, we would suggest something like:

Byte n				Compression	Tape Alert	Clean Required	Clean Recommended	Drive Initialized
Byte n+1	Access Allowed			Tape Ejected	Tape Inserted	Tape Loaded to MAM	Tape Loaded to Thread	Tape Loaded and Threaded
Byte n+2	Tape Motion Status							
Byte n+3	Drive Error Code							
Byte n+4	FLAG 08	FLAG 07	FLAG 06	FLAG 05	FLAG 04	FLAG 03	FLAG 02	FLAG 01
Byte n+5	FLAG 16	FLAG 15	FLAG 14	FLAG 13	FLAG 12	FLAG 11	FLAG 10	FLAG 09
Byte n+6	FLAG 24	FLAG 23	FLAG 22	FLAG 21	FLAG 20	FLAG 19	FLAG 18	FLAG 17
Byte n+7	FLAG 32	FLAG 31	FLAG 30	FLAG 29	FLAG 28	FLAG 27	FLAG 26	FLAG 25
Byte n+8	FLAG 40	FLAG 39	FLAG 38	FLAG 37	FLAG 36	FLAG 35	FLAG 34	FLAG 33
Byte n+9	FLAG 48	FLAG 47	FLAG 46	FLAG 45	FLAG 44	FLAG 43	FLAG 42	FLAG 41
Byte n+10	FLAG 56	FLAG 55	FLAG 54	FLAG 53	FLAG 52	FLAG 51	FLAG 50	FLAG 49
Byte n+11	FLAG 64	FLAG 63	FLAG 62	FLAG 61	FLAG 60	FLAG 59	FLAG 58	FLAG 57

Where:

- *Drive Initialized* indicates whether the drive is ready for operation, or whether drive initialization is required
- *Clean Recommended* indicates that the drive suggests a cleaning operation be done
- *Clean Required* indicates that a cleaning operation must be done before a data cartridge can be loaded
- *Tape Alert* indicates whether the drive supports reporting of Tape Alert information
- *Compression* indicates that compression is enabled on the drive
- *Tape Loaded and Threaded* indicates that the tape is fully loaded and threaded
- *Tape Loaded to Thread* indicates that the tape is fully loaded to thread point
- *Tape Loaded to MAM* indicated that the tape is loaded to memory readable point
- *Tape Inserted* indicates that the tape is present inside the drive (needs better definition)
- *Tape Ejected* indicates that the tape is present, but outside the drive some distance (needs better definition)
- *Access Allowed* indicates that the library may move a cartridge to or from the drive
- *Tape Motion Status* indicates the following:
  - 0x00 No tape motion is progress
  - 0x01 Cleaning operation in progress
  - 0x02 Firmware upgrade in progress
  - 0x03 Tape is being loaded
  - 0x04 Tape is being unloaded
  - 0x05 Tape in motion
- *Drive Error Code* is a drive reported error, which could be hardware, media, or both (0x00 would be no error)
- *Flags 01-64* are the Tape Alert flags

With this type of approach we can obtain all the information we need to understand the state of the drive in only 12 bytes. This type of information tends to be frequently retrieved, so a small number of bytes is advantageous.

In addition to this frequent drive status information, we would also suggest a second drive counter or statistics data set that can be obtained. This might include:

- Total number of loads

- Total number of cleans
- Time since last clean
- Time since power on
- Error counters, etc.

For cartridge status, we would suggest a separate data set that can be obtained for an individual cartridge when it's loaded. This might include:

- Number of loads
- Tape type
- Format type (or density)
- Capacity used
- Capacity remaining
- Label
- Cleans used (for cleaning tapes)
- Cleans remaining (for cleaning tapes)
- MAM data
- Error counters

3.3 – 3.6 Load/Unload

We would like to establish a consistent set of terminology to use when describing loaded and unloaded states. From a library perspective, the most important information pertains to the physical position of the cartridge. Towards that end we would suggest:

- Ejected – cartridge is as physically far out of the drive as can be controlled by the drive
- Inserted – cartridge is as physically far into the drive as can be controlled by the drive, which may or may not be equivalent to the load point, depending on how many other mechanical positions exist
- Loaded and memory chip readable
- Loaded and thread-able
- Loaded and threaded

Some of these positions (such as the middle three) may be the same, depending on the drive and its load mechanism. These are only suggestions, but the intent is to capture the physical positions where the cartridge could be. We should probably also indicate whether the drive is still able to manipulate the cartridge at these positions, especially the ejected position.

3.7 Set Operational Modes

We would suggest a corresponding Get Operational Modes command. We would also suggest a "vendor unique mode" which would just be some number of bytes that can be passed to the drive for interpretation, as defined by the vendor.

3.7.3 Baud Rate

We would suggest not saving a changed baud rate across power cycles. This should always be negotiated upon power up for full recovery ability. We also need to define when a change to baud rate takes affect (hopefully after the change command acknowledgement). We would suggest 9600 as default.

3.7.4 Stop bits transmitted

Selectable is fine. Again, we would not want this saved across power cycles. We would suggest one stop bit as default.

Nothing is mentioned for Parity, but we would suggest no parity.

The default behavior would then be 8-N-1, 9600 baud.

3.7.6 Alerts

In addition to being enabled or disabled, we need to decide whether they can be sent asynchronously or not (we would prefer they not be), and how they are cleared or consumed (and if they affect the primary path). We would suggest alerts also be selectively enabled (configurable).

3.7.8 – 3.7.9 Autoload/Autounload mode

We like this concept, and would suggest for both loading and unloading we allow behavior to be set by cartridge type. We would suggest three types – Data, Cleaning, and Firmware. And in addition to defining the behavior, we would suggest the behavior be set to either "automatic" or "wait for instruction" to provide the most flexibility.

4 SCSI command transport

In general we like the idea of converging onto SCSI as the protocol. However, we think it is very important that the implementation does not interfere with the data path operation, and that the library has full operational concurrency in using it.

If SCSI is used as the serial port approach, we also see value in having any SCSI extensions created also made available on the primary interface. In some bridged environments (e.g. routers), the possibility to eliminate the use of the serial port and thus reduce cost is real. This requires any capability available through the serial interface also be available through the primary interface.

- Tagged queuing – we think not
- Contingent allegiance – as a general rule, library activity should be transparent to the primary path, and vice versa
- Reservation conflict – by the above rule, no

5 Logical unit bridging

We like this capability, but we would suggest that this be an additional option over and above the standard interface protocol, such that each drive can chose whether to offer this extension. This can be indicated as part of the protocol support in the primitive. The library should first enable this extension for use.

Additionally, we would suggest that logical unit bridging be strictly a pass-through operation, such that the drive does not do any additional processing for commands to LUN 1 (at the CDB level).

6 Topologies

We would prefer point to point.

7 Link Layer

7.1 Protocol type detection

We prefer a standard primitive command, such as 0x00.

7.3 End of frame recognition

We prefer a length field.

7.4 Frame ACK/NAK characters?

We prefer more detailed information as to what was wrong with the frame, instead of just a generic “bad frame.”

7.5 Error detection and recovery

Simple checksum should be sufficient. Hamming distance > 1 would be nice.

8 Physical layer

8.1 Connector type

Per one of our engineers that has interfaced to many of the existing drives…

“Ideally, a connector that mates to an IDC ribbon cable connector would allow for ease of manufacture and mass termination of cables...saves much money. The connector should be polarized AND provide some type of locking/latching. At the least, the connector should support mates that accept a range of wire gauges. Small is good, but if the goal of this universal serial interface is ease of integration, then other flexibilities need to be considered.”

8.2 Cable length

Maybe a maximum length should be specified.

8.3 Levels

We would prefer RS-422.

8.6 Stop bits

Selectable.

Parity?