ADDENDUM TO MIDI INTERFACE USER GUIDE

FOR USE WITH MIDI INTERFACE USER GUIDE ISSUE 1

INFORMATION FOR MIDI MODULE (UPGRADE TO I/O PODULE) USERS

The Midi Module does not support the mode of operation set by:

SWI "MIDI_SoundEnable" with R0=1 (connection to the MIDI IN buffer)

This mode allows an external MIDI instrument to drive the Archimedes internal sound system (see page 16). Although this mode can be used to verify software function (as in the example in the chapter *Getting Started*, pages 7/8), with fast MIDI data such as chords or pitch bend, overrun errors will occur.

The effect of this for Midi Module users is that the Archimedes cannot itself be used as an instrument, only as a controller or editor for other instruments.

The dedicated Midi Podule does however have the appropriate additional buffering to allow correct operation of this mode.

Changes to documentation:

The following information, given in the chapter *Introducing MIDI*, section 'What the interface can do', is **not** applicable to MIDI Module users:

By installing the MIDI interface in your Archimedes and connecting a music keyboard, you can use the computer as a musical instrument. The sound can be made by the speaker on your computer, or you can plug in headphones, or your hi-fi system.

Appendix A:Arthur MIDI Specification refers to the MIDI interpreter and sound system driver being driven directly by data from the MIDI IN socket (see pages 11, 14, 16). This information is **not** applicable to MIDI Module users.

ERRATA TO MIDI INTERFACE USER GUIDE

Applicable to both MIDI MODULE and MIDI PODULE users

Page 8

The last sentence should read:

Button 3 will activate Archimedes voice 3 and so on.

Page 13

In the section CHANNEL MODES, the Mode 4 entry should read:

Voice messages are received in voice channels N to N+M-1 and assigned monophonically to voices 1 to M, respectively. The.....etc

Page 18

The description for SWI "MIDI_InqBufferSize" should read:

Return the number of empty bytes in the transmit or receive buffer. The maximum size of these buffers is 128 bytes, and they can fill (rx buffer) or empty (tx buffer) at a maximum rate of 320 microseconds per byte.

Page 20

The entry for SWI "MIDI_RxCommand" should read:

On exit: R0= bytes:

byte 0 = status
byte 1 = data byte 1, or 0
byte 2 = data byte 2, or 0
byte 3 = number of bytes in command

NOTE: Byte 3 returned by SWI "MIDI_RxCommand" in R0 is the total number of bytes comprising the command (excluding the timestamp) and thus is the number of data bytes + one. The 'MIDI Specification 1.0' specifies the number of data bytes with each command, which is thus different by one. (See the 'Summary of Status Bytes' table, pages 34/35 and Appendix C:Bibliography for further information on the MIDI Specification 1.0)

Page 21

The entry for SWI "MIDI_TxNoteOff" should read:

R1 = key off velocity, 0-127

Page 22

The entry for SWI "MIDI_TxNoteOn" should read:

R1 = key on velocity, 0-127

Page 29

The section 'MIDI Timing Clock transmission counting', paragraph three should read:

For example, if the sound system is programmed to the default tempo of &1000(=100 per second), then, when enabled by SWI "MIDI_TxStart", "Timing Clock" messages will be transmitted at a rate of 6.25 per second", and the Song Position Pointer will increment.....etc

ADDITIONAL INFORMATION AND NOTES FOR PROGRAMMERS

Applicable to both MIDI MODULE and MIDI PODULE users

1 The Archimedes sound system beat counter is 'off' by default (BEATS=0), so BEATS must be set to a positive value to set the beat count in operation and give non-zero values for internal BAR/BEAT time-stamping. (See section *Timing*, page 28.)

2 If a sound module (SoundDMA, SoundChannels, SoundScheduler) returns an error to the MIDI interpreter, for example, if the sound module were removed, the background MIDI task is automatically disabled completely. The MIDI module, although apparently still present, ceases to function correctly. To restore correct operation in this case, the sound module that caused the error must first be reinstalled and then the MIDI module reinitialised using *RMREINIT MIDI.

3 The pitch bend algorithm used by the internal sound system driver is such that if notes are newly triggered while pitch bend is in operation they will start with the normal pitch, and will only be pitch-changed when the next pitch-bend message is received.

4 The MIDI interpreter does not respond to notes of very short duration, of less than about 20ms. This means that certain drum machines cannot be used to trigger notes on the Archimedes sound voices.

5 In the Arthur operating system, the overhead involved in calling a SWI can be up to about 200 microseconds regardless of how simple the SWI is. As this time is not much shorter than the time between receiving successive MIDI bytes when operating at the maximum data rate (320 microseconds per byte), it is evident that a program should attempt to use SWIs in the most economical manner possible, and in any case should average fewer than one SWI per byte received. Any program should be tested for receive buffer overflow under conditions of maximum MIDI data rate.

6 The internal MIDI interpreter will discover and update certain operations only about every 200ms. These are:

- change of BEATS
- change of TEMPO

Some operations that work independently of the state of the MIDI interpreter are updated immediately that the complete message is received. These are:

- "O", "F" or "B" error status
- Song Position Pointer
- Start
- Timing Clock
- Continue
- Stop
- Active Sensing

All other messages that act upon the internal MIDI interpreter are interpreted every 10ms, though they will be returned through SWI "MIDI_RxCommand" as soon as they are received.

This information is important if, for example, you have coupled the MIDI interpreter to the outgoing data, and your program sends a System Reset message immediately followed by another Real Time message. For example:

```
SWI "MIDI_TxSystemReset"
SWI "MIDI_TxStart"
```

In this case the System Reset will not take place until the next centisecond tick, which will probably be after the Start message was received and acted upon. The System Reset will then clear the Started state and the end result will be as if the Start message was never received. This also applies if an external instrument sends a System Reset followed immediately by another Real Time message if the internal MIDI interpreter is coupled to the incoming data. If the MIDI interpreter is not coupled then the System Reset will not have any background effect and hence this will not happen.

Published by Acorn Computers Limited

January 1988
Part Number 0476,212