Apple //e Annunciator 0 MIDI Output

This project involves building a circuit that interfaces with the Apple //e internal Joystick socket connector, and running software which:

- Reads Format 0 Standard Midi Files stored on the ProDos operating system
- Sends MIDI output via the Annunciator #0 pin on the joystick connector
- Displays the names of the notes being played in real time on the Apple //e 80 column screen.

The hardware and software driver may be used on a 64K Apple //e, but the Midi File player software requires an extended 80 column card (which contains 64K memory on board). The software will work on an Unenhanced Apple //e, but the 80 column lyrics display will have glitches

The circuit comes from the Midi 1.0 Spec dated August 5, 1983.

Contact info for obtaining the spec:
International MIDI Association
12439 Magnolia Blvd
North Hollywood, CA 91607
818-505-8964

It can also be found in the book: ***Synthesizers and Computers*** from the Keyboard Magazine Basic Library - published by GPI Publications and Hal Leonard, 1985, 1987. ISBN 0-88188-716-1

The following pins are used on the Apple //e joystick socket inside the case:

Pin 1 = +5V
Pin 8 = GND
Pin 15 = AN0

First test your connections by wiring an LED with a resistor between AN0 and +5V. The short lead of the LED is the negative terminal that goes to AN0.

To turn the LED on: PRINT PEEK(49241)
To turn the LED off: PRINT PEEK(49240)
In hex: $C059 turns it on, $C058 turns it off.

The MIDI 5 pin DIN socket is wired up as follows:

Pin 15 AN0 --> Inverter --> Inverter --> 220 ohm resistor --> Midi Out Pin 5
Midi Out Pin 4 --> 220 ohm resistor --> +5V Pin 1
Midi Out Pin 2 --> GND Pin 8

The inverters are on a 7404 IC (2 of the 6 Hex inverters are used).
http://www.makeyourownchip.com/7404.html

Current ON is logical 0, so to send a logical 1 you turn the annunciator off.

To send midi data, you need to send 10 bits per byte at 31.25K Baud.

The Start bit is a logical 1 (no current: 0) on AN0.
Then 8 data bytes are sent for the MIDI byte: D0 thru D7
so you send the least significant bit first, following the protocol of inverting the data bit to convert the logical bit to current on/off.
The Stop bit is a logical 0 (current:1) on AN0.
The line stays high until the Start bit for the next byte.

The driver program, **MIDIDRVR.OBJ**, is an assembly language subroutine that sends one midi byte out AN0 by precisely timing the delay loop between bits to achieve 320 microseconds per serial byte. An Apple //e has a 1MHZ clock, and machine language instructions have specific cycle times, so it is possible to calculate precision timings for delay loops. Midi File playback is done using: **MFF0PLAY.OBJ** and **MFF0DRVR.OBJ**.

Apple // Oasis is an Apple][series emulator for Windows PC's available from:
http://www.geocities.com/apl24win/DOWNLOAD.HTM

The .DSK image supplied with this download can be loaded by the **Apple // Oasis** emulator. The help file for the ADSERVER program has info about how to build a cable to transfer the disk image from a PC to an apple floppy disk. The disk a 5.25 140K floppy in Prodos format. If you get it to work, it boots up a basic program for selecting a song, loads it, and runs machine language programs that drive the annunciator and display the notes being played. There are 3 sample songs on the disk - they have file types of $D7 which is the Prodos file type for Standard MIDI.

A good reference book for Apple //e hardware and programming is:
Inside the Apple //e by Gary B. Little, published by Brady Communications Company and Prentice-Hall. ISBN: 0-89303-551-3

Have fun!
Eric Rangell
erangell@gmail.com