

What is the MIDIo?

The MIDIo Programmable Visualizer is a platform designed to allow real-time control of timed visuals for performance or entertainment. The kit is designed with musicians and visualists in mind, allowing people to pre-program designs or perform realtime using already existing music interfaces and software such as Renoise, Ableton, EnergyXT, Reaper, Max/MSP/Jitter, PureData, and devices such as the Monome, Bliptonome, MIDI grids, touchpads, and keyboards.

The main goals in the kit's design are:
affordability, having a DIY Kit Version Available, upgradability, and user programming.

The kit was designed with a low parts-count and most work being done in software. Power is drawn from a computer, eliminating bulky and problematic “Wall Warts” and power circuits. All control is done via midi, allowing the user to decide on their own interface while eliminating expensive switches & knobs.

The kit is designed for an intermediate kit solderer, and a pre-built versions can be had.

The ANSI firmware resides on a single 8-pin chip. As the kit is designed in a modular way, alternative firmwares can be used by simply switching out the chip for another. This allows for cheap upgrades, new visual and programming systems, and an ability to use more than one visualizer system without buying new hardware for each.

What is the ANSI MIDIo?

The ANSI MIDIo is the first in a series of pre-programmed firmware modules for the MIDIo platform. Though the MIDIo platform could allow any translation of midi data to video output, the ANSI MIDIo specifically creates an ANSI art paradigm known throughout the 1980s and 90s via online “Bulletin Board Systems ” (BBS). The firmware gives the user the ability to generate realtime patterns or create custom artwork such as those seen in Art Packs and on the BBSs of the era. As such, the system was loosely based on the 12 most important ANSI characters, and the same 16-color palette used on old DOS-based computers and other systems that could interpret ANSI via a Terminal Emulator.

Installing the Firmware:

If the ANSI_MIDIo firmware is an upgrade or firmware change from a different MIDIo firmware the 8-pin IC on the MIDIo will need to be replaced, as each 8-Pin IC holds it's respective firmware giving the MIDIo unlimited potential functions. To change or replace the firmware, simply disconnect the power, and carefully remove the 8-pin IC from it's socket by sliding a small blade underneath it's end (between IC and Socket lip) and twisting back and forth. Slightly bent or mis-formed pins can be gently bent to a correct straight position with little fear of damaging the IC. The old firmware can be replaced at any time by following the seating method below. Store the old IC in the static foam or static holder provided with the new firmware IC.

To install the new firmware, verify that the new IC's legs are perpendicular to the IC body. They can be gently corrected with a fingernail or small blade if needed. It is important that the IC is facing the

correct direction when installed. Looking at the IC's "top" (back of the bug, legs away) note a small semi-circle cut into one end. This semi-circle must line-up with the small notch in the IC socket closest to the nearby metal canister (crystal – IC clock). Verify this direction, and that all 8 legs are inside the socket lip, then press down firmly. A slight rocking pressure on the back of the IC will allow the IC to seat and align with the socket. A tiny air-gap between IC and socket is normal.

How it works:

The system is mostly software based as it keeps parts-count and cost to a minimum. As such, control of the kit is created via a modular system:

human → software or controller → computer midi → KIT_MIDI driver → FTDI cable → MIDIo Kit → TV, monitor or projector → eyes → brain

or

human → midi controller → midi → 5-PIN DIN Module → MIDIo Kit → TV, monitor or projector → eyes → brain

In order for this chain of modules to work, a certain amount of software setup is required. The system should work roughly the same way on various flavors of Windows or OSX, though there will always be some differences. I'll address the modules above in reverse order as the higher level modules are dependent on everything else working:

TV, monitor or projector

I've used a range of monitors, projectors, TVs, LCDs, video-mixers, and computer capture devices with this circuit. In general, your device needs to be capable of receiving a composite signal, which is VERY common. Depending on your firmware version the MIDIo kit can produce PAL or NTSC (but not both). You will need a common RCA (male-to-male) cable to route the signal from the MIDIo kit to your chosen viewing device.

MIDIo kit

The kit is a fairly simple circuit as most functions are handled in software on the main microcontroller. The kit receives its power and the MIDI signal from your computer via the FTDI cable. The microcontroller receives and parses the MIDI messages, and then outputs video through a small resistor network to the composite RCA jack.

The kit only includes a few other important components: 2 buttons & 2 LED lights : 1 LED will light whenever power is applied to the kit. The other will blink 5 times demonstrating proper health on startup. It will then turn "on" for any received MIDI "noteOn" messages received, and turn "off" for any MIDI "noteOff" messages received. This is designed to help with troubleshooting if no video is produced.

The PCB mounted button is used only as a reset button. This action will clear all midi values to their default, and reset the microcontroller, which will go through the blink process and wait for its first midi command. No settings are preserved. For more information on the kit's circuit, please see the "How-to-build" documentation located at <http://www.straytechnologies.com/resources> .

The button on the top of the 40-pin IC is to control Midi mode and is explained below in the section marked **Direct MIDI Control (computer free!)**.

FTDI Cable and Drivers

The MIDIo receives its command and power directly from a computer's USB port via an FTDI cable or USB-to-Serial FTDI breakout board. This cable maintains the chore of converting modern USB to something the controller can handle, while allowing the option for 6-pin control by other hardware or microcontroller based systems.

Cables or breakout boards are available with the kit or here:

<http://search.digikey.com/scripts/DkSearch/dksus.dll?Detail&name=768-1028-ND>

or here:

<http://www.moderndevice.com>

or many other hobby electronic shops.

These cables require a “virtual com port” driver to be installed on Windows, OS X, or Linux which is available here:

<http://www.ftdichip.com/Drivers/VCP.htm> in MANY flavors.

Users will need to know how the device is referred to on their specific system. On Windows, check your 'Device Manager' for the newly installed Port (COM & LPT) to find the setting: something like “COM3” is normal. On OS X and Linux the installed driver should show up as something like “/DEV/TTY/USBVIRTUAL-FTxxx” in the KIT_MIDI software mentioned below.

KIT_MIDI_V2 Synth DRIVER software

This software will be occasionally updated, with the newest version available at:

<http://www.straytechnologies.com/resources/>

The software is available in WIN, and OS X versions.

This is Java based software, and will need to be run every time the video-synth is used from a computer. On opening the software (yeah..it can start slowly) the user will need to click the check box for the Serial Port supplied by the FTDI driver mentioned above. The user then needs to select what virtual midi “port” or “virtual cable” the video-synth will be connected to. After a few seconds of internal software chugging, the driver will display “Ports Connected” and can be minimized (but needs to be left running) for the duration of your session. If you're running multiple kits, each physical kit will need its own running copy of the KIT_MIDI driver addressing its own FTDI cable.

I've had great success with this system, though Windows users will have a smoother experience if the Serial Port is always selected before the Midi port. I've also found that plugging-unplugging hardware controllers such as the M-Audio Key Stations (or the kit for that matter) sometimes causes some panic in the software, and will likely require a restart of the software.

Midi

Software use of the synthesizer will require some sort of Virtual Midi Cable system to be installed on your system before using the KIT_MIDI Driver and the MIDIo kit. If you don't already use midi software, setup is somewhat different on various operating systems, and is quickly outlined below.

OS X based systems:

On OS X there is a built in system called IAC (Inter Application Communication) which can be found in Applications > Utilities > Audio Midi setup. Click on IAC Driver in the Midi Window, and verify that the “Device is online” box is checked. Under “More Information,” you can add additional midi busses or ports as needed. In KIT_MIDI and other software these should show up as “IAC DRIVER MIDI 1” or whatever name/number your port is.

Folks running pre- Mac OS X 10.6 or those who have reverted Java to an earlier version will discover that Apple and Java do not always play together well. In this case you may need to install “mmj” midi

for Java, as Apple evaded implementing midi in some java updates. Info and the files are at :
<http://www.humatic.de/htools/mmj.htm>

WIN based systems:

In Windows you'll need a "Midi loopback device" installed on your system. There are several versions such as "LoopBe1" and "Hubi's Midi loopback system", found on the Internet with their respective installation instructions.

I can't speak highly enough of Jamie O'Connell's "Midi Yoke" at:

<http://www.midiox.com/index.htm> this program has been around since the AOL/CompuServe days and is still maintained and works great! LoopBe1 from <http://nerds.de> is tested and working on 64bit Windows 7. With either solution, the virtual midi device is installed like real hardware and appears to be an actual midi port to the OS and software. It will show up in the KIT_MIDI software under it's respective name in the right hand column, once correctly installed.

Linux based systems:

This is a WIP, and Linux users are probably already used to doing things on their own. So, I will simply point you to the ALSA library's virtual midi card "VirMidi" and hopefully fire all of this up on my midi box for detailed installation instructions in the near future. There are also Python options for those ready to explore the nuts and bolts.

Direct MIDI Control (computer free !)

As an alternative, the MIDIo device has been designed to respond to a standard MIDI signal without using a computer or FTDI cable. This requires a MIDI module to be "plugged into" the 6 pin FTDI header. The module supplies the interface for direct MIDI control of the MIDIo, as well as filtered power from an external power supply. (See Midi Module documentation and the blog at www.straytechnologies.com for more information)

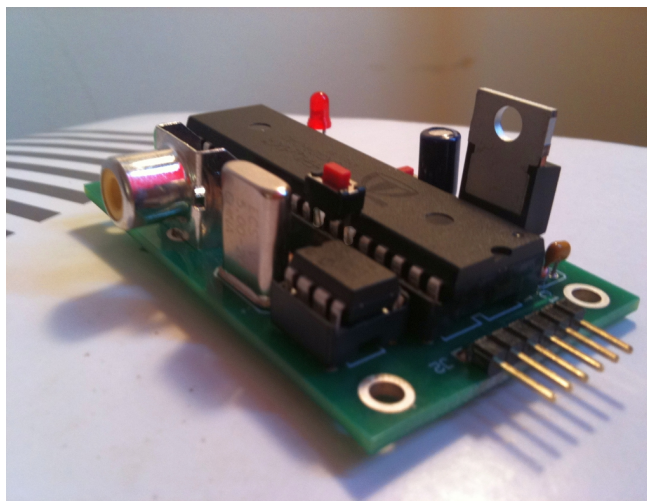
Normally the device defaults to FTDI mode, and expects signal and power to come from an FTDI cable attached to a computer. If direct MIDI control will be used (Computer Free Mode). The MIDI Mode button (bridging pins 33 & 35 on the 40-pin IC itself) needs to be pressed and held-down during power-on or reset (using the reset button). Hold the button down until the green LED blinks. The LED will blink only 3 times (instead of the normal 5-times) signifying it is ready for direct 5-pin MIDI control. This is only maintained for one "power" cycle, and will need to be done every time the MIDIo is started while used with a MIDI module. It will otherwise default to five LED blinks and FTDI control only, ignoring 5-pin MIDI based signals.

Controlling the ANSI MIDIo Kit

Control of the kit can be quite simple or complex depending on your level of interest. I've included an ability to draw logos and images, though a simple realtime triggered pattern-generator may be enough for many live performances and testing. For those wanting to control the kit with finite detail, the MIDI specifications on the last page of this document should be at your fingertips for the foreseeable future.

In short, the system has been tested and works happily with Renoise, Reaper, EnergyXT, PureData, Max/MSP, MidiOx, Bliptronome (Arduinome/Monome) hardware and the Oxygen8 hardware midi controller.

I firmly believe that the implementation is hardy enough to handle control from any software that can connect to a virtual midi cable. I've also tested the system on several OS's and various machines including a sluggish Dell netbook, and find that it's very stable. That said, remember that you're simultaneously running a LOT of software, and your Aunt's Pentium-I may not be up for the challenge. Faster systems will mean less midi timing glitches and Java crashes. Running 5 VST's, FruityLoops, browsing the Interweb, and doing your taxes at the same time will mean more problems.



GENERAL USAGE (How to Draw)

The ANSI MIDIo has three main modes and several effects all controlled via MIDI NoteOn, NoteOff and ControlChange Messages. The output can be thought of as a small Ansi terminal on a very old computer or a BBS being viewed from a Telnet program via modem. Characters are written from left to right, top to bottom in sequence. When the bottom line is completed, the entire visual screen is scrolled up by one row and a new row begins. The entire screen is composed of 32 characters across by 12 characters down with an endless vertical scroll.

To draw a character on the screen in the default "DRAW MODE", a NoteOn command is sent with two components: Note & Velocity. In the case of the MIDIo, Any note value is converted to one of 12 graphic character regardless of note octave. C = Char 1, C# = Char 2, D = Char 3 and so forth... (see Character table on MIDI Implementation page). The velocity is similarly derived from the velocity of the note, such that one of 64 different 2-color palette entries can be used. Velocity of 1 = palette 1 color pair, 2 = palette 2 color pair, and so forth. (see Color Palette table on MIDI Implementation page). Velocity of 64 (hex 0x40) produces a special black "space" regardless of the note/character selected. In this way notes are drawn to the screen in a selected order. Large scale images can be drawn and scrolled up the screen in this manner.

Ex:

C-4 40, C-4 08, C-4 40, C-4 08 would produce “black,red,black,red” in order using a solid character. E-4 13, F-4 13, E-4 13, F-4 13, would produce the beginning of a green & blue checkerboard pattern.

There are 2 additional commands to make life easier, which are available via MIDI Control Change messages. Any MIDI Value sent to CC26 (hex 0x1A) will produce a single NEWLINE making the next character appear on far screen left one line down: Sort of an ANSI “carriage return.” Any MIDI Value sent to CC27 (hex 0x1B) will clear the screen completely. The next sent character will be placed in the top-left most position as if the MIDIo kit was restarted.

MODES

There are three pattern modes selectable on the ANSI MIDIo via Control Change CC22. Once a mode is selected, all messages sent will be for that mode until CC22 is changed again.

DRAW MODE is the default mode explained above. This can be selected with a VALUE of 1 (hex 0x01) sent to CC22.

REPEAT MODE is the same as DRAW MODE, with multiple characters sent for each NoteOn message sent. The VALUE sent to MIDI CC22 2-127 (hex 0x02-0x7F) will produce that number of repeated blocks. This creates an effect of a faster drawing time and elongated visual patterns.

INFINITE MODE reacts to both NoteOn and NoteOff messages and is selected by sending a value of 0 (hex 0x00) to MIDI CC22. A NoteOn message will produce a continuous stream of the chosen character and color until either another NoteOn is sent (replacing it), or a NoteOff is sent (terminating the stream). This creates very fast pattern changes, and can be used as a color organ where pre-drawn designs are not desired.

EFFECTS

There are three built-in effects accessible with three other MIDI Control Change commands as documented below. Effects can be used simultaneously in any combination.

REPEAT SPEED can be changed for REPEAT MODE or INFINITE MODE with MIDI CC23. Any value 0-127 (hex 0x00-0x7F) is interpreted as a different draw speed between MIDI noteOn commands. The default is the fastest speed 127 (hex 0x7F). 0 (hex 0x00) is the slowest speed, designed to roughly simulate a 9600 baud modem speed from 1980. REPEAT SPEED is overridden by the speed or tempo at which you send commands. Any new command will cut-off an incomplete previous command to maintain correct timing.

VERTICAL MIRROR MODE is controlled via MIDI CC24. Any value sent greater than 64 (hex 0x3F) will split the screen in half vertically and mirror the visual stream. Any commands sent will be mirrored until the effect is turned-off by sending a value less than 63 (hex 0x3F) to CC24. The screen is not cleared or erased during this effect change.

HORIZONTAL MIRROR MODE is controlled via MIDI CC25. Any value sent greater than 64 (hex 0x3F) will split the screen in half horizontally and mirror the visual stream. Any commands sent will be mirrored until the effect is turned-off by sending a value less than 63 (hex 0x3F) to CC25. The screen is not cleared or erased while this effect is changed.

Caveats (or “Features”)

MIRROR MODES can be mixed and used together!

There is no polyphony– if you send multiple noteOn messages without a noteOff, only the last-sent note/character will be rendered.

Any Drawing will be halted if a new command is sent before completion. This is done to maintain visual timing when syncing the work to music.

A noteOn command with zero velocity appears the same as a noteOff command to the MIDIo software.

Settings are not remembered during power-off.

The unit will always default to FTDI mode when reset or powered-on, unless the MIDI mode button is pressed during start-up or reset.

There is no scroll-back buffer. Once a design leaves the screen it is gone from the MIDIo memory.

Let me know how it goes. Feedback means better firmware. Awesome videos online and example midi or tracker files, means inspiration to keep working. Drop me a line, and let me know how it goes.

-Wil

wil@straytechnologies.com

ANSI MIDIo : Current Midi Implementation

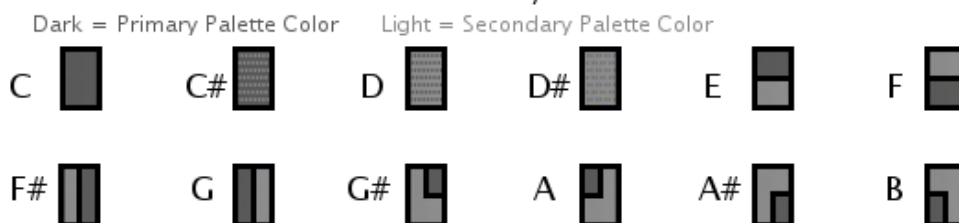
All messages will respond to CH 16 (0x0F) only

Command	Data1	Data2	ANSI MIDIo control
NoteOn	Note 0-127	Velocity 0	Same as NoteOff
	Note 0-127	Velocity 1-63	Draw character/s (n/1-12), color palette (v/1-63)
	Note 0-127	Velocity 64	Draw black "space"
	Note 0-127	Velocity 65-127	Draw character/s (n/1-12), color palette (v/1-63)
NoteOff	Ignored	Ignored	Stop Drawing in INFINITE REPEAT MODE

Midi Ccs CH 16 (0x0F) only

CC22	0x16	VALUE 0	INFINITE MODE repeat noteOn until noteOff
		VALUE 1	DRAW MODE : draw one character on noteOn
		VALUE 2-127	REPEAT character 2-127 times on noteOn
CC23	0x17	VALUE 0-127	REPEAT SPEED : 0-127 (127 is fastest)
CC24	0x18	VALUE 0-63	NO VERTICAL MIRROR (mode off)
		VALUE 64-127	VERTICAL MIRROR MODE (mode on)
CC25	0x19	VALUE 0-63	NO HORIZONTAL MIRROR (mode off)
		VALUE 64-127	HORIZONTAL MIRROR MODE (mode on)
CC26	0x1A	VALUE ignored	NEWLINE (any value sent = 1 newline)
CC27	0x1B	VALUE ignored	CLEARSCREEN (any value sent)

Character Created for Each Note in any Octave:



Color Palette for Each Hex Velocity Value:

	40	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
Primary	Black	Grey	Grey	Grey	Grey	Grey	Grey	Grey	Red	Red	Red	Red	Red	Red	Red	Red
Secondary	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey
Primary	Green	Green	Green	Green	Green	Green	Green	Blue	Blue	Blue	Blue	Blue	Blue	Blue	Blue	Blue
Secondary	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey
Primary	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow	Yellow	Purple	Purple	Purple	Purple	Purple	Purple	Purple	Purple	Purple
Secondary	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey
Primary	Teal	Teal	Teal	Teal	Teal	Teal	Teal	White	White	White	White	White	White	White	White	White
Secondary	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey	Black	Brown	Green	Blue	Olive	Purple	Teal	Grey