One of the projects in the Alten Tech Challenge is the MIDI-2-USB project. This project is about the use of older music synthesizers in a modern environment. Many of those instruments had progressive technology at their time: they have non-volatile memory to store parameters and also a MIDI[1] connection. Compared to modern instruments they do have some disadvantages:

- Every manufacturer and each model used its own dedicated system for memory storage cards. Of course these were not compatible with one another, let alone with modern memory storage systems.
- USB did not exist at that time
- MIDI[1] implementation was limited primarily to musical information. System exclusive data dumps were regarded as a gadget.

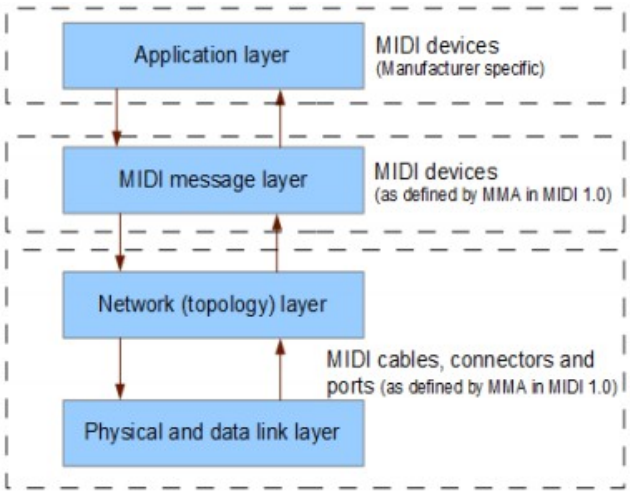
You may find a good overview of some of these older instruments on my web page[5].

Goal of the MIDI-2-USB project is to increase usability of older instruments by coupling system exclusive dumps to a USB drive or USB memory stick. Please read on, as this article attempts to answer some of your obvious questions, like “Interesting, but what is he talking about?”, and “What has Alten TSW to do with all of this?”.

MIDI is generaly known for its .mid files: small files allowing playback of instrumental music on a computer. These SMF files are only a small part of the ever growing list of standards that are published since 1982 by the MIDI Manufacturers Association (MMA)[1] , like:

- MIDI 1.0: The basic MIDI communication protocol
- GM: The General MIDI device specification
- SMF: Standard MIDI Files
- MTC: MIDI Time code
- MSC: MIDI Show control
- MMC: MIDI Machine control

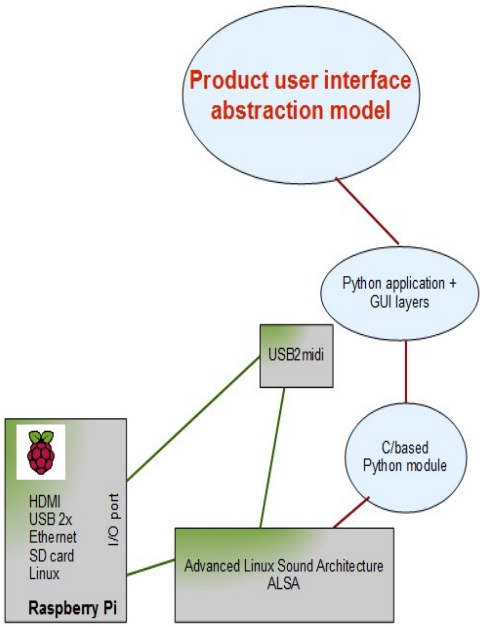
The scope of this article is limited to MIDI 1.0 and specifically system exclusive data dumps. System exclusive dumps transfer data between devices through a standard MIDI connection. In a layered architecture (there's your Alten connection) as depicted below, our project is mainly concerned with the MIDI message layer[4] and the application layer.



Data is packaged in system exclusive messages that are defined in de MIDI message layer. Packaging strategies are dependent on the manufacturer and on the instrument model. Specific software in the application layer is needed to support this.

The concept and goal of MIDI-2-USB appears to becomes feasible with the rise of small, smart development systems like the Raspberry Pi[6]. The Raspberry Pi commonly uses the Linux operating

system. Usually the ALSA[2] library is an installed package on this Linux distribution. ALSA is an abstract interface for soundcards, which also supports MIDI. The Raspberry Pi does not have a MIDI interface, but it does provide two USB ports. An overview of the actual status of the project is given in the image below:



The Raspberry Pi, with Linux and ALSA library installed, is the core of the project. The two lower layers of the MIDI 1.0 model are implemented very simple via a plug-and-play USB-MIDI kabel. Inside this cable's connectors there is a tiny processor with memory buffer, which is responsible for the conversion between USB and MIDI protocols and transmission speeds on hardware level.
The second USB port on the Raspberry Pi will be used to connect a USB-drive or USB memory stick.

Application software is written in Python. Coupling Python with the ALSA library is done via Python's ctypes module. Real-time reception of data via MIDI system exclusive, using the maximum MIDI bandwidth, has been proven.

As User Interface Management Framework[3] has been developed in Python as part of this project. The UIMF allows us to decouple the application from the actual user interface. It also supports parallel use of multiple user interfaces that in turn may use different implementation techniques. Currently a single user interface, implemented in software using tkinter, is being used.

Possible next steps in the project's roadmap are:

- Plugin manager voor Manufacturer / instrument specific applicaton layer modules. These are needed to support data packaging and transmission of the MIDI system exclusive messages;
- Plugin module for the Roland D50[5];
- Hardware user interface board coupled to the Raspberry Pi GPIO;
- MIDI controller functions: hold (sustain) on 4 channels, arpeggio on 4 channels;

And finally, when dreams may come true, the Raspberry Pi will be replaced by a dedicated lightweight hardware platform that may be produced and distributed via Kickstarter.

Resources.

1. MIDI Manufacturers Association (MMA)

http://www.midi.org
2. Advanced Linux Sound Architecture (ALSA) project homepage
   http://www.alsa-project.org/main/index.php/Main_Page
3. A User Interface Management Framework to assist in product development of hardware I/O user interfaces
   https://home.alten.nl/UserFiles/mediawiki/5/5b/User_interface_management_framework.pdf
4. Inside the MIDI message layer
   https://home.alten.nl/UserFiles/mediawiki/5/53/Inside_the_midi_message_layer.pdf
5. Overview of instruments and controllers
   http://www.rven.eu/music/instruments/index.html#overview of instruments and controllers
6. Raspberry Pi
   https://www.raspberrypi.org/
7. Contact the author
   mailto:ad.van.gerven@alten.nl?subject=MIDI-2-USB