Build the "REMI Synth" (mk2)

Monophonic MIDI Sound Synthesizer designed for EWI's

A DIY Project by M.J. Bauer

This post describes the design of the "second generation" (mk2) REMI synth module.
For a general introduction to the project and news updates, see the page: "**Introduction to the REMI**".

REMI 'mk2' Synth Module prototype

Overview

The REMI Synth Module is a monophonic MIDI-controlled sound synthesizer designed primarily for use with electronic wind instument (EWI) MIDI controllers, in particular the **REMI mk2 handset**.

Provision of a standard 'MIDI IN' port allows the synth to be played by any MIDI controller, for example, a keyboard or another EWI with a standard MIDI output. Using a low-cost MIDI-USB adapter, the REMI synth can also be controlled by a computer running music software, for example a MIDI sequencer.

Features

- High quality audio output: 40kHz sample rate, 32-bit precision DSP

- High accuracy oscillator pitch for musical application
- Dual wave-table sound synthesis with mix-ratio modulation (morphing)
- Graphical user interface (2.5" monochrome GLCD, 128x64 pixels) - **optional**
- Command-line interface (CLI) for setup and patching (using PC as terminal)
- Instrument presets (8) selectable from GUI, CLI or MIDI input source
- User-programmable synth patches and wave-table creator (using CLI)
- Noise Generator and Noise Filter (for "pitched noise" effects)
- Effect modulation by breath pressure (CC2) and/or modulation messages (CC1)
- Filter with variable cutoff frequency and resonance, pitch tracking
- Filter frequency control by Expression (CC2), Ampld Env, Mod'n (CC1), LFO, etc.
- Reverberation effect

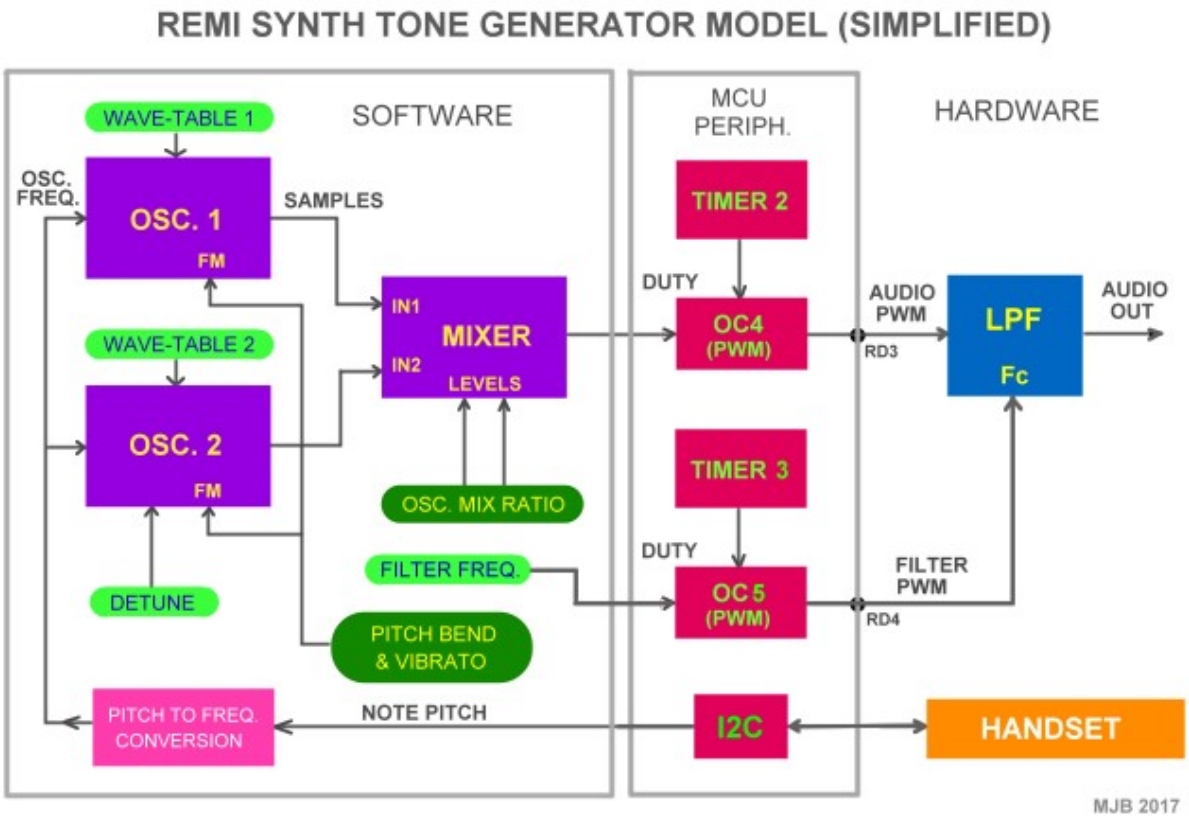
Right-hand side connector panel

Synthesizer Design

The REMI synthesizer is implemented almost entirely in software, requiring minimal circuitry outside of the microcontroller chip - just a low-pass filter in the audio output circuit. To generate audio tones, the synth uses a dual "wave-table oscillator" algorithm offering a variety of waveforms which can range from very simple to rich and complex sounds, some resembling acoustic instruments.

A PIC32 on-chip timer module is used to generate a PWM audio output signal. The PWM "DAC" has a resolution of 11 bits, which gives adequate sound quality. For improved sound quality, an external 12-bit or 16-bit SPI DAC chip may be added. Software DSP computations use 32-bit normalized fixed-point numbers with 20-bit fractional part, allowing the application to run on 32-bit microcontrollers without hardware floating-point capability.

The "synth engine" code has been migrated successfully to a Teensy 3.2 microcontroller, on which the author's new "REMI 3" (all-in-one EWI) is based. (The Teensy 3.2 MCU has an on-chip 12-bit DAC.)



For clarity, the diagram omits the LFO, noise generator, envelope shapers and output attenuator.
(The above model is somewhat obsolete. See sketch of revised patch model here.)

The synth model comprises a pair of wave-table oscillators which use independent wave-tables. The two oscillator outputs are fed into a "mixer" which scales and adds the two signals in a variable ratio. The mix ratio can be fixed, or it can be varied in time as the note progresses. The mixer has its own dedicated envelope shaper to control the oscillator mix ratio. This capability is used to implement "waveform morphing", a technique used to vary the harmonic content of the sound with time. Waveform morphing can be used to realise a range of effects, beyond what is possible to achieve with filtering techniques.

The pitch of the secondary oscillator can be "detuned", i.e. increased or decreased relative to the primary oscillator. The "detune" factor is a patch parameter having units of "cents", so that the detune resolution is 1/100th of a semitone. If the detune factor is a fraction of a semitone, typically in the range 3 to 30 cents, and both oscillators are driven from wave-tables with similar harmonic content, the resulting effect is known as "Voix Celeste" (heavenly voice). This effect greatly enriches the soundscape possibilities of the synthesizer.

In addition to the two wave-table oscillators, a low-frequency oscillator (LFO) is provided. The LFO can be used to modulate the audio oscillator frequency to implement vibrato, or the LFO can be used to modulate the oscillator mix ratio. In the latter case, the mixer envelope output level determines the modulation depth.

An external analog filter of some sort is necessary to remove the 40kHz carrier frequency from the PWM audio output signal. The chosen filter is a 3rd-order "Sallen-Key" low-pass design with a cut-off frequency of 10kHz and roll-off slope of -60dB per decade.

The amplitude (loudness) of the note-in-progress can be varied with time in a variety of ways depending on the instrument patch. A five-segment envelope shaper provides the classic "attack, peak-hold, decay, sustain, release" (AHDSR) amplitude profile. In addition to the envelope shaper, amplitude can be controlled by breath pressure or other MIDI IN Control Change messages.

The firmware includes several "pre-defined" synth patches providing a good variety of instrument sounds. Any pre-defined patch may be assigned to any of the 8 Presets via the user interface (GUI or CLI).

How the Synthesizer is Patched

The REMI synth can be programmed (patched) by the user to create a new sound, without needing to modify and re-compile the firmware. Instead of using knobs and switches like a "real" synthesizer, however, the REMI synth is patched by means of a set of numeric parameters... (see table below). A CLI command "patch" is provided for the purpose of setting patch parameter values. A user-created patch can be saved in non-volatile memory (EEPROM) for later recall. A stored "user patch" may also be assigned to any of the instrument Presets.

Table 1: REMI Synth Patch Parameters

Oscillators	Mixer & Contour Env.	Noise Mix & Filter	Envelope & Ampld Ctrl
OSC1 Wave-Table	Mixer Control Mode	Noise Filter Mode	Amplitude Env Attack
OSC2 Wave-Table	Mixer OSC2 Level (%)	Noise Level Control Mode	Amplitude Env Peak Time
OSC2 Detune (cents)	Contour Env Start Level (%)	Filter Freq. Control Mode	Amplitude Env Decay Time
Pitch Bend Range (cents)	Contour Env Delay Time	Filter Corner Frequency	Amplitude Env Sustain Level
LFO Frequency	Contour Env Ramp Time	Filter Resonance	Amplitude Env Release Time
Vibrato Depth (cents)	Contour Env Hold Level (%)	Filter Pitch Tracking (offset)	Output Ampld Control Mode
Vibrato Delay/Ramp Time			

Two patch parameters specify which wave-tables out of a large selection will be used by the synth oscillators. The assigned wave-tables determine the waveforms and hence the harmonic content of the oscillator outputs. A "user patch" can specify any pre-compiled wave-table (stored in MCU program memory).

The firmware also provides a utility for users to create their own wave-tables. A CLI command "wav" is provided for this purpose. A user-created wave-table can be tested in a user patch.

REMI makers who are prepared to re-compile the firmware can add their own patches and wave-tables, limited only by the amount of MCU flash program memory. CLI commands "patch" and "wav" include options to dump patch parameters and wave-table data (resp.) as C source code definitions.

Details of REMI synth functionality and operation using the console CLI and (optional) front-panel GUI are provided in the Synth User Guide.

Sample sound clips made with the REMI synthesizer

These clips are very old! The latest firmware produces better sounds. More sample sound clips and a better quality demo video will be posted shortly.


Recorder (plain)


Recorder with Celeste


Oboe -- Hard Reed


Jazz Organ with Celeste

Construction

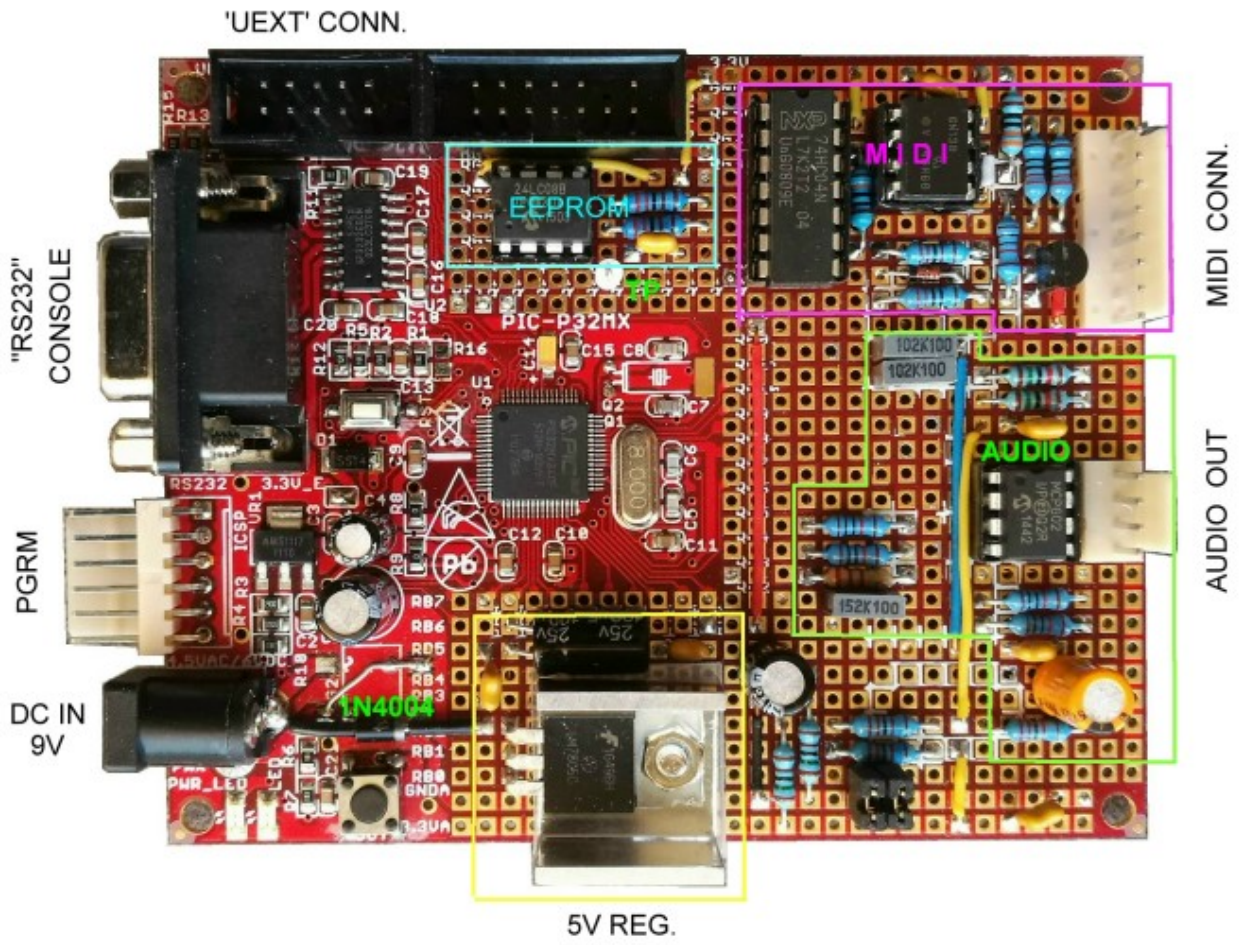
Following is a description of the author's prototype construction. There is enough information given here to allow experienced electronics hobbyists to replicate the synth design. Detailed information such as step-by-step instructions, parts lists, etc, are not given in this post.

Compared with the earler (mk1) prototype, the new PIC32 synth module is greatly simplified. It is based on a PIC32-MX340 proto board from Olimex (pictured below), priced at €19.95 (US$22 approx). The board has parts added for the MIDI IN and (optional) MIDI OUT interface circuits, plus PWM audio output circuitry, 5V regulator for the MIDI interface and LCD panel, plus an IIC EEPROM to store synth configuration and preset parameters. A separate proto board may be added to carry a headphone amplifier and volume control.

View synth schematic diagram
Erratum: The schematic shows EEPROM type 24LC64, but the current firmware supports type 24LC08B only.
The device pinouts are identical. Future firmware revisions may support both types.

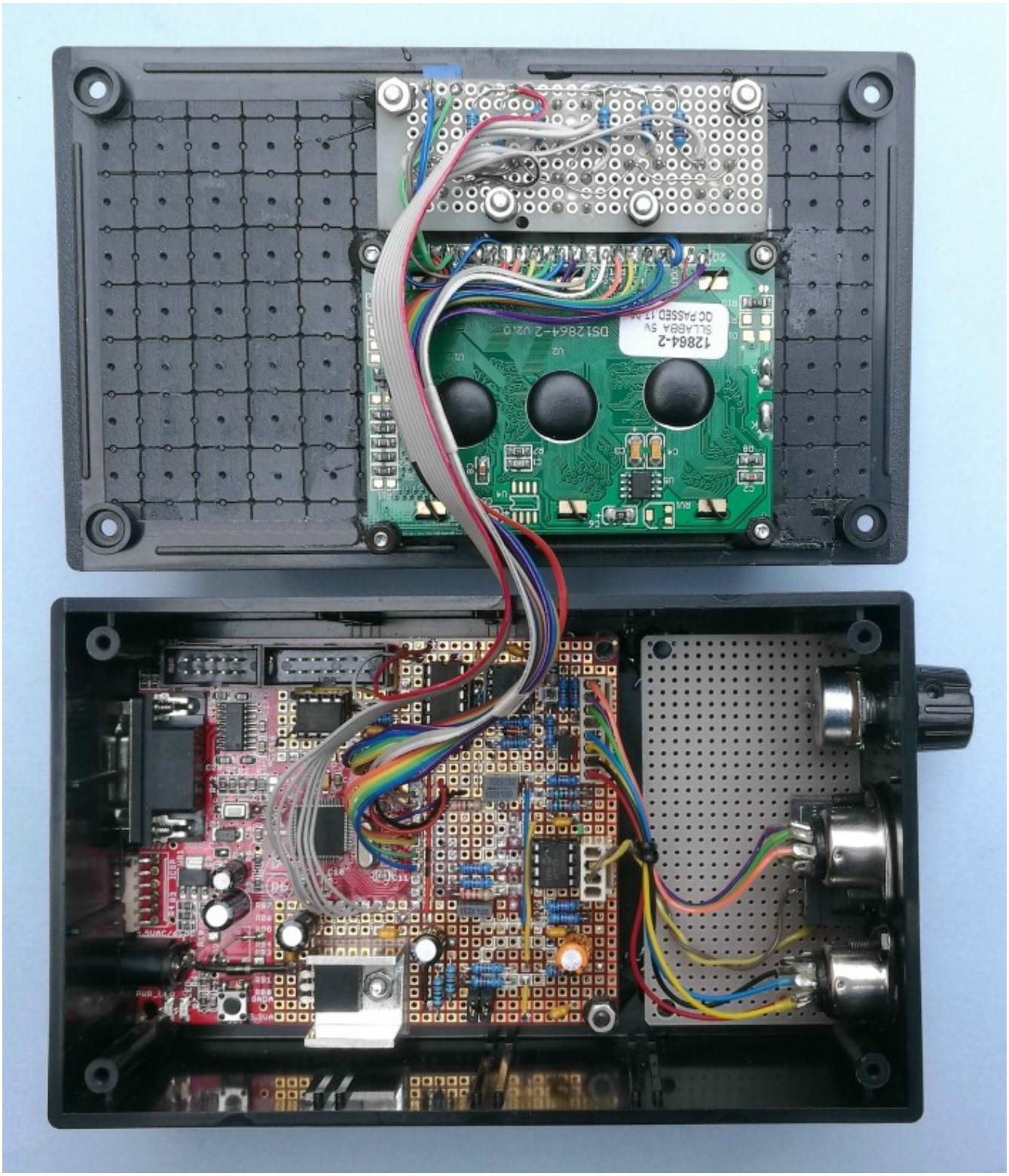
The complete module incorporates a front-panel user interface (GUI) consisting of a low-cost monochrome graphic LCD panel and 6 push-buttons. The LCD module and key-switches are wired directly to I/O pads on the Olimex PIC32 board. This is the quickest and easiest wiring method. (See internal view below.)

NB: Provision of a front-panel (LCD module and push-buttons) is optional. The REMI synth can be operated completely with the CLI alone. The firmware adapts itself automatically if the front-panel is absent.

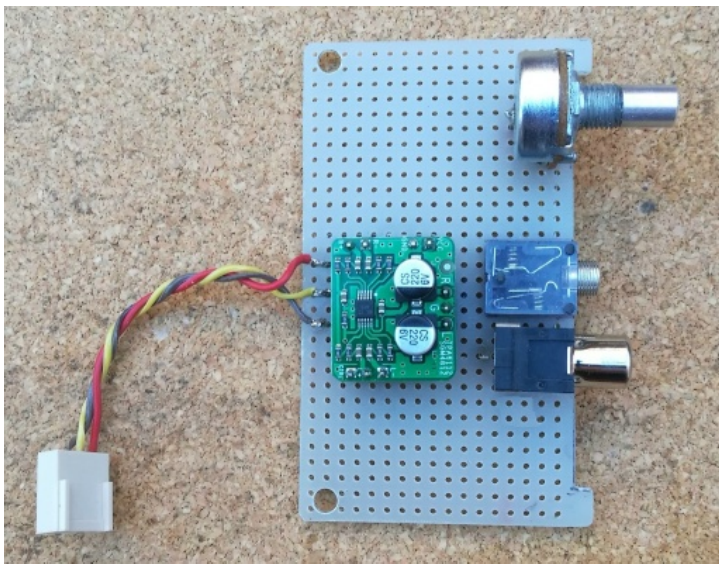


Olimex PIC32MX prototype board with REMI add-ons (excluding optional LCD panel)

A suitable LCD panel is available from Sparkfun. Alternative modules are available at low cost from online suppliers, e.g. Ali-Express. This type of LCD module is also available with smaller overall dimensions and a smaller dot pitch (~0.4mm). There are variants of the connector pinout, chip-select polarity, etc, so be sure to observe the datasheet of the display module you choose.



The internal view above shows the LCD panel and push-button board wired to the PIC32 MCU board. Also shown are the MIDI sockets and volume control. A pre-built headphone amplifier (breakout board with TPA6112 IC) was fitted on the prototyping board later, as shown in the photo below.



Boards are mounted on plastic spacers or standoffs. The PIC32 MCU board and audio output board are mounted on the bottom panel of the box using 20mm x M3 machine screws (countersink heads). Standoffs for the LCD module and button board are glued to the inside of the lid (with 5-minute epoxy) so that screw heads are not visible on the outside.

Left side panel showing modified ICSP header protruding thru a cut-out

---

Firmware

The REMI synth firmware includes a command-line user interface (CLI), accessible via the "RS-232" serial port provided on the PIC32 board. The CLI was originally intended mainly for firmware development, diagnostic and testing purposes, but the CLI now provides equivalent functions for all operations performed by the front-panel GUI -- and much more. You'll need a USB/Serial adapter cable (~ $10) and a PC terminal emulator application (e.g. 'PuTTY').

Provision of an LCD panel and keypad is optional. The synth can be operated using just the console CLI. This is like the "Command Prompt" in Windows, or the "Console Terminal" in Linux. User interaction is all text-based. The REMI CLI has commands to do everything, e.g. set up configuration options, set up instrument Preset parameters, synth patch programming operations, wave-table creation, sound testing, etc.

In contrast, the front-panel GUI can perform only a subset of the CLI functions -- it's main purpose is to change the instrument 'Preset' parameters, for example to select the synthesizer "patch" (and/or MIDI voice for external synth) assigned to each Preset. It is convenient to be able to do this without the need to connect a computer.

New features added in firmware update v2.3.5x:

- Noise Generator and Noise Filter (for "pitched noise" effects)
- Effect modulation by breath pressure (CC2) and/or modulation messages (CC1)
- DSP Filter with variable cutoff frequency and resonance, pitch tracking
- Filter frequency control by Expression (CC2), Ampld Env, Mod'n (CC1) or LFO
- Reverberation effect

Programming Tool

A PIC programming tool, e.g. Microchip PICkit-3, is required to install the synth application firmware. Low-cost PICkit-3 clones are available from online suppliers via AliExpress, eBay, etc.

REMI firmware is built using Microchip PIC development tools - MPLAB.X IDE with XC32 and XC8 compilers - free to download from Microchip's website. If you intend to modify or extend the firmware, you will need these tools. Otherwise, you just need to install the PIC programmer application (IPE - included with MPLAB.X download) on your computer.

User Guide

Details of REMI synth functionality and operation using the CLI are provided in the **Synth User Guide**.

Downloads & Links
Synth PIC32MX proto schematic
PIC32MX MCU pinout diagram
Synth MIDI Implementation Chart
Synth Firmware Development Kit (Source code, MPLAB.X project files, etc)
Synth User Guide
Build the REMI Handset (mk2)

If you are interested in building a REMI synth module and/or EWI handset, or if you have enjoyed following the project here, kindly send me an email. Support is offerred to readers who wish to build a REMI or some other electronic music device. [MJB]

mjbauer@iprimus.com.au

Last update: 29-DEC-2021

MJB Resources for Embedded Firmware Development