A first look at THE

Harmonync

a Dynamic Harmonics Calculator

DRAFT

OF

SPECIFICATIONS

AND

USER GUIDE

Walter Mantovani

Anyone interested in this field of research
and a desire to contribute, collaborate, cooperate
or simply find out more, please contact me:

Walter Mantovani
email: armonici.it@gmail.com

A first look at THE HARMONYNC: A Dynamic Harmonics Calculator – Draft of Specifications and User Guide

Release 0.2.8 – August 2015
(1st release March 2014)

Published by CREATIVE INDUSTRIES PRESS, Turin – Italy

ISBN 978-88-940077-0-1 (PDF)
ISBN-A (DOI) 10.978.88940077/01

TABLE OF CONTENTS

1. AN OPEN ~~PATENT~~ IDEA

1.1. LICENSING

This work was born in freedom.

I worked on this Mechanism gaining informations from the global knowledge that the humankind shares in brotherhood. The contents of this document, the code of the Harmonync and the further development improved by the community are and always will be a free content.

The generic concept/specification of a D*ynamic Harmonics Calculator* is to be intended as a Public Domain release. The specific implementation, here called *Harmonync,* is under **GNU Affero General Public License v.3.0 (AGPL v3)**.

I hope no one is so vile to claim patents or closed licenses to make money round the functioning and the code of this Mechanism. Thank you.

1.2. ABSTRACT

The Mechanism produces a fundamental-variable, user-controlled, Harmonic Series-based tone-scale. The Mechanism allows the user to play the tones of the Harmonic Series of whatever fundamental tone/pitch. The user can play the tones alone (monophonic) or together (polyphonic). It can re-tune MIDI instruments and eventually Controlled Voltage devices.

The first implementation of the fundamental concept standing at the basis of The Mechanism become from the Nature: the human vocal tract is able to finely filter the harmonics that are already present in the timbre of the human voice, like in **Overtone Singing**. Also some musical instruments and playing-techniques can reproduce (play) the Harmonic Series.

So, the base concept is a natural phenomenon: human voice and primitive musical instruments can exploit this phenomenon.

Any implementation of The Mechanism is a simple ratio or multiplication by two numbers or physical quantities. This multiplication/division can be achieved computationally or physically.

The frequency data are stored in a set of low-latency multidimensional arrays (accessible for writing and reading in real-time).

The outputs of The Mechanism can be **ratios** and **numeric values** of frequencies or other units, **tones** (sound frequencies) or eventually **visual signals** (graphs, patterns etc.).

It is possible define this type of mechanism a D*ynamic Harmonics Calculator*, hereinafter called “DHC”.

A DHC is a platform that allow the user/musician to play the tones of an harmonic series and changing the fundamental tone in real-time just like the overtone singers do with the *polyphonic overtone singing technique*. The idea is to play a fundamental-variable harmonic series on electronic and electro-acoustical instruments retuned via MIDI and in real-time.

I started a **project** to implement a DHC.

The first **implementation** is written in **Pure Data visual programming language** using the **Extended Libraries**. In other words, actually the DHC is prototyped as a patch for Pure Data. In particular the first release was developed with *Pd-extended 0.43.4* for Mac OS X but it should run on other platforms.

The project and the DHC implementation take the name of **Harmonync**, as the union of the words “Harmonic” and “Sync”.

Note:
tone == frequency == pitch
harmonic == overtone

1.3. THE CODE

At the time of publication of this document, the first release of Harmonync software is a Pure Data Extended patch available under AGPL v3 license on GitHub at the following URLs:

- Download Project Page: http://industriecreative.github.io/Harmonync/
- Official Repository: https://github.com/IndustrieCreative/Harmonync

1.4. THE GUI

The graphical user interface of the first public release is the following:

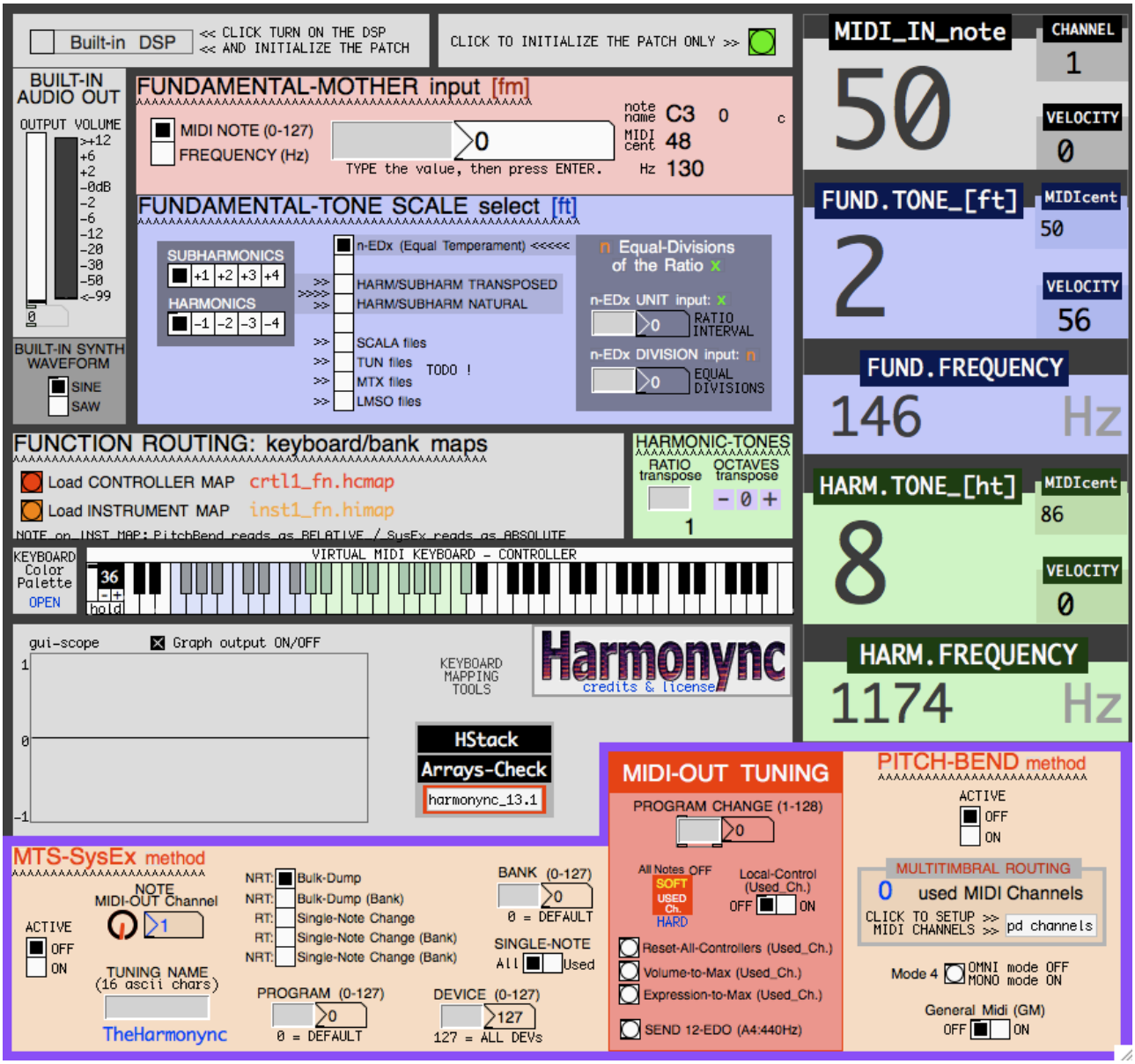


Fig. 1

Fig.1 shows the main window of the Harmonync Pure Data patch.

2. FUNCTIONING

As written in the first chapter, the Harmonync produces a fundamental-variable, user-controlled, Harmonic Series-based tone-scale. It allows the user to play the tones of the Harmonic Series of whatever fundamental tone/pitch. The user can play the tones alone (monophonic) or together (polyphonic).

There are 3 layers of tuning.

- THE FUNDAMENTAL MOTHER (FM)
  The **master frequency** on witch all the calculations are done.
- FUNDAMENTAL TONES (FT)
  It is a set of tones/intervals chosen by the user. The purpose is to offer a palette of intervals to the user. There are many methods to generate or set them (see section 5.2.2 of the Proposed TOC Draft). All the fundamental tones are in a well known ratio with the Fundamental Mother.
  We could define the Fundamental Tones a sort of **interval palette**. Each tone is available to generate a corresponding harmonic series (or subharmonic). The default key-map is set up with a range of 15 keys: from -7 to +7 steps tuned in the standard Equal Temperament (12-EDO or 12-TET) where each step is a semitone. The FT *zero key* ever matches with the Fundamental Mother.
- HARMONIC TONES (HT)
  They are tones taken from the Harmonic Series of the last pressed[1] Fundamental Tone. The default key-map is set up with the first 16 overtones. They are recomputed every time a FT key is pressed.

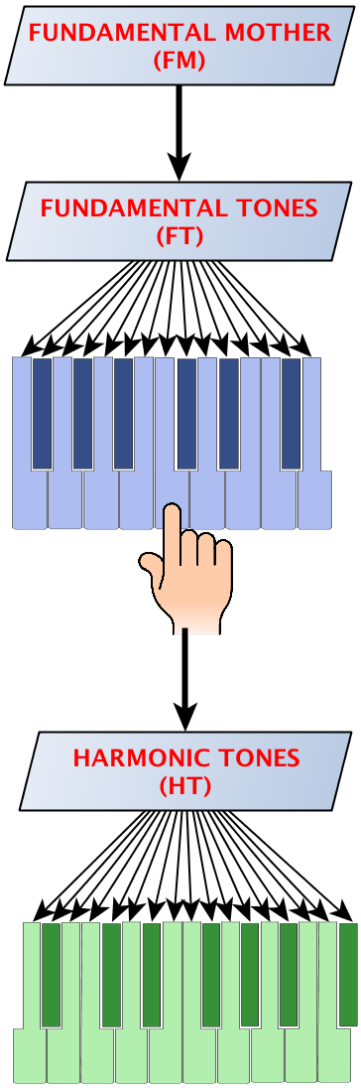
2.1. THE HARMONYNC ALGORITHM

The functioning is pretty simple. Reducing it to its bare bones, we have:

1. the user sets a FM and choose a palette of intervals;
2. the user have now two layer of keys, FT and HT;
3. when the user press a FT key, the frequencies of the HT layer change instantly.

[1] There's a bug: releasing a key when one or more other keys are pressed at the same time, the reference frequency for HTs change to the released key/tome. (TODO: ref. to bug # on bug tracking system)

We can represent the base functioning as follows:



set a FM

choose a palette of intervals

press a FT key

the HT layer change instantly

Fig. 2

For example, if the FM is a C note, more precisely a C3, the frequency to input will be 130.813 Hz (or MIDI note 48) and the *zero key* in the FT layer will be the same. With the default key-map and FT settings (n-EDx, unit 2, divisions 12) you have from -7 to +7 semitones available. It means that you have:

- -7 → F2
- 0 → C3
- +7 → G3

You can notice that the interval -7 > 0 is a perfect fifth and the same is for the interval 0 > +7.

This, because in this configuration you can obtain all the available intervals in the diatonic scale of the classic equal temperament (12-EDO or 12-TET). When you press a key in the FT layer, instantly the Harmonync recomputes the HT layer.

The choice to use just 15 FTs and 16 HTs can be interpreted as a limitation, but this and other reasons need to be detailed in the section 2.2.1 of the Proposed TOC Draft.

We can figure out this example using the following diagram:

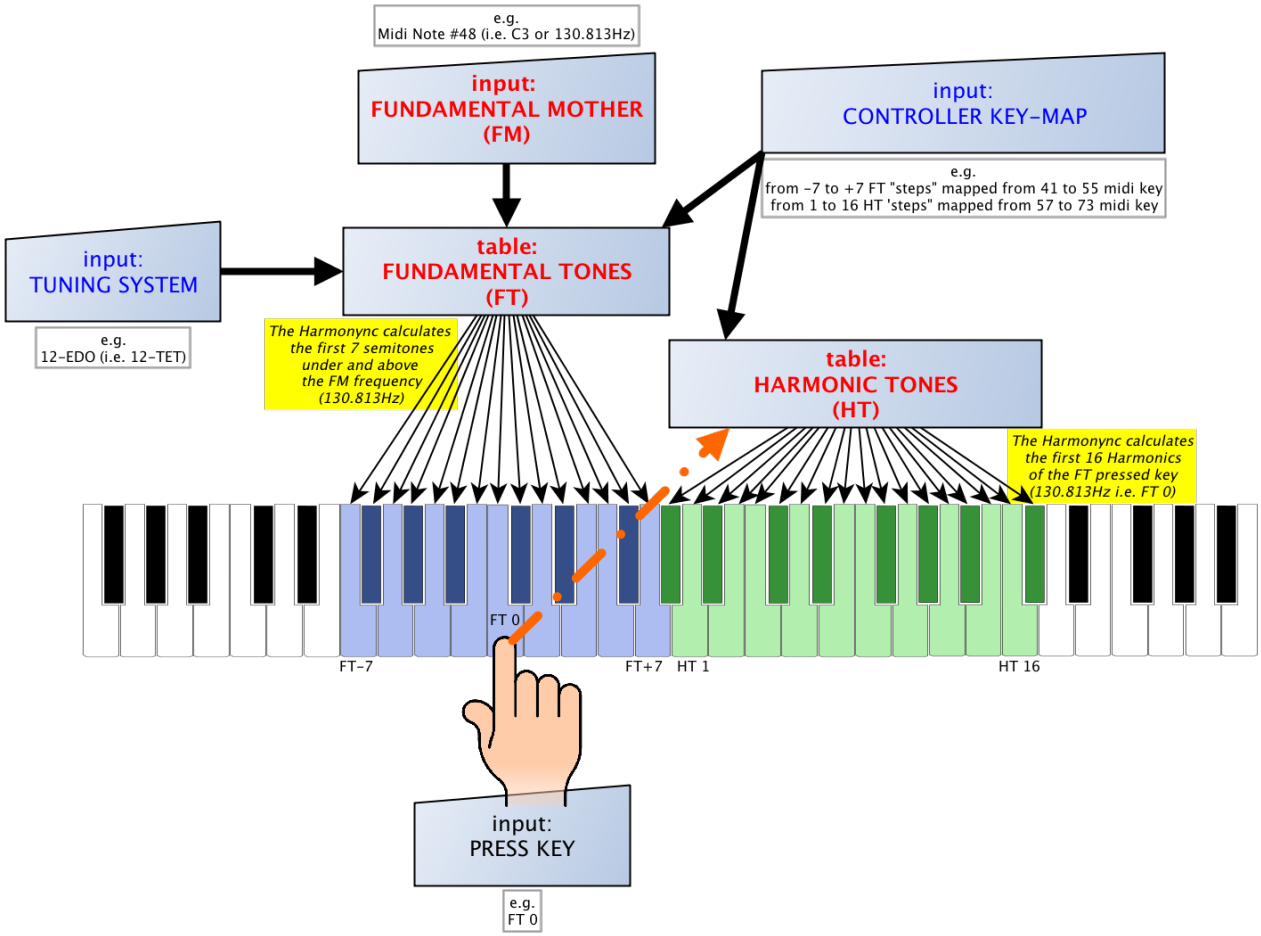


Fig. 3

This is just an example of course. Farther it's using the default settings. Anybody can edit the parameters and the key-maps to fit the DHC to your needs. The guide about User Interface and available settings will be created after this draft and published in the next edition.

2.2. HARDWARE & SOFTWARE REQUIREMENT

A DHC is composed by a:

- central processing unit;
- input and output ports;
- controller device/s;
- instrument device/s.

To be a good DHC, a DHC should be:

- Free Software (free as in freedom);
- Multi-Platform;
- Player-friendly;
- Plug-and-Play;
- MIDI compliant;
- Open Sound Control compliant.

Actually the Harmonync doesn't have all these properties. Some of these goals will be implemented in the future development of this DHC. The goals currently achieved are:

- **[YES] Free Software (free as in freedom);**
- **[YES] Multi-Platform;**
- [NO] Player-friendly;
- [NO] Plug-and-Play;
- **[YES] MIDI compliant;**
- [NO] Open Sound Control compliant.

I officially invite the entire community to participate in the further development to make the Harmonync suitable for production use.

2.2.1. DYNAMIC HARMONICS CALCULATOR

Actually the Harmonync DHC is implemented as a **Pure Data** patch and has been successfully tested on Mac OS X, Linux and Windows platforms. In detail, Harmonync was initially written in Pure Data Extended v0.43.4 for Mac OS X.

The current implementation in Pure Data is intended to quickly draw the logic of the algorithm and to be used as a sort of common reference for the re-implementation of a DHC in other

programming languages that are better suited to the creation of embedded systems with higher efficiency, accuracy and stability.

A DHC can be easily implemented. It is a sort of signal translator between two or more devices. At least there must be: a controller and an instrument. Controller and instrument could be the same physical device but, in this case, its keyboard must be set on MIDI Local OFF Mode.

When you press a key on the controller device the DHC read the signal and the related function. There are two types of functions: **Fundamental Tones** and **Harmonic Tones**. From the point of view of the controller, FT and HT are **two layers**. In default key-maps they are never overlying but you can do this, if you need. Instantly, the HT layer is recomputed with the Harmonic Series based on the frequency associated to the last FT key pressed.

If you press a HT key you'll get the frequency data about the related overtone.

Finally, the frequency data of HT and FT keys can be heard in different ways:

- Built-in Synth/DSP.
- MIDI OUT signals.
    - Pitch Bend Method.
    - MIDI Tuning Standard System Exclusive (MTS SysEx).
- Open Sound Control signals (not implemented yet).
- Control Voltage (not implemented yet).

Harmonync has an **internal synthesizer** and a **built-in DSP**, so you can use them like an instrument device. Since the internal synthesizer is quite limited (you currently have just a sine and a saw waveforms) you can use a physical (external/hardware) or virtual (internal/software) synthesizers and samplers.

Since most of modern electronic musical instruments implements the **MIDI protocol**, MIDI is the best way to drive a instrument. There are two methods to retune a MIDI instrument: by using the **Pitch Bend Change** or the **MTS SysEx** signals. This two methods are mutually exclusive.

Some hardware architectures could use **Control Voltage** signals to send pitch informations. The actual hardware-embedded implementation of the Harmonync does not provide a Control Voltage OUT port, so you need a MIDI-to-CV module.

> NOTE:
>
> Using the **"hard sync" feature** of many synthesizers, you can exploit an electro-acoustical phenomenon to generate harmonics. In short, two oscillators can be hard-synced: one oscillator is zeroed on the other oscillator's frequency. Changing the frequency of the salve oscillator, the resulting waveform generates a an audible pitch that follows the Harmonic Series ratios. So, it is possible to compute the right frequency of two oscillators routed in sync mode, to generate a complex waveform tuned to the desired frequency.
>
> Further developments in this way are planned.

2.2.2. CONTROLLER

A controller is a device that can send signals to the DHC. For example:

- Caps-Lock Keyboard (PC keyboard).
- Virtual MIDI Keyboard (Harmonync GUI).
- MIDI Master Keyboard.
- MIDI Instrument Keyboard (Local OFF Mode).
- Any device in general that can send Note ON/OFF MIDI signals.
- Control Voltage devices.

2.2.3. INSTRUMENT

An instrument is a physical or virtual device that can play the data frequency received from the DHC. For example:

- Synthesizer, sampler or any other playable device.
- Any device in general with a DSP able to generate sounds pitched at a certain frequencies.

A **monophonic** instrument is easiest to set-up but it provides limited performances.

A **polyphonic** instrument is more complete, but is more difficult to set-up.

Some polyphonic instruments are also **multitimbral**. It means that you can set different timbres (or waveforms) for each voice of polyphony. Usually a multitimbral instrument support

the MIDI Channel Mode 4, that is the best configuration to connect the DHC to the instrument using the **MIDI OUT Pitch Bend Method**.

The next sub-section is about the MIDI Channel Modes.

2.2.3.1. MIDI Channel Modes

If you have to play instruments by using the **MIDI OUT Pitch Bend Method** you should know about MIDI Cannels, OMNI, POLY and MONO settings.

According to the MIDI specifications, Polyphonic Instruments could be set in four different ways called “Channel Modes”.

- Mode 1 [OMNI:ON + POLY]
- Mode 2 [OMNI:ON + MONO]
- Mode 3 [OMNI:OFF + POLY]
- Mode 4 [OMNI:OFF + MONO]

Mode 4, also called *Guitar Mode*[2], is the best mode to set the instrument in order to play a DHC. In this mode you can avoid all possible sound artifacts and aberrations.

Mode 3 is also good, but if you use sounds with a long sustain/release you could get into bad sound phenomena. By setting more polyphony voices you can reduce the likelihood of sound artifacts.

2.2.4. AUDIO & MIDI

In order to use all the features provided by the Harmonync the computer should be equipped with a MIDI port (in/out) and a soundcard.

The MIDI protocol is used to receive signals from the Controller and send the output processed signals to the Instrument. Naturally, you need some type of acoustic actuator in order to hear the sound frequencies you generate with the Instrument.

[2] Guitar Mode means that is a mode suitable for guitar synthesizers.

2.3. HARDWARE SETUP SCENARIOS

Since the internal Caps-Lock Keyboard is not implemented yet, the only methods to input a key pressure are:

- a physical MIDI keyboard;
- the Virtual MIDI Keyboard provided with Harmonync;
- a software utility that emulate a MIIDI device and grab the key pressure from the PC keyboard, like the Apple Logic Pro's Caps Lock Keyboard.

2.3.1. BUILT-IN DSP

The simplest configuration is made by connecting a MIDI Controller to a computer running Harmonync, then use the internal DSP and simple Synth and output the audio to the soundcard. The sound is generated by the built-in DSP provided with Pure Data. You have to turn on the DSP and leave the MIDI Tuning Methods turned OFF. The Harmonync patch have 10 voices of polyphony: 2 for Fundamental Tones and 8 for Harmonic Tones. You can also select from two available waveforms, sine and saw.

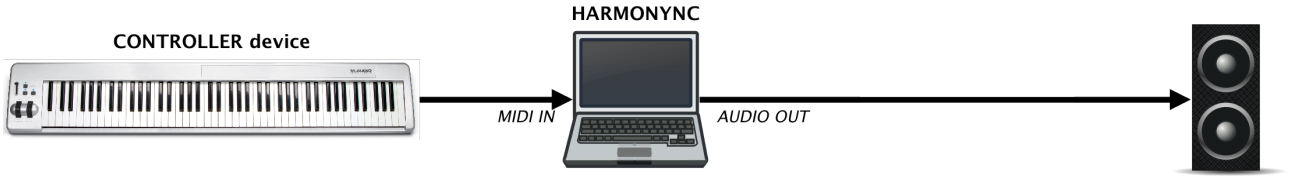


Fig. 4

As in Fig.4, the output sound is generated directly by the DHC.

2.3.2. VIRTUAL INSTRUMENT

In order to retune and play a virtual instrument (hereinafter called "VI"), you must have a MIDI port between the Harmonync and the virtual instrument. If the VI is loaded in the same computer running the Harmonync (that is the most common set-up) you could use a virtual MIDI port. According to your Operating System, you could have none, one or more virtual MIDI ports. If you need one, you can google "virtual midi port" plus the name of your OS.

Whether you use a VI in stand-alone mode or loaded as a plugin in a DAW you have to set the right MIDI channel routing. Read the documentation provided with your DAW, OS and/or your MIDI virtual port software in order to correctly route the virtual devices.

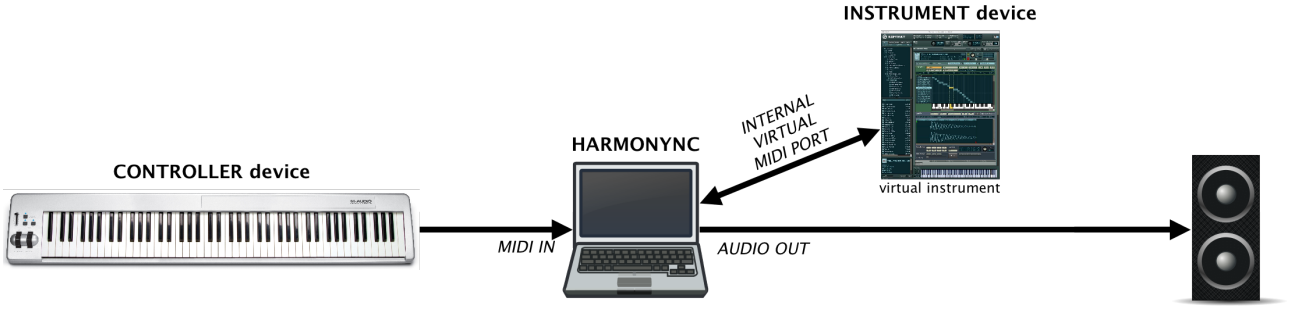


Fig. 5

2.3.3. PHYSICAL INSTRUMENT

Obviously, you can play any external physical instrument that implements the MIDI protocol.

The most natural configuration is by using a separated Controller and Instrument. For example a Master Keyboard and a Synthesizer as in Fig.6.

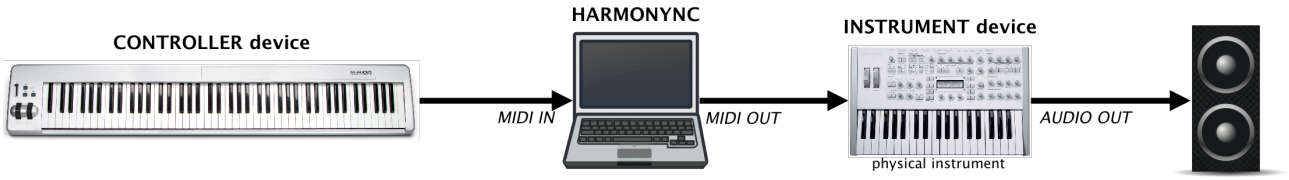


Fig. 6

If your Instrument is able to set **MIDI Local Mode OFF** you can use its keyboard to play the Instrument itself. The Fig.7 figures out a virtual piano having MIDI Local Mode ON/OFF option.

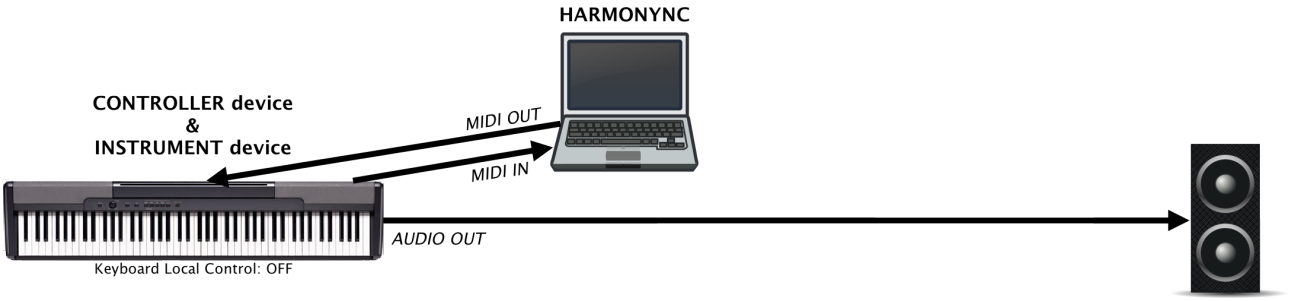


Fig. 7

A good DHC should be able to retune instruments by Control Voltage. This option could be implemented directly with an hardware module controlled by the DHC software or via a MIDI-to-CV converter device.

3. USAGE

3.1. BEST SCENARIOS

Usually musicians need to play physical or virtual instruments, and record them with a DAW. This configuration is also suitable for scientific purposes.

Actually there are two best ways to retune a MIDI compliant instrument:

- using the ratified MIDI Tuning Standard protocol (MTS);
- using the standard 12-TET (12-EDO) notes with the right amount of Pitch Bend Change.

The two methods are mutually exclusive.

Since practically no hardware manufacturer implemented the MTS protocol, it is almost useless.

The Pitch Bend Method is potentially compatible with any MIDI instrument capable to receive Pitch Bend Change signals.

4. PROPOSED TOC DRAFT

This is the draft of the proposed “Table of Contents” of the upcoming Harmonync guide.

5. HARMONYNC SOFTWARE GUI

5.1. INITIALIZE THE PATCH

5.1.1. TURN ON THE DSP...

5.2. THE TUNING SYSTEM

5.2.1. THE FUNDAMENTAL MOTHER

5.2.1.1. SELECT UNIT OF MEASURE

5.2.1.2. INPUT TEXT BOX

5.2.1.3. INFO LABELS

5.2.2. FUNCTIONAL TONES

5.2.2.1. FUNDAMENTAL TONES

5.2.2.1.1. TUNING COMPUTATION METHOD

5.2.2.1.1.1. nED-x (aka Equal Tempetaments)

####### 5.2.2.1.1.1.1. UNIT (ratio interval to divide)

####### 5.2.2.1.1.1.2. DIVISION (equal divisions)

5.2.2.1.1.2. HARMONICS/SUBHARMONICS

####### 5.2.2.1.1.2.1. TRANSPOSED

####### 5.2.2.1.1.2.2. NATURAL

####### 5.2.2.1.1.2.3. Adjust the Transposition

5.2.2.1.1.3. todo: TUNING FILES

####### 5.2.2.1.1.3.1. SCALA

####### 5.2.2.1.1.3.2. TUN

####### 5.2.2.1.1.3.3. MTX

####### 5.2.2.1.1.3.4. LMSO

5.2.2.2. HARMONIC TONES

5.2.2.2.1. TRANSPOSITION

5.2.2.2.1.1. BY INPUT RATIO

5.2.2.2.1.2. BY INCREMENT/DECREMENT OCTAVES

5.3. THE DEVICE MAPPING SYSTEM (Function Routing)

5.3.1. DEVICES

5.3.1.1. CONTROLLER

5.3.1.1.1. Map syntax

5.3.1.2. INSTRUMENT

5.3.1.2.1. Map syntax

5.3.2. FUNCTIONS

5.3.2.1. FUNDAMENTAL TONE (FT)

5.3.2.1.1. Available range

5. ACKNOWLEDGEMENTS

Thanks to the Internet Global Community.

And thanks to:

- Jeff X. Scott (LMSO - Li'l Miss' Scale Oven)
- Aaron Andrew Hunt (H-Pi Instruments)
- Manuel Op de Coul (SCALA)
- Miller S. Puckette (Pure Data)
- Carlo Serafini (thesis: Tecnologia e Sistemi di Accordatura)
- Victor Cerullo (Max Magic Microtuner)
- Benjamin Frederick Denckla (thesis: Dynamic Intonation for Synthesizer Performance)
- Jacky Ligon (Xen-Arts)
- Robert Walker (Tune Smithy)
- Mark Henning (TUN file format)
- Joe Monzo and Chris Wittmann (Tonescape / Tonalsoft Encyclopedia of Microtonal Music-theory)
- Bill Gannon and Shelly Kantrow (Justonic)