https://doi.org/10.7494/978-83-67427-06-7_4

Software emulation of MIDI arrangers

Kamil Zimny

AGH University of Science and Technology in Krakow

kzimny@agh.edu.pl

Abstract

MIDI arrangers are the type of keyboard instruments, used for song arrangement and playing live. It's done by using the so-called automatic accompaniment, controlled by the user in real time. With auto-accompaniment play mode, some song-creation operations are moved to the MIDI arrangers' algorithms. Therefore the user can focus on manually played part, while not giving up the following of other instruments. So far, the idea of MIDI arrangers has been used only for hardware devices and its software emulation. Therefore, the use of MIDI arranger features in any MIDI-device systems became the subject of this paper. For this purpose, an application which emulates the operation of MIDI arranger's algorithms has been designed.

1. Introduction

The advent of electronic musical instruments has completely changed the musicians' workflow and the music itself. Evolving technology offers new ways to generate sound or playing, providing unprecedented possibilities for conveying emotions. What's more, the miniaturization and growing computer power of the devices allows for implementation of more advanced features. In electronic instruments, this means e.g. better sound synthesis engines, which enable obtaining various sounds, including the faithful emulation of acoustic instruments.

Over the years, most of the written music contains an certain repetition. It concerns both short instrumental parts or longer fragments of the song (like verse, chorus). As technology developed, it became possible to create music tracks in real time, using prepared or overdubbed parts played in a loop. Hardware manufacturers, however, went a step further, developing algorithms to control the course of the harmonic content of a song in a similar way.

1.1. MIDI arrangers

Among many electronic musical instrument, there is a separate group of them called MIDI arrangers. These are the keyboard instruments, which works as multi-timbral synthesizers, so can generate a sound for more than one instrumental part in the same time. In addition to manual play, MIDI arrangers allow to use the so-called automatic accompaniment (auto-accompaniment). This is nothing more than following the user's play by the other instrumental parts, which enables the creation and performance of extensive musical arrangements. What's important, the course of the entire song is controlled by the user in real-time.

When playing with auto-accompaniment, the device's keyboard is split into two zones (Fig. 1).

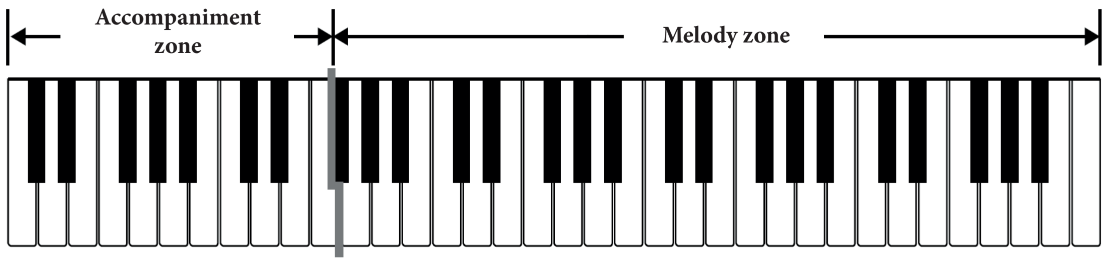


Fig. 1. MIDI arranger's keyboard zones in the auto-accompaniment play mode

When pressing the keys in *Accompaniment zone*, the MIDI arranger recognizes the chord and transforms the auto-accompaniment to match the harmony of new chord. The *Melody zone* is used for manual play, like in other keyboard instruments (piano, synthesizer).

MIDI arrangers fit perfectly into the *One-Man-Band* playing concept. Thanks to the advanced control of the song course in real-time and well emulated sound, these instruments could successfully replace the conventional music bands. Interestingly, the ideology of work and use MIDI arrangers has so far been used only in hardware devices, and virtual emulations of them.

1.2. MIDI

MIDI (*Musical Instrument Digital Interface*) is an digital standard covering the data transmission protocol, connectors and file format [1]. This protocol is used for communication between electronic musical instruments and other compatible devices. The appearance of the MIDI was an response to the needs of musicians, who want to use their instruments as an integrated system. MIDI gave them this feature and much more, starting a new era in working with electronic musical instruments.

That is why MIDI has been closely related to the electronic musical instruments. In the development of this technology, another big milestone was the appearance of MIDI sequencers – hardware/software devices that allow for recording (capturing), editing and playback the MIDI sequences. It should be remembered that before the MIDI development, the philosophy of working with recorded audio material was only available to professional recording studios, properly equipped. Thanks to MIDI sequencers, this workflow was moved to the musicians' homes, because in many cases working with MIDI is similar to the audio data. It was actually the beginning of so-called home recording.

In the MIDI standard, the data is sent in form of so-called MIDI messages [2]. Some of them are sent on one of the **16 MIDI channels** (*Channel Messages*), i.e. independent MIDI data sub-streams; the others work as global messages. Among the channel messages, there is one group, which is mainly responsible for the harmonic content of MIDI sequences. There are called **Note-On** and **Note-Off** messages, and are usually used to turn on/off sounds of a given pitch and (often) volume. Note-On message consist of three bytes of data (Fig. 2)[1].

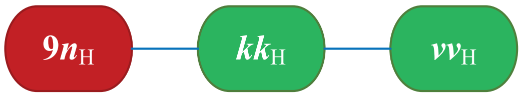


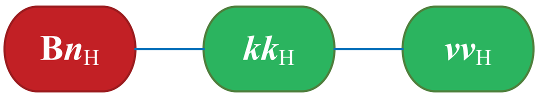
Fig. 2. Structure of Note-On MIDI message (where: *n* – number of MIDI channel, *kk* – number of MIDI note, *vv* – velocity value)

In MIDI standard each note is assigned a numerical value, within the range 0_D–127_D. The velocity parameter relates to the speed at which a key is pressed, and is usually assigned to control the volume of the sound. In this case the higher velocity value, the louder the sound. The Note-Off messages are sent

[1] Two hexadecimal digits (two characters) represent the 8-bit byte content, one character is half of the content (4 bits).

when note (key) is released. In most applications, Note-Off messages are replaced by the Note-On, with number of released note and velocity value equal to zero.

There is also another group of Channel Messages, called **Control Change** (CC), made up of three bytes of data (Fig. 3).



$\mathbf{B}\boldsymbol{n}_{\mathrm{H}}$
$\boldsymbol{kk}_{\mathrm{H}}$
$\boldsymbol{vv}_{\mathrm{H}}$

Fig. 3. Structure of Control Change MIDI message
(where: *n* – number of MIDI channel, *kk* – controller number, *vv* – value)

CC messages are used for supporting various physical controllers (like faders, knobs, pedals, sliders). There are also an subgroup – called Channel Mode Messages – used for special functions, i.e. turning off all notes, all sounds, resetting controller values etc.

2. Project assumptions

The aim of this paper was to design a system which allows for using the features of MIDI arrangers in any MIDI-device system. For this reason, research and tests were carried out on a hardware MIDI arranger KETRON SD9 PRO LIVE STATION [3, 4]. Based on it, a general MIDI arranger scheme has been designed (Fig. 4).

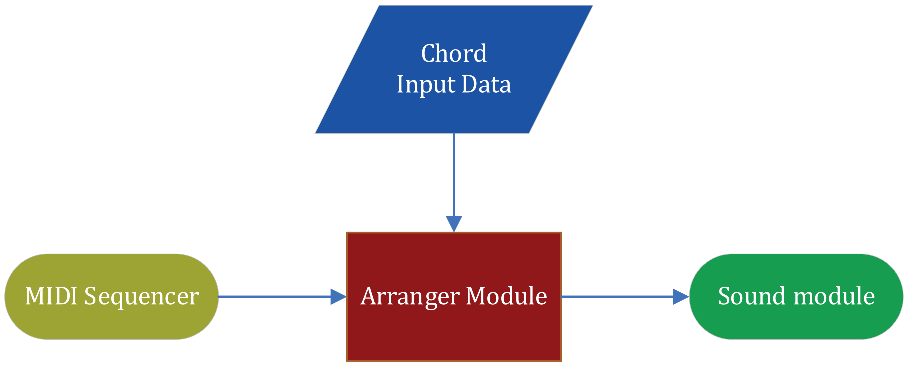


Fig. 4. General scheme of MIDI arranger's data flow

Each MIDI arranger is equipped with a large number of auto-accompaniments, in a variety of musical styles[2]. After selecting the desired one, an appropriate MIDI sequence is loaded into the internal MIDI sequencer. It consist of several parts corresponding to different parts of the song (verse, chorus etc.). These are chosen during playing by the user using dedicated buttons on device's panel. What's more, the entire harmonic content of the MIDI sequence is based on one so-called base chord – usually C major.

The Arranger module – which receives MIDI data from the sequencer – transposes the notes of the auto-accompaniment, according to the so-called current chord and selected operating modes. The current chord is defined by the user during playing via pressing the keys in the accompaniment zone (so it is the generating of *Chord Input Data*). Finally the processed data is sent to the internal sound module, which executes the MIDI commands, e.g. play sounds.

When creating the software emulation of MIDI arranger, each module has to be replaced by separate device. There are lots of MIDI sequencers – hardware of software – which could send the auto-accompaniment MIDI data. Nowadays, this role is often done by the computer applications called *Digital Audio Workstation* (DAW) – professional software for working with audio and MIDI. The same is

[2] For this reason, the auto-accompaniments are often called *styles*.

with generating *Chord Input Data*, which could be done with any type of MIDI controller. There are also many sound modules, which could be connected in many ways for executing the MIDI commands. The only missing and key element is the emulation of Arranger module, which has been realised as author's software application named *MIDI Arranger Emulator* (MAE). This app receives MIDI data (e.g. from MIDI sequencer), process it according to recognized chord and selected modes and sent to the outputs (e.g. sound modules). Due to the above, an example system which emulates the working of MIDI arrangers has been shown on Figure 5.

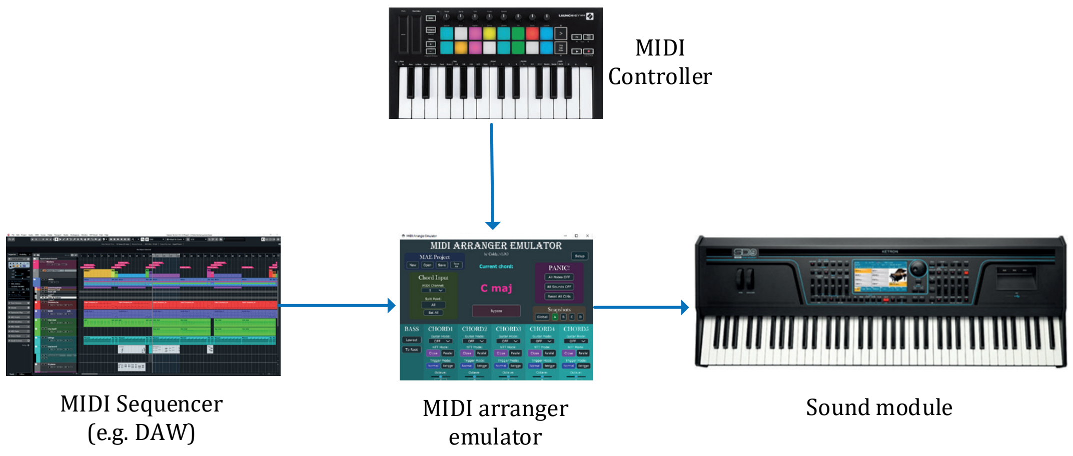


Fig. 5. An example MIDI arranger emulation system with MAE app[3]

3. Implementation

The MIDI arranger emulator app emulates the operation of the arranger module. Thus a detailed scheme of it was designed (Fig. 6).

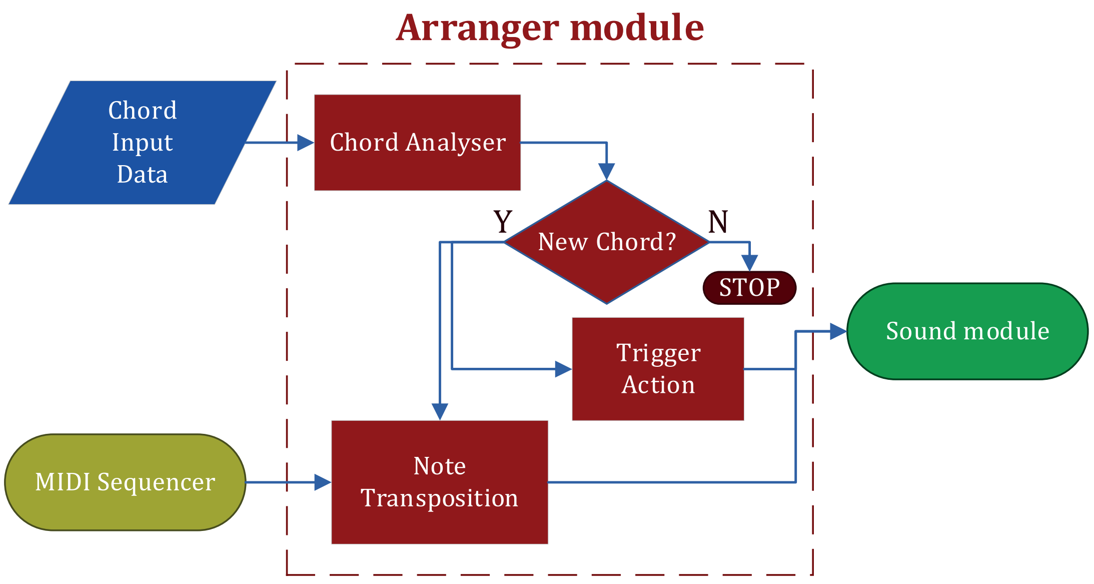


Fig. 6. Detailed arranger module scheme

The details of implementations of arranger module's blocks are discussed in the following sections.

[3] Sources: https://www.ketron.it/en/prodotto/sd9-2#&gid=1&pid=1, access 13.11.2021; https://novationmusic.com/sites/novation/files/LKM-overhead-1018-530.png, access 13.11.2021.

3.1. Chord Analyser

As mentioned before, each MIDI note has its own number. Thus chord recognition algorithm is based on calculation of the distances (intervals) between keys pressed by the user (Fig. 7).

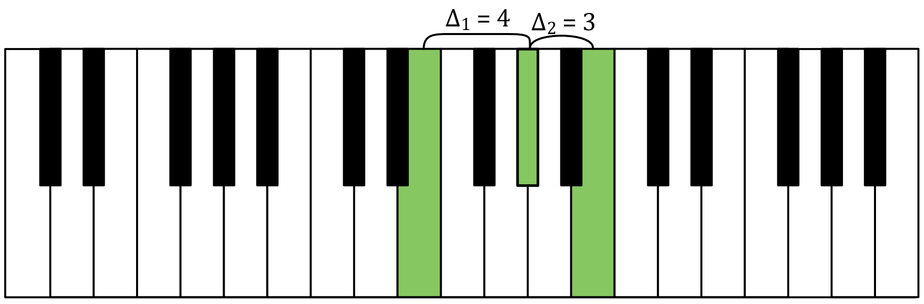


Fig. 7. Counting the intervals between pressed keys

After calculating the differences, the algorithm searches the defined database for the pattern. If found, the chord is recognized. The latest version of MAE supports 13 chord types with up to 10-key combinations, and also the so-called *Pianist mode*. When enabled, the entire keyboard of the MIDI arranger act as accompaniment and melody zone simultaneously. This allows for e.g. playing like a piano with the following of the auto-accompaniment. Additionally, it is possible to turn on freezing the current chord by using the sustain pedal (CC# 64). On the other hand, for easier play the simplified chord patterns (1- or 2-keys) are also supported. Finally, the lowest chord note can be additionally recognized (*Lowest* mode), which is used by the Note Transposition block for alternate bass part processing.

The decision block *New Chord?* checks whether the newly recognized chord differs from the previous one. If so, appropriate actions are performed, otherwise the algorithm terminates.

3.2. Note Transposition

The MIDI arranger's auto-accompaniment consists of several sound parts, usually:

– drum parts (usually 1–2),
– bass part,
– instrumental parts (usually 5).

Each part is assigned to its own MIDI channel, played back by the MIDI sequencer and sent to the Note Transposition block. As mentioned above, all harmonic content of the auto-accompaniment parts is based on one chord, usually C-major. Obviously, this does not apply to the drum parts, which by definition don't contain the harmonic content. For this reason, there are not processed in the Note Transposition block, but simply sent to the MAE outputs.

The Note Transposition block is responsible for the note transposition, so moving them to new pitches, according to the current chord. The current chord data is provided in real time from the Chord Analyser block – when the new chord appears, the processing is immediately performed for the new current chord. In the MAE application, the Note Transposition block has two operation modes:

– **Close (fixed) mode** – the transposition is performed so that the distances (intervals) between original and processed notes are as small as possible. This is done taking the rules of conventional harmony into account. An example transposition from C major to F major chord in close mode has been shown in Figure 8.

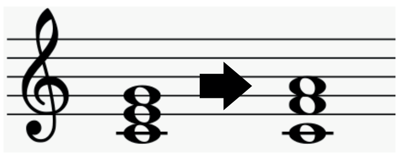


Fig. 8. Close mode transposition

– **Parallel mode** – the notes are transposed without changing the intervals between them. To avoid too high transposition values, the so-called inversion chord root point is defined; if root of current chord exceeds it, transposed notes are shifted down by the octave (12 semitones). An example transposition from C major to F major chord in parallel mode has been shown in Figure 9.

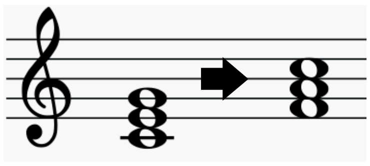


Fig. 9. Parallel mode transposition

Due to the finite number of chromatic sound pitches, supported chords and transposition modes, in practical implementation of Note Transposition block the matrix of all possible transposition values is used. This matrix is called Note Transposition Table (NTT), and for MAE application, the NTT was created based on emulated MIDI arranger – KETRON SD9.

The data was extracted from the device using prepared input data, which had been saved as auto-accompaniment file, and played back by the hardware MIDI arranger. Then the output MIDI data was captured using the external MIDI sequencer – DAW application, and compared to the prepared input data (Fig. 10).

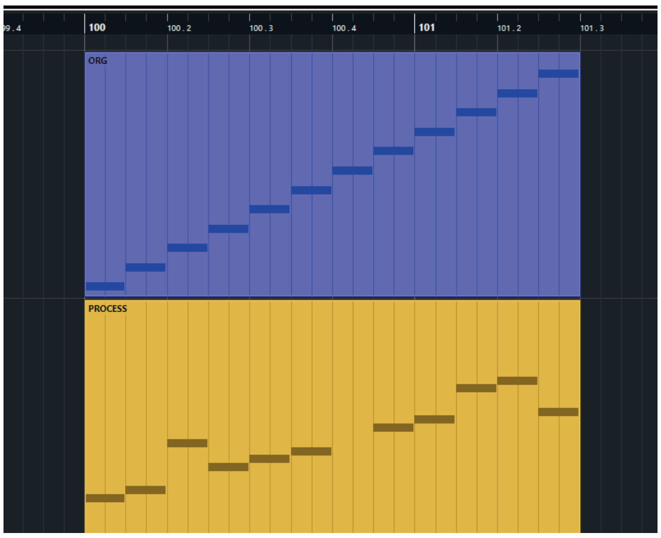


Fig. 10. Extracting the NTT data (violet – input data, yellow – captured output data)

After that, the intervals (distances) between the corresponding MIDI notes was calculated, and put into the Note Transposition Table (Tab. 1).

Table 1

Example part of the NTT for close mode, E major chord

Close mode, E major												
Note:	C	C#	D	Eb	E	F	F#	G	Ab	A	Bb	B
Input[4]:	48	49	50	51	52	53	54	55	56	57	58	59
Output[4]:	47	48	54	51	52	53	n/d[5]	56	57	61	62	58
Δ	**–1**	**–1**	**4**	**0**	**0**	**0**	**n/d[5]**	**1**	**1**	**4**	**4**	–1

[4] Input – input MIDI note number; output – output MIDI note number.

[5] For some notes the output value is not defined (n/d) which causes no sound; thus the Δ value is also undefined.

Using NTT, the algorithm finds the appropriate note transposition value for each coming note, and then transpose it by adding the transposition value to the note number, or disable it (Δ = n/d). The processing of bass part is performed using a separate NTT, which works similar to the Parallel processing. Instead of Close/Parallel operating modes, the bass part section provides two others:
– **Lowest mode** – instead of chord root note, lowest chord note is played.
– **To Root mode** – play all notes as chord root note; when the **Lowest** mode is also enabled, play all notes as lowest chord note.

3.3. Trigger Action

In general, the current chord may change anytime, also when some notes are sounding, so when they're turned on by the Note-On message, and not turned off by the Note-Off. They are called active notes. In order for the auto-accompaniment to correctly follow the chords, an action for active notes has to be taken. That is the task for Trigger Action block, which operates in two modes:
– **Normal mode** – after changing the current chord, all active notes are turned off (Fig. 11).

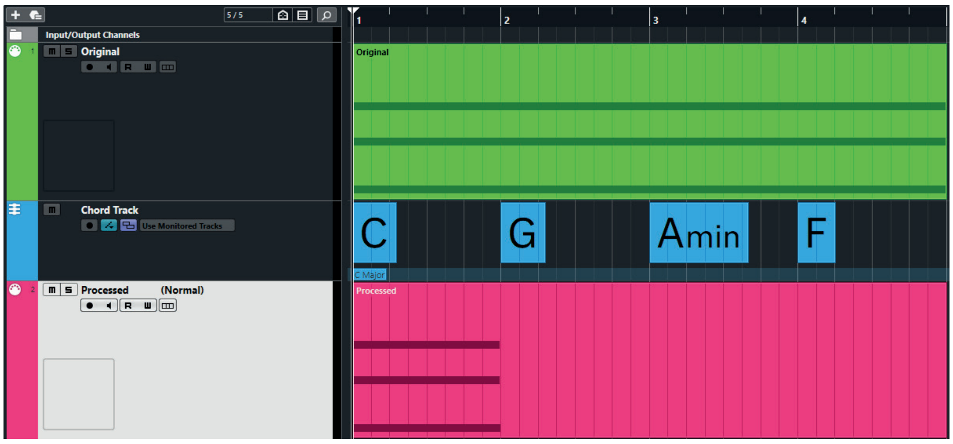


Fig. 11. Operation of Trigger Action block in Normal mode

– **Retrigger mode** – after changing the current chord, all active notes are turned off, transposed to a new current chord, and then turned on (Fig. 12).

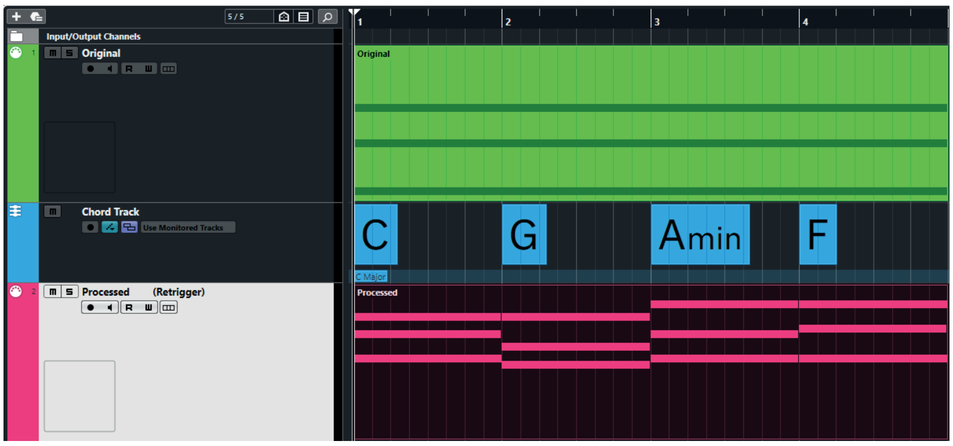


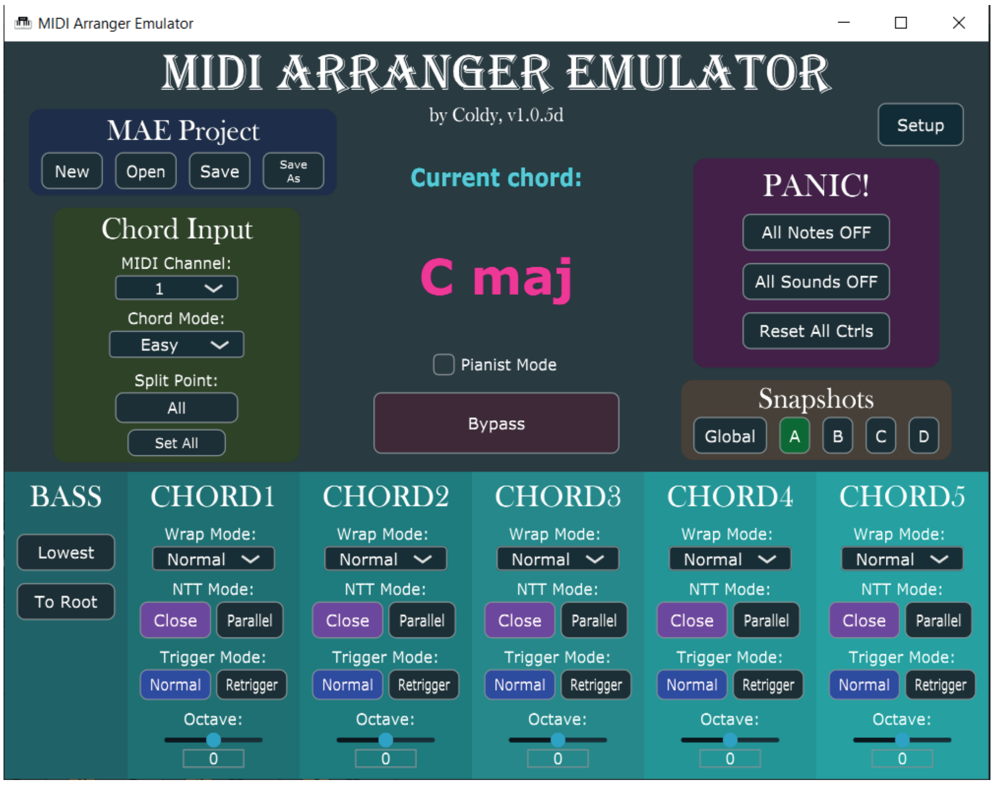
Fig. 12. Operation of Trigger Action block in Retrigger mode

Each mode is used for a different purpose. When the part consists of many Note-On and Note-Off messages, it is better to use the Normal mode to avoid unnecessary repetition of the MIDI notes. On the other hand, for static parts made of long MIDI notes (like strings, pads, organs) the Retrigger mode is more suitable, for resuming the notes after changing the current chord.

Some MIDI arrangers also offer a third mode, often called **Repitch** mode. It works similar to the Retrigger mode, except that the notes are returned to the new pitch instead of being turned off and on. Thus the sounding of the notes is not interrupted. The Repitch mode has not been implemented in the MAE app as it is not supported by the emulated MIDI arranger (KETRON SD9).

4. MAE distribution

The key element of MIDI arranger's emulation – MIDI Arranger Emulator app – has been designed with JUCE framework, C++ language (Fig. 13).



MIDI Channel: 1
Chord Mode: Easy
Split Point: All
Current chord: C maj

	CHORD1	CHORD2	CHORD3	CHORD4	CHORD5
Wrap Mode:	Normal	Normal	Normal	Normal	Normal
Octave:	0	0	0	0	0

Fig. 13. MIDI Arranger Emulator (MAE) app window

MAE is the standalone app, which could be run on computers with Windows operating system. During start up, it searches for available MIDI Input/Output devices for receiving, processing and sending out the MIDI data. All project assumptions have been successfully implemented – the application works stable and has low resource consumption.

In addition to the main processing blocks, MAE is equipped with several useful features. The MIDI I/O configuration is saved and restored during each MAE start up. The user can also store the operating modes for processing the instrumental parts (*CHORD 1–5*) in four banks of memory (*Snapshots*). The whole MAE configuration is called MAE project, which could be saved to file with *.mae extension for further loading. Some features of the app could be remote controlled, using the appropriate MIDI commands. The application is being prepared for commercial distribution. Thus the app installer was created, as well as security mechanisms for avoiding the non-authorized usage. The user manual for the MAE app has also been prepared.

4.1. MAE-based systems

As mentioned at the beginning of the paper, MIDI arrangers exist as single devices that contain all the necessary elements (MIDI sequencer, arranger module, in hardware devices also a MIDI controller and a sound module). However, with MAE application there are no obstacles to use the idea of MIDI arrangers freely, in any MIDI system. Thanks to its modularity, MAE could be used in many ways with lots of MIDI devices. This gives new features to the owned system and new ways of artistic expression, using relatively simple idea of MIDI arrangers.

The MAE-based MIDI emulation systems are also very helpful when creating auto-accompaniments for existing MIDI arrangers. For this purpose, an simply system consisting of the hardware arranger (which acts as MIDI controller and sound module), DAW and MAE apps is shown on Figure 14.

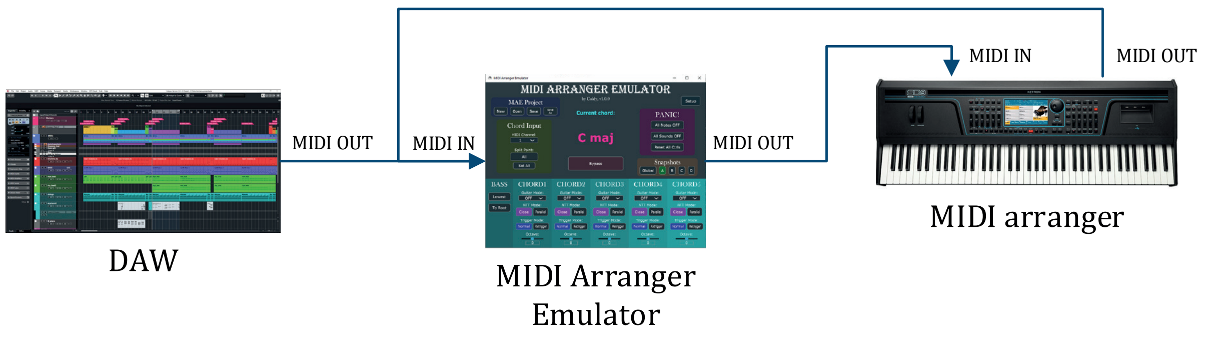


Fig. 14. MAE-based system for creating the auto-accompaniments

In the above system, the creation of the auto-accompaniment MIDI sequence is performed in the DAW application, which acts as MIDI sequencer. When playing back MIDI sequences, the data is sent to the MAE application, which processes it in the same way as an emulated hardware MIDI arranger. The output is sent to the hardware device, which executes the MIDI commands. Moreover, the arranger's keyboard is used to generate the Chord Input Data, which is also sent to the MAE MIDI inputs. When working on the auto-accompaniment without MAE, it is necessary to export the MIDI sequence as file and load it to target device in order to check its operation. Often this process has to be repeated many times, and on the other hand, it can be eliminated using the MAE-based system. Thanks to it, composing the auto-accompaniment could significantly speeded up – even 3–4 times, according to the author's observations.

In the future, further expansion of MAE is planned. The number of supported chords will gradually increase, and in addiction to the chords, the Note Transposition block will process the MIDI data also for intervals like thirds or fifths. This requires the new operating modes for this block, as well as the expansion of Chord Analyser, for delivering the appropriate data. Another major expansion will be the integration with some DAW apps, that are currently most used in auto-accompaniments creation. Thanks to it, some useful (as for MIDI arrangers) features could be controlled directly from MAE app, to make it simple for the users. As for security, the USB-key authorizing system will be designed, as well as term licencing system. Finally, general improvements will be made to improve the stability and reliability of the MAE.

5. Conclusion

This paper presents an software emulation of MIDI arranger's, created with author's application MIDI Arranger Emulator. Thanks to it, the MIDI arranger's musical workflow can be used outside of typical devices, in any MIDI device system. The modularity of MAE allows for going far beyond the typical use of MIDI arrangers, which gives new ways of music creation and performance. On the other hand, the MAE-based systems could help in creating the auto-accompaniments for existing hardware/software devices.

All project assumptions of designing the software emulation of MIDI arrangers were successfully implemented. As for the MAE app, further development is planned in terms of extending its functionality and preparation for commercial distribution.

References

[1] The MIDI Manufactures Association, *The Complete MIDI 1.0 Detailed Specification*, Los Angeles, 2014
[2] P.D. Lehrman, *What is MIDI?*, Tufts University, 2017
[3] K. Zimny, *Inżynieria wsteczna algorytmów instrumentu KETRON SD9 PRO LIVE STATION*, AGH, Kraków, 2020
[4] AJAM Inc. USA, *KETRON SD9/SD60/SD90 PRO MANUAL, HINTS & TIPS (V 1.5)*, USA, 2018