Avon - KHSB8

Group 1 Proposal: Step Sequencer

Hönnun Tölva

Bragi Marinósson

Hallgrímur H. Einarsson

Karl James Pestka

Sindri Bergsson

Teacher: Helgi Þorbergsson

9. september 2016

Háskóli Íslands

Contents

List of Figures

List of Tables

1 Executive summary

We propose to build a MIDI step sequencer centered around the Raspberry Pi microcomputer. Step sequencers allow a musician to compose for virtually any MIDI instrument by manipulating a physical, accessible interface consisting of a grid of buttons which map pitches over a fixed and repeated length of time. Our sequencer will provide these features in a fully interactive and deeply controllable interface, which at the same time should be easy enough to operate that a performer can begin to compose with no prior instruction. Additionally, we hope to incorporate computer vision to allow for greater expressive possibilities.

In short, our step sequencer will satisfy a crucial musical need, while at the same time using camera-guided controls to add an fresh, innovative angle to the traditional sequencer.

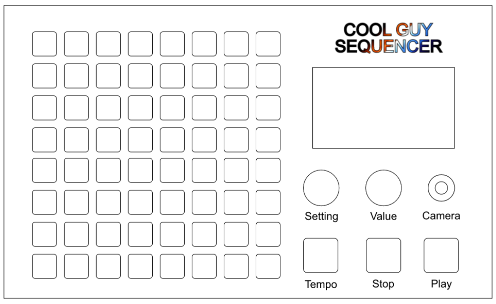


Figure 1: First prototype design.

2 Detailed Description

2.1 Usage

Hardware step sequencers tend to lie outside the range of affordability for most young or pre-professional musicians, who turn instead to software sequencers and digital audio workstations with much more sophisticated sequencing capabilities. However, we believe that the simplicity and accessibility of a hardware controller offers the formidable advantage of composing through a stripped-down, distraction-free interface. Unlike the soft-sequencer, the hardware step sequencer allows the composer to avoid time-consuming considerations of instrument choice, effects, or even volume, focusing strictly on the notes themselves. This makes it the ideal solution for sketching on-the-fly the moment inspiration strikes, as well as for young composers or musicians who are not as fluent in software.

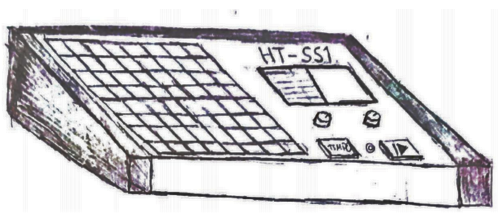


Figure 2: Sketch of prototype.

Perhaps its most useful application, however, is the use of the step sequencer as a live performance tool, one upon which the performer/composer can improvise and trigger melodic and rhythmic patterns for multiple instruments in real time. It is this live use-case that we hope to maximize in particular, adding camera-control and LED feedback in a way that encourages expressive interaction without increasing cognitive load.

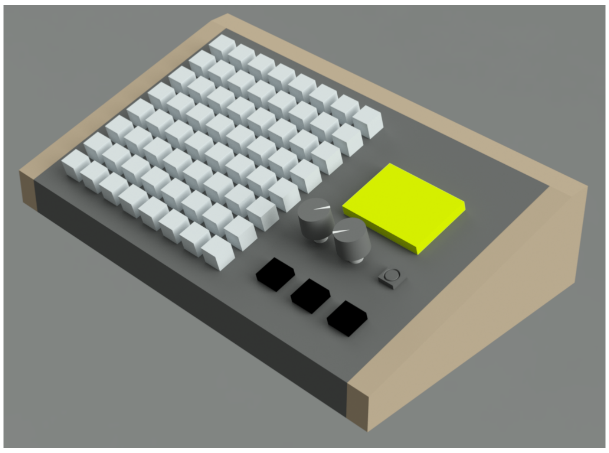


Figure 3: Render of the prototype from an isometric point of view.

2.2 Interface

To achieve our goals of simplicity and maximum expression, our interface will center around a traditional 8x8 grid of LED-lit buttons (hereby referred to as the "keypad"), which represent eight ascending pitches on the vertical axis, played over the course of eight beats from left to right. Upon power-up, the keypad buttons will appear dark, with a separate "tempo" button flashing continuously at the current tempo setting. The user may tap the tempo button at a new pace to alter tempo at any time.

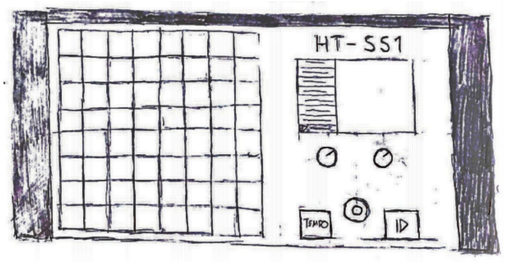


Figure 4: Sketch of top view.

As the user depresses various keypad buttons, each button will light up and remain so. Each row of buttons from the bottom to top of the keypad represent ascending pitches on the connected MIDI instrument (for example, the eight "white keys" of the C-major scale on a piano). Each column of buttons from the left to right side of the keypad represent the 8 beats of the musical phrase upon which the "lit notes" will be played.

Upon hitting the "play" button, the leftmost column of buttons will completely light up indicating the performance of all of that column's former "lit" notes on the first beat, then the second column/beat, third, etc. At each beat, the Raspberry Pi converts that beat's lit notes to MIDI "note-on" signals and sends them to the MIDI output on the back of the sequencer.

Thus a diagonal line of lit keypad buttons from bottom left to top right could, once the play button is triggered, perform a scale in ascending order on a MIDI piano, or strike each drum on a connected drum machine. Multiple notes are allowed on each beat, making chords possible. The row of notes/chords will repeat on an endless loop until the user presses Stop button. At any time the user may add notes by depressing dark buttons, or remove prior notes by depressing lit buttons.

MIDI
in
out
THRU

Figure 5: Sketch of rear view
with MIDI IN, OUT, THRU, and USB ports.

Via the MIDI OUT port, one can connect a MIDI instrument such as a keyboard, drum machine, GarageBand instrument, or even an array of MIDI-controllable lights. The MIDI protocol[1] allows for any instrument to receive the same note on/off signals, so no modification is needed at the step-sequencer level to allow it to control any MIDI-capable instrument(s).

A MIDI IN port on the back of the device allows the sequencer to receive master clock signals from MIDI master devices (such as a computer digital audio workstation) which may dictate tempo to the sequencer. MIDI THRU allows the sequencer to be connected to a long chain of MIDI instruments, and to retransmit MIDI IN messages not intended for it to another instrument connected to its THRU port. A USB port adjacent to the MIDI ports provides power and computer connectivity to update the internal software.

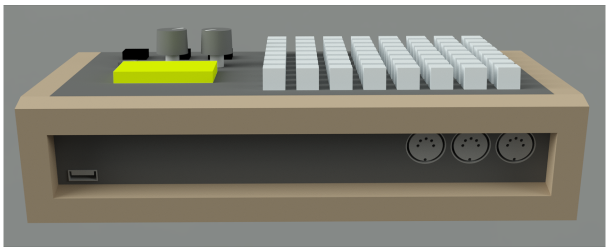


Figure 6: Render of the back of the prototype with MIDI and USB ports.

In addition to the extensive sequencer functionality, the user may manipulate two additional MIDI controls by placing a predetermined colored object such as a red ball in view of the built-in camera. By changing the vertical and horizontal position of the ball, the user can manipulate two MIDI control values, such as note length or volume. The camera will output a live video feed to the built-in screen, and display the parameters affected by the object's position.

The user may also access the advanced settings of the sequencer via the screen and encoder knobs beneath it. For example, one might change which type of notes are sent to the MIDI instrument (instrument notes or drums), alter the time signature, or save the current pattern in memory.

Below the screen and menu knobs are a set of dedicated buttons: tempo, stop, and start.

2.3 Software

The software for the step sequencer will be written in Python and run on the Raspberry Pi 3 microcomputer. The Raspberry Pi is the ideal platform[2] and language as there already exists a Raspberry Pi library in Python for the Trellis LED-button hardware we wish to use[3]. The Pi also has the needed input/outputs to connect the 8x8 keypad, camera, screen, and dedicated buttons for tempo and menu options. The high level of the programming language will also allow us to create objects with flexible sizes to represent the array of keypad buttons; this opens the possibility of adding an advanced feature later extending the range of pitches and doubling the phrase length, without having to change the underlying data structures.

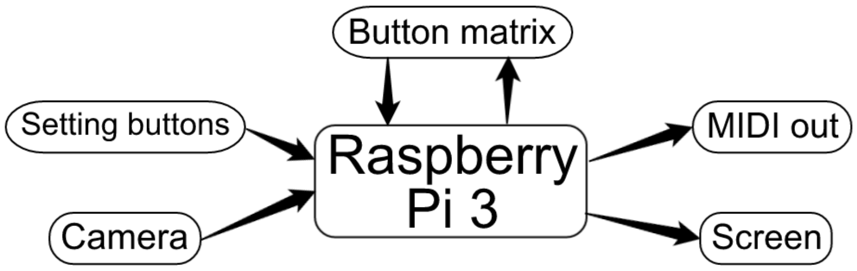


Figure 7: Signal flow within the sequencer.
The Raspberry Pi receives data from the 8x8 keypad, menu/setting/play/tempo buttons, and built-in camera, and sends data back to the keypad, as well as to the MIDI OUT port and screen.

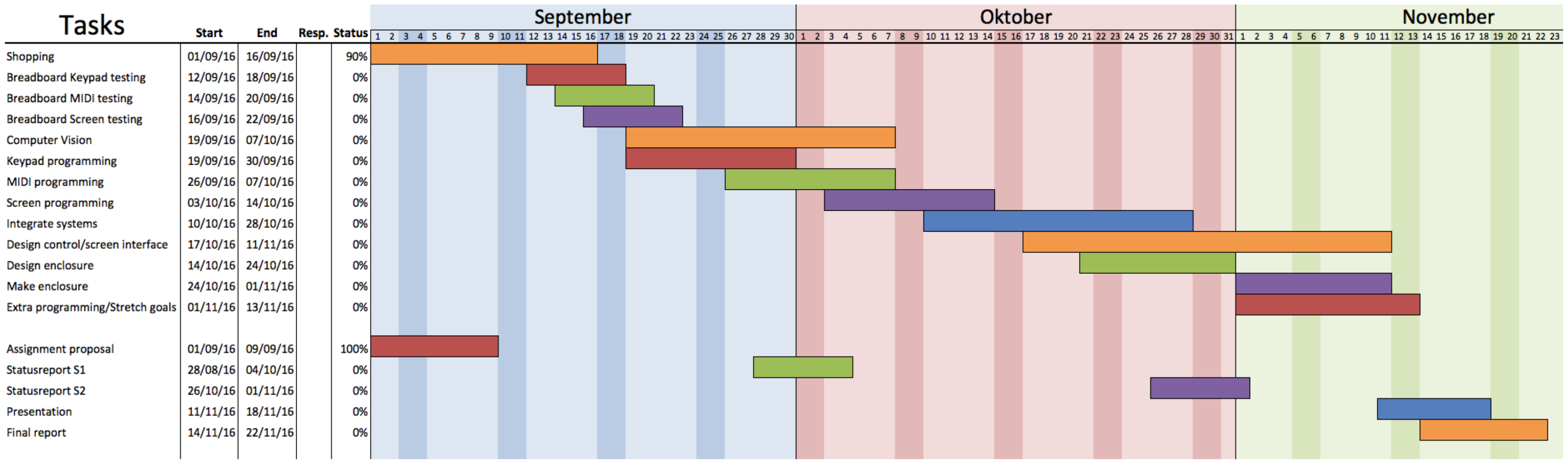
3 Cost Estimate

Partur	Verð [kr]
Raspberry Pi 3	4.891
SD card og reader	3.280
LED lights	2.423
Camera	3.666
Screen	3.504
Keypads	7.295
Cables, wires	4.000
Chassis	10.000
Import duties	8.218
	47277

Table 1: Cost Estimate

The majority of hardware has already been ordered. The only remaining unknown factor is the design and cutting of the chassis enclosure, which we will complete in the final sprint of the project.

4 Project Plan



Tasks	Start	End	Resp.	Status
Shopping	01/09/16	16/09/16		90%
Breadboard Keypad testing	12/09/16	18/09/16		0%
Breadboard MIDI testing	14/09/16	20/09/16		0%
Breadboard Screen testing	16/09/16	22/09/16		0%
Computer Vision	19/09/16	07/10/16		0%
Keypad programming	19/09/16	30/09/16		0%
MIDI programming	26/09/16	07/10/16		0%
Screen programming	03/10/16	14/10/16		0%
Integrate systems	10/10/16	28/10/16		0%
Design control/screen interface	17/10/16	11/11/16		0%
Design enclosure	14/10/16	24/10/16		0%
Make enclosure	24/10/16	01/11/16		0%
Extra programming/Stretch goals	01/11/16	13/11/16		0%
Assignment proposal	01/09/16	09/09/16		100%
Statusreport S1	28/08/16	04/10/16		0%
Statusreport S2	26/10/16	01/11/16		0%
Presentation	11/11/16	18/11/16		0%
Final report	14/11/16	22/11/16		0%

Figure 8: Gantt chart for the sequencer project.

The individual tasks for the project can be grouped into three major sprints, each one completed before the deadline of a status report or class demonstration.

4.1 Sprint 1

In the first sprint leading up to our first status report, we will aim to finish purchasing all the needed hardware, wire together all components on a temporary breadboard, and set up the Raspberry Pi. Specifically, we will run basic Python test programs on the Pi to practice lighting up and reading from the Trellis keypad (the 8x8 grid). We then want to make sure we can send MIDI-compliant signals out from the MIDI OUT port, which we can test by "wire-tapping" the MIDI signal to view its contents via a program such as Max/MSP. We'll then work on displaying the camera's video input on the computer screen along with menu text. At the same time, we want to begin to develop the computer vision abilities of the camera, allowing it to track specific objects. Once we are comfortable programming for all hardware components, we'll commence work on basic keypad operations, with the aim of sending basic note on/off signals out of MIDI OUT before we present our first status report.

4.2 Sprint 2

In the second sprint leading up to our second status report, we will finish developing the computer vision program and link it with our keypad/MIDI program, allowing the camera to control two MIDI effects. During this sprint we'll also focus on the design of the interface, especially the menu, encoder wheels, and dedicated tempo, stop and start buttons. MIDI IN and THRU will be tested and integrated into the sequencer program. Depending on how much time we have, we hope to allow for some basic usability tests before the second status report.

4.3 Sprint 3

In our final sprint we will finish any leftover integration tasks connecting the MIDI ports, keypad, screen, and camera to the sequencer program, thoroughly test for bugs, and design the hardware enclosure. One integration-related concern at this stage is to find out whether the Pi can keep perfect tempo during playback while performing computer vision tasks, and to make sure it prioritizes tempo above all else. Now that we have a clearer picture of the user experience, we can also make any needed last-minute changes to the keypad layout and menu design before we finish designing the software and enclosure. Finally, we will print or cut a chassis for the sequencer and all its hardware, solder all the parts together, and mount them inside the chassis. We hope to finalize the software before the class demonstration, and to finish the enclosure before the final report. If there is additional time during this stage, we can start to add some "luxury" features to the basic sequencer program.

5 Stretch Goals

The crowdfunding platform Kickstarter uses the term "stretch goals" to describe a list of optional "luxury" features the developers promise to add in case their product receives an overabundance of funding. Similarly, in the case that we have an overabundance of time, we have a similar wish list of luxury features for our sequencer.

5.1 Extended phrase length

Users are likely to quickly run up against the sequencer's strict eight-note limit on musical phrases. Thus we would hope to enable longer phrases as our first stretch goal. A simple menu operation (dedicated button or menu option) could allow the user to make the phrase 16, 32, or 64 beats long. Such an operation would copy the existing 8-beat pattern to the next 8 beats, making it possible to perform during playback without affecting the sound. In practice, the user would only be able to view and edit 8 beats at a time on the keypad, but he can quickly "page" or scroll to beats 9-16, 17-32, etc. The paging/scrolling button would need to be easily accessible (requiring no more than two clicks on the mode-knob) in order to be performable during playback. The currently editable eight beats should be at all times visible on the screen.

5.2 Scrolling octaves

As with phrase length, the user will likely want to compose in more than one octave (an 8-note scale). We could thus provide the option to use one of the rotary controls to "scroll" the available notes "down" or "up," so that notes below and above the default 8-note scale become editable. For example, rotating the menu encoder wheel one click clockwise would make the visible pattern shift "upward" by one row, providing a new blank row at the bottom of the keypad representing the note one step below the formerly lowest note. The range of currently editable notes should always be clearly visible on the screen as well.

5.3 Custom scales

By default the eight notes on the sequencer will represent the 8 "white piano keys" of the C major scale. In order to make the device more versatile we could add a menu option to select a different scale such as the minor, Lydian etc. while also picking a different key. Stretching the feature even further, we might even go as far as to allow the user to create his own custom scale with a fundamental tone of his own choosing. This would be especially useful when programming for MIDI drum instruments, where the desired drum sounds often are only available in higher octaves.

5.4 Alternate time signatures

The default time signature of 4/4 (four quarter notes per bar equivalent to eight 8th notes) could be editable to 3/4, 6/8, or even more exotic prime-number time signatures such as 5/4 or 7/8.

5.5 Modulation

This option would allow the user to modulate a specific parameter available in the instrument connected to the device, which most often adjusts the cutoff level of that instrument's filter.

5.6 Pitch bend, portamento

Another option might be to allow the user to control the fine-tuning of every pitch, a MIDI-defined control called "pitch bend". Additionally the user might adjust the rate of "glide" or "glissando" between notes, a MIDI control value known as portamento.

5.7 Swing

In order to allow the user to make the music more lively, the user could manually move each note a little bit earlier or later in time, depending on the user's preference. The most common application is to create a "swing" effect where even-numbered beats arrive slightly later.

Such an implementation might behave as follows: The user chooses a menu item indicating that he may edit the "swing" of each note. The sequencer then enters "swing" edit mode, where instead of displaying the pattern of pitches/beats, the sequencer would light up the bottom four pads of each column on the 8x8 keypad. Pressing a dark key in the upper half of any column would move the light so it lights up all keys up to and including that button. This would delay that column's beat. Pressing a key lower then the topmost lit up key would reduce the size of the light strip as well as making that beat play a little earlier. A column with only the bottom four buttons lit thus represents a beat that arrives exactly on time.

5.8 Zoom

By default, the sequencer divides one bar into eight beats, or eight 8th notes. We could add a "zoom" feature that allows the user to increase or decrease the resolution of the keypad, choosing for example whether each button programs one 8th note, or faster-moving 16th or 32nd notes.

For example, the user would be able to choose the zoom function from the menu; he then would adjust one of the rotary knobs to zoom in or out. Zooming in would increase the resolution such that the user would be able to program 16th notes, 32nd notes, etc. In any zoomed state, the user only sees the first eight notes, and must manually "scroll" to see the next group or zoom back out to view the whole bar.

5.9 Live mode / MIDI controller

This function would allow the 8x8 keypad to function as a set of live MIDI pads that can trigger 64 different notes, generally referred to as a MIDI controller. In this "live mode", one could strike a pad and immediately send that MIDI note out from the device in real time. This allows the user to improvise on the device as if it were an electronic drum set or a keyboard.

5.10 Single-note edit mode

This major interface feature would make the process of adjusting note-related parameters more intuitive, accessible, and faster. The user would simply hold down a dedicated control button and simultaneously press a lit note on the keypad in order to edit all the settings of that specific note. Doing this would reset the LEDs so that each column would show the current level of each of eight note-specific settings, such as velocity, swing, portamento, etc. Similar to swing mode, the number of lit keys ascending from the bottom of each column could be manipulated up or down by pressing a dark key (up) or a lit key (down).

References

[1] Summary of MIDI Messages. The MIDI Association. https://www.midi.org/specifications/item/table-1-summary-of-midi-message (accessed September 9, 2016).

[2] Raspberry Pi 3 - Model B - ARMv8 with 1G RAM. Adafruit Industries. https://www.adafruit.com/products/3055 (accessed September 9, 2016).

[3] DiCola T. Adafruit Trellis Library For Python. GitHub. https://github.com/tdicola/Adafruit_Trellis_Python/blob/master/README.md (accessed September 9, 2016).