Axoloti (and Ksoloti) panel controls

zenpho@zenpho.co.uk github.com/zenpho

7 June 2024

Axoloti boards designed by Johannes Taelman (and clones like Ksoloti) are programmed and configured using a software editor. I find on-screen controls do not provide tactile visceral joyful control of sound. I like joyful sound... In 2016, I *hastily* constructed a panel with 12 rotary potentiometers, 8 push switches in an R_2R ladder to select banks of controls, and a 2x16 character display over I2C. Accompanying software (*also* hastilly constructed!) uses undocumented features for convenient tactile joyful etc workflow. Simply "paste-in" objects to any pre-existing patch, nominate controls (up to a maximum total of 96 in 8 banks of 12), and everything just works automatically. Yay!

1 Overview

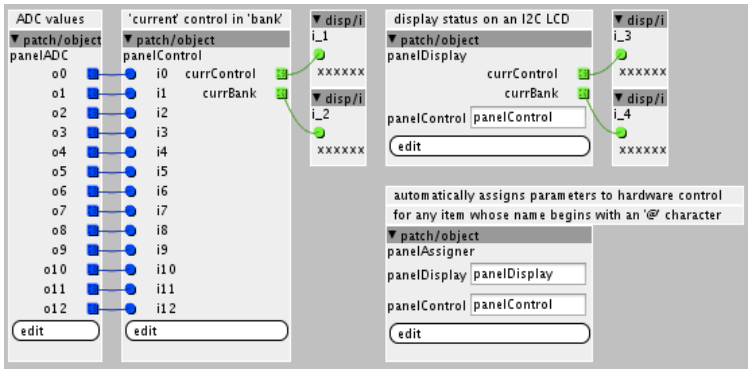

Figure 1: Detail of labelsystem5a.axp

Figure 2 and 3 show the front and rear of the hastily constructed instrument. Figure 4 shows schematic wiring connected to Axoloti Core pins. The system uses 12 rotary potentiometers, 8 push switches in an R_2R ladder to select banks of controls, and a 2x16 character display over I2C. A software overview is provided on the following pages.

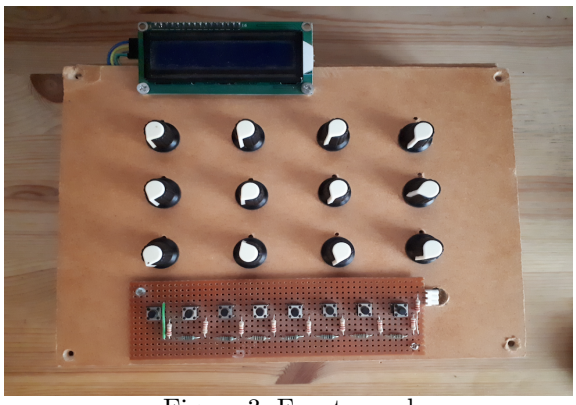

Figure 3: Front panel

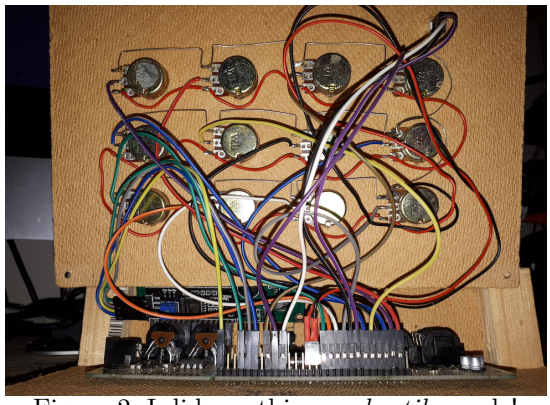

Figure 2: I did say this was *hastily* made!

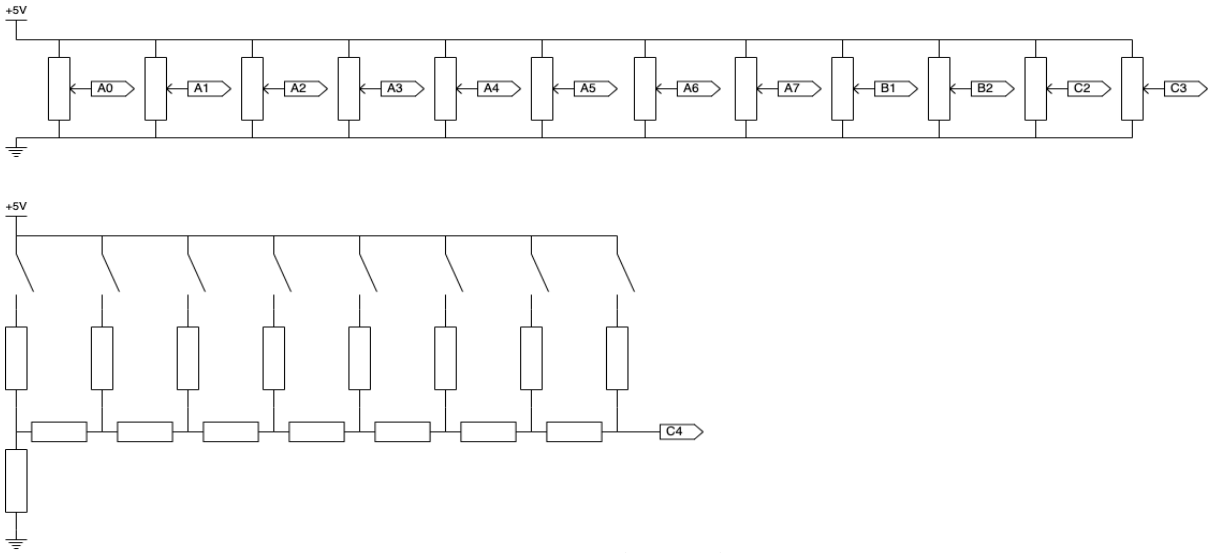

Figure 4: Panel schematic showing Axoloti ADC input pin identifiers

2 Software overview

The software monitors rotary control and bank selection switch state and reacts appropriately. Preferred workflow is to add labelsystem objects to an existing patch and *nominate* (more on that later) which parameters will be assigned to hardware controls. If more than 12 parameters are nominated, the system can use upto 8 banks of 12. When turning a physical control the 2x16 LCD clearly identifies the parameter label, current bank, and any unused banks or controls. All changes are "hooked" (aka "pickup") which avoids sudden jumps when switching banks.

See Table 1 for a description of files provided. A good starting point is `labelsystem5a.axp` (shown in Figure 1 on page 1) which includes a complete demonstration of the system. Since AXP "patch" files can contain embedded C sourcecode you may copy-paste into your own patches to enjoy.

Filename	Description
`labelsystem5a.axp`	Demonstration. *Start here!*
`labelsystem5a-chart.axp`	With `disp/chart/p` useful to debug hardware.

Table 1: A brief description of all provided files

2.1 Objects

All objects described in Table 2 are required. Do you need support for OLED displays over SPI? Controls from I2C ADC modules? Hack on my code. :)

Object	Description
`panelADC`	Reports 13 ADC values from hardware controls. Requires 12 potentiometers and a button array with R-2R ladder.
`panelControl`	Controls patch parameters using inputs from `panelADC`. Reports the state of the system as two int32 values. curControl = which control is touched, curBank = which bank of controls is selected.
`panelAssigner`	Assigns *nominated* parameters to `panelControl` and `panelDisplay`. Checks objects whose name begins with an '@' character. Truncates names if necessary. Not all parmameter types are supported yet – can you help? Subpatches with 'child' parameters are supported somewhat.
`panelDisplay`	Display recently touched control labels and identifiy used/unused clearly. Requires I2C 2x16 LCD using pins PB8=SCL and PB9=SDA.

Table 2: A brief description of objects

The C code (ab)uses, to my knowledge undocumented, features of the Java based software editor related to parameter handling. Everything works well on firmware E95BAC96 and editor 1.0.12, but no other configuration has been tested. Send bug reports, fix bugs, teamwork, yay! :)

3 Suggested usage

Seee the demonstration in `labelsystem5a.axp` for suggested methods of usage and workflow. The simplest way to assign parameters to hardware control is to rename with the first character as an '@' symbol. For example '`gain_1`' becomes '`@gain_1`' as in Figure 5.

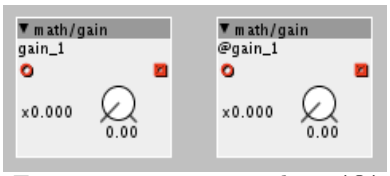

Figure 5: renaming with an '@' character

Parameters inside subpatches using 'on parent' are supported. A portion of the parent identifier of the parent will be shown on the LCD. Long identifiers are truncated preserving first and last characters.

Not all parameter types are supported. I help to support `ctrl/toggle`, `ctrl/button`, `ctrl/cb16`, `ctrl/i`, and `ctrl/i radio` types. CAN YOU HELP?