MEng Thesis!

Josh Gordonson

May 7, 2015

1 Motivation

The breadboard has been a staple substrate for electronic construction over the last century. At the dawn of a growing interest in amatuer radio, resourceful tinkerers used planks of wood to secure and ruggedize their electrical handiwork. Conductive nodes, such as nails or copper rails, were driven into the non-conductive boards, providing anchors and contact points that were electrically isolated from the rest of the circuit. Components were soldered or wire-wrapped to the nodes, and sometimes secured by non-energized nails or screws. This construction technique provided a lot of artistic freedom in circuit construction, but was relatively time consuming and required relatively heavy hand tools, such as a hammer or drill.

The solderless breadboard is the cannonical tool given to students taking introductory courses in the field. Rather than driving nodes into arbitrary locations, component leads are inserted into contact points that allow rapid semi-rigid construction of circuits with no other tools. The layout of a solderless breadboard is designed to be compatible with a plethora of powerful integrated circuits, enabling complex designs. Modern solderless breadboards have come a long way from their namesake wooden ancestors, but there is still room for improvement.

The shortcomings of solderless breadboards lie entirely within the art of breadboarding. The intent of breadboarding is to physically realize a circuit. Often, this involves designing or using a reference schematic to guide construction, but circuit improvisation is not uncommon. Inserting components and jumper wires into contact points is straightforward, but poor contacts, broken wires (inside insulation), and mis-inserting leads can plague designers for hours on end or even destroy components. A meticulous breadboarder can successfully realize a circuit without error, but for the uninitiated it is difficult to justify the additional time and care required to plan and build. Breadboarding is a skill that is learned over time, but small errors can lead to excessive frustration and turn students off to the field. To satisfy the requirements of the Masters in Engineering program, I propose a solution to some of the issues with the solderless breadboard.

2 Overview

A confident linkage between the the electrical and mechanical domains is required to construct a circuit. On larger scales, the mechanical structures (eg. contacts, wires, components) that create electrical circuits are visible in plain sight. It is simple and reliable, then, to determine the circuit representation of a mechanical structure that has no hidden connections. Breadboards often obscure connections due to their construction. The regular nature of a breadboard, a grid of contact points arranged in rails, makes it difficult to keep track of which rail is connected to what circuit node. When combined with the opaque nature of most components and the poor reliability of breadboard contacts, visually verifying complex circuits on a breadboard becomes infeasible. Many of the inherent problems with breadboards stem from this open-loop nature of breadboarding, where visual cues are not enough to determine the electrical circuit from the mechanical structure. A symptom of open-loop construction techniques is a mismatch between the mental model and the physical realization of the system at hand. I seek to close this loop by designing and implementing a circuit-sensing breadboard.

At first glance, the proposed problem is straightforward. A standard-issue breadboard has 130 conductive rails. Each of these rails can be connected to an array of ADCs (analog to digital converters) and DACs (digital to analog converters) controlled by a microcontroller. With access to every node in the circuit, by iteratively stimulating and probing the circuit it is possible 1 to determine the circuit topology constructed on the breadboard. Once the topology has been determined, a schematic representation of the constructed circuit, from the perspective of the breadboard, can be displayed. This visual display will provide an accurate depiction of the electrical circuit at hand, effectively closing the loop on breadboard construction. Designing a circuit-sensing breadboard will require a substantial amount of hardware and software infrastructure. Fortunately, the majority of this infrastructure is useful in its own right. A hardware test-bench will have to be constructed to interface each of the 130 rails on the breadboard with a hand full of ADCs, DACs, and the microcontroller. The test-bench should include a pcb-mounted breadboard, an ADC and DAC multiplexing board, firmware to control sampling and stimulating the breadboard, and software to stream the sampling data to a computer for display. Effectively, the test-bench is a 130-channel digital breadboard-oscilloscope/function generator. Since software is faster to prototype with, I will be doing most of the theoretical circuit topology work in software. A test-bench for a circuit simulator will have to be built. Ideally, this test-bench will allow a circuit-sensing program to probe and stimulate node voltages of an automatically generated circuit, as if it were running on a microcontroller connected to the hardware test-bench with an arbitrary circuit built up. After the test-benches have been constructed, they need to be integrated and the circuit-sensing program will be tested on physical circuits. Finally, a schematic drawing program will be written to visualize the circuit-sensed topology.

3 Work to Date

3.1 PCB integration of Breadboard

Tool design is an art. There are many fields of study dedicated to the art of design, none of which I've spent much time exploring. Despite this, I have developed my own sense for how tools should feel when they're in use. The circuit-sensing breadboard should seem no different in both size and weight from a standard breadboard. (I'm not saying that standard breadboards are perfect, but I am deciding not to spend time redesigning the breadboard) To that end, I've decided that the best way to interface with an existing breadboard is to physically mount that breadboard on a printed circuit board. I've developed a new technique for mounting a breadboard on a PCB.

A modern solderless breadboard is usually composed of three components - a plastic face, metal finger springs (rails), and a foam backing. The finger springs are designed to fit snugly in the plastic face, which provides structure for the entire board. The foam backing keeps the finger springs from being pushed out the back of the plastic face. A breadboard can be mounted to a PCB by removing the foam backing and surface mount soldering the finger springs to a PCB, then pressing the plastic face onto the mounted rails.

Initial tests of this technique involved small numbers of finger springs hand-soldered one-by-one. The results of these tests were promising, but alignment proved to be an issue. To mitigate this issue, a rail-holder was 3d-printed to losely hold each finger spring in alignment. Once all of the rails were inserted upside-down in the rail-holder, a dot of super glue between two dots of solderpaste were dispensed on each rail. The rails were then adhered to the PCB by pressing the PCB upside-down onto the array of aligned rails, such that the rails lined up with pads on the PCB. To aide with alignment during mounting, the rail-holder had four global alignment pegs that mated with four holes drilled into the PCB. Finally the assembly was reflowed with hot air and the plastic faceplate was pressed on.

This technique was successful and seems feasible for high-production runs, though I should probably talk to an expert on the subject.

3.2 Resistor network sensing algorithm

3.2.1 Pre-fabbed circuit algorithm/requirements (topology reconstruction)

3.2.2 Incremental modeling algorithm/requirements

3.3 Software Testbench

3.4 Hardware Testbench

3.4.1 Programming

In order to handle the high datarates associated with capturing eight 1MSPS ADCs we use the Direct Memory Access (DMA) controller to asynchronously store samples. Triggered by a timer interrupt, the DMA controller can store 8-bit wide GPIO port data. Since the ADCs are serial output devices each additional ADC-sample-bit read will create an 8-bit wide piece of data, one bit for each ADC. This data will have to be decoded later on to achieve high sample rates.

3.4.2 Timers

To clock the SPI ADCs we need a clock signal. There are several timers to choose from on our microcontroller of choice, but we'll look at timers 2-5.

3.4.3 STM32 stuff

3.4.4 Arduino stuff

References