Educational Smart Breadboard

Team 68 - Chinemelum Chibuko, Minseong Kim and Mostafa Elkabir
ECE 445 Project Proposal - Spring 2018
TA: Kexin Hui

1. Introduction

1.1. Objective

The first time any kid or college student learns circuits, he/she will encounter breadboards. A breadboard is a board for making an experimental model of an electric circuit. It is essentially a learning tool for students to gain more knowledge about circuits. Breadboards can be used for prototyping which is the process of testing out an idea by creating a preliminary model from which other forms are developed or copied. However, one of the big problems any student will face with breadboards is the issue of debugging. Debugging on the breadboard is particularly difficult. This is because of the relative small spacing between holes in the breadboard. As a result, wires tend to become clustered and make it a bit difficult to debug. Thus, the need for the Educational Smart Breadboard.

The goal of this project is to help solve the problem of debugging on the breadboard, with educational emphasis.This is so the students can focus on actual debugging skills and not on the mess of clustered and intertwined wires.

There are two main ways in which we intend to help make debugging easier for the student. One is checking the voltage of each line by just inputting line location. The second is by allowing a chip test. It will improve the quality of learning in introductory electronics classes such as ECE 110.

1.2. Background:

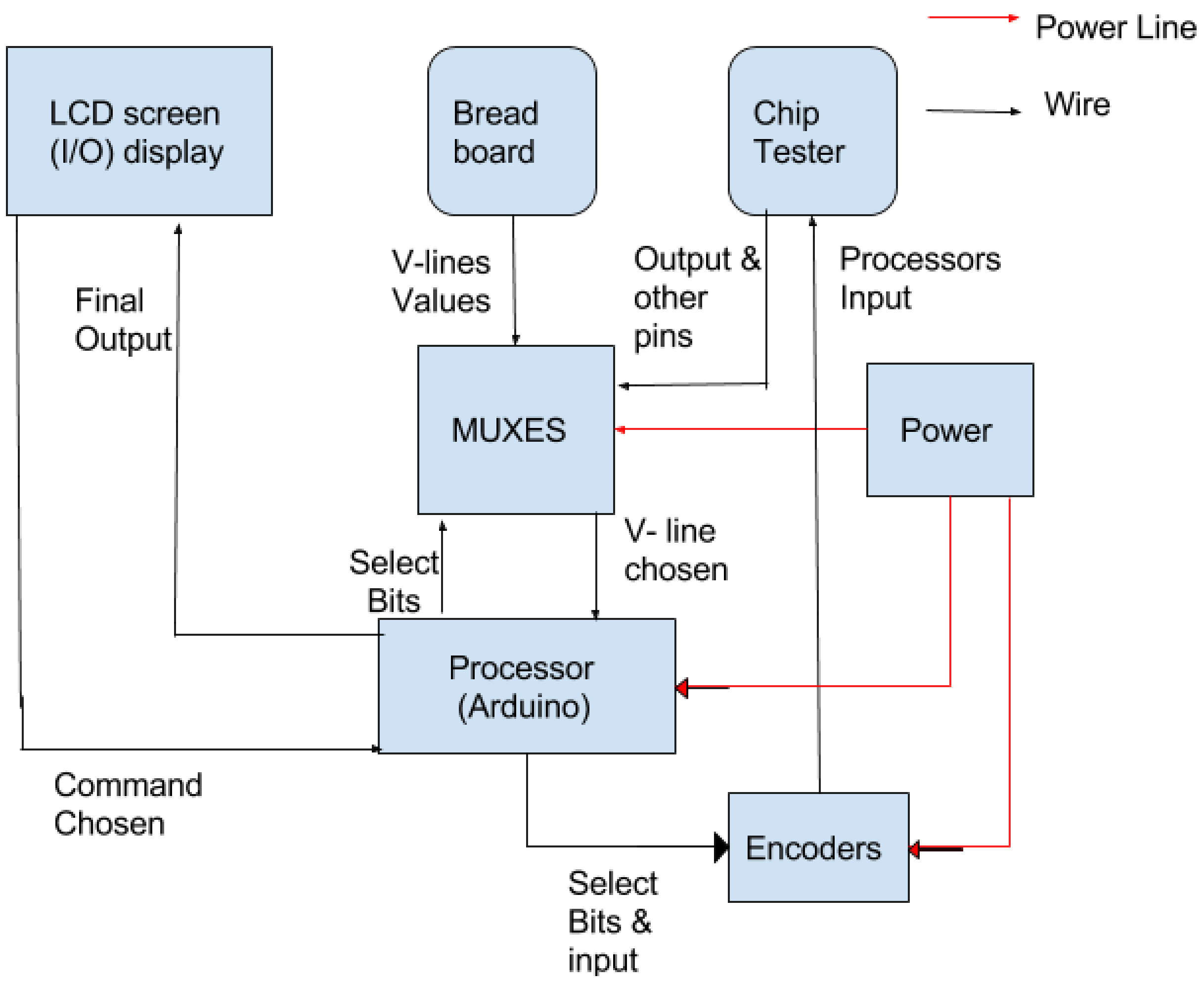
Debugging on the breadboard could get a little bit difficult. The small spacing between the holes causes the wires to be clustered and messy. Sometimes the chips could be faulty without the student knowing and the student might end up taking out all the wires without knowing it is just a faulty chip. Our smart breadboard for that purpose functions as a quick chip tester, when a chip is placed in the chip testing area. The main idea is running all possible configurations to input pins and generate truth table then the processor will compare it to a lookup table. Also, the smart breadboard will function as the voltage and logical value checker when the user wishes to check what is going around for his/her specified pin.

Surprisingly, there has not been really a simple debugging-purpose-extended breadboard. This may be because breadboard is not actually used for industry purposes, and we usually learn debugging by connecting it to multimeters, voltmeters and oscilloscope. But while these natural debugging methods are versatile, using them can be waste of time for digital circuitry purposes. For example we do not wish to manually check all pins to check whether chips are functioning correctly.This project is particularly important because it will help the learning experience and would help improve the standard of teaching electronic circuits. It will make it easier for students to debug and concentrate on the functionality of the circuit

1.3 High-level requirements list:

- The breadboard must test any listed chip correctly.
- The project can display any voltage value at voltage line
- The whole project must cost less than $40

2. Design:



LCD touch screen (I/O):

Physically, LCD touch screen will be located at either left or right of the breadboard. LCD screen is the main user debugging interface. At the start of any debugging operation, a user would see the welcome message, asking for the user to select out of options. While we may expand on the options in case we successfully implements our design, for now our options are: checking physical voltage and logical value (which can be said to fluctuate) of each line of pins, and chip testing. Users will then press the button on the sides of the screen. The screen, receiving messages from the processor, will then print out the sub-options related to each option.

For checking voltage and logical value of each pin, the screen will ask the user input for which line of pins it wishes to evaluate. The user presses the button on the screen using the touchscreen pen, then the screen prints out voltage and logical value. For chip testing, a user would put a gate on the dedicated region of the breadboard. Then the user would enter the chip testing mode by pressing on the screen, and the screen would ask for the user to specify Vcc, Gnd, input pins and output pins, and what logical value each output pin would be relative to each input pin configuration. This allows us to test chips that have stable and fixed input-output relationship. Thus, this excludes stateful chips like counters, but this can be extended - we will see this more in the description for the processor.

On the screen, in case messages that need to be printed out is long, then an additional button will be printed in the screen so that messages can be scrolled.
For hardware specification of our touchscreen, we will use the 2.4-inch TFT Arduino touchscreen shield. (For example, see [1].)

Power for a LCD screen will be provided by Arduino, as we are going to use LCD screens tailored for Arduino uses. Thus, the power concerns for the LCD screen are relegated to the power concerns for Arduino.

Requirement: Big enough (approximately 2.4 inch screen) to print out relevant messages and relevant buttons without looking too cluttered. Clock concerns will not be there, as our eyes operate slower than screens available today. Using market-available Arduino-compatible LCD screens, though depends on what Arduino we are going to use.

Processor:

Our first choice for the processing unit would be Arduino - more specifically, Arduino UNO. [2]
Arduino provides 14 digital input/output pins, which are enough for processing breadboard signals, as we rely on MUXes to narrow down to one signal from just one line of pins.
Arduino is capable of providing power, and thus Arduino will be used as the voltage source for MUXes and the chip testing unit. Current-wise, maximum current for each MUX, to use the reference SN74LS150, is 1mA. Arduino has 500mA max current limitation for the output voltage [2][3], so this is not a problem, since we are going to less than 500 chips.
Arduino will be supplied with 7V voltage battery, according to the specification limit [2][3].
For the correct operation of our smart breadboard, the processor needs to be programmed so that appropriate actions can be taken. This literally a copy-paste of our specification, so will not constitute a serious problem.

Requirement: The total current for all the pins must be less than 500 mA, 7V power supply, with the given tolerance in the power supply. Must be programmed with the correct program.

MUXes:

There is a limited number of inputs and outputs that Arduino can accept. To overcome this - allowing accesses to every pin, multiplexers are needed. One can say that the MUX unit is the intermediate interface between the user breadboard and Arduino, and the sole way the breadboard unit interacts with the processor.
Physical design-wise, we would like to have the MUXes sit next to the Arduino. The unit of MUXes and the processor would sit below the breadboard so that user breadboard actions do not get interfered.
Based on a 60 voltage lines bradboard. We will be using nine 8-to-1 multiplexers. Where the breadboard is broken into 8's and each of them will have as input each voltage line representing each ⅛ area of the breadboard. As the processor gets the input from the screen, it will first give the select bits to all the 8 voltage lines, then those output will be inputted in a general 8-to-1 mux representing the whole breadboard. The processor will choose which to access by controlling the select bits.
Our starting choice for MUXes is SN74LS150, and will be our guiding reference when we implement the project. The specification for SN74LS150 states that minimum supply voltage allowed is 4.75V, while maximum supply voltage allowed is 5.25V. This is the power supply tolerance we have. [5]

Requirement:
Enough power to supply all the muxes. Using 5 chips, will require ~705 mw. 5V regulated power usually required, with tolerance level of ±0.25V.

Encoders:

Based on a chip with max pins of 24, the encoder will receive the different combinations to feed the ship through the processor. We will use 6 1-4 demultiplexers. The processor will provide select bit and inputs to the encoder, which will output 4 bits going to 4 locations in the chip testing area and do that for all the area. Then the processor will change the encoders outputs simultaneously by controlling its inputs.

Requirements: limit of number of pins <25.

Power Supply:

We will have a 7 volts battery supplying the arduino and MUXes. We will have a voltage regulator to ensure the voltage going into MUXes is ~5V.

Requirement: A market voltage regulator regulating/supplying 5V voltage up to tolerance level of ±0.25V. 7V battery, up tolerance level of ±1V. It is easy to find a power supply and a regulator satisfying these demands, so not heavily binding.

Breadboard:

The user breadboard will be a traditional breadboard, but very minimally modified so that the breadboard can connect to MUXes without cluttering user inputs. a breadboard has five pins on each line of pins other than vertical line of pins assumed to provide Vcc and Gnd. We will follow the same design of the reference breadboard. It is assumed that maximum operating Vcc voltage of user chips of the breadboard is around 5V, with tolerance level of +0.25V. Furthermore, user chips are assumed to have maximum current drawing of 1mA. To allow flexibility, a user will provide separate power sources for their user chips. However, in case we assume all chips have 5V reference voltage, then we can easily modify our setup so that our power supply provides power for user chips.

Requirement: # of pins <65, as we are going to use eight 8-to-1 multiplexers. Each horizontal line of pins has five user-accessible pins, but actual specification-wise, it has six pins. User chips assumed to have maximum operating voltage of 5V, with assumption of +0.25V tolerance, with 1mA max current limit.

Chip Tester:

The breadboard will have an area for chip testing. The user will place a chip, and input the specifications of the chip in the touch screen. The user will provide the Vcc voltage of the chip less than or equal to 5V following the screen message. Then the user will provide which pins are Vcc, Gnd, input and output. For input pin configuration, the screen will ask for the logical value

of an output pin, which allows the processor to build a lookup table. The processor will then input all configurations to the input pin using Arduino-supplied output pins, and compare output to a lookup table it stored in the memory. The screen will alert the user of discrepancy or logical value fluctuation. We will assume that maximum voltage of 34% of the user-specified Vcc of the chip, relative to the provided Gnd, as logical 0, with the minimum voltage of 66% of the user-specified Vcc of the chip as logical 1. The processor will print out to the screen fluctuating value message (specification thus not met) in case it receive values in between.
We will have to sample output pin voltage several (20, to give an approximate value) times in order to check output voltage fluctuation. The processor clock is usually fast enough detect any fluctuation, so this is not a binding problem.
We can extend the chip tester to cover some stateful chips like counters. In that case, we would have to program the processor so that it asks for counting behavior of chips. But there will be no way any extension would cover all stateful chips with unstable input-output relationship.

Requirement: Proper connections to Arduino.

Risk Analysis:

The project's computation is based on the processor. If during the computations too much current, or voltage that might make the processor have wrong values, or stop working. All of the computations have to be done simultaneously, then gate delays is something we will need to consider when doing computations. The muxes have to connected in a way where they function parallely as well, to make sure each segment is done at the same time.
A faulty battery or a large current drain can cause the battery to explode. Safety is our first concern, as the main users will be students, and children.

Ethical and Safety Issues:

There is a number of safety issue first the battery can be faulty. If there is a huge current drain by the circuit of faulty connections by the user smoke might rise and the user might inhale which would lead to a variety of health issues.
As the user will supply the power for his own circuit, a number of safety issue might occur, one example is having a short circuit or applying too much voltage to the circuit. This might damage the chips they are using or any other electric tool such as applying too much voltage to a BJT which could burn the metal.
The ethical codes of IEEE [6] and ACM [4] will be followed strictly when implementing this project. However, code 1.2 of ACM code of ethics states “Avoid Harm to others“, harm might occur in wrong usage of the circuit or applying too much power of some sort. In the other hand

the capability for students to learn from this tool is beyond that. Also the safety issues rising come from the wrong usage of outside tools**.**

Reference:

1. Amazon.com. (2018). *HiLetgo 2.4 Inch TFT LCD Display Shield Touch Panel ILI9341 240X320 for Arduino UNO MEGA*. [online] Available at: https://www.amazon.com/HiLetgo-Display-ILI9341-240X320-Arduino/dp/B0722DPHN6/ [Accessed 8 Feb. 2018].
2. Arduino. (2018). *Arduino Uno Rev3*. [online] Available at: https://store.arduino.cc/usa/arduino-uno-rev3 [Accessed 8 Feb. 2018].
3. Staples, G. (2018). *Arduino Power, Current, and Voltage Limitations*. [online] Electricrcaircraftguy.com. Available at: http://www.electricrcaircraftguy.com/2014/02/arduino-power-current-and-voltage.html [Accessed 8 Feb. 2018].
4. Acm.org. (2018). *ACM Code of Ethics and Professional Conduct*. [online] Available at: https://www.acm.org/about-acm/acm-code-of-ethics-and-professional-conduct [Accessed 8 Feb. 2018].
5. Ee-classes.usc.edu. (2018). *SN74LS150 data sheet*. [online] Available at: http://ee-classes.usc.edu/ee459/library/datasheets/sn74150.pdf [Accessed 8 Feb. 2018].
6. Ieee.org. (2016). IEEE Code of Ethics. [online]. Available at: http://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 29- Feb- 2016].