Design Document

Team 68: Educational Smart Breadboard
Chinnies Chibuko, Mostafa Elkabir, Minseong Kim
TA:Kexin Hui
2/22/18

I. Introduction

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. Furthermore, there may be too many wires in a TTL circuit such that using a voltmeter can be such a headache. 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 row by just inputting row location. The second is by allowing a chip test. It will improve the quality of learning in introductory electronics classes such as ECE 110.

Background

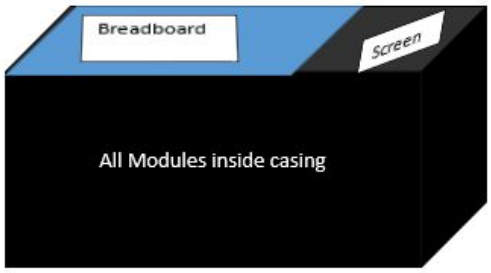
Debugging on the breadboard could get a little bit difficult. The small spacing between the holes causes the wires to be clustered and messy. Sometimes the chips could be faulty without the student knowing and the student might end up taking out all the wires without knowing it is just a faulty chip. 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

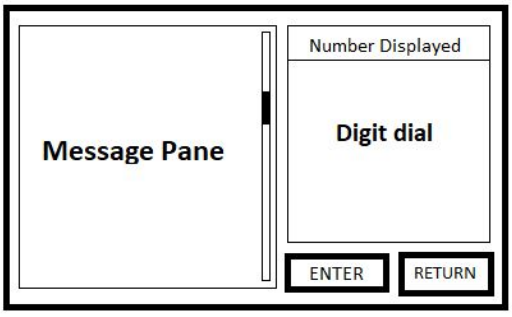
High Level Requirements

● The breadboard must test any listed chip - a chip with stable and fixed input/output - correctly.
● The project can display voltage value at each row
● The whole project must cost less than $40

II. Top-level Physical Diagram



Top-level

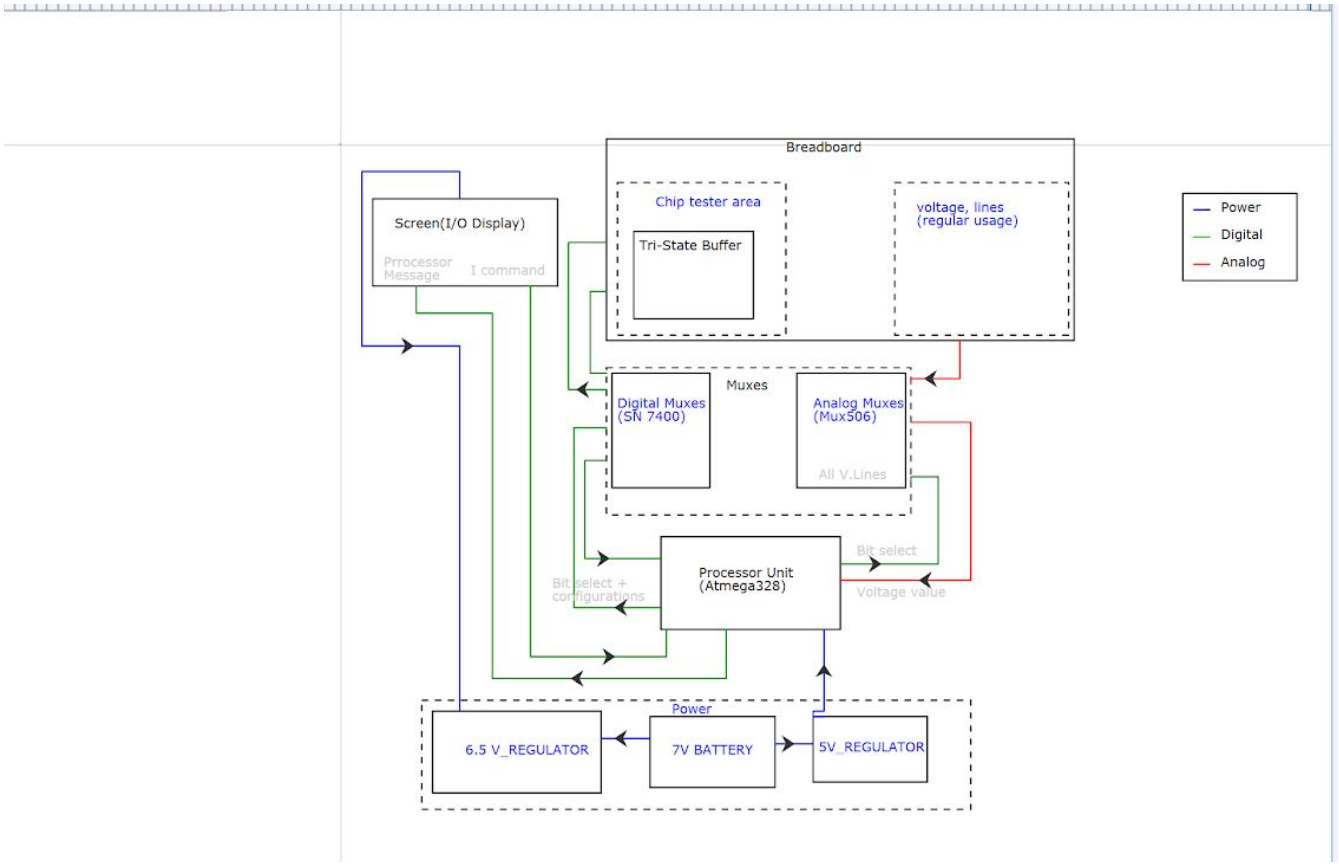


Screen Pane structure

The breadboard module, in its external appearance will be a typical breadboard. The size of the smart breadboard would depend on the number of rows of the breadboard module and the size of the screen. We plan to support 60 rows, so our physical dimension of the final smart

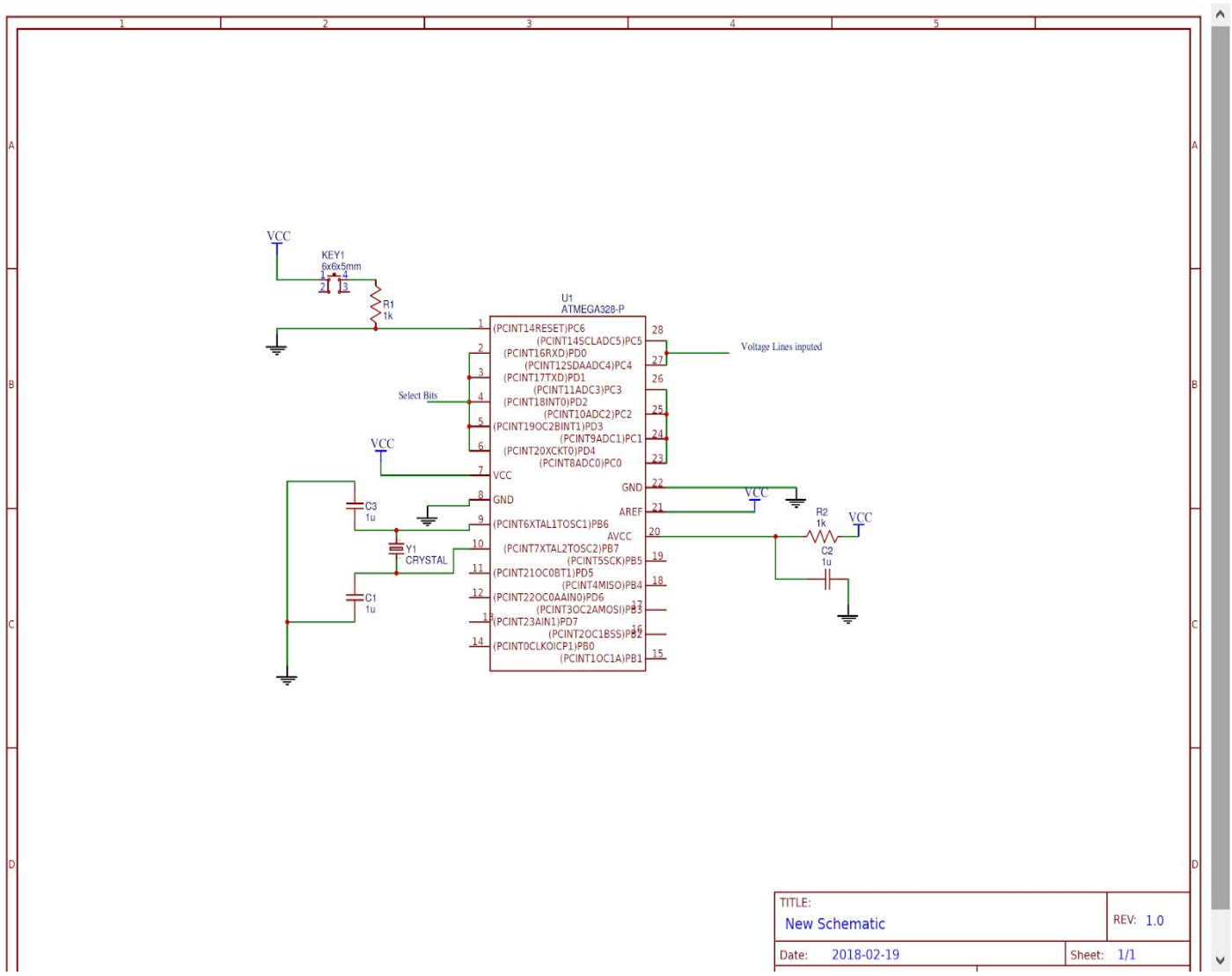
breadboard and the screen would directly depend on this measure, with 30 rows horizontally and 10 holes per each horizontal side.

III. Block Diagram

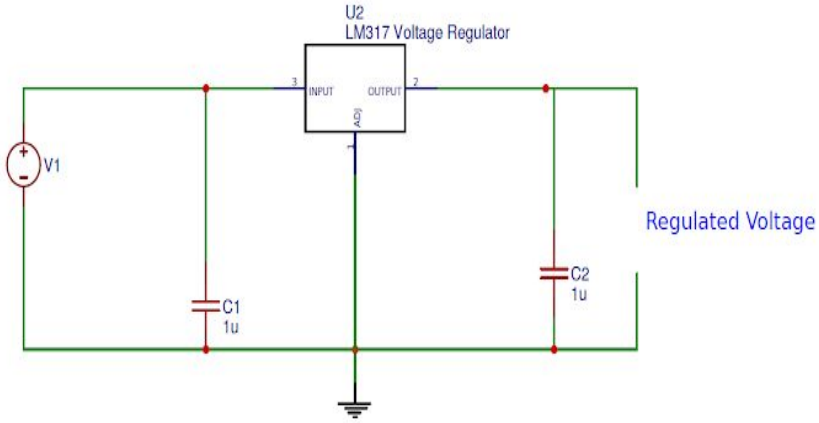


IV. Circuit Schematics

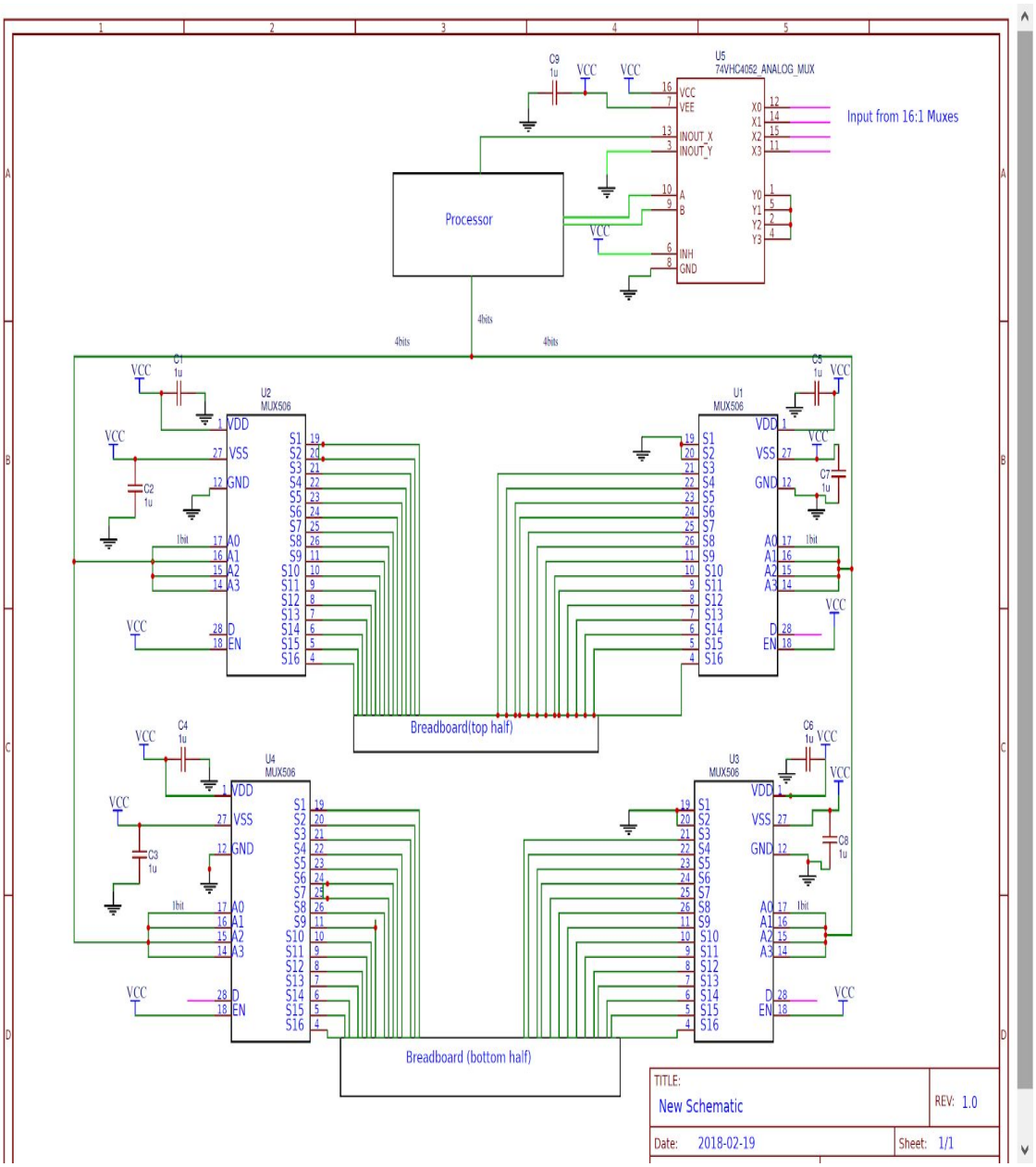
Processor Unit:



Voltage Regulator:

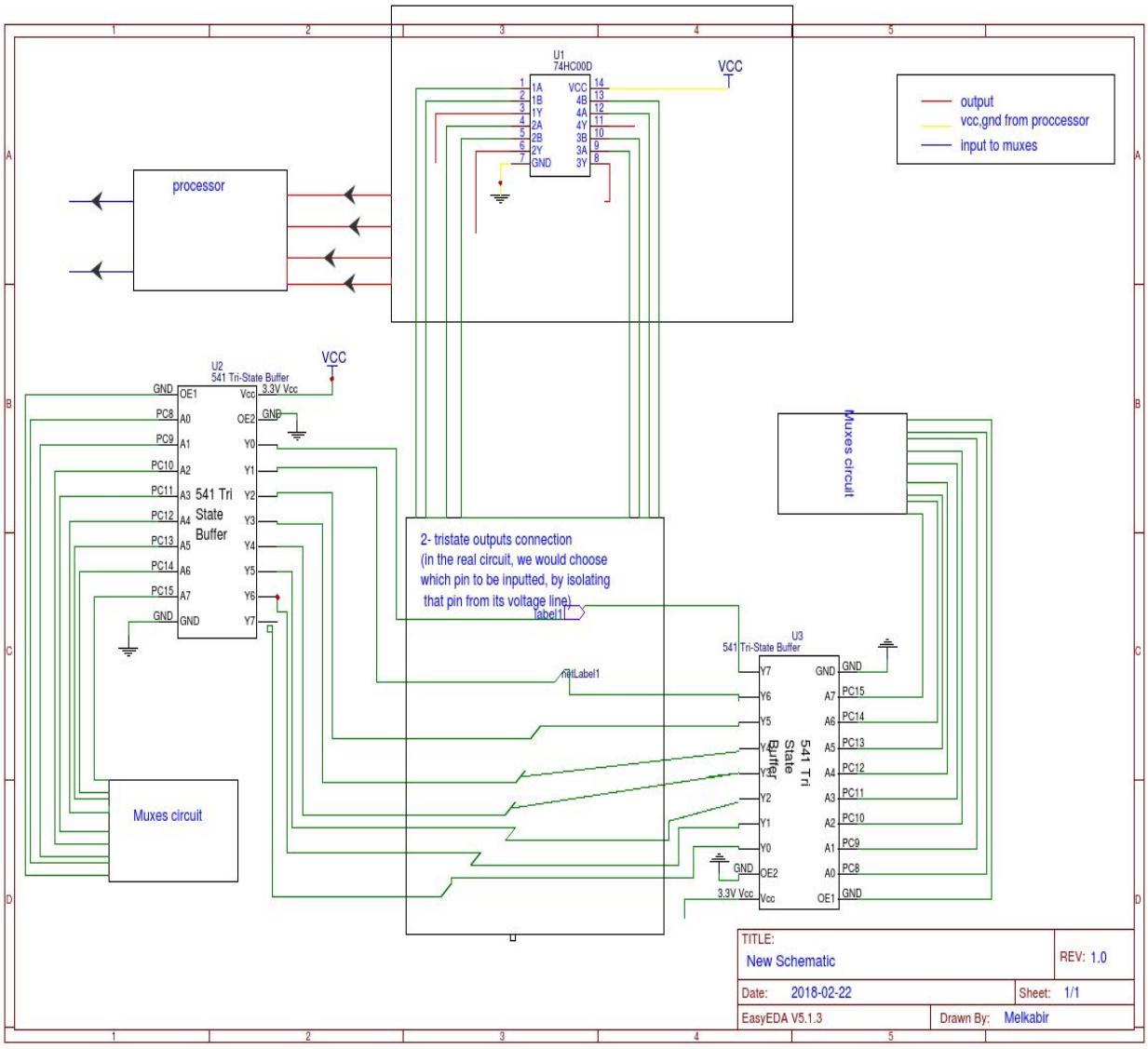


Processor-Logic Module:



Chip_tester module:

This schematic was implemented, using a NAND ship as a test ship, and used a 2-input bin to connect the 2-inputs from the different muxes, where the 2-Inputs for each pin represent 1) the current value in that row, and 2) the value assigned by the processor.



7805 +5V Voltage Regulator TO-220

For the 5V voltage regulator, it has a range of input voltages from 6.5 – 30V and a fixed output voltage of 5V. The dropout voltage for this regulator is 1.5V at 5A. The power consumed would thus be estimated to be 30mW.

Decoder:

We will be using a 74LS138 decoder. It will have a power dissipation of 32mw.It requires a minimum supply voltage of 4.75 and a max voltage of 5. 25V. For the DC characteristic, it has a minimum input high voltage of 2.0 and a max input low voltage of 0.8. We will need two decoders.

16:1 mux(74HC4067)

The supply voltage should typically be 5V with a minimum of 2V and a max of 10V.a power dissipation of 500mW. Three of these will be needed.

AtMega328

Operating voltage of 1.8-5.5V, Power consumption at 1.8V, Active mode:0.2mA, Power-down mode:0.1uA, Power-save Mode:0.75uA, Power consumption at active mode will be 3.6*10^-4W.

Thus the overall power needed for this circuit will be 0.59436W.

V. Module Description

Chip Testing Area of the Breadboard

Is the fifth or first pin of each breadboard row, which has five holes. This is the area we expect to place chips. We now restrict users of the breadboard not to use this chip-assigned hole for wiring. Thus, the user is now condemned to four active holes for wiring purposes per each row. As usual, columns of the breadboard are assumed to be assigned either Vcc or Gnd.

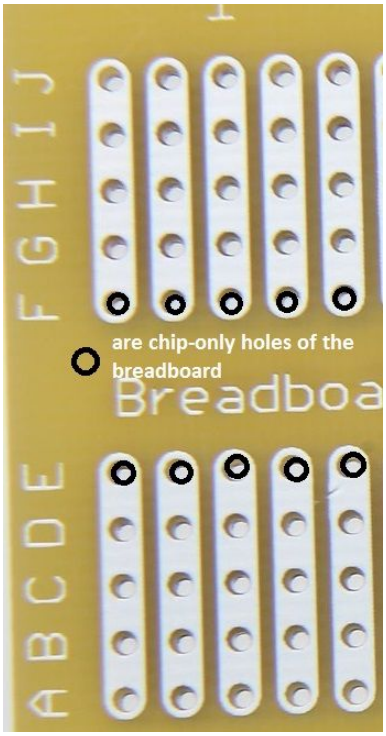
The chip testing area is separately considered from the breadboard module, because how the holes should be printed on the board are different. Each hole of the chipt testing area is connected to two tri-state buffers, instead of directly being connected to the rest of the circuit.

These tri-state buffers are what allow us not to assign what row must be input or output before the user gives the screen information about what row they are using as input or output by placing a chip.

We are only allowing maximum number of 6 input pins, which amount to 64 possible input configurations. This seems to be a lot to enter, even if this is the maximum number, and thus we will let the user to assign output, input, Vcc and Gnd rows directly, extending our minimal screen flowchart. Any unassigned rows/holes will be considered “Do not care,” and will simply be left in high-impedance mode. In practice, this reduces the number of input configurations to be considered, which means reduction of touchscreen interactions required for proper chip testing. Suppose a AND chip consists of 4 two-input AND gates. Then we only need to test for (2^2)*4=16 configurations.

The maximum number of pins of a single chip a chip tester can accomodate is defined to be 16.

Two tri-state buffers are needed for each hole where chips can be placed - one was described above, the other is necessary to disconnect user wires from the board without physically pulling off wires. User wires are thus connected to the input port of the tri-state buffer, with the control bit controlled by the processor.



Logic Module (Tri-state buffer module + demultiplexer module + mux module)

Demultiplexer modules provide necessary control bits of tri-state buffers and multiplexers. Control bits of multiplexers control which row output is read by the processor. Multiplexers are used to determine which row of the bread will be read by the processor. Demultiplexer also provides digital inputs to a chip being tested. The roles of tri-state buffers are already described in the description for the chip testing area.

The reason why the logic module is “big” relative to its operation is similar to the case when a computer deals with different devices. Just like how the processor is often restricted to dealing with one interrupt at a time, our processor also has similar limitation, but we need to have access to multiple rows and chips. We are devising a method that does not have to rely on a relatively complex concept of interrupts - thus our use of tri-state buffers, multiplexers and demultiplexers.

The logic module is externally invisible, as it is inside the smart breadboard casing.

Chip Testing-Logic Interface

For digital logic value checking of each row, user wire connections are required to affect our testing - obviously. For chip testing, however, user wire connections are required to be isolated away so that we can provide processor-provided input values to input pins of the chip. Tri-state buffers, as mentioned, are used to achieve this goal. (Though note that we extend our digital logic value checking to analog voltage checking by the use of analog multiplexers.)

Thus, even if a user uses all four holes of the row, or if a TTL circuit designed is filled with wires that testing is difficult, a user does not need to disconnect wires to test chips or to use a voltmeter to measure digital logic value. This is an important motivation for our project, along with increasing speed of debugging, so getting this right is very critical.

Breadboard Module

The breadboard module is the usual breadboard, except users now have four holes per row that can be used for wiring instead of five. But actual printed board-wise, each hole of the breadboard is also connected to the logic module inside the smart breadboard casing. The top of the smart breadboard has our breadboard module and the screen on the top-right, which are the only accessible parts of the smart breadboard along with power connection parts.

Screen Module

The screen module is the interface in which chip testing and digital logic value checking of each row are done. Every smart capacity starts with the screen module and ends with the screen output.

Processor Module

As mentioned in the logic module, our processor module has very limited number of input and output ports. Our processor used has clock speed of 20 MIPS. This will be more than sufficient to serve our purpose, as logic module is an asynchronous circuit, thus latency is mostly about signal propagation delay, which almost never is a problem.

VI. High-level Module Requirement

High-level module requirement is organized around the two options a user can use in the screen.

	Requirement	Validation
Chip Tester	The smart breadboard must be able to test correctly whether chips that have fixed input-output relationship work as specified by a user. In other words, all chips that are stateless must be testable against user specification of functioning of chips. User specification is simply required input-output pair, provided into the touchscreen module. Only a single chip is explored for each test.	At the full-functioning level (or high-level), we would partly be able to validate correct functioning of the chip testing feature by simply placing a chip in the breadboard module and providing a false specification through the screen at the start of the chip testing. For all false specifications of the chip, the screen module will warn the user that the chip functions incorrectly at the end of chip testing, but for the true specification, the screen module will print the message that the chip works as specified. If any

		deviation is seen, then there is flaw in implementation or design of our smart breadboard circuit.
		As we do not require users to move chips to a separate gate testing area to test chips, the smart breadboard needs to turn off effects wires connecting pins of a chip to other pins of the same chip or other chips to test individual chip. Whether the chip testing feature actually does this needs to be explicitly tested. In order to test one, we may build a slightly complex TTL circuit involving multiple chips and interconnected wires on the breadboard module and run chip testing procedures for each testing trial through the screen. For each trial, different specifications are provided so that what was mentioned in 1A gets tested as well. We will validate each module and its connections to other modules separately, but this will be presented when validating for each module.
Logic Value Printer	The smart breadboard must be able to display the logic value (0 or 1) of the row of the breadboard module that the user requested through the touchscreen. (Each row is supposed to share the same voltage.)	High-level validation of just the logic value printing feature of the smart breadboard is relatively easy: measure the pin voltage by voltmeter, and check what the screen says. Thus, much of validation lies in providing a complex circuit scenario so that how the smart breadboard finishes processing

		one feature and moves onto another user-requested feature can be fully checked. Ideally, it would be good if we can test all possible scenarios. This is infeasible, even simulation-wise. Thus we would have to rely on the benchmark circuit to test our features, just like how samples are used to perform statistical analysis.
For general validation		We may use SPICE simulation to validate our circuit, incorporating validation ideas in 1A, 1B and 2A, though actual operational validation would definitely have to be done, and this thus is a complementary validation.

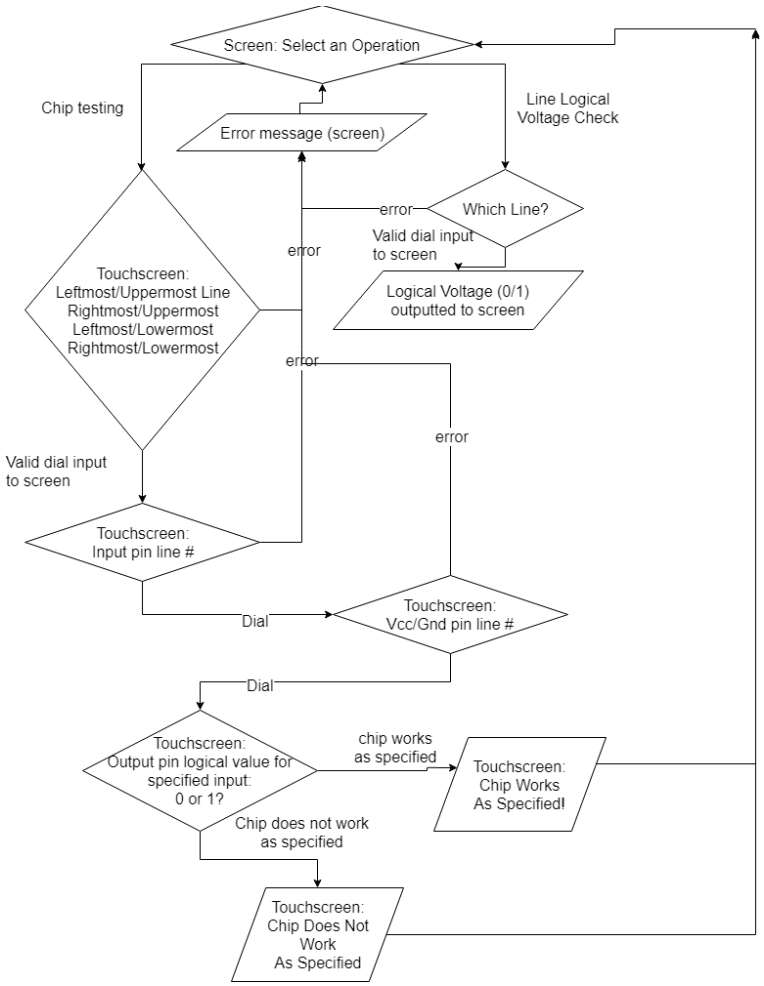
VII. Module Requirement

As our design is modular, we are performing modular tests by each module or module interface after or before checking high-level validation.

	Requirement	Validation
Screen-Processor module interface	The screen module only communicates with the processor module. Screen should print out the message the processor sent out.	Using voltage sources, provide an input signal to the screen module so that we can verify (screen output,input) pair as specification would require. (We will not be building screen module ourselves, so this depends on the choice of our screen.)
Processor-logic module interface	Consistency of (processor input, logic module output to actual	Using voltage sources and voltmeter, we can easily verify whether our module interface is

	pins for a chip) to our desired specification.	working correctly. Note that final logic module output is equivalent to voltage level of pins connected directly to chips.
Breadboard	Breadboard requirements are inevitably tied to the processor-logic module requirements. The below is thus user operational aspects. Allow user wires to be connected above the printed board using typical breadboard casing, while also connecting the same printed pin to the logic module. Printed board-wise without casing, there may not be any problem, but we need to verify that the user will be able to use the cased breadboard module without problems.	User experience validation, and technical validation is the same one in the processor-logic module validation.
	The chip-reserved pin should be connected directly to two tri-state buffers instead of being part of the same voltage row with other pins of the row.	Validation is done through a voltmeter and a voltage source connected to one of non-reserved pins of each row. Also Eagle visual checks.
Programming	Should assign correct logical values to control/select bits of multiplexers and tri-state buffers. Also must consider whether the signal the screen module can print off is being outputted to the screen module.	Programming checks using a test script.

Screen flowchart:



VIII. Hardware Complexity

Chip testing requires clever use of tri-state buffers, multiplexers and encoders. Otherwise, we would not be able to deal with required separation of inputs and outputs. Turning off effects of user wires for chip testing and then providing proper inputs for testing without pre-assigning location of input and output rows necessitates some complexity in the implementation of the project.

IX. Tolerance Analysis

The heart of our project is the processor. One important tolerance we have to maintain is timing. We want a fast computation rate. The number of pins is also important. We have to have a processor that can supply a number of select bits.

We have 5, 16:1 mux, 4 of them will have the same select bits; then one will have another 4 different bits. This makes it 8-select bits in total. That is the case when using the analog muxes to pass in the values. When we choose to do the gate testing part, we will make usage of the same number of muxes, plus two different tri-state buffer ships in this way we will have to provide 8-bits + 2 (1 for each tri-state buffer) = 10- bit select. This way we come to realise that we need a 10 pins in the processor that will only be used to assign select bits.

For testing purposes we will have a limitations of input to ships with 6 input pins. This way we can need 6 other pins and 3 for output. This summing up for 19 pins for I/O operations.
The number of pins can be tolerated to minimum of 10 + 2+1 = 13 pins for I/O operations , meaning we can manage to get the same result with 0.07 tolerance. The only cost in that is time. Since before we pass in input in a parallel way, but using 13 I/O pins, we will have to run each input pair by itself. Then do the next pair. We will also use muxes to locate the next input pins.

As for time constraints, the processor have a speed of 20(MIPS) ; that is each instruction takes 50 nsec to be implemented. Then we will have to consider the delay by each ship, where our muxes range in delay-rate from 32-54 nsec, but for our purpose we will make the calculations based on 54 nsec delay. Tri-state buffer have a delay rate of 15-30 nsec, similarly we will choose 30 nsec. With those information we can calculate how long it takes to provide 2 inputs for a chip.
50nsec *2(for assigning,2 values to the ship) + 54 nsec*4 (mux passing: 2*Input, VCC,GND) + 30nsec*2 (2 select bits for tri-state buffer)+54nsec(mux passing back the output) = 430nsec;
In this way we can see having 6 inputs, will take 2^6*time-delay= 27520nsec=27.5usec;
And 2^10 will take .44 msec. That is mainly the time the processor takes to run through all the configurations, then we will need to look through saving into memory, and comparing with lookup tables. Considering making 6 input pins a our limit, then this way let's say worst case scenario, the processor takes up to 5 sec for saving the outputs, and comparing outputs, then its still a very suitable rate for users. As for the time it takes for processing the output and comparing it, this will be determined based on the cpu speed, and then the memory size, since both will determine the time it takes for processing. The more lookup-tables we have, the longer it will take to run through them.
Due to those facts we see that atmega328 is a good fit for our project. It have a suitable memory space = 32KB, and considerably fast CPU speed.

X. Cost Analysis

We believe this project would be not that expensive to make.The fixed cost is estimated at $40/hour at 15 hours per week for three people.Shown below is our estimated incurred costs

Part	Manufacturer	Part#	Quantity	Cost
16:1 Analog mux	Texas Instruments	MUX506IPWR	10	$3.14
Tri-state buffer	Texas Instruments	SN74AHC126N	10	$0.329
Voltage regulator	ST microelectronics	L7805CV	10	$9.50
Breadboard			1	$5.69
AtMeg328	Microchip Technology	74HC4514	1	$2.01
LCD screen	Foxnovo		1	$17.49

The total cost will be $38.159.

Schedule

Week	Chinnies	Mostafa	Minseong
3/5/18	Ordering of parts	Making the processor circuit	Taking care of tri-state buffer and demultiplexer circuit
3/12/18	Assemble parts	Taking care of the muxes circuit	Write programs inside the processor
3/19/18	Spring Break	Spring Break	Spring Break
3/26/18	Wiring of the socket	Testing area circuit	Writing test scripts for programs
4/2/18	Making the design layout of the breadboard	debugging + measurements	With inputs from Mostafa: check connections of the mux module with the rest of the logic module. And test the assembled design. Pass the program to Chinnies for final testing of the program in the processor
4/9/18	Program the processor	Making look-up tables + screen commands	User interface checks. Physical casing checks.
4/16/18	Finalizing our project		
4/24/18	Begin final report	Prepare final presentation	Prepare Final presentation

XI. Safety Statements

There is a number of safety issues, firstly the battery could 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.

Our design is considerably safe as it uses a safe range of voltage and current. Our projects runs on a 5v, processor, and 6.5 V screen. One ethical issue might rise is that students become dependent on this project, and do not follow other methods of testing. Students may use our project as the main testing tool without being supervised by a professional. This might cause harm for young children and new users as well. In the times of huge amount of computation is done, the processor, or any of the parts might be damaged, or even burned in the process. That will result in sometime a chemical reaction and smoke might rise. The voltage regulator have a high rate of watts being dissipated, which it needs a heat sink. That much power could be dangerous for young children when heat sink fails.

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].