Lab 05: Introduction to LabView

EG1003 Section G1

Alexander Lopez-Matos

Lab Partner: Kevin

Date of Experiment: October 6, 2014

Date due: October 19, 2014

**Abstract**

The objective of this lab “Digital Logic” was to engineer a “combination logic circuit” that would switch on a LED light when certain conditions are met using LabView and the NI-ELVIS prototype board. The result of this lab was a functioning combination logic equation that was successfully implemented on both software and hardware.

Combination logic is the foundation for all advanced hardware that uses circuitry. Stop lights, cell phones, airplanes, and computers are just a few examples of physical devices that use digital logic. The implications of this lab are important for all forms of future hardware that needs to interact with software and use logic to accomplish real world tasks.

**Introduction**

At the heart of all digital circuitry is Boolean logic. The mathematician George Boole created the symbols we currently use while representing Boolean logic. Within Boolean logic, there are two values, or outputs, true and false or one and zero. These values can be thought of as “on” and “off”. These 1 and 0 inputs are used along side logic gates to carry out Boolean logic. Logic gates are input and output devices that take in one or more inputs and give back one output. There are seven different types of logic gates: NOT, AND, OR, NAND, NOR, XOR, and XNOR. Only NOT, AND, and OR gates are used for this lab.

The first of the gates used in this lab is the NOT gate is given an input and returns the opposite of that input. If a 0 is given to a NOT gate, a 1 would be returned. This process is called “inverting” the input. The second gate used is the AND gate. The AND gate takes in two inputs and returns one output. If the first input and the second input are both true or 1, the AND gate will return true. If either the first or the second input is false, the AND gate will return false, or 0. The third gate used is the OR gate. The or gate takes in two inputs similar to the AND gate, but returns true if either inputs are true. If an OR gate was given 0 and 1 for inputs, it would return 1 because the second input is true.

Boolean logic is formed by creating a “truth table” like the table shown in Figure 1 which shows all possible input and output combinations. From a truth table, a Boolean equation can be generated as show in Figure 2. The Boolean equation is created by writing out all input that generate a 1 value for output. Any values that need to be inverted using a NOT gate are represented with a bar line above their respective character like so: This Boolean equation can then be simplified using Karnaugh map (K map) shown in Figure 3. With this simplified Boolean equation, a logic circuit can be implemented, such as the one shown in Figure 5.

In this lab, a digital logic circuit needed to be created to protect a farmers hens from foxes and his corn from his hens. There are two barns that need to be monitored, barn one and barn two which are represented as 1 and 0 respectively. When a fox and hen are in the same barn, an alarm needs to be set off. When a hen is in the same barn with corn, an alarm needs to be set off.

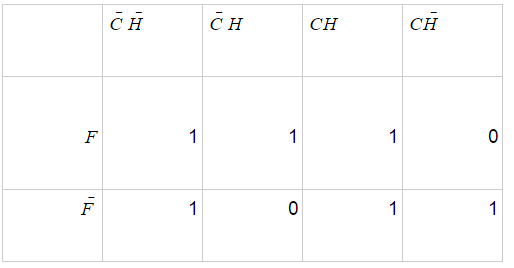
**Figure 1 - Truth Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Corn** | **Hen** | **Fox** | **Result** |
| **0** | **0** | **0** | **1** |
| **1** | **0** | **0** | **1** |
| **1** | **1** | **0** | **1** |
| **1** | **1** | **1** | **1** |
| **0** | **1** | **1** | **1** |
| **0** | **0** | **1** | **1** |
| **0** | **1** | **0** | **0** |
| **1** | **0** | **1** | **0** |

**Figure 2 - Boolean equation**

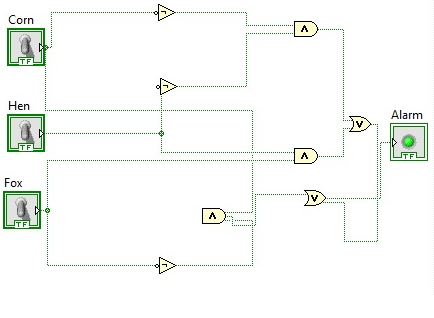


**Figure 3 - Karnaugh Map**



**Figure 4 - Simplified Boolean Equation**



**Figure 5 - Logic Circuit**

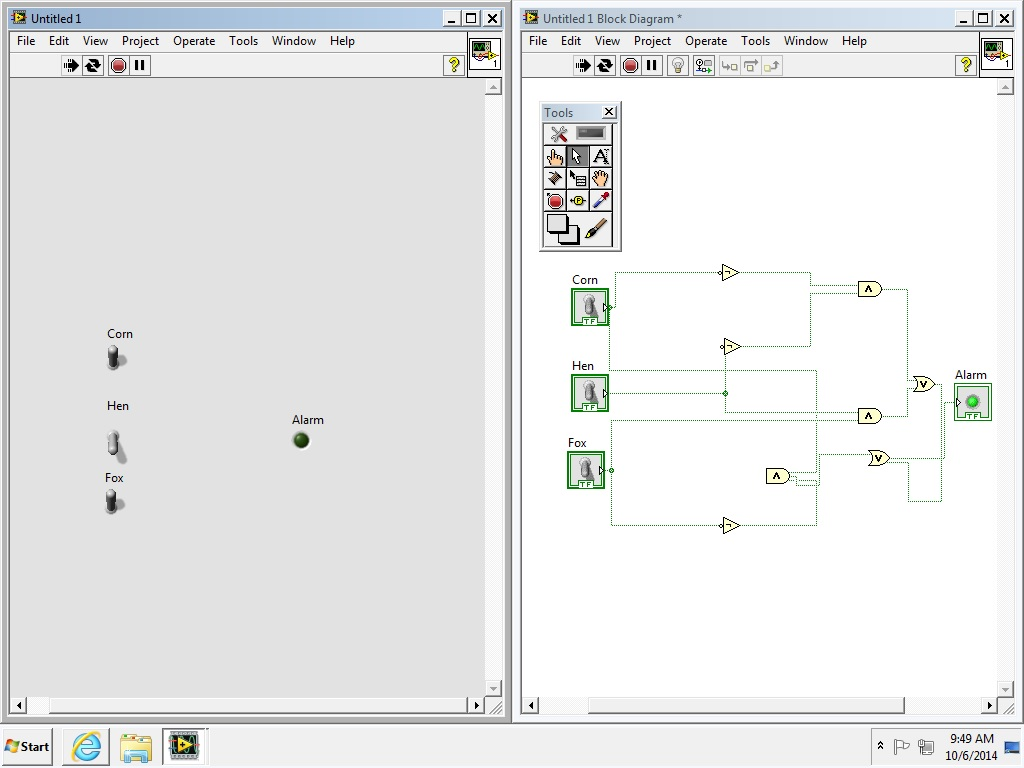
**Procedure**

A truth table was constructed listing all possible combinations of fox, hen, and corn in the two barns (Figure 1). The output column was generated by determining if a fox and hen were in the same barn, or if a hen is in the same barn as some corn. If hen and corn are both 0, then we know that they are both in Barn 2 and the alarm needs to be set off. Likewise if fox and hen are both 1, then we know there is a fox where the hen is and the alarm again needs to be set off.

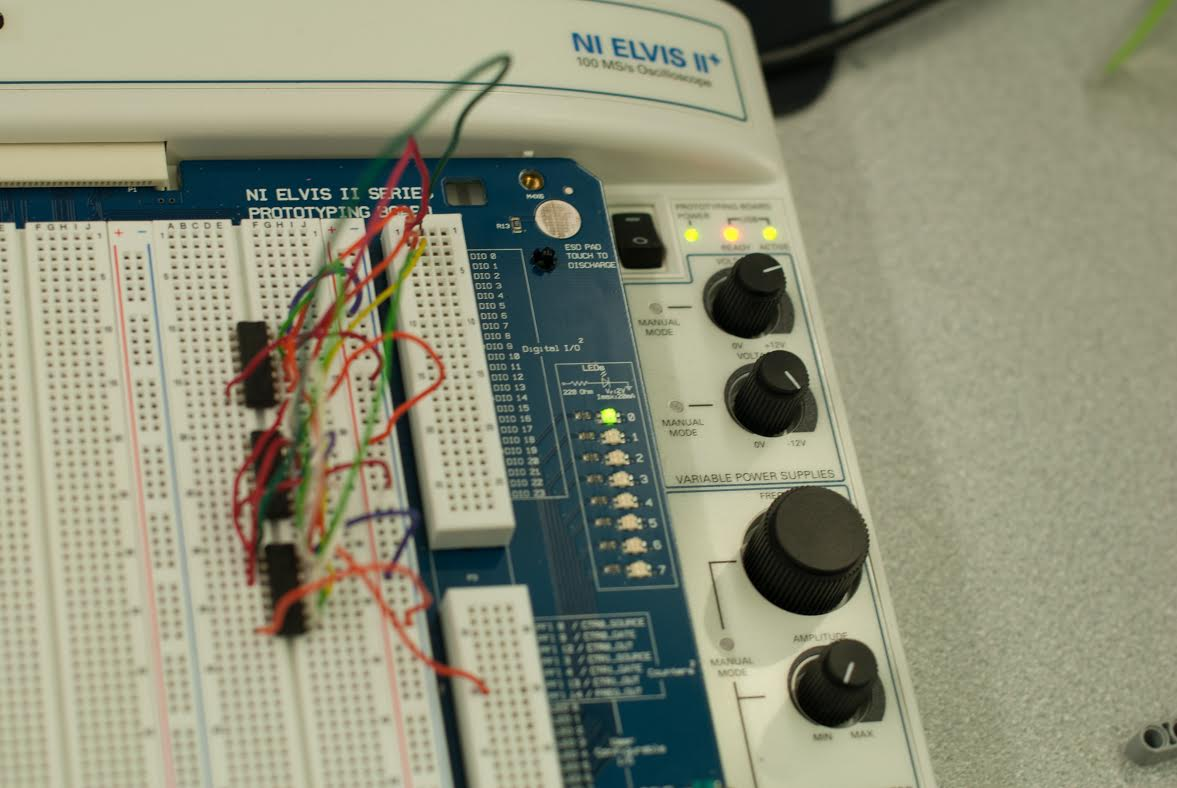
All combinations of fox, hen, and corn that produced an output of 1, meaning the alarm needed to be set off, were used to create the Boolean equation shown in Figure 2. Using this Boolean equation, the Karnaugh Map shown in figure 3 was created to simplify the equation. On the Karnaugh Map, pairs of output value 1 were grouped together and used to create the simplified Boolean equation shown in Figure 4. With this simplified Boolean equation, the logic circuit in Figure 5 was sketched and then approved by the Lab TA.

Once the sketch of the logic circuit was approved, the schematic was created in LabView and tested using a front panel with 3 switches and a LED light to represent an alarm as shown in Figure 6. When the LabView circuit board worked successfully, the circuitry was then built on the NI-ELVIS Board with wires and NOT, OR, and AND gates. Power was drawn from the bottom right column and input was received from the computer through the top right column.

**Figure 6 - LabView Application**



**Figure 7 - NI-ELVIS Board Implementation**



**Data/Observations**

While finding the Boolean equation to solve the problem in this lab, the truth table in Figure 1 represented the foundation for the rest of the problem. The concept of using NOT gates was clarified during the creation of the truth table because when a fox and hen are both in barn two, their values are both 0. Since an output of 1 is required to set off the alarm in this scenario, the values of the fox and hen need to be routed through a NOT gate to convert both values to 1 before being given to an AND gate. Since an AND gate takes in two inputs and returns 1 if and only if both inputs are 1, the conversion of the zeroes to one’s is a necessary step to set off the alarm.

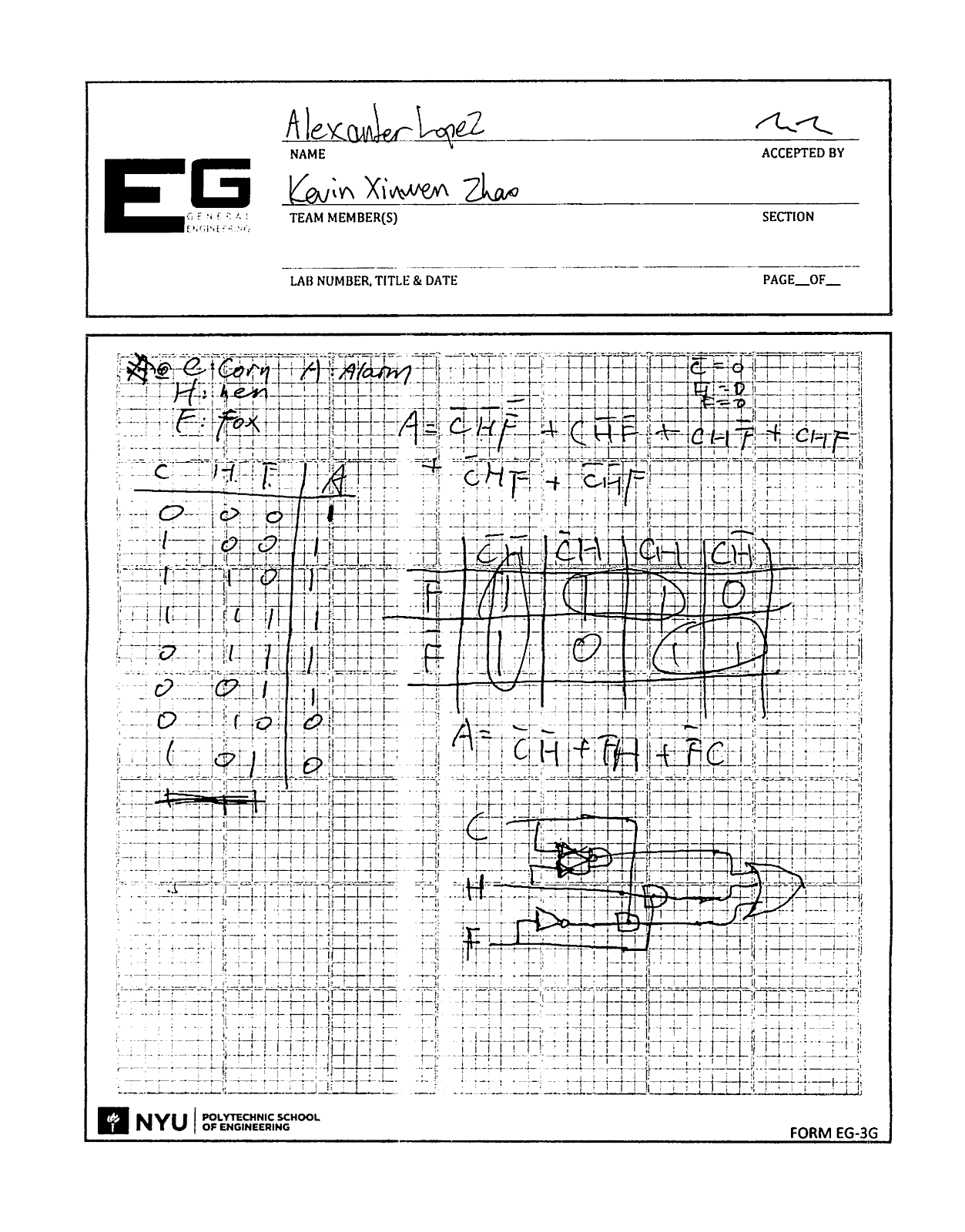
The final combinational logic board was created using wires and gates, and was modeled after the LabView back panel logic. Input was received from the computer and through wires shown in Figure 7 on the top right. These wires were then routed to either AND gates directly or to NOT gates if the input needed to be inverted. If an input was inverted, it would then be routed to an AND gate and them from there to the final step which was an OR gate. The OR gate connected directly to the LED light and would cause the light to appear when either inputs were equal to 1.

**Discussion/Conclusions**

Simplification of the Boolean equation through the use of a Karnaugh map shown in Figure 3 significantly reduces the complexity of the circuit board, and is a necessary step to creating a efficient and effective circuit board implementation. If a Boolean equation were not simplified before it was implemented on a circuit board, the circuitry would quickly become a confusing mess and would not only waste time but waste costly materials in a real world scenario.

If one barn used a bell and the other used a horn, the inputs that are inverted would need to go to one output, and the non inverted inputs would go to another.

The experiment was a success. The concepts of Truth Tables, Boolean equations, K maps, and logic circuits are fundamental aspects of creating any technology that interfaces with the real world. This base knowledge is necessary in order to make advances on the technology of today.



Work Cited

"EG Manual" *EG1003*. N.p., n.d. Web. 28 Sept. 2014. (manual.eg.poly.edu/index.php/Main\_Page)