Spring, 2013

Luminous Chessboard

ECE445 – FINAL REPORT

TA: MUSTAFA MUKADAM

QIANLIANG LIU, KE MA

UNIVERSITY OF ILLINOIS AT URBANA - CHAMPAIGN | ECE

Abstract

We tried to create a luminous chessboard with simple artificial intelligent built inside in order to provide a fun way for people who have never played chess before to learn the basic rules. The LED built inside provides a direct and beautiful instruction for the beginners especially for children.

The project is mainly built on a chessboard size PCB board. With the PIC microcontroller controlling the signals through the circuits on the PCB, the chessboard will be able to light up any desired LEDs on the chessboard. By turning on LEDs on the different locations on the chessboard corresponding to the different chess piece that users picked up, the users can see the allowable moves for the piece that they pick up.

Although we successfully made each individual module works independently, we failed making our project works on the PCB board. With limited time and resource, we won't be able to fix our project completely.

The cost is higher than we thought, since we had unexpected problems on programming PIC chip.

Table of Contents

1. Introduction

We tried to create a luminous chessboard with simple artificial intelligent built inside in order to provide a fun way for people who have never played chess before to learn the basic rules. The LED built inside provides a direct and beautiful instruction for the beginners especially for

children. Using 4-pin sockets from service shop, we made 32 chess pieces and the locations to hold them on the chessboard. The chessboard is made by an 11.5*12 inch PCB board, so we can connect the LED circuit directly on the chessboard.

1.1 Purpose

Chess is a popular board game with a long history for which the chessboard and pieces are usually made of wood or plastics. These years with the fast development of the techniques of mobile computer, chess is often played on a touch screen without real chessman pieces. However, we believe the feeling of moving a tangible chessman piece is irreplaceable. So that we are willing to make efforts to combine the benefits of the two by designing a chessboard computer with real board and pieces which can "see" what is happening and can give computed feedback.

The goal of our project is to design an electronic chessboard that can recognize chessman pieces and is equipped with a lighting system showing potential moves with an AI algorithm so that beginners can learn the rules with more fun.

1.2 Functions

The chessboard will be able to recognize every chess piece. Instead of memorizing the locations of each piece, the microcontroller is designed to recognize the unique signal feedbacks from different chess pieces. Therefore, even if pieces are not placed as the beginning positions, the AI still will be able to give instructions based on current situation on the chessboard.

Once a piece is picked up, the chessboard should light up the available positions it can go, and indicate good moves and bad moves with different colors of LEDs, and afterwards detect the move the player conducts. The AI algorithm will be simple and be able to look one or two steps ahead.

1.3 Subprojects

1.3.1 Detection module

This module is a simple circuit that made of 4-pin sockets, eight 16:1 mux and a 4:1 dual mux. The purpose of this module is to scan the whole chessboard by receiving the signals from the microcontroller and then feedback to the microcontroller through the 4-pin socket.

1.3.2 LED module/lighting module

This module is used to receiving the signals from the microcontroller and turn on the desired LEDs.

1.3.3 Microcontroller module

The PIC needs to send out 2 identical signals with 180 phase difference to the chessboard, and read the feedbacks in order to understand the situation on the chessboard. After receiving the feedbacks, the arterial intelligent will identify the piece that got picked up and send the signals to turn on the LEDs that corresponding to the allowable moves of the piece.

2. Design

2.1 General design alternatives

Since the service shop has limited size of PCB, we have to use a much smaller PCB. Therefore, we weren't able to use LEDs to show users the bad moves. Our alternative design can only show users the allowable moves. Although some details have been changed, the design still contains three modules as the same as the original design. Figure A1 and A2 are used to present the overall block diagram.

2.2. Overall Summary

1) Detection Module

This module connects the positions with or without the pieces on to the processor. The pieces are plugged on the chessboard as black boxes.

a. Chessman Piece

A chessman has a circuit which may consist of wires and diodes that can transfer the input signals to the output port to be detected.

b. MUX

Since we cannot receive all the 128 feedback signals, namely, 64 pairs of signals, at the same time, MUXs are used to focus on one specific position.

2) Lighting Module

This model lights up the chessboard to indicate allowable moves which is the main function of the design.

a. LEDs

One of the main features of the chessboard is to indicate good and bad moves with lighting system. The feature may be implemented by installing 64 LEDs under the squares of the chessboard.

b. Decoder and NMOS

The micro-controller output digital signals, while the LEDs are basically operated with currents. This model converts binary control bits to determine which LED to be turned on. It can also limit the analog current in the LED's operation range.

3) Micro-controller

It is in charge of recognizing what is happening on each square with received signals, and choosing which position to be light up with what color. The AI will be programmed in C language.

2.3 Detail design descriptions

2.3.1 Detection Module

	Inputs	Ouptuts
Pieces	input1 = 10 and an input2 = 01	identifiable signals ouput1 = ** and output2 = **
Positions	input1 = 10 and an input2 = 01	Empty position output1 = 00 and ouput2 = 00 Occupied position identifiable signals from the piece.
MUX	64 pairs of output signals from all the squares; 4 select bits to the 16-1 MUX 2 select bits to the 4-1 MUX	Output signals from the chosen position

Figure.1

On every position of the chessboard, we put a 4-pin socket on it. Two of the pins are connected to the micro-controller and receives two digital signals, namely two voltages. The other two pins are also connected to the micro-controller as the chessboard outputs. In every loop, the micro-controller gives input1 = 1 and input2 =0 in the first half cycle, and receive a pair of outputs. Then it reverses the input signals to be input1 = 0 and input2 = 1 and receive another pair of outputs. With the 2 pairs of output signals, we have 4 digital bits to make up 16 combinations, covering the 12 chessman type and 1 empty case.

Originally, there are no wire connections among the four pins, so the outputs of an empty position are always zero, output1 = 00, output2 = 00.

As discussed above, a chessman piece has the same 4-pin socket inside it so that it can be plugged onto the board squares. Inside the chessman, we build a circuit which may consist of wires and diodes that can transfer the input voltages to the output port to be detected. Different circuits indicate different types of piece. These pieces act as “black boxes” for the board to recognize.

Figure.2

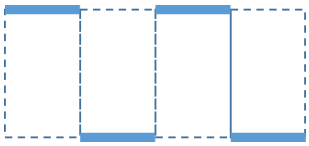


Figure.3 Input 1 Pulse

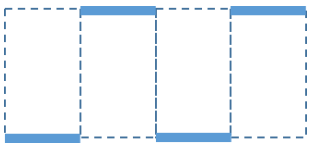


Figure.4 Input 2 Pulse

In the figures above, we use pulses to represents the changing inputs.

The design of all chess pieces and their unique feedbacks are presented at Appendix B Figure.B1.

Based on the number of squares on a chessboard, we know that there are 2 feedback outputs for each square, 128 in total.

2 outputs/square * 64 squares = 128 outputs

We will use 4 16-to-1 MUXs (MC14067BCP) in parallel to reduce the 64 lines of ouputs1 into 4 lines, and do the same thing to get 4 output2 lines. Then we use a dual 4:1 MUX to select one signal for output1 and one signal for output2. By switching the select bits of 16:1 MUX and dual 4:1 MUX, we can get feedbacks from all the squares on the chessboard.

The detection module circuit can be divided into two parts, the sending part (Figure.B3) and the receiving part (Figure.B4). Due to the size of the figure, the figures of both parts can be found in Appendix B.

2.3.2 Lighting Module

	Inputs	Ouptuts
LEDs	Currents	N/A
Decoder	Power supply 2 bits input to enable two rows	5V from the power supply to the chosen two rows. 0V to the other six rows.
MOSFETs	5V / 0V voltage on Drain Digital signals (4V / 0V) on Gate Source is grounded	If Vdrain = 5V and Vgate = 4V, the MOSFET will allow operation current to pass thought. Otherwise, no current will pass.

The LEDs we use are green LEDs (HLMP3507).

The main circuit is a simple combination of demux, LED and nmos.

The demux will get a 5V DC input from the power source and output to different rows by receiving different select bits from the microcontroller. Every outputs of the demux are connected with LEDs and the drain of the nmos in series. Therefore in order to turn on a single LED, the nmos also needs to receive a high gate voltage which will be provided by the microcontroller. More details about the LED circuits on the chessboard can be found in Appendix B, figure.B5.

Each LED has a current that is controlled by 3 bits, two row bits and one column bit. In order to avoid burning the LEDs, we regulate that the current through each LED cannot exceed 25mA.

From the data sheet of nmos[5], it says the drain voltage will be approximately 4V.

By the equation: $V_D=V_{DD}-I_D\times R_D$ equation (2.3.2.a)

And the values we know: $V_{DD}=5V$, $I_D=20mA$, $V_D\approx 4V$

We got that $R_D=200\Omega$

Schematic Figure.B6

Simulation Figure.B7

2.3.3 Micro-controller

Detection functions

Software input: void - a call

Software output: int board[8][8] - an 8*8 array corresponding to the 8*8 positions, with information in it.

Hardware inputs: Read output1 from RA7, output2 from RA6.

Hardware output: RA2:5 as 16-1 MUX select bits, RE0:1 as 4-1 MUX select bits. RA0 as input1, RA1 as input2. Loop of RA0:1 = 10; RA0:1 = 01.

When AI functions calls the detection functions, the detection functions will make the micro-controller give select bits to choose a position. Then it runs a detection cycle. Set RA0:1 = 10, read RA6:7 = BA, set RA0:1 = 01, read RA6:7 = DC. With the four bits of ABCD, the micro-controller is able to know the piece type.

Then it moves on to another position by changing the select bits.

After it detects all the positions, the function will return an 8*8 array with different number in it indicating different chessman.

LED functions

Software input: int1 result - LED on or off; int x y, -at position (x,y)

Software output: void

Hardware inputs: N/A

Hardware outputs: RB0:7 as MOSFETs gate voltages in even rows, RC0:7 as MOSFETs gate voltages in odd rows, RD0:1 as select bits to decoder.

When AI decides to turn on of off an LED, it tells the LED functions of its position x, y, and an intl result = 1 as ON, 0 as OFF.
The LED functions will set the MOSFETs pins and the decoder pins according to the command.

Main AI functions
Software input: Board[8][8] from the detection functions
Software output: result, x, y to the LED functions
Hardware input: N/A
Hardware output: N/A
This is the brain of the program.
It will recognize the action of the players, and give suggestions by lighting up the LEDs. It can handle error and regret for one step.

Flow chart of AI algorithm: Figure.B8
Pin assignments: Figure.B9

The chessboard schematic: Figire.B10

3. Design Verification

Our testing doesn't require many equations or calculation. The detection module is only a variation of wiring so that different feedbacks will be received by the microcontroller. For the microcontroller module, the testing is mainly operated on computer. And for the LED module, the bright LEDs at desired position is quite a direct success. For details, refer to Appendix D.

3.1 Testing

3.1.1 Detection module

We soldered a 4-pin socket with 4 wires and then connect it on the breadboard. We applied a 5V Pk-Pk AC voltage to one of four pins, and used an inverter to invert the AC source and connected the output of the inverter to another pin.

Next, we use connected the remaining two pins to the two different channels of oscilloscope. We also connect AC input to the third channel of oscilloscope for comparing. Next, we put the chess pieces which also made from 4-pin sockets on the socket on the breadboard then observe the results on the oscilloscope and see if the results match our predictions.

Results: Appendix C, Figure C1 to C7

From the results, we are pleased that our detection module worked independently.

However, the detection module failed on the PCB board, since we didn't put any capacitors for decoupling. Therefore, the voltage is highly unstable. The max voltage of feedbacks we can get is only about 2.3V, which is much lower than what we wanted.

3.1.2 LED module

First, we built a single LED circuit on the breadboard, and check if an AC power supply could work on nmos. We provide a 5V DC to the anode of the LED and 4V Pk-Pk 2Hz AC supply to the gate of nmos and check if the LED is flashing. After the LED flashed, we increased the frequency up to 200Hz until the LED looks like constant on. Moreover, we found that 200Ω is too large for the LED circuit. Therefore, we changed to 56Ω so that LED looks brighter.

Then we built a 2-by-2 size of LED circuit, and check if switching the gate voltage of nmos and the select bits of demux could change the turned on LED position.

Both circuits worked. We have verified that LED module could work independently. But on the PCB board the LED circuits failed because of two factors. We didn't put any capacitors on the PCB for decoupling so that the voltage on the PCB dropped rapidly and the EAGLE model of the demux was wrongly picked so that the DC supply cannot get to the anodes of LEDs

3.1.3 Microcontroller module

We changed the program so that we can play chess on the computer by entering proper inputs. The simulation of the chessboard works fine on the computer. (Refer to Figure. C10)

In practice, the PIC starter is hard to use, we burned our PIC chip and failed importing program into it. However, we made a simple detection program to run on the PIC16F877A. We set the two output pins and the two input pins as discussed in the detection module design section, and connect for LEDs to show the detection result. The first two show the output1 as input changes from 10 to 01, the last two show the output2. For instance, a piece with parallel wires connecting the two input pins to the output pins, will result in the four LEDs to be ON, OFF, OFF, ON, indicating the output1 is 10, identical to input1, and output2 connected to input2, is 01.

4. Cost

For this project, we mostly used the parts from service shop, which are free. Since our circuits use only simple components. We will analyze the cost of parts based on the price list on the DIGIKEY.

4.1 Labor

For this project, we spent much more hours every week than we thought. Based on our hourly payment, the total labor cost will be analyzed. Since the soldering and designing PCB board took lots of time, we spent average 20 hours per week after spring break and 10 hours per week before the break.

The total hours we spent can be estimate by the formula:

$$\text{Total hours} = 20\text{hours/week} \times 5 \text{ weeks} + 10\text{hours/week} \times 5 \text{ weeks} = 150\text{hours} \quad (5.1.a)$$

By using the formula:

$$\text{Labor Cost} = \text{Ideal Hourly Salary} \times \text{Hours Spent} \times 2.5 \quad (5.1.b)$$

Members	Payment/hour	Hours spent	#weeks	Total (2.5)
Qianliang Liu	$30	150	10	$11250
Ke Ma	$30	150	10	$11250

4. 2 Parts

For the parts, we didn't purchase any expensive devices for our project, since our circuit is not very complex. However, to build a chessboard takes quite amount of parts to make all the circuits on the chessboard, which is the main reason that our parts cost is high (refer to figure.E1). Despite the parts we broke during the tests and modifying, we still need approximately $78.71 to build this project.

4.3 Total cost

By using the equation:

$$\text{Total cost} = \text{Labor cost} + \text{parts cost} = \$12500 + \$78.71 = \$12578.71$$

5. Conclusion

Overall, we have successfully created all three modules we need and made them work perfectly independently. The detection and LED modules were completely functional on the breadboard. And we can even simulate the whole chess game on the computer with the program we wrote.

However, our project highly relies on the performances of the PCB board and the PIC chip, which we failed on. Because of the size of the PCB board and the huge number of IC chips we put on it, there is a large impendence on the PCB board. Since we didn't have any decoupling experience, we weren't able to properly fix our PCB board.

There are also other uncertainties in this project. For example, the PIC chip is really fragile and requires complex connection to import the program. With little experience, we accidently burned our chip and failed importing the program. Moreover, we didn't expect that the PIC starter cannot support our chips, which caused us to change the chip at the last few weeks.

At the beginning of the semester, we were warned that there are several similar products on the market. But we can guarantee that our design is purely original, and it can even do the things that none of those similar products can do, like recognizing the chess piece.

For the future work, we might consider ordering a much better PCB board outside the campus, and changed our microcontroller to Arduino. At first, we thought that PIC can easily accomplish our goal, since it recognizes C language and our algorithm is simple. Furthermore, it's cheaper than Arduino. Now, we realize that Arduino is easier to use, though it has its own language. Also, with the program kit, the Arduino only cost $20, which is much cheaper than the PIC program kit. Also, we will need help from ECE machine shop, because our project (refer to Figure.G1) is not very good looking. To make out project looks more like an awesome luminous chessboard, we will need ECE machine shop to help us build a better shell.

6. Reference

[1] IEEE Code of Ethics [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html

[2] T-13/4 (5 mm) Diffused LED Lamps Data Sheet, Agilent Technologies, no date, Web

[3] PIC18F4685 Data Sheet, Microchip Technology Incorporated, 2009, Web

[4] MC14067B Analog Multiplexers /Demultiplexers data sheet, Semiconductor Components Industries, LLC, 2011,June, 2011 − Rev. 7 web

[5] IRF520 power mosfet data sheet, Vishay Siliconix, Rev. B, 21-Mar-11, web

[6] CD74HCT139 decoder data sheet, Texas Instrument, September 1997 - Revised October 2003, web

[7]Sn74LS153N Dual 4:1 selector/ MUX data sheet, Texas Instrument, Dec 1972- Revised May 2007, web.

Appendix A Block diagram

Figure.A1to A3 are the block diagrams of our project. We added more details toward the progress of our project

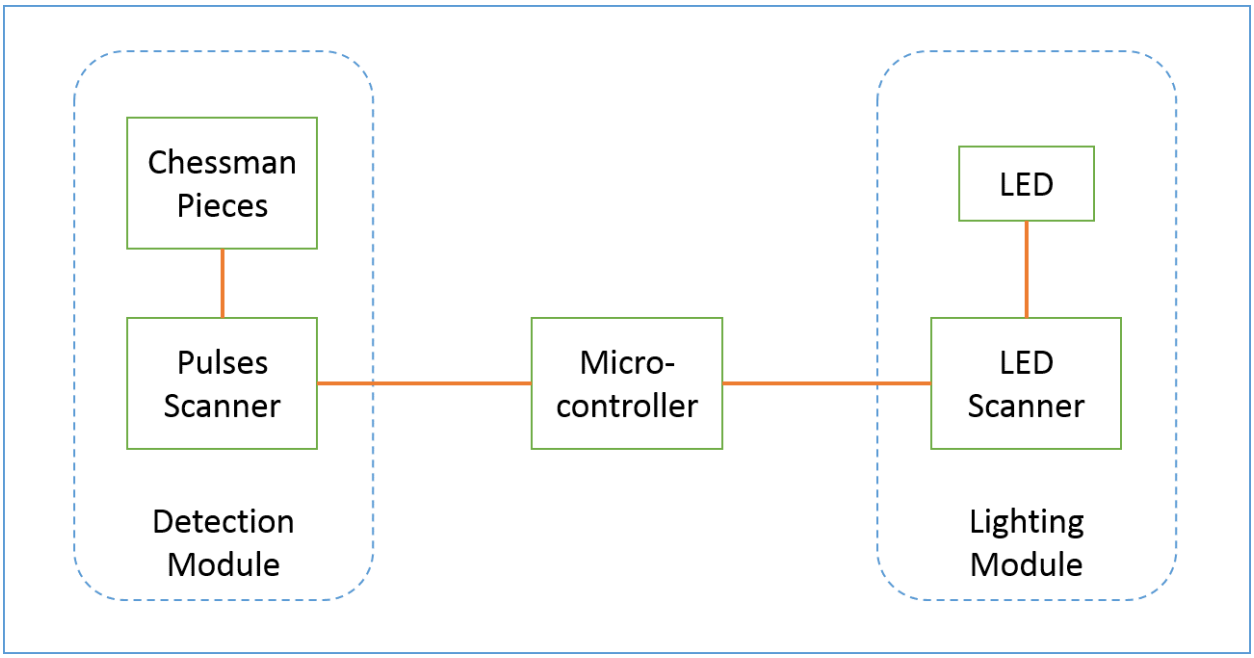


Figure.A1

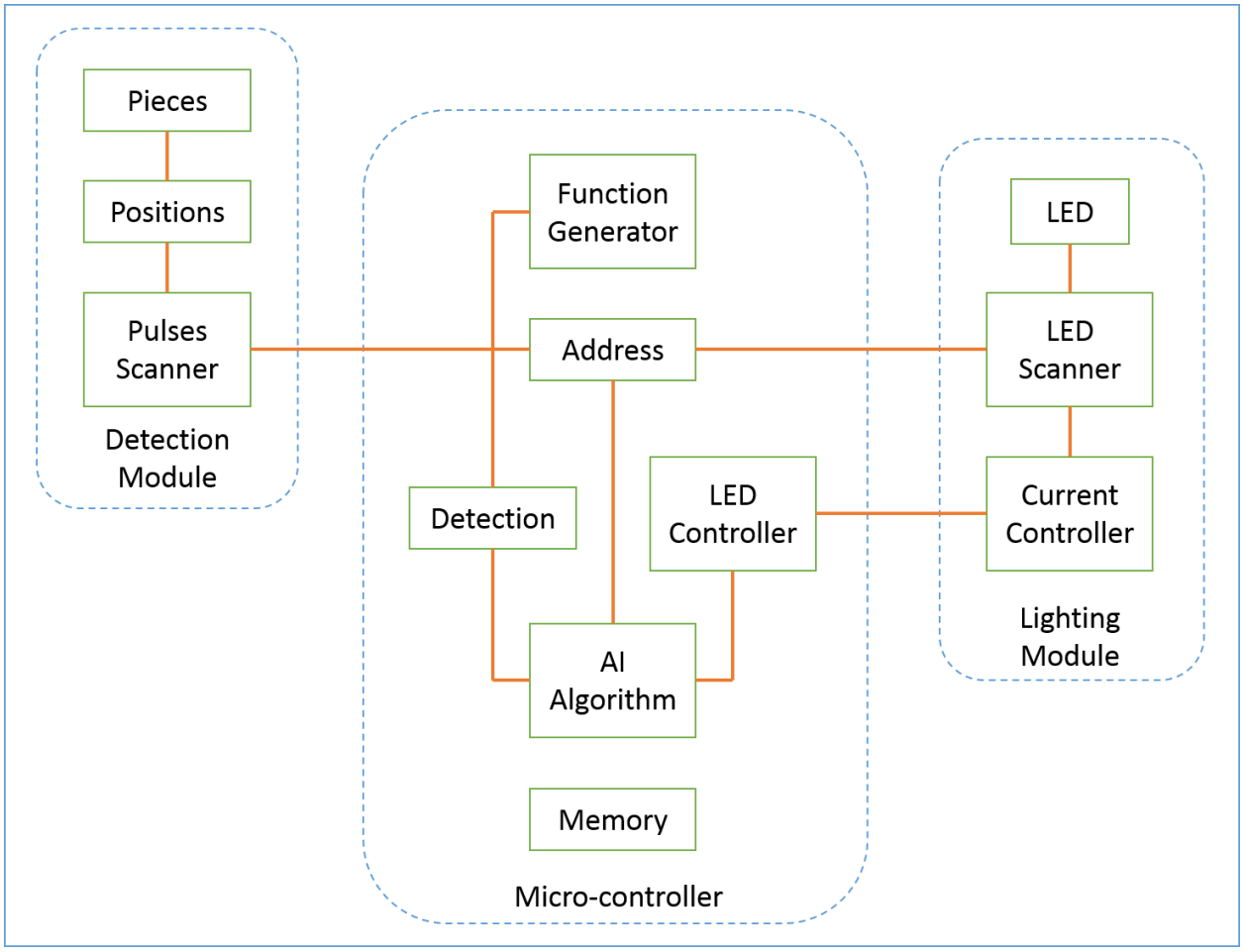

Pieces
Positions
Pulses Scanner
Detection Module
Function Generator
Address
Detection
LED Controller
AI Algorithm
Memory
Micro-controller
LED
LED Scanner
Current Controller
Lighting Module

Figure.A2

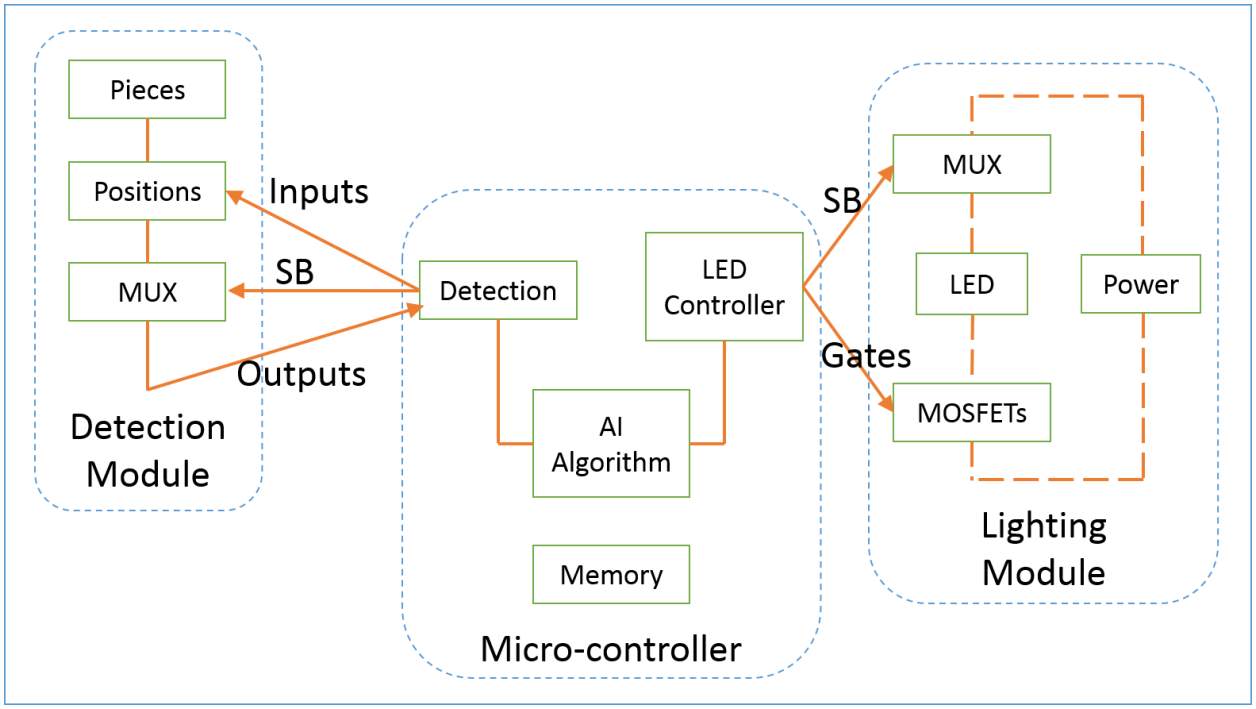
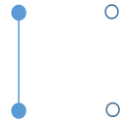
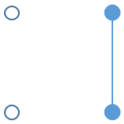
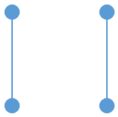
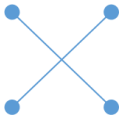
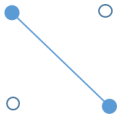

Pieces
Positions
MUX
Inputs
SB
Outputs
Detection Module
Detection
LED Controller
AI Algorithm
Memory
Micro-controller
SB
Gates
MUX
LED
Power
MOSFETs
Lighting Module

Figure.A3

Appendix B Schematic/ Design

Chessman Type	Connections	Outputs
Pawn (Side A)		1.  2. 
Pawn (Side B)		1.  2. 
Rook (Side A)		1.  2. 
Rook (Side B)		1.  2. 
Bishop (Side A)		1.  2. 
Bishop (Side B)		1.  2. 
Knight (Side A)		1.  2. 

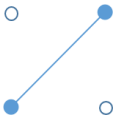
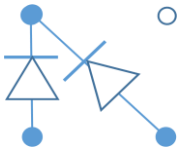
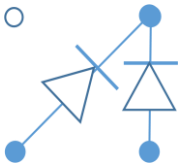
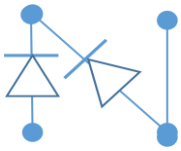
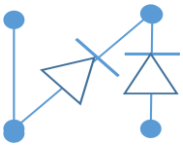
Knight (Side B)		1.  2. 
King (Side A)		1.  2. 
King (Side B)		1.  2. 
Queen (Side A)		1.  2. 
Queen (Side B)		1.  2. 

Figure.B1

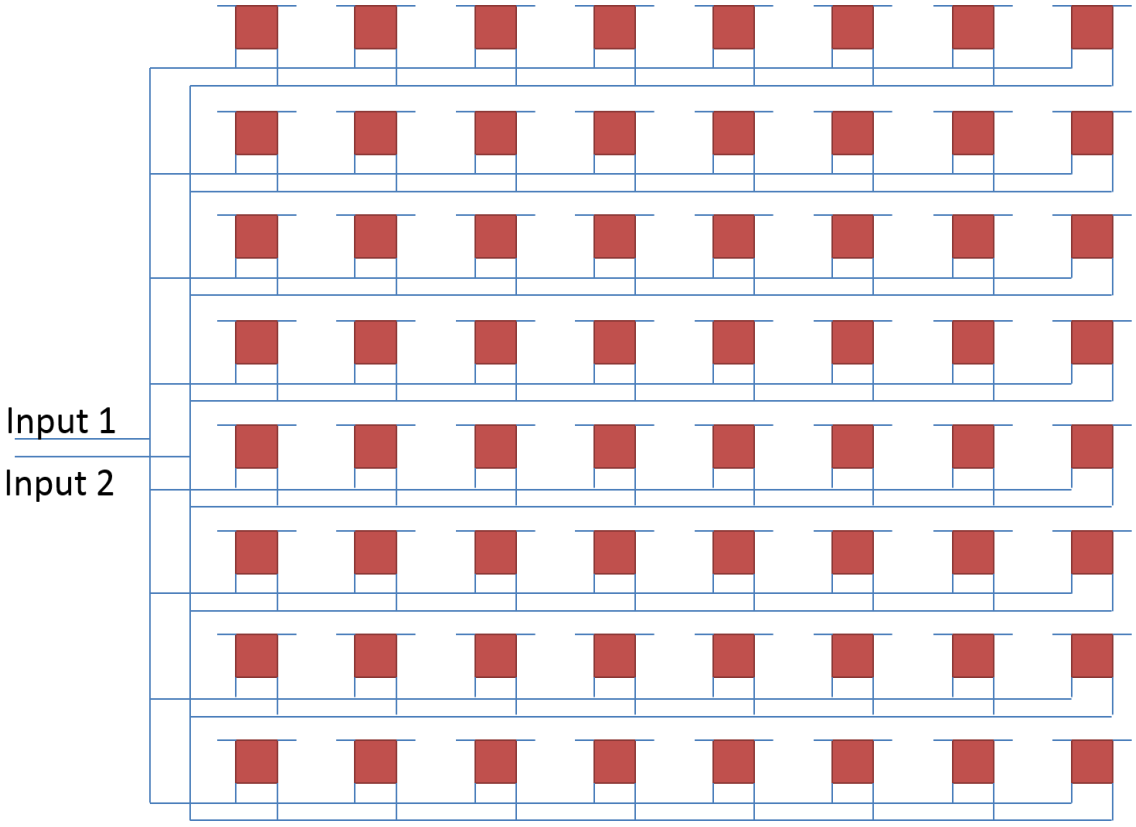


Figure.B3 Pulse sending

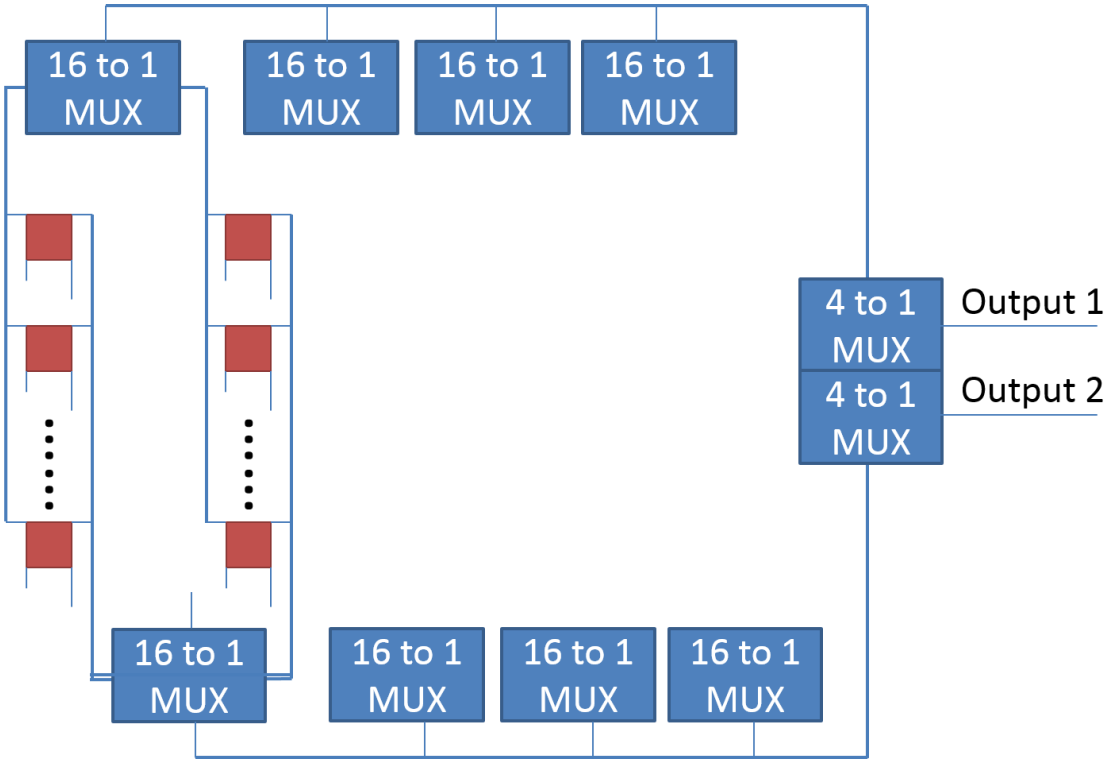


Figure.B4 Feedbacks receiving

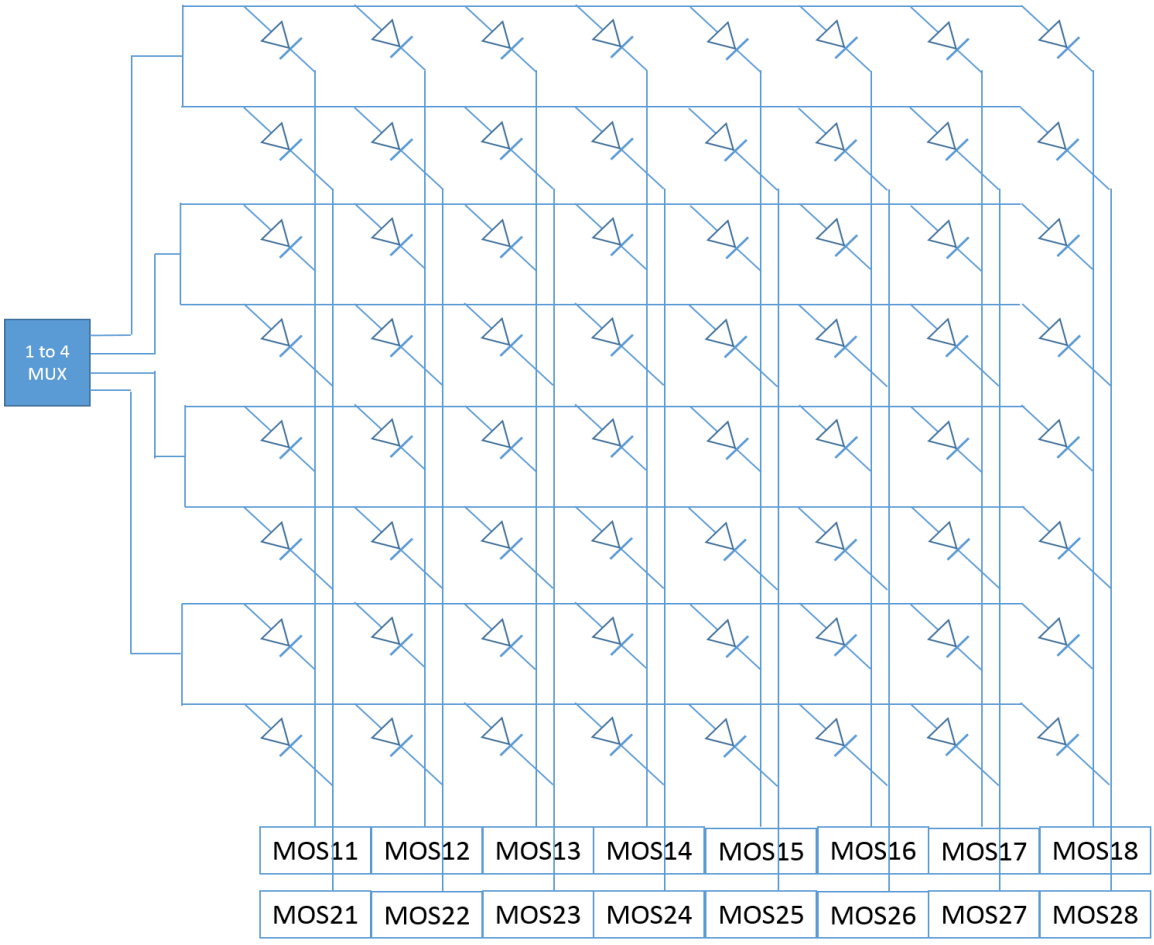


Figure.B5, LED circuits of the whole chessboard.

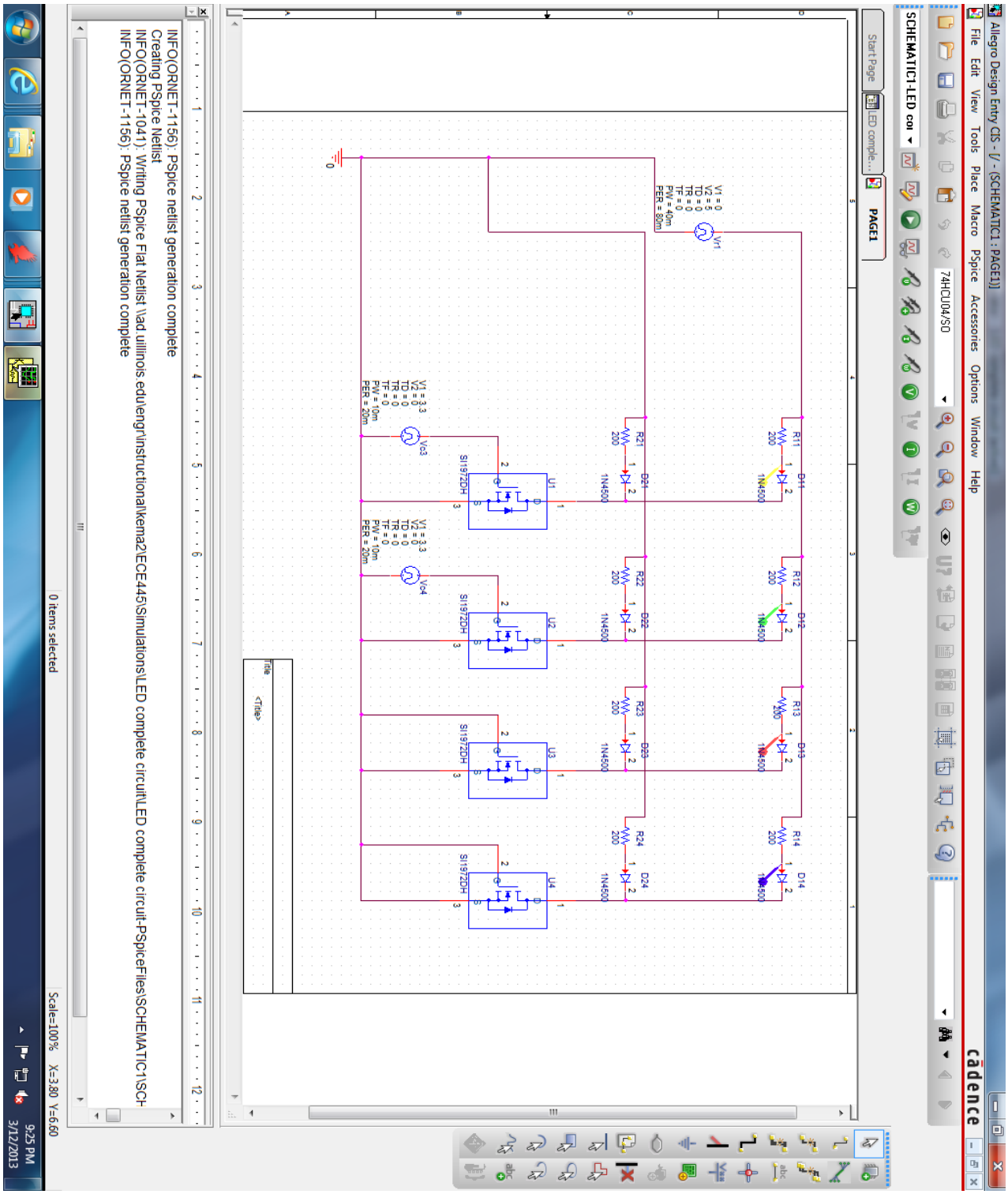


Figure.B6, Schematic of a 2by4 LED circuit

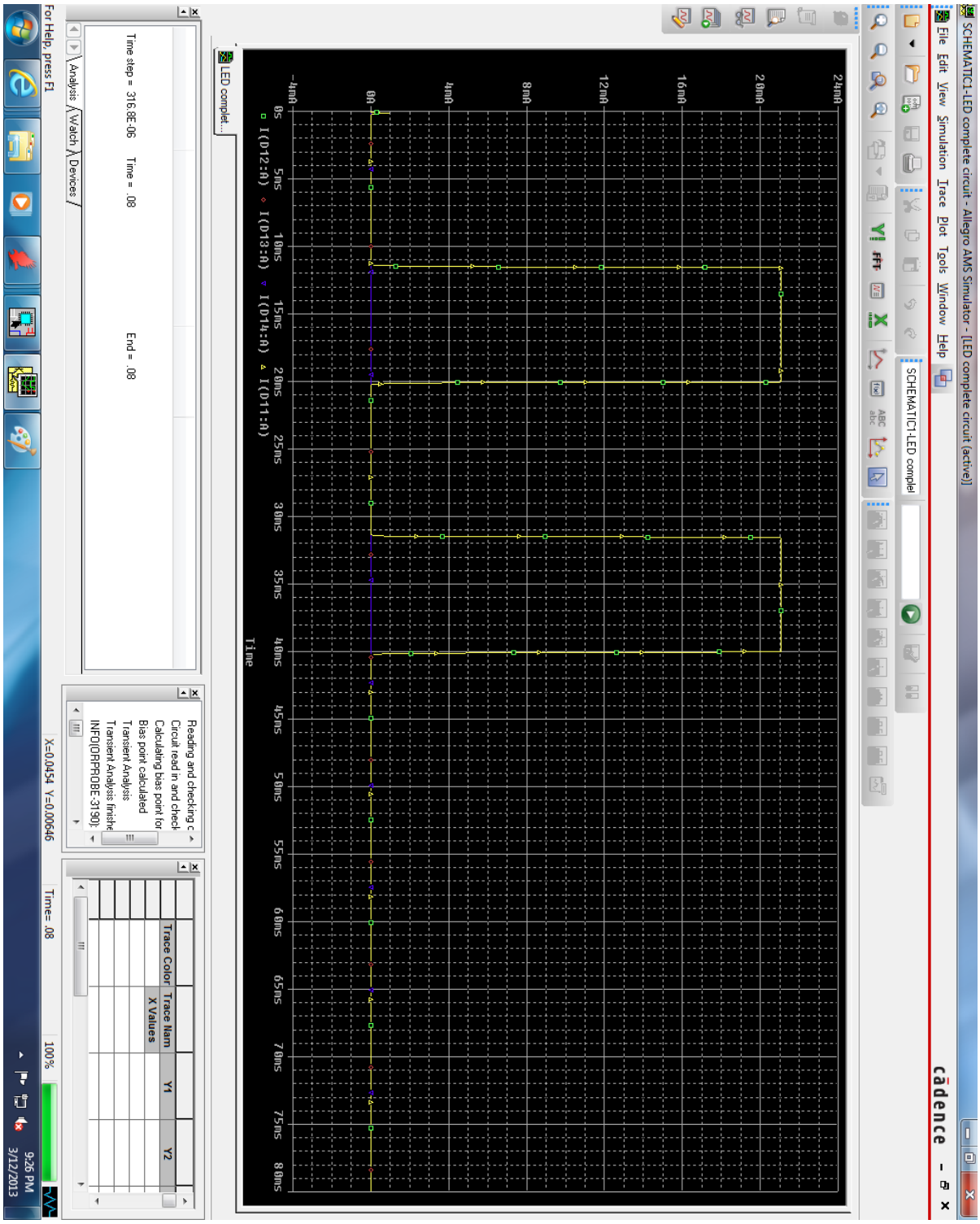


Figure.B7 simulation results of the circuit in Figure.B6

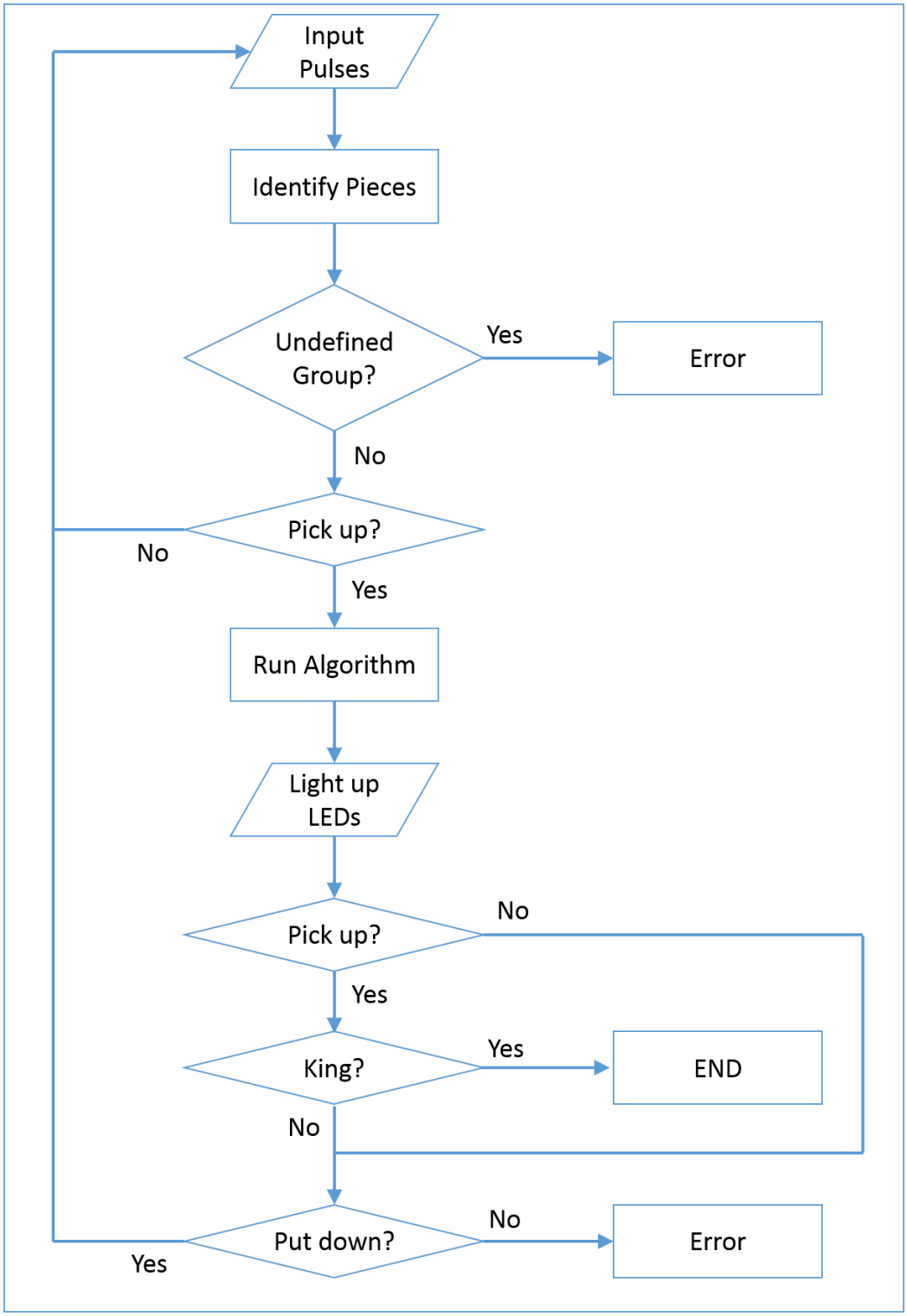


Figure.B8 Flow chart of our program

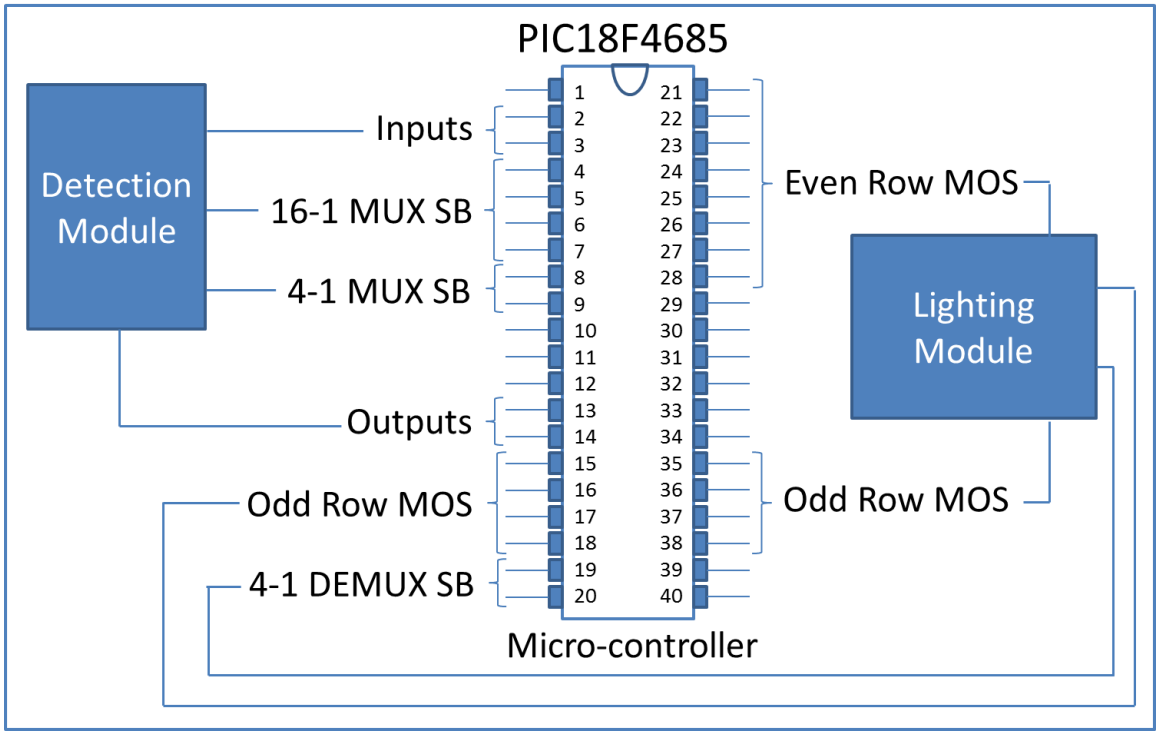

PIC18F4685
Detection Module
Inputs
16-1 MUX SB
4-1 MUX SB
Outputs
Odd Row MOS
4-1 DEMUX SB
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
Even Row MOS
Lighting Module
Odd Row MOS
Micro-controller

Figure.B9

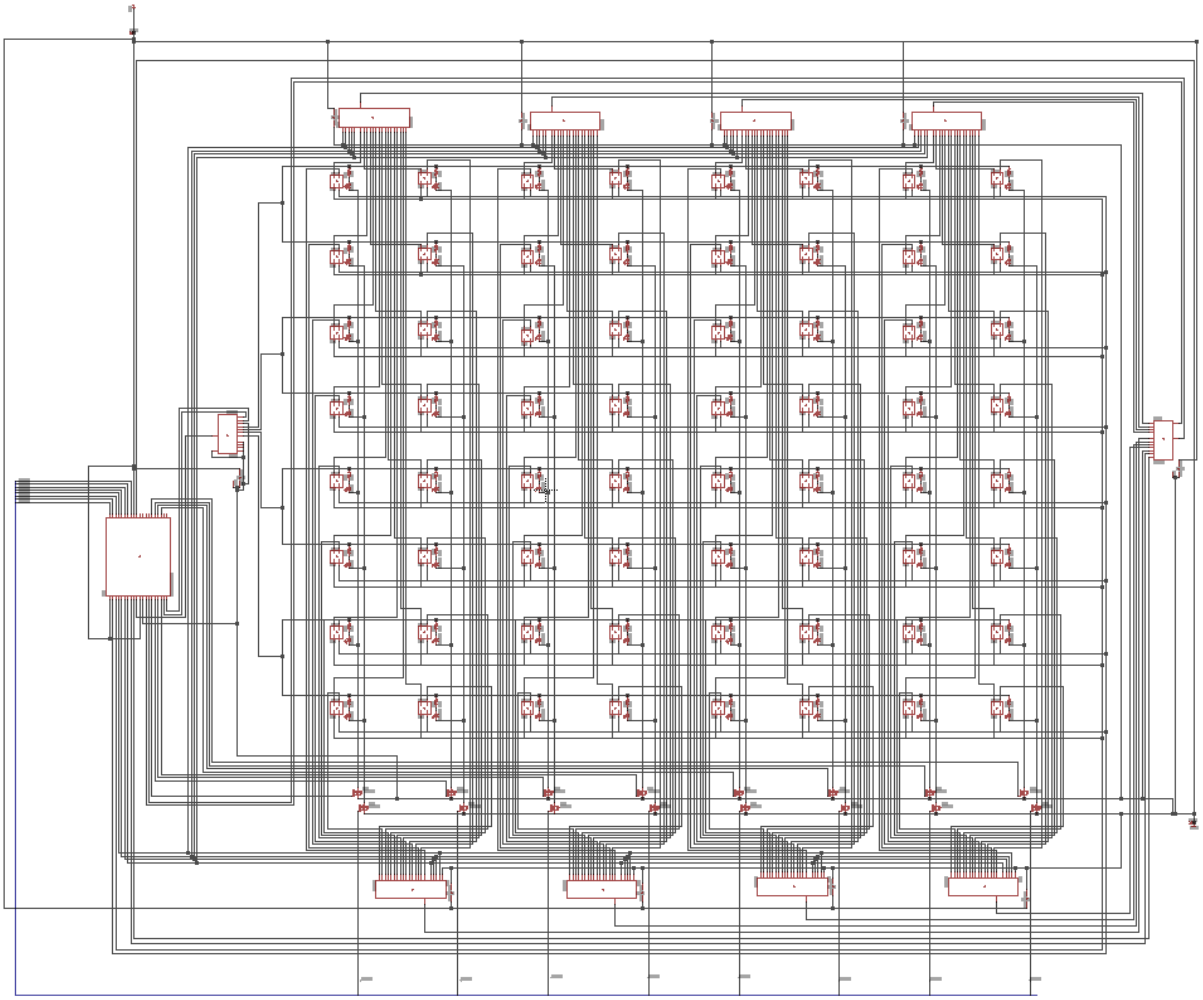


Figure.B10, The schematic of the whole chessboard

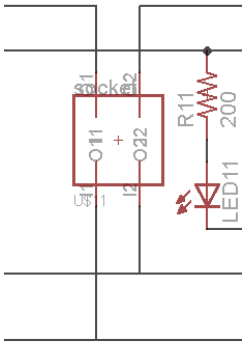


Figure.B11 the sample circuit in a square of the chessboard

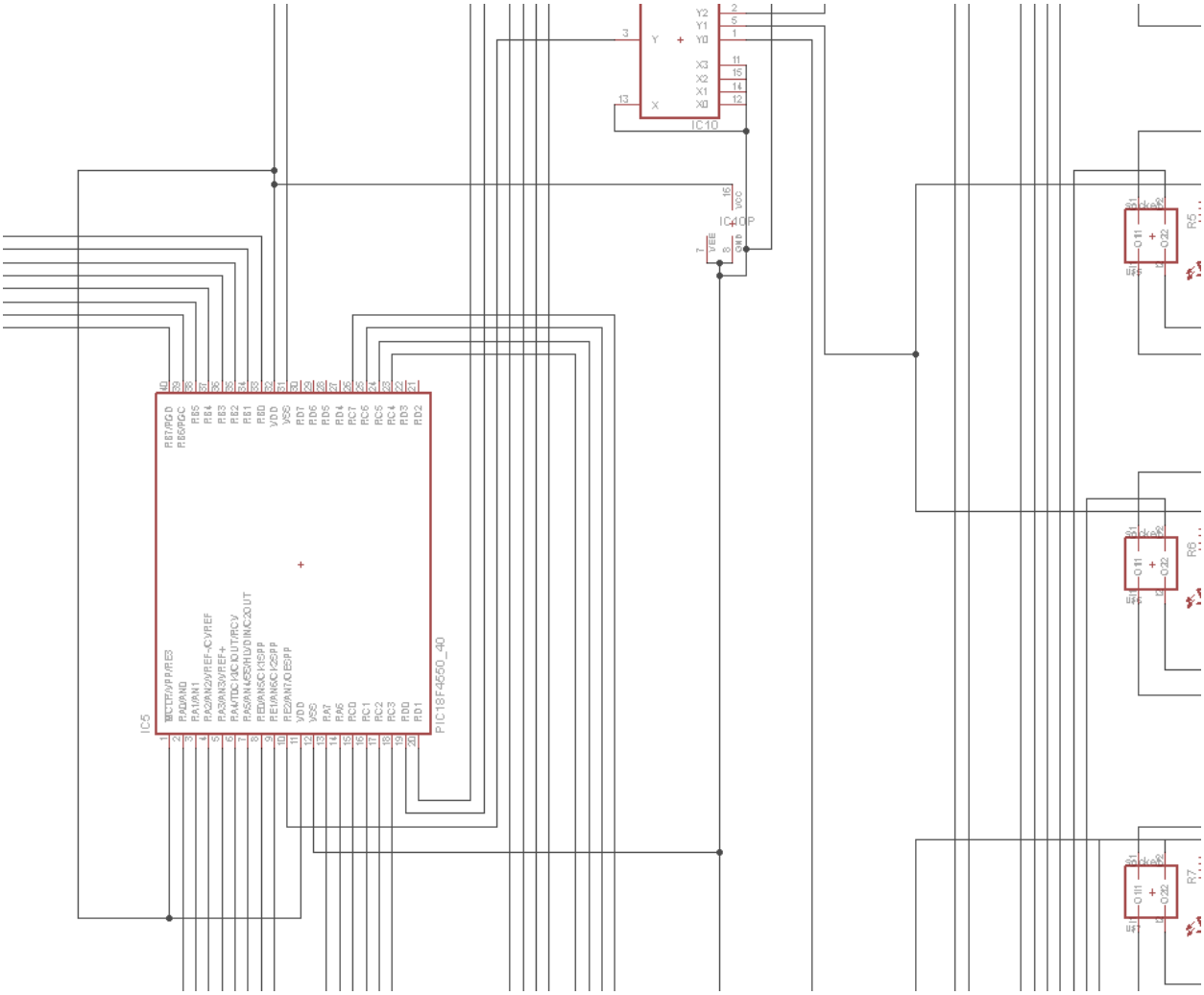


Figure.B12 the PIC we are using, since PIC18F4550 shared the same pin assignment with PIC18F4685, we used this model instead of creating a new one for PIC18F4685.

Appendix C Testing

The figrue.C1 to C7 are the testing results for detection module. For the oscillioscope, channel A (yellow) is the feedback from output1, channelB(green) is the feedback from output2 and channel C(blue) is the input signal.

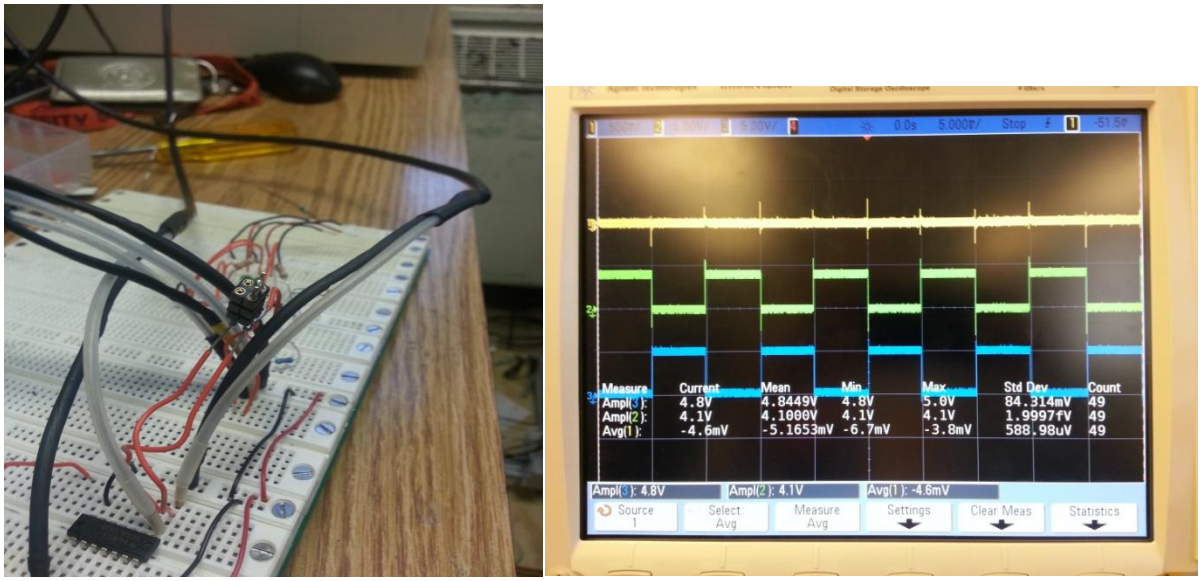


Figure.C1Pawn (A or B)

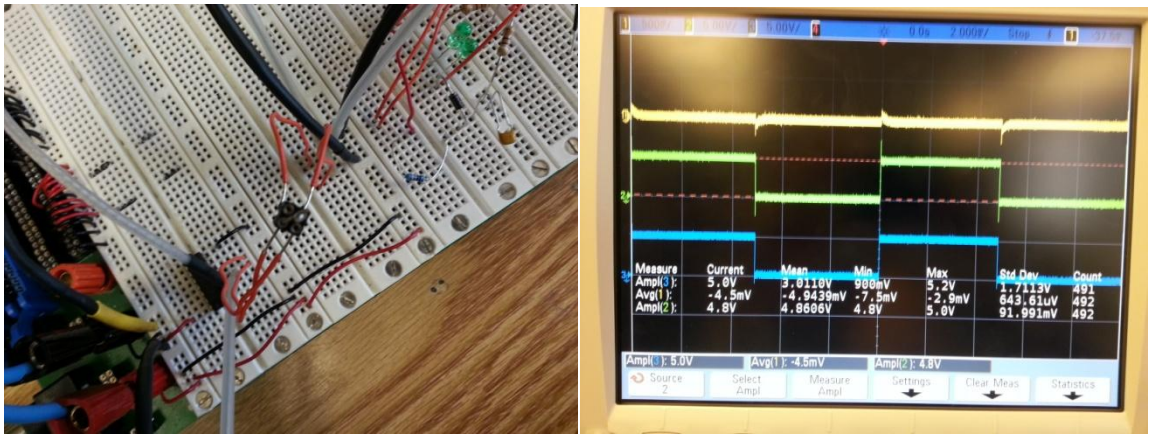


Figure.C2 Knight (A or B)

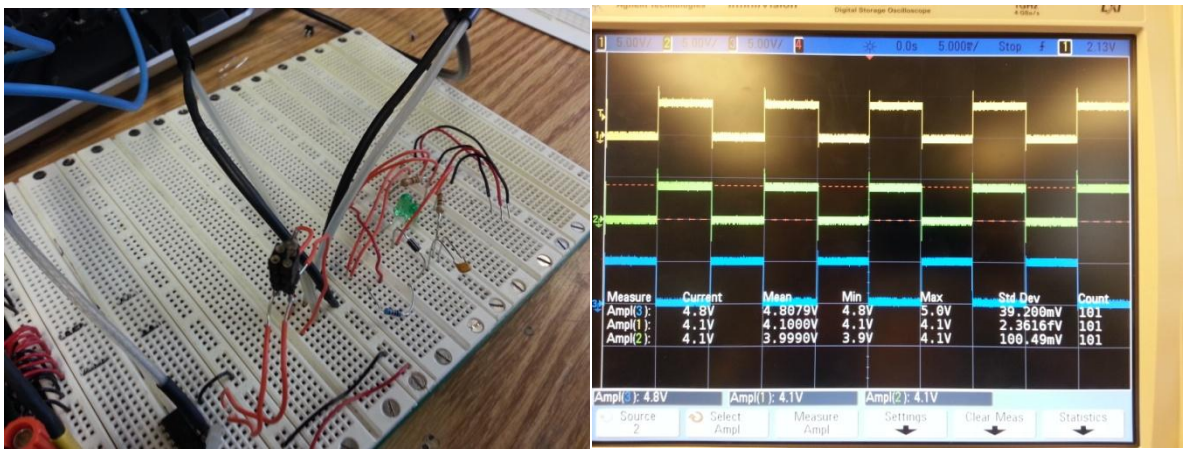


Figure.C3 Rook (A or B)

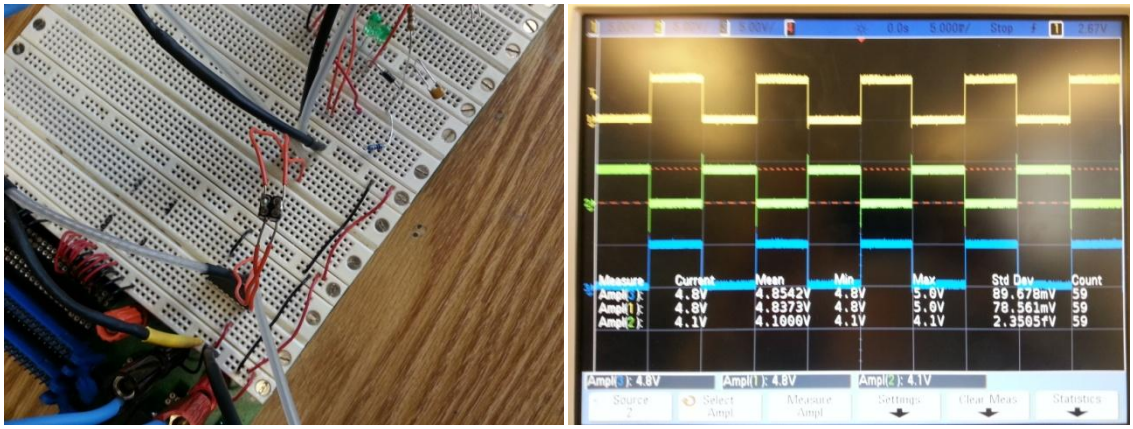


Figure.C4 Bishop (Side A)

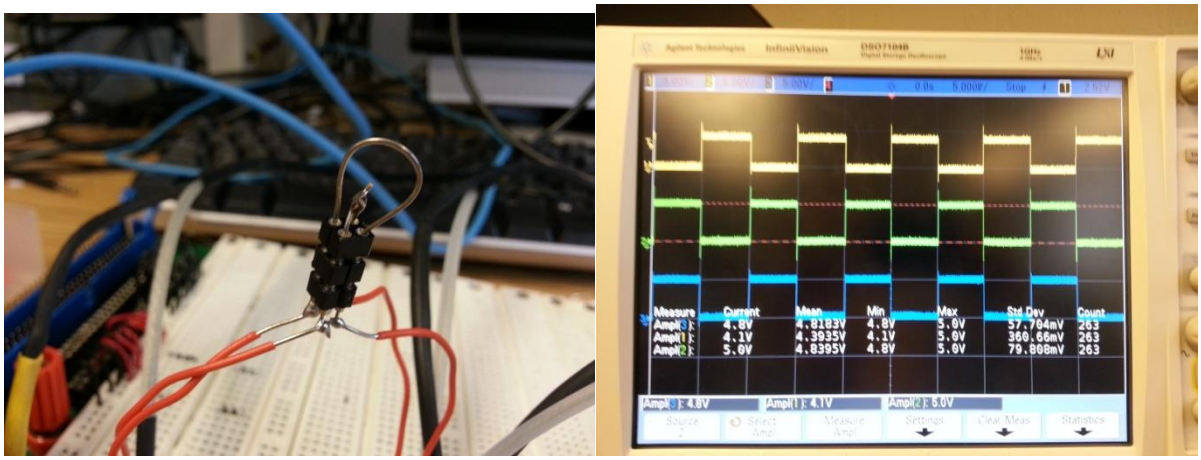


Figure.C5 Bishop (Side B)

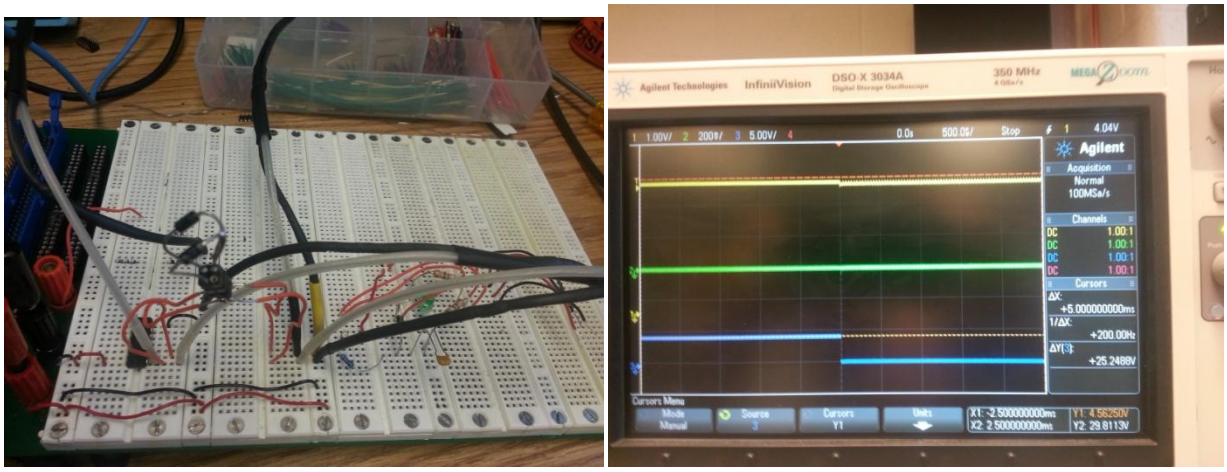


Figure.C6 King (A or B)

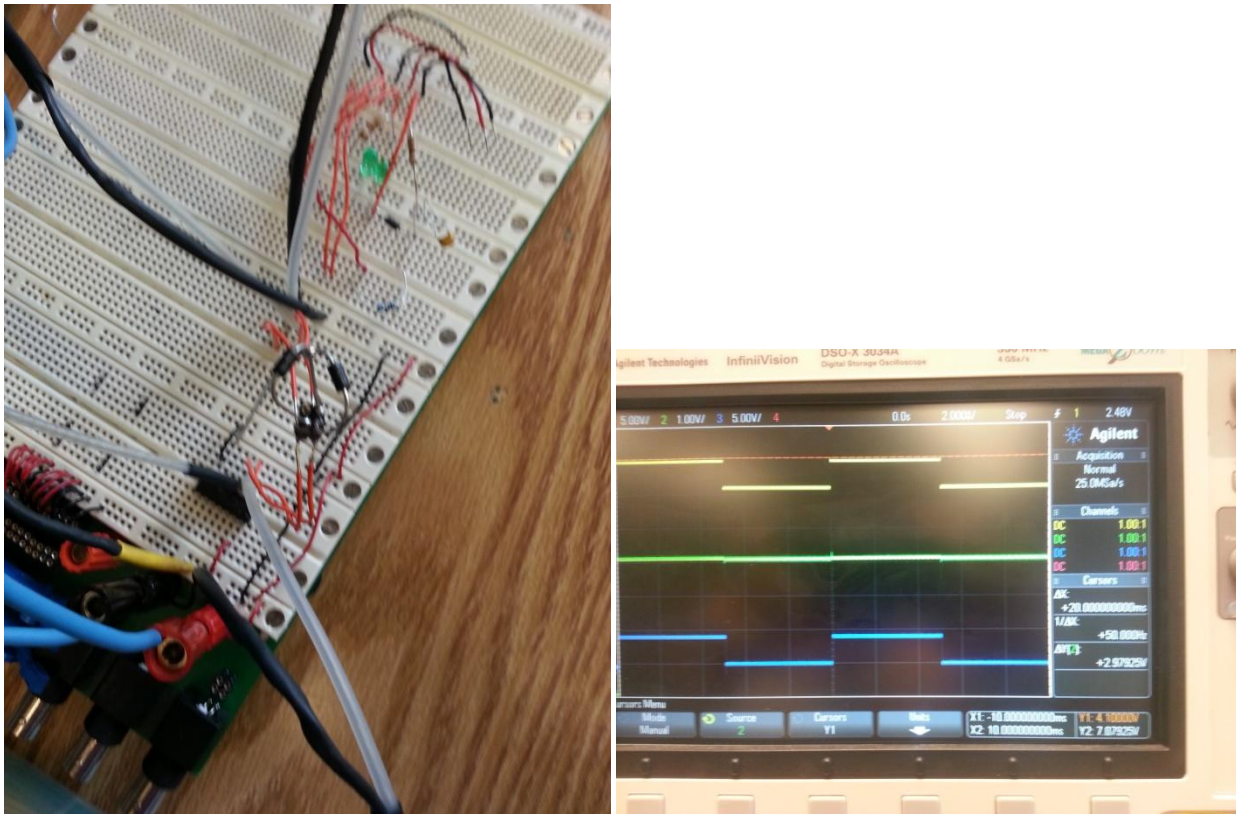


Figure.C7 Queen (A or B)

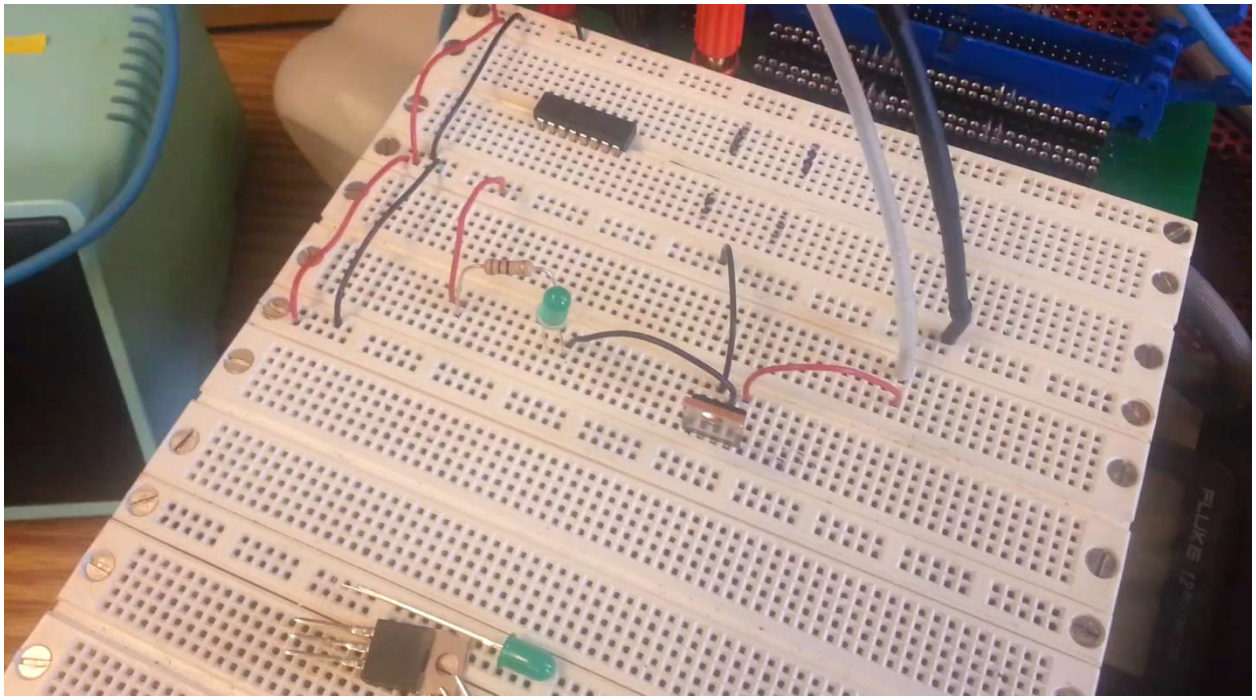


Figure.C8 (single LED circuit)

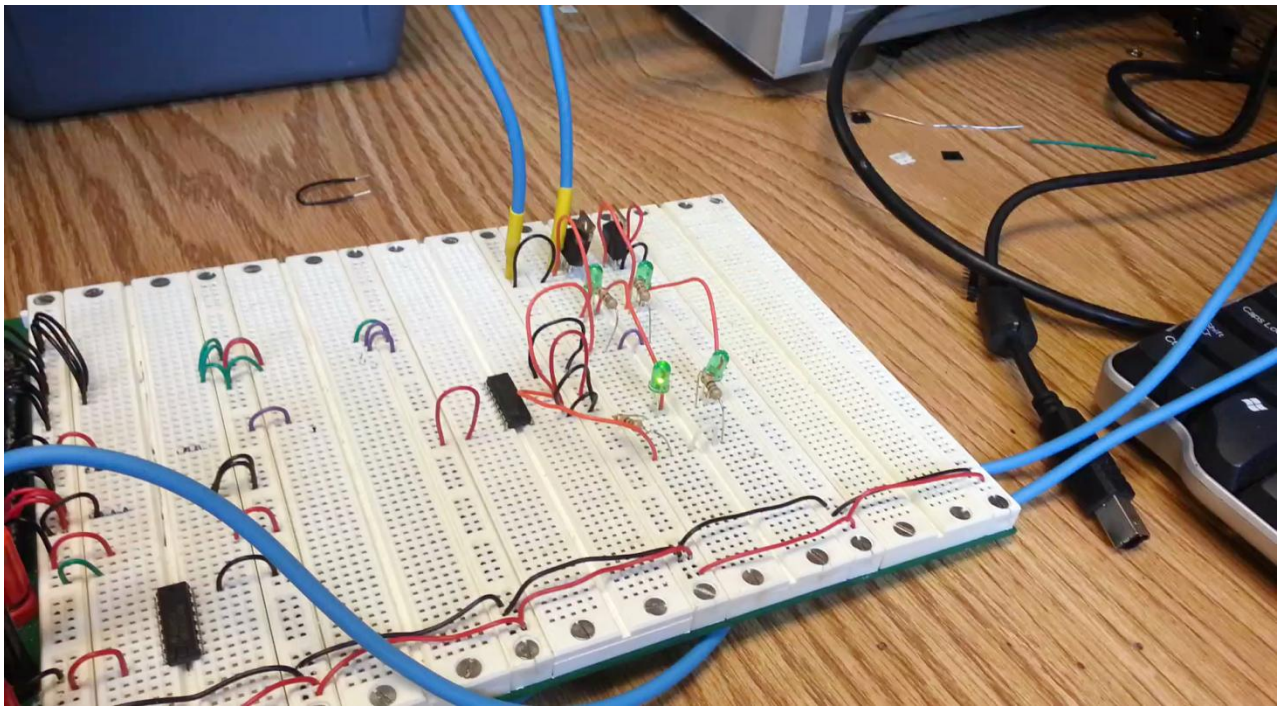


Figure.C9 (2-by-2 LED circuit)

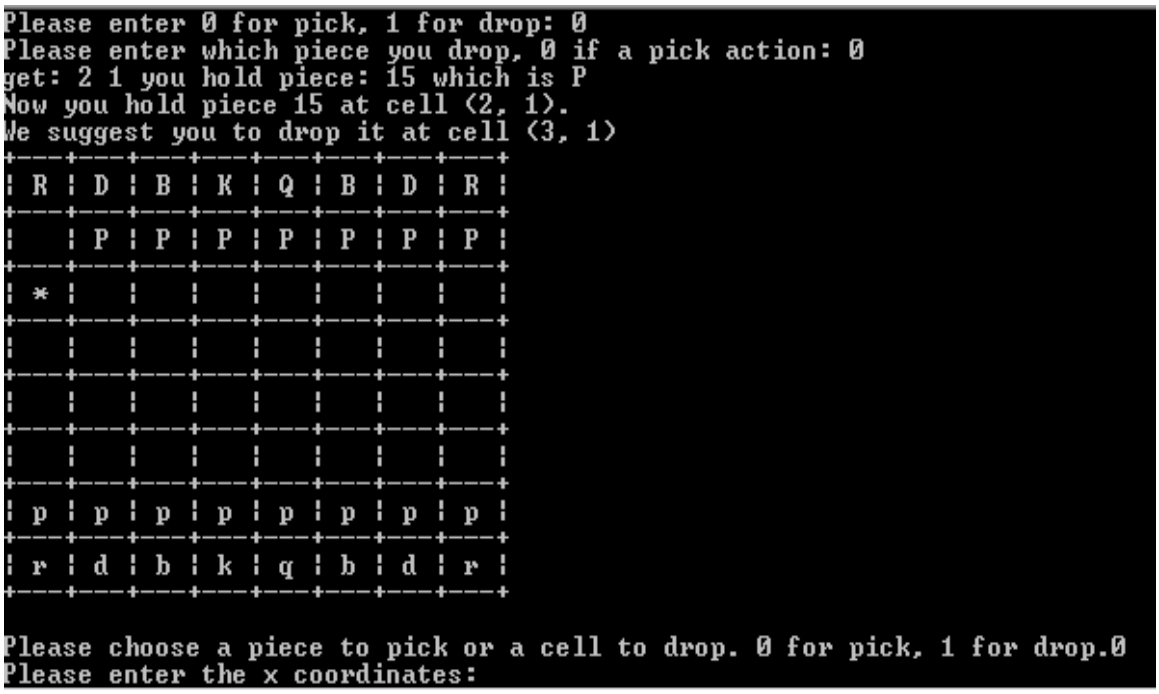


Figure.C10 (simulating chessboard)

Appendix D Requirements and Verifications

1. Detection module

Requirements	Verifications
Piece: 1. The socket could conduct current from 2 input pins to 2 output pins 2. 2. Different connections on the sockets can have different ac signal responses.	1. Connect the output pins and inputs pins, then connect socket with resistor then apply DC power across it, use multimeter test for current. Checked 2. Using 2 function generators to generate 2 identical signals with phase difference of 180. Then use oscilloscope to test 2 outputs from the chessman, and see if different kinds of chessman has different signal combinations. Checked
Pulse scanner: 1. MUX can output from 16 different input pins. 2. MUX can switch outputs when its select bits are controlled by PIC processor.	1. Connect MUX with a 5V DC power supply and test output voltage with multi-meter for different selected input pin. checked 2. Write a counter up code in PIC processor; then connect the outputs of processor to the select pins of MUX. Apply a high voltage to a specific input pin. Connect the output of the MUX to oscilloscope, and observe if the result matches the input series. Failed, since we failed importing program into the chip.
Position: 1. No current should be conduct on an empty socket when no pieces are put on a square.	1. Apply a 5V power supply on the input sides, then use multi-meter to test the current on the output pin and observe if the current is zero. Checked
Modify on PCB:	Failed, without proper decoupling by capacitors, the voltage dropped rapidly.

Figure.D1

2. Lighting module

Requirements	Verifications
LED: 1. LEDs work properly under DC power supply. 2. LEDs need to be able to work when it's connected with nmos. 3. LEDs need to be able to flash fast enough that look like a constant on state when powered by ac source.	1. Connect LED with power supply one by one and see if LEDs light up. Checked, works fine connected with 56 Ohms resistor and 5V DC supply. 2. Connect the LEDs with nmos, then supply a 5V voltage across the circuit, then supply a 4V voltage to the gate of nmos. Checked, the LED circuit works fine. 3. Connect the LED circuit with 5V DC source. Connect the gate of nmos to a function generator with 4V pk-pk ac. increase the frequency of the function generator until the LEDs look like constant. Checked, the LED looks like constant on when the frequency is above 200 Hz.
LED scanner: 1. The Demux should be able to output at different output pins when select bits are changing. 2. The Demux needs to be able to work when the select bits are controlled by processor.	1. Connect the select bits of MUX with the outputs of a 4-bits counter. Then connect the inputs of MUX to 5V power supply. Manually trigger the counter to count up and switch the multi-meter to different outputs. Checked 2. Use the same counter up code in PIC processor; then connect the outputs of processor to the select pins of MUX. Apply a high voltage to the input pin. Use oscilloscope to test the voltage of one specific pin and observe if the duty cycle of the voltage wave of that pin is. Failed, Since we failed importing program into the PIC chip.
Modify on PCB	Connect the LED circuit on the PCB board. Then power up, check if LEDs work fine. Failed, the voltage on the PCB dropped rapidly. The nmos can only receive 2.62 volts, when a 4V input was given. Wrong model were used for demux.

Figure.D2

3. Microcontroller

Requirements	Verifications
Programming 1. The codes can be successfully compiled in MPLAB. 2. The design can fit in with the program memory and the data memory of the PIC. 3. A HEX file can be correctly exported.	 1.In a project in MPLAB, add the C file into the source file folder and compile it. Checked. The program is compiled successfully. 2. In the project wizard, set the PIC to be PIC18F4685, compile again. Checked. The program is compiled successfully. 3. Export the HEX file. Checked. The HEX file is successfully exported.
Upload 1. The PIC is correctly connected with the power and the PICKIT2. 2. The computer can communicate with the PIC. 3. HEX file can be written into the PIC chip.	 1. Connect the PICKIT to the computer, and then connect its PORT 1 - MCLR with a pull-up resistor. PORT 2 - Vdd PORT 3 - Vss PORT 4 - PGD PORT 5 - PGC Checked. 2. Check communication. Checked. The Pickit 2 v2.61 can detect the PIC. 3. Import the HEX file and write it to the chip. Checked. The HEX file is successfully written to the chip.
Basic Functions 1. The PIC can output digital signals as voltages, digital 1>3.5V, digital 0 < 0.5V.	 1. Write a simple program into the PIC, set the PORTA pins to be 0b11111111. Measure the voltages at the PORTA pins. They should all be over 3.5V. Checked. The PORTA pins can light up LEDS.

	We didn't have the time to measure the voltages before the PIC was broken
2. The PIC can read voltages as digital signals.	2. Connect a 3.5V voltage to the PORTB pins, read the signals and set it to PORTB pins. Measure the voltages at the PORTB pins. They should all be over 3.5V. Failed. PIC is broken. The other PIC we have cannot work with the pic kits we have in the lab.
LED control Interface 1. With an LED ON command on a given position, the program is able to recognize the corresponding outputs. 2. The microcontroller should be able to set the pins of the outputs above, including the LED enable, the select bits of the MUX and the MOSFETS gate voltages. 3. With an LED OFF command, the PIC can set the LED enable to be low.	 1. In VS, print the outputs values to the screen. They should match to the expectations. Checked. 2. Give an ON command to the PIC, measure the corresponding pins. See if the 1s are high voltages and the 0s are low voltages. Failed. 3. Give an OFF command to the PIC, measure pin RE2. It should be low. Failed.
Detection Interface 1. With a command of detections, the loops should work to count the select bits of the 16 to 1 MUX and the 4 to 1 MUX. 2. The microcontroller should be able to recognize the feedbacks as piece types. 3. The microcontroller can give an new array of the chessboard to the AI module.	 1. In VS, print the outputs values to the screen. They should match to the expectations. Checked. 2. With a given position, plug a piece on that position. Connect the unused pins to LEDs to tell the piece type it recognize. Failed 3. Write code in the main function to compare the new and the old chessboard. Light up the changed position to see if it is correctly recognized.

	Failed.
Algorithm 1. The algorithm can suggest moves. 2. The algorithm can recognize regrets. 3. The algorithm can work well on screen. 4. The algorithm can work well on PIC.	1. Add input and print functions to the algorithm. Test it with commands. Checked. 2. Same with 1. Checked. 3. Same with 1. Checked. 4. Write the entire codes on the PIC. Test it on the breadboard. Failed.

Figure.D3

Appendix E Cost and Labor

Parts

Name	Price/Unit ($)	Quantity	Cost($)
4-pin socket	0.4	64	25.6
HLMP3507 LED GREEN T-1 3/4 (5MM)	0.15	64	9.6
CD74HCT139 decoder	0.64	1	0.64
PCB	0	1	0
Miscellaneous (resistors, capacitors, wire, diode)			10
IRF 520 NMOS	0.96	16	15.36
MC14067B-D 16:1 MUX/DEMUX	1.29	8	10.32
PIC18F4685	6.75	1	6.75
SN74LS153N DUAL 4:1 MUX	0.44	1	0.44
Total			78.71

Figure.E1 Parts cost

Appendix G Project image

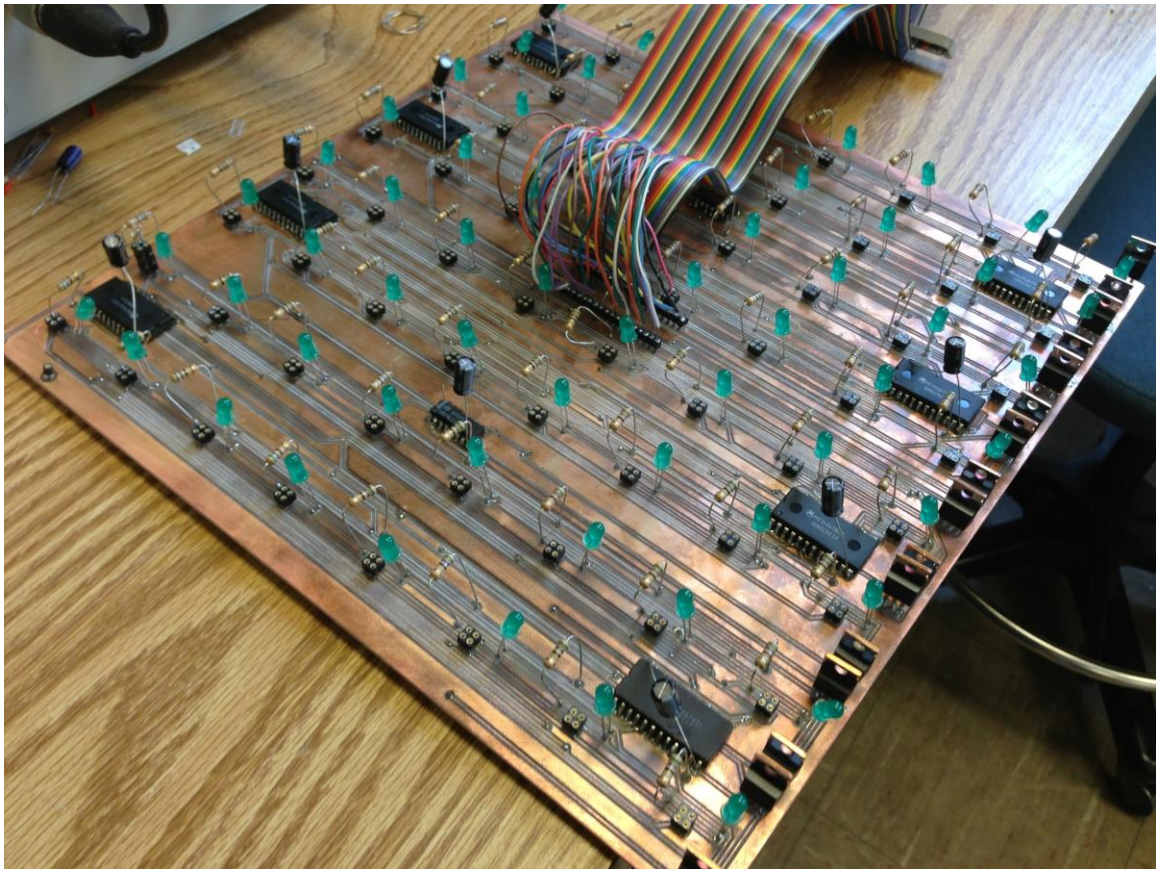


Figure.G1 The completed chessboard