School of Engineering

Board Interface

Prototype and PCB design

NAME: Adam Bance Student ID:X00183440 Date:3/10/22

Declaration

All students are expected to complete their courses in compliance with university regulations and standards. No student shall engage in any activity that involves attempting to receive a grade by means other than honest effort, for example:

1. No student shall complete, in part or in total, any examination or assignment for another person.

2. No student shall knowingly allow any examination or assignment to be completed, in part or in total, for himself or herself by another person.

3. No student shall plagiarize or copy the work of another and submit it as his or her own work.

4. No student shall employ aids excluded by the instructor in undertaking course work.

5. No student shall knowingly procure, provide, or accept any materials that contain questions or answers to any examination or assignment to be given at a subsequent time.

6. No student shall procure or accept assignments from any other student from current or prior classes of this course.

7. No student shall provide their assignments, in part or in total, to any other student in current or future classes of this course.

8. No student shall submit substantially the same material in more than one course without prior authorization.

9. No student shall alter graded assignments or examinations and then resubmit them for regrading.

10. All programming code and documentation submitted for evaluation or existing inside the student's computer accounts must be the students original work or material specifically authorized by the instructor.

11. Collaborating with other students to develop, complete or correct course work is limited to activities explicitly authorized by the instructor.

12. For all group assignments, each member of the group is responsible for the academic integrity of the entire submission.

13. Each student is responsible for knowing and abiding by TU Dublin Academic Regulations and Policies. Any student violating these regulation/policies will earn an 'F' in the course and will be reported to the University for the violation.

By signing this document, I understand will abide by these rules and regulations

Signature **Adam Bance**

Date: 27/10/22

Contents

Declaration 2
Abstract 5
Acknowledgements 5
Introduction 6
Objective 8
Overall project planning 9
Semester 3 Prototype plan 9
Semester 4 PCB design plan 10
Circuit overview 12
Theoretical Operation 13
1. THE OR GATE 13
2.THE NOR GATE 16
3. 74HCT245 18
4. 74HCT138 21
5. 74HCT251 23
6. The 74HCT259 26
Bill of materials 29
Multisim 30
Ulti-Board 32
PCB unrouted 32
PCB routed 34
Copper top and Bottom 35
Trace width 35
Trace clearance 37
Final board outline 38
Hole pin diameter 38
Interfacing MSP430 to processing 40
Objectives 40
Processing code #1 41
MSP code #1 42
Processing and MSP430 42
GUI 42
Processing code #2 43
MSP code #2 44
Using feedback from MSP430 45

Processing code #3 45
Surface mount technology 47
What is Surface mount technology? 47
Soldering in surface mount technology 47
Operation 47
Switches matrix 49
Objective 49
Understanding of operation 49
LEDS 49
Software code for motherboard 50
Turning on a bit 51
Turning on a byte 51
Full code #1 52
Full code #2 54
Test procedure for The prototype 55
Test procedure for the PCB 56
Problems and solutions (prototype) 57
Problems and solutions PCB 58
Conclusion and recommendations 59
Bibliography 60
Appendices 61
Prototype 61
PCB and motherboard 61
Ribbon cable and MSP430 62
PCB routed (copper top and bottom) 63
User manual Guide 64
Safety And Hazard 64
Environmental impacts 65
References 66

Abstract

The main aim of this project is to design, build, test and verify a board interface in a professional manner. The design will be constructed on a 39x62 vera board and will consist about 8 different IC chips, resistors, switches and LEDS. The IC Chips that were used were the OR gate, NOR gate, 3-to-8-bit decoder ,8 to 1 multiplexer, the octal bus transceiver and the 8-bit latch. The LED was our original idea for the output but was replaced with a 7-segment display. Before the actual building of the board interface, we would first build and verify each IC board using the MSP4-30 on Energia. Verify and construct a circuit with each chip made a greater understanding of how the logic gates work so that when it comes to the actual building on the vera board there would be little or no confusion and more conclusions. The aim of this project is to design, build, test and verify a board interface in a professional manner.

For the design constructed was the board interface PCB which measured at 57.65 x 57.65 mm. The board also will consist about 8 different IC chips, resistors, switches and LEDS. The IC Chips that were used were the OR gate, NOR gate, 3-to-8-bit decoder ,8 to 1 multiplexer, the octal bus transceiver and the 8-bit latch.

In the actual design its self the student will use Multisim, Ult-board, Energia and processing code to break down and figure out how the board works while connected to the mother board.

Acknowledgements

I would like to give all my thanks to Tom Murray who is my lecture and mentor who inspired me to help construct and test my board interface. His words of wisdom through suggestions and advice helped me during the whole process. I also would like to give a big thanks to my fellow colleagues including Bryan who made great understanding what was to be completed and helped me think about possible ways of making this project.

Introduction

- So, the overall breakdown of the board Interface is 60 percent hardware and 40 percent software. Before any build or testing there must be an overall planning done to demined what and when must a certain thing be done by so that at the last minute everything is not overlapping each other and rushed. The importance of planning is very significant and the key thing to constructing this project such as truth tables schematic, pin layouts, K- maps, Boolean equations etc.

- The Great things you can get from planning is understanding, acknowledgment, achievements knowledge Time management and without this thing this will make everything even tougher to complete. Once the planning is laid out then there would be a phase of logic gate building and testing to further understand the logic operation of the chip that will be place on the board interface its self. Conclusions and recommendations can be made from this. The final testing will be writing the code on a java program called Energia Using the MSP4-30 to verify. The codes for each IC chip will be used as a testing code for each IC chip when demonstrate it towards the end.

- The software Element is as highly important as the actual circuit build because one without the other cannot successfully work. Whatever result shows on the on the circuit should also show on the program. Observing great knowledge from all this is crucial and good in a professionally in terms of understanding the board interface.

- The board interface app can be used in the world of Engineering for someone taking over a previous employee position. Once someone comes in and is asked to figure the old employers work, the individual would find out it difficult to know how it operates. So then comes the board interface app. It's used to figure out problems and gives us an understanding of how the board works. For this project the student will need a:

- Designed PCB board with 8 different IC header pins
- MSP430
- The motherboard

- The motherboard consists of various components such as about 55 LEDS, 14 IC chips and 21 pushbuttons. In order for everything to work together we would need feedback from processing all the way to Energia in order to say which button was pressed in order to turn on an LED. This simply means that the student will have to implement a GUI which consist of your LCD, MCLR, D_IN, S1, A, C, E2, VCC, GRN, E3, E1, B, S2, S0, OR and DOUT.

- In this project the student will use his skill and knowledge from last semester to understand and design the board interface. This design of both prototype and PCB can benefit the student in the long run by supplying them with knowledge and problem-solving skills. With mistakes likely to happen it would be learning curve for the student to get a grasp of the world of engineering today. .

Objective

- The objective is to create and test the board interface it fully working with test codes, planning and sketches of how it will operate.

- All hardware must be designed neatly in professional manner with a log ready to catch and record errors as I go along.

- All safety precautions and hazards should be acknowledged in the process of the build.

- All time schedule is up to date on tackling this project before the end of the semester.

Overall project planning

- This is the overall layout plan off the board interface prototype semester 3 and semester 4 PCB design. This plan was taking as worst-case scenario as it shows the deadline of certain plans and shows the direction in which the student is heading.

Semester 3 Prototype plan

- **First week (30/9/22) 74HCT32 (OR) 74HCT02(NOR):** Building and testing the two logic gates then connecting them to the msp430 to run a code to verify its operation.

- **Second week (7/10/22) 74HCT138**: same plan as the OR and NOR gates using the MSP40 with Energia.

- **Third week (14/10/22)74HCT251**: understanding the logic behind its inputs and outputs and most importantly its enable pin while building and testing it.

- **Fourth week (21/10/22) 74HCT245**: looking into the operation of the 8-bit transceiver and what makes it work then testing it on Energia.

- **Fifth week (28/10/22) 74HCT259**: figuring out the last chip which is called the 8-bit addressable latch and its different modes when testing it

- **Sixth week (4/11/22):** planning the layout of the chip on board

- **Seventh week (11/11/22):** the start date of soldering and building.

- **Tenth week (2/12/22):** end of building and testing.

Semester 4 PCB design plan

Table 1 shows project plan for semester 4

week number	Dates	Task	completed
1	1/27/2023	project planning	
2	2/3/2023	footprints	
3	2/10/2023	PCB-Unrouted	
4	2/17/2023	PCB sent out	
5	2/24/2023	LEDS	
6	3/3/2023	surface mount tech	
7	3/10/2023	building/Testing/mods	
9	3/24/2023	switches Direct	
10	3/31/2023	switches matrix	
13	4/21/2023	LCD	
14	4/28/2023	Quiz	
15	5/5/2023	logbook/report/hardware	

- The 1st week (1/27/2023) the student should have everything drawn out and everything planned which includes schematics, sketches and measurements. This used with Multisim.

- In the 2nd week (2/3/2023) PCB footprints would be expected to be started and finished with everything checked back to the schematic layout and components in Multisim. Ulti-board is used to design the foot prints in this process.

- At the 3rd week (2/10/2023) The PCB should be routed and triple checked with every measurement, each component and connection. The copper should be finalized and the overall board size should be agreed on.

- Realistically in the 4th week (2/17/2023) the PCB should be sent out but since this worst-case scenario, the deadline to be sent out should be this week and no later.

- In the 5th week (2/24/2023) the student should create 2 different codes from processing and Energia to create input boxes which have the gates used to turn on a LED.

- In the 6th week (3/3/2023) The student should be able to solder and desolder on a PCB design if there may be an error and is able to do it correctly even fixing while fixing it.

- By the 7th week (3/10/2023) the PCB should sent back to the student. This would be the time to perform tests to see if there are any errors and if there is modify it. This would also be the time to add in header pins using surface mount technology skills.

- Week 8: reading week: During this week of college, the student could use this week to catch up on missed weeks or could use this time to catch up on the project report.

- Week 9: Switches +Direct: Similarly, to week 5, this week (3/24/2023) would use to set up the switches using processing while using the MSP430.

- Week 10: switches matrixes: At this stage (3/31/2023) the motherboard should be connected up to the MSP430 which is also connected to the PCB design. Using Energia, the student would have to figure out what chip turns on or triggers which to turn on a single LED on the motherboard.

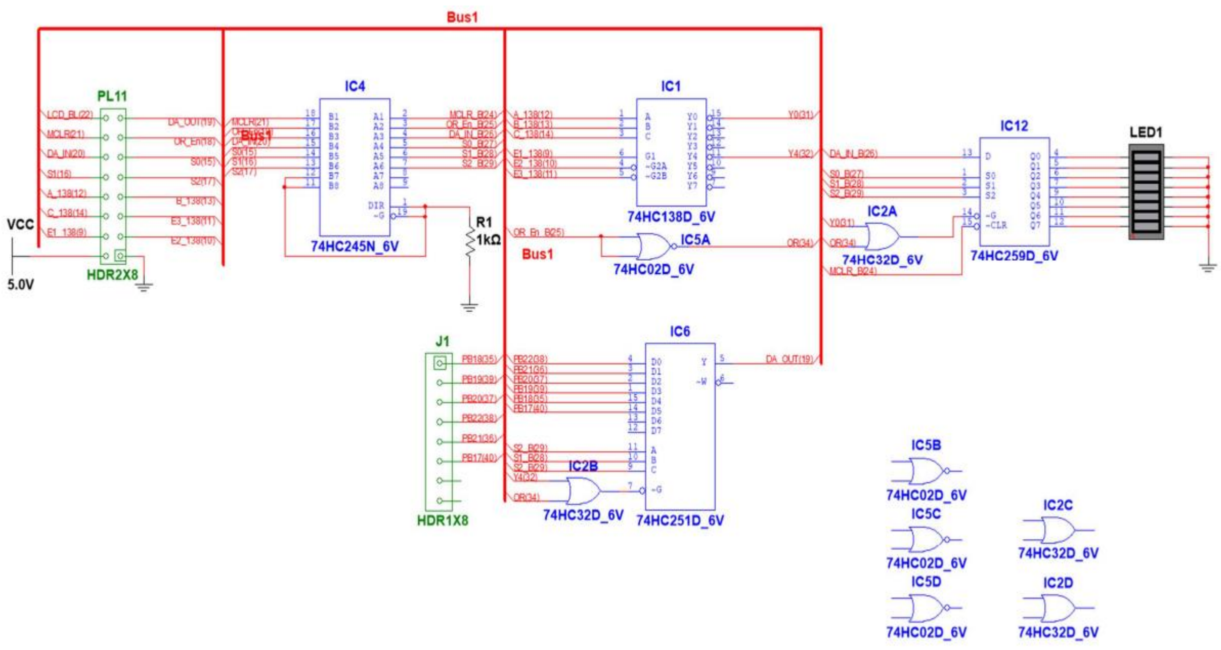
- Week 11 and 12 Easter break: This would be the perfect 2 weeks for the student to work on the project report and finish any building or coding on the project.

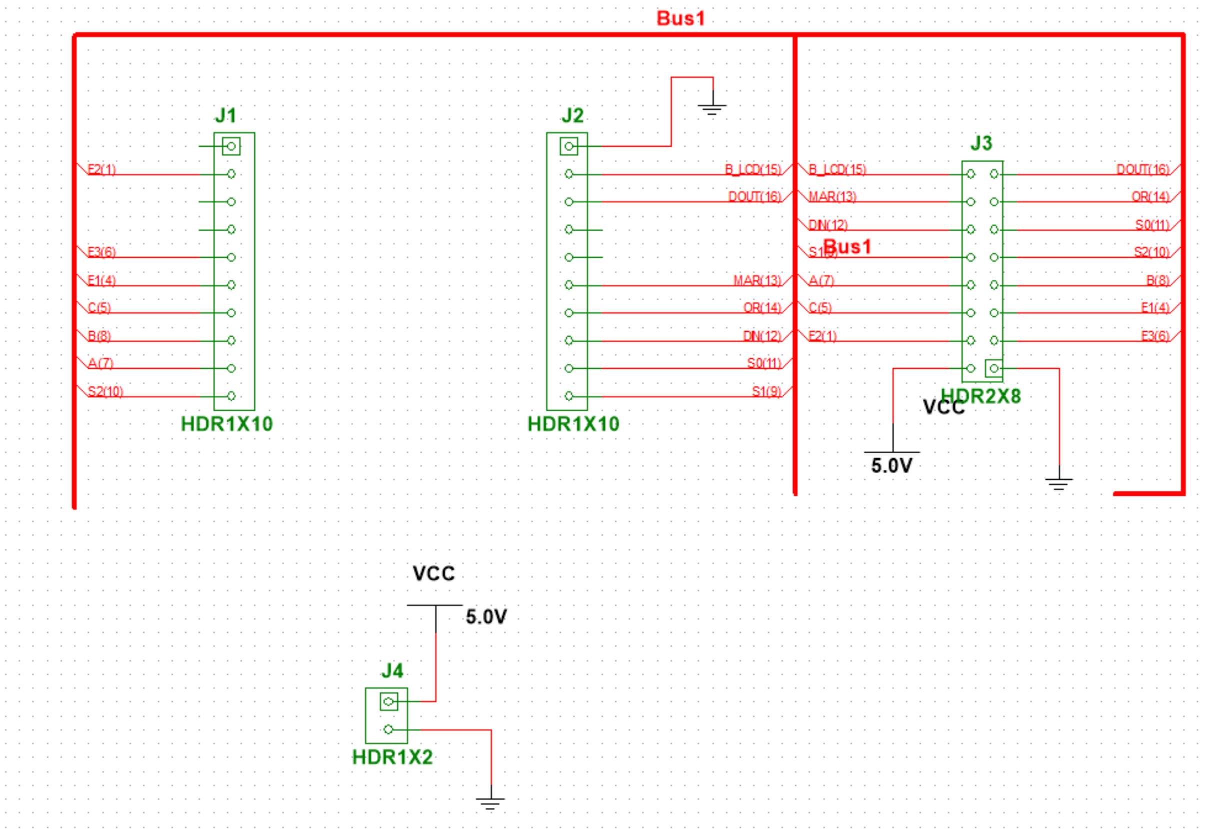
- Week 13: LCD

- Week 14: Quiz: During this week (4/28/2023) student will be quizzed on everything up to date. This will see what the student has learned and what student understands about the project.
- Week 15: Hardware/logbook/report

This week (5/5/2023) would be the deadline for everything. physical PCB hardware would be tested and examined by the lecturer, the logbook and report should be all handed up and given to the lecturer.

Circuit overview



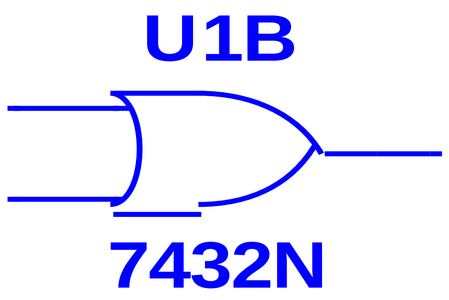


Theoretical Operation

In this theoretical operation in will explain and break down each circuit with a chip that the student has built and tested on a patch board which can then further explain on how it operates. The theoretical operation is very important which can hold very crucial information

1. THE OR GATE

An OR gate is a 2 or more input gate with one out put that can actually performs the logical disjunction. if one out of two of the inputs are a one then the output is a one as well as if both input if they were all ones as shown in figure



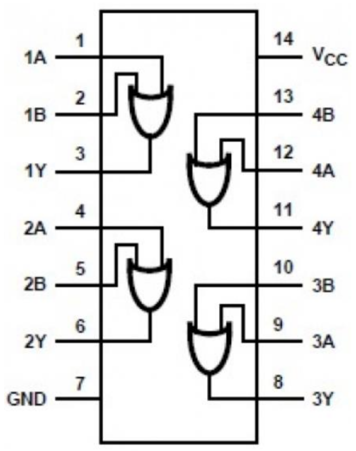


Figure 1.2 logic chip layout with pin numbers

Figure 1.1 Logic OR gate symbol from google images

In figure 1.2 it shows four logic gates inside the 74HCT32. This chip can produce 4 outputs. These outputs are 1Y,2Y,3Y and 4Y.pin 14 is where VCC is regulated and pin 7 is the Ground pin. IN order to know how it works we must look at truth tables and k- maps.

Supply Voltage :5 Volts

Supply Current: 20mA

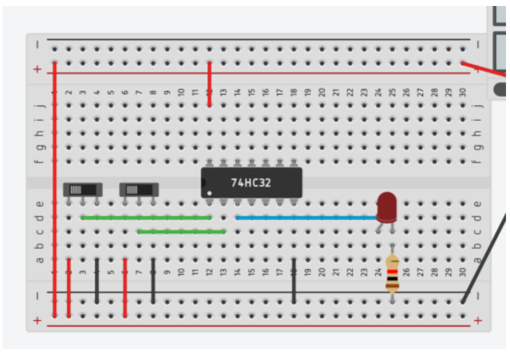
State	A	B	Y
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

0	1
1	1

Figure 2.2 K-map for logic 0R

Figure 2.1 0R gate truth table

Boolean Equation: A+B =Y



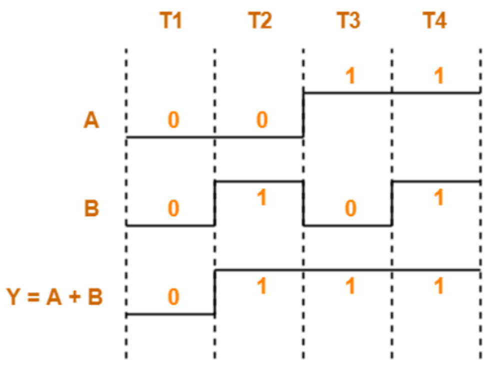


Figure 3.2 timing diagram showing the inputs and outputs from google images

Figure 3.1 built circuit for OR gate

Above in figure 3.2 the timing diagram confirms and shows us what's taking place inside the chip. This tells gives me a better understanding of the operation of the OR gate logic. I then had to use Energia and show the same results using this code below.

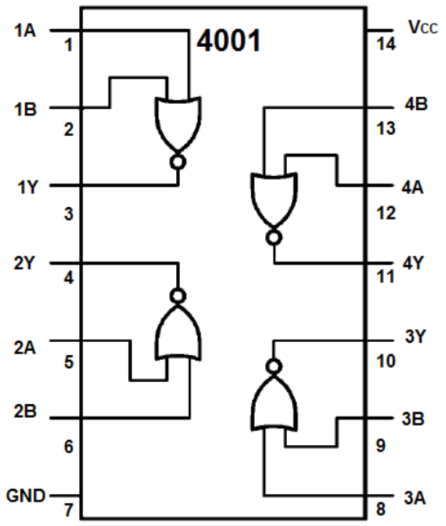
```
#define A1 P1_4
#define A2 p1_5
#define Y1 P1_6

void setup() {
  for(int dec=0 ; dec <=3;dec++){
    digitalWrite(A1,bitRead(dec,1));
    Serial.print(dec);
    Serial.print("\t");
    Serial.println(digitalRead(Y1));
    delay(1000);
  }
}
```

- Code Operation: In this code I went to check how the OR gate works by code so I first enter the pins and what I set the pins to. Next, I put in serial as 9600.After that I use pin mode so it knows that we are going to set them as inputs and outputs. The reason why A1 &2 are outputs is because it's going out from the MSP into the circuit with the OR gate. Once that done, we then use a for loop in void loop by declaring dec and saying if dec is less than 3 then increment. I then say A1 as dec 1 and A2 as dec 0 which in this case is the output.it will then print out 1 and 0s underneath each other after every 10 milliseconds

2.THE NOR GATE

The nor gate is a contraction of a not and or and also implies an or gate function with a complemented output. The nor gate has 2 inputs with one output.it behaves according to the truth table down below.

Truth table and logic symbol:



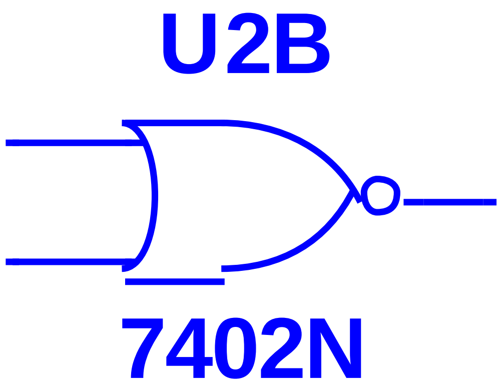


Figure 4.2 logic gate of the 02

Figure 4.3 logic pin layout of NOR gate from Texas instruments

State	A	B	Y
0	0	0	1
1	0	1	0
2	1	0	0
3	1	1	0

1	1
1	0

Figure 4.4 truth table and K-map

Logic circuit diagram

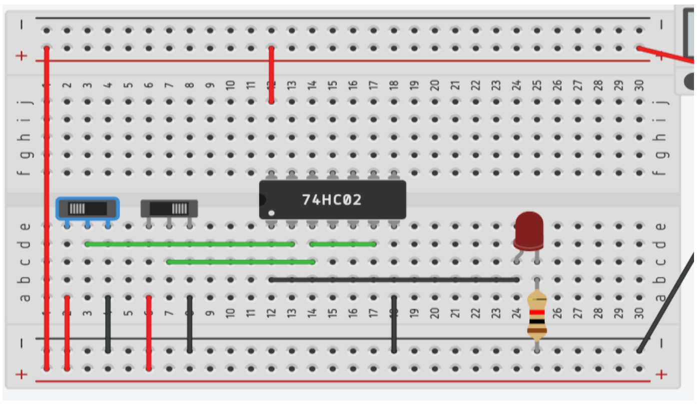


Figure 5 logic built circuit

```
   #define A2 P1_5

#define Y1 P2_0
#define Y2 P2_3

void setup() {
  Serial.begin(9600);
  pinMode(A1,OUTPUT);
  pinMode(A2,OUTPUT);
  pinMode(Y1,INPUT);
}
void loop(){
  Serial.println("or\nor");

  for(int dec=0 ; dec <=3;dec++){
    digitalWrite(A1,bitRead(dec,1));
    digitalWrite(A2,bitRead(dec,0));
    Serial.print(dec);
    Serial.print("\t");
    Serial.println(digitalRead(Y1));
     Serial.print("\t");
     Serial.println(digitalRead(Y2));
    delay(1000);
  }
}
```

Figure 5.1 combination code of 32 and 02

This code has both the OR and NOR put together as one. The main difference from this and the last is that both outputs will display at the same time in the void loop using the same for loop to implement it. \t separates both and give space for each to show.

3. 74HCT245

The 245 is known as the 8-bit transceiver with having a 3 state outputs. The outputs the 245 has is DIR which is for direction control and OE which sends and receives. How It works is that once OE is forced to be HIGH it will assume High impedance in off state. This is also known as a buffer will I explain down below.

Logic truth table, circuit diagram and schematic

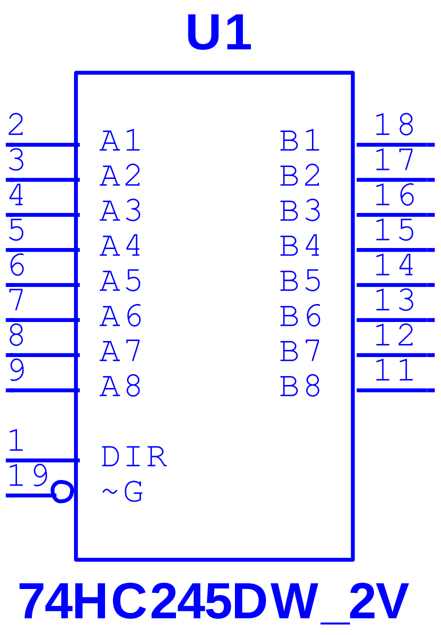


Figure 5.2 pin layout

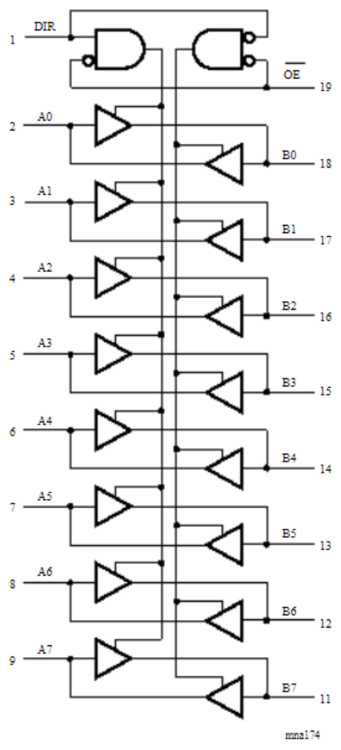


Figure 5.3 logic figuration from Texas instruments

Operation: DIR and OE bar are connected in order for the buffer to work. So in this case if A0 is a LOW then B0 will also be a LOW the other sided shown in figure 5.3 . if A7 is HIGH then B7 will also be HIGH.

OE	DIR	operation
L	L	B data to A bus
L	H	A data to B bus
H	X	Isolation

Figure 5.4 logic truth table

This table verifies exactly what I'm talking about. When both OE and DIR are LOW then its from B to A. when DIR becomes HIGH then it changes direction. And when OE is HIGH it will then isolate all pins.

```
#define A0 P2_4
#define B7 P2_5
#define OE P2_3
#define D P1_3
#define S0 P2_7
#define S1 P2_7

void setup () {
  Serial.begin(9600);
  pinMode(S0, INPUT_PULLUP);
  pinMode(S1, INPUT_PULLUP);
  pinMode(D, OUTPUT);
  pinMode(A0, OUTPUT);
  pinMode(B7, OUTPUT);
  pinMode(OE, OUTPUT);
  digitalWrite(OE, LOW);
}
void loop() {
  digitalWrite(A0, HIGH);
  digitalWrite(A0, LOW);
  digitalWrite(B7, HIGH);
  digitalWrite(B7, LOW);
  if (digitalRead(S0) == HIGH) {
    while (digitalRead(S0) == HIGH)
      digitalWrite(D, LOW);
      }
    if (digitalRead(S1) == HIGH) {
    while (digitalRead(S1) == HIGH)
      digitalWrite(D, HIGH);
  }
}
```

Figure 5.5 logic code for 245

As usual the pins are all initialized up top including D(DIR) and OE. I then set S0 and S1 as input-pullup because in this they represent a constant true or false. So, in this case we set OE as low outside the loop. Then inside we A0 and B7 HIGH and use an if loop to say if digital read SO is equal to HIGH, I use the while loop in the if loop to get S0 to equal to HIGH. And if that's the case then DIR is LOW. but if S1 is HIGH then DIR is HIGH.

4. 74HCT138

Logic operation of a 1 to 8 decoder

The 1 to 8 decoder is also known for having three weighted input binary address, Address 1, Address 2 , Address 3

These address liners go to 8 outputs(Y0-Y7)

This chip also features 3 enable inputs such as

- Enable (bar)1
- Enable(bar)2
- Enable 3
  All output will be HIGH unless Enable bar 1 and 2 are LOW and Enable 3 is LOW. This will allow easy parallel expansions.

Supply voltage Unregulated :	12 Volts
Supply Voltage regulated :	5 Volts

Supply Current: 20mA

Diagram and layout and truth table

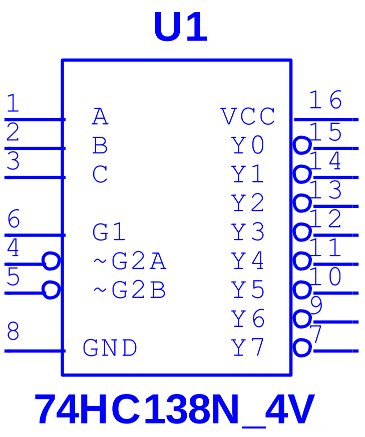


Figure 6 pin layout of 138

- X =doesn't care
- L = logic LOW
- H =Logic HIGH

Figure 6.1 functional table of the 138

INPUTS						OUTPUTS							
ENABLE			ADDRESS										
E1	E2	E3	A1	A2	A3	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	H	L	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	H	H	L	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

```
#define S2 P2_5
#define SW P1_3
int Ybar = 0;
void setup() {
  Serial.begin(9600);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(SW, INPUT_PULLUP);
}
void loop() {
  if (digitalRead(SW) == LOW)
  {
while (digitalRead(SW) == LOW) {}
Serial.println(Ybar);
digitalWrite(S0, bitRead(Ybar, 0));
digitalWrite(S1, bitRead(Ybar, 1));
digitalWrite(S2, bitRead(Ybar, 2));
if (Ybar < 7) {
  Ybar = Ybar++;
} else
```

- For this chip I set int Y-bar as 0 since we will use later in the code. Then i initialize inputs and outputs. In the loop we say if SW is equal to LOW, I then enter the while and say the same thing. If its true it will print out whatever Y bar is from S0 to S2. This increment once its less than 7 and if its bigger then go back to 0 which it will repeat the process.

5. 74HCT251

Logic operation:

The 8 to 1 multiplexer or MUX is actually a digital device in the chip that allows you to select one of the 8 inputs that will go to the one output line using a three-bit selection line.

Logic truth table and mech layout:

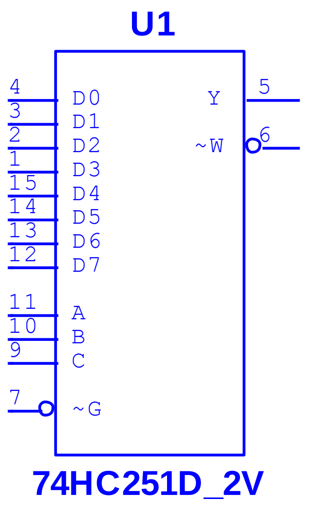


Figure 7 pin layout for 251

INPUTS			OUTPUTS
S2	S1	S0	O
0	0	0	i0
0	0	1	i1
0	1	0	i2
0	1	1	i3
1	0	0	i4
1	0	1	i5
1	1	0	i6
1	1	1	i7

Figure 7.2 logic table of operation

Logic symbol,voltages and kurnaph map:

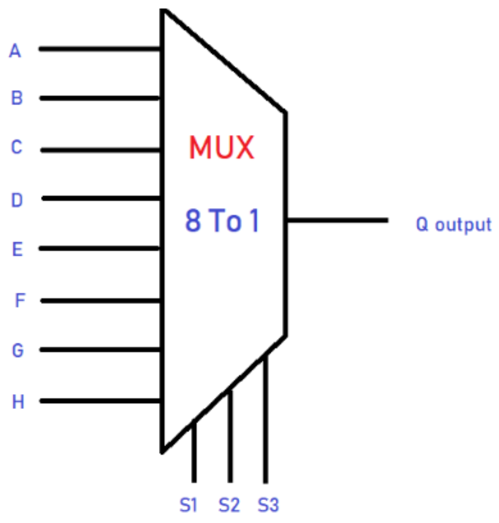


Figure 7.3 mux layout with sel 1,2 &3 from internet

Supply voltage Unregulated	12 Volts
Supply Voltage regulated	5 Volts
Supply Current	10mA

```
#define S0 P2_5
#define S1 P2_4
#define S2 P2_3
#define Y1 P1_7

void setup() {
 Serial.begin(9600);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(Y1, INPUT_PULLDOWN);
}
void loop() {
  for (int i = 0 ; i <= 7; i++) {
    digitalWrite(S0, bitRead(i, 0));
    digitalWrite(S1, bitRead(i, 1));
    digitalWrite(S2, bitRead(i, 2));
    if (digitalRead(Y1) == HIGH) {
      Serial.print("Pin ");
      Serial.print(i);
      Serial.println(" Is Aactive High ");
    }
  }
  Serial.println();
  delay(100);
}
```

Figure 7.4 code for 251

- The main things when it comes to this particular code is that we set y1 as a pulldown in pin mode and I use if to say if its less or equal to 7 it will then increment one each time. This will happen for each select line. Once Y1 is active HIGH it will print whatever value I is and print out the message active high in the serial monitor. This goes at a much faster rate about 1 milli second.

6. The 74HCT259

Logical operation:

The 259-logic gate is an 8-bit addressable latch. This chip has a enable (OE BAR) and also features a send and receive (DIR). The DIR is known for its direction control.

Once enable is HIGH it would cause the output to have a HIGH impedance OFF state.

Features and powering:

- Wide supply voltage ranges from 2 to 6 volts
- CMOS low power dissipation
- High noise immunity
- Supply Current 10mA
- Non-inverting 3-state outputs

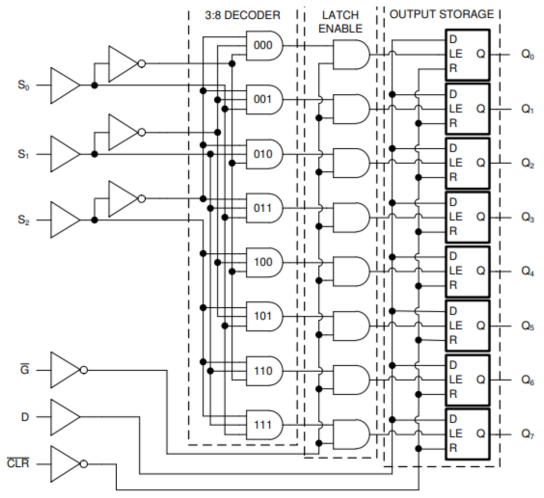


Figure 8 schematic of 74hct259 from Texas instruments

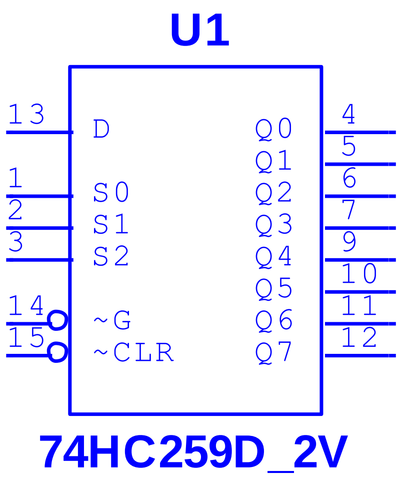


Figure 8.1 pin layout of the 259

Device function modes and truth table:

INPUTS		ADDRESSED LATCH	OTHER OUTPUT	FUNCTION
CLR BAR	G BAR			
H	L	D	Q10	ADDRESS LATCH
H	H	Q10	Q10	MEMORY
L	L	D	L	8 DEPLEXER
L	H	L	L	CLEAR

Figure 8.2 functional modes

H =high voltage level

L= low voltage level

Q^{10}= previous output state of a selected latch

INPUTS			ADDRESSED LATCH
S2	S1	S0	
L	L	L	0
L	L	H	1
L	H	L	2
L	H	H	3
H	L	L	4
H	L	H	5
H	H	L	6
H	H	H	7

Figure 8.2 logical truth table

H= High voltage level

L = Low voltage level

```
#define CLR P1_3
#define G P1_7
#define D P1_6
#define S2 P2_5
#define S1 P2_4
#define S0 P2_3
#define SW P2_7

int count = 0;
void setup() {
  Serial.begin(9600);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(CLR, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(D, OUTPUT);
  pinMode(SW, INPUT_PULLUP);
  digitalWrite(S0, LOW);
  digitalWrite(S1, HIGH);
  digitalWrite(S2, LOW);
  digitalWrite(CLR, LOW);
  digitalWrite(G, LOW);
  digitalWrite(D, HIGH);
}

void loop() {

   digitalWrite(S0, bitRead(count, 0));
   digitalWrite(S1, bitRead(count, 1));
   digitalWrite(S2, bitRead(count, 2));
    if (digitalRead(SW) == LOW) {
      count++;
      Serial.print("Pin ");
      Serial.print(count);
      Serial.println(" Active High ");
      delay(1500);
    }
}
```

Figure 9 first part of the 74hct 8 duplexer mode

- Outside the loop i set int count as 0. We declare all pins as outputs except for SW which is an input pullup. Then we set S0 to S2 as HIGH or LOW. For the CLR, G and D I want to enter 8 duplexer mode so the I set CLR as LOW, G as LOW and G as HIGH.

- In the loop Its using digital write for S0, S1, and S2. Like in any other code i use the if statement and say if SW is equal to LOW increment by 1 and print out pin and whatever count is. When its HIGH it will say active HIGH. I delayed this by 15 milli seconds.

Bill of materials

component and tools	item no	designator	Quantity	supliers	product code	cost	
vera board	N/A	ltd	1	RK Education	B08D2HP4B3	€5:73	
74hct245	62S8876	N/A	1	Texas instruments	4.10E+12	€4:28	
74hct138	KY32	N/A	1	Kynix	N/A	N/A	
74hct02	CD74HCT02E	N/A	1	Farnell	1105963	€0.50	
74hct32	62S8772	N/A	1	burklin	4.10E+12	€4:82	
74hct251	252-0656	N/A	1	Texas instruments	N/A	€0.94	
74hct259	85X2830	N/A	1	Newark	N/A	€0.60	
7 segment display	TDSG3150	N/a	1	Farnell	2748728	€2:30	
IC holders	N/A	N/A	6	eBay	N/A	€9:14	
Soldering iron	40326	N/A	1	screwfix	N/A	€11:50	
soldering flux	159120	N/A	1	plumnation	5.05E+12	€28.75	
Snippers	S-TOL-10447	N/A	2	Robosavvy	N/A	€22:88	
soldering stands	N/A	ltd	1	amazon	N/A	€13:80	
track cutters	N/A	N/A	1	manomano	ME14879644	€6:90	
desoldering pump	N/A	N/A	1	Hobbycraft	6219031000	€7:50	
MSP430	MSP-430G2ET	N/A	1	Texas insruments	N/A	€9:18	
Ribbon cable	R2651HTSY10SC85	ltd	1	Multicomp pro	2628357	€18:82	
header pins	826629-8	N/A	3	Farnell	3418364	€1:50	
PCB board	N/A	N/A	1	JLC PCB	N/A	€50:00	
PCB holder	N/A	N/A	1	grandado	8.72023E+12	€48:19	
						€247.33	Total

Multisim

In the beginning of this project work, the student must design a schematic of the pcb design using header pin, busses and correct pins. It is also important to know that each of the student progress must be saved in case of a slight error which could cause problems in the later stages of the design.

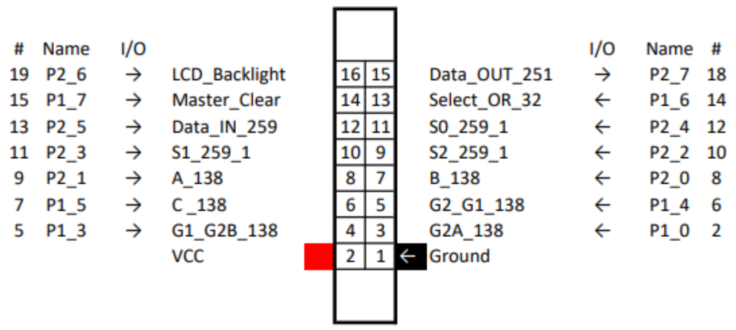
The step goes as follows:

First step on Multisim is go to place, then click on component. Once clicked, go to all groups and search for HDR which is short for header pins. We will need 4 of these

- J1 1X10
- J2 1X10
- J3 2X8
- J4 1X2

Second step is getting the busses by moving the mouse to place again and clicking on buss. Once clicked click the screen and guide the bus in where and how you want as long as it's around the HDR.

The third step is very important. In order to connect the HDR to the buses the student must go to the pin layout and see what pins are which.



#	Name	I/O					I/O	Name	#
19	P2_6	→	LCD_Backlight	16	15	Data_OUT_251	→	P2_7	18
15	P1_7	→	Master_Clear	14	13	Select_OR_32	←	P1_6	14
13	P2_5	→	Data_IN_259	12	11	S0_259_1	←	P2_4	12
11	P2_3	→	S1_259_1	10	9	S2_259_1	←	P2_2	10
9	P2_1	→	A_138	8	7	B_138	←	P2_0	8
7	P1_5	→	C_138	6	5	G2_G1_138	←	P1_4	6
5	P1_3	→	G1_G2B_138	4	3	G2A_138	←	P1_0	2
			VCC 🟥	2	1	← Ground			

Figure 1 shows pin layout

As shown in the figure above me of the connector's layout diagram, we can see that VCC and ground are 2 and 1. They are fixed and cannot change at all. Same with the other pins like S0, A, LCD backlight etc. say if the pins were wrong and sent out and came back, we would have to make modification by making connections to the right pins which would be a lot of hassle and would not be neat and presentable. This will be the students J3. The pin numbers on each side will be a great significance in this project which will talked about later.

The fourth step is to now connect every pin in each HDR to the buses with each connection having a label so the student knows where the connections are going to and say if there were a connection error we could trace down the fault.

Fifth step is to then go back and triple check everything and seeing if there bad connections, unnecessary connection's or even a double one. Always check if the pins are correct and especially VCC and GND

Once everything had triple checked and certain the final schematic should look like this before moving on .

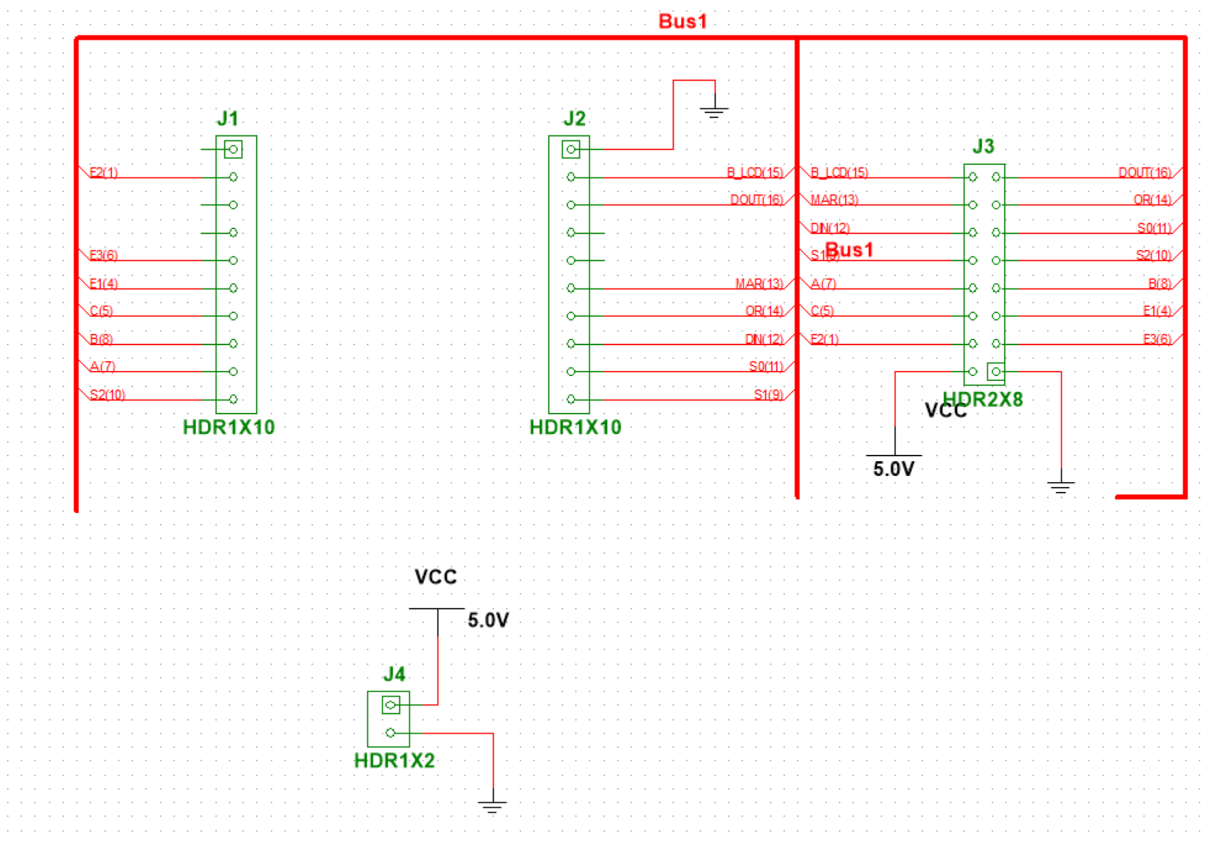


Figure 2 finished schematic of PCB design

The student must then save this because this could be used as a reference to look back on when taking the design one step further. It's very important point to point out since it's the foundation of the board design. If the students have any errors and slight changes to make, it would recommended to do it in the early stage rather than the late stage .

Ulti-Board

This part of the project is very important since it will be sent to manufactured and sent exactly the way the student designed it. This part be explained to fullest to show what the student did in order to complete it a correct professional manner.

Once the student has the schematic saved and backed up for reference student must then open up ulti-board.

NI ulti-board is an electronic printed circuit board which has a layout software program that integrates with Multisim. Its feature are great for engineers around the globe. One such feature is the Real time design rule check and was offered on expensive work stations. It was created by Ultimate technology.

Figure 3 Ulitboard logo from bucknell website

With great tools to use from Ulti-board, the student can transfer the schematic from Multisim to Ulti-board. To do so the student must move the mouse to transfer. It should give the options to transfer as long as it was saved.

PCB unrouted

Once the entire schematic has been transferred it will appear outside the yellow square on the top left.it will be the student's job to move it to the yellow square by dragging it and figuring out the layout of design. The yellow square is known as the board outline and be changed to any size possible. The aim of not just this project but in the world of engineering is to make things as small as possible. The minimum board outline was to be about 90mm both height and length

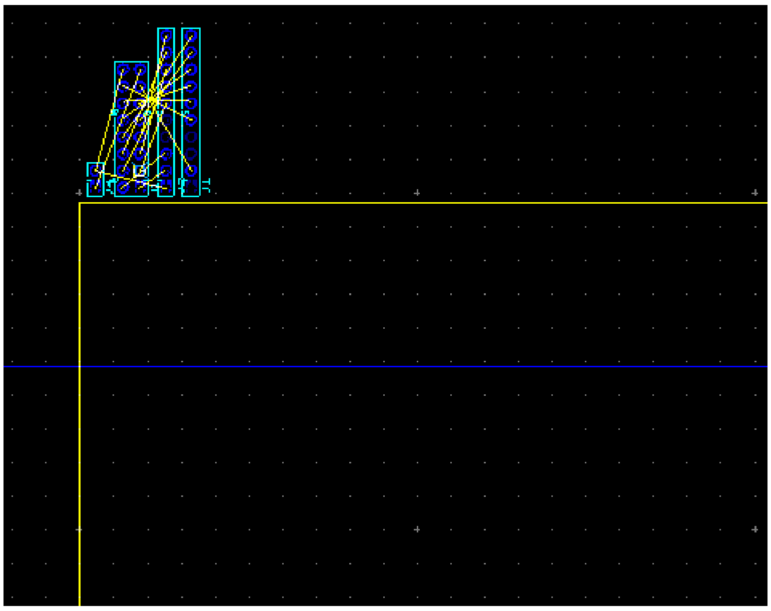
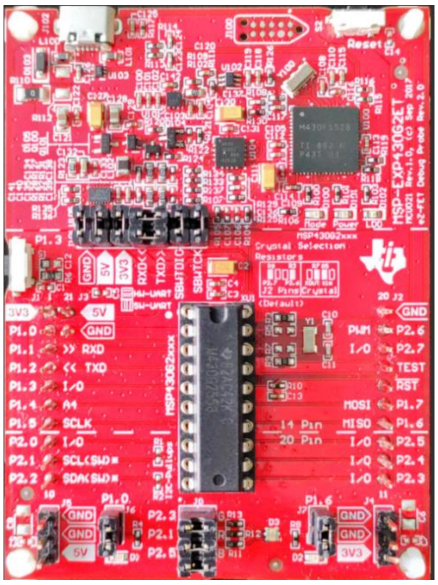


Figure 4 schematic of PBC transferred

In order for the student to position where each component will go the student must first know that the PCB will be connected under the MSP430.



This is where proper and carful measurements come in

Figure 5 shows MSP430

- The length between J1 and J2 centre pin to centre pin is about 46.2 mm
- J4 should be closed in on the J1 leaving no spaces
- The Measurement between the gaps the 3 components above and the J3 which the student had decided to be vertical should be about 18.2mm in spacing.

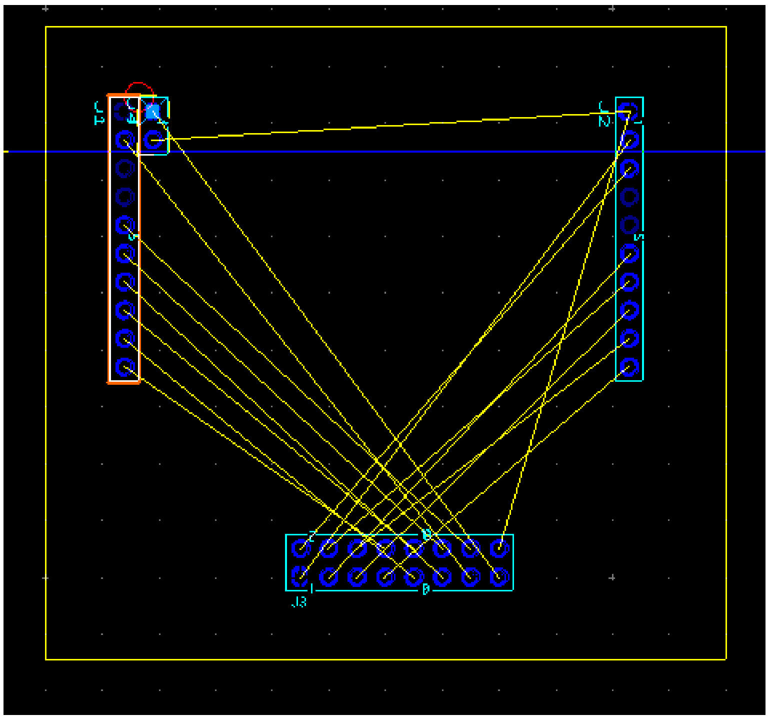


Figure 6 rough layout of components according to measurements

Once at this stage is very important to confirm that everything is correctly measured by triple checking it again and making sure the components are the right way around and there are no errors showing. The only error that’s meant to show is the J4 and J1 which are close to each other. This is completely fine as these are intended for the connection of the MSP430.

Next thing to check with be the Nets by following each line and checking to see if there connected or are going to correct net. What happens is that say if the student assumed that the net was all perfect and sent them of to be manufactured and they came with errors on the board, the student would then have to modify the hardware and software and worst-case scenario it was VCC and GND. Cheeking back to the schematic from Multisim is a big must as its where you can find faults and errors and suggestions along the way then later as said before.

So, its clear that even nets must be cheeked once something is moved or added to the design. At this point once everything is checked, the student would then have to save this file as PCB unrouted since it’s a backup in case the student can go back to it figure an alternative.

PCB routed

This part has many key aspects to pay attention to such as:

- Copper top and copper bottom
- Trace width

- Trace clearance
- Final board outline
- Net clearance

After saving the unrouted, the student then clicked autoroute. This shows the copper screen on the board and shows the clear connections.

Copper top and Bottom

In every PCB design in the world there 2 layers of the PCB. The top and bottom. This design is very useful for engineers when it comes to complex designs that involve a lot of tracks and connections. Say for instance if the PCB design was very limited on space and had no more room for a few connections, then the top or bottom layer would be used as well.

Here is an example from Ulti-board that was used to help design the PCB

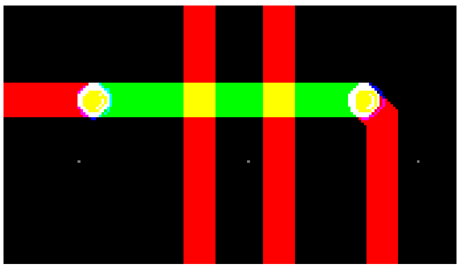


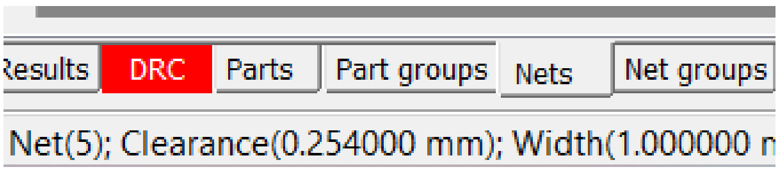
Figure 7 PCB copper tracks from Ulti-board

Here you can see that tracks are going over or through each other but that not the case. It actually over each other but do not have contact with each other. From this view we can see that the green is the copper top and the red is the copper bottom.

In this figure it is used to get around areas of the PCB which would make it easier to design. Although there are no shorts, it can actually connect to the copper bottom at each end.

Trace width

The trace width needs to be big enough to see where the copper tracks were going. Remember this is mm we are talking about so when its first auto routed its really tiny and will be easy to see. In order to set the right amount of trace to become visible enough, the student clicked on the net tab.



Net(5); Clearance(0.254000 mm); Width(1.000000 n

Figure 8 shows tabs for PBC settings

Once clicked on it should show every single net .if you click on a net it will show what net is connected to which.

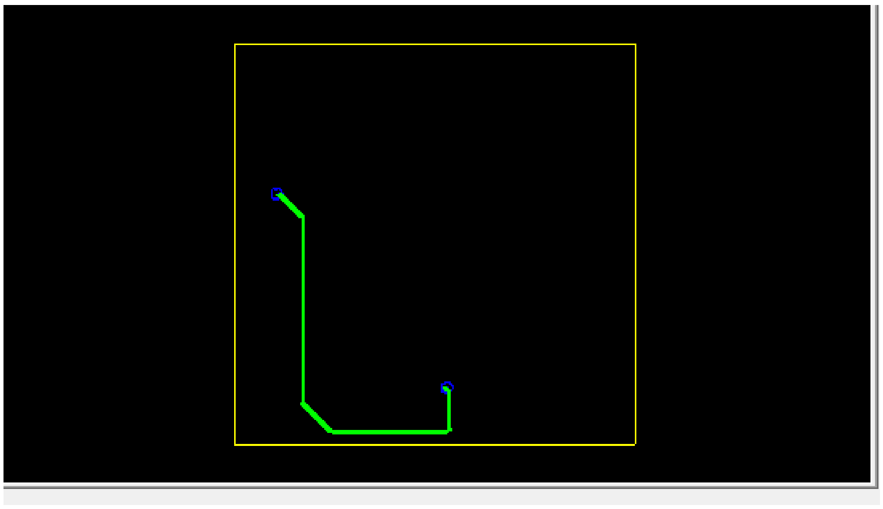
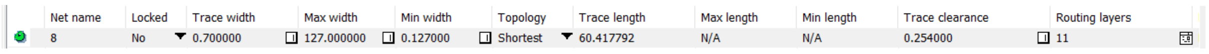


Figure 9 displayes the net selected

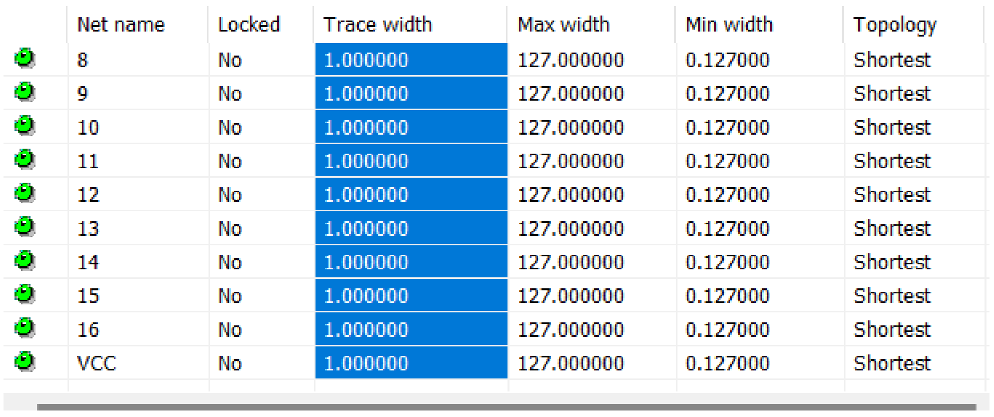
Table 2 shows the net clicked



	Net name	Locked	Trace width	Max width	Min width	Topology	Trace length	Max length	Min length	Trace clearance	Routing layers
●	8	No	0.700000	127.000000	0.127000	Shortest	60.417792	N/A	N/A	0.254000	11

There are 2 ways of actually making the trace width bigger. The first way is by doing it automatically. To do this click on the trace width tab above and the entire table should go blue. All the student has to do is to set what ever number (in this case the student put in 1.0000000) this will automatically set all to that same exact number.

Table 3 shows all nets selected and set as the same number



	Net name	Locked	Trace width	Max width	Min width	Topology
●	8	No	1.000000	127.000000	0.127000	Shortest
●	9	No	1.000000	127.000000	0.127000	Shortest
●	10	No	1.000000	127.000000	0.127000	Shortest
●	11	No	1.000000	127.000000	0.127000	Shortest
●	12	No	1.000000	127.000000	0.127000	Shortest
●	13	No	1.000000	127.000000	0.127000	Shortest
●	14	No	1.000000	127.000000	0.127000	Shortest
●	15	No	1.000000	127.000000	0.127000	Shortest
●	16	No	1.000000	127.000000	0.127000	Shortest
●	VCC	No	1.000000	127.000000	0.127000	Shortest

The manual way of doing it is by going to the follow me icon on the top right-hand side of the screen.

Figure 10 the follow me icon

After clicking it, the student can manually move and adjust the tracks with the mouse. The benefit of using the follow me icon is that its flexible and can be moved around for the designer's pleasure. Wherever the mouse goes the copper track will also follow.

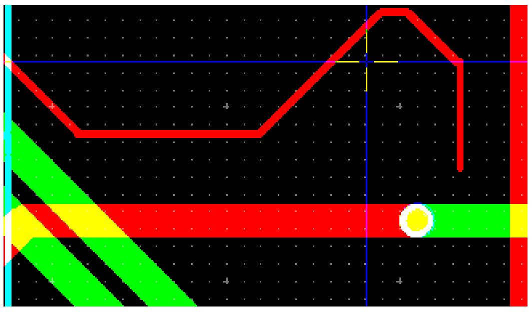


Figure 11 manual connection with the follow me icon

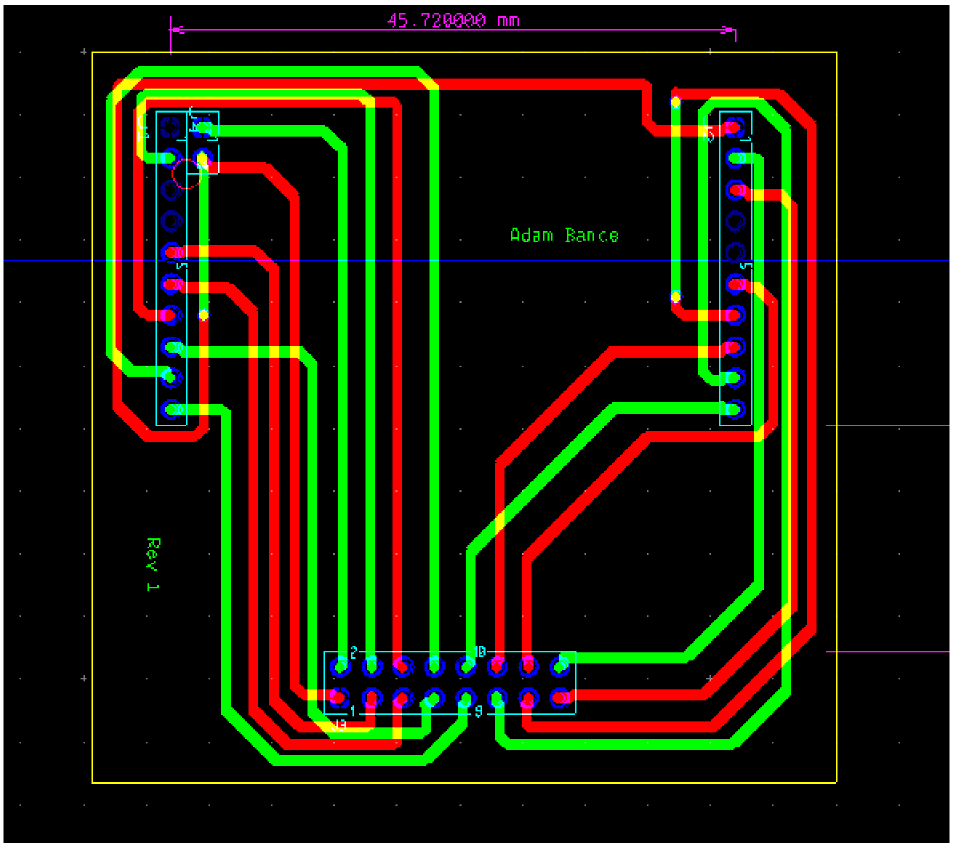
Trace clearance

Trace clearance is very important for the design and place a big part in all designs. The reason is because once it comes back from the manufacturing company, the last thing you want is to tracks to short circuit each other. So, it's always nice to leave clear space between the tracks. All tracks should get equal amount of clearance if possible but also bearing in mind that there is not too much space for when condensing down the board to make smaller. It was agreed with the student and his colleges that the default trace clearance was way to small and would come back with a lot of errors. The agreed trace clearance was then set 0.5mm each

Final board outline

Over the few attempts of designing the PCB on Ulti-board, the student managed to fit the board down to about 57.6mm in height and 57.6mm in length. The problem with going any smaller would be that the tracks would be way to close to the edge of the PCB design. At this stage the student could possibly move a few tracks inside the board to try and condense it even smaller without them being to close to each other of the edge of the board.

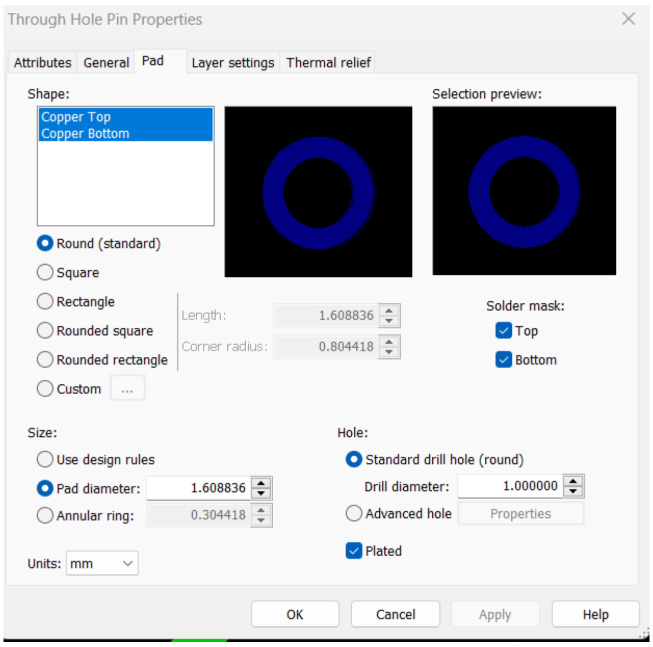
Inside the board the student wrote his name and versions with the copper top to identify his piece of work.



Hole pin diameter

The last thing checked was the diameter of the pin holes. This is again crucial as if it's too small the Pin will not fit through at all but if its big it won't be perfect either but can filled with solder in order to connect the pins. The student left clicked into each pin, then left clicked to go into properties.

Once clicked onto it should show options to change pin hole in different ways but in this case the student only wants to change only the diameter spacing.



Through Hole Pin Properties

Attributes | General | Pad | Layer settings | Thermal relief

Shape:
Copper Top
Copper Bottom

[✓ Round (standard)] Square Rectangle Rounded square Rounded rectangle Custom

Length: 1.608836
Corner radius: 0.804418

Selection preview:

Solder mask:
[✓] Top
[✓] Bottom

Size:
Use design rules
[✓ Pad diameter:] 1.608836
Annular ring: 0.304418

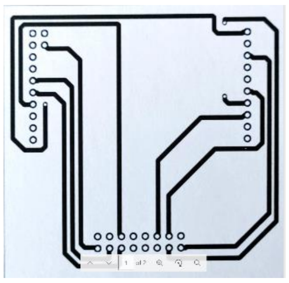
Units: mm

Hole:
[✓ Standard drill hole (round)]
Drill diameter: 1.000000
Advanced hole Properties
[✓] Plated

OK | Cancel | Apply | Help

Figure 12 settings for net 7 pin of the j3

As shown on the figure above the student set the pin diameter to 1.00mm. although its not an exact accurate measurement of the pin hole it is still better than a smaller one.



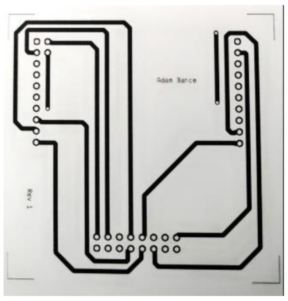


Figure 13 copper bottom print

Figure 14 Copper top printed PCB

After every net and been set to 1.00 mm, The student goes back through everything and triple cheeks again as this is the final step before it will be set off to the manufacturing company. For the final inspection the student would print of the PCB design and just about hold it underneath the MSP to see if it fits perfectly fine.

Once all checks have been conducted it is then submitted as a file to the lectures link

Interfacing MSP430 to processing

The objectives for this chapter goes as follows:

Objectives

Recap on Processing from semester one.

How to import and use a Library within Processing programs.

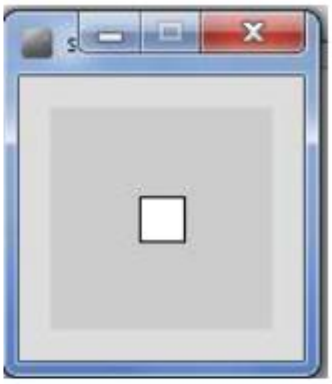
Write MSP430 code to Receive a data byte from the Serial monitor.

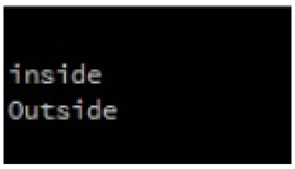
Create a simple GUI to transmit a data byte from Processing to the MSP430.

Implement feedback between the Processing Sketch and MSP430.

The student must recap on previous programs or basics programs that were created in processing in semester one. This will be used to test the output of the PCB when it arrives from the manufacturing company. To start off with the basic the student should implement a code to draw a square.

The first thing to do is create a window with 100 X 100 pixels. Then to create a square with 20 X 20 pixels. The last thing is print out inside and outside with the void mousse-pressed function. Once everything is all done the student must then cheek it before running the code. When running the code, it should look like this:



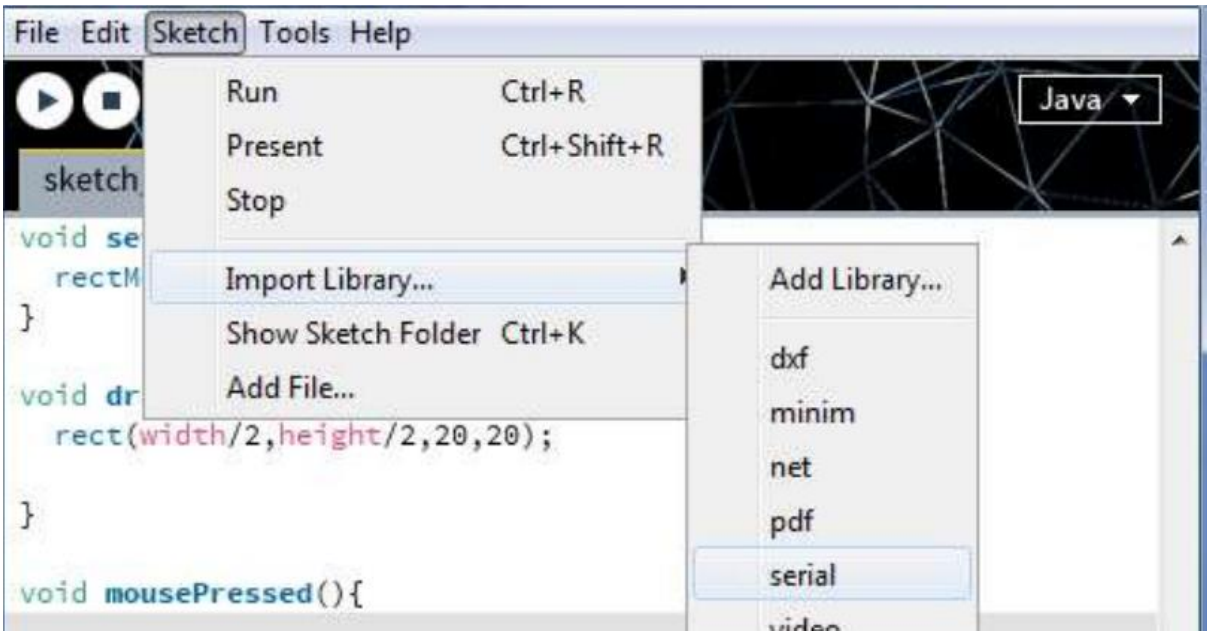


```
inside
Outside
```

Figure 15 displays the printed message by the void mouse pressed

Figure 16 shows windows and square created

To access the library the student must go to sketch, then go to import Library. It should give an option for serial which will help us connect and receive data to the MSP430 .



```
void se
  rectM
}

void dr
  rect(width/2,height/2,20,20);

}

void mousePressed(){
```

Figure 17 importing the serial into the program

Processing code #1

```
import processing.serial.*;
Serial commPort;



void setup(){
  rectMode(CENTER);

  println(Serial.list());
  commPort = new Serial(this, Serial.list()[1],9600);
}
void draw (){
  rect(width/2,height/2,20,20);
}
void mousePressed(){
  if((mouseX > 40 && mouseX < 60)&&(mouseY > 40 && mouseY <60)){
    println("inside");
    commPort.write('1');
  }
  else{
    println("Outside");
    commPort.write('2');
  }
}
```

Next would be to write the code for Energia using the MSP430 .For this the student would need to set incoming bytes as 0.

MSP code #1

```
int incomingByte = 0;
void setup(){
  Serial.begin(9600);
  pinMode(2,OUTPUT);

}
void loop() {
  if(Serial.available() > 0) {
    incomingByte = Serial.read();
    if(incomingByte ==49){
      digitalWrite(2,HIGH);
    }
    else{
      digitalWrite(2,LOW);
    }
    Serial.print(" I received: ");
      Serial.print(incomingByte,DEC);
      Serial.print("\t");
      Serial.print(incomingByte,HEX);
  }
}
```

Once executed, open up serial monitor and it should show the bytes received. It is important to make sure that line ending is selected in the serial monitor. If a 1 is sent the red LED should be on and if another character has been sent it should be turned off.

Processing and MSP430

Moving into the next part is to sketch processing and MSP together. This is done by keeping the code from Energia and by little changes to the processing code. The idea is to make the LED go on when the mouse clicked inside the square and when it clicked outside the square it should turn off. In order to face error, the student need to make sure that serial monitor is not running which could Cause the program to not run at all. Once all this is done correctly, the program should work and do what's it's supposed to do.

GUI

The last part was getting feedback from the MSP430.

On processing the student drew a new square with a new code. In the void draw section, the student set rec as ("commPort.read") and used if else statements to determine if the LED is on or off with true and false. At void mouse pressed function depending on where the mouse is like bigger or smaller then the pixel number it will print that its inside. If not then its outside. All this used by the if, else statement.

- For the MSP430 the student used a lot more if, else statements to say if this number is high or low then then serial. Write it as 1 or a 0 of course depending on what number was Entered. This section of the code is sent data only when you received data. It will then read the incoming byte and when done correctly should be turning the LED on as well as the student clicking the mouse in the square to change colour. If its white then its off and if its red its on. The MSP430 should be responding to this.

Processing code #2

```
import processing.serial.*;

Serial commPort;
int rec = 0;

void setup() {
  rectMode(CENTER);
  println(Serial.list());
  commPort = new Serial(this, Serial.list()[1], 9600);
}




void draw() {
  if (commPort.available () >0) {
    //int rec = 0;//


    rec=commPort.read();
    if (rec == 48) {
      // LEDstatus is false
      fill(255);
    } else {
      //LED is true
      fill(255, 0, 0);
    }
  }
  //println(LEDstatus);
  rect(width/2, height/2, 25, 25);
}

void mousePressed() {

 if ((mouseX > 38 && mouseX <61)&&(mouseY >38 && mouseY <61)) {
    println("inside");
    commPort.write('1');

 } else {
    println("Outside");
    commPort.write('2');
  }
}
```

MSP code #2

```
int incomingByte = 0;

void setup() {
  Serial.begin(9600);
  pinMode(2, OUTPUT);

}
void loop() {

 if (Serial.available() > 0) {
    incomingByte = Serial.read();

 if (incomingByte == 49) {
      digitalWrite(2, HIGH);
      digitalWrite(2, LOW);
      Serial.write("0");
    }
    else {
      digitalWrite(2, HIGH);
      Serial.write("1");
    }
  }
  else {
    digitalWrite(2, LOW);
    Serial.write("0");
  }

}
}
```

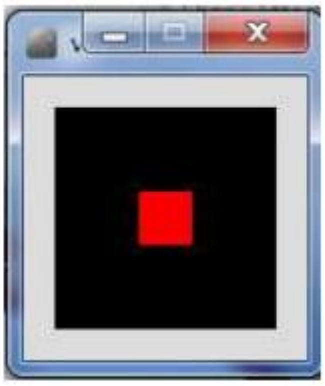


Figure 18 indicate that The LED is on

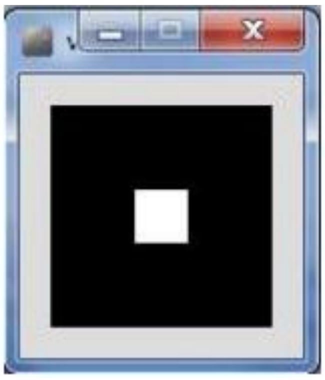


Figure 19 indicate that LED turned off

Using feedback from MSP430

From this progress the student knows how to make one box turn off and on. Now the student needs to make 15 more boxes as inputs. Remember the aim by the end is have 16 boxes so that the student can test the output by clicking on them so that it can turn on an LED.

In order to do this column and row must be initialized along with r and c which be used for a loop later on in the code to create the rest of the squares.

In this code there should be strings with the names to placed in each box. Each box a default colour as white. Depending where the mouse is clicked it will know that a square has been clicked. Once it knows it should change colour of the particular box and print out the name of the box that has been pressed. The final code should look like this.

Processing code #3

```
int col=0;
int row=0;
int c=1;
int r=7;

String boxNames[][] = {{"LCD_Backlight", "MCLR", "DA_IN", "S1", "A", "C",
"G1", "VCC"}, {"DATA_OUT", "SEL_OR", "S0", "S2", "B", "G3", "G2", "GND"}};


void setup() {
  size(500, 500);
  background(255);
  textAlign(CENTER, CENTER);
}

void draw() {
  background(255);
  drawBox(1, boxNames[0].length - 1);
}

void drawBox(int c, int r) {

  for (col=0; col<=c; col++) {
    for ( row =0; row<=r; row++) {
      fill(255);
      rect(col*100, row*50, 100, 50);// draws each box

      fill(0);// sets the colour of the text(black)
      text(boxNames[col][row], 50+(col*100), 25+(row*50)); // draws the
text in each box
    }
  }
}

void  mousePressed() {
  for ( col=0; col<=c; col++) {
    for ( row =0; row<=r; row++) {

      if (mouseX>col*100 && mouseX<((col*100)+100) && mouseY>row*50 &&
mouseY<((row*50)+50))
      {
        println(boxNames[col][row]);
      }
    }
  }
}
```

Surface mount technology

In this part of the project the student learns the importance of surface mount technology and is able to perform carful operations to any design made. Have such a great skill from this can really benefit the student in the long run, especially in projects in future jobs and etc. To understand surface mount technology the student must know what it is.

What is Surface mount technology?

Surface mount technology is considered to be a component assembly technology that would be related to printed circuit board where there are components mounted to the board and also connected together. The method is slightly different than other PCB design methods as surface mount.

These methods include:

- Inserting the component leads in the holes
- Soldering from the bottom to fill up the holes
- Interconnecting the components

Soldering in surface mount technology

During the soldering process its important that the student understands the safety precautions while conducting this activity.

The equipment used where:

- Portable PCB holder: used to hold the printed circuit board while modifying or adding to the circuit.
- A solder fan: used to blow away the fumes.
- Soldering Lead: used to weld the component and the board together
- Sponge: used to clean the tip of the soldering iron
- Solder holder: used for holding the solder iron in place once finished
- Soldering pump: to help remove a component
- Soldering iron: the tool used to fix or place components to the circuit board

Operation

At this stage the PCB were sent back from the manufactures with unknown errors. It was recommended to conduct a continuity test on the PCB before any modifications could be carried out. This testing process will give assure the student on what needs to be change if there were an error and how then figure out how to modify it with surface mount.

The header pins we given out by the lecture Tom once the continuity test had been properly done and conducted. The pins were all in one line and had to be broken up into the appropriate size and numbers to fit it the PCB. Wire snippers were used to break it up neatly by apply pressure on the middle week part of the component.

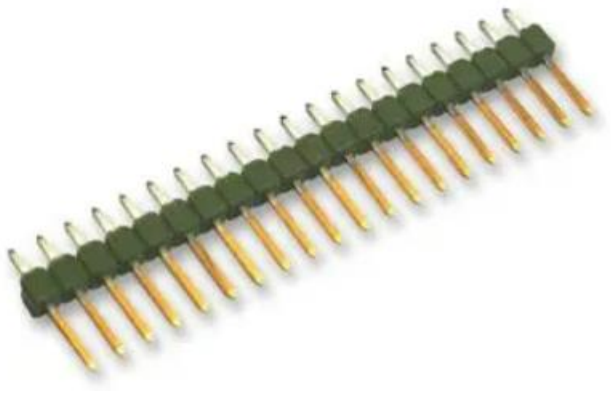


Figure 20 image of header pin from farnell website

The first pins to place were the j1 and j2. The student needed to assure that there would be no unnecessary movement with the operation of the soldering so the student used tape and used it to hold the header pins in place. Once the Soldering stand and PCB were steady the student then used some lead and heat to begin soldering. The student went about doing by first soldering the first end of the pin to the last end of the pin. This would assure that the pins were fixed of right and were neatly fixed to the board.

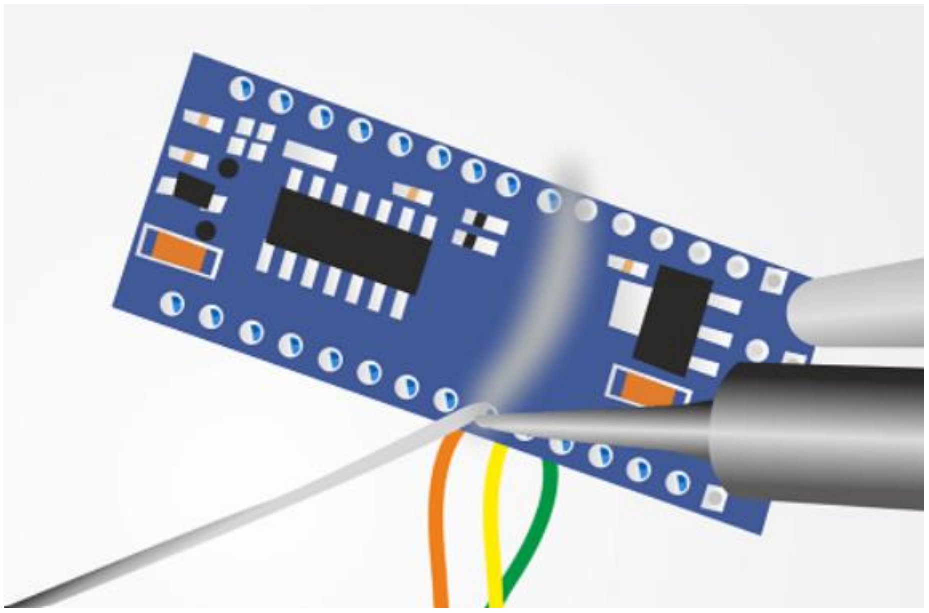


Figure 21 displays an example of surface mount by electronic hub

Once the J1 and J2 were added perfectly to the PCB, J4 and j3 were added next. Just like the J1 and J2 tape was used to hold the components to the board until they were soldered perfectly.

After completing and soldering all components together the student then showed it to the lecture tom for approval to see if any pins need more solder or need to be resoldered.

Switches matrix

Objective

In this chapter the objective was to get the student to turn on a single bit and then turn on a byte using the modes memory, latch, reset and demultiplexer through the chips of the 74HCT138,74HCT245 and the 74HCT259.

Understanding of operation

In order to understand how to turn on a single bit, the student must first understand what each chip actually does.

The first chip used is the 138 decoder. It has its address (A0-A2) , its enables (E1-E2) and its 7 outputs (Y0-Y7). The address represents the streets and the enables represent the town. There are 2 of these chips in the mother board. The top 138 is (1) and the bottom 138 is (4). This will simply tell the student which set of LEDS will activate and turn on.

The 259 has its selects (S0-S2) which is the house. It also has the pins G , Data_In and master clear. Through this the student can figure out what modes are which and what they should do.

LEDS

On the motherboard itself there are 20 LEDS top and bottom. they are divided into 8 LEDS that make up a byte. The last 4 can still be operational depending on the piece of code that will make it turn on. Each of these 8 sets of LEDS can be traced back to one of the 259 chips on the motherboard.

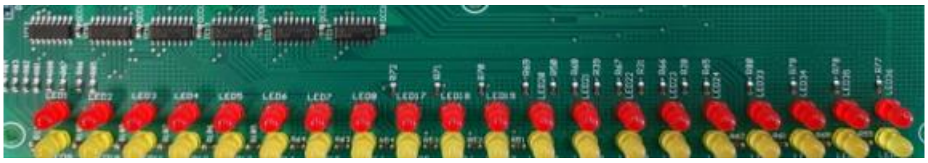


Figure 22 LEDS and 259 chips on motherboard - TU d Moodle

Once there is an understanding of chips the student begins the software part to drive on a single bit

Software code for motherboard

since the student will use a lot of defining and loops for turning on a single LED a tab is needed to separate them so that its clear to student and its easy to fix if there are errors.

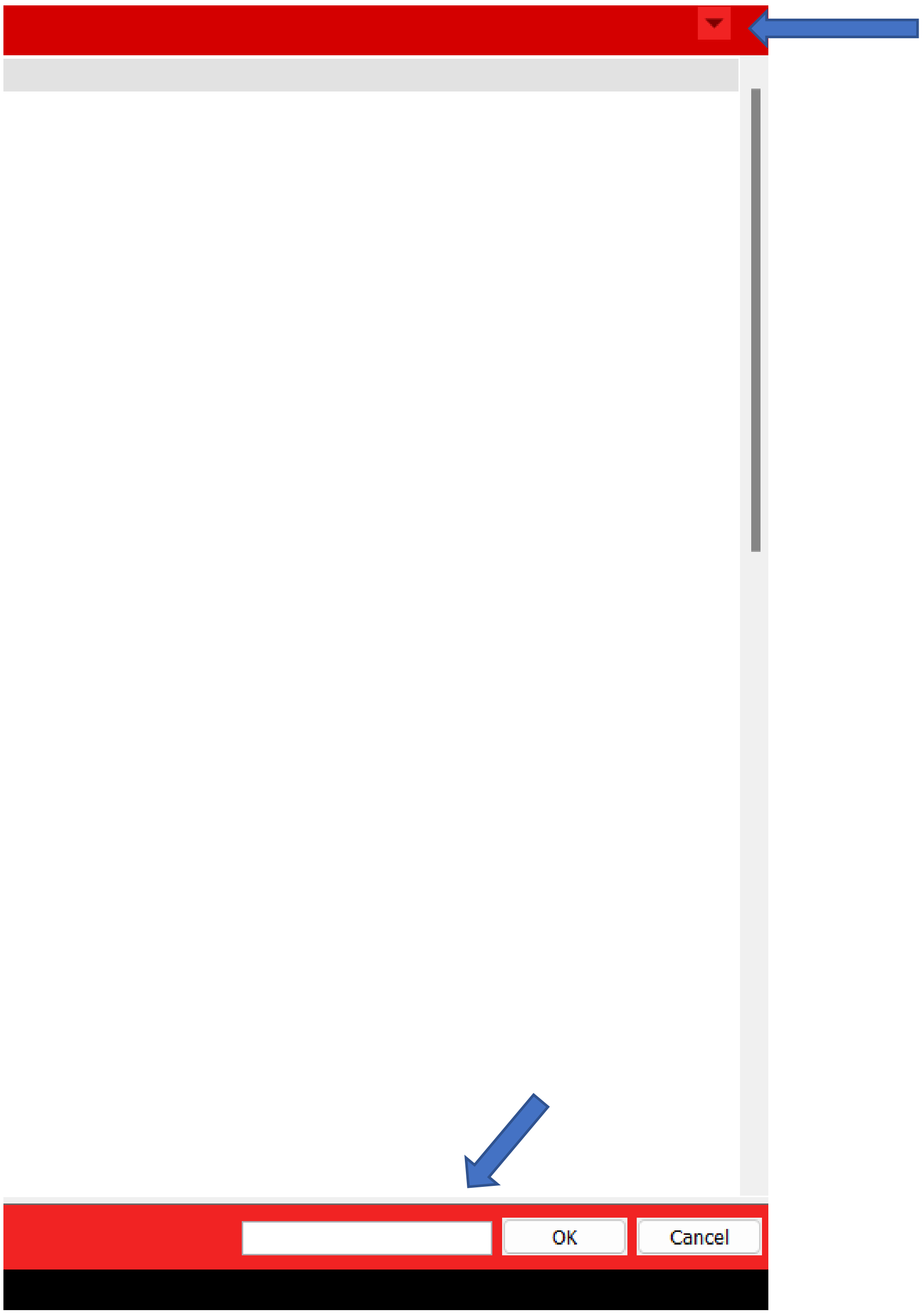


Figure 23 showing a tab being created

The student made this tab MSP430IO.H which will connect to the main code on a separate tab. The student was original meant to create tables for the house, town and streets but due to an error, all had to be cut and pasted to the main code. This will be explained in the PCB errors section.

Turning on a bit

```
void loop() {
  flashPIN();
  en(4);
  adr(0);
   sel(3);
   mode(3);
   digitalWrite(Data_IN_LOW);
```

this piece of code above turns on a single LED which in this case a Bit. As shown the enbles is 4 which means its bottom row, the streets are set 0 telling me which set of LEDS on that row, the sel is 3 which is the house and mode 3 is memory mode after concluding it. This should turn on the 4th LED from the bottom row first sets of LEDS

Turning on a byte

```
void WriteByte (int IC, int Y, int VAL ) {
  for (int S = 0; S <= 7; S++) {
    sel(S);
    digitalWrite(Data_IN, bitRead(VAL, S));
    en(IC);
    adr(Y);
    clockEnable();
    //delay(250);
  }
}
void clockEnable() {
  digitalWrite(Sel_OR, HIGH);
  digitalWrite(Sel_OR, LOW);
}
WriteByte (4, 0, 255);
    delay(250);
  }
```

After turning a bit, the student was able to figure what was need to turn on a byte. From the code fragment above we see that a function was need called write a byte with IC, Y , and VAL. Clock enable is used to set SEL or high and low. Once all factions were in place the numbers were placed into the bracket. The 4 is the top LEDS, the 0 means the first set of 8 and 255 meaning putting 1s all 8 LEDS since 1,2,4,8,16,32,64,128 add up to 255.

```
for (int count = 0; count <= 7; count ++) {
    WriteByte (4, 0, count);
    delay(250);
  }
```

This code fragment is simply using loops to turn on all 8 LEDS. instead of 255 its count which will go through states to flash on LEDS

Full code #1

```
#include "MSP430IO.H"
#define testPIN S1

void setup() {
  pinMode(E1, OUTPUT);
  pinMode(E2, OUTPUT);
  pinMode(E3, OUTPUT);
  pinMode(A, OUTPUT);
  pinMode(B, OUTPUT);
  pinMode(C, OUTPUT);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(Master_C, OUTPUT);
  pinMode(Sel_OR, OUTPUT);
  pinMode(Data_IN , OUTPUT);


  // Wait for the software reset to occur
 // while (255);
unsigned char WriteByte = 0xFF; // Initialize byte to 0xFF (all bits set to
1)

  WriteByte = 0x00; // Clear byte to 0 (all bits set to 0)

}
void WriteByte (int IC, int Y, int VAL ) {
  for (int S = 0; S <= 7; S++) {
    sel(S);
    digitalWrite(Data_IN, bitRead(VAL, S));
    en(IC);
    adr(Y);
    clockEnable();
    //delay(250);
  }
}
void clockEnable() {
  digitalWrite(Sel_OR, HIGH);
  digitalWrite(Sel_OR, LOW);
}
void sel (int num) {
  // for( int num = 0; num <= 7; num ++){
  digitalWrite(S0, bitRead(num, 0));
  digitalWrite(S1, bitRead(num, 1));
  digitalWrite(S2, bitRead(num, 2));
  //delay(500);
  //}
}
void en (int num) {
  // for( int num = 0; num <= 7; num ++){
  digitalWrite(E1, bitRead(num, 0));
  digitalWrite(E2, bitRead(num, 1));
  digitalWrite(E3, bitRead(num, 2));
  //delay(500);
  //}
}
void adr (int num) {
  // for( int num = 0; num <= 7; num ++){
```

```
  digitalWrite(A, bitRead(num, 0));
  digitalWrite(B, bitRead(num, 1));
  digitalWrite(C, bitRead(num, 2));
  //delay(500);
  //}
}
/* mode 0
    mode 1 is demultiplex
    mode 2
    mode 3 is memory
*/
void loop() {
  //flashPIN();
  //en(4);
  //adr(0);
  // sel(5);
  // mode(3);
  // digitalWrite(Data_IN_LOW);
 // for (int count = 0; count <= 7; count ++) {
    WriteByte (4, 0, 255);
    delay(250);
  }
//}
```

Full code #2

```
#define LCD_B P2_6
#define Master_C P1_7
#define Data_IN P2_5
#define S1 P2_3
#define A P2_1
#define B P1_5
#define E2 P1_3



#define Data_OUT P2_7
#define Sel_OR P1_6
#define S0 P1_4
#define S2 P2_2
#define B P2_0
#define E3 P1_4
#define E1 P1_0
```

This code fragment above is the define tab which defines all pins

Test procedure for the prototype

In the testing stage the student first did a power and Ground check to see if there were no short circuits and see if the supplied voltage was coming into the pins. The power supply was used in case to conduct a voltage test on the vera board. The reason why we used the power supply first instead of the MSP430 was because if there was a short circuit there would be no or little time before the port gets destroyed and blow up whereas if I were to use the power supply there would be time to react and there would be an indication in which the mA unit would flash to let us know that there is a short in the board. The test procedure goes as follows

. Step 1: would be to connect a wire from the power supply to the vera board.

Step 2: than set the power to about 5 volts or 3.3 and set the mA to 0.20 and flip the switch and if their mA unit doesn’t flash then it would conclude that there no short circuit and further testing can be done.

Step 3: use the multi meter set it to volts and set the range to 100volts.Then you the red and black probe to first Verify that Ground and VCC are getting power. If it reads the supplied voltage on multi meter from the power supply then there indeed power being supplied into the board.

Step 4: then use the probes with chips VVC and Ground, Inputs and Outputs if all have same voltage going into then the board is grounded and is reading for code testing which will determine if the chip works or not.

Step 5: take of the wire from the power supply and connect the vera board with the MSP430 using VCC and Ground.

Step 6: instead of testing all pins at once we would use just one pin to verify.

Step 7: pick a pin from the MSP430 and connect it to the input that is selected first and use the blink on Energia to see if the output will flash HIGH or LOW.

Step 8: get a logic probe tester to verify the results on the output. Get another wire and connect it to VCC and ground.

Step 9: connect the probe to ground and VCC and depending on results errors, analysis and conclusions can be made from this

Test procedure for the PCB

For this test the student needed to make sure that there are no unknown problems, having the right equipment to test with and most importantly the log to right down results, mistakes, conclusions and recommendation. Although the student had already conducted a hardware test to obviously verify that it works in order to write to it the actual test part was of the software opposite to the prototype(hardware).

The procedure goes as follows:

Step 1: firstly, is to have your log book, ribbon cable, MSP with USB, PCB design and motherboard ready and set.

Step 2: is to connect the USB cable to the MSP and PC. Once it's connected then proceed to connect the PCB design under the MSP430.

Step 3: is to use the ribbon cable to connect from the PCB design to the motherboard correctly. If it's not connected correctly it may result in short circuit or even damaging both boards.

Step 4: after all connectivity is in place, go onto the PC and open up energeia.

Step 5: using the code that was created by student to write a bit and a byte test the outputs and see the results of test. Do I binary count using a loop or even change the code to write to the desired LED. Check to see if there is a message on the LCD which in this case the student didn't complete.

Step 6: on the motherboard itself use the Switches direct to control which output is on and see the results

Problems and solutions (prototype)

Problems can be good for us young engineers because without problems we wouldn't have the skill to solve problems which what we are required to do in the world of engineering. I will go through the min important problems that I had encounter along the way of building the board interface and solution that came with it.

- Problem 1: was because of the wire insulation not being cut properly leading to short circuits. I had used wire to make connection underneath the boards due to lack of spacing I had on the front side of the board. The wire I had used where thin and more likely to break if not handled carefully. I stripped them using wire cutters and measured the length with my nail for correct length connection so they wouldn't be to lose around the back of the board. The problem was that to much of the wire its self was exposes to much and soldered in the pins and touching off other wire causing shorts in the board.
- Solution 1: a helpful solution I got from my fellow colleague was to get two wire cutters with one holding the wire and the other just about peeling or marking the layer I want to cut of and pull gently to that a I get the exact wire length that I need and so my wire doesn't break in the process.

- Problem 2: there was a lot of complications when the board interface was fully built. The reason for this was because of voltage going into pins that it's not supposed to go which again causes short circuits in the board. I didn't realise the problem be I was blind sides with the lose wiring I had at the back of the board not realising that the tracks had not been cut.
- Solution 2: A recommendation that came from my lecturer Tom was to cut tracks as I go along since I would have a sense of direction on what is working and what needs to be done next instead of leaving it to the very end causing distress due so much tracks to cut or not finding errors.

- Problem 3: This consists of problems I had before and I repeated such as soldering things in the wrong way, making wrong connections, not using the right port for testing my code and unnecessary solder slash between to pins that are not needed what so ever. I found myself repeating the same mistakes again without realising that it had happened very frequently. My mind set was narrow and not wide seeing the that it can be avoided.

- Solution 3: The easy solution to all of that is a log book. I found out the hard way that a log helps record and solve the things that I have already came across. This makes things easier for myself and helps further understand how and why things like that is happening and even help me come up with not just on but to conclusions and analysis. Not only does have problems that I had but it can help my record new problems and give me ideas to come up with a better solution which is a great aspect of learning the world of engineering. A log book shows what you are doing and how you go about doing it.

Problems and solutions PCB

Unlike the prototype and mistakes that the student had made, the student as not only manage to make less problems but also verifying them and checking them frequently as possible even reducing board errors significantly.

- Problem 1: the first problem encountered was with the hardware. Thankfully enough it was wasn't a huge error since all pins were connected and had power. The only issue was that the pins on the design were switched even though the student followed the schematic several times to verify the right connection. This was the only hardware issue
- Solution 2: since these can be changed and modified, they were changed around in the software so that it would send signals to right output. However, if they were ground or VCC it would have been a greater problem since they cannot be changed.

- Problem 2: onto the software the serial port was unfortunately not recognized due the cable not being a USB type and instead a charger.
- Solution 2: to know if its USB type cable it would show a symbol at the top of the USB cable to indict what type of cable it is.

- Problem 3: there was pin that wasn't declared in the code. The student assumed that in fact the pin was even declared but after checking it actually declared.
- Solution 3: after looking at the pins on both tab it was made known that the pins were not exactly typed in the same since the every is case sensitive.

- Problem 4: there was a big software problem which told the student that a pin or a variable was not declared in the scope. After looking through each tab it was clear that everything was define, declared and checked for spelling and cases.
- Solution 4: it was then decided to move every other tab to the main tab not including the define tab. This was done by copying and pasting. The tabs that were copied and pasted from where either commented out or deleted. Once it was all done the program compiled and uploaded on the MSP caring out its task.

Conclusion and recommendations

The conclusion for the prototype is that planning and knowing to conduct a design and test a goes to show ability to understand and solve problems. The actual schematic was complexed at first glance but the breakdown of each layout got the student to understand and figure out the build layout of the project board interface. Yet the board interface wasn't fully co-operative though it classifies as learning for the student to debug the errors of each chip. Time management was not exactly to plan since the student was 2 weeks behind in soldering. The project build planning was planned for 3 weeks when only it should have been for a week. The student believe 2 chips should have been tested and verified first in the prebuilding stage so that it would more time for the building stage. Each chip should be verified each within 2 hours each out of the 4 hours of the lab. More careful and understanding planning should have carried out so that less errors would be caused.

More analysis and elaboration could have been talked so that there would be less confusion with let's say on which side the IC components should be soldered onto. But overall, everything went great with verify, testing, questions and recommendations.

The student learned to record things in my log book better such everything that happened for example, errors, recommendations, theories testing conclusions, diagrams, codes and information about components which I get from data sheets. The student also learned to conduct a test perfectly and can conclude on problems such as short circuits and faulty connections.

The student's soldering has gone better due to my practice with small strip boards in the lab and showing it to my lecturer. Doing this allow me to learn from my mistake at the moment then doing that on the actual project itself. My wire cutting skills drastically improved thanks to my Lecture showing me how to get the exact length measurements of the wire to place on the board. I'm now sharper and faster at doing so.

On the other hand, the PCB design was tackled a lot better the protype with planning, suggested work time, carful layout of design and etc. although there was struggle to get the LCD, feed to MSP using processing and switches direct to work, more than half of the project was complete and ready for testing. one of the biggest helps was the log book. Learning from the mistakes of the prototype, making the PCB design was easy due what was recorded and being recorded.

It made sense to me that design hardware is not as easy as designing on software. The reasons for this are because for hardware it requires a lot of schemes and steps in order to conduct and test. And the test must conduct in specific locations of the hardware. Once there is a mistake in the hardware most things have be changed or checked which would cause a lot of hassle. But when it comes to software it tells you the error and it's easy to modify and change around without accidently causing other problems.

There could be still room for improvement since the plan was worst case scenario. Time management was the only main big issue but was tackled a lot better than before.

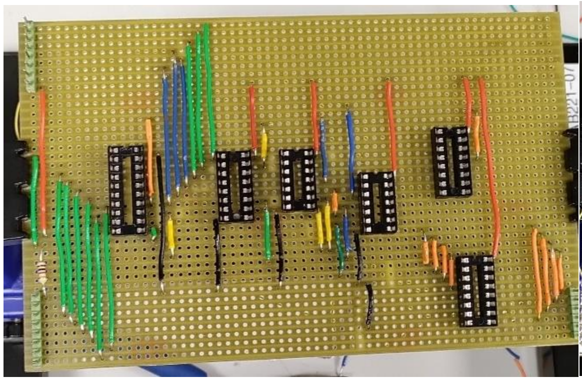
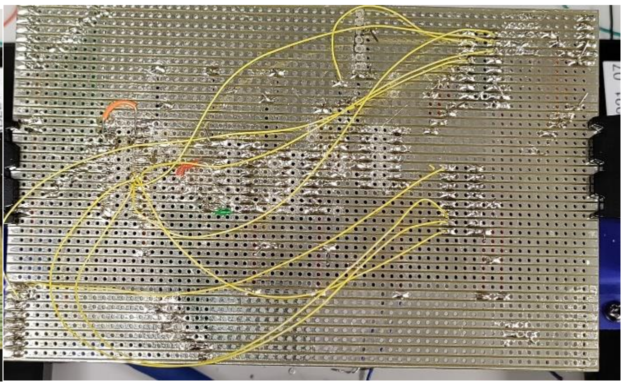
Bibliography

Web sites that I have looked into that inspired me:

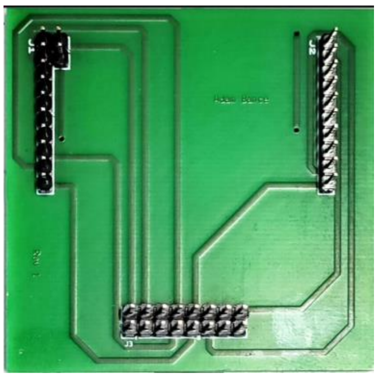
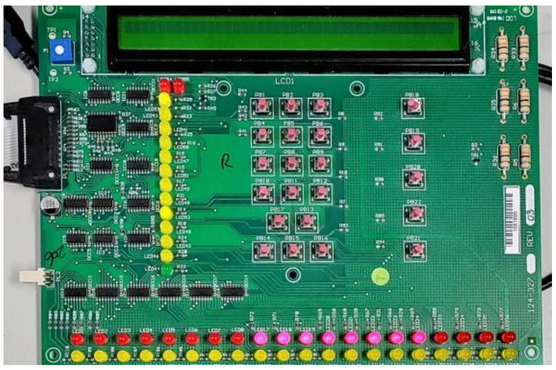
- Texas Instruments
- The manufactuer.com
- Energia
- Nexperia.com
- Farnell.com
- Gatevidyalay.com

Appendices

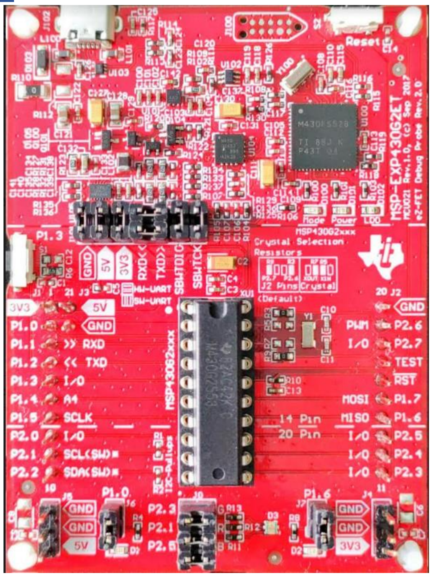
Prototype

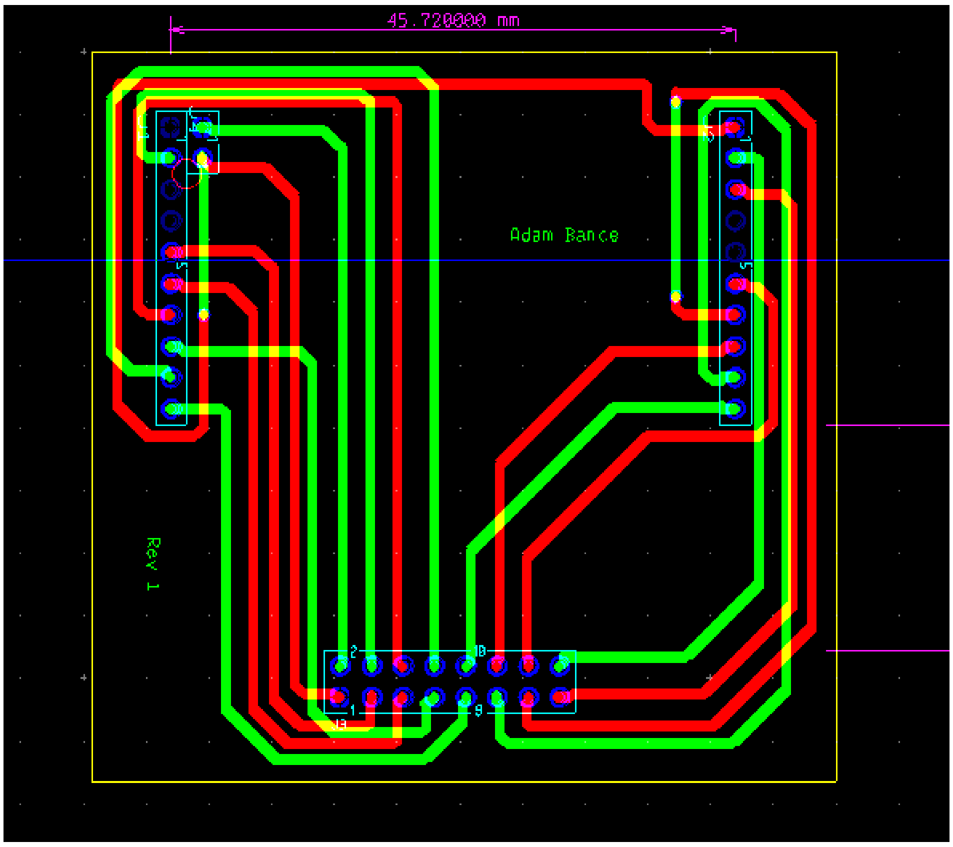
PCB and motherboard

Ribbon cable and MSP430



PCB routed (copper top and bottom)



User manual Guide

The instruction goes as follows:

- Power up the device that you have and connect it to the board interface.
- Make sure the device is connected to an application called energia. If that application is not already on the software install in from the web browser. This application is for the user to create and test a code on any giving device once its connected
- Once its connected you will need to set 2 things. Make sure you are on any other com except for com 1 and to select the right board on board management.
- To change coms, go to tools, then go to Port, and then select the appropriate com.
- The next step is to go to the board manager. To do that you will need to go to tools and scroll down to board. Then click on the device name that you currently have. You may need to update the application to allow your device name to show up.
- Once everything is set do write the program o the application or get a sample program and test each individual chip.
- After each test has been conducted the last test would be on the 74HCT259 which is the chip allows what goes and out towards the output.

Safety And Hazard

- Make sure that your power is at the right amount of volts and not exceeding the voltage limit which could cause damage. Also make sure your current is limited correctly. An example would be for 5 volts use 20mA.

- Make sure that there are not shorts in your device and that everything is grounded perfectly. If failed to do so, there could be possible burning or shock from the device

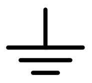


Figure 9.2 ground symbol

- Make sure to the device back in it original packaging to avoid static. The packaging is designed specifically to keep the device from any damages or static.

- If there is a smell of burning, turn off all devices and disconnect every connection from the device.

.
.

Environmental impacts

Recycling plays a huge part in the environment especially when it comes to electronics used by engineers themselves. There can be good friendly Environmental impacts on the world today and there can be bad Environmental impacts. When i say electronics I mean like hairdryers, ATMS, Phones, laptop, security systems, machinery, network equipment and etc. These are the so called things that we interact with as human beings and depending on how we are carful about things could affect everything altogether as technology.

Mining and drilling raw materials are used to actually create components. Things like metal ores become into things like transistors, wire diodes and touchscreen devices. Natural resources like oil which is where plastic is derived from are very limited making it non-renewable. The process of extracting oil and ores by mining and drilling is damaging to the environment and ecosystem.

Another impact would be power and energy were electronic require electricity. That very electricity comes from an outlet that travels back from the power grid or even from batteries. Some countries have eco-unfriendly recourses like coal. The thing with batteries is that they are very dangerous to dispose since a battery carries, lead, cadmium and lithium. These metals can intoxicate soil and water. Such components like this can start to break down and slowly and leach around the Globe.

An eco-friendly impact that I found is that electric cars are good for the environment due to the cars producing fewer green house gasses and air pollutants then a petrol or diesel car would. The electricity generated in the car to keep them going and not realising carbon dioxide into the atmosphere. This is a huge step for engineers around the globe to make the environment better or even help improve the environment. Air pollution would be positively impacted meaning humans can breathe a more breathable air that would not affect their health.

One last thing that I left out is that society can be heavily affected my devices like phones, TV and computers in which they can be used for so long that can really bad for the human body. The main effect of this can cause changes in the human brain and are of increasing prominence according to studies. This goes to show that impacts such as what I have elaborated on can be reversed and changed depending on what we do and how we go about doing it

References

[1] Electrical Technology, "Logic OR Gate – Digital Gates," Apr. 2018. [Online]. Available: https://www.electricaltechnology.org/2018/04/logic-or-gate-digital-gates.html. [Accessed: Apr. 09, 2025].
[2] Learning About Electronics, "Electronics Tutorials," [Online]. Available: http://www.learningaboutelectronics.com. [Accessed: Apr. 09, 2025].
[3] Philips Semiconductor, "74HCT138 datasheet," [Online]. Available: https://html.alldatasheet.com/html-pdf/15536/PHILIPS/74HCT138/499/2/74HCT138.html. [Accessed: Apr. 09, 2025].
[4] EE Paper, "Circuit diagram of 74HC138 composing 32-wire decoder," [Online]. Available: https://ee-paper.com/circuit-diagram-of-74hc138-composing-32-wire-decoder/. [Accessed: Apr. 09, 2025].
[5] Testbook, "Digital Electronics – NAND Gate," [Online]. Available: https://testbook.com/learn/digital-electronics-nand-gate/. [Accessed: Apr. 09, 2025].
[6] Physics Hypertext, "NAND Gate," [Online]. Available: https://physicshyper.wordpress.com/electricity-and-magnetism/electronics/integrated-circuits/gates/nand-gate/. [Accessed: Apr. 09, 2025].
[7] Smarts4k, "8-to-1 MUX Circuit Diagram," [Online]. Available: https://www.smarts4k.com/8-to-1-mux-circuit-diagram/. [Accessed: Apr. 09, 2025].
[8] Plusmanga, [Online]. Available: https://plusmanga.org/. [Accessed: Apr. 09, 2025].
[9] Texas Instruments, "CD74HCT259 datasheet," [Online]. Available: https://www.ti.com/lit/ds/symlink/cd74hct259.pdf. [Accessed: Apr. 09, 2025].
[10] Texas Instruments, [Online]. Available: https://www.ti.com/. [Accessed: Apr. 09, 2025].
[11] Master Bond, [Online]. Available: https://www.masterbond.com/. [Accessed: Apr. 09, 2025].
[12] ScienceDirect, [Online]. Available: https://www.sciencedirect.com/. [Accessed: Apr. 09, 2025].
[13] Farnell Ireland, [Online]. Available: https://ie.farnell.com/. [Accessed: Apr. 09, 2025].
[14] The Manufacturer, [Online]. Available: https://www.themanufacturer.com/. [Accessed: Apr. 09, 2025].
[15] Energia, [Online]. Available: https://energia.nu/. [Accessed: Apr. 09, 2025].