

**LABORATORY WORK BOOK**  
**For The Course**  
**EL-101 Electronic Engineering Drawing & Workshop**

(Batch 2013-14)



Name: \_\_\_\_\_

Roll No. : \_\_\_\_\_

Batch: \_\_\_\_\_

Year: \_\_\_\_\_

Dept.: \_\_\_\_\_

**Department of Electronic Engineering**  
**N.E.D. University of Engineering & Technology,**  
**Karachi –75270, Pakistan**

**LABORATORY WORK BOOK**  
**for the course**

**Electronic Engineering Drawing & Workshop**  
**(EL-101)**

PREPARED BY:

**Mr. Danish Mahmood Khan (Sr. Lab. Engineer)**  
**Ms. Komal Masroor (Lecturer)**

REVIEWED BY:

**Dr. Syed Shoaib Hasan Zaidi (Chairman)**



APPROVED BY:

**The Board of Studies of Department of Electronic Engineering**

## **ACKNOWLEDGEMENT**

We owe thanks to a number of people who helped and supported us in the compilation of this work book.

We would like to pay our heartfelt gratitude to **Ms. Madiha Shabbir**, Lecturer (dept. of Electronic engineering), who has played an important role in project assignment and its evaluation. Our deepest thanks to **Ms. Tooba Khan** for structuring lab sessions based on LabView. Also, we sincerely appreciate **Mr. Zeeshan Nafees**, hardware technician (dept. of Electronic engineering), for extending his support to re-engineer practical work related to PCB designing using OrCAD.

We are also grateful to Prof. Dr. Shoaib Hasan Zaidi for being a wonderful guide to encourage implementation of problem based learning.

Above all, we would like to acknowledge resources from the INTERNET, cited in the bibliography section, that have been helpful in assembling this workbook.

# **PREFACE**

This document has been divided into four sections:

- Computer Aided Designing
- Soldering
- Data Visualization and Analysis
- Data Acquisition and Control

Each of these segments has been designed to enhance essential hands-on learning and application skills. Lectures and demonstrations are supplemented with project based learning. This process introduces critical thinking and problem solving skills.

- Computer Aided Design (CAD) software are now a very common method of producing technical drawings. OrCAD is one such tool which is used for designing Printed Circuit Boards (PCBs). The sequence of steps, starting from simulating an electronic circuit, converting this to a pattern for the PCB using the software, transferring this pattern to the PCB and subsequently processing the PCB are taught in this segment.

Note: Tasks covered in this manual may not cover all the features of a tool. The emphasis however, is on the steps that you will need to perform in each OrCAD tool so that your design works smoothly through the flow.

- Hands-on Soldering will allow students to make sophisticated and permanent Electronic circuits easily, efficiently and accurately which will help them in their hardware based Term Projects as well as in their Final Year projects.

- The availability of technical computing environment such as MATLAB is now reshaping the role and applications of computer laboratory projects.

It is a powerful and accessible tool for data visualization (both 2D and 3D graphing) and for data analysis. Also, it gives the students an opportunity to easily conduct numerical experiments and to tackle realistic and more complicated problems. The lab sessions have been designed to familiarize students with a computer software (e.g. MATLAB) to address real world problems.

- LabVIEW is a graphical programming language which is widely utilized for data acquisition and process control. This manual therefore, will help the students to get started with the graphical programming.

# ELECTRONIC ENGINEERING DRAWING & WORKSHOP

## CONTENTS

Lab #	List of Experiments	Remarks
	<b><u>Section 1: Project Based Learning</u></b>	
1	Introduction to Breadboard	
2	a) To learn how to solder b) To learn how to de- solder	
3	Projects	
	<b><u>Section 2 : COMPUTER AIDED DESIGNING (CAD)</u></b>	
4	a) Introduction to Orcad Capture b) Generation of schematic c) Generation of Netlist	
5	a) Introduction to Layout Tool. b) Creation of Layout Board.	
6	a) Introduction to component foot prints. b) Editing and creating new footprints.	
7	a) Overview of Autorouting using Orcad Layout. b) Defining DRC. c) Making Copper Pour.	

8	Introduction to Manual Routing using OrCAD Layout Tools.	
9	a) Introduction to Post Processing b) Introduction to Gerbtool	
10	a) Practical on the identification of Drill Bits and Drilling on the Manual Drill Machine. b) Practical to understand the electroplating processes on ABC plating line in COMPACTA machine BUNGARD. c) Practical on plotting, fixing and developing a film for exposing of PCB.	
	<b><u>Section 3: DATA VISUALIZATION &amp; ANALYSIS</u></b>	
11	a) Introduction to MATLAB and MATLAB GUI b) Introduction to basic operations	
12	Introduction to Mathematical functions, Variables and Matrix operations	
13	Array Operations And Programming in MATALB	
14	Introduction to Graph plotting in MATLAB	
15	Introduction to Control Flow and Loops in MATLAB	
	<b><u>Section 4: INTRODUCTION TO DATA ACQUISITION, CONTROL AND HARDWARE INTERFACE</u></b>	
16	Introduction to LabView	
17	To learn how to create subVI and use it in another VI	
18	To learn how loops are designed and work in LabVIEW.	

	To write a VI for the Multiplication of a random number with 10 and displaying the result continuously, until it is stopped.	
19	To learn how for loops are designed and work in LabVIEW	
20	To learn different functionalities of arrays	
21	To display quantities on waveform charts and waveform graphs	
22	To learn the use of structures in LabVIEW for decision-making	
23	To learn how to build clusters and their various functions.	
24	To learn the use of various functions to edit and manipulate strings To learn the use of basic file I/O functions	
25	<b><u>Bibliography</u></b>	

**Note:** *All the contents of this manual have been extracted from internet; the links have been cited where possible. If any of the references have been missed, we would be more than pleased to add it to the list of citations in the revisions of this manual.*

*Thank you*

**Dr. Shoaib Zaidi**  
**Chairman**  
**Department of Electronic and Telecommunications Engineering**

## **Lab Session 01**

### **OBJECTIVE:**

Introduction to Breadboard

### **Breadboard**

A breadboard is used to make up temporary circuits for testing or to try out an idea. No soldering is required so it is easy to change connections and replace components. Parts will not be damaged so they will be available to re-use afterwards. Thus, a breadboard is used to check whether the circuit works as intended.

The photograph shows a typical small breadboard which is suitable for beginners building simple circuits with one or two ICs (chips).



Small Breadboard  
Photograph © [Rapid Electronics](#)

### **Connections on Breadboard**

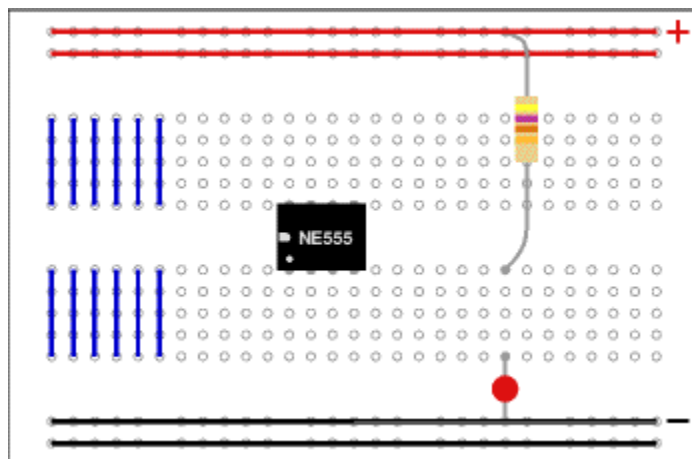
Breadboards have many tiny sockets (called 'holes') arranged on a 0.1" grid. The leads of most components can be pushed straight into the holes. ICs are inserted across the central gap with their notch or dot to the left.

Wire links can be made with single-core plastic-coated wire of 0.6mm diameter (the standard size). Stranded wire is not suitable because it will crumple when pushed into a hole and it may damage the board if strands break off.

The diagram shows how the breadboard holes are connected:

The top and bottom rows are linked horizontally all the way across as shown by the red and black lines on the diagram. The power supply is connected to these rows, + at the top and 0V (zero volts) at the bottom.

Use upper row of the bottom pair for 0V, then you can use the lower row for the negative supply with circuits requiring a dual supply (e.g. +9V, 0V, -9V).





The other holes are linked vertically in blocks of 5 with no link across the centre as shown by the blue lines on the diagram. Notices how there are separate blocks of connections to each pin of ICs.

On larger breadboards there may be a break halfway along the top and bottom power supply rows. It is a good idea to link across the gap before you start to build a circuit, otherwise you may forget and part of your circuit will have no power!

## **Building a Circuit on Breadboard**

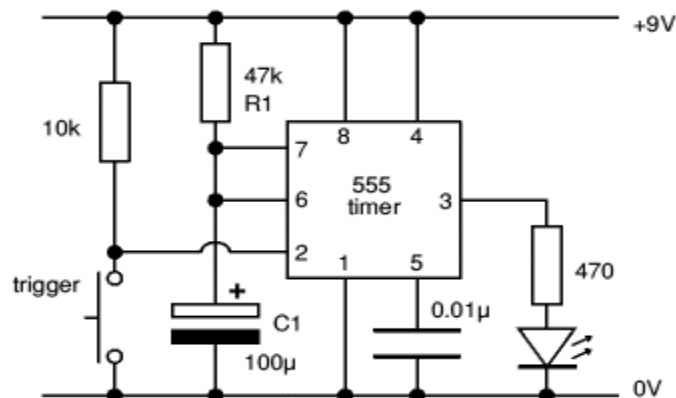
Converting a circuit diagram to a breadboard layout is not straightforward because the arrangement of components on breadboard will look quite different from the circuit diagram.

When putting parts on breadboard you must concentrate on their connections, not their positions on the circuit diagram. The IC (chip) is a good starting point so place it in the centre of the breadboard and work round it pin by pin, putting in all the connections and components for each pin in turn.

The best way to explain this is by example, so the process of building this 555 timer circuit on breadboard is listed step-by-step below.

The circuit is a monostable which means it will turn on the LED for about 5 seconds when the 'trigger' button is pressed. The time period is determined by R1 and C1 and you may wish to try changing their values. R1 should be in the range 1kΩ to 1MΩ

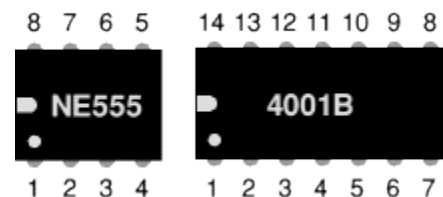
Time Period,  $T = 1.1 \times R1 \times C1$



Monostable Circuit Diagram

## **IC pin numbers**

IC pins are numbered anti-clockwise around the IC starting near the notch or dot. The diagram shows the numbering for 8-pin and 14-pin ICs, but the principle is the same for all sizes.

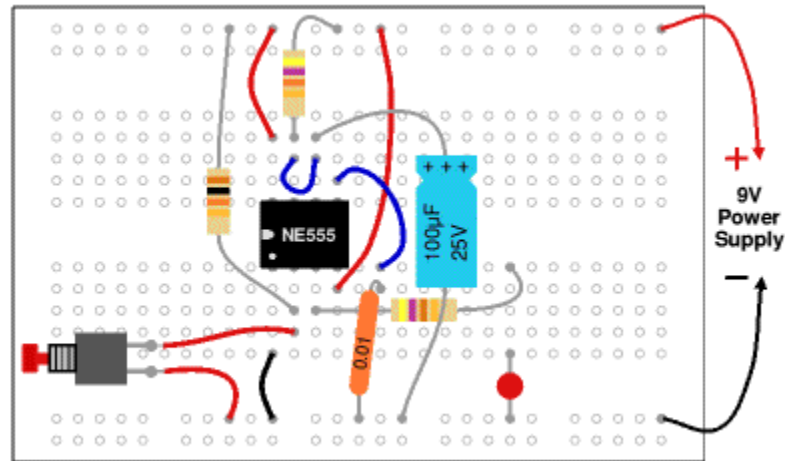


## **Building the example circuit**

Begin by carefully insert the 555 IC in the centre of the breadboard with its notch or dot to the left.

Then deal with each pin of the 555:

1. Connect a wire (black) to 0V.
2. Connect the 10k resistor to +9V.  
Connect a push switch to 0V (you will need to solder leads onto the switch)
3. Connect the 470 resistor to an unused block of 5 holes, then...  
Connect an LED (any colour) from that block to 0V (short lead to 0V).



Monostable Circuit on Breadboard

4. Connect a wire (red) to +9V.
5. Connect the 0.01µF capacitor to 0V.  
You will probably find that its leads are too short to connect directly, so put in a wire link to an unused block of holes and connect to that.
6. Connect the 100µF capacitor to 0V (+ lead to pin 6).  
Connect a wire (blue) to pin 7.
7. Connect 47k resistor to +9V.  
Check: there should be a wire already connected to pin 6.
8. Connect a wire (red) to +9V.

Finally,

- Check all the connections carefully.
- Check that parts are the correct way round (LED and 100µF capacitor).
- **Check that no leads are touching** (unless they connect to the same block).
- Connect the breadboard to a 9V supply and press the push switch to test the circuit.

If your circuit does not work disconnect (or switch off) the power supply and very carefully re-check every connection against the circuit diagram.

## **Lab Session 02(a)**

**Objective:** To learn how to solder

### **What is solder?**

Solder is an alloy (mixture) of tin and lead, typically 60% tin and 40% lead. It melts at a temperature of about 200°C. Coating a surface with solder is called 'tinning' because of the tin content of solder. Lead is poisonous and one should always wash his/her hands after using solder.



Reels of solder

Photograph © [Rapid Electronics](#)

Solder for electronics use contains tiny cores of flux, like the wires inside a mains flex. The flux is corrosive, like an acid, and it cleans the metal surfaces as the solder melts. This is why solder must be melted actually on the joint, not on the iron tip. Without flux most joints would fail because metals quickly oxidize and the solder itself will not flow properly onto a dirty, oxidized, metal surface.

The best size of solder for electronics is 22swg (swg = standard wire gauge).

### **How to Solder**

#### **Few safety precautions:**

- ✦ **Never touch the element or tip of the soldering iron.**  
They are very hot (about 400°C) and will give a nasty burn.
- ✦ **Take great care to avoid touching the mains flex with the tip of the iron**  
The iron should have a heatproof flex for extra protection. An ordinary plastic flex will melt immediately if touched by a hot iron and there is a serious risk of burns and electric shock.
- ✦ **Always return the soldering iron to its stand when not in use**  
Never put it down on the workbench, even for a moment!
- ✦ **Work in a well-ventilated area**  
The smoke formed when solder is melting mostly from the flux and quite irritating. Avoid breathing it by keeping head to the side of, not above, the work.
- ✦ **Wash hands after using solder**  
Solder contains lead which is a poisonous metal.

### Preparing the soldering iron:

- ✦ **Place the soldering iron in its stand and plug in**  
The iron will take a few minutes to reach its operating temperature of about 400°C.
- ✦ **Dampen the sponge in the stand.**  
The best way to do this is to lift it out the stand and hold it under a cold tap for a moment, then squeeze to remove excess water. It should be damp, not dripping wet.
- ✦ **Wait a few minutes for the soldering iron to warm up.**  
Check if it is ready by trying to melt a little solder on the tip.
- ✦ **Wipe the tip of the iron on the damp sponge.**  
This will clean the tip.
- ✦ **Melt a little solder on the tip of the iron.**  
This is called 'tinning' and it will help the heat to flow from the iron's tip to the joint. It only needs to be done when the iron is plug in and occasionally while soldering if the tip is cleaned on the sponge.

### Start Soldering:

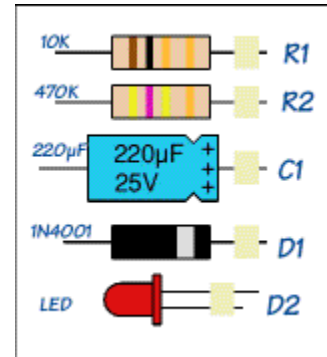
- ✦ **Hold the soldering iron like a pen, near the base of the handle.**  
Remember to never touch the hot element or tip.
  - ✦ **Touch the soldering iron onto the joint to be made.**  
Make sure it touches both the component lead and the track.  
Hold the tip there for a few seconds and,
- The diagram illustrates two types of solder joints on a PCB or stripboard. On the left, a 'GOOD JOINT' is shown with a 'volcano shape' of 'shiny solder' bridging a 'component lead' and 'copper tracks'. A green checkmark is placed above it. On the right, a 'BAD JOINT' is shown as a 'dry joint' with 'dull solder' that does not properly connect the 'component lead' to the 'copper tracks'. A red X is placed above it. The components are labeled 'component' and the board is labeled 'PCB or stripboard'.
- ✦ **Feed a little solder onto the joint.**  
It should flow smoothly onto the lead and track to form a volcano shape as shown in the diagram. Apply the solder to the joint, not the iron.
  - ✦ **Remove the solder, then the iron, while keeping the joint still.**  
Allow the joint a few seconds to cool before the circuit board will be moved.
  - ✦ **Inspect the joint closely.**  
It should look shiny and have a 'volcano' shape. If not, reheat it and feed in a little more solder. This time ensure that **both** the lead and track are heated fully before applying solder.

### Using a heat sink

Some components, such as transistors, can be damaged by heat when soldering so it is wise to use a heat sink clipped to the lead between the joint and the component body. One can buy a special tool, but a standard crocodile clip works just as well and is cheaper.

## Soldering Advice for Components

1. **Stick all the components onto a sheet of paper using sticky tape.**
2. **Identify each component** and write its name or value beside it.
3. **Add the code (R1, R2, C1 etc.) if necessary.** Many projects from books and magazines label the components with codes (R1, R2, C1, D1 etc.)
4. **Note down the values of Resistors and Capacitors by decoding schemes.**

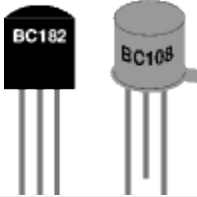


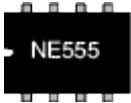


Some components require special care when soldering.

Many must be placed the correct way round and a few are easily damaged by the heat from soldering. Appropriate warnings are given in the table below, together with other advice which may be useful when soldering.

For most projects it is best to put the components onto the board in the order given below:

	Components	Pictures	Reminders and Warnings
1	<b>IC Holders</b> (DIL sockets)		<b>Connect the correct way round</b> by making sure the notch is at the correct end. Do NOT put the ICs (chips) in yet.
2	<b>Resistors</b>		No special precautions are needed with resistors.
3	<b>Small value capacitors</b> (usually less than 1µF)		These may be connected either way round. Take care with polystyrene capacitors because they are easily damaged by heat.
4	<b>Electrolytic capacitors</b> (1µF and greater)		<b>Connect the correct way round.</b> They will be marked with a + or - near one lead.
5	<b>Diodes</b>		<b>Connect the correct way round.</b> Take care with germanium diodes (e.g. OA91) because they are easily damaged by heat.
6	<b>LEDs</b>		<b>Connect the correct way round.</b> The diagram may be labelled <b>a</b> or + for anode and <b>k</b> or - for cathode; The cathode is the short lead and there may be a slight flat on the body of round LEDs.

7	<b><u>Transistors</u></b>		<b>Connect the correct way round.</b> Transistors have 3 'legs' (leads) so extra care is needed to ensure the connections are correct. Easily damaged by heat.
8	<b><u>Wire Links</u></b> between points on the circuit board.	 single core wire	Use single core wire; this is one solid wire which is plastic-coated. If there is no danger of touching other parts tinned copper wires can also be used, this has no plastic coating and looks just like solder but it is stiffer.
9	<b><u>Battery clips</u></b> , buzzers and other parts with their own wires		<b>Connect the correct way round.</b>
10	<b><u>Wires</u></b> to parts off the circuit board, including <b><u>switches</u></b> , <b><u>relays</u></b> , <b><u>variable resistors</u></b> and <b><u>loudspeakers</u></b> .	 stranded wire	use stranded wire which is flexible and plastic-coated. Do not use single core wire because this will break when it is repeatedly flexed.
11	<b><u>ICs (chips)</u></b>		<b>Connect the correct way round.</b> <b>Many ICs are static sensitive.</b> Leave ICs in their antistatic packaging until they are needed, and then earth hands by touching a metal water pipe or window frame before touching the ICs. <b>Carefully insert ICs in their holders:</b> make sure all the pins are lined up with the socket then push down firmly with thumb

## Lab Session 02(b)

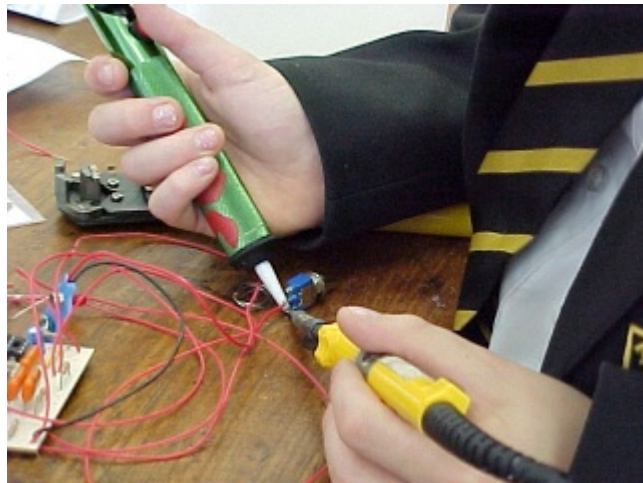
**Objective:** To learn how to de-solder

### **De-soldering:**

In order to de-solder a joint to remove or re-position a wire or component, there are two ways to remove the solder:

#### **1. With a de-soldering pump (solder sucker)**

- ✦ Set the pump by pushing the spring-loaded plunger down until it locks.
- ✦ Apply both the pump nozzle and the tip of soldering iron to the joint.
- ✦ Wait a second or two for the solder to melt.
- ✦ Then press the button on the pump to release the plunger and suck the molten solder into the tool.
- ✦ Repeat if necessary to remove as much solder as possible.
- ✦ The pump will need emptying occasionally by unscrewing the nozzle.



Using a de-soldering pump (solder sucker)

#### **2. With solder remover wick (copper braid)**

- ✦ Apply both the end of the wick and the tip of soldering iron to the joint.
- ✦ As the solder melts most of it will flow onto the wick, away from the joint.
- ✦ Remove the wick first, then the soldering iron.
- ✦ Cut off and discard the end of the wick coated with solder.



Solder remover wick  
Photograph © [Rapid Electronics](#)

After removing most of the solder from the joint(s) one may be able to remove the wire or component lead straight away (allow a few seconds for it to cool). If the joint will not come apart easily apply soldering iron to melt the remaining traces of solder at the same time as pulling the joint apart, taking care to avoid burning .

## **Lab Session 03**

### **OBJECTIVE:**

- **Introduction to OrCAD Capture**
- **Generation of schematic**
- **Generation of Netlist**

### **OrCAD:**

This software is used for complete design and fabrication of PCB(Printed Circuit Board).

### **OrCAD CAPTURE:**

It is a schematic design tool.

### **SCHEMATIC:**

- It is a symbolic representation of electronic components which are connected with virtual wires
- Schematic provides input (Netlist) to the layout tool.

### **NETLIST:**

A Netlist is a file, usually ASCII text, which defines connections between the components in your design.

### **ASCII TEXT:**

The American Standard Code for Information Interchange (ASCII) represents text in computer & communication equipment.

### **PSpice:**

It is OrCAD's simulation tool.

### **OrCAD LAYOUT:**

This OrCAD tool is used for routing of tracks.

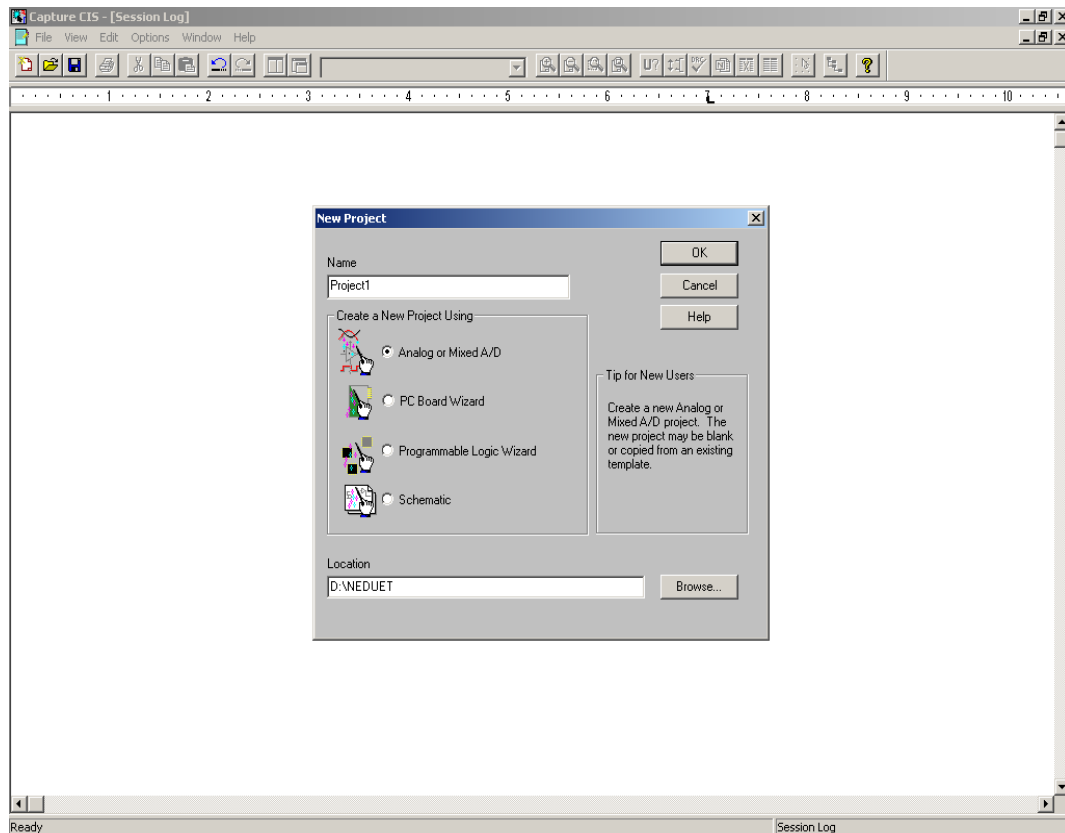
### **CREATING PROJECT:**

Creating Project Wizard provides framework for creating project.

#### **Follow these steps:**

1. Open OrCAD 16.0→Capture CIS
2. OrCAD\_Capture\_CIS\_Option\_with\_Capture→ok  
[CIS: Component Information System]
3. OrCAD Capture CIS – [Session Log]      \*// window will appear
4. From Menu bar select File→New→Project.
- 5.

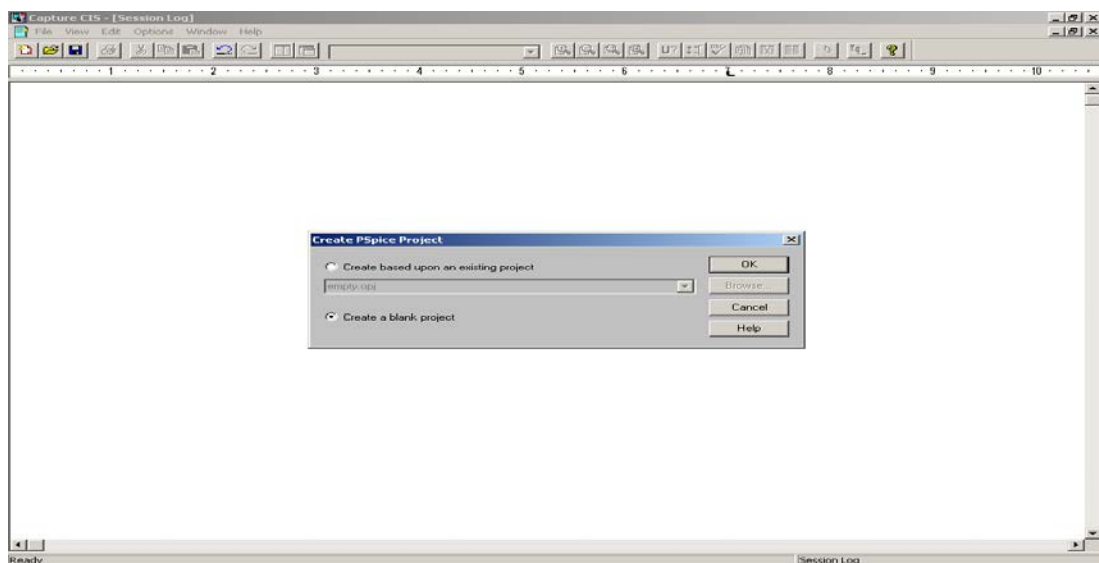




- Give name to your project.
- Either select Analog or Mixed A/D or select Schematic.
- Select the location where you want to save your project  
[Note: *This folder must be unique.*]

*[\* If you have chosen Analog or Mixed A/D.]*

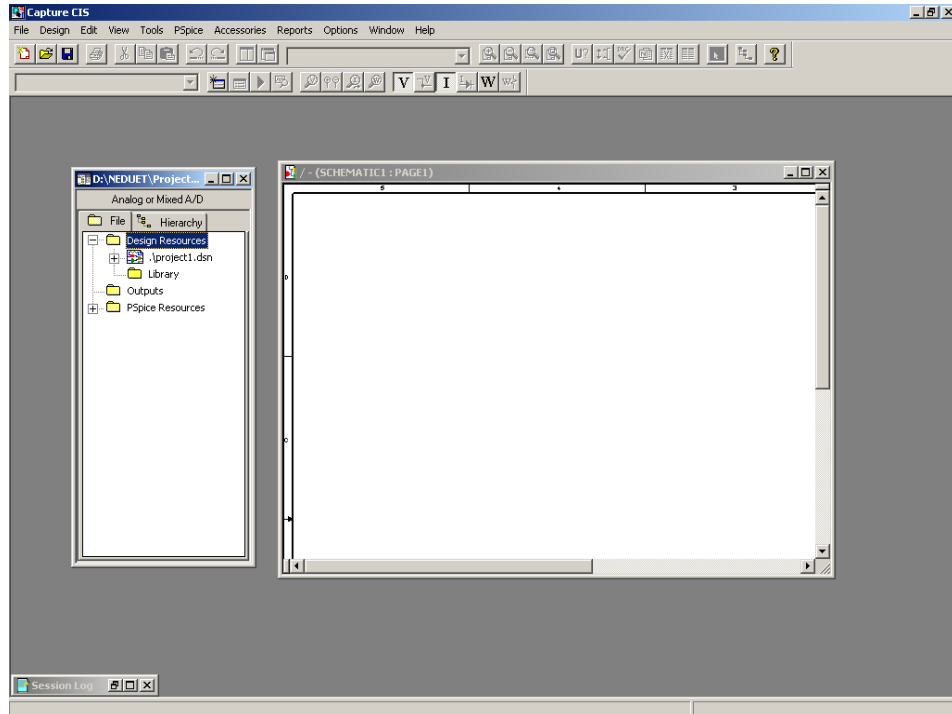
6.



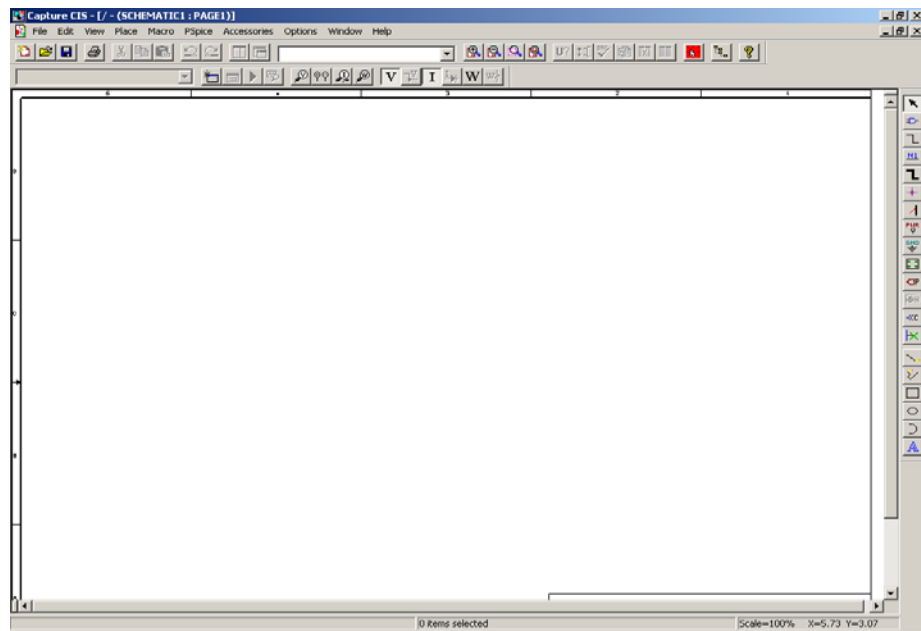
7. Two frame-works will appear:

A. **OrCAD Capture CIS-[/-/(SCHEMATIC1:PAGE1)]**.

B. **OrCAD Capture CIS-[D:\NEDUET\Project1.opj]**. \*// .opj is the project file

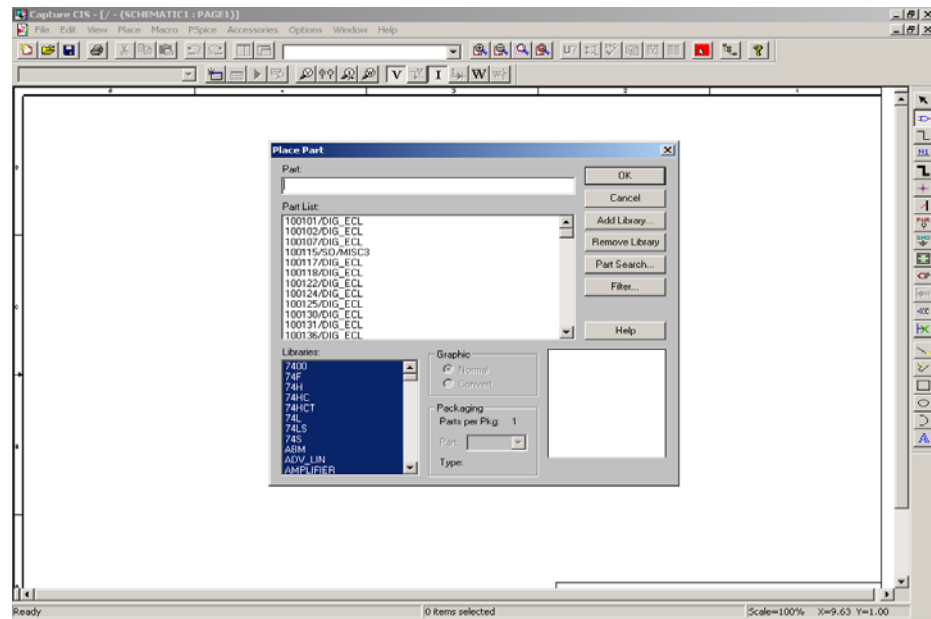


A. **First we Select: OrCAD Capture CIS-[/-/(SCHEMATIC1:PAGE1)] to work and making Schematic Circuit.**

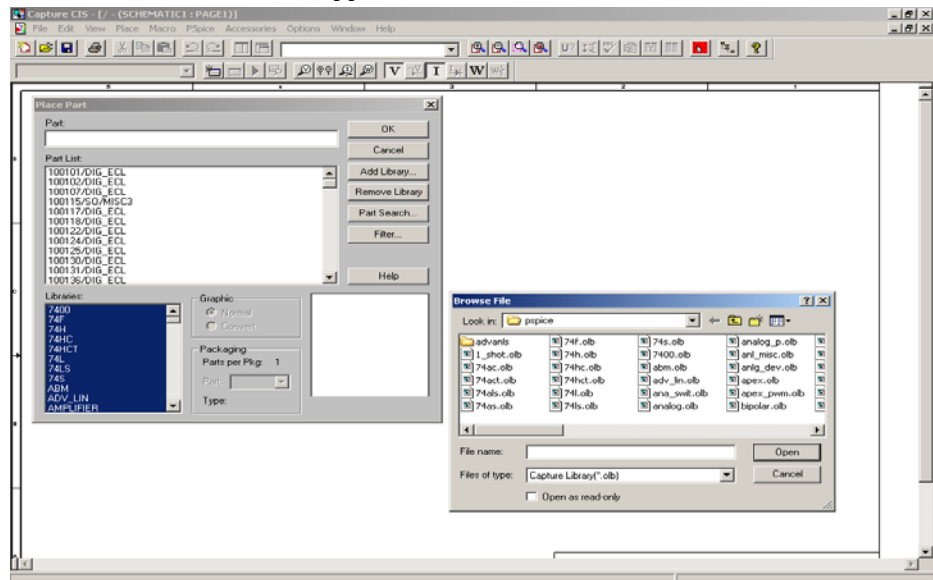


a) **ADDING PARTS:**

- Select Place Part tool from Menu bar.  
*[Place Part window will appear.]*



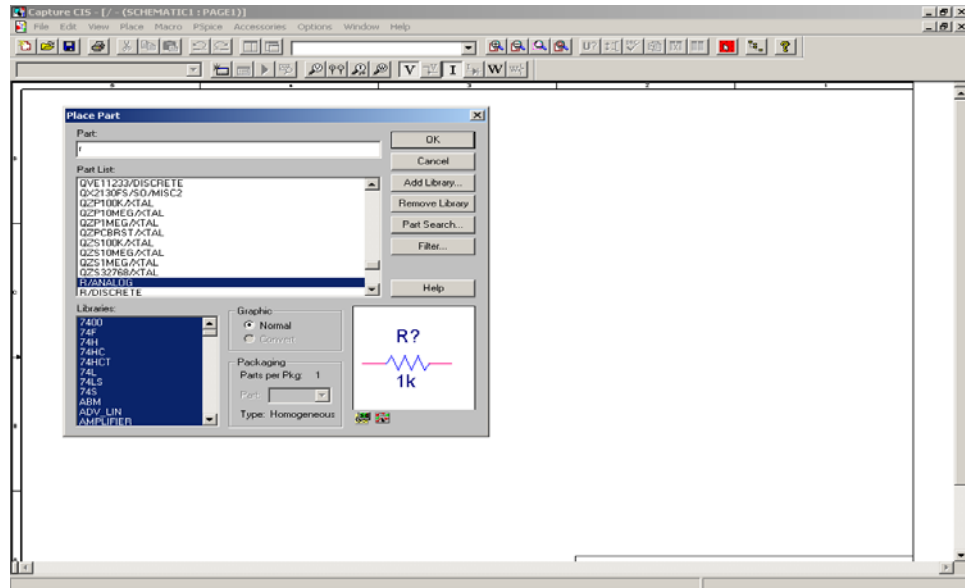
- If you are a beginner then click on Add Library.  
*[Browse File window will appear]*



- Library file extension is .olb.

After adding library files .olb we can select parts (electronic) to make our schematic diagram.

[Note: Select all the file of Libraries then we can easily search component by writing its name.]

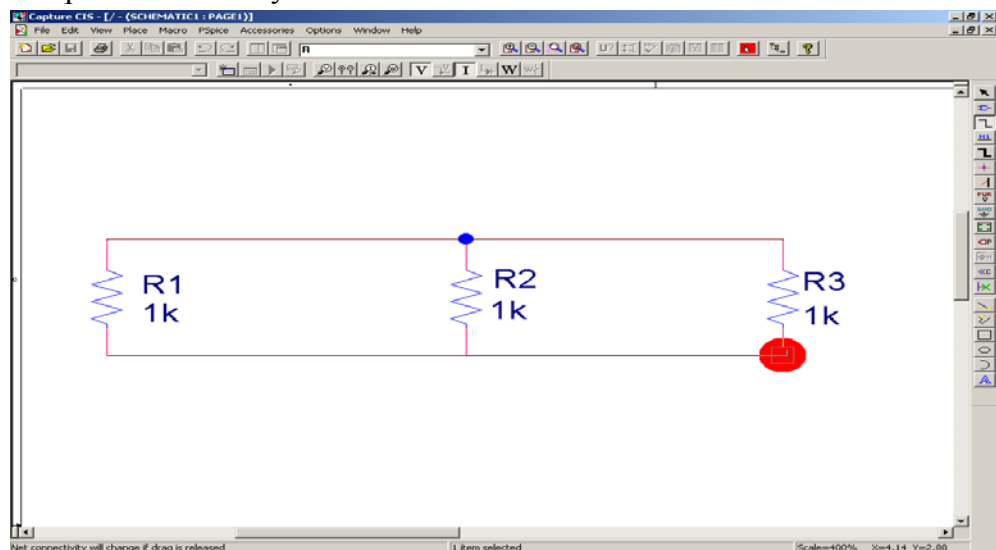


**b) CONNECTING PARTS:**

- After adding required component on schematic page we join these components by virtual “wires”.
- Click Place Wire from place menu.

**c) ZOOM IN/OUT:**

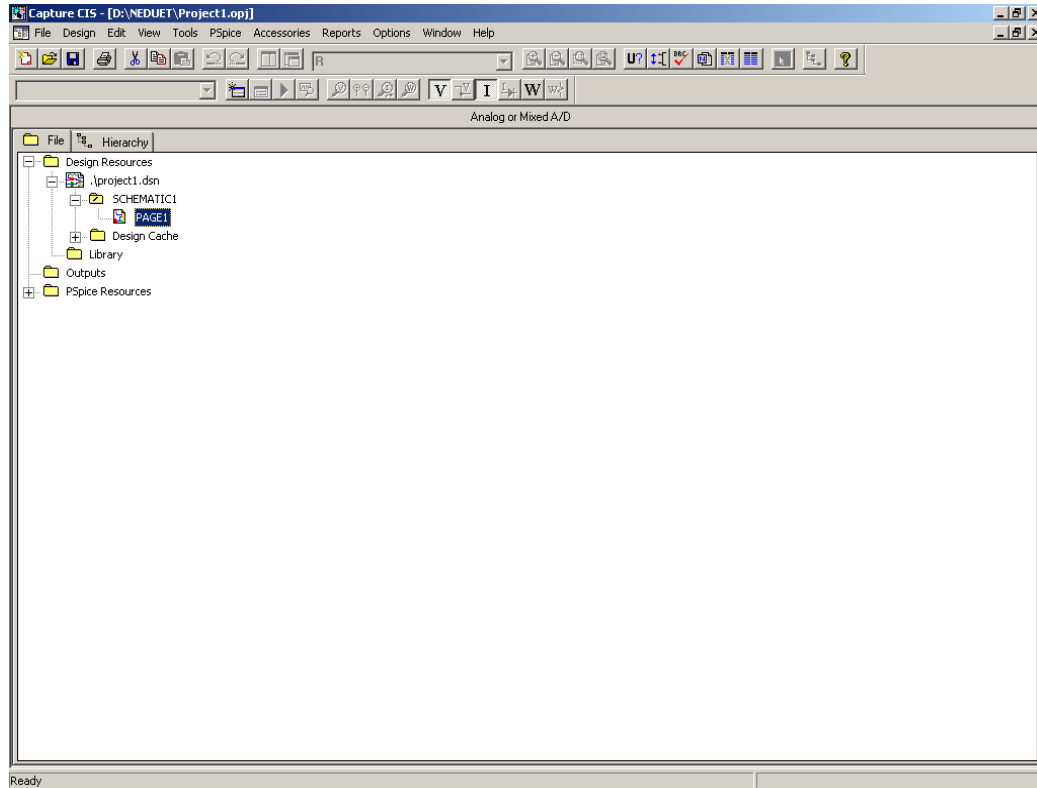
Use “I” and “O” keys to zoom in and out it will help you to connect the components correctly.



After finally making Schematic diagram on schematic page select File→Save→Close.

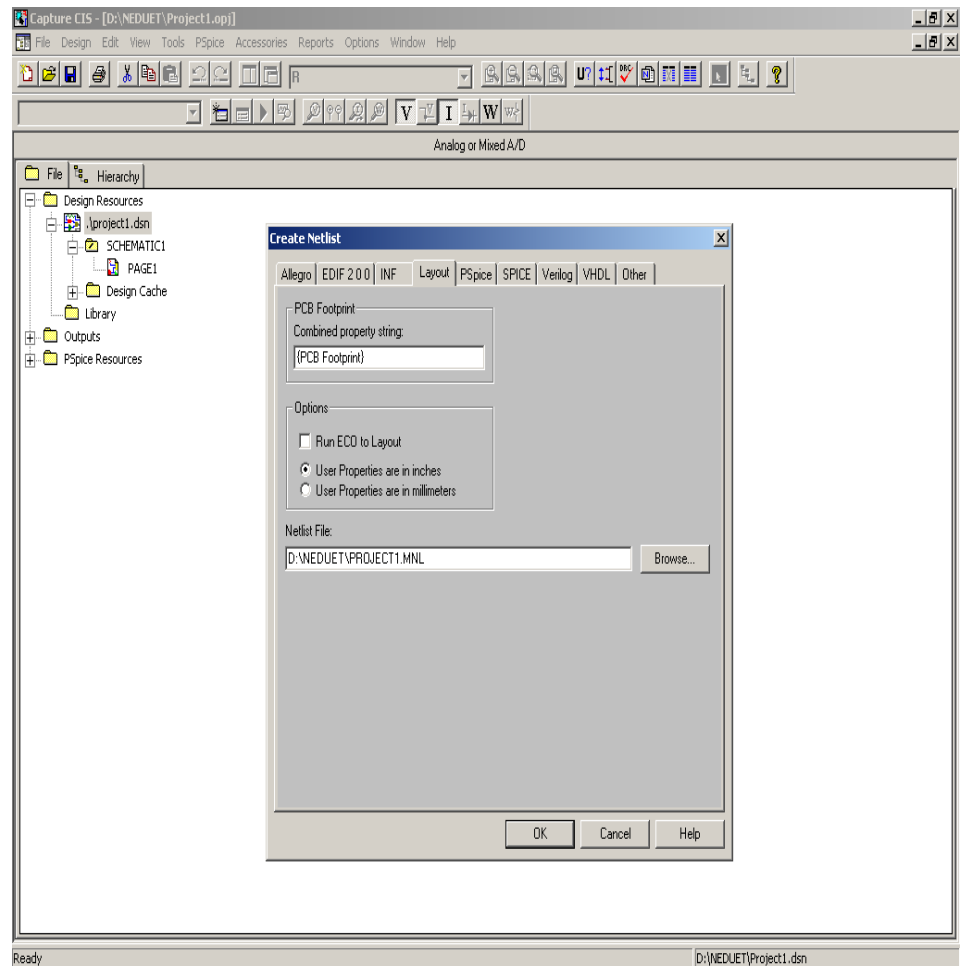
- We are now out from the Schematic page.
- Enter into the project page. \*// .opj is project extension
- .dsn is the schematic file format.

**B. Now we have OrCAD Capture CIS-[D:\NEDUET\Project1.opj] framework window.**



**CREATING NETLIST:**

Select .dsn file by mouse cursor and then go to Tools→Create Netlist.  
[Create Netlist Window will appear]



- Select Layout
- Select ☐ User properties are in inches.
- Put .MNL in a unique folder. \*// It will automatically generate  
[D:\NEDUET\Project1.MNL]
- Click OK.
- Go to File→Save→Close Project.
- The Netlist file .MNL is used as input for Layout.
- This .MNL file will be used to make .MAX file.

[Note: If we make any changes in our schematic then we always make update using Netlist option.]

## **Lab Session 4**

### **OBJECTIVE:**

- **Introduction to Layout Tool.**
- **Creation of Layout Board.**

### **OrCAD LAYOUT:**

OrCAD Layout tool is used for PCB routing and floor planning.

### **LAYOUT:**

Layout Tool accepts Netlist as input (created in Capture CIS) and generates output layout file that suitable for PCB fabrication.

### **CREATING LAYOUT BOARD FILE:**

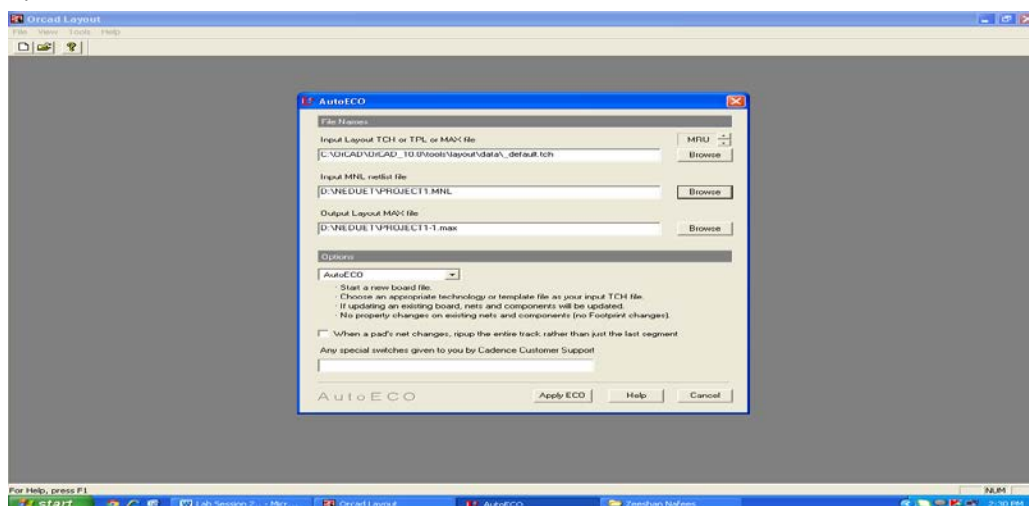
When we create a new board file in OrCAD Layout, we merge the electrical information from the Layout Netlist (.MNL) and physical information from template file (.TPL) or a technology file (.TCH) to create a new board design (.MAX). To be able to create a board file for a new design in Layout, we need to provide template file & netlist.

- A template (.TPL) file describes the characteristics of a physical board. A template includes information such as board outline, layer definition, grid settings, spacing & track width.
- A layout Netlist file (.MNL) describes the parts & interconnection in a schematic.
- A technology file (.TCH) can be considered as a subset of the template, but don't provide physical characteristic information of board.

1. Open OrCAD Layout→File→New.

*[Auto ECO framework window will appear]*

- 2.



- In the input layout TCH or TPL or MAX file text box, specify the name & the location of the technology file to be used for your board.
- In the input MNL netlist file text box, specify the location of Project1.MNL created in layout netlist section.
- The output layout MAX file text box will automatically generate the name and location of file, Project1.MAX.

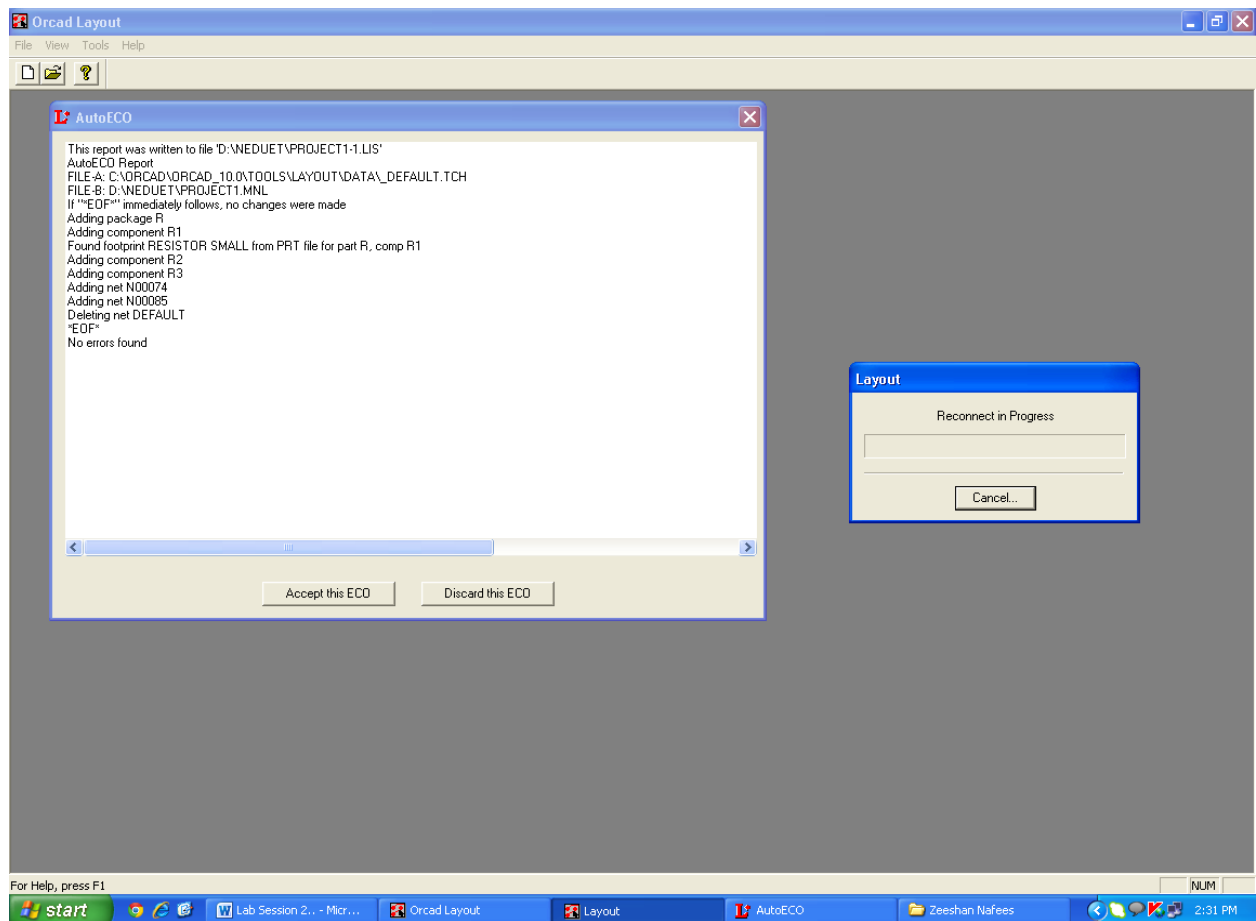
[Note: The Layout board file (.MAX) contains complete electrical & physical information of board.]

3. Apply AutoECO.

4. To create the Layout board file with the settings specified by you, click Apply ECO.

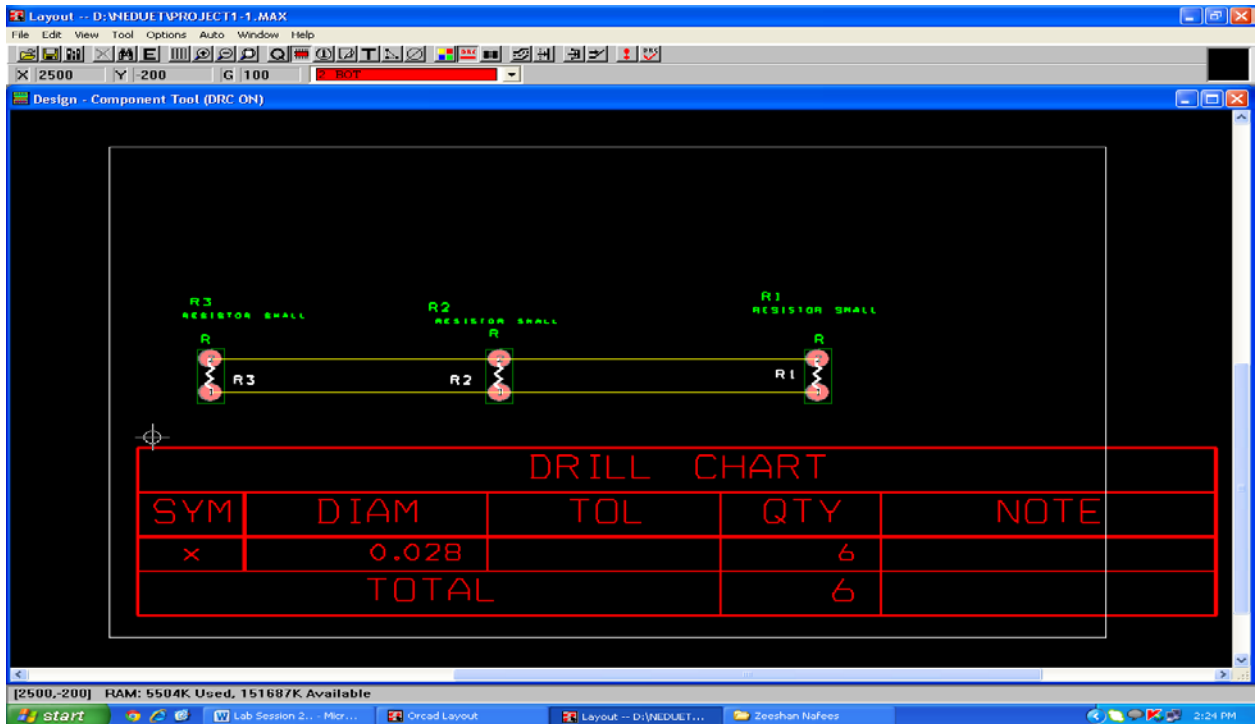
- The Layout progress box appears indicating that the board file is being created.
- The process of creating a board file will be completed only if the footprint information is available for all the components in the design.

*[At this stage beginner students may face some difficulty consult your INSTRUCTOR.]*





- Once AutoECO completed, AutoEco dialog box appears with the report, click Accept this ECO button.
- Click OK.
- Now the framework window of .MAX file will be appeared.



## **Lab Session 05**

### **OBJECTIVE:**

- **Introduction to component foot prints.**
- **Editing and creating new footprints.**

### **COMPONENT FOOTPRINT CREATION:**

- Go to→OrCAD 16.0→OrCAD Layout
- Select Tools from Menu bar→Library Manager (On left side Library Manager Window will appear.)
- Select “Create New Footprint”.  
*[This will open Create New footprint Window.]*
  - I. Give name to your footprint “TO-220 REGULATOR IC” which will be saved with unique identity.
  - II. Select “English” unit system because most of the designs uses mils & inches unit. [Note: 1mm= 39.4mils]
- After pressing/clicking “OK” some text with one pin will be appear in Library Manager Framework window.

We are designing a “Regulator IC” having 3-pins, first of all we will define one padstack (pin) which is already present in Library Manager framework window.

- Select View Spreadsheet→Padstack.

Padstack spreadsheet will be open. We will edit the padstack T1, which is already used by Pin1.

- Double click the padstack name T1, this will open the Edit Padstack dialog box for all the layers in Padstack.
- Change the name of the Padstack to something like PSU\_HOLE. Next select ☐ undefined radio button. Click “OK”. In Spreadsheet you will see Padstack name change from T1 to PSU\_HOLE & all the layers become undefined.

We will now set each layer individually. We can also select multiple layers at a time by holding CTRL key when you click the layer name.

First, let's define the size of the drill used for this part. The datasheet tells us that the pin diameter can vary from 0.027inches to 0.037 inches. So we use a drill bit (for drilling PCB) of 40 mils. [Note: mils is very small unit.]

- Now select layers DRLDWG & DRILL from Padstack Spreadsheet. Press right click →Properties→Edit Padstack window will appear.
- Select ☒ Round & give value of “40” to Pad Width & Pad Height. Press “OK” changes will be appear in Spreadsheet(Padstack).

Now we will define the amount of metal on the routing layers beyond the size of the drill. This is called the “Annular Ring”. Each board shape will have requirements on the minimum “Annular Ring” size based on the drill diameter. In most cases 20 mils is a safe bet.

- Select the Top & Bottom layers→Right Click→Properties→Edit padstack dialog box will appear. Here we can select different shapes for instance we select **Round** & giving the Pad Width 60 & Pad Height 60.  
[ $60=40(\text{Drill}) + 20(\text{Annular Ring})$ ]

The last thing we need to define is the “Solder Mask”. This is usually defined as slightly larger (about 5mils) then the “Annular Ring” on the Top & Bottom layers.

- Select “SMTOP” & “SMBOT” and make round pads with height & width of 65(60mils+5mils).  
[ $65=60(\text{Annular Ring})+5(\text{Solder Mask})$ ]

We can notice the value change in spreadsheet.

We just define those layers which are used to define parts of THT( Through-Hole Technology) & SMT( Surface Mount Technology).

- ✓ THT Components: Top, Bottom, SMTOP, SMBOTTOM, DRLDWG, DRILL.
- ✓ SMT Components: Top, SMTOP, SPTOP.

As far as padstack are concerned, SMT’s are handled easily but requires special equipment’s for soldering.

- Now Click “Save” in “Library Manager”. Since you have designed a new footprint and saving it first time, you will be asked to select the library to keep the footprint in. We have not yet created a footprint library, so you will need to click the “Create NEW Library” button. Browse libraries directory & name the library “PSU\_FOOTPRINTS”.

Let’s now clean up a few things before adding the rest of the pins. You will see a lot of text on your screen. Most of it is on the layers “ASYTOP” which we will not use. This text is safe to delete.

- Click Spreadsheet→Text.  
[Now five text items will be appear in text dialog box]
- Select all text on “ASYTOP” layers by clicking mouse cursor on “ASYTOP” layers. Right click→Delete This will leave the “reference designator” text on the “SSTOP” layer. We will use it later.

Before creating all the pins of our “Regulator IC”, please note few things. The name of the pad is very crucial. It “MUST MATCH” the number property of the corresponding pins in the Schematic Symbol.

- To open the schematic of “Regulator IC”. Go to →OrCAD 16.0→OrCAD Capture CIS→File→New→Project→ Give name Regulator→click **Analog** or **Mixed A/D**→OK→**Create a blank project**→OK.

- Click Place Part
- To search any component, click Part search, write \*LM317\* in part name box and then click begin search.

**[Important Note: library Path:**

**C\OrCAD\OrCAD\_16.0\tools\capture\library]**

- Double Click on the Schematic of LM317, Click on Pins tab at the bottom side of framework window.

*[Property Editor framework will appear]*

- The numbering of pins of Regulator IC are 1,2,3 but this is not always the case.
- Refer to the datasheet of “Regulator IC” the adjacent pins space is given from 90 mils to 110 mils. We are using the mean spacing i.e. 100 mils.
- Open the spreadsheet→Foot-prints.

It is shown that pin1/pad1 (x,y) location is (0,0). Always place pin1/pad1 at (0,0).

- To create a new pin, just highlight/select pin1 in spreadsheet footprint & type CTRL+C this will open Add Pad dialogue box.
- Type “2” in the pad name because the next pin number in schematic symbol is 2. Type “100” as the X-Coordinate & “0” to Y→OK.
- This will create second Pad 100mils apart from origin i.e. pad1.
- Choose the PSU\_HOLE padstack for the pin. In most cases, you will leave the other settings as they are by default. Add the third pin in similar way by making X “200 mils” & Y “0”.
- The spreadsheet should now look like this:  
*[Footprints framework window will appear]*
- To place the outline of “New Footprint”, select Obstacle tool from Menubar→go to Tools→Obstacle→New→Make outline around the three pins/pads then double click mouse→Edit Obstacle window will appear.
  - Select Obstacle Type : Place Outline
  - Select Obstacle layer : Global Layer
  - Click “OK”→ End Command.

## **Lab Session 06**

### **OBJECTIVE:**

- **Overview of Autorouting using Orcad Layout.**
- **Defining DRC.**
- **Making Copper Pour.**

### **AUTOMATIC ROUTING USING OrCAD LAYOUT:**

OrCAD Layout supports Automatic Routing (Autorouting) of components, board and DRC.

- Component routing will help to route the nets associated with the selected component.
- Board autorouting will route all the nets on the complete board.
- DRC will implicate that all the nets within the DRC box which is defined by the user are routed.

### **AUTOROUTING OF A COMPONENT OR BOARD:**

1. From the *Auto* menu, choose *Autoroute > Component*.
2. Select the component that you want to route. All the nets connected to that component are routed.
3. Similarly, to automatically route a complete board, choose *Board* from the *Autoroute* submenu. To autoroute region choose *DRC* from the *Autoroute* submenu.

Auto route helps us in complex circuits to show the suggested path of nets.

### **DEFINING DRC BOX:**

By using DRC box, we can define the place where we want to initiate routing. Once we initiate the autoroute (available in Layout and Layout Plus) it will automatically start routing the board at the area you designate. If we are manually routing, Layout zooms in to the area encompassed by the DRC box and centers it on the screen.

### **TO DEFINE A DRC BOX:**

1. If the current DRC box is not showing, choose the online DRC from toolbar buttons, and then choose the refresh all from toolbar buttons. The current DRC box displays.

2. From the View menu, select Zoom DRC/Route Box. The cursor will change to “Z.”
3. Click the left mouse button at one corner of the box you would like to define, and while holding down the left mouse button, drag the cursor to the opposite corner of the area you would like to define, and then release the left mouse button.
4. Layout zooms in on the area, centering it on the screen.

#### **TO MOVE A DRC BOX:**

1. If the current DRC box is not displaying, choose the online DRC toolbar button, then choose the refresh all toolbar button. The current DRC box displays.
2. From the View menu, choose Zoom DRC/Route Box. The cursor changes to “Z.”
3. Move the cursor to the target location and click the left mouse button. Layout zooms in at the new location, centering it on the screen.

#### **COPPER POUR:**

A copper-filled zone on the board that features automatic voiding where there are tracks or pads. Tracks can pass through it. Copper pour can be used for noise suppression, shielding, to draw heat away from components that tend to get hot, or to isolate signals. It can be assigned to a net or attached to a component pin. It doesn't affect placement. It can be filled with hatched lines or it can be solid. It re-pours when you choose the refresh all toolbar buttons.

## **Lab Session 07**

### **OBJECTIVE:**

- **Manual Routing using OrCAD Layout Tools.**

### **USING OF MANUAL ROUTING TOOLS:**

- By using add/edit route mode we can make new tracks from a ratsnest. Without using unrout option we can edit or modify already existing tracks by placing the cursor on any routed segment or vertex and then click the left mouse button.
- By using edit segment mode we can move existing segments of tracks, can create new segments, or can remove the segment.
- The Any Angle Corner option allows us to make an angle of any kind. When we select this option, the connection segment attached to the routing tool's crosshairs rotates freely through 360°.
- The 135° Corners option allows us to create angles of 90° or 135° while we route.
- The 90° Corners option restricts angles to 90° only.
- The Curve Corners option allows us to place curved tracks on our board while we route manually. While routing using this tool we can create vertical, horizontal and curved tracks (however, we cannot freely create 135° angles with this option selected).

Select Edit Segment Option from Menu Bar → Then from Options (Menu Bar) → Route Settings from drawing method selects the type of Corner (Any angle Corners/135 Corners/90 Corners/Curve Corners).

- We can practice the add/edit route mode to route new tracks and edit tracks already exists. If we select a partially routed track, we can continue routing the track, one segment at a time, at an angle of 90° or 135°. When you select a track at a location where there is copper on more than one layer, the router edits the track that is on the current layer.
- If you choose an existing track, press the SPACEBAR button, and type a layer number (for example 1 or 2), the track switches to the new layer, and vias will be installed automatically where necessary. If it is not possible to clear room for the vias, the router reacts with beeps and does not switch the track. (Note: By default, DRC (Design Rule Check) is always on for routing.)

### **MANUAL ROUTING:**

- Choose the add/edit route toolbar button.
- Choose the zoom in toolbar button, and then click the left mouse button to magnify the area to route. Press ESC to exit zoom mode.
- Select a ratsnest with the left mouse button. The ratsnest attaches to the pointer.
- Drag the pointer to draw a track on the board.

- Click the left mouse button or press the SPACEBAR to create vertices (corners) in the track.
- When drawing the last segment for the connection, choose Finish from the popup menu. The track automatically connects to the centre of the pad. A complete connection is indicated by the cursor changing size and the ratsnest disappearing from the pointer.

### **BOARD OUTLINE CREATION:**

Layout requires exactly one board outline, on the global layer.

### **TO CREATE BOARD OUTLINE:**

1. From the Tool menu, choose Dimension, and then choose Datum. Click on the lower left corner of the board outline to place the datum (to provide a starting grid for component placement). Press HOME to redraw the screen.
2. Choose the obstacle toolbar button.
3. From the pop-up menu, choose New, then from the pop-up menu; choose Properties. The Edit Obstacle dialog box displays.
4. From the Obstacle Type drop-down list, select Board outline.
5. In the Width text box, enter a value for the outline's width.
6. From the Obstacle Layer drop-down list, select Global Layer, then choose the OK button. The Edit Obstacle dialog box closes.
7. Move to the point on the board at which you want to start drawing the outline, then click the left mouse button to insert the first corner.
8. Continue clicking the left mouse button to insert corners. After you click to insert the last corner, choose Finish from the pop-up menu.
9. Layout automatically completes the board outline.

### **TO SET MEASUREMENT UNITS:**

Open your board in Layout.

1. From the Options menu, choose System Settings. The System Settings dialog box displays.
2. Select the measuring's to mils, inches, microns, millimetres, or centimetres.
3. Choose the OK button.

### **SETTING SYSTEM GRIDS:**

Using the System Settings dialog box, you can set five distinct grid settings.

#### **Here are some rules of thumb for setting the grids:**

- For efficient routing performance, the routing grid and via grid should have the same value.
- The place grid must be a multiple of the routing and via grids.
- The routing grid should never be less than 5 mils.
- The detail grid can be set as low as 1 mil for better resolution.
- Components are placed on the place grid using the component datum, which is typically pad 1 (unless the component has been modified).



### **TO SET SYSTEM GRIDS:**

1. From the Options menu, choose System Settings. The System Settings dialog box displays.
2. Set these options, and then choose the OK button.
  - **Visible grid** assigns a display grid based on the X and Y coordinates (for example, if you're using mils, a setting of 200 would place a grid dot at every 200 mils).
  - **Detail grid** assigns a drawing grid (for lines and text) based on the X and Y coordinates.
  - **Place grid** assigns a component placement grid based on the X and Y coordinates. For greatest routing efficiency, this value needs to be a multiple of the routing grid. The datum, or origin, of footprints is constrained to this grid.
  - **Routing grid** assigns a grid used for routing (see the routing grid chart below for suggested settings).
  - **Via grid** assigns a grid upon which you or the router can place vias.

### **DEFINING GLOBAL SPACING VALUES:**

Global spacing values set rules for spacing between the various objects on the board.

### **TO DEFINE GLOBAL SPACING VALUES:**

Choose the spreadsheet toolbar button, choose Strategy, then choose Route Spacing. The Route Spacing spreadsheet displays.

1. Double-click on the layer you want to modify. The Edit Spacing dialog box displays.
2. Set these options, and then choose the OK button.
  - **Track to Track Spacing:** Tracks are defined as any routed connections and copper obstacles (such as keep outs and place outlines). Track-to-track spacing specifies the minimum space required between tracks of different nets, and between tracks and obstacles of different nets.
  - **Track to Via Spacing** Track-to-via (and obstacle-to-via) spacing specifies the minimum space required between vias and tracks of different nets.
  - **Track to Pad Spacing** Track-to-pad (and obstacle-to-pad) spacing specifies the minimum space required between pads and tracks of different nets.
  - **Via to Via Spacing** Specifies the minimum space required between vias of different nets.
  - **Via to Pad Spacing** Specifies the minimum space required between pads and vias of the same net (as well as different nets, which is the usual case). For instance, to keep a distance of 25 mils between your SMT pads and the fanout vias connected to the pads, set Via to Pad Spacing to 25.
  - **Pad to Pad Spacing** Specifies the minimum space required between pads of different nets.

## **Lab Session 08 (a)**

### **OBJECTIVES**

Introduction to Post Processing

### **POST PROCESSING:**

This section introduces some of the tasks that are not a part of the placement and routing process, but are related and can be performed using OrCAD Layout.

### **Renaming components:**

After you have completed the placement and routing of your PCB board, you can rename the components on the PCB board in a specific order.

1. From the *Options* menu, choose *Components Renaming*. The Rename Direction dialog box appears.
2. Select one of the renaming strategies. For the full adder design, select *Right, Down*.
3. Click OK.
- 4 From the *Auto* menu, choose *Rename Components*. Layout renames the components. The reference designators for the component on the board changes.

### **Back annotation:**

While creating a PCB board, you might make some changes in the layout board (.MAX) file. As a result, the board file and the design file in Capture may be out of sync. To ensure that both these file are in sync, you can backannotate the changes in the PCB board file to the Capture.

When you backannotate, information, such as component location and component names (changed due to renaming) gets added on to the schematic in Capture.

To generate the backannotation file:

- 1 From the *Auto* menu in Layout, choose *Back Annotate*.
- 2 A message box indicating the location of FULLADD.SWP file appears. Click OK. The .SWP file generated by Layout after backannotation is read by OrCAD Capture.

To backannotate the changes to the schematic:

- 1 Open FullAdd.opj in Capture
- 2 In the Project Manager window, select fulladd.dsn.
- 3 From the *Tools* menu in Capture, select *Back Annotate*.

- 4 In the Backannotate dialog box, select the *Process entire design* option button.
- 5 Select the *Update Occurrences* option button.
- 6 Specify the location of the .SWP created by Layout.
- 7 Click OK.

The schematic is updated with the changes in the board file. Similarly, if the board file is open in Layout and you make changes in the schematic design, you can ensure that these changes are forwarded to the board during Layout netlist creation.

To do this:

- 1 In the Project Manager window, select thefulladd.dsn.
- 2 From the *Tools* menu, choose *Create Netlist*.
- 3 In the Layout tab of the Create Netlist dialog box, select the *Run ECO to Layout* check box.
- 4 Click OK. The changes in the schematic design will appear in the board file.

### **Generating output:**

The final task in creating a board design is to generate output files. You can create Gerber files, drill files, DXF files, and printer/plotter files.

Before you generate reports and output files, you should clean up the design. To clean up your design:

1. From the *Auto* menu, choose *Cleanup Design*. The Cleanup Design dialog box appears.
2. In the Cleanup Routing section, click the Select All button.
3. In the Cleanup Database section, select all three check boxes, to ensure that unused padstacks, footprints, and Nets are removed.
4. Click OK. Message boxes appear indicating the cleanup process being performed. You can now generate the desired output files and reports.

### **Output files:**

Using OrCAD Layout, you can generate various files that can further be used with various third-party tools, such as GerbTool, IntelliCAD, VisualCAD, AutoCAD, and so on.

To generate these output files, complete the following steps:

1. From the *Options* menu, choose *Post Process Settings*.  
The Post Process spreadsheet appears.

Post Process				
Plot output File Name	Batch Enabled	Device	Shift	Plot Title
*.TOP	Yes	EXTENDED GERBER	No shift	Top Layer
*.BOT	Yes	EXTENDED GERBER	No shift	Bottom Layer
*.GND	Yes	EXTENDED GERBER	No shift	Ground Layer
*.PWR	Yes	EXTENDED GERBER	No shift	Power Layer
*.IN1	No	EXTENDED GERBER	No shift	Inner Layer 1
*.IN2	No	EXTENDED GERBER	No shift	Inner Layer 2
*.IN3	No	EXTENDED GERBER	No shift	Inner Layer 3

2. Select the Device column.
3. Right-click and select *Properties* from the pop-up menu. The Post Process Settings dialog box appears.
4. Select the required options.
  - To create files to be used with Gerber tools, select Gerber RS-2740 or extended Gerber.
  - To create files with mechanical information that is to be used with the CAD tools, select DXF.
  - To create HPGL files select Print Manager. To create HPGL file, you must have the HPGLprinter installed.
  - In the Post Process Settings dialog box, select theExtended Gerber check box.
5. Select the Create Drill Files, Overwrite Existing Files, and Enable for Post Processing option buttons.
6. Click OK.

Gerber files are generated.

7. Close the Post Process spreadsheet. You can also generate output files using the Run Post Processor command.
  - i. From the *Auto* menu, choose *Run Post Processor*.
  - ii. A message box appears indicating the generated filename. Click OK.

After Layout creates the post processing files, a post processing log file displays.

## **Lab Session 08 (b)**

### **OBJECTIVES**

Introduction to Gerbtool

### **GERB TOOL:**

GerbTool provides a powerful set of Windows-based CAM tools, including a feature-rich and robust Gerber/NC editor for ensuring a seamless link between PCB design and manufacturing. GerbTool is designed to provide CAD/CAM professionals with the tools they need for complete control over their CAM databases.

From visual verification to high-level CAM tools, GerbTool simplifies and automates your PCB layout post processing and pre-manufacturing tasks. GerbTool's consistent and intuitive graphical user interface, and programmable mouse buttons and function keys, allow you to focus on accomplishing tasks, rather than on the technical details of operating the software.

### **GerbTool features:**

- Easy to use, Fast and unlimited file sizes. Accurate to 1/100 mil (.00001 in.).
- Fully automatic panelization and venting.
- Complete undo to beginning of session.
- Full design rule checking (DRC), including annular ring checking and stub detection. Snoman™ pad/trace filleting.
- Teardrop pads. Isolated pad removal.
- NC drill optimizing, including step and repeat.
- Automatic removal of silkscreen data from pads.
- Full support for true multilayer netlists, including net highlighting.
- Scalable check plots to HPGL, PostScript®, Laser printers, and all printers/plotters supported by Windows.
- Conversion of drawn pads to flashes.
- Macro language allows the addition of new commands.
- Metric and Imperial formats supported.
- Photoplotter support includes extended Gerber, FIRE9xxx, EIE, BARCO DPF and IPC-D-350.
- Accurate display of power and ground plane composites.
- Allows aperture scaling to create soldermasks, shrink/expand traces, and so on.
- Ability to scale layers to shrink or expand the database.
- Merge a complete design or a single Gerber file into another.
- Import NC Drill, HPGL, or BARCO files.
- View up to 999 layers simultaneously.
- Handles over 4000 apertures in up to 999 aperture lists.
- Aperture list conversion tools allow the addition of custom aperture list converters.
- Easily created custom apertures and custom fonts.

## **Electronic Engineering Drawing & Workshop Lab Session 09**

*NED University of Engineering and Technology- Department of Electronic Engineering*

### **Lab Session 09 (a)**

**Practical on the identification of Drill Bits and Drilling on the manual drill Machine**

#### **OBJECTIVES:**

Upon the completion of the experiment, you will be able to:

1. Know about the sizes of the different drill bits for drilling holes in PCB.
2. Perform the drilling of holes of any suitable size on a PCB.

#### **BACKGROUND:**

The drill bits come in different sizes as the boards vary in the size too. The drill bit can be less than a fraction of mm to any No. of millimeters. The minimum size of the drill bit is related to the technology. If the technology is so precise than the drill bit size goes on reducing to the limitations permitted by the resources at hand and the machinery involved. As the drill bit size is reduced and the automation is introduced so is the size increased. If someone wants to perform the drilling on the CNC (COMPUTER PNEUMATIC CONTROL) machine, it is more precise, especially in the working of IC pads. As the sizes involved are mils only.

So it also depends upon how you plan to go ahead. First of all, the person who wants to drill must do sketches on the copper board if he is performing the drilling manually. If he is precise at the handling of the machine then he can work without the sketching and can use any other method for getting the exact point where he has to drill a hole.

Normally the manual machine has a motor attached to it that revolves around and there is a slot vacant for inserting the drill bit.

#### **CNC DRILLING:**

The techniques for drilling copper clad for double-sided and multilayer PCBs with automated equipment are identical, with the exception that multiple drilling steps will be needed if your multilayer design includes buried or blind vias. Refer to the documentation that came with your drilling machine for more information (standard boilerplate copy-out). Items to remember include:

- Set the STACK HEIGHT parameter to clear all dowel pins during traverse
- Set the SPINDLE FEED (inches per minute) and SPINDLE SPEED (RPM for each drill size to values consistent with drilling standard 0.062" (1.6mm) FR-4 copper clad.
- Set the SPINDLE PLUNGE DEPTH so that the tip of the largest diameter drill bit fully enters the backing material. Otherwise, these large diameter holes will not totally penetrate to bottom laminate and exit foil.

- DO NOT contour route the board immediately after drilling the stack. This should only be done after all other processing is complete.

### **MANUAL DRILLING:**

- With the laminate stack formatted as detailed above, manual drilling is a straightforward, if somewhat mind-numbing process. Items to consider include.
- When using a conventional drill press, hole placement accuracy can be improved and drill breakage minimized through the use of a “sensitive drilling” or “finger” chuck. Small format, precision high speed drill precision, ideal for PCB fabrication, is also available from a number of sources.
- Regardless of the type of drill press being used, a pressure foot should be employed if available.
- If available, position a work lamp on a flexible mount as close to the work surface as possible.
- Although more brittle than conventional high speed steel (HSS) drills, tungsten carbide bits designed specifically for PCB drilling will yield far superior hole wall quality. Minimize burr formation, and outlast HSS bits almost 10 to 1. The downside is that, with smaller break and must be handled carefully.
- Always use drill bits that have been fitted with depth setting rings. This will allow you to set the plunge depth stop on your drill press to a single value that will work for all bit
- Diameters.
- Prepare a chart that links the various diameter bits with the symbols used in the drill master.

### **THROUGH-HOLES:**

1- Load the largest diameter bit to be used into the drill chuck, making sure that the depth ring is pressed firmly against the ends of the chuck jaws when they are fully tightened.

2- Using a piece of scrap backing material as a gauge, adjust the spindle travel stop on your drill press to a depth that insures that the entire tip of the drill bit penetrates at least half of the material’s thickness. You can also use two pieces of entry foil as a feeler gauge” to set the depth. Under no circumstances allow a PCB drill bit to drill into the table of your drill press. PCB bits are specifically designed to drill copper clad and will shatter if plunged into cast iron, steel, or aluminum.

3- Starting with largest diameter drill bit, drill all of the through holes, stopping periodically to insure that the drill bit have not snapped off and that the spindle travel stop has not slipped.

4- As you drill each hole size (from the largest to the smallest) check off that diameter on the drilling chart. This is a good bookkeeping technique that will help you keep track of your progress and insure that no hole size is missed.

5- After all of the holes have been drilled, remove the backing material from the stack and re-tape the remaining sheet with the dowel pins in places.

6- Hold the stack up to the light for visual inspection. Ascertain that all of the holes have been drilled through and that none are blocked by drill debris. If some debris is seen, remove by carefully pushing a smaller diameter drill bit through the hole.

7- If all of the holes ;in your circuit design go all the way ;through the board, you are now ready to activate hole walls to prepare for through-hole plating.

### **BLIND OR BURIED VIAS:**

Designs that sue blind or buried vias (vias that do not penetrate through the PCB) need supplementary drilling operations before proceeding. Unfortunately, they are also quite a bit more difficult to activate and through plate since each must be processed singly.

1. Fully disassemble the drilled stack.
2. Reassemble a sub stack consisting of the backing sheet, one of the copper clad substrates that need additional drilling, and the entry foil that carries the drillmaster.
3. Re-pin with the dowels and tape as before.
4. Playing close attention to the drill master symbols representing the holes needed by the included substrate, drill the sub stack.
5. Disassemble the sub stack and repeat steps 2 through 4 for each layer that needs further drilling.
6. Inspect each layer after it is drilled and remove any debris that might be blocking the holes.
7. If all of the holes are drilled to your satisfaction, the individual layers are now ready for activation.



## **SUMMARY:**

- When using a conventional drill press, hole placement accuracy can be improved and drill breakage minimized through the use of a “sensitive drilling” or “finger” chuck. Small format, precision high speed drill presses, ideal for PCB fabrication, is also available from a number of sources.
- Regardless of the type of drill press being used, a pressure foot should be employed if available.
- If available, position a work lamp on a flexible mount as close to the work surface as possible.
- Although more brittle than conventional high speed steel (HSS) drill, tungsten carbide bits designed specifically for PCB drilling will yield far superior hole wall quality, Minimize burr formation, and outlast HSS bits almost 10 to 1. The downside is that, with smaller diameters [0.018” (0.46mm) and less], the carbide drills are easier to break and must be handled carefully.
- Always use drill bits that have been fitted with depth setting rings. This will allow you to set the plunge depth stop on your drill press to a single value that will work for all bit diameters.
- Prepare a chart that links the various diameter bits with the symbols used in the drillmaster.

## **TESTS AND MEASUREMENTS:**

### **MINIMUM DRILL HOLE SIZE, PCB THICKNESS:**

S. No.	PCB THICKNESS MM	PCB THICKNESS INCHES	DRILL HOLE SIZE MM	DRILL HOLE SIZE INCHES

### **PRELIMINARY INSTRUCTIONS:**

1. Always wear safety glasses when operating a drill press, especially if you are drilling with carbide PCB drill bits.

2. If available, always use a vacuum cleaner to remove debris and collect airborne dust during the drilling operation.
3. The dust generated during PCB drilling can pose a very; serious health hazard and should not be inhaled or ingested under any circumstances.

### **EXPERIMENTAL PROCEDURE:**

#### **Through-holes:**

1. Load the largest diameter bit be used into the drill chuck, making sure that the depth ring is pressed firmly against the ends of the chuck jaws when they are fully tightened.
2. Using a piece of scrap backing material as a gauge, adjust the spindle travel stop on your drill press to a depth that insures that the entire tip of the drill bit penetrates at least half of the material's thickness. You can also sue two pieces of entry foil as a "feeler gauge" to set the depth. Under no circumstances allow a PCB drill bit to drill into the table of your drill press. PCB bits are specifically designed to drill copper clad and will shatter if plunged into cast iron, steel, or aluminum.
3. Starting with the largest diameter drill bit, drill all of the through holes, stopping periodically to insure that the drill bit have not snapped off and that the spindle travel stop has not slipped.
4. As you drill each hole size (from the largest to the smallest) check off that diameter on the drilling chart. This is a good bookkeeping technique that will help you keep track of your progress and insure that no hole size is missed.
5. After all of the holes have been drilled, remove the backing material from the stack and re tape the remaining sheets with the dowel pins in place.
6. Hold the stack up to the light for visual inspection. Ascertain that all of the holes have been drilled through and that none are blocked by drill debris. If some debris is seen, remove by carefully pushing a smaller.
7. If all of the holes in your circuit design go all the way through the board, you are now ready to activate the hole wall to prepare for through-hole plating.

#### **BLIND OR BURIED VIAS:**

8. Designs that sue blind or buried vias (vias that do not penetrate through the PCB) need supplementary drilling operations before proceeding. Unfortunately, they are also quite a bit more difficult to activate and through plate since each must be processed singly.
9. Fully disassemble the drilled stack.
10. Reassemble a sub stack consisting of the backing sheet, one of the copper clad substrates that need additional drilling, and the entry foil that carries the drillmaster.

11. Re-pin with the wholes and tape as before
12. Paying close attention to the drillmaster symbols representing the holes needed by the included substrate, drill the sub stack.
13. Disassemble the sub stack and repeat steps 2 through 4 from each layer that needs further drilling.
14. Inspect each layer after it is drilled and remove any debris that might be blocking the holes.
15. If all of the holes are drilled to your satisfaction, the individual layers are now ready for activation.

## **Lab Session 09 (b)**

**To understand the electroplating processes on ABC plating line in  
COMPACTA machine BUNGARD**

### **OBJECTIVE:**

To learn the principles and methods involved in electroplating.

### **BACKGROUND:**

There is a sequence of Baths involved in plating system:

The total No. of bathes is six (6).

Bath = 1: This Bath is for cleaning and conditioning

Bath = 2: This bath is for Pre-Dip. We must use the pre-Dip solution for cleaning tank 3.

Bath = 3: Catalyst: To stir the bath, always use a very clean glass or plastic rod.

Bath = 4: Intensifier.

Bath = 5: Spare bath.

This spare bath can be used for electro less tin plating at the room temperature. If not in use, please fill this tank with water.

Bath = 6: Copper plating: Fix both anode holders, use anode bags to cover anodes and use the strings to form a knot so that the bags are kept in place.

For proper copper plating results, it is necessary to run the anodes under working conditioned but with reduced current of 1 A/dm .sq.

The COMPACTA ABC unit is equipped with triple cascade rinse section. Pre rinse always in the bath with the highest water level followed by the one with the lower level.

### **SUMMARY:**

We must connect the proper supply to the plating unit. The cable should be placed in a proper way, Remove the plug if you are not using the machine. Take special care that the liquid must not pass the housings, set up the appliance in a proper room. Close the drain valves before filling the tanks. The temperature of the solution can differ from the actual temperature

Wear goggles and protective glasses for work. In the system, the conveyer arms are removable and adjustable. Here is a front rinsing system for the rinsing of PCB at different stages. There are five treatment tanks in the system. They can be divided into

1. Treatment tanks
2. Electroplating tanks
3. Control section

## **TESTS & MEASUREMENTS:**

### **1. The treatment Dimensions:**

Internal Dimensions:

300\*100\*400 mm (D\*W\*H)

Content: approx. 10 Liters

Two tanks are equipped with PTFE coated heating elements: 220 V, 400Watt

The first tank can be heated upto: 70 Degree Celsius

The first tank can be used upto: 50 Degree Celsius

### **2. Electroplating Tank:**

Internal Dimensions:

400\*275\*400 (D\*W\*H)

Capacity: Approx 30 Liters.

### **3. Control Section:**

Two thermostats with a switch, one air pump 400l/h with switch, main switch, 5 electronics timers, Conveyor potentiometer with switch, rectifier adjustable up to 6V, 40A. And switch, internal fuses.

The cleaning unit is made of PVC.

## **PRELIMINARY INSTRUCTIONS:**

### **Safety Instructions:**

- Do not use for any other application than through hole plating.
- Read all safety instructions. Take extra care that there is no extra humid or wet. Environment.
- To avoid electric shock do not remove the housing.
- Keep the safety and operating instructions somewhere safe in use.
- In your interest, pay attention to all safety warnings.
- Whenever you use the unit ensure that there is the sufficient ventilation in the room.

## **EXPERIMENTAL PROCEDURES:**

- Cut the PCB to size with board cutter. The blank size must be 20mm larger.
- Drill your PCB board to required hole pattern. Allow 0.05 to 0.01 mm extra dia for the drill bits.
- Fix the cleaned board in the MACHINE. Fix it in the three-finger board holder. Start from bath 1 (Left side)
- Process the board as per instructions from bath 1 to 6 with the sequence and timings as per the instructions in the machine manual.

## **Lab Session 09 (c)**

### **PRACTICAL ON PLOTTING, FIXING AND DEVELOPING A FILM FOR EXPOSING OF PCB'S**

#### **OBJECTIVES**

Upon the completion of this experiment we will be able to

1. Plot the film using photo plotter unit for exposing of PCB.
2. The fixing and development of the film.

#### **BACKGROUND**

For plotting of film a photo plotter unit is used. It exposes the film using a diode laser. The unit has an external power supply and is connected to PC via parallel port.

For photo plotting a dark room must be arranged, with a special green safe light illumination for standard development and fixing process. Therefore a unit must be installed for development and fixation of the film, consisting of three tanks one each for development, fixation and rinse.

The extent of delivery of photo plotter comprises of three software programs. The 1<sup>st</sup>, Gerb2Bitmap, serves for data preparation. This software exports the data in a proprietary FPF format for use on the plotter driver software. Further use of this software is i.e. to mount several layouts on one sheet of film, creation of step and repeat artwork. The second Run Plotter takes the so prepared data to control the plotter via the parallel port of your PC. It reads bitmap data in the above mentioned FPF format or at user's choice in windows BMP or EAGLE TIFF format. In Run Plotter, you may select the output resolution as well as positive or negative, direct or mirrored output.

The third application supplied is View Mate, a Gerber viewer and aperture converter tool. A built-in aperture converter helps transform the aperture tables of all CAD and CAM software into a standard format suitable for use with Gerb2Bitmap.

#### **OBSERVATION**

**Plotting:**

Material	Size( sq. mm)	Plot Area (sq. mm)
----------	---------------	--------------------

**Fixing and development:**

Chemical	Concentration
----------	---------------

## **SUMMARY**

Enter the dark room mount the film in line with white arrow on the drum of the photo plotter, with some tape. The emulsion side must be facing the drum and sheet edges must be parallel to drum axis. Close the lid and leave the dark room.

Start the “Run\_Plotter” software, set image quality and size, and run the photo plotter software. The drum starts rotating and the plotting starts. The computer shows the estimated time countdown i.e. how much time is left in completion of plotting. After plotting is complete, enter the dark room, open the lid and remove the film from the drum and start the development of the film.

The next part is the development and fixing of the film. Place the film removed from the photo plotter in the tank containing fixer. The film is then rinsed for 10 sec in tank containing water and after rinsing the film is dried. The film is then placed in developer solution for 30 seconds. After it rinse again and expose it to sunlight for few seconds.

### **TESTS AND MEASUREMENTS:**

- Light Source (Laser Diode): \_\_\_\_\_;  $\lambda =$  \_\_\_\_\_
- Resolution X: \_\_\_\_\_.
- Resolution Y: \_\_\_\_\_.
- Light Source (Dark room): \_\_\_\_\_.
- Speed of Plotting (per second): \_\_\_\_\_.

### **PRELIMINARY INSTRUCTIONS:**

1. The film must not be exposed to normal light before its development and fixing, because the film consist of crystals of silver halides, which when exposed to light form an image, and as normal light falls on the screen all the crystals are exposed to light and the film gets black.
2. The film must not be twisted or bent and must be parallel to drum axis otherwise the image will be distorted.
3. The parallel port connecting PC to photo plotter unit must not be loose.
4. Film from different manufacturers show different light intensity. Therefore it is necessary to adapt that light intensity.
5. Keep the drum surface clean. Do not use spray cleaners for removing residues of tape.
6. During plotting process the film is exposed to the laser light. The light can injure the human eye. Never open the plotter lid during operation.
7. The emulsion side must be facing the drum.

## **PROCEDURE**

### **a) Plotting:**

Enter the dark room turn the green safe light ON and make sure there is no extra light from outside dropping in take a sheet pf film from its box and mount it in line with white arrow on the drum of the plotter, with some tape, which itself is in line with white arrow on the left of the drum housing.

The emulsion side as mentioned earlier must be facing the drum, under green light, the emulsion side of the film normally looks grey, and the opposite side is darker. After fixing upper edges turn the drum by hand and sweep over the sheet so it goes tightly on the drum, and is parallel to it and that the upper and the lower sheet corners are facing each other.

Close the lid and leave the dark room. Start the “Run Plotter” software, set image quality and size, (the output X resolution that can be set is 1016, 1355, 2032 and 4064, the Y resolution is fixed on 3000dpi) and run the photo plotter software. The drum starts rotating and after reaching normal speed the computer starts plotting with red diode laser. The red LED blinks according to the ON/OFF condition of the laser head. The compute shows the estimated time countdown i.e. how much time is left in completion of plotting. After plotting is complete, enter the dark room, open the lid and remove the film from the drum and start the development of the film.

### **b) Film Development and fixing:**

The next part is the development and fixing of the film. Place the film removed from the photo plotter in the tank containing fixer.(with a concentration of 1 part of fixer chemical and 2 parts of water) for 30 sec. The film is then rinsed for 10 sec in tank containing water. This step is taking because the developer will be destroyed if fixer drops in it.

After rinsing the film dry it, place it on the smooth surface; wipe the 1<sup>st</sup> side with a smooth fresh tissue and similarly the 2<sup>nd</sup> side. The film is then placed in developer solution (with concentration 1 part of developer and 3 part of water) for 30 sec. After it, rinse again and expose it to sunlight for few seconds.



## **Lab Session 10(a)**

**Objective:** Introduction to MATLAB and MATLAB GUI

### **Introduction:**

The name MATLAB stands for MATrixLABoratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects.







MATLAB is a high-performance language for technical computing. It integrates Computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make MATLAB an excellent tool for teaching and research.

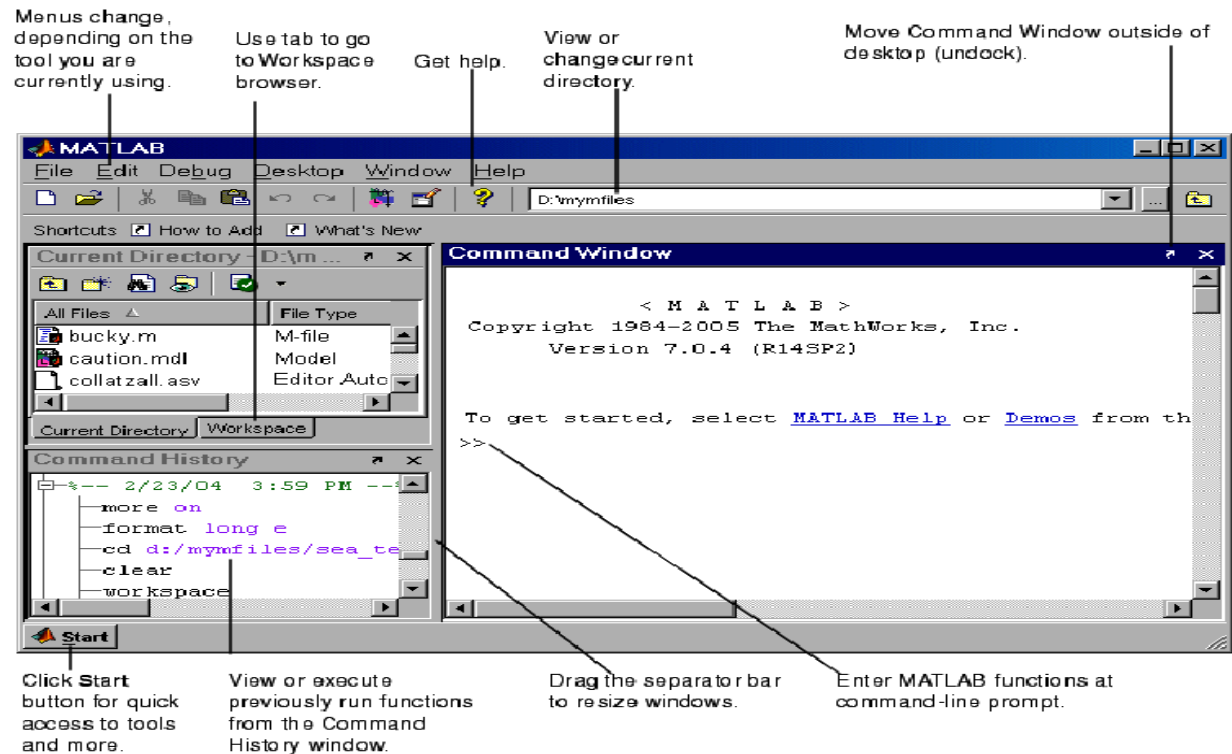
MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide.

It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

### **Starting MATLAB**

After logging into account, double-click on the MATLAB shortcut icon (MATLAB) on Windows desktop. A special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible from the desktop are:

-  The Command Window
-  The Command History
-  The Workspace
-  The Current Directory
-  The Help Browser
-  The Start button



## clc

Clear Command Window

**GUI Alternatives:** As an alternative to the `clc` function, select **Edit > Clear Command Window** in the MATLAB desktop.

**Syntax:** `clc`

**Description:** clears all input and output from the Command Window display, giving a "clean screen." After using `clc`, the scroll bar to see the history of functions cannot be used, but use of the up arrow to recall statements from the command history is still possible.

## clear

Remove items from workspace, freeing up system memory

**GUI Alternatives:** As an alternative to the `clear` function, use **Edit > Clear Workspace** in the MATLAB desktop.

**Syntax:**

- ▶ `clearall`
- ▶ `clear variable`

**Description:** `clear` removes all variables from the workspace. This frees up system memory.

## LAB Session 10(b)

**Objective:** Introduction to basic operations

### **Basic Arithmetic Operators:**

Symbol	Operation	Example
+	Addition	2+3
-	Subtraction	2-3
*	Multiplication	2*3
/	Division	2/3
^	Power	2^3

### **Creating MATLAB variables**




MATLAB variables are created with an assignment statement. The syntax of variable assignment is

Variable name = a value (or an expression)

For example,

>> x = expression

Where, expression is a combination of numerical values, mathematical operators, variables, and function calls. On other words, expression can involve:

-  manual entry
-  built-in functions
-  user-defined functions

### **Precedence**

Precedence is the order of performing arithmetic operations

The hierarchy of arithmetic operations	
PRECEDENCE	MATHEMATICAL OPERATIONS
First	The contents of all parentheses are evaluated first, starting from the innermost parentheses and working outward.
Second	All exponentials are evaluated, working from left to right
Third	All multiplications and divisions are evaluated, working from left to right
Fourth	All additions and subtractions are evaluated, starting from left to right

## **Exercise**

Q1. Calculate:

$$\frac{1}{2 + 3^2} + \frac{4}{5} \times \frac{6}{7}$$

- In MATLAB
  - Manually
- a. Compare the outputs.
  - b. If there is a difference in the two outputs, explain why it is so and how it can be solved?

Q2. Implement following in MATLAB

- a. Fahrenheit to Celsius formula
- b. Lens formula
- c. 3-Equations of motion

## LAB Session 11

**Objective:** Introduction to Mathematical functions, Variables and Matrix operations

### **Mathematical functions**

MATLAB offers many predefined mathematical functions for technical computing which contains a large set of mathematical functions. Typing `help elfun` and `help specfun` calls up full lists of elementary and special functions respectively. There is a long list of mathematical functions that are built into MATLAB. These functions are called built-ins. Many standard mathematical functions, such as  $\sin(x)$ ,  $\cos(x)$ ,  $\tan(x)$ ,  $e^x$ ,  $\ln(x)$ , are evaluated by the functions `sin`, `cos`, `tan`, `exp`, and `log` respectively in MATLAB.

Following are some commonly used functions, where variables  $x$  and  $y$  can be numbers, vectors, or matrices.

Table 2.1: Elementary functions

<code>cos(x)</code>	Cosine	<code>abs(x)</code>	Absolute value
<code>sin(x)</code>	Sine	<code>sign(x)</code>	Signum function
<code>tan(x)</code>	Tangent	<code>max(x)</code>	Maximum value
<code>acos(x)</code>	Arc cosine	<code>min(x)</code>	Minimum value
<code>asin(x)</code>	Arc sine	<code>ceil(x)</code>	Round towards $+\infty$
<code>atan(x)</code>	Arc tangent	<code>floor(x)</code>	Round towards $-\infty$
<code>exp(x)</code>	Exponential	<code>round(x)</code>	Round to nearest integer
<code>sqrt(x)</code>	Square root	<code>rem(x)</code>	Remainder after division
<code>log(x)</code>	Natural logarithm	<code>angle(x)</code>	Phase angle
<code>log10(x)</code>	Common logarithm	<code>conj(x)</code>	Complex conjugate

### **Types of Variables:**

A MATLAB variable is essentially a tag that assigns to a value while that value remains in memory. The tag gives a way to reference the value in memory so that programs can read it, operate on it with other data, and save it back to memory.

MATLAB provides three basic types of variables:

- 📌 Local Variables
- 📌 Global Variables

### **Local Variables**

Each MATLAB function has its own local variables. These are separate from those of other functions (except for nested functions), and from those of the base workspace. Variables defined in a function do not remain in memory from one function call to the next, unless they are defined as global.

Scripts, on the other hand, do not have a separate workspace. They store their

variables in a workspace that is shared with the caller of the script. When called from the command line, they share the base workspace. When called from a function, they share that function's workspace.

### **Global Variables**

If several functions, and possibly the base workspace, all declare a particular name as global, then they all share a single copy of that variable. Any assignment to that variable, in any function, is available to all the other functions declaring it global.

### **Matrix Generation:**

Matrices are the basic elements of the MATLAB environment. A matrix is a two-dimensional array consisting of m rows and n columns. Special cases are column vectors ( $n = 1$ ) and row vectors ( $m = 1$ ).

#### **Entering a vector**

The elements of vectors in MATLAB are enclosed by square brackets and are separated by spaces or by commas. For example, to enter a row vector, v, type

```
>> v = [1 4 7 10 13]
v =
1 4 7 10 13
```

Column vectors are created in a similar way; however, semicolon (;) must separate the components of a column vector,

```
>> w = [1;4;7;10;13]
w =
1
4
7
10
13
```

#### **Entering a matrix**

A matrix is an array of numbers. To type a matrix into MATLAB

- ▶ begin with a square bracket, [
- ▶ separate elements in a row with spaces or commas (,)
- ▶ use a semicolon (;) to separate rows
- ▶ End the matrix with another square bracket, ].

Here is a typical example. To enter a matrix A, such as,

```
A =
1 2 3
4 5 6
7 8 9
Type,
>> A = [1 2 3; 4 5 6; 7 8 9]
```

## **Matrix indexing**

If  $i$  and  $j$  are two indices, then element of row  $i$  and column  $j$  of the matrix  $A$  is denoted by  $A(i,j)$ . Thus,  $A(i,j)$  in MATLAB refers to the element  $A_{ij}$  of matrix  $A$ . The first index is the row number and the second index is the column number.

For example,  $A(1,3)$  is an element of first row and third column.

Here,  $A(1,3)=3$ . Correcting any entry is easy through indexing. Here we substitute  $A(3,3)=9$  by  $A(3,3)=0$ . The result is

```
>> A(3,3) = 0
```

```
A =
```

```
1 2 3
```

```
4 5 6
```

```
7 8 0
```

## **Matrix Operations:**

### **Colon Operator:**

To enter a vector  $k$  from say  $x$  to  $z$  having step size of  $y$  then colon operator is used instead of typing whole matrix manually.

```
>> k = x : y : z
```

The colon operator can also be used to pick out a certain row or column. For example, the statement  $A(m:n,k:l)$  specifies rows  $m$  to  $n$  and column  $k$  to  $l$ . Subscript expressions refer to portions of a matrix.

Only  $':'$  shows whole row or column to be extracted i.e  $A(m:n,:)$  specifies rows from  $m$  to  $n$  with all columns.

### **Deleting row or column**

To delete a row or column of a matrix, use the empty vector operator,  $[]$

```
>> A(3,:) = []
```

```
A =
```

```
1 2 3
```

```
4 5 6
```

Third row of matrix  $A$  is now deleted. To restore the third row, we use a technique for creating a matrix

```
>> A = [A(1,:);A(2,:);[7 8 0]]
```

```
A =
```

```
1 2 3
```

```
4 5 6
```

```
7 8 0
```

Matrix  $A$  is now restored to its original form.

### **Dimension (size)**

To determine the dimensions of a matrix or vector, use the command `size`. For example,

```
>> size(A)
```

```
ans =  
    3 3  
means 3 rows and 3 columns. Or more explicitly with,  
>> [m,n]=size(A)
```

### **Transposing a matrix**

The transpose operation is denoted by an apostrophe or a single quote ('). It flips a matrix about its main diagonal and it turns a row vector into a column vector. Thus,

```
>> A'  
ans =  
    1 4 7  
    2 5 8  
    3 6 0
```

### **Exercise:**

1. Construct a matrix containing tables of numbers 2 to 10.
2. Solve the following system of Equations using MATLAB
  - a)  $x + 2y + 3z = 1$  ;  $3x + 3y + 4z = 1$  ;  $2x + 3y + 3z = 2$
  - b)  $3x - 4y + 5z = 6$  ;  $-4x + 9y + 8z = 2$  ;  $0.6x + 1.9y - 2.2z = -10$



## LAB Session 12

**Objective:** Array Operations  
Introduction to Programming in MATAB

### **Array arithmetic operations**

Array arithmetic operations or array operations for short, are done element-by-element. The period character ‘.’ distinguishes the array operations from the matrix operations. However, since the matrix and array operations are the same for addition (+) and subtraction (-), the character pairs (:+) and (:-) are not used. The list of array operators is shown below in Table 3.2. If A and B are two matrices of the same size with elements  $A = [a_{ij}]$  and  $B = [b_{ij}]$ , then the command

>> C = A.\*B

Produces another matrix C of the same size with elements  $c_{ij} = a_{ij}b_{ij}$ . For example, using the same 3 x 3 matrices,

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}, \quad B = \begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix}$$

As

>> C = A.\*B

C =  
10    40    90  
160   250   360  
490   640   810

.*	Element-by-element multiplication
./	Element-by-element division
.^	Element-by-element exponentiation

Table 3.1: Array operators

### **Programming In MATLAB:**

#### **M-File Scripts**

A script file is an external file that contains a sequence of MATLAB statements. Script files have a filename extension .m and are often called M-files. M-files can be scripts that simply execute a series of MATLAB statements, or they can be functions that can accept arguments and can produce one or more outputs.

## **M-File functions**

Functions are programs (or routines) that accept input arguments and return output arguments. Each M-file function (or function or M-file for short) has its own area of workspace, separated from the MATLAB base workspace.

### **Input and output arguments**

The input arguments are listed inside parentheses following the function name. The output arguments are listed inside the brackets on the left side. They are used to transfer the output from the function file. The general form looks like this

function [outputs] = function\_name(inputs)

### **Input And Output to a script file**

**input** Taking input from keyboard

**Syntax:**

Variable = input('string') ;

**Description** The response to the input prompt can be any MATLAB expression, which is evaluated using the variables in the current workspace

**disp** Display text or array

**Syntax**

disp(X)

**Description** disp(X) displays an array, without printing the array name. If X contains a text string, the string is displayed.

### **Exercise:**

1. Write a MATLAB program to find the average of two numbers input by the user.
2. Write a program to prepare HSC marksheet.
3. Write a function to calculate geometric mean of three numbers input by the user in separate program and display the output.
4. Implement three equations of motion as separate MATLAB functions and write a program to calculate Vf, S1 and S2 through these functions.

## LAB Session 13

**Objective:** Introduction to Graph plotting in MATLAB

### **Basic Plotting**

MATLAB has an excellent set of graphic tools. Plotting a given data set or the results of computation is possible with very few commands

### **Creating simple plots**

The basic MATLAB graphing procedure, for example in 2D, is to take a vector of x-coordinates,

$x = (x_1, \dots, x_N)$ , and a vector of y-coordinates,  $y = (y_1, \dots, y_N)$ , locate the points  $(x_i, y_i)$ , with  $i = 1, 2, \dots, n$  and then join them by straight lines. Prepare x and y in an identical array form; namely, x and y are both row arrays and column arrays of the same length.

The MATLAB command to plot a graph is `plot(x,y)`.

```
>> x = [1 2 3 4 5 6];  
>> y = [3 -1 2 4 5 1];  
>> plot(x,y)
```

### **Plotting a Sine Wave:**

To plot the function  $\sin(x)$  on the interval  $[0; 2\pi]$ , first create a vector of x values ranging from 0 to  $2\pi$ , then compute the sine of these values, and finally plot the result

```
>> x = 0:pi/100:2*pi;  
>> y = sin(x);  
>> plot(x,y)
```

### **Adding titles, axis labels, and annotations**

MATLAB enables to add axis labels and titles. For example, using the graph from the previous example, add x- and y-axis labels. Now label the axes and add a title. The character `\pi` creates the symbol  $\pi$

```
>> xlabel('x = 0:2\pi')  
>> ylabel('Sine of x')  
>> title('Plot of the Sine function')
```

### **Specifying line styles and colors**

It is possible to specify line styles, colors, and markers (e.g., circles, plus signs . .) using the plot command:

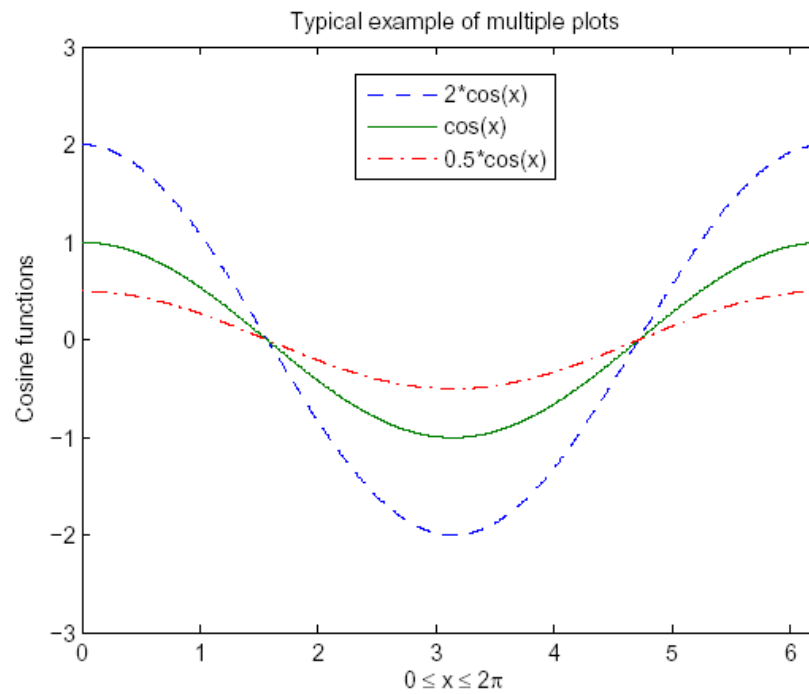
```
plot(x,y,'style_color_marker')
```

where `style_color_marker` is a triplet of values from the Table.

SYMBOL	COLOR	SYMBOL	LINE STYLE	SYMBOL	MARKER
k	Black	—	Solid	+	Plus sign
r	Red	--	Dashed	o	Circle
b	Blue	:	Dotted	*	Asterisk
g	Green	—.	Dash-dot	.	Point
c	Cyan	none	No line	×	Cross
m	Magenta			s	Square
y	Yellow			d	Diamond

### **Exercise:**

Q. Write MATLAB code for following output.



## LAB Session 14

**Objective:** Introduction to Control Flow and Loops in MATLAB

### **Control flow and operators**

MATLAB is also a programming language. Like other computer programming languages, MATLAB has some decision making structures for control of command execution. These decision making or control flow structures include for loops, while loops, and if-else-end constructions. Control flow structures are often used in script M-files and function M-files. MATLAB provides several tools that can be used to control the flow of a program (script or function). In a simple program the commands are executed one after the other. But the flow control structure that make possible to skip commands or to execute specific group of commands.

### **Relational and logical operators**

A relational operator compares two numbers by determining whether a comparison is true or false.

OPERATOR	DESCRIPTION
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
~=	Not equal to
&	AND operator
	OR operator
~	NOT operator

### **Loops:**

a) **For Loop**

Execute block of code specified number of times

**Syntax/General Format:**

```
for variable = initval:endval
    statement
...
statement
end
```

**Description** for statement repeatedly executes one or more MATLAB statements in a loop. Loop counter variable x is initialized to value initval at the start of the first pass through the loop, and automatically increments

by 1 each time through the loop. The program makes repeated passes through statements until either x has incremented to the value endval, or MATLAB encounters a break, or return instruction, thus forcing an immediate exit of the loop

**b) While Loop**

Repeatedly execute statements while condition is true

**Syntax**

while expression, statements, end

**Description** repeatedly executes one or more MATLAB statements in a loop, continuing until expression no longer holds true or until MATLAB encounters a break, or return instruction. thus forcing an immediate exit of the loop

## **Conditional Statements**

**If else statement**

Execute statements if condition is true

**Syntax**

if expression, statements, end

**Description** evaluates expression and, if the evaluation yields logical 1 (true) or a nonzero result, executes one or more MATLAB commands denoted here as statements. expression is a MATLAB expression, usually consisting of variables or smaller expressions joined by relational operators or logical functions.

Simple expressions can be combined by logical operators (&&, ||, ~) .

## **Exercise:**

1. Write a program to generate counting from 1 to 20 and then 20 to 1.
2. Write a program which takes two numbers from the user and displays the relationship b/w them.
3. Write a program which takes three numbers input from user, and displays them in ascending order.
4. Write a program to take 10 numbers input from user using 'For Loop' and display the Sum of all numbers, even numbers and odd numbers separately.

## Lab Session 15

**Objective:** Introduction to LabView

### **Introduction**

LabVIEW programs are called virtual instruments, or VIs, because they appear as physical instruments, such as oscilloscopes and multi-meters. LabVIEW VIs contain three components-the front panel, the block diagram, and the control and function palette. You can use LabVIEW to communicate with hardware such as data acquisition, vision, and motion control devices

In this Lab, you will build simple VIs to incorporate basic programming structures in LabVIEW. This section will teach you fundamentals of LabVIEW front panel, block diagram and tool palette, and also explains data flow in the block diagram.

### **Front Panel**

In LabVIEW, you build a user interface, or front panel, with controls and indicators.

**Controls** are knobs, push buttons, dials, and other input devices. **Indicators** are graphs, LEDs, and other displays.

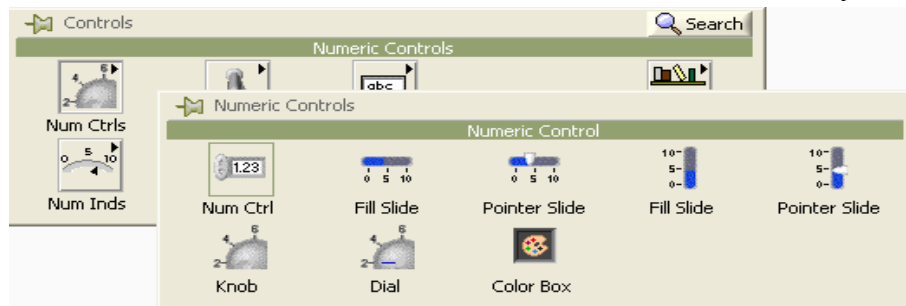
### **Block Diagram**

After you build the user interface, you add code using different functions to control the front panel objects. The block diagram contains this code. In some ways, the block diagram resembles a flowchart.

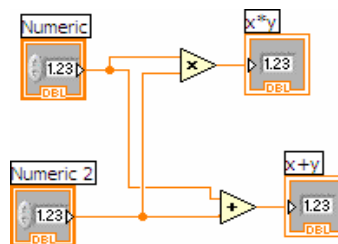
### **Building a VI**

1. Start 'LabVIEW' program.
2. Open a 'blank VI'
3. You will have two screens available. One is the 'front panel' and the second is called 'block diagram'. Front Panel will always have the 'controls' and 'indicators' required by the program to show the input and output parameters. 'Block diagram' will have the detailed graphical representation of the actual program.
4. Go to the 'Front Panel'. The proposed program needs two numbers as inputs for 'addition' and 'multiplication'. Remember all inputs are 'controls' and all outputs are 'indicators'.
5. Right click your mouse keeping the cursor anywhere on the front panel. That opens the 'controls' palette. Then select 'Numerical controls' and a 'Numeric control', which brings a numeric control display onto the front panel. This can be used to input the required number as input 1.

6. These steps may be repeated to obtain a second control input, where you can input the second number. This will be your input2.



7. Now shift over to the 'Block diagram' by pressing CTRL+E characters on your keyboard.
8. You will see the icons representing the two numeric control displays provided on the front panel. They are called control terminals. Right click the mouse (keeping the cursor in the free space on the block diagram). That opens up the 'functions' palette.
9. Select Arith/compare palette. This opens up the Arithmetic/comparison palette. Select 'Numeric' palette. This opens up the 'express numeric' palette, which shows all the arithmetic functions. Select and drag the 'add' and 'multiply' functions on to the block diagram.
10. Right click on out put side of the 'add' and 'multiply' functions and create indicator by selecting Create>>Indicator. This provides the display indicators for showing the results of the respective functions.
11. Get wiring tool and wire both the numeric controls to both the input terminals of the 'add' and 'multiply' functions. The block diagram will then show the following graphical program on the screen.



12. Go to the front panel. Your program is now ready for execution. Input the numbers you want to add and multiply in the respective numeric input boxes. Hit the run button on the tool bar and the output display boxes will show the results of addition and multiplication functions. Save the VI using the 'File' pull down menu as 'example1-add+multiply'.



**Exercise 1:** Build a VI to convert Rupees to Dollars. Take input from the user by using control.

**Exercise 2:** Solve the given equation  $x^2 - 9x + 20 = 0$  by using quadratic formula  $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

**Exercise 3:** Verify De Morgan's laws;

$$\overline{A + B} = \overline{A} \cdot \overline{B}$$

$$\overline{A \cdot B} = \overline{A} + \overline{B}$$

## Lab Session 16



### Objective

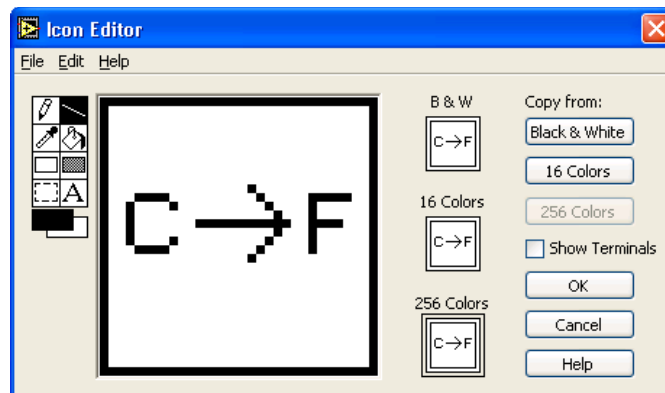
To learn how to create subVI and use it in another VI



### Building subVI


Build a VI to convert temperature from degree centigrade to degree Fahrenheit.

### Icon and Connector Pane

1. Right-click the icon in the upper right corner of the front panel window and select Edit Icon from the shortcut menu. The Icon Editor Dialog box appears.
2.  Double-click the Select tool, shown in pg 65, on the left side of the Icon Editor dialog box to select the default icon.
3. Press the <Delete> key to remove the default icon.
4.  Double-click the Rectangle tool, shown in pg 65, to redraw the border.
5. Create the icon in Figure.



- a)  Double-click the Text tool, shown in pg 66, and change the font to Small Fonts .
- b) Use the Text tool to click the editing area where you will begin typing.
- c) Type C and F. While the text is active, you can move the text by pressing the arrow keys.
- d)  Use the Pencil tool, shown in pg 66, to create the arrow.
- e) Note: To draw horizontal or vertical straight lines, press the <Shift> key while you use the Pencil tool to drag the cursor.
- f) Use the Select tool and the arrow keys to move the text and arrow you created.
- g) Select the B & W icon and click the 256 Colors button in the Copy from section to create a black and white icon, which LabVIEW uses for printing unless you have a color printer.
- h) Select the 16 Colors icon and click the 256 Colors button in the Copy from section.

- i) When you complete the icon, click the OK button to close the Icon Editor dialog box. The icon appears in the upper right corner of the front panel and block diagram.
6.  Right-click the icon on the front panel and select Show Connector from the shortcut menu to define the connector pane terminal pattern. LabVIEW selects a default connector pane pattern based on the number of controls and indicators on the front panel. For example, this front panel has two terminals, deg C and deg F, so LabVIEW selects a connector pane pattern with two terminals.
7. Assign the terminals to the numeric control and numeric indicator.
  - a) Select Help>>Show Context Help to display the Context Help window.
  - b) Click the left terminal in the connector pane. The tool automatically changes to the Wiring tool, and the terminal turns black.
  - c) Click the deg C control. A marquee highlights the control on the front panel.
  - d) Click an open space on the front panel. The marquee disappears, and the terminal changes to the data type color of the control to indicate that you connected the terminal.
  - e) Click the right terminal in the connector pane, and click the deg F indicator.
  - f) Click an open space on the front panel. Both terminals of the connector pane are orange.
  - g) Move the cursor over the connector pane. The Context Help window shows that both terminals are connected to double-precision, floating-point values.
8. Save and close the VI. You will use this VI later in the course.

## Using SubVIs

After you build a VI and create its icon and connector pane, you can use the VI as a subVI. To place a subVI on the block diagram, select Functions>>All Functions>>Select a VI. Navigate to the VI you want to use as a subVI and double-click to place it on the block diagram. You also can place an open VI on the block diagram of another open VI. Use the Positioning tool to click the icon in the upper right corner of the front panel or block diagram of the VI you want to use as a subVI and drag the icon to the block diagram of the other VI.

### Exercise 1

Use the VI built to calculate square root and convert it into subVI.

## Lab Session 17

### **Objective**

To learn how loops are designed and work in LabVIEW.

To write a VI for the Multiplication of a random number with 10 and displaying the result continuously, until it is stopped.

### **Introduction:**


Structures are graphical representations of the loops. Loops use structures on the block diagram to repeat code and to execute code conditionally or in a specific order.


Like other nodes, structures have terminals that connect them to other block diagram nodes, execute automatically when input data are available, and supply data to output wires when execution completes.

- **For Loop** Executes a code a set number of times.
- **While Loop** Executes a code until a condition is met.
- **Case structure** executes depending on the input value passed to the structure.
- **Sequence structure** Contains one or more sequential code, which execute in sequential order.

In text base programming Do Loops repeat-until a condition is met. While loop iterates until the condition is true. The condition can be tested at the beginning or the end of the loop. In LabVIEW, the while Loop executes a graphical code while a condition is true or until a condition is met. A LabVIEW while loop always executes at least once.





Look at the Loop and identify the iteration index, conditional terminal and Boolean operator from the figure.

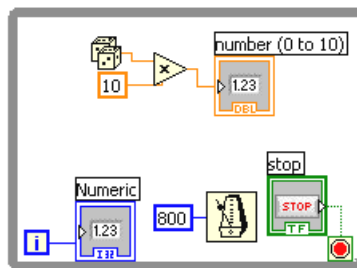
**Stop If True**  When a conditional terminal is in the Stop If True state, the While Loop executes its Block diagram until the conditional terminal receives a TRUE value from the front panel. The VI checks the conditional terminal at the end of each iteration.

**Continue If True**  When a conditional terminal is in the Continue If True state, the While Loop executes its sub-diagram until the Conditional Terminal receives false value.

### **Building the VI:**

1. Open LabVIEW and start a new VI. A blank VI opens the front panel and block diagram simultaneously. If you save one VI, second one will be saved automatically. Put a while loop from the function palette on the block diagram. This comes from Functions >> Structures >> While Loop
2. Right click on the conditional terminal and Select Create>>Control. This provides a True or False Control for the conditional terminal.

3. Get a random number function  by right clicking on blank space from All function » Functions » Numeric » Random number. Same way get multiply function from the numeric palette.
4. Get wiring tool and wire the random number function to one input of 'multiply' function. [By default, you are in automatic tool selection mode, which provides you as soon as you bring the cursor near input/output terminal of any functional operator. In case, you can't get the wiring tool automatically, go to windows pull down menu and click on the 'show tools palette'. Then click on 'connect wire' tool on the palette.] Right click on 2nd input terminal of the multiply function and select Create>>Constant, and type 10 in the blank box.
5. Right click on the output side of the multiply function and create an indicator by selecting Create>>Indicator. Your block diagram will now show:  
This step will show the output value from the multiply function. Right click on the iteration  and create indicator to see number of iterations so far done. [Follow the same procedure you did for creating the indicator in the last step].
6. Double click on the stop button, this will show you location of the stop button in front panel. Identify the indicator of the number of iterations and Output number indicator and change the title of these indicators, if necessary.
7. Hit the run button on the tool bar and observe the running program. It will be too fast to see the numbers.
8. Stop the program by pressing the button and go to block diagram. Use CTRL-E whenever you want to move from front panel to block diagram and vice versa. Get a metronome  by right clicking Functions »All Function »Time & Dialog » Wait until next ms multiple. Right click on left hand side (at its input) of  and create a constant and type 800 in blue box.
9. Your block diagram should look like the diagram below



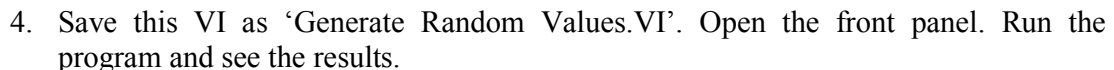
**Exercise 1:** Generate random numbers and multiply it with 10. If the answer is in between 4 and 6, then program should automatically abort. Also introduce a delay of 500ms after each iteration and display the number of iterations that it to ok till the condition is met. As soon as the condition is met, a dialogue box should appear indicating that your required instance has occurred.

NED University of Engineering and Technology- Department of Electronic Engineering

## To learn how for loops are designed and work in LabVIEW

**A For Loop** executes a sub-diagram a set number of times specified by a constant or control at its count terminal (**N**).

1. Open LabVIEW and start a new VI. A blank VI opens the front panel and block diagram simultaneously. If you save one VI, second one will be saved automatically. Put a while loop from the function palette on the block diagram. This comes from Functions >>Structures >>For Loop
2. Get 'all functions' from the 'Functions' palette and connect all wire as given in the figure.



63

## Lab Session 19

### Objective:

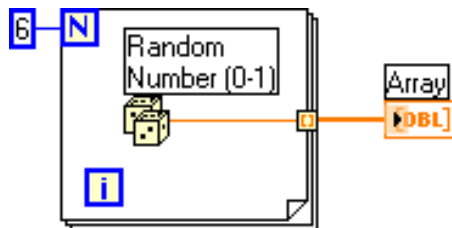
To learn different functionalities of arrays

### Arrays

Arrays group data elements of the same type. An array consists of elements and dimensions. Elements are the data that make up the array. A dimension is the length, height, or depth of an array. You can build arrays of numeric, Boolean, path, string, waveform, and cluster data types. Consider using arrays when you work with a collection of similar data and when you perform repetitive computations. Arrays are ideal for storing data you collect from waveforms or data generated in loops, where each iteration of a loop produces one element of the array.

#### **Auto-Indexing**

If you wire an array to a For Loop or While Loop input tunnel, you can read and process every element in that array by enabling auto-indexing. When you auto-index an array output tunnel, the output array receives a new element from every iteration of the loop. The wire from the output tunnel to the array indicator becomes thicker as it changes to an array at the loop border, and the output tunnel contains square brackets representing an array, as shown in Figure



### Array Functions

- **Array Size** - Returns the number of elements in each dimension of an array.
- **Initialize Array** - Creates an n-dimensional array in which every element is initialized to the value of element.
- **Array Subset** - Returns a portion of an array starting at index and containing length elements.
- **Build Array** - Concatenates multiple arrays or appends elements to an n-dimensional array.
- **Index Array** - Returns the element or sub-array of n-dimension array at index.

### Building a VI

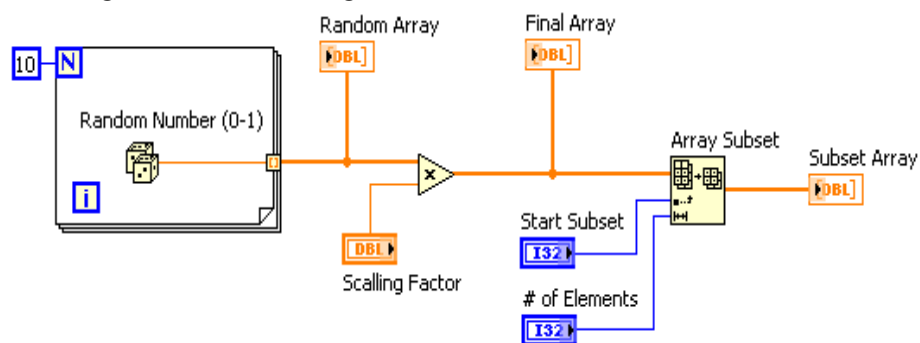
#### **Front Panel**

Build your front panel as shown in the following figure.

Random Array										
0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Scaling Factor										
0.00										
Final Array										
0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Start Subset			# of Elements							
0			0							
Subset Array										
0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

### Block Diagram

Build a block diagram as shown in figure.



**Exercise 1:** Create a two-dimensional array of the order 5x5..

**Exercise 2:** Verify that integral of constant is a ramp and integral of ramp is a square. Display the two results in separate arrays.



## Lab Session 20

### **Objective:**

To display quantities on waveform charts and waveform graphs

### **Introduction**

#### **Waveform Charts**

The waveform chart is a numeric indicator that displays one or more plots. The waveform chart is located on the Controls>>Graph Indicators palette. You can wire a scalar output directly to a waveform chart. The data type in the waveform chart terminal matches the input data type.

Waveform charts can display multiple plots. Bundle multiple plots together using the Bundle function located on the Cluster palette.

#### **Waveform graphs**

VI with graphs usually collects the data in an array and then plot the data to the graph. Waveform graph accept data in the form of array.

### **Building a VI**

#### **Front Panel**

1. Put a waveform chart selecting from Control>>Graph indicators.

#### **Block Diagram**

1. Select a random number generator from numeric palette.
2. Multiply it with 10
3. Put this VI inside WHILE loop.
4. Join the output of the multiplier to the input of waveform chart terminal.
5. Click RUN


**Exercise 1:** Generate a sine wave and plot it on a wave form chart. Also double the amplitude of the generated sine wave and plot the two sine waves on the same waveform graph.


## Lab Session 21

### **Objective:**

To learn the use of structures in LabVIEW for decision-making

### **Introduction:**

**Select**  The Select function, located on the Functions>>Express>>Comparison palette, selects between two values dependent on a Boolean input.

**Case**  A Case structure has two or more subdiagrams, or cases. Only one subdiagram is visible at a time, and the structure executes only one case at a time. An input value determines which subdiagram executes. You must wire an integer, Boolean value, string, or enumerated type value to the selector terminal. You can specify a default case for the Case structure. You must specify a default case to handle out-of-range values or explicitly list every possible input value.

### **Square Root VI**

#### **Front Panel**

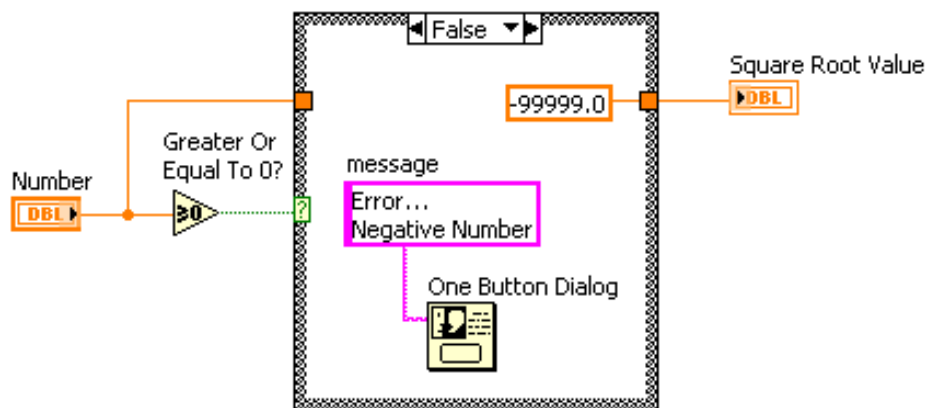
1. Open a blank VI and build the front panel shown in Figure 1.




**Figure 1**


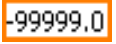

#### **Block Diagram**

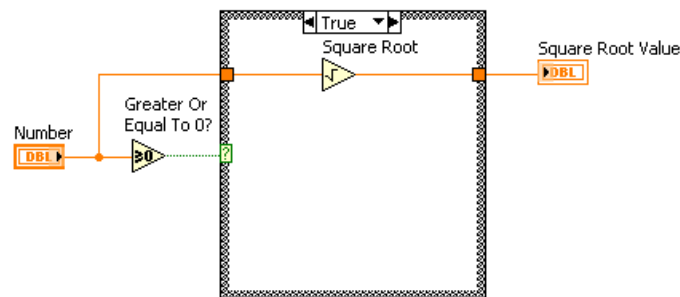
1. Build the block diagram shown in Figure 2



**Figure 2**

- a)  Place the Case structure, located on the Functions>>Execution Control palette, on the block diagram. Click the decrement or increment button to select the False case.

- b)  Place the Greater or Equal to 0? function, located on the Functions>>Arithmetic & Comparison>>Express Comparison palette, on the block diagram. This function returns True if Number is greater than or equal to 0.
  - c)  Right-click the numeric constant and select Properties from the shortcut menu. Select the Format and Precision tab. Set Digits of precision to 1, select Floating point notation, and click the OK button to ensure there is no data conversion between the constant and the numeric indicator outside the Case structure.
  - d)  Place the One Button Dialog function, located on the Functions>>All Functions>>Time & Dialog palette, on the block diagram. This function displays a dialog box that contains the message “Error...Negative Number”.
  - e) Right-click the message terminal of the One Button Dialog function, select Create>>Constant from the shortcut menu, type Error...Negative Number in the constant, and click the Enter button on the toolbar or click outside the control.
  - f) Complete the diagram as shown in Figure 2.
2. Select the True case of the Case structure. Place the Square Root function, located on the Functions>>Arithmetic & Comparison>>Express Numeric palette, on the block diagram. This function returns the square root of Number. Wire the function as shown in Figure 3.



**Figure 3**

**Exercise 1:** Build a VI to convert temperature from °C to °F and °F to °C depending upon the user’s choice and display the result using a single indicator for both conversions. Develop a mechanism to distinguish between the results by their respective units.

**Exercise 2:** Build a VI to perform addition, subtraction, multiplication and division on two numbers taken as input from the users. Only one of the functions can be performed at a time according to user’s wish.

## Lab Session 22

### **Objective:**

To learn how to build clusters and their various functions.

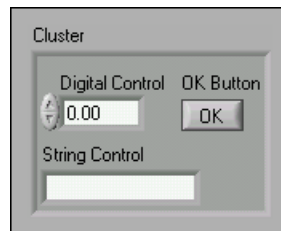
### **Clusters**

Clusters group data elements of mixed types, such as a bundle of wires, as in a telephone cable, where each wire in the cable represents a different element of the cluster.

Bundling several data elements into clusters eliminates wire clutter on the block diagram and reduces the number of connector pane terminals that subVIs need.

#### **Creating Cluster Controls and Indicators**

To create a cluster control or indicator, select a cluster on the Controls>>All Controls>>Array & Cluster palette, place it on the front panel, and drag controls or indicators into the cluster shell.



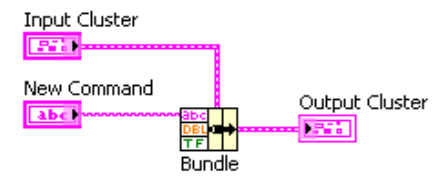
#### **Cluster Order**

Cluster elements have a logical order unrelated to their position in the shell. The first object you place in the cluster is element 0; the second is element 1, and so on. If you delete an element, the order adjusts automatically.

### **Cluster Functions**

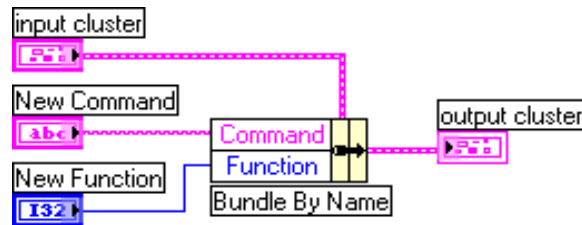
#### **Assembling Clusters**

Use the Bundle function to assemble a cluster from individual elements or to change the values of individual elements in an existing cluster without having to specify new values for all elements.



#### **Replacing or Accessing Cluster Elements**

Use the Bundle by Name function to replace or access labeled elements of an existing cluster. Bundle by Name works similarly to the Bundle function, but instead of referencing cluster elements by their cluster order, it references them by their owned labels.



## Disassembling Clusters

Use the Unbundle function to split a cluster into its individual elements.

Applicant Cluster

Name

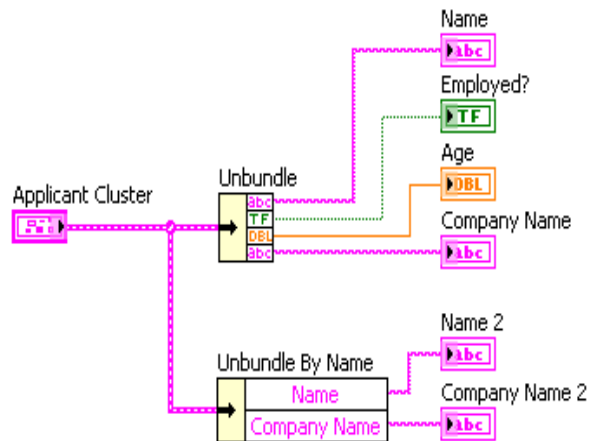
Age
 0

Employed?

Yes
☐

No
☐

Company Name



## Exercise 1

Make a cluster control containing one numeric control, two toggle buttons and one slider control. Increment numerical control by one and invert the logic of second toggle button and display the modified cluster. Also display the values of slider and first toggle button by in a separate cluster indicator.

## Lab Session 23

### **Objective:**

To learn the use of various functions to edit and manipulate strings  
To learn the use of basic file I/O functions

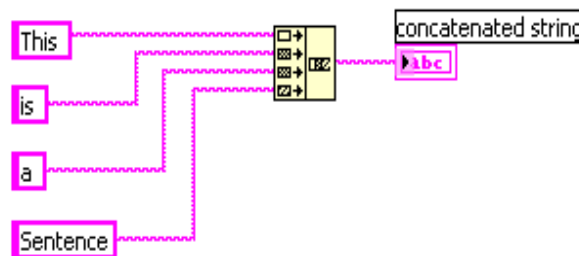
### **Introduction**

#### **Strings**

A string is a sequence of displayable or non-displayable ASCII characters. Strings provide a platform independent format for information and data.

#### **String Functions**

- **String Length** - Returns in length the number of characters (bytes) string, including space characters.
- **Concatenate Strings** - Concatenates input strings and 1D array of strings into a single output string.



- **String Subset** - Returns the substring of the input string beginning at offset and containing length number of characters.
- **Match Pattern** - Searches for regular expression in string beginning at offset, and if it finds a match, splits string into three substrings.
- **Format into string-**



**Exercise 1:** Replace the string “brown” with “yellow” in the sentence “The quick brown fox jumps over the lazy dog”.

**Exercise 2:** Sort the characters in the string “The quick brown fox jumps over the lazy dog” alphabetically

## **File I/O VIs and Functions**

File I/O operations pass data to and from files. Use the File I/O VIs and functions located on the Functions>>All Functions>>File I/O palette to handle all aspects of file I/O, including the following:

- Opening and closing data files
- Reading data from and writing data to files
- Reading from and writing to spreadsheet-formatted files
- Moving and renaming files and directories
- Changing file characteristics

### **Basics of File I/O**

A typical file I/O operation involves the following process:

1. Create or open a file. Indicate where an existing file resides or where you want to create a new file by specifying a path or responding to a dialog box to direct LabVIEW to the file location. After the file opens, a refnum represents the file. A reference number, or refnum, is a unique identifier for an object, such as a file, device, or network connection.
2. Read from or write to the file.
3. Close the file.

**Exercise 4:** Verify that integral of constant is a ramp and integral of ramp is a square as done in lab 5. Display the two results in separate arrays. Write results to a spreadsheet and plot them on graphs.

# Bibliography:

- INTRODUCTION TO MATLAB FOR ENGINEERING STUDENTS  
By David Houcque (Northwestern University)
- <http://www.kpsec.freeuk.com/breadb.htm>
- [www.mathworks.com](http://www.mathworks.com)
- [www.rapidonline.com](http://www.rapidonline.com)
- <http://en.wikipedia.org/>
- <http://www.kpsec.freeuk.com/solder.htm>
- Lab View : LabVIEW Graphical Programming Course  
Collection edited by:National Instruments