

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY
CAMPUS MONTERREY



DEPARTAMENTO DE COMPUTACIÓN

TE2024 MICROCONTROLLERS

STUDENTS' MANUAL

Elaborated by:

Jonathan Félix Gaxiola

Juan Hinojosa Olivares

Israel Alanís

Fernando Elizondo Uribe

Translated and updated by:

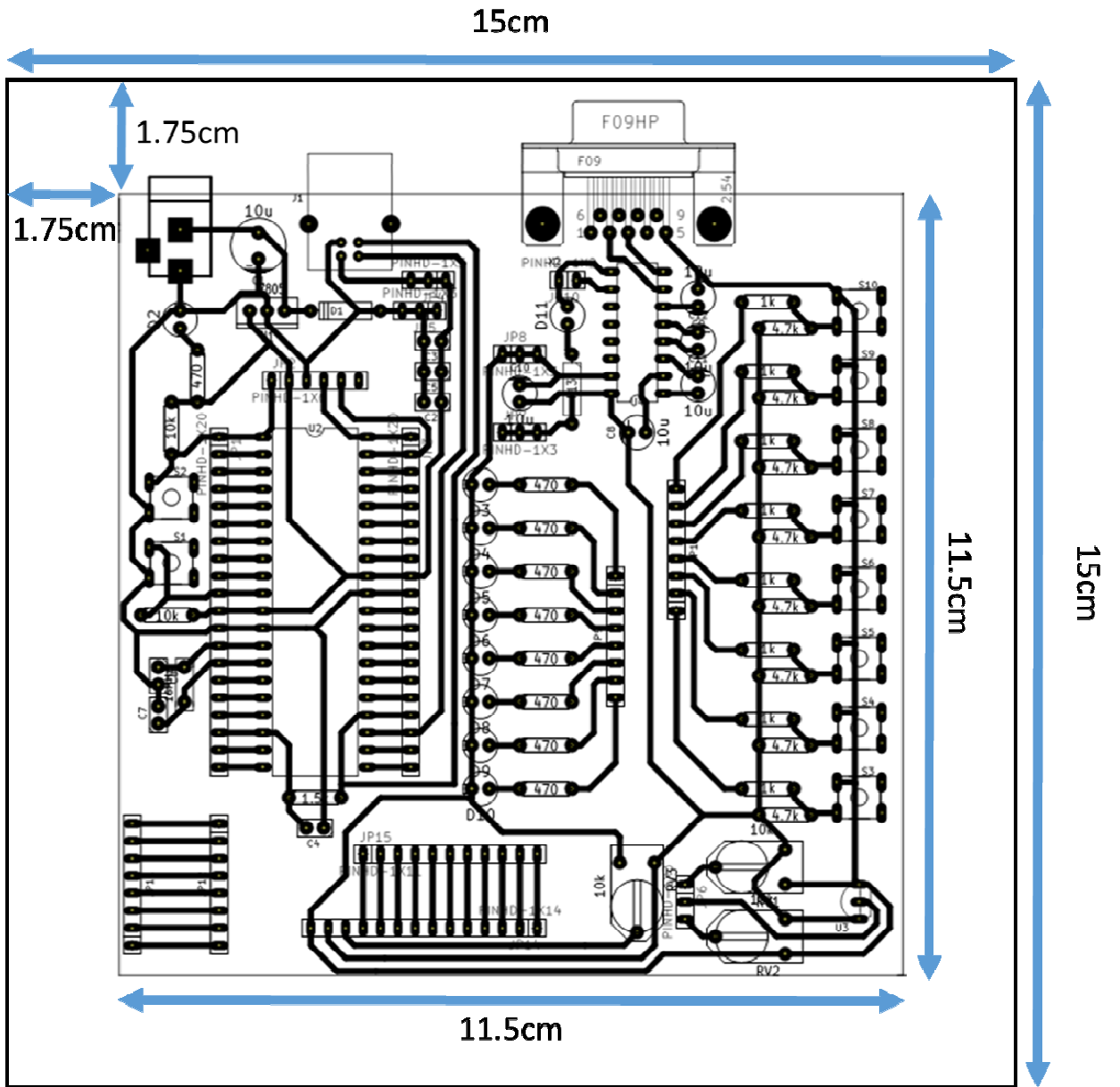
Sebastián Estrada

Luis Enrique Mota

January 2018

Index

Instructor's Information	4
Reglamento de Laboratorio	5
Report Format	9
Introduction to Microcontrollers	10
Practice 1 - Developing the PCB (Schematic)	11
Practice 2 - Development the PCB (layout)	21
Practice 3 - Development of PCB (fabrication)	29
Practice 4 - Introduction to using IDE MPLAB X	34
Practice 5 - Input/Output Ports	41
Practice 6 - Matrix Keyboard	44
Practice 7 - LCD (Liquid Crystal Display)	47
Practice 8 - Timers	49
Practice 9 - Interruptions	52
Practice 10 - Module Capture/Compare/PWM/ADC	54
Practice 11 - Serial Port	57
Annexes	60
Annex 1 (Schematic):	60
Annex 2 (Layout):	61



Instructor's Information

Name: Procopio Villarreal Garza

E-mail: procopio@itesm.mx

pvillarr@expertis.com.mx

Consulting hours: Send an e-mail to appoint an hour.

**Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Monterrey**

Departamento de Computación

Reglamento de Laboratorio

1. Código de conducta

- 1.1. Está prohibido el uso de palabras ofensivas y/u obscenas dentro del laboratorio. Cualquier falta de respeto hacia sus compañeros o hacia el instructor se les penalizará con 50 puntos menos en la práctica y/o ser expulsados del laboratorio.
- 1.2 Es obligación del alumno hacer uso adecuado y cauteloso del equipo electrónico, componentes, herramientas, y en general del mobiliario con el que se trabaja dentro del laboratorio.
- 1.3 Se prohíbe estrictamente introducir alimentos, bebidas, mascotas y demás objetos que puedan llegar a provocar algún accidente o daño dentro del laboratorio.
- 1.4 Por seguridad se debe utilizar zapato cerrado dentro del laboratorio (prohibido el uso de chanclas y zapatos abiertos).

2. Asistencia

- 2.1 El alumno deberá presentarse puntualmente al laboratorio correspondiente el día y a la hora señalados por su horario de clase oficial.
- 2.2 Se concederá una tolerancia de 5 minutos después de la hora señalada en el horario de clases oficial para permitir el acceso al laboratorio, de lo contrario se les penalizará con 15 puntos menos en la práctica de esa sesión.
- 2.3 En caso de falta justificada, para poder ser considerada como tal, el alumno deberá entregar al instructor un justificante por escrito. Así mismo es indispensable que el alumno reponga la sesión del laboratorio en **esa semana** de lo contrario **perderá los puntos de la práctica**.
- 2.4 Las explicaciones de la práctica de esa sesión y de la sesión siguiente se darán los primeros minutos de que comience el laboratorio. Si el alumno no llegó a

tiempo será atendido una vez que las dudas, de los compañeros que llegaron a tiempo, han sido resueltas.

3. Equipos de trabajo

- 3.1 Para facilitar el aprendizaje se recomienda trabajar en equipos de 2 integrantes, mismos que se deberán mantener durante todo el ciclo escolar. Una vez hecho el equipo no se permiten cambios.
- 3.2 En caso de baja académica o cambio de grupo de un integrante de equipo, el integrante restante podría trabajar solo o integrarse a otro equipo según lo considere conveniente el instructor de Laboratorio.
- 3.3 En caso que el Laboratorio lo requiera y a solicitud del instructor, en un equipo de trabajo deberá haber al menos una computadora portátil con suficiente carga de batería y/o con su respectivo adaptador de corriente.

4. Actividades dentro del laboratorio

- 4.1 Es requisito indispensable para que se permita el acceso al laboratorio, llevar los componentes solicitados por el instructor para realizar la práctica correspondiente.
- 4.2 En caso que el Laboratorio lo requiera, se deberá tener instalado y funcional el software requerido por la práctica en la computadora portátil con la que se trabajará dentro del laboratorio.
- 4.3 Todos los integrantes de un equipo de trabajo son responsables por el buen uso de los materiales y equipo electrónico con los que trabajan. Si por el uso incorrecto o falta de atención por parte del usuario, algún equipo electrónico (osciloscopio, fuente de poder, generadores de funciones, multímetro, etc.), herramientas, accesorios o componentes resultan dañado, se hará el cargo correspondiente por reposición o reparación, según sea el caso, a los integrantes del equipo responsables del cuidado de dicho material.
- 4.4 En caso de detectar alguna anomalía en los materiales y/o equipo electrónico con el que se trabaja, se deberá reportar inmediatamente al instructor.
- 4.5 Está prohibido cambiar la ubicación del equipo electrónico que no sea solicitado al almacén.
- 4.6 Para solicitar material de trabajo en el almacén, un integrante del equipo de trabajo deberá llenar la forma proporcionada por el almacenista y dejar su credencial de estudiante, misma que se devolverá al regresar el equipo prestado.

- 4.7 Es responsabilidad de cada alumno devolver el material y/o equipo electrónico que haya solicitado personalmente al almacén. El almacenista no devolverá credenciales a alumnos que no sean quienes hayan solicitado el equipo prestado.
- 4.8 Es bien importante que llegar preparados para la sesión del laboratorio. Es indispensable haber leído su práctica ANTES de ingresar al laboratorio. El instructor tiene la responsabilidad de explicar la práctica y orientar a los alumnos en la solución de los problemas asignados. El instructor no proporcionará la solución del problema asignado.
- 4.9 Queda estrictamente prohibido prestarse los circuitos contruidos para la práctica. Así como prestarse los códigos requeridos en las mismas. Si el instructor tiene sospecha que alguien de otro semestre, de otro salón o algún compañero les haya pasado cualquiera de las dos cosas mencionadas se les acreditará un 0 en la práctica y un DA en su calificación.
- 4.10 Para poder obtener la calificación máxima en la práctica el alumno deberá entregar lo pedido en la misma y además el alumno deberá de responder una serie de preguntas individuales acerca de la práctica desarrollada en esa sesión.
- 4.11 Está prohibido el uso del Messenger, Facebook, Twitter o cualquier página de internet que no tenga nada que ver con el curso. Si algún alumno hace caso omiso a esta regla se le penalizara con 50 puntos menos en la sesión.
- 4.12 La práctica se deberá mostrar al instructor en el momento que el calendario lo dicte. Si la sesión del laboratorio es de 2 horas, se les recomienda que se tomen su tiempo para mostrar la práctica al instructor. Ya que el instructor tiene toda la libertad de irse exactamente a las 2 horas después de que comience el laboratorio. Si el alumno no llegara a terminar las prácticas, cuenta con un total hasta TRES días para entregarlas sin penalización. Cada día tarde que pase sin entregar la práctica son 25 puntos menos de la calificación.
- 4.13 Los pre-reportes se entregan al comenzar la sesión de laboratorio en el momento que el instructor tome lista. No se aceptan reportes fuera de tiempo.
- 4.14 El reporte se entregará 1 semana después de haber concluido la práctica en el laboratorio. Se barajarán 10 puntos por cada semana tarde que se entregue.
- 4.15 La calificación de cada práctica se dará a conocer 1 o 2 semanas después de haber concluido la práctica. Por lo que cualquier aclaración se tendrá que hacer en el momento que se les dé a conocer su nota (ese mismo día).

4.16 Es responsabilidad del alumno traer a clase su material a usar como componentes, laptop, cuaderno, libro etc. Es bien importante que se pongan de acuerdo los dos integrantes del equipo sobre quien trae las cosas, ya que si no traen el material no perderán puntos en la práctica por no terminar el trabajo asignado.

4.17 Es responsabilidad del alumno anotar el nombre, matricula, teléfono de casa, celular y mail del equipo de trabajo para cualquier futuro problema.

Nota: Este reglamento tiene los puntos generales que deben observarse en los Laboratorios del Departamento de Computación del Campus Monterrey, sin embargo, algunos puntos podrían ampliarse de acuerdo al criterio del profesor/instructor.

Report Format

Report of Practice #X

Introduction {10% | Sum of all point}

- [5%] **Topic's summary.** What topics (course content) does the lab cover/address? A paragraph with at least three sentences.
- [5%] **Objective/purpose.** What was the objective and purpose of this lab? A paragraph with at least three sentences.

Body {60% | Average of all the exercises}

Exercise #:

- [10%] **Description.** Brief description of the goal and purpose
- [30%] **Results.** Pictures of the board working. Screen-prints (printouts) from the computer windows (example MPLAB screen). Don't forget to include the explanation of each picture. (one sentence at least)
- [20%] **Problems/Solutions.** Problems found and your solutions to those problems.
- [40%] **List of code/commands.** it could be in 2 columns with a small font. **THE CODE MUST BE COMMENTED.**

Conclusion {30% | Sum of all point} (for each member of the team)

- [25%] **Reflection** about your leaning process
- [5%] **Suggestions** about the practice.

Introduction to Microcontrollers

Microcontrollers are all around the world. Each day, Microcontrollers, are more present in the many aspects of our lives: in our work, inside our houses, and in more. We can find them controlling small devices like cellphones, microwaves, washing machines, and televisions.

A microcontroller is one device or chip that is used to govern one or more processes. For example, the controller that regulates the room temperature of an air conditioner; it has a sensor that continuously measures the internal temperature and, when the preset limits are exceeded, it generates the necessary signals to adjust the temperature.

Currently a microcontroller has the following elements:

- Processor o CPU (Central Unit Processor)
- RAM memory for data.
- ROM/PROM/EPROM memory for the program
- Peripheral control modules (temporizers, serial ports, parallel ports, Digital-to-Analog Converter, Analog-to-Digital Converter, and many more).
- Clock pulse generator to synchronize the operation of the entire system

The main objective of this manual is to provide students the foundation to fully understand the operation of the PIC18F45K50 microcontroller. This will be achieved through 11 practices that will guide the reader to create their own electronic card or PCB (Printed Circuit Board) and to be able to program it; in order to, execute different functions.

The advantages of the PIC microcontroller to others on the market, which is why it will be used throughout this manual, are as follows:

- Easy to operate.
- There is enough documentation to work with it and it's easy to obtain it.
- The price is comparatively lower than its competitors.
- It has a high operating speed.
- Development tools are cheap and easy to use.
- There are a variety of hardware that can record, erase and check the behavior of PIC.
- Once you learn to handle a PIC, it will be easier to handle any other models of microcontrollers.

Practice 1 – Developing the PCB (Schematic)

Objective

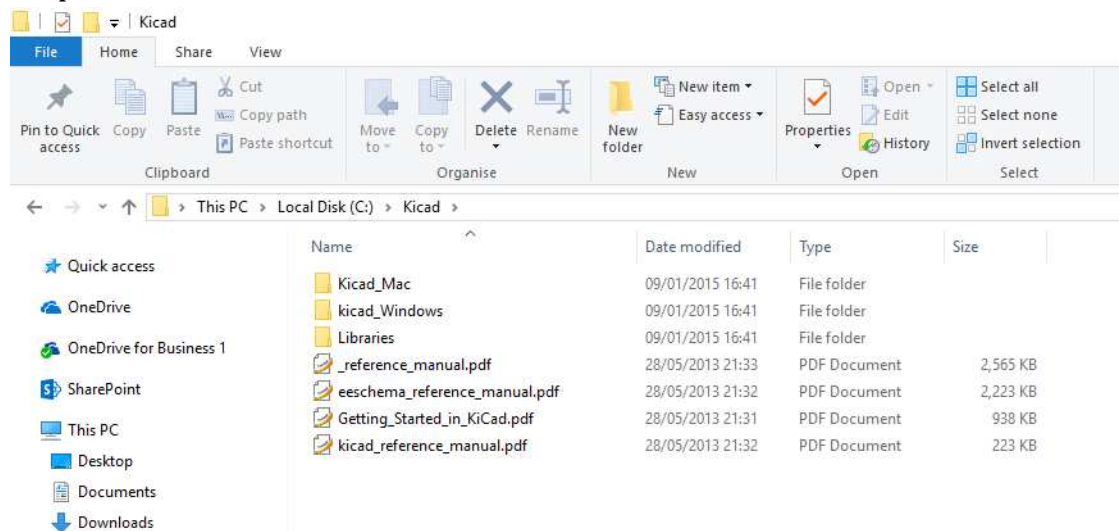
The student will make the schematic design of a minimum card system, which he/she will work with throughout the semester. The minimum system contains: microcontroller, switches, LEDs, +5V source power, serial port, keyboard connections and LCD.

The students will use the design software 'Kicad' for the entire design (schematic and layout).

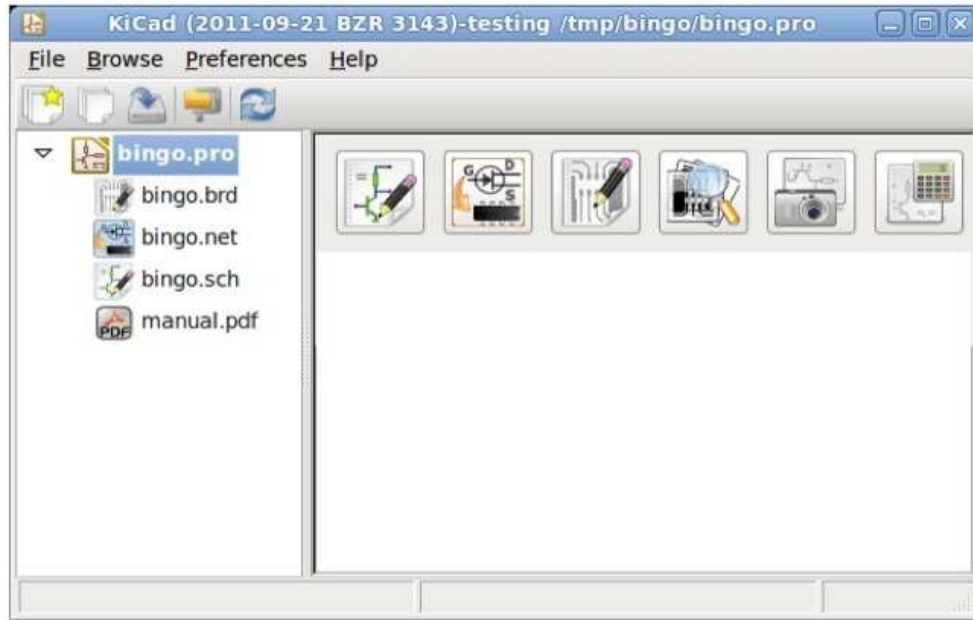
Activities




NOTE: PLEASE REVIEW ALL THE INSTRUCTION ONCE BEFORE STARTING TO WORK ON IT.

1. Download and install the Open source (free) software 'Kicad' from its official web page: <http://www.kicad-pcb.org/>. You can also find a more stable version on the Blackboard Platform.
2. Copy the libraries, which came inside the folder with the manual, to your computer. You may save them in the 'Kicad' folder. Should look just like the picture below:




3. Double click on kicad.exe. Now the main page will open 'kicad Project manager'. From here, you will be capable of accessing 5 different tools: EESchema, Cvp pcb, PCBnew, GerbView and Bitmap2Component.



4. Create a kicad folder at C:/kicad
 Copy the files from ...\\TE-2024\\Softwares\\Kicad\\kicad_Windows into C:/kicad
 Copy the folder "Libraries" found at ...\\TE-2024\\Softwares\\Kicad\\Libraries into the C:/kicad folder
 Create a folder where you want to make the project files into the C:/kicad folder. **NOTE: Make sure that the PATH and NAME of your project does not include special characters like "spaces", accentuations, letter "ñ", etc.**
5. Create a new project. File -> new->New from template. Open the project folder, all files in your project will be saved there. The project will be saved with the '.pro' extension. A Template selection screen will appear; Click OK and ignore the error screen that may appear.
6. Click on the 'update'/'refresh' button  to display the new file extension '.pro' in the 'project manager'.
7. Begin to create the schematic. Let us open the EESchema editor "". It is the first button on the left of the 'project manager'. If an error message pops and says: 'project file was not found', ignore it and click OK.
8. First we need to save all the project of the schematic: File -> Save Whole Schematic Project. Click the icon 'Page Settings' "" located in the top of the toolbar. Select the size 'A3', and enter the title of your schematic. You can see there is more information that can be provided if necessary. Once

completed the necessary information, click OK. This information will appear in the lower right corner of the design page.

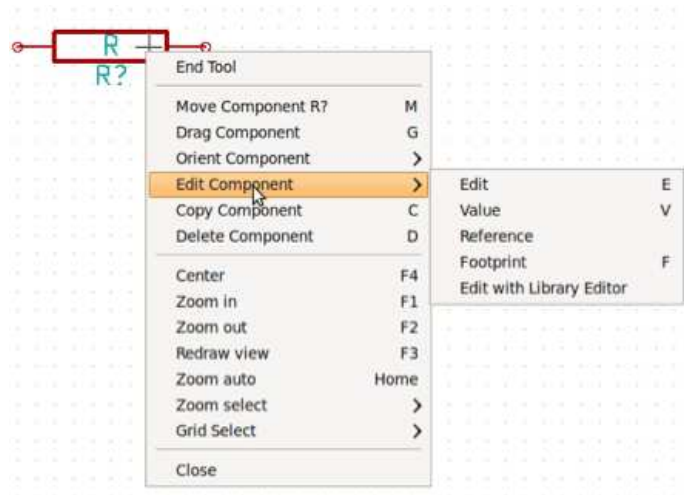
9. Add all the libraries to the 'Kicad': Preferences -> Library. Click on the button 'Add' and search for the folder where you save them beforehand. Select all the libraries and click 'Open'.
10. Now let's add the first component of the design. Click the icon 'Add

components'  " in the toolbar on the right. The same functionality is obtained by pressing the shortcut 'Key A'.

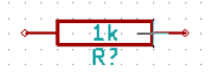
NOTE: You can see a list of all available shortcut by pressing 'Key ?'.

The component selection window appears. Click on the 'List All' button. Now the library selection window will appear, here you have a list of all libraries available.




11. Select with a double click the library 'device'. A window to select components will appear. Here you have a list of all available components in the library 'device', which is the one that contains the most commonly used components.
12. Scroll down and click the resistance R. This action will close the component selection window and take you back to the schematic sheet.
13. Place the component in the schematic by clicking wherever you want it to be. You can zoom in or out the 'scroll' of your mouse.
14. With the mouse over the 'R' component, press the 'Key R'; you will see how the component rotates 90 degrees, you do not need to click on the component to rotate it.
15. Click on the middle of the component and select: Edit Component -> Value. You can get the same result on the component by positioning the cursor above it and pressing the 'Key V'; also with the 'Key E' you will have a window with the more general properties. When you click the right mouse button on the component, all the possible 'shortcut keys' actions will appear.






16. The value of the component appears. Replace the current value 'R' for '1k' and click OK. Do not modify the reference field (R?), this option is automatically modified later. The value inside the resistor should now be "1k".




17. If you made a mistake and want to delete a component, you must right-click the mouse on the component and select 'Delete Component'. You can delete it by pressing the 'Key Del' by hovering the mouse over it. NOTE: You can rename any 'shortcut key' by going to: Settings -> Hotkeys. Remember to save the new keys: Preferences -> Save preferences.
18. You can duplicate an existing component in your schematic by positioning the cursor on the component and pressing 'Key C'.
19. Press right click on the resistance. Select 'Drag Component'. Reposition the component and left click to drop it. The same function can be done by clicking a "Key G". You can use 'key R' to rotate the component. (NOTE: The option right click -> Move component (equivalent to 'Key M') is a viable option to move any component, but is used for components that have not been connected.)


20. To add the symbol +5V  and GND (reference) , select the button 'Place a power port'  in the toolbar located in at the right. You may also use the shortcut 'Key A' and select the 'power' library.


21. This is also the same procedure to add +5VA (+5V alternative)  y GND (alternative reference) .


22. To wire the components, you must click on the icon 'Place a wire' "  " located in the toolbar on the right. NOTE: Be careful not to select the 'Place to bus' below this button. The use of that button will be explained later.

23. Sometimes it is good to give names to the wires using 'labels' which is an alternative form of wiring connections between components. To do this, you must click on the icon 'Place net name' "  " in the toolbar on the right.

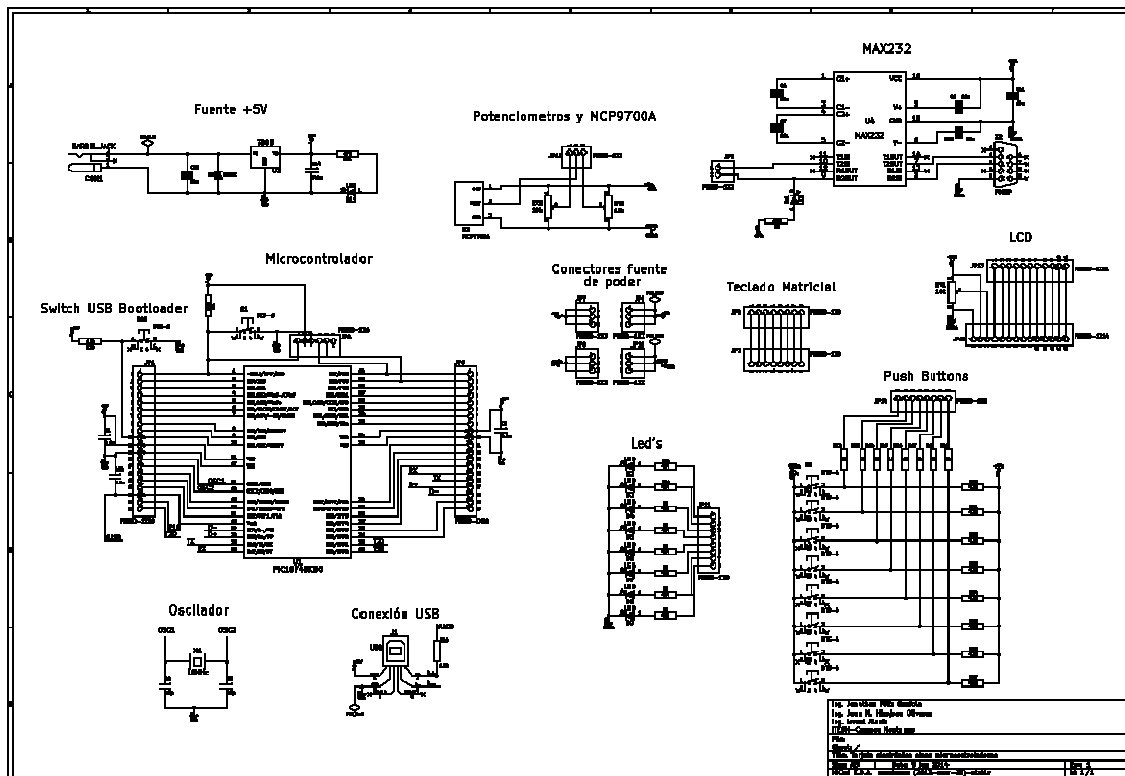
You can also press 'Key L'. Sometimes 'labels' are used only for purposes of information on one line.

24. The non-connected terminals must be identified to avoid errors or warnings in the schematic. To mark a connections that will be free, you must select the icon 'Place no connect flag' “” located in the toolbar on the right and place the 'X' symbol on the free terminal.

25. It is necessary to add 'Power Flags' to signal the 'Kicad' that the power comes from a valid source. Press 'Key A', select 'List All', double click on the 'power' library and search for a 'PWR_FLAG' “”.

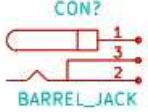

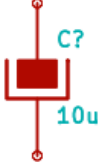


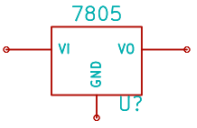
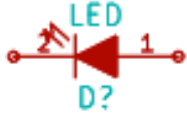

26. Sometimes it is good to write comments or notes in the schematic. To do so, you can use the icon 'Place graphic text'  located in the toolbar at the right.

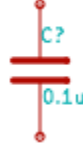

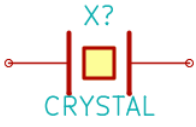
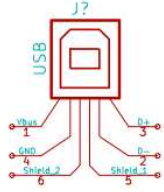


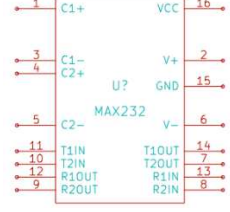
27. Now assemble the schematic design needed to make the PCB of this manual:

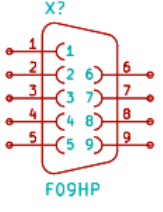
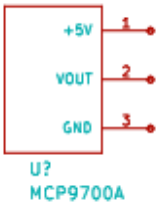


NOTE: A larger schematic image is added as 'Annex 1' on the part of annexes to this manual.


To help you find all the components more quickly, a table indicating in which library is included each component is noted below:

Name	Library	Image
Barrel Jack Connector	Conn -> BARREL_JACK	
PWR_FLAG	power -> PWR_FLAG	
Polarized Capacitor	device -> CP	
Diode	device -> DIODE	
Resistor	device -> R	
+5V rectifier	regul -> 7805	
Led	device -> LED	
Push button	user -> DTS-6	

Non-polarized capacitor	device -> C	
PIC18F45K50	user -> PIC18F45K50	
Crystal	device -> CRYSTAL	
USB connector	conn -> USB	
Pot	device -> POT	
Pinheads	user -> PINHD-??X??	
Max232	user -> MAX232	


Serial Connector	user -> F09HP	
MCP9700A	user -> MCP9700A	

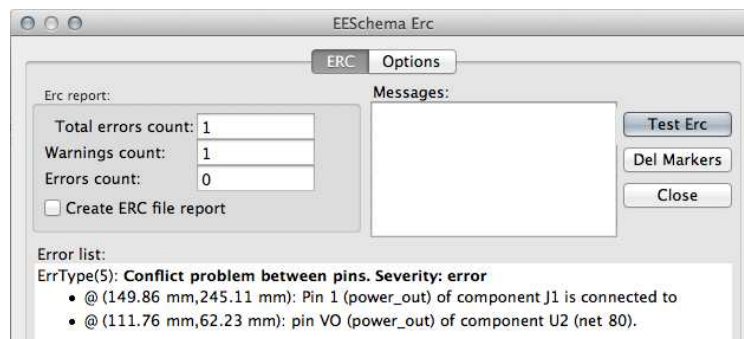
28. All components must have a unique identifier. In fact, many of the added components are named after 'R?' or 'J?'. To assign an ID automatically, you

can clicking on the icon 'annotated schematic' “”.


29. In the window 'Annotate Schematic' select 'Use The entire schematic' and click on the 'Annotation' button. Click OK in the confirmation message that appears on the screen; then click on 'Close'. You will see how all the '?' Have been replaced by numbers. This identifier is unique.

30. Now let's check the schematic for error. Click the icon 'Perform Electric Rules

Check' “”. Click the "Test ERC" button. A report informing you of any error, warning and/or disconnected terminals will appear. You should have 0 errors and 0 warnings. If any of the cases, a green arrow appears in the schematic positioned where the error or warning exists. For further information about the errors, click on 'Write ERC report' and then press 'Test ERC'. In the schematic of this manual, just one error must appear; and do to the nature of this error, you may ignore it. The following error should look like this:



The error occurs because of the outdated nature of the “regul” library.

31. The schematic is finished. Now we can create a "Netlist file" that will be added to the impression of each component. Click the icon 'Netlist generation' " " in the toolbar on the top. Click on 'Netlist' and then 'save'. Save it with the name that appears by default.
32. Now you may close the schematic editor.

Practice 2 – Development the PCB (layout)


Objective



Develop the connection diagrams (layout) from the schematic diagram.

Activities

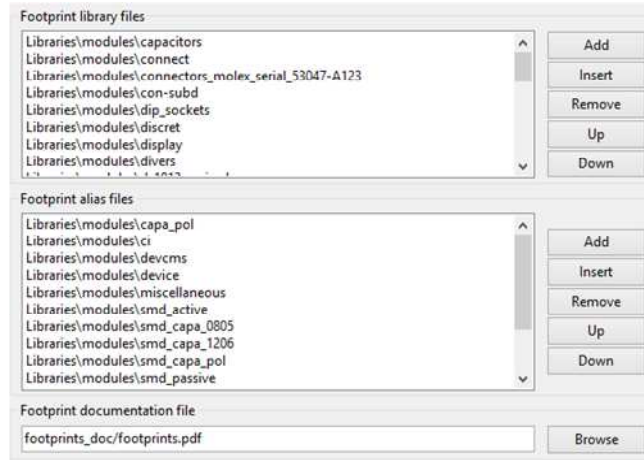
NOTE: PLEASE REVIEW ALL THE INSTRUCTION ONCE BEFORE STARTING TO WORK ON IT.

The next step consist on creating a Netlist file, in which, you will add the footprints to each and every one of our components. To do this, open the schematic design and

click on the icon 'Netlist Generation'  in the toolbar above. Then click on the 'Netlist' option and by doing so it will create and save a .net file in the workspace Folder with the same name as the .sch file. This type of file shows all the used components with their respective connections for each pin. The netlist file is can be opened as a text based file, which you can see and edit, without a problem. }after creating the netlist file, go back to the main window of Kicad and click on 'Run

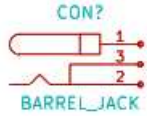
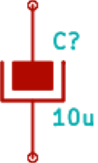

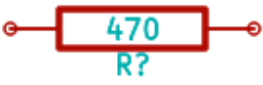
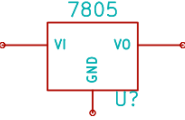
Cvpcb' , by doing so, you will be able to watch the footprints associations in a list format. If there is no list displayed, click on 'Open a net list file'  and open the .net file directly.

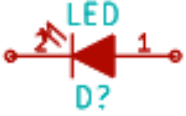

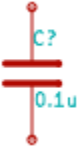

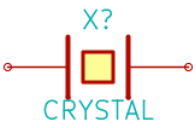
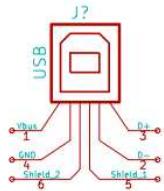

Now it's the time to add the corresponding libraries to the Cvpcb program. Click on Preferences -> Libraries and in this way you will be able to add the modules from the folder named 'modules' (which is inside the Libraries folder). Below you will find the window where the modules linked to the program.

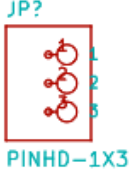
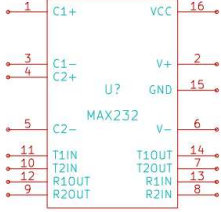
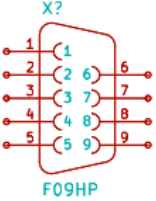
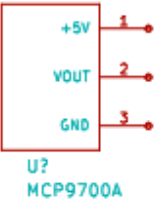


Once that's done, you must associate the list of components with its respective footprint, use the table below for guidance; me sure to activate the option of 'Display

the full footprint list'  from the toolbar of the Cypcb.

Name	Schematic Image	Footprint name
Barrel Jack Connector		BARREL_JACK_MOD
Polarized Capacitor *1		Capacitor_elect
Diode		Diode
Resistor *2		Resistor
+5V rectifier		LM7805

Led		Led
Push button		DTS-6
Non-polarized capacitor		Capacitor
PIC18F45K50		PIC18F45K50
Crystal		Crystal16MHz
USB connector		USB_B_MOD
Pot		RV2X4

Pinheads		Pinhead-??X??
Max232		Maxim232
Serial Connector		Conector_F09HP
MCP9700A		MCP9700A

NOTE 1: For the capacitor of 10uF near the Barrel Jack Connector, use the footprint of 'Capacitor_elect_8x13mm'.



NOTE 2: For the resistor of 470 ohms near the Max232, use the footprint of 'Resistor_large'.

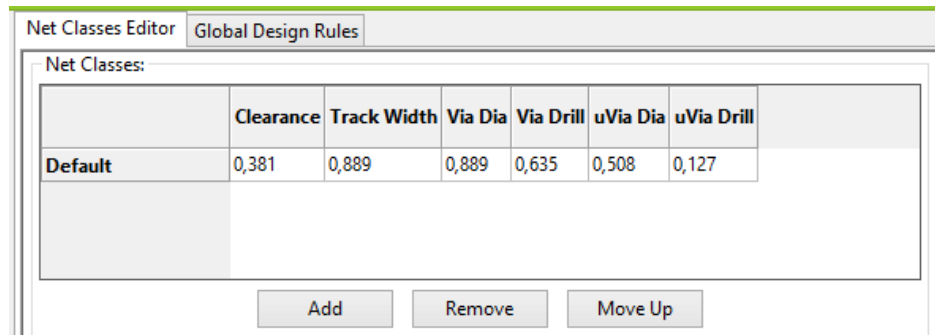
Save your modification with 'File -> Save' or doing click on .

After that, you may close the Cvp pcb and go back to the EEAchema (Schematic Editor). Save youe project by clicking on 'File → Save Whole Schematic Project'. Close the editor of the Schematic and switch back to the 'Project Manager'.

PCB Layout Design

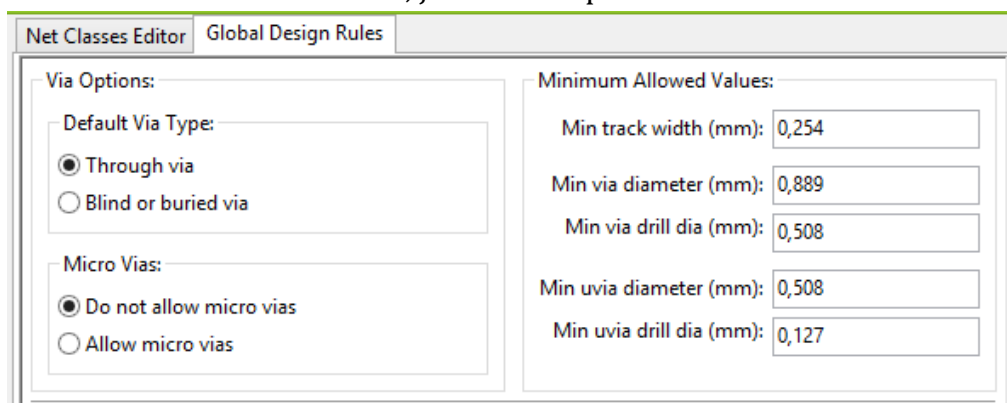
Once you have the design of the circuit on the schematic, it's time to generate the 'layout' of our PCB using the netlist file:


1. On the 'project manager' from the kicad, click on the icon 'PCBNew' . A new window will appear from the 'PCBNew'. Don't mind any error message that may appear and just select 'OK'.
2. On the toolbar menu from the 'PCBNew' click on the 'Page Settings'  and modify the size settings to 'US Letter'.
3. After that, it will be time to focus more on design concepts. You may do so by opening the menu of 'Design Rules → Design Rules'. A window like below must show on your screen:




On it, some final design characteristics will appear, with many millimeter values, and between them, you should find the 'Clearance' specification, which is the smallest spaces that it must exist between you tracks on the PCB. You shall set the value of this cell to 0.254 mm (0.010 inches). Next, we set a value of 0.800 mm (0.0314960 inches) on the cell of 'Track Width'. The other cell values must be set like the picture above.

4. Switch now to the 'Global Design Rules' tab and write, if the instructor said so, the minimum values for each and every one of the parameters on the 'Minimum Allowed Values' box, just like the picture below:





5. Time to import the netlist file. Go to the toolbar and click the icon 'Read Netlist'  and on the field 'Netlist File' it must appear the folder path to

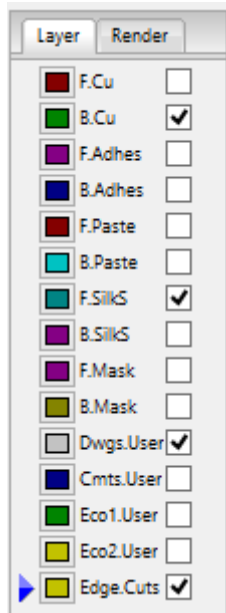
the netlist that you created before. For the final step, click on the 'Read Netlist' button and close the window. All the components will appear on the screen.

6. Now make sure that the toolbar in the upper size of the 'PCBNew' has the the option 'Mode Footprint'  highlighted. Right click on any blank section of the screen and select 'Global Move and Place' -> 'Move All Modules', select 'Yes'.

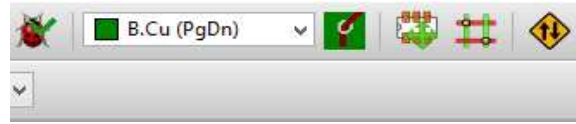
NOTE: Make sure that all the resistors, capacitors, and any other component are present on the 'PCBNew' screen, if there is a missing component, please contact your lab instructions for further instructions.

7. It's your turn to move all the components in the circuit, in order to avoid any track intersection, inside the working area. You can move the components by positioning your mouse pointer above it and pressing the keyboard letter 'G'. All the components are connected to each other by a group of wires called *ratsnest*. Make sure that the icon of 'Hide board ratsnest'  located in the toolbar of the left is highlighted.

8. Now you have to define the limits of the PCB. Select the tool 'Add graphic line or polygon' . First select the type of line to draw on the screen, by using the toolbar on the right side, like the picture below. Select 'Edge.Cuts' and began to trace the original measures of your copper board (15x20 cm). You must also define the cutting area; don't forget to add a 1.75 cm margin around the board.



9. Now you must link all the connections of you PCB. But before that, you must change the drawing option from the right toolbar from 'Edge.Cuts' to 'B.Cu' (B=> Bottom, Cu=> Copper).

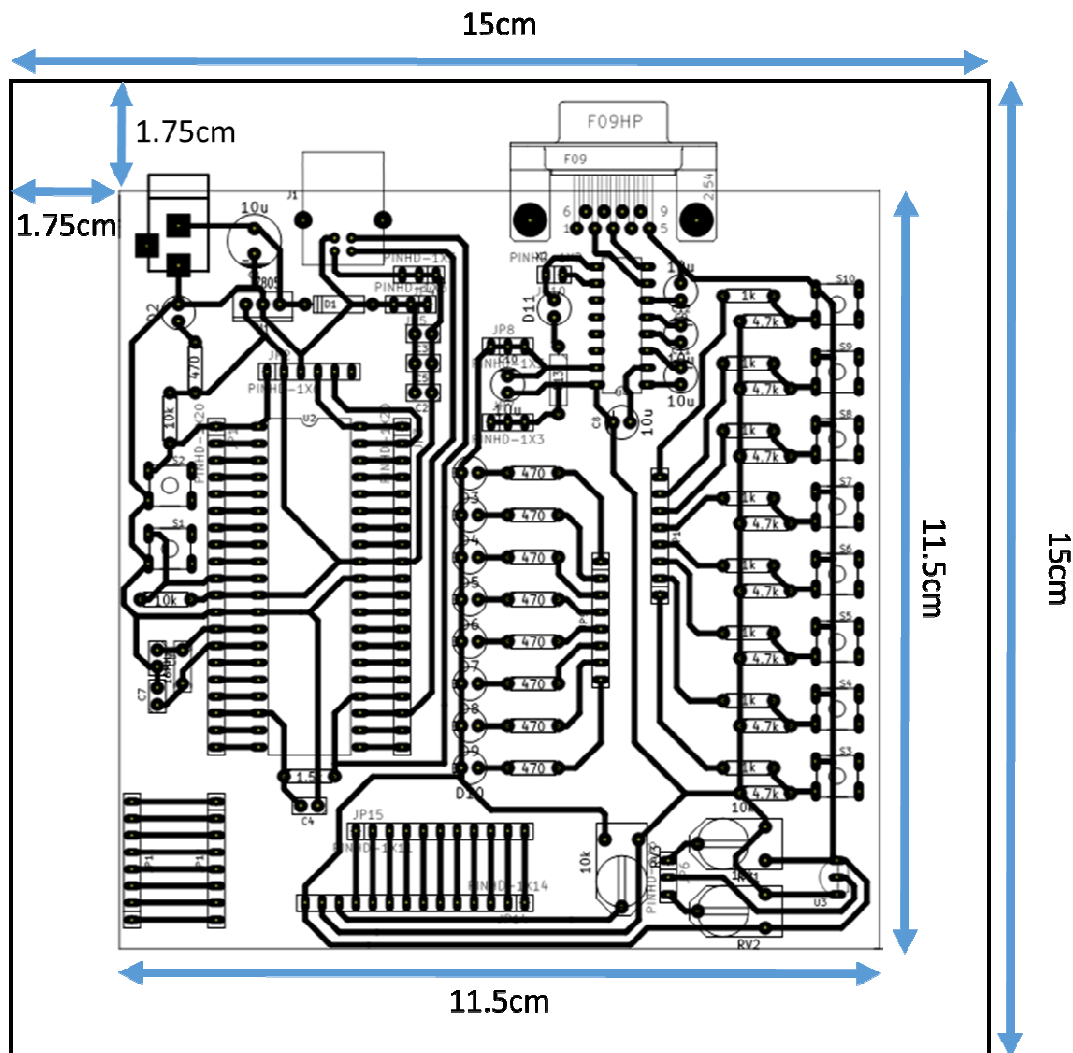


10. Start to connect your tracks to each and every one pin on the schematic by

using the icon 'Add track or vias'  or by using the keyboard letter 'X'.

11. Once you are done, save your file on File → Save or using **Ctrl + S**.

12. Your board must look like the picture below:



NO

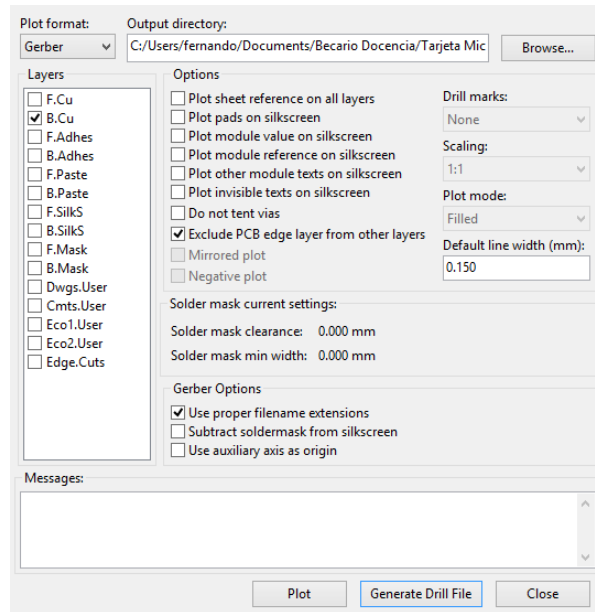
TE: A larger images of the layout is located on the annex section.

13. Contact your instruction for further review of your design.

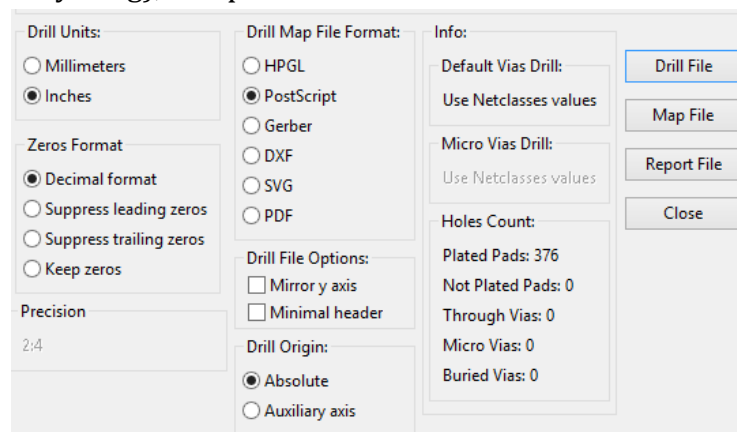
Generate Gerber files

Now that your PCB is completed, you can generate the Gerber Files for each layer of your board.

1. Click on File → Plot. Select 'Gerber' as 'Plot Format', also you must choose a folder to save your generated files. Make sure that the only option selected is the 'B.Cu' from the 'Layers' box.



2. Finally, click on 'Plot' and then 'Generate Drill File'. A new window will appear with some options on default (don't change anything), and press 'Drill File'. Then on 'Drill Map File Format' change to PDF and once again press 'Drill File' (don't change anything), and press 'Drill File'.



Practice 3 – Development of PCB (fabrication)

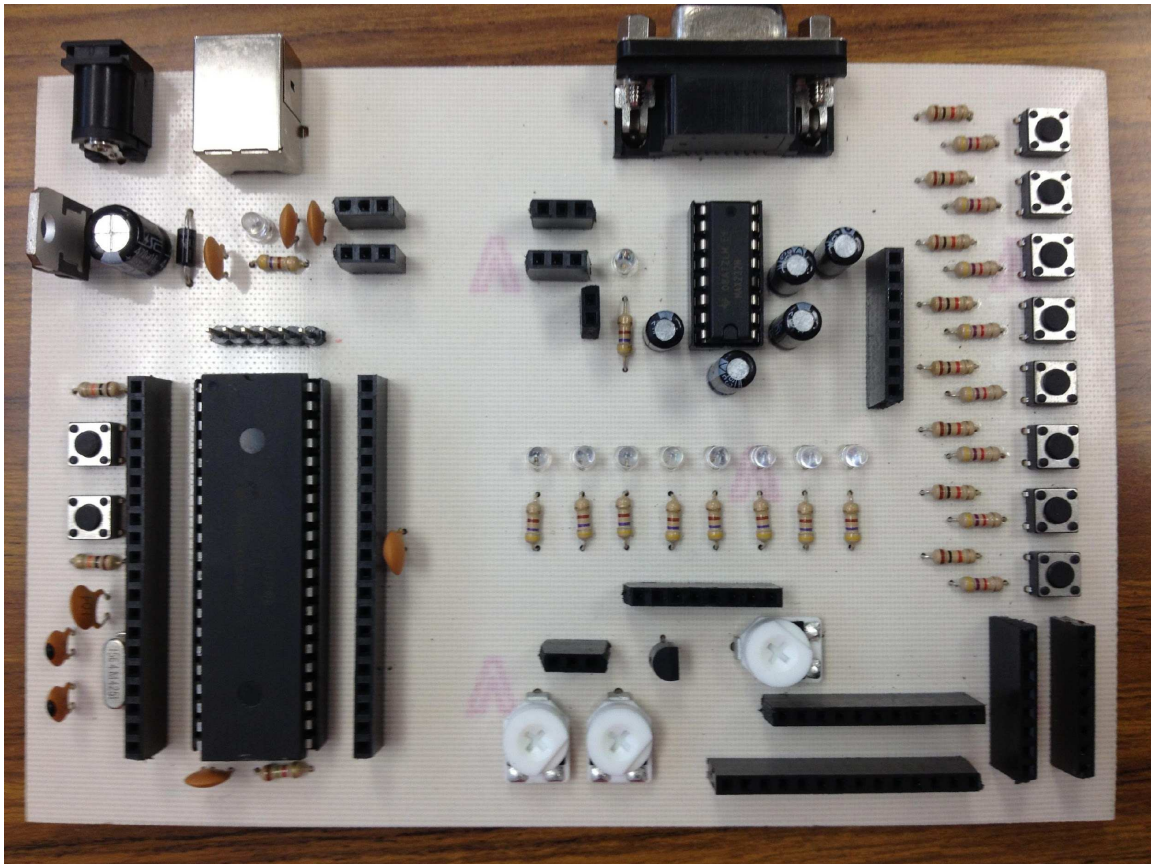
Objective:

Build the minimum system from the designs that the student made on the practices 1 and 2.

Activities

NOTE: From now on, you will work as teams of 2. Teams with at least one ISD students **MUST** realize this activity with option NUMBER 2, no exceptions.

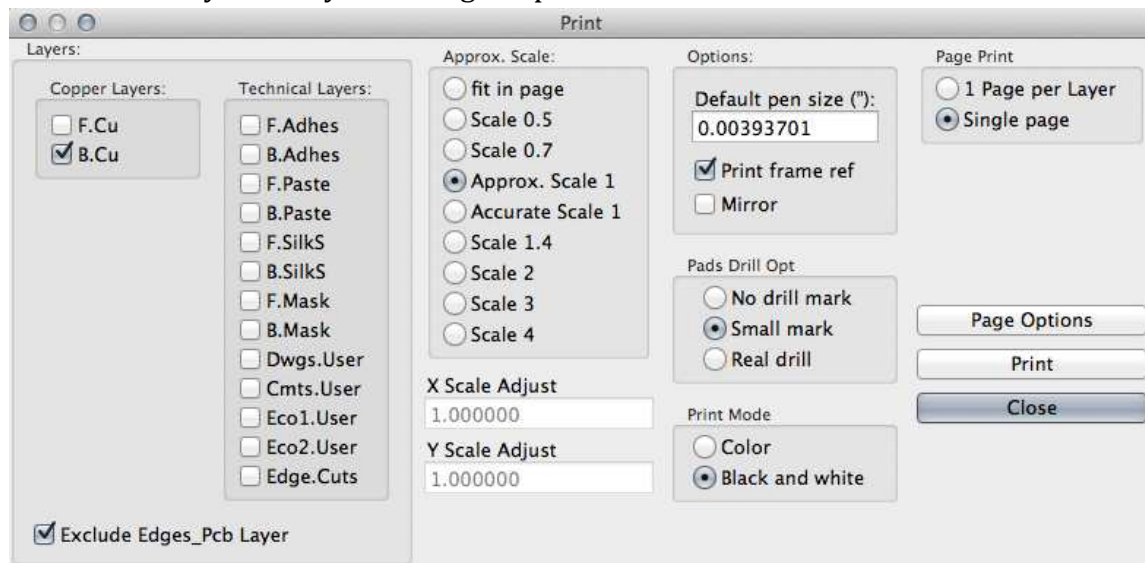
Your team must decide which design to use to build the board inside the laboratory. Just like the picture below:



NOTE: DO NOT USE this picture as an example to drill holes or place components. Each design is unique and there might be some variations. This model is a 20 x 15. Yours must be a 15 x 15 model.

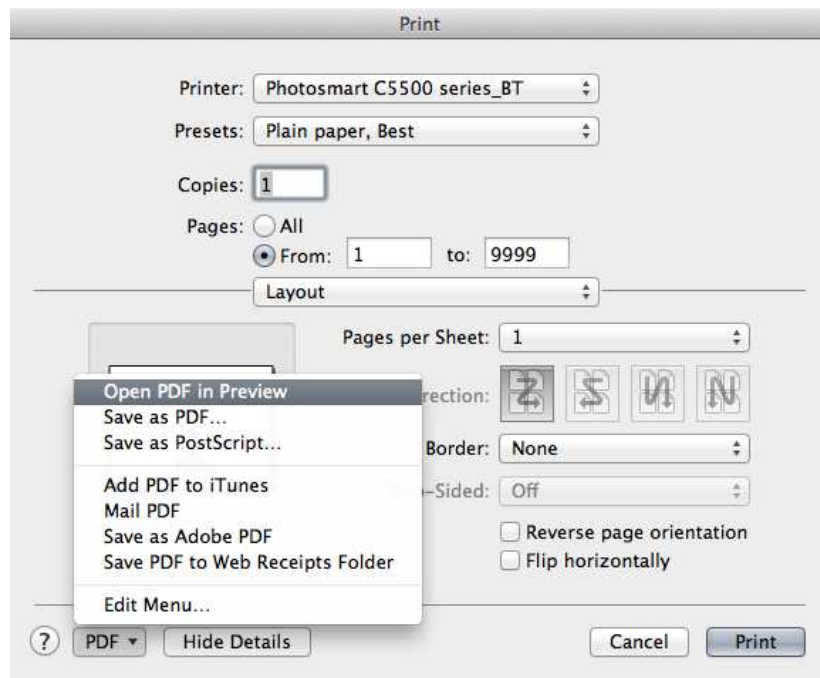
Option 1:

1. Sand the copper surface (phenolic plate) with a little rough sandpaper (200 or higher number preferably, the purpose is to clear the board not to remove the copper).
2. Clean with soap and water phenolic plate (once clean, do not touch it, the goal is that the surface is free of dirt, once this is done the board will have a brighter surface).
3. Open the project in Kicad and run the PCBnew program.
4. Print in the transfer page the tracks and pads of your design: File -> Print. You may do so by following the picture below:

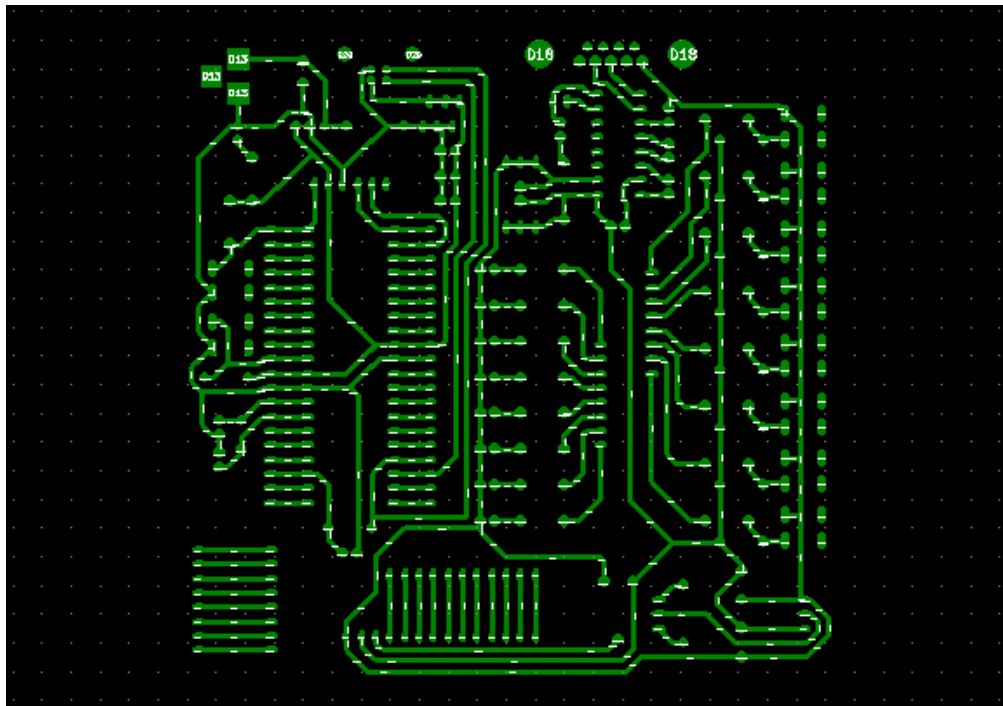


NOTE: B.Cu (Bottom Cu) means that only the copper tracks are going to be printed.

5. Click on 'Print'.
6. Another window will show up convert the design in a pdf format.

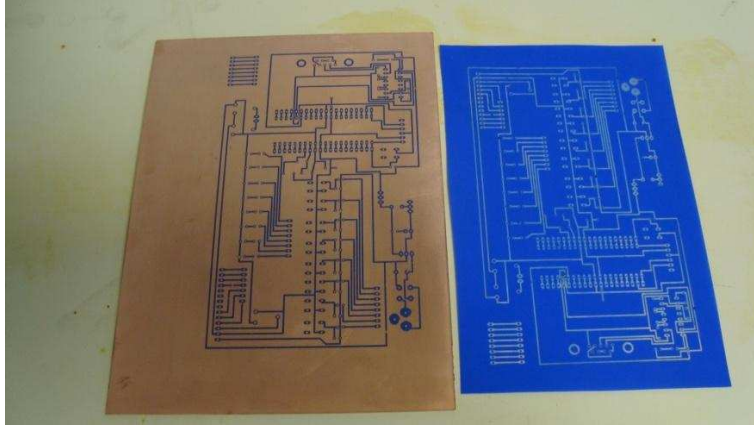


7. On the pdf, you will be able to see the copper tracks with the right thickness and scale:



NOTE: DO NOT USE this picture as an example to drill holes or place components. Each design is unique and there might be some variations.

8. Print this pdf on the Transfer page.
9. Give over the tracks, of the transfer sheet, to the phenolic plate with the help of a hot iron. Place the sheet with print side on the copper surface and apply heat until the complete transfer of the tracks.



NOTE: DO NOT USE this picture as an example to drill holes or place components. Each design is unique and there might be some variations. This model is a 20 x 15. Yours must be a 15 x 15 model.

10. Once you have the phenolic plate with the layout printed on it, immerse it in the ferric chloride until the copper is dissolved, the instructions of the chloride are locate in its container. The mixture must be moving constantly.

WARNING.

Ferric chloride or **iron chloride (iii)** is harmful and unstable at high temperatures (above 37 ° C). Do not dispose of it in the sewers. Wash thoroughly with water in case of skin contact.



The pH of this chemical is within the acid realm. Please, use glass or plastic containers for the treatment of the copper plate. Adding water helps to dilute the solution but it may generate an exothermic reaction and releases dangerous gases in the process, including hydrogen. Avoid prolonged exposure with this gases.

Steps for the acid bath.

- a. Get a container (glass or plastic are fine) large enough to hold the copper plate inside.
- b. Add enough ferric chloride (along with a quarter part of water) to cover the whole copper plate like in the picture below:



NOTE: DO NOT USE this picture as an example to drill holes or place components. Each design is unique and there might be some variations. This model is a 20 x 15. Yours must be a 15 x 15 model.

- c. Shake (gently) the container for around 15 to 20 minutes.
 - d. Once the exposed copper plate is wasted, remove the plate and dispose of the acid in the laboratory storage room.
-
11. Clean the plate with soap and water and remove the black ink with a sandpaper, preferably under running water to avoid staining the card.
 12. Drill the hole on the plate according to the size of each of the elements, the bits that you will use are around 30 mils, there are some components who need larger perforations.
 13. Once you are done, clean with soap and water again.
 14. Weld one component at a time.
 15. Check the operation of the card with the Instructor.

Option 2:

Visit the page <http://diec.mty.itesm.mx/labpcb/>

Follow the procedure for the one face board **“without Through-Holes”**.

Practice 4 – Introduction to using IDE MPLAB X

Pre-Lab 4

NOTE: Solve these activities using the datasheet for the PICF45K50. Failure to meet this specification, the exercise will NOT be accepted.

1. Make a table where the following instructions are thoroughly described. The exercise should be done in the same way like the following example (60%):

Instructions: BCF, BSF, BTFSS, CLRF, INCF. (12% each)

Instruction	Description	Example	Initial Conditions	Description
BTFSC Page 418 from the datasheet	This instruction tests a single bit from a register. If the bit is a zero, then it skips the next instruction after the BTFSC. If the bit is a one, the then next instruction is executed without change. The syntax for this instruction is BTFSC register, # bit. Where #bit can be a number between 0-7.	cycle: BTFSC PORTB, 1 BCF PORTA,0 GOTO cycle	If PORTB = 1001 0000	First, it ask if the PORTB line has a zero in bit 1 (the second bit from right to left) Since the judgment is affirmative, the BCF instruction is ignored and it performs the GOTO instruction. This instruction will jump to the label "cycle" and repeat the process again.

2. Describe your words what is the difference between the following instructions, and provide a simple process for each instance (40%).
MOVF, MOVFF, MOVLW, MOVWF (10% each)

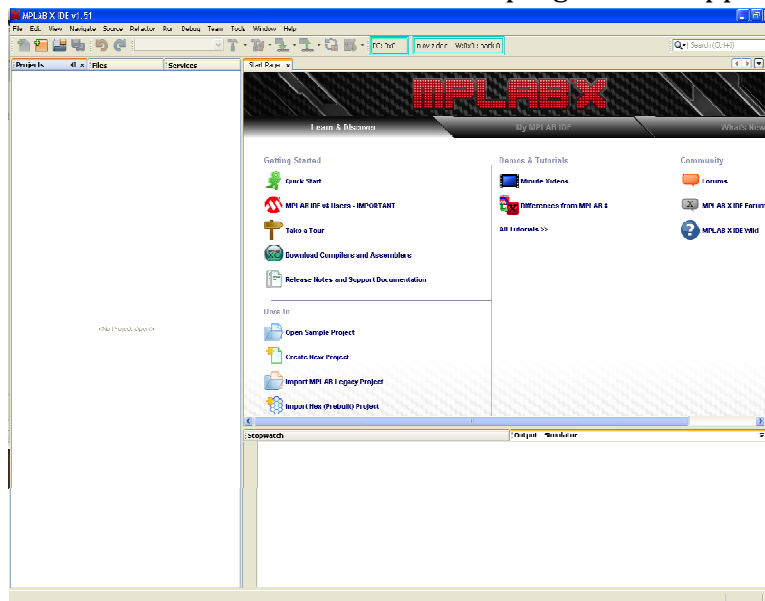
Objective

MPLAB X IDE is a software program running on a PC (Windows, Mac OS, Linux or Web) to develop applications for Microchip microcontrollers and digital signal controllers.

The objective of this practice is to help you discover and learn to use the various features available to simplify and improve development activities.

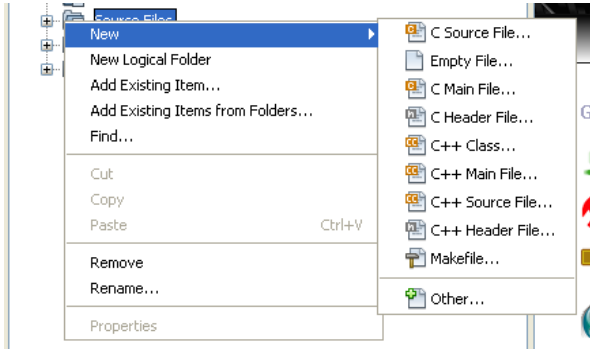
Activities

1. Install MPLAB X IDE, the installer is in the folder 'software' USB annex to the manual. If a warning message about a compiler not being installed, ignore it.
ALTERNATIVE: You can use the Web IDE by going to this page and signing up to Microchip Web service.
(<https://mplabxpress.microchip.com/mplabcloud/ide>)
2. Open IDE MPLAB X. The main screen of the program will appear.



3. Create a new project: File -> New Project. Select Microchip Embedded as 'Category' and Standalone Project as 'Project'. Click on 'Next'.
4. In the 'Family' option select: 8-bit MCUs Advanced (PIC18). In 'Device' select PIC18F45K50. Click on 'Next'.
5. Select Simulator as Hardware Tools and click 'Next'.
6. As a compiler, select: MPASM (v5.47) [C: \ Program Files \ Microchip \ MPLABX \ mpasmx]. Click on 'Next'.

7. Name the project “Practice4” and select the folder in which all our files are going to be stored. Click on ‘Finish’.
8. To start to program, you need to create an ‘.asm’ file. Right-click on the ‘Source File’ folder, then ‘New’ and click on ‘Empty File’.



9. A window will show up with the name ‘New Empty File’, Change its name to “Practice4.asm”. When done, click on ‘Finish’. Expand the ‘Source Files’ folder and the ‘Practice4.asm’ file should appear. There, you can start to write code.



10. TYPE BY HAND THE FOLLOWING CODE into the Practice4.asm:

```

RADIX      DEC                                ; SET DECIMAL AS DEFAULT BASE
PROCESSOR  18F45K50                          ; SET PROCESSOR TYPE
#include    <P18F45K50.INC>

;
;   VARIABLE'S DEFINITION SECTION
;
variable1  EQU            0
variable2  EQU            1
;
;   *** ONLY NEEDED FOR SOFTWARE SIMULATION ***
;
ORG        0                                ; RESET VECTOR
GOTO       0X1000
;
ORG        0X08                             ; HIGH INTERRUPT VECTOR
GOTO       0X1008
;
ORG        0X18                             ; LOW INTERRUPT VECTOR
GOTO       0X1018
;
;   *** END OF CODE FOR SOFTWARE SIMULATION ***
;
;
;   *** START OF PROGRAM ***
;
;   JUMP VECTORS
;
ORG        0X1000                             ; RESET VECTOR
GOTO       MAIN
ORG        0X1008                             ; HIGH INTERRUPT VECTOR
; GOTO     ISR_HIGH                          ; UNCOMMENT WHEN NEEDED
ORG        0X1018                             ; LOW INTERRUPT VECTOR
; GOTO     ISR_LOW                           ; UNCOMMENT WHEN NEEDED
;
;

```

STUDENTS' MANUAL

```
; RESOURCE INITIALIZATION
;

MAIN:
    MOVLB    15
    CLRF     ANSEL, BANKED           ; SET ALL PINS AS DIGITAL
    CLRF     ANSELB, BANKED          ; SET ALL PINS AS DIGITAL
    BSF      TRISC, 0                ; SET RC0 AS OUTPUT
    BSF      TRISC, 1
    CLRF     TRISB
    CLRF     LATB

;
; MAIN LOOP
;

LOOP:
    BTFSS    PORTC, 0
    CALL     boton1
    BTFSS    PORTC, 1
    CALL     boton2
    BRA      LOOP

boton1:
    CALL     delay
    BSF      LATB, 0
    RETURN

boton2:
    CALL     delay
    BSF      LATB, 1
    RETURN


delay:
    MOVLW    0XFF
    MOVWF    variable1
    MOVWF    variable2

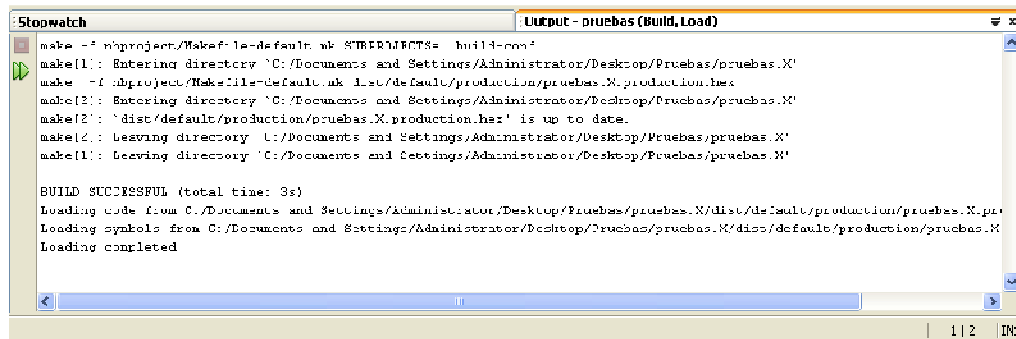
loop1:
    DECFSZ   variable2
    GOTO     loop1
    DECFSZ   variable1
    GOTO     loop1
    RETURN

END
```

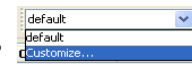
Code Explanation:

- Text after a ';' is considered as a comment
- Each port is controlled by three main registers: TRISx, LATx and PORTx.
- In this scenario, we will work with registers B and C.
- To indicate the pin as an Output, we set a zero in the TRISB
- If we want a pin to be Input, we set its TRISx to one.
- To assign a value to a pin, you can do so by using the register PORTx.
- One way to do it is by using WREG.
- The part indicated as 'Software Simulation', is needed in all your codes to allow programming through USB cable.

11. Once you are done writing your code, click on the icon 'Build Project' , located in the upper side of the toolbar menu.
12. If the program was written correctly, the 'output' tab at the bottom of MPLAB X will display a success message.





13. You can transfer the program to the microcontroller and test it.
14. You already know how to program, how to compile the code and how to pass it on to the electronic card. It's time to learn to simulate your program. To do this, you must first configure your project data. Click on 'Configuration',




- located in the upper toolbar, then select 'Customize...'
15. Select the 'Simulator' tab. In the option 'Processor frequency (Fcyc)' set the value to "4". In the section of 'Frequency In' select "MHz". Click on OK.
16. Compile again the program with the option 'Build Project'. To simulate the




- program, click on the icon 'Debug Project'  located on the upper toolbar.
17. The program must now be running as a simulation, and to prove it, you can find the loading rectangle on the bottom-right corner of the screen .
18. It is possible to simulate the program line by line. To do so, you may use the



- button 'pause'  located on the upper toolbar. You will notice a highlighted line in the code (here is where the program has stopped). Now you can use the button 'Step Over' to execute each line, one by one.
19. Now that you can control the execution of each line, you may want to check on the registers and ports values during each step of the code. Go to the toolbar and click on Window->Debugging-Watches. Click on the tab named



'Watches' . Right click on the row '<Enter new watch>' and select 'New Watch'. The variables created in your program are located in

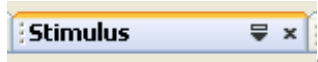
the option 'Global Symbols'; the registers of the microcontroller are located in the tab 'SFR's'. Select does you wish to inspect and click OK.

20. On some occasions, you will need to measure execution times in the program; and to do it, you may use the function 'Stopwatch'. Go to the toolbar and click on Window->Debugging->Stopwatch. Click on the tab:

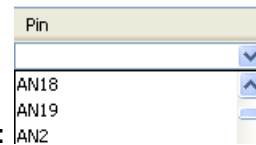


. Once inside the tab, to left, will find 4 buttons to help you in measuring execution time. The function of each button can be found by hovering the mouse pointer over each icon.

21. As it was mentioned before, the tab 'Watches' its used to inspect the values of the variables, ports, registers, etc. inside our program. In some cases it is necessary to force values on this variables, so that we can recreate an event like pushing a button. To do it, you must use the option 'Stimulus'. Go to the toolbar and click on Window->Simulator->Stimulus. Click on the tab:



. To add a port, click on the white row under the 'Pin'



section and select the Pin you wish to use: . On the row of 'action' select the behavior of that Pin: set to high, low or toggle. To activate the behavior, you must click on the box 'fire', by doing so, the program will respond to the new signal on the next 'Step over'.

22. To stop the step by step simulation and go back to automatic/continuous

execution, you may click on the button 'Play'  on the upper toolbar.

23. You can also set 'breakpoints' on the lines of the code, by doing a double click on the line number of that instruction. A red box should appear on it



. A 'breakpoint' help to indicate a full stop in the simulation once the program reach that line in particular.

24. Record with screenshots the step-by-step of the example code with PORTC0 as 1 and PORTC1 as 0. Include the watches and stimulus windows.

Practice 5 – Input/Output Ports

Pre-lab 5

NOTE: Solve these activities using the datasheet for the PICF45K50. Failure to meet this specification, the exercise will NOT be accepted.

1. Answer the following questions (make sure to add a reference) (100%):

- What is the purpose of the registers TRISA, TRISB, TRISC, and TRISD? (15%)
- What is the loading value for TRISB, so that all the Register-B worked as an input port? (15%)
- What is the loading value for TRISB, so that the 7 more significant bits (XXXX-XXXN) of the Register-B worked as an output port? (15%)
- What is the main difference between PORTB, TRISB and LATB? (15%)
- **Explain in your own words.** What is an anti-rebounds routine and why it is important? (40%)

Objective

Familiarize yourself with the basic instructions for configuration and management of input/output of Microcontroller; as well as, the use and management of the level assembler routines.

At the end of this practice, the student will be capable of:

- Configure, read and write about the different ports of the device
- Apply common routines anti-rebounds and timing through cycles.

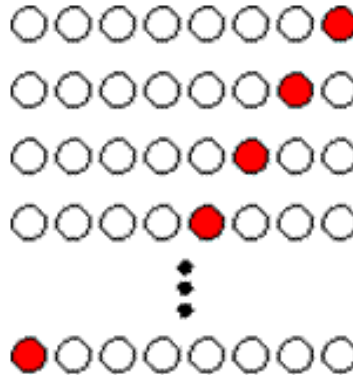
Requirements

- PCB developed on practices 1, 2, and 3.
- Knowing the basic tools and functions of the Debugging software IDE MPLAB X from practice 4.

Activities

1. Elaborate a program that upon activation it should light up the less significant led (NNNN-NNNX) for one second, and then turn it off for another second, this should behave like a loop. You will also program 3 different buttons. Each button will activate a routine; first button "B1" will activate routine 'a', second "B2" is 'b' and third "B3" is 'c'. Program can't jump from routine one routine to another. User must deactivate the routine with the same button if he wishes to switch to another routine.
 - a. Toggle led will start to move to the left like in picture 1. On each second the led jumps to the next position. When the led reaches the last position (bit 7), it should reappear on the other side again (bit 0). This is an endless loop until the button B1 is pressed again. Upon exiting the routine, the led must keep blinking on its last position with the same 1 second interval as before.
 - b. Toggle led will start to move this time to the right. On each second the led jumps to the next position. When the led reaches the last position (bit 0), it should reappear on the other side again (bit 7). This is an endless loop until the button B2 is pressed again. Upon exiting the routine, the led must keep blinking on its last position with the same 1 second interval as before.
 - c. Toggle led will start to move this time it MUST start to the left. On each second the led jumps to the next position. When the led reaches the last position (bit 7 or bit 0), it will switch direction and bounce back to its previous position until it reaches the other end. This is an endless loop until the button B3 is pressed again. Upon exiting the

routine, the led must keep blinking on its last position with the same 1 second interval as before.



Picture 1. Led behavior for section 1a.

RECOMMENDATION: USE PORTB FOR THE LEDS.

2. Elaborate a program with 2 counters. Program must include 4 buttons: Gray1 (for the gray counter), BCD2 (for the BCD counter), Plus3 (for adding +1), and Minus4 (for adding -1). The program must have 2 routines: Gray1 is for routine 'a' and BCD2 is for routine 'b'. Student may decide initial conditions.
 - a. First state is counter Gray. Leds should represent a Gray counter. Each time the user presses Plus3, the counter must go up by one and down by one with Minus4 (according to Gray sequence).
 - b. Second state is counter BCD. Leds should represent a BCD counter. Each time the user presses Plus3, the counter must go up by one and down by one with Minus4 (according to BCD sequence).

RECOMMENDATION: USE PORTB FOR THE LEDS.

Practice 6 – Matrix Keyboard

Pre-lab 6

NOTE: Solve these activities using the datasheet for the PICF45K50. Failure to meet this specification, the exercise will NOT be accepted.

1. Answer the following questions (make sure to add a reference) (60%)

- What is the main function of a keyboard? (5%)
- How many pins does a 4x4 Matrix Keyboard has? (5%)
- With an illustration, explain with pins are columns and rows? [Tip: you may use a multi-meter] (10%)
- Which port or ports, from the PIC18, will you use as Input or Output, to link the Matrix Keyboard? (20%)
- What is a pull-up resistor? (10%)
- On the 4x4 Matrix Keyboard that we are using, where does the pull-up should be located? (10%)

2. Investigate how to use and enable the pull-ups of the PORTB? (40%)

Objective

Knowing the functions and interactions of the pull-up with an input/output port, and acquiring data from keyboard matrix by the sweeping method.

At the end of this practice, the student will be capable to:

- Identify and know the importance of the pull-up/pull-down settings in a microcontroller.
- Understand concept of sweeping for data acquisition in the matrices.

Requirements

- Having the pre-lab 3 delivered.
- Read the "I/O ports" section of the Microcontroller Datasheet.
- Requested, from the storage room, a Matrix Keyboard.

Activities

1. Connect the matrix keyboard to the Microcontroller. Configure the microcontroller so that when a button from the matrix keyboard is pressed, it should send a pattern to another port (connected to the leds).

Button (Input)	Leds (Output)
1	0000 0001
2	0000 0010
3	0000 0011
4	0000 0100
5	0000 0101
6	0000 0110
7	0000 0111
8	0000 1000
9	0000 1001
0	0000 0000
A	0001 1111
B	0010 1111
C	0011 1111
D	0100 1111
# / E	0101 1111
* / F	0110 1111

2. Program the microcontroller so that the following operations are capable:
A as Addition, B as Subtraction, C as Multiplication and D as Division.
 Handle the **E as Equal** and **F as Clear Screen**.

3. **{EXTRA POINTS}** Try to display the result of the operation in units and tens. What does this means? Well, instead of displaying the following answer:

$6 \times 8 = 48 \rightarrow \{\text{■ ■ ■ ■ ■ ■ ■ ■}\}$ stand for 48 in decimal

stand for 30 in hexadecimal

stand for 00110000 in binary

We write the result in a different way

$6 \times 8 = 48 \rightarrow \{\text{■ ■ ■ ■ ■ ■ ■ ■}\}$ stand for 72 in decimal

stand for 48 in hexadecimal

stand for 01001000 in binary

The 4 most significant bits are the tens and the 4 less significant bits are the units.

You can try using the DAW operation (read the PIC18 datasheet for more information); still, there are many ways to do this kind of conversions. Use your imagination.

Practice 7 – LCD (Liquid Crystal Display)

Pre-lab 7

NOTE: Solve these activities using the datasheet for the PICF45K50. Failure to meet this specification, the exercise will NOT be accepted.

Read the file “LCD.pdf” to answer the following questions:

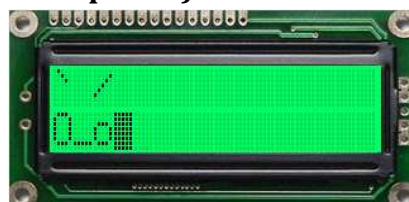
1. How many pins do you need to connect between the LCD and the PIC? (5%)
2. Pin number for the following signals in the LCD device: (5%)

GND _____
 VCC _____
 Dimmer _____

4. Identify the symbols and the function of the following pins of the LCD. (15%)

Pin	Symbols and Functions
4	
5	
6	
7 – 14	

5. The data bus can be configure to work with 4 bits or 8 bits. What kind of changes must be done, to switch from 4 to 8 and vice versa? (10%)
6. Which pins would you use for a 4 bits mode? (5%)
7. Describe the main difference of working with 4 bits or 8 bits in the data bus. (10%)
8. Draw a Flow Diagram for the LCD initialization with the 8 bit mode. (20%)
9. Write a Pseudo Code or table input, to show the following image in the display: (30%) **(+10% extra points)**



You may use the following [page](#) for help. Try your best.

Objective

To study and use a Liquid Crystal Display (LCD) as a graphic terminal for the Microcontroller. Knowing the main functions and characteristics of the device.

At the end of this practice, the student will be capable to:

- Configurator and integrate an LCD to the following practices of this lab.

Requirements

- Read the "LCD.pdf" and "HD44780.pdf" files.
- Requested, from the storage room, a LCD device.

Activities

1. Realize the same activity as Lab 6.2 using the same Matrix Keyboard and operations, but instead of LED use the LCD as your graphic terminal. The program must have the next extra requirements:
 - Configure the LCD with the 8 bits mode
 - Operation must appear in the first row.
 - Result of the operation must appear in the second row.
 - F key must clear the page and return the cursor to home position on the first row.

Practice 8 – Timers

Pre-lab 8

1. Answer the following questions:

- Which is the register to configure the timer0? (12%)
- What does the bit TMR0ON from register T0CON do? (12%)
- What does the bit T08BIT from register T0CON do? (12%)
- What does the bit T0CS from register T0CON do? (12%)
- What does the bit PSA from register T0CON do? (12%)
- How long does it take to the timer0 to increment by one, without a pre-scaler assigned? (40%)

Objective

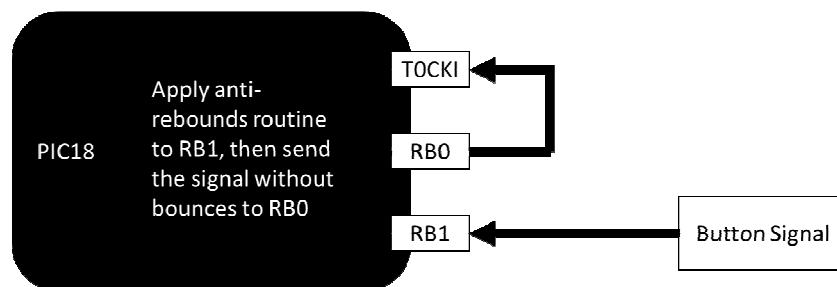
Implement programs in assembler modules which use timers to count events and times.

At the end of this practice, the student will be capable to:

- Perform the calculations required in their applications to configure and program timers.
- Perform miscellaneous timing routines as signal generators.

Activities

1. Write a program that allows a LED to be lit for 60ms and off for another 60ms, using the TIMER0. Check that the times are met using an oscilloscope.
 - Report and clearly explain to the instructor how the exercise was planned and how it was calculated.
 - Report formulas and calculations as justifying the necessary account for TIMER0.
 - Report your reason to choose the mode 8 or 16 bits, T0CON configuration and the initial count for the timer.
2. Repeat the same activity 1 but using the maximum time available by TIMER0 using the 8-bit mode. What is this maximum value? Show calculations.
3. Write a program that allows a led to go on, after pressing one button 10 times: Follow the next specifications.
 - a. Remember to use an anti-rebounds routine. You can do it by implementing a filter with the PIC18.

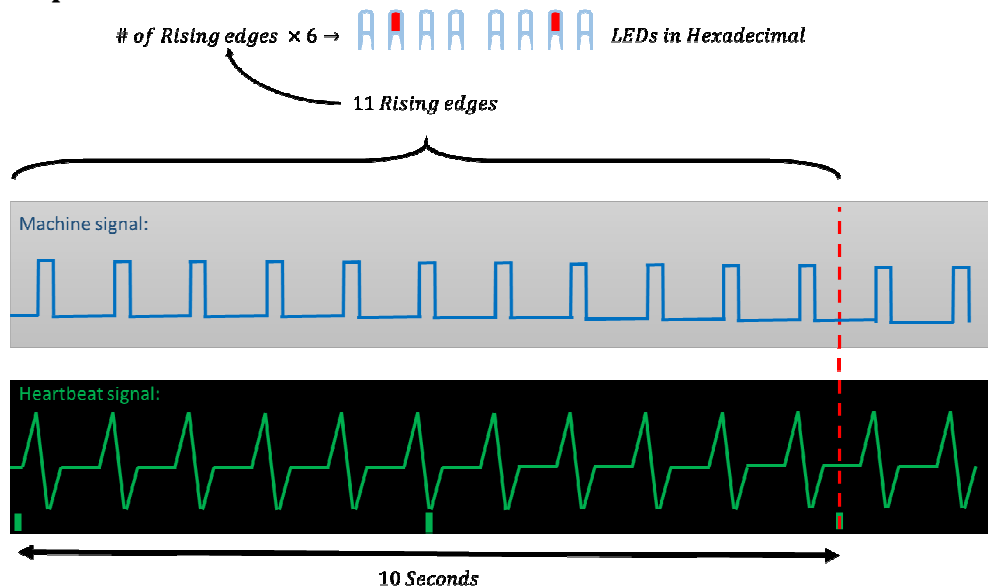


- b. Use input T0CKI to accomplish this (Consult the datasheet).
- c. Configure TIMER0 to use T0CKI as an external asynchronous clock.

4. Heartbeat Monitoring System (HMS). THIS IS A MUST DO FOR ENGINEERS IN BIOMEDICINE.

Assume that a HMS measures the heartbeats during ten seconds. After ten seconds, the system will display on the LEDs the amount of Heartbeats, during that cycle, multiplied by six. You will need to configure TIMER0 and TIMER1 (read the datasheet for more help). The following list will include some specifications of the software:

- Read the heartbeats with the T0CKI using a push button. A heartbeat will be interpreted as a rising-edge (signal going from 0 to 1). Remember to remove the button bounces.
- Display amount count times six on the LEDs every 10 seconds (if there was no heartbeat during a 10 seconds period, the LED must display a zero in hexadecimal).
- Configure the TIMER0 to count the heartbeat pulses (this signal will be simulated by a push button).
- Configure the TIMER1 to count the 10 seconds period between measurements. **Include on the report your configuration procedure.**



Practice 9 – Interruptions

Pre-lab 9

1. **Answer the following questions:**

- What is the purpose of the pin INT0? (15%)
- How do you enable the interrupt INT0? **Tip:** Review the INTCON section on the datasheet. (15%)
- If you connect a push button on the pin INT0, how would you remove the bounce noise on the signal? (30%)

2. **For a 8 Hz square signal:**

- Determine the period? (10%)
- How would generate this signal using only interruptions? (30%)

Objective

Know and use the services offered by the internal and external interruptions, and highlighting the importance that these have in the software.

At the end of this practice, the student will be capable to:

- Make interruption routines for both software and hardware.
- Replicate the function of a wave generators by using interruptions.

Activities

1. Count the events using interrupts. Use an anti-rebounds routine by connecting PushButton1 to the PIC, remove any bounce noise, gather the output on pin RD7 and connect it to INT0. Keep all the LEDs in off state and only activate LED0 when PushButton1 is pressed 10 times. Inside the interrupt routine, you must have a signal counter. **YOU CAN NOT USE TIMERS, USE INTERRUPTS TO SIMULATE THIS BEHAVIOR.**
2. Generate a 2 Hz square signal with a duty cycle of 50% (50% on, 50% off), using the TIMER1. Use an oscilloscope to measure the signal. Include your procedures and operations on the report. **DO NOT USE THE PWM MODULE, USE**

Practice 10 – Module Capture/Compare/PWM/ADC

Pre lab 10

1. Answer the following questions:
 - What is duty cycle? (10%)
 - Describe the Sampling Theorem. (10%)
 - How do you initialize the PWM module to generate a square signal with a duty cycle of 75%? (15%)
 - List and describe all the necessary registers for the PWM module. (15%)
 - Which is the maximum period that you can obtain with the PWM module, if you are using a 4 MHz oscillator? (15%)
2. Describe the procedure to calculate the time between two events, using the CAPTURE module. (35%)

Objective

Develop routines for using the modules ADC and CCP; applying them in the control of the width of a pulse periodic signal.

At the end of this practice, the student will be capable to:

- Controlling the speed of a DC motors by varying the pulse width of a square wave.
- Implement programs involving conversions of Analog – Digital data.

Activities

1. PWM. Generate a square wave using the PWM module, with a duty cycle of 75% and the max period available. Connect the output to a LED to view the behavior.
2. COMPARE. Make your own program to test the usefulness and the capacities of the COMPARE module.
 - a. Include a small description of what your program does.
 - b. Include a picture of the code's states behavior.
 - c. **(IF REQUIRED)** Include a picture or table for the necessary hardware connections.
3. CAPTURE. Measure the period of a square signal with the CCP1 (pin 17).
 - a. Configure the CCP1 pin to work as Capture.
 - b. You should work with rising-edge signals.
 - c. Sampling speed should be 1MHz.
 - d. DO NOT USE TIMER 3.
 - e. Display the period value (in micro-Seconds) in a BCD, or Hexadecimal format, using the PORTD as outputs. Use the following table to guide you.

Input Signal		Output PORTD
Frequency (Hz)	Period (ms)	
1 MHz	1~2	0000 00XX
500 KHz	2	0000 0010
100 KHz	10	0000 1010
10 KHz	100	0110 0100
4 KHz	250	1111 1010

4. ADC. Write a program that performs analog - digital conversion of the AN0 channel and change the pulse width of the PWM. You may chose the behavior between the PWM and the analog signal. Connect AN0 pin to a pot and the output pin to an oscilloscope.

Practice 11 – Serial Port

Pre lab 11

Practice 11 doesn't have a Pre-lab.

Objective

Configure and establish a communication line between the PIC and a computer, using only the module USART.

At the end of this practice, the student will be capable to:

- Achieve a data interchange using the protocol RS232 (Serial Interface)

Activities

Part 1:

Send a continuous 0x55 through the TX pin and connect it to an Oscilloscope. Try to identify the header, tail and parity section of the data package.

Part 2:

Using an USB cable – Serial DB9 and a hyper terminal software on a PC (putty.exe), send a letter into the PIC18. If the letter was an UPPERCASE, the PIC18 should return the letter to the PC in the lowercase format, and vice versa. Remember to use the ACSII format.

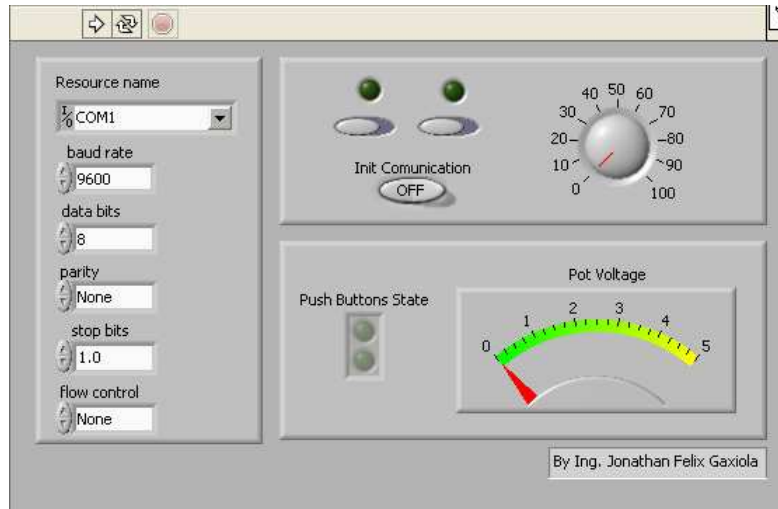
Parte 3 (Demonstrative only):


1. Download and install 'Run-Time-Engine 2011' to open LabView projects.
2. Download and install 'NI Visa' to enable a serial port into LabView.
3. Download and install the drivers for the DB9 serial cable.
4. Copy into a computer the folder 'Archivos_USB_Serial_Emulator'. Given to you at the beginning of the semester.
5. Open the MPLABX Project inside the folder. Look into 'Proyecto_PIC'.
6. Compile the code of this project and load it to the PIC.
7. Do the following connections:

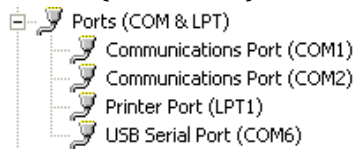
PIC18 Pins	Device
RB0	Led
RB1	Led
RB2	Button
RB3	Button
RC2	Led
RE0	Pot


8. Connect the PIC18 to your computer (Both power and DB9).

9. Open the .exe program located in: Archivos_LabView -> USB_Serial_Emulator_Exe -> USB_Serial_Emulator.exe
10. Set the values of baud rate, data bits, parity, stop bits and flow control, like the picture below.



11. Click on 'Run Continuosly' 
12. On 'Resource Name' select the COMx corresponding to the PIC. Go inside the 'Device Manager' window to find the COM number, under the section 'Ports (COM & LP)'.



13. Start the communication by pressing the switch 'Init Communication' 
14. You may now interact with the tools of the program.
15. Observe and report the behavior of the PIC or the USB_Serial_Emulator.exe as you test each tool.

Annexes

Annex 1 (Schematic):

