



Nivel intermedio

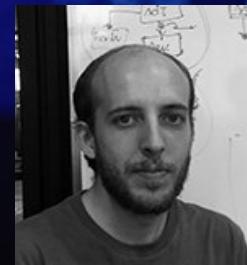
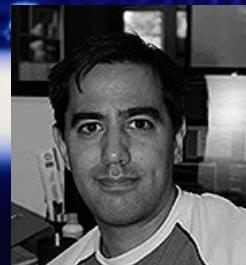
Día 2:

- Uso de GIT.
- Editor de Esquemático: Práctica intermedia.
- Editor de PCB: Práctica intermedia.
- Editores de símbolos y módulos.

Esta presentación corresponde al día 2 del curso ofrecido por INTI-CMNB en relación al proyecto CIAA.

Autores: Diego Brengi - Noelia Scotti - Diego Alamon

Versión 1.0
30/11/15





Conceptos básicos

Control de versiones

Un sistema de control de versiones permite llevar registro (historial) de la evolución de un conjunto de archivos. Esto es de suma importancia en diseños de hardware de mediana y alta complejidad.

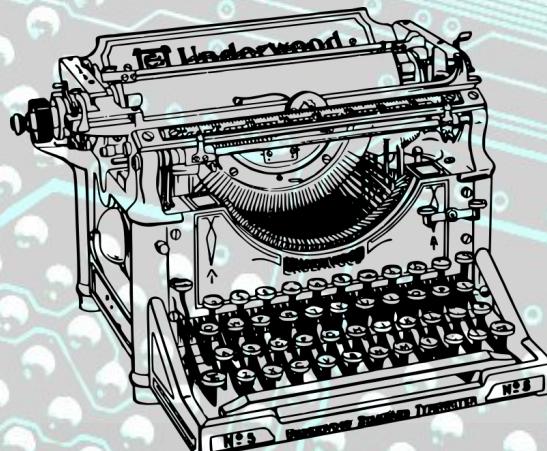
También permite:

- El trabajo en grupo.
- Identificación de etapas o momentos importantes de esos archivos o del conjunto.
- Realizar trabajos derivados.
- Resguardo de los archivos.

GIT es un sistema para el control de versiones

Este NO es un curso de GIT, pero un buen diseño de Hardware conviene realizarlo bajo algún sistema de control de versiones.

Veremos de una manera muy práctica y básica el uso de GIT para el desarrollo de PCBs con KiCad.



Para las explicaciones de GIT nos tomaremos algunas “licencias poéticas”. Consultar su documentación ante cualquier duda.

Presentando a GIT

GIT es un sistema distribuido de control de versiones

URL oficial:

[http://git-scm.com/](http://git-scm.com)

The screenshot shows the official Git website at git-scm.com. The page features a large header with the Git logo and the tagline "git --distributed-even-if-your-workflow-isn't". Below the header, there's a brief introduction: "Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency." To the right of this text is a 3D-style diagram of seven computer towers connected by red and yellow lines, representing a distributed network. Below the introduction, a button says "Learn Git in your browser for free with Try Git." The main content area is divided into several sections: "About" (with a gear icon), "Documentation" (with an open book icon), "Downloads" (with a download arrow icon), and "Community" (with a speech bubble icon). On the right side, there's a large monitor icon displaying "Latest source Release 2.6.3" and "Downloads for Linux". A search bar at the top right says "Search entire site...".

Aprendiendo GIT

El libro de referencia sobre GIT es de descarga gratuita:

URL:

<http://git-scm.com/book/en/v2>

Git - Book

git-scm.com/book/en/v2

git --fast-version-control

About

Documentation

- Reference
- Book
- Videos
- External Links

Blog

Downloads

Community

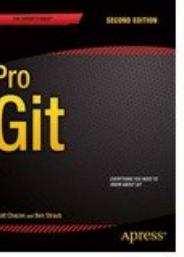
Download this book in PDF, mobi, or ePub form for free.

This book is translated into Deutsch, 简体中文, 正體中文, Français, 日本語, Nederlands, Русский, 한국어, Português (Brasil) and Čeština.

Partial translations available in Arabic, Español, Indonesian,

Book

The entire Pro Git book, written by Scott Chacon and Ben Straub and published by Apress, is available here. All content is licensed under the [Creative Commons Attribution Non Commercial Share Alike 3.0 license](#). Print versions of the book are available on [Amazon.com](#).



2nd Edition (2014)
Switch to 1st Edition

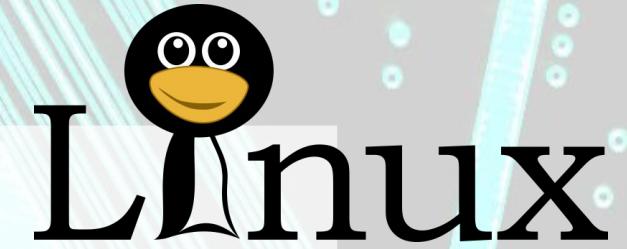
1. Getting Started

- 1.1 About Version Control
- 1.2 A Short History of Git
- 1.3 Git Basics
- 1.4 The Command Line
- 1.5 Installing Git
- 1.6 First-Time Git Setup
- 1.7 Getting Help
- 1.8 Summary

Download Ebook

- pdf
- epub
- mobi
- html

GIT en pocas palabras



Algunas palabras sobre GIT:

- Es un sistema distribuido.
- Fue pensando para el desarrollo del Kernel Linux.
- Fue creado por Linus Torvalds.
- Permite varias modalidades de trabajo.
- Permite trabajar sin conexión a red y luego "sincronizar".
- Facilita el trabajo con ramas de código.
- Al ser distribuido, el contenido del repositorio, con todo el historial, es replicado por cada usuario.
- GIT es una herramienta que funciona por línea de comandos.
- Existen muchas herramientas gráficas que incorporan GIT o lo utilizan por debajo.
- Siempre hay un repositorio local GIT donde se realiza el trabajo. Puede haber remotos o no.

Modos de trabajo (algunos)

USUARIO LOCAL



COMPARTIDO



USUARIO/APORTE



DESARROLLADORES



REPO CENTRAL

USUARIO



REPO CENTRAL

Instalando GIT (Windows y GNU/Linux)

Git es una herramienta de software libre y por lo tanto hay muchas versiones y formas de utilizarlo.

Usaremos su forma básica de línea de comandos, que es la que permite conocerlo cara a cara.

URL Windows:

<https://git-for-windows.github.io/>

En GNU/Linux, viene con cualquier distribución decente.

Por ejemplo para instalarlo:

`apt-get install git`

`yum install git`

`pacman -S git`

`emerge --ask --verbose dev-vcs/git`

`pkg_add git`

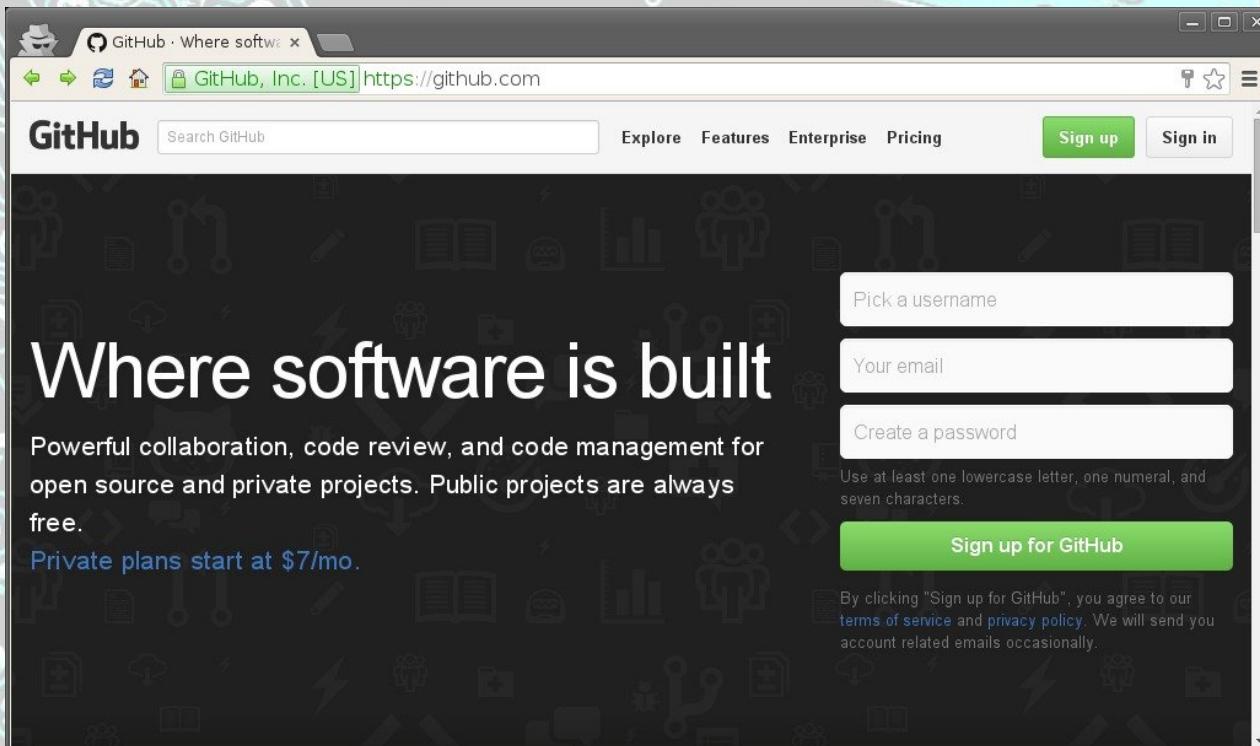


1) Instalar GIT o GIT-Bash según su sistema operativo.

No confundir Github con GIT.

GitHub es un sitio que provee repositorios GIT (alojados en un servidor en internet), pero además brinda acceso y configuración mediante web, facilita la creación de un repositorio compartido y la administración del mismo.

Con GitHub se pueden manejar usuarios y permisos de esos usuarios, consultar y editar un repositorio con el navegador entre otras cosas.



GitHub ofrece servicios de repositorios gratuitos, siempre que uno este dispuesto a que “cualquiera” tenga acceso al repositorio.

Para usar Github con repositorios privados es necesario abonar el servicio. Por esto es muy utilizado para proyectos abiertos o de software libre.

Primeros pasos con GIT (local)

Configuración de GIT

Antes de usar git conviene realizar algunas configuraciones globales básicas. Estas configuraciones aplicarán a todos los repositorios que utilicemos y se guardan en nuestro “home”.

```
$ git config --global user.name "Tu Nombre Completo"  
$ git config --global user.email Tu-email@servidor.com  
$ git config --global http.proxy http://IP_PROXY:PORT  
$ git config --global credential.helper "cache --timeout=28800"  
$ git config --global core.editor emacs
```



- 1) Realizar las configuraciones globales necesarias.
- 2) Verificar listando la configuración.

```
$ git config --list
```

Creando un repositorio GIT

Vamos a crear un repositorio GIT local para experimentar un poco.

```
$ mkdir prueba
```

```
$ cd prueba
```

```
$ git init
```

```
Initialized empty Git repository in /home/diego/prueba/.git
```



- 1) Crear una carpeta nueva.
- 2) Ingresar al directorio.
- 3) Ejecutar “git init”.

Zonas en GIT

Git define tres zonas en el repositorio local.



Directorio de trabajo (Working directory)

Zona de preparación (staging)

Repositorio

Preparación

Commit

Checkout

Es el directorio de trabajo donde editamos, borramos, movemos y creamos archivos con las herramientas tradicionales.

Es donde se va preparando el próximo “commit” o modificación al repositorio. Se usa mediante instrucciones GIT.

Es el repositorio propiamente dicho, donde se guarda la historia de evolución de los archivos. Se lo puede ver como un conjunto ordenado de “commits”

Evolución de un archivo en GIT



Directorio de trabajo
(Working directory)

Zona de preparación
(staging)

Repositorio

lista.txt



Creamos un archivo de texto con una lista de compras.

En este momento para GIT el archivo es desconocido y por lo tanto su estado es:

SIN SEGUIMIENTO (??)

- 1) Seguir esta actividad en su PC.
- 2) Consultar el estado con "git status -s"

Evolución de un archivo en GIT



Directorio de trabajo
(Working directory)

Zona de preparación
(staging)

Repositorio

lista.txt



```
$ git add lista.txt
```

lista.txt



Agregamos el archivo al área de preparación.
Como el archivo no existe en el repo, para GIT es un archivo agregado.

AGREGADO (A)

- 1) Seguir esta actividad en su PC.
- 2) Consultar el estado con "git status -s"

Evolución de un archivo en GIT



Directorio de trabajo
(Working directory)

Zona de preparación
(staging)

Repositorio

lista.txt



```
$ git commit -m  
"Agrego listado."
```

Luego pasamos el archivo del área de preparación al repositorio.

Como el archivo en nuestro directorio de trabajo y en el repo coinciden, GIT esta tranquilo y el estado es “Sin modificación”.

lista.txt



ESTADO:
Sin modificación ()

- 1) Seguir esta actividad en su PC.
- 2) Consultar el estado con “git status -s”

Evolución de un archivo en GIT



Directorio de trabajo
(Working directory)

Zona de preparación
(staging)

Repositorio

lista.txt



Si editamos nuestro archivo en el directorio de trabajo, su estado pasará a modificado.

ESTADO:

Modificado (M) en directorio de trabajo.

lista.txt



- 1) Seguir esta actividad en su PC.
- 2) Consultar el estado con "git status -s"

Evolución de un archivo en GIT



Directorio de trabajo
(Working directory)

lista.txt



```
$ git add lista.txt
```

Zona de preparación
(staging)

lista.txt



Pasamos el archivo modificado al área de preparación.

ESTADO:
Modificado (M) en staging o preparado.

Repositorio

lista.txt



- 1) Seguir esta actividad en su PC.
- 2) Consultar el estado con "git status -s"

Evolución de un archivo en GIT



Directorio de trabajo
(Working directory)

Zona de preparación
(staging)

Repositorio

lista.txt



```
$ git commit -m  
"Agrego una fruta."
```

Realizamos el commit y pasamos las modificaciones en staging al repositorio.

lista.txt



ESTADO:
Sin modificación ()

- 1) Seguir esta actividad en su PC.
- 2) Consultar el estado con "git status -s"

Primeros pasos con GIT (remoto)

Clonando un repositorio

DESARROLLADORES



```
$ git clone  
https://github.com/INTI-CMNB/Practicas-Curso-Kicad.git
```



Podemos conocer donde está nuestro repositorio remoto
(de donde hemos clonado).

```
$ cd Practicas-Curso-Kicad/  
$ git remote -v  
origin https://github.com/INTI-CMNB/Practicas-Curso-Kicad.git (fetch)  
origin https://github.com/INTI-CMNB/Practicas-Curso-Kicad.git (push)  
diego@demeter:~/git/Practicas-Curso-Kicad$
```

Clonando un repositorio

DESARROLLADORES

Cada usuario tiene una copia del repositorio.

Al momento de clonar el repositorio remoto, la copia local del remoto, el repositorio local y el directorio de trabajo están sincronizados.

Repo remoto

README.md



REPO CENTRAL

Directorio de trabajo
(Working directory)

README.md



Zona de preparación
(staging)

README.md

Repo local

README.md

Origin
(copia del remoto)

README.md



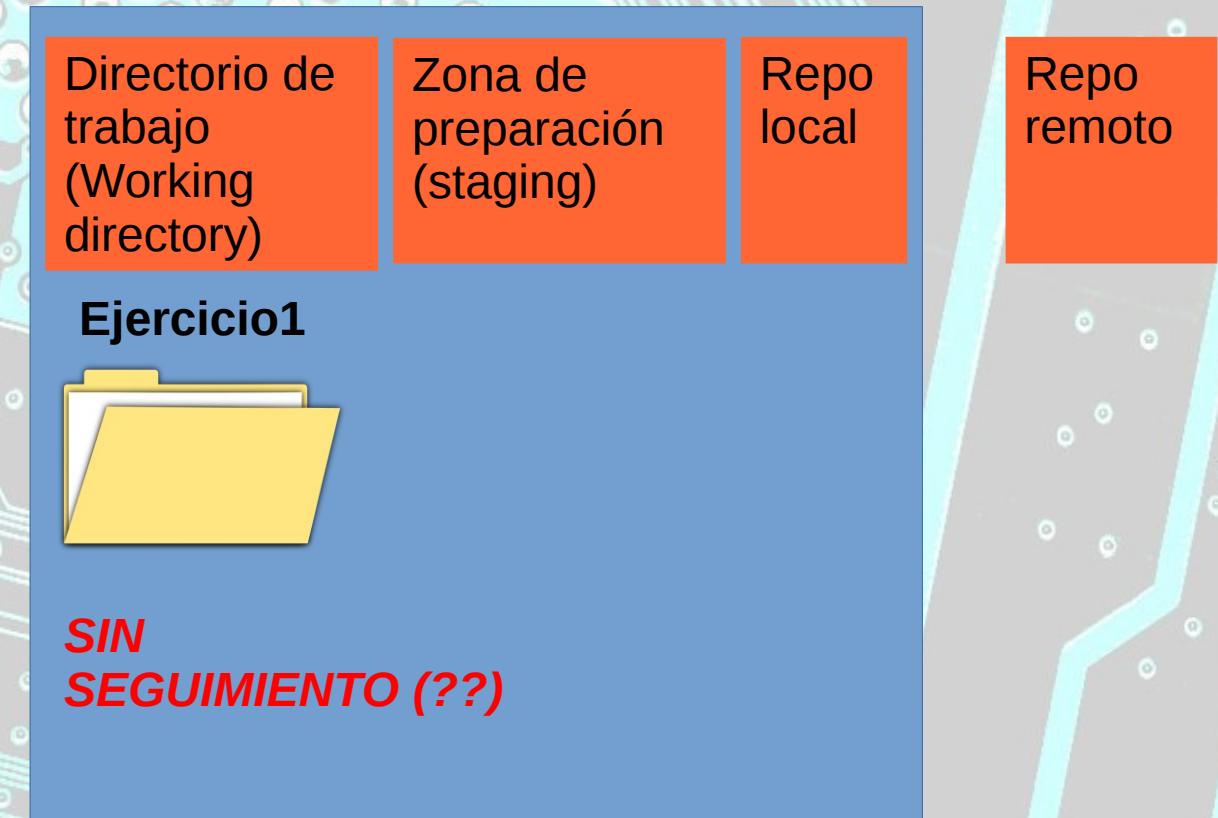
GIT LOCAL

```
$ cd Practicas-Curso-Kicad  
$ ls
```

Ingresando datos nuevos al directorio de trabajo

El ejercicio consiste en subir al repositorio local y luego al remoto el diseño realizado en el ejercicio 1.

- 1) Al traer el repositorio, se observará un directorio con el nombre y apellido de cada participante (creado previamente).
- 2) Mover el directorio que contiene el proyecto Kicad del Ejercicio 1 dentro del directorio correspondiente al participante.
- 3) Renombrar la carpeta a Ejercicio1.
- 4) Verificar el estado “sin seguimiento” de nuestro ejercicio 1.



```
$ cd Diego_Brengi
$ mv /home/diego/Ej1 Ejercicio1
$ git status -s
```

Ingresando datos al área de preparación

- 1) Agregar el directorio al área de staging.
- 2) Verificar el estado de los archivos.

Directorio de trabajo
(Working directory)

Ejercicio1



Zona de preparación
(staging)

Ejercicio1



Repo local

Repo remoto

*Se verán todos los archivos como:
Agregado (A)*

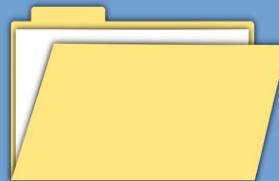
```
$ git add Ejercicio1  
$ git status -s
```

Ingresando datos al repositorio local

- 1)Realizar el commit.
- 2)Verificar el estado de los archivos.
- 3)Ver el log.

Directorio de trabajo
(Working directory)

Ejercicio1



Zona de preparación
(staging)

Ejercicio1



Repo local

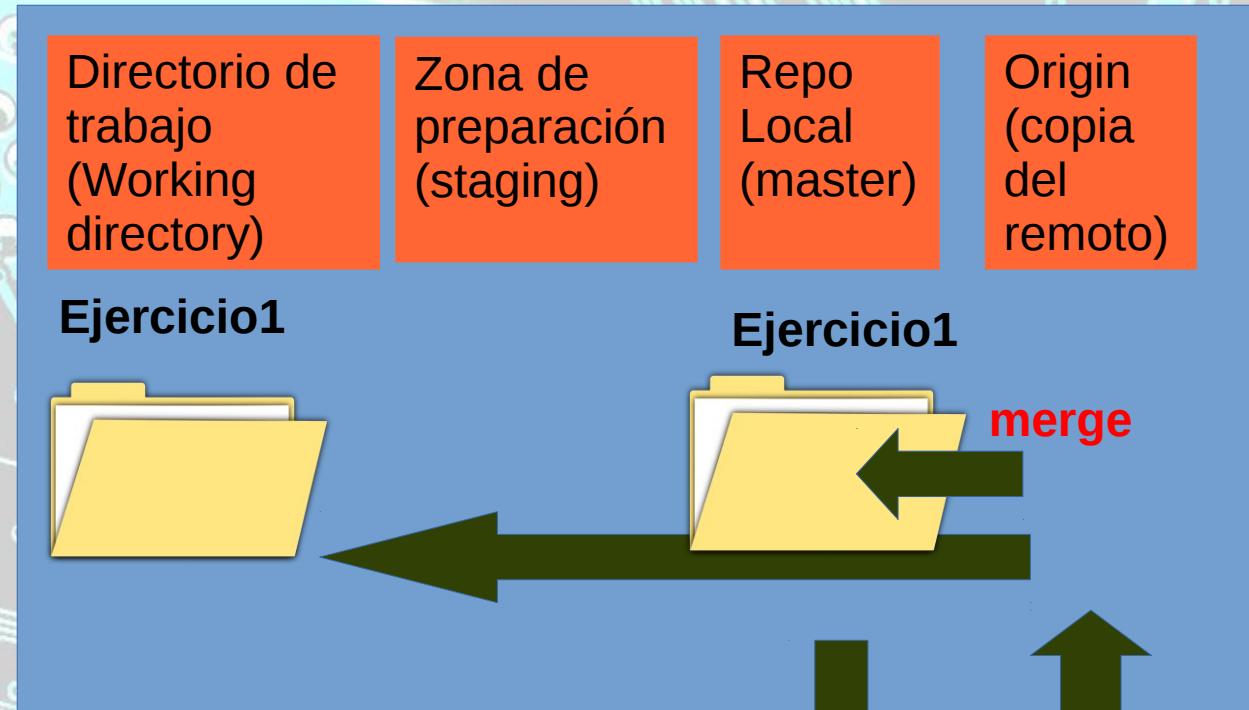
Repo remoto

**SIN
MODIFICAR ()**

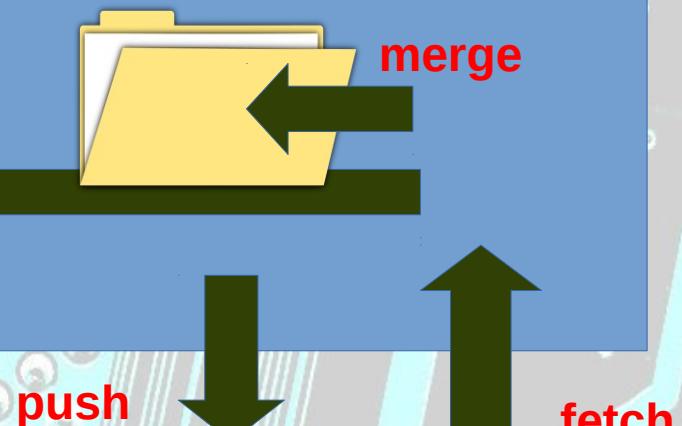
```
$ git commit -m "Agrego mi ejercicio 1."  
$ git status -s  
$ git log
```

Ingresando datos al repositorio remoto

- 1) Ver las diferencias en entre el repo local y el remoto.
- 2) Sincronizar nuestro repositorio con el remoto.
- 3) Enviar nuestro agregado al repo local hacia el repo remoto.



```
$ git fetch  
$ git diff origin master --name-status  
$ git merge  
$ git push origin master
```



Repo remoto

Con el *fetch* y el *merge* es posible que tomemos localmente los ejercicios que han subido antes que nosotros nuestros compañeros.

Como estamos trabajando en directorios diferentes no debería haber conflictos en el *merge*.

Explicando fetch y merge I

Directorio de trabajo (Working directory)



Zona de preparación (staging)

Repo Local (master)

Origin (copia del remoto)



Repo remoto



Todo sincronizado

Directorio de trabajo (Working directory)



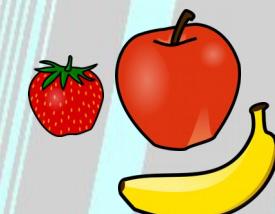
Zona de preparación (staging)

Repo Local (master)

Origin (copia del remoto)



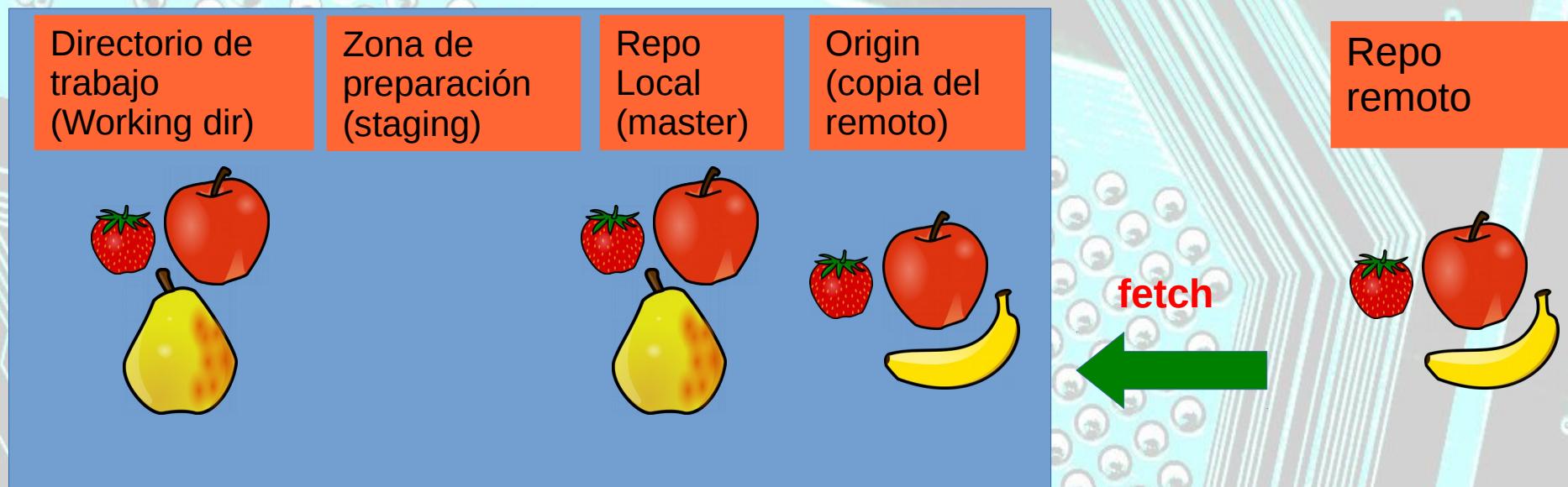
Repo remoto



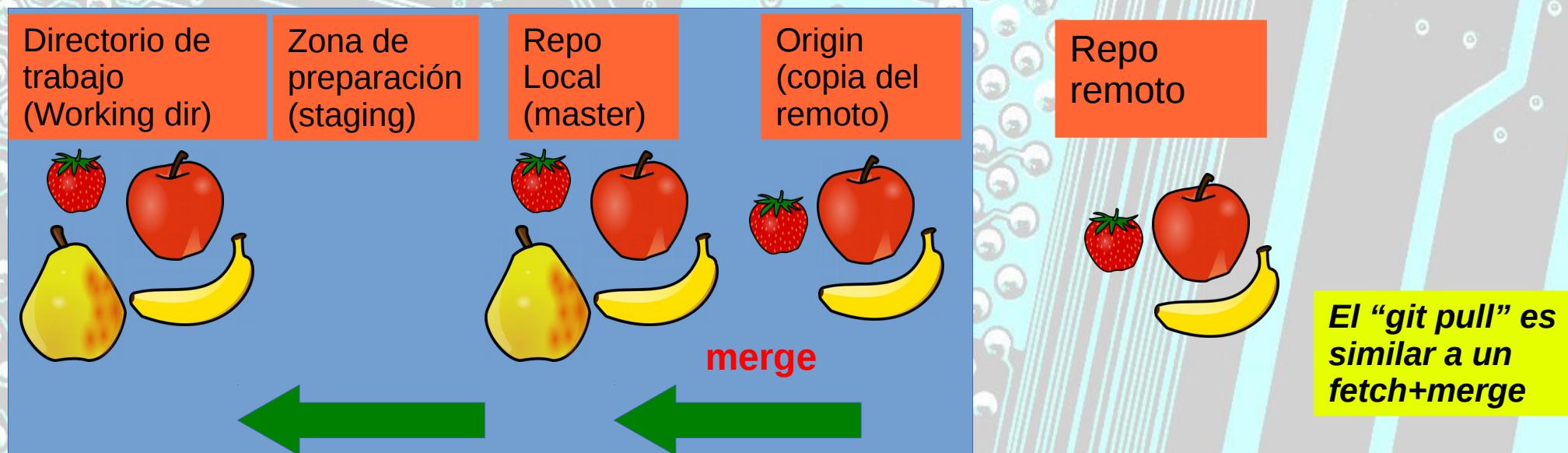
Se agrega el archivo “peras” en el repo local

Se agrega banana en el remoto

Explicando fetch y merge II

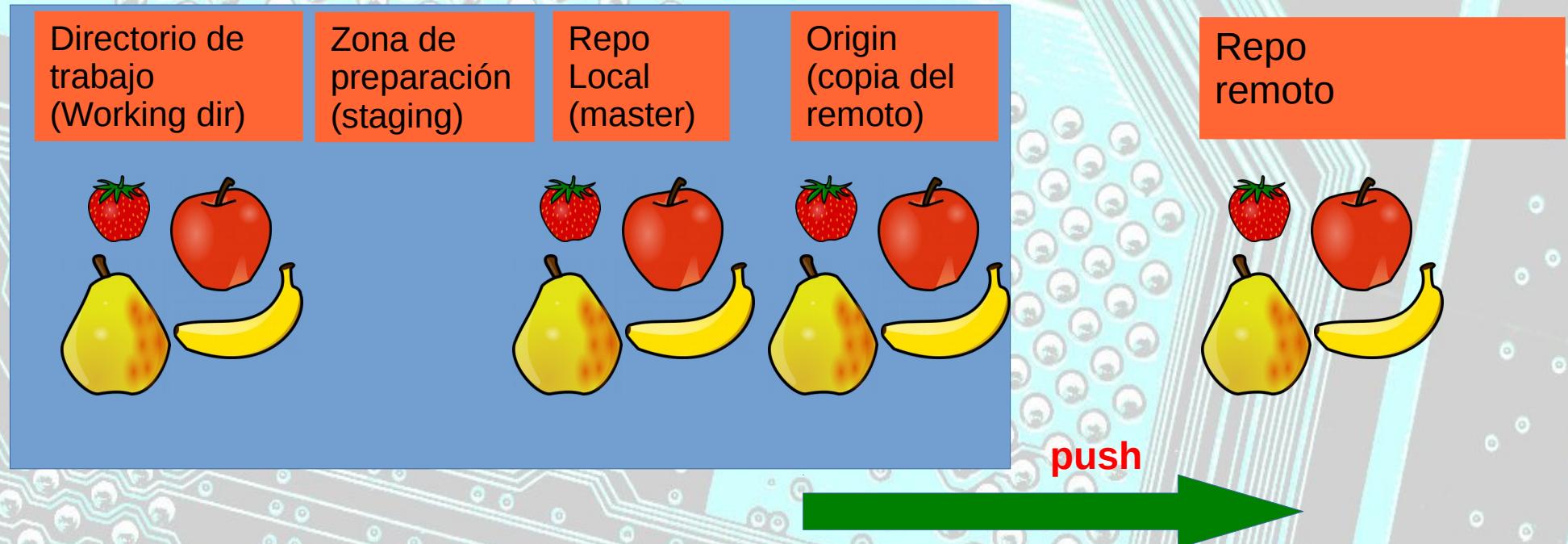


El fetch actualiza la copia local del remoto (origin)



El merge hace una mezcla del repo local con el remoto.

Explicando el push



El push envía el trabajo propio al repo remoto.

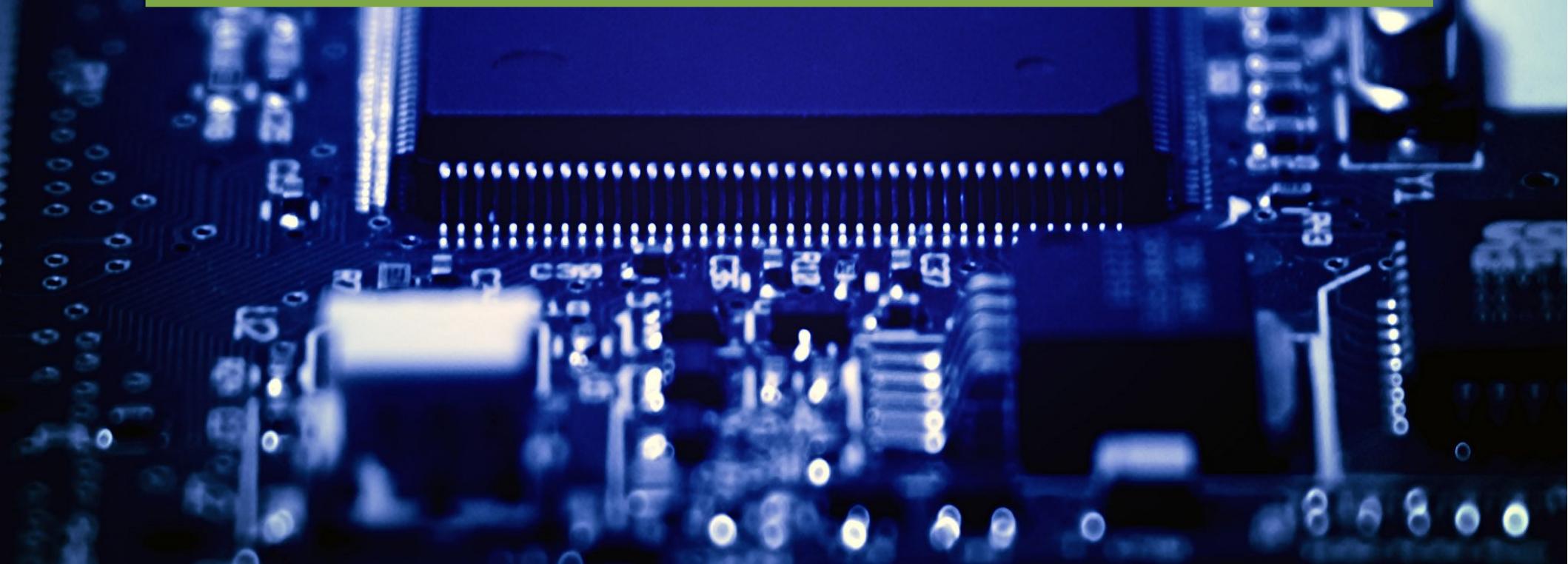
Con el push al repositorio del ejercicio 1, se da por concluido el mismo.

Nota: Las frutas pueden ser archivos o pueden ser distintos números de línea de un archivo de texto.

La explicación es conceptual y se trata de una simplificación. Se recomienda consultar la bibliografía para mayor detalle.

Ejercicio 2

Profundizando en el uso de KiCad



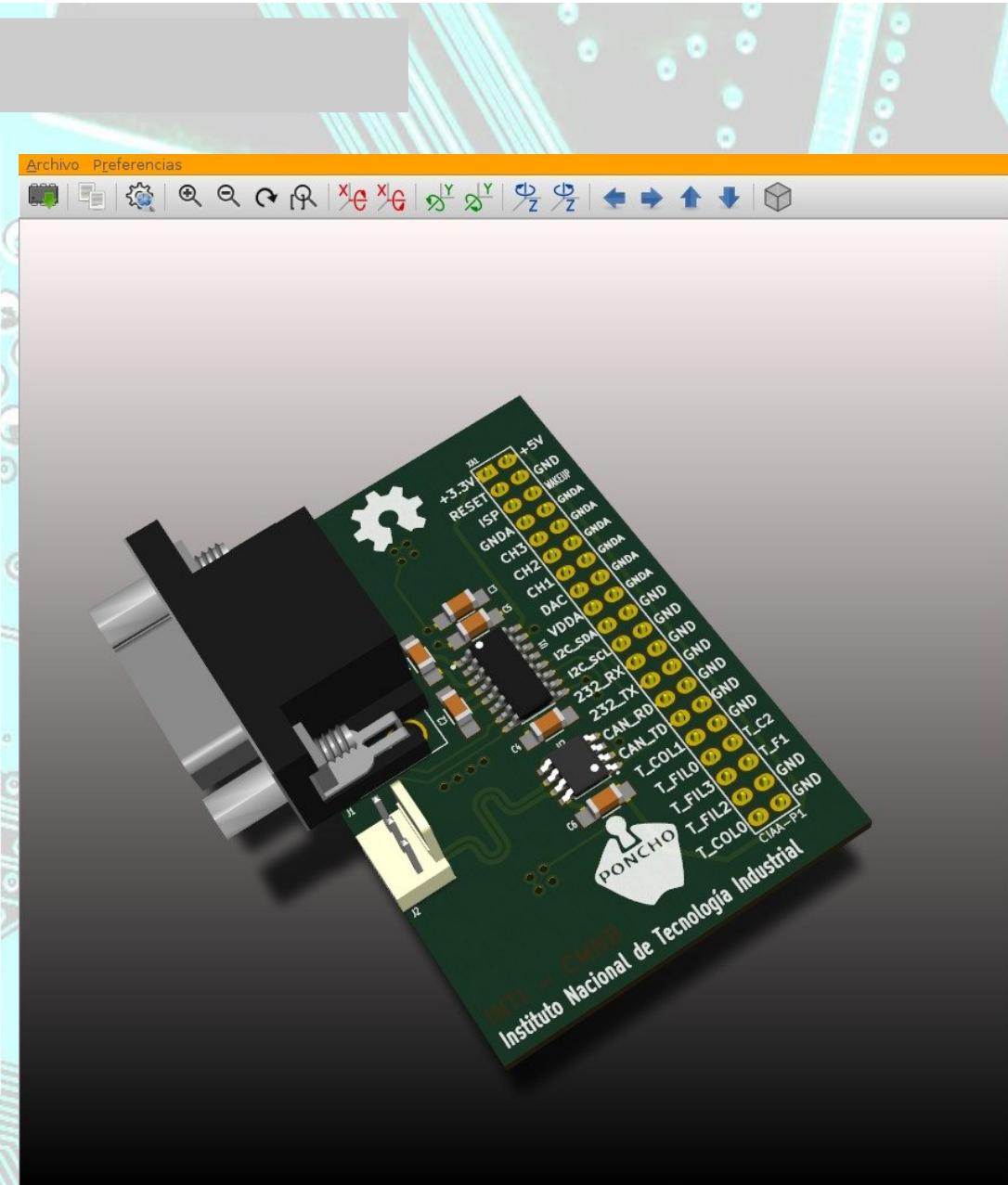
Ejercicio 2 - Inicio

En este ejercicio iremos viendo un poco más en profundidad las opciones de KiCad.

El ejercicio consiste en un circuito doble faz simple, pero se enfocará en tareas y métodos que son utilizados de igual forma en un diseño de mediana complejidad y en circuitos multicapa:

- Registro de la evolución del trabajo usando GIT.
- Uso de jerarquía de hojas.
- Uso de etiquetas y buses.
- Funcionamiento del ERC.
- Creación de símbolos y footprints de biblioteca.
- Mayor profundidad en las opciones del Pcbnew.
- Zonas de cobre.
- Vista 3D.

- 1) Crear un directorio “Ejercicio2”.
- 2) Abrir un proyecto nuevo.
- 3) Colocar información en el rótulo.
- 4) Guardar.



Ejercicio 2 - Ingreso a GIT - .gitignore

Ingresamos el proyecto nuevo a GIT.

El archivo .gitignore permite definir archivos que no queremos subir al repositorio y no queremos que GIT los reporte.

ARCHIVO .gitignore

```
*.bak  
*.kicad_pcb-bak  
_autosave-*.*.kicad_pcb
```

Comandos GIT

```
git add [Filename]  
git commit -m "msj"  
git status -s
```



- 1) Investigar los archivos generados.
- 2) Crear un archivo .gitignore.
- 3) Ingresar al repositorio solamente el .gitignore
- 4) Probar el reporte de estado de git.

Ejercicio 2 - Planificación de directorios

Nuestro diseño tendrá una estructura de directorios planificada previamente. No hay una regla estricta para esta organización, pero es muy conveniente tenerla y mantenerla en todos nuestros diseños.

Sugerencia:

Ejercicio2

libs: Biblioteca de símbolos de nuestro diseño.

ej2.pretty: Footprints.

ej2.3dshapes: Modelos 3D.

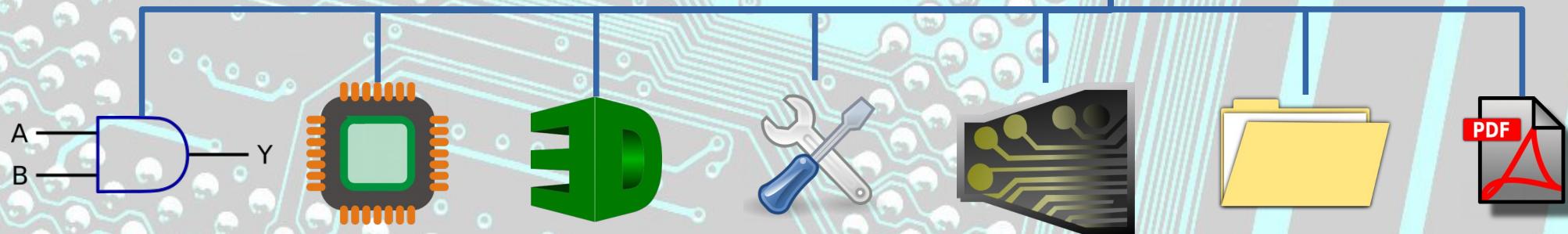
tools: Herramientas complementarias.

gerber: Archivos gerber.

doc: Documentación y licencia.

datasheets: Hojas de datos relevantes.

1) Crear la estructura de directorios.



Ejercicio 2

Editor de esquemáticos

Ejercicio 2 - Configuración de bibliotecas

Para este ejercicio intentaremos solamente utilizar la biblioteca de símbolos provista.

Tip: Es una buena práctica al finalizar el diseño, dejar solamente una biblioteca con todos los componentes utilizados dentro.

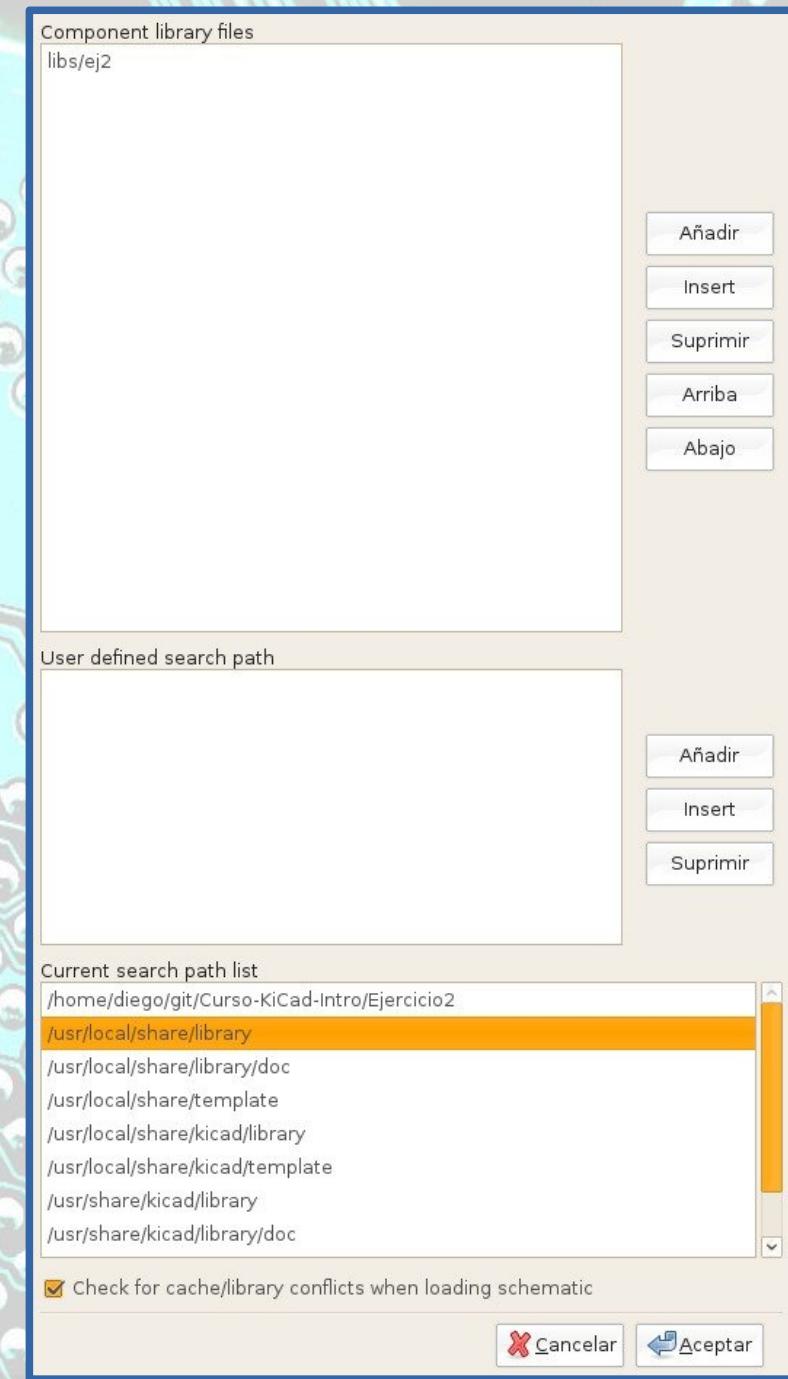
Tip: Al agregar una ruta (path) que está dentro de nuestro proyecto, es conveniente agregarla en forma relativa. Así no depende del lugar absoluto de trabajo.

Las bibliotecas provistas por KiCad pueden modificarse en el tiempo, pero nuestro diseño debe mantenerse inalterado.

Las bibliotecas de símbolos más usadas en KiCad:

- device (dispositivos de uso común)
- power (Etiquetas para alimentación)
- conn (Conectores)

1) Dejar disponible solo la biblioteca provista y con ruta relativa.



Eschema - Jerarquía de hojas

Diseño Jerárquico: Qué es?

Diseño con más de una hoja en el que las interconexiones entre hojas se realizan en una hoja principal o root.

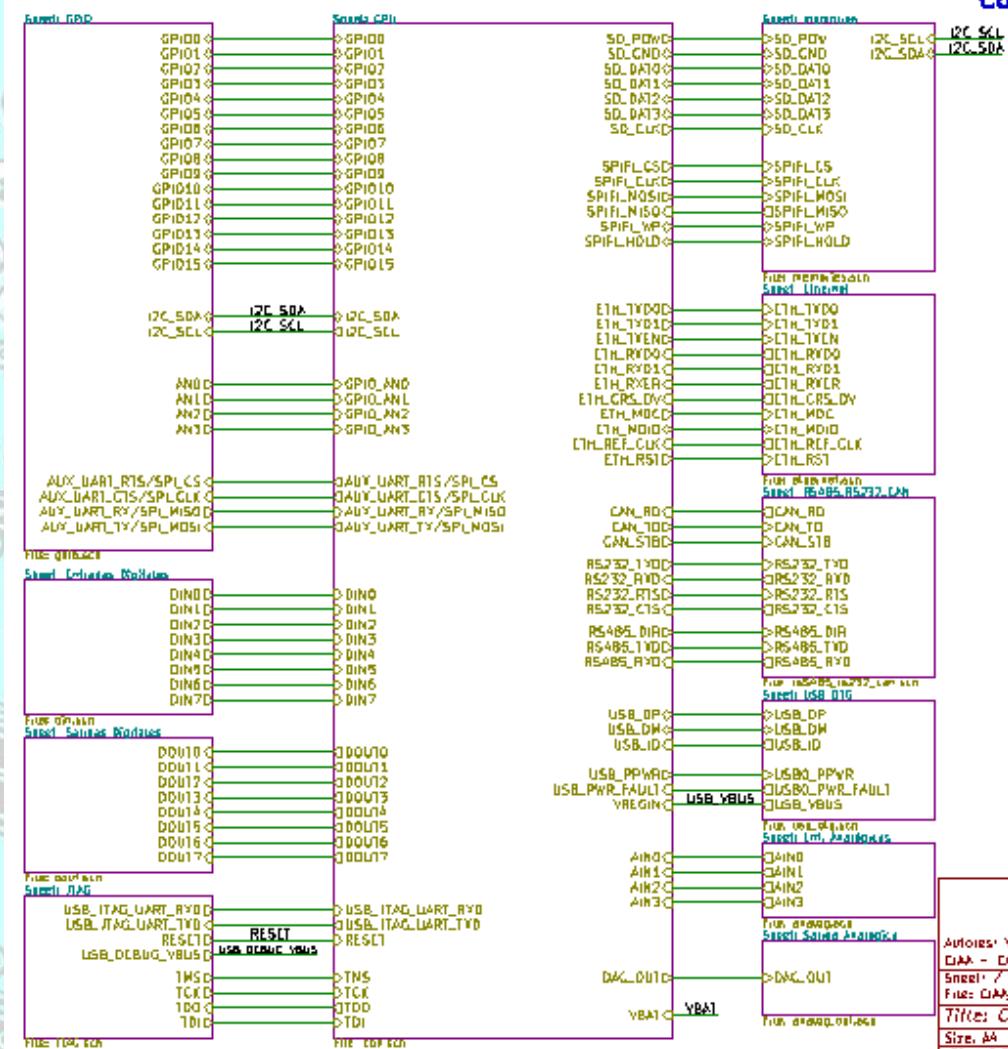
Cada hoja jerárquica se denomina sub-hoja.

Diseño Jerárquico: Cuándo utilizarlo?

- Cuando el diseño es muy complejo y no entra en una hoja.
 - Cuando se desea trabajar en grupo.

Diagrama Jerárquico: Ventajas

- Posibilidad de trabajar en grupo fácil y cómodamente.
 - Legibilidad y prolijidad.
 - Fácil seguimiento de las señales que van a varias hojas.



Tip: Configurar siempre las páginas en tamaño A4 para permitir una correcta impresión del esquemático.

Eschema - Jerarquía de hojas - Botones



Navegador de jerarquía

Diseño Jerárquico: Flujo de trabajo

- 1) Agregar en la hoja principal las sub-hojas.
- 2) Editar cada sub-hoja creando los labels jerárquicos correspondientes.
- 3) Importar los pines jerárquicos en cada sub-hoja de la hoja principal.
- 4) Realizar las interconexiones entre hojas.

Navegación entre hojas

- Para entrar en una hoja hacer DOBLE CLICK (IZQ.) sobre la misma.
- Para salir y volver a la hoja principal hacer DOBLE CLICK DERECHO.
- Para ir a cualquier hoja usar el navegador de la barra superior.



Subir/bajar jerarquía

Tips

- Evitar usar labels globales ya que son difíciles de rastrear en el esquemático.
- Ubicar los labels jerárquicos en los bordes de las páginas para encontrarlos rápidamente.

2

- Crear label jerárquico en sub-hoja
- Crear nueva hoja jerárquica en hoja principal
- Importar pin jerárquico en la hoja jerárquica
- Crear pin jerárquico en hoja jerárquica

Este último botón no conviene usarlo, no es práctico!



Ejercicio 2 - Jerarquía de hojas

Realizaremos una jerarquía de hojas con:

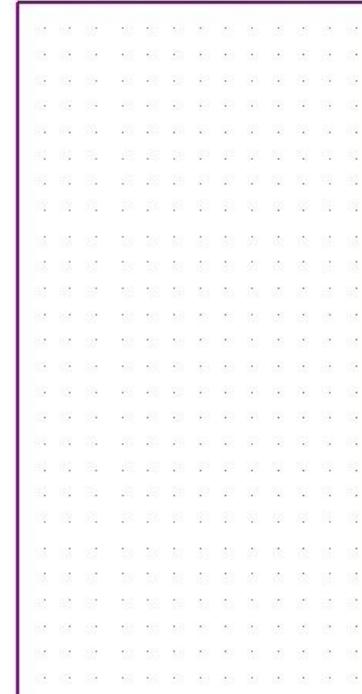
- Una página principal
- Una sub-hoja para el conector y el CAN.
- Una sub-hoja para el driver RS-232.

Sheet: Conector



File: Conecotor.sch

Sheet: RS-232



File: Rs232.sch

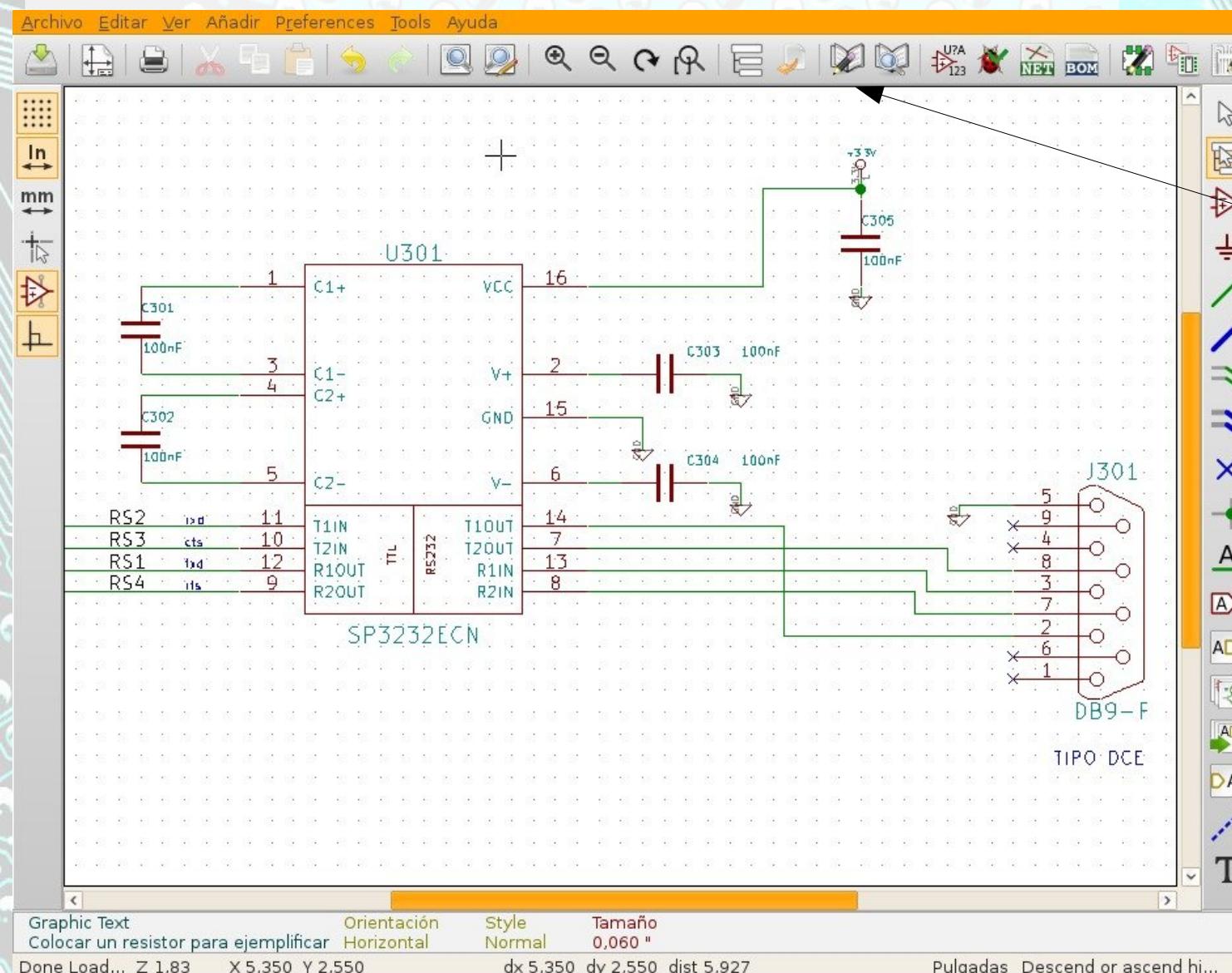
- 1) Crear dos hojas jerárquicas.
- 2) Crear los pines jerárquicos en la hoja del conector y CAN.
- 3) Importar los pines jerárquicos en la hoja principal.

Ejercicio 2

Editor de símbolos

Ejercicio 2 - Creación de un símbolo

En la hoja RS-232 realizaremos el siguiente circuito.
El SP3232ECN lo crearemos para incursionar en la
creación de símbolos.



- 1)Abrir el editor de bibliotecas de símbolos.
- 2)Ingresar los datos solicitados (ver siguiente slide).

Ejercicio 2 - Creación de un símbolo

Actualmente el proyecto KiCad tiene pautas para la creación de símbolos. Estas pautas nos ayudan a ser organizados, consistentes y estar en sintonía con el software:
<https://github.com/KiCad/kicad-library/wiki/Kicad-Library-Convention>

General Rules

- Writing uses C-style naming with the first letter of each word being capitalized. Ex: "**Socket_Strip_Straight_2x06**"
- Every acronym has all of its letters capitalized.
- Manufacturer name is capitalized as usual. Ex: NEC, Microchip.
- When dimensions are used in part name, they are in millimeters, decimal places separated by a dot, and unit is not mentioned. Ex: "C_Rect_L13_W4_P10"
- **Filename is the same as the part name.**
- The order of elements in names must be the same as the enumerations presented in this document.
- Reference fields are prefilled with the reference designator of the part (IEEE 315-1975).

Symbol Library Names (.lib files)

- Manufacturer.
- Category or family of parts. ex: "Capacitors", "Spartan6", etc.

General Rules for Symbols

- **Using a 100mil grid**, pin ends and origin must lie on grid nodes (IEC-60617).
- For black-box symbols, pins have a length of 100mils. Large pin numbers can be accommodated by incrementing the width in steps of 50mils.
- **Origin is placed in the middle of symbol.**
- Black-box components **group pins logically**, for example by function set, and ports in counter-clockwise position.
- Whenever possible, **inputs are on the left and outputs are on the right**.
- **Field text uses a common size of 50mils.**
- **The Value field is prefilled with the object name.**
- **Description and keywords properties contain the relevant information.**

Ejercicio 2 - Creación de un símbolo

Para cada pin de conexión hay que seleccionar una propiedad:

INPUT,
PASSIVE,
OPEN COLLECTOR,

OUTPUT,
NO ESPECIFICADO,
OPEN EMITTER,

BIDIRECCIONAL,
POWER INPUT,
NO CONNECTION

TRI-STATE,
POWER OUTPUT,

ENTRADA

ENTRADA

ENTRADA

ENTRADA

ENTRADA
ENTRADA
SALIDA
SALIDA E

1

3
4

5

11

10

12

9

C1+

C1-
C2+

C2-

T1IN

T2IN

R1OUT

R2OUT

U?

VCC

GND

V+

V-

16

15

2

6

14

7

13

8

POWER INPUT

POWER INPUT

POWER OUTPUT

POWER OUTPUT

SALIDA
SALIDA
ENTRADA
ENTRADA

SP3232

- 1) Realizar el componente.
- 2) Guardarlo en el directorio local de bibliotecas.
- 3) Ingresarlo al esquemático.

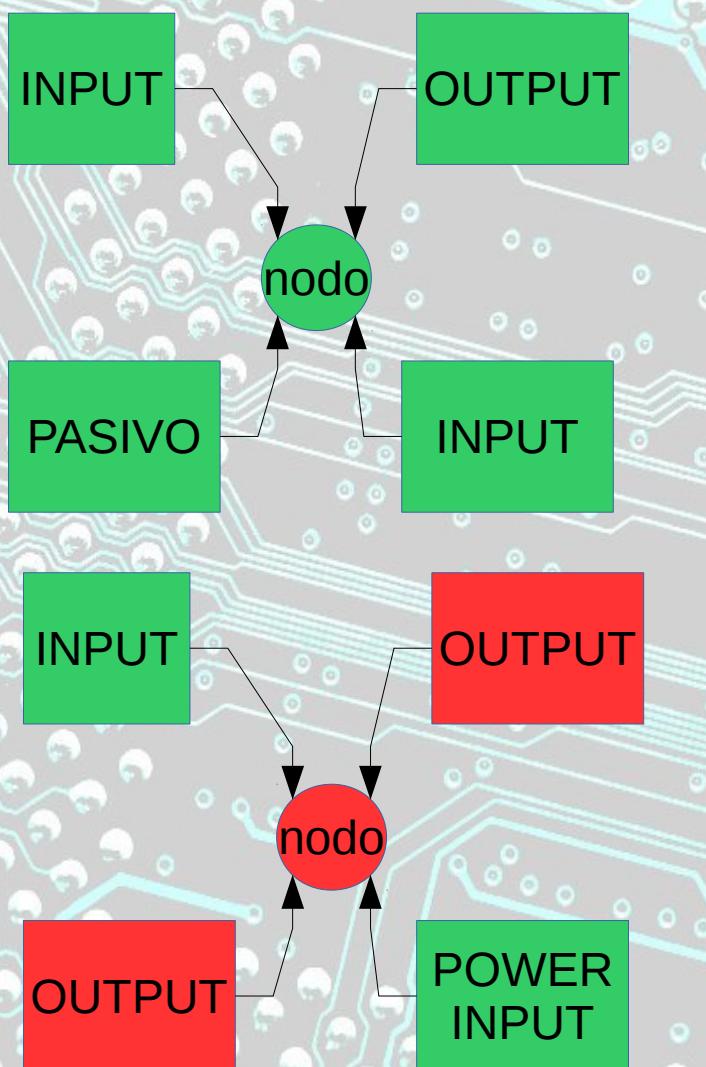
Ejercicio 2

Editor de esquemáticos

Ejercicio 2 - ERC

Las propiedades de los pines son utilizadas por el ERC.

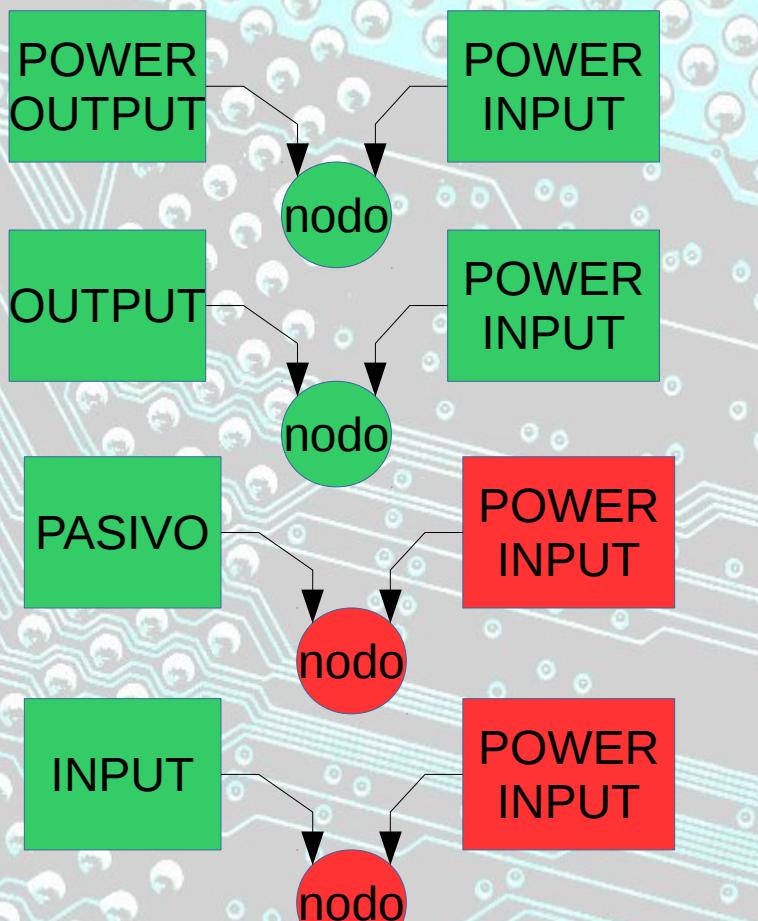
El ERC (Electrical Rules Checker) de KiCad NO SIMULA, solamente aplica esta matriz en cada nodo, considerando las propiedades de los pines conectados al mismo.



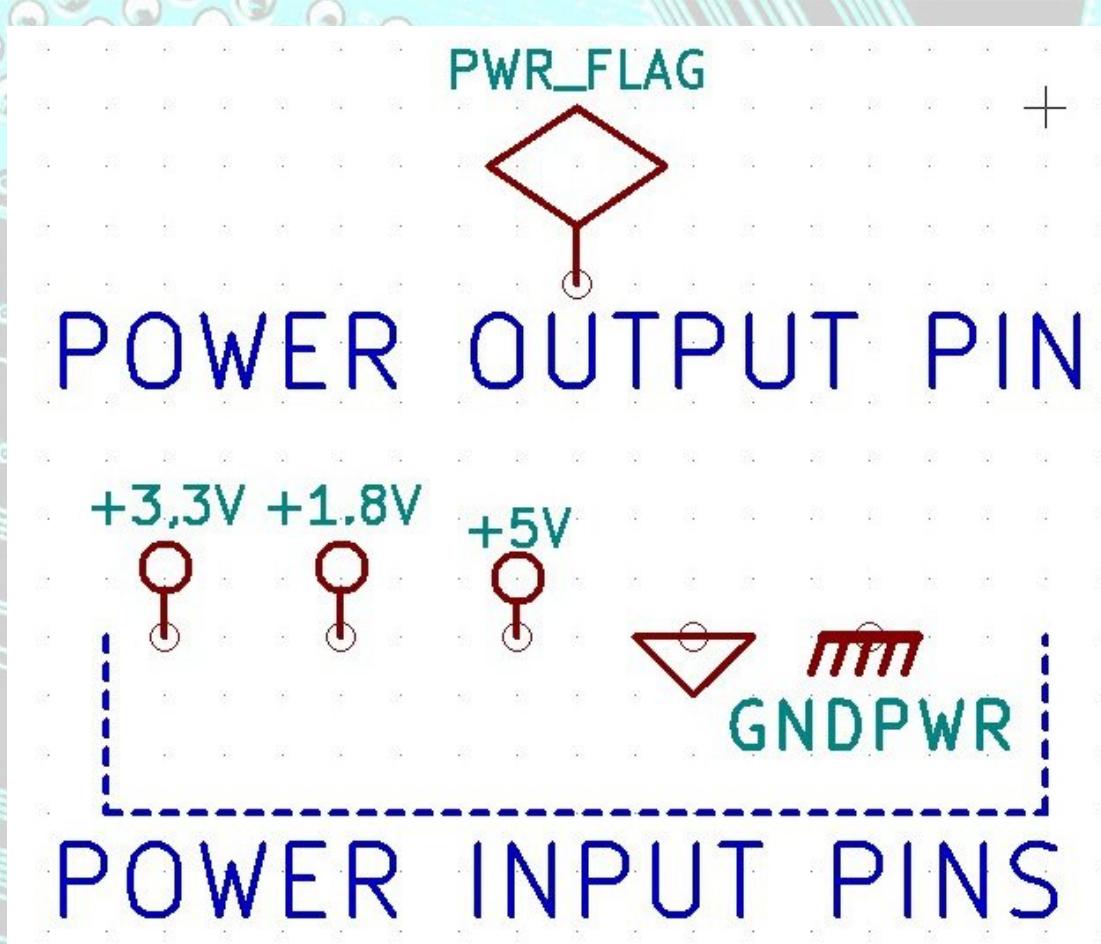
ERC Opciones		Defecto							
	Input Pin.....								
	Output Pin.....								
	Bidirectional Pin..								
	Tri-State Pin.....								
	Passive Pin.....								
	Unspecified Pin....								
	Power Input Pin...								
	Power Output Pin...								
	Open Collector.....								
	Open Emitter.....								
	No Connection.....								

Ejercicio 2 - ERC - Power

KiCad verifica que un pin de entrada de alimentación (power input) esté alimentado por alguno de los componentes del nodo.



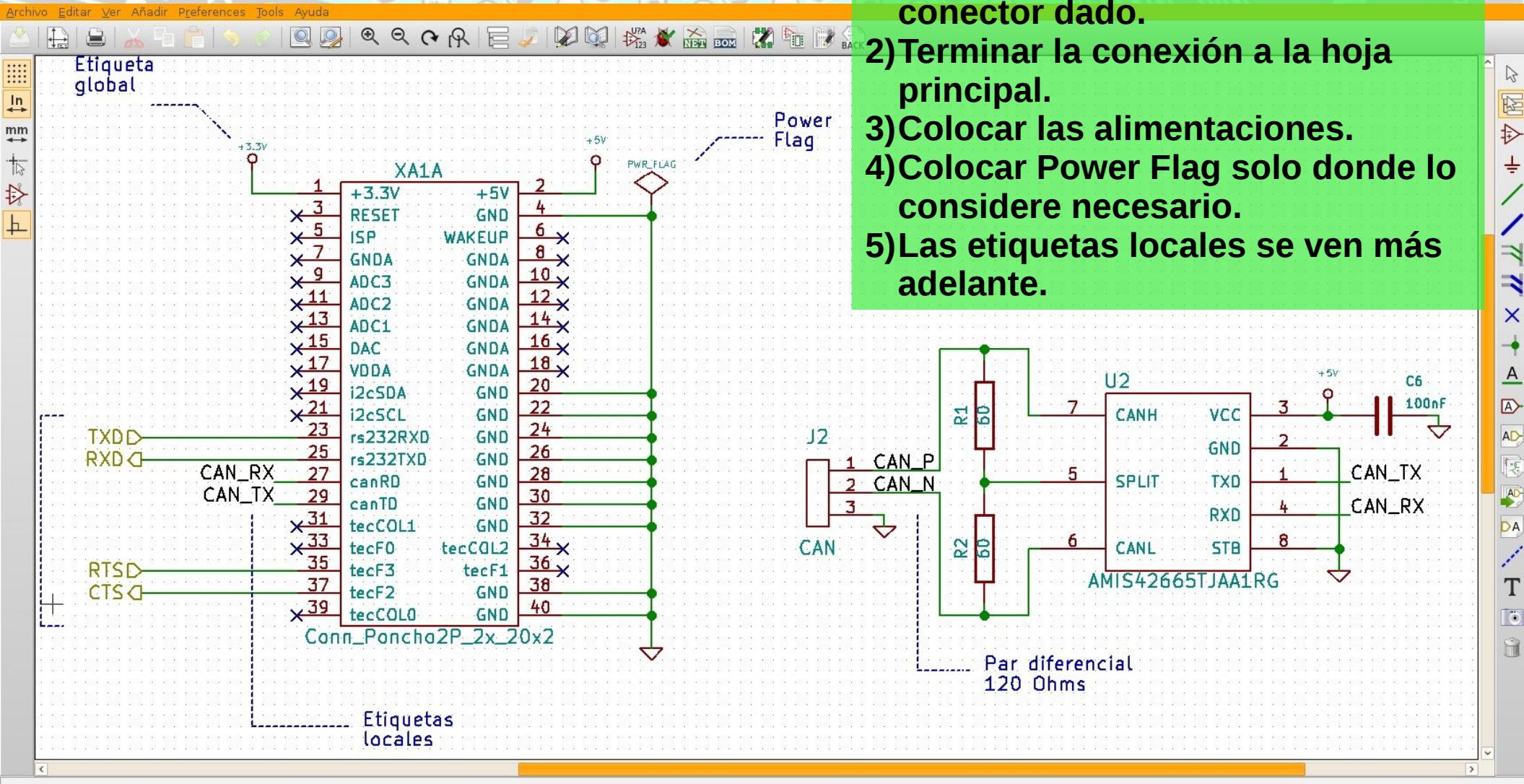
Además el ERC verifica que todos los componentes tengan una referencia válida, elementos no conectados, etc.



El POWER FLAG se utiliza para evitar el error de ERC cuando la alimentación proviene de un conector o viene de un componente con propiedad pasiva (fusible, diodo, resistor, etc.).

Ejercicio 2 - Hoja conector

Completamos la hoja del conector, prestando atención a los power flags y las alimentaciones. El conector es un componente multipartido (se ve en el día 3).



Eschema - Etiqueta local

La etiqueta local permite realizar una conexión dentro de una hoja, sin que una línea realice todo el trayecto de conexión.

La etiqueta debe aplicarse sobre un pin o sobre un tramo de cable.

El punto de conexión es un cuadrado debajo de la primera letra de la etiqueta, y debe estar en contacto con el pin o con el tramo de cable.

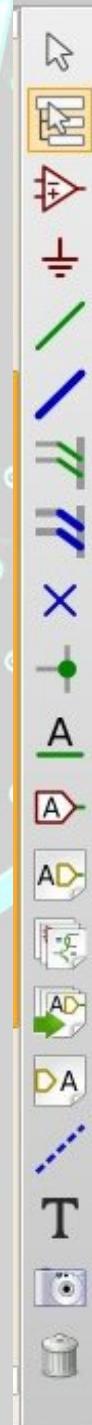
TIP: Las etiquetas pueden hacer más claro un circuito esquemático al eliminar líneas que cruzan por la hoja. Pero su abuso conduce a esquemáticos sin interconexiones visibles que dificultan la interpretación (como leer un archivo netlist).

Las etiquetas también sirven para definir buses.

RX

Etiqueta

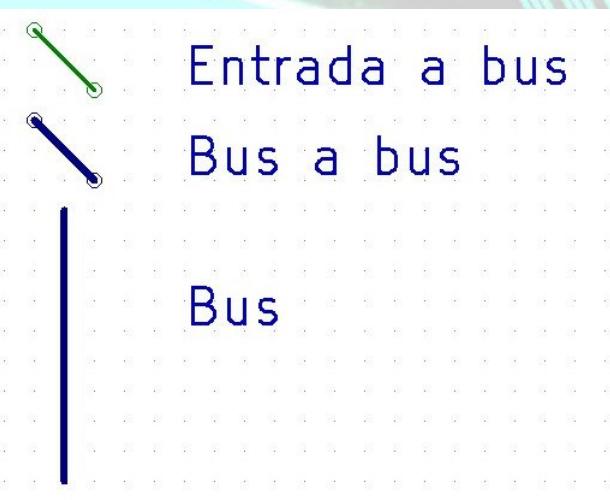
Etiqueta local



Eschema - Buses

Los símbolos de entrada a bus y de bus a bus son solo dibujos para prolijidad en la hoja. Son estéticos y no cumplen función ni interconectan elementos.

La línea de bus también es estética y no cumple función de interconexión, salvo que se utilice una etiqueta con nomenclatura de bus (siguiente slide) ,

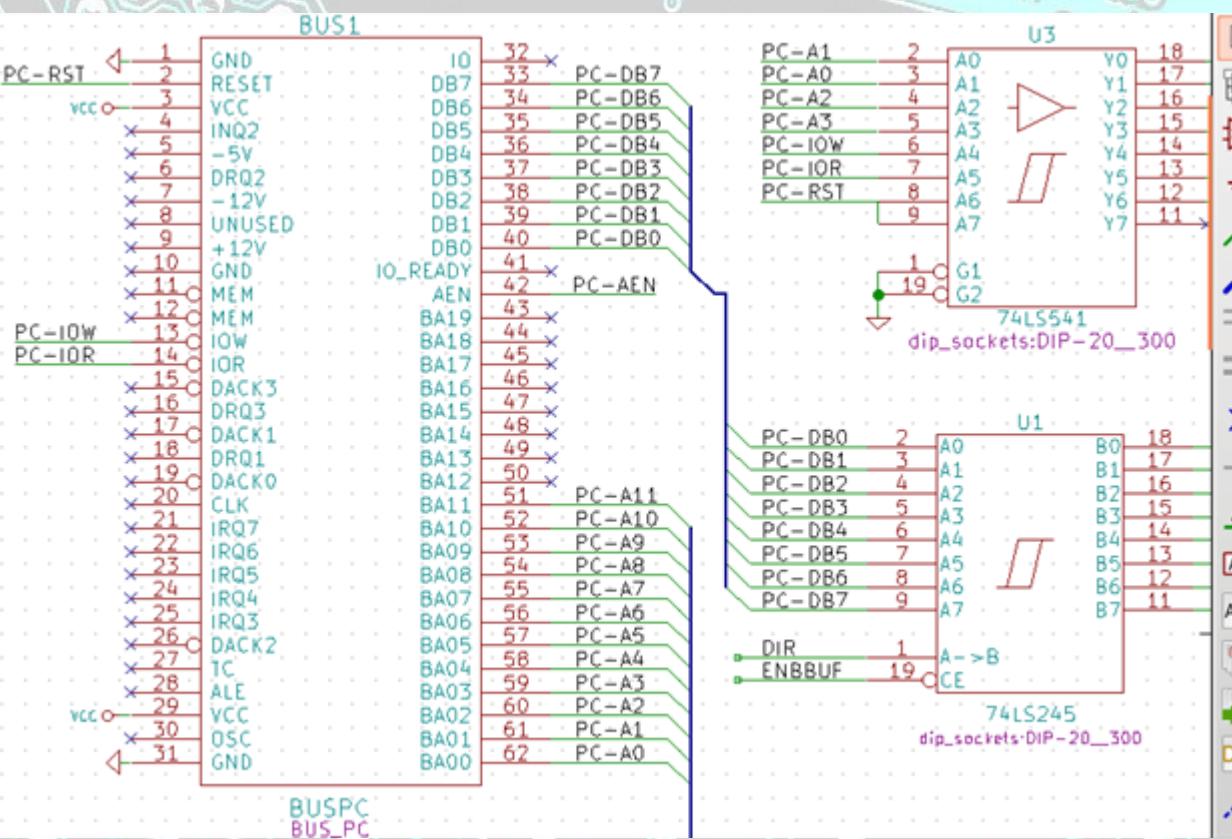


Línea de bus
Entrada a bus
Entrada de bus a bus

Etiqueta local

En el circuito de la izquierda (PC-DBx) la interconexión real se hace con etiquetas locales. Las entradas de bus y los tramos de bus podrían borrarse y las interconexiones seguirían correctas.

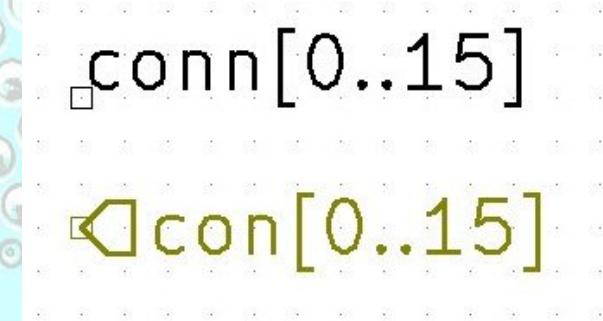
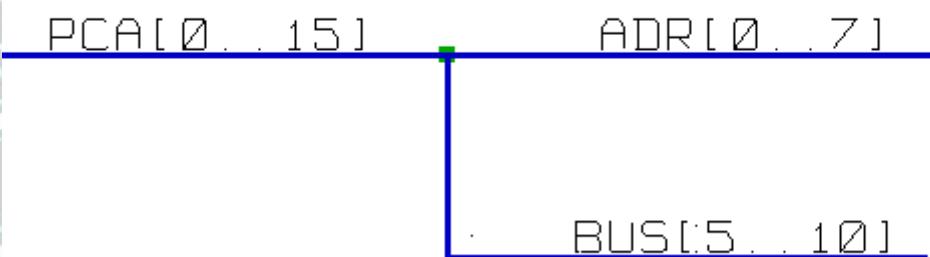
Tip: Para colocar las etiquetas de un BUS, colocar solo la primera y luego la tecla [INS] (Repeat las item). En las opciones de edición se configuran los parámetros de repetición.



Eschema - Buses

Nomenclatura de BUS

- Se utiliza un nombre de base (prefijo) y entre corchetes el rango de señales del bus [n..N], siendo el primer número de señal y N el último.
- Esta nomenclatura se puede utilizar en labels de conexión jerárquicos o sobre líneas de buses.
- Para mezclar buses, kicad simplemente conecta números iguales entre sí.
- Para referenciar cada señal del bus se utiliza la herramienta de etiqueta local.



CONEXIONES

- PCA0-ADR0
- PCA1-ADR1
- PCA2-ADR2
- PCA3-ADR3
- PCA4-ADR4
- PCA5-ADR5-BUS5
- PCA6-ADR6-BUS6
- PCA7-ADR7-BUS7
- PCA8-BUS8
- PCA9-BUS9
- PCA10-BUS10
- PCA11
- PCA12
- PCA13
- PCA14
- PCA15

Línea de bus

Etiqueta local

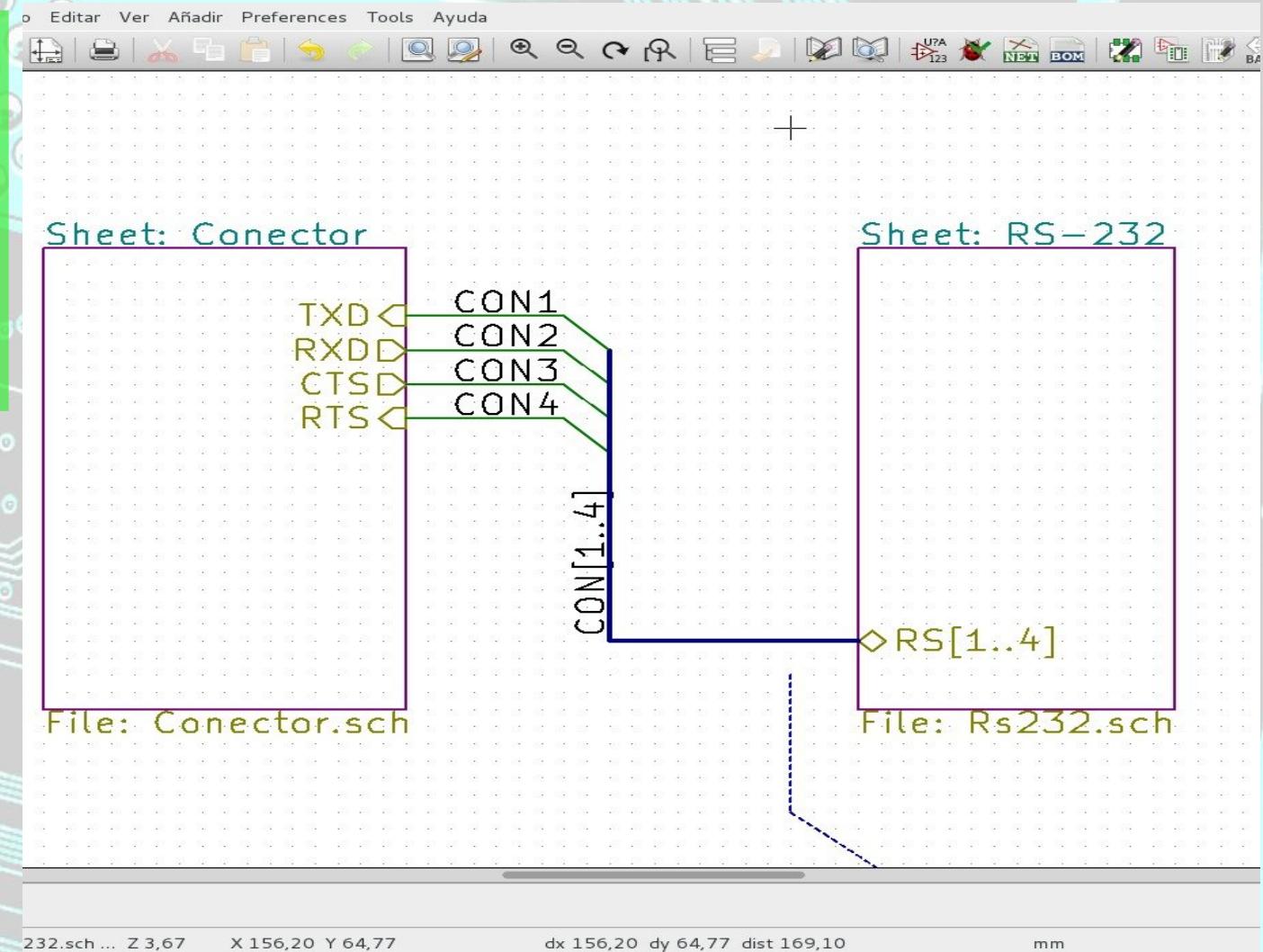
label en sub-hoja

Label en hoja

Eschema - Jerarquía de hojas y buses - Ejercicio 2

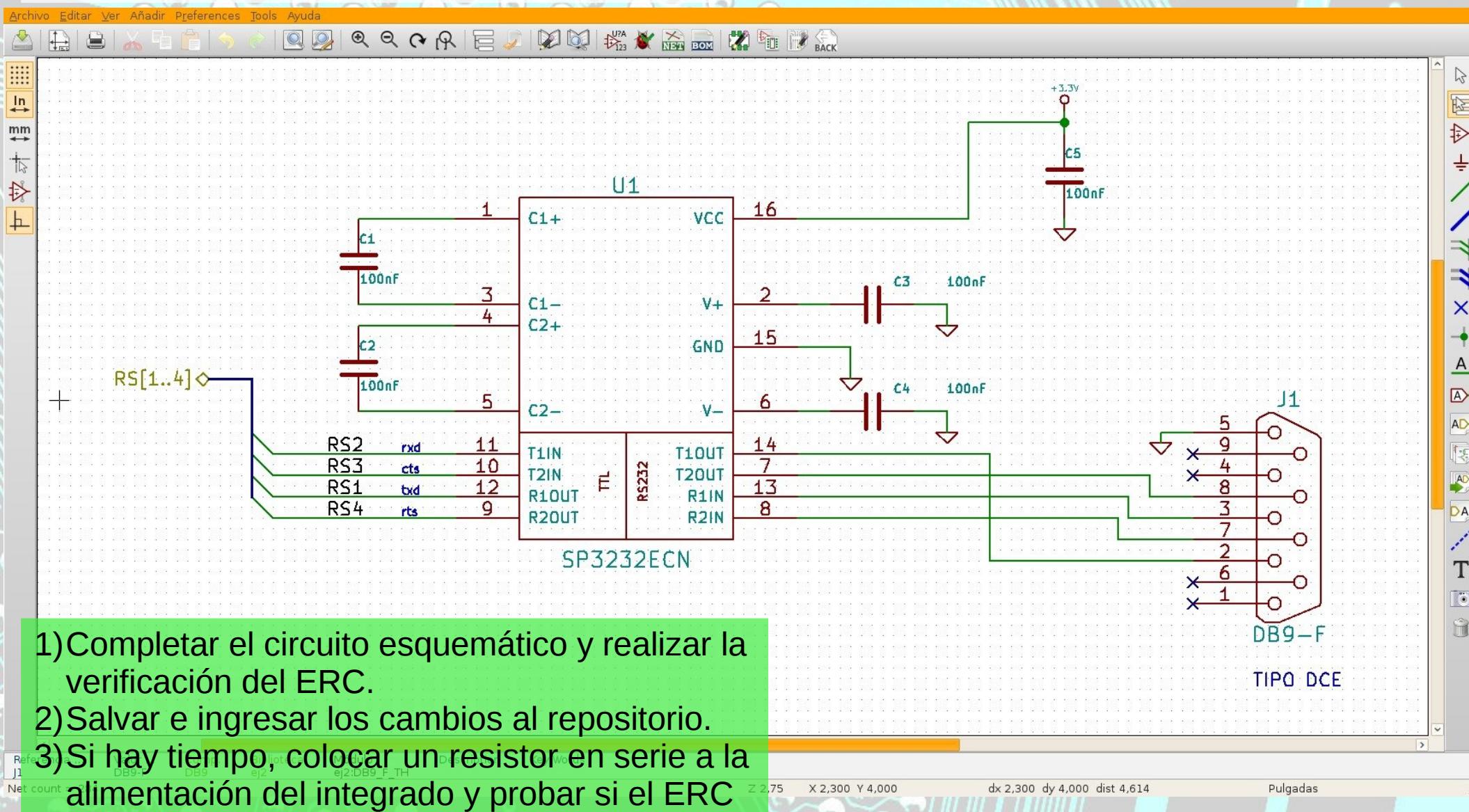
1) Utilizar un bus para interconectar ambas hojas.

2) En la hoja RS-232 ingresar con el bus para luego descomponerlo dentro de la sub-hoja.



Ejercicio 2 - Interconexiones y verificación general

Terminar el conexionado completo, pasar el ERC e ingresar a GIT.



Eschema - Campos de Información

Campos de información: Por qué se usan?

- Correcta identificación de cada componente desde el momento que el desarrollador lo agrega al esquemático.
- Uniformidad de la descripción de todos los componentes.
- Obtención de una lista de componentes completa y ordenada para realizar la compra de materiales.

Campos de información: Cómo se usan?

- 1) Acordar con todos los desarrolladores los campos que se utilizarán y cómo se completarán.
- 2) Chequear que todos tengan los campos escritos exactamente igual, no es lo mismo NRO. DE PARTE que NÚMERO DE PARTE!
- 3) Completar los campos a medida que se van agregando los componentes al esquemático, no dejar para el final.
- 4) No cambiar ni agregar campos durante el transcurso del proyecto para no generar conflictos.

Schematic Editor Options		
General Options		Template Field Names
Field Name	Default Value	Visible
Descripción		Hidden
Fabricante		Hidden
Nro. parte		Hidden
Path datasheet		Hidden
Digikey/Mouser		Hidden
Footprint estandar		Hidden

Preferences → Schematic Editor Options → Template Field Names

Field Settings

Name: Footprint estandar

Default Value:

Visible

Add

Component Properties

Fields

Name	Value
Reference	R206
Value	10K
Footprint	r_0603
Datasheet	~
Descripción	RES 4.7K OHM 1/10W
Fabricante	Yageo
Nro. parte	RC0603JR-0710KL
Path datasheet	~
Digikey/Mouser	311-10KGRCT-ND
Footprint estandar	0603

Horiz. Justify:

- Left
- Center
- Right

Vert. Justify:

- Bottom
- Center
- Top

Visibility:

- Show
- Rotate

Style:

- Normal
- Italic
- Bold
- Bold Italic

Field Name: Reference

Field Value: R206

Tip: Es posible editar campos desde los archivos de texto!

Ejercicio 2 - Campos de información

- 1) Crear el campo de información Digikey/Mouser.
- 2) Completar el campo Digikey/Mouser y el campo datasheet con la url de la hoja de datos (usar la que da Digikey).

Reference(s)	Value	LibPart	Footprint	Digikey/Mouser
C1, C2, C3, C4, C5, C6	100nF	ej2:C	ej2:C_0603_HandSoldering	311-1179-1-ND
J1	DB9-F	ej2:DB9	ej2:DB9_F_TH	S9554-ND
J2	CAN	ej2:CONN_3	ej2:CON_PALETA_3	A19470-ND
R1, R2	60	ej2:R	ej2:R_1206_HandSoldering	Y149660R0000A0R-ND
U1	SP3232ECN	ej2:SP3232	ej2:SP3232ECN-SOIC16N	1016-1803-1-ND
U2	AMIS42665TJAA 1RG	ej2:AMIS4266 5TJAA1RG	ej2:SOIC-8	766-1006-1-ND
XA1	Conn_Poncho2P_ 2x_20x2	ej2:Conn_Pon cho2P_2x_20 x2	ej2:Conn_Poncho_Derecha	952-1387-ND

BOM - Listado de materiales

Actualmente kicad utiliza scripts phyton para generar distintos tipos de lista de materiales.

En el directorio tools copiarse los siguientes archivos:

kicad_netlist_reader.py (1)
bom_csv_grouped_by_value.py (2)

- (1) Script base que obtiene la información y genera un xml.
- (2) Script que genera el BOM deseado

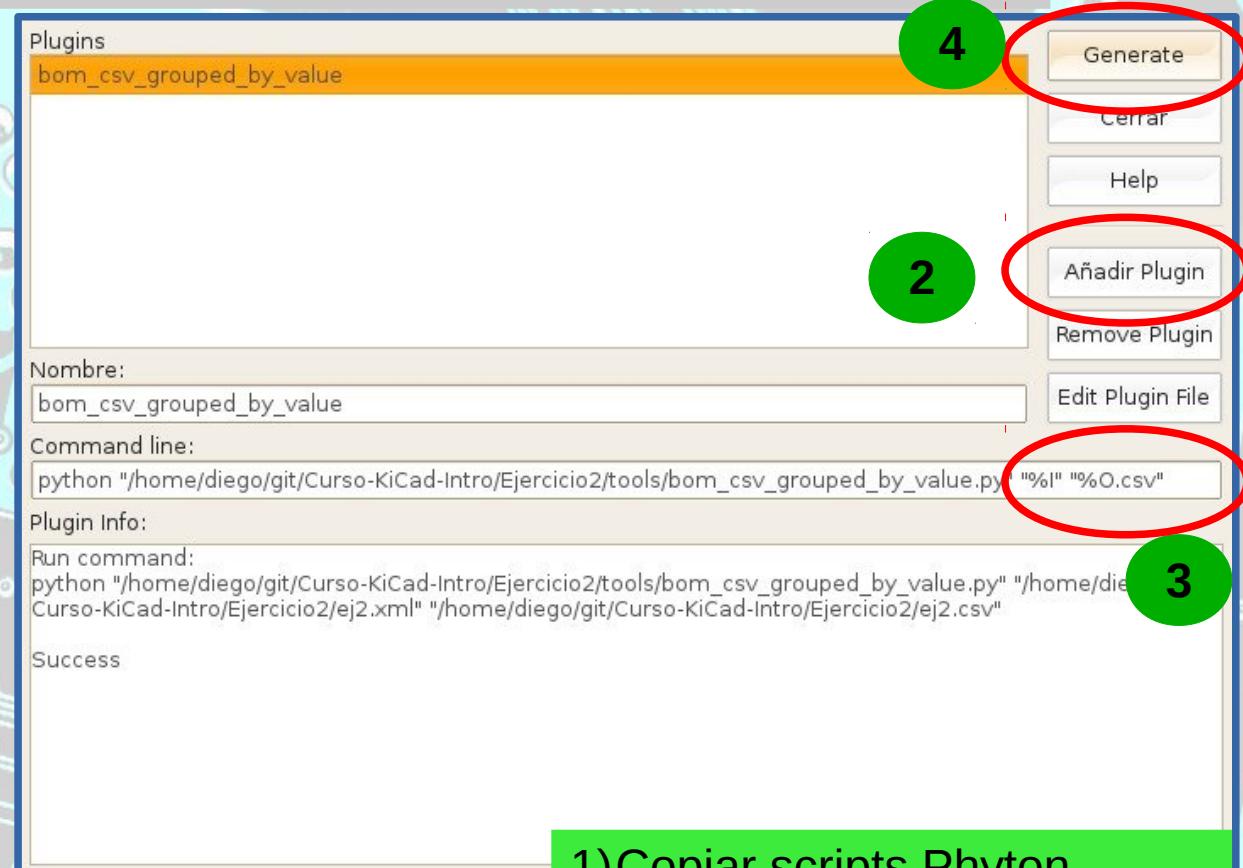
Luego añadir el plugin deseado.

Se puede editar la línea de comandos que usará Kicad. Por ejemplo cambiar a "%O.csv"

Generar el BOM y abrirlo lo una planilla de cálculo.

TIP: Hay más scripts en:

<https://github.com/KiCad/kicad-source-mirror/tree/master/scripts/bom-in-python>



- 1) Copiar scripts Phyton.
- 2) Agregar el plugin.
- 3) Editar la línea de comandos.
- 4) Generar el BOM.
- 5) Abrirlo con un software de planilla de cálculo.

Kicad generará un bytecode de Phyton:
kicad_netlist_reader.pyc

Alerta: Por el momento KiCad guarda las rutas absolutas de estos scripts.

Ejercicio 2 - Asociación de footprints

Con las bibliotecas de footprints dadas para el ejercicio realizar la asociación de todos los footprints. Ver tabla anterior.

- 1)Copiar los footprints dados al directorio ej2.pretty
- 2)Usando el Wizard ingresar el directorio como en el ejercicio 1 (solo para este proyecto).
- 3)Usando Cvpcb o mediante el campo “Módulo” de cada símbolo realizar la asociación de footprints.
- 4)Generar el netlist.
- 5)Grabar el esquemático.



Sobre la asociación de footprints

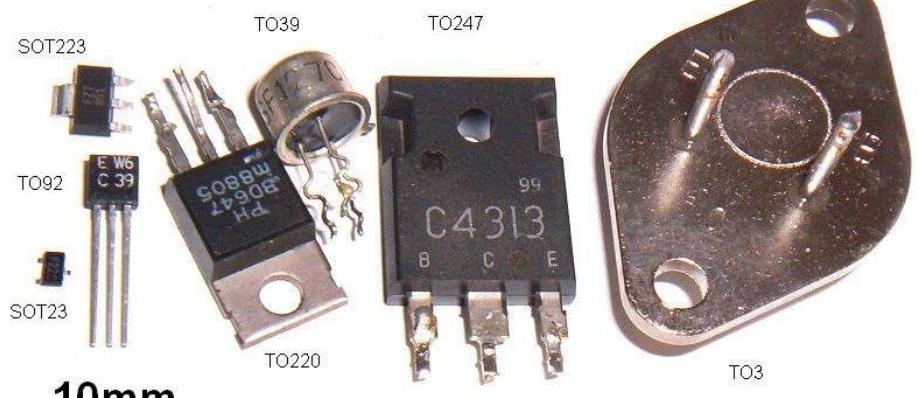
Se debe tener especial cuidado al asociar un símbolo con un footprint.
La problemática es la siguiente:



1) Existe una gran cantidad de símbolos genéricos que se pueden utilizar para una gran variedad de componentes diferentes (como por ejemplo el símbolo de un transistor NPN).



2) Existe una gran cantidad de footprints estandarizados como por ejemplo TO-92, TO-220, TO-3, TO-18, SOT-23. El estándar del encapsulado define la forma y las dimensiones, pero no la función de cada pin o el tipo de dispositivo.



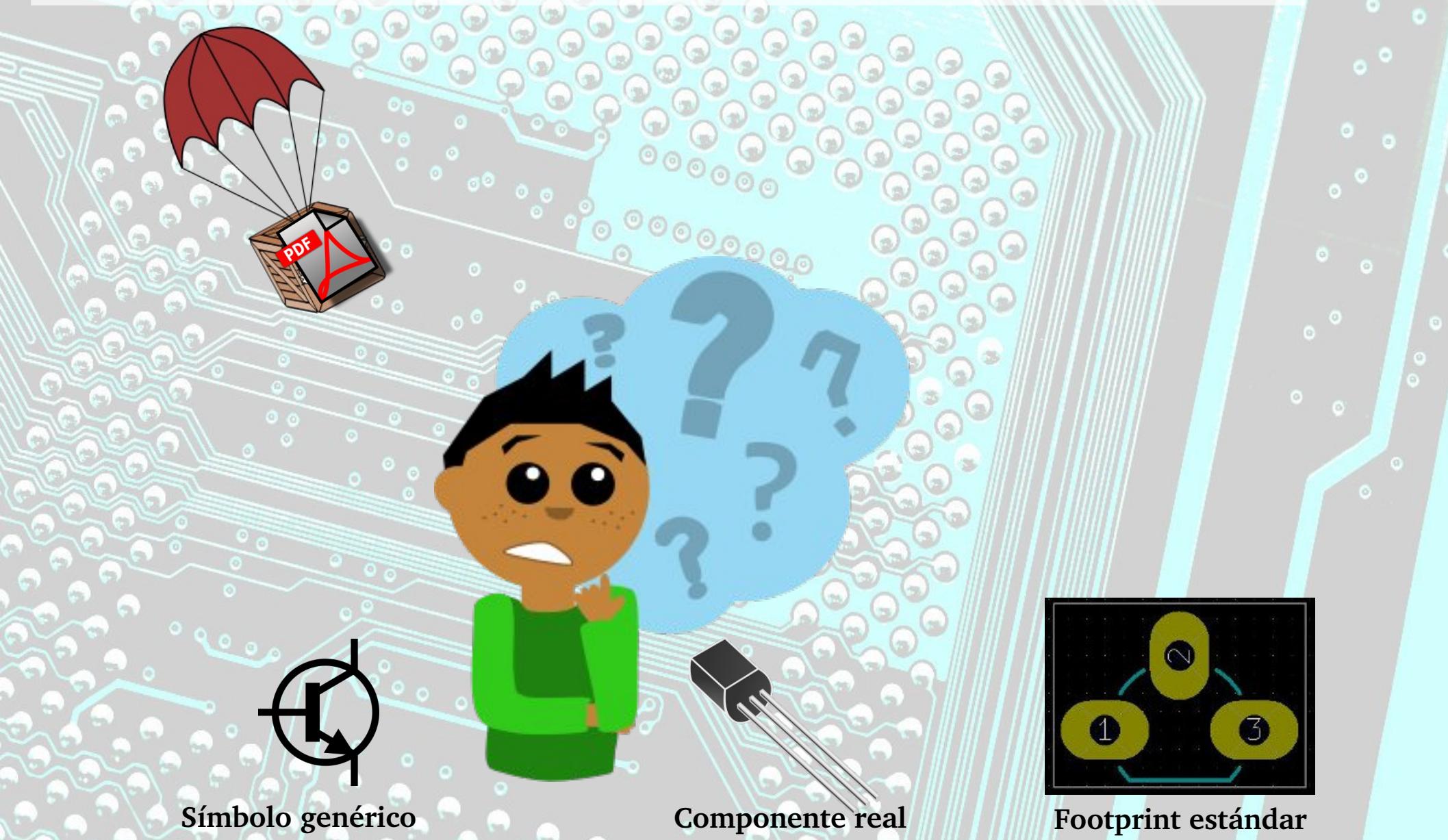
3) Lo que termina definiendo la relación entre la funcionalidad del pin y su correspondiente pin en el encapsulado es la hoja de datos.



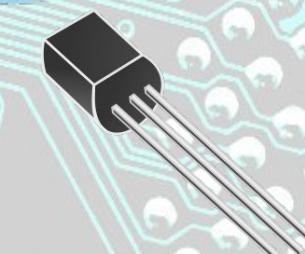
Se presenta al transistor NPN como ejemplo. La problemática es general de las bibliotecas, que evitan considerar la infinidad de componentes electrónicos existentes cuando no es necesario.

Sobre la asociación de footprints

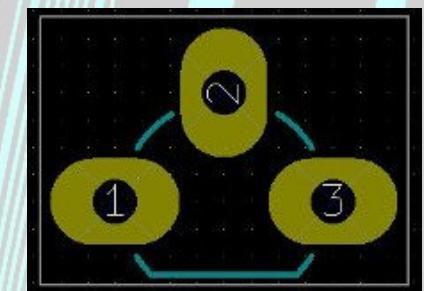
Es necesario consultar la hoja de datos, analizar y realizar los ajustes necesarios:



Símbolo genérico



Componente real



Footprint estándar

Sobre la asociación de footprints

El ejemplo del transistor NPN en KiCad.
Supongamos tenemos un BC548:

1) El símbolo existente está en la biblioteca "devices" y se llama NPN. Solo se debe editar el campo valor para reflejar nuestro modelo particular de transistor.

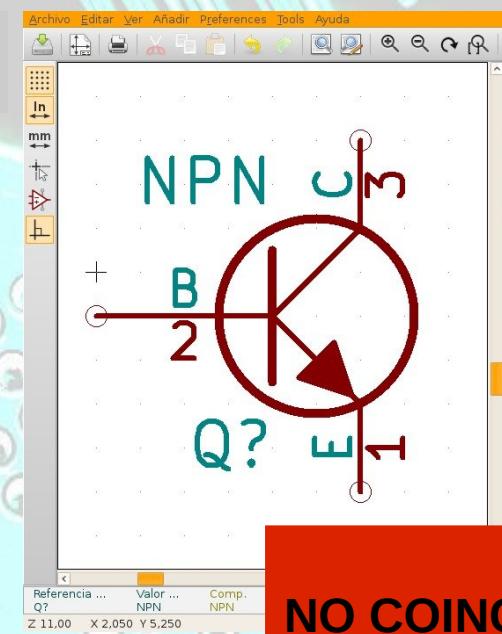
2) Seleccionamos de la biblioteca de footprints (TO_SOT-Packages-THT) alguno de los modelos TO-92 existentes.

3) La asociación que realiza KiCad es simple:

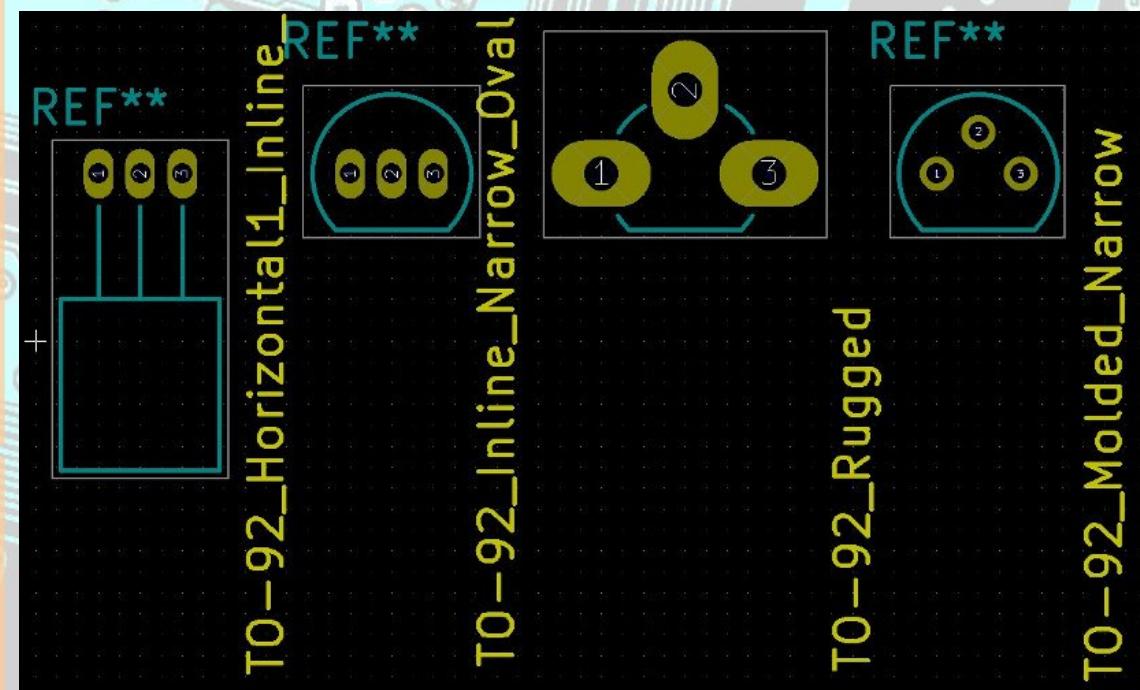
- 1- emisor - 1 footprint
- 2- base - 2 footprint
- 3- colector – 3 footprint

Verificamos si coincide la numeración en el símbolo con la del footprint, mirando la hoja de datos.

4) Si no coincide, editamos con el editor de módulos, cambiamos la numeración, guardamos el módulo nuevo en el directorio de footprints y aplicamos el cambio en el PCB. Finalmente actualizar la conexiones (diálogo de netlist).



NO COINCIDE



Ejercicio 2 - Seguimiento con GIT

Ingresamos a GIT nuestros avances, con la versión casi terminada de nuestro esquemático.

- 1) Verificar ERC.
- 2) Generar netlist.
- 3) Generar BOM.
- 4) Revisar BOM.
- 5) Agregar los archivos nuevos y los modificados a GIT.

Tip: En general los archivos que pueden generarse a partir de otros **NO** deben subirse a GIT. Sin embargo a veces generar estos archivos puede ser complicado. En esos casos se puede subirlos solamente a fines de documentar *releases* importantes.

Ejemplo: El pdf del esquemático resulta redundante, pero en una release se facilita la lectura del esquemático a cualquier interesado. Otros ejemplos son el BOM o el .net.

AGREGAR

ej2.kicad_pcb: Al momento está vacío.
ej2.sch: Hoja principal.
Rs232.sch: Hoja secundaria.
Conektor.sch: Hoja secundaria.
ej2.pro: Proyecto.
ej2-cache.lib: Una copia de los símbolos usados.

AGREGAR DEPENDE

ej2.net: Agregar si el cambio impacta en el PCB.
ej2.csv: Agregar en versiones importantes/finales.

NO SUBIR

***.bak :** Archivos de backup
ej2.xml: (agregar al .gitignore)

Directorios ejercicio2

libs: biblioteca provista + creada.
datasheets: Podemos colocar una copia del pdf.
doc: Un texto comentando el ejercicio.
ej2.pretty: Aún no hay nada aquí.
ej2.3dshapes: Aún no hay nada aquí.
tools: Scripts para el BOM.
gerbers: Aún no hay nada aquí.

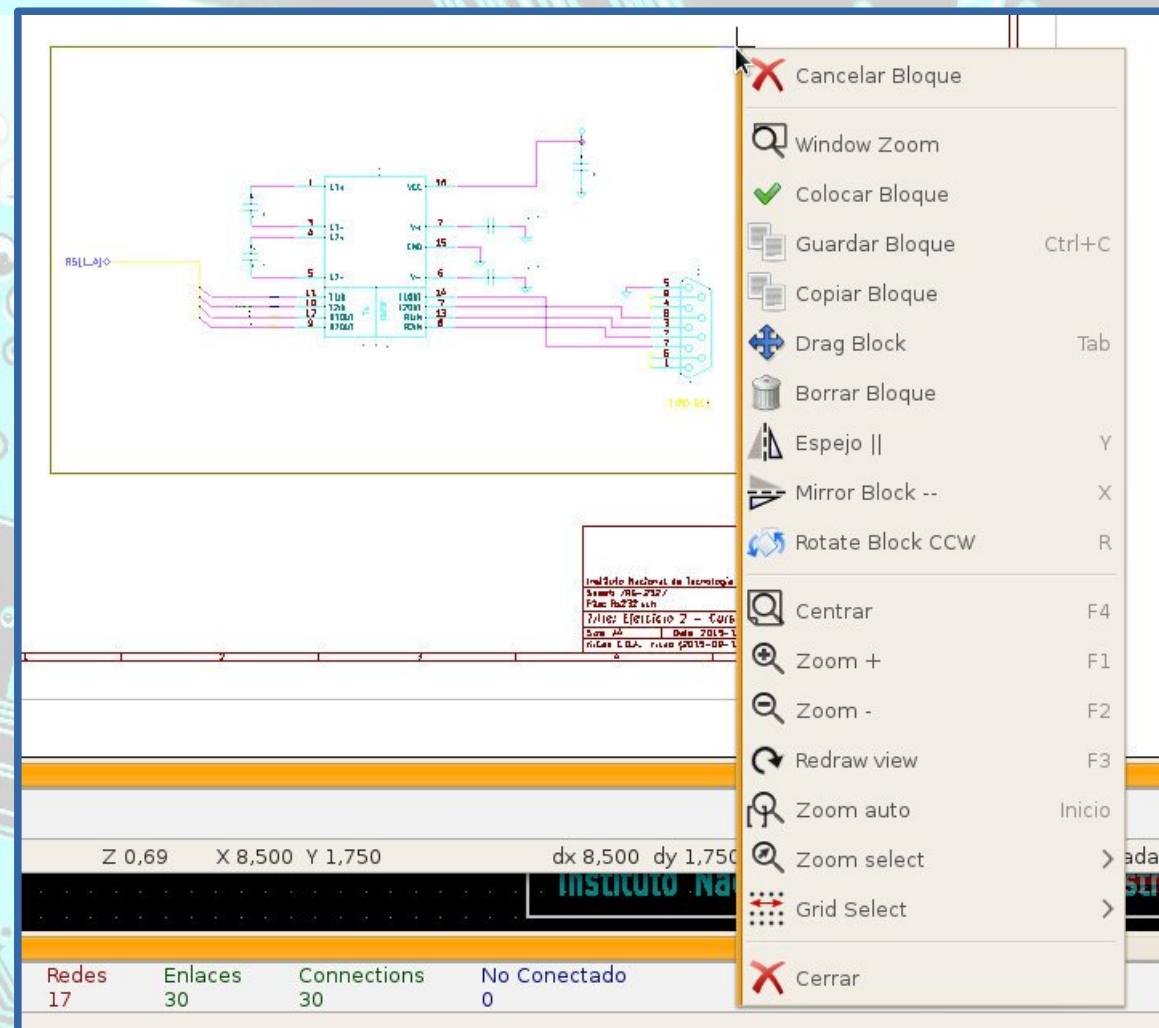
Eschema - Operaciones de bloques

En el esquemático podemos realizar algunas operaciones de bloques como mover, copiar, espejar o borrar.

Para realizar las operaciones de bloques:

- 1)Presionar el botón izquierdo de mouse en la primer esquina del bloque.
- 2)Manteniendo presionado, mover el mouse hasta la otra esquina del bloque a definir.
- 3)Soltar el botón izquierdo y presionar el derecho para desplegar el menú de operaciones de bloques.
- 4)Si no se presiona el botón derecho, se aplica la operación mover bloque.

- 1)Probar operaciones de bloques, usando el “deshacer” luego de cada una.
- 2)Intentar copiar un bloque de una hoja a la otra. Ayuda: usar “Guardar bloque”.



Tip: En KiCad todavía no se puede confiar “ciegamente” en la función deshacer. Es una característica agregada recientemente y algunas pocas operaciones en Eeschema y Pcbnew no tienen deshacer.

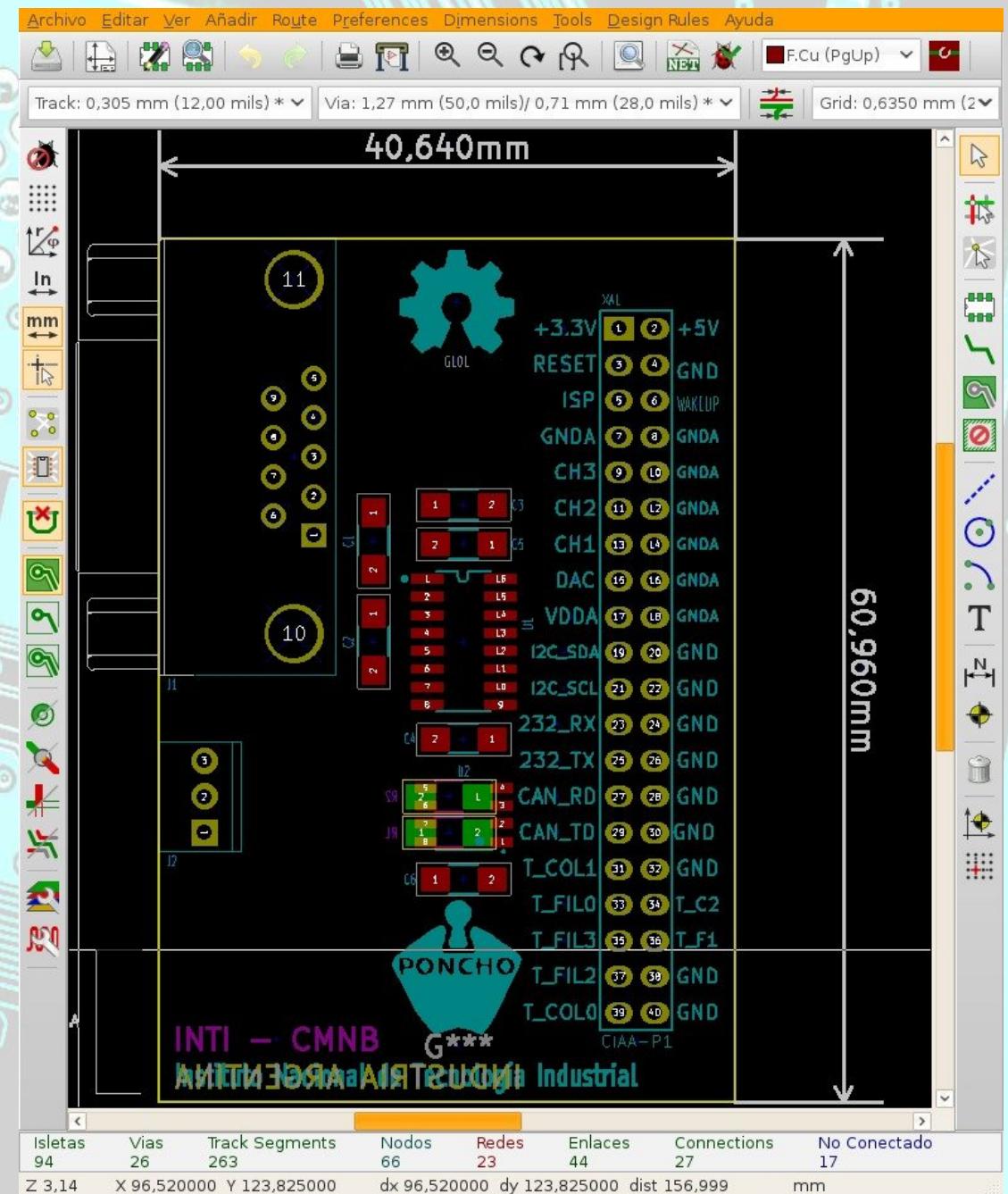
Ejercicio 2

Editor de PCB

Ejercicio 2 - Importar netlist

En el PCBnew ingresamos el netlist, separamos los componentes, dibujamos el borde de impreso y realizamos un placement rápido.

- 1)Ingresar el netlist al pcbnew.
- 2)Separar componentes automáticamente.
- 3)Dibujar borde de impreso (Usar plantilla).
- 4)Realizar un placement tentativo.
- 5)Colocar componentes en ambas capas (Front y Bottom).



Ejercicio 2 - Cambio de footprint

MALAS NOTICIAS:

Desde el proyecto CIAA nos piden que utilicemos footprints SMD 1206 que se puedan soldar fácilmente a mano en vez del 0603 que hemos usado!



Para incursionar en el editor de footprints realizaremos el encapsulado SMD 1206 dese cero, para luego reemplazarlo en nuestro PCB.

1206!

1206

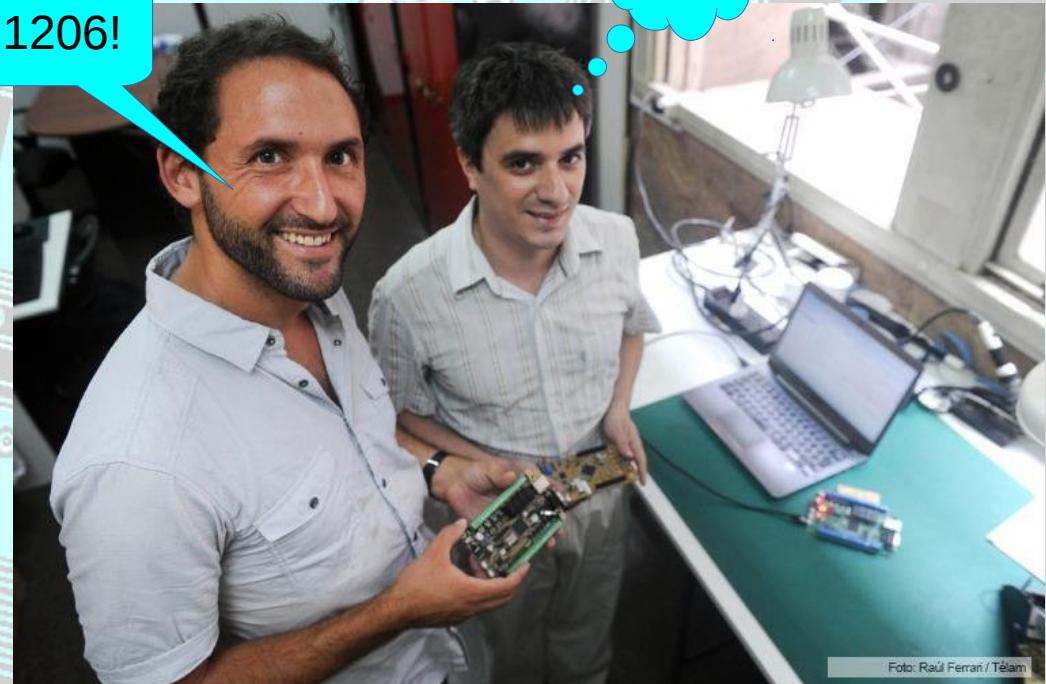


Foto: Raúl Ferrari / Télam

Imagen cortesía de <http://www.telam.com.ar/>

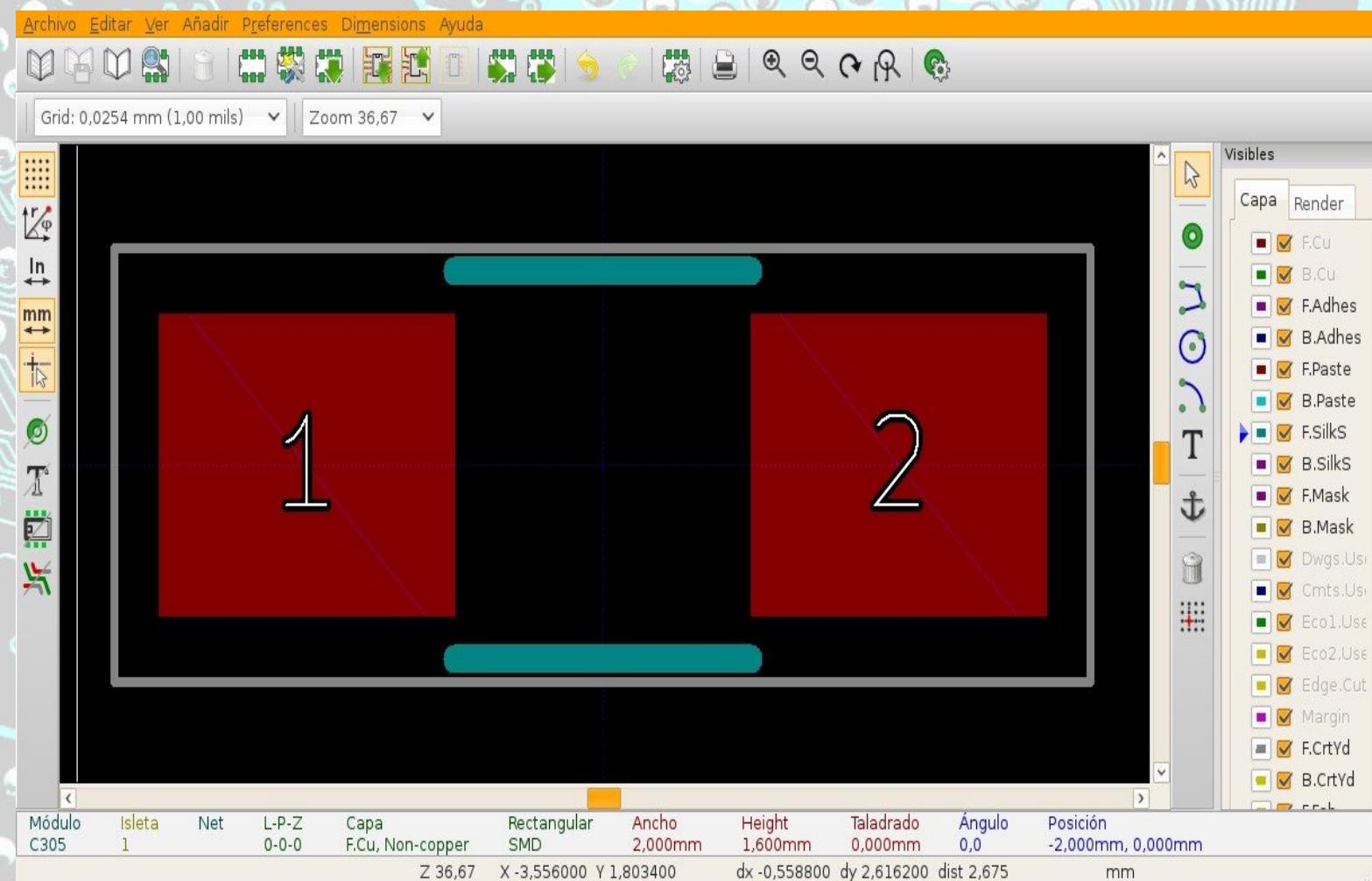
Ejercicio 2

Editor de Footprints

Ejercicio 2 - Creación de footprints

Vamos a realizar un footprint SMD para soldar a mano un 1206. Los pads son un poco más grandes que el 1206 normal.

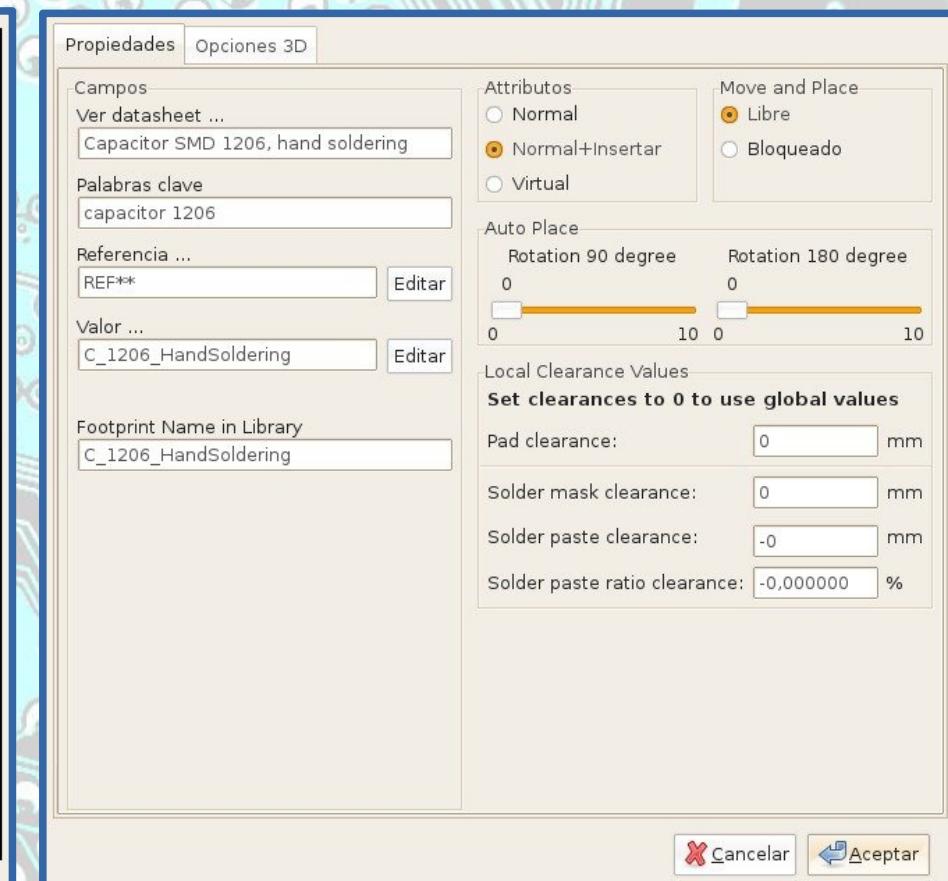
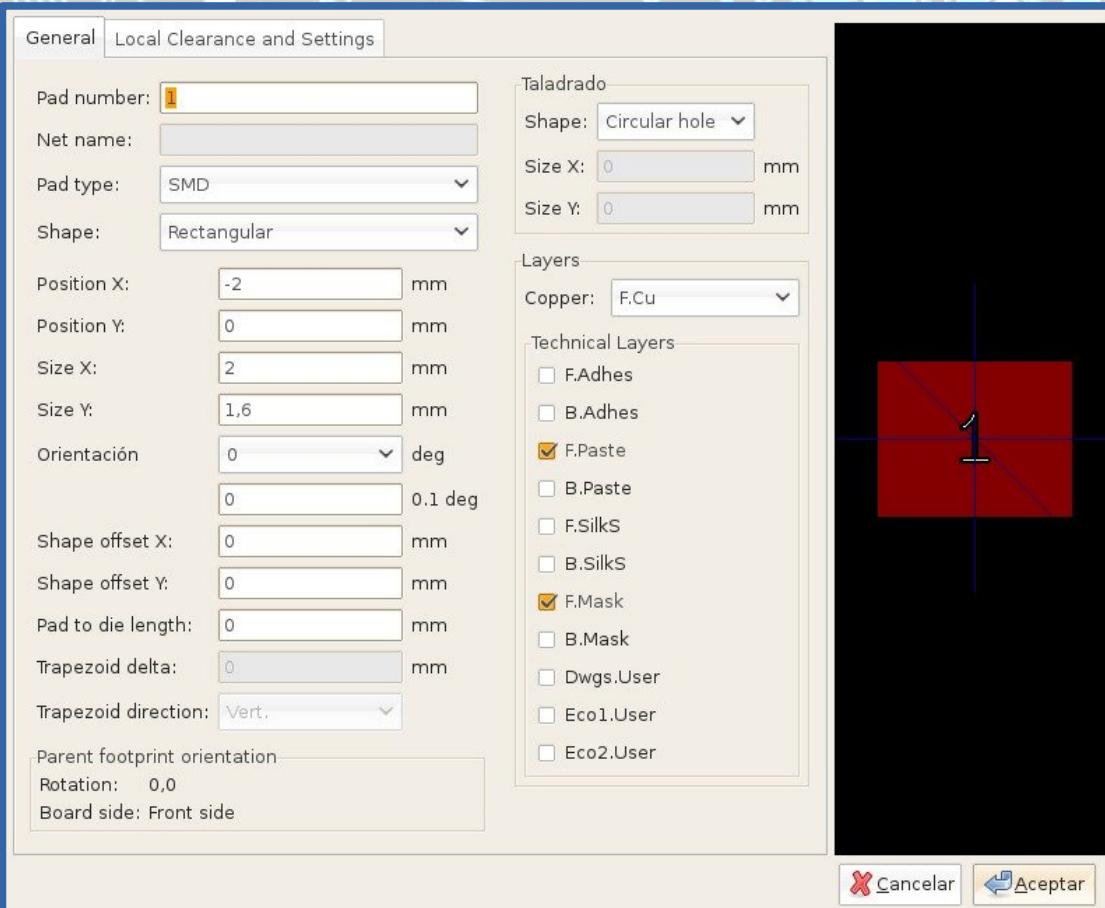
- 1)Abrir el editor de módulos.
- 2)Nuevo footprint.



Ejercicio 2 - Creación de footprints

Crearemos un footprint para SMD 1206 con facilidades para soldar a mano.

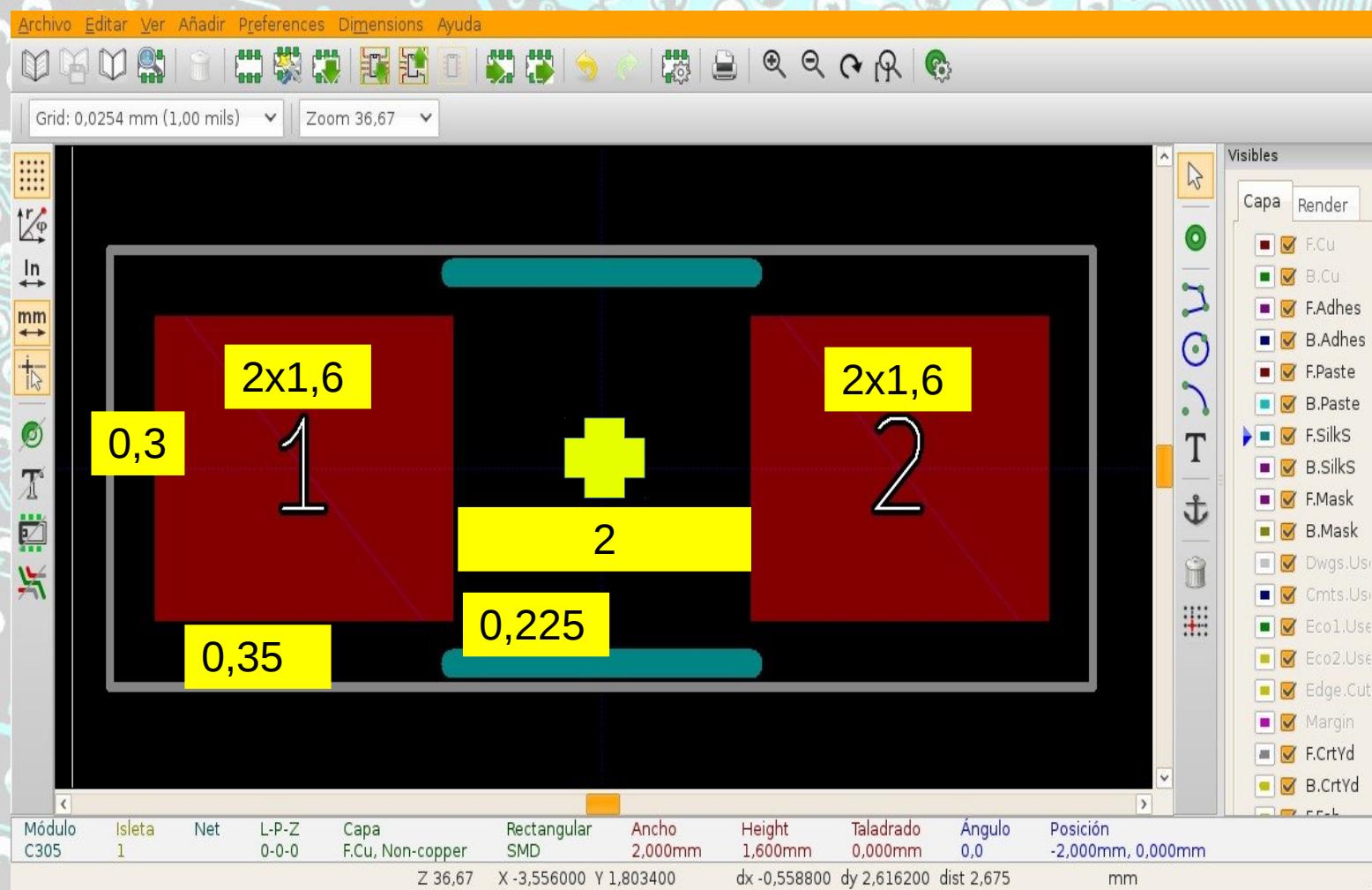
- 1) Crear los pads 1 y 2.
- 2) Completar las propiedades.



Ejercicio 2 - Creación de footprints

Dimensiones en mm.

- 1) Ubicar los pads.
- 2) Dibujar y ubicar los demás elementos (ver información adicional en el siguiente slide).

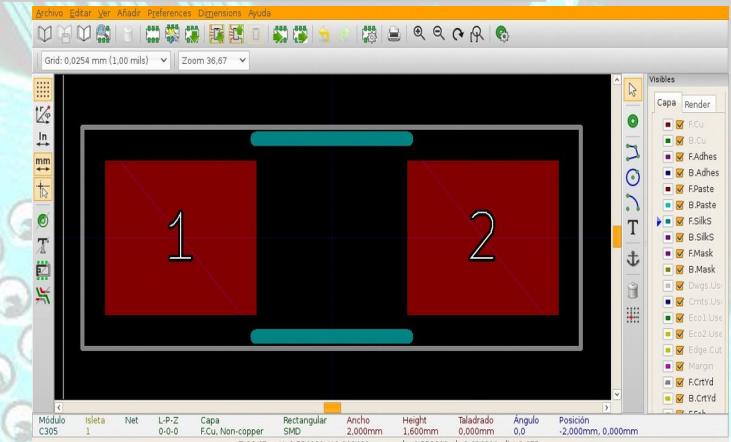


Origen

Ejercicio 2 - Creación de footprints

Líneas de serigrafía y courtyard.

- 1) Dibujar el courtyard y la serigrafía.
- 2) Una vez finalizado exportar el footprint al directorio ej2.pretty.
- 3) En el esquemático o en el Cvpcb asociar los capacitores al footprint nuevo.
- 4) Exportar el netlist.



Front Silkscreen

Start point X:	1	mm	Item thickness:	0,15	mm
Start point Y:	-1,025	mm	Default thickness:	0,15	mm
End point X:	-1	mm	Capa:	F.Silks	
End point Y:	-1,025	mm			

Front Courtyard

Start point X:	-3,3	mm	Item thickness:	0,05	mm
Start point Y:	-1,15	mm	Default thickness:	0,15	mm
End point X:	3,3	mm	Capa:	F.CrtYd	
End point Y:	-1,15	mm			

Ejercicio 2

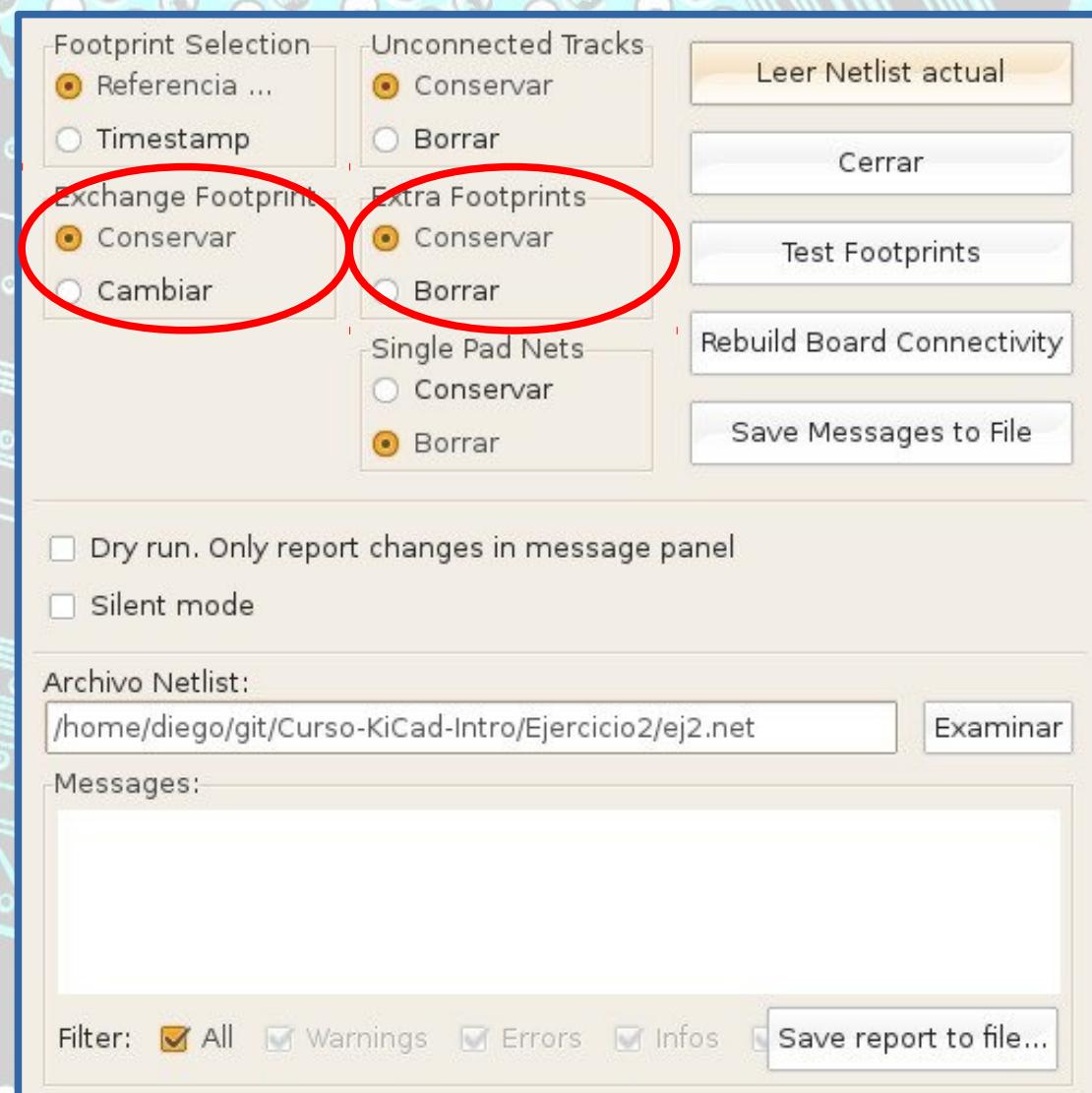
Editor de PCB

Ejercicio 2 - Reemplazo de footprints

En el Pcbnew ingresamos el netlist nuevo.

Cambiar las opciones por defecto de “Exchange Footprint” y “Extra Footprint”. Probaremos primero la opción “Dry run”.

- 1) Ir al diálogo de ingreso de netlist.
- 2) Habilitar “Dry run”.
- 3) Configurar para cambiar los footprints y borrar los footprints que sobren (esta última opción no es necesaria en este ejemplo).
- 4) Leer netlist y analizar la ventana de mensajes.
- 5) Deshabilitar “Dry run” y realizar la operación.
- 6) Realizar nuevamente la ubicación de componentes.



Ejercicio 2 - Tags de GIT

Ingresamos el estado actual a GIT, dejando todo listo para el ruteo de nuestro PCB.

Algunas etapas de nuestro desarrollo es conveniente marcarlas porque tal vez sea útil rastrear fácilmente ese momento.

Para esto se pueden usar los TAGs de GIT.
Un TAG es simplemente una marca con un nombre fácil que podamos identificar.

Veremos los “*lightweight*” que son los más sencillos.

```
$ git tag NOMBRETAG  
$ git push origin NOMBRETAG  
$ git tag
```

El primer comando aplica el tag al repositorio local.
El segundo envía el tag al repositorio remoto.
El tercero muestra los tags.

Tip: Aunque no utilicemos tags, cada commit tiene un código único (SHA-1) que es equivalente a un tag. Es útil para identificar una versión o un estado de nuestro PCB.

```
$ git log --abbrev-commit  
commit e7360dc  
Author: Diego Alamon <dalamon@inti.gob.ar>  
Date: Wed Sep 30 17:08:32 2015 -0300  
Se agregaron los tornillos de fijación para el formato uQseven.
```

Tip: Dependiendo el caso puede ser útil *taggear* por ejemplo al momento de cerrar el esquemático, antes del ruteo, al finalizar el ruteo, al enviar a cotizar y al enviar a fabricar.

- 1) Actualizar el repositorio con pull, y luego git add y commit.
- 2) Enviar los cambios al repositorio remoto (push).
- 3) Aplicar un tag “INICIALES_pre_route” con git tag.
- 4) Enviar el tag al repositorio central.

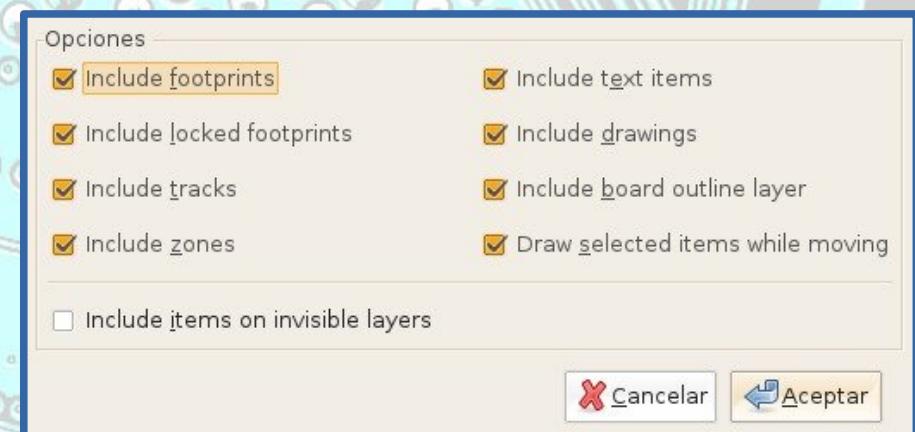


Pcbnew - Operaciones con bloques

En el editor de PCB podemos realizar algunas operaciones de bloques como mover, copiar, espejar, girar y borrar.

Para realizar las operaciones de bloques:

- 1) Presionar el botón izquierdo de mouse en la primer esquina del bloque.
- 2) Manteniendo presionado, mover el mouse hasta la otra esquina del bloque a definir.
- 3) Soltar el botón izquierdo y aparecerá un diálogo para seleccionar qué elementos del bloque considerar y cuales no.



Tip: Para desrutear un circuito o una parte del mismo, incluir solo los tracks.

Pcbnew - Operaciones con bloques

Luego del diálogo de selección de objetos.

MOVER BLOQUE

- 1) Simplemente mover el mouse para mover el bloque.
- 2) Click izquierdo para colocar el bloque.

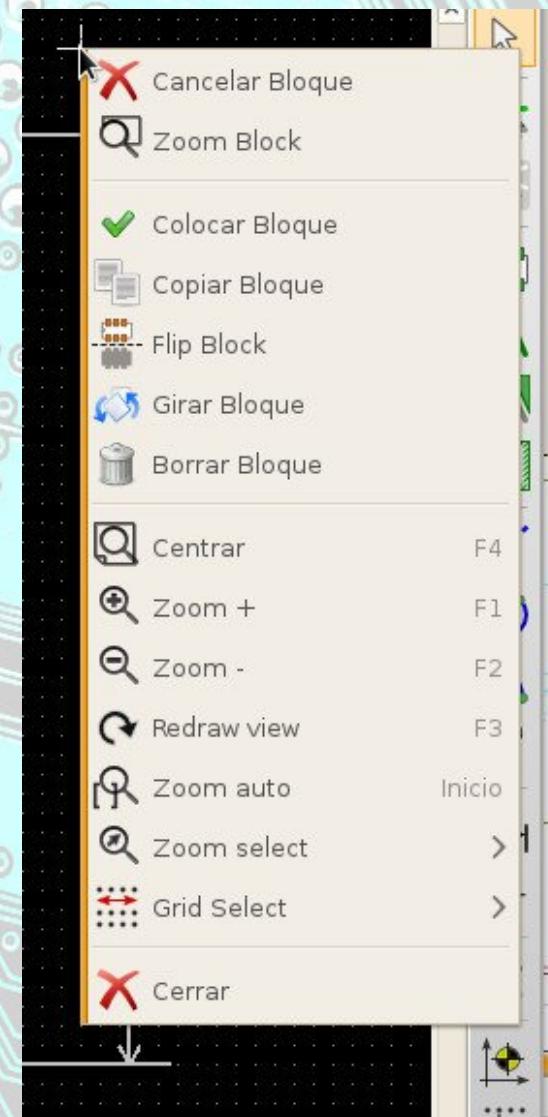
COPIAR BLOQUE

- 1) Mover el bloque seleccionado a la posición donde se desea la copia (igual que mover bloque).
- 2) Botón derecho y seleccionar “Copiar bloque”.

ESPEJO, GIRO y BORRAR

- 1) Botón derecho y elegir la opción correspondiente.

- 1) Probar operaciones de bloques, usando el “deshacer” luego de cada una.



Ejercicio 2 - Ruteo

Elegiremos la tecnología de 12 mils para nuestro diseño.

- 1) Configurar ancho de traza, vías y márgenes según la tabla del fabricante.

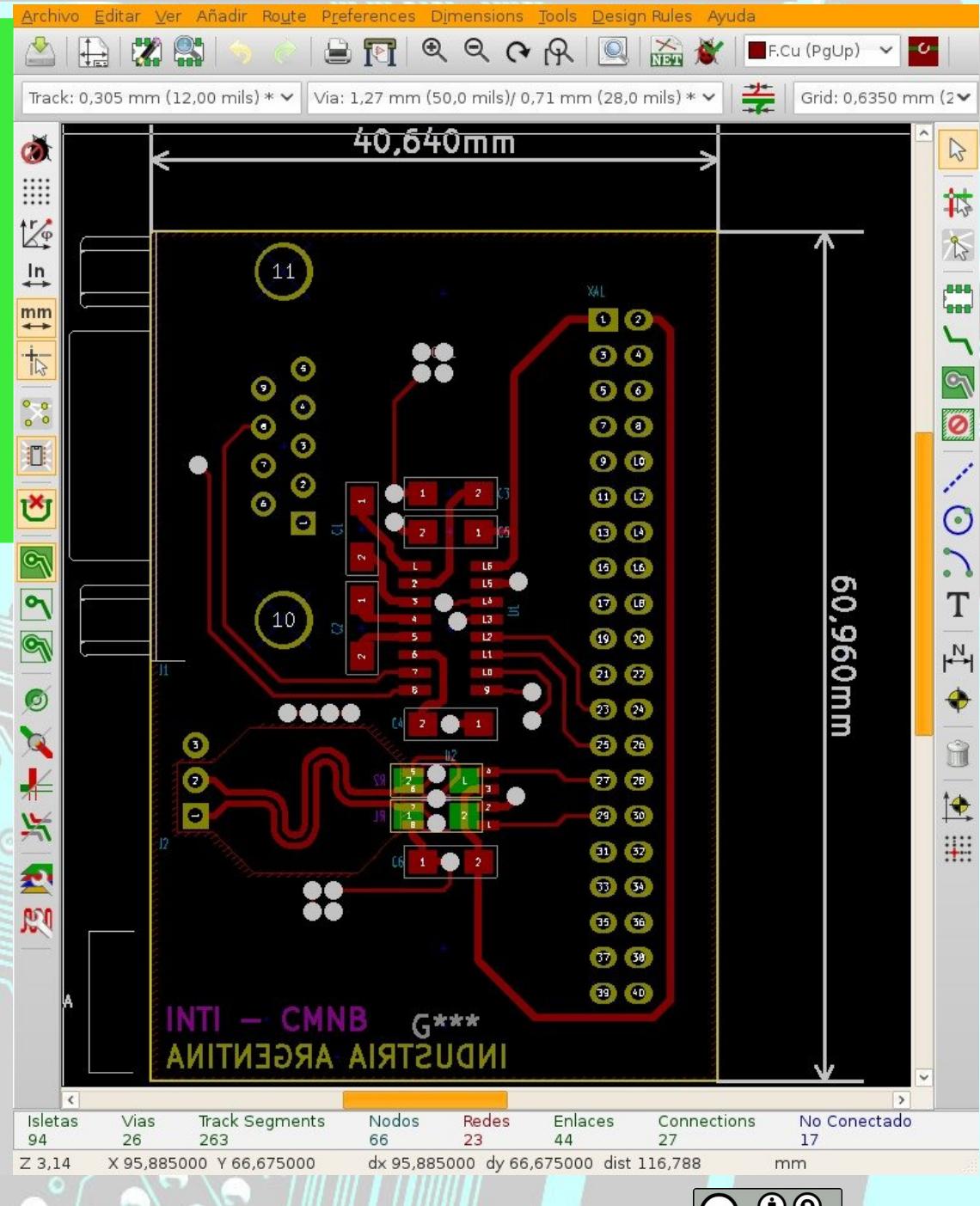
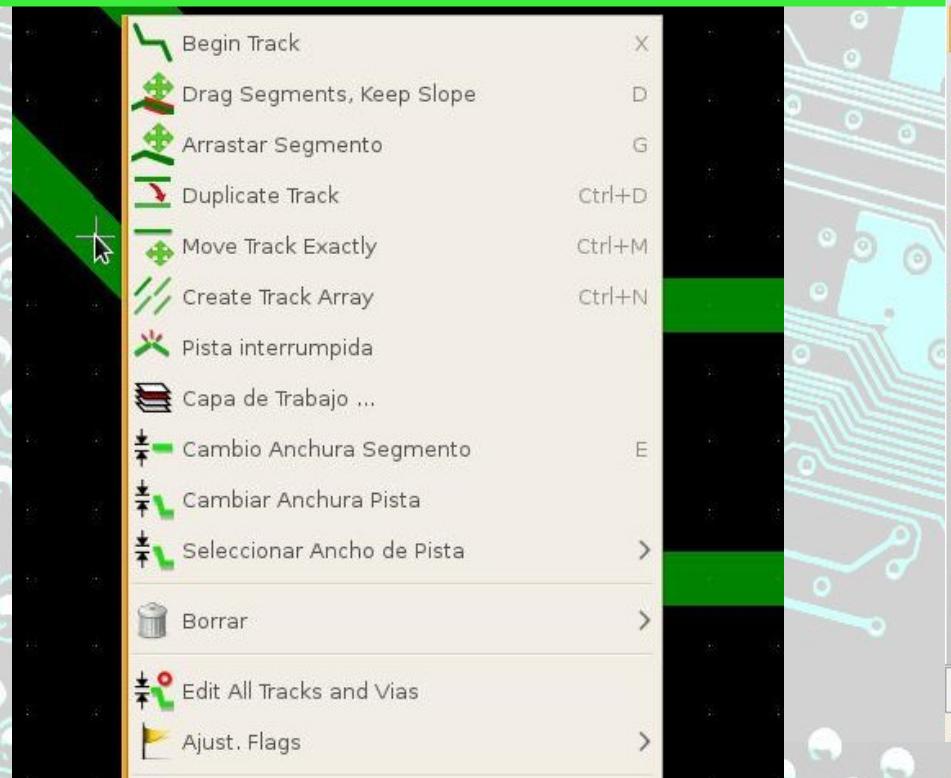
Estándares de fabricación

TECNOLOGIA	12 MILS	10 MILS	8 MILS	6 MILS
DIAMETRO DE AGUJERO	28 (0.70 mm)	20 (0.50 mm)	16 (0.40 mm)	16 (0.40 mm)
DIAMETRO DE PAD O VIA	50 (1.27 mm)	40 (1.00 mm)	32 (0.80mm)	28 (0.70mm)
ANCHO DE TRAZA	12 (0.30mm)	10 (0.25 mm)	8 (0.20 mm)	6 (0.15mm)
SEPARACION ENTRE TRAZAS	12 (0.30mm)	10 (0.25 mm)	7 (0.18 mm)	6 (0.15mm)
SEPARACION ENTRE TRAZA Y PAD/VIA	10 (0.25 mm)	8 (0.20 mm)	6 (0.15mm)	6 (0.15mm)
DISTANCIA DE COBRE A BORDE	12 (0.30mm)	12 (0.30mm)	12 (0.30mm)	12 (0.30mm)
ALTURA - TRAZO DE LETRAS	48-8	36-6	30-5	30-5

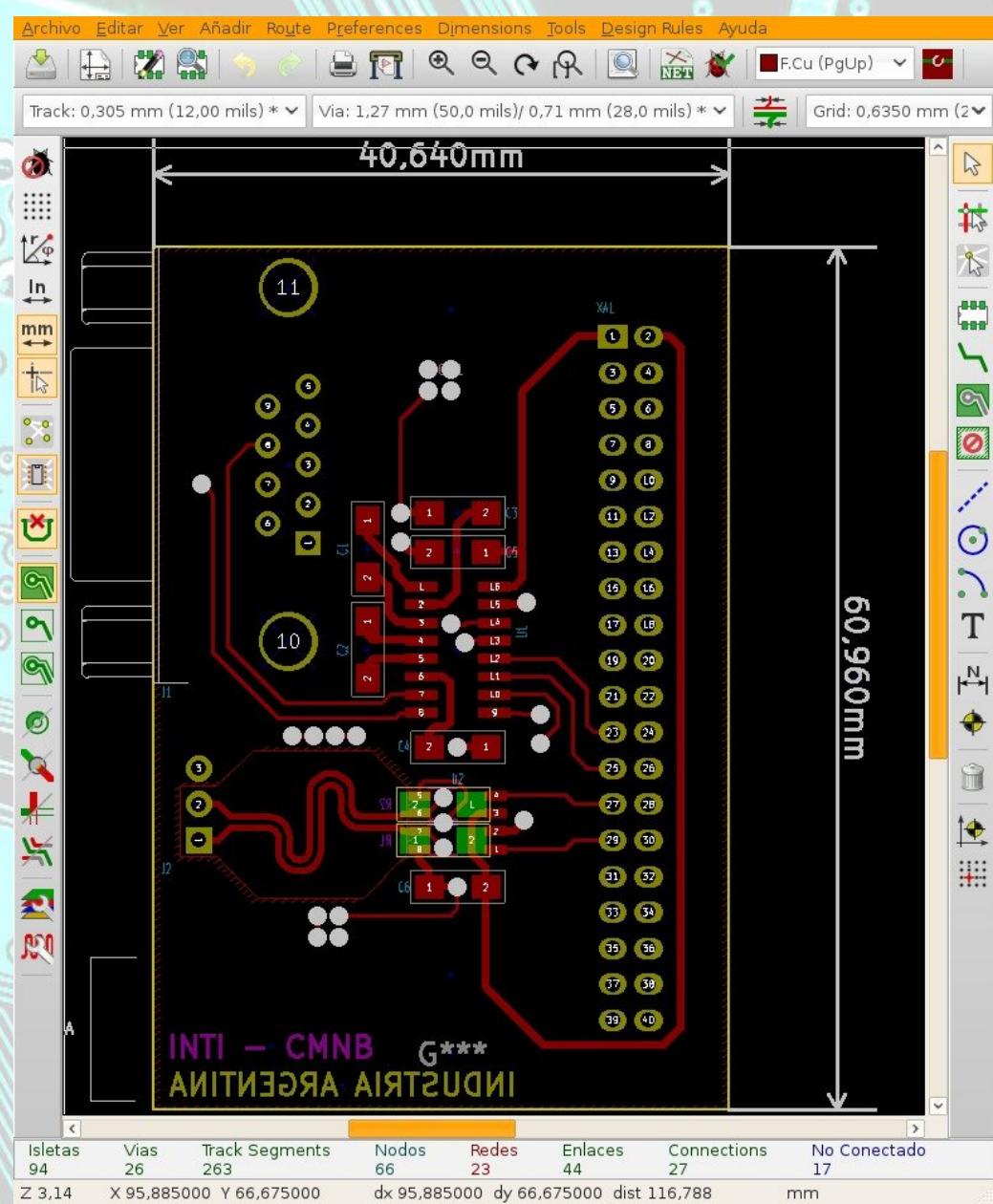
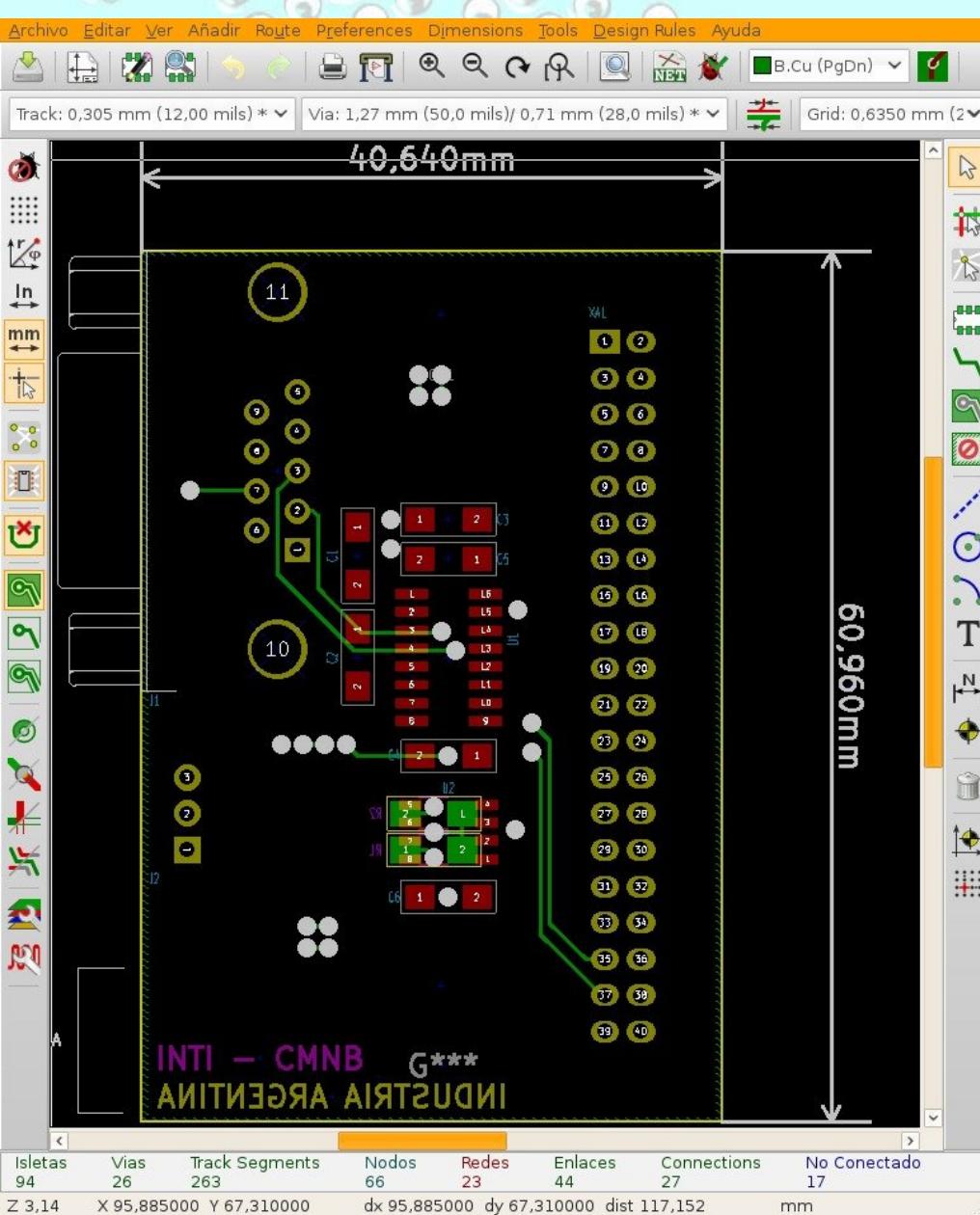
Imagen cortesía de <http://www.mayerpcb.com/>

Ejercicio 2 - Ruteo

- 1) Realizar el ruteo, colocando los resistores en el Bottom y el resto en el Front.
- 2) Investigar las opciones de mover nodo, mover segmento, mantener pendiente, etc.
- 3) Las líneas del CAN se rutean en forma tradicional y las haremos diferentes el día 3 del curso.
- 4) Los planos de tierra se explican en las siguientes diapositivas. No conectar las tierras.



Ejercicio 2 - Ruteo

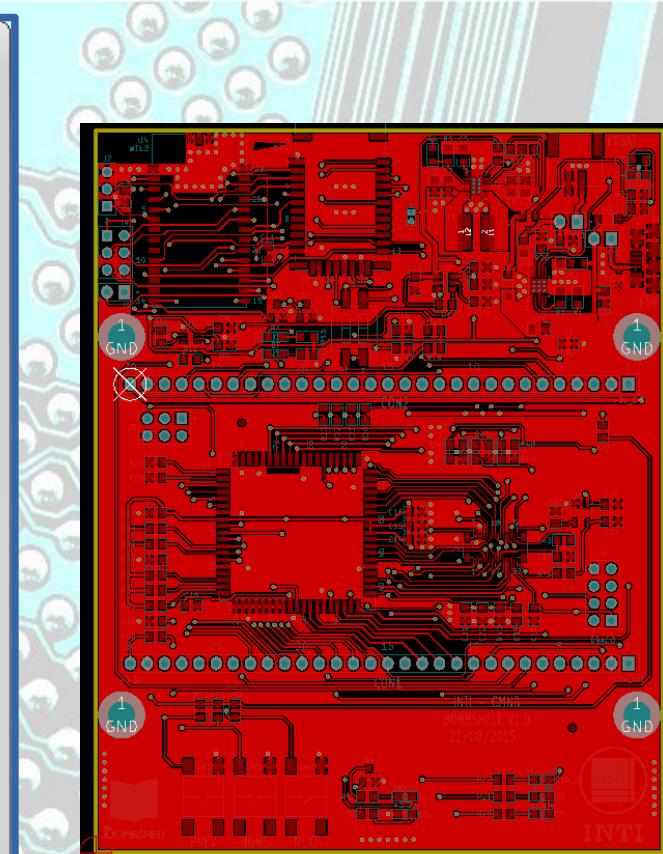
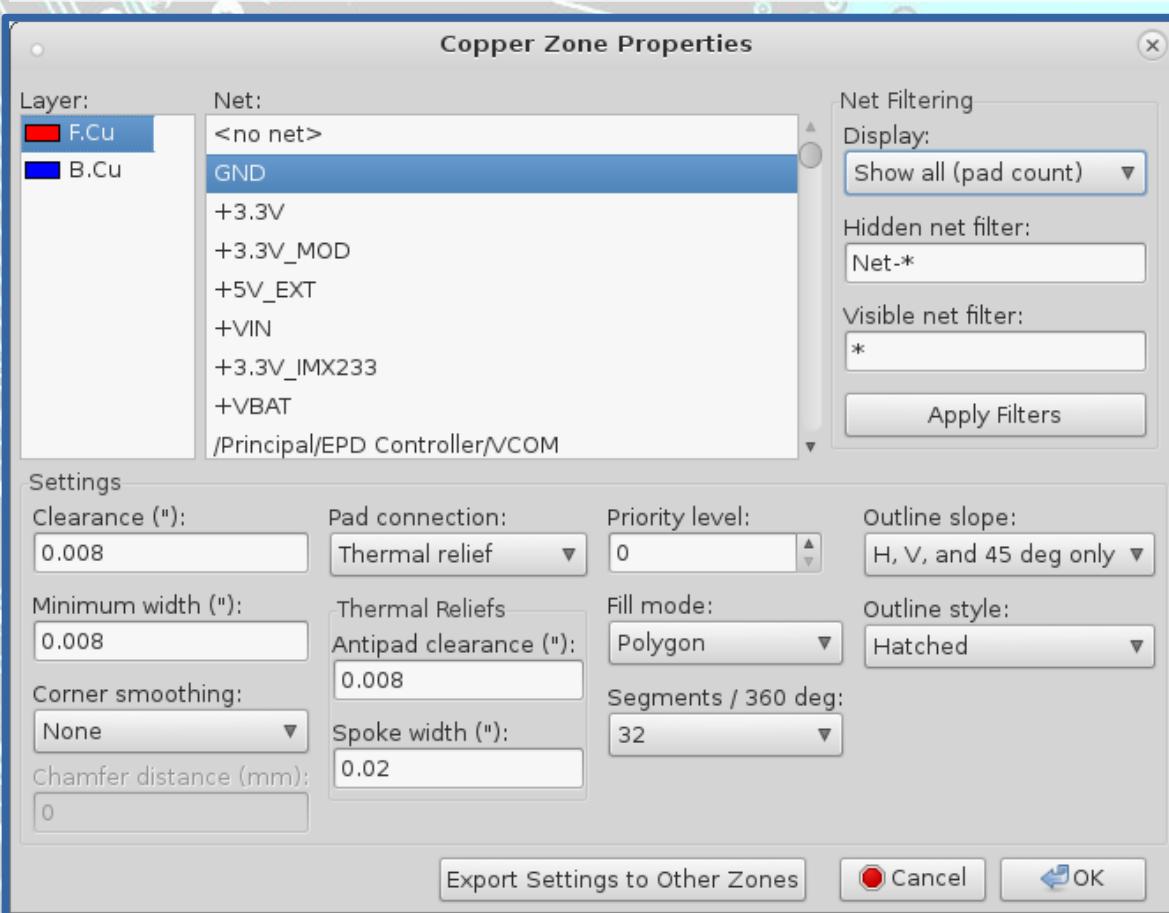


En esta nueva versión de KiCad no hace falta rutear lo que irá conectado a las áreas de cobre (ver diapositivas siguientes).

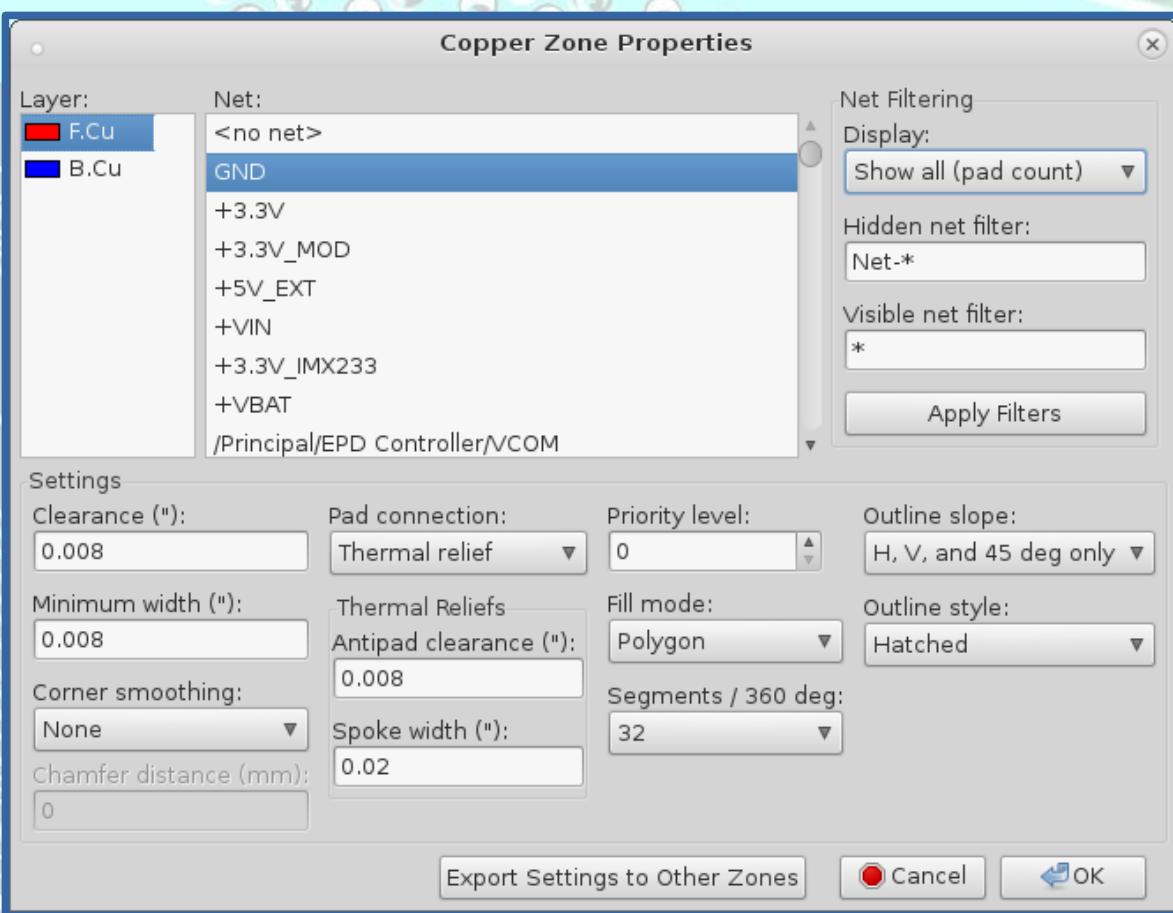
Pcbnew - Áreas de cobre (Copper zones) I

Creación de áreas de cobre

- 1) Activar cualquier layer de cobre y clickear en el botón **Agregar Zona de Cobre**.
- 2) Seleccionar Layer y Net.
- 3) Configurar todos los parámetros deseados: Clearance, Minimum Width, Pad Connection, Fill mode, Priority level.
- 4) Trazar los bordes de la zona y finalizar con doble click.
- 5) Para llenar hacer click derecho en el borde del área e ir a **Zones → Fill Zone**.

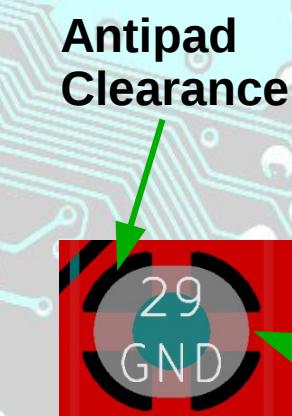


Pcbnew - Áreas de cobre (Copper zones) II



Clearance: Poner un valor un poco superior al clearance general de la placa.

Minimum Width: Se utiliza para limitar la formación de áreas de cobre muy pequeñas. No se debe agrandar mucho porque afecta a los thermal reliefs.



Priority Level: Se utiliza para crear zonas dentro de otras zonas. Poner en 0 para grandes zonas y en 1 ó más para sub-zonas.

Fill mode: Dejar en Polygons.

Segments: Dejar en 32, ya que no afecta la performance.

Outline Slope: Dejar en H, V y 45º.

Outline Style: Dejar en Hatched.

Pad Connection: Por lo general se utiliza Thermal Relief para que no se dificulte la soldadura de los componentes. En los casos de planos que se agregan para disipación, puede dejarse en Solid.

Antipad Clearance: Un valor muy grande puede impedir que se forme un thermal relief en un pad muy pequeño.

Spoke width: Debe ser mayor que el valor Minimum Width.

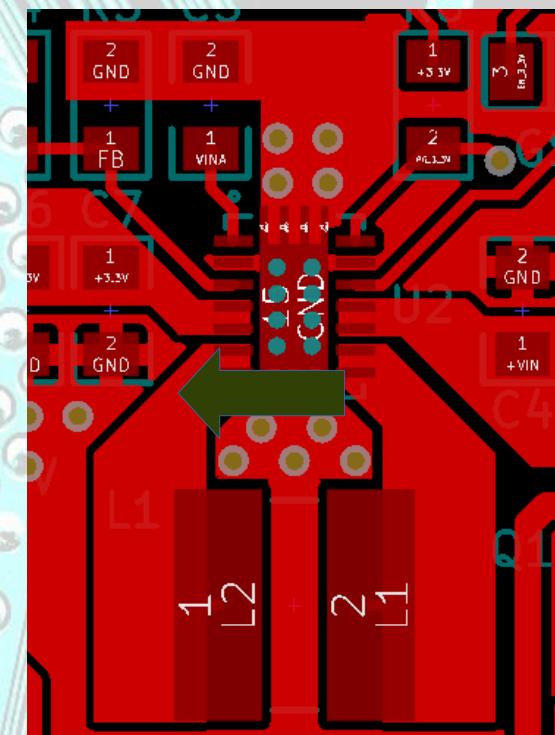
**Spoke Width
(Track Width)**

Pcbnew - Áreas de cobre (Copper zones) III

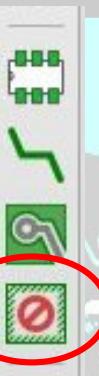
Fill Zones: Para reconstruir zonas usar la tecla '**B**'. Utilizar periódicamente para tener siempre actualizados los planos!

Unfill Zones: Para sacar temporalmente el relleno de las zonas, usar '**Ctrl+B**'. Es útil si se debe rutear nuevamente un sector del PCB o mover algún componente.

Para editar la forma de una zona existente se pueden crear y eliminar **corners (puntos de inflexión)**, esto resulta útil para perfeccionar el contorno de la zona!



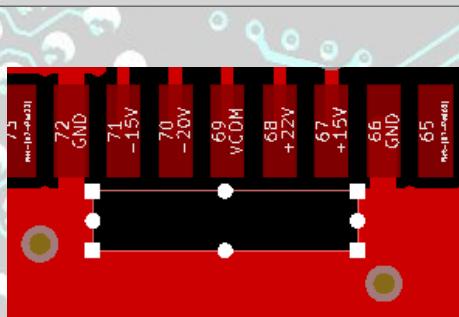
Pcbnew - Áreas de cobre (Copper zones) IV



Keepout Area

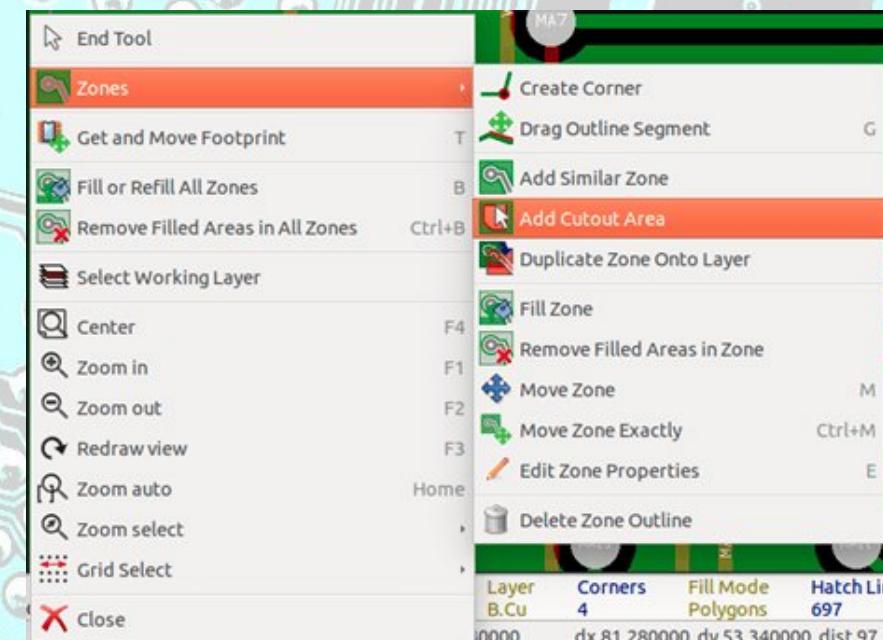
- 1) Click en Add Keepout Area
- 2) Seleccionar Layer.
- 3) Configurar opciones (No Tracks, No Vias, No Copper Pour).
- 4) Trazar los límites de la zona.
- 5) Rellenar las zonas nuevamente.

Keepout Area Properties



Cutout Area (default mode)

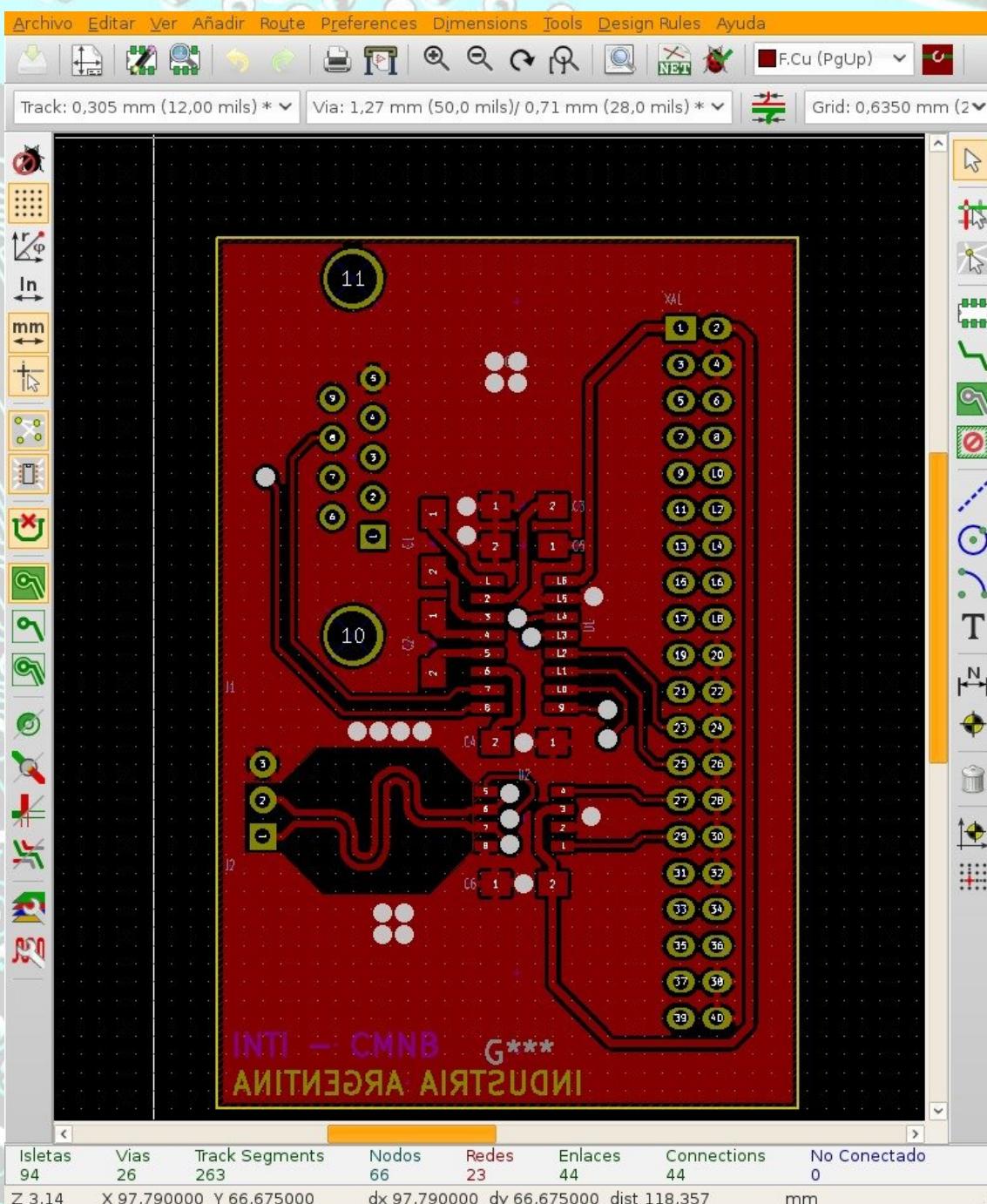
- 1) Click derecho en el borde de la zona.
- 2) Seleccionar Add Cutout Area.
- 3) Trazar los límites de la zona.
- 4) Rellenar las zonas nuevamente.



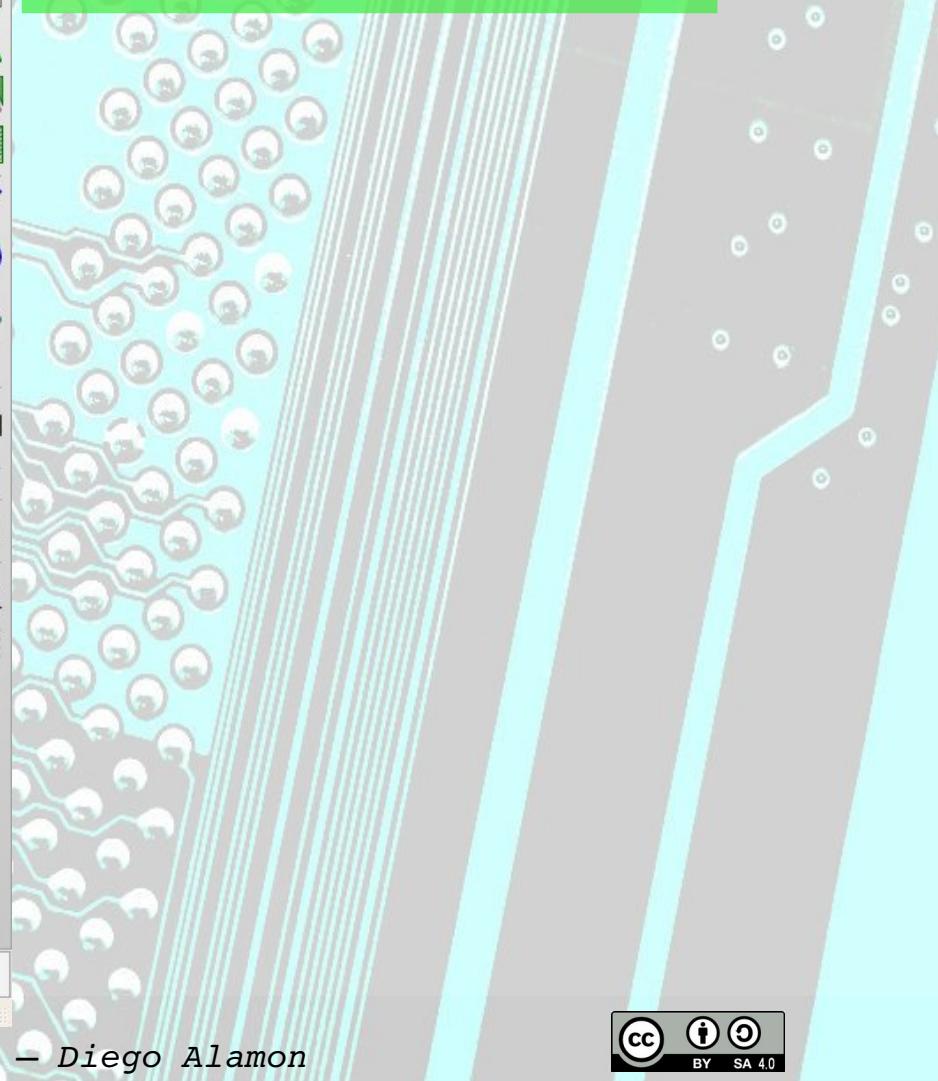
Diferencias

Si se traza una pista en una **Keepout Area** con la opción **No Tracks** habilitada, se produce un error de DRC. Esto no sucede en una **Cutout Area** que simplemente es un recorte en la zona.

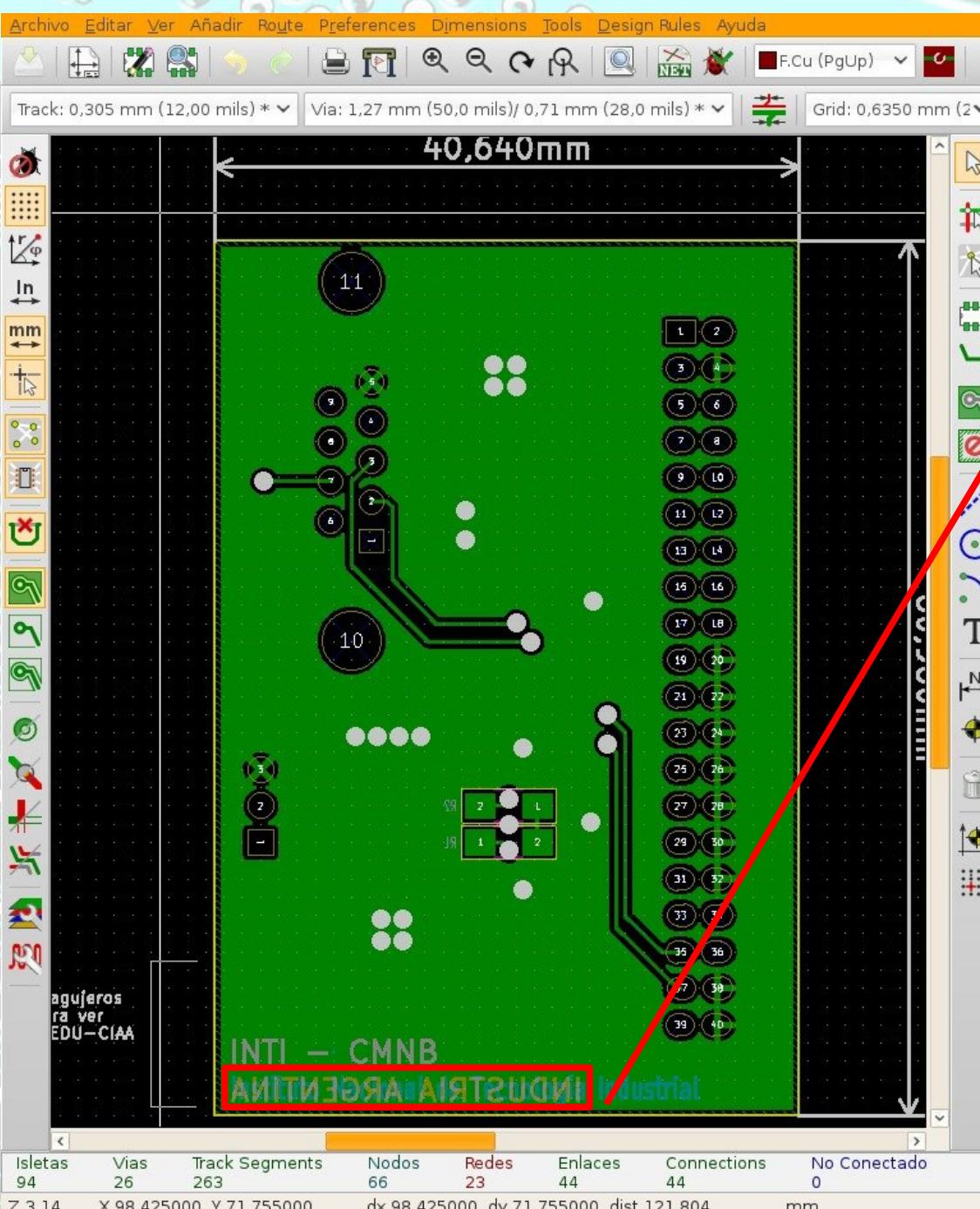
Ejercicio 2 - Áreas de cobre y Keepout



- 1) Colocaremos áreas de cobre en ambas capas.
- 2) En las líneas del CAN agregar una zona de Keepout, que será de utilidad más adelante.



Ejercicio 2 - Áreas de cobre y Keepout



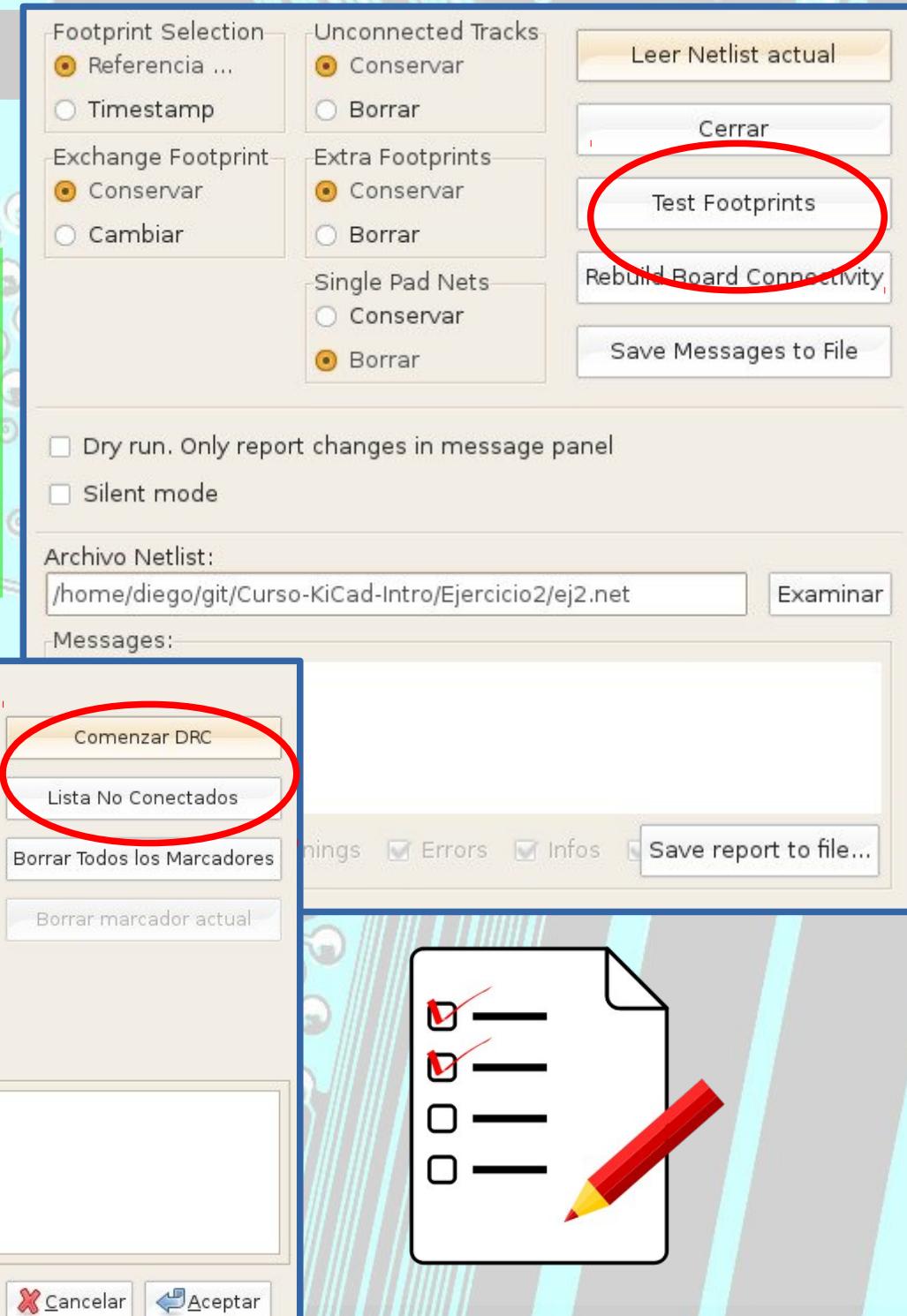
- 1) Colocar textos en capa superior, capa inferior y serigrafía superior.
- 2) Colocar el texto INDUSTRIA ARGENTINA en la capa bottom de cobre.
- 3) Agregar una cutout area que contenga al texto anterior.
- 4) Colocar logos.
- 5) No olvidar también acomodar la serigrafía (campo referencia).

TIP: Algunos textos adicionales a colocar:
INDUSTRIA
ARGENTINA, Empresa o institución, Fecha, SHA-1 corto o tag, URL, Autor/es, Proyecto, etc.

Ejercicio 2 - Verificaciones

Una vez terminado el ruteo realizaremos algunas verificaciones simples.

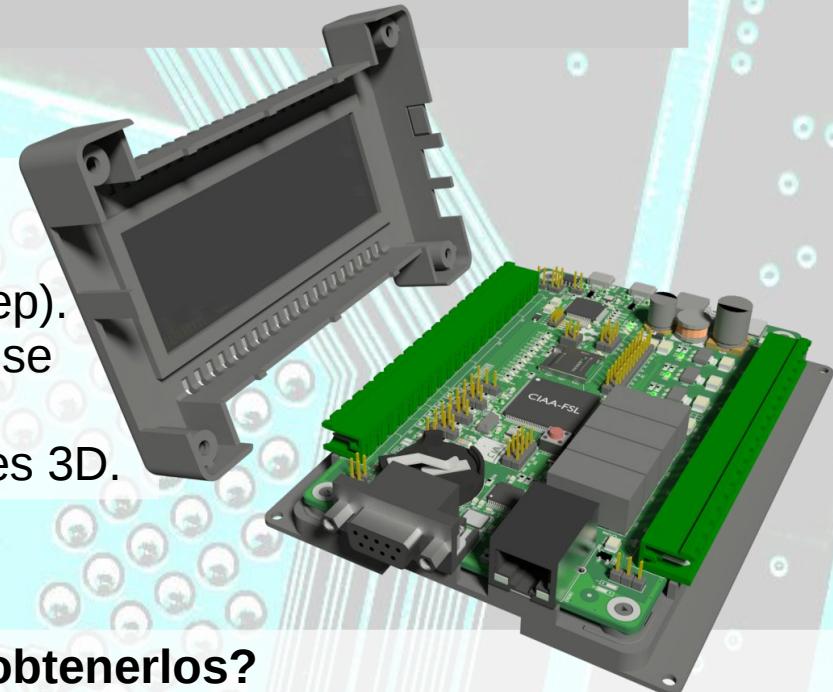
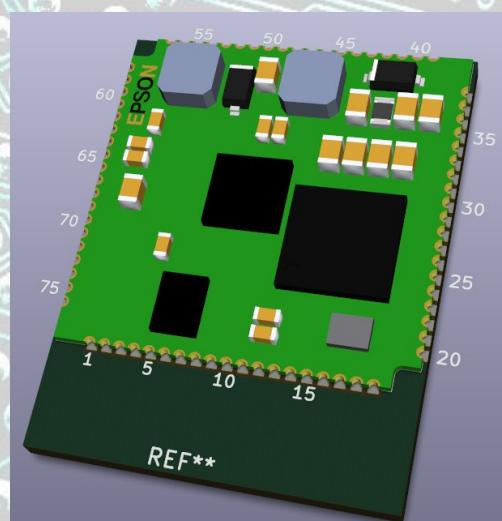
- 1) En el diálogo de netlist usar el botón de "Test Footprints".
- 2) Correr el DRC y la opción de "Lista No conectados."
- 3) También se puede generar el netlist en el esquemático y leerlo nuevamente para asegurarnos que todo coincide.



Pcbnew - Modelos 3D I

Modelos 3D: Cuándo usarlos?

- Cuando se va a diseñar un gabinete o se necesita chequear uno existente (se debe tener el modelo step).
- Cuando se quiere verificar que los componentes no se toquen entre sí o no tapen la serigrafía.
- Para promocionar un producto o realizar animaciones 3D.



Modelos 3D: De dónde obtenerlos?

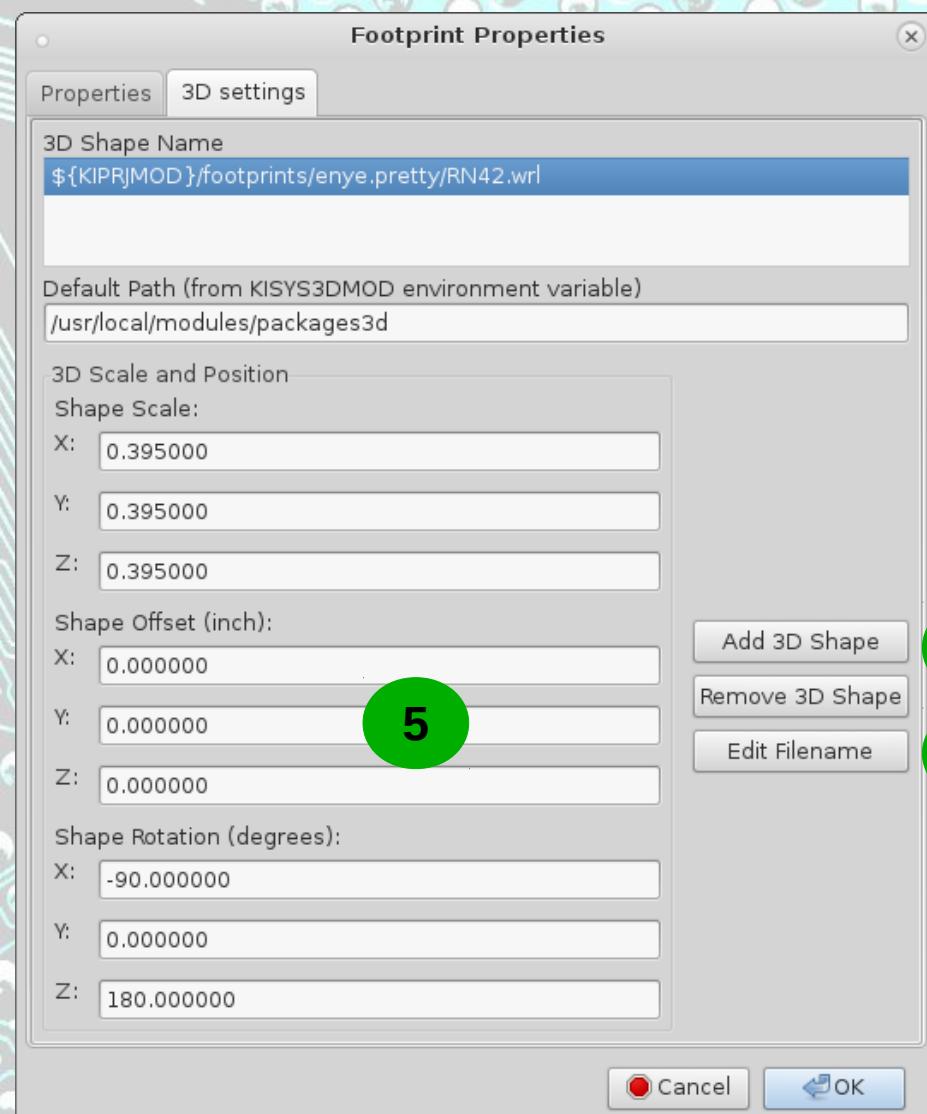
- Librerías estándar de KiCad
<https://github.com/KiCad/kicad-library/tree/master/modules/packages3d>
también en (/usr/local/share/kicad/modules/packages3d).
- Bajar el modelo IGES o STEP que provea el fabricante y convertirlo al formato VRML (.WRL) con FreeCad.
- Bajar el modelo de la página www.3dcontentcentral.com (ver licencias).
- Crear el modelo 3D con FreeCad.



Tip: Usar FreeCad para convertir archivos o superponer modelos 3D.

Pcbnew - Modelos 3D II

Footprint Properties → 3D Settings

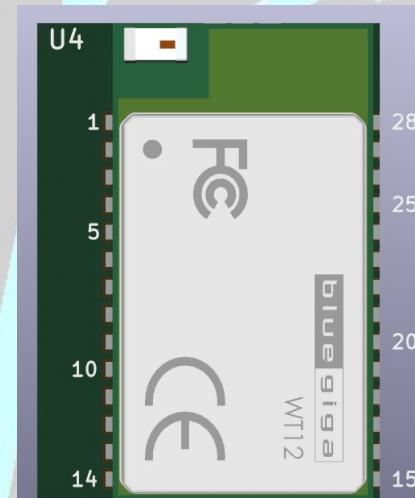


Modelos 3D: Cómo asignarlos a un componente?

- 1) Copiar el archivo WRL en el directorio de 3D , por ej. /ej2.3dshapes/.
- 2) Seleccionar un componente, presionar la tecla 'E' y abrir el cuadro **3D Settings**.
- 3) Presionar **Add 3D Shape** y buscar el modelo.
- 4) Presionar **Edit Filename** y agregar en la ruta del archivo:
\${KIPRJMOD}/ej2.3dshapes/
- 5) Si fuera necesario, escalar, desplazar o rotar el modelo.

El cambio de ruta permitirá que al abrir el PCB desde otra ubicación, los modelos 3D se carguen correctamente!

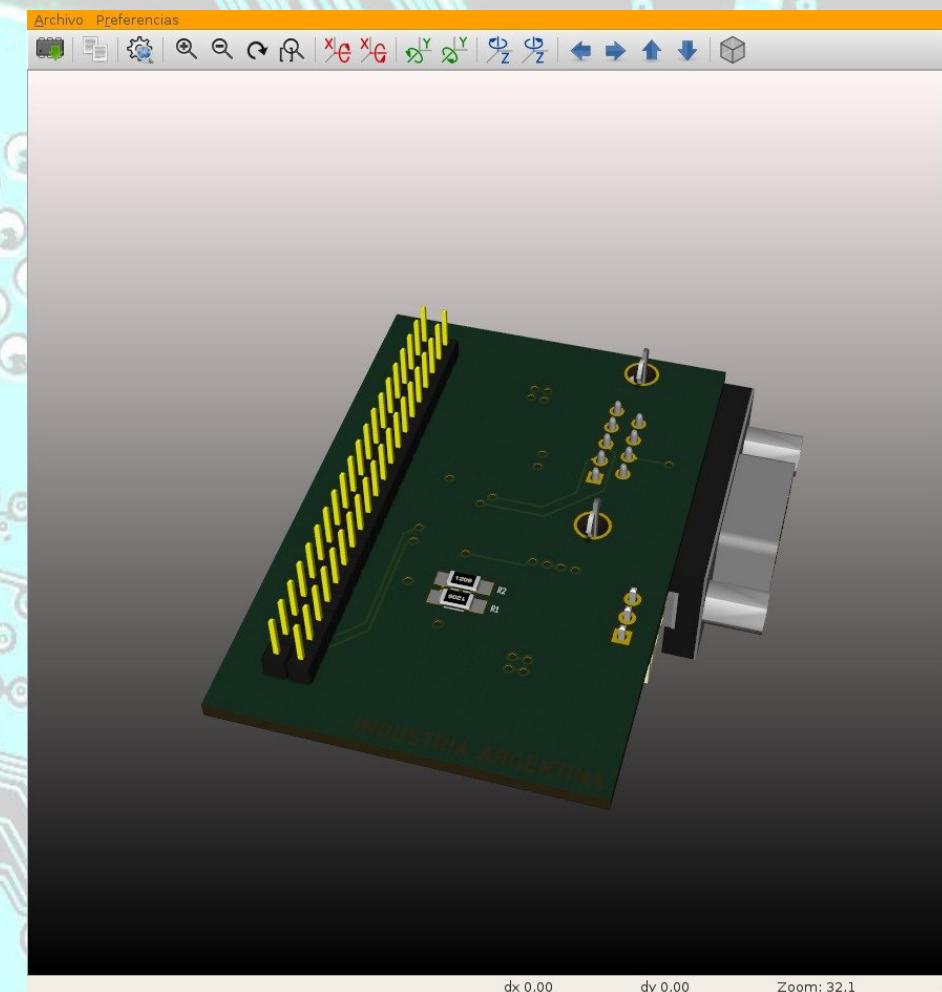
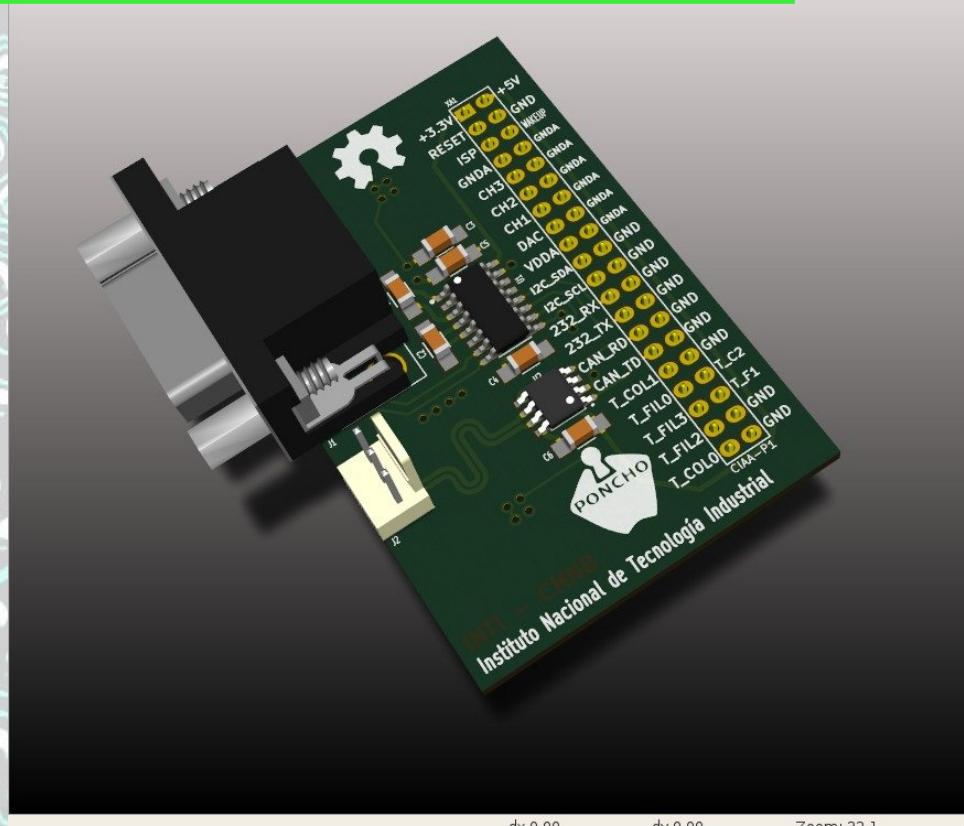
Tip: Para la escala probar con el factor 0.395!



Ejercicio 2 - Modelos 3D

Colocar los modelos 3D de nuestro diseño.

- 1)Copiar los modelos 3D provistos en el directorio ej2.3dshapes.
- 2)Vincular los componentes según las instrucciones previas.
- 3)Ver la vista 3D.
- 4)Actualizar el repositorio local y el remoto con nuestro diseño.



Ejercicio 2 - Verificaciones

Una vez terminado nuestro diseño, abrimos los ejercicios de los demás participantes.



- 1) Elegir compañero y abrir su proyecto.
- 2) Verificar correcta apertura del esquemático.
- 3) Verificar correcta apertura del PCB.
- 4) Verificar vista 3D.
- 5) Verificar estructura de directorios.

Imágenes de esta presentación

Carátula principal:

Foto titulada “Circuit” de Yuri Samoilov bajo licencia CC-BY disponible en
<https://www.flickr.com/photos/yusamoilov/14011462899/>

Fondo de la presentación:

Foto titulada “computer motherboard tracks” de Creativity103 bajo licencia CC-BY disponible en: https://www.flickr.com/photos/creative_stock/5228433146/

Las imágenes de clipart se tomaron de: <https://openclipart.org/>

El Logo INTI es de uso exclusivo del Instituto Nacional de Tecnología Industrial y debe removese en versiones derivadas.

Los demás logos corresponden a proyectos de Software Libre u Open Source. Consultar cada licencia en particular.

Todas las capturas de pantalla fueron realizadas por los autores y están bajo la misma licencia que esta presentación.

Las imágenes del circuito esquemático con un BUS y la interconexión de tres BUSES fueron obtenidas del manual de KiCad <http://docs.kicad-pcb.org/en/eeschema.html> que pose licencia CC BY 3.0.

El resto de las imágenes se cita la fuente debajo de cada una.





Nivel intermedio

Esta presentación posee licencia:
(CC BY-SA 4.0)

Attribution-ShareAlike 4.0 International

<https://creativecommons.org/licenses/by-sa/4.0/>



LibreOffice®



Autores: Diego Brengi, Noelia Scotti y Diego Alamon