

Electric VLSI Design System

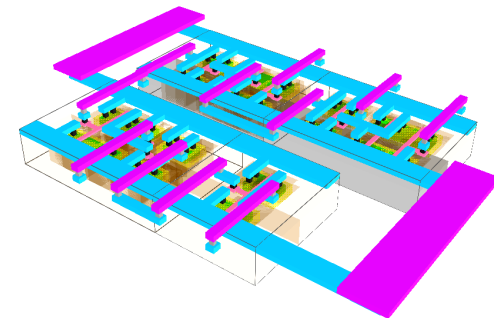
Liliana Arias Torres.

Walter Alberto Pulido Chiguasuque.

Director : Sebastian Eslava Garzón PhD.

Departamento de Ingeniería Eléctrica y Electrónica.

Universidad Nacional de Colombia



Instalación y ejecución del programa

Instalación de Electric

- Para obtener la ultima versión de Electric se descarga la versión *source* de Electric de la siguiente pagina:

<http://www.staticfreesoft.com/productsFree.html>

- La carpeta descargada es un empaquetado JAR por lo tanto es necesario tener instalado una versión de Java 1.6 o posteriores, para Ubuntu desde el centro de software se instala OpenJDK Java7.



Home
About
Products
Documentation
Contact Us



Static Free Software

Home of the Electric VLSI Design System

Products:

[Software](#)[Code](#)[Libraries](#)[Documentation](#)

Run Electric Now

You can run Electric instantly, just [click here](#). This uses Java Web Start, which downloads necessary files and then runs Electric. It takes longer to start when first run or when new releases of Electric are made available, but starts quickly at other times because the downloading is already done.

Download Source Code for Electric

That's right, Electric is free software! You can download Electric source code from the web.

There is no better way to get to know a CAD system than to use it for a while. Now you can use it at no charge! If you like it, keep it! If you don't, you've lost nothing.

Why are we doing this? Because we believe that software should not be proprietary, and we want everyone to be able to use Electric.

Download Java Source and Binary Code

The Java distribution consists of a single ".jar" file that will run on any platform with Java 1.6 or later (available from [Oracle](#) or [Apache Harmony](#)). You can get a ".jar" file with or without source code (the source version is larger). Both versions have the built-in user's manual, supporting libraries, and binary code, so they are ready to run.

The binary distribution is 19 megabytes in size, and the source distribution is 24 megabytes in size. For specific information about running Electric on your system, read the [setup manual page](#).

The distributions are located on the [Free Software Foundation \(GNU\)](#) web pages. You can read about downloading [GNU Electric](#) there, or you can...

GET THE GNU ELECTRIC BINARY RELEASE, [version 9.03](#)

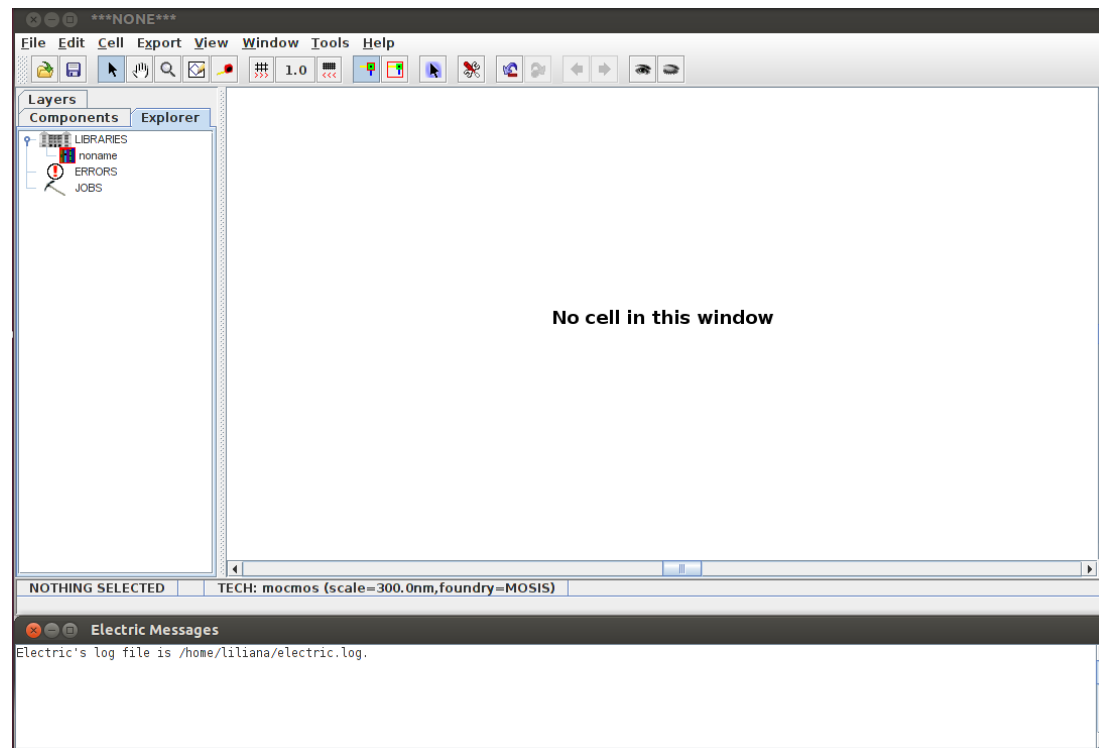
GET THE GNU ELECTRIC SOURCE RELEASE, [version 9.03](#)

...from the main GNU FTP area now. If this server is busy, check the list of [mirrors](#) and find Electric locally.

Due to copyright restrictions, GNU is unable to distribute those parts of Electric that come from external sources. Therefore, these facilities must be downloaded separately. After downloading GNU Electric distribution, you can download these extras (none of which are necessary):

Ejecución del programa

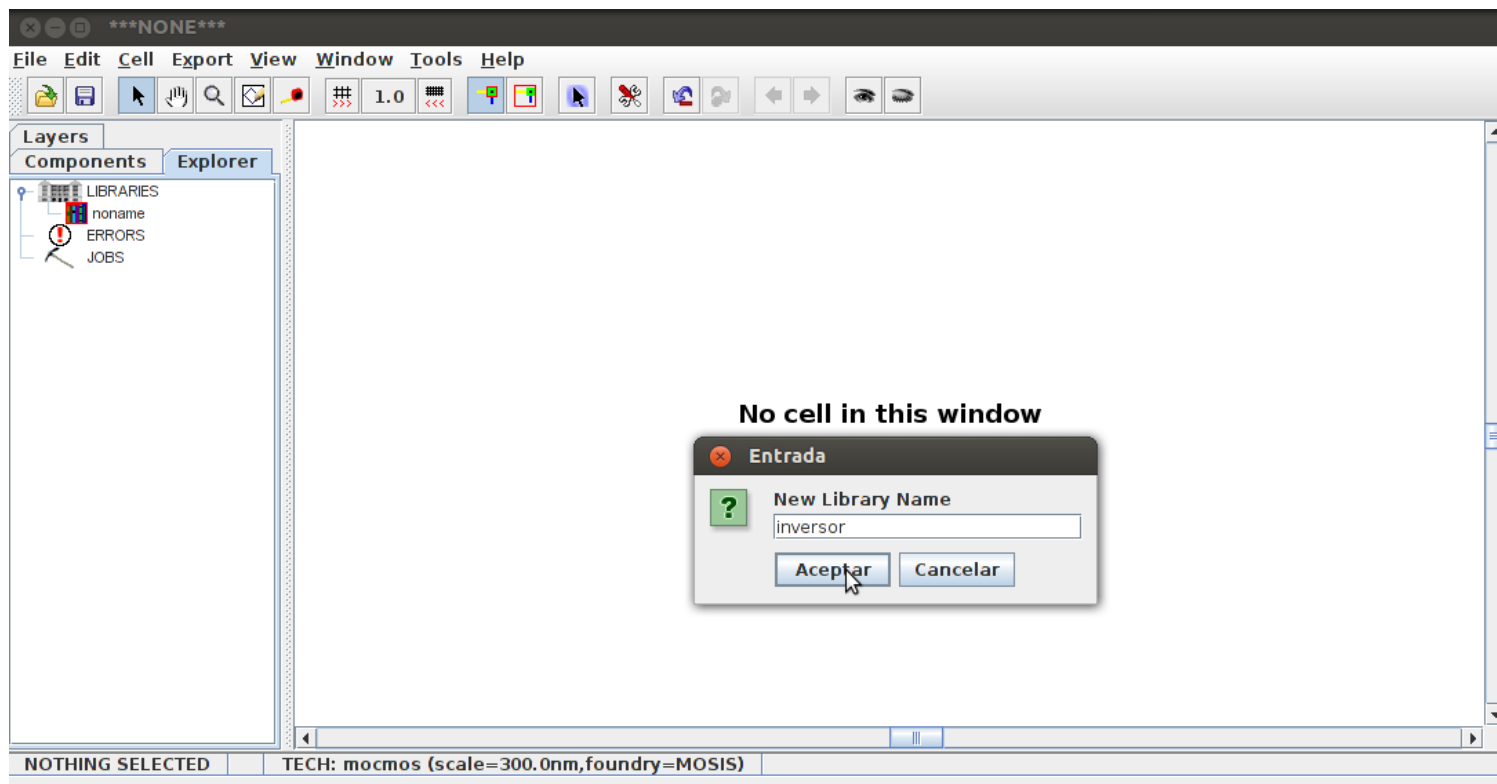
- Para ejecutar el .jar (debe estar ubicado en la carpeta personal) se escribe en la terminal la siguiente línea de comando: *sudo java-jar electric-9.03.jar*



Creación de un nuevo proyecto

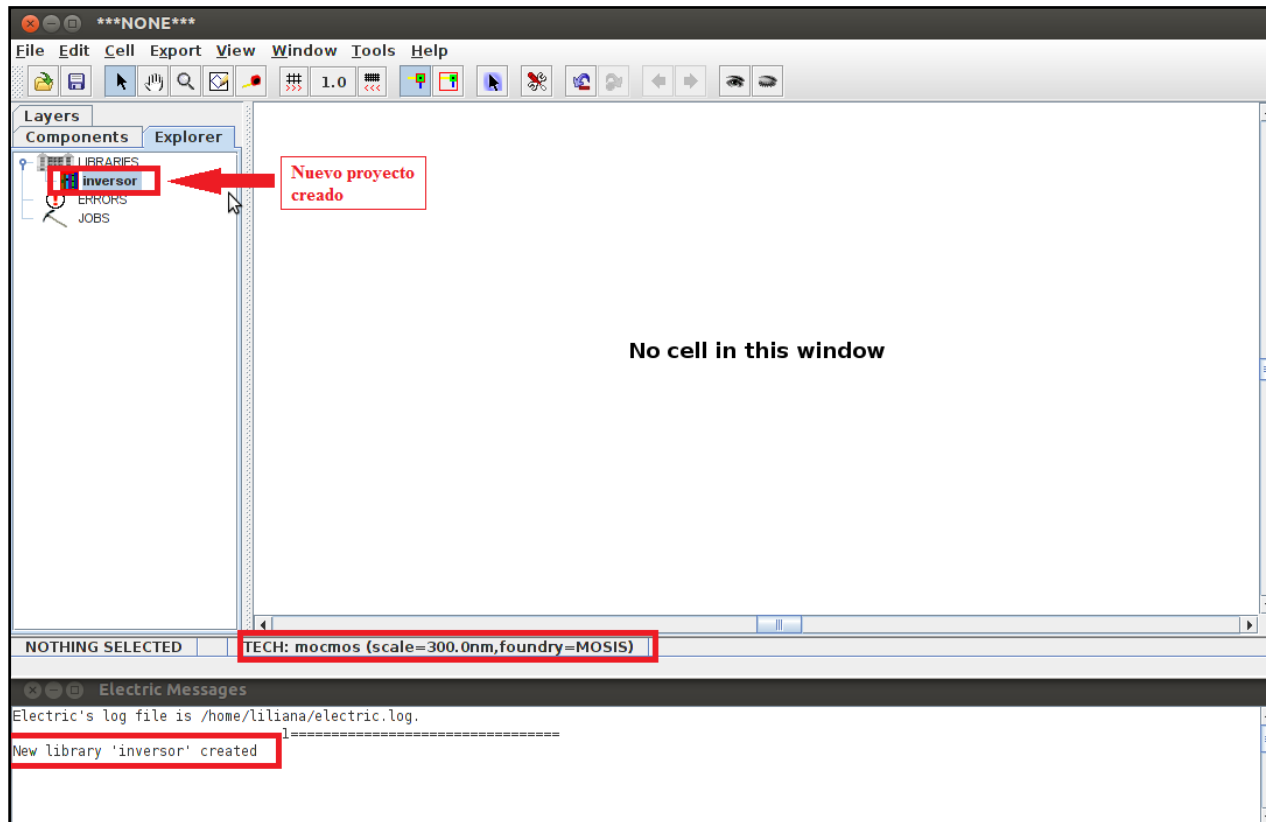
Creación de un nuevo proyecto

- Para crear un nuevo proyecto realizamos los siguientes pasos: **file**→**New Library**



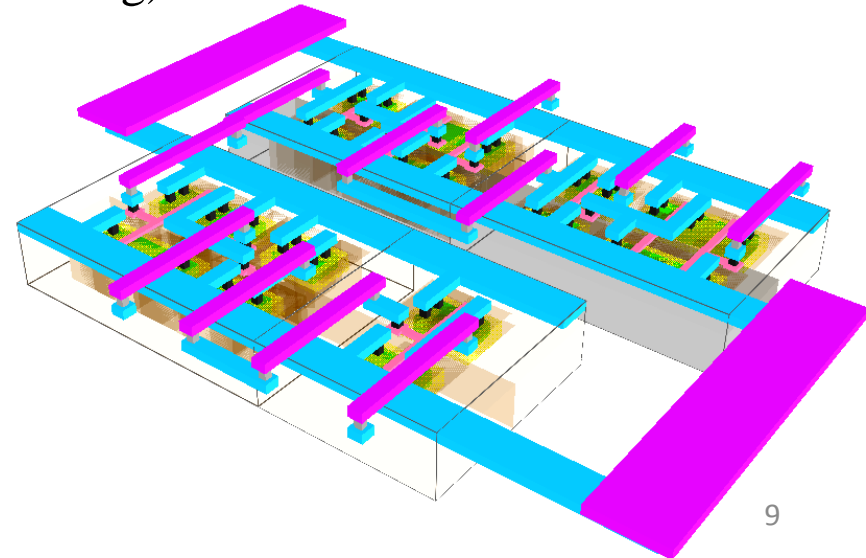
Creación de un nuevo proyecto

- Guardar el proyecto en: **file→Save as**
- La ventana se observará así:

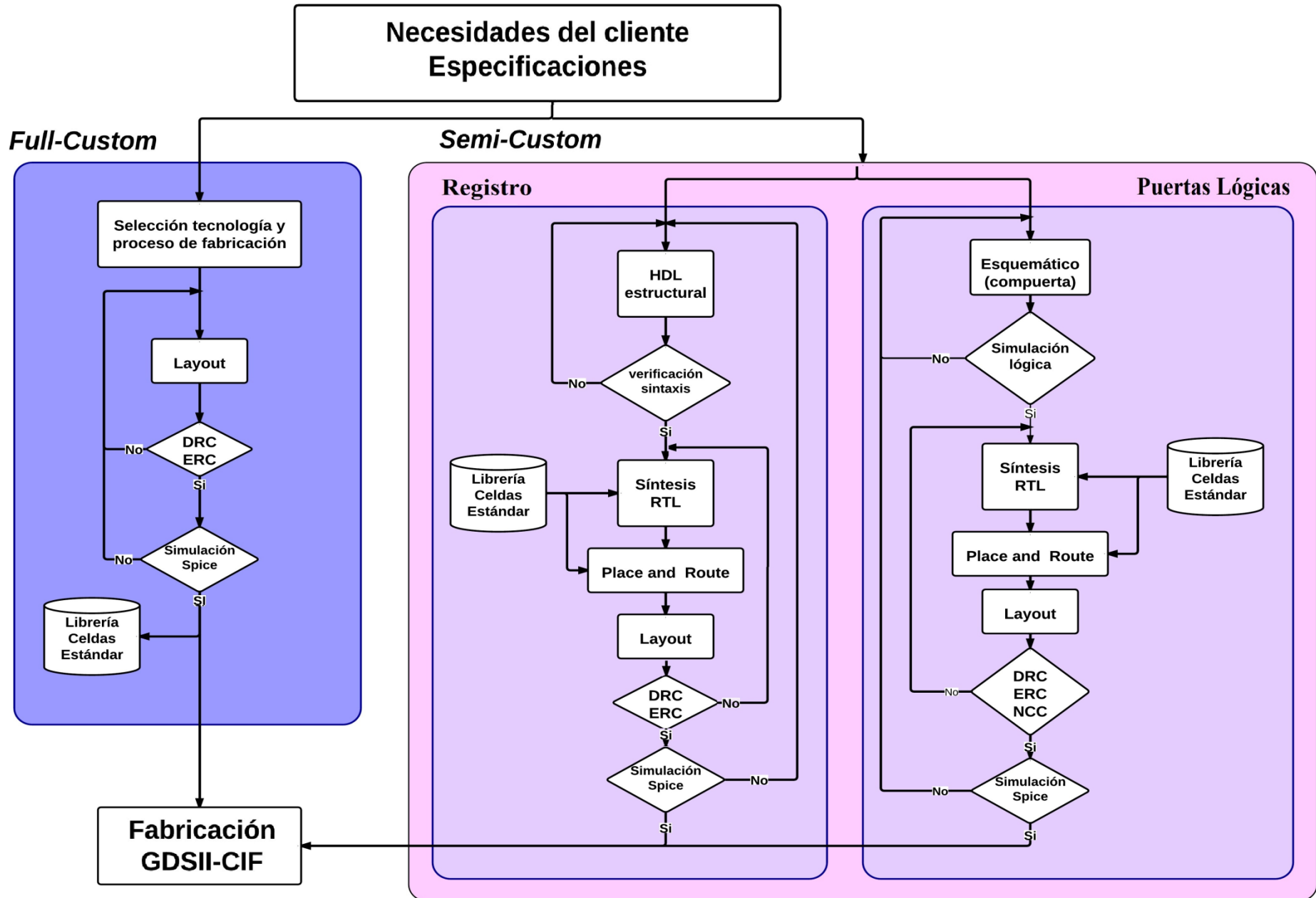


Características de Electric VLSI Design System

- Diseño:
 - Full-custom: nivel de transistor (layout)
 - Semi-custom: nivel de registro y puertas lógicas (HDL y esquemático)
- Verificación de diseño
 - DRC (Design Ruler Checker)
 - ERC (Electrical Ruler Checking)
 - NCC (Network Consistency Checking)
- Simulación:
 - Ltspice
- Archivos de extracción:
 - GDSII
 - CIF



Ambiente de diseño para Electric

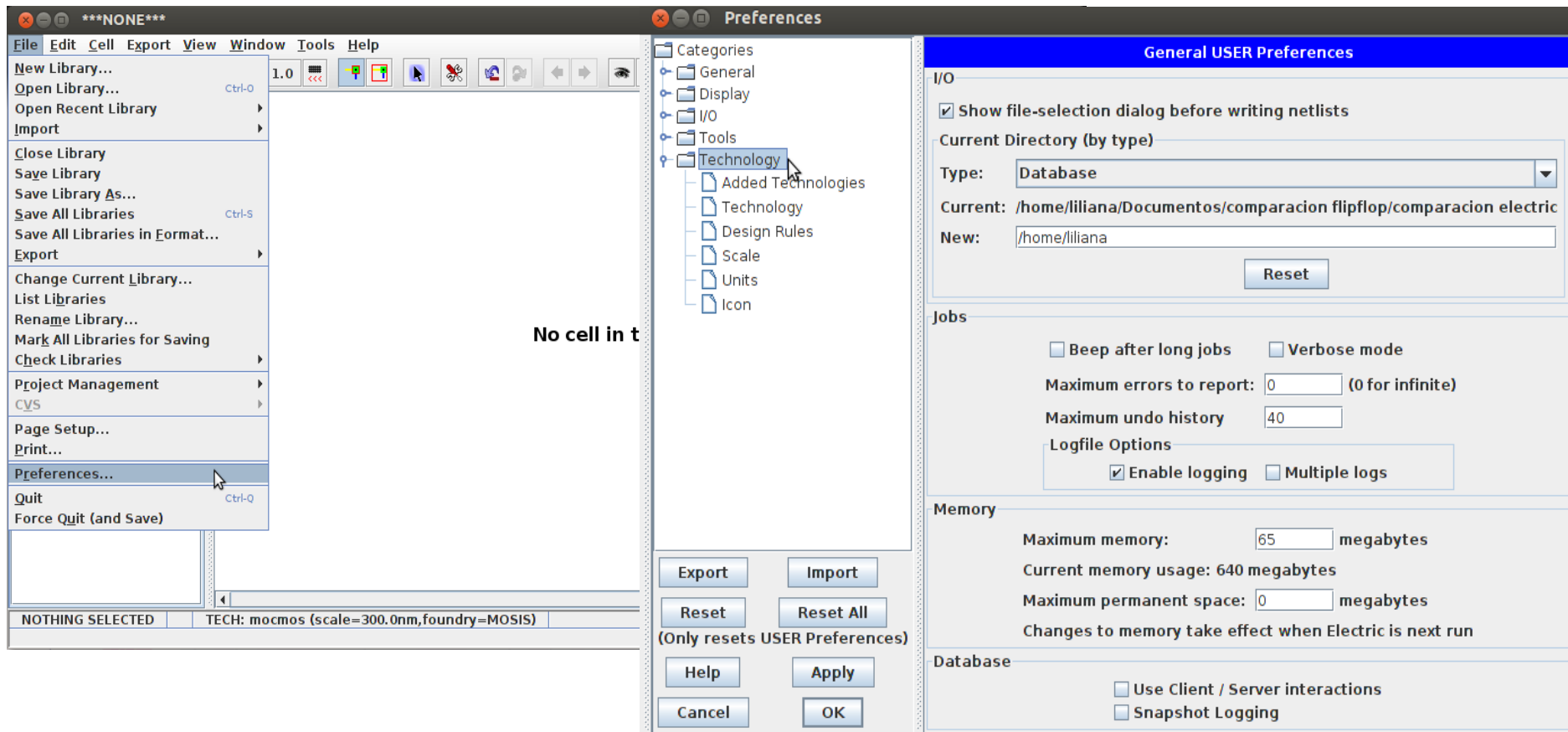


Diseño Full-Custom

Selección de tecnología

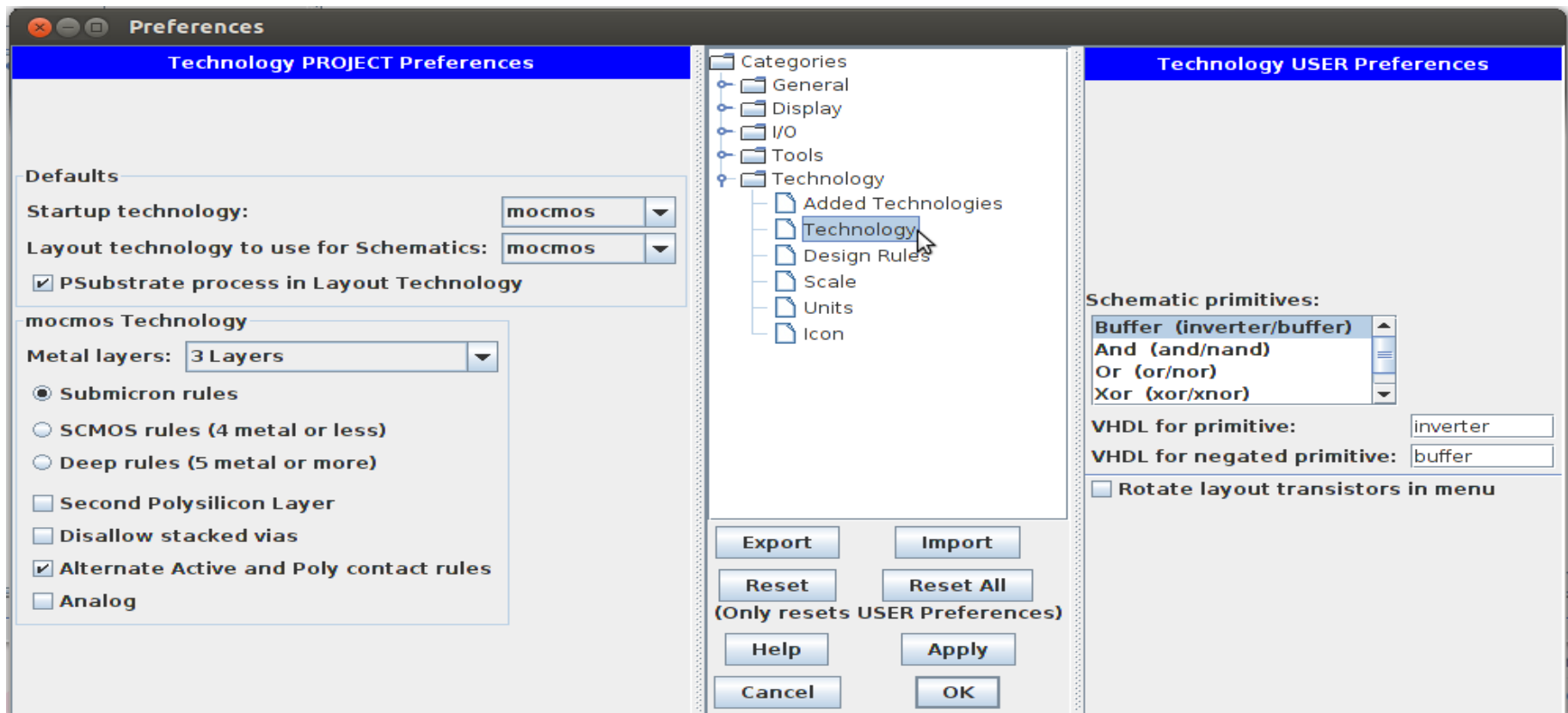
- Electric ofrece varias opciones para crear y escoger diferentes tecnologías para el diseño VLSI. Dentro de estas tenemos las reglas MOSIS las cuales son predeterminadas por la herramienta.
- Escogemos la tecnología mocos:
file → preferences → Technology

Selección de tecnología



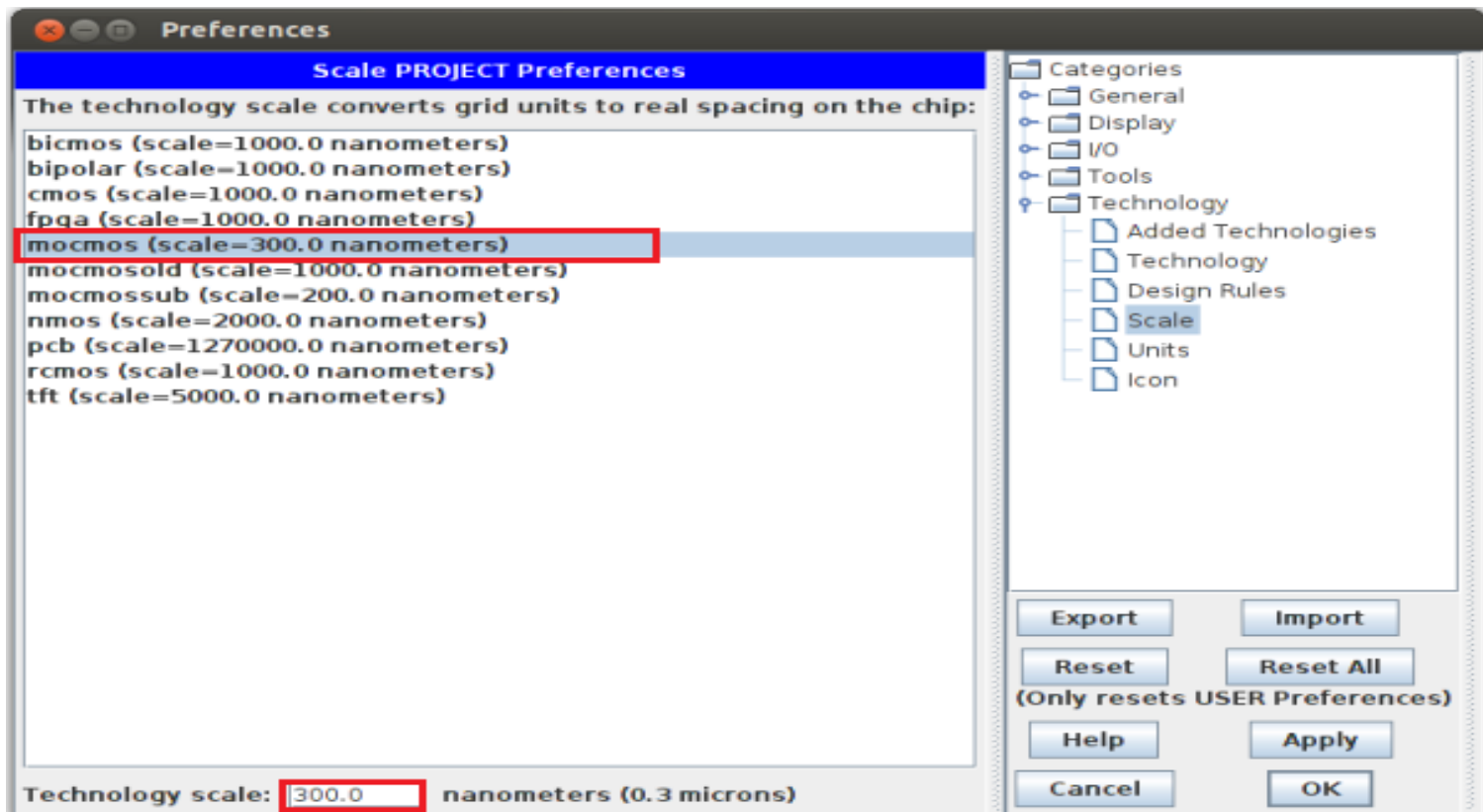
Selección de tecnología

- Escogemos la tecnología mocmos:
file → preferences → Technology → Technology



Selección de tecnología

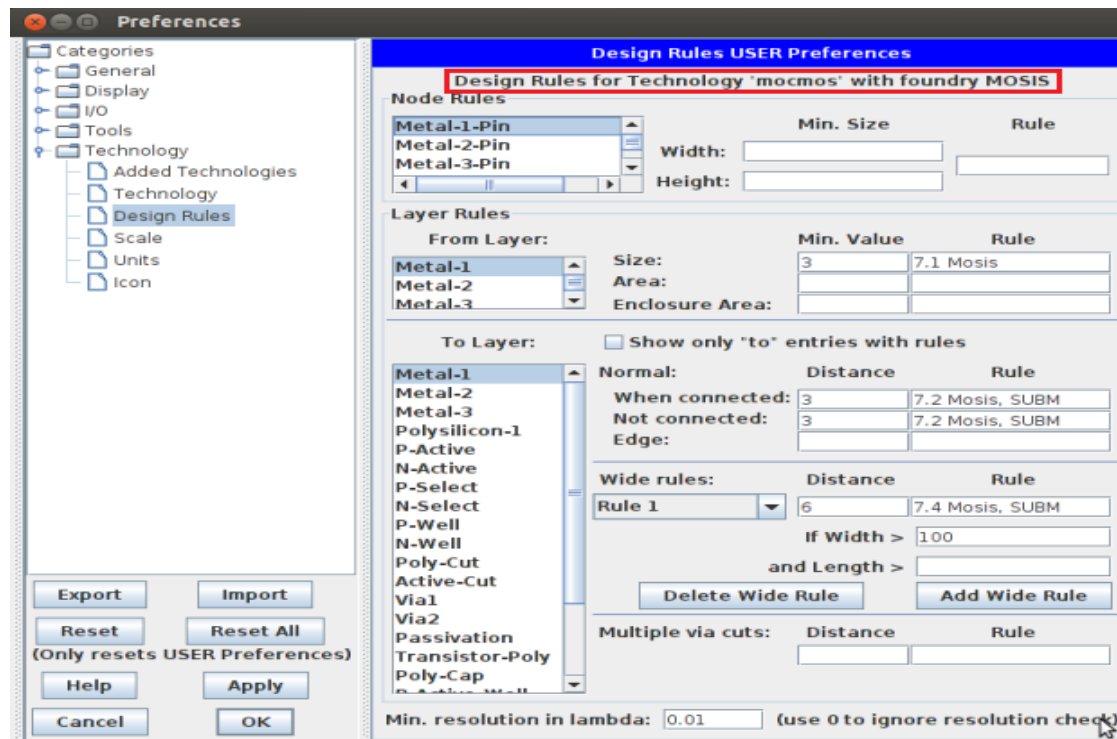
- Como ejemplo seleccionamos el proceso C5 de 0.5 μ m, entonces se debe escoger la escala en lambda de 300nm en: **file** \rightarrow **preferences** \rightarrow **Technology** \rightarrow **Scale**



Selección de tecnología

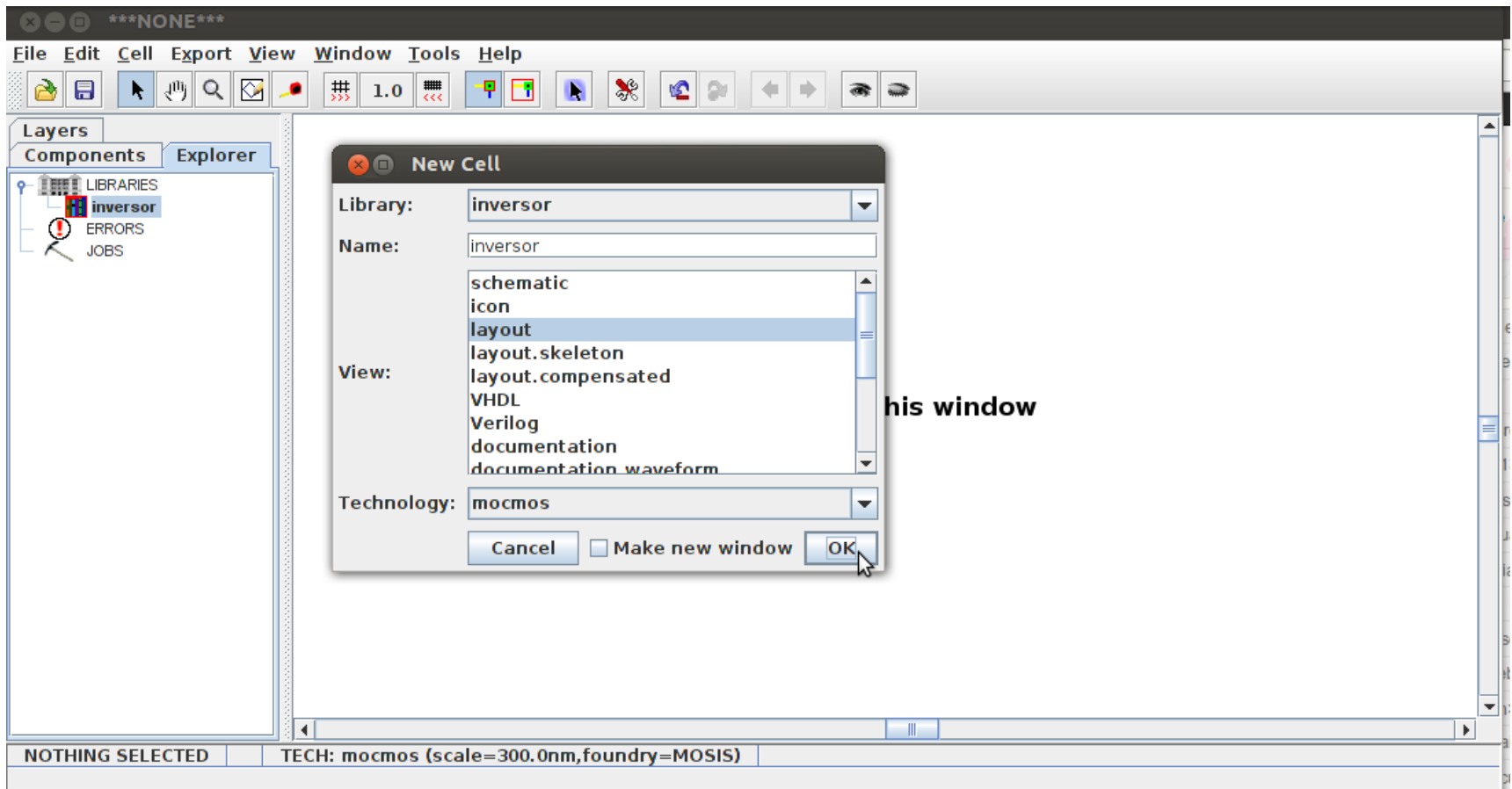
- Para observar cada una de las reglas de la tecnología escogida **MOSIS Scalable CMOS (SCMOS)** llamada mocmos en electric nos dirigimos a:

file → preferences → Technology → Design Rules



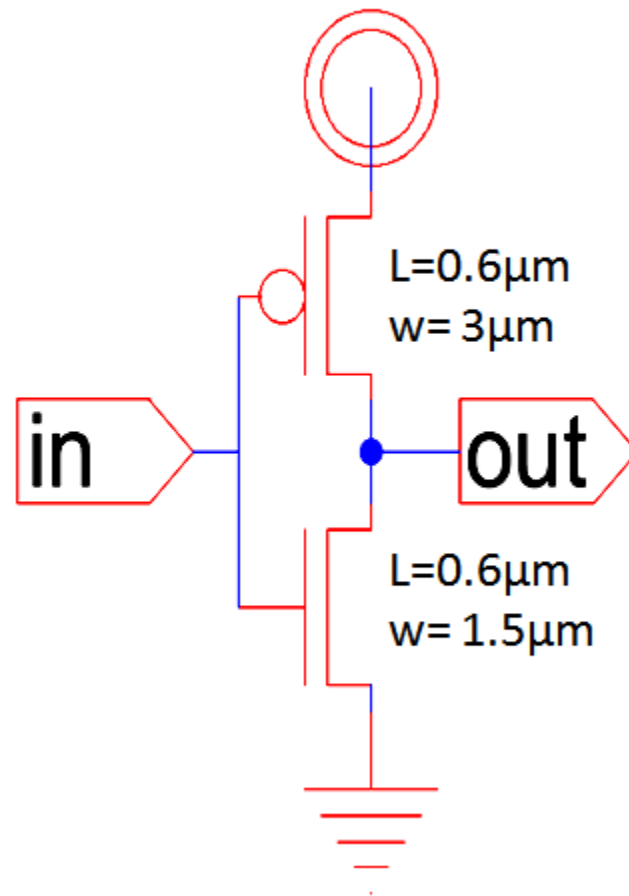
Creación del layout

- Para crear una nueva celda tipo layout nos dirigimos:
Cell→New Cell



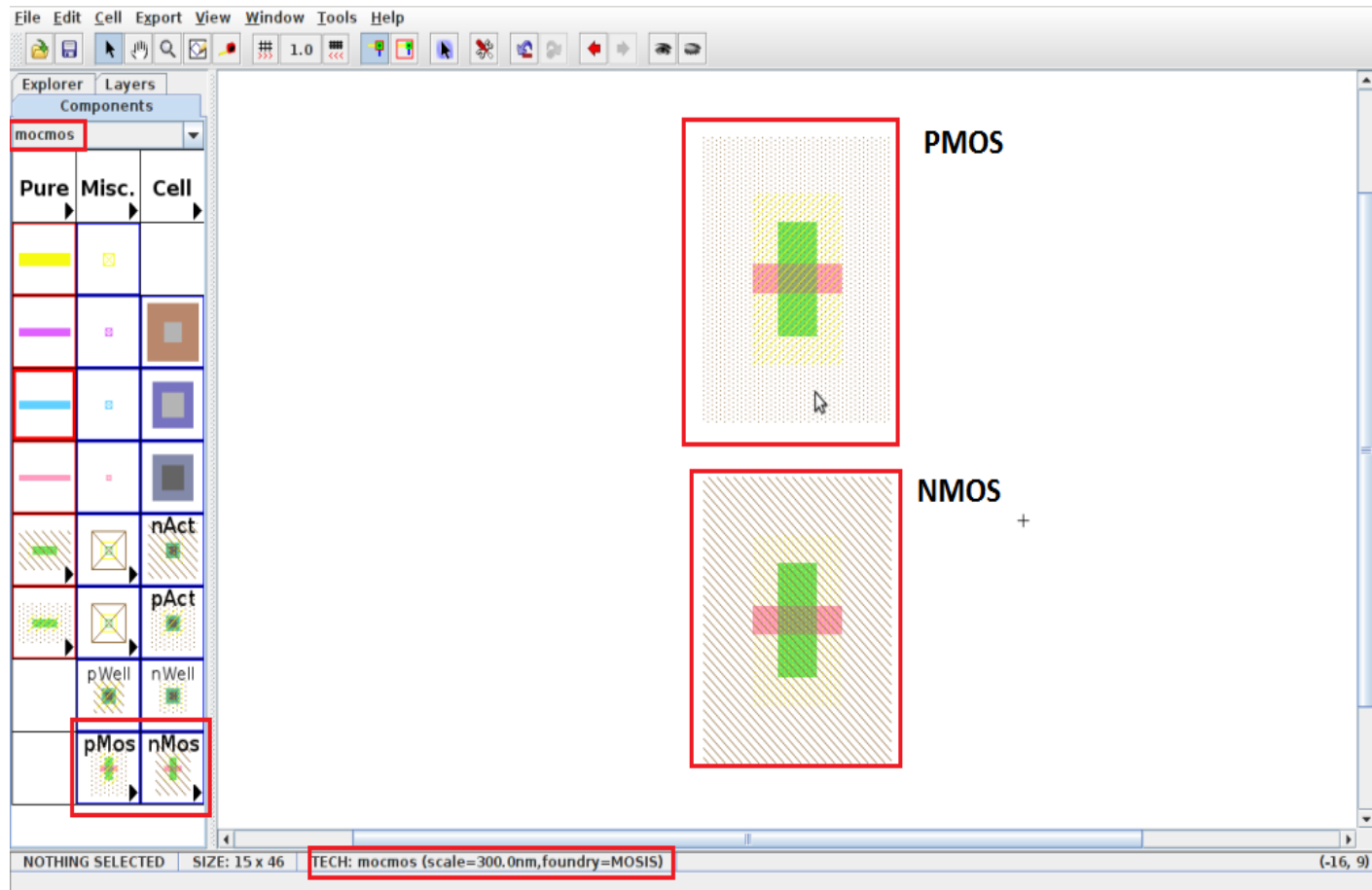
Creación del layout

- Vamos a crear un inversor cuyos transistores tengan las siguientes propiedades:



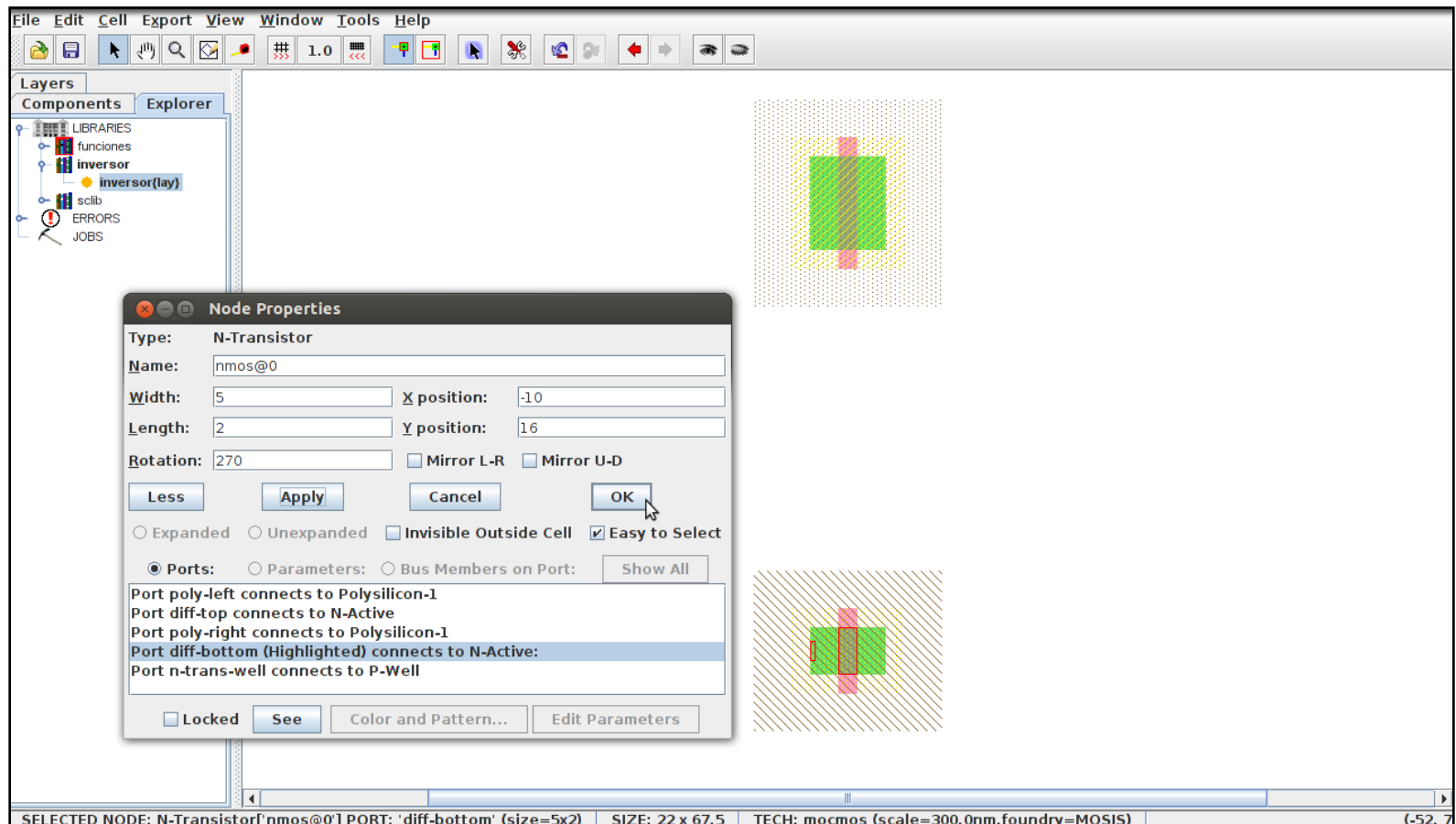
Creación del layout

- Para crear el inversor vamos a Components y seleccionamos los transistores Pmos y Nmos



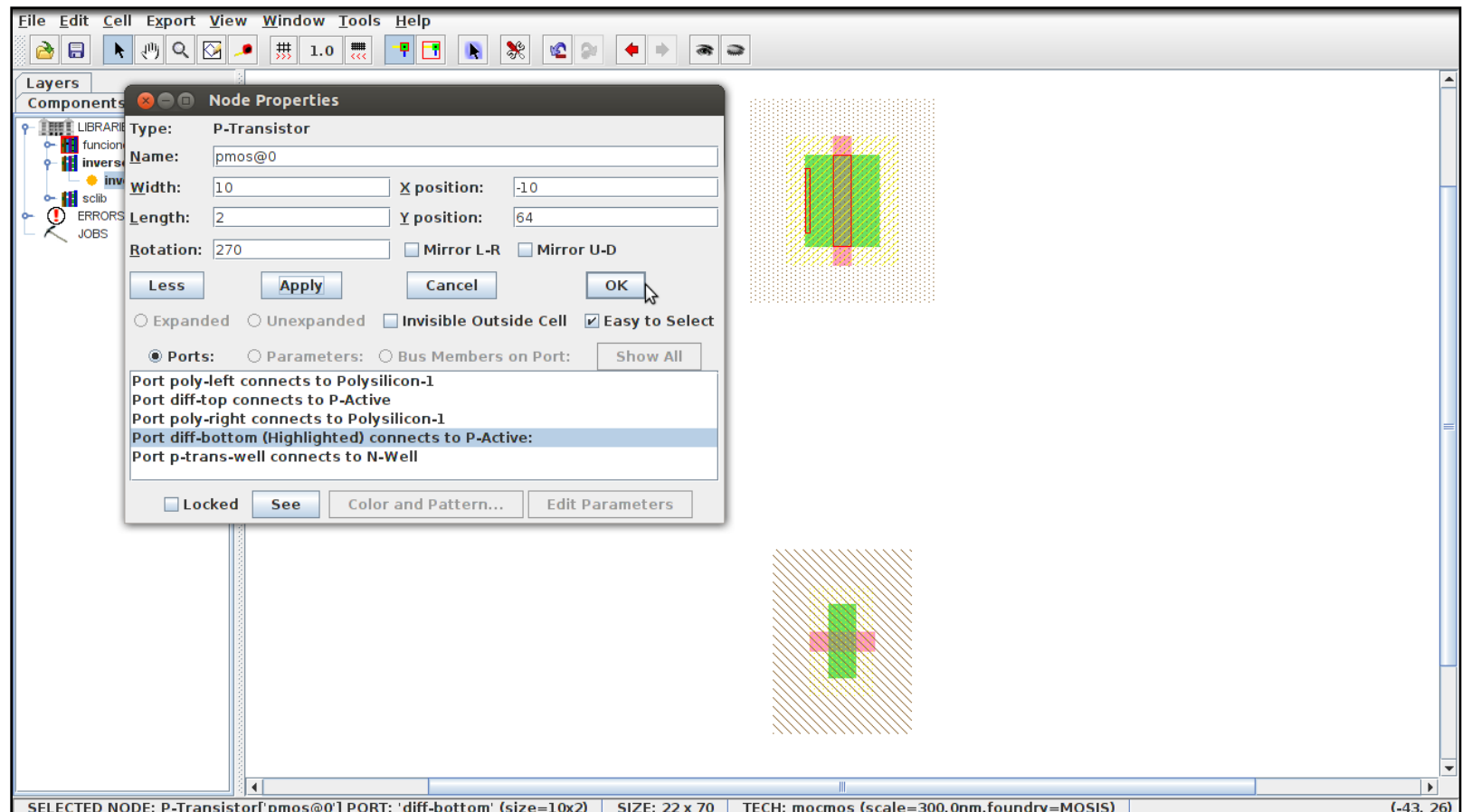
Creación del layout

- Para editar las propiedades del transistor Nmos se selecciona y pulsando Ctrl+I se asignan W=5 y L=2. (nota: la escala es de 300 nm)



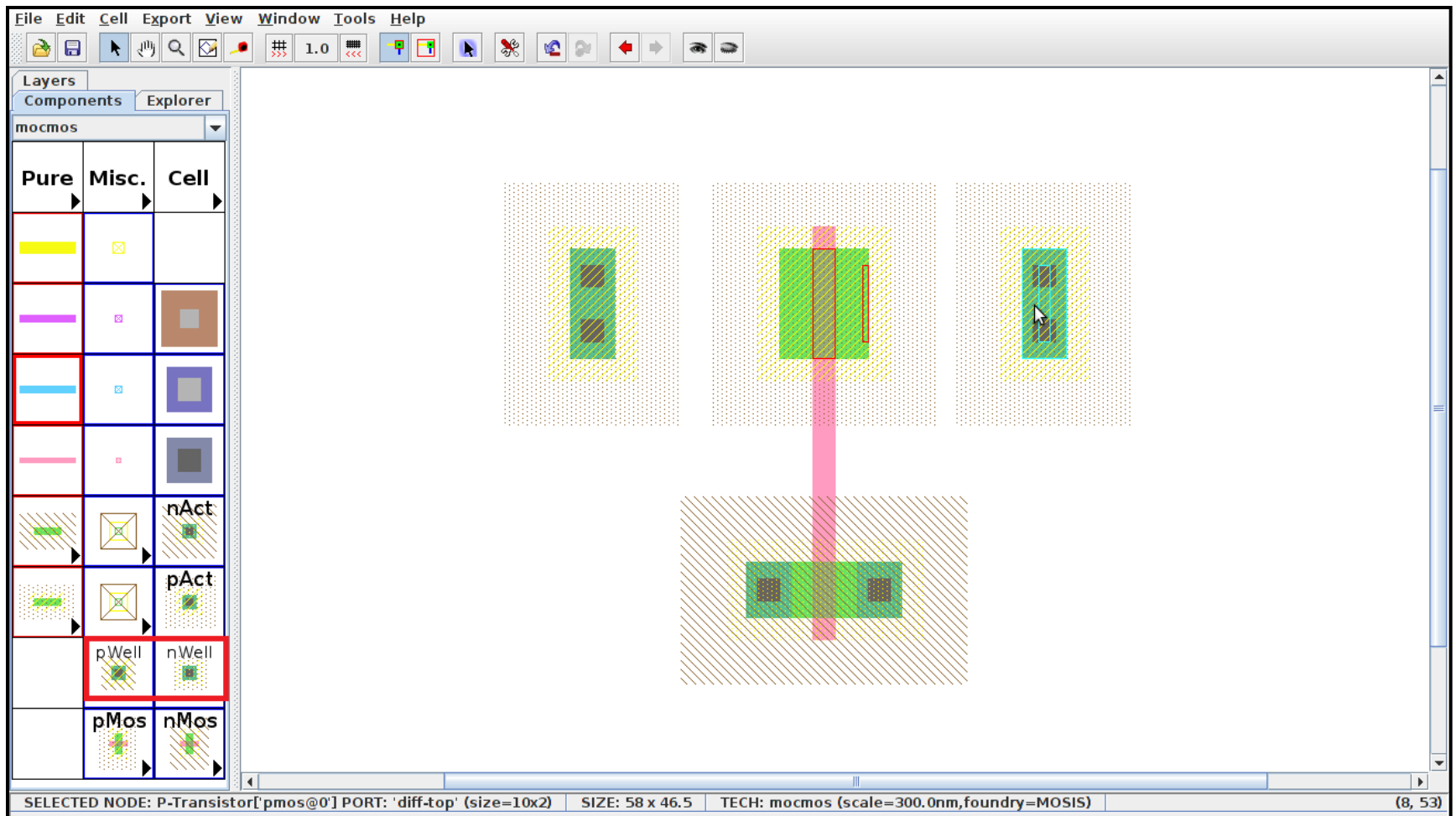
Creación del layout

- Para editar las propiedades del transistor Pmos se selecciona y pulsando Ctrl+I se asignan W=10 y L=2. (nota: la escala es de 300 nm)



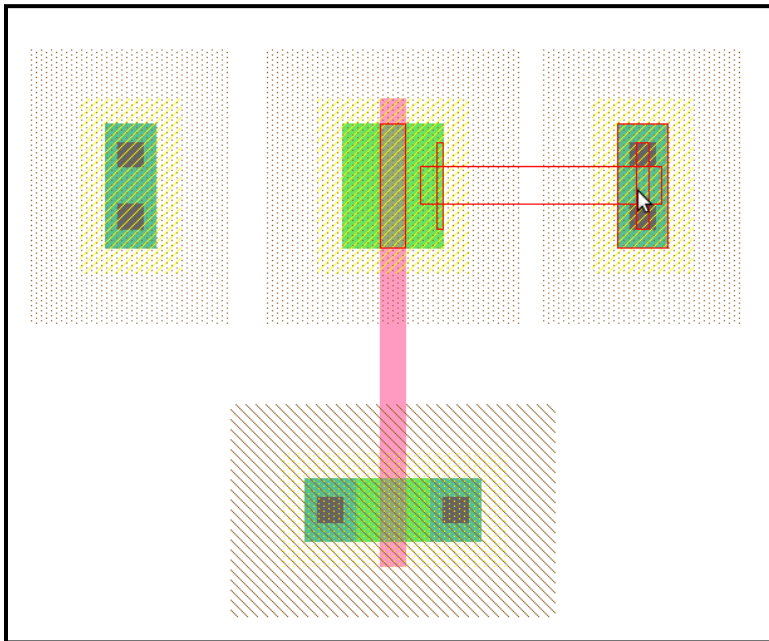
Creación del layout

- Una vez editados los transistores Nmos y Pmos procedemos a ubicar los contactos, para Nmos con pWell y Pmos con nWell

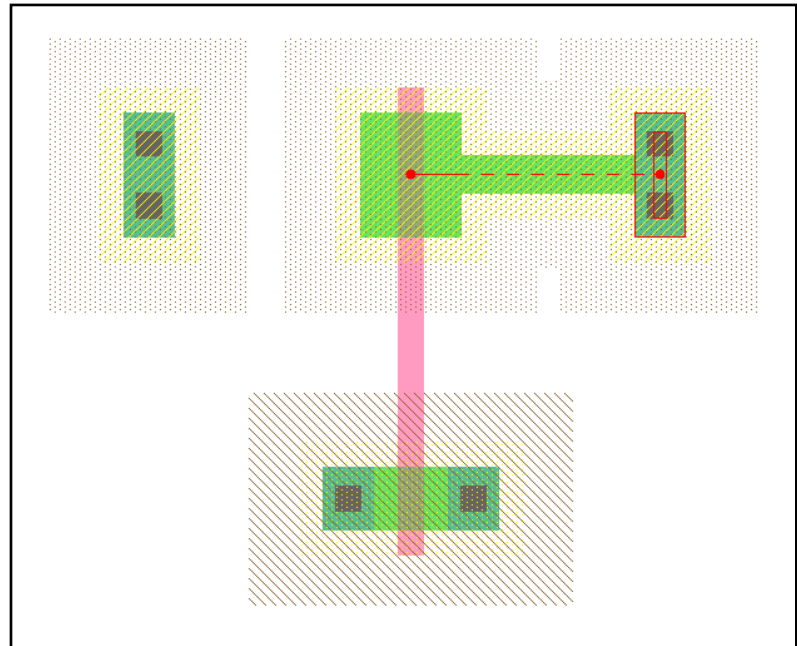


Creación del layout

- Para unir los contactos al transistor simplemente le damos un click derecho al transistor (1) y luego un click izquierdo al contacto (2).



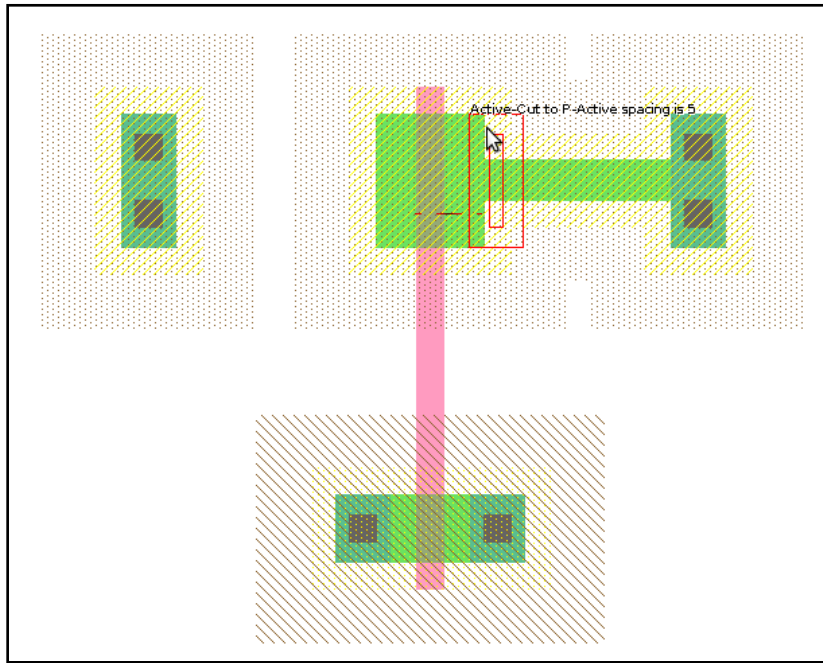
1.



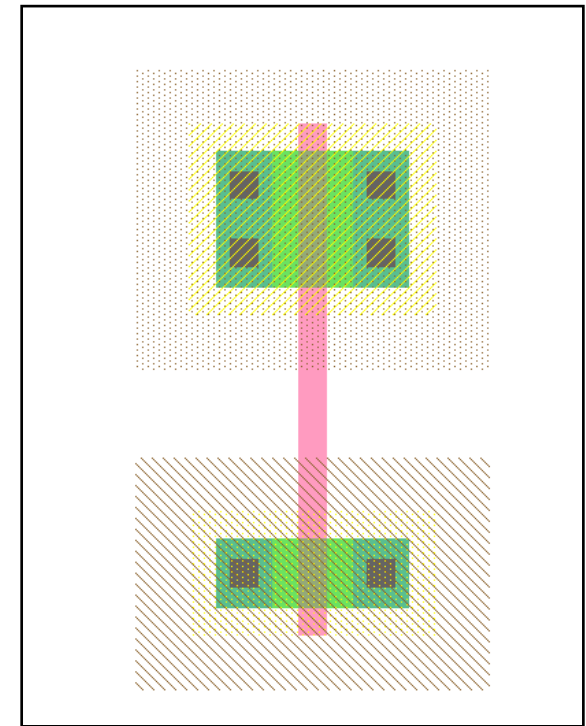
2.

Creación del layout

- Una vez unidos los pozos (3) se aproxima el contacto al transistor evidenciando que no se presente error alguno (4).



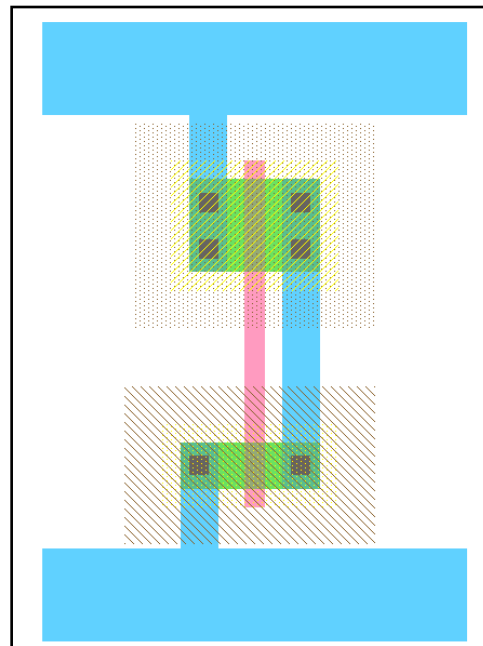
3.



4.

Creación del layout

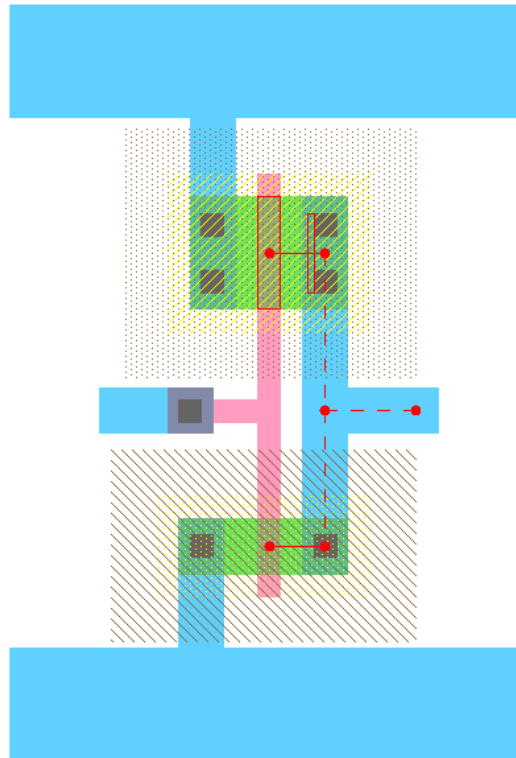
Creamos los barrajes de VDD en la parte superior y GND en la parte inferior, luego unimos el contacto izquierdo del Pmos a VDD y el contacto izquierdo del Nmos a GND y los contactos derechos de los transistores entre si. (Nota: revisar periódicamente el DRC con F5 o desde **Tools** → **DRC** → **Check Hierarchically**)



Creación del layout

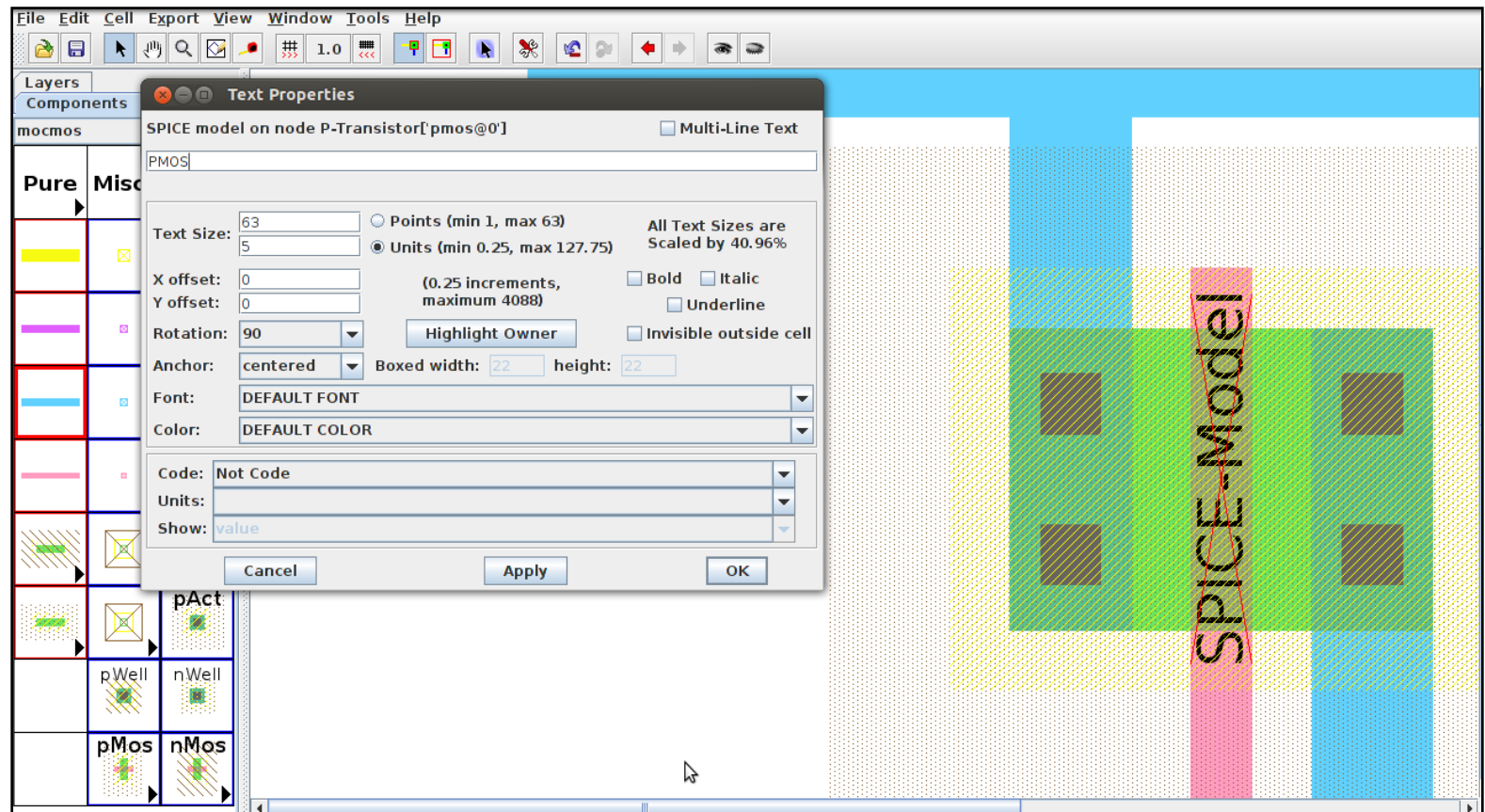
- Para poder simular es necesario brindar atributos de modelo spice a los transistores para esto seleccionamos el transistor y nos dirigimos a :

Tools→Simulation (Spice)→Set Spice Model



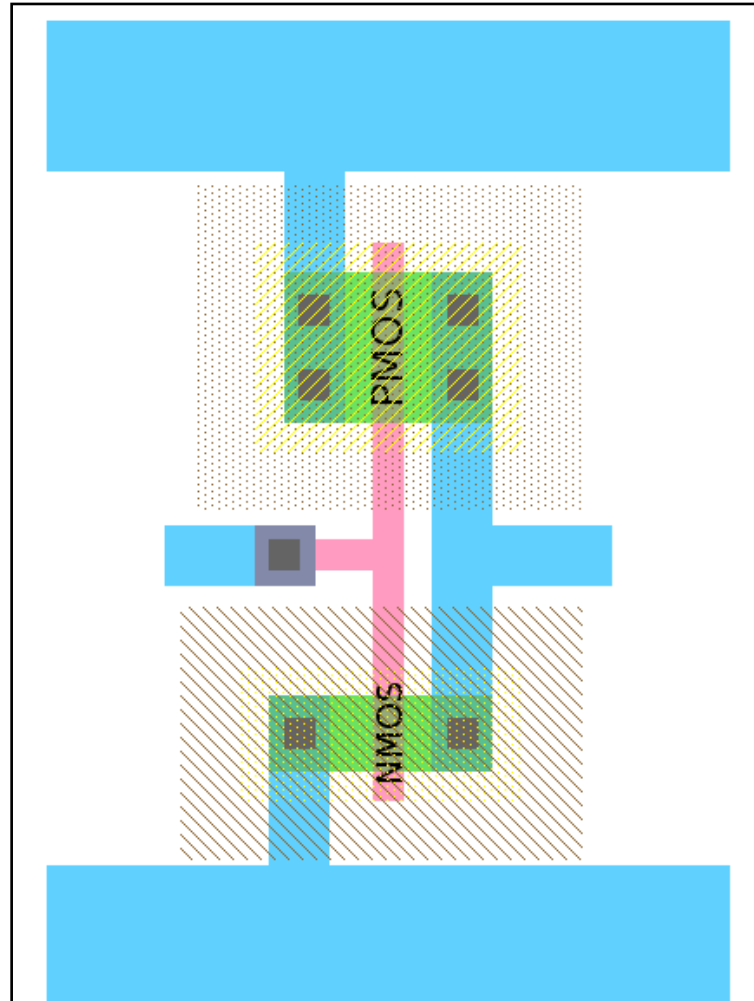
Creación del layout

- Aparecerá sobre el transistor una leyenda SPICE-Model la seleccionamos y oprimimos Ctrl+I para editar el nombre, le colocamos PMOS o NMOS según sea el caso.



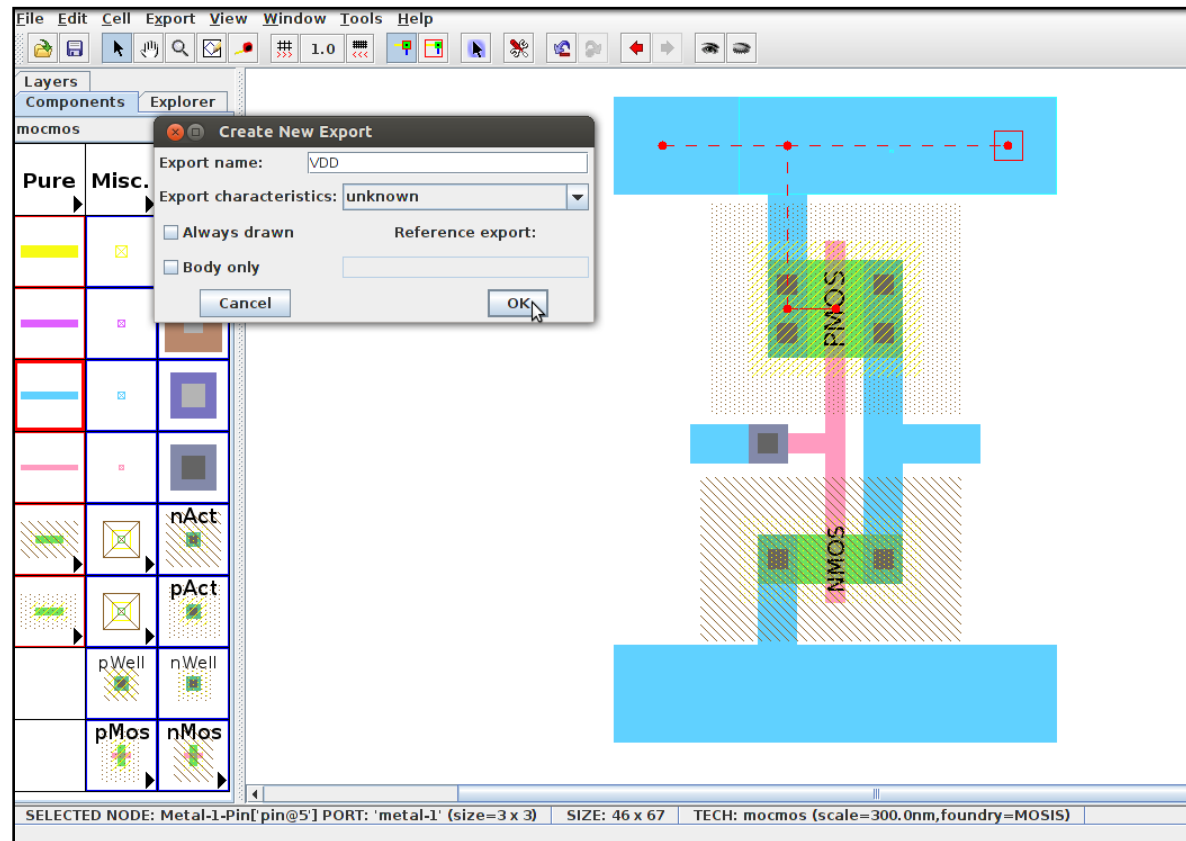
Creación del layout

- El inversor con el modelo spice tendrá el siguiente aspecto:



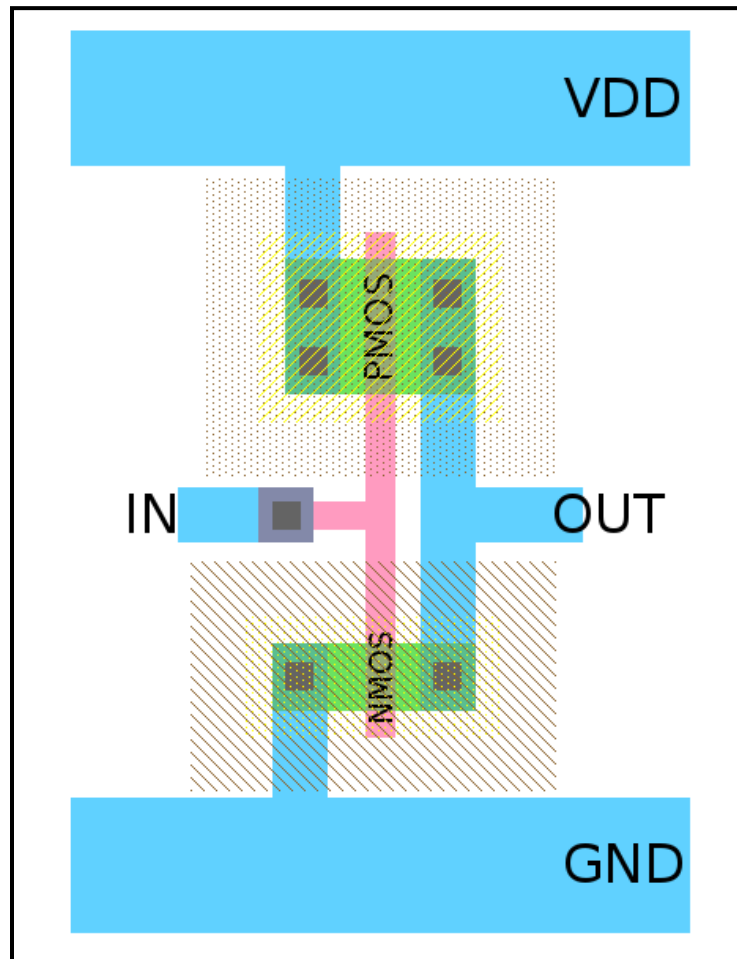
Creación del layout

- Para nombrar los puertos de la celda nos ubicamos sobre el extremo donde queremos que quede el puerto, oprimimos Ctrl+E y nombramos el puerto.



Creación del layout

- El inversor con los puertos nombrados se presenta a continuación.



Simulación Spice

- Para simular se requiere adicionar un código de simulación spice para esto vamos a *Components* y en **Misc.** → **spice code**. A continuación se muestra el código de simulación.

```
vdd VDD 0 DC 5  
vin IN 0 dc 0 pulse 0 5 5m 10n 10n 5m 10m  
.tran 0 100m  
.include /home/liliana/Documentos/ejem-  
electric/C5_models.txt
```

Simulación Spice

- Se requiere incluir un archivo .txt con el modelo Spice de los transistores de la tecnología escogida. (Nota: este archivo se adjunta a la presentación)

```
C5_models.txt *
* BSIM3 models for AMI Semiconductor's C5 process
*
* Don't forget the .options scale=300nm if using drawn lengths
* and the MOSIS SUBM design rules
*
* 2<Ldrawn<500 10<Wdrawn<10000 Vdd=5V
* Note minimum L is 0.6 um while minimum W is 3 um
* Change to level=49 when using HSPICE

.MODEL NMOS NMOS (
+VERSION = 3.1          TNOM    = 27          LEVEL   = 8
+XJ      = 1.5E-7       NCH     = 1.7E17      TOX      = 1.39E-8
+K1      = 0.8351612    K2      = -0.0839158  VTH0     = 0.6696061
+K3B     = -7.6841108   W0      = 1E-8        K3       = 23.1023856
+DVT0W   = 0           DVT1W   = 0          NLX      = 1E-9
+DVT0     = 2.9047241   DVT1    = 0.4302695   DVT2W    = 0
+U0      = 458.439679   UA      = 1E-13      DVT2     = -0.134857
+UC      = 1.629939E-11 VSAT    = 1.643993E5    UB       = 1.485499E-18
+AGS     = 0.1194608    B0      = 2.674756E-6   A0       = 0.6103537
+KETA    = -2.640681E-3 A1      = 8.219585E-5   B1       = 5E-6
+RDSW    = 1.387108E3  PRWG   = 0.0299916   A2       = 0.3564792
+WR      = 1           WINT    = 2.472348E-7  PRWB     = 0.0363981
+XL      = 0           XW      = 0          LINT     = 3.597605E-8
+DWB     = 5.306586E-8 VOFF    = 0          DWG      = -1.287163E-8
+CIT      = 0          CDSC    = 2.4E-4      NFACTOR  = 0.8365585
+CDSCB   = 0          ETA0    = 0.0246738   CDSCD    = 0
+DSUB    = 0.2543458   PCLM   = 2.5945188   ETAB     = -1.406123E-3
+PDIBLC2 = 2.311743E-3 PDIBLCB = -0.0272914  PDIBLC1  = -0.4282336
+PSCBE1  = 5.598623E8  PSCBE2 = 5.461645E-5  DROUT    = 0.7283566
+DELTA   = 0.01       RSH     = 81.8        PVAG     = 0
+PRT      = 8.621      UTE     = -1          MOBMOD   = 1
+KT1L    = -2.58E-9   KT2     = 0          KT1      = -0.2501
+UB1     = -4.8E-19   UC1     = -7.5E-11   UA1     = 5.4E-10
+WL      = 0          WLN     = 1          AT       = 1E5
+WWN     = 1          WWL     = 0          WW       = 0
+LLN     = 1          LW      = 0          LL      = 0
+LLN     = 1          LW      = 0          LLN     = 1
```


Simulación Spice

- Necesitamos generar el archivo de simulación *.spi* para esto nos dirigimos a:

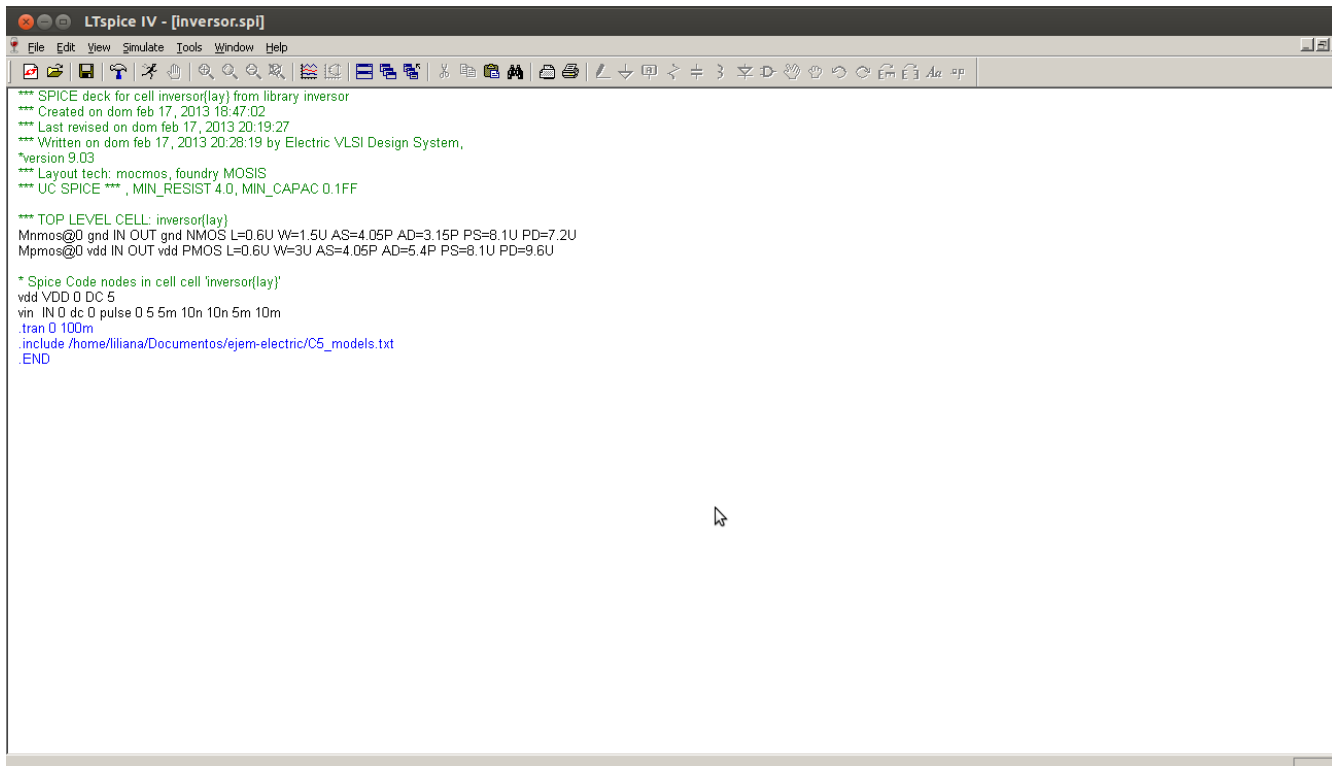
Tools→Simulation (Spice)→Writte Spice Model

- Para simular utilizaremos LTspice IV, es una herramienta libre que funciona sobre windows, entonces debemos instalar *wine* y luego descargar el simulador desde la página:

<http://www.linear.com/designtools/software/>

Simulación Spice

- Luego de abrir LTspice nos dirigimos a la carpeta donde guardamos el proyecto y abrimos el archivo *.spi* que fue generado por Electric en este caso *inversor.spi*



The screenshot shows the LTspice IV window titled "LTspice IV - [inversor.spi]". The interface includes a menu bar (File, Edit, View, Simulate, Tools, Window, Help) and a toolbar with various simulation and editing icons. The main text area displays the SPICE deck for a cell named 'inversor'.

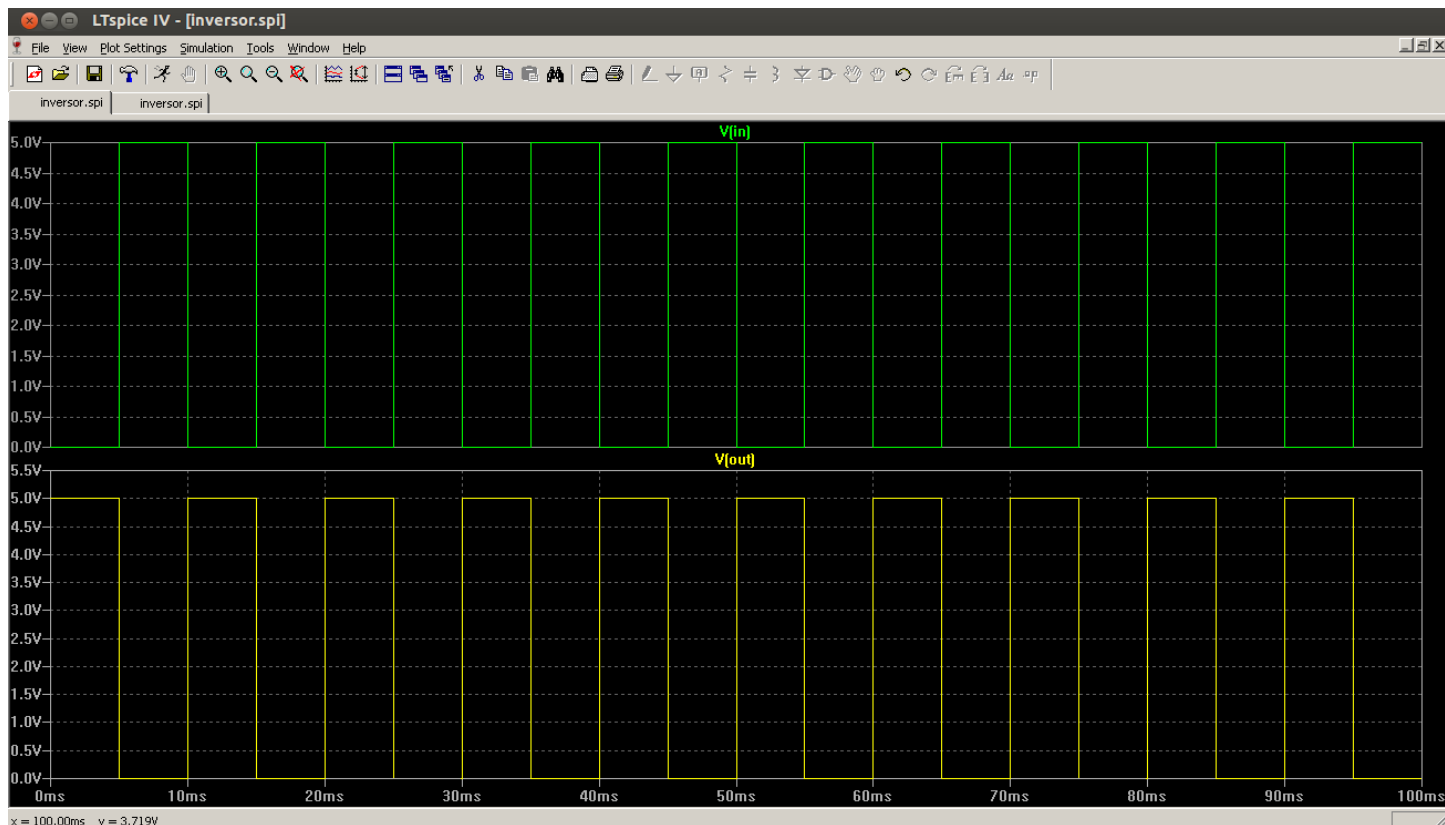
```
*** SPICE deck for cell inversor[lay] from library inversor
*** Created on dom feb 17, 2013 18:47:02
*** Last revised on dom feb 17, 2013 20:19:27
*** Written on dom feb 17, 2013 20:28:19 by Electric VLSI Design System,
*version 9.03
*** Layout tech: mcmos, foundry MOSIS
*** UC SPICE *** , MIN_RESIST 4.0, MIN_CAPAC 0.1FF

*** TOP LEVEL CELL: inversor[lay]
Mnmos@0 gnd IN OUT gnd NMOS L=0.6U W=1.5U AS=4.05P AD=3.15P PS=8.1U PD=7.2U
Mpmos@0 vdd IN OUT vdd PMOS L=0.6U W=3U AS=4.05P AD=5.4P PS=8.1U PD=9.6U

* Spice Code nodes in cell cell 'inversor[lay]'
vdd VDD 0 DC 5
vin IN 0 dc 0 pulse 0 5 5m 10n 10n 5m 10m
.tran 0 100m
.include /home/liliana/Documentos/ejem-electric/C5_models.txt
.END
```

Simulación Spice

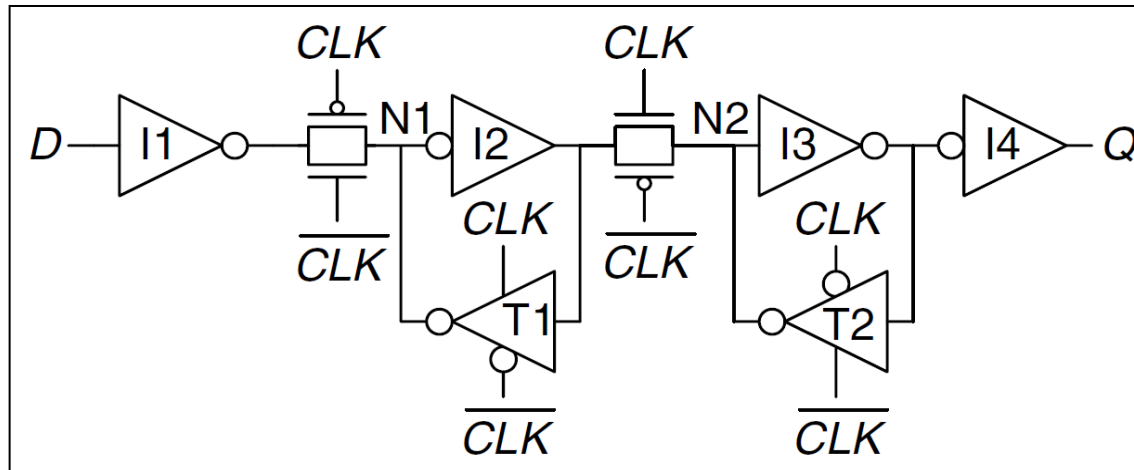
Corremos la simulación y adicionamos las señales de interés, en este caso $V(in)$ y $V(out)$ donde se puede observar que el inversor esta funcionando.



Diseño Semi-Custom Registro

HDL estructural

- Como ejemplo realizaremos un flip-flop D como se muestra en el siguiente esquemático, adicionando buffer a las entradas y salidas.

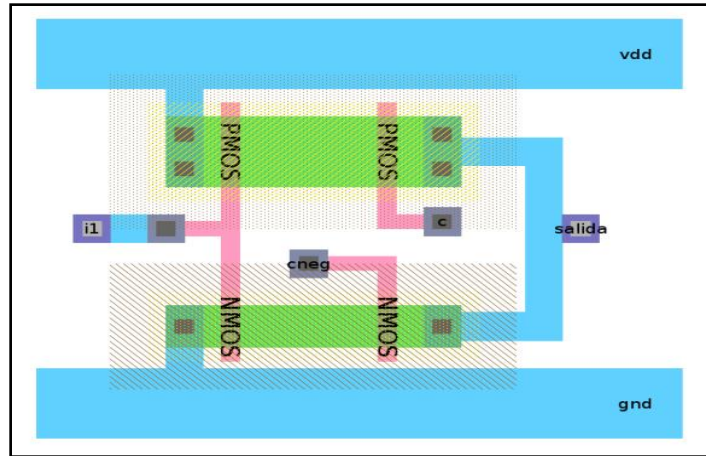


Fuente: David Money Harris and Sarah L. Harris. Digital Design and Computer Architecture

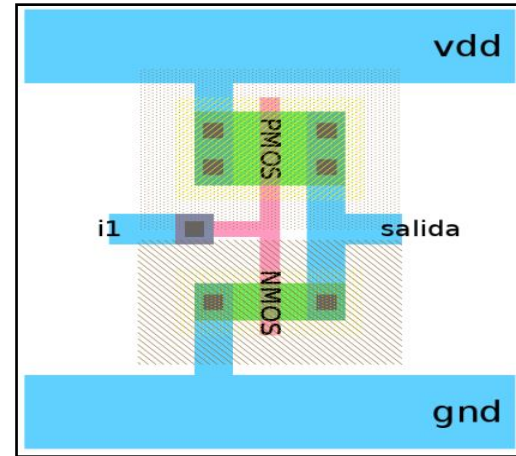
HDL estructural

- El primer paso es tener una librería de celdas estándar con las compuertas necesarias, dicha librería se crea con el método full-custom que ya conocemos.
- Las alturas de la celdas y de los barrages de VDD y GND de deben ser iguales para todas las celdas.

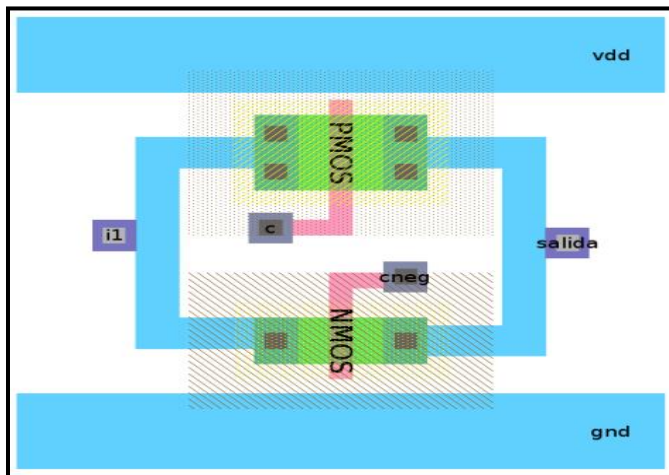
HDL estructural



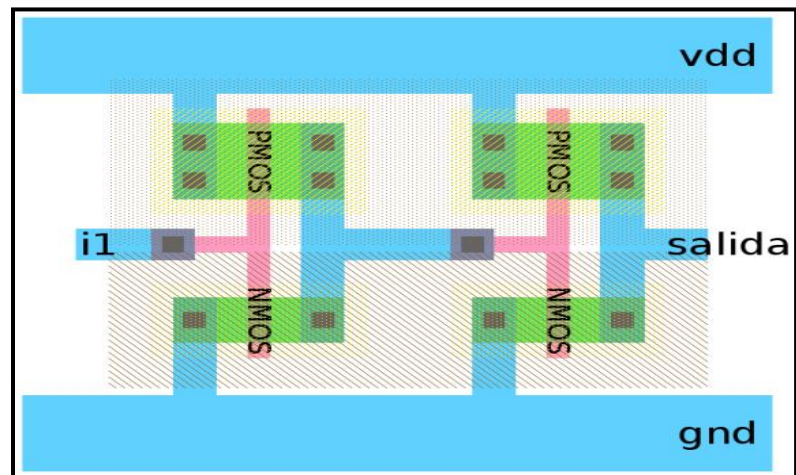
Tristate



Inversor



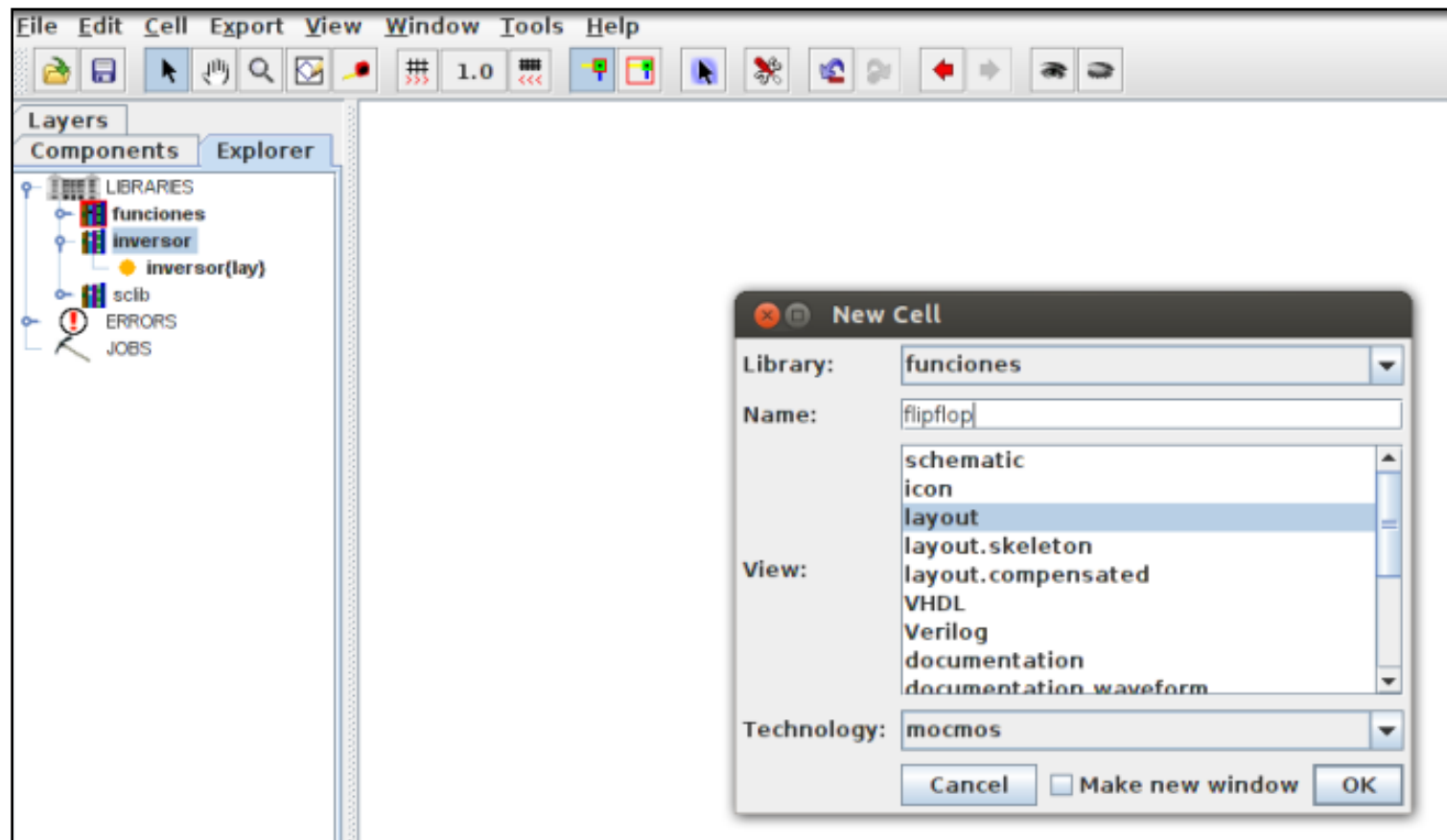
Transmisión



Buffer

HDL estructural

- Se crea una nueva celda tipo VHDL Cell→New Cell



HDL estructural

- Se realiza el diseño en código VHDL estructural.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.all;

entity flipflop is
port(clk, D: in std_logic;
      Q: out std_logic);
end flipflop;

architecture flipflop_BODY of flipflop is

    component inversor port(i1: in std_logic; salida: out std_logic);
    end component;
    component transmision port(i1, c, cneg: in std_logic; salida: out std_logic);
    end component;
    component tristate port(i1, c, cneg: in std_logic; salida: out std_logic);
    end component;
    component buffer port(i1: in BIT; salida: out BIT);
    end component;

    signal clkneg, abuf, bbuf, cbuf, Dneg, n1, n2, n2neg, n1neg: std_logic;

begin
    buf_1: buffer port map(clk, abuf);
    buf_2: buffer port map(D, bbuf);
    buf_3: buffer port map(cbuf, Q);

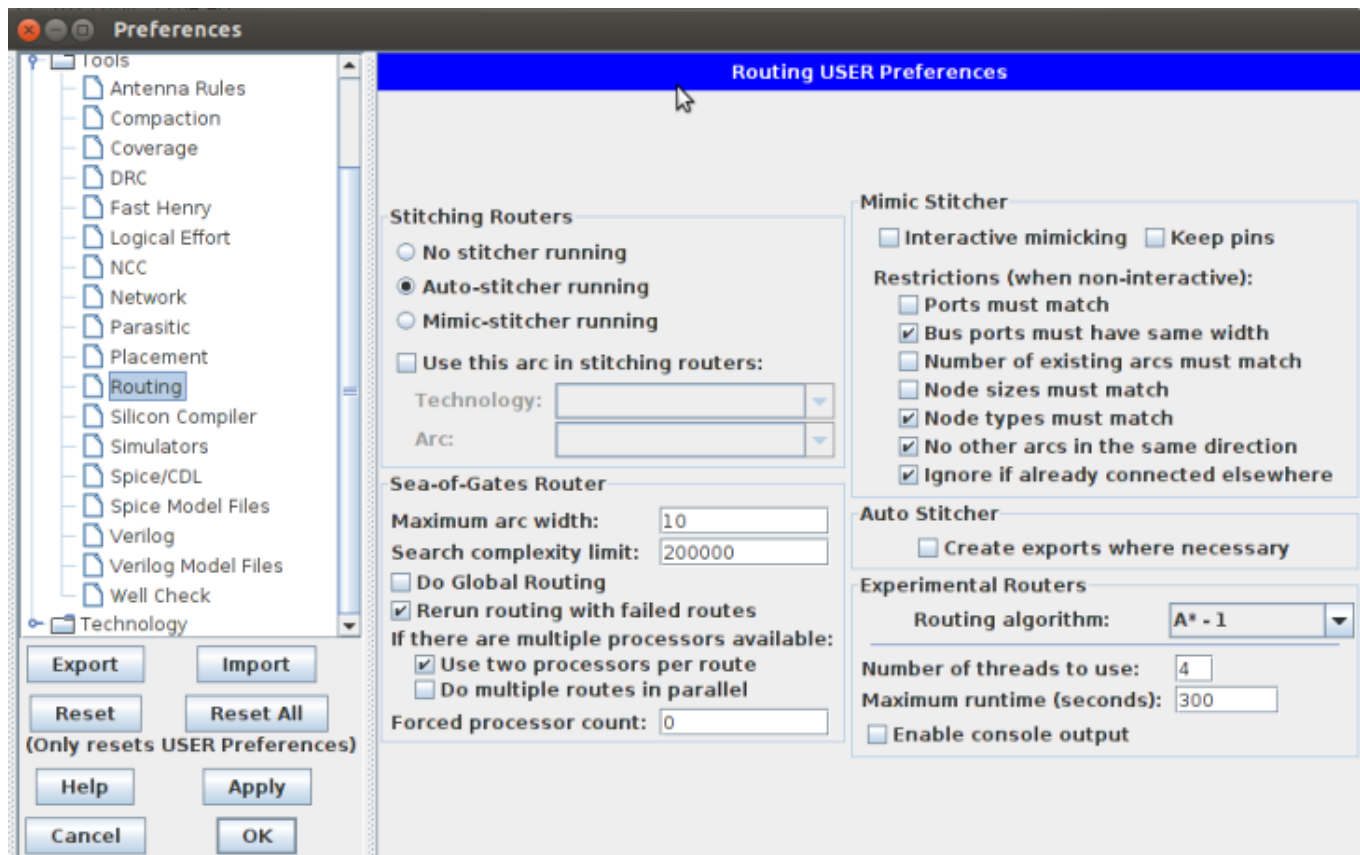
    inv_1: inversor port map(abuf, clkneg);
    inv_2: inversor port map(bbuf, Dneg);
    tran_1: transmision port map(Dneg, abuf, clkneg, n1);
    inv_3: inversor port map(n1, n1neg);
    tri_1: tristate port map (n1neg, clkneg, abuf, n1);
    tran_2: transmision port map(n1neg, clkneg, abuf, n2);
    inv_4: inversor port map(n2, n2neg);
    tri_2: tristate port map (n2neg, abuf, clkneg, n2);
    inv_5: inversor port map(n2neg, cbuf);

end flipflop_BODY;
```

HDL estructural

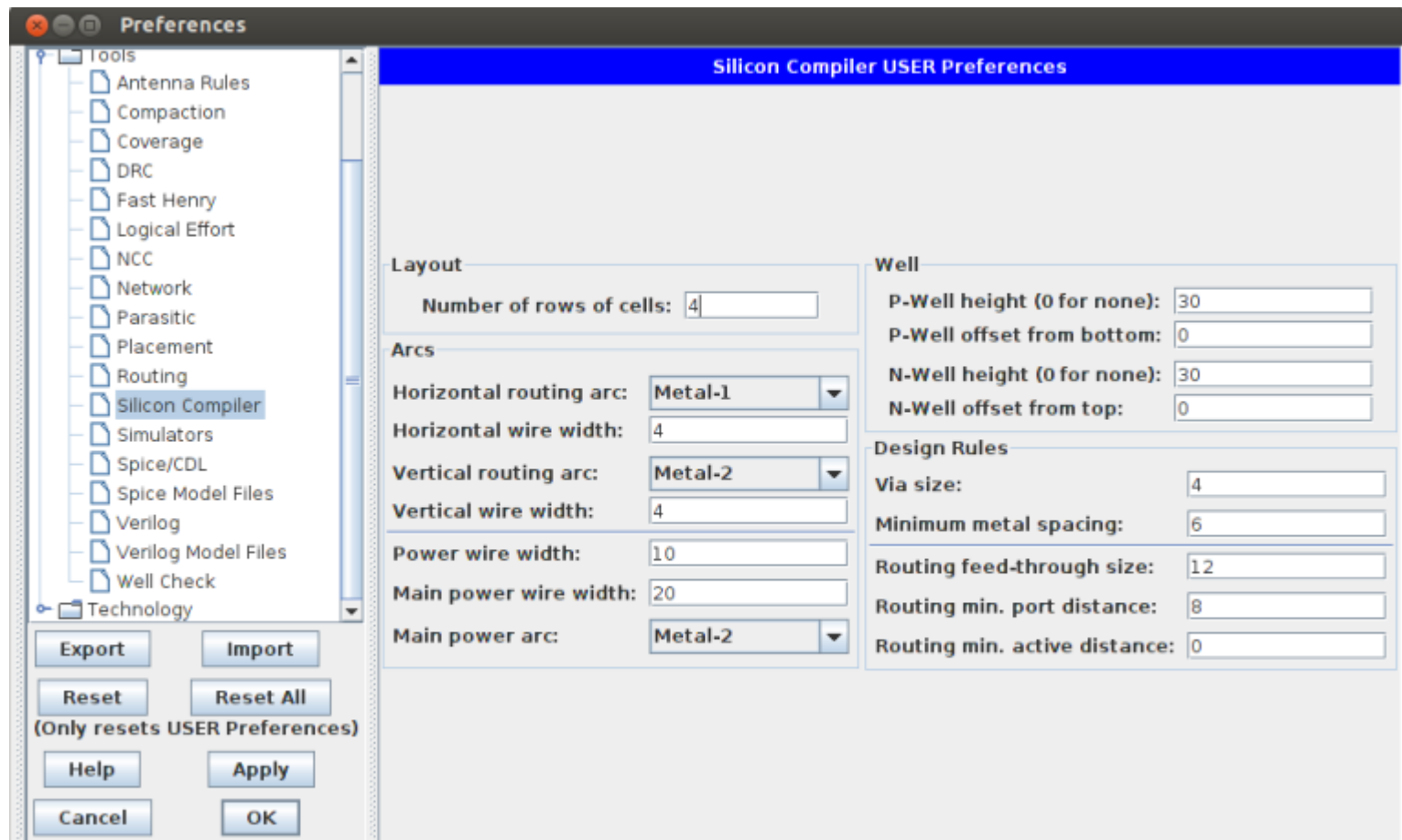
- Para generar el layout se deben fijar primero los parámetros en:

file → preferences → Technology → Tools → Routing.



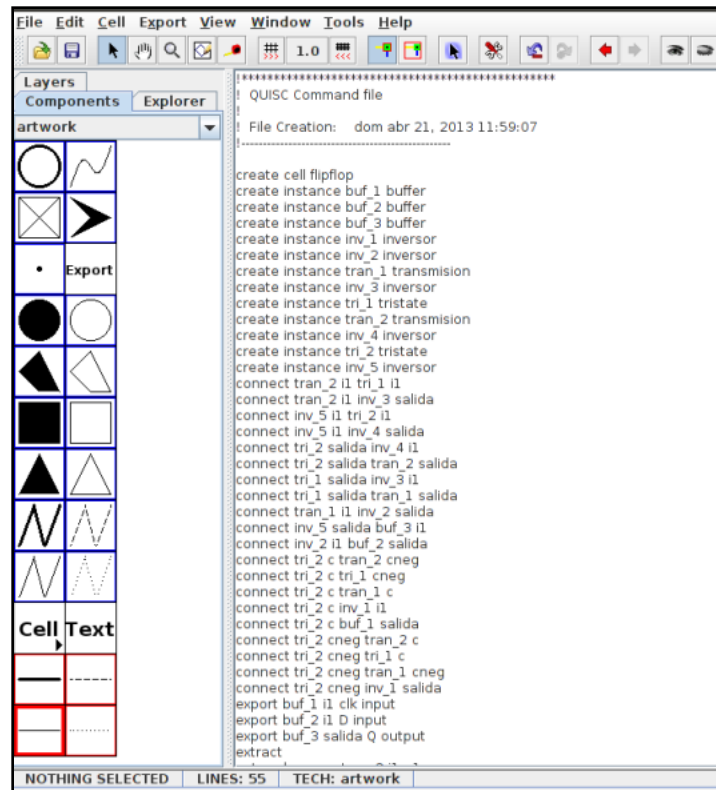
HDL estructural

- Fijar parámetros en:
file → preferences → Technology → Tools → Silicon Compiler.



Verificación sintaxis, síntesis RTL

- Para generar el flip-flop vamos a **Tools** → **Silicon Compiler** → **Compile VHDL to Netlist View**.
- Esto genera la netlist: (Nota: si no se crea la netlist se debe corregir la sintaxis del código VHDL)

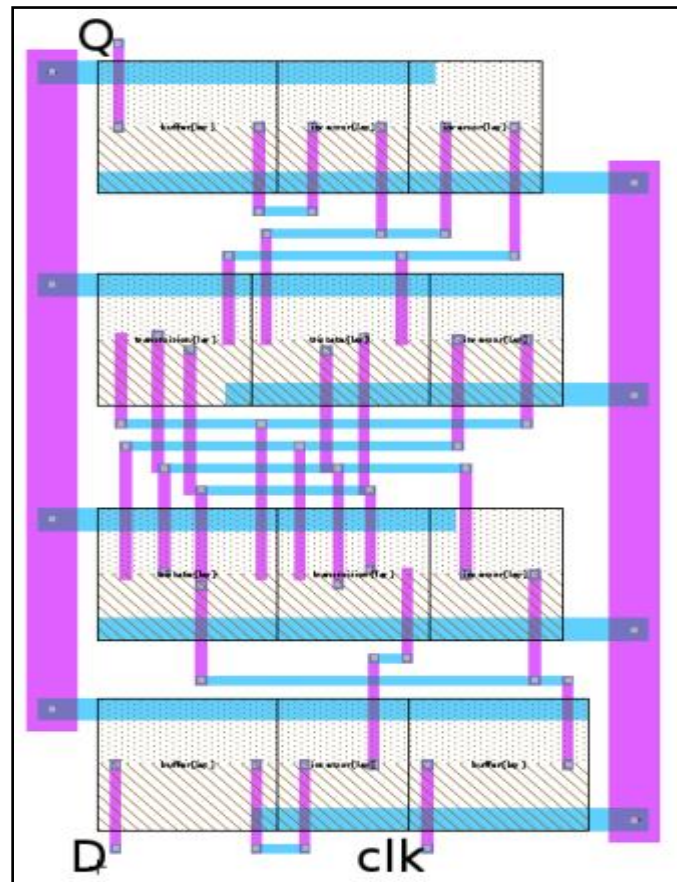


Place and Route

- Para generar el layout primero se debe seleccionar la tecnología, esto se hace en components y se cambia *artwork* a *mocmos*.
- Luego se genera el layout, vamos a:
Tools → Silicon Compiler → Convert Current Cell to Layout.

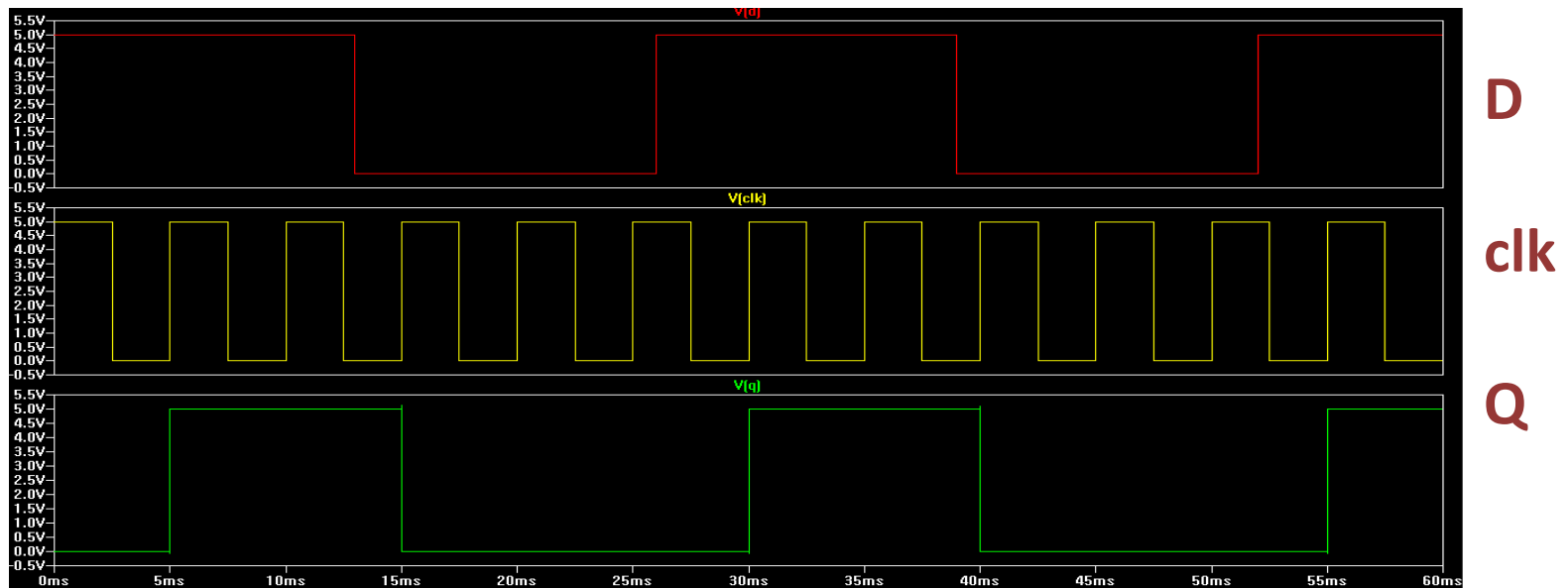
Layout y DRC

- Aparece la nueva celda flipflop{lay}. Con F5 verificamos si hay algún error.



Simulación Spice

- Al igual que el inversor se crea el código de simulación y se genera el .spi para correrlo en Ltspice.



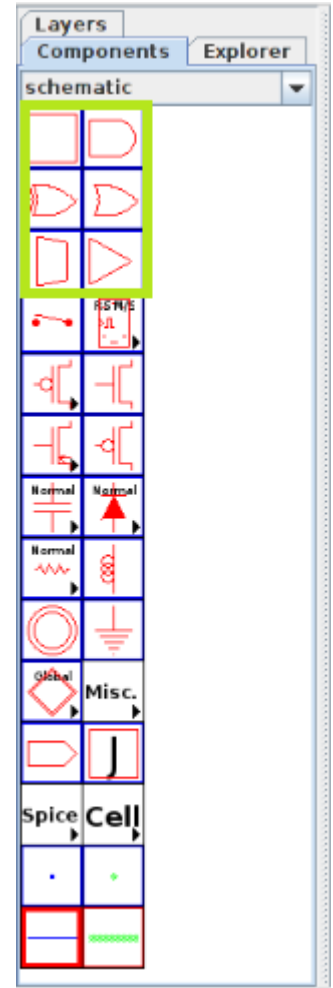
Diseño Semi-Custom Compuertas

Esquemático

- se realizará como ejemplo la siguiente función

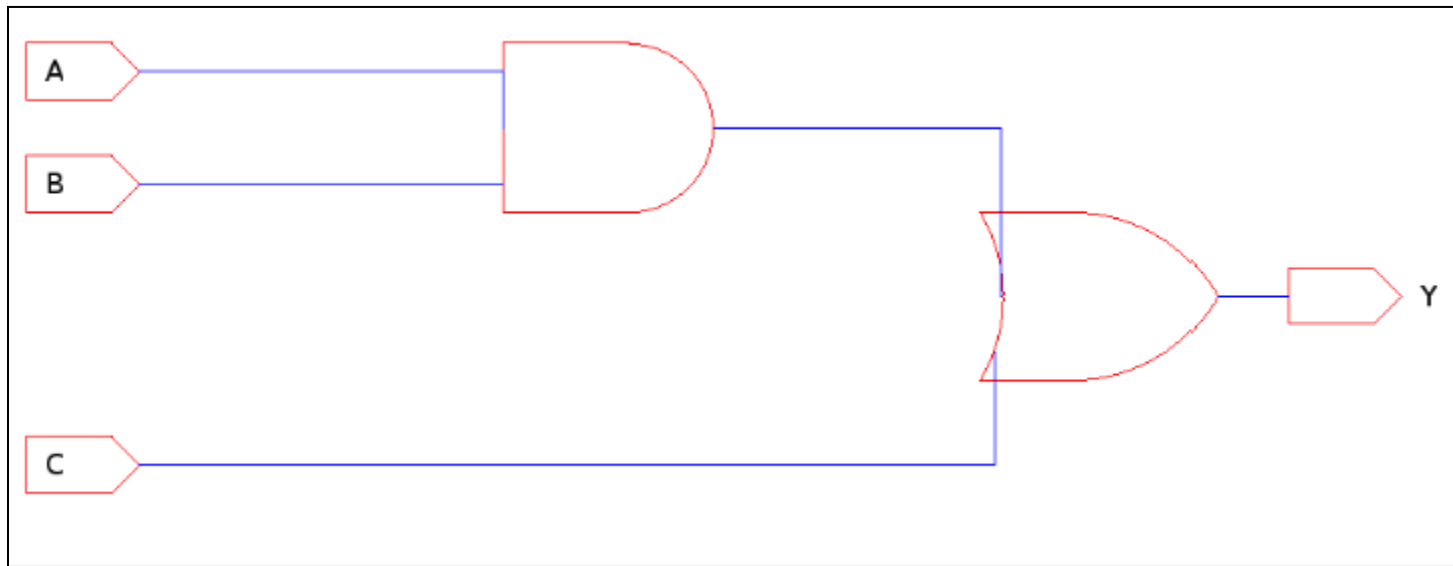
$$Y = AB + C$$

- Se crea una nueva celda tipo Schematic y en components varias símbolos digitales y análogos, de los cuales utilizaremos las compuertas AND y OR.



Esquemático

- Realizamos la función con las compuertas AND y OR, tres puertos de entradas y uno de salida.



Sintesis

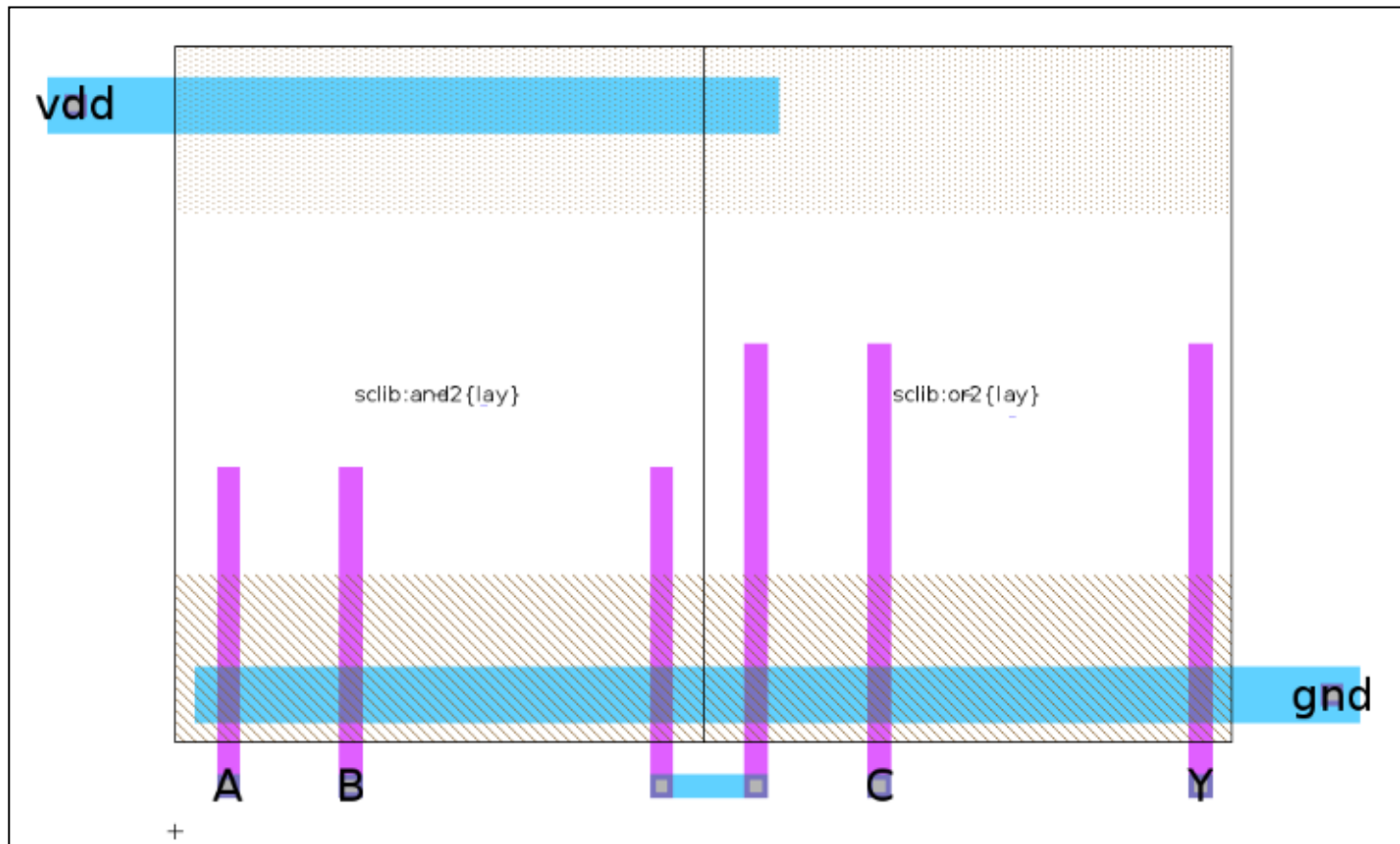
- Para poder convertir del esquemático a layout se necesita tener una librería de celdas estándar previa, para el ejemplo Electric trae por defecto una librería llamada sclib.
- La síntesis automática se realiza:

Tools → Silicon Compiler → Convert Current Cell to Layout.

Esto nos genera 3 archivos con extensiones: vhdl, net.quisc y lay

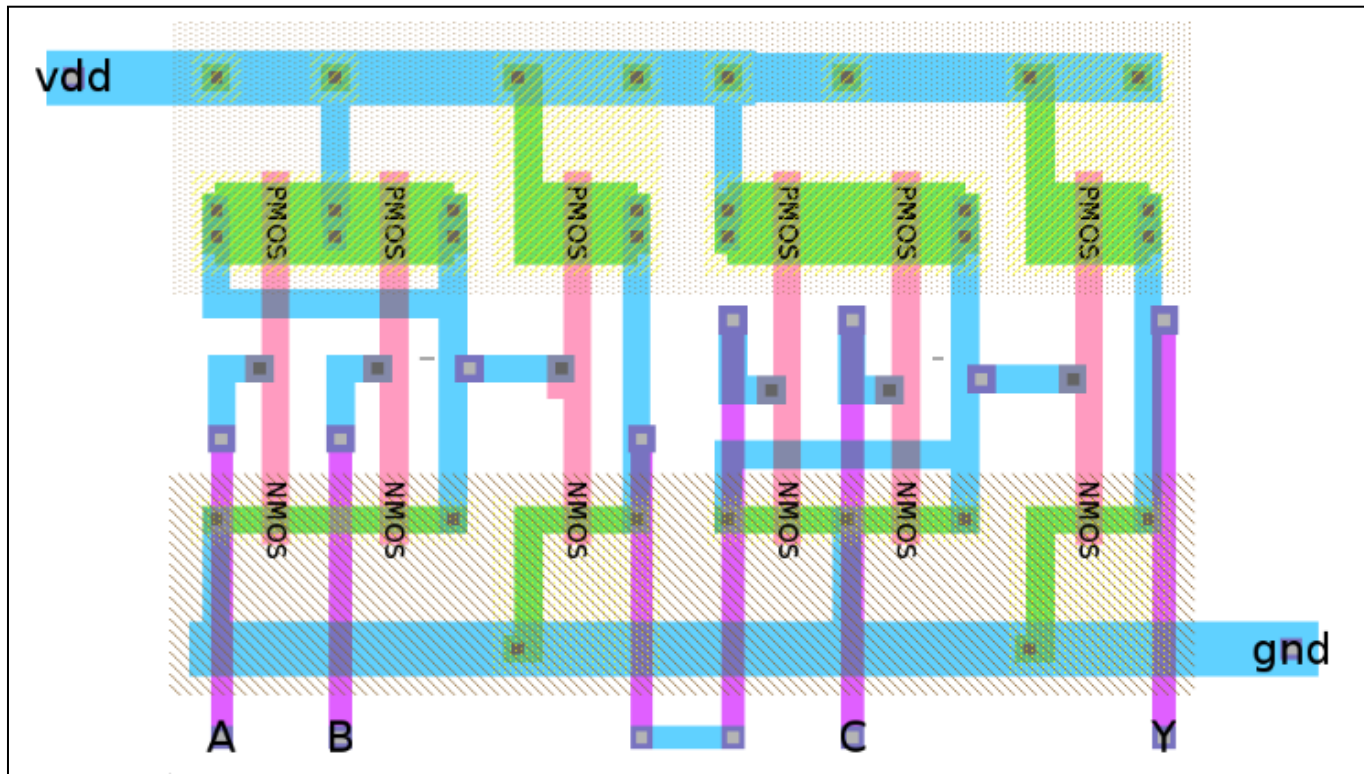
Layout

- A continuación se muestra el layout con las celdas de la librería sclib.



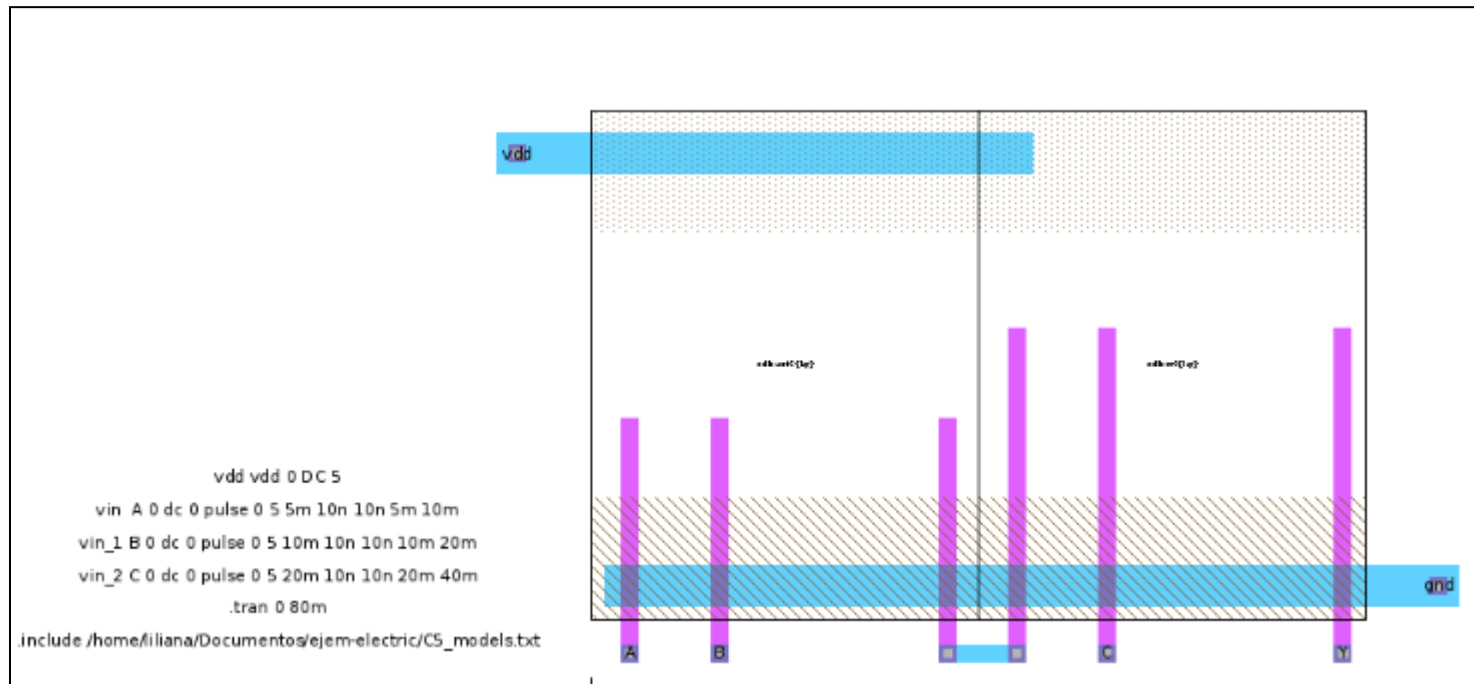
Simulación Spice

- Para simular es necesario asignar a los transistores de las celdas utilizadas el modelo spice.



Simulación Spice

- Adicionamos el código de simulación y generamos el archivo de simulación spi.



Simulación Spice

- A la simulación obtenida en LTspice se muestra a continuación.

