CSE3142 Integrated Circuit Design

Creating Schematic Driven Layout

Practical Experiment 3 **Duration**: two weeks

3.1 About this tutorial

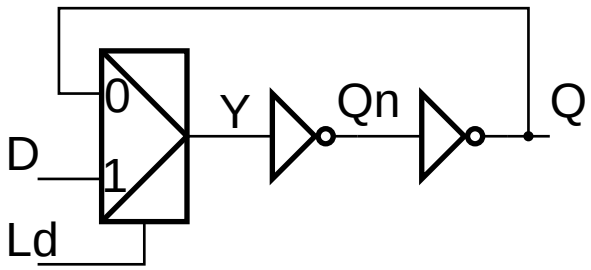
In this tutorial we will be employing a Schematic Driven Layout (SDL) methodology to facilitate creation of a circuit layout given its schematic. This tutorial is based on: *Using the ASIC Design Kit for Schematic Driven Layout* by David M. Zar from Washington University in St. Louis available from `http://ge.ee.wustl.edu/dzar/tutorials/adk_sdl/sdl_toc.html`
For your convenience we quote the following overview:

> Schematic Driven Layout (SDL) is a design process you can use to create IC layouts with information from a logic database (schematic).
>
> The SDL design flow generally uses two windows in ICgraph. One window displays the schematic design viewpoint (logic database) and the other displays the physical cell you are creating. You identify schematic objects to add to the new cell by selecting them, and SDL uses a database to determine library cells or devices that correspond to the selected schematic objects and makes them available for placement in the new cell.
>
> You can use SDL to place devices or library cells in a cell **automatically or manually** (interactively). SDL handles the cells, parameters, and interconnections automatically, but the designer creates the detailed final layout.
>
> SDL uses an EDDM design database (schematic). You will use Design Architect to create the schematic using components previously defined for this purpose. These include N- and P-FETS, resistors, capacitors as well as power and ground connections.
>
> The back-end tool SDL uses to create the IC layout is ICgraph, which requires a design viewpoint to open the logic database (schematic). Creating this viewpoint has been automated for the components that have been defined for you.
>
> During SDL operations, ICgraph uses technology data (from the Process file and rule file), fixed cells, and device generators to create devices for the layout.
>
> Both prior to and after layout, you can simulate your designs using both the digital (QuickSim II) and analog (Eldo, Accusim II) simulators. This allows for both functional testing of your designs as well as post-layout timing verification of the layout.

We will be designing a layout of a D-latch implemented in a combination of the switch and gate logic.

3.2 Creating a schematic of a D-latch

From a directory **$ICDES/prac3** invoke: **adk_daic &** and create a schematic of a D-latch with a weak multiplexer using a method described in **prac1**.



Ld	operation	
0	Q <= Q	Hold
1	Q <= D	Load

Figure 1: A logic diagram of a D Latch and its function table

A logic diagram of a D Latch and its function table is given in Figure 1. A transistor level schematic that you have to enter using Design Architect is given in Figure 2

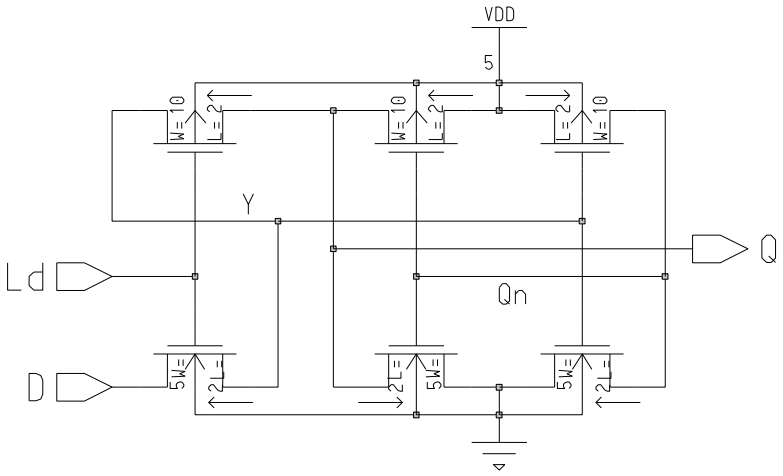


Figure 2: A schematic of a D latch.

- The schematic should be called **UDlatch** where **U** represents your initials.
- Remember to capture your design steps in an **Ample script** . It will save you lots of time if something goes wrong.
- The **length** of all MOS FETs should be: $L = 2\lambda$.
  The **width** of FETs should be $W = 10\lambda$ for pMOS, and $W = 5\lambda$ for nMOS.
- Remember about **portin** and **portout** terminals.
- Assign names to all nets as in Figure 2. One way of doing this is through a pop-up menu:
  Right Click $>$ **Name Nets:** Then change the Property Value to the appropriate name.

- **Check & Save** the schematic sheet.
- Generate a symbol using **Miscellaneous** > **Generate Symbol** menu.
- Add the name **UDLatch** using **Add** > **Text** menu. Place this text in the symbol. You can adjust the hight of the text if required.
- Check and save the symbol.
- You can print the schematic for encapsulation in your report using:

  File > **Export Graphics ...**

  Select the 'Export EPS' format and set up the path name for your schematic. I suggest to change the path name to: `$ICDES/prac3/UDlatchSch.eps`.

At this stage you can safely quit and re-enter **adk_daic** (from the directory `$ICDES/prac3`) if required.

3.3 Selecting an IC technology

Prior to simulation we have to specify the technology in which our integrated circuit will be manufactured. Simulation results strongly depend on the selected technology.

From the **schematic_edit** palette select

Simulation > **TSMC0.35**

This will specify the TSMC0.35 technology and create the following **viewpoints** which will be used in the analog and digital simulation and in the circuit layout creation and verification.

tsmc035	for use with the digital QuickSim II Simulator
tsmc035a	for use with the analog simulators
layout	for use with IC Station when doing standard cell place and route
lvs	for use with IC Station when performing LVS of your design
sdl	for use with IC Station when doing SDL place and route

3.4 Analog simulation with Eldo

Simulate the design using eldo, refer back to the notes for prac1 if you don't feel confident about this.

If you are feeling adventurous then have a go at it now, but remeber the key step before you simulate :

- Setting the library (`$ADK/technology/accusim/tsmc035.mod, NOM`)
- Choosing the type of simulation to perform (analyses)
- Forcing inputs
- Forcing power
- Choosing nets to view

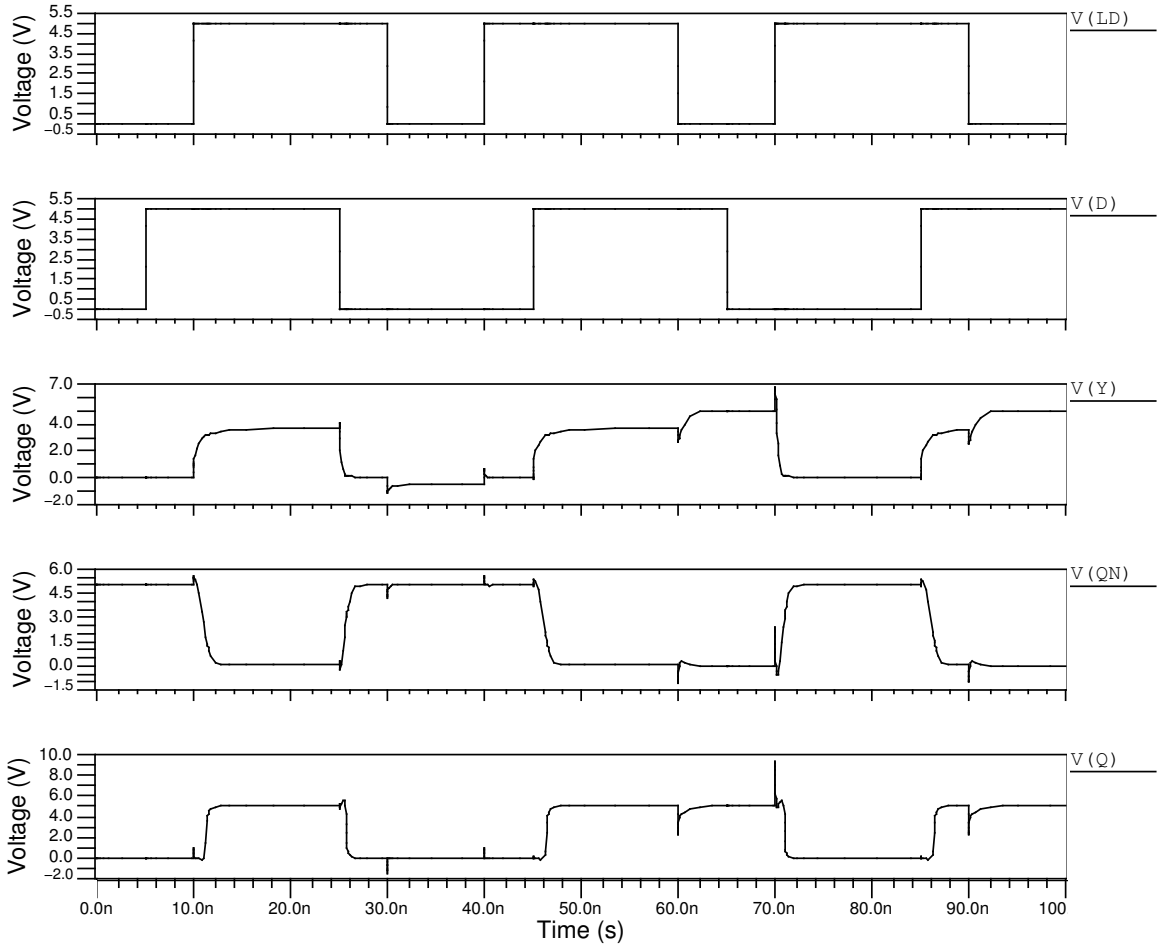


Figure 3: Results of simulation with Eldo

Figure 3 shows some sane input forces and simulation results.

Note and **measure the delay** between the edges of the input signals, Ld, Q, and the output signals, Qn, Q.

3.5 Creating a Schematic Driven Layout (SDL) using IC Station

3.5.1 Creating an IC cell

Before you attempt this section you have to **design a stick diagram**of your D Latch. You will waste lots of time if you do not do that. One possible stick diagram of the D latch which stems from the schematic of Figure 2 is given in Figure 4.

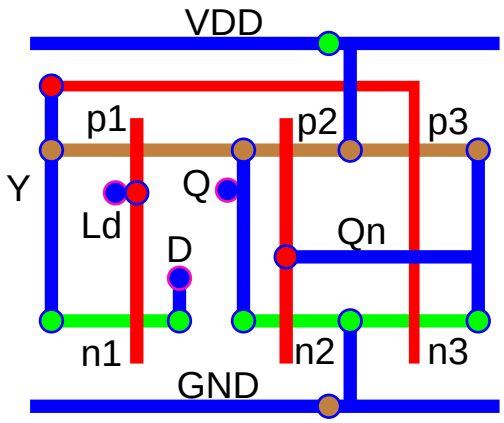


Figure 4: A stick diagram of the D Latch

In the stick diagram you can identify three pairs of transistors, relevant contacts, the input and output nodes and associated nets, (Ld, D, Q), and important internal nets (Y, Qn).

- From the directory **$ICDES/prac3** invoke **adk_ic &**.
- Now you can create a new cell. From the **Session** palette select **Create**.

  In the Create Cell dialog box enter the following (use the Navigator where possible):

  Cell Name : UDlatch
  select button: With connectivity
  Attach Library : $ADK/technology/ic/tsmc035
  Process : $ADK/technology/ic/tsmc035
  Rules File : $ADK/technology/ic/tsmc035.rules
  EDDM Schematic Viewpoint : UDlatch/sdl
  Logic Loading Options
  Logic Loading: Flat (IMPORTANT!)

  All other options should be left at default values.

  A cell window will now appear having the name UDlatch. In the top left bar of the window it should read Process: tsmc035(-R).
- Once the empty cell has been opened, you should go to the **ADK Edit** menu via the **IC palettes** menu. This will bring up the **ADK Edit** palette that you used in the previous tutorial.

  You can also use the **DLA Layout** palette which contains similar functions as the **ADK Edit** one.
- **Opening the schematic:**

  From the **ADK Edit** palette, you need to open your schematic sheet. Click on the **Open** button from the **SDL** section of the palette.

  The schematic sheet should appear in a separate window. It is a good idea to organize the cell and schematic windows in a way convenient for easy switching between them.

- **Placing the devices:**

  Now you can place the devices into your cell. There are many ways to do this as outlined in Chapter 1 of http://ge.ee.wustl.edu/dzar/tutorials/adk_sdl/sdl_6.html

  You will use the **automatic placement** method for this exercise. To do this, click on **AutoInst** on the palette menu.

  At this point, the tools will locate devices that are of the same type and have connectivity such that **diffusion regions** may be **shared** . The devices will be automatically generated based on the length and width parameters specified on the instances and placed into the open cell.

  The first thing you should probably do is to **View** > **All** to see all of your devices.

  Overflow lines (in yellow) will show you the connectivity points in your circuit. As you wire the circuit together, these overflows will disappear when you have made the correct connections.

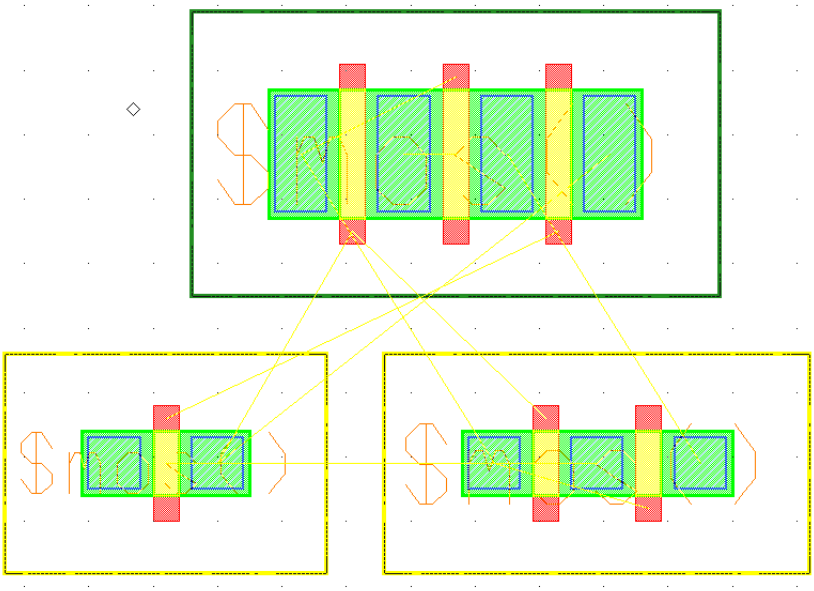


Figure 5: The initial placement of transistors of the D Latch

An example of the automatic placement result is given in Figure 5. Note that three pMOS transistors can share a diffusion area as indicated by their serial-like connection on the schematic. The nMOS transistors are arranged in the two groups consisting a single nMOS and a pair of nMOS transistors.

Note that the placement algorithm places transistors fairly tight, therefore, in order to have all interconnections made we will need to move transistors around.

Note also that transistors come complete with all details including **diffusion-to-metal** contacts. To see them you have to peek down through the hierarchy. Use **Context** > **Hierarchy** > **Peek Area** to perform the necessary operation.

- First, practice the selection of all devices, connections and pins. Notice that when you select a

device or a connection it is also selected in the schematic window. This cross-selection is very useful.

In particular, identify the GND and VDD nets and related overflow lines.

Play around with selecting things for a moment to get a better feel for how this all works.

- Selecting the overflow lines or gates one-by-one you can identify the order transistors. After consulting the schematic (Figure 2) or the stick diagram (Figure 4), the circuit can be described by the following layout matrix:

$$\mathcal{L}1 = \begin{bmatrix} Qn & (p3) & V_{DD} & (p2) & Q & (p1) & Y \\ & Y & & Qn & & Ld & \\ & Ld & & Y & & Qn & \\ D & (n1) & Y\ !\ Qn & (n3) & \mathrm{GND} & (n2) & Q \end{bmatrix} \tag{1}$$

  where '!' describes the discontinuity in the diffusion.

  In order to achieve the order of transistors as in the stick diagram we have to **flipped horizontally** all three groups of transistors. To do this, select carefully each block individually and type in the command **fli in p h** (flip in place horizontally). If overlapping occurs execute **mov re 20 0** (move relative by x = 20, y = 0) or something similar. Press F2 do de-select after each command.

  After flipping, we should have transistors arranged in the same order as in the stick diagram, and the circuit layout is now as follows

$$\mathcal{L}2 = \begin{bmatrix} Y & (p1) & Q & (p2) & V_{DD} & (p3) & Qn \\ & Ld & & Qn & & Y & \\ & Ld & & Qn & & Y & \\ Y & (n1) & D\ !\ Q & (n2) & \mathrm{GND} & (n3) & Qn \end{bmatrix} \tag{2}$$

- At this stage we need to have a **floorplan** specifying the location of the layout components. We do not aim initially at the minimum size, but rather to satisfy the design rules.

  First, identify the origin of the coordinate system and note that the grid is setup to 0.5λ.

  Second, measure all three Wells. In my case the measurements are as follows.

  The nWell: $41 \times 22\lambda$, the two-transistor pWell2: $33 \times 17\lambda$, and the single-transistor pWell1: $25 \times 17\lambda$.

- The main considerations for the floorplan are as follows:

  – In general, the layout will consist of the following **rows** (from top to bottom): VDD rail, (horizontal) interconnections, pMOS transistors, interconnections, nMOS transistors, interconnections, GND rail.

  – The hight of the GND and VDD metal1 rails should be 10λ to match the size of the well-to-power contacts.

- In the original stick diagram there are no horizontal interconnections between GND and nMOS transistors.
- You need space to fit in the metal1-to-poly contact between the VDD rail and pMOS transistors.
- The area between the wells needed for contacts and interconnections can be estimated as, say, 23λ high. If you write a script for your layout you can easily reduce this distance later on.
- The width of the cell will be initially dictated by the combine width of the pWells.

From the above considerations we decide that the **Wells** with the devices can be initially placed at, say:

nWell: [16, 50] ; pWell-1: [0, 10] ; pWell-2: [25, 10]

In this way we should have gates of transistors (p2, n2) and (p3, n3) aligned vertically.

- Select carefully every well in turn and place it at the planned position. Remember to use the F2 key to **unselect** after a move has been completed.

  You can use the **Move** button from the **ADK Edit** palette to perform approximate moves, and then the **Rel Move** (mov re x y command) to numerically specified the x, y offset and position layout components where required.

- Now we will move a p-well with the transistor n1 in it horizontally to overlap with n2, n3 as much as possible.

  First, from the **ADK Edit** palette select **Drc** $>$ **Check** and verify that there are only two results (that is, the design rule violations) regarding well/substrate contacts.

  Next move the well with n1 horizontally by, say, 10λ and again verify the design rules violation. You find that the diffusion (active) separation must be 3λ.

  In my case the total resulting p-Well has the horizontal length of 49λ. The overall result should be similar to that presented in Figure 6 (without metal interconnections).

- Before you do wiring enter the following command:

  SHOw LAyer Palette 46, 49

  This will display the **Layer Palette** for polysilicon and metal1 layers.

3.5.2 Creating power rails, ports and well contacts.

- We create first the GND rail and add the p-well contact.

  Select **metal1** from the layer palette and then from the **ADK Edit** palette select **Shape*** . Create metal1 shape from [9, 0] to [58, 10]. This will be the GND rail. The resulting shape should be adjacent to the pWell.

- Select the GND rail and from the pull-down menu select **Objects** > **Make** > **Port** . You will be prompted for the type (make it **power**) and for the port name (make it **GND**). Leave the direction as is since the power will be coming into this cell. You cannot make it bidirectional so do not even try!

- Place the p-Well contact using the **pwc** macro. The p-well contact should be placed on the **GND** rail so that its bottom edge is aligned with the bottom edge of the rail.

- Now we will move the GND rail and the the p-Well contact up until we violate the distance rule.

  Select the GND rail and the the p-Well contact and move it up by 3λ. Check the design rules and you should have just one result. One more λ and we are in trouble.

- If we are planning to create a polysilicon connections between the gates of the (n3, p3) transistors and the Y net, we must be aware of the fact that polysilicon cannot cross the well contact. Therefore we have to make room for the polysilicon connection **adding** at least 3λ of the **n-Well** on the top of the existing n-Well. Use the **Shape** command.

- The VDD rail can now be placed on the top of the extended n-well. Place the metal1 shape at [9, 75], [58, 85].

  Select the VDD rail and from the pull-down menu select **Objects** > **Make** > **Port** . You will be prompted for the type (make it **power**) and for the port name (make it **VDD**).

- Place the n-Well contact using the **nwc** macro. The n-well contact should be placed on the **VDD** rail so that its top edge is aligned with the top edge of the rail.

3.5.3 Wiring up.

- Now we will wire up all metal1 and polysilicon connection. There are many methods for accomplishing this task. You know two already, namely, using **Shape** and **Path** commands. In addition there is an **AddR** command very useful for **wiring metal** together.

  With reference to our stick diagram let us create first the vertical metal1 connections, namely, Qn, Q and connections to VDD and GND.

  Use the **AddR** item on the **ADK Edit** palette menu. This will prompt you for a starting and ending point of a route. Make the starting point on the metal contact at the right pMOS being part of the Qn net. Notice how, to the right of the pointer, it says **route_1** ? This shows that you are on routing layer 1 (or metal 1 for the AMI process). Once you have selected the starting point for the route, a guide will appear showing you the clearance where you can place the route without any problems. Draw the route down to the right nMOS. This is a part of the net Qn that we will complete later.

  Similarly wire the Q net, the metal part of the Y net, and the vertical connections to VDD and GND.

  At this stage my layout looks as in Figure 6.

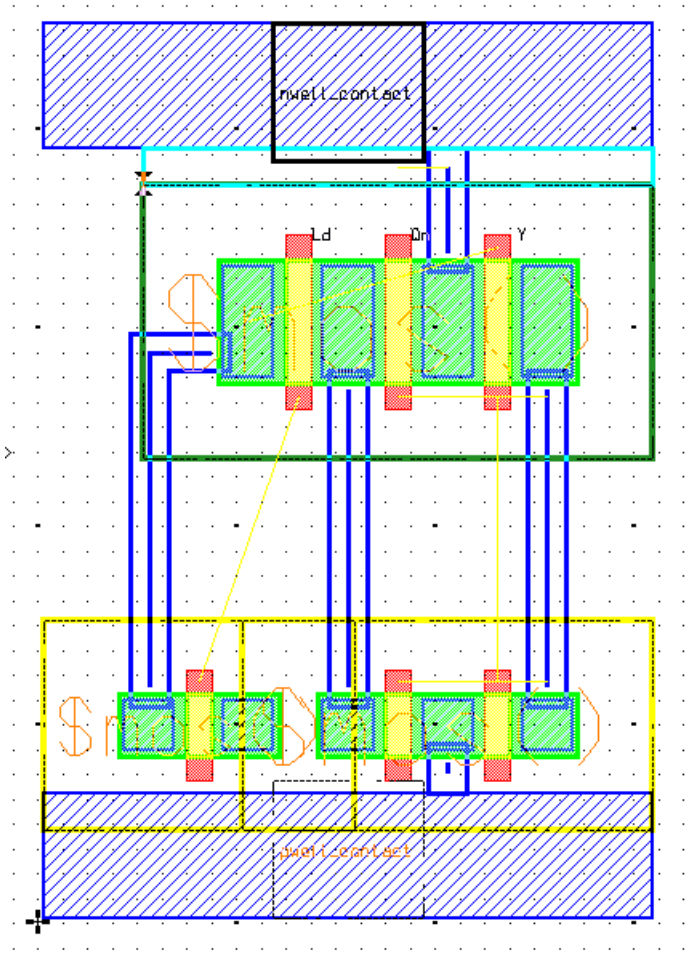


Figure 6: Partial layout with VDD and GND nets and sections of the Qn, Q and Y nets

- **Polysilicon wires** can be conveniently created using the **Path** command. First, click on the **poly layer** in the **layer palette** and click on **Path** on the **ADK Edit** palette. Now you will place a route on the poly layer.

  Click on the **Options** button on the prompt bar and in the dialog box change the **width** to 2λ, select the **Keep Option Settings** and leave everything else alone. Now you are ready to place poly paths.

  Wire up the **gate** inputs with **poly** using **Path** . When you have made a complete connection, the overflow line will disappear.

  Be sure to follow the design rules! Two λ minimum width, three λ separation, etc.

- In order to complete the Qn and Y nets you have to place first polysilicon-to-metal1 contacts. Use the **pc** (poly contact) command which will place the contact wherever you point to with a mouse.

- Note that you also need a poly contact on the Ld net. This contact is needed to create the input port.

- When you finish wiring up your circuit, so that all the overflows vanished, you are ready to place input-output ports.

  Before you can place ports, you have to select the type of port you wish to use. Go to the **Setup** $>$ **SDL...** pull-down menu to do this. Click on **SDL Port Styles** and then press the **Setup...** button. You should then select **Process Port** on the dialog box to display the ports defined for this process.

  We will put our ports on metal 2, so select the **default** line and then click on **Preview** to see its parameters. Notice that it is on metal 2 and it is $4 \times 4\lambda$. Click OK to select this as the default port style.

- To **place the ports** , select the two inputs and the output in your schematic window. Then make the layout window active, again, and click on **Port** in the **DLA Layout** palette menu.

  One by one, you are prompted to move these ports to where you want them. The **net** the **port** will be placed on is highlighted in the layout window to make it easier to see which port is being placed.

  Note that geometrically, the port is 4×4 square of metal 2, and it should be placed adjacent to a metal 1 path representing the required electrical net. Make sure to connect ports to the appropriate nets.

  If your net is on polysilicon you have to use a polysilicon-to-metal1 contact to move the net from poly to metal1.

- In order to make the **via** between the metal2 port and metal1 you can use the following command:

  pp : The **Place Port via** command will get a metal1-metal2 via for you and let you place it on top of your port (or anywhere else).

- **Final Design Rule checking**.

  As previously, you can invoke design rule checking from the **ADK Edit** palette be selecting **Drc** $>$ **Check**. Click OK on the prompt bar and DRC will be performed.

  You should fix all DRC errors before moving on to the next section.

- Save your cell. If any subsequent editing is required you will have to execute **File** $>$ **Cell** $>$ **Reserve** $>$ **Current Context**.

  A compact version of the final layout satisfying the design rules is given in Figure 7.

- There is one problem with the layout, namely, the ports Ld and D overlap in a vertical row. This is not reported as the design rule violation, however, It would be inconvenient, if not impossible to use such a cell in a bigger circuitry where metal 2 is used as vertical connections. I will leave it to you to fix the circuit accordingly.

Finally, you should place **metal1 blockages** over all the metal1 in your cell. The reason is that by default, everything but the ports are invisible at upper layers of the hierarchy. Later, if we use these cells to make a larger cell we do not want to route metal1 over our metal1 wires in this cell! Fortunately, the ADK provides a script that will do this for you automatically.

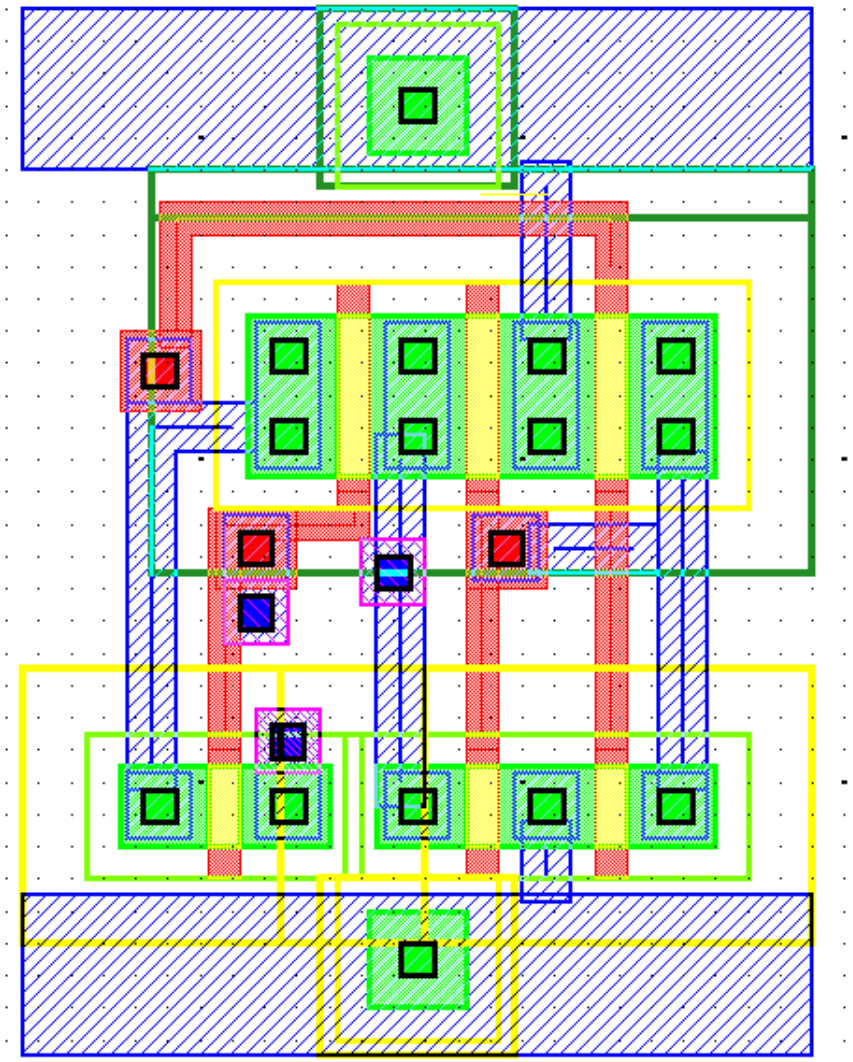


Figure 7: The final layout of latch which satisfies the design rules. Details of hierarchical structures are shown.

ab1 : to include all metal1 blockages over metal1 paths (reserve the cell for editing).

rab1 : to remove metal1 blockages (to make re-editing of the cell easier)

3.6 Layout Versus Schematic (LVS)

After the Design Rule checking we check Layout Versus Schematic (LVS) to find and correct electrical errors.

- First, from the **IC Palletes** we navigate to the **ICtrace(M)** palette menu. This is mask-based LVS checking. It will ignore all hierarchy and look only at the actual connectivity of your devices.
- Once in the **ICtrace(M)** menu, select **Load Rules** from the palette menu to load a rules file that includes data needed for LVS verification. Specify the rule file

$ADK/technology/ic/tsmc035.rules

- Select the **LVS** item to bring up the LVS dialog box.

  You will need to supply the schematic source name. Use the navigator or type in the source name including the viewpoint, e.g. **UDlatch/sdl**. If you use the wrong viewpoint, LVS will fail. You can use the **sdl** viewpoint for LVS as well.

  Accept the setup dialog box and the LVS dialog box and then LVS will begin.

- When LVS is complete, you should look at the report by using the **Report** > **LVS** item from the **ICtrace(M)** palette.

  A text window will appear displaying the report. Be sure is compared correctly. If it did not, you will need to fix any errors before performing the next step.

- Once you have checked your LVS report and everything is OK, close the LVS report window.

3.7 Advanced topic : Parasitic Extraction

This section is for advanced students and is not compulsory,
if you do not feel like a lot of a challenge then proceed to the section 3.9

The notes for these two sections worked with previous versions of IC_Flow, but the current version of IC_Flow doesn't support parasitic extraction.
Instead you must use Calbre to do this job.

The notes below serve as an outline for the process, but to work out how to do it with calibre you should refer to :

usonic:/sw/mentor/Calibre/2003.6/ixl_cal_2003.6_7/shared/pdfdocs/xrc_trans_user.pdf

After your layout passes **DRC** and **LVS** , you are ready to perform **parasitic extraction** . This information will allow you to simulate your circuits with **backannotated RC delays** in Eldo or AccuSim II. You will be able to analyze the performance of your final circuit's layout. At this point, you can only extract for analog simulation (Eldo, AccuSim II). Hence, **lumped extraction** is what you will be performing.
Lumped extraction will treat each distinct net as one lumped capacitor and/or resistor. Therefore, every point on the net will see the full extracted RC delays. Hence, a transmission line will look the same so matter where you are on that transmission line. Keep this in mind if you expect different results!

- To prepare for extraction, be sure your cell is reserved for edit.
- Go to the **ICextract(M)** palette menu.
- If you have the **sdl viewpoint** already open, you need to close it, first. You can close it from the **Logic** > **Close** palette menu.

- Select **Load Rules** from the palette menu to load a rules file that includes data needed for Eldo or Accusim extraction. Specify the file

  $ADK/technology/ic/tsmc035.rules for Eldo

  (or, **$ADK/technology/ic/tsmc035.accusim.rules** for AccuSim).

- Now select **Lumped** from the palette menu to display the extraction dialog box.

- You need to specify a schematic source since you will be backannotating to it. Click on **Yes** to the right of the **Specify Schematic Source** to show the **Source Name** field. You will then be able to enter the appropriate source and viewpoint name. Unfortunately, Mentor did not see the need for a navigator button here, so you will have to type your cell's viewpoint name: **UDlatch/tsmc035a**. (For AccuSim the name should be **UDlatch/accusim**).

- Now you should specify that you want to backannotate the results. Select "**Yes**" by the **BackAnnotate** option.

  Specify the name of a BA file to use. A suggested name is **lumped** but you can choose anything. If you choose a previously used name, then you should also select the "**Yes**" button under the **Clear** option to remove any previous annotations.

  Alternatively, you can have several BA files where the most recently created is used until you change the BA file in DVE, yourself.

- Now it is time to select the parameters to backannotate. Simulators will use the **cap_net** property for net capacitance between a node and any others, plus the intrinsic capacitance to ground for the node. This is what you will use. Since this property will be overridden by the **cpl_cap_net** property (which is just the coupling capacitance) **turn off** the **Coupling Capacitance** extraction.

  You should check **Lumped capacitance** and **Lumped resistance** .

  Accept this dialog box and the extraction will begin.

- When the extraction is done, select **Logic** $>$ **Open** (UDlatch/tsmc035a viewpoint) you will see that your schematic is automatically updated and you will see the back annotations on it.

  There is a pair of values associated with every net, resistance in Ω and capacitance in fF = 10^{-15}F

  Now you are ready to simulate your circuit with extracted parameters.

- You can now quit the IC Station.

3.8 Post Layout Simulation

The final step in the design process is post-layout simulation. Previously you have designed a circuit by capturing a transistor-level schematic, functionally tested it, done a mask design and extracted parasitic capacitances and resistances from the mask. Now you will see if the circuit has the desired timing and drive characteristics that you had hoped for.

We will check the delay through your circuit as well as rise and fall times at the inputs and outputs. For simplicity we will not have a load connected to the output of the circuit, but we need to remember that in reality the circuit will drive other cells.
We will use an analog simulator Eldo that you have used in tutorial 1. Eldo can be conveniently invoked from the Design Architect, **adk_daic**.

- It is a good idea to prepare first a **simulation commands/parameters** file, **UDlatch.sim** and place it in the `$ICDES/prac3` directory. You can use the file `$ICDES/prac1/Uinv.sim` as a template and modify it appropriately.
- From the directory `$ICDES/prac3` invoke `adk_daic &` and open schematic **UDlatch**.
- From the **schematic_edit** palette select

  Simulation > **TSMC0.35**

  This will load your cell with the backannotated **lumped resistances** and **capacitances** that ICextract placed on your design.
- **Setting up the default viewer.**

  From the pull-down menu select:

  Setup > **Simulation** > **Setup Simulator/Viewer...**

  and in the dialog window select your favourite viewer.
- **Creating the netlist file**.

  From the **schematic_sim_palette** invoke Netlist > Write.

  This will create a file `.../tsmc035a/*.spi` with the backannotated circuit description.
- **Including simulation commands/parameters file**.

  Include the simulation commands/parameters file **$ICDES/prac3/UDlatch.sim** invoking in the **schematic_sim_palette** Setup Other > Include command. This will create a `Uinv_tsmc035a.cir` file that you can examine using Commands > Edit command.
- **Invoking Simulation**.

  From the **schematic_sim_palette** invoke Simulation > Run. You can examine the resulting log file using the Simulation > View Log command.
- **Invoking a viewer**

  From the **schematic_sim_palette** invoke Results > View > Invoke viewer. This should produce the plots similar to those in Figure 8.

From the waveform you can estimate that the propagation delay is approximately 150ps.

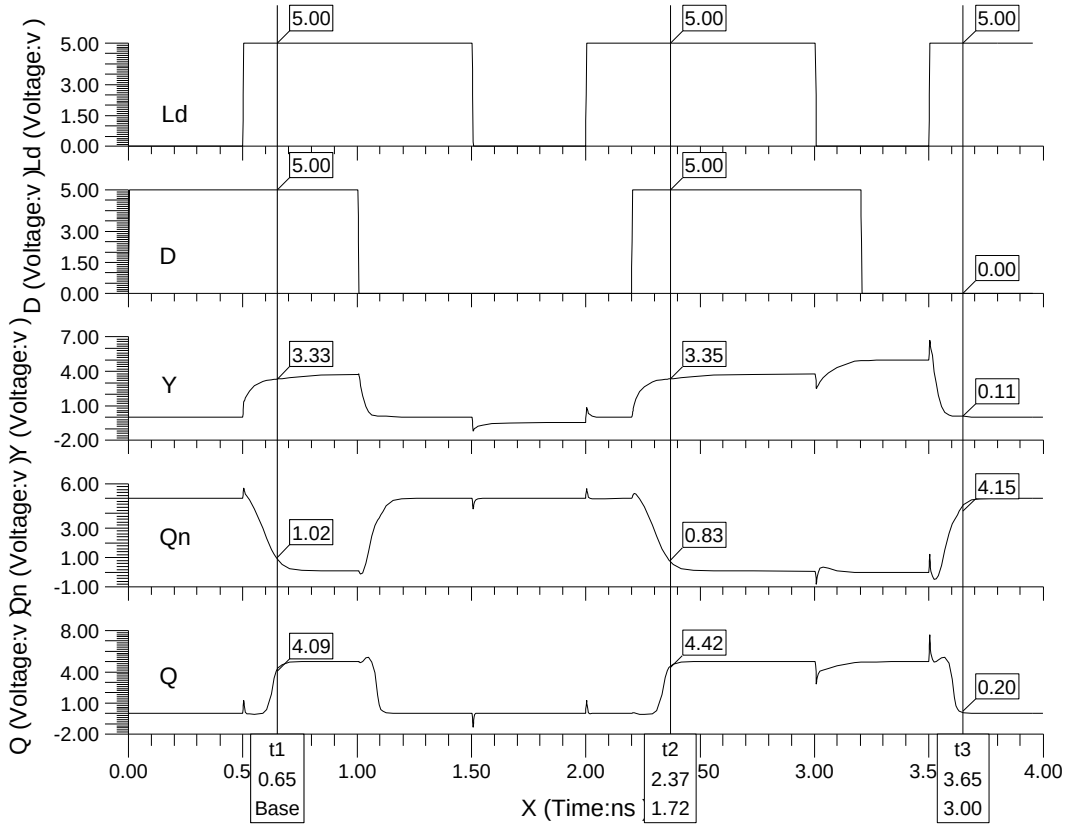


Figure 8: Post-Layout simulation of the D latch with Eldo.

3.9 Demonstration and Report

- Demonstrate your working circuits to your tutor.
- Write a brief report in which you include a short description of your activities and the results obtained. Include schematics, simulation results, layouts, stick diagrams and relevant explanations regarding the structure of the library and your own inverters.

  Edit postscript files as described in prac1 if required.