EE382M: VLSI I
Monolithic 3D IC Basic Cell Library

Emily Bragg, Tengchieh Huang
emily.bragg@utexas.edu, tengchieh@utexas.edu

December 7, 2014

1 Introduction

3D integrated circuits (3DICs) are one potential advancement that will allow circuit designers to save area by having different parts of a design on different layers of silicon which are then bonded together. There are several different styles of 3DIC technology, but we will be focusing on monolithic 3DICs. We designed a standard cell library for 3DIC technology, and simulated the cells to determine the performances differences as compared to the standard version of each cell.

2 Technology Background

2.1 3D ICs

There are two common types of 3D ICs. The more common variety, which the library we plan to use was designed for, uses Through-Silicon Vias (TSVs) to connect layers, with each layer typically having full transistors. This allows vertical logic, but requires careful planning so as to ensure that the layers line up. Another issue with this approach is that a single TSV is very large relative to the cells it connects; at a 45nm technology node, the minimum width of a TSV is 6000nm [3]. This obviously cannot be used for fine-grained modifications to a design, and must be planned for when the design is initially made.

2.2 Monolithic 3D ICs

One way of avoiding the problem with TSV is through monolithic 3D IC technology. This technology is also a vertical technology, but it builds up very thin layers of devices instead of bonding them together. Instead of having TSVs, monolithic integration has 'monolithic inter-tier vias' (MIV). These MIVs, at 70nm, are a similar size to the contacts used at a 45nm technology node [2], and thus can be used to design individual cells across multiple layers. They additionally have negligible parasitic capacitance. Because a significant portion of cell area comes from the necessary separation between the pwell and the nwell, by splitting the pull-up network and pull-down networks and putting each on a separate layer, we can achieve significantly smaller area without having to alter logic. We also considered doing the method presented in [1]; however, due to the limitations of the tools, the advantages of these method (stacking cells on top of other cells by layers to reduce wire lengths and delays across designs) would be difficult to illustrate.

3 Specifications

3.1 Overview

We are designing a library of standard cells (listed in Table 1) for reducing area using monolithic 3D IC technology. Once completed, these cells should be able to be placed in a design andreduce the area by up to 40% in the ideal case. This also should allow for less area consumed by routing, as routing can be run in the layer that has less area dedicated to logic (typically the same layer as the NMOS, due to the smaller transistor widths).

Cell	Sizes
AND	2,3,4
OR	2,3, 4
NAND	2, 3,4
NOR	2, 3, 4
AOI	21, 22
OAI	21, 22
INV	N/A
BUF	N/A

Table 1: Cell Types and Sizes

3.2 Interface

The goal of cell design is to ensure that the inputs and outputs match identically with standard cell libraries of traditional technology. As such, each cell has the standard outputs. The primary difference is that the target library has VDD and GND rails stacked on separate layers, and as such they are not at opposing ends of the cell as they traditionally are in standard cells. All inputs and outputs come from the opposite side of the cell from the power and ground rails in our designs.

3.3 Design Targets

Our primary goal is area reduction, and we expect that using 3D IC technology can achieve area improvements approaching 50% in the ideal case. However, we expect most of our better gains to be around 40% less area, due to the overhead created by the MIVs. The best cases for area reduction come from gates that have equally sized PMOS and NMOS transistors; however, this is rarely the case (with size typically larger on the PMOS network), and even if it were, we could not achieve 50% area reduction due to the MIV area. We plan to create reference versions of all of the standard cells, and the power and timing requirements for the reference versions are our design targets for the power and timing requirements of the 3D versions.

4 Design

4.1 Basic Architecture

The most basic goal of our design is to achieve the functionality of each standard cell. Because we used the technique outlined in [2], the PMOS and NMOS networks remained the same as they were in the standard cell, except for removing the poly between and replacing it with a contact to metal1.

4.2 Toolset

NCSU has a library for 3D ICs that we initially considered using. However, while it does support multiple layers, it has no functionality for MIVs (focusing rather on TSV-based ICs). This meant two things for our design. Firstly, if we built our designs using the 3D IC library provided, there would be no way of achieving power and timing simulation, as TSV technology has radically different effects. Secondly, TSVs would dominate the area of any cell, and thus we would not be able to get valid area results either. The only advantage this path would have had would be for purposes of simulating the logic, which was basic and thus would not be important information.
We instead decided to utilize the standard 45nm NCSU libraries. While this does not support laying directly, we were able to simulate the effect of the MIV by separating the layers with a metal1 wire, with the length calculate to provide the same resistance as the MIV would, for power and timing simulations. Additionally, as the contacts connecting the poly to the wire are within 5nm of the size of a MIV, we can accurately measure the area. This approach allowed us to calculate the area, power, and timing achievable if we had tools that supported the technology.

4.3 Design Flow

To illustrate our proceedure, here are the steps taken for a NAND2 gate.

1. Create a schematic (Fig. 1)
2. Generate the pins from the schematic
3. Create a layout for the standard cell (Fig. 2)
4. Take the standard cell layout and remove the interconnecting polysilicon
5. Add a 65nm by 3420nm metal1 connection between the poly contacts and the inputs. This size was chosen by calculating the wire length necessary, when created from a material with the resistivity of

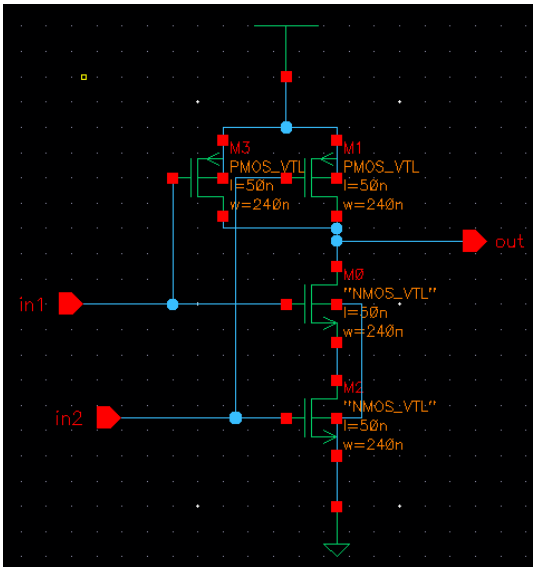

Figure 1: NAND2 Schematic

metal1 (.25Ω / square) [3], to simulate a MIV resistance of 20 Ω [2] . We chose 65nm width to minimize the required length. This is shown in Figure 3.

6. Measure each half of the cell (the 3D IC area is equal to the max of the two sizes)

7. Simulate the 3D version of the cell pre- and post- layout to determine the power and timing. Timing is shown for both rising and falling worst case scenarios for planar (Figures 4 and 5) and 3D (Figures 6 and 7) designs.

4.4 Optimization Techniques

We employed the standard optimization techniques used for standard cell libraries, as the pull-up and pull-down networks remain intact. These include sizing the transistors so that they are balanced, adding additional pins on the larger transistors to reduce the resistance from the contacts, and carefully routing the metal1 so as to avoid unnecessary area due to routing. The last one is particularly relevant, as space that would have been available between transistors due to minimum distances between wells is no longer available. One unique advantage that monolithic 3D IC technology gives us is the ability to reduce the length of the poly. As it notes in [2], the unit resistance of polysilicon is significantly higher than that of metal; by only having polysilicon on the transistor and not forming a wire between them, we can reduce the internal wire resistance. This counteracts the additional resistance provided by the MIV, producing what we anticipate to be a negligible timing difference.

4.5 Naming Conventions

Our naming conventions were to have either <CELLNAME><CELLSIZE>, for a standard version of a cell, or <CELLNAME><CELLSIZE>_3D, for the version that has the additional wiring to simulate a 3D IC.

4.6 Issue Tracking

We kept track of the cells through git, using it to track progress, bugs, to-do items, and notes on each design as we worked on them.

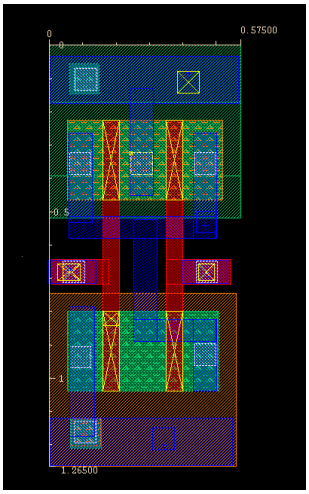

Figure 2: NAND2 Layout

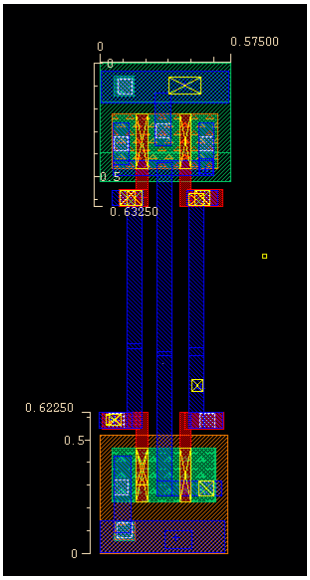

Figure 3: Simulated 3D NAND2 Layout

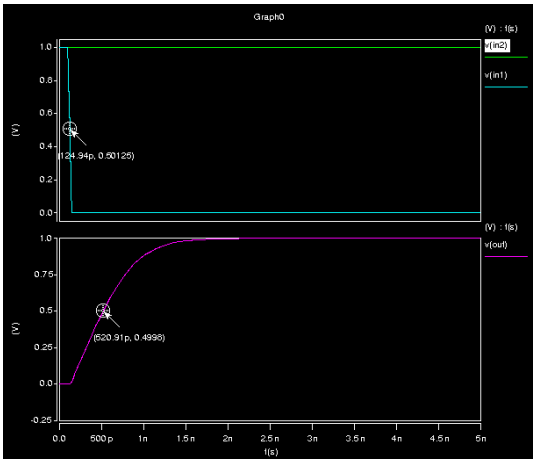

Figure 4: Worst-case NAND2 Rising Time

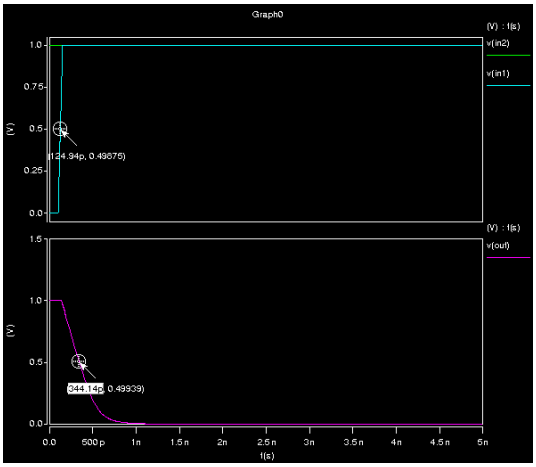

Figure 5: Worst-case NAND2 Falling Time

5 User Document

6 Testing

7 Optimization

7.1 Optimizing Individual Cells

Individual cells were primarily optimized using prior knowledge, as the design process (as noted in Section 3) does not differ significantly from the standard process. The methods that we used have all been thoroughly investigated in Lab 1 of this semester, and thus are not shown in detail. We used multiple contacts for all of the cells that had transistors large enough to fit multiple contacts, to reduce the resistance. Additionally, all transistors were sized to have 2:1 PMOS:NMOS transistors, to minimize delay.

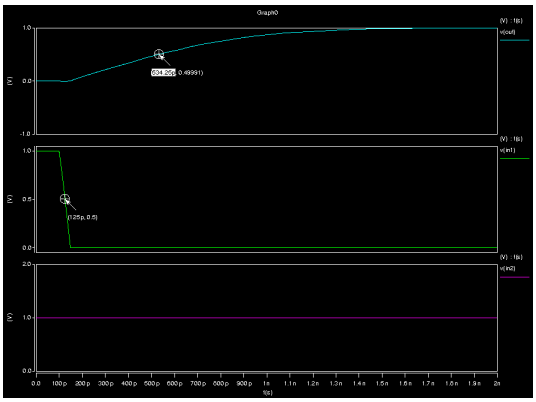

Figure 6: Worst-case NAND2_3D Rising Time

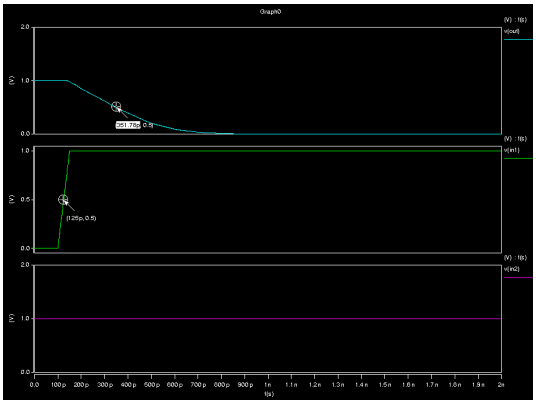

Figure 7: Worst-case NAND2_3D Falling Time

7.2 Using Monolithic 3D ICs to Optimize Full Designs

The more interesting problem is how to use these completed cells to optimize existing designs. There are several facets, which we will analyze here; without support from the tools, it is difficult or impossible to show these in an actual design.

1. **Total Area** The biggest optimization goal of using 3D ICs is to reduce area. In the instances of balanced networks, as is shown in Section 6, the area saving can approach 40 – 50 % on individual cells. However, the more optimal the split between the two sections is, the less room is left for routing, and thus cells must be spaced out to allow for wires. If we assume an optimal gate (using the NAND2 numbers from above) that must route wires, the area savings goes down by 130nm for every second wire added (as they can be split between layers if there are an even number). For a NAND2 gate, that is equivalent to a third of the area savings for each additional wire.

2. **Routing** Another interesting case, and a more realistic one, is that the user has a balance of cell sizes and types. In this case, while the overall area savings are less, you can hide the routing overhead in the underutilized layer (typically with the NMOS, though the routing can also use MIVs to switch to the other layer if it becomes advantageous). This can help mitigate the pressure on routing. Designs have been found by [2] to approach 40% reduction in area, even with the additional difficulties of routing.

A Source Code

B Cell Layouts

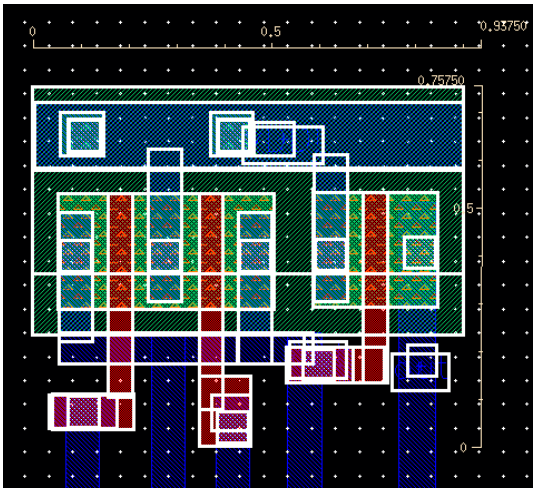

Figure 8: AND2 Pull-up Network

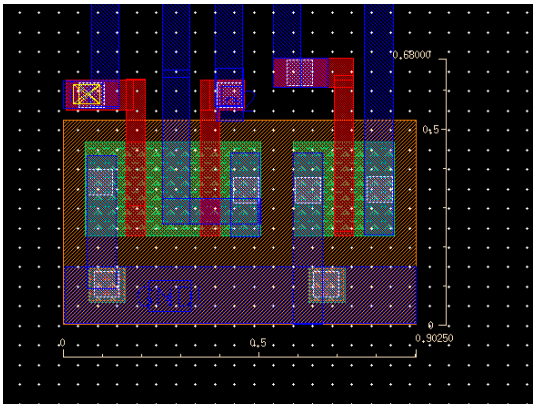

Figure 9: AND2 Pull-down Network

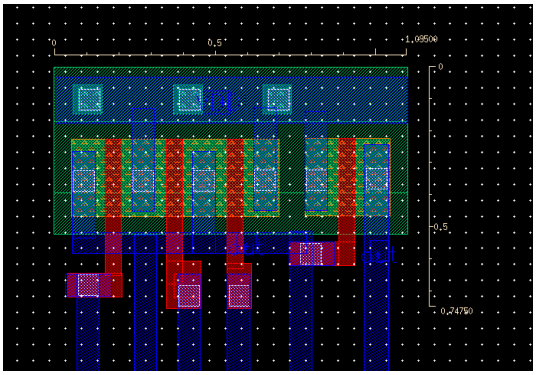

Figure 10: AND3 Pull-up Network

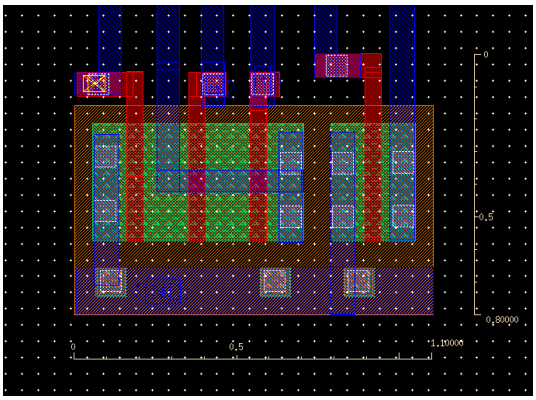

Figure 11: AND3 Pull-down Network

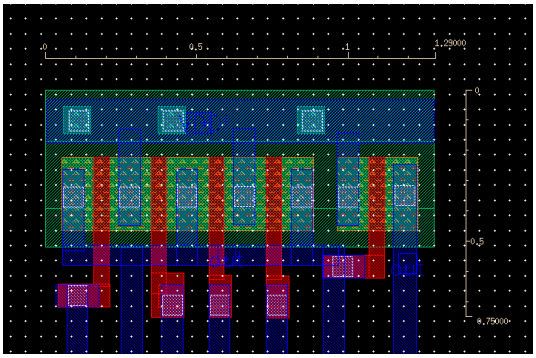

Figure 12: AND4 Pull-up Network

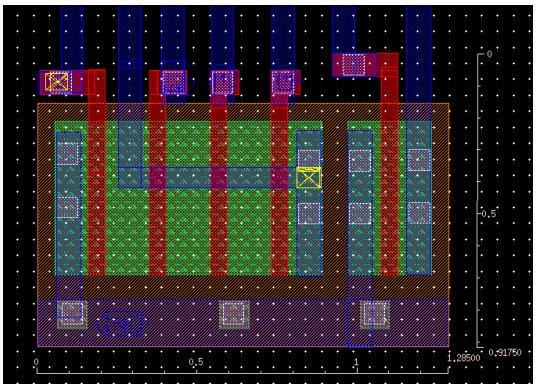

Figure 13: AND4 Pull-down Network

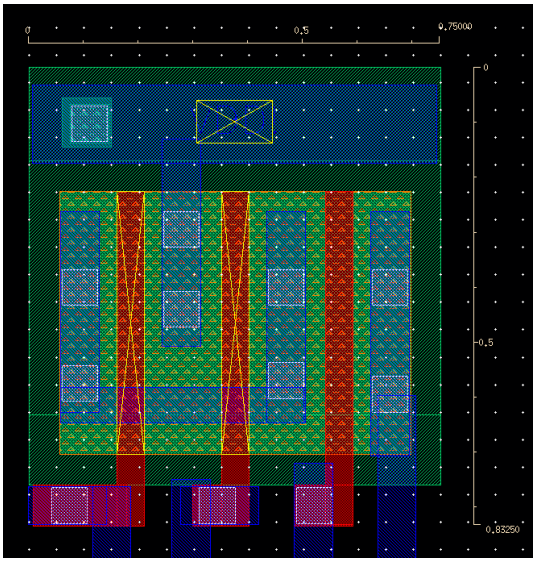

Figure 14: AOI21 Pull-up Network

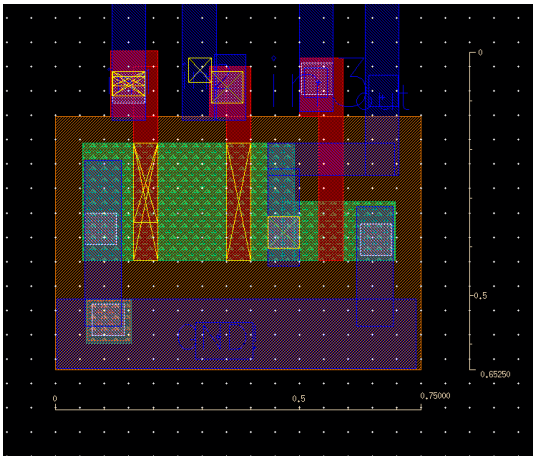

Figure 15: AOI21 Pull-down Network

References

[1] S. Bobba, A. Chakraborty, O. Thomas, P. Batude, T. Ernst, O. Faynot, D.Z. Pan, and G. De Micheli. Celoncel: Effective design technique for 3-d monolithic integration targeting high performance integrated circuits. In *Design Automation Conference (ASP-DAC), 2011 16th Asia and South Pacific*, pages 336–343, Jan 2011.

[2] Young-Joon Lee, P. Morrow, and Sung Kyu Lim. Ultra high density logic designs using transistor-level monolithic 3d integration. In *Computer-Aided Design (ICCAD), 2012 IEEE/ACM International Conference on*, pages 539–546, Nov 2012.

[3] North Carolina State University. Electronic design automation (eda) wiki, 2012.

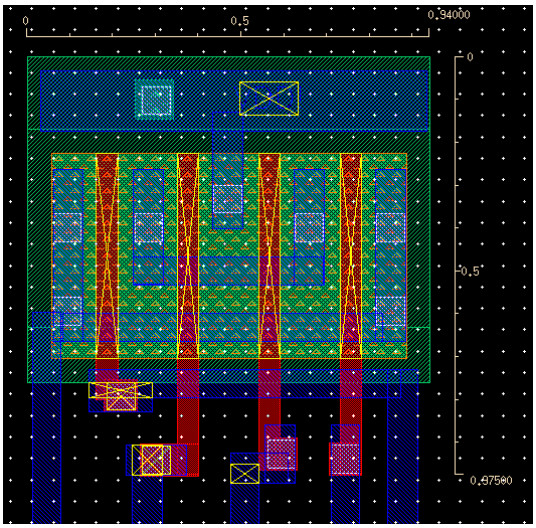

Figure 16: AOI22 Pull-up Network

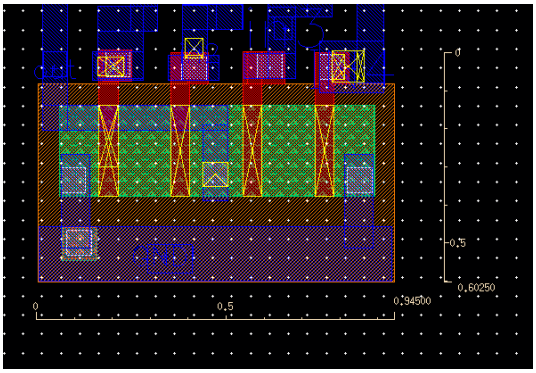

Figure 17: AOI22 Pull-down Network

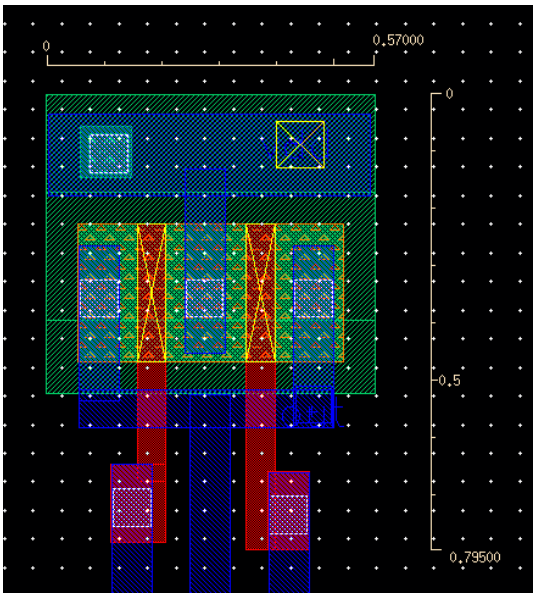

Figure 18: NAND2 Pull-up Network

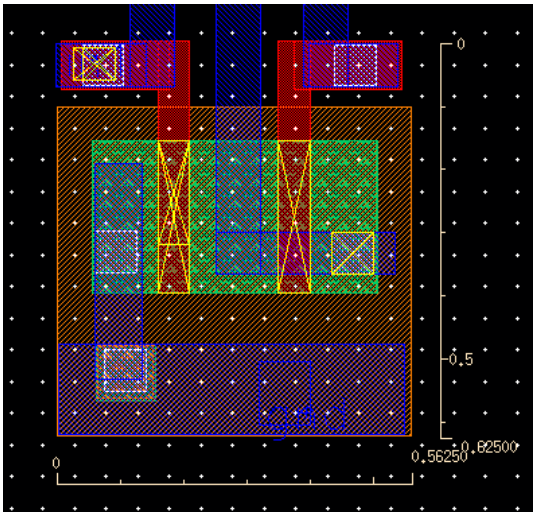

Figure 19: NAND2 Pull-down Network

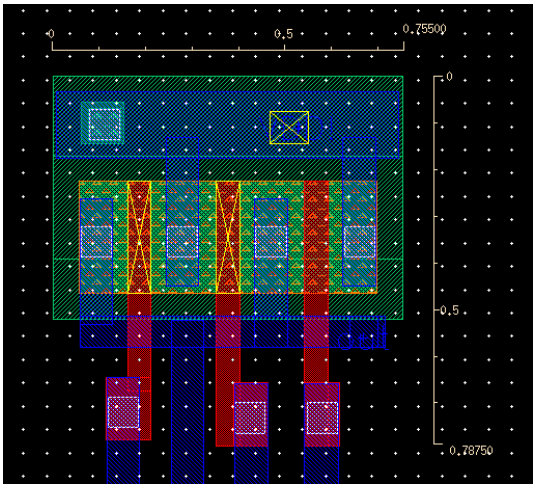

Figure 20: NAND3 Pull-up Network

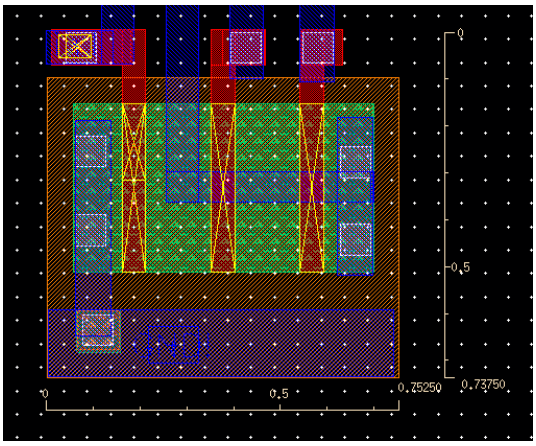

Figure 21: NAND3 Pull-down Network

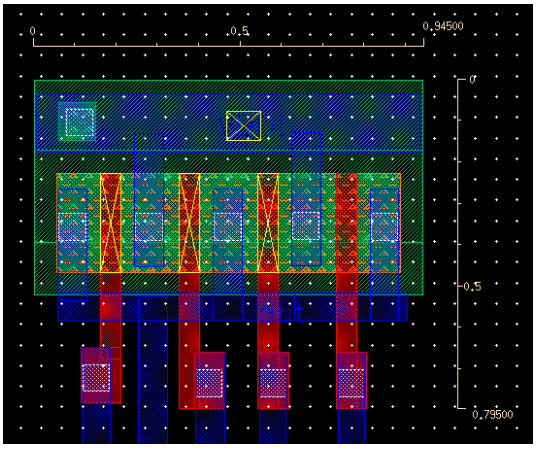

Figure 22: NAND4 Pull-up Network

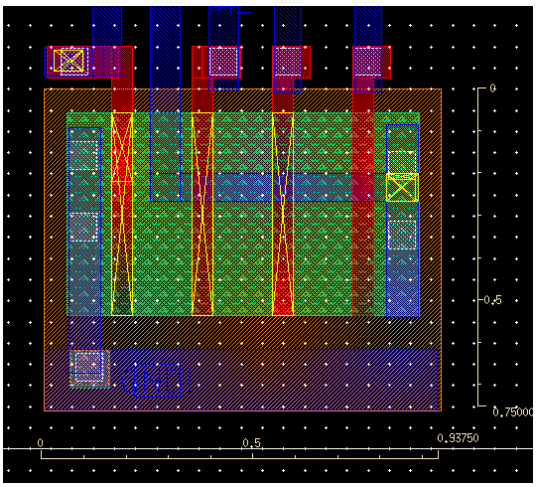

Figure 23: NAND4 Pull-down Network

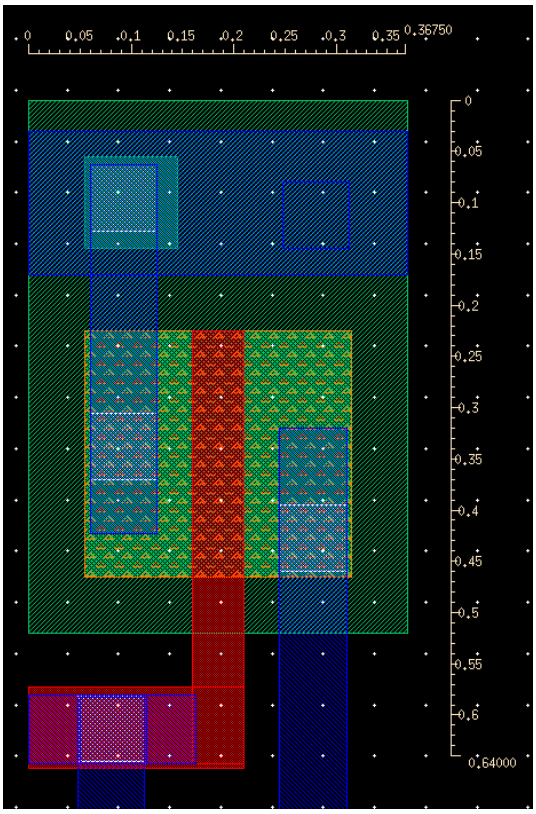

Figure 24: INV Pull-up Network

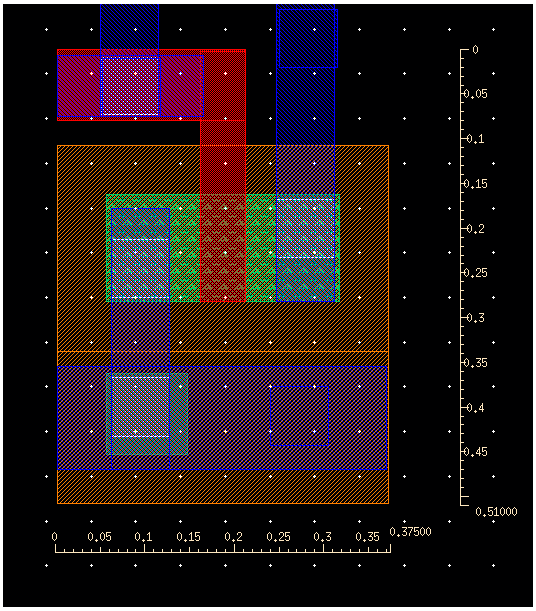

Figure 25: INV Pull-down Network

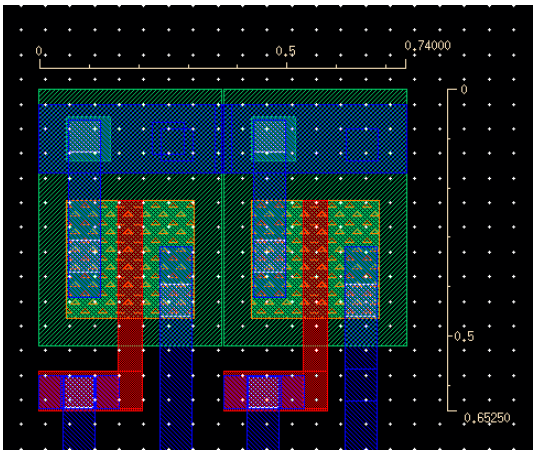

Figure 26: BUF Pull-up Network

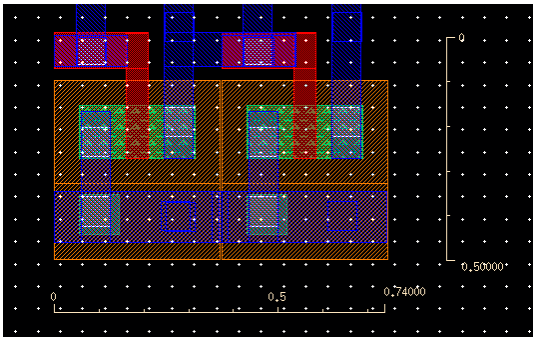

Figure 27: BUF Pull-down Network

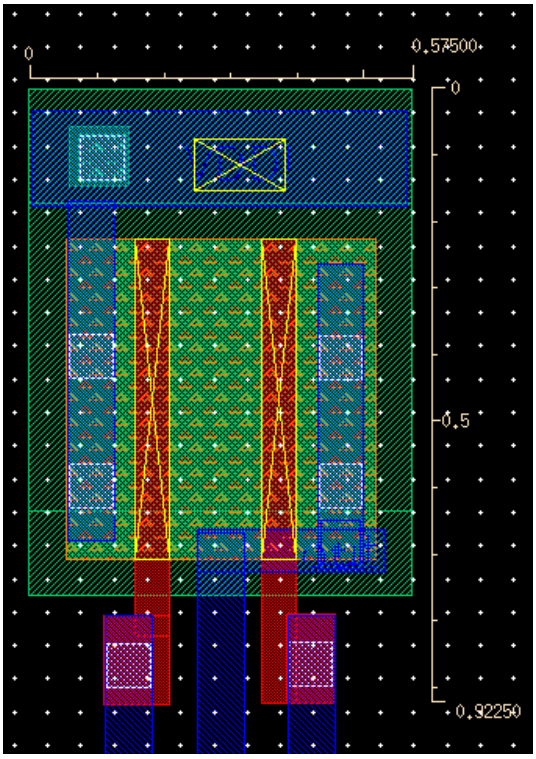

Figure 28: NOR2 Pull-up Network

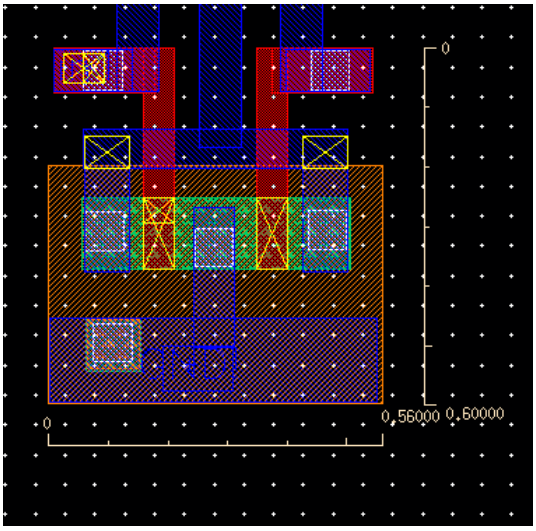

Figure 29: NOR2 Pull-down Network

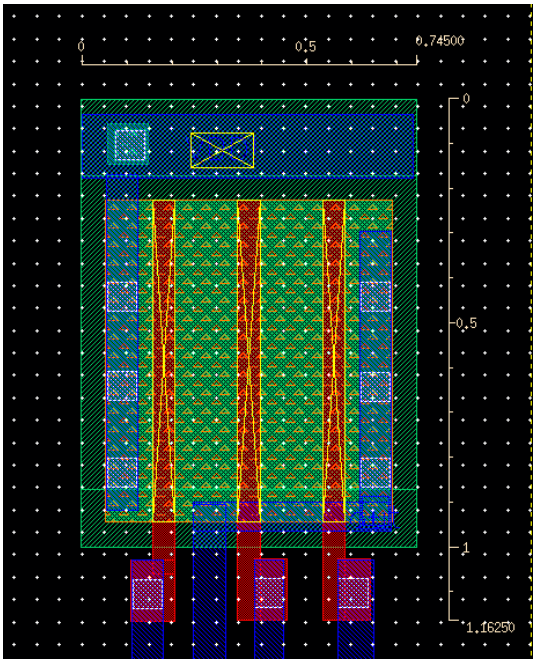

Figure 30: NOR3 Pull-up Network

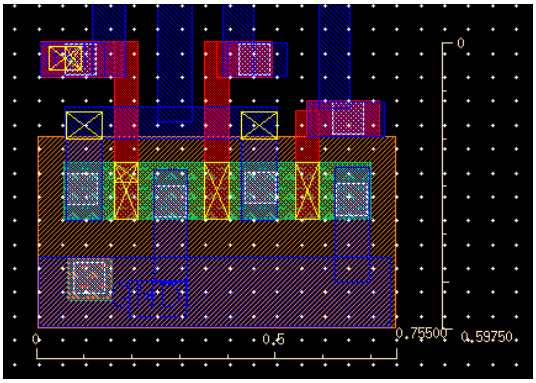

Figure 31: NOR3 Pull-down Network

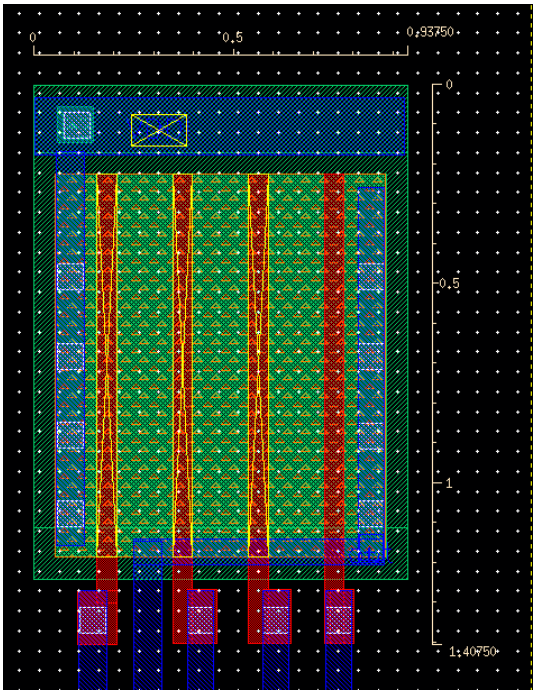

Figure 32: NOR4 Pull-up Network

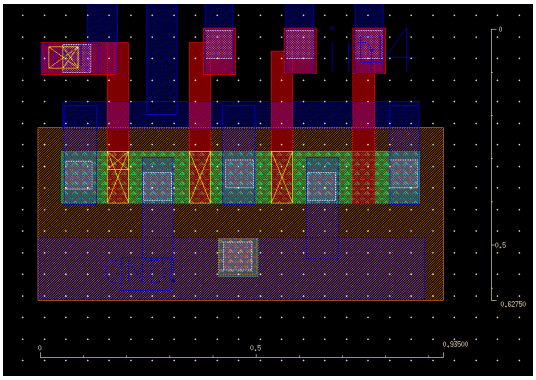

Figure 33: NOR4 Pull-down Network

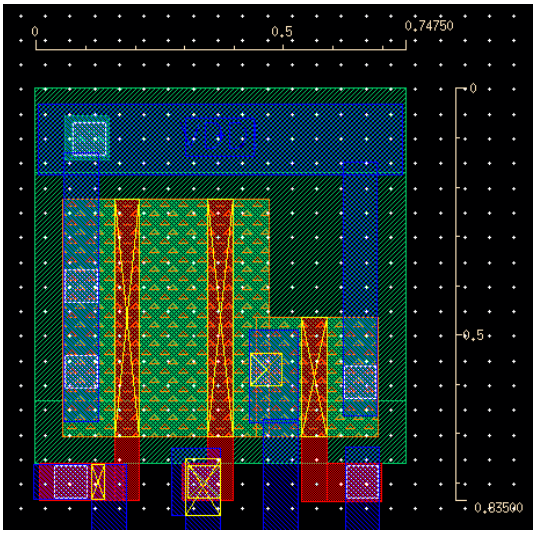

Figure 34: OAI21 Pull-up Network

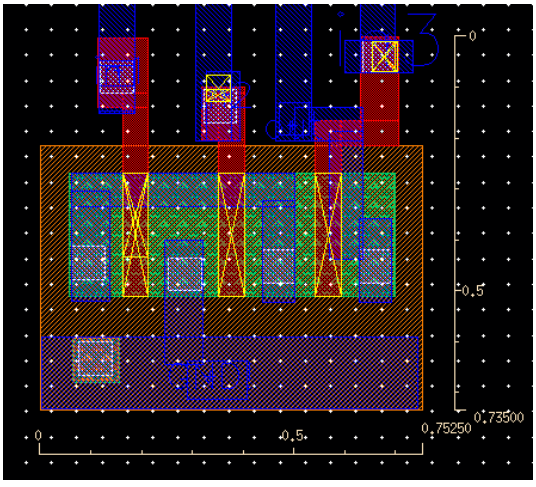

Figure 35: OAI21 Pull-down Network

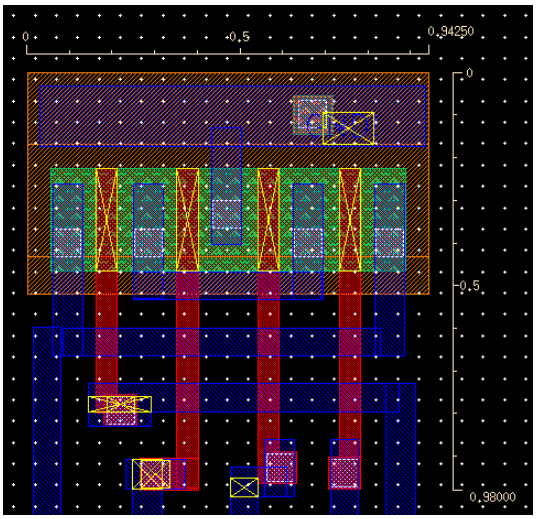

Figure 36: OAI22 Pull-up Network

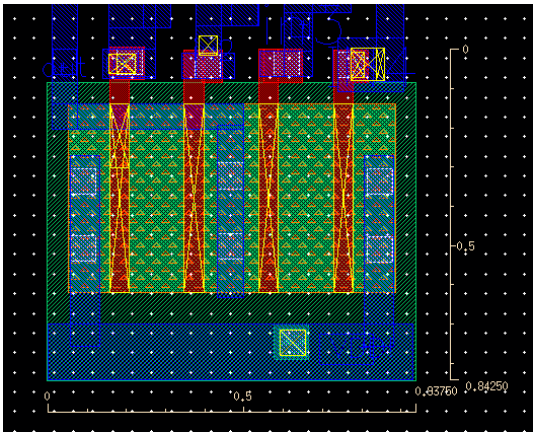

Figure 37: OAI22 Pull-down Network

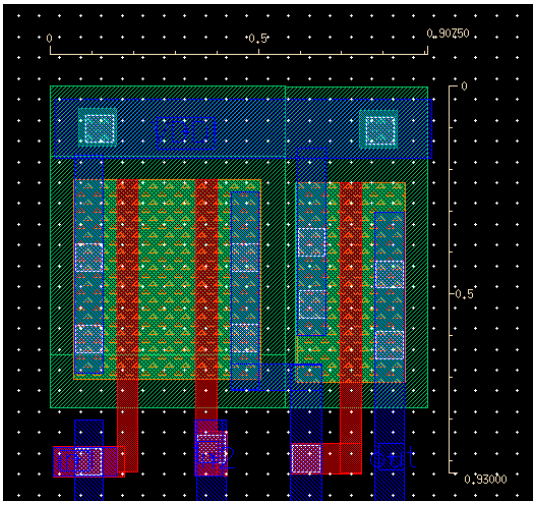

Figure 38: OR2 Pull-up Network

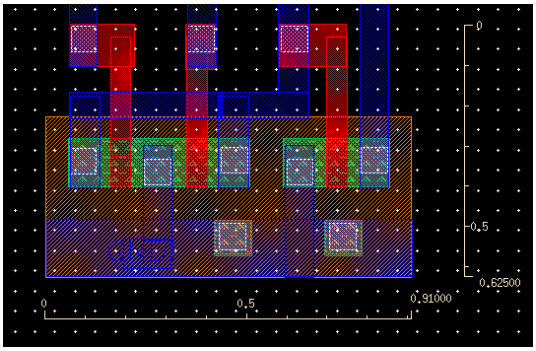

Figure 39: OR2 Pull-down Network

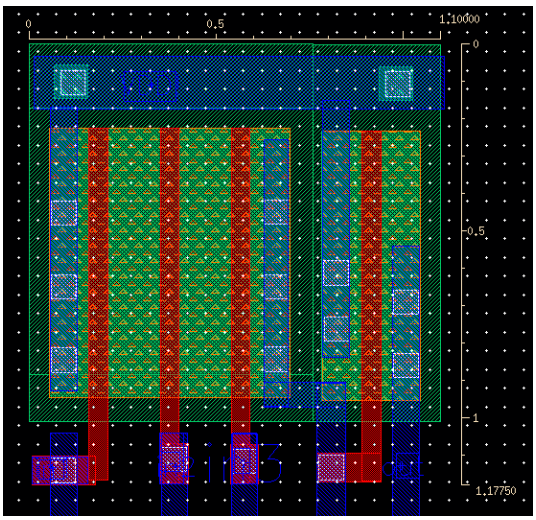

Figure 40: OR3 Pull-up Network

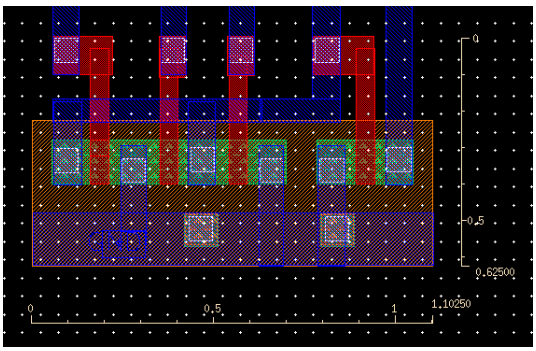

Figure 41: OR3 Pull-down Network

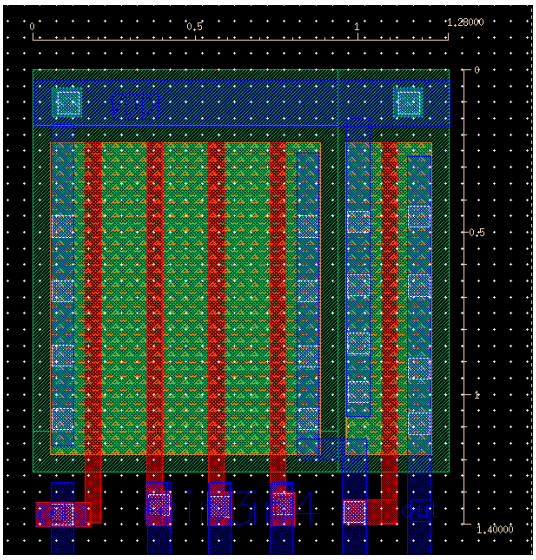

Figure 42: OR4 Pull-up Network

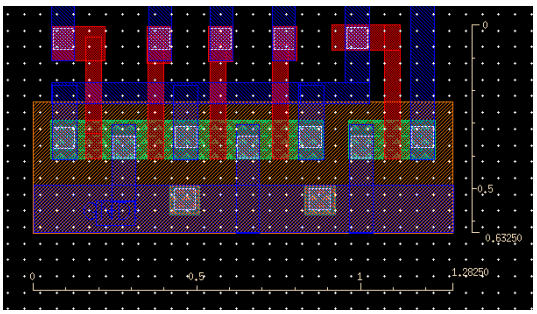

Figure 43: OR4 Pull-down Network