

ECE 425 – VLSI Circuit Design

Lab 5 and 6 – Hierarchical Layout / Digital-Analog Conversion.

Revised February 18, 2004

Introduction

This lab will involve the development of a hierarchical layout that requires the design of several components. Specifically, we will design a circuit known as a *voltage scaling digital-to-analog converter* (DAC). This circuit converts a digital value into an analog voltage that is proportional to the digital value. DACs are an essential component in any electronic system that uses digital values internally and must produce analog outputs. It is also a key building block in analog-digital converters (ADCs).

Figure 1 shows a possible design for a 4-bit voltage-scaling DAC. A network of 15 resistors with value R is connected in series implements a voltage divider that creates 16 distinct voltage values; each of these values correspond to a binary value from “0000” to “1111”.

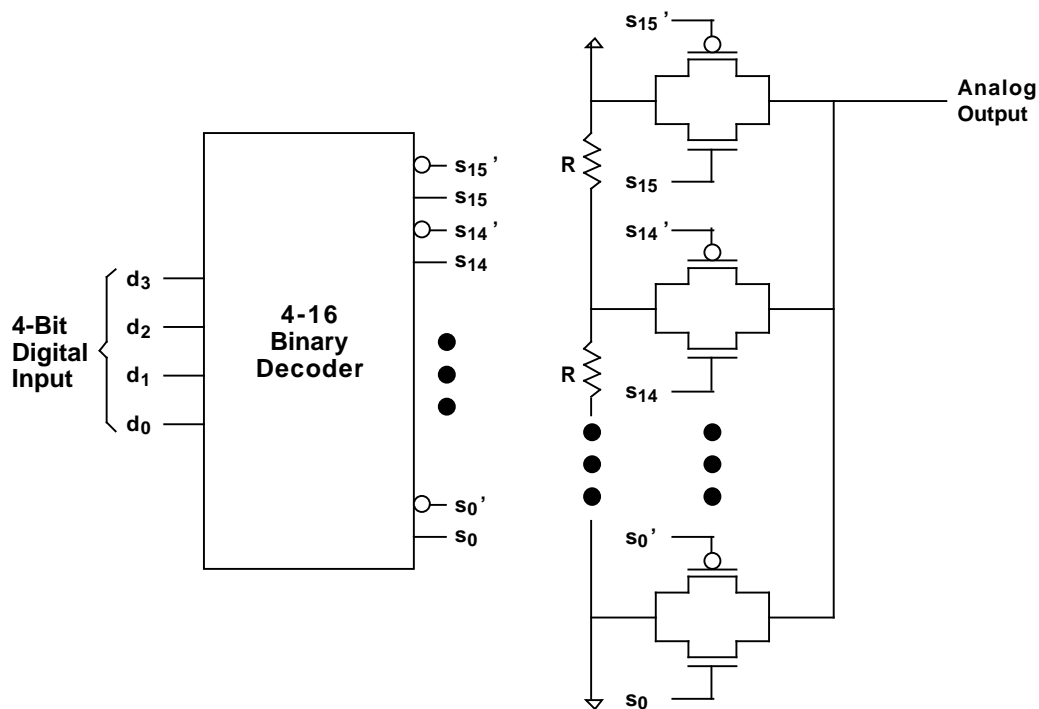


Figure 1 – Voltage-Scaling DAC – First Attempt

The key to the conversion is the network of *transmission gates*. Each transmission gate consists of a P and N transistor in parallel and implements a voltage controlled switch. The gates of these transistors are connected to a *binary decoder* so that for any given binary input, exactly one transmission gate will be turned on. This implements an *analog multiplexer* which selects one of the 16 distinct voltages.

This type of circuit requires a VLSI layout that is “tall and narrow”, particularly when the conversion is being done with a larger number of bits. In addition, the circuit for a 4-16 decoder is fairly complex. For this reason, an alternative approach is often used, as shown in Figure 2.

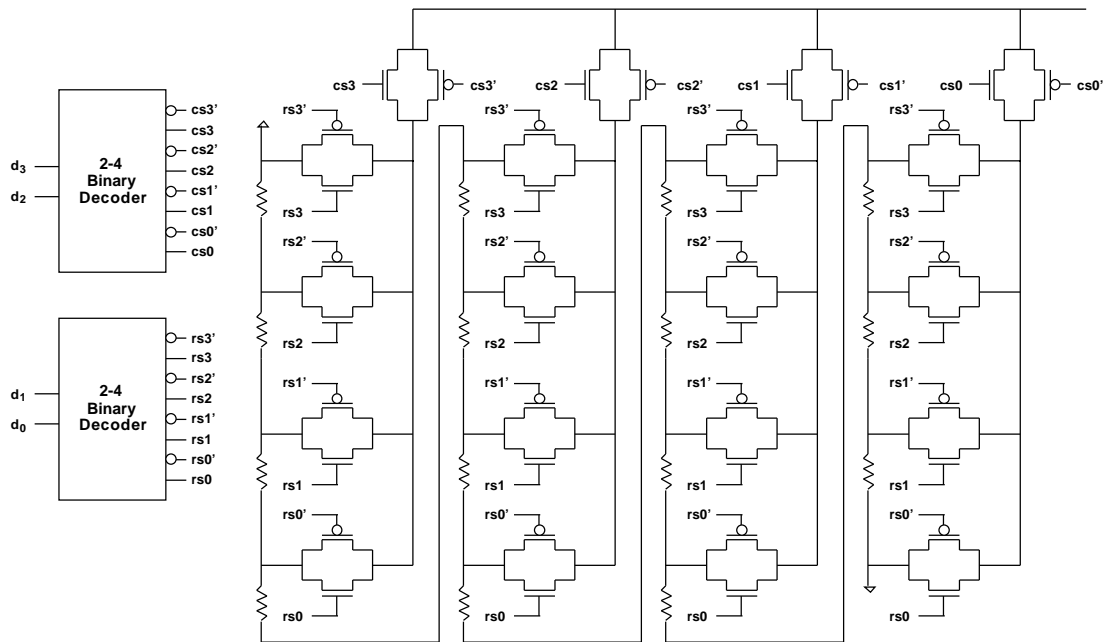


Figure 2 – Voltage Scaling DAC – Modified Design

In this approach the resistor network is split into four columns. The columns are connected in series, so that the overall network continues to operate as a voltage divider. Each column uses the transmission gates to form a four-input analog multiplexer which is controlled by the two least significant bits of the digital input. Each column output then feeds into a separate analog multiplexer; this multiplexer is controlled by the two most significant bits of the digital input. This produces a layout that is “more square” and easier to fit on a chip die. Moreover, the two 2-4 decoders that are required for the row and column decoders are much simpler than the 4-16 decoder required in the first approach.

Over the next three weeks, we will design, test, and verify a VLSI layout for the circuit shown in Figure 2. In Lab 5, we will design a hierarchical layout for the row and column cells of the DAC circuit. In Lab 6, we will design and add layouts for the row decoder and column multiplexer that is necessary to form the complete design. Finally, in Lab 7 (discussed in an additional handout) we will extract the circuit and simulate it using the PSpice circuit simulator.

Hierarchical Layout of a Voltage Scaling DAC

In Labs 5 and 6 we will construct a VLSI layout for the circuit shown in Figure 2. Given the number of repeated elements in this circuit and its regular structure, we can use a hierarchical layout to save a large amount of design effort. However, each cell in the hierarchy must be designed carefully to allow connection by abutment.

Figure 3 shows a possible hierarchical design for the DAC. The key cell in the hierarchy is labeled “RPTx” in the figure. Figure 4 shows the detailed design of the RPT cell. It includes a resistor, which can be constructed from a material like polysilicon, and a transmission gate. This cell must be designed so that the row select lines rs' and rs can be connected by abutment horizontally using metal2, while the multiplexer line out must be connected by abutment vertically using metal1 (note that the cell for the least significant bit “PT0” varies slightly from the other cell in that it contains no resistor)

Lab 5 will focus on the design of a layout for this cell, along with a “column connector” cell which connects adjacent columns together together. In Lab 6 we will complete the design by adding the row decoder and column multiplexer layouts. The design will be done assuming that the chip will be fabricated using the MOSIS AMI 1.5 μ m n-well process.

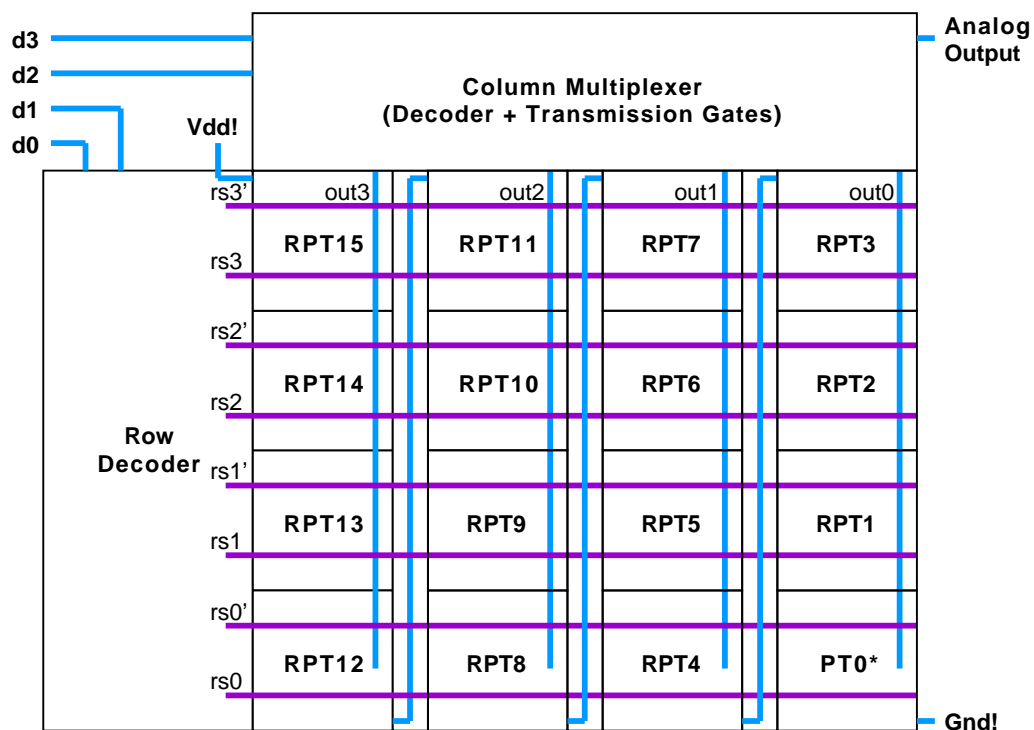


Figure 3 – Hierarchical Stick Diagram of the DAC

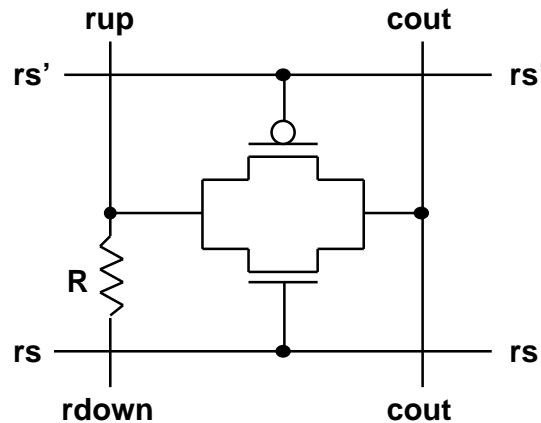


Figure 4 – RPT Cell

Prelab – Lab 5

1. Review the “Cell Hierarchies” section the tutorial “IC Layout Using Magic”.
2. Read [Magic Tutorial #4: Cell Hierarchies](#) (available on the course website).
3. We are going to design the DAC so that the current through the resistor network is approximately 1mA. Assuming $V_{DD}=5V$, choose a value of R that will assure this.
4. The resistor R will be implemented using a piece of minimum-width polysilicon wire connected to metal1 straps. According to MOSIS, the sheet resistance of polysilicon in the AMI 1.5 μ m process is 25.7 Ω /sq, and the polysilicon-to-metal contact resistance is 25.1 Ω . Calculate the size of a minimum-width polysilicon wire needed to realize the value of R computed in the previous step, accounting for both the polysilicon and the contacts.
5. Sketch a stick diagram for the RPT Cell, assuming that the rs and rs' signals connect by abutment horizontally on metal2, and the rup/rdown and cout connections connect vertically on metal1. Include a n-substrate contact next to the p-transistor and a p-substrate contact next to the n-transistor, and include Vdd! and Gnd! connections to the substrate contacts (feed these vertically on metal1 from the top of the cell).
6. Draw a stick diagram of a modified RPT cell “PT0” that does *not* include the resistor but is otherwise identical.
7. Draw a stick diagram of a “column connect” cell, which provides a vertical metal1 connection between the “rdown” terminal of the bottom of the left column and the “rup” terminal at the top of the right column. Be sure to show where “n-well” regions will be extended from left to right.

In the Lab – Lab 5

1. Use Magic to create a layout of the RPT cell that corresponds to your stick diagram. Save this layout using a filename of your choice (e.g., “RPT”). Be sure to size your polysilicon resistor so that it has the proper value.

2. Use Magic to create the layout of the modified RPT cell that corresponds to your stick diagram.
3. Use Magic to create a “column connect” cell that will connect together the four columns.
4. Use Magic to create a “root cell” that will contain instances of the RPT, modified RPT, and column connect cells to form the overall core of the DAC. Use the “:getcell” command to load instances of these cells, and use the “:array” command to create an array for each column. Note that cell instances may be edited “in place” using the “:edit cellname” command. Make sure that there are no design rule violations in your layout, and check that connections are made through cells using the “s” macro.

Lab 6 – Prelab

1. Create a stick diagram of a 2-4 decoder that produces both inverted and uninverted outputs to use as a “row decoder” for your DAC array. Thus the decoder outputs will be connected to the “row select” lines in the array (rs0, rs0', rs1, rs1', rs2, rs2', rs3, and rs3'). Note that this can be designed with four NAND gates and 6 inverters. To save on layout effort, design the decoder as a hierarchical circuit in which the leaf cells are the basic gates (inverters, 2-input NAND) and the root cell is the 2-4 decoder. Keep in mind that your decoder will be connected to the rest of your DAC circuit. To make the layout as small as possible, the decoder outputs should be designed to “pitch match” with the row select lines in your DAC array so that they can be connected by abutment.
2. Extend your decoder design to create a 4-1 column multiplexer that selects the column outputs of the DAC array. Ideally, you should be able to use your decoder circuit as a subcircuit, and add transmission gates for each column. However, some changes may be necessary since the spacing of the column outputs may be different than the spacing of the row select lines.

In the Lab – Lab 6

1. Use Magic to create the leaf cells of your row decoder design, and create instances in the row decoder leaf cell following your stick diagram. Verify the operation of your layout by extracting and simulating with IRSIM.
2. Use Magic to create your column multiplexer design. Verify the operation of your layout by extracting and simulating with IRSIM.
3. Assemble the DAC array, row decoder, and column multiplexer into a complete Digital/Analog Converter.

Report

For your lab report, hand in the following:

1. A short technical memorandum which describes (a) what was done, (b) what you learned, and (c) what difficulties you encountered. Discuss any errors you found in your layout, describing how you found out what was wrong and how you fixed the problems.
2. Plots of all leaf cells.

3. Plots of important intermediate cells (e.g., col, col0, row decoder, column multiplexer). Use “flea -r 2” to show the contents of the intermediate cells.
4. Plot of your complete root cell. Use “flea -r 2” to display the contents of the hierarchy.
5. IRSIM timing diagram plots showing the correct operation of your row decoder and column multiplexer circuits.