# CAIRO UNIVERSITY Elective Course :VLSI Design

## FACULTY OF ENGINEERING Dr. Serag Habib

## Electronics and Communications. DEPT., Fouth Year, Jan. 2024

#### Final Project

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Introduction:**

During the last few years, students of this course developed a digital standard cell library based on the scalable N well CMOS MOSIS (SCN3M) process. This library is named CUSCLIB (Cairo University Standard Cell LIBrary). This standard cell library includes 104 cells. In this project, you are required to layout MSI circuits utilizing the standard cells of CUSCLIB. Layouts and characterization of these cells are given in the attached Handout folder.

**Project Task: Design a binary multiplier / divider array**

You studied in the course lectures how to design a modular integer multiplier using a single logic cell. Such multiplier design is called an array multiplier. The array multiplier studied in the lectures is called add-and-left shift integer multiplier, where the first row is the result of multiplying the least significant bit of the multiplier with the multiplicand. The next row is shifted left from the first row, and so on. You may correctly notice that the order of rows is insignificant. Thus, an alternative design is an add-and-right shift integer multiplier where we assign the first row to the multiplication o**f the most significant bit of the multiplier** with the multiplicand. The next row is now shifted right from this row, and so on. Figure 1 illustrates the left-shift and right-shift arrangements of an array multiplier.

|  |  |
| --- | --- |
|  |  |

Figure 1

The integer division operation is given by:

where A is the dividend, B the divisor, Q the quotient, and R the remainder. Note that fixed point division can be transformed to integer division by simple **scaling** of the operands. For example, using a decimal number representation, consider the fixed-point division of 12.1d by 7d . Let us require that the result should be available in fixed point format, and be accurate up to three decimal points. This fixed-point division operation can be transformed to the integer division of 12100 by 7 as follows:

Unlike multiplication, division includes decisions. At each division step, you have to decide if you need to borrow one more digit from the right part of dividend or not. There are two famous binary division algorithms that handle these decisions differently. These two algorithms are called the restoring and non-restoring division algorithms. A simple introduction to logic circuit implementations of these division algorithms is included in the handout of this project (Ref 1- Restoring and non-restoring binary dividers.pdf). This Ref 1 is extracted from the following reference:

A. E. A. Almaini, "Electronic Logic Systems," Third edition, Prentice Hall.

Multiplication is based on repeated additions. Division is based on repeated subtractions. Since 2’s complement adders can carry out both additions and subtractions, you would correctly expect that a single array logic circuit can either multiply or divide. This project involves the design of such multiplier/divider array.

You are required to merge the two cells of the array multiplier (modified using the right shift approach) and **the non-restoring** array divider of Ref. 1 to create a unified cell for an array multiplier/divider. **The objective of this project is to design down to the layout level an array multiplier / divider for unsigned binary numbers with the following parameters:**

* **Multiplier mode: 5bit \* 5bit**
* **Divider mode: 10 bit dividend and 5 bit divisor**

You are to use the static CMOS approach at the logic level. The IC layout should be done using the SCN3M technology studied throughout the course.

The IC design should include the circuit floorplan with the I/O pads provided. Use the 0.5 µm N well CMOS process defined in your lecture notes. You are allowed to use the CUSCLIB library cells **only**. You may also design your own standard cells that conform to the CUSCLIB standard cell format given below.

**Design Procedure:**

Follow a hierarchical approach, namely:

* + - 1. Logic design Phase:

1. ~~Draw the block diagram of your array multiplier / divider.~~
2. ~~Identify the cells used and their logic diagrams.~~
3. ~~Enter your design into a logic simulator~~ of your choice. Although **~~RTL design entry is preferred~~**, you may use also Proteus, Logisim, or any other logic simulation tool. If you select an RTL entry tool, you should enter your design at the logic level **~~structurally not behaviorally~~**~~.~~
4. **~~Simulate your design and verify its proper operation~~**~~.~~ Be careful to include reasonable number of test vectors to verify your design at a well selected test vectors. For the multiplier mode, you should cover corner cases like: zero x zero, max multiplicand x max multiplier, max multiplicand x zero , …etc. For the divider mode, you should cover corner cases like divide: 1 by 1, max dividend by max divisor, max dividend by 1, 1 by max divisor, ..etc.). **This verification step is a vital step of the design.**
5. Prepare a subfolder for your logic design. This subfolder should include a **pdf report** describing your design as well as all logic **design files generated during this phase.** Please name your circuit after the first names of your group partners (e.g., if Ahmed Hamdy, Yassin Sabry and Bassem Fawzy formed one group, then their logic design file should be named "Ahmed\_Yassin\_Bassem Logic.pdf ").
   * + 1. Layout Phase:
6. ~~Layout the basic cells of your array multiplier /divider module.~~ Again layout should be hierarchal. Instance (not copy) CUSCLIB cells to build these cells. Ledit layout tool should of course be used. Refer to the Ledit manual (handed to you as part of the Ledit lab.) for the proper hierarchical design procedure.
7. Assemble your multiplier / divider array to form the core of your IC.
8. DR check your design
9. Extract and simulate your multiplier/divider core. Use PSPICE circuit simulator. Do not Use TopSpice. Refer to the attached comments on the use of the extractor so that the extracted PSPICE file does keep the original node names of your design.
10. Calculate the worst case delay, and hence, the maximum frequency of operation. Find out the dynamic power at the max. clock frequency. Refer to the document attached explaining how to determine the critical path in your design.
11. Add the IO cells. Repeat steps **c** to **e** above.
12. Prepare a separate subfolder for this layout phase of your design. This subfolder should include a pdf report describing your design as well as all layout design files generated during this phase. Again, if the names of your team members are Ahmed Hamdy, Yassin Sabry and Bassem Fawzy, then their layout design file should be named "Ahmed\_Yassin\_Bassem Layout.pdf ").

**A Bonus of 20% of this project grade is offered to the core design with highest working frequency (excluding pads).** Note that this is only a bonus, not part of the project grade. Speed maximization or delay minimization can be achieved by modifying the architecture, logic, circuit or layout levels.

* At the architecture level, you may modify your divider structure to achieve faster performance. The divider mode is of course slower than the multiplier mode.
* At logic level you may consider for example different implementations of the basic multiplier / divider cells used.
* At circuit level, you may select different CUSCLIB cells with different transistor sizing. You may also design your own standard cells compatible with CUSCLIB rules.
* At layout level, you may, for example, try to minimize parasitic capacitances or use metal wires in place of poly wires.

Groups of three students can work on a single project.

**Bonus and Penalty policy**

* A bonus will be given to the design(s) achieving the maximum frequency of operation, provided, of course, that it does not include design errors!.
* A bonus will be given to groups that detect and correct errors in any of CUSCLIB cells.
* A Bonus will be given to groups that reduce the area of any of CUSCLIB cells by more than 50%.
* Plagiarism ( ( السرقة الأدبية is strictly forbidden. A strict penalty policy will be enforced for any case of plagiarism.

**Project Inputs**

* A handout folder is attached including:
* introductory Power point presentation that describes the project.
* CUSCLIB cell library
* MOSIS scalable N well process description, design rules, and device models.
* I/O pads ( for layout only. Do not use for estimating area or delay)
* Section 6.8 of book: A. E. A. Almaini, "Electronic Logic Systems," Third edition, Prentice Hall, covering the main binary division algorithms.

**CUSCLIB Standard cell specifications**

Each standard cell satisfies the following guidelines:

1. Cell height 40 λ
2. VDD runs on a 5λ strip of Metal 1 located at height 34.5 λ to 39.5 λ
3. GND runs on a 5λ strip of Metal 1 located at height 0.5 λ to 5.5 λ
4. Inputs and outputs of each cell are available via vertical Metal 2 wires as shown in Fig. 3



Fig. 3

**Project Outputs**

Each Group should submit a report organized as follows:

* + - * 1. **Abstract**
        2. Logic design subfolder including:
* **Logic design pdf report**
* logic simulation results to verify the proper operation of the multiplier / divider array.
  + - * 1. Layout design subfolder including:
* Layout design pdf report
* layout files (.tdb files) with and without pads.
* Post-layout Simulation results to verify the proper operation of the multiplier.

For PSPICE simulation curves, please do the following:

Go to the “Windows” pull down menu, select copy to clipboard, open wordpad, paste and save to a suitable file format; e.g. design\_simulation.doc. Submit this file along with .cir and .spc files.

* Core performance metrics (post-layout delay, fmax, dynamic power, and core area)

**Please Note that Report files are very important components of your project,** as documentation always is. Good part of the project mark is dedicated to the logic and layout reports.

Upload your design to elecvlsi2021@gmail.com no later than midnight of Sat. Feb. 3, 2024.

**Submission to other Emails, whatsapp pages, links to cloud drives, or file sharing sites …etc is not acceptable and will be ignored.**