Educational aspects of VLSI training at postgraduate level

Prof.dr. Alain GUYOT
Institut National Polytechnique du Grenoble

Mihai T. LĂZĂRESCU
"POLITEHNICA" University of Bucharest

Conf.dr.ing. Daniel C. IOAN
"POLITEHNICA" University of Bucharest
Facultatea de Electrotehnică, sala EB206,
Spl. Independenţei 313, 77206 Bucureşti,
ROMÂNIA

ABSTRACT

This paper will describe the way a VLSI circuit project is intended to be used for training in the *Postgraduate School for Computer Aided Electrical Engineering* in Bucharest, Romania. The stress will be focused on the main design steps, on the use of various CADENCE Edge^{TM} VLSI design environment facilities, and on strong team collaboration stimulation.

I – INTRODUCTION — The Romanian PSCAEE school

The TEMPUS Joint European Project JEP 2717, entiteled "Initiation of Formal Training in Computer Aided Electrical Engineering in Romanian Universities" is coordinated by Polytechnic University of Bucharest and has as contractor COREP – Politecnico di Torino. Among other institutions, the partners interested in VLSI circuit design are: INPG-CMP, Università di Genova, National Technical University of Athens, Thames Valley University and ISEN Lille. The project is financed mainly by CEC, having for the three academic years 1991-1994 a total budget of 586,000 ECU.

The main goals to of this project are:

- establishing a Postgraduate Training Center of CAD/CAE in Electrical Engineering (PSCAEE) at Polytechnic University of Bucharest;
- organizing and operating a CAD/CAE laboratory and associated documentation center;
- curriculum development and textbooks editing;
- EC-RO mobility program for professors and students;
- organization of a workshop in CAD/CAE, with international participation.

The postgraduate center of CAEE could train about 20 high level specialists annually. Two main directions of study are available: electromagnetic field modelling and (micro)electronic systems design. Boths directions' curriculum includes a restricted number of compulsory courses (assuring theoretical background and Unix operating system knowledges) and a large choice of optional courses. In addition, individual studies and computer-laboratory works are included in the student's training. During the two years of part-time study, the students will have to achieve both a mid-term project and a graduation project working in Polytechnic University Bucharest or in one of the partners' university.

In the frame of mobility program of TEMPUS JEP 2717, five Romanian professors and one graduate student spent each 3-6 months at INPG-CMP, France. Three specialists from INPG visited Polytechnic University of Bucharest. Following this program, 14 volumes of textbooks, summing more than 3,500 pages were edited and printed in 30 copies.

The contact with the JEP partner representative, Dr. Bernard Courtois from INPG-CMP offered to this newly founded Romanian CAEE training center from Polytechnic University Bucharest the opportunity to apply and to be accepted as partner in the EUROCHIP scheme. The success of this TEMPUS JEP allows to improve essentialy the VLSI systems design training at Polytechnic University. Software packages for VLSI systems design such as Cadence-Edge, Hilo and HSpice were aquired by JEP 2717 using the EUROCHIP offer and were instaled for PSCAEE laboratory on workstations (DECStation 5000/25, SUN LX, HP-Apollo 9000/710, and 705) purcased by JEP budget.

The organization of the Postgraduate Training Center in CAD/CAEE is the first attempt of this kind in Romania.

As for every begin, there were many difficulties consisting mainly in the lack of information in Romania regarding: Unix-RISC workstation (software and hardware); appropriate books and periodicals, and important delay in CAD tools delivery due to restrictions imposed for advanced technology transfer towards Eastern European countries.

Among the successes, one of the most important and encouraging is the fruitful collaboration with the JEP partners representative. The joint with EUROCHIP scheme is an excelent opportunity to be integrated in the European efforts aimed to improve VLSI systems design training.

II – The laboratory work

The students will have free access to the whole laboratory documentation. They will receive the theme, the work scheduling and will be helped with hints whenever necessary. There is no continously supervising provided for the lab. A few hours several days a week, a qualified person will be available for inquiries and help.

A theme suitable for this almost self-teaching working method has to be simple enough to be easy in-deepth understood by each student, and a splittable one in order to allow a high degree of parallel development by assigning the subparts to separate teams.

Since mathematical operators are inherently involved in data processing irrespective of the data type, they are to be found in almost every processor. Moreover, the overall automaton performances are definitevely influenced by the quality of the used operators.

Given the above considerations, we intend to propose as laboratory work for the VLSI division of PSCAEE a design of a fast binary divider.

The internal structure of the proposed divider can easily be splitted into several quasiindependent laboratory works. As the final goal is to put all the pieces together into the full custom layout of the divider, the laboratory work implies also a permanent collaboration among students to coordinate their implementations, that becames more tight as the project is narrowing the end phase. Also, optimizations in schematics and layout for each part as well as for the whole design have to be realized, as the low computation delay and a minimum area on silicium are important criteria for the circuit's quality.

III – Circuit's structure

Two operands are needed as divider's input: the dividend and the divisor, both written in a standard binary notation, that is using only 0 and 1 as binary digit values. They are supposed normalized binary numbers, according to IEEE 754-1985 standard. Internal computations inside the divider are done in a binary borrow-save notation, that is using -1, 0, and 1 as possibles values for each binary digit. The quotient is then computed in the same redundant notation. This is why an extra block was added to perform the quotient conversion from the redundant notation to the standard binary one, supposed to be used in the rest of the circuitry.

The divider is composed of three parts (fig. 1), each of them playing a distinct role in internal dataflow:

- the *head cell* receives as inputs the three MSB of the partial remainder, written in redundant notation. The cell has three roles at its turn.

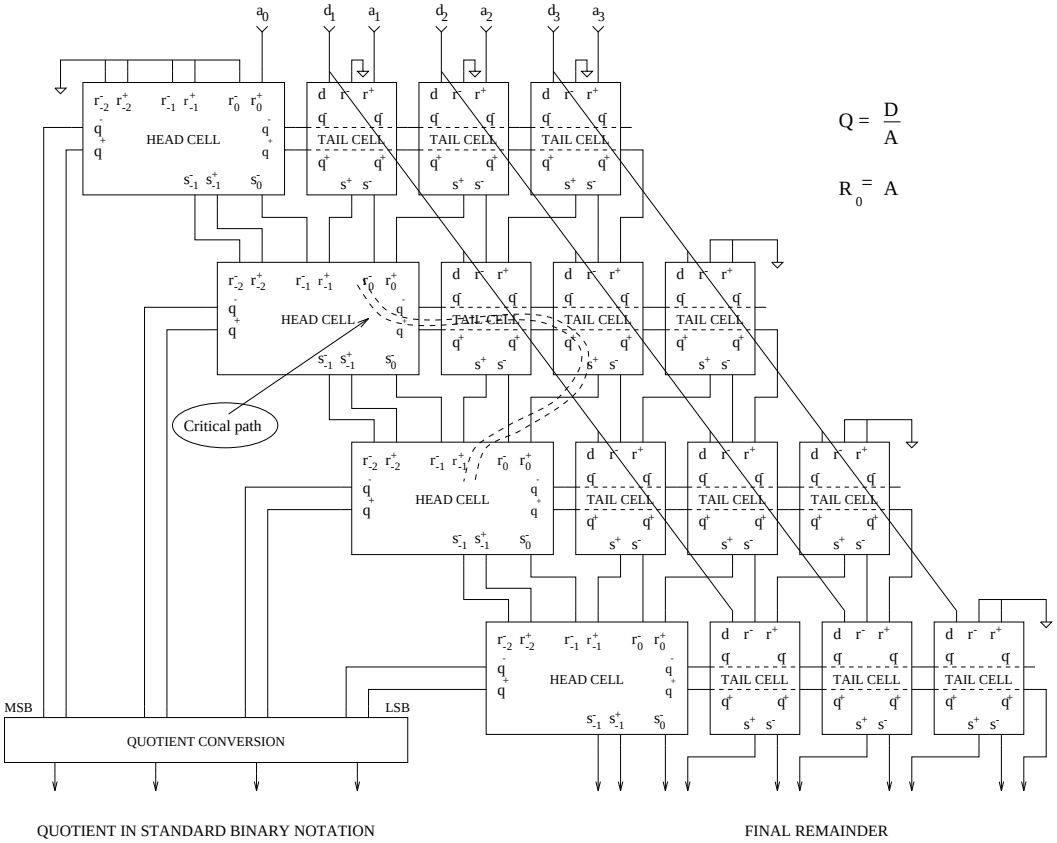

Figure 1: the block schematic of a 4 bit divider

First it computes a quotient digit. This quotient digit is used then by the subsequent tail cells in the same row to choose the operation to execute on the rest of the partial remainder digits — an addition, a subtraction, or nothing.

The second role of the head cell is to do that computation on the MSBs of partial remainder as the tail cells do with the rest of the remainder's bits. The MSB of the normalized divisor D is always 1, so it is considered as an implicit constant signal in the head cell during computations.

Finally, the head cell take advantage of the redundant notation of the partial remainder and rewrite its MSB in an equivalent form, so that the next remainder has one most significant digit less. This is to avoid truncation errors when shifted one bit to the left for the next stage.

The five signals generated by the head cell (fig. 1) have the following boolean expressions:

$$\begin{aligned}
q_j^+ &= r_{-2}^+ + \overline{r_{-2}^-}\,(r_{-1}^+\,\overline{r_{-1}^-} + \overline{r_{-1}^-}\,r_0^+\,\overline{r_0^-} + r_{-1}^+\,r_0^+\,\overline{r_0^-})\;;\\
q_j^- &= r_{-2}^- + \overline{r_{-2}^+}\,(\overline{r_{-1}^+}\,r_{-1}^- + \overline{r_{-1}^+}\,\overline{r_0^+}\,r_0^- + r_{-1}^-\,\overline{r_0^+}\,r_0^-)\;;\\
s_{-1}^+ &= r_{-2}^+\,(r_{-1}^+ + \overline{r_{-1}^-} + r_0^+\,\overline{r_0^-}) + \overline{r_{-2}^-}\,r_{-1}^+\,\overline{r_{-1}^-}\,r_0^+\,\overline{r_0^-}\;;\\
s_{-1}^- &= r_{-2}^-\,(r_{-1}^- + \overline{r_{-1}^+} + r_0^-\,\overline{r_0^+}) + \overline{r_{-2}^+}\,r_{-1}^-\,\overline{r_{-1}^+}\,r_0^-\,\overline{r_0^+}\;;\\
s_0^- &= q_j^- \oplus r_0^+ \oplus r_0^-\;.
\end{aligned}$$

- the *tail cell* receives as inputs a digit of the divisor (in binary standard notation), a digit of the partial remainder (in binary redundant notation), and the quotient digit computed by the head cell of its row. The tail cell's role is to perform an addition, a subtraction or no-operation on its two input digits (one of the divisor and one of the partial remainder), according to the value of the received quotient digit.

  The two signals generated by the tail cell, s_{i-1}^+ and s_i^- satisfy the equality:

$$2 \cdot s_{i-1}^+ - s_i^- = r_i^+ - r_i^- + (q_j^+\,\overline{d_i} + q_j^-\,d_i)$$

- the *quotient conversion block*, as the name shows directely, has to convert the computed quotient in redundant notation to binary standard notation, the last notation beeing supposed to be used by the rest of the circuit that the divider is part of. This block is composed from small cells disposed in a regular structure.

IV – One possible laboratory work. Complexity evaluation

The students can be organized in three teams, each team having to design one of the divider's parts: the head cell, the tail cell, and the quotient conversion block.

Clearly it seems that the head cell is the most complicated cell with respect to the others, since it has to generate 5 signals, all of them using all the 6 input signals. Anyway, due to the perfect symmetry of the borrow-save notation, the q_j^+ and q_j^- signals have the same structure for their expression, only the input signals are different. The same stands for s_{-1}^+ and s_{-1}^- too. This means that this two pairs of signals will have the same schematic at transistor level, but the transistors themselves will be driven by different input signals. The fifth head signal, s_0^- is generated by a very simple gate anyway.

So, besides the simpler gate for the s_0^- signal, there are to be designed only two structuraly different complex gates: one for the q_j^+ and q_j^- signals, and the other for s_{-1}^+ and s_{-1}^-. On the other hand, the surface of the head cell layout isn't a very important constraint to be observed, because for a n-bit divider there are just n head cells in the layout.

In figure 1 we can see that the critical path of the divider is passing through the head cell. More precisely, the computation time of the q_j^- and q_j^+ signals is very important. This is why the complex gates which generate this two signals have to be carefully optimized in terms of speed.

The tail cell is much smaller than the head cell. Depending on the implementation, the transistor count is two to four times smaller than in the head cell. The input signals are 5 for each tail cell and the generated ones are just two. This means that the complexity of the gates inside a tail cell is lower than those in the head cell. Instead, the main issue with the tail cell is the layout area, because for a n-bit divider the number of its tail cells is $n\,(n-1)$. This makes the total area of the tail cells to represent almost all the divider's area, especially for a large number of bits (n).

The divider's critical path runs through the tail cells too (fig. 1), so that its computation time is affecting the overall computation speed of the whole divider. Schematics optimization for area and speed are thus mandatory for this cell.

Finally, the quotient conversion block is a rather scholastic one. It consists of several identic cells disposed in a regular structure. If applied, area optimization criteria will impose however the use of two different types of cells in order to take advantage of the higher computation speed of the quotient conversion versus quotient digits output rate. This may complicate a whit the quotient conversion block layout design. Quotient conversion is done "on the fly", so the main optimization for this cell is to choose the minimum number of rows in the quotient conversion cell based on the ratio between the computation time of q_j signals and the traversal time of a row of cells in the quotient conversion cell.

Given the considerations above it shows up that the three parts of the divider's structure could constitute well balanced design projects for laboratory work. The teams in charge to design the divider's parts are stimulated to cooperate to deal efficiently with issues like minimizing the divider's area, optimizing the critical path, choosing the rows number in the quotient conversion cell based on delays, power distribution, common cell pitch, communication pins position for assembling by abbutment, etc.

In the same time, each of the divider's parts has its own specific hot-point. The designers of the head cell will have to deal with complex gates and the computation time minimization for q_j^- and q_j^+ signals. The tail cell layout has to be as small as possible and the computation time for its signals minimized too. The more novices could be grouped in the team charged to design the quotient conversion cell, which is likely to be a good starting project for them.

For an arithmetic operator, the worst case computation time is the only one considered at performances evaluation. The students will have to pay attention to the evaluation and minimization of the worst case delay for the critical path signals of their cells. This implies worst case input vectors sequence determination, and several schematic structure simulation.

V – Some variants

There are several implementations for the divider's cells that can be choosed from. There are also for each cell's schematic several choices. Let's skim some of them.

Fairly the most complex one, the head cell can be implemented using either static or dynamic CMOS logic. Of course, to efficiently use the dynamic logic major changes in the whole divider's structure are needed, but for an educational purpose project it's use may be restricted just for the head cell.

The tail cell is structurally a Full Adder completed with a multiplexer for one of the input signals. The cell can be implemented in this block structure fully with CMOS gates or with transmission gates. Another choice may be to draw out the cell's schematic from the signals' truth tables, which case one of the most advantageous choice is that using just transmission gate based multiplexers, and inverters.

For the quotient conversion cell it is possible also to use a combination of pass transistors and CMOS gates in order to minimize the transistor number.

So far we discussed only about the full custom (i.e. at transistor level) implementation of the cells. This is the most performant one, but for CAD tools teaching purposes one might consider a standard cell based design too. In this latter case, the freedom of choice in schematic conception is somehow restricted with respect to the full custom design, and there are less optimizations to be searched. But the standard cell approach combined with the Automatic Place and Route facility of CADENCE Edge™ design environment is by far the fastest way to get the layout of a circuit using this CAD tool.

The most time consuming and tedious stage of the project is certainly the layout design if is done by hand. This can be rended considerably shorter and a lot more pleasent making use of "The Layout Synthesizer" tool (LAS), combined with Symbolic Layout Editor and Compactor. This tools are both provided in the CADENCE Edge™ environment. The LAS is able to automatically generate a symbolic layout of a cell using the transistor level schematic as its input. The generated symbolic layout may be further optimized using the Symbolic Layout Editor and Compactor. Of course, to have a more complete control over the cell's layout, one may use from start only the symbolic Layout Editor and Compactor, designing the symbolic layout itself.

VI – Scheduling

Depending on the number of laboratory hours allocated for the project and the student's knowledge level in microelectronics, the theme definition can be biased either to a more automatic or a more manual approach.

In any case, the theme should compel the hardworking student to go through and master all the facilities offered by the design environment for the given approach. Logic simulation to check schematic validity, electric simulation for timing optimization, and post-layout simulations on extracted schematics with parasitics are a must. This should be joined by the use of the various features of the Automatic Place and Route tool or, for the other extreme, those of the LAS and Symbolic Layout Editor and Compactor tool.

Viewing the theme's complexity, it would be a good idea to extend the allocated laboratory hours on two semesters. The most part of the time shall be devoted by the students to circuit and layout conception as well as studying various optimization alternatives. This way, the computer working time will be efficiently used to implement the best choices and to learn the design environment. This means that there is no need to use important computing resources for this project, which is an important issue for a poor hardware dotation lab.

VII – Expected results and comments

The proposed work is rather large and complex. It should finally clearly distinguish the hardworking students from the others.

The advantages of such a work are important and multiples:

- offers the feeling of a real project (although a small one) and could constitute a valuable starting experience for a further work in VLSI ASIC design.
- develop team collaboration relationship and inter-team cooperation on various stages of the design,

mainly in the floor planning phase (layout design).

- as a high performance circuit, this should stimulate the student's commitment in searching optimisations making use creatively of their electronics knowledges as well as learning new ones.
- there is already available a CMOS static logic implementation of the divider, whose performances could be taken as reference by the students in order to self-evaluate their own work. This should make them willing to achieve better results, stimulating their inventivity.
- the division is a very well known operation, that does not need intricate specifications.

Skills in using the design environment should be appreciated in the final notation.

References

[1] ANSI/IEEE, *IEEE Standard for Binary Floating-Point Arithmetic*, ANSI/IEEE Standard 754-1985, 1985

[2] A. Avizienis, "Signed-digit number representation for fast parallel arithmetic", *IRE Transactions on electronic computers*, 10, 1961, pp. 389-400

[3] C. Y. Chow and J. E. Robertson, "Logical design of a redundant binary adder", in *Proc. 4$^{\text{th}}$ symposium on computer arithmetic*, 1978, pp. 109-115

[4] L. Montalvo and A. Guyot, *A hybrid radix-4 divider with operand scaling.* Unpublished internal paper, TIMA Grenoble, 1993

[5] I. Moussa, A. Skaf and A. Guyot, *Design of a GaAs redundant divider*, VLSI'93 Grenoble, 1993

[6] I. Moussa, A. Guyot and P. Rost, *Design and comparison of GaAs and CMOS dividers*, ESSIRC'93, Sevilla, 1993

[7] J. M. Muller, *Arithmetique des ordinateurs*, Paris: Masson 1989

[8] J. E. Robertson, "The correspondence between methods of digital division and multiplier recoding procedures", *IEEE Transactions on computers*, C-19 N$^{\text{o}}$ 8, 1970

[9] N. Weste & K. Eshraghian, *Principles of CMOS VLSI Design*, Reading Massachusetts: Addison-Wesley Publishing Company, 1985

[10] J. William & V. C. Hamacher, "A linear-time divider array", *Canadian Electrical Engineering Journal*, Vol. 6 n$^{\text{o}}$ 4, 1981

[11] Mead & Conway, *Introduction to VLSI systems*, Addison-Wesley Publishing Company, 1980