1

INTRODUCTION TO VLSI DESIGN

1.1 INTRODUCTION

The word digital has made a dramatic impact on our society. More significant is a continuous trend towards digital solutions in all areas – from electronic instrumentation, control, data manipulation, signals processing, telecommunications *etc*., to consumer electronics. Development of such solutions has been possible due to good digital system design and modeling techniques.

1.2 CONVENTIONAL APPROACH TO DIGITAL DESIGN

Digital ICs of SSI and MSI types have become universally standardized and have beenaccepted for use. Whenever a designer has to realize a digital function, he uses a standard set of ICs along with a minimal set of additional discrete circuitry. Consider a simple example of realizing a function as

$$Q_{n+1} = Q_n + (A\ B)$$

Here Q_n, A, and B are Boolean variables, with Q_n being the value of Q at the nth time step. Here $A\ B$ signifies the logical AND of A and B; the '+' symbol signifies the logical OR of the logic variables on either side. A circuit to realize the function is shown in Figure 1.1. The circuit can be realized in terms of two ICs – an A-O-I gate and a flip-flop. It can be directly wired up, tested, and used.

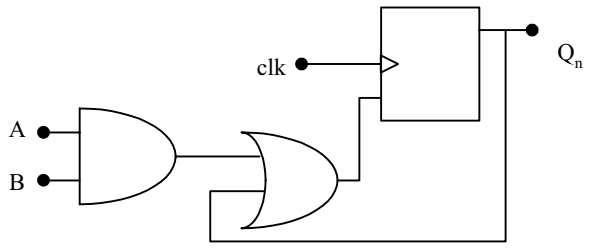

Figure 1.1 A simple digital circuit.

Design Through Verilog HDL. T. R. Padmanabhan and B. Bala Tripura Sundari

ISBN: 0-471-44148-1

With comparatively larger circuits, the task mostly reduces to one of identifying the set of ICs necessary for the job and interconnecting; rarely does one have to resort to a microlevel design [Wakerly]. The accepted approach to digital design here is a mix of the top-down and bottom-up approaches as follows [Hill & Peterson]:

- Decide the requirements at the system level and translate them to circuit requirements.
- Identify the major functional blocks required like timer, DMA unit, register-file *etc*., say as in the design of a processor.
- Whenever a function can be realized using a standard IC, use the same –for example programmable counter, mux, demux, *etc*.
- Whenever the above is not possible, form the circuit to carry out the block functions using standard SSI – for example gates, flip-flops, *etc*.
- Use additional components like transistor, diode, resistor, capacitor, *etc*., wherever essential.

Once the above steps are gone through, a paper design is ready. Starting with the paper design, one has to do a circuit layout. The physical location of all the components is tentatively decided; they are interconnected and the 'circuit-on-paper' is made ready. Once a paper design is done, a layout is carried out and a net-list prepared. Based on this, the PCB is fabricated, and populated and all the populated cards tested and debugged. The procedure is shown as a process flowchart in Figure 1.2.

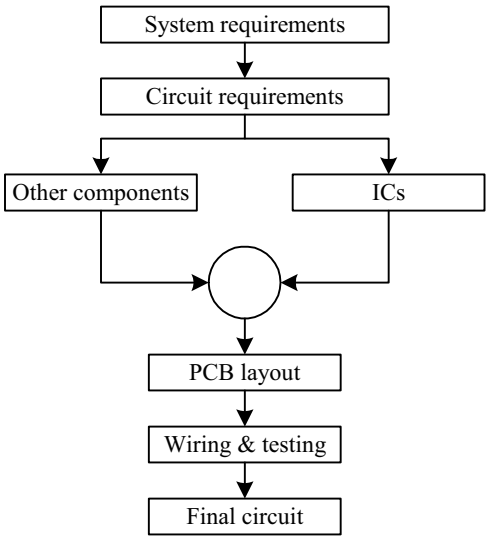

Figure 1.2 Sequence of steps in conventional electronic circuit design.

At the debugging stage one may encounter three types of problems:

- *Functional mismatch*: The realized and expected functions are different. One may have to go through the relevant functional block carefully and locate any error logically. Finally the necessary correction has to be carried out in hardware.
- *Timing mismatch*: The problem can manifest in different forms. One possibility is due to the signal going through different propagation delays in two paths and arriving at a point with a timing mismatch. This can cause faulty operation. Another possibility is a race condition in a circuit involving asynchronous feedback. This kind of problem may call for elaborate debugging. The preferred practice is to do debugging at smaller module stages and ensuring that feedback through larger loops is avoided: It becomes essential to check for the existence of long asynchronous loops.
- *Overload*: Some signals may be overloaded to such an extent that the signal transition may be unduly delayed or even suppressed. The problem manifests as reflections and erratic behavior in some cases (The signal has to be suitably buffered here.). In fact, overload on a signal can lead to timing mismatches.

The above have to be carried out after completion of the prototype PCB manufacturing; it involves cost, time, and also a redesigning process to develop a bugfree design.

1.3 VLSI DESIGN

The complexity of VLSIs being designed and used today makes the manual approach to design impractical. Design automation is the order of the day. With the rapid technological developments in the last two decades, the status of VLSI technology is characterized by the following [Wai-kai, Gopalan]:

- A steady increase in the size and hence the functionality of the ICs.
- A steady reduction in feature size and hence increase in the speed of operation as well as gate or transistor density.
- A steady improvement in the predictability of circuit behavior.
- A steady increase in the variety and size of software tools for VLSI design.

The above developments have resulted in a proliferation of approaches to VLSI design. We briefly describe the procedure of automated design flow [Rabaey, Smith MJ]. The aim is more to bring out the role of a Hardware Description Language (HDL) in the design process. An abstraction based model is the basis of the automated design.

1.3.1 Abstraction Model

The model divides the whole design cycle into various domains (see Figure 1.3). With such an abstraction through a division process the design is carried out in different layers. The designer at one layer can function without bothering about the layers above or below. The thick horizontal lines separating the layers in the figure signify the compartmentalization. As an example, let us consider design at the gate level. The circuit to be designed would be described in terms of truth tables and state tables. With these as available inputs, he has to express them as Boolean logic equations and realize them in terms of gates and flip-flops. In turn, these form the inputs to the layer immediately below. Compartmentalization of the approach to design in the manner described here is the essence of abstraction; it is the basis for development and use of CAD tools in VLSI design at various levels.

The design methods at different levels use the respective aids such as Boolean equations, truth tables, state transition table, *etc*. But the aids play only a small role in the process. To complete a design, one may have to switch from one tool to another, raising the issues of tool compatibility and learning new environments.

1.4 ASIC DESIGN FLOW

As with any other technical activity, development of an ASIC starts with an idea and takes tangible shape through the stages of development as shown in Figure 1.4 and shown in detail in Figure 1.5. The first step in the process is to expand the idea in terms of behavior of the target circuit. Through stages of programming, the same is fully developed into a design description – in terms of well defined standard constructs and conventions.

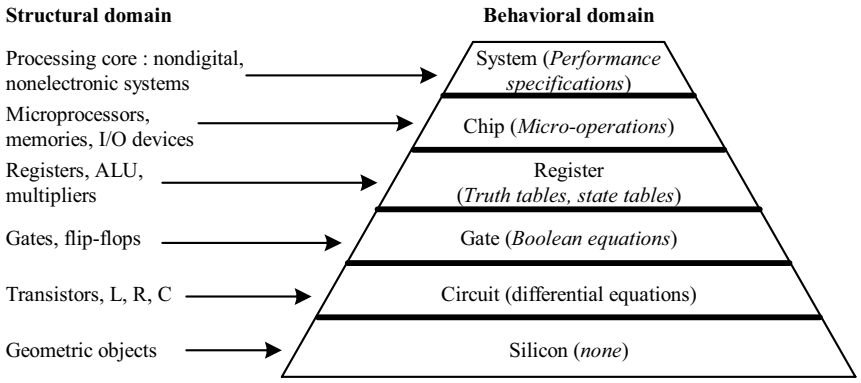

Figure 1.3 Design domain and levels of abstraction.

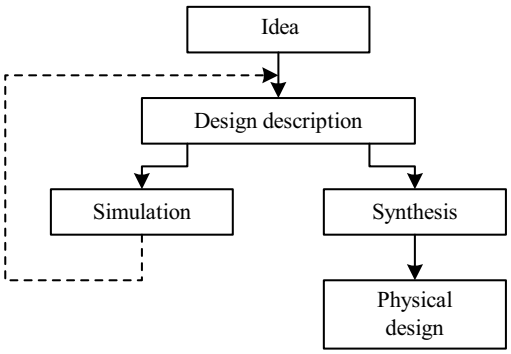

Figure 1.4 Major activities in ASIC design.

The design is tested through a simulation process; it is to check, verify, and ensure that what is wanted is what is described. Simulation is carried out through dedicated tools. With every simulation run, the simulation results are studied to identify errors in the design description. The errors are corrected and another simulation run carried out. Simulation and changes to design description together form a cyclic iterative process, repeated until an error-free design is evolved.

Design description is an activity independent of the target technology or manufacturer. It results in a description of the digital circuit. To translate it into a tangible circuit, one goes through the physical design process. The same constitutes a set of activities closely linked to the manufacturer and the target technology

1.4.1 Design Description

The design is carried out in stages. The process of transforming the idea into a detailed circuit description in terms of the elementary circuit components constitutes design description. The final circuit of such an IC can have up to a billion such components; it is arrived at in a step-by-step manner.

The first step in evolving the design description is to describe the circuit in terms of its behavior. The description looks like a program in a high level language like C. Once the behavioral level design description is ready, it is tested extensively with the help of a simulation tool; it checks and confirms that all the expected functions are carried out satisfactorily. If necessary, this behavioral level routine is edited, modified, and rerun – all done manually. Finally, one has a design for the expected system – described at the behavioral level. The behavioral design forms the input to the synthesis tools, for circuit synthesis. The behavioral constructs not supported by the synthesis tools are replaced by data flow and gate level constructs. To surmise, the designer has to develop synthesizable codes for his design.

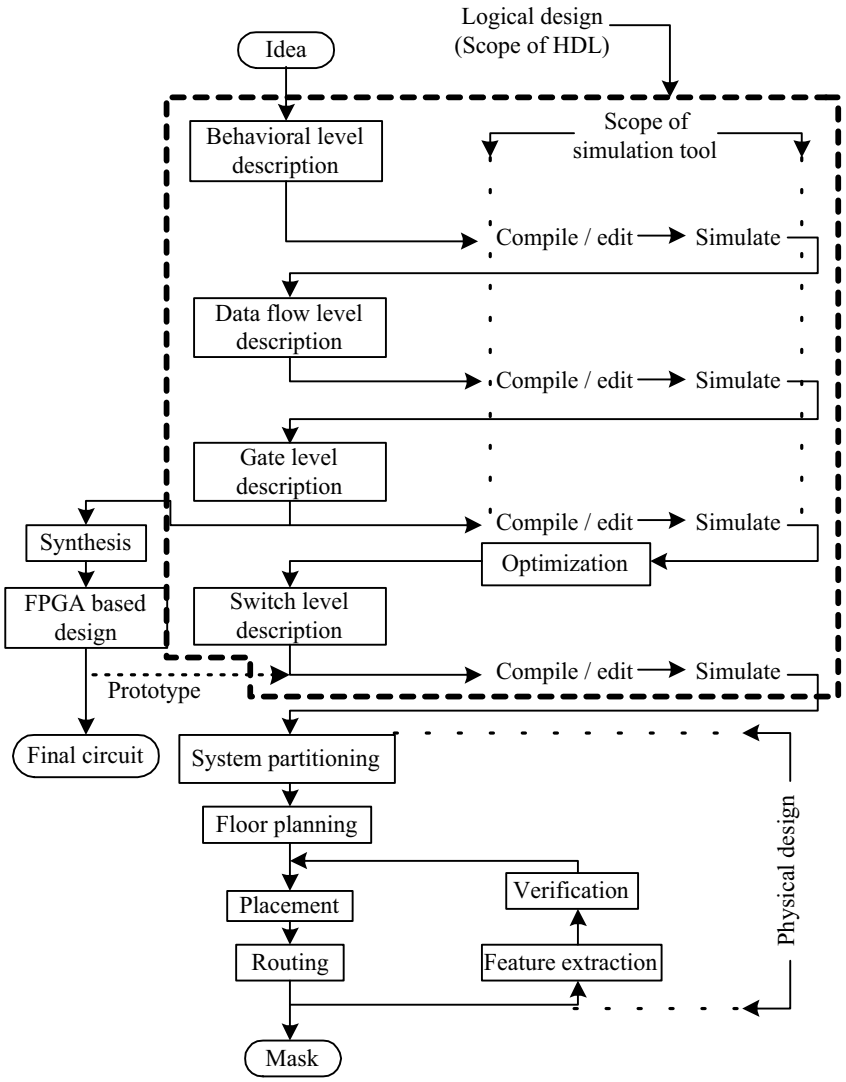

Figure 1.5ASIC design and development flow.

The design at the behavioral level is to be elaborated in terms of known and acknowledged functional blocks. It forms the next detailed level of design description. Once again the design is to be tested through simulation and iteratively corrected for errors. The elaboration can be continued one or two steps further. It leads to a detailed design description in terms of logic gates and transistor switches.

1.4.2 Optimization

The circuit at the gate level – in terms of the gates and flip-flops – can be redundant in nature. The same can be minimized with the help of minimization tools. The step is not shown separately in the figure. The minimized logical design is converted to a circuit in terms of the switch level cells from standard libraries provided by the foundries. The cell based design generated by the tool is the last step in the logical design process; it forms the input to the first level of physical design [Micheli].

1.4.3 Simulation

The design descriptions are tested for their functionality at every level – behavioral, data flow, and gate. One has to check here whether all the functions are carried out as expected and rectify them. All such activities are carried out by the simulation tool. The tool also has an editor to carry out any corrections to the source code. Simulation involves testing the design for all its functions, functional sequences, timing constraints, and specifications. Normally testing and simulation at all the levels – behavioral to switch level – are carried out by a single tool; the same is identified as "scope of simulation tool" in Figure 1.5.

1.4.4 Synthesis

With the availability of design at the gate (switch) level, the logical design is complete. The corresponding circuit hardware realization is carried out by a synthesis tool. Two common approaches are as follows:

- The circuit is realized through an FPGA [Oldfield]. The gate level design description is the starting point for the synthesis here. The FPGA vendors provide an interface to the synthesis tool. Through the interface the gate level design is realized as a final circuit. With many synthesis tools, one can directly use the design description at the data flow level itself to realize the final circuit through an FPGA. The FPGA route is attractive for limited volume production or a fast development cycle.
- The circuit is realized as an ASIC. A typical ASIC vendor will have his own library of basic components like elementary gates and flip-flops. Eventually the circuit is to be realized by selecting such components and interconnecting them conforming to the required design. This constitutes the physical design. Being an elaborate and costly process, a physical design may call for an intermediate functional verification through the FPGA route. The circuit realized through the FPGA is tested as a prototype. It provides another opportunity for testing the design closer to the final circuit.

1.4.5 Physical Design

A fully tested and error-free design at the switch level can be the starting point for a physical design [Baker & Boyce, Wolf]. It is to be realized as the final circuit using (typically) a million components in the foundry's library. The step-by-step activities in the process are described briefly as follows:

- *System partitioning*: The design is partitioned into convenient compartments or functional blocks. Often it would have been done at an earlier stage itself and the software design prepared in terms of such blocks. Interconnection of the blocks is part of the partition process.
- *Floor planning*: The positions of the partitioned blocks are planned and the blocks are arranged accordingly. The procedure is analogous to the planning and arrangement of domestic furniture in a residence. Blocks with I/O pins are kept close to the periphery; those which interact frequently or through a large number of interconnections are kept close together, and so on. Partitioning and floor planning may have to be carried out and refined iteratively to yield best results.
- *Placement*: The selected components from the ASIC library are placed in position on the "Silicon floor." It is done with each of the blocks above.
- *Routing*: The components placed as described above are to be interconnected to the rest of the block: It is done with each of the blocks by suitably routing the interconnects. Once the routing is complete, the physical design cam is taken as complete. The final mask for the design can be made at this stage and the ASIC manufactured in the foundry.

1.4.6 Post Layout Simulation

Once the placement and routing are completed, the performance specifications like silicon area, power consumed, path delays, *etc*., can be computed. Equivalent circuit can be extracted at the component level and performance analysis carried out. This constitutes the final stage called "verification." One may have to go through the placement and routing activity once again to improve performance.

1.4.7 Critical Subsystems

The design may have critical subsystems. Their performance may be crucial to the overall performance; in other words, to improve the system performance substantially, one may have to design such subsystems afresh. The design here may imply redefinition of the basic feature size of the component, component design, placement of components, or routing done separately and specifically for the subsystem. A set of masks used in the foundry may have to be done afresh for the purpose.

1.5 ROLE OF HDL

An HDL provides the framework for the complete logical design of the ASIC. All the activities coming under the purview of an HDL are shown enclosed in bold dotted lines in Figure 1.4. Verilog and VHDL are the two most commonly used HDLs today. Both have constructs with which the design can be fully described at all the levels. There are additional constructs available to facilitate setting up of the test bench, spelling out test vectors for them and "observing" the outputs from the designed unit.

IEEE has brought out Standards for the HDLs, and the software tools conform to them. Verilog as an HDL was introduced by Cadence Design Systems; they placed it into the public domain in 1990. It was established as a formal IEEE Standard in 1995. The revised version has been brought out in 2001. However, most of the simulation tools available today conform only to the 1995 version of the standard.

Verilog HDL used by a substantial number of the VLSI designers today is the topic of discussion of the book.