Introduction to HDL-based design methodology

Prof. Chien-Nan Liu
TEL: 03-4227151 ext: 34534
Email: jimmy@ee.ncu.edu.tw

Milestones for IC Industry

- **1947**: Bardeen, Brattain & Shockley invented the transistor, foundation of the IC industry
- **1952**: SONY introduced the first transistor-based radio
- **1958**: Kilby invented integrated circuits (ICs)
- **1965**: Moore’s law
- **1968**: Noyce and Moore founded Intel
- **1971**: Intel announced 4-bit 4004 microprocessors (2300 transistors)
- **1976/81**: Apple / IBM PC
- **1985**: Intel began focusing on microprocessor products
- **1987**: TSMC was founded (implication: fabless design)
Milestones for IC Industry

- **1991**: ARM introduced its first embeddable RISC IP core (implication: chipless design)
- **1996**: Samsung introduced prototype 1G DRAM
- **1998**: IBM Austin Res. Lab announced 1GHz experimental microprocessor; Ericsson, etc. founded Bluetooth Special Interest Group -- designs go Systems-on-chip (SOC)!
- Today, Intel-PIII has > 10M transistors (up to 1.13 GHz; 0.18 um)
- Today we produce > 10M transistors/person (1 billion/person by 2008)
- Semiconductor/IC: #1 key field for advancing into Y2K (Business Week, Jan. 1995)

The First Transistor

![Image of the first transistor](image)
Pioneers of the Electronic Age

Silicon Wafers
Moore’s Law

- Logic capacity doubles per IC per year at regular intervals (1965)
- Logic capacity doubles per IC every 18 months (1975)
The Dies of Intel CPUs

Pentium Pro

Semiconductor Technology Roadmap

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology node (nm)</td>
<td>250</td>
<td>180</td>
<td>130</td>
<td>100</td>
<td>70</td>
<td>50</td>
<td>35</td>
</tr>
<tr>
<td>On-chip local clock (GHz)</td>
<td>0.75</td>
<td>1.25</td>
<td>2.1</td>
<td>3.5</td>
<td>6.0</td>
<td>10</td>
<td>16.9</td>
</tr>
<tr>
<td>Microprocessor chip size (mm²)</td>
<td>300</td>
<td>340</td>
<td>430</td>
<td>520</td>
<td>620</td>
<td>750</td>
<td>901</td>
</tr>
<tr>
<td>Microprocessor transistor/chip</td>
<td>11M</td>
<td>21M</td>
<td>76M</td>
<td>200M</td>
<td>520M</td>
<td>1.40B</td>
<td>3.62B</td>
</tr>
<tr>
<td>Microprocessor cost/transistor (x10⁸ USD)</td>
<td>3000</td>
<td>1735</td>
<td>580</td>
<td>255</td>
<td>110</td>
<td>49</td>
<td>22</td>
</tr>
<tr>
<td>DRAM bits per chip</td>
<td>256M</td>
<td>1G</td>
<td>4G</td>
<td>16G</td>
<td>64G</td>
<td>256G</td>
<td>1T</td>
</tr>
<tr>
<td>Wiring level</td>
<td>6</td>
<td>6-7</td>
<td>7</td>
<td>7-8</td>
<td>8-9</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>Supply voltage (V)</td>
<td>1.8-2.5</td>
<td>1.5-1.8</td>
<td>1.2-1.5</td>
<td>0.9-1.2</td>
<td>0.6-0.9</td>
<td>0.5-0.6</td>
<td>0.37-0.42</td>
</tr>
<tr>
<td>Power (W)</td>
<td>70</td>
<td>90</td>
<td>130</td>
<td>160</td>
<td>170</td>
<td>175</td>
<td>183</td>
</tr>
</tbody>
</table>

Source: SIA99
Trends of VLSI Design

- Design Complexity
- Reuse
- Synthesis
- Place & route
- Design Productivity

- Transistors
- Gates
- RTL
- SoC


VLSI Design Considerations

- Several conflicting considerations
  - **Design complexity**: large number of devices/transistors
  - **Cost**: die area, packing, testing, etc.
  - **Performance**: optimization requirements for high performance
  - **Time-to-market**: about 15% gain for early birds
  - Others: power, noise, testability, reliability, manufacturability, etc.
- **Keys**: hierarchical design, abstraction, **CAD**
Design Representation

Three Different Domains

- **Behavior**
  - Functionality of components

```verilog
code
module add4 (s4, c4, ci, a, b);
  input [3:0] a, b;
  input ci;
  output [3:0] s;
  output c4;
  wire [2:0] co;
  add f0 (co[0], s[0], a[0], b[0], ci);
  add f1 (co[1], s[1], a[1], b[1], co[0]);
  add f2 (co[2], s[2], a[2], b[2], co[1]);
  add f3 (c4, s[3], a[3], b[3], co[2]);
endmodule
code
```
Three Different Domains

- **Structural**
  - Connectivity between components

![Diagram showing structural connectivity between components](image)

- **Physical**
  - A layout description

![Diagram showing physical layout](image)
Hierarchy of Description

- System Level
- Algorithm Level
- Register Transfer Level
- Logic Gate Level
- Switch Level

Design on Different levels

(a) silicon
(b) circuit
(c) gate F-F
(d) Register
(e) Chip

Parallel bus
Data available
 Chip

Serial in
Serial out
Clock
Design Styles

Full Custom Design Style

- Handcraft functional & physical designs
- Design efforts and cost are high
- Expect high quality and high volume
Cell-Based Design

- Cells are characterized and stored in library
- Need update when technology advances
- Compatible to custom designs
- Easier to develop CAD tools for design and optimization

Standard Cell Design Style

- Cell-based: standard cells, macro cells
  - Standard cells: FFs, AND, OR..
  - Macro cells: Memory, PLA, ALU…
- Manufacturing process is not simplified
- Design process is simplified
- Need tremendous characterization effort
- Parameterized area and delay over ranges of temperatures and operating voltage
**Gate Array Design Style**

- Arrays of sites are pre-manufactured
- Metal and contact layers are used to program the chip
- Fewer manufacturing steps correlate to lower fabrication time and cost

**CPLD/FPGA Design Style**

- Illustrated by a symmetric array-based FPGA
Field Programmable Gate Array

- Arrays of programmable modules with the capability of implementing a generic logic function
- Wires can be connected by programming antifuses
- Reduce development and production time
- Low cost prototyping

SPLD Design Style

SPLD (PLA) implementation
SSI Design Style

* SSI = Small Scaled Integrated circuit

Comparison of Design Styles

<table>
<thead>
<tr>
<th></th>
<th>Full custom</th>
<th>Standard cell</th>
<th>Gate array</th>
<th>FPGA</th>
<th>SPLD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Density</strong></td>
<td>Very high</td>
<td>High</td>
<td>High</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td>Very high</td>
<td>High</td>
<td>High</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Flexibility</strong></td>
<td>Very high</td>
<td>High</td>
<td>Medium</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Design time</strong></td>
<td>Very long</td>
<td>Short</td>
<td>Short</td>
<td>Very short</td>
<td>Very short</td>
</tr>
<tr>
<td><strong>Manufacturing time</strong></td>
<td>Medium</td>
<td>Medium</td>
<td>Short</td>
<td>Very short</td>
<td>Very short</td>
</tr>
<tr>
<td><strong>Unit cost – small quantity</strong></td>
<td>Very high</td>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>Very Low</td>
</tr>
<tr>
<td><strong>Unit cost – large quantity</strong></td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>Very High</td>
</tr>
</tbody>
</table>
Design-Style Trade-offs

- Performance/cost tradeoff
  - Customize design to achieve desired performance

Traditional VLSI Design Flow

<table>
<thead>
<tr>
<th>System Specification</th>
<th>Functional / Architecture Design</th>
<th>Logic Synthesis</th>
<th>Circuit Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>functional simulation</td>
<td>logic simulation</td>
<td>circuit analysis</td>
<td>extract &amp; verification</td>
</tr>
</tbody>
</table>

module fz(a,b,c,d,e,Z);
input a,b,c,d;
output Z;
assign Z = ~a|(b&C)|(d&e);
endmodule

behavior representation

HDL description

gate-level representation

switch-level representation
Traditional VLSI Design Flow

Specification to Circuits

- Specification: Specify what the system does
  - The functional appearance of a system to its users
- Design: Determine
  - The structure of a system
  - Different methods to achieve a function
  - Logical structures that perform the architecture
- Realization: Materialize the physical structures
  - Technology decision (CMOS, ECL)
  - Design style decision (Full Custom, Semi-Custom ...)
Design Specification

- Informal description by English
  - These full adders perform the addition of two 2-bit binary numbers. The sum output is provided for each bit and the resultant carry is obtained from the second bit. Design for medium-to-high speed, multiple bits, parallel-add/serial carry applications.

Design Specification

- Formal description using HDL
  - Verify the specification through simulation or verification
  - Easy to change
  - Enable automatic synthesis
Hardware Description Language

- Can describe a design at some levels of abstraction
  - Behavioral, RTL, Gate-level
- Can be used to document the complete system design tasks
  - testing, simulation ... related activities
- Comprehensive and easy to learn

HDL-Based Design Flow

- Describe desired functionality and timing using HDL such as VHDL, Verilog
- Use high-level synthesis tool to obtain structural level design
- Using placement and routing tools to obtain physical level design
- Very popular design approach for standard cell and gate array design
Logic Synthesis

- Logic synthesis programs transform HDL descriptions into logic gate networks in a particular library
  - Involve translation and optimization
- Optimization criteria: area, performance, power, and testability
- Logic optimization: technology independent
  - Optimize Boolean expressions but still keep their equivalence
- Technology mapping: technology dependent
  - Map Boolean expressions into a particular cell library
Gate-Level Netlist

Drivers for Synthesis: Competition

- Time to market
  - Delayed market entry means lost revenue
- Reducing costs
  - 2-4X more complex, 2-4X higher quality for 50% of the cost as the last system
- Higher quality: shipping fewer defective products
- Managing the complexity
- Technology independent
Synthesis v.s. Schematic Capture

- Traditional way of design
  - Architecture design
  - Functional design with logic
  - Design capture using schematic capture tools

- HDL-based design
  - Architecture design
  - Functional capture using HDL
  - Implementation using synthesis tools

How HDL-Based Design Helps

- Shortens the design verification loop
- Allows architectural tradeoffs with short turnaround
- Reduces time for design capture
- Encourages focus on functionality
- Ensures design documentation consistency
- Facilitates design modification
Physical Design

- Physical design converts a circuit description into a geometric description (GDSII file)
- The description is used to manufacture a chip

Standard Cells
Physical Design Flow

- Physical design cycle:
  1. Logic partitioning
  2. Floorplanning, placement, and pin assignment
  3. Routing (global and detailed)
  4. Compaction
  5. RLC extraction & verification

PowerPC Chip Floorplan

Figure 9-38
A PowerPC microprocessor chip with various functional areas indicated. This microprocessor uses the Harvard instruction and computing (RISC) architecture.
(Photograph courtesy of IBM Corp.)
IC Fabrication

Wafer and Dices

A dice fabricated with other die on the silicon wafer

Wafer diameter is typically 5 to 8 inches.

Sawing a Wafer into Chips

Figure 9-33: After testing and sawing, the individual chips are picked up by a robotic arm and placed in the package for die bonding. (Photograph courtesy of Intel Corp.)

IC and Die

(a) 0.1 inch  (b) silicon die
Chip Packing

Pentium-MMX with PGA Packging
FPGA Design Flow

- Advantages: Fast and reusable prototyping
  - Can be reprogrammed and reused
  - Implementation time is very short
- Disadvantages: Expensive and high volume

FPGA Structure (Altera)

Fig. 6-28 Altera® MAX 7000™ Structure (Reprinted with Permission of Altera Corporation, © Altera Corp., 1991)
Store the Programming Info.

- SRAM technology is used
  - M = 1-bit SRAM
  - Loaded from the PROM after power on
- Store control values
  - Control pass transistor
  - Control multiplexer
- Store logic functions
  - Store the value of each minterm in the truth table
Xilinx FPGA Routing

- Fast direct interconnect
  - Adjacent CLBs
- General purpose interconnect
  - CLB – CLB or CLB – IOB
  - Through switch matrix
- Long lines
  - Across whole chip
  - High fan-out, low skew
  - Suitable for global signals (CLK) and buses
  - 2 tri-states per CLB for busses

Xilinx Switch Matrix

- Six pass transistors to control each switch node
- The two lines at point 1 are joined together
- At point 2, two distinct signal paths pass through one switch node

Fig. 6-31 Example of Xilinx® Switch Matrix (Adapted with Permission of Xilinx®, Inc.)
Configurable Logic Block (CLB)

- Combinational logic via lookup table
  - Any function(s) of available inputs
  - Output registered and/or combinational

Simplified CLB Structure
Internal Functions of a CLB

- Two 4-input tables implement two distinct functions (F’ and G’)
- F’ and G’ with another control (H1) feed into a third lookup table (H’)
- Two arbitrary functions of up to four variables and selected functions of up to nine variables can be implemented
- Properly setting the two MUXes can assign any pair of F’, G’, and H’ to the two combinational outputs (X and Y)

Internal Functions of a CLB

- Two D flip-flops directly drive outputs XQ and YQ
- Each of the D inputs can be selected from F’, G’, H’ and input DIN
- Two XORs select each flip-flop individually to be positive or negative edge triggered
- Two SR controls select the signal S/R to be an asynchronous Set or Reset for the flip-flops
- Two multiplexers allow the input EC to optionally act as a clock ENABLE signal for each flip-flop
I/O Block (IOB)

- Periphery of identical I/O blocks
  - Input, output, or bidirectional
  - Registered, latched, or combinational
  - Three-state output
  - Programmable output slew rate

Input/Output Mode of an IOB

- **Input**
  - 3-state control places the output buffer into high impedance
  - Direct in and/or registered in

- **Output**
  - 3-state driver should be enabled by TS signal
  - Direct output or registered output
Design with FPGA

- Using HDL, schematic editor, SM chart or FSM diagram to capture the design
- Simulate and debug HDL codes
- Work out detail logic manually or by using synthesis tools
- Simulate and debug again at logic level
- Feed the logic into CLBs and IOBs
  - Completed by a CAD tool

Design with FPGA (cont’d)

- Partitioning (Technology mapping)
  - Chip-level
  - CLB-level
- Place and route
- Generate bit pattern for programming the FPGA
- Download the bit pattern into the internal configurable memory cells and test the operations
Acknowledgement

- Some of the presentation materials were modified from the previous works of
  - Prof. Jing-Yang Jou (周景揚, Dept. of EE, NCTU)
  - Prof. Yao-Wen Chang (張耀文, Dept. of EE, NTU)
  - Ms. I-Ling Chen (陳怡伶, ITRI/STC)
- Their supports are greatly appreciated