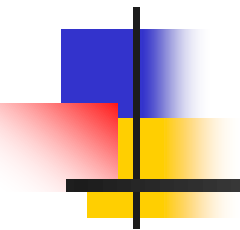
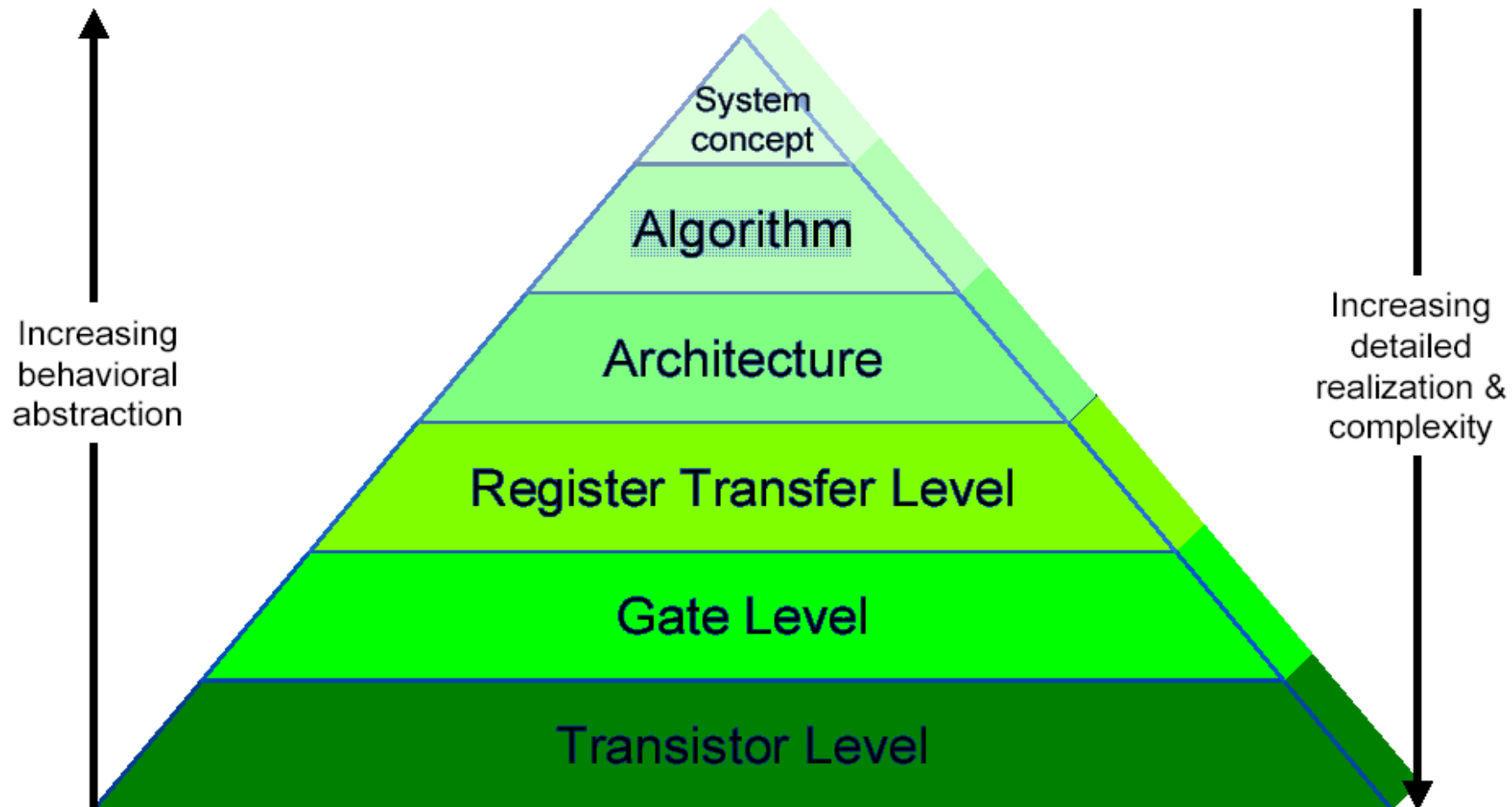


數位IC設計

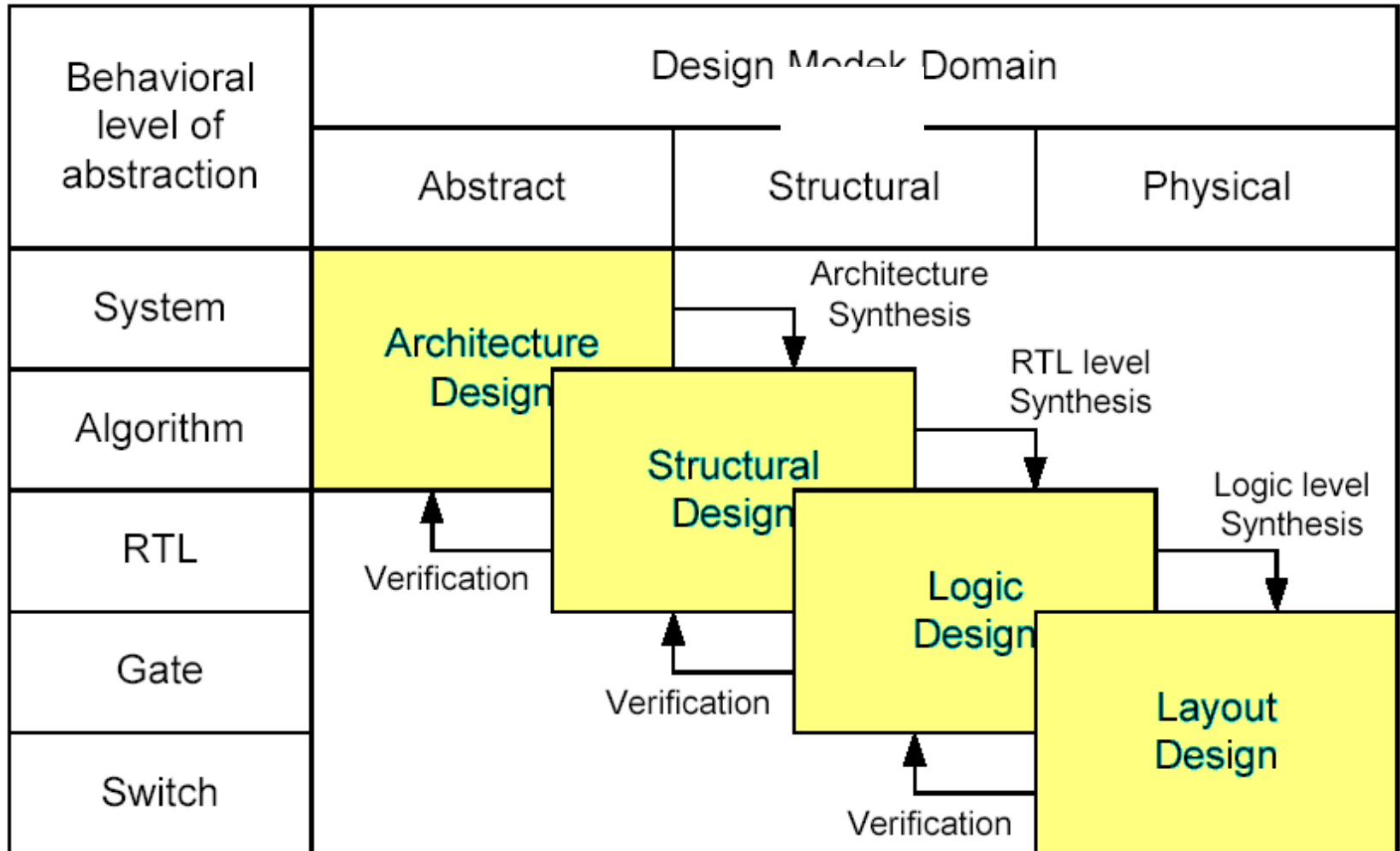


Semi Custom Design Flow

Top-Down Design Methodology



Design Domain



Register Transfer Level -- RTL

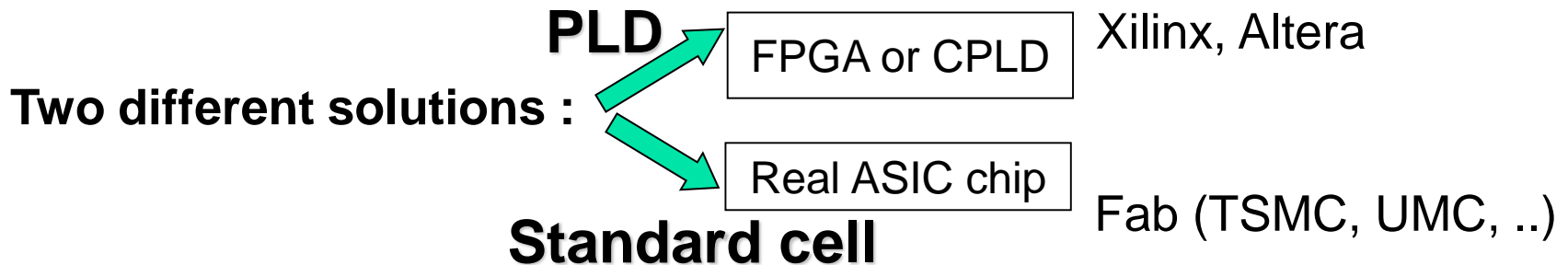
Cell-Based Design Flow (1/2)

Semi Custom Design

- a. Product specification
- b. Modeling with HDL
- c. Synthesis (by using suitable standard cell)
- d. Simulation and verification
- e. Physical placement and layout
- f. Tape-out (real chip) -- implemented by suitable Fab companies
- g. Testing -- implemented by suitable tools and mechanisms

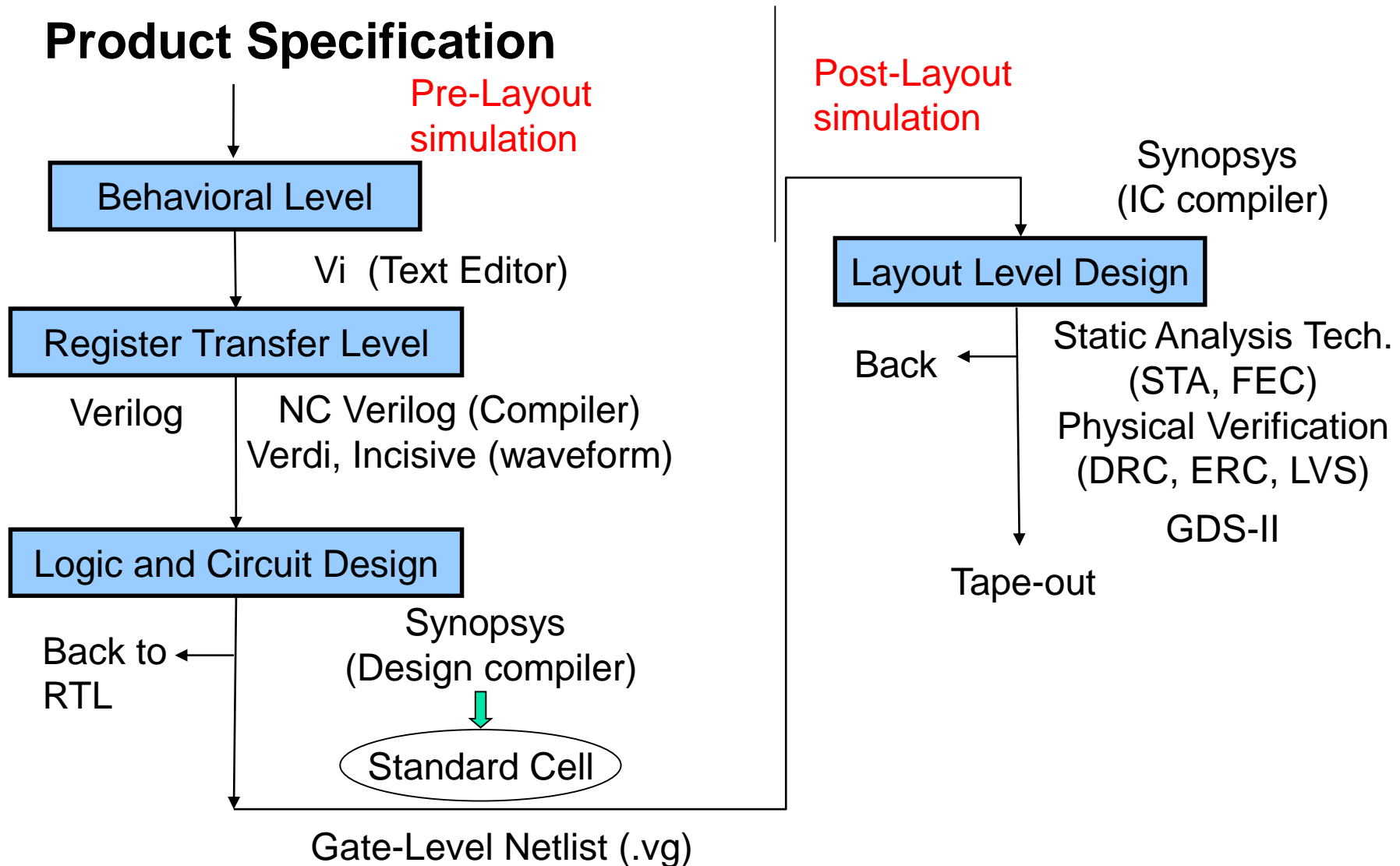
-- implemented with suitable tools

more flexible, shorter design cycle, suitable for smaller production

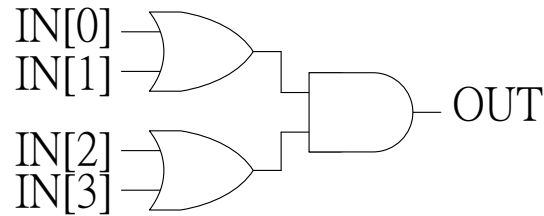


less flexible, long design cycle, larger-scale production to reduce price

Cell-Based Design Flow (2/2)



Hardware Description Language (1/3)

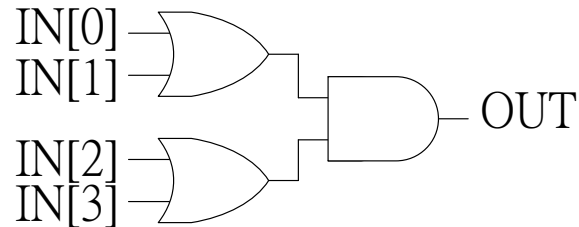


Structural description

1. module OR_AND_STRUCTURAL(IN,OUT);
2. input [3:0] IN;
3. output OUT;
4. wire [1:0] TEMP;
5. or u1(TEMP[0], IN[0], IN[1]);
6. or u2(TEMP[1], IN[2], IN[3]);
7. and (OUT, TEMP[0], TEMP[1]);
8. endmodule

Register Transfer Level -- RTL

Hardware Description Language (2/3)



- Behavioral description

```
1. module OR_AND_BEHAVIORAL(IN, OUT);  
  
2.   input [3:0]  IN;  
3.   output      OUT;  
4.   reg         OUT;  
  
5.   always @(IN)  
6.   begin  
7.     OUT = (IN[0] | IN[1]) & (IN[2] | IN[3]);  
8.   end  
9. endmodule
```

Register Transfer Level -- RTL

Hardware Description Language (3/3)

■ 3 to 8 decoder

E	In[2]	In[1]	In[0]	Out
0	X	X	X	00000000
1	0	0	0	00000001
1	0	0	1	00000010
1	0	1	0	00000100
1	0	1	1	00001000
1	1	0	0	00010000
1	1	0	1	00100000
1	1	1	0	01000000
1	1	1	1	10000000

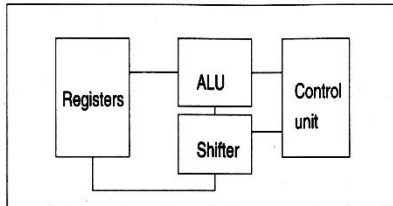
Behavioral description

```
1. module Decoder_Behavioral (E, In, Out);
2.   Input    E;
3.   input    [2:0]    In;
4.   output   [7:0]    Out;
5.   reg      [7:0]    Out;
```

```
always @(E or In)
begin
  if(!E)
    Out = 8'h00;
  else
    begin
      case(In)
        3'b000: Out = 8'h01;
        3'b001: Out = 8'h02;
        3'b010: Out = 8'h04;
        3'b011: Out = 8'h08;
        3'b100: Out = 8'h10;
        3'b101: Out = 8'h20;
        3'b110: Out = 8'h40;
        default: Out = 8'h80;
      endcase
    end
  end
end
endmodule
```

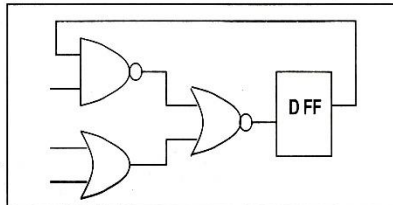

Design Flow Using ModelSim

Functional design

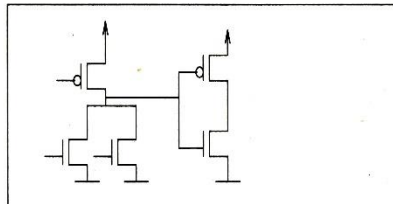


HDL

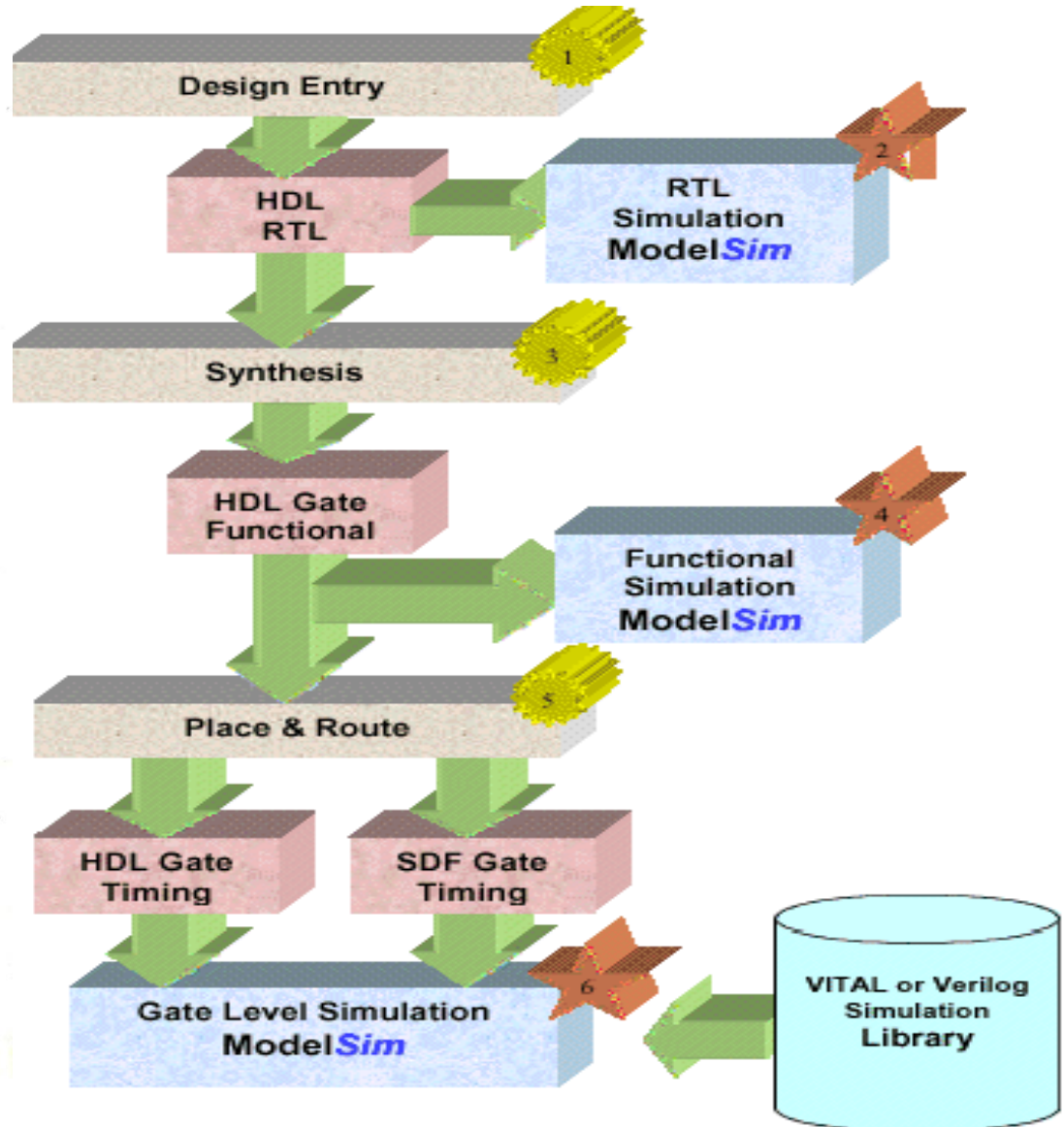
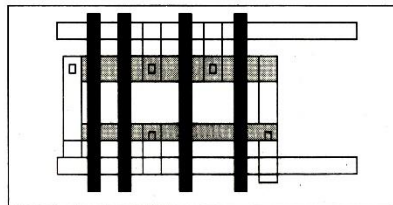
Logic design



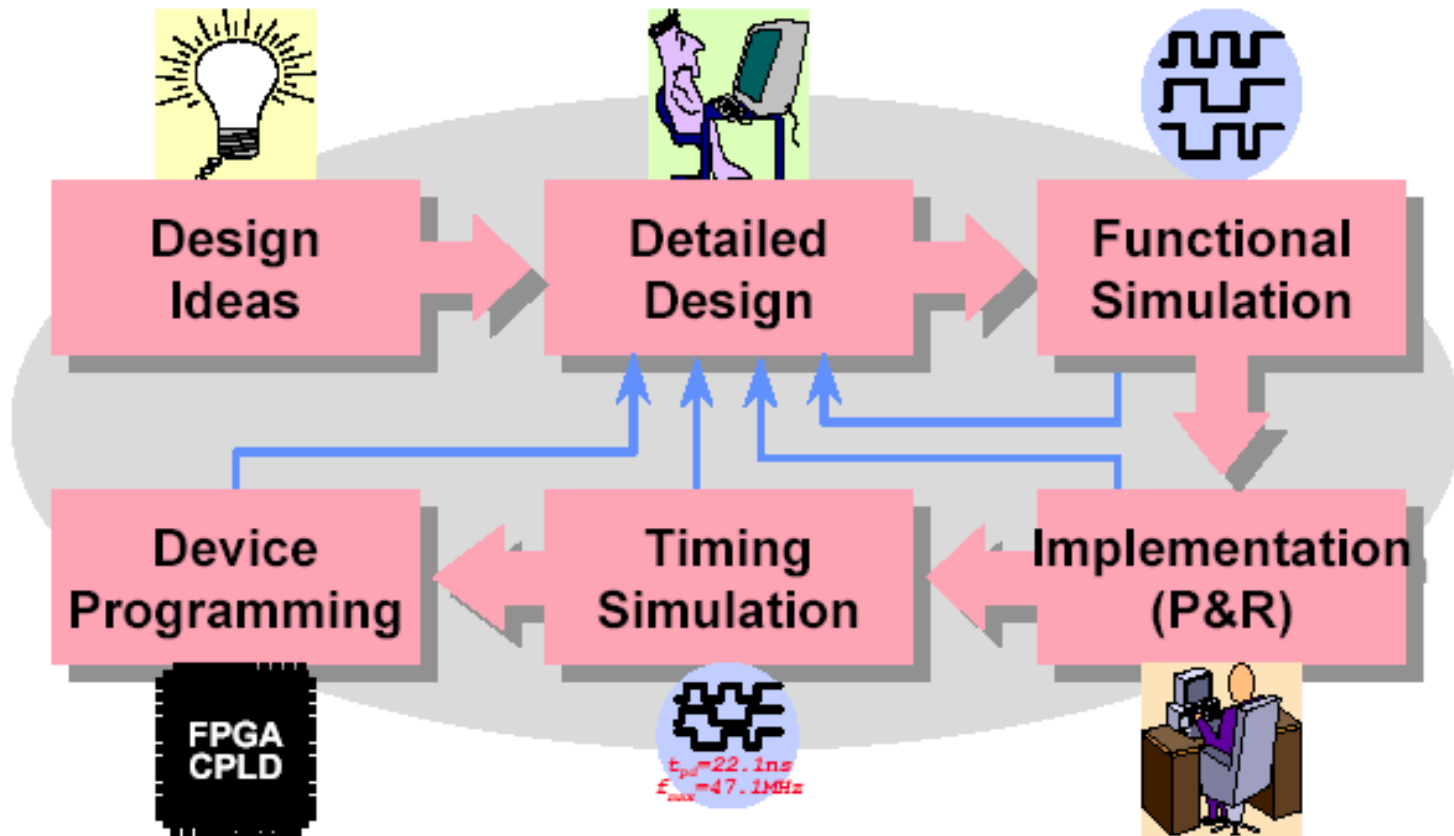
Circuit design



Physical design



FPGA/CPLD Design Flow



Note: What is the difference between standard-cell design and FPGA/CPLD design ? 何者相同?何者不同?
Flow, tool, cost, time, ...



Design Ideas

- Main design consideration
 - Design feasibility?
 - Design spec.?
 - Cost?
 - FPGA or standard-cell ASIC?
 - FPGA vendor?
 - Device family?
 - Development time?



Programmable Logic Device

- PLD (Programmable Logic Device)依其架構及密度可分成三類:
 - SPLD (Simple Programmable Logic Device)
 - CPLD (Complex Programmable Logic Device)
 - FPGA (Field Programmable Gate Array)



SPLD

- SPLD (Simple Programmable Logic Device):
 - 邏輯閘約在數百閘左右
 - IC腳位在28pin以內
 - Bipolar process，只能作單次燒錄，資料無法抹除
 - GAL process，多次燒錄，可縮短設計時程



CPLD

- CPLD (Complex Programmable Logic Device)
 - 邏輯閘在800~5000之間
 - IC腳位高於28pin
 - 44pin以上IC的封裝以PLCC為主
 - CMOS設計技術
 - 多次燒錄抹除



FPGA

- **FPGA:(Field Programmable Gate Array)**
 - 使用和CPLD不同的架構設計方式
 - 以暫存器居多，其密度在5K以上
 - 腳位數多
 - Routing複雜，非固定式，延遲時間較長
- **FPGA的架構**
 - SRAM Base -可重覆燒錄但需外部電源維持資料
 - Anti-fuse -只有一次燒錄，提供較佳保密性



Configurable VLSI

- Advantages

- Short time-to-market
- Low tooling costs
- Low penalty on design changes
- Low testing cost
- Product advantage (new process, .35, .25, .18, .13)

- Disadvantages

- capacity
- cost
- speed



Altera Devices

- Products

- MAX3000/5000/7000/9000 devices family
- FELX6K/8K/10K devices family
- APEX20K devices family
- ACEX1K

- Workbench

- MaxplusII & QuartusII
- FPGA express, Leonard Spectrum



Xilinx Devices

- Products

- XC9500/4000, Coolrunner(XPLA3) ,Spartan,Vertex

- Workbench

- Foundation & ISE

- **Virtex Architecture**

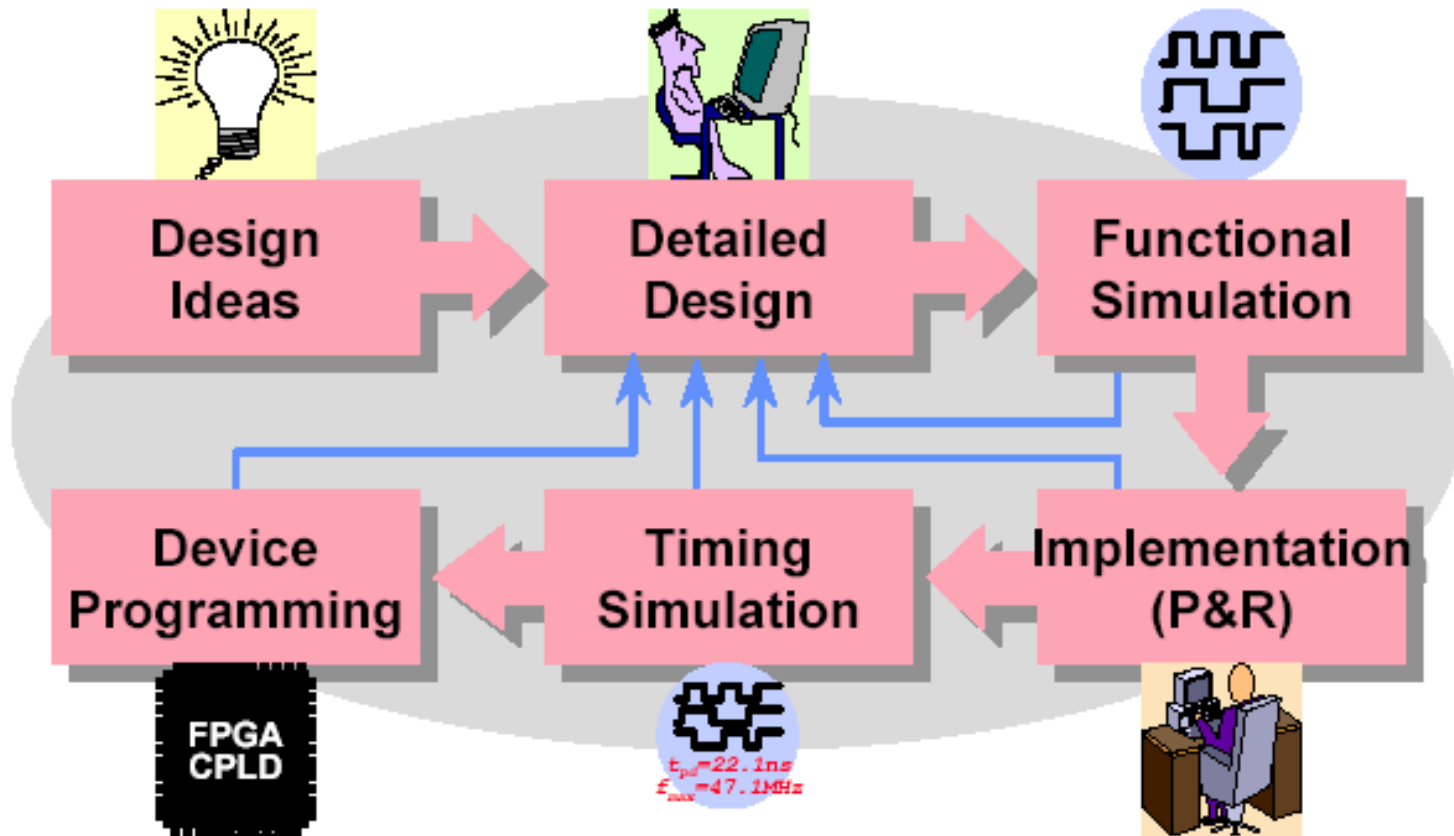
Gate-array like architecture, Configurable logic blocks

I/O blocks 16 signal standards, Block RAM

On-chip memory for higher performance

Clocks & delay-locked loop, Interconnect resources

FPGA/CPLD Design Flow



Note: What is the main difference between cell-based design and FPGA/CPLD design ? 何者相同?何者不同?
Flow, tool, cost, time, ...

Detailed Design (1/2)

- Choose the suitable architecture
 - Algorithm, spec. to architecture mapping
 - Parallel, pipeline, serial, DA,
- Choose the design entry method
 - Schematic
 - Gate level design
 - Intuitive & easy to debug
 - HDL (Hardware Description Language), Verilog & VHDL
 - Descriptive & portable
 - Easy to modify
 - RTL (register transfer level) coding
 - Mixed HDL & schematic



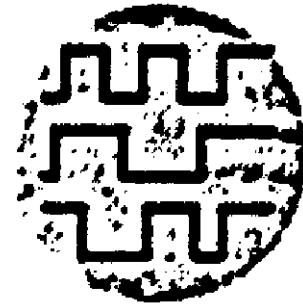


Detailed Design (2/2)

- Manage the design hierarchy
 - Design partitioning
 - Chip partitioning
 - Logic partitioning
 - Use vendor-supplied libraries or parameterized libraries to reduce design time
 - Create & manage user-created libraries (circuits)

Functional Simulation

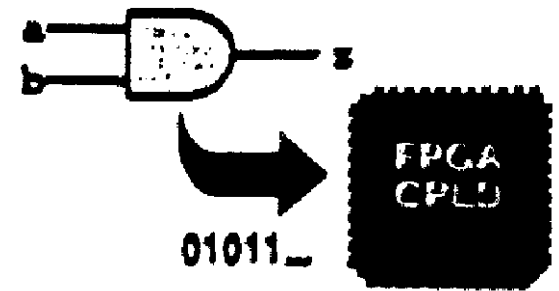
- Preparation for simulation
 - Generate simulation patterns
 - Waveform entry
 - HDL testbench
 - Generate simulation netlist
- Functional simulation
 - To verify the functionality of your design only
- Simulation results
 - Waveform display
 - Text output
- Challenge
 - Sufficient & efficient test patterns



Design Implementation (1/2)

- Implementation flow

- Natlist merging, flattening, data base building
- Design rule checking
- Logic optimization
- Block mapping & placement
- Net routing
- Configuration bitstream generation





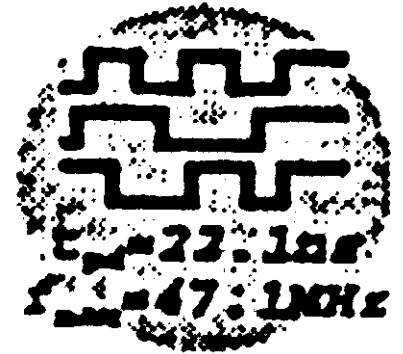
Design Implementation (2/2)

- Implementation results
 - Design error or warnings
 - Device utilization
 - Timing reports
- Challenge
 - How to reach high performance & high utilization implementation?

Timing Analysis & Simulation

■ Timing analysis

- Timing analysis is static, i.e., independent of input & output patterns
- To examine the timing constraints
- To show the detailed timing paths
- Can find the critical path

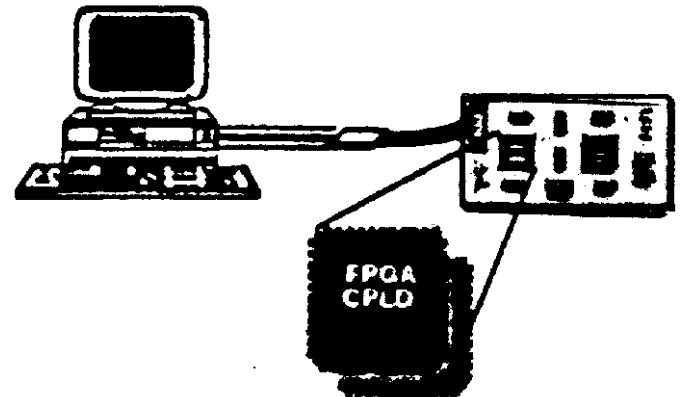


■ Timing simulation

- To verify both the functionality & timing of the design

Device Programming (1/2)

- Choose the appropriate configuration scheme
 - SRAM-based FPGA/CPLD devices
 - Downloading the bitstream via a download cable
 - Programming onto a non-volatile memory device & attaching it on the circuit board
 - OTP, EPROM, EEPROM or Flash-based FPGA/CPLD devices
 - Using hardware programmer
 - ISP



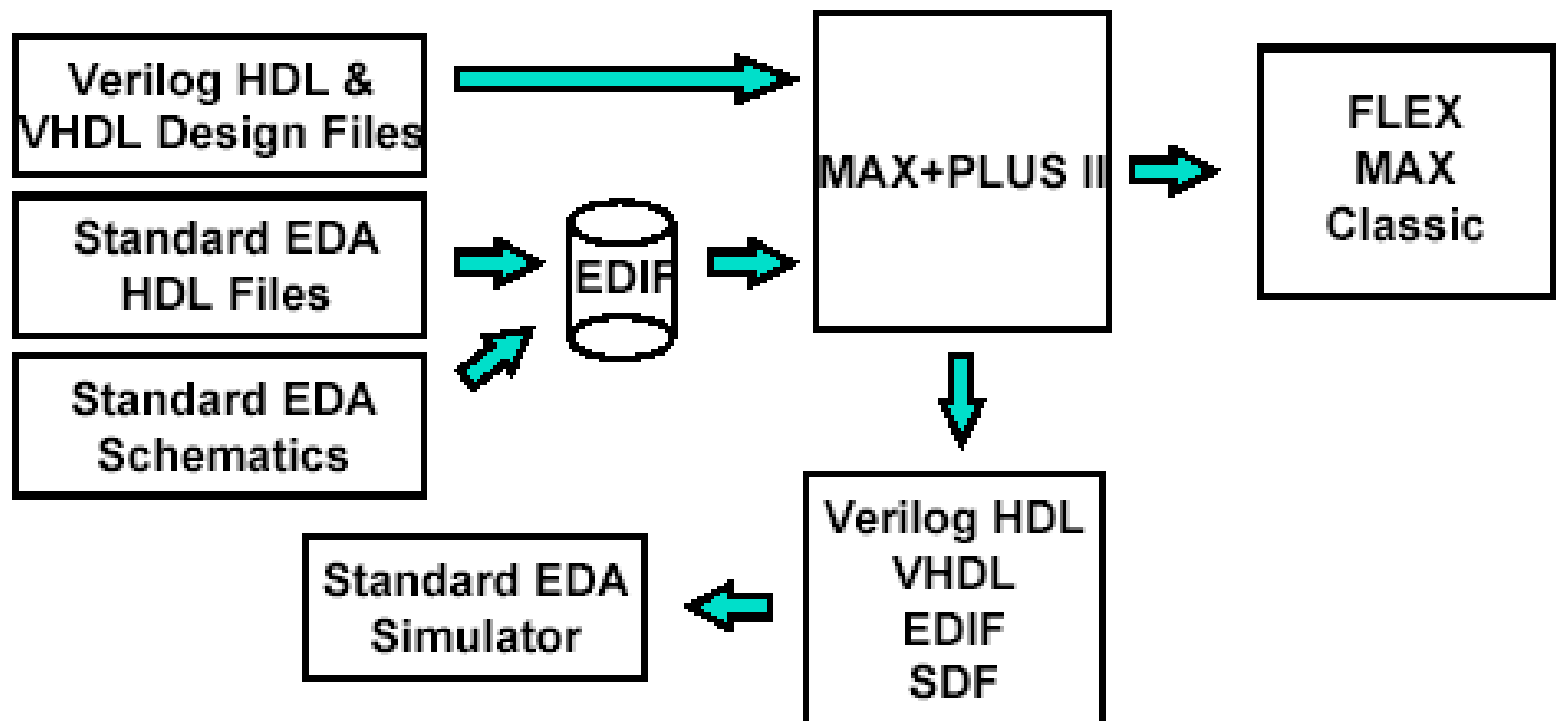


Device Programming (2/2)

- Finish the board design
- Program the device
- Challenge
 - Board design
 - System considerations

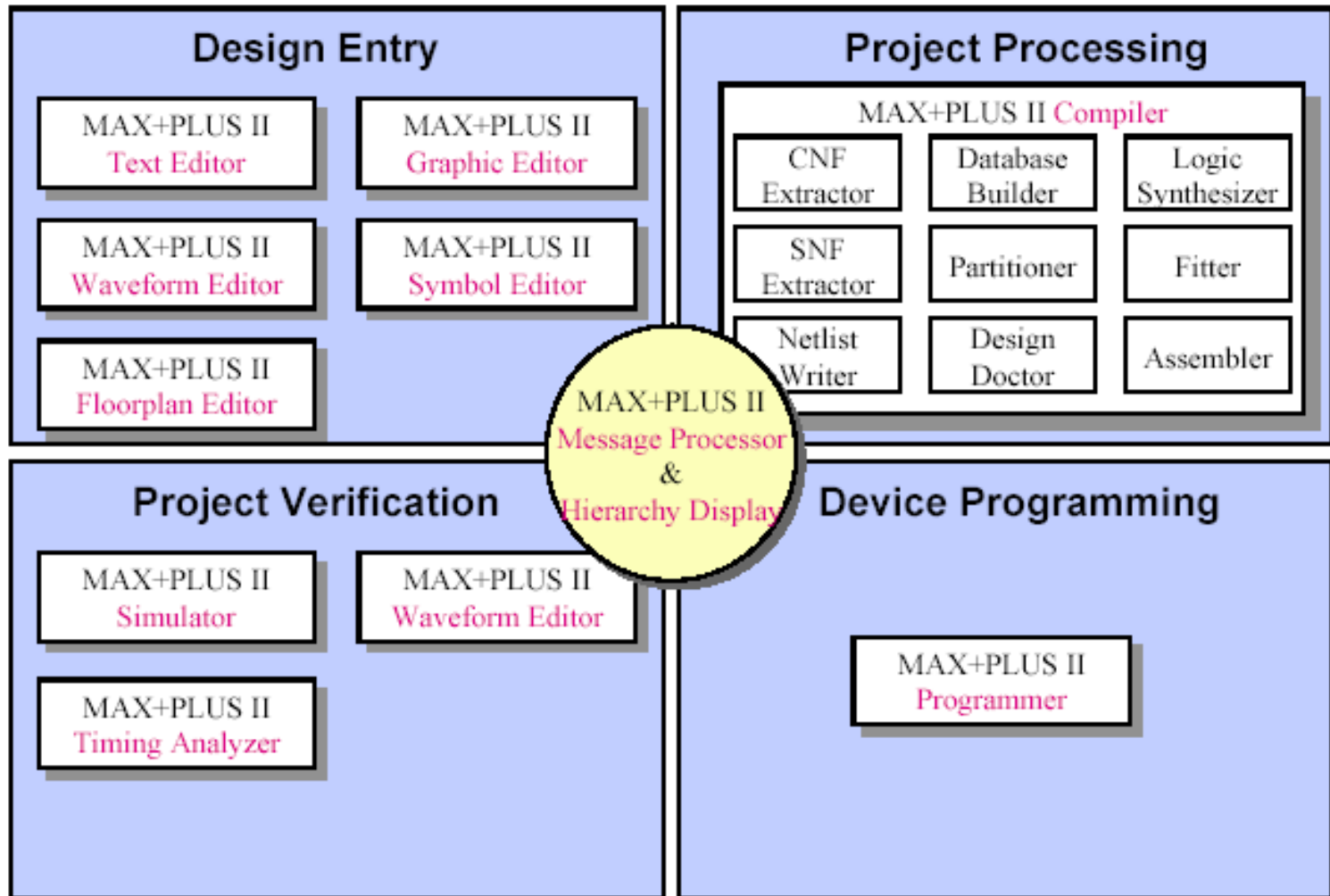
Altera Design Flow (Max-Plus II)

- Operate seamlessly with other EDA tools





MAX+PLUS II Altera Fully-Integrated Development System

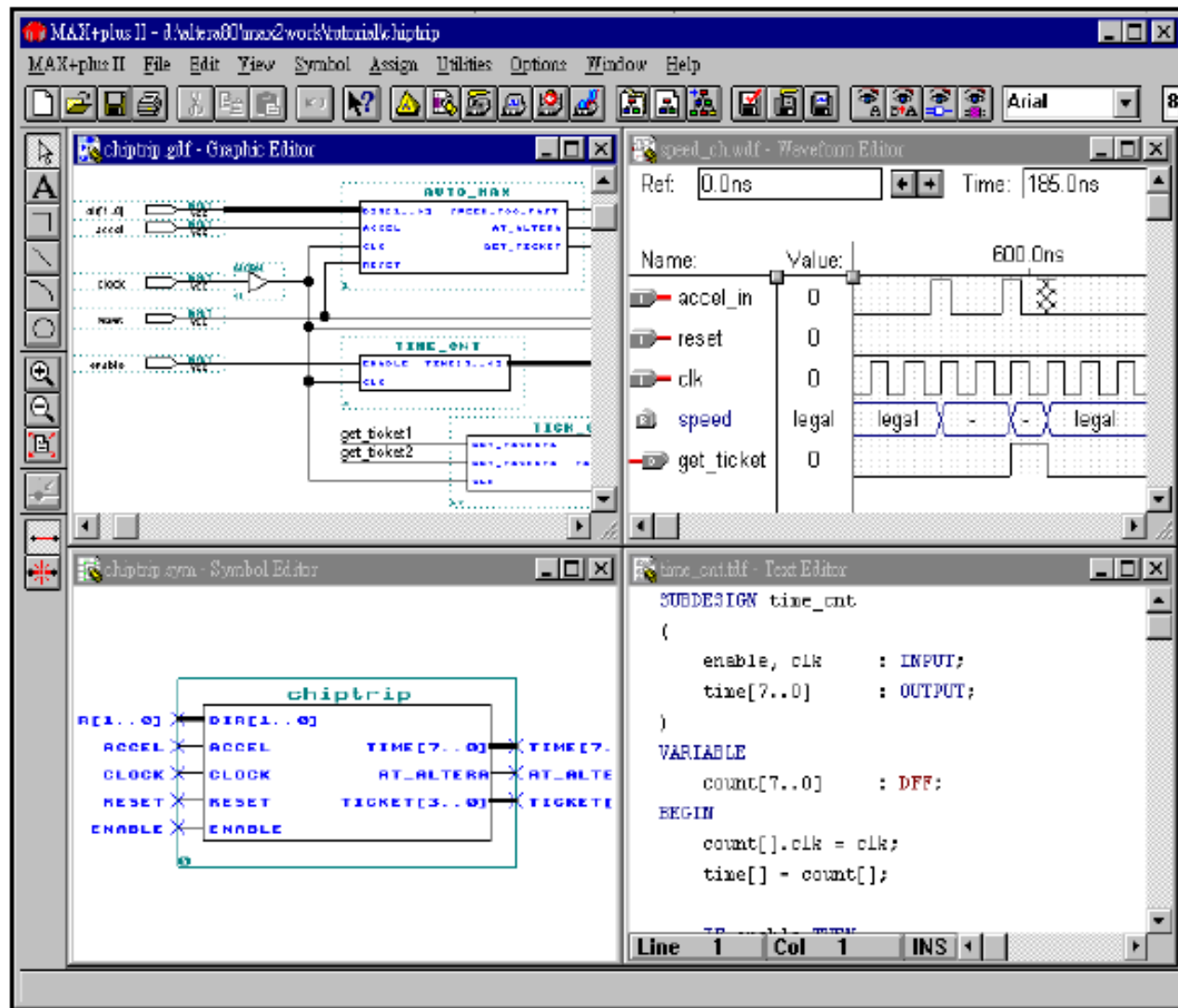




Design Entry (1/2)

- MAX+PLUS II design entry tools
 - Graphic Editor & Symbol Editor
 - For schematic designs
 - Text Editor
 - For AHDL, Verilog HDL, and VHDL designs
 - Waveform Editor
 - Floorplan Editor
 - Hierarchy Display

Design Entry (2/2)



The screenshot displays the MAX+plus II software interface with four active windows:

- chiptrip.gdf - Graphic Editor:** Shows a logic diagram with three main blocks: **AUTO_HRM**, **TIME_CNT**, and **TICKET**. Inputs include **accel**, **clock**, **enable**, and **reset**. Outputs include **AT_ALTERA**, **AT_ALTE**, and **TICKET**. The **TIME_CNT** block is a counter that increments on the clock edge when enabled.
- speed_clk.wdf - Waveform Editor:** Displays a timing diagram for signals: **accel_in**, **reset**, **clk**, **speed**, and **get_ticket**. The **speed** signal is a periodic square wave. The **get_ticket** signal is a single pulse. The **clk** signal is a periodic square wave. The **reset** signal is a single pulse. The **accel_in** signal is a single pulse. The **speed** signal is labeled with a value of 600.0ns.
- chiptrip.sym - Symbol Editor:** Shows a symbol for the **chiptrip** component. Inputs are **DIR[1..0]**, **ACCEL**, **CLOCK**, **RESET**, and **ENABLE**. Outputs are **TIME[7..0]**, **AT_ALTE**, and **TICKET**.
- time_cnt.blf - Text Editor:** Contains the Verilog code for the **time_cnt** component:

```
SUBDESIGN time_cnt
(
    enable, clk      : INPUT;
    time[7..0]      : OUTPUT;
)
VARIABLE
    count[7..0]    : DFF;
BEGIN
    count[0].clk = clk;
    time[] = count[];
```



Project Processing (1/2)

- MAX+PLUS II tools for project processing (implementation)
 - MAX+PLUS II Compiler
 - MAX+PLUS II Floorplan Editor
 - For pin, logic cell location assignments
 - Message Processor
 - For error detection & location

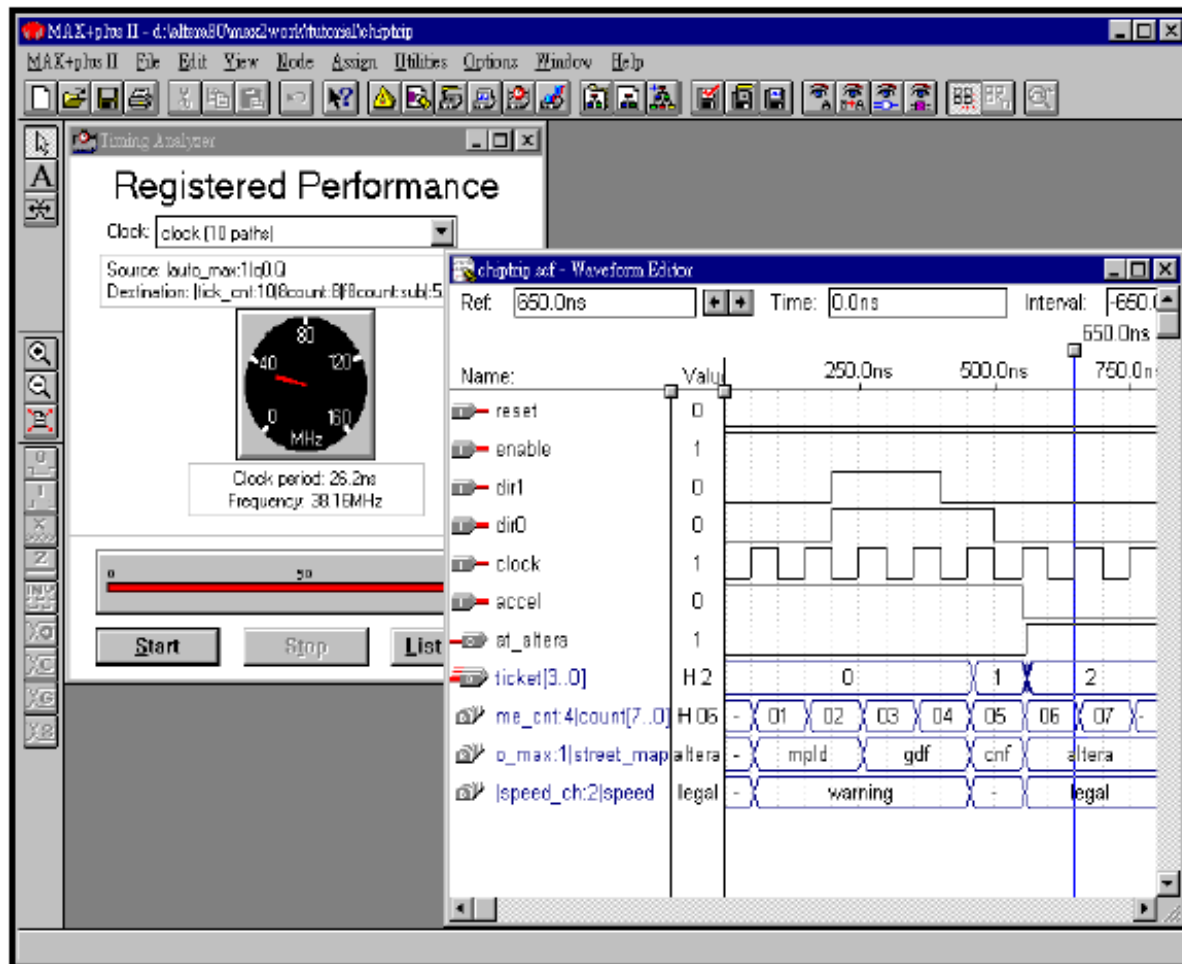
The screenshot displays the MAX+plus II software interface. The top menu bar includes File, Edit, View, Layout, Assign, Utilities, Options, Window, and Help. Below the menu is a toolbar with various icons. The main window is titled "MAX+plus II - d:\altera80max2\work\tutorial\ehp\hp". The "Compiler" window is open, showing buttons for Compiler Netlist Extractor, Database Builder, Logic Synthesizer, Partitioner, Fitter, Timing SNF Extractor, and Assembler. A progress bar is visible below these buttons. The "Floorplan Editor" window is open, showing a grid of columns (AoD, Any Col, Col 1 to Col 15) and rows (Row A, Row B). A purple line is drawn across the grid, and a callout box points to a specific cell in Row B, Column 2, labeled "time_cnt4.time7 @ LC2_B1".



Project Verification (1/2)

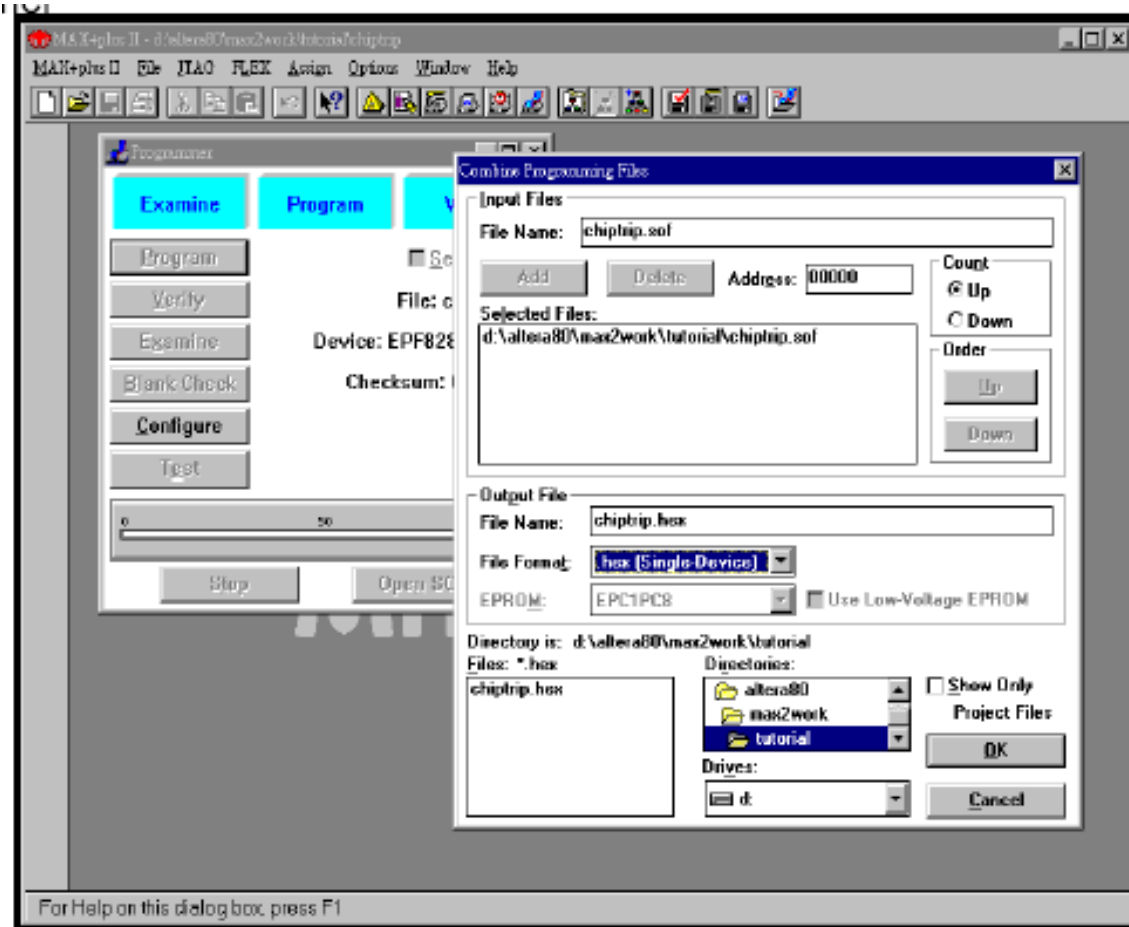
- MAX+PLUS II tools for project verification
 - MAX+PLUS II Simulator
 - MAX+PLUS II Waveform Editor
 - MAX+PLUS II Timing Analyzer

Project Verification (2/2)



Device Programming

- MAX+PLUS tool for device programming
 - MAX+PLUS II Programmer





MAX+PLUS II Features (1/2)

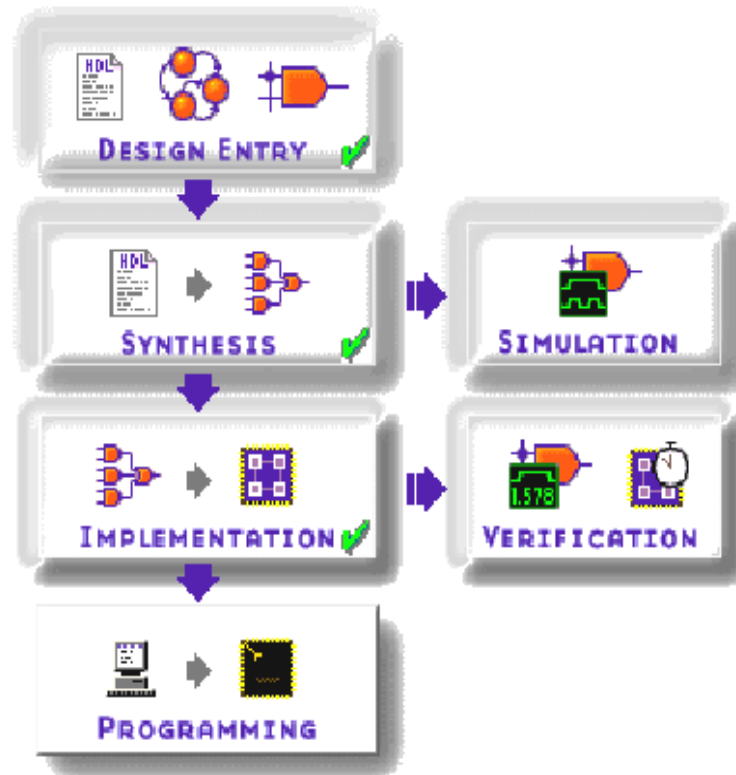
- MAX+PLUS II, Altera's fully integrated design environment ([jump to page 4](#))
 - Schematic, text (AHDL, VHDL, and Verilog HDL), waveform design entry & hierarchy display
 - Floorplan editing
 - RC, logic synthesis & fitting, timing-driven compilation
 - Multi-device partitioning
 - Automatic error location



MAX+PLUS II Features (2/2)

- Functional simulation, timing simulation, and multi-device simulation
- Timing analysis
- Programming file generation & device programming
- EDA interface : industry-standard library support, EDA design entry & output formats (EDIF, Verilog & VHDL)
- On-line help

Xilinx Design Flow (Foundation)





References

- (1) [http:// www.xilinx.com/xilinx FPGA.pdf](http://www.xilinx.com/xilinx_FPGA.pdf)
- (2) http://www.altera.com /alt_lab.pdf
- (3) <http://www.cic.edu.tw/~steven/stratix.htm/>
- (4) <http://www.altera.com/literature/ds/acex.pdf>
- (5) 教育部P&L聯盟-FPGA系統設計實務(摘自陳漢臣老師部份)