電子系統層級設計與驗證

Electronic System-Level Design and Verification

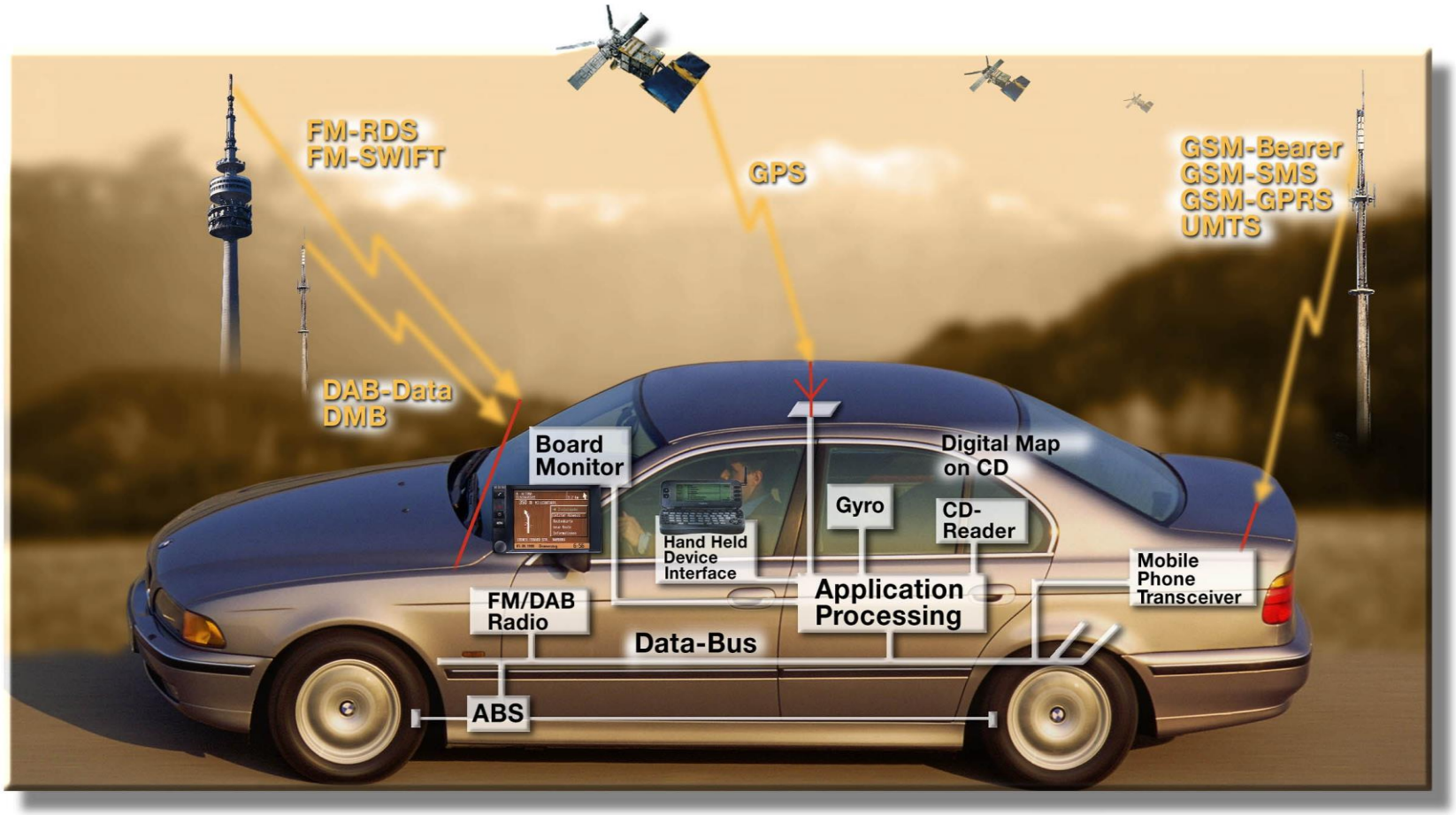
Shiann-Rong Kuang (鄺獻榮)

Dept. of Computer Science and Engineering
National Sun Yat-sen University

Tel: (07)5252000 ext. 4340
E-mail: srkuang@cse.nsysu.edu.tw
Office: エEC3008

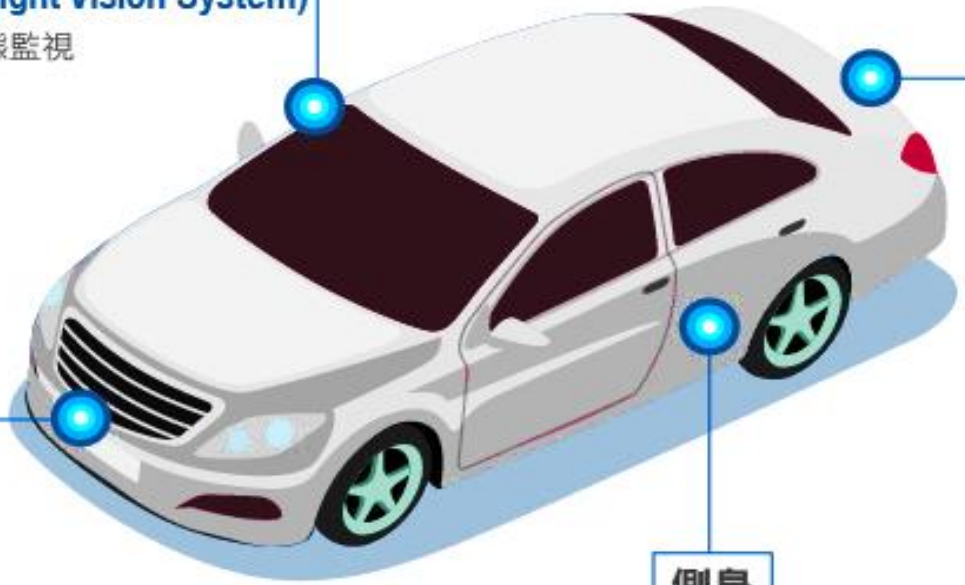
Chip Everywhere!

An Example


FM-RDS
FM-SWIFT
GPS
GSM-Bearer
GSM-SMS
GSM-GPRS
UMTS
DAB-Data
DMB
Board Monitor
Digital Map on CD
Gyro
CD-Reader
Hand Held Device Interface
Mobile Phone Transceiver
Application Processing
FM/DAB Radio
Data-Bus
ABS

Google Driverless Car

ADAS, Advanced Driver-assistance System



駕駛艙

抬頭顯示器 (Head Up Display)

夜視系統 (Night Vision System)

駕駛人生理狀態監視

車尾

後方碰撞警示系統
(Rear Crash Collision Warning System)

停車輔助系統 (Parking Aid System)

車頭

緩解撞擊煞車系統 (Collision Mitigation System)

主動車距控制巡航系統 (Adaptive Cruise Control System)

碰撞預防系統 (Pre Crash System)

行人偵測 (Pedestrian Detection)

交通號誌偵測與辨識 (Traffic Sign/ Signal Recognition)

適路性車燈系統 (Adaptive Front/lighting System)

側身

偏離車道警示系統 (Lane Departure Warning System)

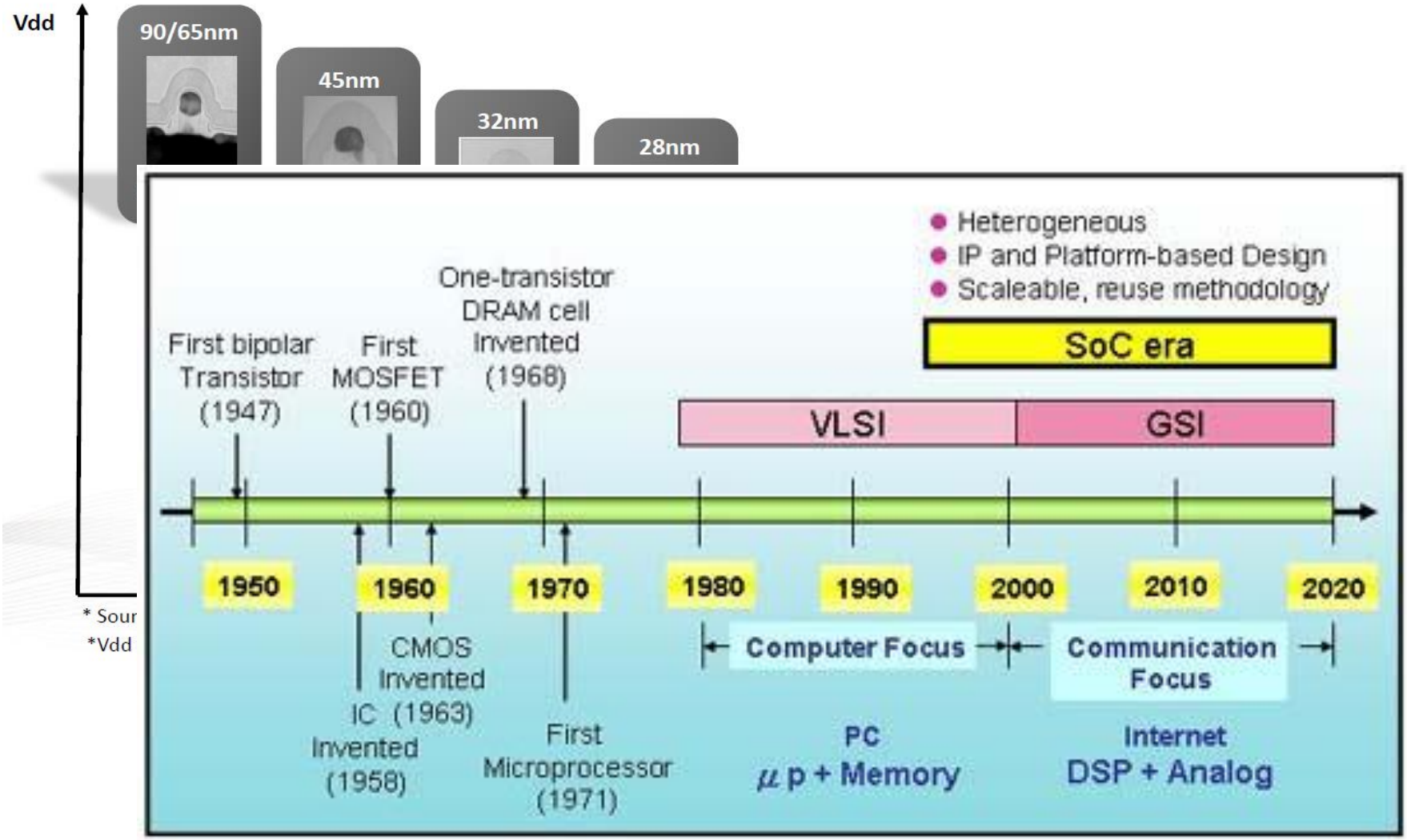
盲點偵測系統 (Blind Spot Detection System)

360度車身環景系統

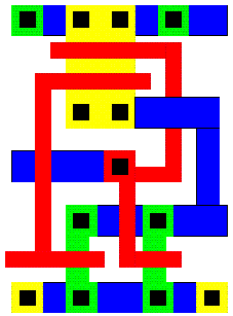
胎壓偵測系統 (TPMS)

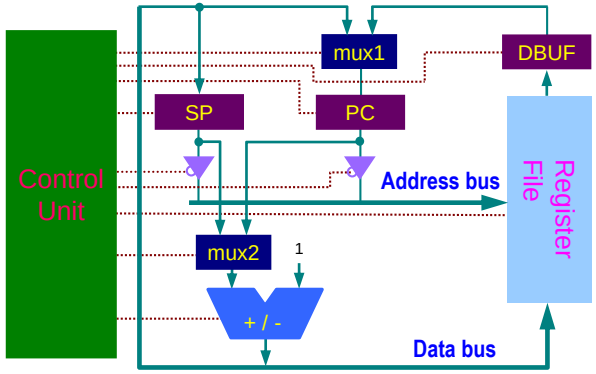
車身穩定控制系統 (Stability Control System)

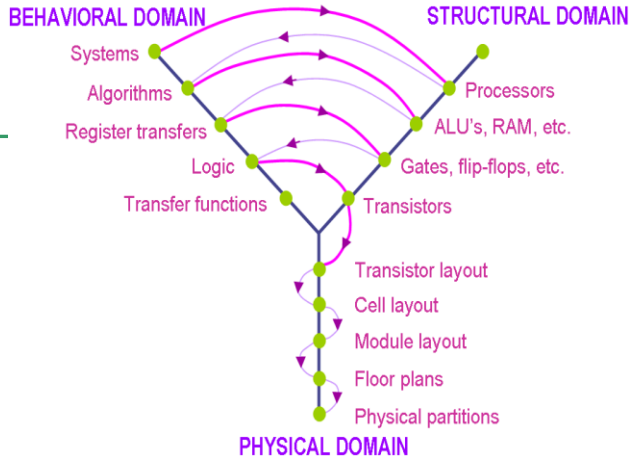
Evolution of IC Technology

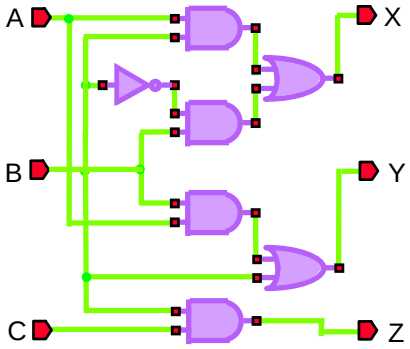


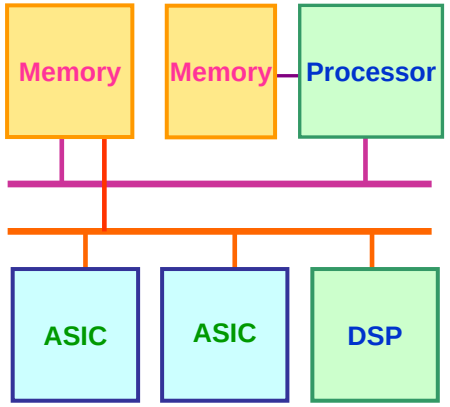
Evolution of IC Design

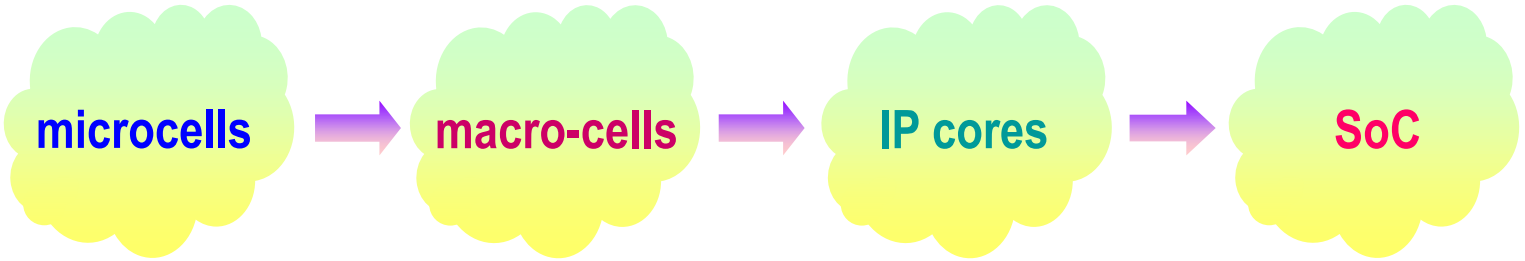




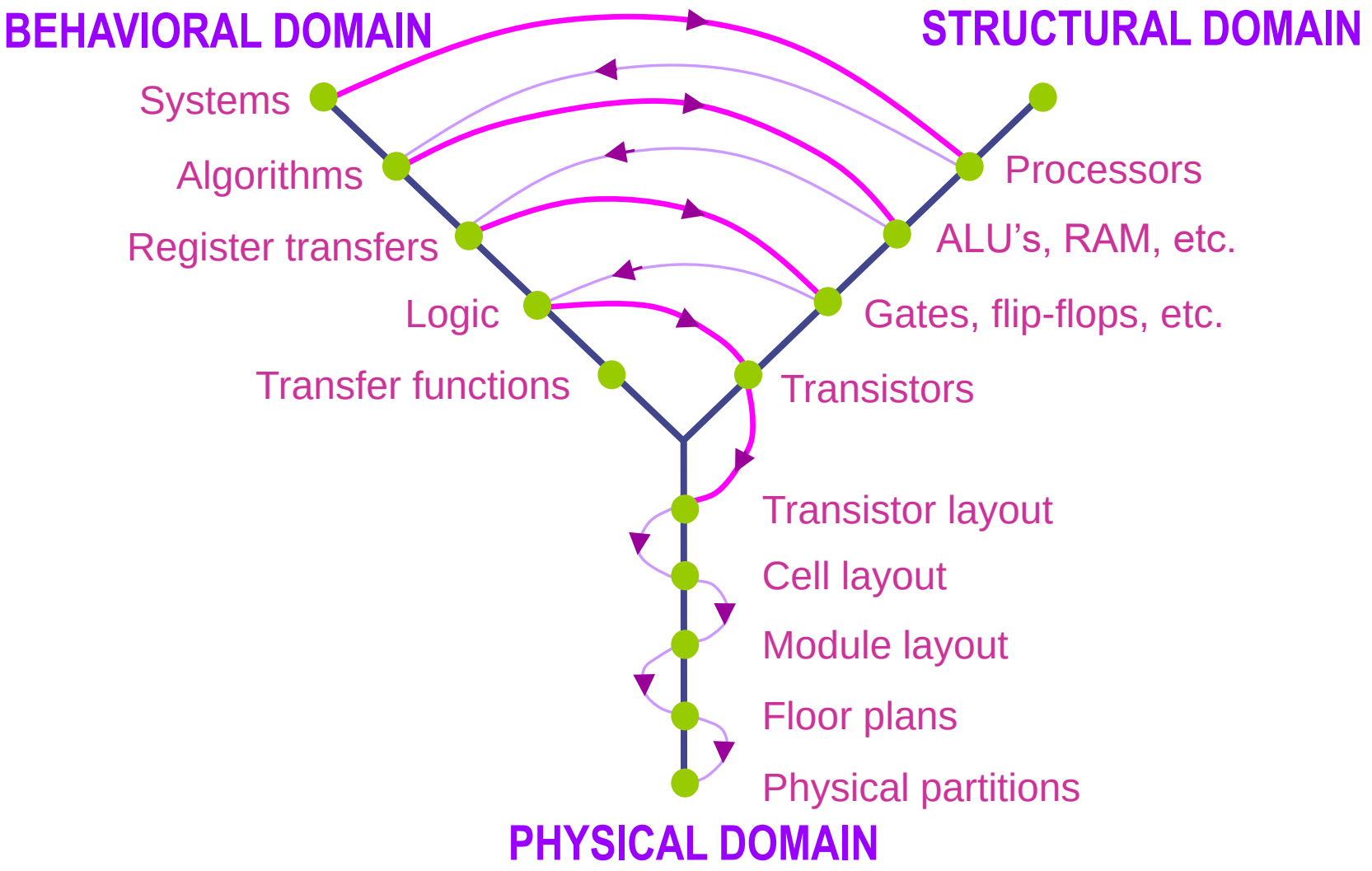






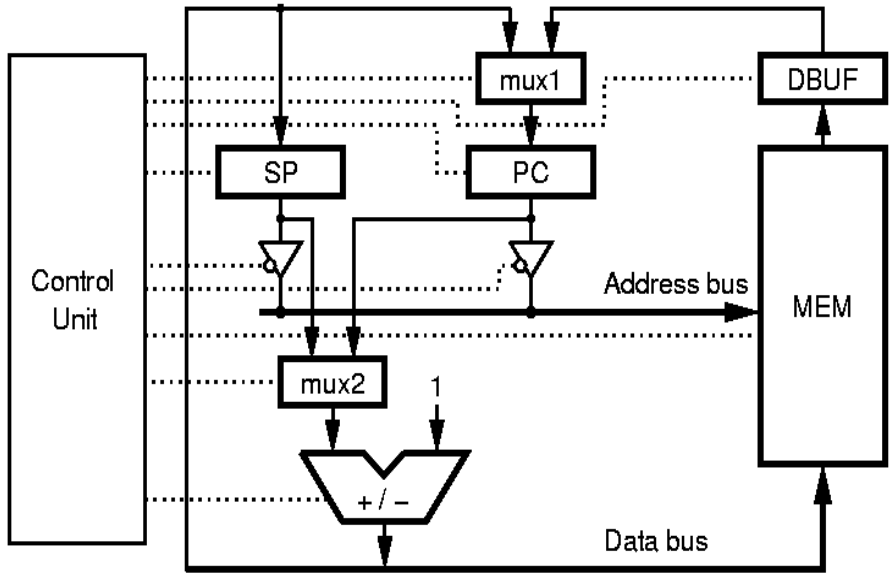


Design Flow

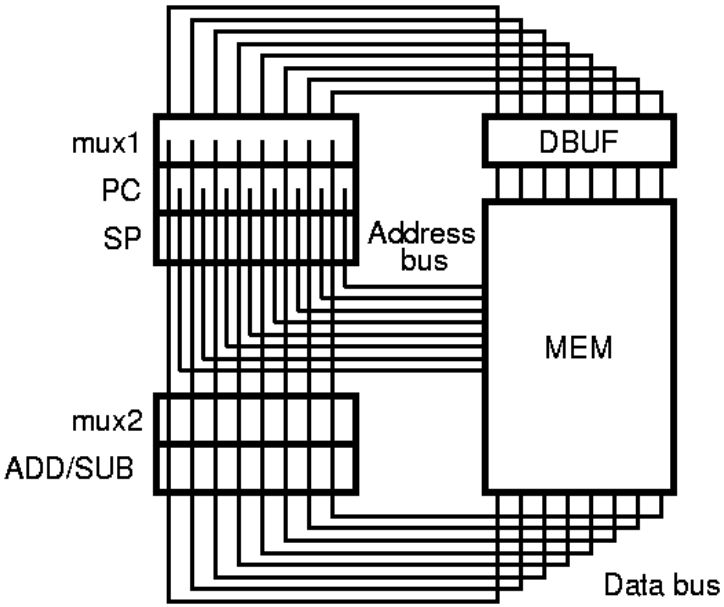

BEHAVIORAL DOMAIN
STRUCTURAL DOMAIN
Systems
Algorithms
Register transfers
Logic
Transfer functions
Processors
ALU's, RAM, etc.
Gates, flip-flops, etc.
Transistors
Transistor layout
Cell layout
Module layout
Floor plans
Physical partitions
PHYSICAL DOMAIN

```
if IR(3) = '0' then
    PC          := PC + 1;
else
    DBUF        := MEM(PC);
    MEM(SP)     := PC + 1;
    SP          := SP – 1;
    PC          := DBUF;
end if;
```

Behavior

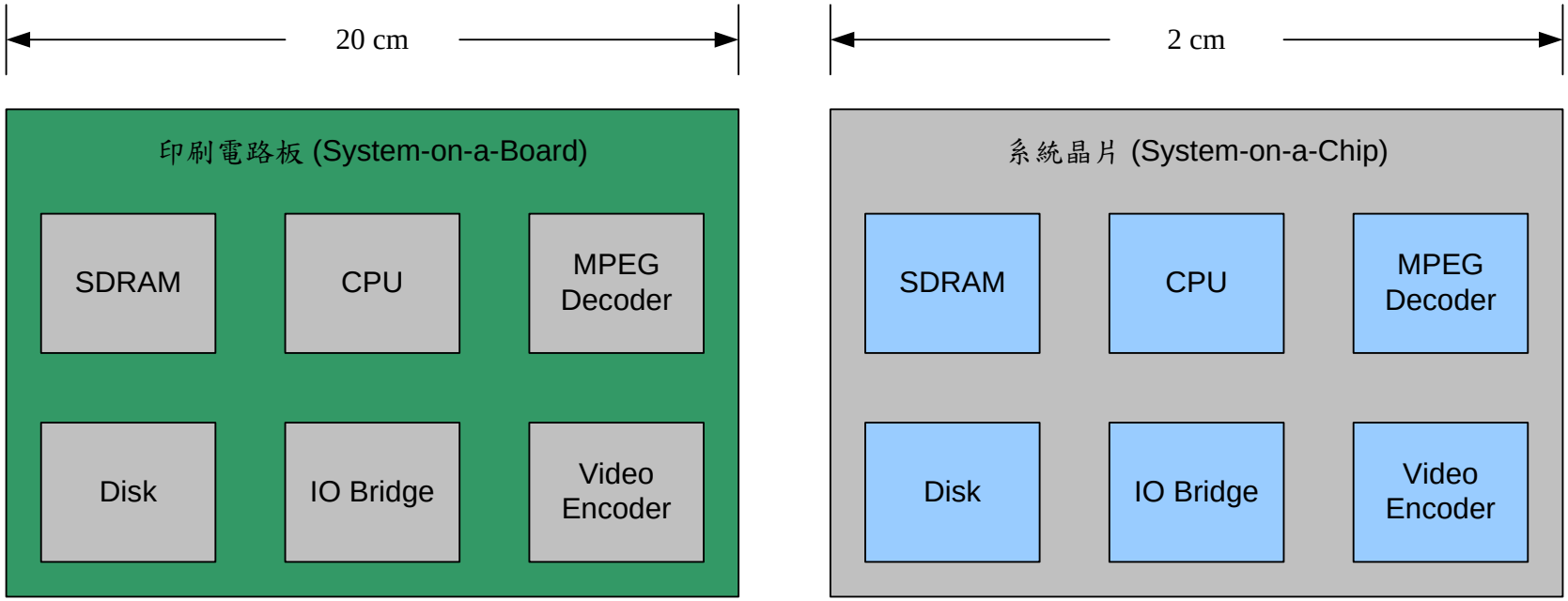


Structure



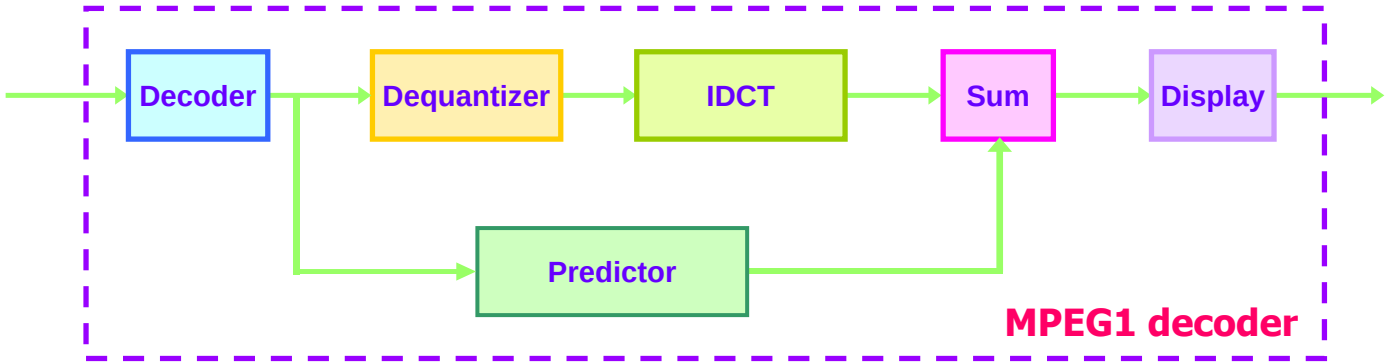
Floorplan

System-on-a-Board v.s. System-on-a-Chip

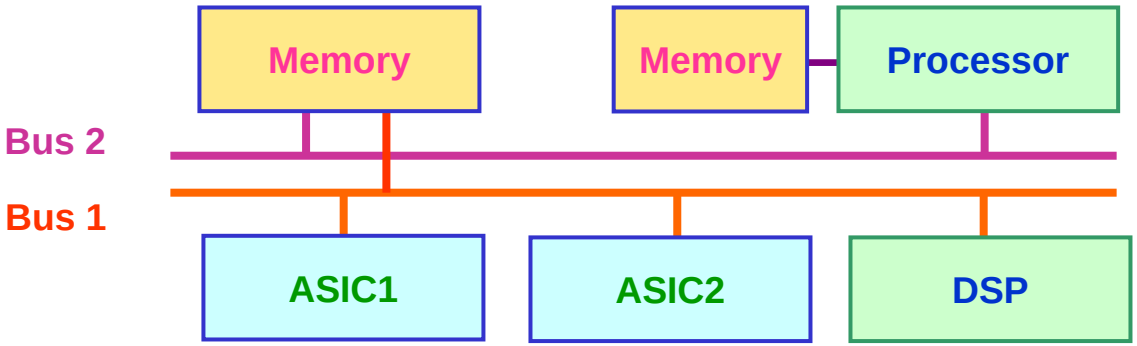


System-level Design

system specification

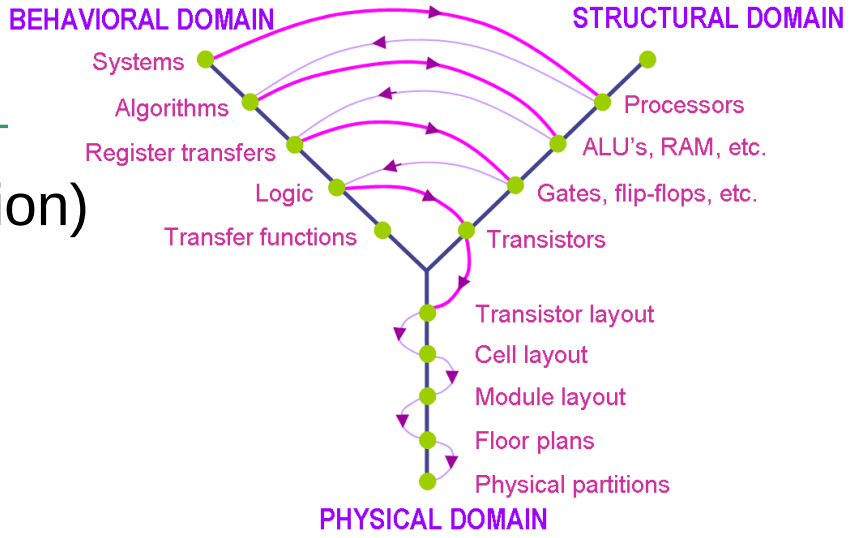


Hardware-software co-design



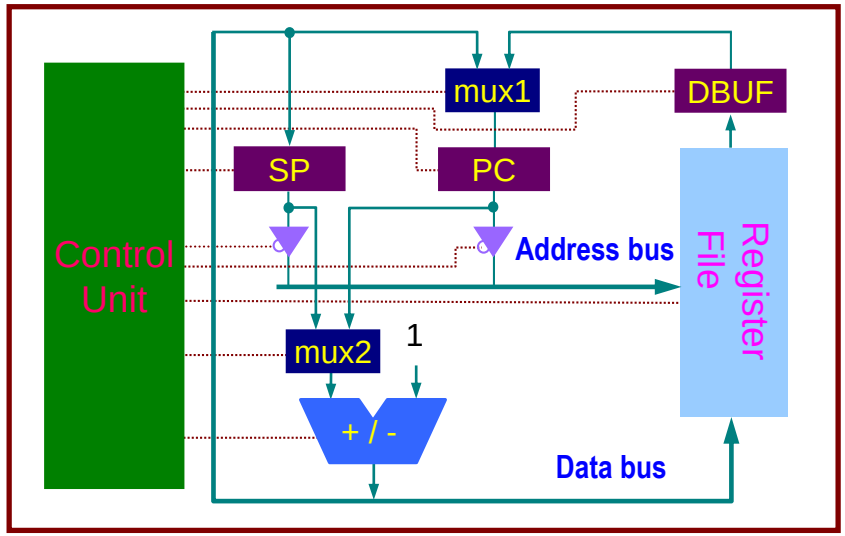
system architecture

Algorithm-level Design



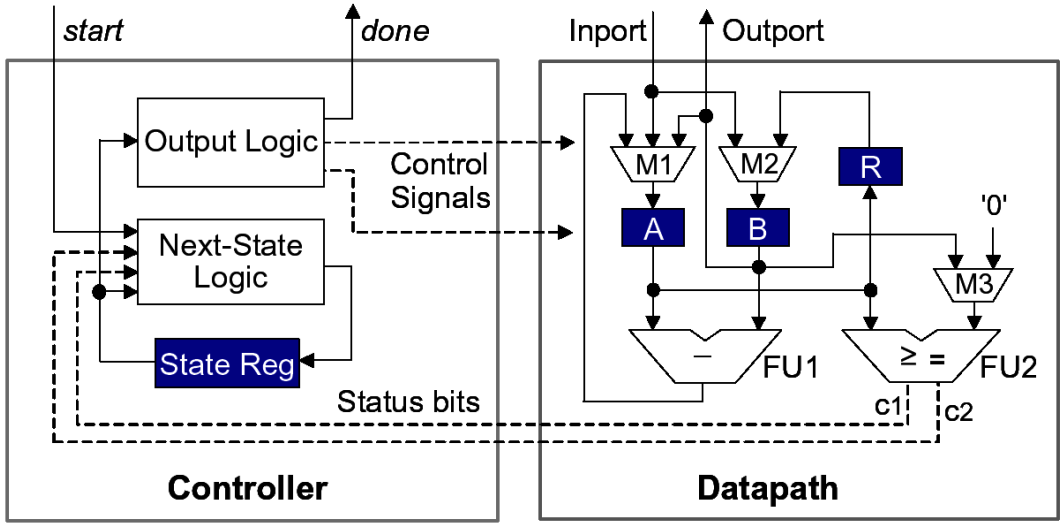
- Specification (behavioral description)
  - Programming languages
    - C or Pascal
  - Hardware description language
    - VHDL or Verilog
- Behavioral (or High-level) synthesis
  - algorithmic behavioral level ⇒ RTL structural descriptions

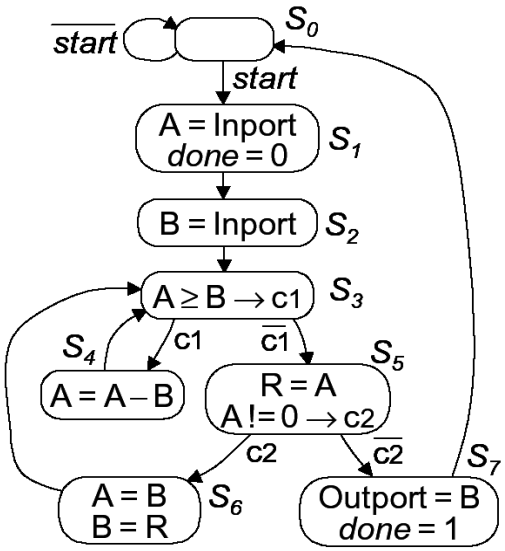
```
if IR(3) = '0' then
    PC          := PC + 1;
else
    DBUF        := MEM(PC);
    MEM(SP)     := PC + 1;
    SP          := SP – 1;
    PC          := DBUF;
end if;
```



Behavioral (High-level) Synthesis

```
          A = Inport;
          B = Inport;
          done = 0;
Repeat:   while (A ≥ B)
              A = A – B;
          R = A;
          if (A != 0) {
              A = B;
              B = R;
              goto Repeat;
          }
          else  goto End;
End:      Outport = B;
          done = 1;
```

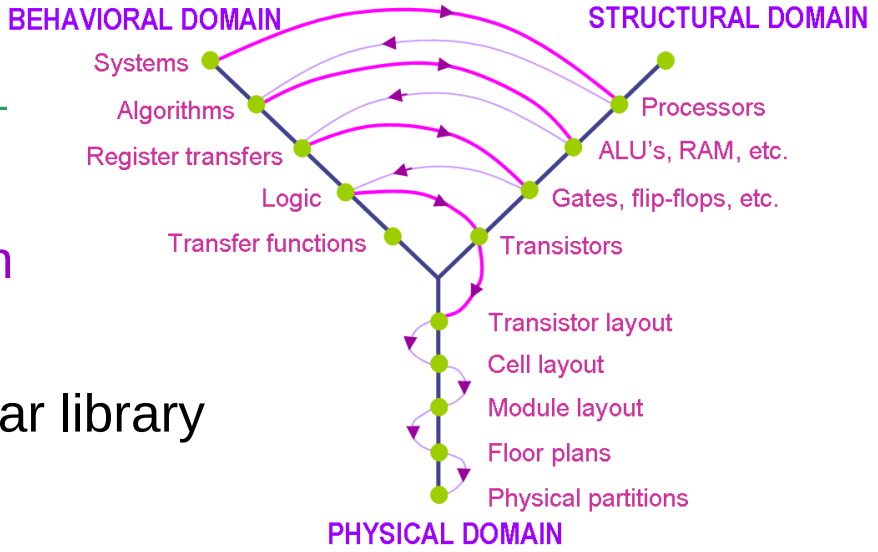


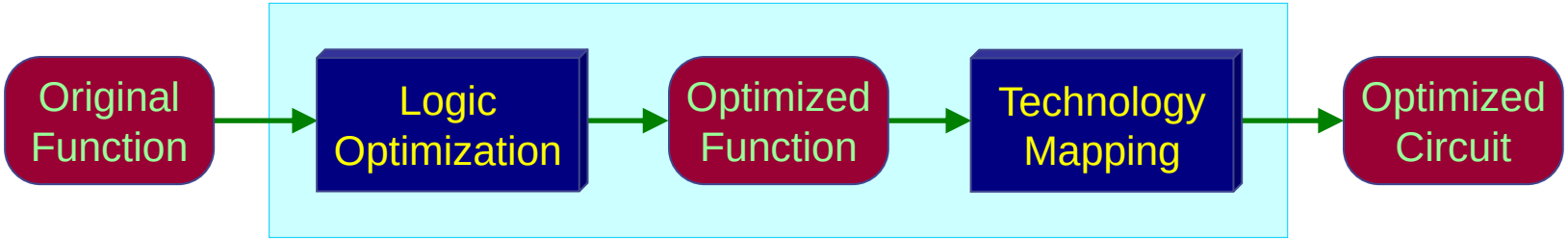


PS	Input	NS	Control signals						
			A	B	R	M1	M2	M3	*done*
S_0	*start* = 0	S_0	-	-	-	-	-	-	0
	start = 1	S_1							
S_1	-	S_2	1	-	-	Inport	-	-	0
S_2	-	S_3	0	1	-	-	Inport	-	0
S_3	c1 = 1	S_4	0	0	0	-	-	B	0
	c1 = 0	S_5							
S_4	-	S_3	1	0	0	FU1	-	-	0
S_5	c2 = 1	S_6	0	0	1	-	-	'0'	0
	c2 = 0	S_7							
S_6	-	S_3	1	1	0	B	R	-	0
S_7	-	S_0	0	0	0	-	-	-	1

Logic Design

- Logic synthesis
  - RTL design ⇒ gate-level design
  - Boolean expressions ⇒ logic gate networks in a particular library

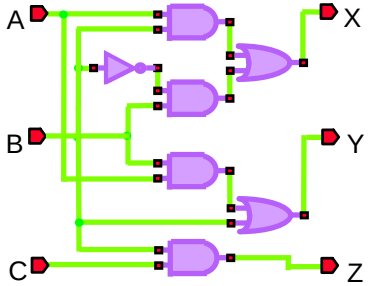




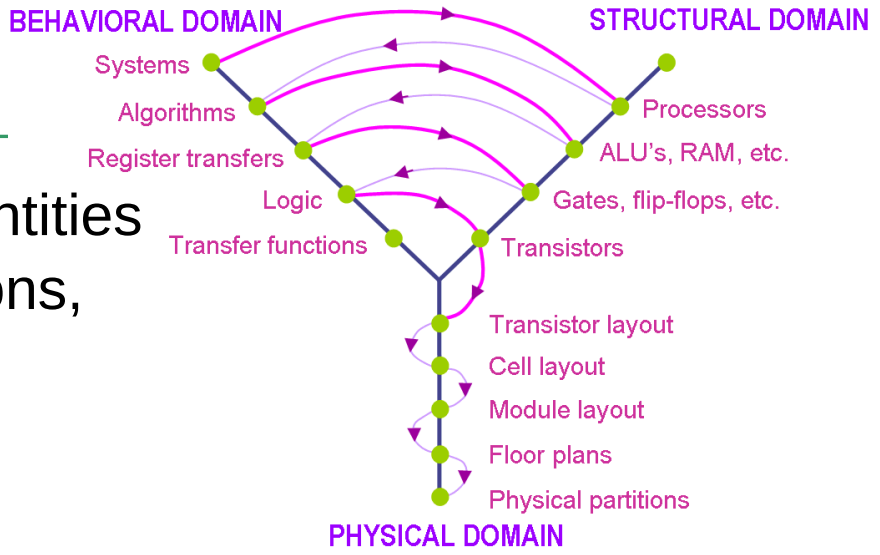
X = AB + AB'C + A'BC

Y = …

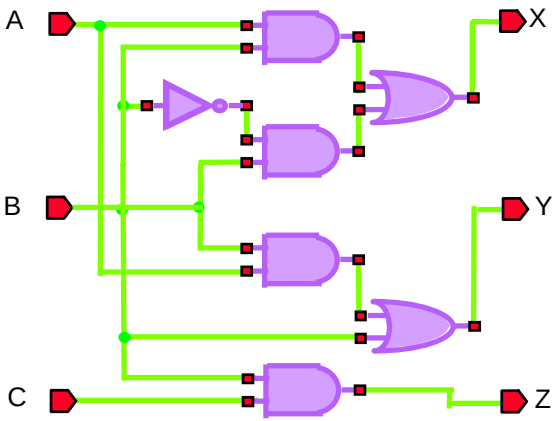
Z = …

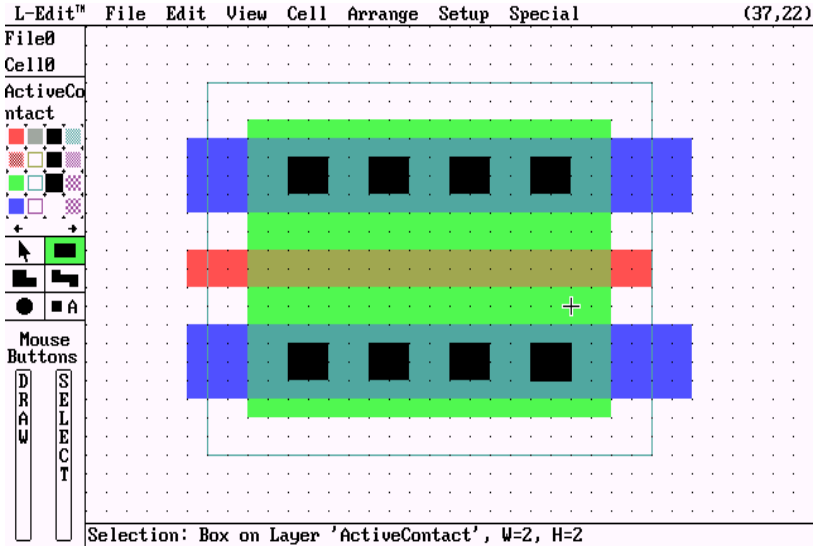


Physical Design

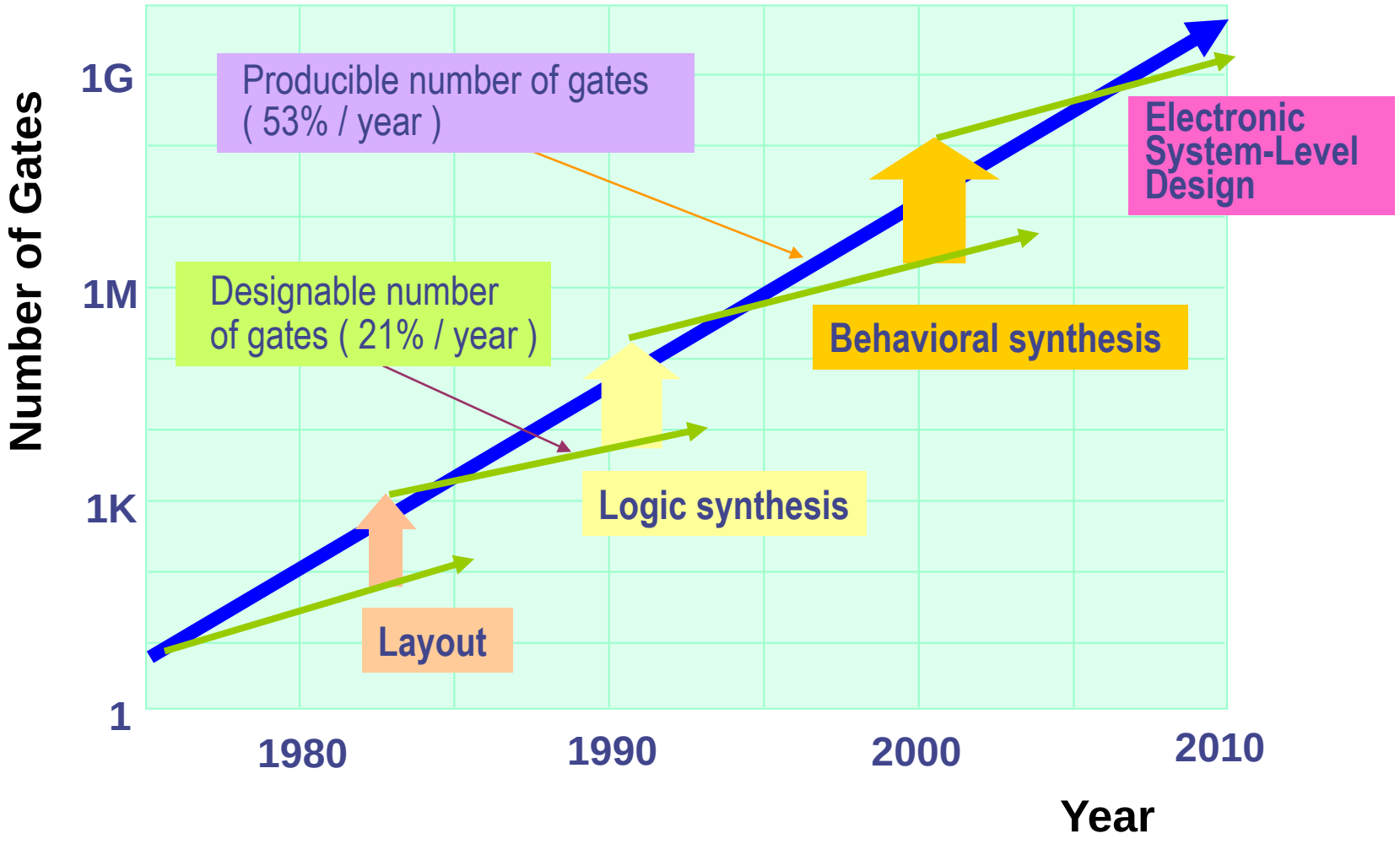


- Create a layout of geometrical entities indicating the transistor dimensions, locations, and their connections





Electronic Design Automation (EDA) tools

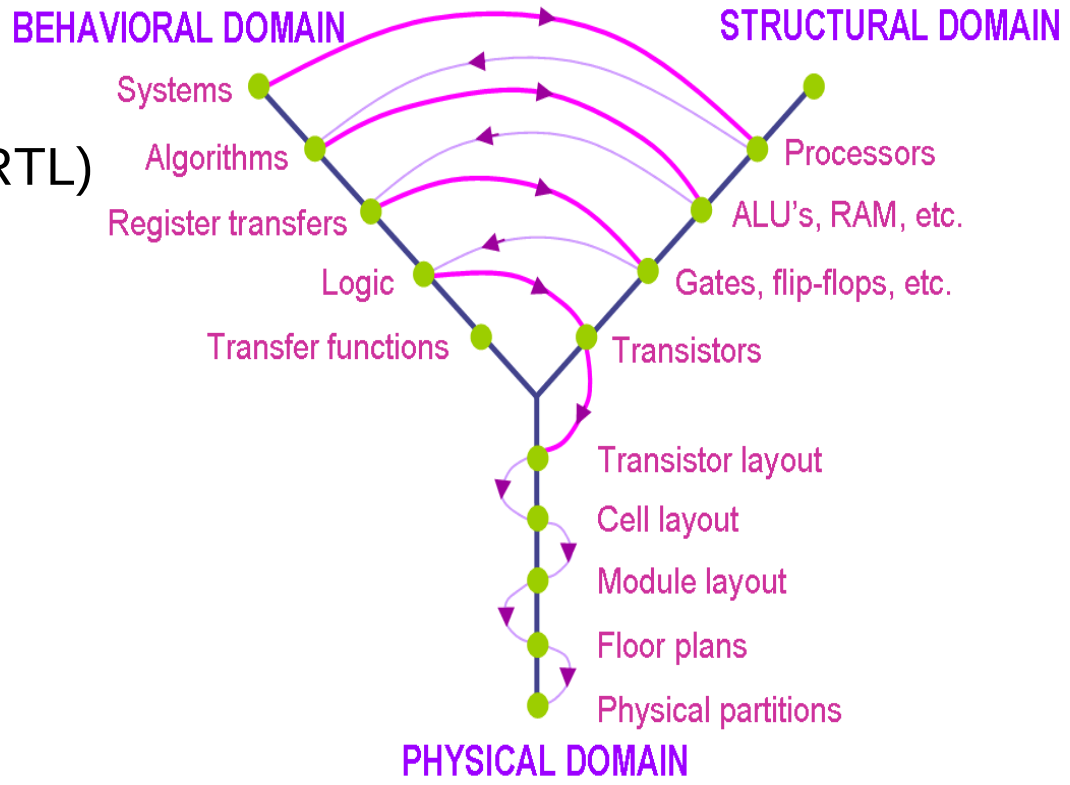


What is ESL?

- ESL = Electronic System Level
- ESL doesn't specify which levels of design should be employed. It focuses on the concepts of designing a system, instead of specific components
- Then, what is Electronic System Level design flow?
- Design, debug, verify the system using ESL methodologies, languages, tools and CONCEPTS

What Does Level Mean?

- In HW design, level means the degree of the design details, or the level of abstraction, of the model of the target design
- For example,
  - Transistor level
  - Gate level
  - Register transfer level (RTL)
  - Transaction level
  - Behavior level
  - Architecture level
  - Algorithmic level
  - ….

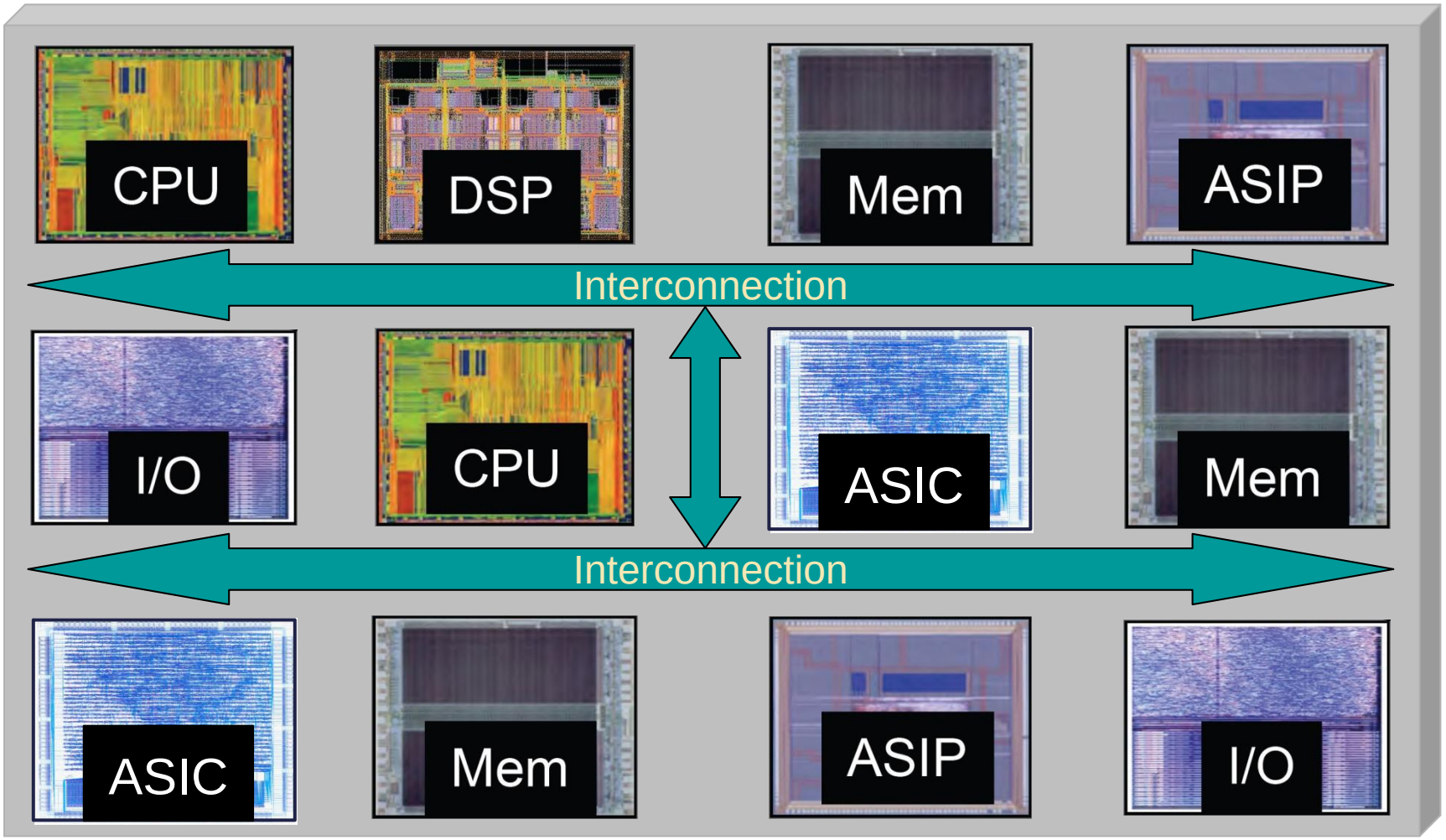


The Problems

- Imagine
  - You have a system with 10 processor cores, each having its own memory system. There are shared memory spaces for the cores. 20 different peripherals to control. There are 20 programmers using 8 different languages to develop 30 different applications on this system which needs to support 2 different OS.
  - And the biggest problem is
    - The FPGA emulator is not even ready yet
- To cope with these problems, what do we need?

Multiprocessor System-on-Chip (MPSoC)

ASIP: Application-Specific Instruction-set Processor

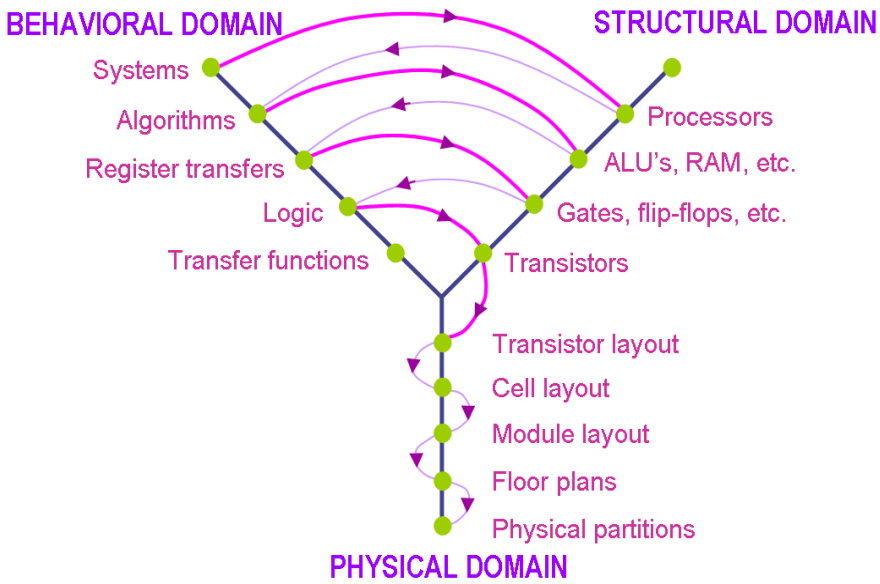


Why ESL Design Flow is Needed?

- Huge system
- Extraordinarily high complexity
- Design reuse
- Slow simulation speed
- Difficulty in integration
- Mixed/multiple disciplines
- HW/SW co-design/co-simulation/co-….
- …..
- Most importantly, time-to-market

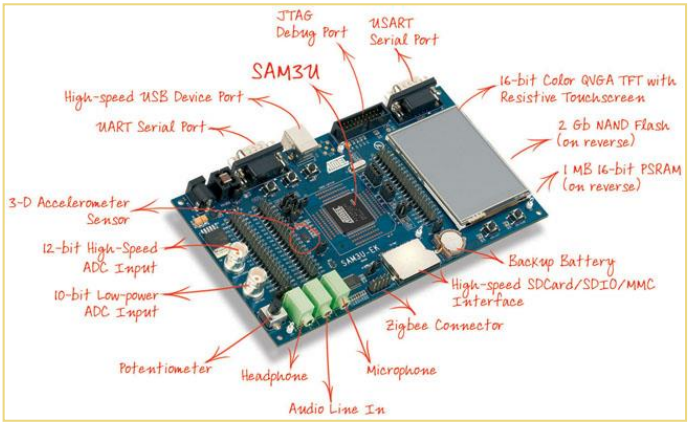
We Need

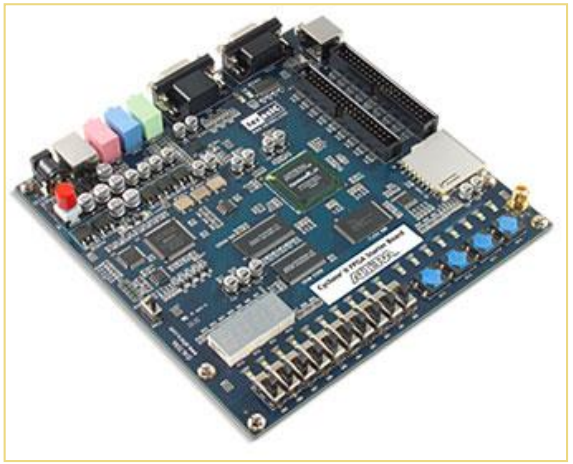
- A super fast simulator
- A simulator supports mixed abstraction level designs
- An integrated HW/SW co-development environment
- A super fast simulation environment
- ….
- To do this, what are the first few steps?



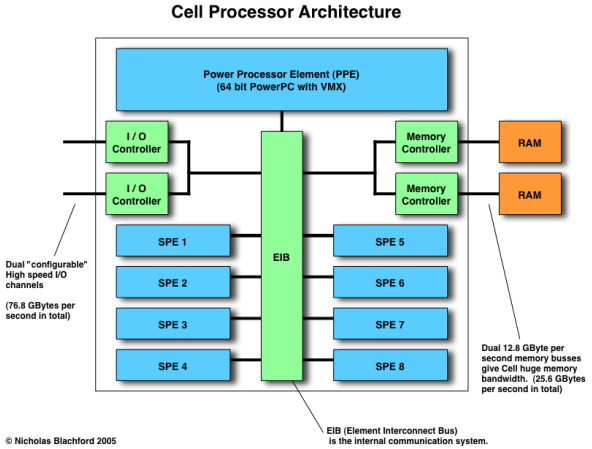
Hardware? FPGA? MCU?

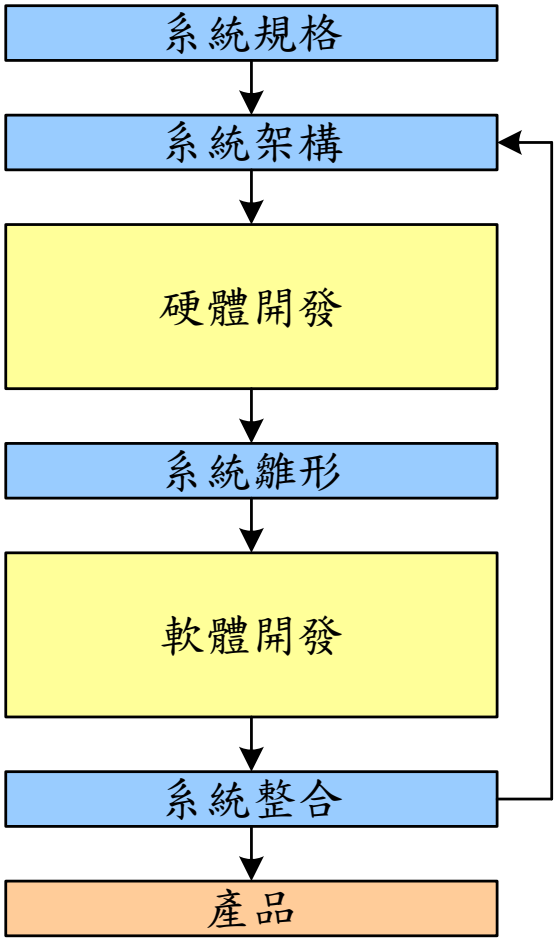
- Microcontroller
  - CPU, Memory, …, etc.
  - Software program
    - (.C->.asm->.out)
- FPGA
  - Verilog code, VHDL code
  - Hardware program
    - (.v->.bit)

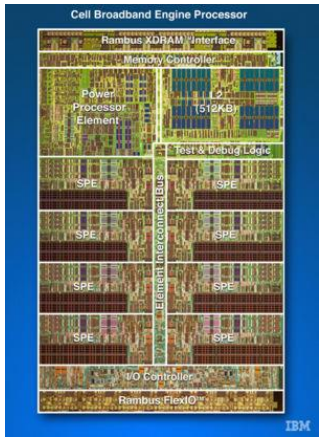


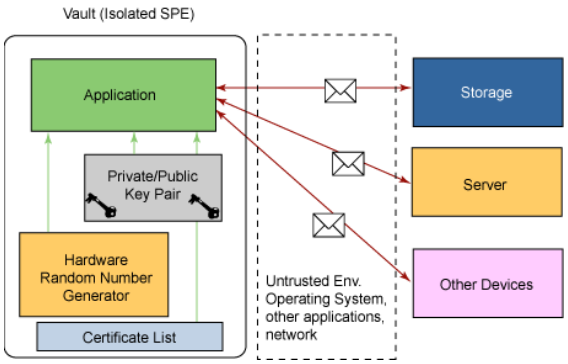


Traditional System Design Flow



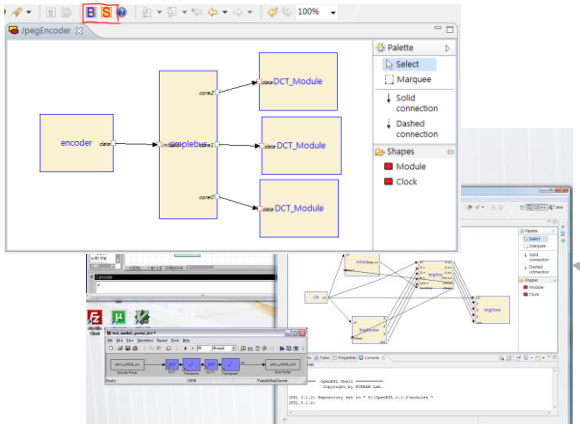


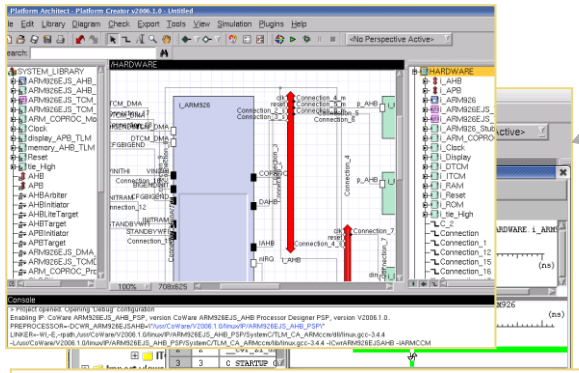




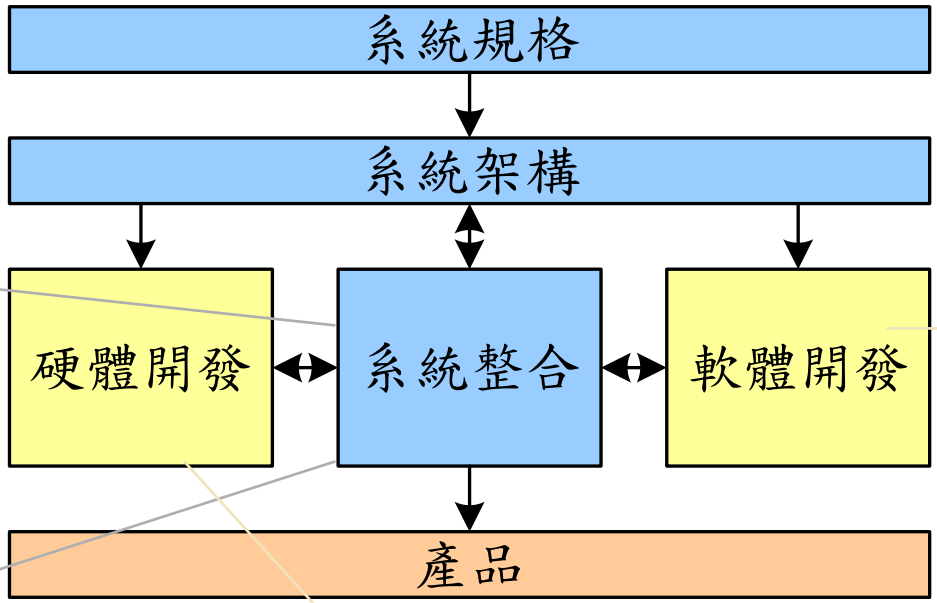
Ideal System Design Flow

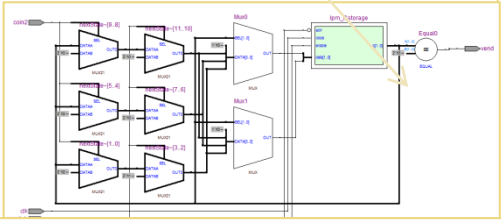
OPENESL：ESL design hardware

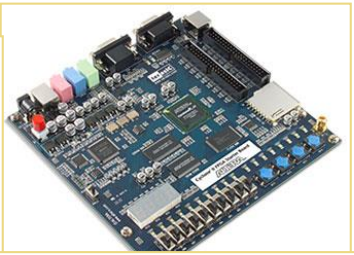




CoWare：ESL design hardware



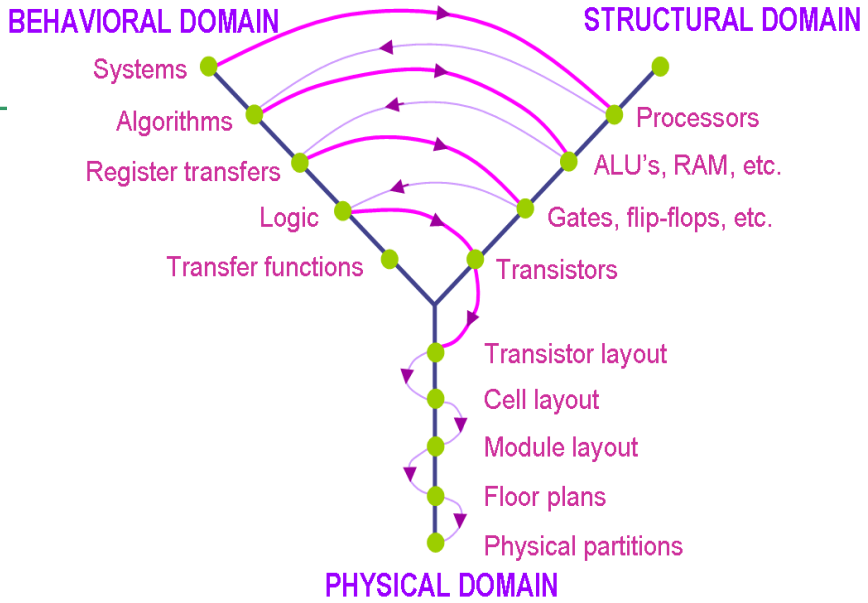


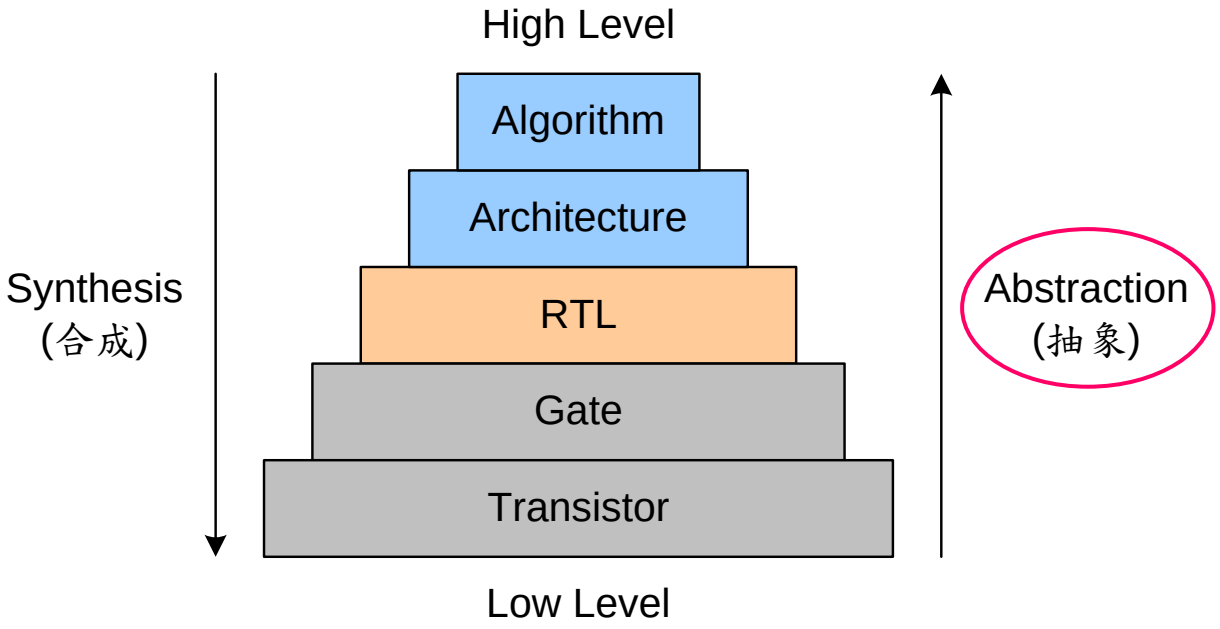


FPGA：Hardware circuit design and verify

Traditional HW Design

- HDL (硬體描述語言)
  - Verilog, VHDL
  - RTL (Register Transfer Level)
- Simulation (模擬)
  - Very slow (e.g. Linux開機)





Why SystemC?

- 標準語法 (IEEE Standard 1666)
  - 模擬系統
  - 更高抽象化
  - 既有的C++開發環境
  - 硬體合成

SystemC Doesn't Speedup Simulation,

High Level does!!

SystemC Use Case

- HW engineer
  - 硬體描述語言(HDL)
  - 更高階、更抽象描述硬體
- System engineer
  - System Virtual Platform (系統虛擬平台)
  - Architecture Analysis (架構分析)
  - Performance Exploration (效能評估)
  - System Verification (系統驗證)
- SW engineer
  - 早期開發環境
  - 更多硬體資訊

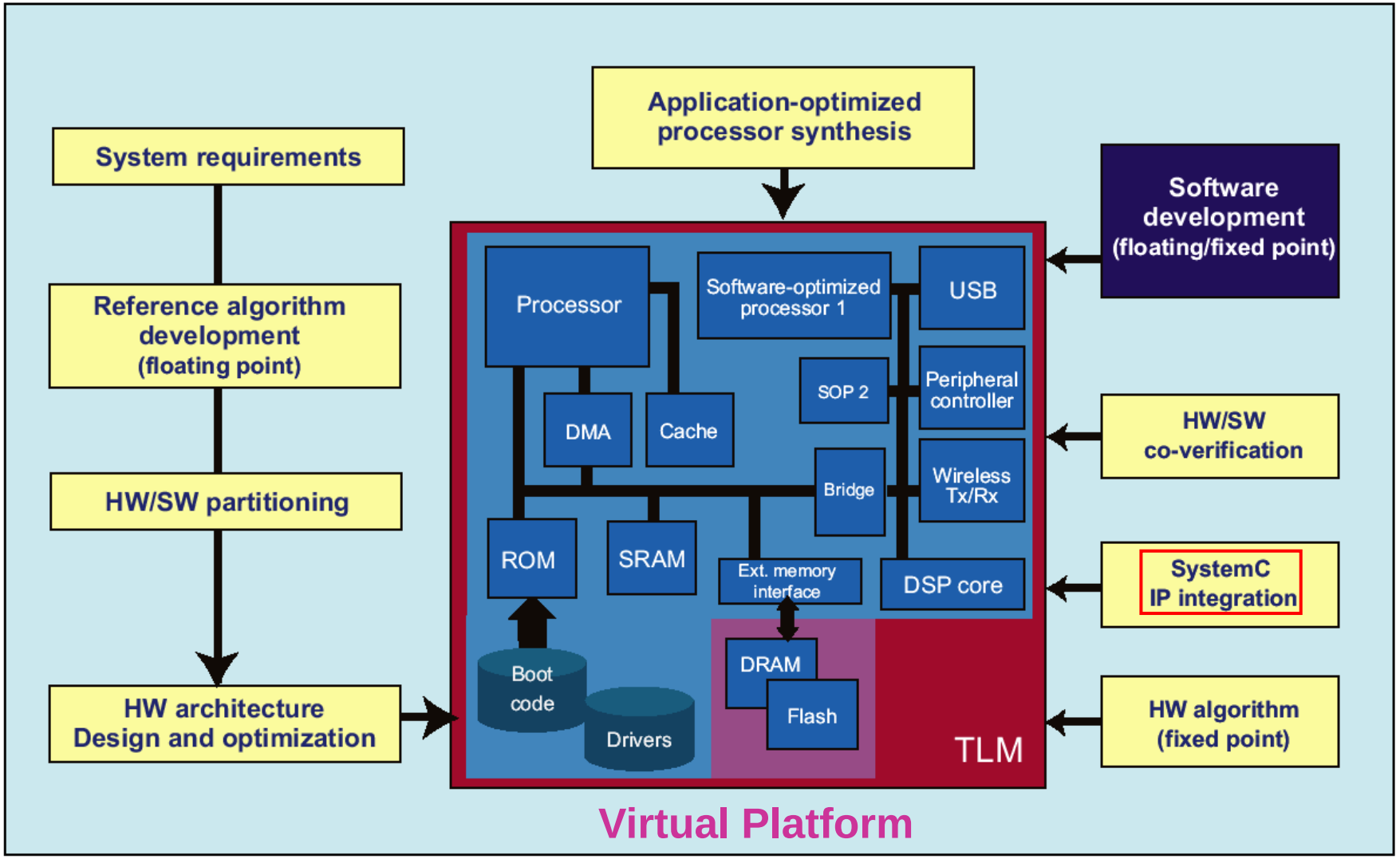
SystemC Background

- C++
  - 物件導向程式語言 (Object-Oriented)
    - 類別(Class)
    - 純虛擬函式(pure-virtual-function)
- HW
  - 元件(Component)
    - 模組(Module)
    - 介面(Interface)

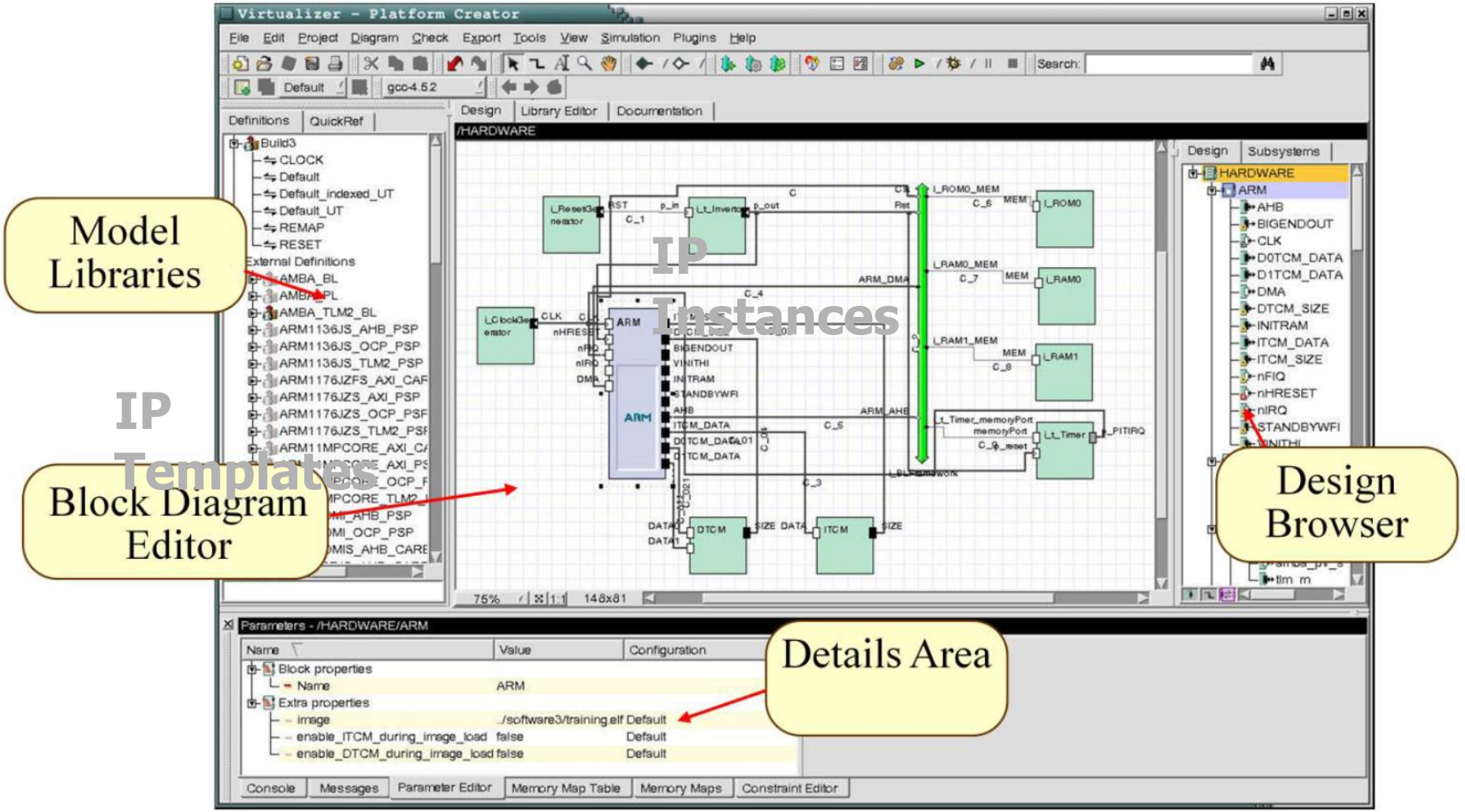
Virtual Platform

- Virtual Platform
  - A SW model of a HW SoC platform
- Enables...
  - HW platform architecture exploration and optimization
  - SW development, debugging, and optimization
  - Concurrent HW/SW design (HW/SW codesign)
- Requirements
  - High simulation speed
  - Speed/accuracy trade-off
  - Flexibility
  - Usability for non-HW-experts

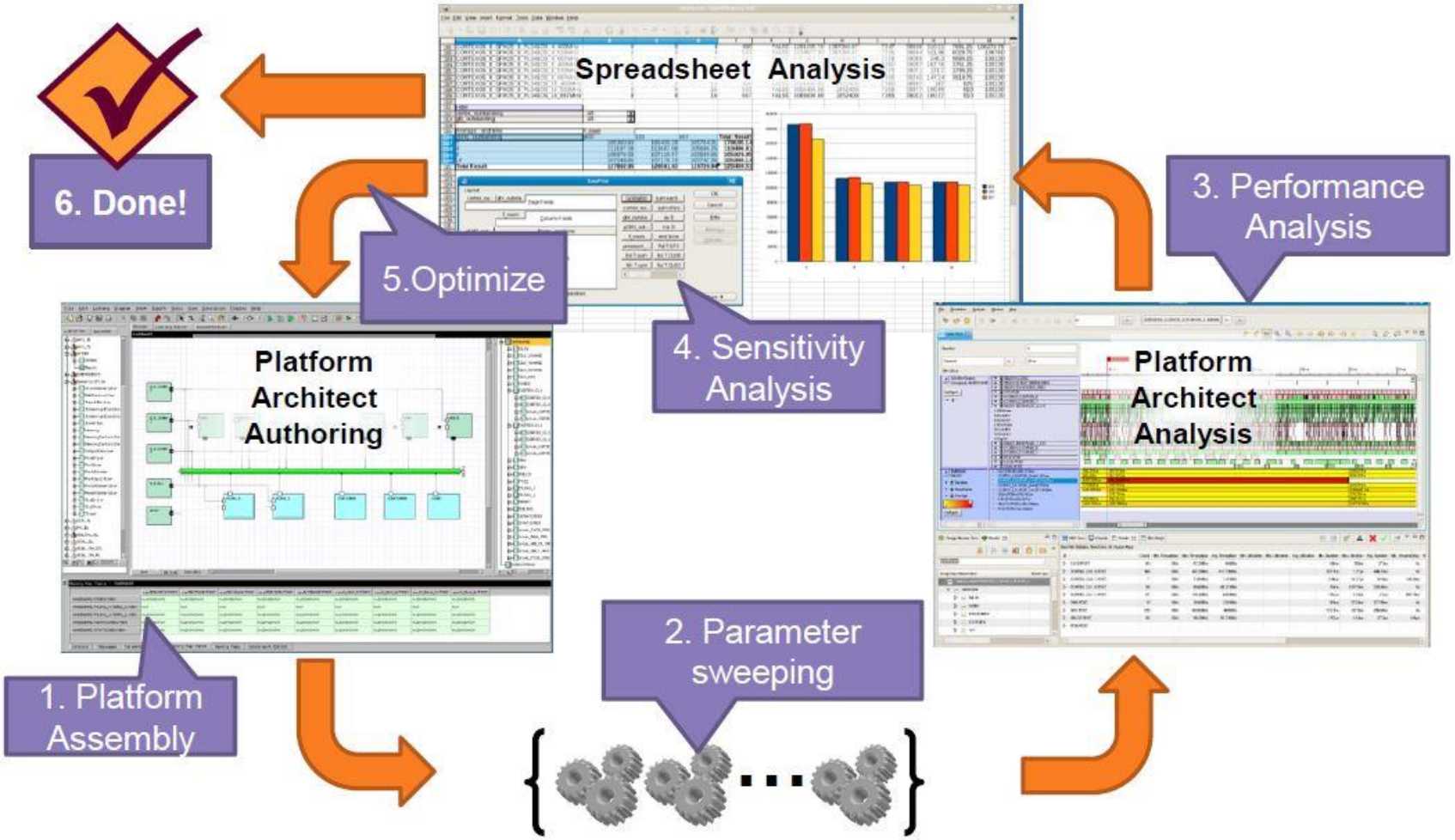
Transaction-Level Modeling



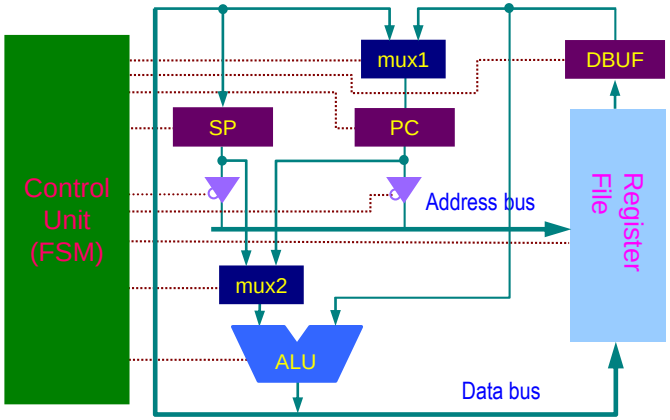
Synopsys Platform Architect

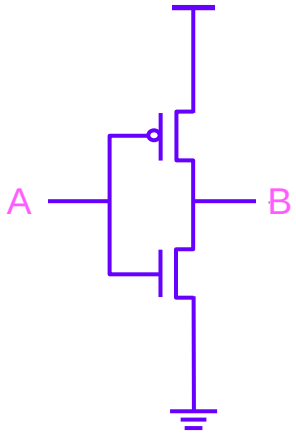


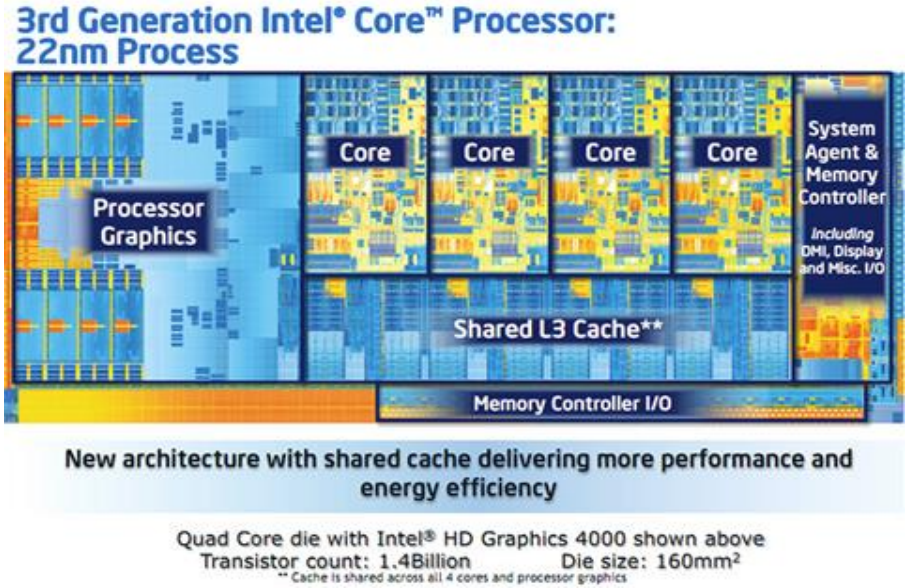
Iterative Architecture Optimization

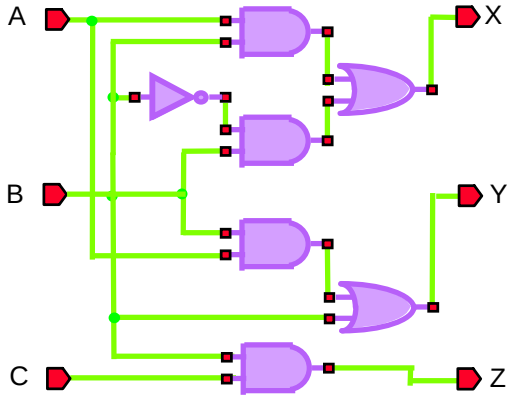


Sources of Power Dissipation (1/2)



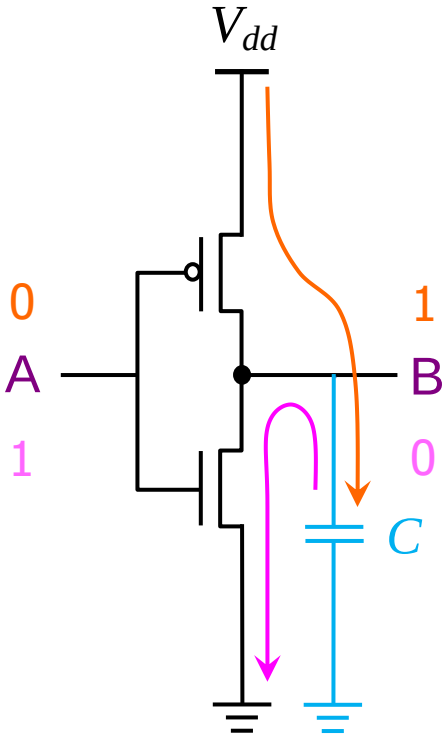


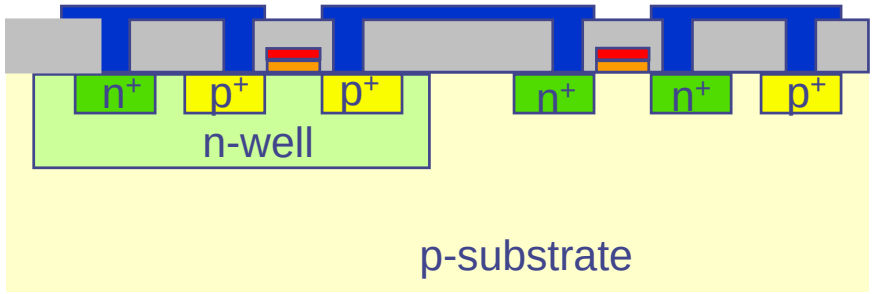




Sources of Power Dissipation (2/2)

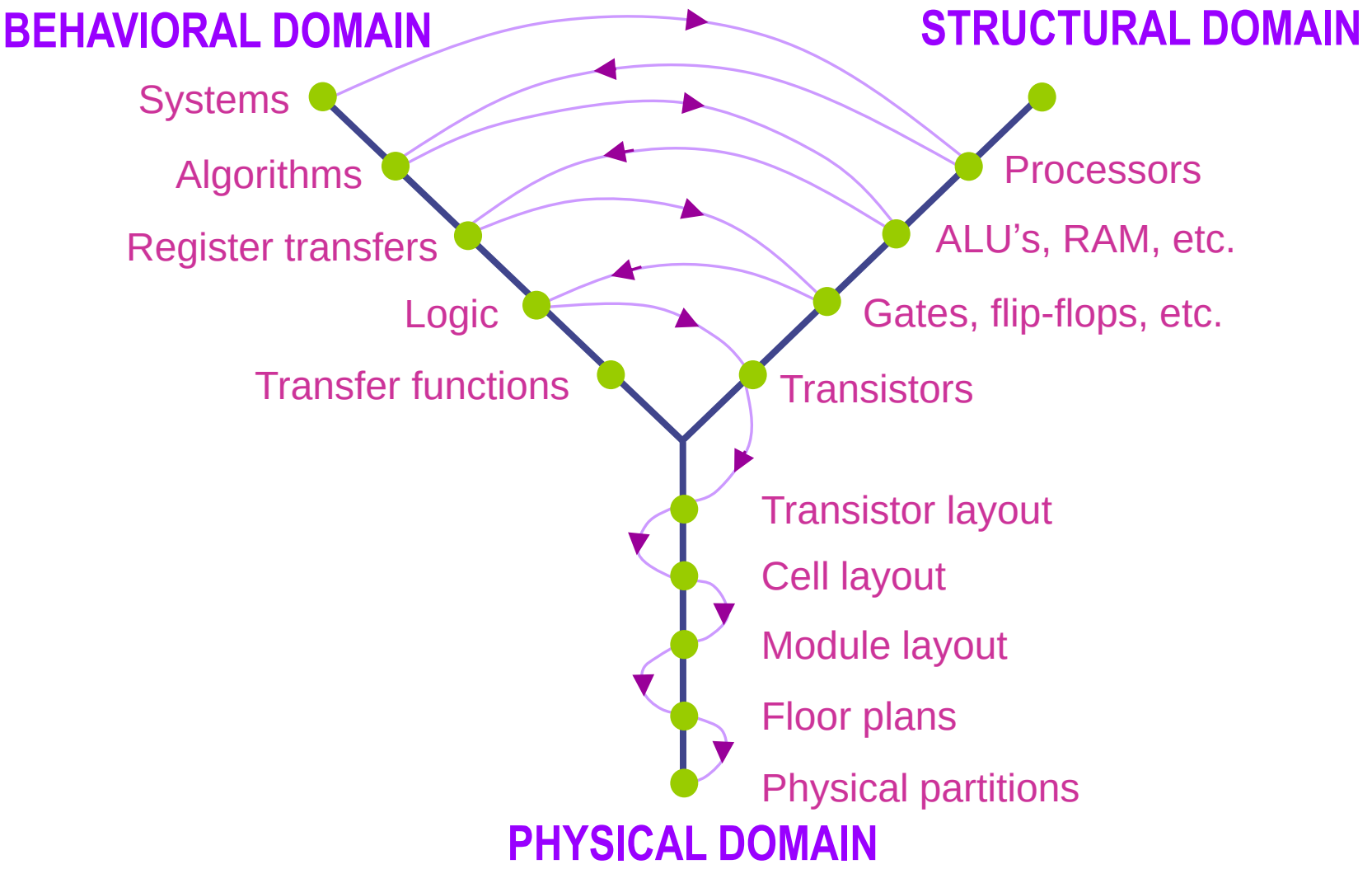
- Dynamic power, Short circuit power, Leakage power





$$P = C \cdot V_{dd}^2 \cdot f + V_{dd} \cdot I_{leak}$$

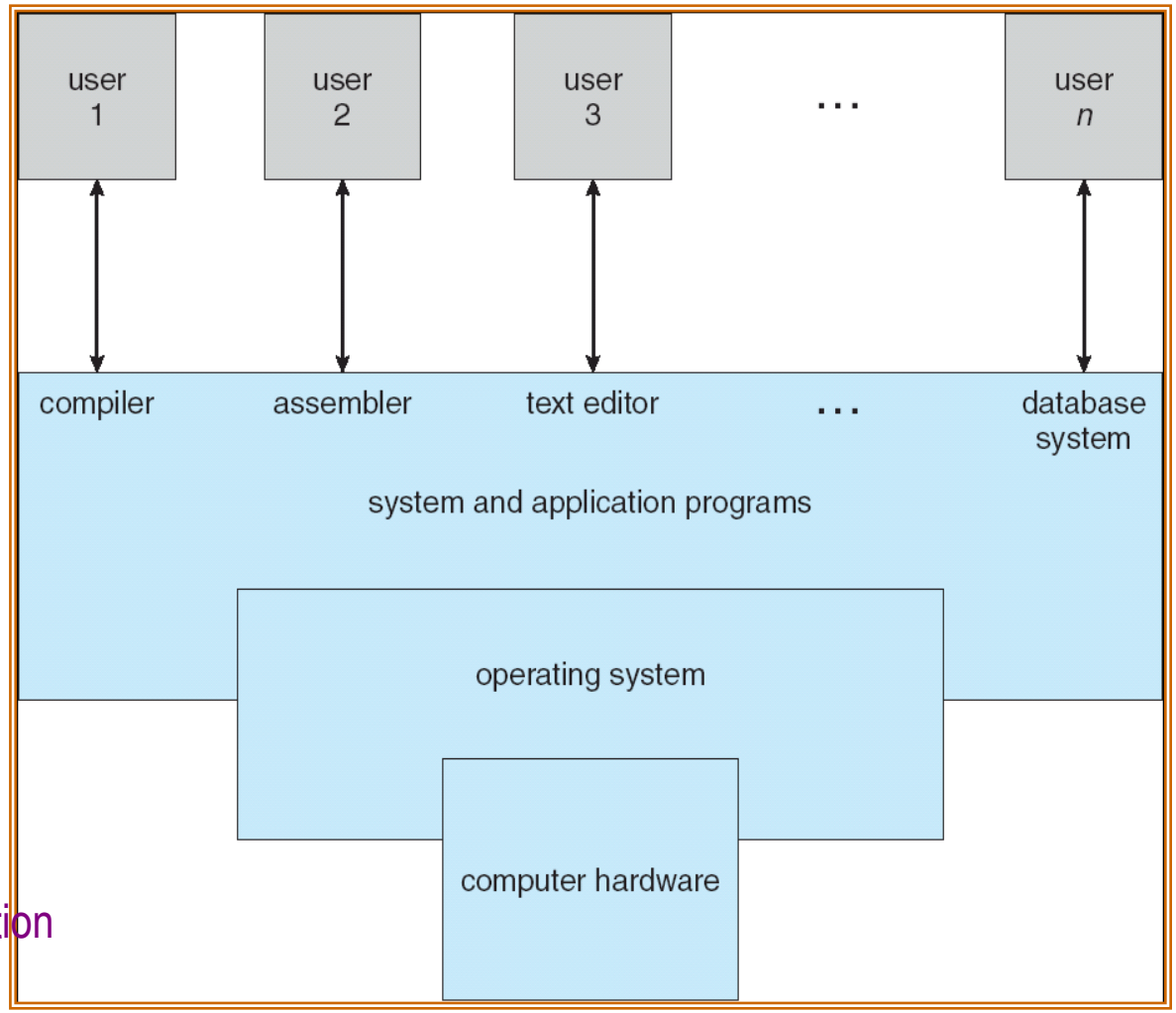
Design Flow


BEHAVIORAL DOMAIN
STRUCTURAL DOMAIN
Systems
Algorithms
Register transfers
Logic
Transfer functions
Processors
ALU's, RAM, etc.
Gates, flip-flops, etc.
Transistors
Transistor layout
Cell layout
Module layout
Floor plans
Physical partitions
PHYSICAL DOMAIN

Levels for Low Power Design

Abstraction Level	Available Techniques	Expected Saving
System	Design partitioning, Power management	>75%
Algorithm	Complexity, Concurrency, Locality, Regularity, Data representation	50 ~ 75%
Architecture	Voltage scaling, Parallelism, Instruction set, Signal correlations	15 ~ 50%
Circuit/Logic	Transistor sizing, Logic optimization, Activity driven power down, Low-swing logic	5 - 15%
Technology	Threshold Reduction, Multithresholds	3 ~ 5%

Levels for Low-Power Processor Design (1/2)



Application

Compiler

OS

Architecture

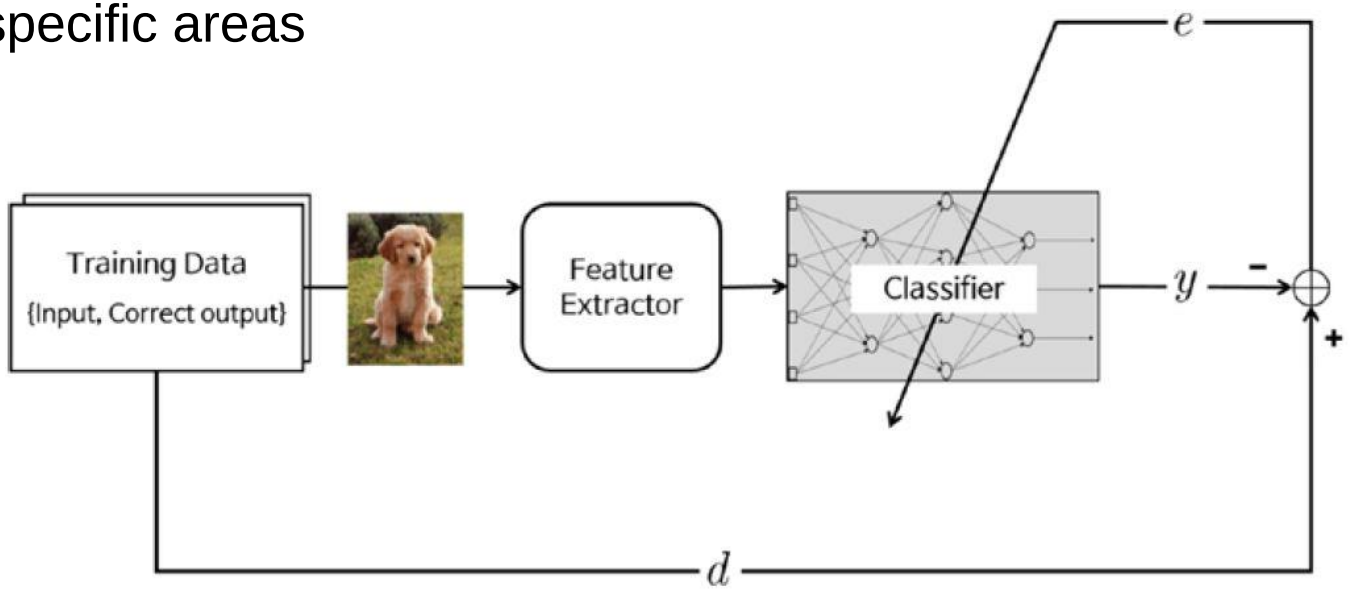
Implementation

Levels for Low-Power Processor Design (2/2)

- Hierarchies of power-aware designs for processors
  - Application level
    - Selection of an algorithm, Choice of the physical parameters, Number Representations, …
  - Compiler/Assembler level
    - Reduce switching activity, Reduce External Memory Accesses, …
  - OS level
    - Dynamic Power Management
  - Architectural level
    - CPU core, Input/Output
  - Implementation level
    - Organization, Logic and Circuit, Device Technology

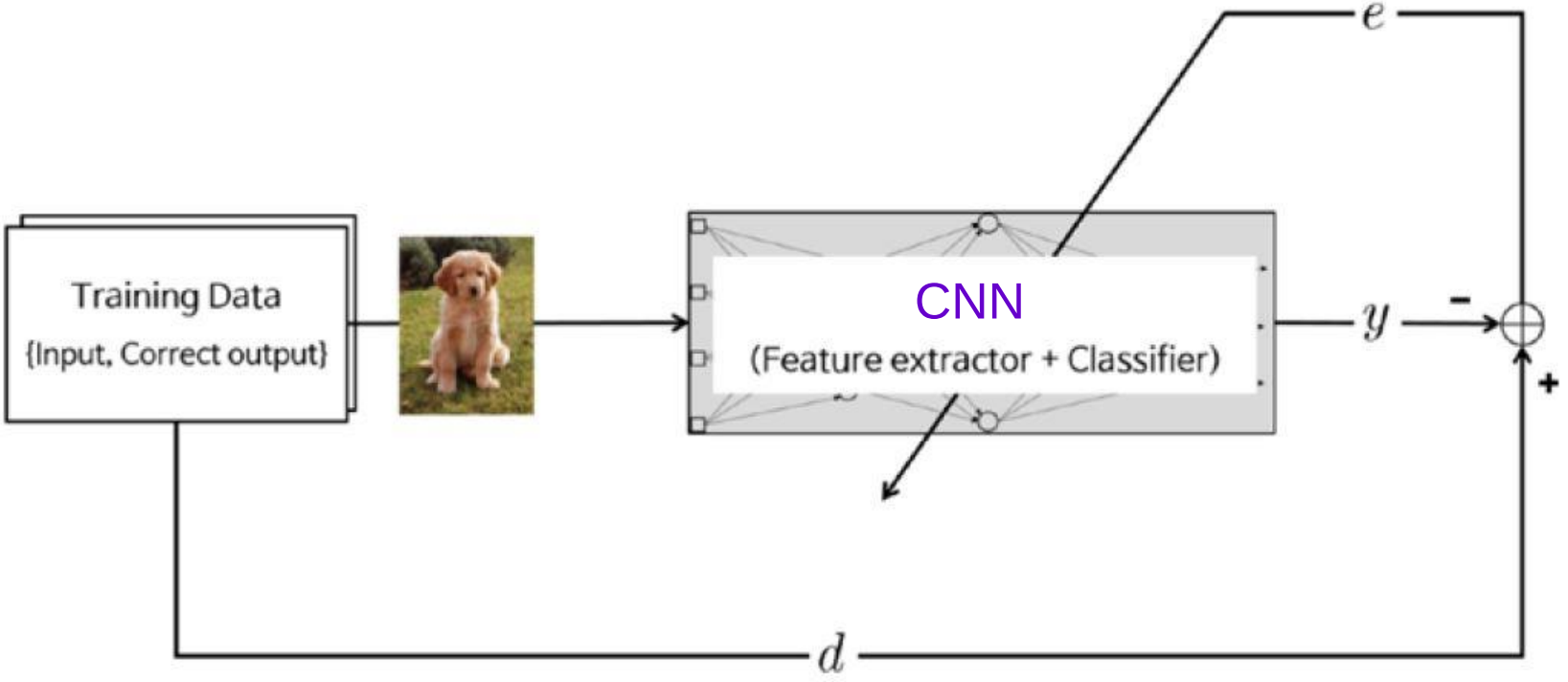
Convolutional Neural Network (1/3)

- Convolutional Neural Network (CNN)
  - is a deep neural network specialized for image recognition
- Basically, image recognition is the classification
  - the images should be processed to contrast the features
  - various techniques for image feature extraction have been developed
  - before CNN, the feature extractor has been designed by experts of specific areas



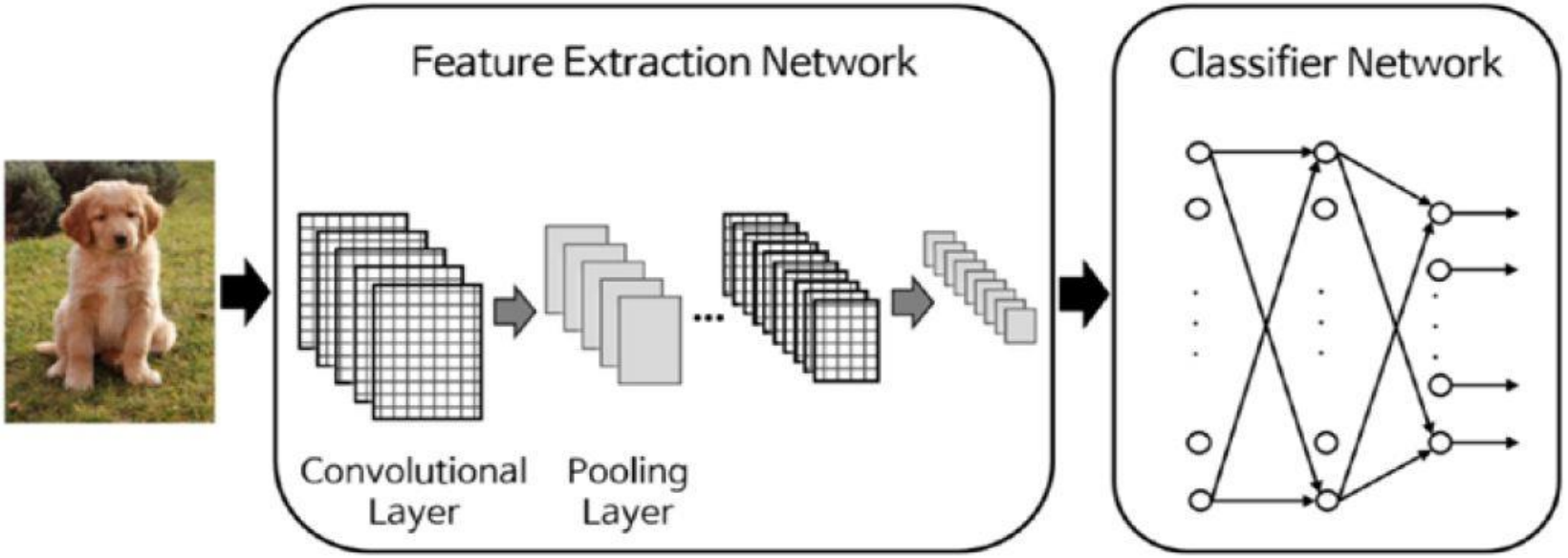
Convolutional Neural Network (2/3)

- CNN
  - includes the feature extractor in the training process rather than designing it manually
  - The feature extractor of CNN is composed of special kinds of neural networks, of which the weights are determined via the training process



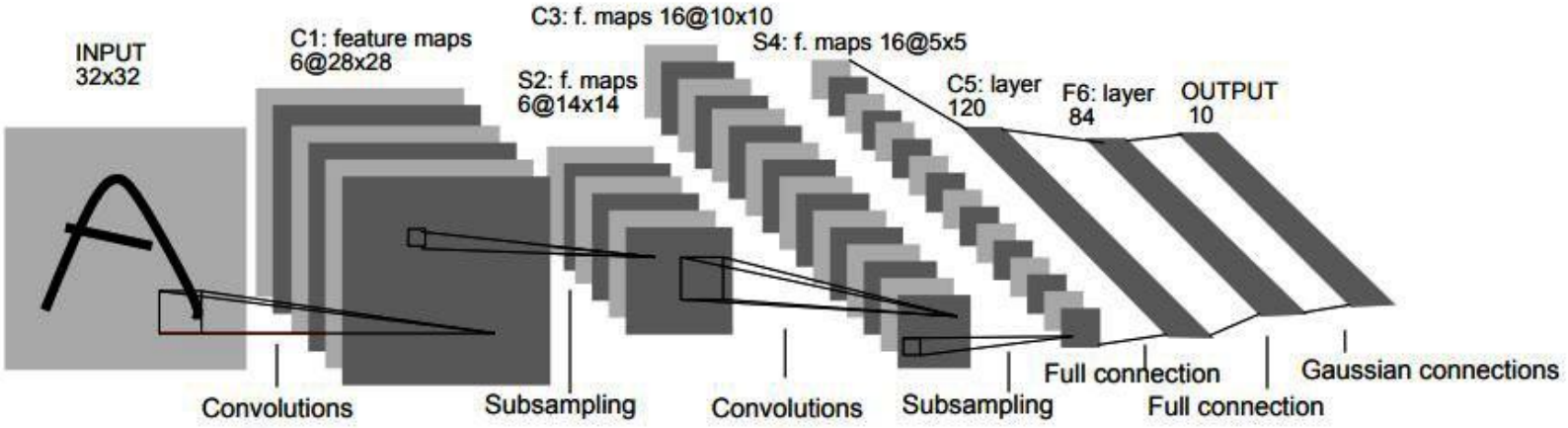
Convolutional Neural Network (3/3)

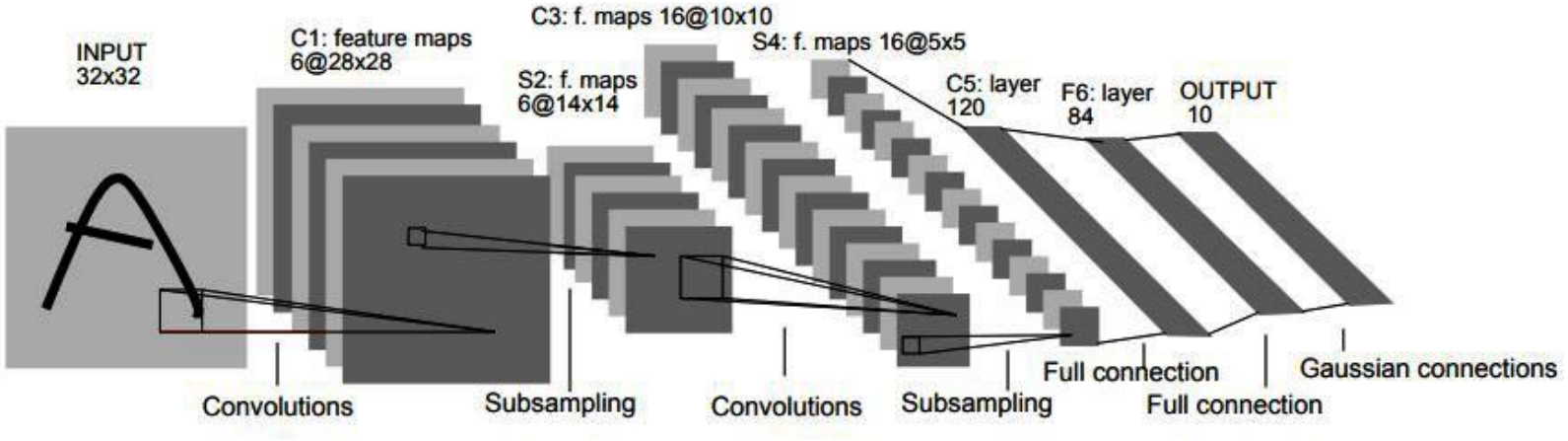
- Typical architecture of CNN
  - The convolution layer converts the image using the convolution operation, it can be thought of as a collection of digital filters
  - The pooling layer combines the neighboring pixels into a single pixel, thus reduces the dimension of the image
  - The classification network usually employs the ordinary multiclass classification neural network



Classic CNN Architectures - LeNet-5

- LeNet-5 (1998)
  - handwritten and machine-printed character recognition
  - activation function: tanh, sigmoid





Layer		Feature Map	Size	Kernel Size	Stride
Input	Image	1	32x32	-	-
1	Convolution	6	28x28	5x5	1
2	Ave. Pooling	6	14x14	2x2	2
3	Convolution	16	10x10	5x5	1
4	Ave. Pooling	16	5x5	2x2	2
5	Convolution	120	1x1	5x5	1
6	FC	-	84	-	-
Output	FC	-	10	-	-

課程教學大綱 (1/2)

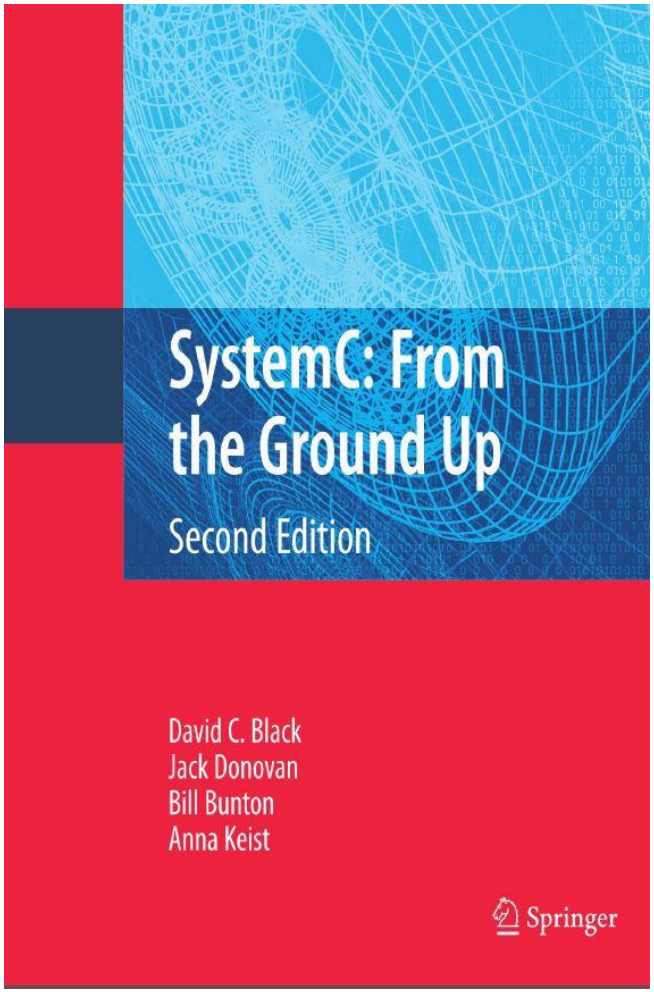
- 討論時間
  - 週(三) (10:00～12:00)、週(四) (10:00～12:00)
  - 地點: 工EC3008
- **課程大綱**
  - Introduction to System Level Design
  - Introduction to High-Level Synthesis
  - System Level Modeling Language
  - Transaction Level Modeling
  - Integrated System Level Modeling
  - Low-power System Design
  - Convolutional Neural Networks
  - Case Study

課程教學大綱 (2/2)

- **課程目標**
  - 本課程介紹系統層次設計與驗證的概念及方法，包含系統層次模型建立、系統層次模擬與效能評估、系統單晶片軟/硬體評估與分割、軟硬體快速雛形驗證等，以建立學生SOC系統架構設計與驗證的概念及實務經驗。
- **授課方式**
  - Lecture and Practice
- **評分標準**
  - Examination (40%)
  - Labs & Report (60%)
- **參考書＼教科書**
  - 教育部相關課程教材及自編講義

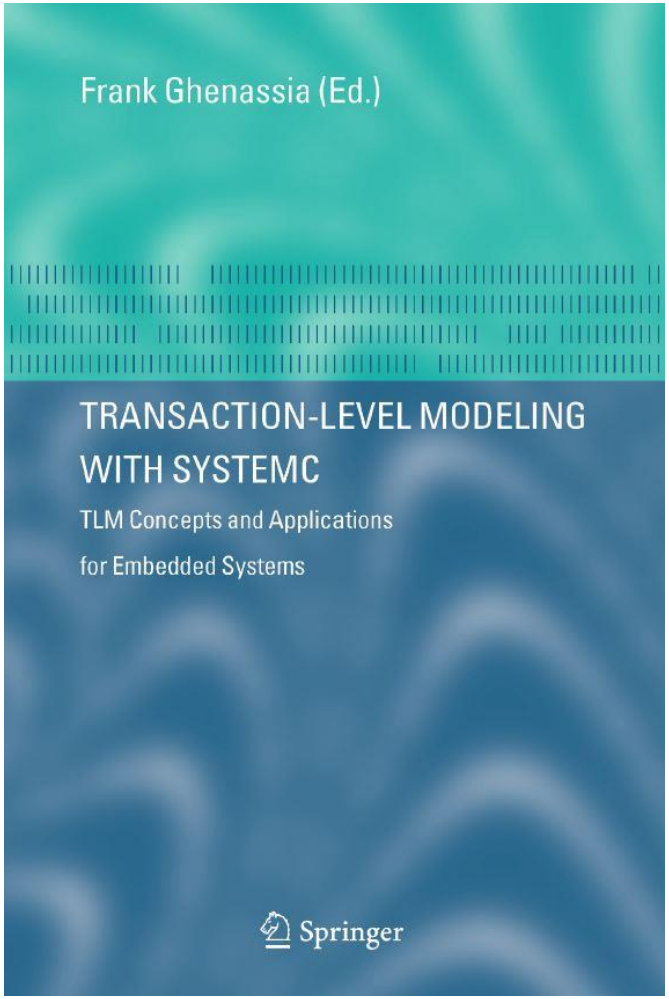
參考書＼教科書

- **SystemC: From the Ground Up, Second Edition**
  - David C. Black, Jack Donovan, Bill Bunton, and Anna Keist
  - Springer, 2010



參考書＼教科書

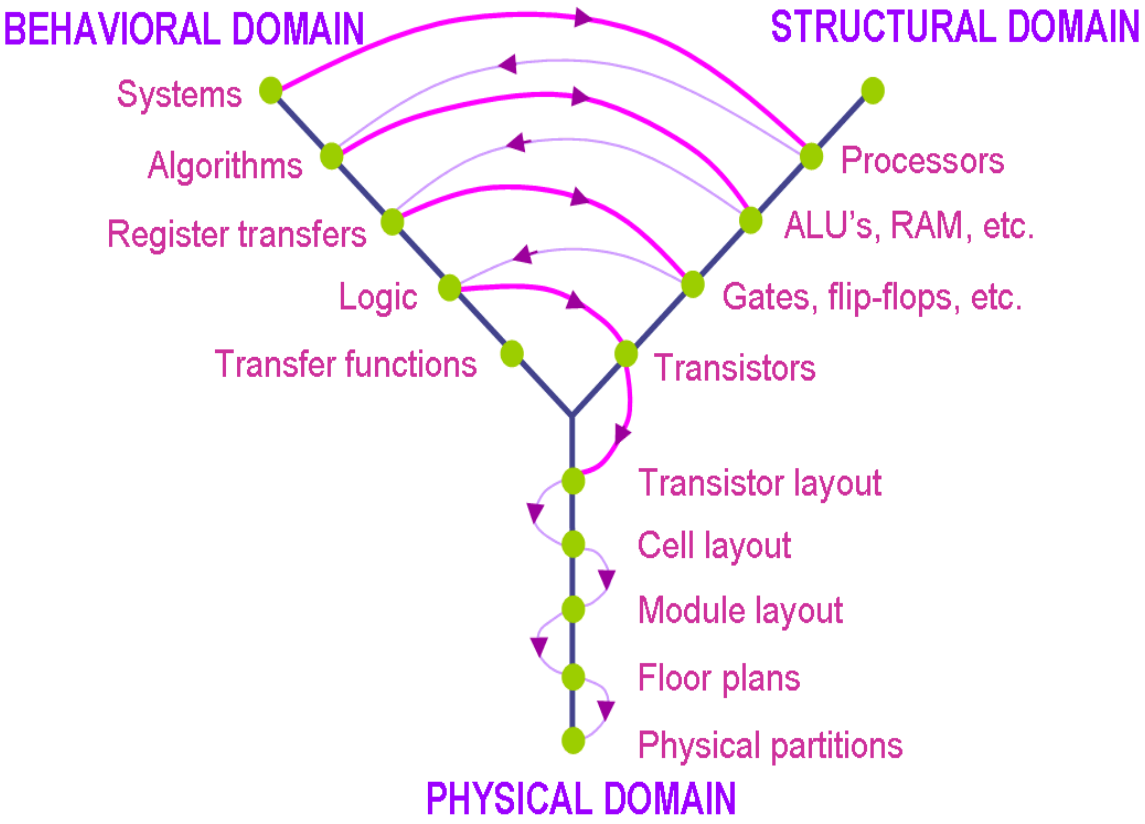
- **Transaction-Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems**
  - Frank Ghenassia, Springer, 2005



Week	Date	Course
第一週	02/14	Introduction to System Level Design
第二週	02/21	Introduction to System Level Design
第三週	02/28	Introduction to System Level Design
第四週	03/07	Introduction to High-Level Synthesis
第五週	03/14	Case Study
第六週	03/21	System Level Modeling Language
第七週	03/28	System Level Modeling Language
第八週	04/04	Case Study
第九週	04/11	Transaction Level Modeling
第十週	04/18	Transaction Level Modeling
第十一週	04/25	Midterm
第十二週	05/02	Integrated System Level Modeling
第十三週	05/09	Low-power System Design
第十四週	05/16	Convolutional Neural Networks
第十五週	05/23	Case Study
第十六週	05/30	Term Project
第十七週	06/06	彈性學習
第十八週	06/13	彈性學習

Background Knowledge

- C/C++ (for System C)
- Verilog
- VLSI design flow (RTL to Layout)
- FPGA



系所學生專業能力

- 1.具有資訊科學及工程領域之專業知識的能力
- 2.具有策劃、設計及執行資訊科學及工程軟硬體系統專案研究的能力
- 3.具有撰寫資訊科學及工程領域專業論文及專業的能力
- 4.具有創新思考及獨立解決資訊科學及工程問題的能力
- 5.具有與資訊科學及工程領域內外的專業人員溝通、協調與團隊合作的能力
- 6.具有良好的國際觀，了解國際上資訊科學及工程相關學術研究與產業發展
- 7.具有領導、管理及規劃資訊科學及工程專案計畫的能力及專業倫理素養
- 8.具有終身自我學習成長的能力

全校學生基本素養與核心能力

- 1.表達與溝通能力。
- 2.探究與批判思考能力。
- 3.終身學習能力。
- 4.倫理與社會責任。
- 5.美感品味。
- 6.創造力。
- 7.全球視野。
- 8.合作與領導能力。
- 9.山海胸襟與自然情懷。

Teaching Assistant

- 王昱翔、王裕祥
  - Tel: 5254340
  - E-mail
    - m103040052@student.nsysu.edu.tw
    - wangyuxiung@gmail.com
  - 超大型積體電路設計與自動化實驗室 (EC5024A)
- 課程網頁
  - 中山網路大學
  - https://cu.nsysu.edu.tw

實驗報告+程式碼

- 實驗報告及程式碼以壓縮檔繳交，每位同學均須繳交
- 實驗報告壓縮檔請以實驗編號及自己的學號姓名命名，例如：Lab1_M999999999陳小華.rar，於規定時間內上傳至“中山大學網路大學-作業評量區”繳交
- 實驗報告內容包含
  - 實驗主題、實驗日期、學號姓名
  - 實驗內容、過程及結果
    - 實驗內容
    - 實驗畫面、電路圖、程式碼...
    - 實驗結果及分析
  - 實驗心得