

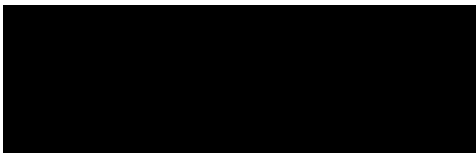
The Tides of EDA

Based on the keynote speech at the 40th Design Automation Conference
by Alberto Sangiovanni-Vincenteli

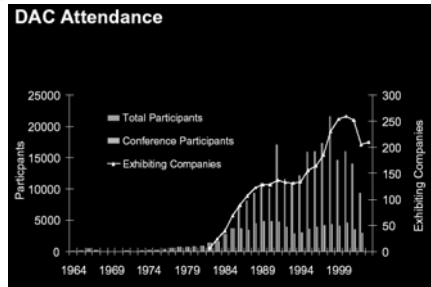
Ages of Mankind



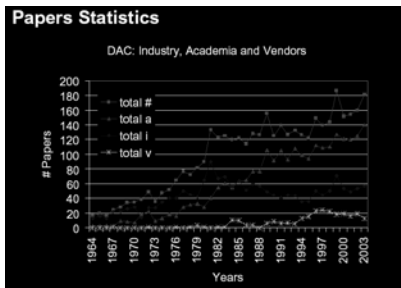
Ages of EDA



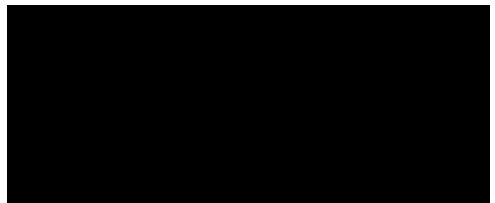
DAC Attendance 1964-2003



DAC Paper Statistics



Three Phases of EDA History



Age of Gods (1964-1978)



7

Age of Gods – Business Side

The Age of the Gods (1964-1978): Business

- CAD of Strategic Value for System industry
- Internal CAD groups very strong
- Strong Japanese presence (NEC, NTT, Hitachi, Fujitsu, etc.)
- First Generation CAD Vendors:
 - Applicon (1969), Calma (1970), Computer Vision (1972): artwork editing and custom hardware workstations



8

First Generation of EDA Companies

Life Cycles of First Generation CAD Companies

- They all disappeared
- Difficult to stay abreast of technology due to software complexity and to hardware evolution
- Little customer loyalty due to "low-level" technology
- Incapacity to read the market and the evolution of new technologies



9

Age of Heroes

The Age of the Heroes (1978-1992)

- Vibrant period for creativity and expansion both in research outcome and industrial development
- Key results in all areas of EDA:
 - Verification and Testing
 - Layout and Data bases
 - Physical Verification
 - Logic Synthesis
 - HDL
 - HW Accelerators
 - FPGA
 - High-level Synthesis and System Level Design



- Largest Vendors today founded in this period

10

Age of Heroes - Contributions

The Age of the Heroes (1979-1992)



11

Verification & Testing

The Age of the Heroes (1979-1992)

- Relaxation-based techniques (Berkeley, Newton and ASV, 1980)
- MOSSIM (MIT, Bryant 1980) Penfield-Rubinstein Model for delays, 1981,
- AWE (CMU, Pileggi and Rohrer, 1988)
- BDDs (CMU, Bryant, 1986)
- Formal Verification: Equivalence and Model Checking (Bull, Coudert and Madre, CMU, Bryant, Clark, Dill, McMillan, and others, 1992)
- PODEM-X (IBM: Goel, 1981)



12

Layout

The Age of the Heroes (1979-1992)

- MAGIC, Squid, OCT, VEM (Berkeley: Ousterhout, Newton)
- Caltech and Bell Labs Silicon Compilation and Symbolic Layout (Mead, Johansen, Buric et al.)
- Sophisticated Place and Route
 - Simulated Annealing: IBM (Kirkpatrick) 1981, Timber/Wolf (Berkeley: Sechen) 1984, Tangant (Fujimura, Nequist, Feig)
 - Quadratic Placement and BBL (Berkeley, Kuh)
 - Channel Routing and River Routing (MIT, Rivest and Pinter)
 - Partitioning (Breuer, Fiduccia and Mattheyses, Lauther) Quadra-section (Kedem)
 - Estimation and Complexity (IBM, Donath and Heller, El Gamal)
 - Floor-planning (Heller, Lauther, Uu, Otten and others)

Layout

13

Logic Synthesis

The Age of the Heroes (1979-1992)

- Logic Synthesis Systems (IBM, Darringer, Joyner, Trevillian, 1979)
- Espresso, YSC, MIS, SIS, HSIS (IBM and Berkeley, Brayton, Hachtel, ASV, Rudell and many others, 1980-1989)
- DAGON (Bell Labs, Keutzer)
- SOCRATES (GE, De Geus and colleagues)
- Fujitsu (Fujita), NTT, NEC
- Redundancy, Testing and Delay (Keutzer, Devadas, McGeer, Brayton, Malik, Saldanha, Mercer, V. Agrawal, Abraham, Cheng and many others)
- Logic Partitioning (Corola, Brglez and others)
- Use of BDDs (Bull, Coudert and Madre, J. Rajski)

Logic Synthesis

14

Hardware Description Language

The Age of the Heroes (1979-1992)

- Hardware Description Languages became a major EDA battle field:
 - Verilog (Thomas, Moorby, Goel)
 - VHDL
- Standards were much debated at DAC in this period
- Synthesizable subsets for synthesis

HDL

A perspective: "...Adoption of VHDL was one of the biggest mistakes in the history of design automation, causing users and EDA vendors to waste hundreds of millions of dollars..."

15

Acceleration

The Age of the Heroes (1979-1992)

- Yorktown Simulation Engine (YSE) (IBM: Pfister and others)
- Wiring machine (IBM, Nair and others)
- Use of parallel processors for EDA (Thinking Machines, Intel, N-cube)
- Rapid Prototyping and Emulation with FPGAs

Acceleration

16

High-Level Design

The Age of the Heroes (1979-1992)

- High-Level Synthesis: CMU (Thomas), USC (Parker), UIUC (Gajski)
- Cathedral, IMEC (De Man, Rabaey)
- SpecChart and SpecC(Gajski)
- Mimola (Marwedel)
- Ptolemy (Lee), Flex (P. Paulin)
- POLIS (Berkeley)
 - Cosyma (Ernst)
 - Synchronous Languages (Estrel, Berry, Lustre, Halwachs, Benveniste)

High-Level Design

17

Age of Heroes – Business Side

- Workstation Companies (hardware and software) (second generation):
 - Daisy (1980) Mentor (1981), Valid(1981) with schematic data capture and logic simulation
- Pure Software companies (third generation):
 - ECAD (1982) - SDA (1983) (later merged into Cadence (1987)), Silicon Compilers (1983), Silicon Design labs, (1983), ViewLogic (1984), Gateway (1985), OSI (1987)
 - Quiz #1: What is the name of Optimal Solutions Inc. today?
- Acceleration Companies:
 - Quickturn (1989), PIE Design (1991), IKOS
- Strong investment in internal EDA by IC companies, e.g., Intel, Motorola, TI, Infineon, Philips, ST, NEC, Hitachi, Toshiba, Fujitsu
- IBM and Bell Labs technical leadership
- Formation of strong groups at Universities

18

Second Generation of EDA Companies

Company Life Cycle

- Second Generation business model proven not sustainable due to dominance of general purpose work-stations
- Daisy, Valid disappeared: technology content problems
- Silicon Compilers and Silicon Design Labs also disappeared: Silicon compilation and procedural layout languages did not appeal hard-core IC designers



19

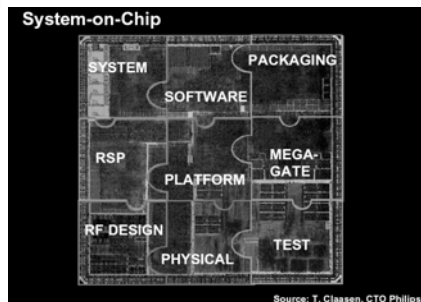
Age of Men (1993-today)



- The emerging Web attracted best energies, minds and capital funding.
- Technical innovation of EDA began slowing down.
- The industry became more mature and less risky.

20

System on Chip (SoC) Became a Reality

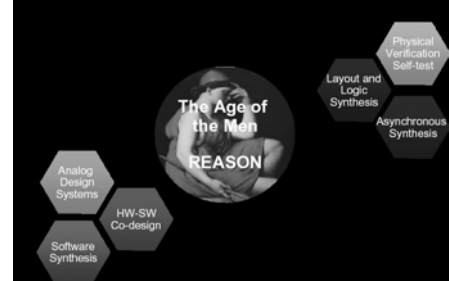


Source: T. Claassen, CTO Philips

21

Age of Men - Contributions

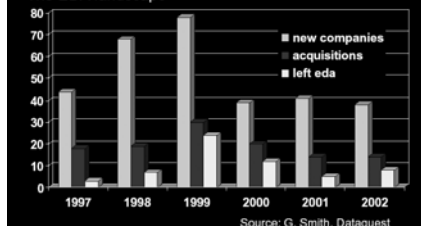
The Age of the Men (1993-2001)



22

Age of Men - Business

- The competition for human resources – WEB-mania
- The EDA landscape

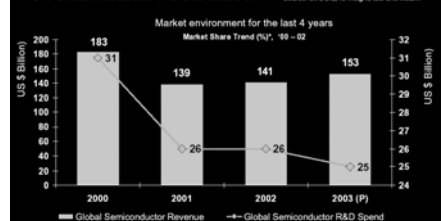


Source: G. Smith, Dataquest

23

The Future – what age?

Overall Market Condition

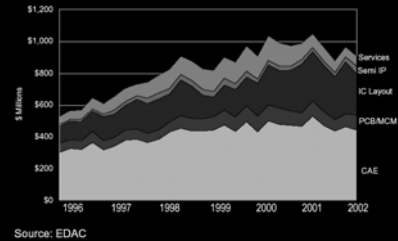


Staying alive and prospering is a major challenge.

24

Current Market Condition

3% EDA Product Decline in 2002
(Revenues are per quarter)



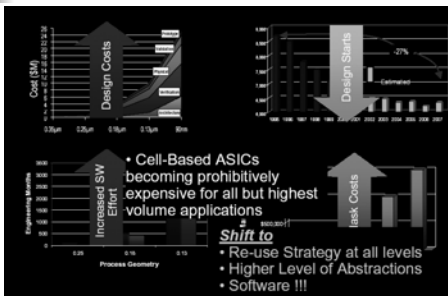
25

Killer Applications



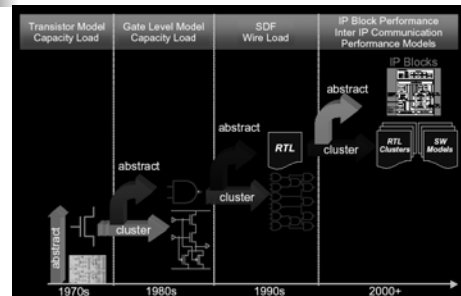
26

Challenges and Trends



27

The Quest for Next Level of Abstraction



28

Complexity, Quality, & Time To Market Today

	PWT UNIT	BODY GATEWAY	INSTRUMENT CLUSTER	TELEMATIC UNIT
Memory	256 Kb	128 Kb	184 Kb	8 Mb
Lines Of Code	50,000	30,000	45,000	300,000
Productivity	6 Lines/Day	10 Lines/Day	6 Lines/Day	10 Lines/Day*
Residual Defect Rate @ End Of Dev	3000 Ppm	2500 ppm	2000ppm	1000 ppm
Changing Rate	3 Years	2 Years	1 Year	< 1 Year
Dev. Effort	40 Man-yr	12 Man-yr	30 Man-yr	200 Man-yr
Validation Time	5 Months	1 Month	2 Months	2 Months
Time To Market	24 Months	18 Months	12 Months	< 12 Months

* C² CODE

FABIO ROMEO, Magneti-Marelli
DAC, Las Vegas, June 20th, 2001

29

Electronic Design: A Vision

- Embedded Software will be increasingly critical in the Electronic Industry
- Embedded Software Design must not be seen as a problem in isolation, it is an, albeit essential, aspect of *EMBEDDED SYSTEM DESIGN*
- The vision is to change radically the way in which ESW is developed today by linking it:
 - Upwards in the abstraction layers to system functionality
 - Downwards in the programmable platforms that support it thus providing the means to verify whether the constraints posed on Embedded Systems are met.

30

System Level Design: Orthogonalization of Concerns and Platform-Based Design

K. Keutzer, S. Malik, R. Newton, J. Rabaey
and A. Sangiovanni-Vincentelli

31

Introduction

- Embedded system market: consumer electronics, industrial automation, retail automation, and medical market
- System design challenge: dramatic expansion of the diversity
- Central importance in such applications: data movement and transformation
- Overall design goal:

balance production costs with development time and cost in view of performance functionality considerations

32

New Approaches are Needed

- Development efforts increase dramatically
- A strict design time budget has to be kept
- A new design methodology must start at high levels of abstraction
- Reuse and early error detection become necessary

33

Features Preferred Approaches Should Have

- Design time and cost dominate the decision-making process for system designers
- Designs must be captured at the highest level of abstraction
- Concurrent systems are most important
- Concurrency implies communication among components of the design
- A few highly complex part-types + many more medium-complexity chips
- Platform based and highly programmable platforms

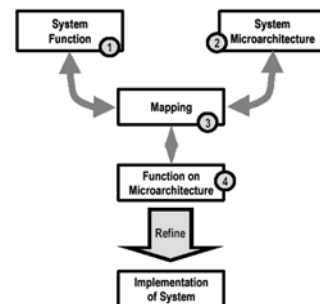
34

System Design Methodology

- Orthogonalization of concerns
 - Separation of various aspects of design to allow more effective design exploration
- Separation between
 - Function and architecture
 - Communication and computation

35

System Design Methodology – Overall View



36

Function and Communication-based Design

- Function: abstract view of the behavior of an aspect of the system (I/O characterization)
- Function design: there are design decisions involved for functions
- Use formal models and transformations to enable verification and synthesis
- Communication based design: communication is specified independently of the modules that compose the design
- Isolation of communication and computation

37

Micro-architecture

- Micro-architecture defines how the functionality is actually realized (different from "architecture")
- Micro-architecture: a set of interconnected components
- Micro-architecture components include microprocessors, peripherals, dedicated logic blocks and memories
- Communication among micro-architecture blocks

38

Mapping

- Functions are assigned (mapped) to the components of the micro-architecture
- Mapping process determines the performance and cost of the design
- Mapping allows moving down the levels of the design flow

39

Link to Implementation

- Implementation of the components of the micro-architecture
- Hardware blocks may be further decomposed into sub-blocks
- Hardware blocks are from either existing libraries or custom design

40

Platform-based Design

- Reuse of software requires the basic micro-architecture of the implementation is essentially "fixed"
- The "basic" micro-architecture consists of programmable cores, I/O subsystems and memories
- Hardware platform – micro-architectures allow reuse of software
- Software platform – a software layer that abstracts the hardware platform and provides a set of APIs
- System platform = Hardware platform + software platform

41

Hardware Platforms

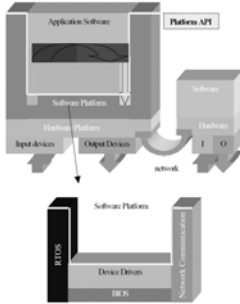
- Constraints that determine the hardware platform are performance and "size"
- Over-design offers more flexibility to reduce design costs and time-to-market
- The design of a hardware platform is the result of a trade-off in a complex space:
 - The flexibility of the hardware platform (size of application space)
 - The degree of freedom in designing hardware platform instances (size of micro-architecture space)

42

Software Platforms

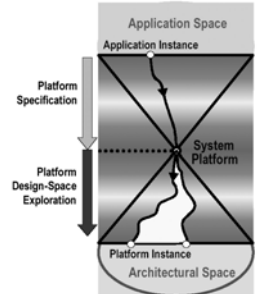
The software layer wraps the essential parts of the hardware platform:

- The programmable cores and the memory subsystem via RTOS
- The I/O subsystem via the device drivers, and
- The network connection via the network communication subsystems



System Platforms

- Mapping an application to a system platform
- Meet-in-the-middle approach for platform-based design



44

Case Studies based on the Methodology

- Philips VideoTop
- Magneti-Marelli Automotive Engine Control
- The Design of Wireless Systems

45

Design Tools based on the Methodology

- MESCAL (Modern Embedded Systems, Compilers, Architectures and Languages) project
 - Methodologies, tools and algorithms to support the efficient development of fully programmable, platform-based designs for specific application domains
- Metropolis project
 - Integrated design environment for heterogeneous embedded systems

46

Summary

- The dramatic increase of design complexity demands new design methodologies
- System level design separates the various aspects of design
 - Function and architecture
 - Communication and computation
- Each level of abstraction is nearly independent
- Advantages
 - Enable reuse of design components (HW & SW)
 - Reduce overall complexity (divide and conquer)

47