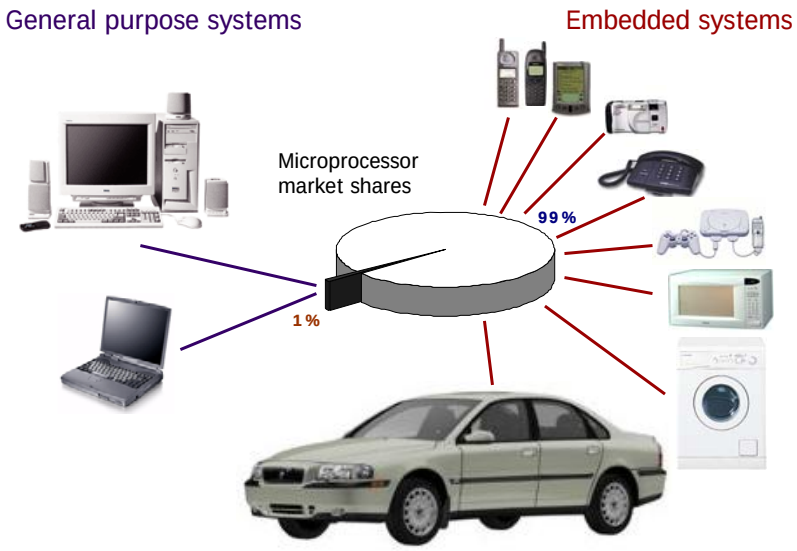
Luca Benini – DEIS Università di Bologna
lbenini@deis.unibo.it

# Part I - SoC Technology

- Silicon technology trends and challenges
- Application drivers
- Architecture evolution



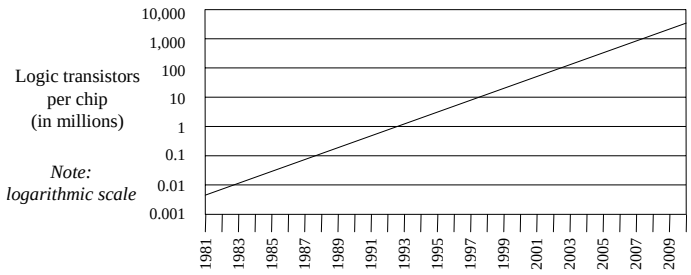
## Embedded Systems



# Embedded Hardware: Moore's law

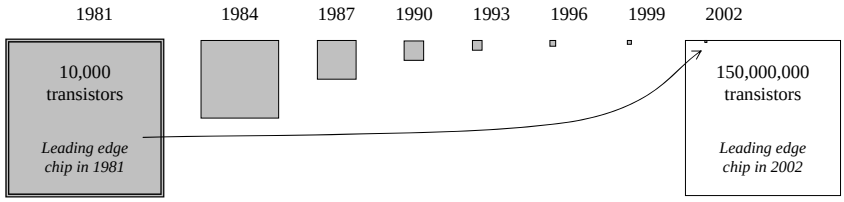
Formulated in 1965 by Intel co-founder G.Moore

***IC transistor capacity has doubled roughly every 18 months for the past several decades***



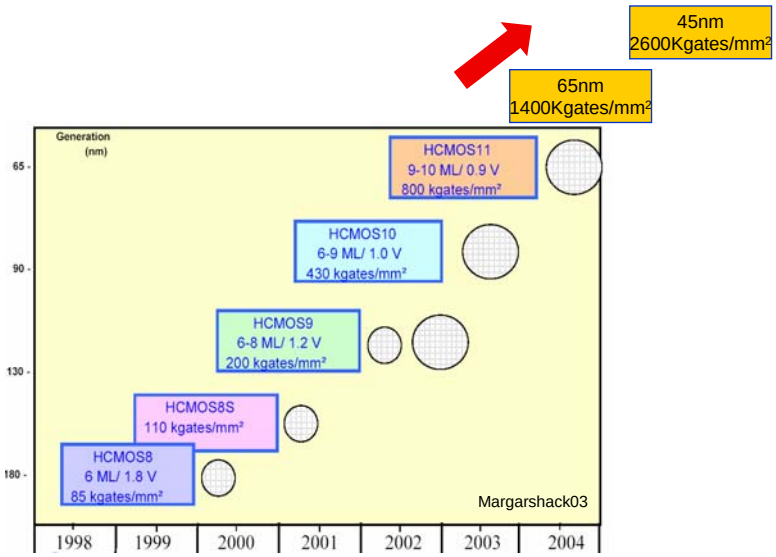



# Graphical illustration of Moore's law




- Something that doubles frequently grows more quickly than most people realize!
  - A 2002 chip can hold about 15,000 1981 chips inside itself

# STMicroelectronics Roadmap

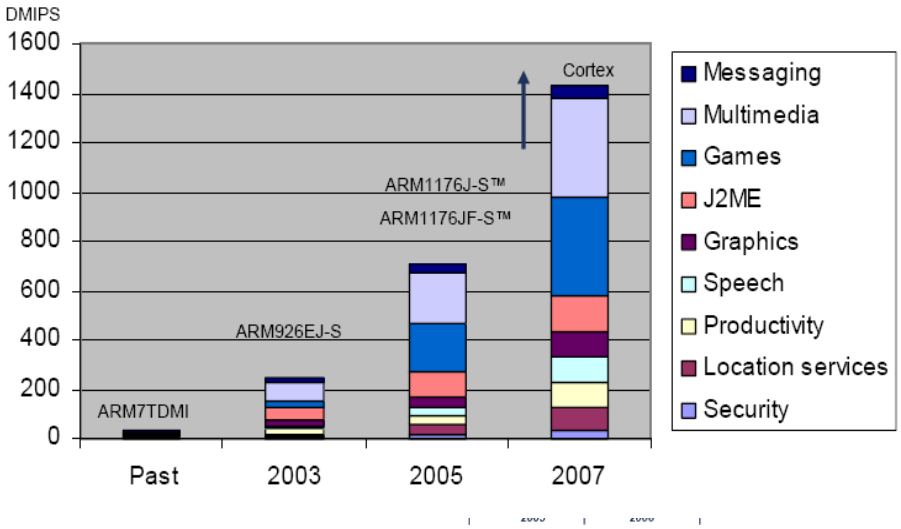





# Technology Progress Overview

- Processor speed improvement: 2x per year (since 85). 100x in last decade.
- DRAM Memory Capacity: 2x in 2 years (since 96). 64x in last decade.
- DISK capacity: 2x per year (since 97). 250x in last decade.

# ARM – Embedded Processor Roadmap






# Technology bottlenecks

# Interconnect Bottleneck

100

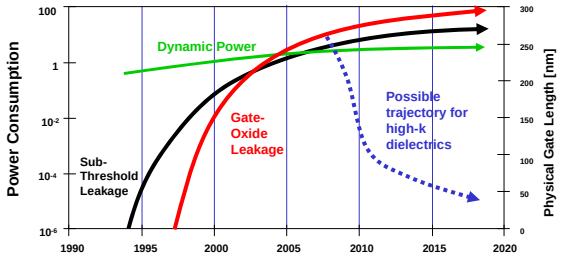
| Operation | Delay (0.13um) | (0.05um) |
|---|---|---|
| 32b ALU Operation | 650ps | 250ps |
| 32b Register Read | 325ps | 125ps |
| Read 32b from 8KB RAM | 780ps | 300ps |
| Transfer 32b across chip (10mm) | 1400ps | 2300ps |
| Transfer 32b across chip (20mm) | 2800ps | 4600ps |

2:1 global on-chip comm to operation delay
9:1 in 2010

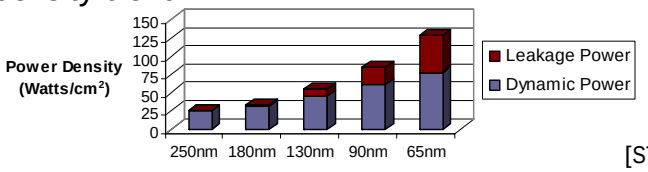


# Power Bottleneck

- Power trend

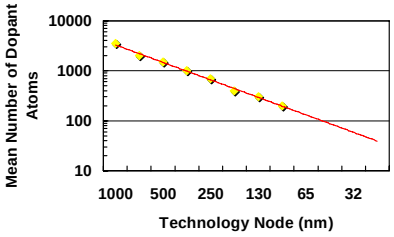



- Power density trend

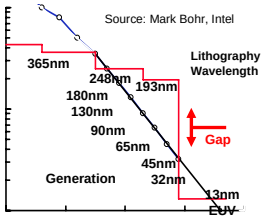



[STM ASIC]

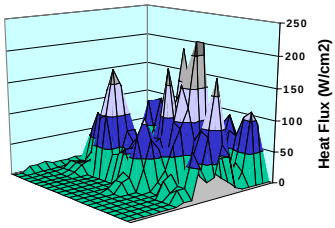
# Variability Bottleneck



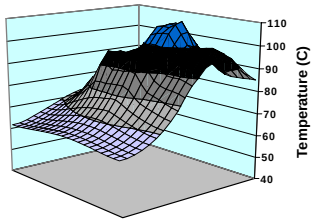

**Random Dopant Fluctuations**




**Sub-wavelength Lithography**




**Heat Flux (W/cm²)—Vcc variation**

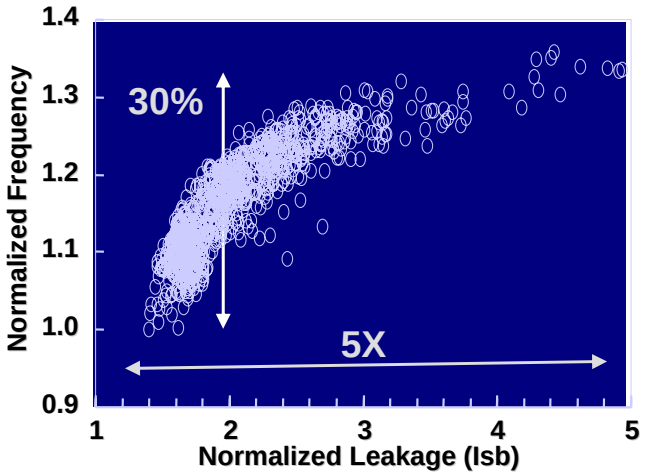



**Temp Variation & Hot spots**

 [Borkar05]

# Impact of Static Variations

***Today…***




**Frequency ~30%**

**Leakage Power ~5-10X**

[Borkar05]

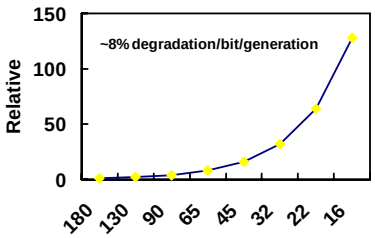
# Technology Outlook

| High Volume Manufacturing | 2004 | 2006 | 2008 | 2010 |
|---|---|---|---|---|
| Technology Node (nm) | 90 | 65 | 45 | 32 |
| Integration Capacity (BT) | 2 | 4 | 8 | 16 |
| Delay = CV/I scaling | 0.7 | ~0.7 | >0.7 | De |
| Energy/Logic Op scaling | >0.35 | >0.5 | >0.5 | Ene |
| Bulk Planar CMOS | High Probability | | | |
| Alternate, 3G etc | Low Probability | | | |
| Variability | Medium | | | Hig |
| ILD (K) | ~3 | <3 | | Redu |
| RC Delay | 1 | 1 | 1 | 1 |
| Metal Layers | 6-7 | 7-8 | 8-9 | 0.5 |

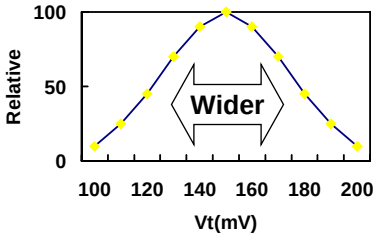
Extreme variations are expected!



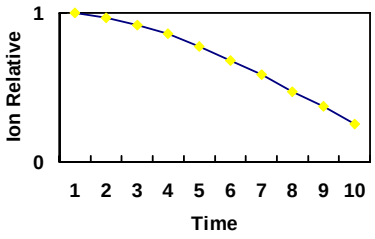
# Reliability Bottleneck




Soft Error FIT/Chip (Logic & Mem)

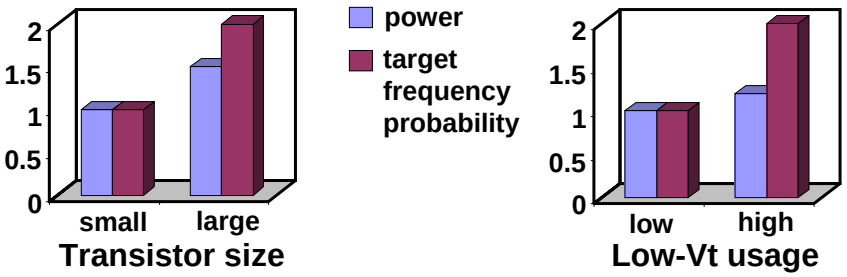



Extreme device variations




Time dependent device degradation

# Circuit Design Tradeoffs



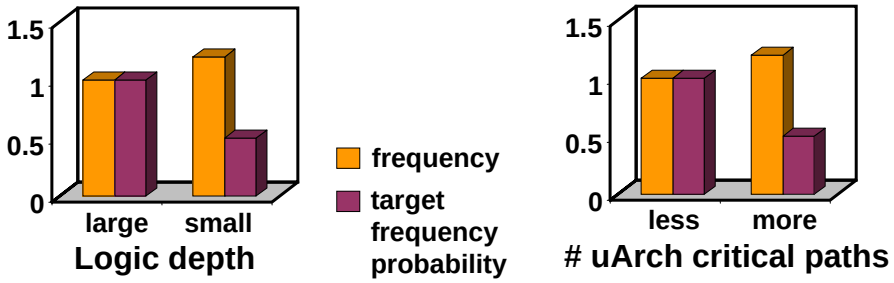

Higher probability of target frequency with:

1. Larger transistor sizes
2. Higher Low-Vt usage

But with power penalty



# μArchitecture Tradeoffs

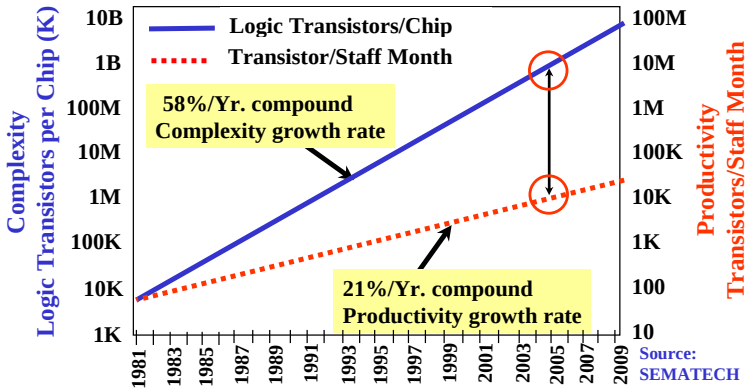



Higher target frequency with:

1. Shallow logic depth
2. Larger number of critical paths

But with lower probability

# Designer Productivity Gap

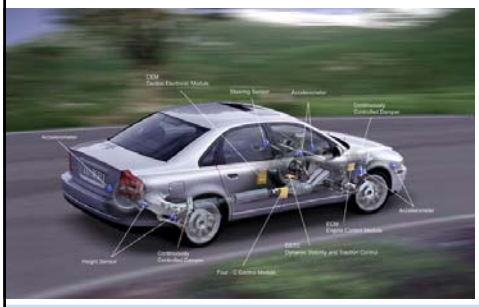



**SoC designs today are complex, characterized by more and more IPs being integrated on a single chip, and a shrinking time-to-market**

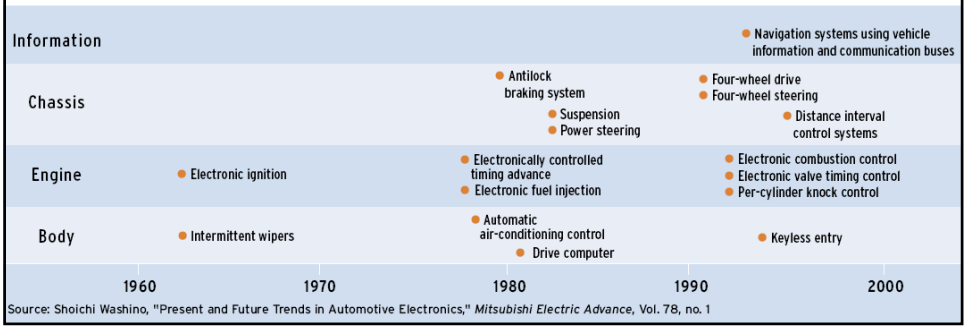


# Application drivers

# Example Area: Automotive Electronics

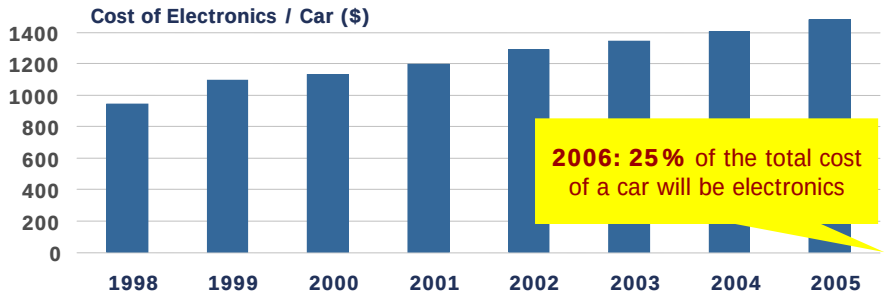


- What is “automotive electronics”?
  - Vehicle functions implemented with electronics
    - Body electronics
    - System electronics: chassis, engine
    - Information/entertainment




Source: Shoichi Washino, "Present and Future Trends in Automotive Electronics," *Mitsubishi Electric Advance*, Vol. 78, no. 1

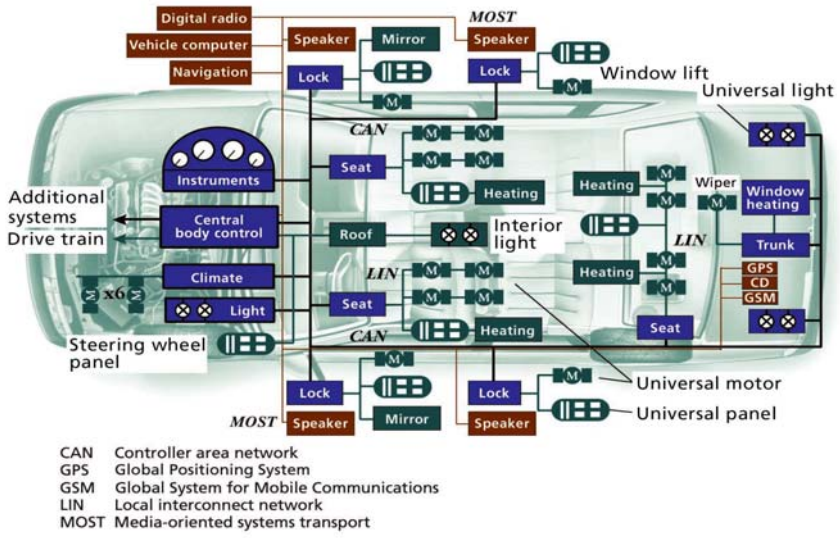
# Automotive Electronics Market Size




| Market ($billions) | 8.9 | 10.5 | 13.1 | 14.1 | 15.8 | 17.4 | 19.3 | 21.0 |
|---|---|---|---|---|---|---|---|---|

**90%** of future innovations in vehicles:
based on electronic **embedded systems**

# Automotive Electronics Platform Example




CAN Controller area network
GPS Global Positioning System
GSM Global System for Mobile Communications
LIN Local interconnect network
MOST Media-oriented systems transport

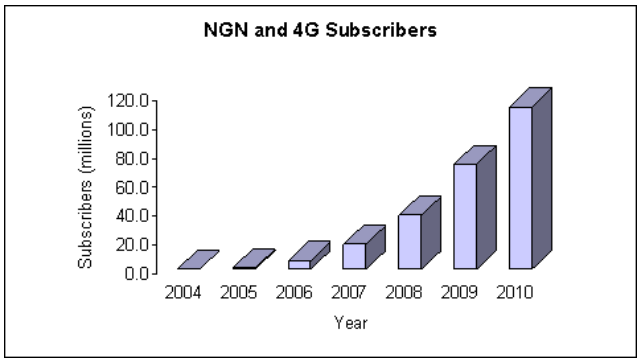
Source: Expanding automotive electronic systems, IEEE Computer, Jan. 2002



# Digital Convergence – Mobile Example


- One device, multiple functions
- Center of ubiquitous media network
- Smart mobile device: next drive for semicon. Industry

# 4th Gen and Next-Gen Networks

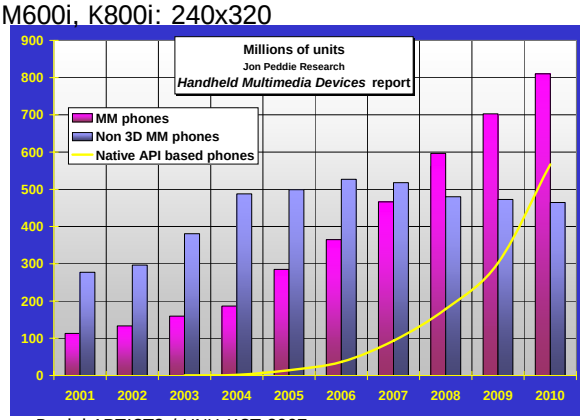



Includes: 802.20, WiMAX (802.16), HSDPA, TDD UMTS, UMTS and future versions of UMTS

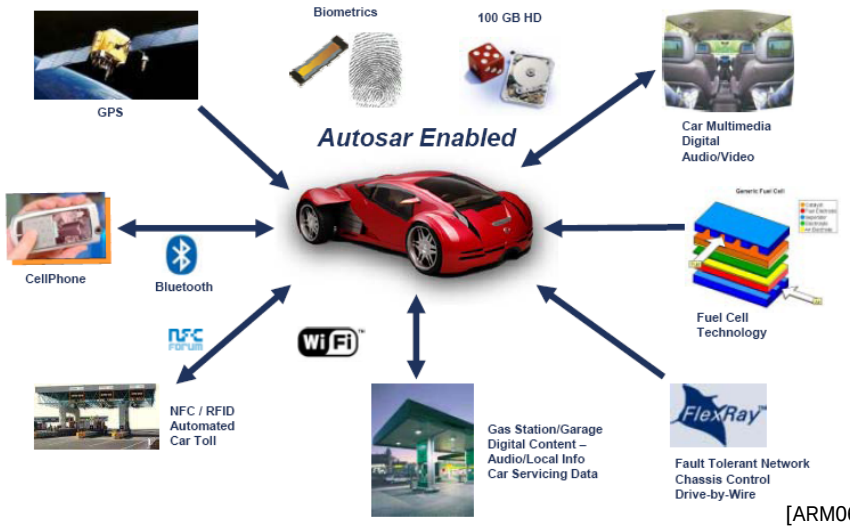


# Mobile graphics/games

- Resolution today ~176x208 –320x240
  - In Japan, QVGA (320x240) is the norm
  - Nokia series 90 is 640x320
  - Nokia N93 is 320x200
  - Sony Ericsson S700i, M600i, K800i: 240x320
- To 1024x768 in the future, VGA (640x400) first



# Automotive applications trends




[ARM06]



# Mobile applications trends




[ARM06]

# Consumer/Home applications trends


Luca Benini ARTIST2 / UNU IIST 2007

[ARM06]

# Media Portability Problem

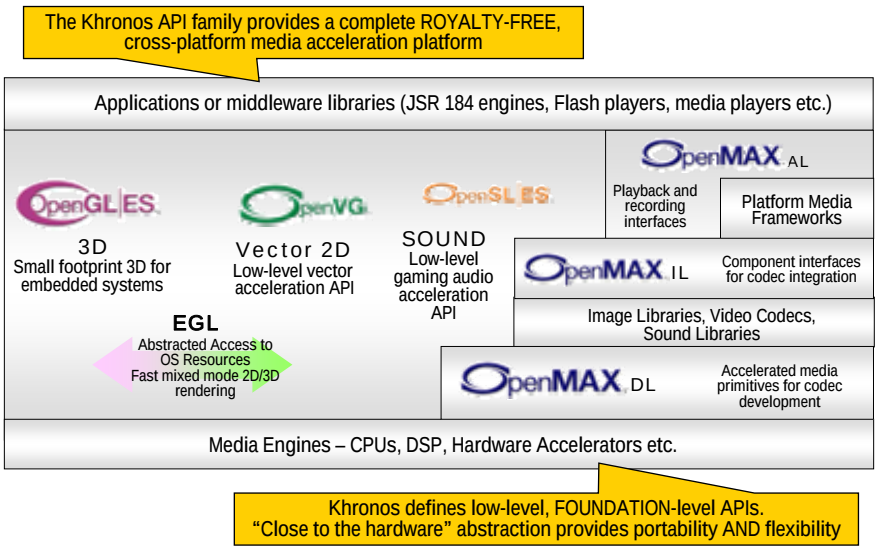
- Media infrastructure portability is a multi-level industry problem
  - Media infrastructure is time-consuming and expensive to develop, integrate and program

| | | |
|---|---|---|
| Media applications are not portable as proprietary hardware-centric libraries are needed to access media acceleration | **Media Applications**<br>Use media networks to deliver a compelling user experience | Application programmers need a cross-platform portable API for controlling high-level media operations |
| Difficult to flexibly integrate diverse, multi-vendor media components into complete media processing solutions | **Media Graphs**<br>Media components connected to process media in real-time | System integrators need cross-vendor standard for media component integration with sophisticated data routing and robust synchronization |
| Software components are not portable across processors - exacerbated by proliferation of media standards and increasing silicon complexity | **Media Components**<br>Inputs, video codecs, image and sound libraries, outputs etc. | Software component & silicon vendors need to a reliable way to accelerate diverse codecs on diverse silicon |

**Media Infrastructure Stack**
Hardware and software to deliver rich media processing solutions

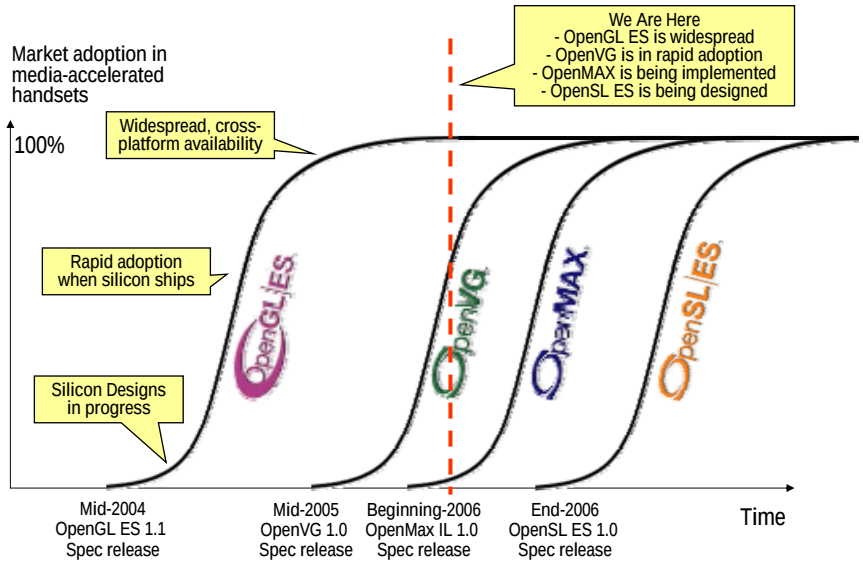
Luca Benini ARTIST2 / UNU IIST 2007

# Complete Khronos Media Stack





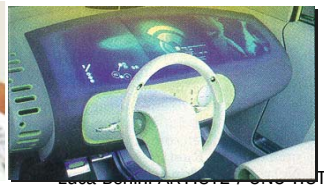

# Adoption of Khronos APIs

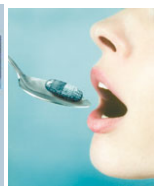


# The Ambient Intelligence Dream




**Secure, trustworthy computing and communication embedded in every-thing and every-one.**
**A pervasive, context aware ambient, sensitive and responsive to the presence of people**



# AmI = Complex Interplay of 4 Technologies driven by Societal Challenges

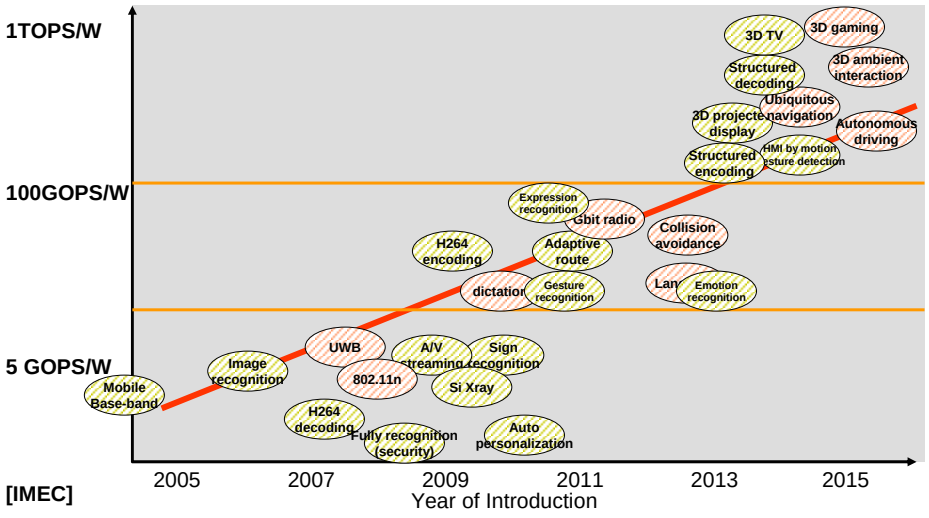



- **Embedded (distributed) computing**
- **Ubiquitous adaptive wireless communication**
- **Transducer networks i/f to physical world**
- **Multi-mode, natural interfaces to the user**

©ISTAG

Luca Benini ARTIST2 / UNU IIST 2007

# Application pull






# Characteristics of AMI systems

- Must be **dependable**,
  - **Reliability R(t) =** probability of system working correctly provided that is was working at t=0
  - **Maintainability M(d) =** probability of system working correctly d time units after error occurred.
  - **Availability**: probability of system working at time t
  - **Safety**: no harm to be caused
  - **Security**: confidential and authentic communication
  - Even perfectly designed systems can fail if the assumptions about the workload and possible errors turn out to be wrong.
  - Making the system dependable must not be an after-thought, it must be considered from the very beginning

## Characteristics of AMI Systems (2)

- Must be **efficient**
  - Energy efficient
  - Code-size efficient (especially for systems on a chip)
  - Run-time efficient
  - Weight efficient
  - Cost efficient
- **Dedicated** towards a certain **application**
  Knowledge about behavior can at design time can be used to minimize resources and to maximize robustness
- **Dedicated user interface**
  (no mouse, keyboard and screen)



## Characteristics of AMI Systems (3)

- Many ES must meet **real-time constraints**
  - A real-time system must react to stimuli from the controlled object (or the operator) within the time interval **dictated** by the environment.
  - For real-time systems, right answers arriving too late are wrong.
  - **„A real-time constraint is called hard, if not meeting that constraint could result in a catastrophe“** [Kopetz, 1997].
  - All other time-constraints are called **soft**.
  - A guaranteed system response has to be explained without statistical arguments

# Characteristics of AMI Systems 4

- Frequently **connected to physical environment** through sensors and actors,
- **Hybrid systems** (analog + digital parts).
- Typically, ES are **reactive systems**:
  „**A reactive system is one which is in continual interaction with is environment and executes at a pace determined by that environment“** [Bergé, 1995]
  Behavior depends on input **and current state**.
  ☞ automata model appropriate,
  model of computable functions inappropriate.



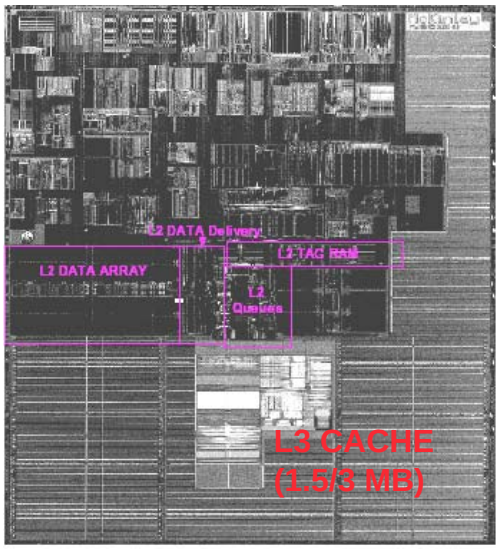
# Architecture Evolution

# The AMI processing Bestiary

- The work-horse
  - Powers the fixed base network machines
  - Power W Performance GB/s
- The hummingbird
  - Powers the wireless base network interfaces
  - Power mW Performance MB/s
- The butterfly
  - The sensor network hardware
  - Power µW Performance KB/s

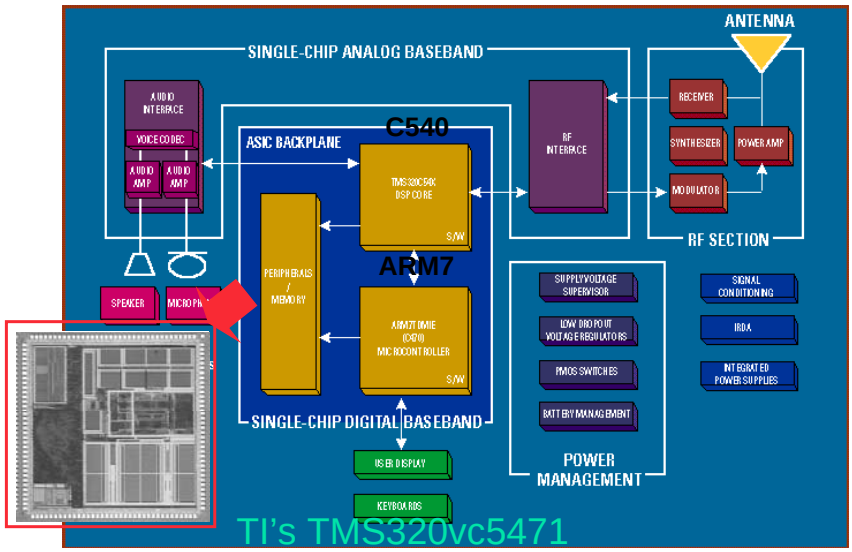


## Itanium® 2 Processor



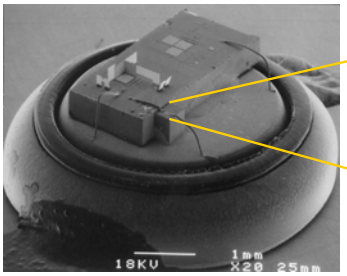

- Released at 733MHz and 800MHz, now 1GHz
- Three level caching system
- 25 million transistors in the CPU and 300 million in the cache (0.18µm)
- 421mm$^2$ die size
- The CPU running at full load draws ~130 Watts
- The clock signals and logic total to approx 84% of the total power usage.
- Leakage power: approx. 2%.
- Power delivery: $V_{dd}$=1.5V, P=130W, P=$V_{dd}$I (!!)

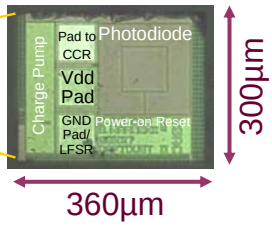
# Handset architecture





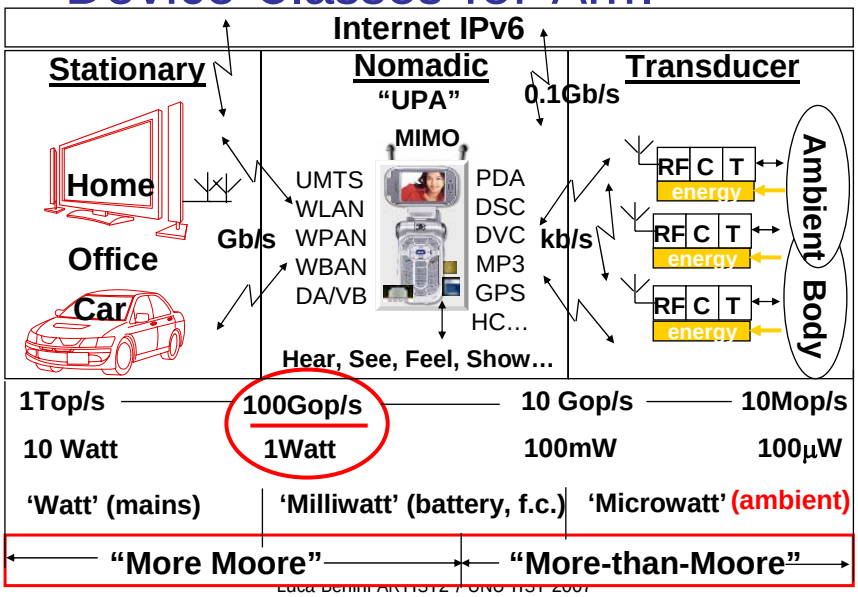

# Berkeley's Daft Dust Device



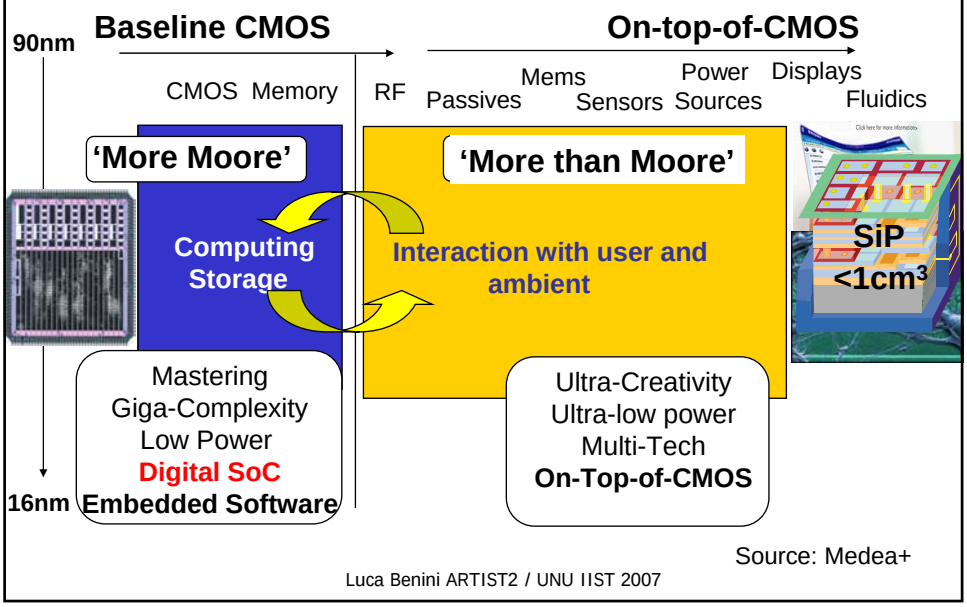




- 63 $mm^3$
- Circuits: 0.25 µm CMOS
  - digital circuits underneath ground pad
  - metal shields to prevent photogenerated carriers
- CCR: Cronos MUMPS
- ☹ Optical wireless connection (line of sight)

# Device Classes for AmI

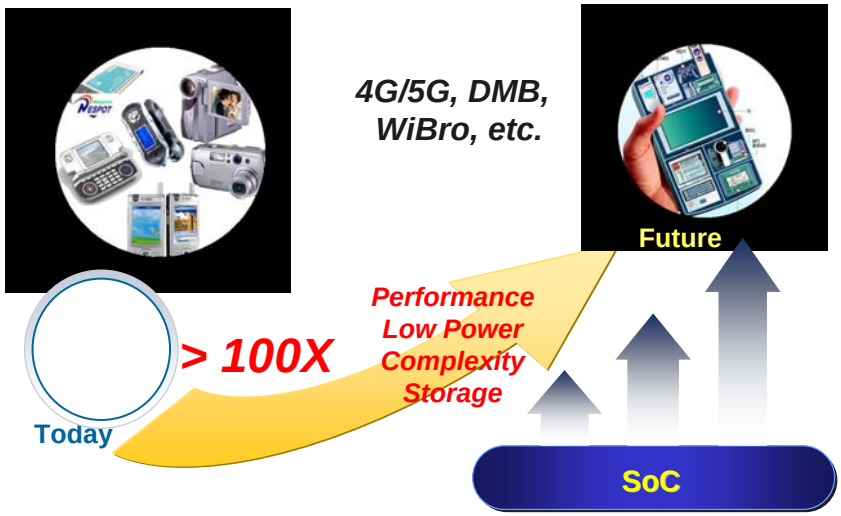



# 'More Moore' vs. 'More than Moore'



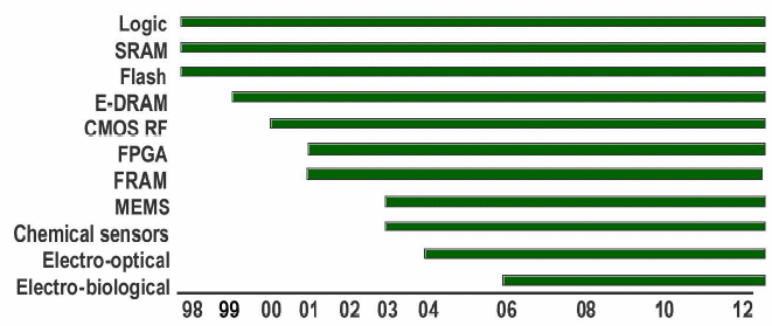

Source: Medea+

## SoC: Enabler for Digital Convergence




Luca Benini ARTIST2 / UNU IIST 2007

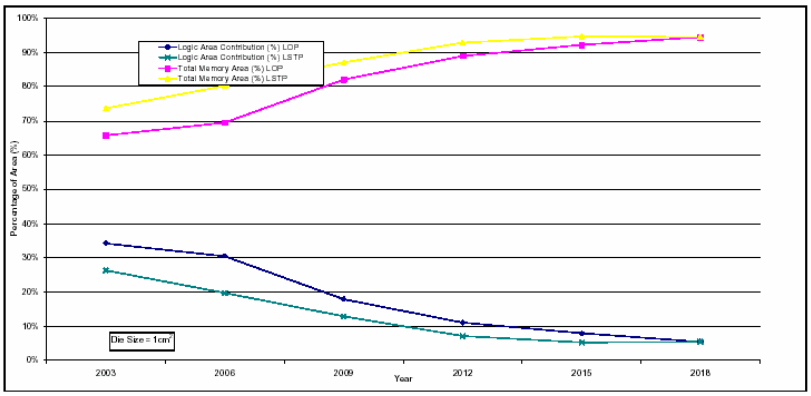
## SoC INTEGRATION




Number of integrated technologies limited by cost considerations
Potential side-by-side integration of Si, GaAs, MEMs

Luca Benini ARTIST2 / UNU IIST 2007

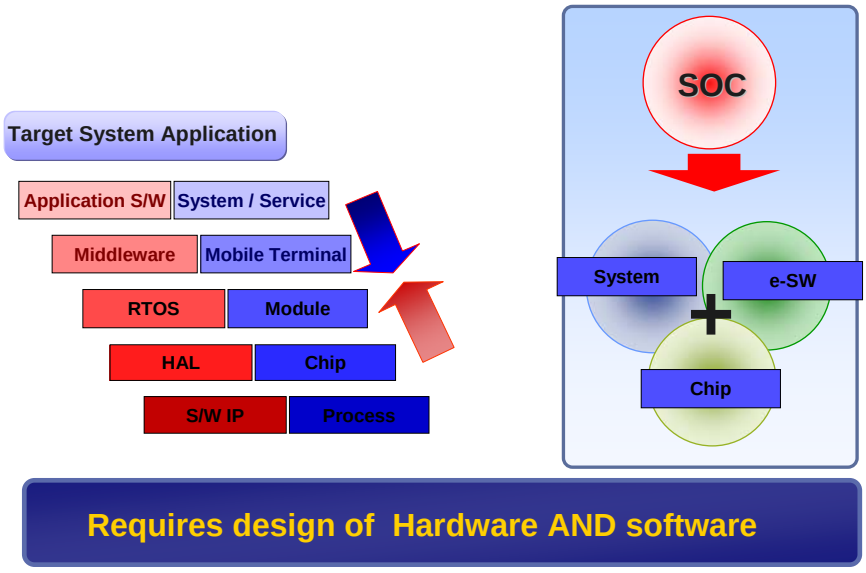
# Chip composition




Strong unbalance between logic and memory area occupancy

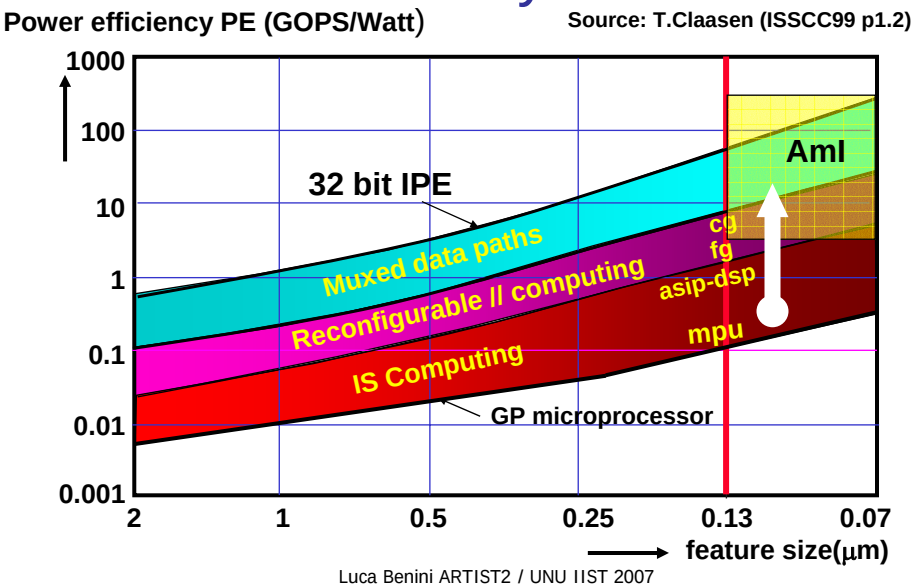


# SoC → Solution-on-a-Chip




**Requires design of Hardware AND software**

# Power-Flexibility Conflict






# The Architectural Gap

**"Managing Giga-Complexity "**

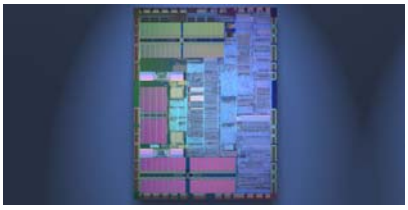
Create SL methods, tools and skills for designing flexible, yet power-efficient platforms

= 1 B uncertain devices + >10M lines of code

**Greatest Challenge:**
**Overcome Power-Flexibility Conflict**
**While coping with CMOS (r)evolution**

# Competing imperatives

- Technology push:
  - high-volume products;
  - feasible design.
- Marketing push:
  - fast turnaround;
  - differentiated products.

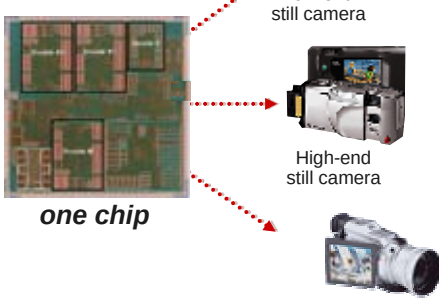


IBM PowerPC 750

Nokia 9210

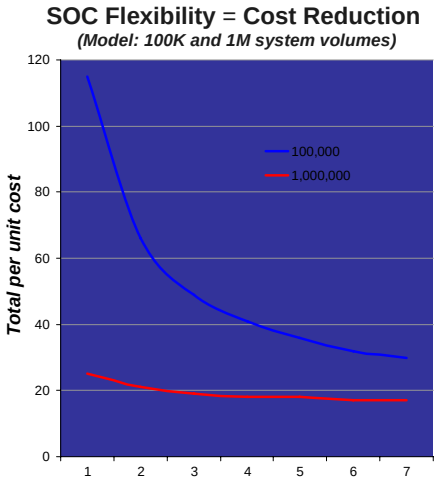


# ROI Goal: One Design, Many Design-ins




***one chip***

***many system designs***




$10M design cost, $15 manf. cost, 5% premium for programmability

# Flexibility is the Key to ROI

**Programmability more "hot features" available**

**Flexibility means more systems per design**

**Little impact on chip cost – pennies per processor**

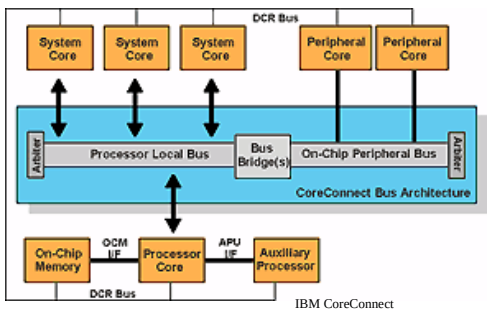
$$ROI = \frac{\text{Return}}{\text{Investment}} = \frac{\text{volume} * (\text{chip ASP} - \text{chip unit cost})}{\text{chip developmen t cost}} = \frac{\uparrow * \uparrow}{\downarrow} = \uparrow$$

**Reduce design time, team size and re-spin risk**



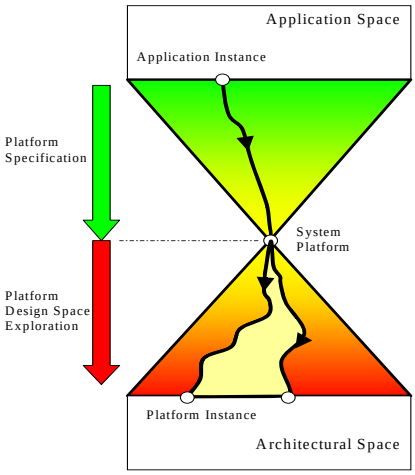
# What is a (HW) platform?

- A partial design:
  - for a particular type of system;
  - includes embedded processor(s);
  - may include embedded software;
  - customizable to requirements:
    - software;
    - component changes.




IBM CoreConnect

# The “hourglass” model




**A Hardware Platform is a family of architectures that satisfy a set of architectural constraints imposed to allow the re-use of hardware and software components.**



# Why platforms?

- Any given space has a limited number of good solutions to its basic problems.
- A platform captures the good solutions to the important design challenges in that space.
- A platform reuses architectures.

## Standards and platforms

- Many high-volume markets are standards-driven:
  - wireless;
  - multimedia;
  - networking.

MPEG Tampere meeting

- Standard defines the basic I/O requirements.

bluetooth.com



## Standards and platforms 2

- Systems house chooses implementation of standards functions:
  - improved quality, lower power, etc.
- Product may be differentiated by added features:
  - cell phone user interface.
- Standards encourage platform-based design.

## Platform vs. full-custom

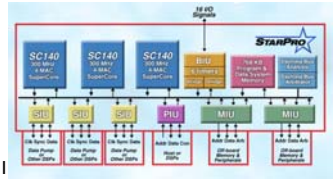
- Platform has many fewer degrees of freedom:
  - harder to differentiate;
  - can analyze design characteristics.
- Full-custom:
  - extremely long design cycles;
  - may use less aggressive design styles if you can't reuse some pieces.



## Platforms and embedded computing

- Platforms rely on embedded processors:
  - can be customized through software;
  - can put considerable design effort into the CPU.
- Many platforms are complex heterogeneous multiprocessors.

Agere StarPro



## Platforms and IP-based design

- Platforms use IP:
  - CPUs;
  - memories;
  - I/O devices.
- Platforms are IP at the next level of abstraction.



## Advantages of platform-based design

- Fast time-to-market.
- Reuse system design---hardware, software.
- Allows chip to be customized to add value.

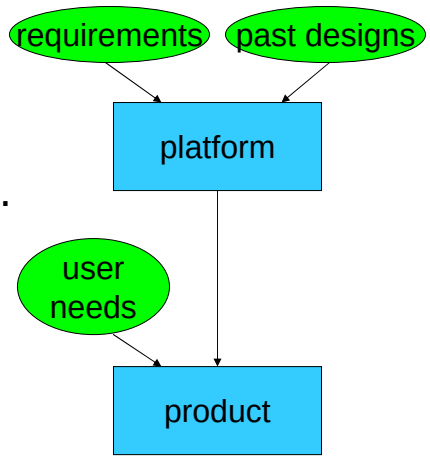
# Costs of platform-based design

- Masks.
- NRE: design of the platform + customization.
- Design verification.



# Two phases of platform-based design

- Design the platform.
- Use the platform.



## Division of labor

- Platform design:
  - choose, characterize hardware units;
  - create the system architecture;
  - optimize for performance, power.
- Platform-based product design:
  - modify hardware architecture;
  - optimize programs.



## Semiconductor vs. systems house

- Semiconductor house designs the platform.
- Systems house customizes the platform for its system:
  - customization may be done in-house or by contractor.

## Platform design challenges

- Does it satisfy the application's basic requirements?
- Is it sufficiently customizable? And in the right ways?
- Is it cost-effective?
- How long does it take to turn a platform into a product?



## Platform design methodology

- Size the problem.
  - How much horsepower? How much power?
- Develop an initial architecture.
- Evaluate for performance, power, etc.
- Evaluate customizability.
- Improve platform after each use.

## Platform use challenges

- How do I understand the platform's design?
- How do I modify it to suit my needs?
- How do I optimize for performance, power, etc.?



## Platform use methodology

- Start with reference design, evaluate differences required for your features.
- Evaluate hardware changes.
- Implement hardware and software changes in parallel.
- Integrate and test.

## Design refinement

- Bad news:
  - hard to learn the platform in order to change it.

    **Worldwide shipping by UPS ...**

    **roughly US$ 50 for CD and US$ 100 for paper copy**

    **(1500 pages, heavy!)**

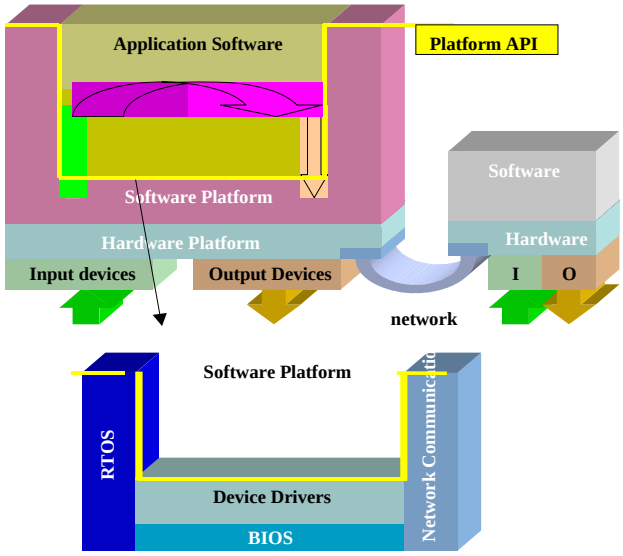
    Bluetooth.com
- Good news:
  - an existing design can be measured, analyzed, and refined.



## Hardware Platforms Not Enough!

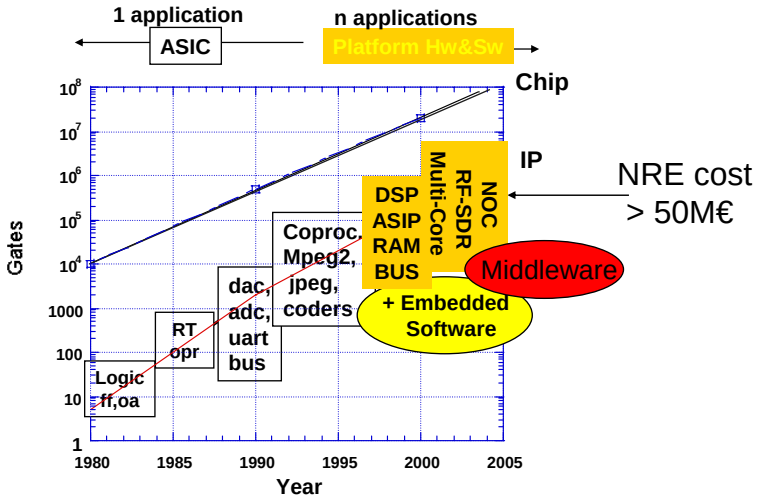
- Hardware platform has to be abstracted
- Interface to the application software is the "API"
- Software layer performs abstraction:
  - Programmable cores and memory subsystem "hidden" by RTOS and compilers
  - I/O subsystem with Device Drivers
  - Network with Network Communication Software

# Software Platforms






# Hw ASIC → Sw SoC Platform

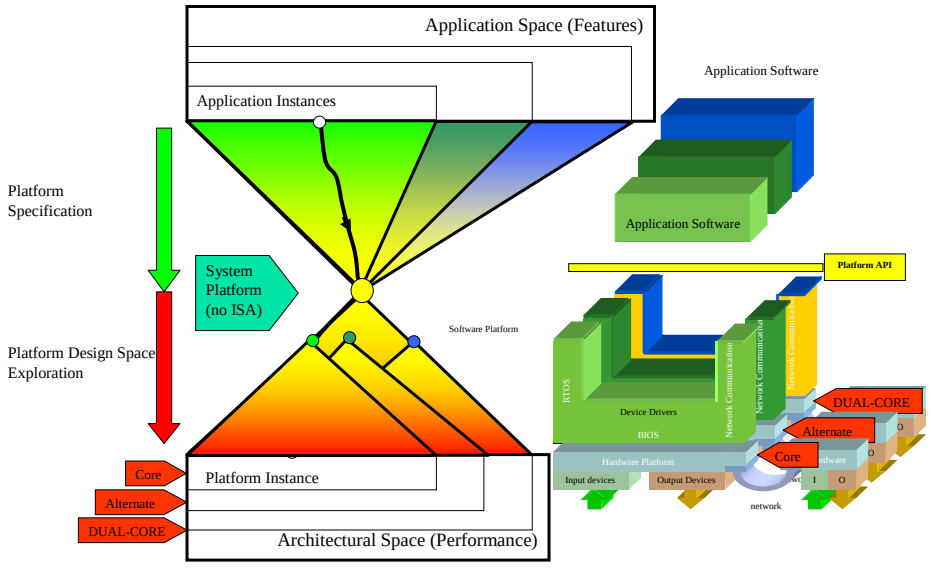


# Software and hardware reuse

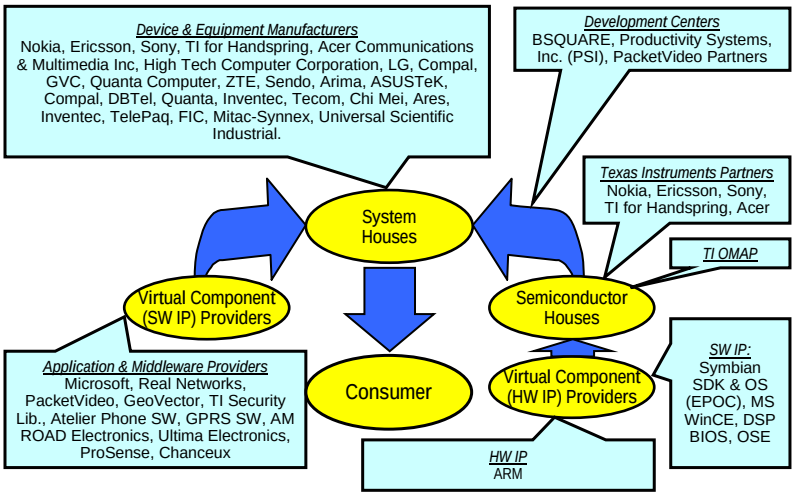
- Want to reuse as many hardware components as possible:
  - known performance, power.
- Want to use software libraries where possible.
- RTOS simplifies design of multi-tasking systems.



# HW-SW Platforms: Exploration and Reuse



# Platform Eco-System: e.g. TI OMAP




**Source: IEEE Computer**



# Pros and cons

- Plentiful hardware options.
- Simple programming semantics.
- Good software development environments.
- Performance-limited.

# Summary

- Technology evolution
  - Bottlenecks
- Applications
  - Application trends
  - Ambient intelligence
- Platforms