Challenges

Shift to
- Reuse Strategy
- Higher Level of Abstractions
- Software !!!

microelectronics group

SoC Landscape 2000+

- Cost of ownership is High
  - Development cost of a high end ASSP can exceed $5M
  - Cost of fabrication and mask making has increased significantly ($500k+ for masks alone)
  - >15x design productivity gap (Spec to Verified Netlist)

- Compounding complexities limiting Time-to-Market, cycle time reduction needed to meet Customer expectations
  - Chip design complexity
  - Silicon process complexity
  - Context complexity
  - End-to-end verification

- New “System to Silicon” methodologies are required that recognize > 50% of the system development is software

Lucent Technologies

PERCENT OF TRANSISTORS WITHIN EMBEDDED IP (EXCLUDES MEMORY)

Random Logic Transistors

Transistors Within Embedded IP


Feature Dimension (µm): 0.35 0.25 0.18 0.15 0.13 0.11 0.10 0.08 0.07

Transistors (%)

100

5
**TRENDS IN EMBEDDED IP**

**SYSTEM IC Market**

**Designs With Embedded IP Will Dominate the System IC Business in the Future**

<table>
<thead>
<tr>
<th>Year</th>
<th>1998</th>
<th>1999</th>
<th>2000</th>
<th>2001</th>
<th>2002</th>
<th>2003</th>
<th>2004</th>
<th>2005</th>
<th>2006</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total IC Value ($M)</td>
<td>112,005</td>
<td>133,426</td>
<td>174,723</td>
<td>164,592</td>
<td>173,506</td>
<td>197,118</td>
<td>227,752</td>
<td>269,078</td>
<td>313,214</td>
</tr>
<tr>
<td>Growth rate (%)</td>
<td>NA</td>
<td>19.1</td>
<td>31.0</td>
<td>(5.8)</td>
<td>5.4</td>
<td>13.6</td>
<td>15.5</td>
<td>18.1</td>
<td>16.4</td>
</tr>
<tr>
<td>System IC Value ($M)</td>
<td>43,506</td>
<td>50,493</td>
<td>64,366</td>
<td>67,581</td>
<td>76,034</td>
<td>92,804</td>
<td>115,123</td>
<td>145,700</td>
<td>175,767</td>
</tr>
<tr>
<td>Growth rate (%)</td>
<td>NA</td>
<td>16.1</td>
<td>27.5</td>
<td>5.0</td>
<td>12.5</td>
<td>22.1</td>
<td>24.0</td>
<td>26.6</td>
<td>20.6</td>
</tr>
<tr>
<td>Percent total (%)</td>
<td>38.8</td>
<td>37.8</td>
<td>36.8</td>
<td>41.1</td>
<td>43.8</td>
<td>47.1</td>
<td>50.5</td>
<td>54.1</td>
<td>56.1</td>
</tr>
<tr>
<td>IP-based design Value ($M)</td>
<td>15,706</td>
<td>19,894</td>
<td>30,123</td>
<td>39,129</td>
<td>51,247</td>
<td>68,582</td>
<td>90,602</td>
<td>122,679</td>
<td>153,093</td>
</tr>
<tr>
<td>Growth rate (%)</td>
<td>NA</td>
<td>26.7</td>
<td>51.4</td>
<td>29.9</td>
<td>31.0</td>
<td>33.8</td>
<td>32.1</td>
<td>35.4</td>
<td>24.8</td>
</tr>
<tr>
<td>Percent system IC (%)</td>
<td>36.1</td>
<td>39.4</td>
<td>46.8</td>
<td>57.9</td>
<td>67.4</td>
<td>73.9</td>
<td>78.7</td>
<td>84.2</td>
<td>87.1</td>
</tr>
</tbody>
</table>
Computing for Embedded Systems

Image "borrowed" from an Iomega advertisement for Y2K software and disk drives, Scientific American, September 1999.
EMBEDDED SYSTEM: THE REAL STORY
FABIO ROMEO

Design Automation Conference
Las Vegas, June 20th, 2001
## COMPLEXITY, QUALITY, TIME-TO-MARKET: TODAY

<table>
<thead>
<tr>
<th></th>
<th>PWT UNIT</th>
<th>BODY GATEWAY</th>
<th>INSTRUMENT CLUSTER</th>
<th>TELEMATIC UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MEMORY</strong></td>
<td>256 KB</td>
<td>128 KB</td>
<td>184 KB</td>
<td>8 MB</td>
</tr>
<tr>
<td><strong>LINES OF CODE</strong></td>
<td>50.000</td>
<td>30.000</td>
<td>45.000</td>
<td>300.000</td>
</tr>
<tr>
<td><strong>PRODUCTIVITY</strong></td>
<td>6 LINES/DAY</td>
<td>10 LINES/DAY</td>
<td>6 LINES/DAY</td>
<td>10 LINES/DAY*</td>
</tr>
<tr>
<td><strong>RESIDUAL DEFECT RATE @ END OF DEV</strong></td>
<td>3000 PPM</td>
<td>2500 PPM</td>
<td>2000 PPM</td>
<td>1000 PPM</td>
</tr>
<tr>
<td><strong>CHANGING RATE</strong></td>
<td>3 YEARS</td>
<td>2 YEARS</td>
<td>1 YEAR</td>
<td>&lt; 1 YEAR</td>
</tr>
<tr>
<td><strong>DEV. EFFORT</strong></td>
<td>40 MAN-YEAR</td>
<td>12 MAN-YEAR</td>
<td>30 MAN-YEAR</td>
<td>200 MAN-YEAR</td>
</tr>
<tr>
<td><strong>VALIDATION TIME</strong></td>
<td>5 MONTHS</td>
<td>1 MONTH</td>
<td>2 MONTHS</td>
<td>2 MONTHS</td>
</tr>
<tr>
<td><strong>TIME TO MARKET</strong></td>
<td>24 MONTHS</td>
<td>18 MONTHS</td>
<td>12 MONTHS</td>
<td>&lt; 12 MONTHS</td>
</tr>
</tbody>
</table>

* C++ CODE
COMPLEXITY, QUALITY, TIME-TO-MARKET: FUTURE TRENDS

KEY DRIVERS
• QUALITY
• TIME-TO-MARKET
• COMPLEXITY MGMT

WINNING SOLUTIONS
• PLATFORM & APPLICATIONS
• DESIGN METHODOLOGIES
• TESTING
Software Productivity

Roger G. Fordham
Director, Performance Excellence
Motorola, Global Software Group
Roger.Fordham@Motorola.com
June 6, 2001
**The Software Development Problem**

- Product Quality is POOR
- Development Productivity is LOW
- Development Cycle-time is TOO LONG

**System Software (of size 10,000 Function Points)**

<table>
<thead>
<tr>
<th>Quality</th>
<th>Industry Average</th>
<th>Ind. Best-in-Class</th>
<th>Customer Expectation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delivered Defects per Function Point</td>
<td>0.44</td>
<td>0.08</td>
<td>&lt;0.000044</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Productivity</th>
<th>Industry Average</th>
<th>Ind. Best-in-Class</th>
<th>Customer Expectation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function Point per Staff Month</td>
<td>4.13</td>
<td>8.76</td>
<td>&gt;40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CycleTime</th>
<th>Industry Average</th>
<th>Ind. Best-in-Class</th>
<th>Customer Expectation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schedule in Months</td>
<td>36</td>
<td>25</td>
<td>&lt;3-6</td>
</tr>
</tbody>
</table>

What are the Remedies?

- Significant commitment to CONTINUOUS IMPROVEMENT
- Effective use of DESIGN METHODOLOGIES
- Effective use of development/management AUTOMATION

P / PC Balance
90 : 10 %

SDL  UML  FML
Software Architecture Today

Poor common infrastructure. Weak specialization of functions. Poor resource management. Poor planning.
Software Architecture Tomorrow?
The “C” or “Java” Paradigm

- Not abstract enough to capture functionality only
- Not detailed enough to capture important parameters such as performance, energy consumption, “size”
What about “real time”? 

“Make it faster!”
Problems with Past Design Method

- Lack of unified hardware-software representation

- Partitions are defined *a priori*
  - Can't verify the entire system
  - Hard to find incompatibilities across HW-SW boundary
    (often found only when prototype is built)

- Lack of well-defined design flow
  - Time-to-market problems
  - Specification revision becomes difficult
Design Effort vs. System Design Value

Level of Abstraction

Effort/Value

Today

Tomorrow

Function

HW/SW

Architecture

RTL - SW

Mask - ASM

Conceptual Gap

RTL/Gate “platform”
Design Effort vs. System Design Value

Level of Abstraction

Function

HW/SW

Architecture

RTL - SW

Mask - ASM

Effort/Value

Today

Tomorrow

Design Entry Level

Hand-off “platform”

Hand-off “platform”

Hand-off “platform”

Hand-off “platform”

Hand-off “platform”

Hand-off “platform”
New Levels of Design Chain Interaction

Today

Effort/Value

Level of Abstraction

Function

HW/SW

Architecture

RTL - SW

Mask - ASM

Tomorrow

Architecture Space

System Platform

Application Space
High-Leverage Paradigms

If we face a problem that has become too complex to solve, eliminate the problem!

- Decompose
- Approximate
- Solve by construction
Separate Behavior from Micro-architecture

◆ System Behavior
  ♦ Functional Specification of System.
  ♦ No notion of hardware or software!

◆ Implementation Architecture
  ♦ Hardware and Software
  ♦ Optimized Computer
Models of Computation: And There are More...

- Continuous time (ODEs)
- Spatial/temporal (PDEs)
- Discrete time
- Rendezvous
- Synchronous/Reactive
- Dataflow
- ...

Each of these provides a formal framework for reasoning about certain aspects of embedded systems.

We are searching for an abstraction that provides the Source for all MoCs that can be obtained by refinement.
Formalization

Model of a design with precise unambiguous semantics:

- Implicit or explicit relations: inputs, outputs and (possibly) state variables
- Properties
- “Cost” functions
- Constraints

Formalization of Design + Environment = closed system of equations and inequalities over some algebra.
Validating Designs

- By construction
  - property is inherent.

- By verification
  - property is provable.

- By simulation
  - check behavior for all inputs.

- By intuition
  - property is true. I just know it is.

- By assertion
  - property is true. Wanna make something of it?

- By intimidation
  - Don’t even try to doubt whether it is true

It is generally better to be higher in this list
**Notion of Time**

- **Continuous time**
- **Discrete time**
- **Totally-ordered discrete events**
- **Multirate discrete time**
- **Partially-ordered discrete events**
- **Synchronous/reactive**

![Salvador Dali, The Persistence of Memory, 1931](image-url)
Two Basic Questions ... Question I - IP Authoring

How to design a system block?
- Starting from the system level
- With a consistent test-bench
- Getting from the abstract, un-timed system model to the clocked HW or SW implementation model

Example
- Rake Receiver
  - Which are the optimal algorithms?
  - How does it work fixed point?
  - How is it best implemented?
  - Does the implementation work as specified in the system level
Two Basic Questions …
Question II – IP Integration

How to integrate system blocks?

- Starting from the system level
- With a consistent test-bench
- Getting from the abstract, un-timed system model to the clocked HW or SW implementation model
- Communication between blocks
- Addressing Platform Based design

Example

- 3G Cell phone
  - Which are the optimal algorithms?
  - Do they work together functionally?
  - Is the architecture sufficient?
  - Does the implementation integration work?
The new approach

- Not the typical stepwise top-down refinement: we rest on platforms!
- Explicit mapping of applications onto architecture components
- The higher the level of abstraction, the faster is the design time
Ptolemy

- E. Lee Project at UC Berkeley
- Multiple models of computation
- DSP beginnings: Static Dataflow
- Many other models: FSM, Discrete Event
- Mixed model verification
A bit of history: the POLIS project

- 1988:

- The problem:

- The target architecture:
Example of System Behavior

12/09/1999
IP-Based Design of the System Behavior
The next level of Abstraction …
IP-Based Design of the Implementation

Which DSP Processor? C50?
Can DSP be done on Microcontroller?

Which Bus? PI?
AMBA?
Dedicated Bus for DSP?

Can I Buy an MPEG2 Processor? Which One?

Which Microcontroller? ARM? HC11?

How fast will my User Interface Software run? How Much can I fit on my Microcontroller?

Can I need a dedicated Audio Decoder?
Can decode be done on Microcontroller?
Architectural Choices

- Dedicated Logic
  - Direct Mapped Hardware
  - Satellite Processor

- Hardware Reconfigurable Processor

- Prog Mem
  -μP
  - MAC Unit
  - Addr Gen
  - Software Programmable DSP

- General Purpose μP

Flexibility vs. 1/Efficiency (power, speed)
Map Between Behavior from Architecture

Transport Decode Implemented as Software Task Running on Microcontroller

Communication Over Bus

Audio Decode Behavior Implemented on Dedicated Hardware
**Classic A/D, HW/SW tradeoff**

- **RF Front End**
- **Can trade custom analog for hardware, even for software**
  - Power, area critical criteria, or easy functional modification
Example: Voice Mail Pager

- Design considerations cross design layers
- Trade-offs require systematic methodology and constraint-based hierarchical approach for clear justification
**Where All is Going**

- **HW/SW Co-Design Paradigm (Felix)**
  - **Rowson, ASV**

- **VSI Design Paradigm**
  - **Chang et al.**

**Convergence of Paradigms**

- **Analog Top-Down Design Methodology**

**Create paradigm shift- not just link methods**
- New levels of abstraction to fluidly tradeoff HW/SW, A/D, HF/IF, interfaces, etc- to exploit heterogeneous nature of components
- Links already being forged
Concluding Remarks

- The Industry Structure is undergoing a revolutionary change
- The Design Problems are changing radically their main characteristics
- System Design is becoming more and more the key to success
- System implies Major Emphasis on Software
- Analog, Sensors, Actuators, RF must be part of design
- Deep Submicron makes most of the tools obsolete