

# Architectural Issues for Embedded Processors and Applications

IEEE Computer Society  
Memphis Chapter / FedEx Institute of Technology  
Tuesday, November 1, 2005

Robert Gezelter Software Consultant  
35 – 20 167th Street, Suite 215  
Flushing, New York 11358 – 1731  
United States of America

+1 (718) 463 1079  
gezelter@rlgsc.com  
<http://www.rlgsc.com>

## Why do things go wrong?

- Therac 25
- review COMP.RISKS
- unexpected coupling

## Evolution is a misleading metaphor

- evolution requires mutation and speciation
- speciation in software dramatically increases costs
- linear progressions only exist as retrospectives

## Architecture: What and When?

- intellectual pathfinding
- framework for expressing problems
- changes are hard to retrofit

## Context

- Capacity – then and now
- What is “Architecture”?
- Technology
- Architecture as leverage

## Capacity – then and now

- Then (1970): Intel 4004
- Now (2005): Intel-based SBCs  
Microchip: Programmable Controllers

## Intel 4004:

- 108KHz
- 0.06 MIPS
- 256 Byte ROM
- 40 Bytes RAM
- 2,300 transistors

## Intel Current Generation: P6 SBC

- 2 GHz
- GIPS
- 10\*\*8 RAM
- 5M+ transistors



## Microchip Controllers:

- 40MHz
- 512 - 64K words ROM
- 20 - 30K RAM
- inexpensive

## Technology

- High Performance – small package
- Very low cost
- The concepts are different, but the issues are unchanged

## Architecture as leverage

- The classic nested diagram is highly misleading
- Architectural elements are frequently independent of each other
- Exponential reduction in code
- Exponential reduction in interaction analysis

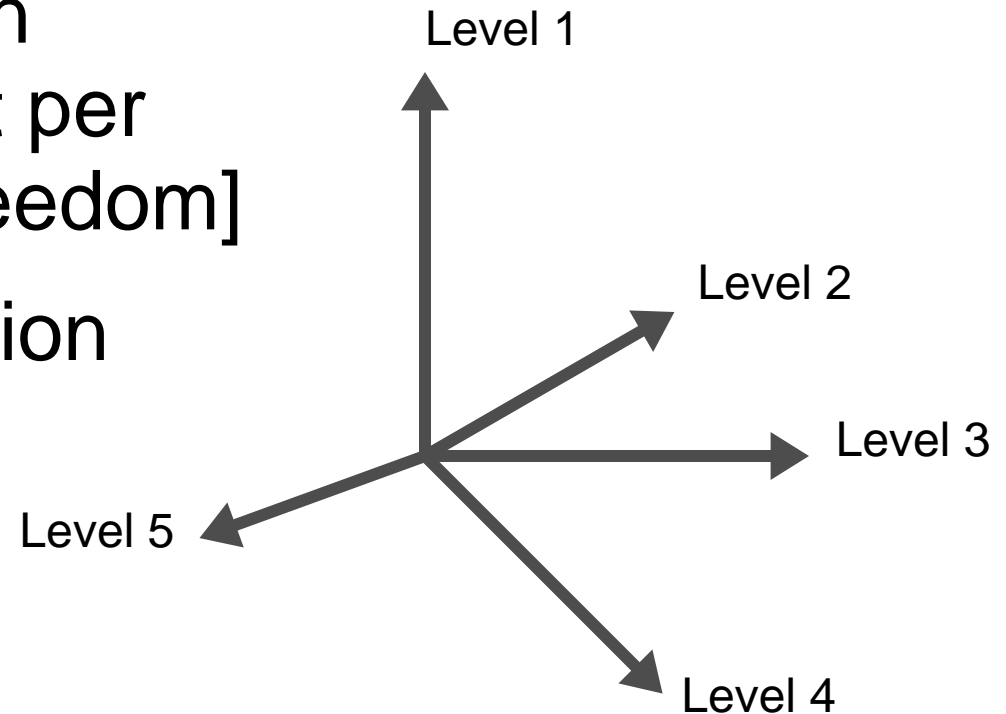
## Tools and Diagramming Drive Designs

- Classic Stack – ISO Open Systems Interconnect model
- One dimension – obvious
- Two dimension – difficult to visualize
- Multi-dimensional – almost impossible to visualize or discuss



## Vector “Degrees of Freedom” Diagram

- ok, my nomenclature (Gezelter, 2004)
- each level is independent
- a full implementation needs  $\geq 1$  element per vector [degree of freedom]
- freedom of substitution

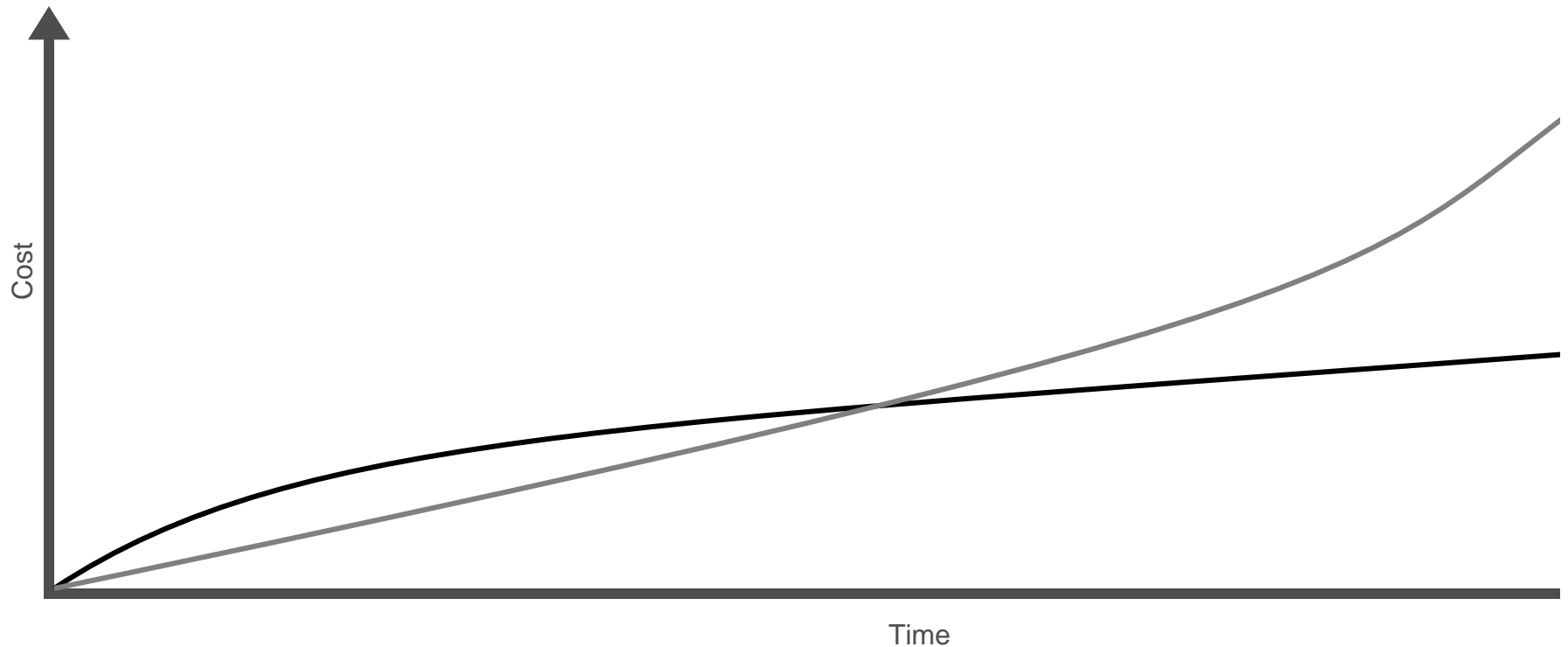


## Classic Representations are Misleading

- classic nested rings are two-dimensional
- accidental interdependencies

## Aesthetic Costs over Time

- reduced interaction effects
- assimilating increasing scope



## The Requirements Trap

- Immediate Necessary is not e.e.  
Long Term Sufficiency
- Change is inevitable



## What is most important?

- Clarity of expression
- Future flexibility
- Defined growth
- Subsetting is safe
- Supersetting inevitably leads to incompatibilities

## What facilities are important?

- Device Interfacing
- Timers and Time Management
- Processes and protection
- Dispatching and CPU allocation

## Is there precedent?

- We have been here before
- 12/16 bit minicomputers
- 1974 – 1978
- What did we learn?
- Exotica is more important than it appears

## Two schools of thought

- minimalist – from the beginning
- large systems in miniature

## Minimalist

- based on minimal monitors
- leads to a form of recapitulation
- leads to the supersetting problem
- the “Stone Soup” syndrome

## Large systems in miniature

- constraints are easier to relax than tighten
- descendants of System/360 (Amdahl, et al.)

## Minicomputer Contributions

- Integration of mapping as a tool
- Separate address spaces on small scale
- loadable drivers and driver abstraction
- common synchronization mechanism
- elimination of polling
- mechanisms for layering facilities

## What is gained by discipline?

- interactions are pre-checked
- expected functionality
- pre-supplied function context
- consistent expectations



## “Trusted Computing Base”

- National Computer Security Center, 1985
- extension of concept to internal work

## An example: Timing Issues

- Interrupts fully masked
- Interrupts partially masked
- Interrupts fully enabled
- Applications processing
- example: RSX-11M/M-PLUS (circa 1978)

## Efficiency of mechanisms

- implementation is everything
- architecturally powerful facilities often take far less resources than imagined
- it is all a matter of representation

## Questions?

Robert Gezelter Software Consultant  
35 – 20 167th Street, Suite 215  
Flushing, New York 11358 – 1731  
United States of America

+1 (718) 463 1079  
gezelter@rlgsc.com  
<http://www.rlgsc.com>

Session Notes & Materials:

<http://www.rlgsc.com/ieee/Memphis/2005-11/index.html>