MERCURY Computer Systems, Inc.

Challenges Drive Innovation ™

Exploiting Processor Heterogeneity Through Reconfigurable Interactions

Shepard Siegel

HOT 19 CHIPS

A Symposium on High Performance Chips
Stanford University, Palo Alto, CA
August 19-21, 2007

Session Recap and Prelude

- **How what I’m going to say relates to what has been said**
  - David’s talk (ASICs and ASSPs)
  - Peter’s talk (20-year FPGA evolution)
  - I’ll share my own wisdom…

- **Abundant technological riches, but**
  - How might we wrestle the complexity?
  - Reason about difficult trade spaces?
  - Minimize time to solution, time to money?

Essential Ideas of Presentation

- **Ability to compose solutions from a collection of heterogeneous components is a value**

- **Constraints upon how components interact can improve interoperability**

- **FPGAs can complement other chip-level processors' capabilities**

Assumptions of the Problem Space

- **Concerns**
  - Throughput
  - Latency
  - Cost
  - Space, weight and power

  Commonly addressed

  - Time-to-solution
  - Reuse, scalability, tech refresh

  Historically 2nd - order

- **A Multi-chip, board, system focus**
  - More layers of physical hierarchy
  - A pure SoC approach can be less-aware

Chip-Level Processor Heterogeneity

- **Classes of chip-level processors include**
  - GPP & multicore-GPP
    - Intel, AMD, PA-SEMI
  - GPU & GP-GPU
    - NVIDIA, ATI
  - CELL
    - IBM
  - DSP
    - TI
  - Coarse-grained heterogeneous
    - Ambric, PCA (Monarch)
  - FPGA
    - Xilinx, Altera

Why (not) FPGA?

- **Why?**
  - Reconfigurable fine-grain circuit composition
    - Optimally-sized concurrent processing granularity
  - Intimate I/O ↔ Logic interaction
  - Application-specific memory hierarchy
  - Future: Time-multiplexing of the FPGA fabric
    - Reconfigurable computing's "final frontier"

- **Why Not?**
  - Spatial programming may not be appropriate
    - More on this later
  - Difficult to decouple designer roles

Divide and Conquer

- **Why**
  - Reason about and solve simpler problems
  - Compose them to attack grand challenge
  - Scalability

- **Separation of**
  - Communication and Computation
  - Application and Architecture
  - Architecture and Implementation
  - Specifier and Implementer

"Velvet Handcuffs"

- **Constraining the number of different kinds of interactions between components promotes well-defined interoperability**
  - Like-Like (obvious)
  - Like-Similar (gaskets, shims)
  - Like-Dissimilar (what where you trying to express?)

What are these Velvet Handcuffs?

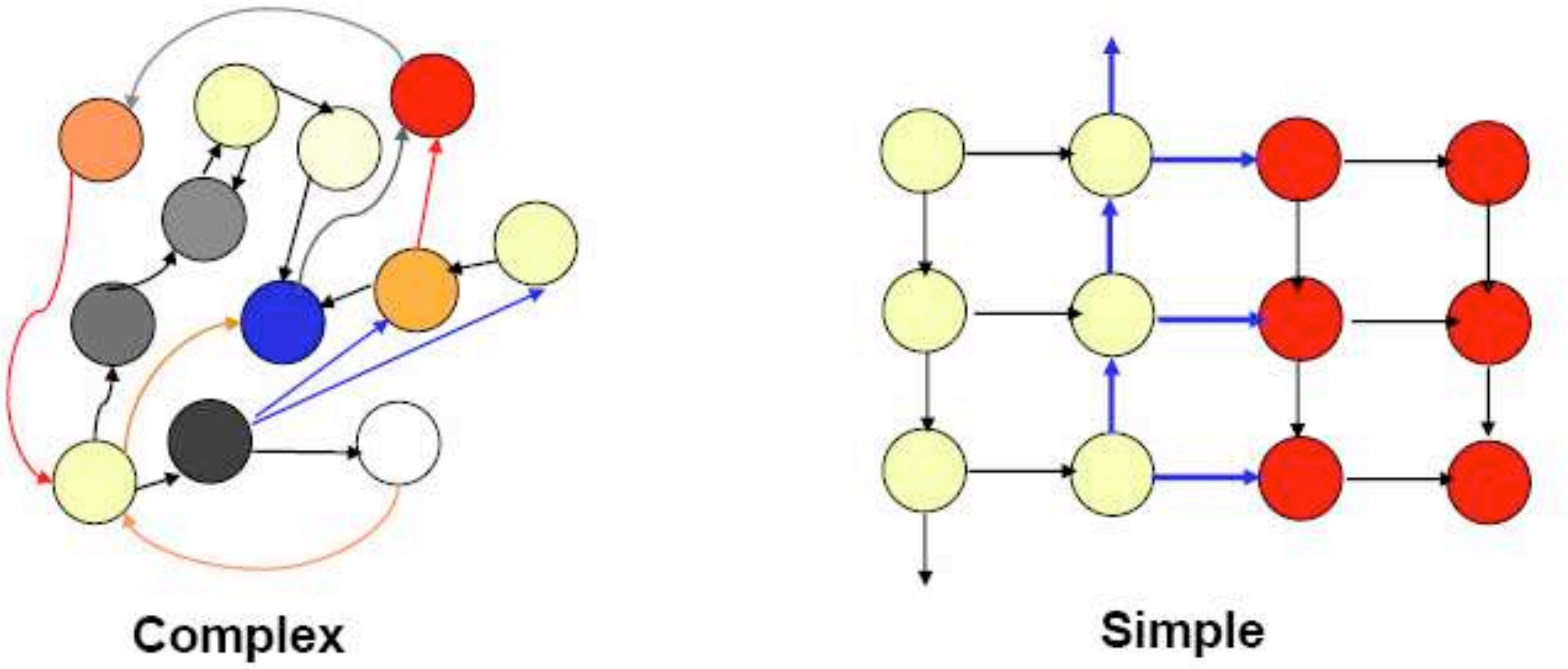
- **Profiles (templates) for**
  - Command and control
  - Data plane streaming
  - Data plane messages
  - Memory access

- **“Intention-Revealing Interfaces” [Evans 2004]**

Fewer Kinds of Interactions are Better

Complexity increases with:

- # different components (not # of components!)
- # different interactions (interconnect, protocol, types)
- Lack of structure in interactions



[DeMan 2002]

Levels of Integration

- **Speak to design at different scales**
  - System
  - Board
  - Chip
  - Infrastructure + application
  - Application assembly
  - Individual IP worker

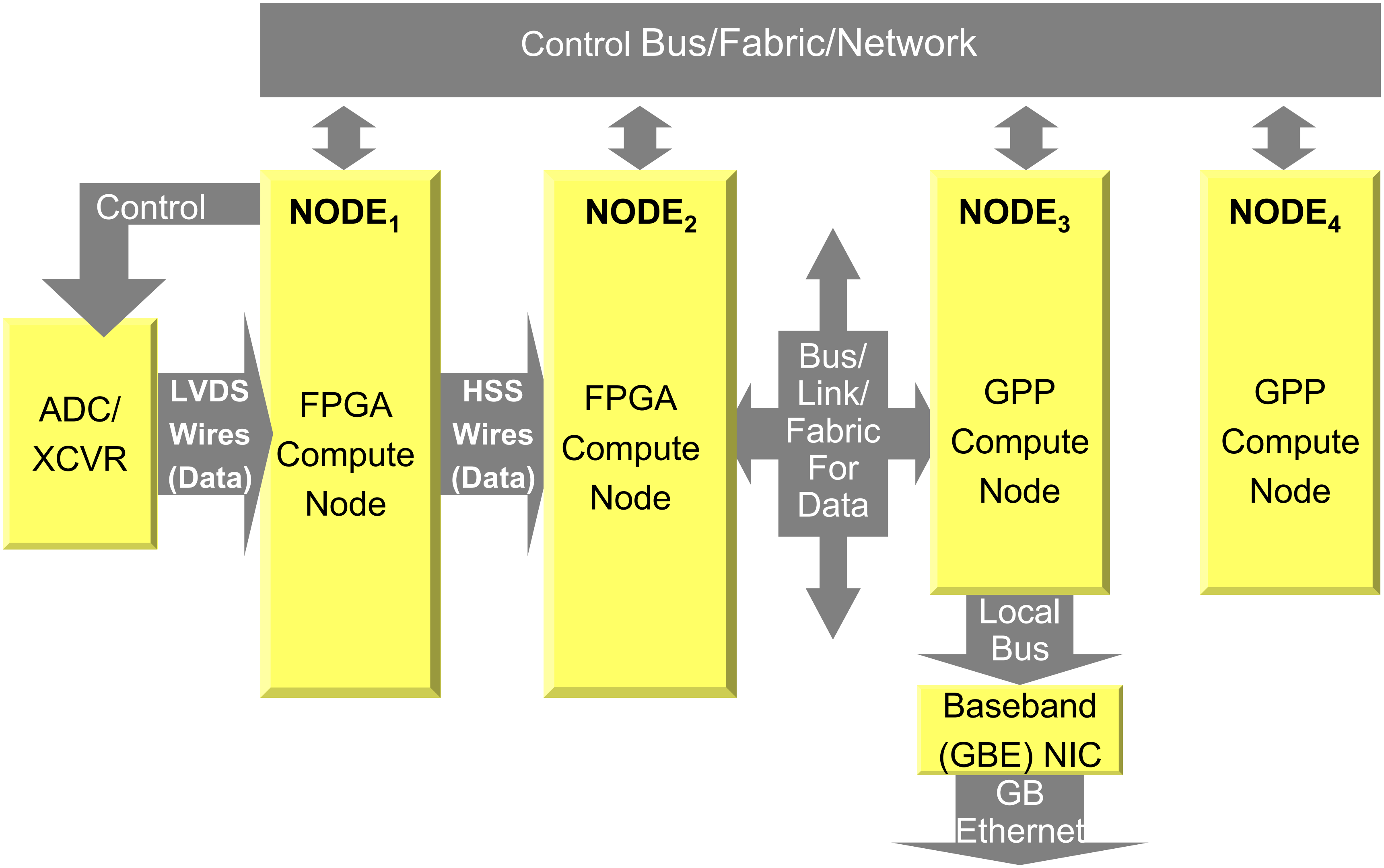
Goals in Abstraction

- **Less coupling between IPs**
  - Workers “work”
  - Infrastructure “copes”

- **Facilitate interchanging components**

- **Provide for (virtual) application-specific platform**
  - Continuous application evolution
  - Continuous architecture evolution
  - Continuous verification refinement

Heterogeneous System of Processors Example



From Architecture to Application Specific

- **Adjust our view of the platform so the application is specifying and the architecture is coping**
  - Favors the value of application over the architecture
  - Future tech refresh is driven by application specification
  - May be less-obvious to map new architecture features to application

An Example

- Specific one, if time permits
- I’ll try to illustrate these points through my own experience

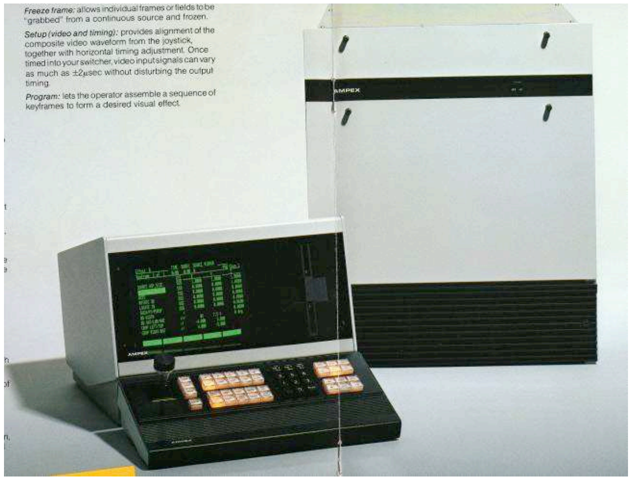
One Engineer's “Arc of Enlightenment”

Author in Ampex ADO Lab (*ca.* 1983)

- **Education**
  - Math + Science = Engineering
- **Ampex**
  - Math + Science = Money, Fame
  - Programmable logic emerges
- **Datacube**
  - Best engineering! = Best business
    - Point-hardware + Point-software = Vulnerable, Niche
  - Programmable logic everywhere
- **Mercury**
  - Hetero processor heritage
  - The cone-beam backprojector story
    - FPGAs don’t have “x86” binary compatibility!
  - Continuous evolution
    - Sustained value in balanced abstraction

Ampex Digital Optics (ADO)

- **Small team in established company dominates Digital Video Effects (DVE) market**
  - Team of youngish engineers (30 was “old”)
  - Company planned on product failing
  - ~ $500M revenue on great margins
  - 27 PWBs, 200A at 5V, 1KW/channel



Freeze frame: allows individual frames or fields to be "grabbed" from a continuous source and frozen.

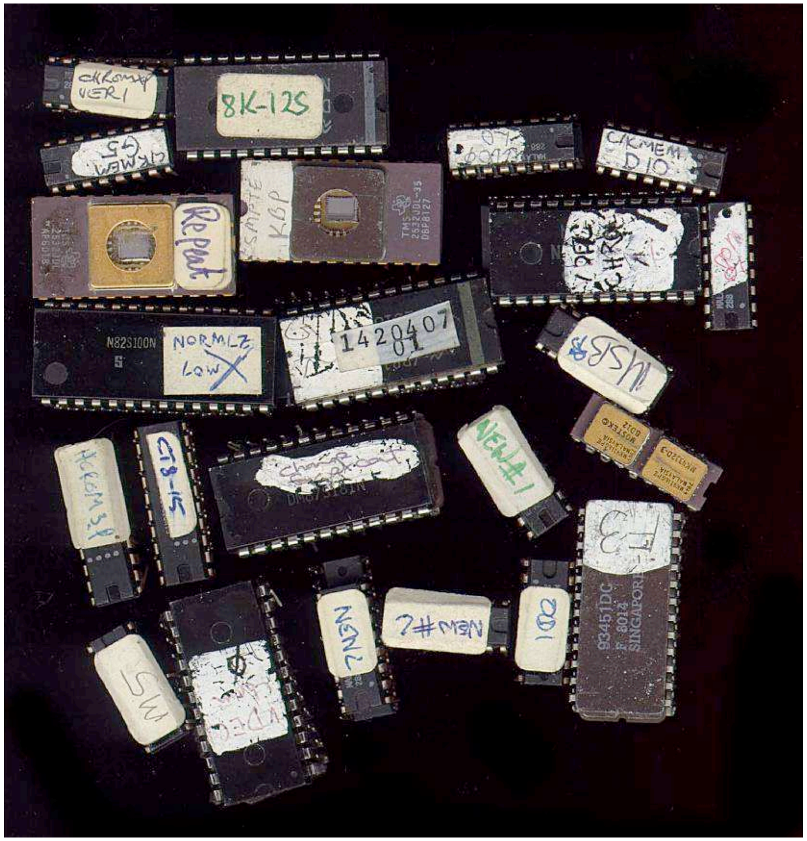
Setup (video and timing): provides alignment of the composite video waveform from the joystick, together with horizontal timing adjustment. Once timed into your switcher, video input signals can vary as much as ±2µsec without disturbing the output timing.

Program: lets the operator assemble a sequence of keyframes to form a desired visual effect.

- **Key technical contributions**
  - Separability of perspective equations [Gabriel 1983]
    - N^2 problem in 2N cost, two serial 1-D problems
  - Transposing field stores (~ 3MB DRAM)
  - TRW 8-bit MPYs → 8-point linear-phase fidelity
  - TRW 16-bit MPYs → denominator of perspective equations
  - Keyframe animation with spline interpolation

ADO Firmware

- **Z8000s for HLC, LLC, CPP, Z80 Keyboard**
  - C and assembler
- **Many PROMs as part of 13.5 MHz signal system**
  - C/UNIX tools to generate
  - Some by hand
  - Mostek 1K x 8 (popular)
- **First PAL**
  - MMI 16R8



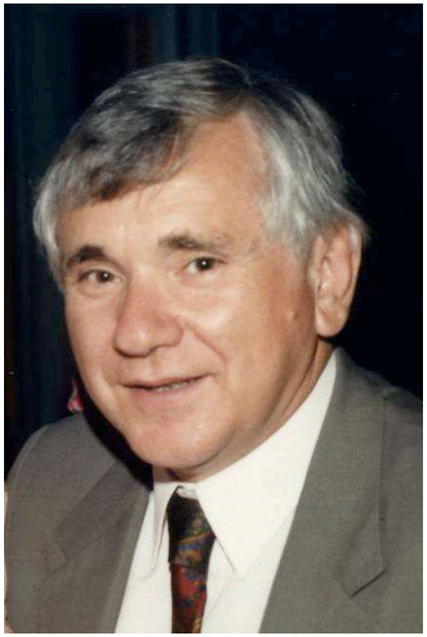
ADO Team

ADO Success

- **Emmy award**
- **“Fame”**
  - Charlex SNL Opener
  - Too many MTV videos
  - Far too many beer commercials
- **Some left to try to repeat this pattern elsewhere**
  - Microsoft
  - Abekas
  - Datacube…

Datacube Inc.

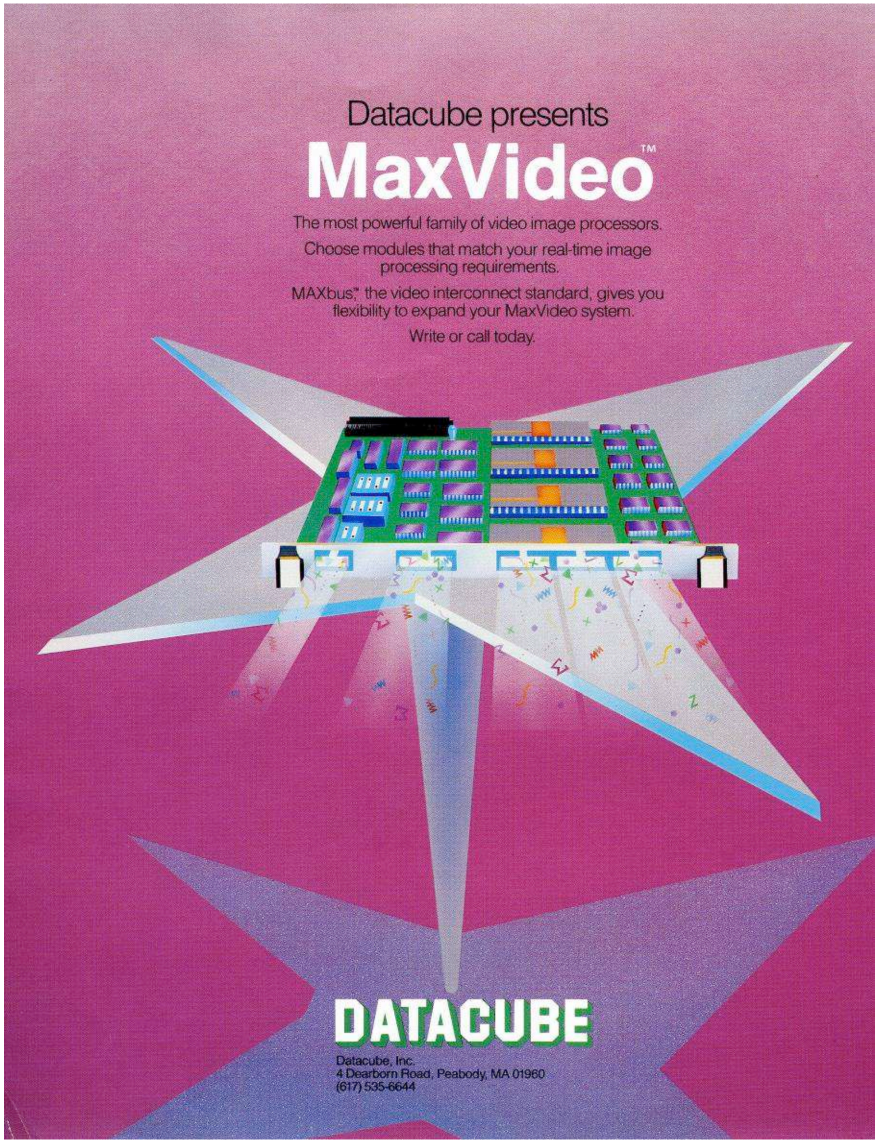
- **Evolved from "frame grabbers" to image processing**
- **Charismatic and entrepreneurial founder, Stan Karandanis put few constraints on engineers**
  - Talk to your customers, constantly
  - Create things of value; not just because they are technically feasible
  - Lean forward, take risks
- **Petri dish for programmable logic**
  - PALs, ASICs, FPGAs



Stanley Karandanis (1934-2007)

Datacube’s Zenith - MaxVideo

- **Late 1980s, early 1990s image processing hardware dominance**
  - Modular 6U VME PWBs
  - 10, 20, 40 MHz heavily-pipelined integer vector processors
    - SHEP = “*Should Have Everything Pipelined*”
  - Transitioned from PALs to FPGAs and ASICs
  - Application-specific connectivity
  - High-bisection bandwidth
    - The blue cables



Datacube presents

MaxVideo™

The most powerful family of video image processors.

Choose modules that match your real-time image processing requirements.

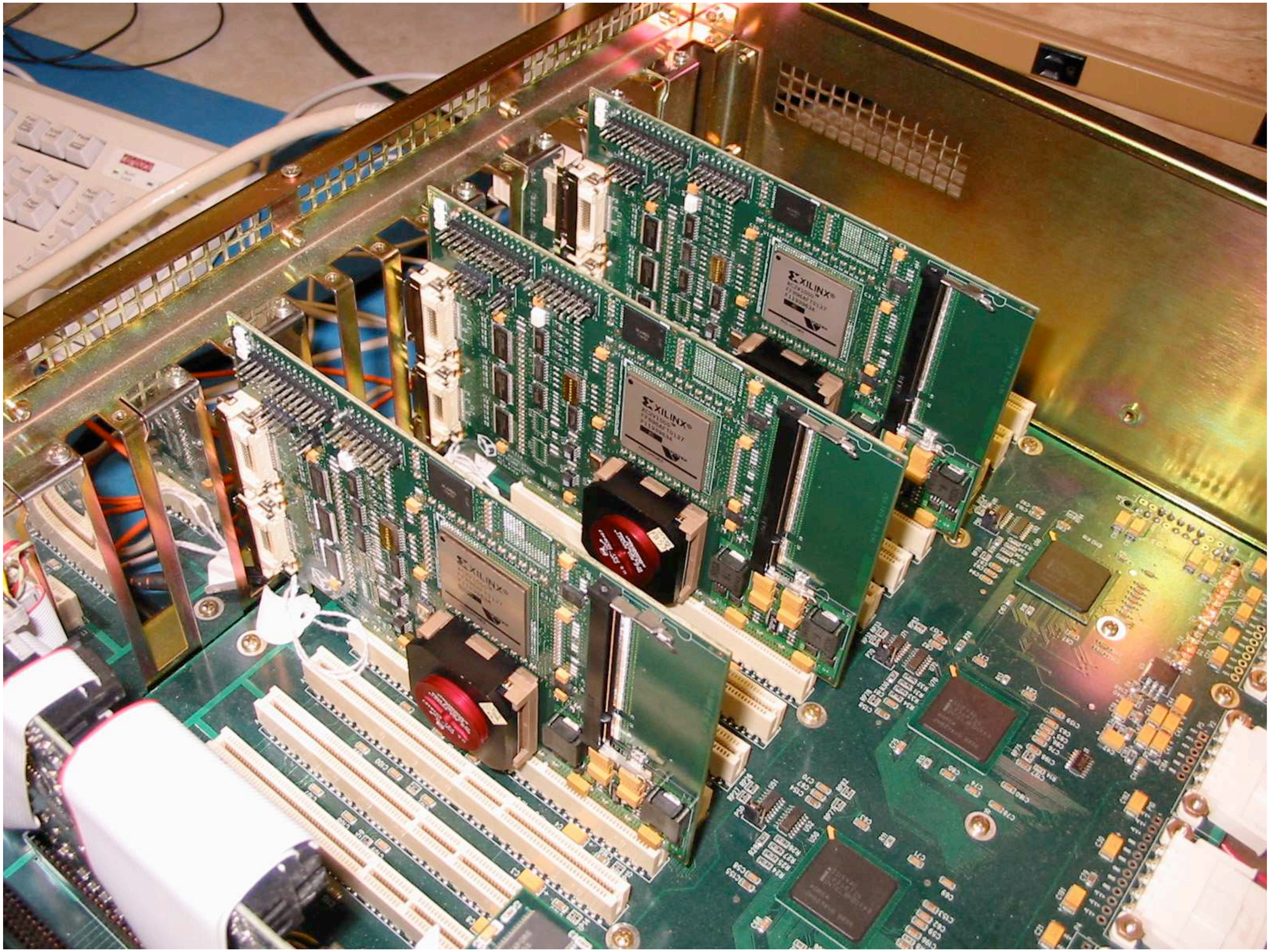
MAXbus,™ the video interconnect standard, gives you flexibility to expand your MaxVideo system.

Write or call today.

DATACUBE

Datacube, Inc.
4 Dearborn Road, Peabody, MA 01960
(617) 535-6644

Datacube MaxRevolution (late 1990s)



Datacube

- **Marketed vector processors that outperformed pure software solutions**
    - Specialized “MaxVideo” hardware
    - OO “ImageFlow” software

- **Ultimately, software alone became “*good enough*” in most of Datacube’s markets**
    - Classic Clayton Christensen “disruptive technology”
        - Too bad the book wasn’t yet written

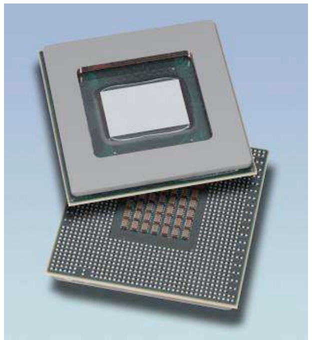
- **Point hardware + point software in niche markets**
    - Not good

…

11:01:47 [Sound of impact]

Mercury Computer Systems, Inc.

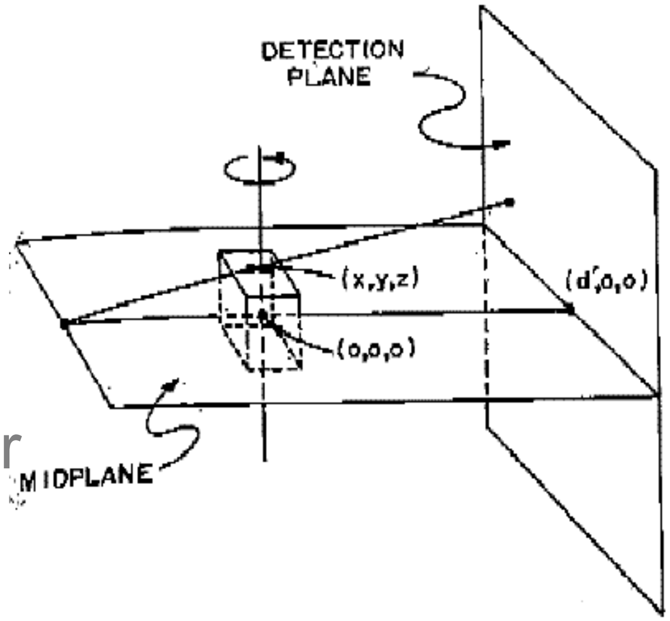
- “Gods of heterogeneity”
  - Communication 1st class citizen with computation
- Communications evolution/diversity
  - RACE, RACE++, PCIe, RapidIO, Ethernet
- Processor evolution/diversity
  - Bit-Slice: 29116
  - GPP: i860, PPC, PA-Semi
  - DSP: SHARC, TI
  - GPU:
  - Cell:
  - FPGA:



Mercury

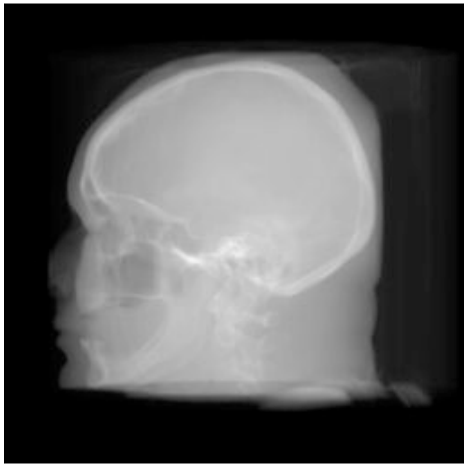
- **Planned in late 1990s to make FPGAs a peer-processor within the multicomputer**
  - Staffed and invested to achieve that goal
  - Before that, used FPGAs as ASICs/ASSPs

- **Backprojector has a story**
  - Feldkamp cone-beam similar to Radon transform (Feldkamp 1983)
  - Opportunity to exploit FPGA application-specific memory hierarchy
  - Opportunity for 10x speedup over GPP solution

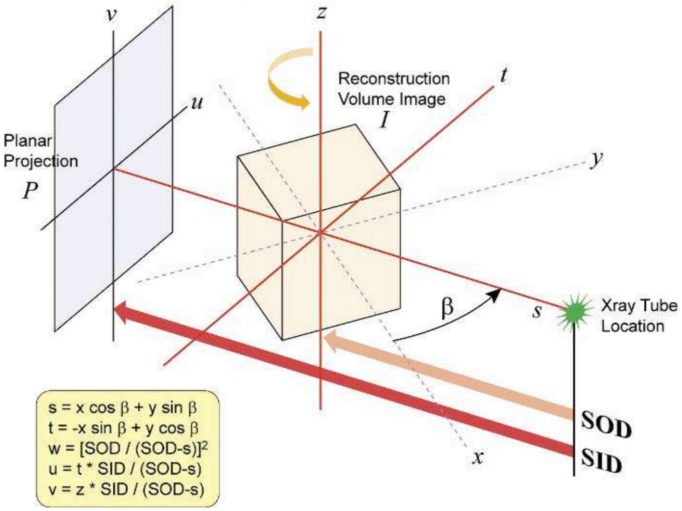


Mercury Backprojector

Sample 512^2 Projection Frame

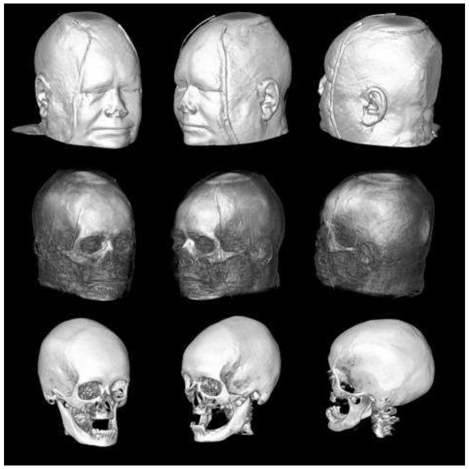


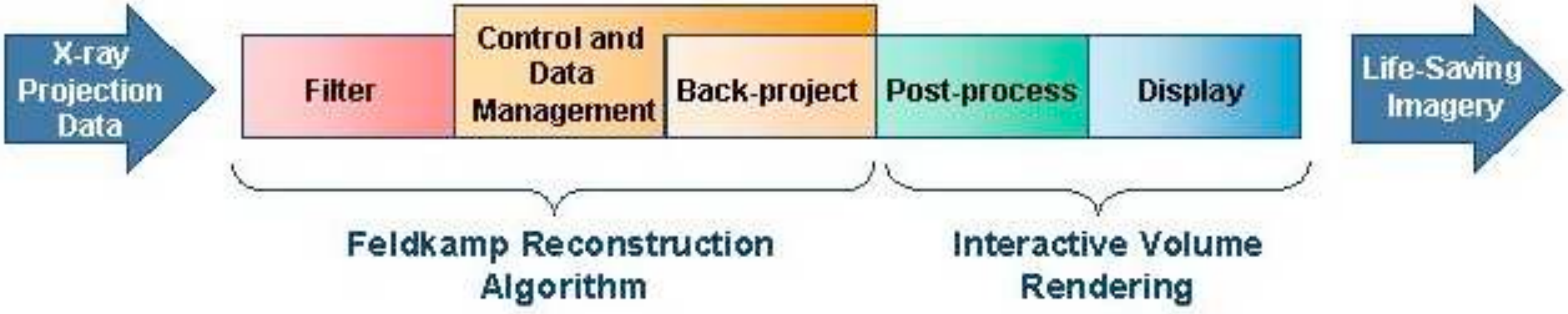
$$I[x, y, z] = \sum_{\beta} P[u(x, y, \beta), v(x, y, \beta), \beta] \bullet w(x, y, \beta)$$



$s = x \cos\beta + y \sin\beta$
$t = -x \sin\beta + y \cos\beta$
$w = [SOD / (SOD-s)]^2$
$u = t * SID / (SOD-s)$
$v = z * SID / (SOD-s)$

Sample 512^3 Volume Renderings



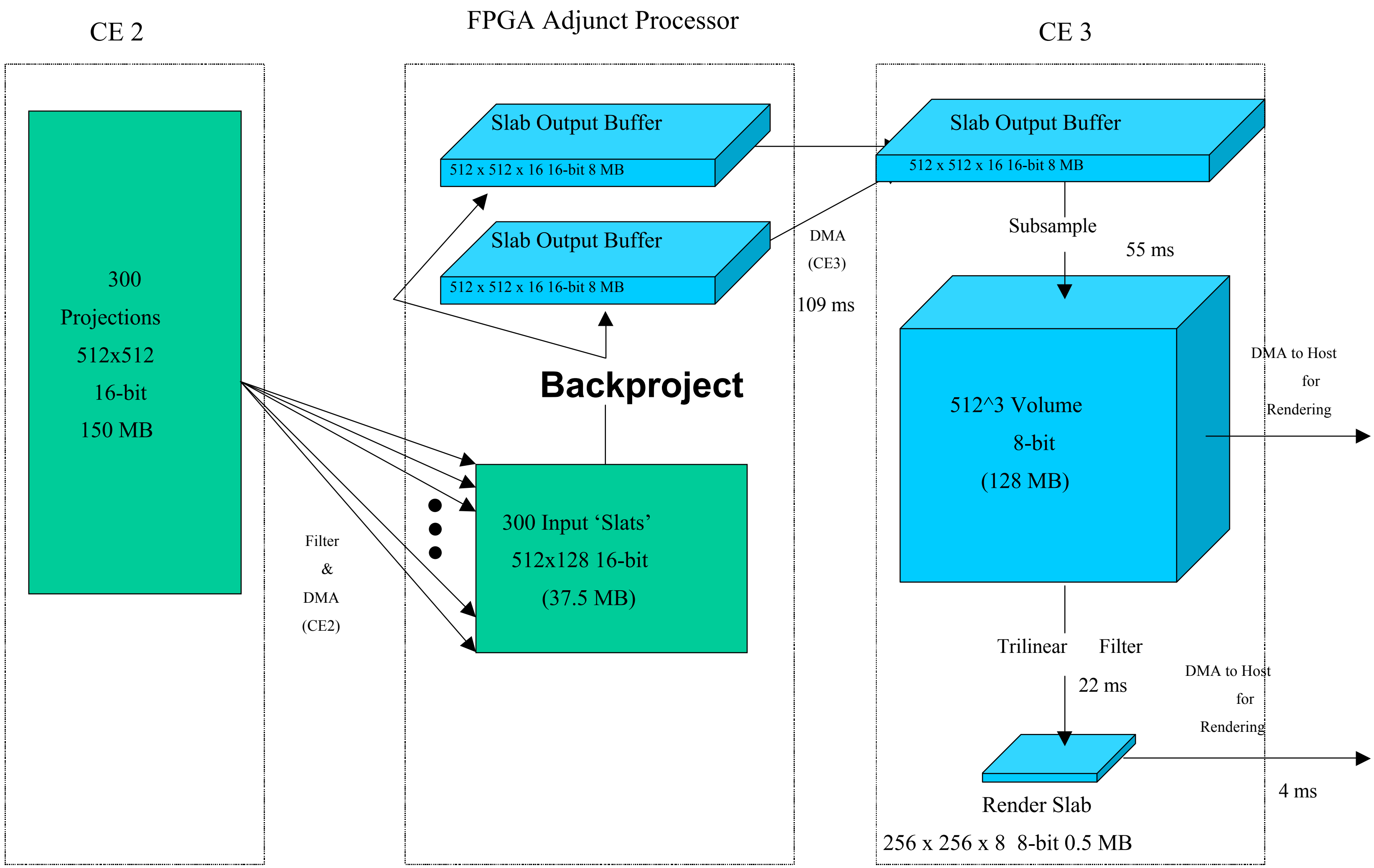


X-ray Projection Data → Filter → Control and Data Management → Back-project → Post-process → Display → Life-Saving Imagery

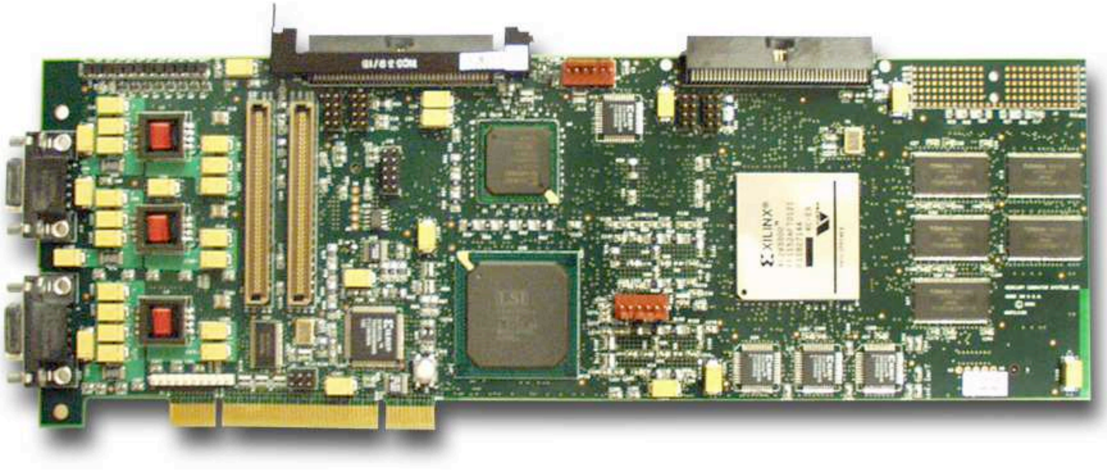
Feldkamp Reconstruction Algorithm

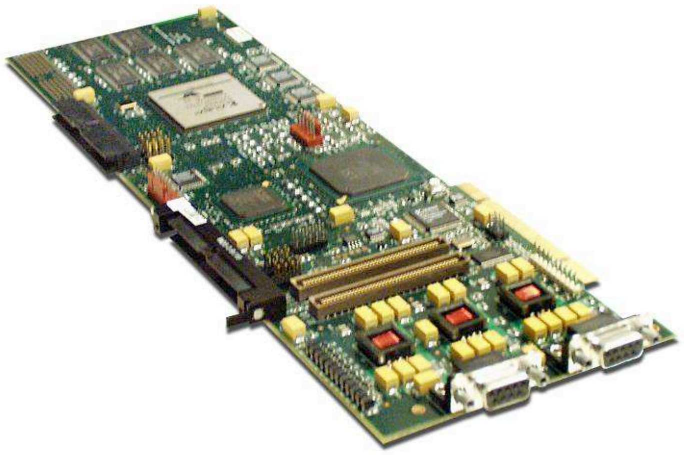
Interactive Volume Rendering

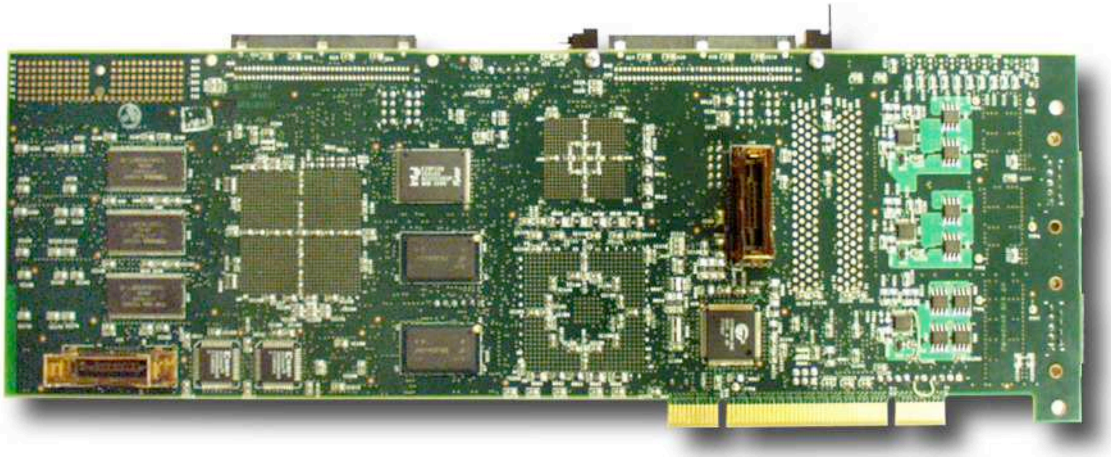
Mercury Backprojector Dataflow



Mercury AP-1 (CY2002)





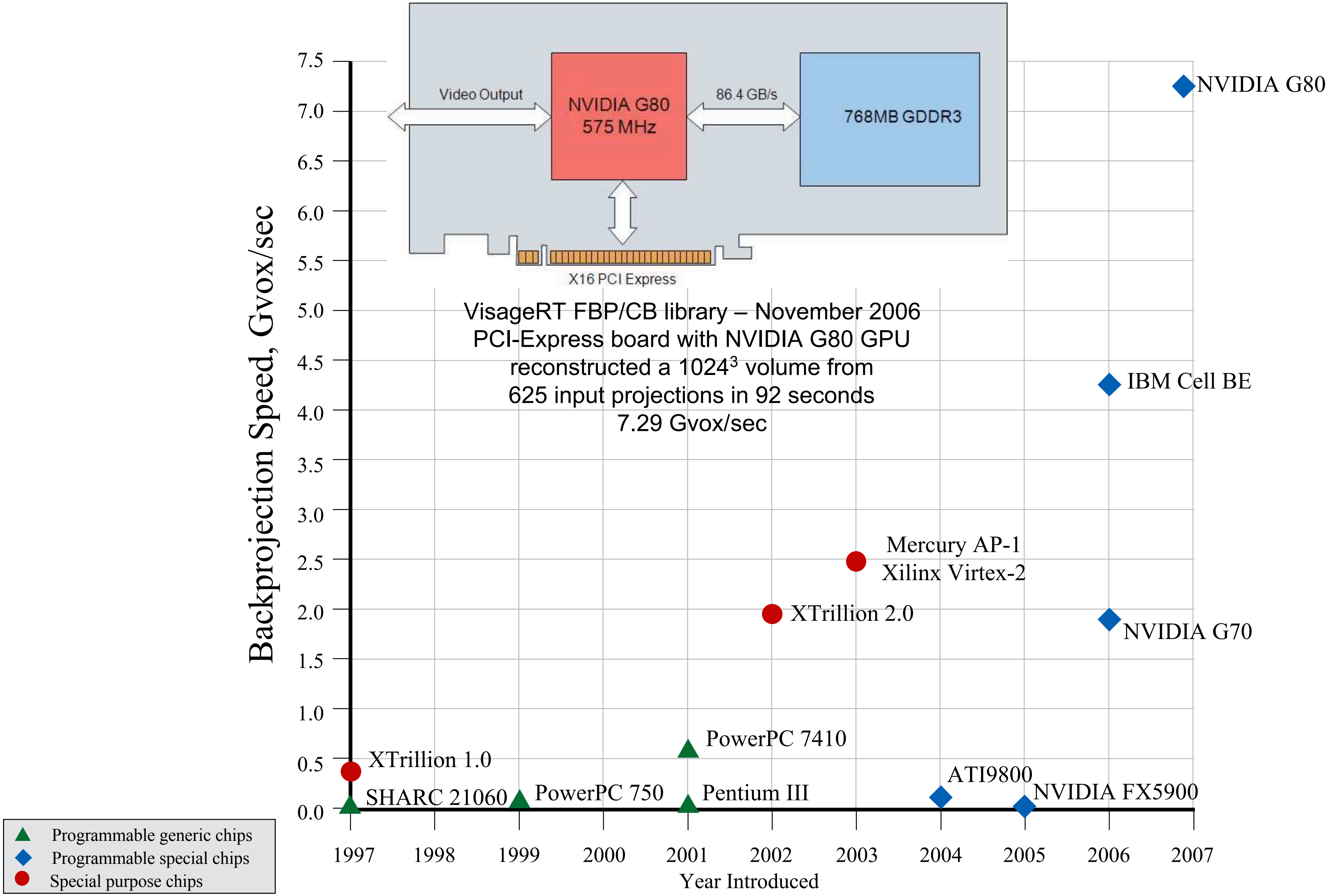


Backprojector Story

- **Technical details are swell [Bloomfield 2002]**
  - Used FPGA in synergy with PPCs
  - A well-mapped “spatial” program upon the FPGA fabric
  - All in all, a textbook FPGA application example by technical merit

- **But…it was later shown that a GPU could perform similar computation with**
  - Less engineering investment
  - More sustained value
  - Lower recurring cost

Backprojector Performance Evolution



VisageRT FBP/CB library – November 2006
PCI-Express board with NVIDIA G80 GPU
reconstructed a 1024^3 volume from
625 input projections in 92 seconds
7.29 Gvox/sec

Source: Scott Thieret

Backprojector Lesson Learned

- **GPUs didn’t kill the backprojector…**
- **A lack of concern for the costs of FPGA did**
  - Put short-term hardware uArchitecture above all else
    - Consequence: poor portability to next technology node

- **Common problem-pattern**
  - Saw it before: Datacube’s HW chased by x86 CPUs
  - We see it now: Application ***agility*** is a value

- **What can we do?**

Performance vs. Reuse

- Is “Performance” $\propto$ 1 / “Reusability”?
  - Plenty of anecdotal evidence
  - Specialization inhibits reuse

- But “Performance” is seldom the sole objective!
- What about…
  - Time to solution, time to money?
  - Sustained application value over time?

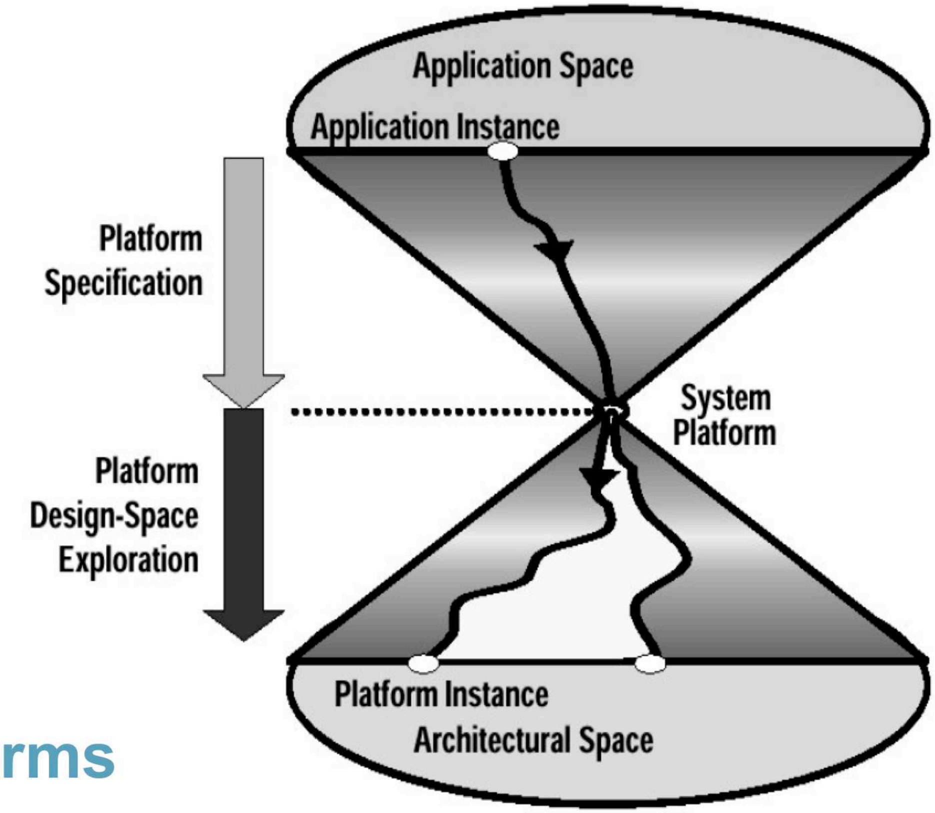
- What can be done to help this situation?

Platform-Based Design

- **Vital concept to empower and exploit "separation of concerns" [ASV 2002]**
  - M+N effort to cover MxN space
    - Effort is additive
    - Effect is multiplicative

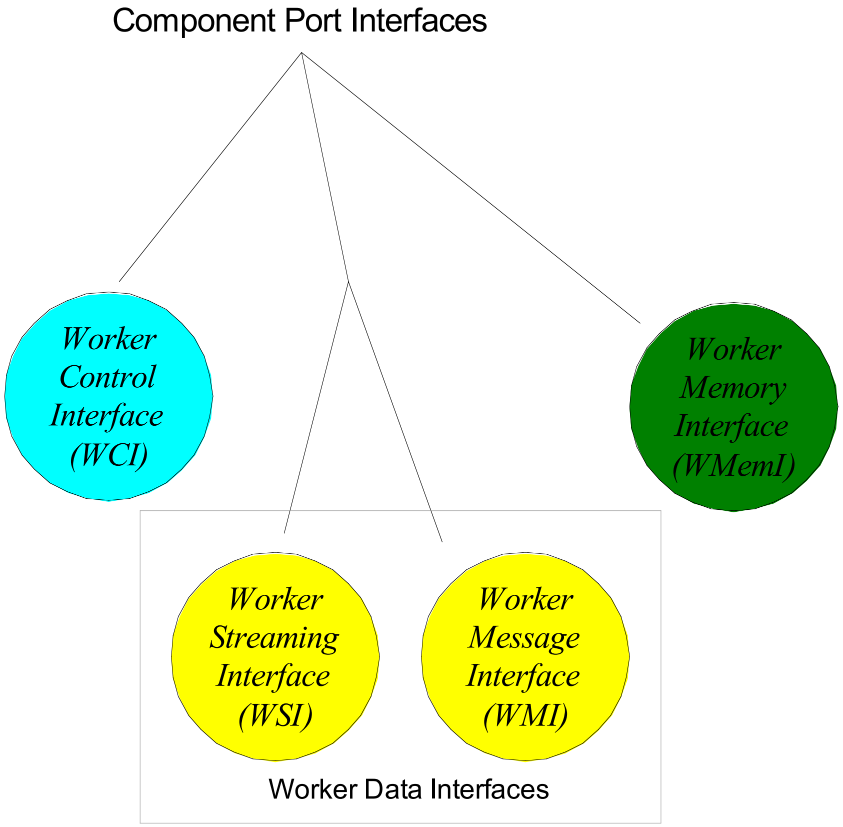
- **Consider**
  - Top-down
  - Bottom-up
  - Middle-out

- **There can be multiple platforms**
  - Amplifies the M+N → MxN leverage



Component Interfaces are Platforms

- **Most IP "worker" interfaces fall into these categories**
  - Control and configuration
  - Streaming data
  - Message data
  - Memory interaction


Component Port Interfaces
Worker Control Interface (WCI)
Worker Memory Interface (WMemI)
Worker Streaming Interface (WSI)
Worker Message Interface (WMI)
Worker Data Interfaces

- **Worker Interface Profiles (WIPs)**
  - Well-Defined Adaptability and Interoperability
  - Independence from Specific Implementations
  - HDL Language-Agnostic, HDL Language-Neutral
  - Semantics hold under Hierarchy

Why Focus on the Interface?

- “Interface before Implementation”
  is a proven software methodology

- Now is time of great change for expressing functionality, especially for hardware (e.g. FPGA) design
  - RTL has run out of runway
  - Reasoning about clock ticks at the system level is crazy!
  - Excellent candidates for a “Hardware Imagination Language” are emerging
    - Bluespec
  - Wish to avoid taking any action that would prevent the use of a particular ESL

- Open, Standard Interfaces allow us this partition

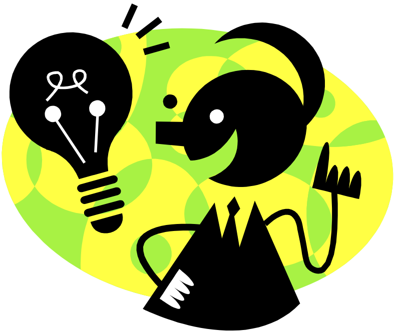
Open Core Protocol (OCP)

- **An Open Interface Standard?**
  - Yes
- **An IDL?**
  - Yes
- **A Platform?**
  - Yes
- **A Bus?**
  - No
- **Weight of Implementation**
  - None explicitly, but
- **Composable in a latency-insensitive fashion?**
  - Yes: Transactions are rule-based interface methods

FPGA Worker IP - Unit Development

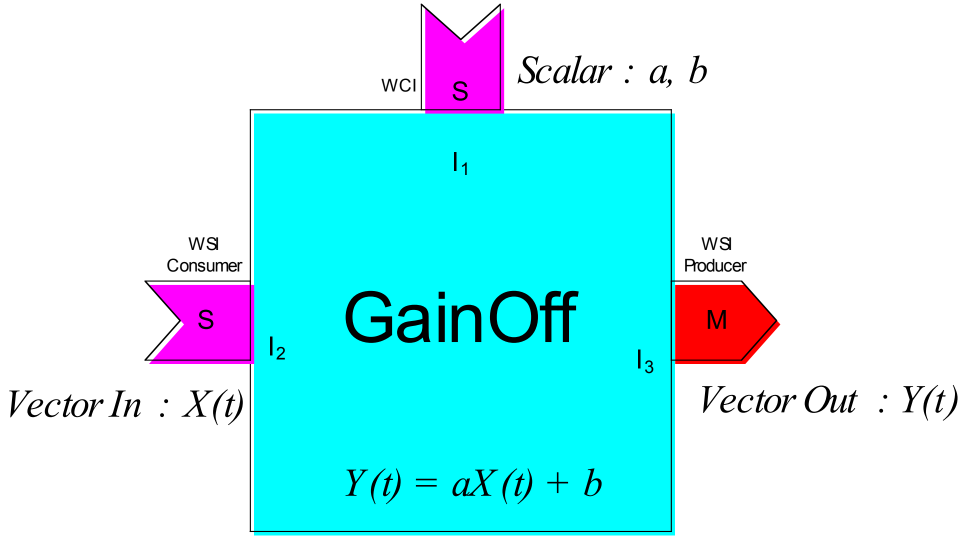
- **The Specifier defines the abstract worker behavior and functional requirements**

- **The Implementer makes additional decisions and refinements**



Gain and Offset Worker Component

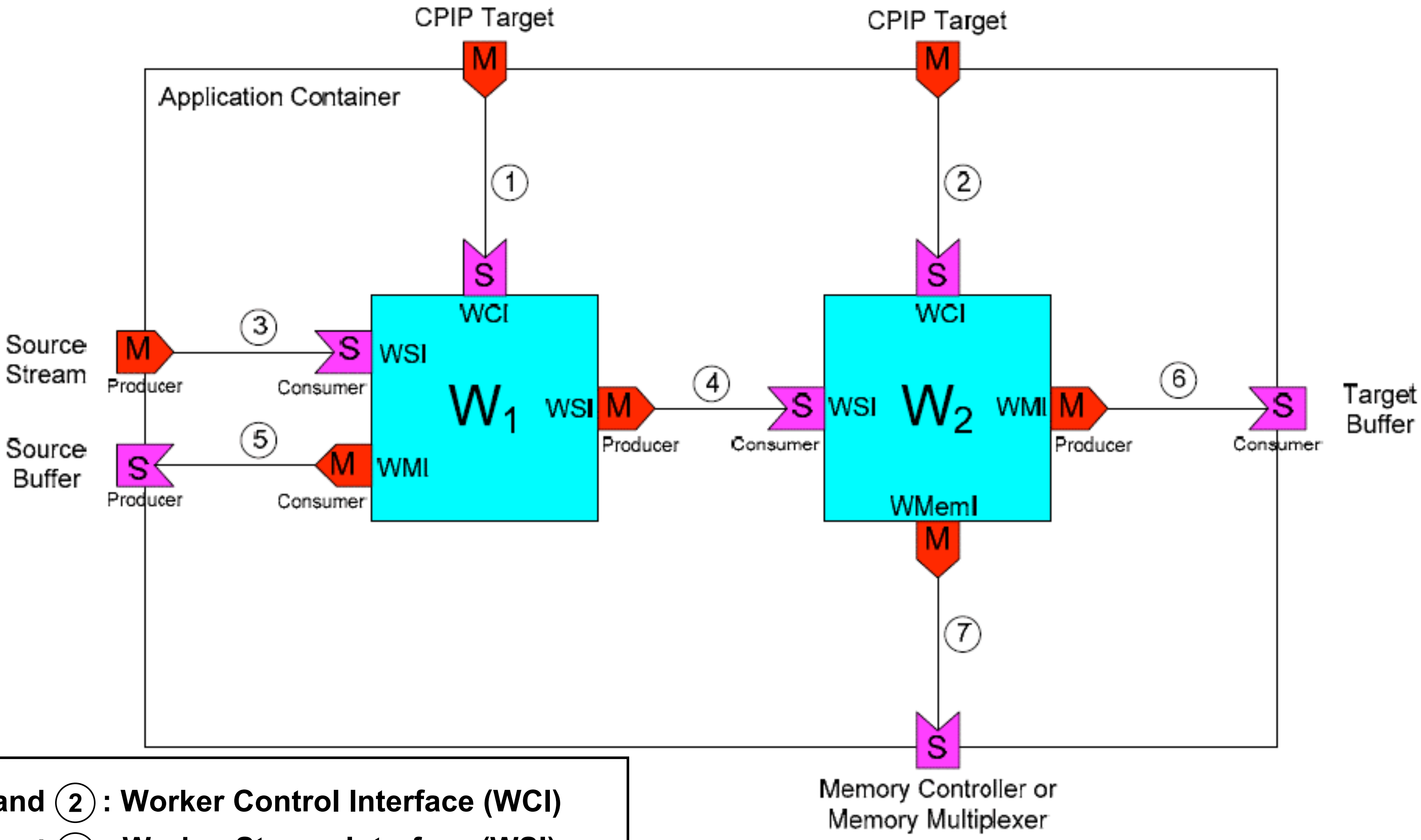
```
<Interface    >
   <SizeOfConfigSpace      >              8     </ SizeOfConfigSpace      >
   <WritableConfigProperties          >  True   </ WritableConfigProperties          >
   <ReadableConfigProperties          >  True   </ ReadableConfigProperties          >
   <Sub 32bitConfigProperties         >  False  </ Sub32 bitConfigProperties         >
</Interface    >
```



```
<Interface    >
   <DataWidth  >                     32    </DataWidth  >
   <NumberOfOpcodes >                1     </NumberOfOpcodes >
   <MessageLengthIsVariable     >    True  </MessageLengthIsVariable     >
   <MessageLengthIsExplicit     >    False </MessageLengthIsExplicit     >
   <PreciseBurstsOnly     >          False </PreciseBurstsOnly     >
   <Producer   >                     False </Producer   >
   <MaxIdleRules     >               False </MaxIdleRules     >
</Interface    >
```

```
<Interface    >
   <DataWidth  >                     32    </DataWidth  >
   <NumberOfOpcodes >                1     </NumberOfOpcodes >
   <MessageLengthIsVariable     >    True  </MessageLengthIsVariable     >
   <MessageLengthIsExplicit     >    False </MessageLengthIsExplicit     >
   <PreciseBurstsOnly     >          False </PreciseBurstsOnly     >
   <Producer   >                     True  </Producer   >
   <MaxIdleRules     >               False </MaxIdleRules     >
</ Interface    >
```

A “Worker” Component



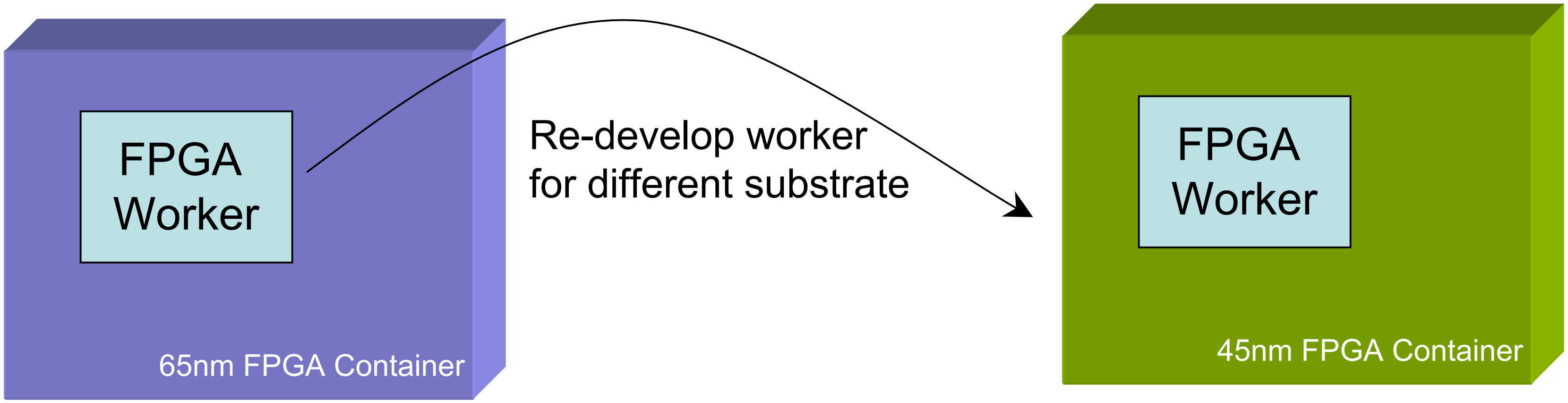
① and ② : Worker Control Interface (WCI)

③ and ④ : Worker Stream Interface (WSI)

⑤ and ⑥ : Worker Message Interface (WMI)

⑦ : Worker Memory Interface (WMemI)

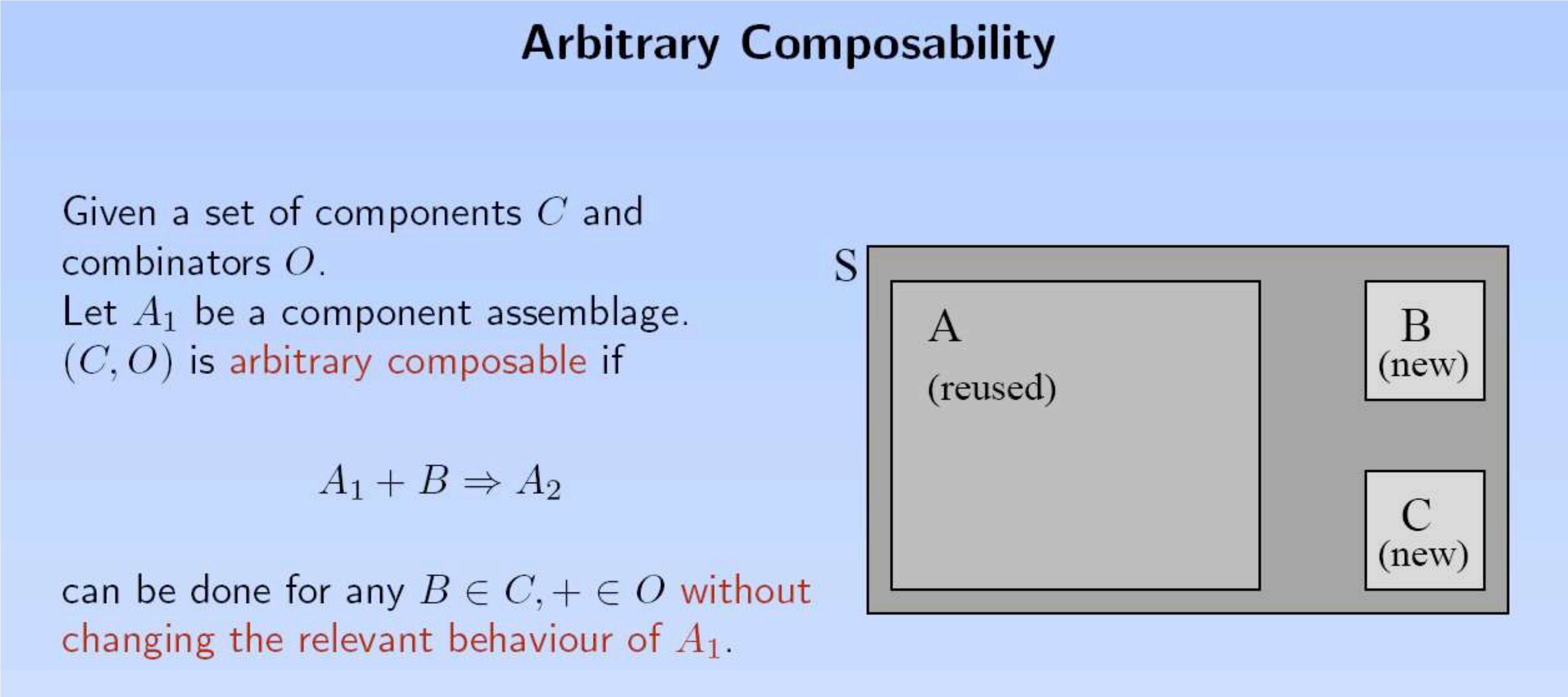
Replace-ability



- **Enables changes in technology/processor class with no impact on the rest of application (model and other components)**
- **Enables addition of component implementations to existing components**
  - Multiple implementations in a component package are possible
  - Allow adding FPGA implementation to component with GPP implementation without impacting application
- **Provide *opaque interoperability* between all classes of component implementations**

Arbitrary Composability

- **Allow the addition of a component without changing the relevant behavior of an existing assembly [Jantsch 2004]**



Arbitrary Composability

Given a set of components C and combinators O.
Let A_1 be a component assemblage.
(C, O) is arbitrary composable if

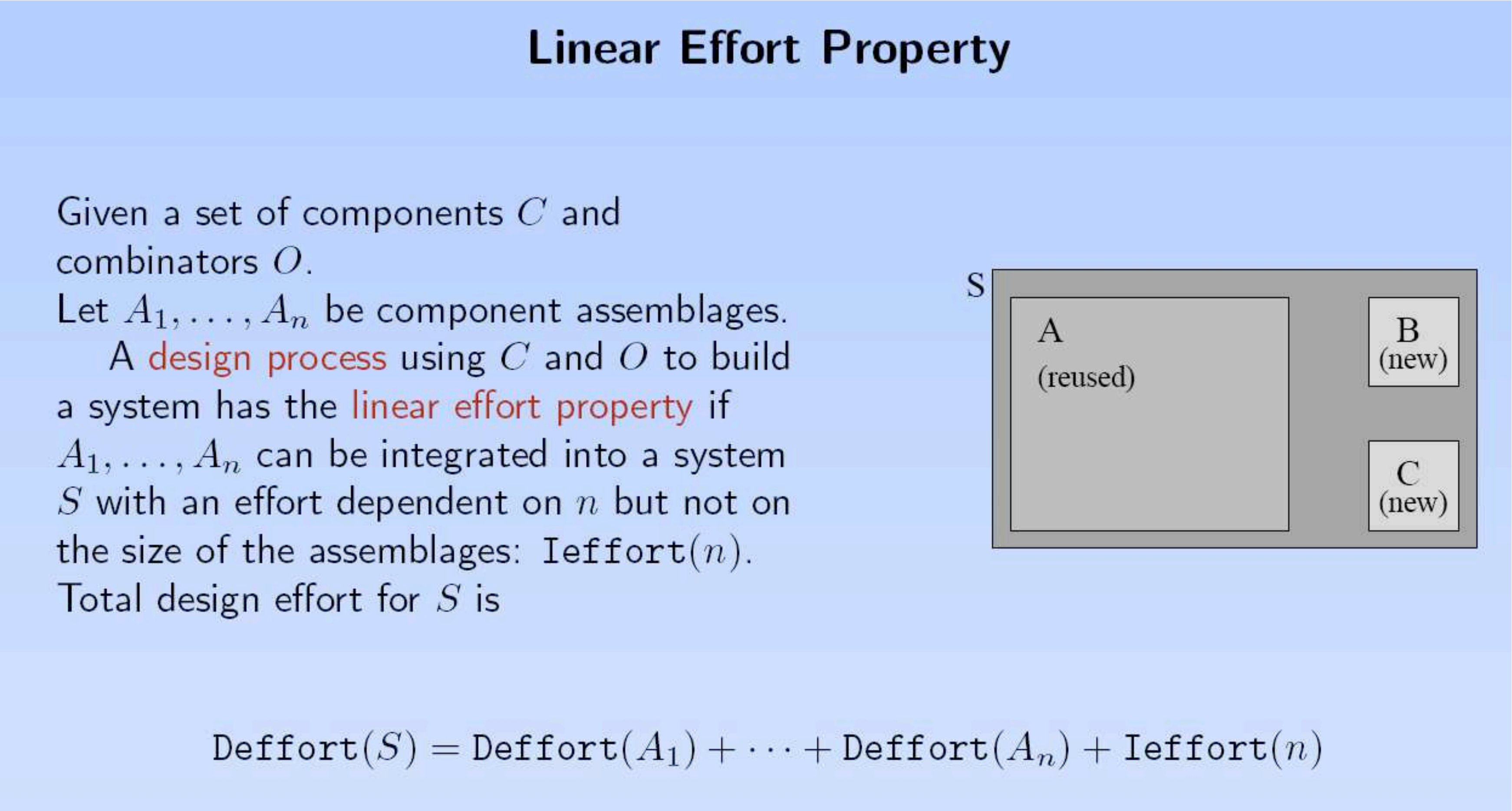
$$A_1 + B \Rightarrow A_2$$

can be done for any $B \in C, + \in O$ without changing the relevant behaviour of A_1.

S
A
(reused)
B
(new)
C
(new)

Linear Effort Property

- **Design process where effort depends on number of (not size of) assemblies [Jantsch 2004]**
  - Related to interaction-complexity



Linear Effort Property

Given a set of components C and combinators O.
Let $A_1, \ldots, A_n$ be component assemblages.
A design process using C and O to build a system has the linear effort property if $A_1, \ldots, A_n$ can be integrated into a system S with an effort dependent on n but not on the size of the assemblages: $\texttt{Ieffort}(n)$.
Total design effort for S is

$$\texttt{Deffort}(S) = \texttt{Deffort}(A_1) + \cdots + \texttt{Deffort}(A_n) + \texttt{Ieffort}(n)$$

Utopia and Reality

- **Utopia**
  - There is near-zero cost at each platform interface
  - Aggregating components is easy
  - Communication is free
  - Homogeneous model

- **Reality**
  - Interfaces frequently have some “weight”
  - Aggregation can be complex
  - Communication is seldom free
  - Processor heterogeneity exists for a reason

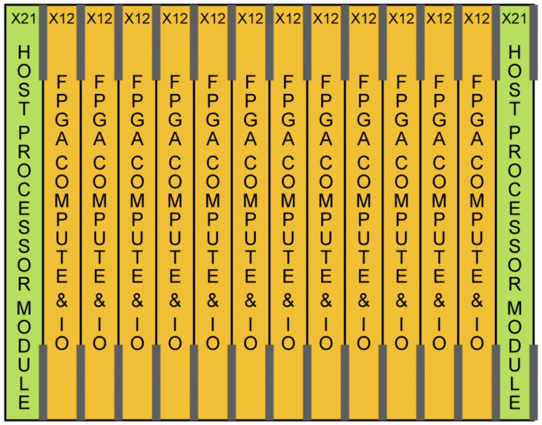
Have it All

- **Top-down**
  - Universe of application truths

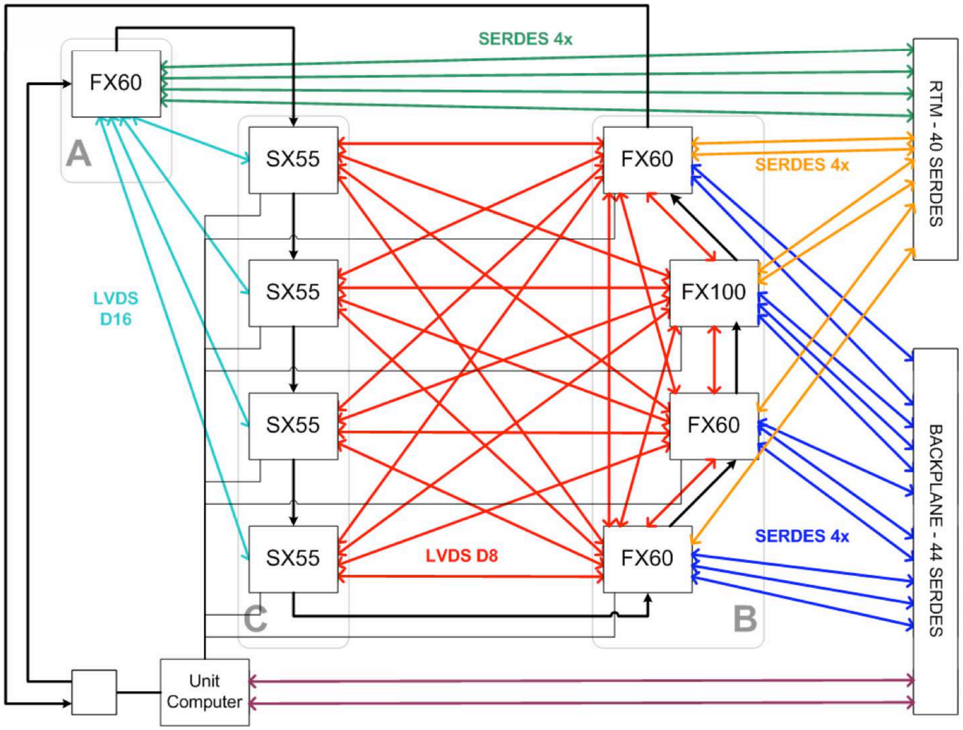
- **Bottom-up**
  - Universe of implementations

- **Middle-out**
  - Constellation of platforms in between

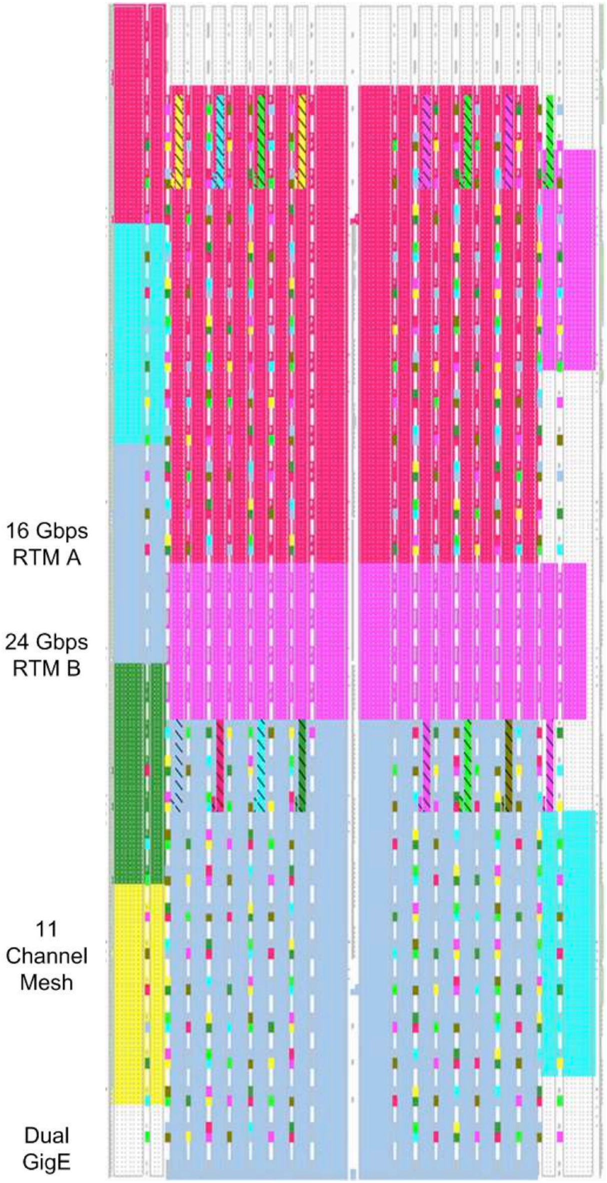
Beamformer Compute Engine (BCE)



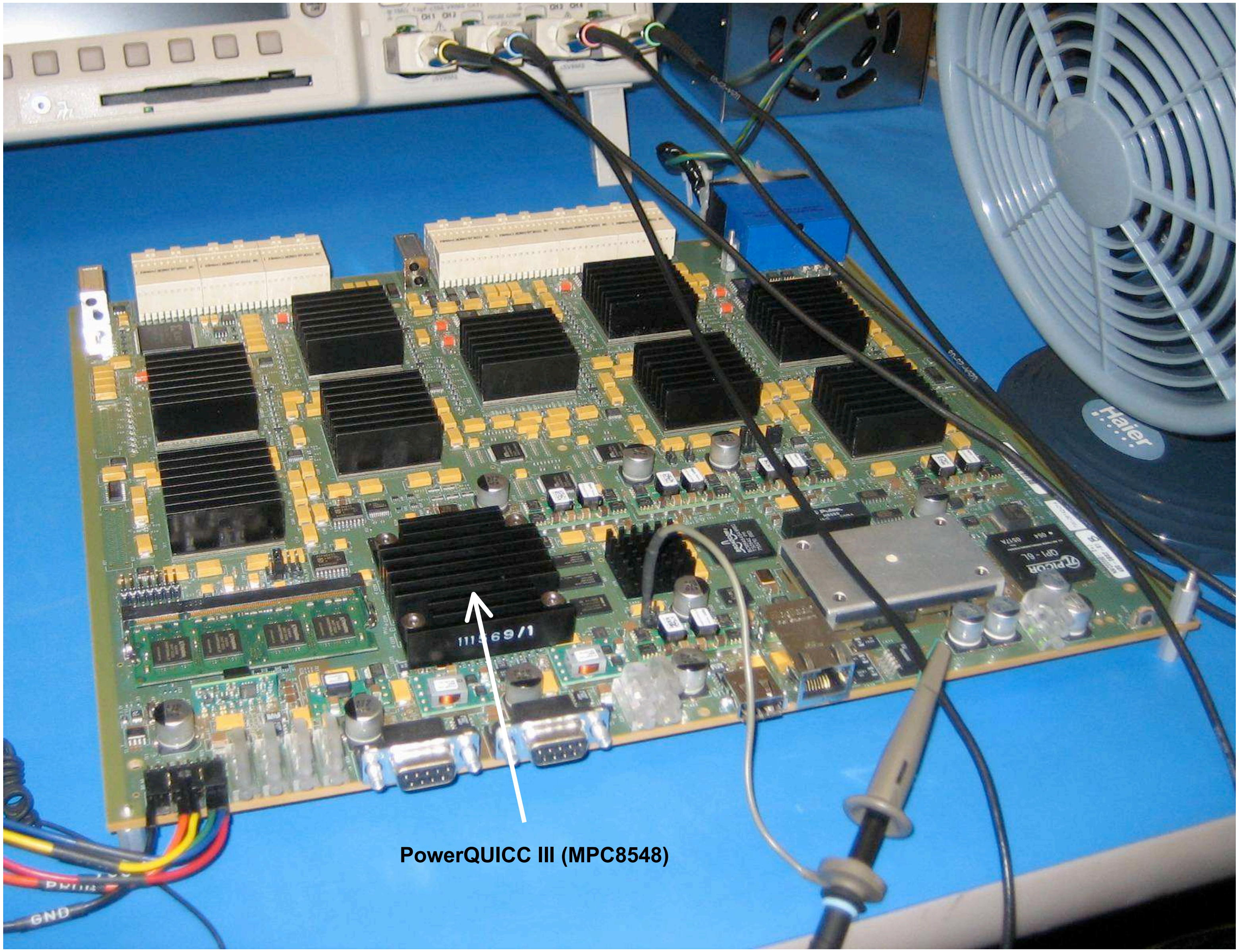
X21	X12	X12	X12	X12	X12	X12	X12	X12	X12	X12	X12	X21
HOST PROCESSOR MODULE	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	FPGA COMPUTE & IO	HOST PROCESSOR MODULE



PowerQUICC III (MPC8548)



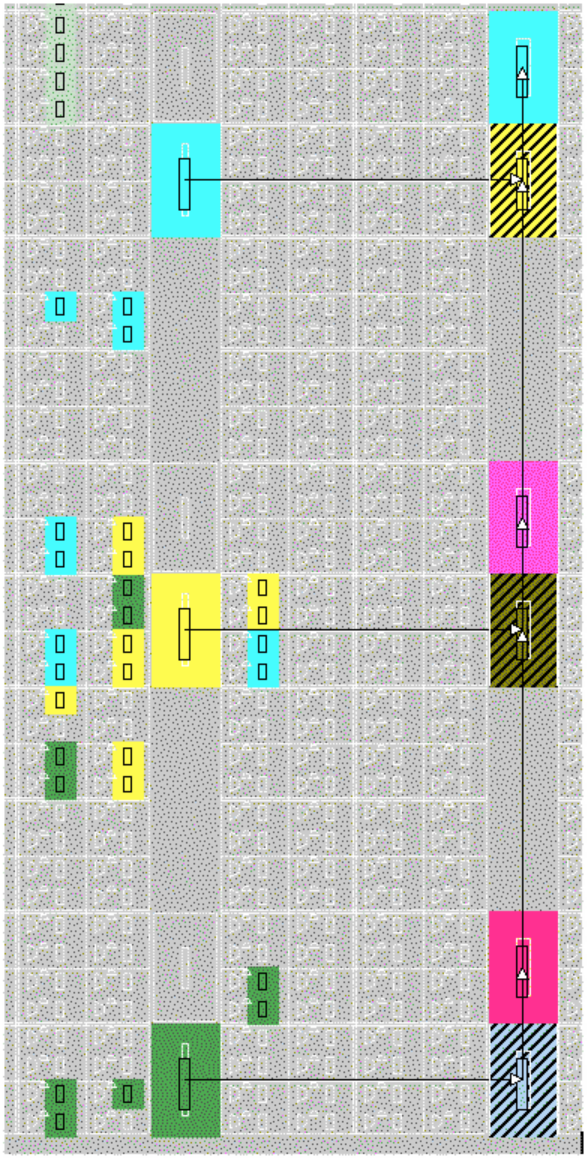
Beamformer Computer Engine (BCE)

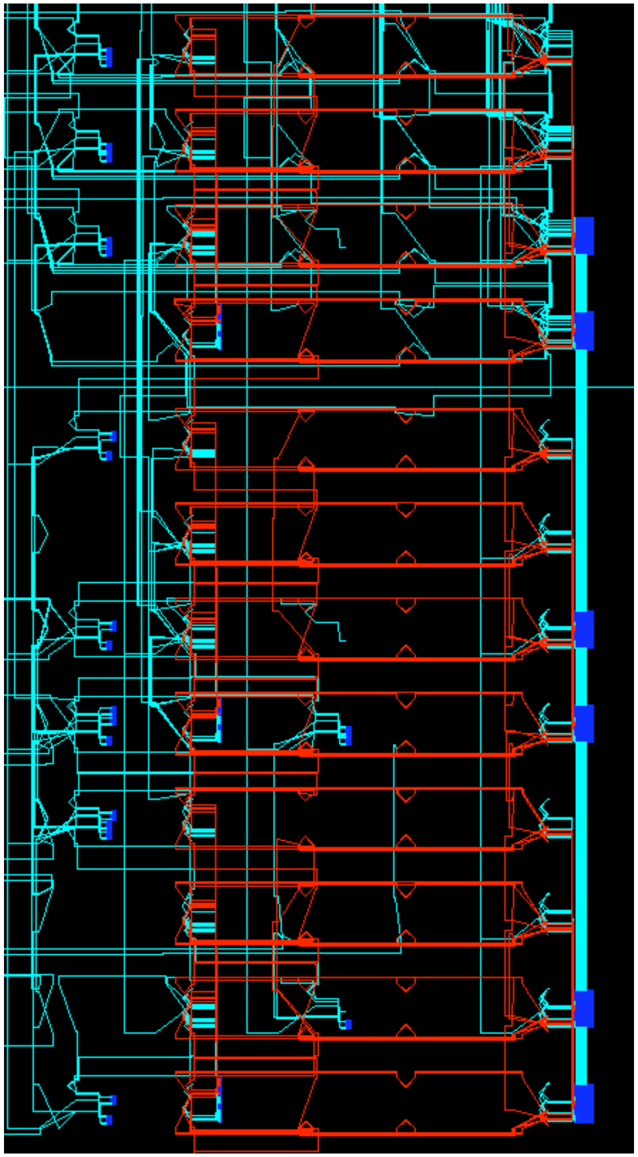


PowerQUICC III (MPC8548)

BCE Debug, Mercury Chelmsford, June 2006

CMAC Detail – (V4-SX) Mapping



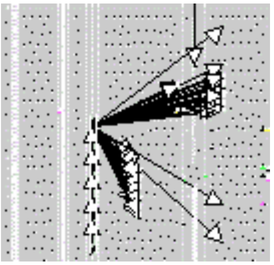


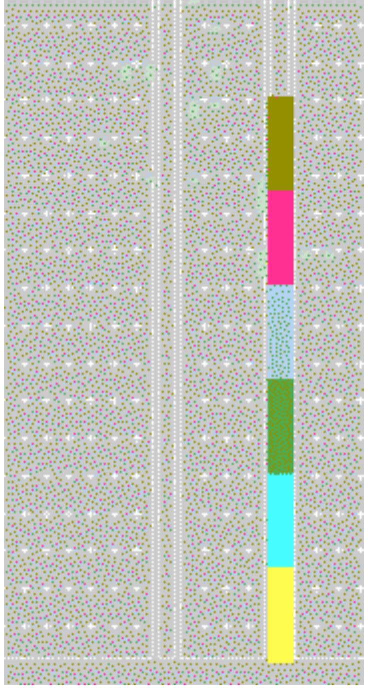
CMAC Detail – (V5-SXT) Mapping

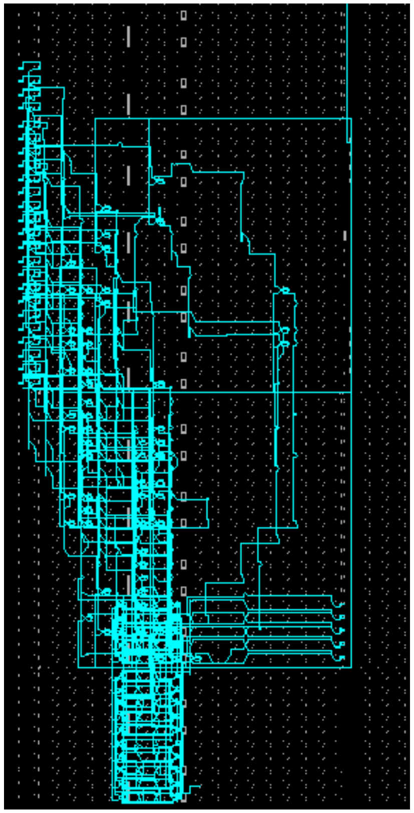
V4/V5: 100% DSP48 (DSP48E) utilization within four area groups, ~80% substrate

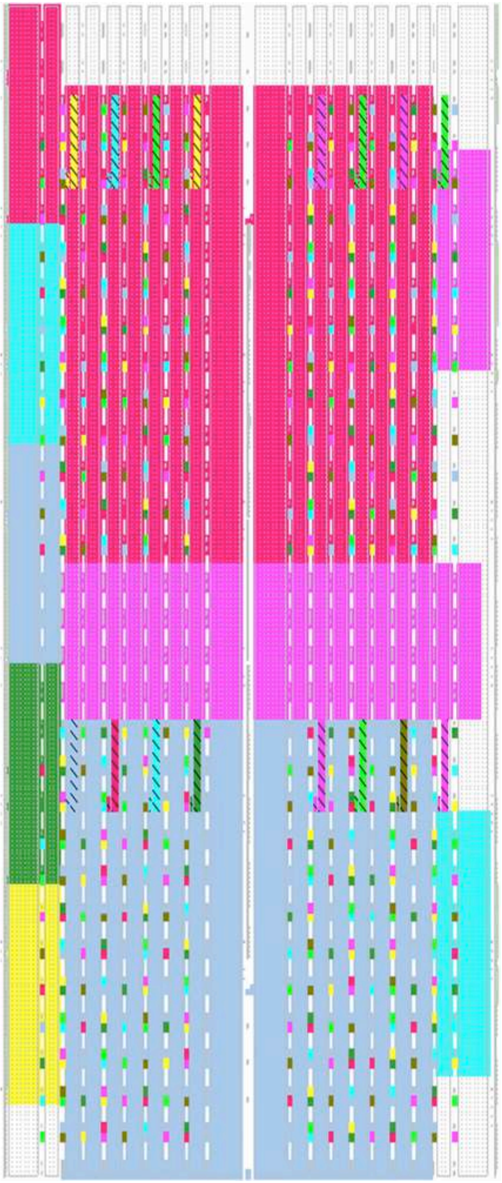
V4-SX55-10: Fmax: 400 MHz (validated)

V5-SX95T-1: Fmax: 450 MHz (estimated)









Processor Strengths / Weaknesses

- GPP, multicore-GPP
  - Familiar ISA
  - Device specific FP assist, 10s of GFLOPs
- Cell
  - Familiar ISA
  - SPEs allow 70~170 GFLOP
  - 100s of Watts
- GP-GPU
  - Increasingly-familiar ISA
  - 250~500 GFLOP
  - 100s of Watts
- FPGA
  - Reconfigurable, application-specific ISA
  - Integrated and specialized IO
  - Integer ~500 GintOP
  - 10s of Watts

Cloistered Real-World Quantitative Data

- **Although peak figures can be derived**
  - “Real-World” may suffer by orders of magnitude
    - I/O or Compute Bound
    - Weight of Implementation
  - Application-specific values are often tightly held
    - Reveal how much (or little) design margin exists

- **Reconfigurable nature of FPGAs**
  - “blessing” in temporal multiplexing of fabric
  - “curse” in obfuscating design effort involved

- **Heterogeneous Processor Comparison is Difficult**
  - Benchmarks will help

BCE Processor Selection

- **Inner-Loop CMACs, Data Reorg**
  - FPGA or ASIC required to meet SWaP goals
    - GPP, GPU, CELL lack combined processing and IO
  - Balanced Compute and Communication
  - Balance between Virtex SX and FX devices

- **QR Decomposition, Everything Else**
  - PowerQUICC PPC chosen
    - Low Risk, Fast Enough, Familiar ISA

- **Why not FPGA Embedded PPC?**
  - Too many coupled concerns
  - Little value in lower latency in this throughput-driven domain

Summary

- **The chip-board-system pattern**
  - Endured decades of use
  - Extended downward into the FPGA
    - And boards + systems composed of FPGA and other processors
  - Evolution is nearly-continuous
    - Re-evaluation of performance trades
    - Reconfiguration of IP

- **Heterogeneity is often a requirement**
  - “One processor does not meet all needs”
  - Processor evolution (tech-refresh) drives us to revisit our choices

- **Interface-centric platforms**
  - Help ease and wrestle the complexity of heterogeneity
  - Improve interoperability
  - Help limit unintended coupling → Ease reconfiguration burden
  - Don’t guarantee success by themselves, but…
  - …can smooth transition to new platforms

Thank you!

Shepard Siegel
Consulting Engineer, Technology Office
Mercury Computer Systems, Inc.
ssiegel@mc.com

References

[Bloomfield 2002] Bloomfield, John 2002 "*Partitioning Computational Tasks…,*" HPEC 2002

http://www.ll.mit.edu/HPEC/agendas/proc02/pdfs/7.3-bloomfield.pdf

[DeMan 2002] De Man, Hugo 2002 "*Design of Microelectronic Systems,*" HJ94 Lecture 2

http://homes.esat.kuleuven.be/~iverbauw/Courses/HJ94/index.html

[Evans 2004] Evans, Eric 2004 "*Domain Driven Design,*" Addison Wesley, Chapter 10

http://domaindrivendesign.org/

[Feldkamp 1983] Feldkamp, L.A. 1983 "*Practical Cone-Beam Algorithm,*" Optical Society of America

http://www.osa.org

[Gabriel 1983] Gabriel, Steven and Bennett, Phillip "*System for Spatially Transforming Images,*" United States patents 4,472,732, 4,463,372 and 4,468,688. (Assigned to Ampex Corporation)

[Jantsch 2004] Jantsch, Axel 2004 "*Networks on Chip,*" Network on Chip Seminar

http://web.it.kth.se/~axel/presentations/2004/LiTH-I.pdf

[ASV 2002] Sangiovanni-Vincentelli, Alberto Feb 2002 "*Defining Platform-Based Design,*" EETimes

http://www.gigascale.org/pubs/141/platformv7eetimes.pdf