Reconfigurable Computing: What, Why, and Implications for Design Automation

André DeHon and John Wawrzynek

June 23, 1999

BRASS Project
University of California at Berkeley
www.cs.berkeley.edu/projects/brass

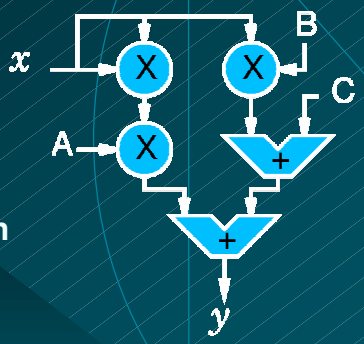
Berkeley Reconfigurable Architectures, Systems, and Software

Outline

- Traditional Hardware vs. Software
- Characteristics of reconfigurable (RC) arrays
- Hybrid: Mixing and Matching
- Opportunities for Design Automation

Traditional Choice: Hardware vs. Software

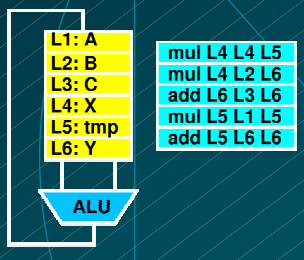
- Hardware fast
  - "spatial execution"
  - fine-grained parallelism
  - no parasitic connections
- Hardware compact
  - operators tailored to function
  - simple control
  - direct wire connections between operators



But fixed!

Traditional Choice: Hardware vs. Software

- Software Slow
  - sequential execution
  - overhead time "interpreting" operations

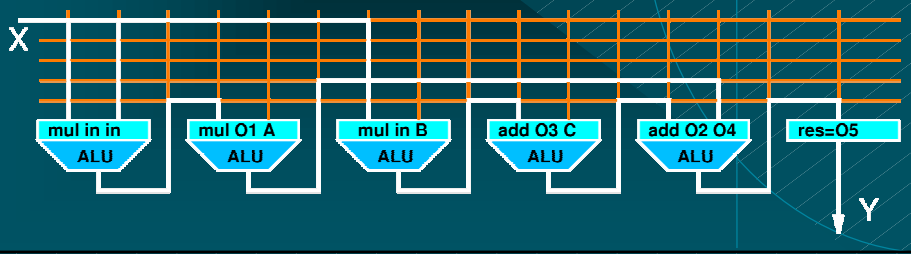


- Software Inefficient Area
  - fixed width operators, may not match problem
  - general operators, bigger than required
  - area to store instructions, control execution

But Flexible!

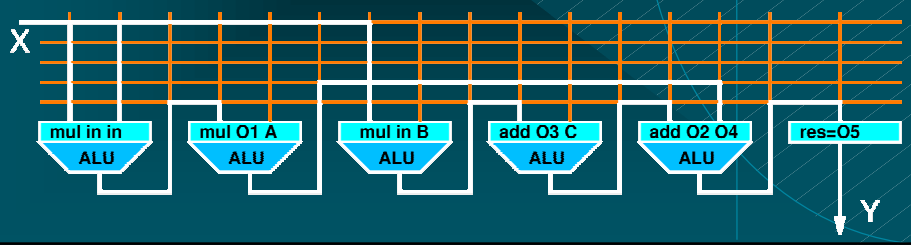
Reconfigurable Hardware

- RC Hardware Fast
  - spatial parallelism like hardware
  - problem specific operators, control
- RC Hardware Flexible
  - operators and interconnect programmable like software

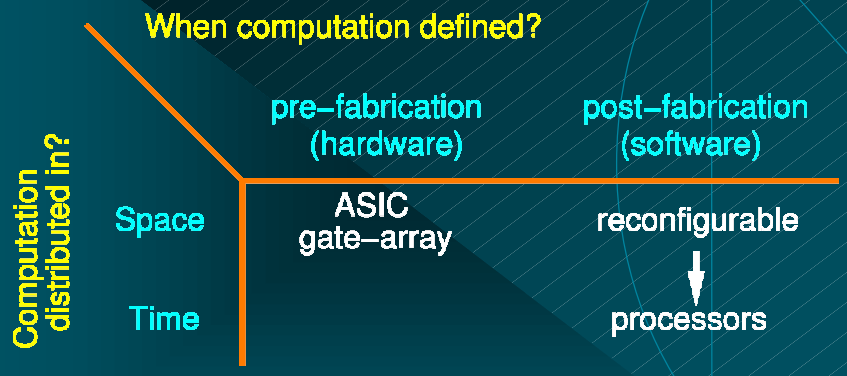


Reconfigurable Hardware

- Flexibility comes at a cost:
  - area in:
    - switches
    - configuration
  - delay in:
    - switches (added resistance)
    - logic (more spread out)
    - modifying configuration (traditionally)
  - Challenging "compiler" target

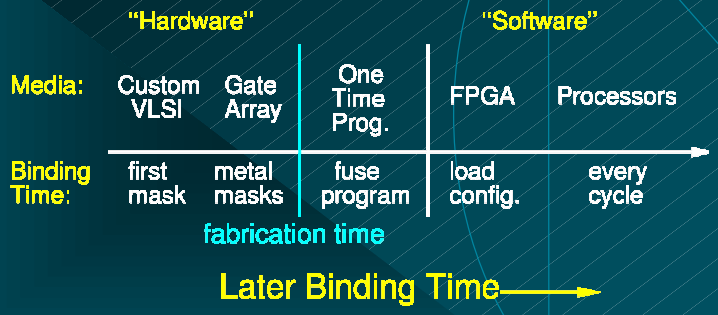


New Design Space



Important Distinction

- Instruction Binding Time
  - When do we decide what operation needs to be performed?



- General Principle
  - Earlier the decision is bound, the less area & delay required for the implementation.

Reconfigurable Advantage

1. Exploit cases where operation can be bound and then reused a large number of times.
2. Customization of operator type, width, and interconnect.
3. Flexible low overhead exploitation of application parallelism.

Specialization

- Late binding of operations
  - exploit cases where data can be "wired" into computation
  - narrows the performance gap between custom hardware and reconfigurable implementation
  - Example: Multiplication

Architecture	16×16		8×8	
	mpy λ^2s	scale λ^2s	mpy λ^2s	scale λ^2s
Custom 16×16	9.6	9.6	9.6	9.6
Custom 8×8			70	70
Gate-Array 16×16	1.3	1.3	1.3	1.3
FPGA	0.097	0.24	0.30	1.5

Runtime Reconfiguration*

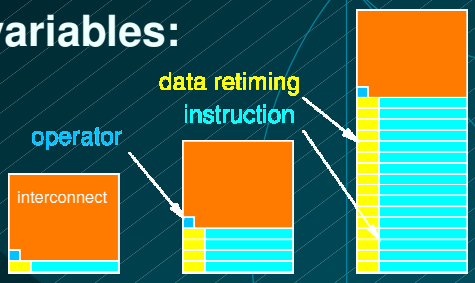
- Data-driven customization
  - ex: MPEG encode with partial reconfiguration between (I,P,B) frame types (every 33ms)
- Hardware Virtualization
  - demand paging, like virtual memory
- Dynamic specialization
  - ex: bind program variables on loop entry

*FPGAs poor at supporting this.

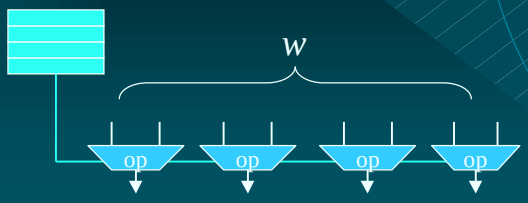
All very experimental.

Programmable Device Space

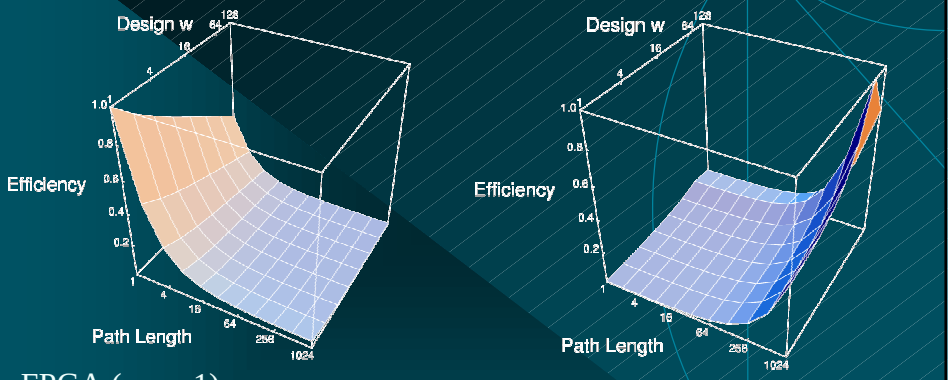
- Two important variables:
  - "instruction" or context depth



  - operator word width



Programmable Application Space Yield



FPGA (c=w=1)

"Processor" (c=1024, w=64)

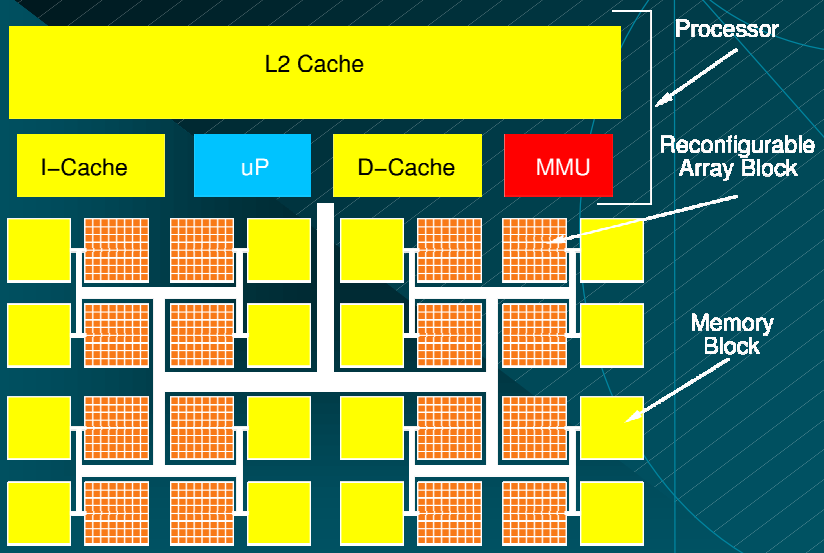
- Bit-level, reconfigurable organization is complimentary to processors

Case for Hybrid Architectures

- In general, applications have a mix of word sizes and binding times
- …and even a mix of fixed and variable processing requirements
- Previous slide suggests no single architecture robust across entire space

⇒ Need heterogenous components to best

Heterogenous Architecture



Design Automation Opportunities

- Currently, a limiter to the advancement of this technology is the state of the software flow.
- The ideal is HLL compilation with short compile/debug cycle.
  - Must combine elements of parallizing compilers,
    - thread- and ILP-level parallelism extraction
  - with elements of *hardware/software co-design,*
    - partitioning of "circuits" for RC array from "software" for processor
    - coordination of memory accesses

Design Automation Opportunities

- and elements of *FPGA and ASIC CAD*.
  - low-level spatial mapping (PPR)
  - more importance on pipelining/retiming
  - fixed resource constraints: wire tracks, memory/compute ratio preallocated
- **Flexible nature of the RC array encourages other optimizations:**
  - specialization of circuit instances around early bound data
  - fast, online algorithms to support run-time specialization

Design Automation Opportunities

- **Most importantly, the tools must run fast**
  - development requirements similar to software only environment
  - need to better understand tool quality/time tradeoff
- **Short of complete integrated HLL compilation**
  - "hand partitioning" between processor and RC array
  - combined FPGA flow with HLL
  - library based approach

Summary

- Reconfigurable architectures
  - spatial computing style like hardware
  - programmable like software
  - more computation per unit area than processors
  - efficient where processors are inefficient
- Heterogenous architectures (mix processors, reconfigurable, custom)
  - "general-purpose" and "application-targeted" processing components
- Exploiting these architectures: new opportunities for DA optimization.

Extra Slides

Brief History

- 1960: Estrin (UCLA) “fixed plus variable structure computer”
- 1980’s: Researchers using FPGAs reports “Supercomputer level performance at orders of magnitude lower costs”
- Mid 1990’s: DARPA invests ≈ $100M in “Adaptive Computing”
- Late 1990’s: 6 startup companies doing “Reconfigurable Computing”

Why the fuss now?

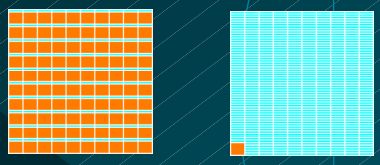
- The Promise: “Programmability of microprocessors with performance of ASICs”
  - *Programmability* key for:
    - standard (low cost) components
    - shorter time to market
    - adapting to changing standards
    - adaptability within a given application
- Technology pull:
  - greater processing capacity per IC
  - higher costs, fewer new designs
  - SOC benefits from on-chip flexibility

Application Successes

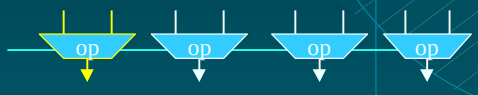
- **Research** *>10x performance density advantage over microprocessors and DSPs*
  - Pattern matching
  - Data encryption
  - Data compression
  - Video and image processing
- **Commercial Push**
  - telecom switches
  - network routers
  - mobile phones

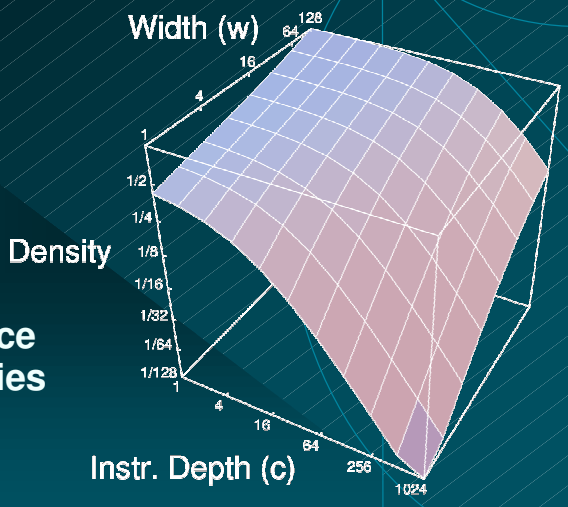
Programmable Design Space

- Variable Effects:
  - operator instruction depth can be order of magnitude density difference



  - operator word width can be order of magnitude in yielded density difference
    - consider narrow (bit) data on wide word architecture



Programmable Design Space Density

- Small slice of space
- 100× density across
- Large difference in peak densities
  - large design space!