

# Integrated Circuits and Systems Design Methodologies Research Program Overview

John Lach

Assistant Professor

Charles L. Brown Department of Electrical and Computer Engineering

University of Virginia

[jlach@virginia.edu](mailto:jlach@virginia.edu)

I don't design circuits.  
I design how to design circuits.

# Modern Integrated Circuit (IC) Design Issues

- Design complexities are increasing
  - Need formal design and verification techniques
- Manufacturing yield and runtime reliability are degrading
  - Need static and dynamic adaptability
- Physical issues (operating temperature, manufacturing variability, etc.) are becoming limiting factors
  - Need design-time modeling and run-time adaptability

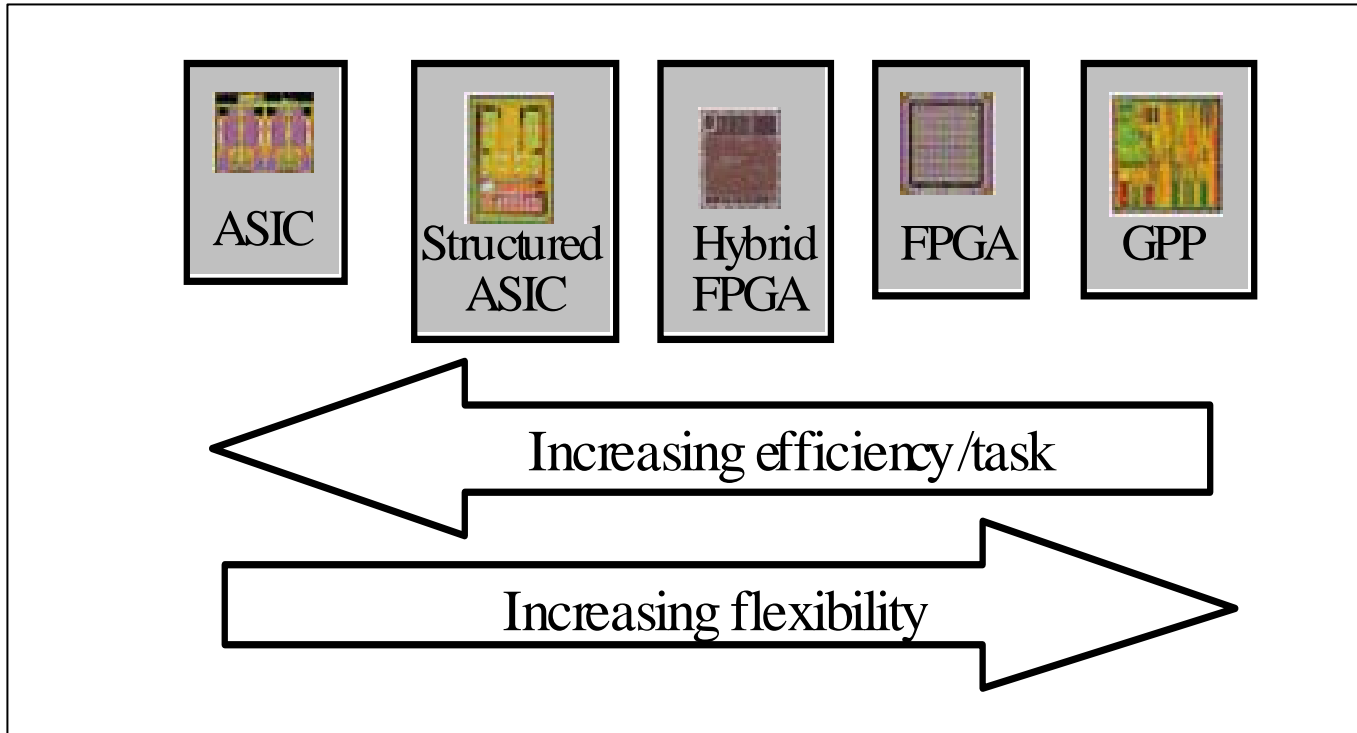
# Ongoing Projects I Won't Be Talking About Today

- Hardware design of extremely computationally complex algorithms
  - Formally explore the “col-space”
- Physically-aware design and synthesis techniques
  - Scheduling, allocating, and binding operations
- Reliability- and yield-aware design
  - Incorporate as metrics into cost function
- Wearable technologies for biomedical applications

# Ongoing Projects I Will Be Talking About Today

- Small-scale reconfigurability (SSR)
  - Dynactable computing
  - Application-specific product generics (ASPGs)
  - Heterogeneous redundancy
- Accessible formal verification
  - Incorporate formal verification into traditional IC design flow
  - Reduce verification costs
  - Enhance safety assurance and strengthens safety arguments for safety-critical systems

# SSR: Motivation

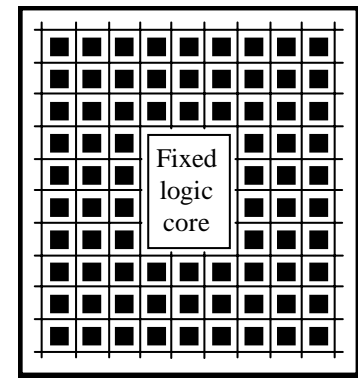


Fixed logic circuits: Best area, delay, power per task

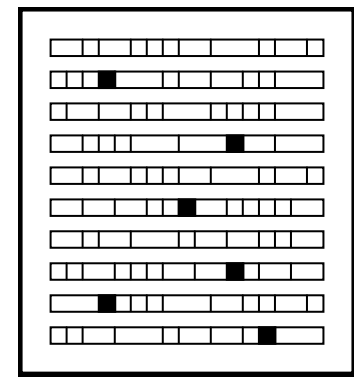
But lack of flexibility => no adaptability, custom fabrication

# SSR: Concept

- A design technique for adding flexibility to predominantly fixed logic circuits
- Reconfigurable logic and interconnect finely integrated at a gate-level granularity
- Application specific integration
  - Just-enough flexibility
  - Performance of ASIC w/ necessary flexibility



Hybrid FPGA



SSR

# Dynaptable Computing

- Collaboration with Mircea Stan and Kevin Skadron
- Efficient design of dynamically adaptable microarchitectures
  - Cache configurations
  - Datapath organizations
- Runtime monitoring to guide adaptation

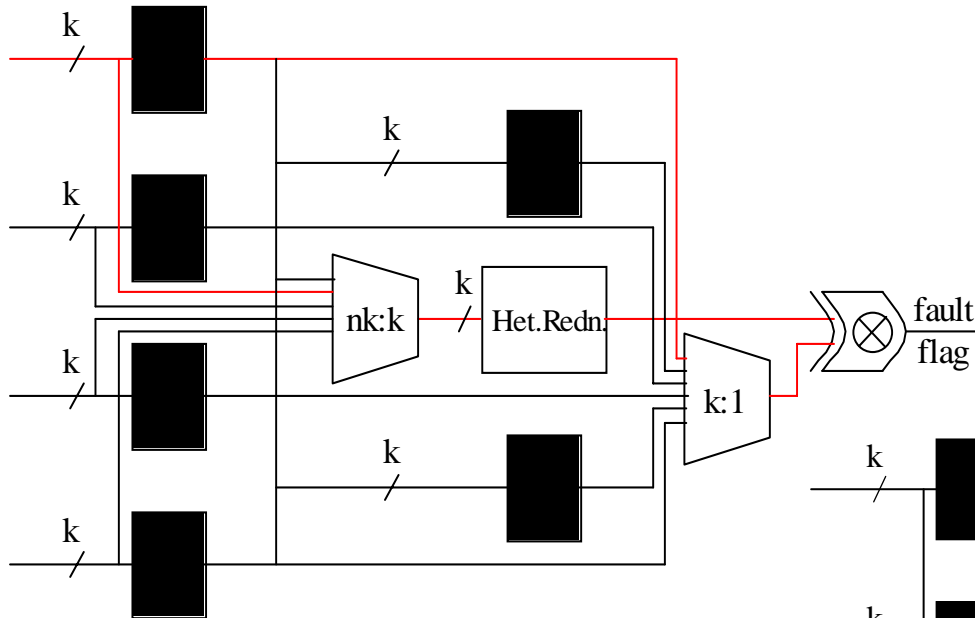
# Application-Specific Product Generics (ASPGs)

- Non-recurring engineering (NRE) costs increasing
  - Design cost (complexity, multiple passes)
  - Fabrication cost (millions for mask set alone)
- Flexible multi-mode systems for NRE cost amortization
  - Design circuit capable of being configured to implement a variety of applications within a domain
  - Customize controller and datapath through interconnect and flexible arithmetic component (FAC) configuration
  - Dynamic reconfiguration provides greater even efficiency

# SSR Application: Heterogeneous Redundancy

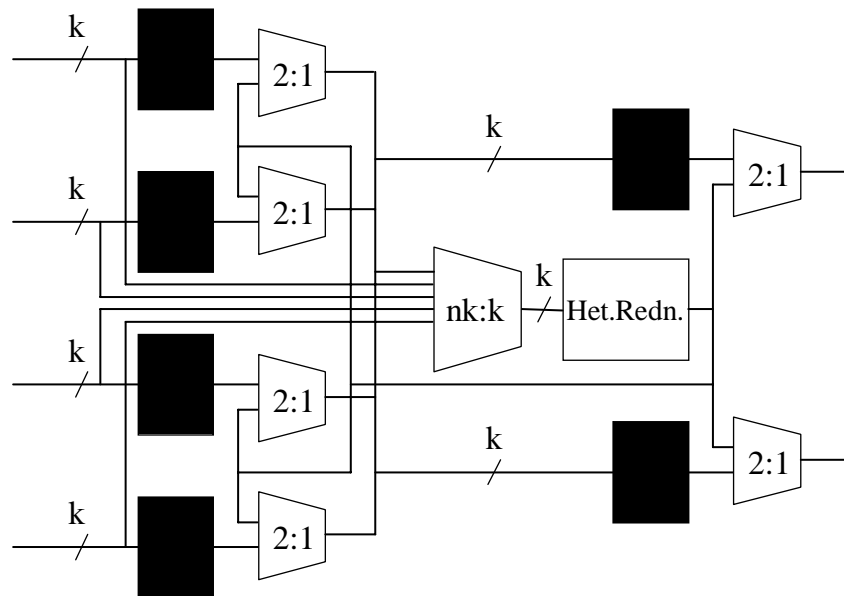
- Importance of embedded self-test
    - Continuing increase in system complexity
    - Relatively stagnant number of available IO pins
  - Importance of built-in tolerance of *permanent* failures
    - Mission critical applications & demanding environments
    - Process complexity, more failure modes
  - Traditional redundancy-based fault tolerance
    - Area scales with circuit size, insufficient reliability
- => **Low complexity fine-grained redundancy needed**

# Fault Tolerance



Fault detection and localization

- Primarily for control path circuits
  - $k$ -input cone of logic replaceable by  $k$ -LUT



Fault recovery

# Heterogeneous Redundancy: Results

- Average 37% logic area reduction in datapath
- Average 86% logic area reduction in control path
  - Limited interconnect area overhead
  - Better in more complex circuits
- Reliability increase
- Limited delay increase

# Accessible Formal Verification

- Electricite de France (EDF) would like to replace existing (aged) circuitry in nuclear power plants with anti-fuse-based FPGAs
  - Implement known functions
  - Replace existing components
    - Implementation details may be unknown
  - Properly use and verify IP blocks
  - Maximize portability and design re-use (across vendors, implementation platforms, and time)
- Circuitry **MUST** perform as specified due to safety-critical application
- Goal: Develop *design* and *safety assessment* techniques for safety-critical ICs
- Side goal: Reduce verification costs for commercial ICs

# What Can Disrupt HW-Based System Safety?

- Random failures
  - Single event upsets (SEU), manufacturing defects, aging (e.g. electromigration), etc.
  - Redundancy helps
- Systematic failures
  - Specification, design, or implementation error
  - Redundancy does NOT always help!

Our focus

# Combating Systematic Failures

- Assure correctness and completeness of safety specifications
  - Including specification of failure modes
- Assure correctness of design with respect to safety specifications
  - Functional/dysfunctional properties
    - Logic/timing properties
  - Freedom from intrinsic design faults
- Assure correctness of manufactured items with respect to design
  - Tool and “naked” FPGA qualification

Our focus

# Assuring Design Correctness

- Development process
  - Necessary, but not sufficient
- Evidence based on sampling
  - Testing, simulation, fault injection, ...
  - Coverage criteria and levels
- Operational experience
  - Credibility, applicability, sufficiency
- Expert judgment
- Formal evidence
  - A priori: systematic fault avoidance
  - A posteriori: formal verification

Our focus

# Formal Evidence

- We must **prove** that a design is correct for safety-critical applications
- Formal verification techniques highly mathematical in nature
  - Hardware specification/design engineers shy away
  - Verification engineers called in
- Incorporate formal verification into traditional FPGA design flow
  - Enable those who specify and design systems to be the same people who verify them
  - Long been the standard in software development
- Independent V&V still necessary

# Must Be Able To...

- Directly implement known functions
- Replace existing components
  - Implementation details may be unknown
- Properly use and verify IP cores
  
- Keep at vendor- and tool-independent level to maximize portability
  - RTL (e.g. VHDL, Verilog, etc.)

# Proposed Methods

- The Library Approach
  - Primarily for the direct implementation of I&C applications
  - Case study: real safety functions provided by EDF
- The Assertion-Based Verification Approach (ABV)
  - Primarily for IP core verification
  - Case study: complete formal verification of Motorola 6800 microprocessor IP VHDL core
- Both provide **accessible** formal evidence
  - Design flows similar to those currently used for current software-based solutions
  - Engineers will actually be willing to use them!



# The Assertion-Based Verification (ABV) Approach

- Embed assertions into RTL code
  - Statements about a design's intended behavior (i.e. properties) which must be verified
- Useful for both direct implementation of safety functions and IP core verification
- Natural part of design flow and IP verification
  - Assert **desired** functionality through **capturing** of design intent
  - Assert **understood** functionality through **probing** of design intent
- The 6800 case study
  - Instruction functional operation
  - Resultant condition code status
  - Instruction end-to-end cycle count
  - Cycle-by-cycle instruction format
  - Limit conditions behavior

# The Future of “Designing How To Design Circuits”

- Traditional IC technology scaling has enable us to get lazy
  - We don’t have to design something well because scaling will give us the performance, cost, power, etc. we need
- What happens when scaling ends and emerging technologies aren’t ready for prime time?
  - I believe that better design methodologies could continue the benefits of scaling for 10+ years after scaling hits the wall

# Integrated Circuits and Systems Design Methodologies Research Program Overview

John Lach

Assistant Professor

Charles L. Brown Department of Electrical and Computer Engineering

University of Virginia

[jlach@virginia.edu](mailto:jlach@virginia.edu)