

# Architectural Support for Design Productivity

Philip Leong ([phwl@cse.cuhk.edu.hk](mailto:phwl@cse.cuhk.edu.hk))  
Department of Computer Science and Engineering  
The Chinese University of Hong Kong



# Introduction

- Productivity a major issue for FPGA/ASIC designers
- Tend to tackle this issue from design tools perspective
  - Distinction between array of gates and a computational device (ASIC vs RC)
- How can we change FPGA architectures to make design easier, particular for RC systems?
  - Research opportunities.
- This talk
  - Part 1: Highlight issues
  - Part 2: Describe a hypothetical FPGA



# Part 1: Design Issues

# Design Issues

- FPGA design difficulties faced which can be addressed by architecture
  - Designs don't fit (**density**)
  - FPGAs slower & larger than ASICs (**speed**)
  - Too much freedom (**flexibility**)
  - Reuse and interoperability (**interfacing**)

# Density

- Improving density reduces partitioning issues and leads to improved speed
- A major overhead is storage of configuration
- Some directions
  - Using new configuration bit technology to reduce area
  - Time multiplexing computation

# Speed

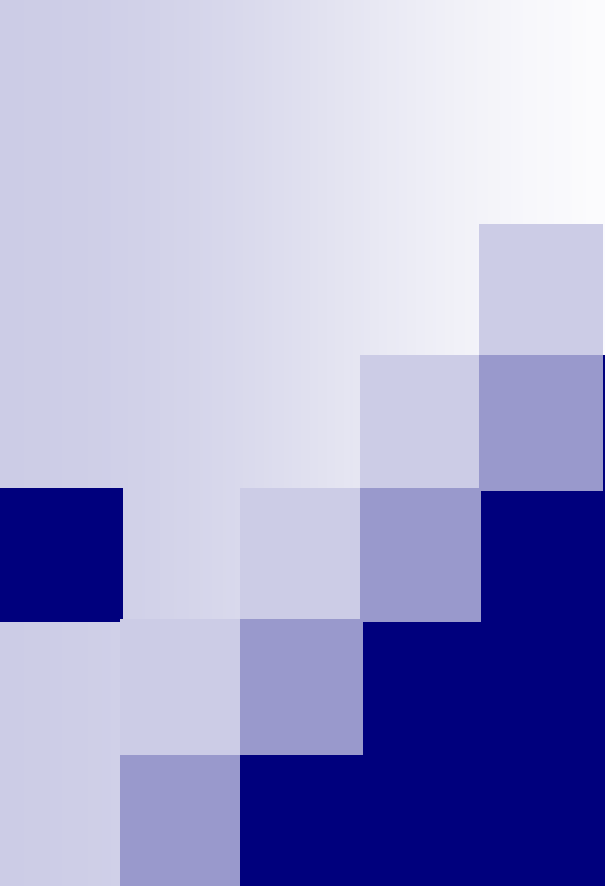
- Improved speed leads to less stringent timing constraints
  - CAD requirements relaxed
  - Timing closure easier to achieve
  - Time multiplexing becomes feasible

# Flexibility

- Flexibility is good but there are associated costs
  - ASIC-style design tools
  - Long design cycles
  - Design exploration necessary
  - Lower speed
- Coarse-grained approach
  - Restricts design space
  - Easier to target
- FPGAs use bit-based interconnect.
  - Word-based interconnect reduces configuration bits so routing requirements relaxed, performance enhanced

# Interfacing

- Difficult to communicate between modules, reuse designs, reuse blocks
- Have bus support (AMBA, Avalon, CoreConnect, Wishbone etc)
  - Need similar lightweight buses for standardised connections

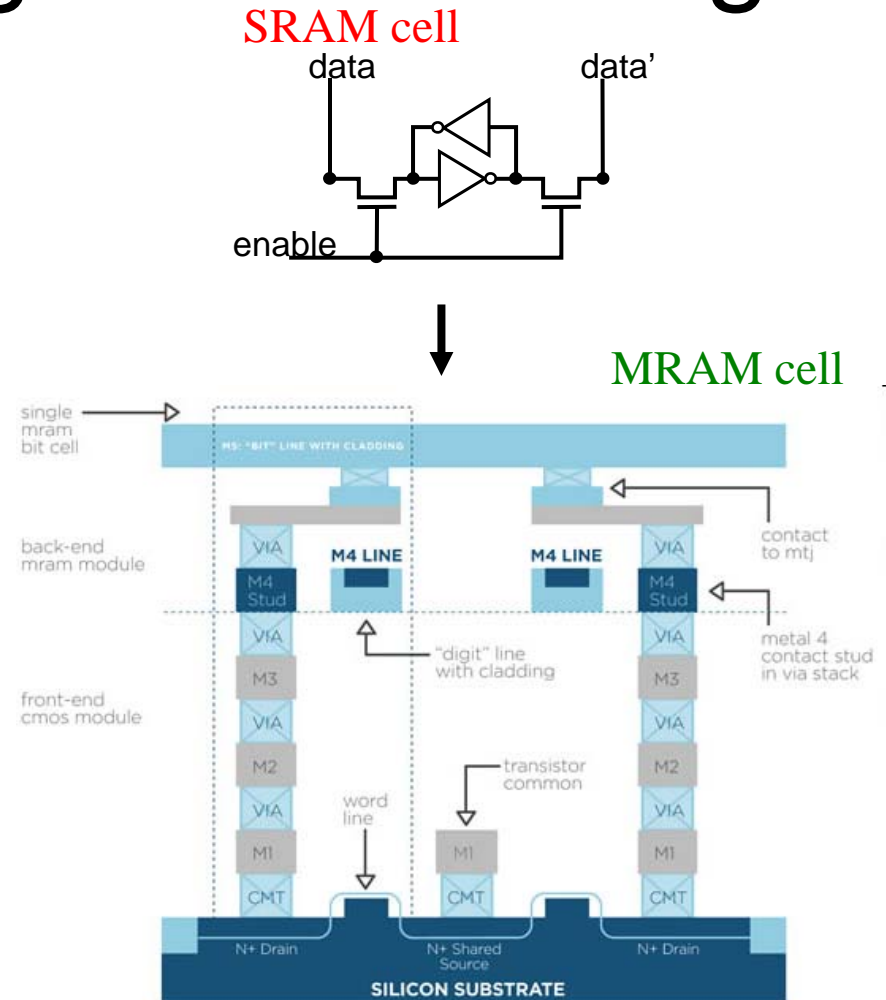


# Part 2: Hypothetical Architecture

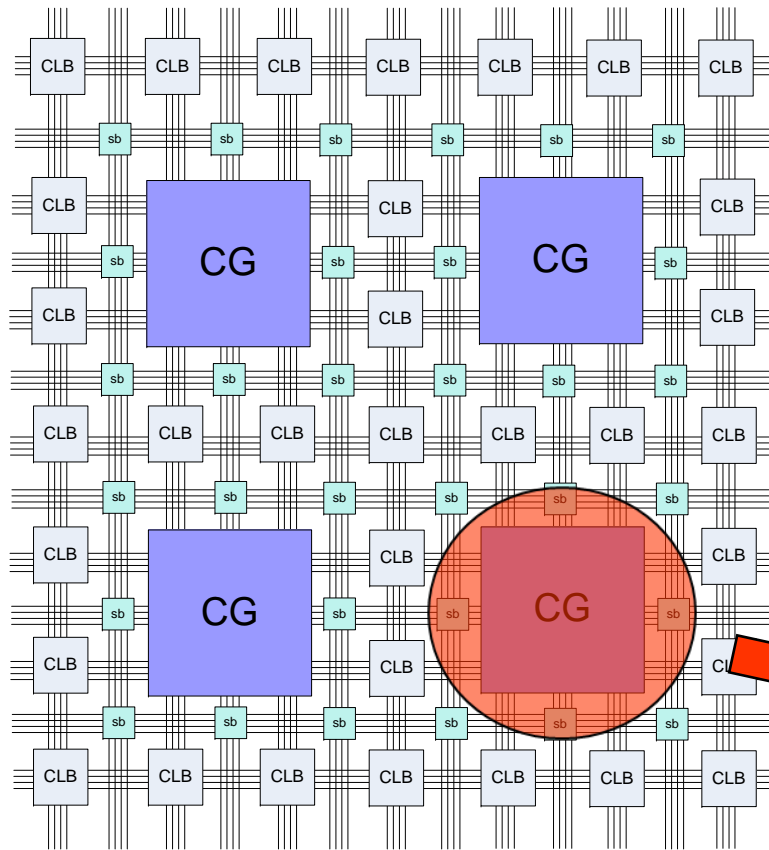


# Improved Configuration Storage

- Configuration bits stored in 6-T cell
- Compact alternatives include flash, phase-change, ferroelectric, magnetoresistive
- Improving density addresses partitioning issues and leads to improved speed



# Hybrid FPGA

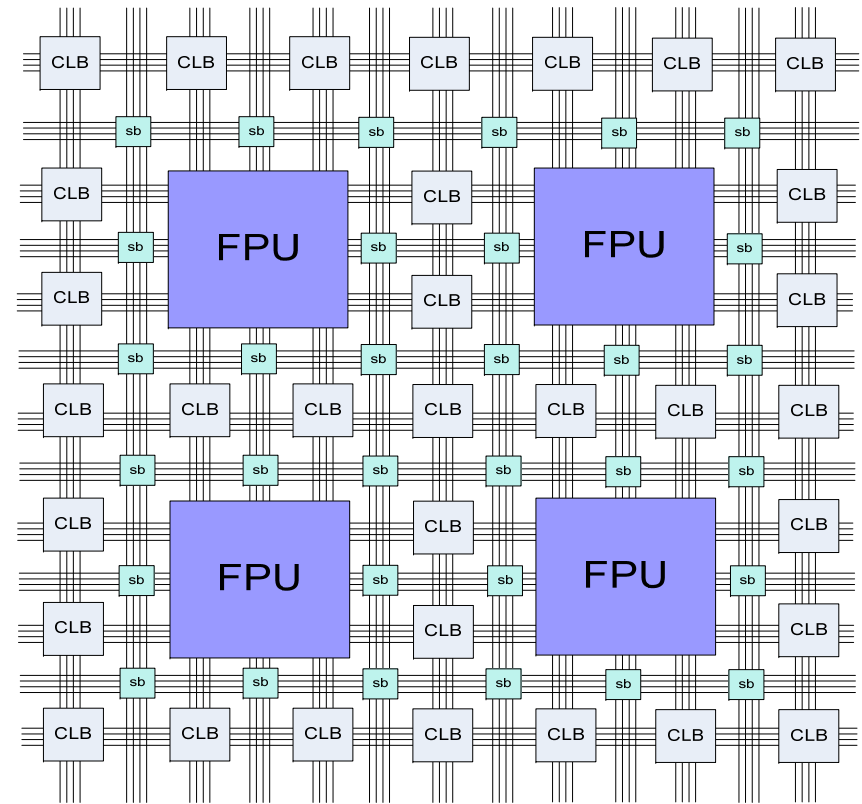


- Coarse grain resources are
  - Parameterised (compile time)
  - Reconfigurable (run time)
  - Domain-specific e.g. fixed/FP, mix of operators, wordlength

**Synthesisable  
Coarse-grained  
Block**

# Floating Point FPGA

- Floating point word-based FPGA
- Specialising reduces flexibility but improves design time, density, speed and power
- Floating point greatly reduces issues associated with determining accuracy requirements, wordlength, verification etc

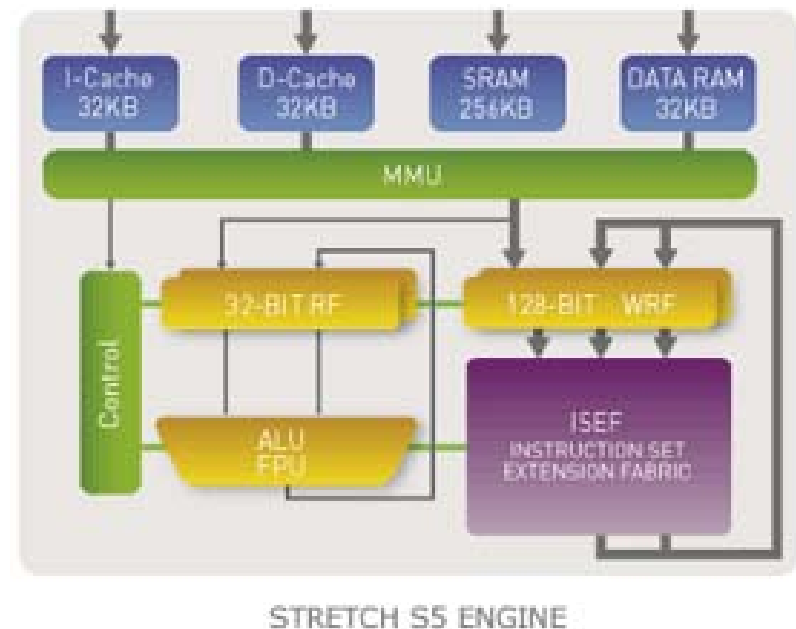


# Dynamic Reconfiguration

- Full runtime reconfigurable virtual-logic systems currently not practical due to hardware support and bandwidth issues
- Coarse-grained blocks have less configuration
- Need
  - Careful studies of the cost/benefits
  - Modelling tools
  - New architectures

# CPU Integration

- For RC applications use processor + customised instructions
- Leads to programming-based design
- Reconfigurable part serves to accelerate application datapath
- Example: Stretch



# Interfacing

- Interconnect: main limit to performance as scaling worse than logic scaling
- Can achieve near-speed-of-light signalling c.f. wire+repeater scheme
  - Increase L, reduce R, use (multiple) carriers
- Multiple carrier system reported by UCLA @ HPCA08, 70 Gb/s at 60 mW in 45nm

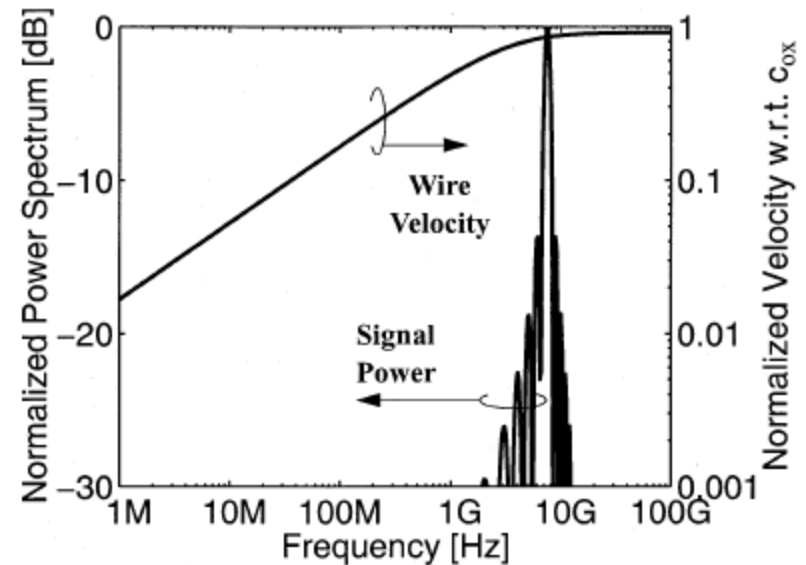
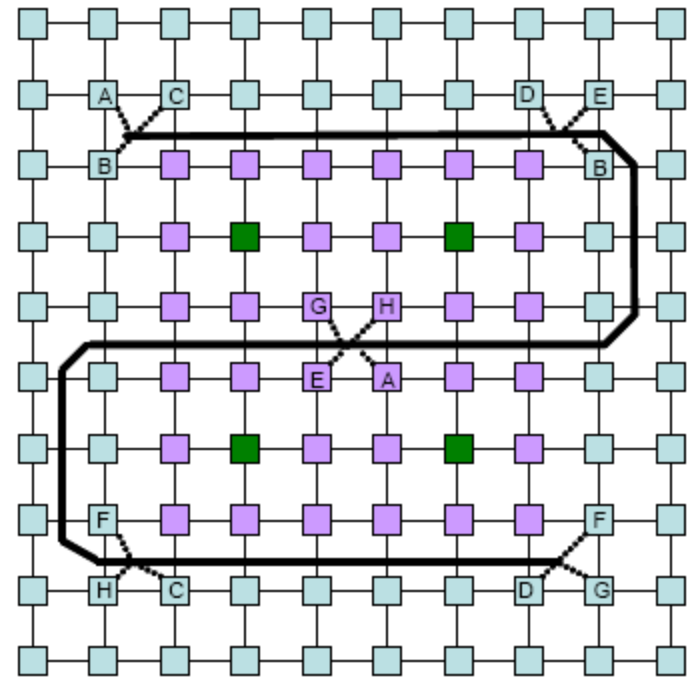


Fig. 4. Frequency characteristics of modulated pulse and low-loss on-chip interconnect.  $c_{ox}$  is the speed of light in silicon dioxide.

Source: Chang et. al., JSSC v38, no 5, 2003

# Hierarchical Interconnect

- Add speed-of-light NoC for connecting distant points on a chip (or off-chip)
- New design methodologies needed to utilise this scheme for ASIC/RC
- Problem partitioning, reduce interconnect delay, improving density and speed



UCLA RF shortcut scheme

# Conclusion

- Is a 10-50x increase in design productivity feasible in the next few years and if so, how?
  - Continue push for density and speed through: scaling, novel storage, domain-specific architectural support
  - Move from a P&R based methodology to compiler-based multithreaded programming one (via multiple embedded CPUs with configurable datapath model using near-speed-of-light interconnect)
  - Give users abstraction of infinite hardware
- Much scope for research in improved FPGA architectures for design productivity