Integrating Asynchronous Digital Design and Testing into the Undergraduate Computer Engineering Curriculum
DUE-0536343, 0717572, and 0717767; Scott C. Smith, Jia Di, and Waleed K. Al-Assaad, University of Arkansas and University of Missouri – Rolla

http://comp.uark.edu/~smithsc/CCLI_async.html

Motivation for using Asynchronous Circuits
- Asynchronous advantages vs. synchronous circuits
  - no global clock
  - lower power
  - less noise
  - less EMI

- International Technology Roadmap for Semiconductors (ITRS) predicts likely shift from asynchronous to synchronous design styles
  - increase circuit robustness
  - decrease power
  - alleviate many clock-related issues

- ITRS predicts that asynchronous circuits will account for 19% of chip area within the next 5 years, and 30% of chip area within the next 10 years

Comparison of Asynchronous Paradigms
- Bounded-Delay
  - Substantial Glitch Power
  - Worse-Case Performance
  - Substantial Timing Analysis Required
    - bundled data convention
    - datapath delay must be matched in control path
- Micropipelines is best known example

- Delay-Insensitive (DI)
  - Glitch Free
  - Average-Case Performance
  - Correct-by-Construction
  - dual-rail signals
    - completion detection
  - Additional Advantages
    - very try robust
    - tolerate process variations
    - tolerate power supply variations
    - tolerate temperature variations
  - design reuse is straightforward
  - easy to interface multi-rate circuits
  - reduced crosstalk

Comparison of Delay-Insensitive Paradigms
- Precharge Half-Buffer (PCHB) logic family
  - dynamic logic
  - synthesis performed at transistor level
  - uses non-standard EDA tools

- Phase-Encoded Logic
  - transforms synchronous circuit to delay-insensitive
  - developed to ease timing constraints, not obtain speed and power benefits

- NULL Convention Logic (NCL)
  - synthesizable at gate level
  - can use standard EDA tools with slight modification
  - Seitz's method, Anantharaman's approach, DIMS, Singh's method, and David's method
    - synthesis performed at gate level
    - only type of state-holding gate: the Nuller

NCL Provides Best Opportunity for Integration into Mainstream Semiconductor Design Industry
- NCL systems consist of DI combinational logic sandwiched between DI registers
  - very similar to synchronous systems
  - automated design flows can be developed to follow standard synchronous circuit design automation
  - enable developed DI design flow to be more easily incorporated into chip design industry

- NCL systems are delay-insensitive
  - design process much easier to automate than non-DI asynchronous paradigms, since minimal delay analysis necessary to ensure correct circuit operation

- NCL's numerous advantages
  - power, noise, and EMI advantages compared to synchronous circuits
  - performance and design reuse advantages compared to synchronous and non-DI asynchronous paradigms
  - area and performance advantages compared to other gate-level DI paradigms

Developed Educational Materials
- Course Modules
  - Introduction to Asynchronous Logic
  - Introduction to NCL (null-convention logic)
  - Transistor-Level NCL Gate Design
  - Input-Completeness and Observability
  - Dual-Rail NCL Design
  - Quad-Rail NCL Design
  - NCL Throughout Optimization
  - Group Projects

- NCL Libraries
  - VHDL Library
    - package that defines fundamental NCL data types
    - file containing standard library
  - NCL AND Function
    - symbolically complete
    - input value unambiguously specified regardless of time reference

- VHDL Course Modifications
  - VHDL course module 3
    - student roughly equivalent to 3.5 weeks of original topics replaced by 2.5 weeks of asynchronous logic topics
    - changes do not eliminate any key VHDL course modules

- VLSI Course Modifications
  - VLSI course module 10
    - Design of Flip-Flops, Latches, and Sequential Circuits
      - Asynchronous Circuit Synthesis
      - Asynchronous Logic Components
      - Asynchronous and Analog Interface
      - Asynchronous Clocking
      - Asynchronous Design Techniques

- End-of-Course Project
  - student group
    - project requires interface of four or fewer variables

- Introduction to NULL Convention Logic

- NULL Convention Logic
  - Denoted THn
  - Output asserted when at least one of n inputs asserted (1 ≤ n ≤ m)
  - Output hysteresis behavior
    - once output asserted, it remains asserted until all inputs are de-asserted
  - NULL Convention Logic (NCL)
    - symbolically complete
    - input value unambiguously specified regardless of time reference

- Transistor-Level NCL Gate Design: TH23

- NCL Throughput Optimization: Pipelining

- VHDL Course Modifications
  - student group
    - project requires interface of four or fewer variables

- VLSI Course Modifications
  - student group
    - project requires interface of four or fewer variables

- Conclusions
  - Students found the asynchronous logic topics very interesting, and performed quite well on the related assignments
  - VHDL course
    - class average on the asynchronous logic homework assignment was the second highest of the six homeworks (i.e., 83% versus 86%, 76%, 73%, 64%, and 44%)
    - asynchronous projects average was approximately the same as the first project (i.e., 86% versus 80%)
    - includes one group of two students who decided not to complete the project, and therefore received a 31% on the partial submission
    - excluding that outlier boosts the asynchronous project’s average to 91%
  - VLSI course
    - students completed the asynchronous lab assignment
    - all three NBL-based semester projects worked correctly
    - course modules were externally evaluated and rated as excellent
    - provides an easy way to integrate cutting-edge technology into standard educational packets to provide a low-cost, informative addition to the Computer Engineering curriculum
    - knowledge gained in these courses will better prepare students for the challenges faced by the digital design community for years to come

- The course modules and libraries are available for download at:
  - http://comp.uark.edu/~smithsc/CCLI_async.html