

Asynchronous IC Interconnect Network Design and Implementation Using a Standard ASIC Flow

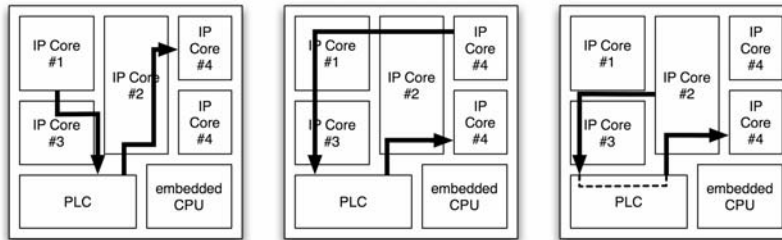
Bradley R. Quinton, Mark R. Greenstreet†, Steven J.E. Wilton*,
*Dept. of Electrical and Computer Engineering,
†Dept. of Computer Science
University of British Columbia
Vancouver, Canada*

Motivation: General

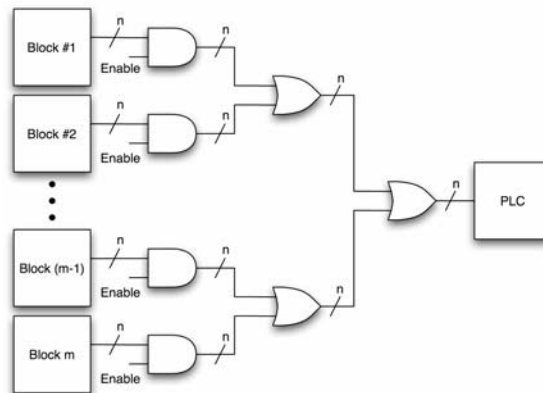
- **Wire delay** is increasing with respect to gate delay
- This can make inter-block **interconnect** the bottle-neck to overall IC performance
- What is the best way to manage this problem?

Motivation: Specific

- Sharing a single physical resource amongst many parts of the design requires a network that spans the **entire die**

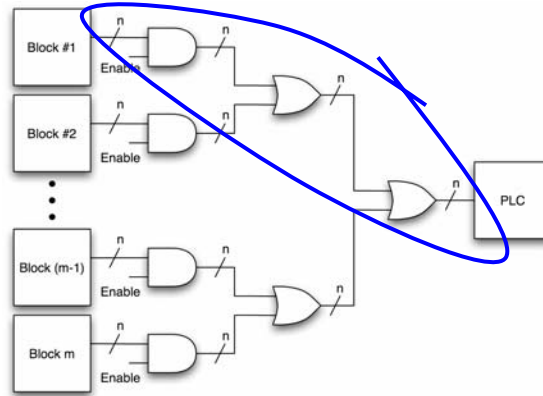


Motivation: Specific



- multiplexed bus spanning the entire chip

Motivation: Specific



- multiplexed bus spanning the entire chip

Past Work: Synchronous

- Algorithms have been proposed to find the optimal **repeater** and **register** locations for synchronous interconnect
- However, these algorithms assume that a **low-skew clock** is available at any location on the die
- Creating this clock is difficult:
 - on-die process variation
 - power supply noise
 - clock jitter
 - placement blockages

Past Work: Asynchronous

- Asynchronous design techniques provide a potential solution since they do not require a global clock
- However, techniques that have been proposed thus far require **custom** designed circuits and **manual design** optimization
- This makes these techniques difficult to compare to synchronous techniques, and infeasible for many ASICs and SoCs designs

Goals of this Work

Goals of this Work

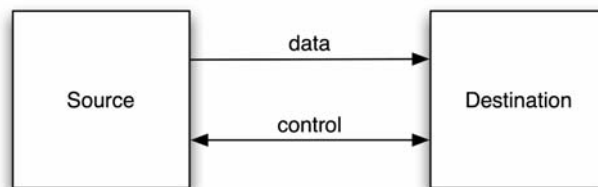
- 1) **Develop an asynchronous design that is feasible using regular standard cells, and off-the-shelf CAD tools.**

Goals of this Work

- 1) **Develop an asynchronous design that is feasible using regular standard cells, and off-the-shelf CAD tools.**
- 2) **Compare synchronous and asynchronous interconnect networks in terms of throughput, area, power and latency for a range of designs.**

Asynchronous Interconnect

Basic Structure



- By **coordinating transfers** between the source and destination asynchronous techniques avoid the requirement of a global clock

Data Formats

- Two broad categories:
 - 1) **Bundled-data**
 - control signaling is separate from the data
 - requires delay-matching*
 - 2) **Delay-insensitive**
 - control signaling encoded with the data
 - no delay-matching* required

* Arbitrary delay-matching is not supported by most design tools.

Handshaking

- Two commonly used handshaking protocols:
 - 1) **2-phase**
 - control signal *transitions* mark data transfers
 - 2) **4-phase**
 - control signal *values* mark data transfers

* Detecting transitions is 'harder' than detecting values, but 4-phase requires more traversals of the interconnect

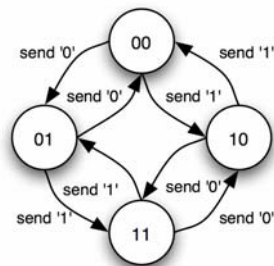
CAD Tool / IP Considerations

- CAD tool limitations from the perspective asynchronous interconnect design:
 - delay-matching
 - automated **glitch** avoidance
 - inference from **combinational loops**
 - path based **delay optimization**
 - automatic insertion of **sequential cells** *
 - non-optimal sequential cells

* This is a significant since it restricts asynchronous pipelines to occur only at network nodes

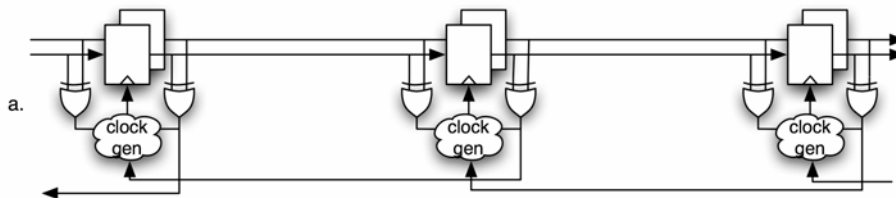
Basic Design - Data Encoding

- Many **data encodings** are possible for delay-insensitive circuits
- We choose 'dual-rail' encoding to minimize the depth of the control decode
- 'dual-rail' encodings allow bit **transitions** to be detected with an simple XOR gate.



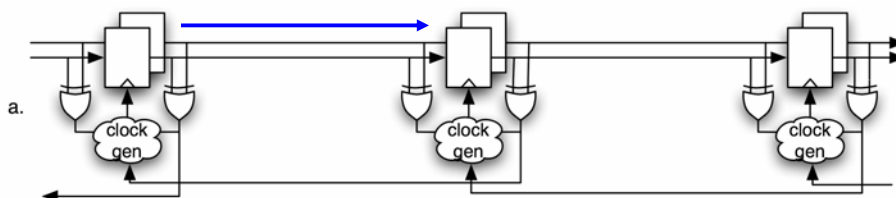
Basic Design - Sequential Gates

- We use a flip-flop based design to conform to standard IP and CAD tools
- 2 flops/bit are require because the data is encoded



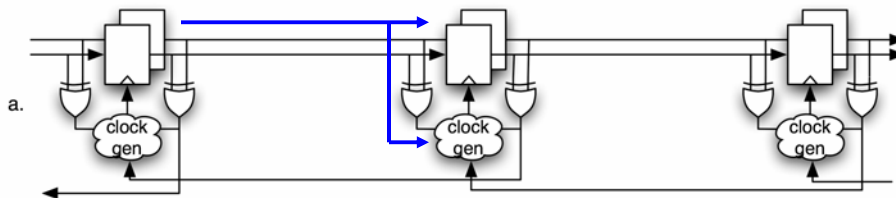
Basic Design - Sequential Gates

- We use a flip-flop based design to conform to standard IP and CAD tools
- 2 flops/bit are require because the data is encoded



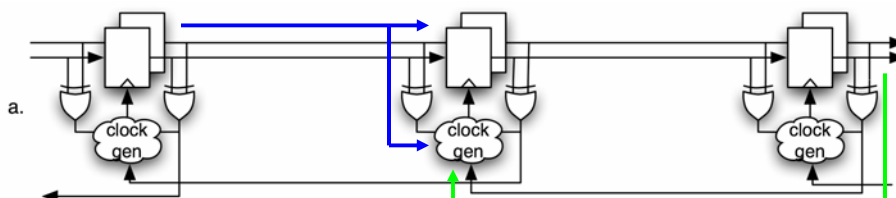
Basic Design - Sequential Gates

- We use a flip-flop based design to conform to standard IP and CAD tools
- 2 flops/bit are require because the data is encoded



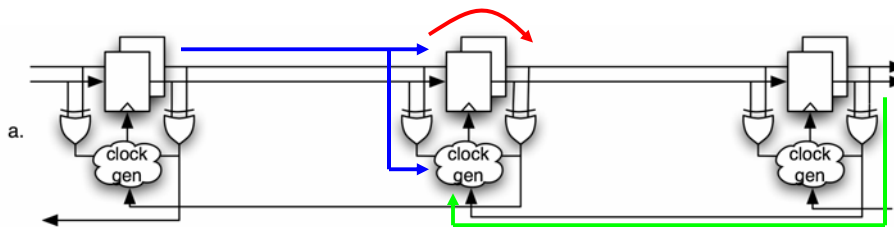
Basic Design - Sequential Gates

- We use a flip-flop based design to conform to standard IP and CAD tools
- 2 flops/bit are require because the data is encoded



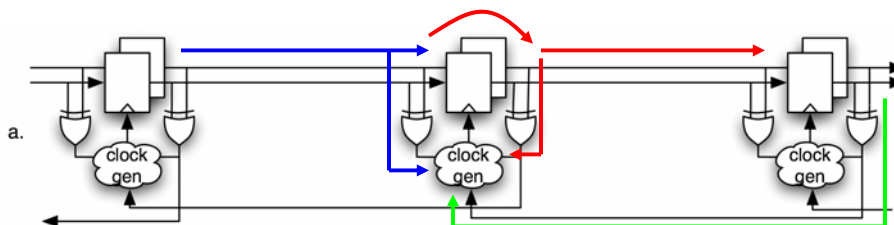
Basic Design - Sequential Gates

- We use a flip-flop based design to conform to standard IP and CAD tools
- 2 flops/bit are require because the data is encoded



Basic Design - Sequential Gates

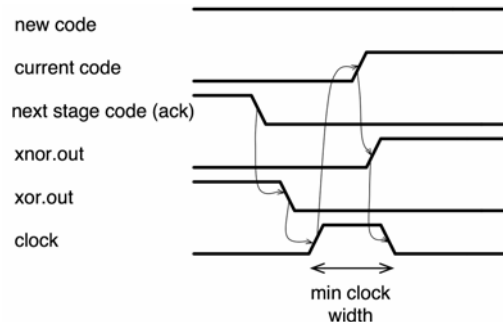
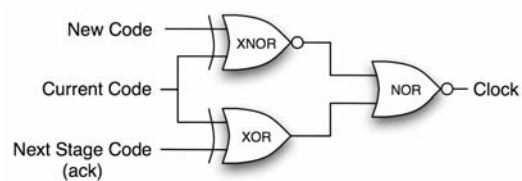
- We use a flip-flop based design to conform to standard IP and CAD tools
- 2 flops/bit are require because the data is encoded



Basic Design - Clock Generation

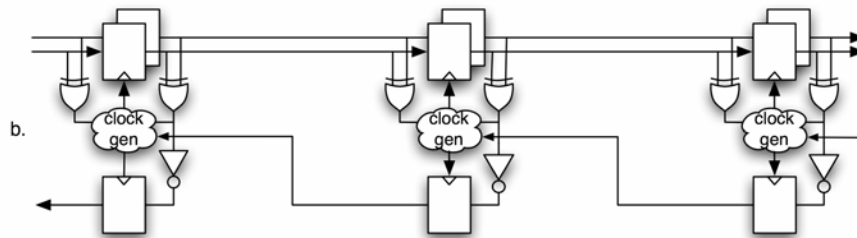
- Clock generation must be done carefully in a flop-based design to avoid **glitches**
- A clock edge is generated if:
 - 1) the code at the **next stage** equals the current stageand,
 - 2) the **incoming code** is different from the current code

Basic Design - Clock Generation



Additional Optimization

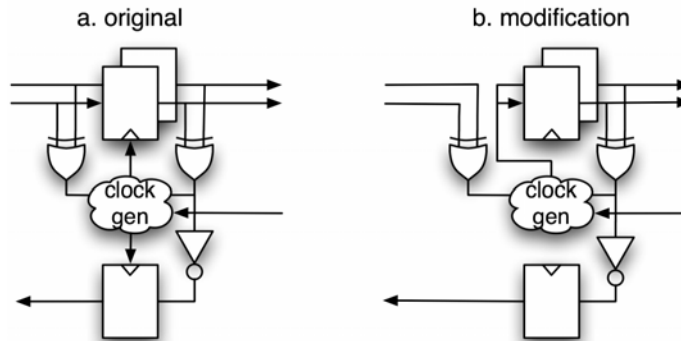
- To further increase the throughput of the design we '**pre-calculate**' the acknowledgement signal



Automatic Delay Optimization

- CAD tools are designed to optimize delay based on paths between **sequential elements**
- This is possible in our design, however it is necessary to explicitly define a large number of paths/clocks
- To avoid this we made a **circuit modification** before delay optimization, and corrected it before routing

Automatic Delay Optimization



- Creates a 'virtual' **global clock** to allow the repeater insertion tool to optimize the correct paths.

Automatic Delay Optimization

- Enabling this automatic repeater insertion had a significant performance impact on the design.
- For the experiments on the largest die size:
 - 8856 cells were **resized**
 - 232 cells were **inserted**
 - the path delay improved by **12.46ns**

Synchronous Interconnect

Clock Constraints

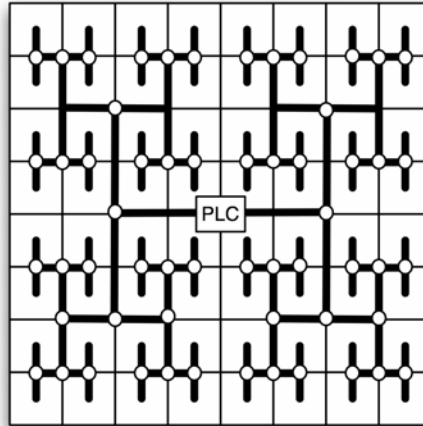
- register pipelining was used for the synchronous design
- registers are restricted to occur at **network nodes**
- the clock modeled with 100 ps of clock uncertainty (jitter) of 100 ps of skew

Experimental Framework

Target ICs

- we created **9 ICs** based on the TSMC 0.18 μm
 - 3 core die sizes:
 - 3830x3830 μm (~1 million gates),
 - 8560x8560 μm (~5 million gates),
 - 12090x12090 μm (~10 million gates)
 - 3 different block partitions:
 - 16 blocks
 - 64 blocks
 - 256 blocks

Block / Network Placement



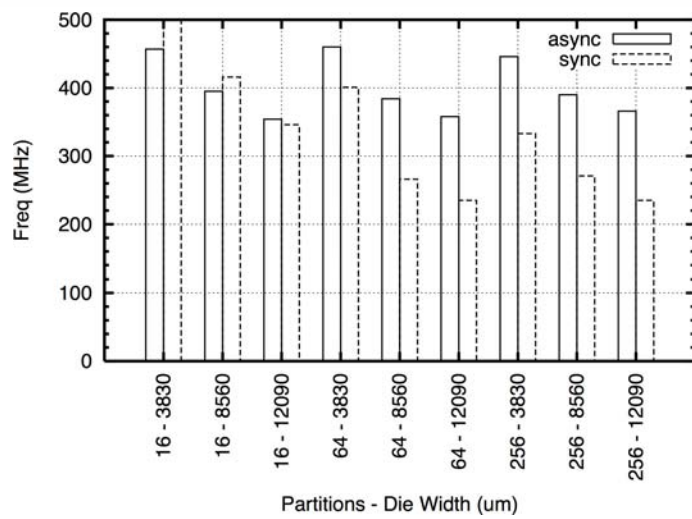
CAD Tool Flow

- Completely automated design flow:
 - **Library:** Artisan SAGE-X 0.18 μ m
 - **Synthesis:** Synopsys Design Compiler
 - **Simulation:** Cadence Verilog-XL
 - **Place and route:** Cadence SoC Encounter
 - **Static Timing:** Synopsys Primetime *
 - **Power :** Synopsys PrimePower *

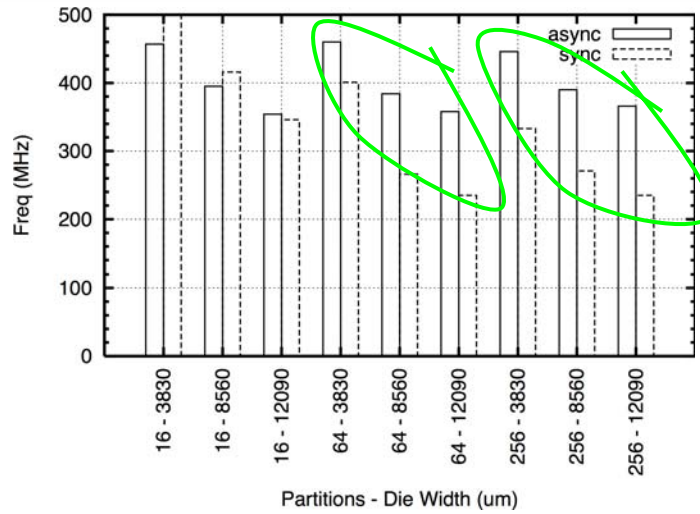
* Results measured from detailed, placed and routed designs

Results

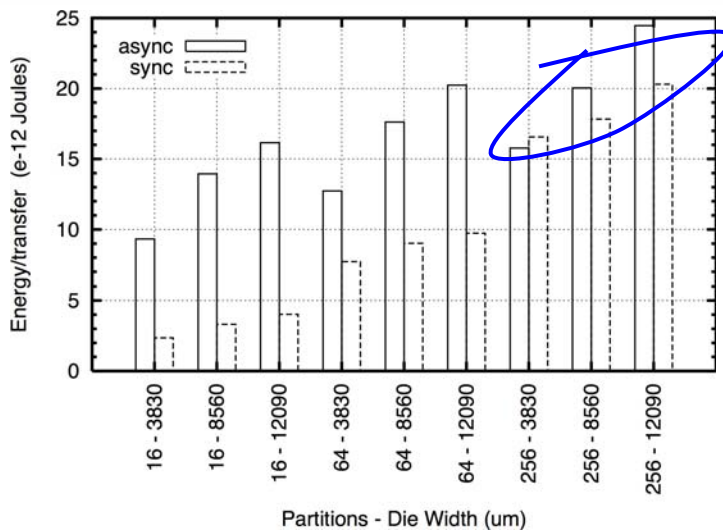
Throughput - No Global Clock



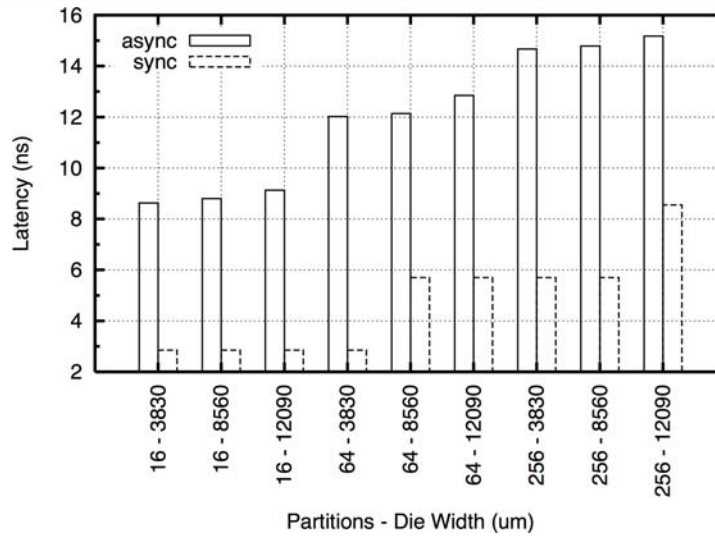
Throughput - No Global Clock



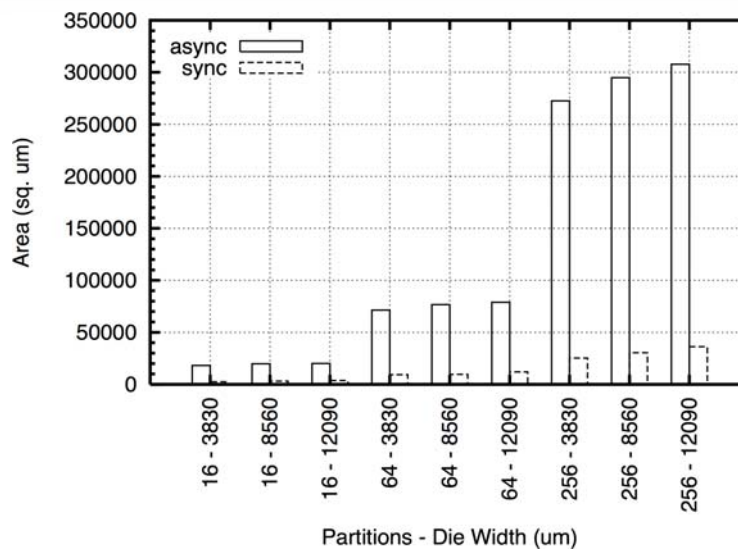
Power - 350 MHz



Latency - 350 MHz



Area - 350 MHz



Conclusion

- It is **feasible** to implement an asynchronous interconnect network using standard cells and CAD tools
- For **large, high-speed ICs** it is possible to achieve a high throughput with asynchronous interconnect while avoiding a global clock for pipeline registers
- Asynchronous interconnect offers **similar power**, but significantly **higher area** than synchronous alternatives

Future Work

- Use **90nm** process - expecting a more significant difference in gate and wire delay
- Investigate the effect of **enhancing** the placement tool to allow automatic insertion of asynchronous pipelines
- Create a **new** sequential “standard **cell**” for asynchronous pipelining

End