EE371

Asynchronous Circuits or Circuits Without Clocks: What makes Them Tick?

Jo Ebergen
Sun Microsystems Laboratories
jo.ebergen@sun.com

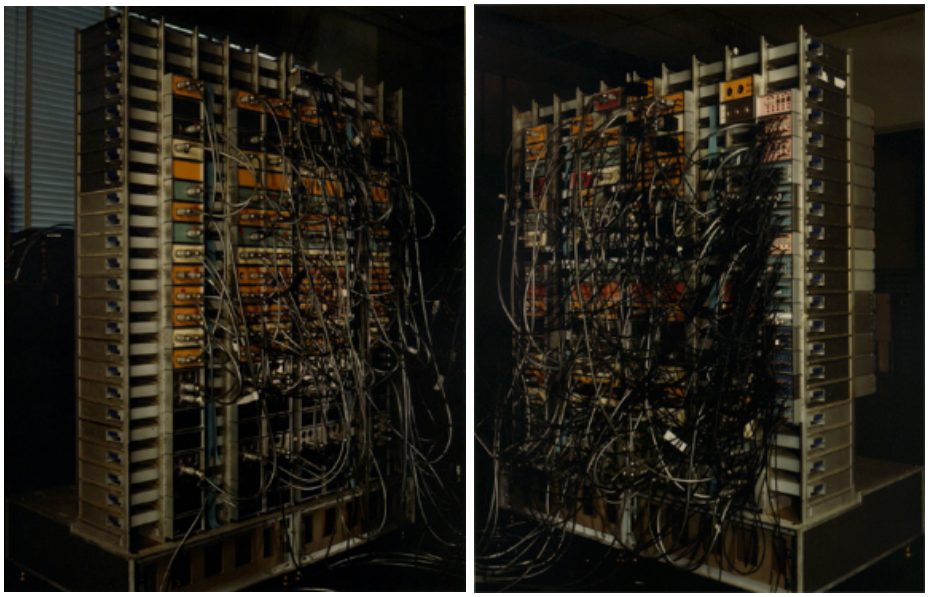
A Historical Perspective

- Clocked versus clockless
- The first computers: clocked or clockless?
- ENIAC('46), ACE('46), EDVAC ('46)
- Enablers for clocked design: Boolean Algebra and FSMs
- A big problem in early days: making *reliable* computers
- Clock is convenient in design and verification
- Terminology: non-synchronous, asynchronous, speed-independent, self-timed, delay-insensitive

Why No Clock?

- Speed
  - Average vs worst-case behavior
- Avoid timing problems related to clock
  - Crossing clock domains, synchronization problems
  - Clock distribution, skew, jitter
- Modularity
  - Reusability
  - Composability
  - Incremental improvement
  - Ex: “Macromodules Project” (Clark & Molnar), “Self-Timed Systems” (Seitz)

An Example: Macromodules (’66-’74)

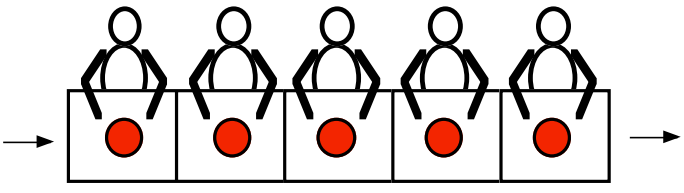


More: Why No Clock?

- Low power
  - Portable electronics
  - Budget constraint: total energy, not cycle time
- Architectural freedom
  - Concurrency at any grain size
- Robustness
  - Correct operation over a large range of process, voltage, and temperature variations
- Reduced electromagnetic radiation
  - Spread spectrum versus isolated peaks

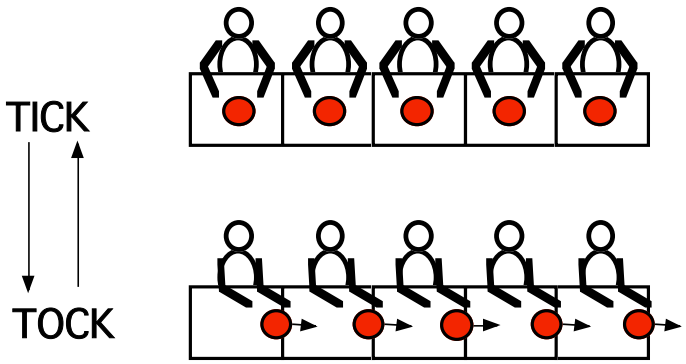
A Live Demo

- Pipelined “food” processing
- Clocked and clockless version



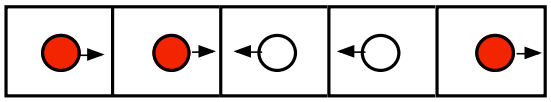
Clocked “Food Processing”

- TICK: process food
- TOCK: move plate to successor



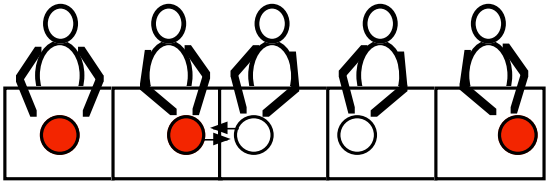
Clockless “Food Processing”

- Two types of plates
- Red plate = stage with data item
- White plate = empty stage
- Red plates move forward
- White plates move backward



Repeat These Actions

- TAKE: Take RED plate from predecessor
  (and give WHITE plate)
- EAT: Process food
- GIVE: Give RED plate to successor
  (and take WHITE plate)



To Notice

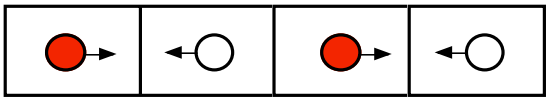
- Clock period must be long enough for slowest person for worst food
- *Clockless pipeline adjusts speed to person and food being processed*

- Clocked pipeline spends energy with every tick or tock, even when there is no food on plate
- *Clockless pipeline spends energy only when food must be processed*

- Clocked pipeline must have the same frequency as the source and sink, to prevent overflow or underflow
- *Clockless pipeline has automatic overflow and underflow protection*

FIFOs

- FIFOs as a basis for pipelines
- Simple local communication

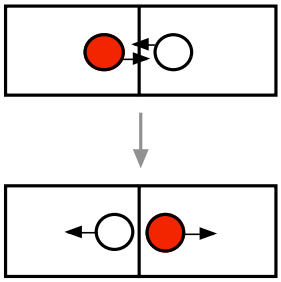


- Red plates move to the right
  - Red plates are data items
- White plates move to the left
  - White plates are often “empty stages”
  - But can be data items as well

Local Communication

- Synchronization at each boundary
- Only one rule:

IF (left plate is red AND right plate is white)
THEN exchange plates



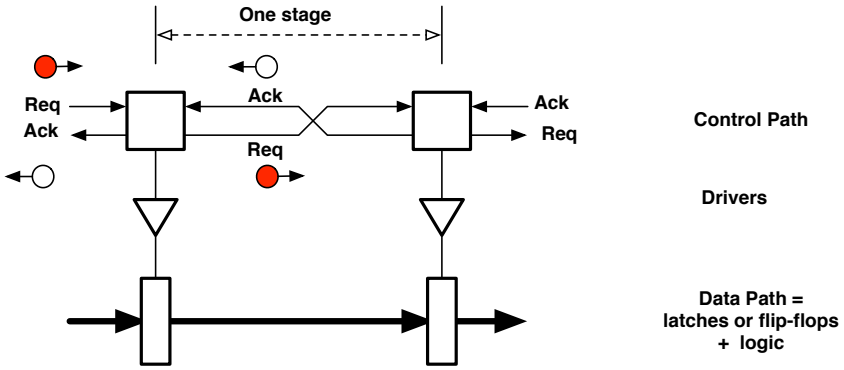
instantaneous event

Many Implementations

- There are many circuits to implement this synchronization
  - Muller C-elements
  - GasP modules
  - Many others...
- Challenge is to design circuit family that is efficient and versatile
  - Small delay
  - Small area
  - Little energy consumption
  - Can do pipelines with Branch, Merge, Fork, Join
- Each circuit family has different signalling conventions
  - Bundled data: Control (clock) says when data must be valid
  - Validity embedded in data encoding

Bundled Data Signaling

- Request indicates arrival of red plate
  - Request is bundled with data
  - Request indicates validity of data
- Acknowledge indicates arrival of white plate

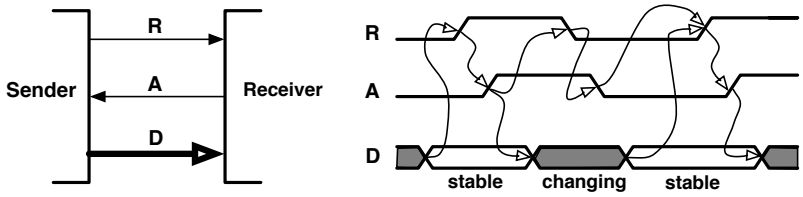


About Bundled Data Signaling

- Control modules produce “clock” signal
  - when needed
  - where needed
- Implementation is similar to clocked design, but with local “clocks”
  - Max and min-time constraints remain
- When data path has logic, insert extra delay in request wire that matches delay in logic
  - Often matched delay is worst-case delay
  - Matched delay may depend on data, if overhead is low
- Use of flip-flop or latch depends on signaling scheme

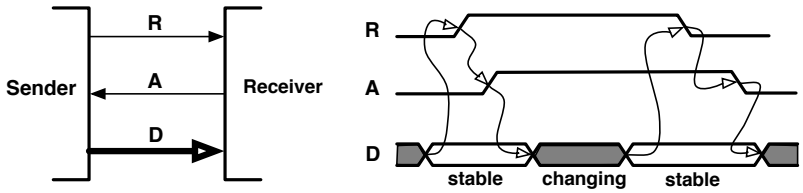
Two-Track Four-Phase Signalling

- Two-track: separate wire for Request and Acknowledge
- Four-phase: four transitions in cycle
- Also known as “return-to-zero” signalling
- Fits well with standard logic gates
- Many variations exist
- Often has long cycle time



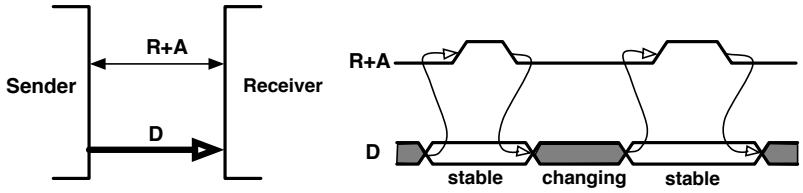
Two-Track Two-Phase Signaling

- Two-phase: two transitions per cycle
- Also known as “transition” signalling
- Uneasy fit with standard logic gates
- Short cycle time for simple pipelines
  - Long cycle time for anything else



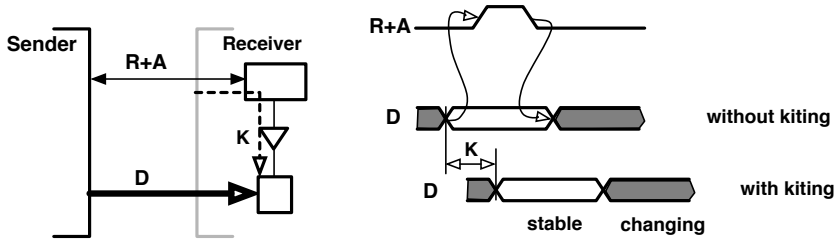
Single-Track, Four-Phase Signalling

- Combines best of both worlds
  - Fits with standard gates
  - Short cycle time
- Uses one tri-state wire, which needs keeper
  - Watch out for noise issues



Kiting Delay

- Control is usually ahead of data
  - head start is called “kiting delay”
- What matters: setup and hold constraints for storage elements

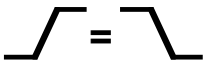


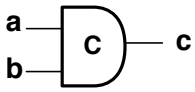
Validity Encoded in Data

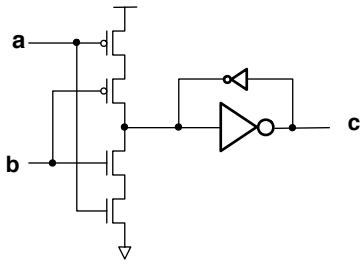
- As opposed to bundled data signaling
- There is no separate signal that says when data must be valid
- Data validity is encoded in data
  - Receiver detects when new data word has arrived
  - Requires completion detection, which may add to cycle time
- Ex: m-out-of-n encoding
  - 1-out-of-2 encoding = dual-rail encoding
    - Often already present in domino gates
    - 1 transition per 2 wires for one bit of information
  - 1-out-of-4 encoding
    - 1 transition per 4 wires for two bits of information
- Can be used with two-phase or four-phase signalling

Muller C-Element

- Also known as Rendezvous or JOIN
- David Muller (60s)
- If both inputs are 1 (0), then output becomes 1 (0)
  - If inputs differ, output remains the same
- C-element = “AND” of transitions

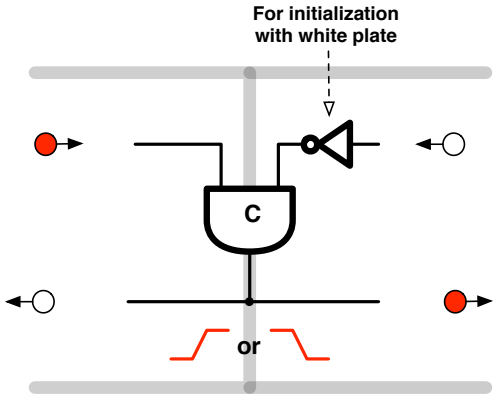






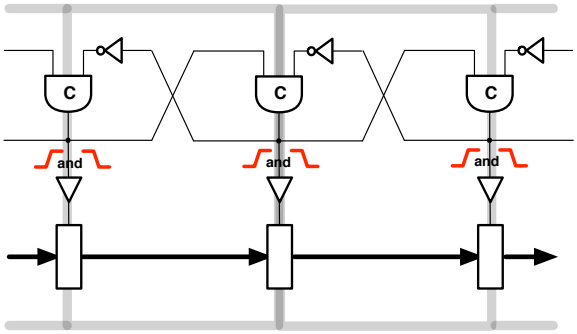
Synchronization with C-Element

- Uses two-track, two-phase signaling
- Every transition counts
- Just control:



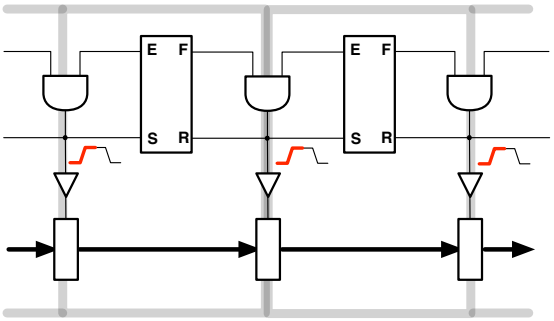
A C-element FIFO

- A simple micropipeline (Sutherland 88)
- Data must be stored with every transition
  - Makes data path more complex
- Cycle time is 5 gate delays
  - An optimized version has cycle time of 3 gate delays



An asP* FIFO

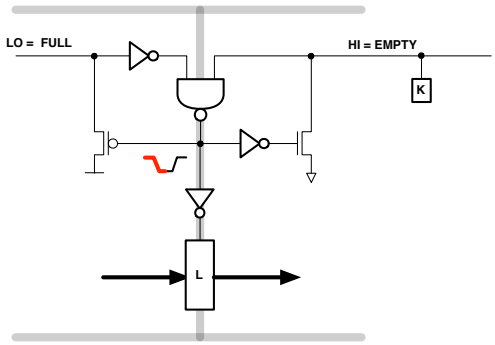
- *a*synchronous *s*ymmetric *P*ulse *P*rotocol
- Simplest control consists of SR latches and AND gates
- Control produces pulses with fixed width (3, 5, 7, .. gate delays)
- Two-track, Four-phase signaling
- Storage elements may be simple latches



E = Empty
F = False
S = Set
R = Reset

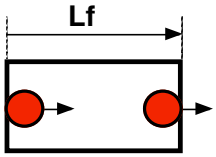
GasP

- Fast derivative of asP*, hence name
- Single-track, four-phase signaling
- Produces pulses with fixed width (3, 5, 7, .. gate delays)
- All gates have equal delay (sizing is important!)

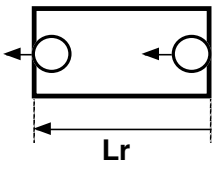


About Speed

- Forward latency of stage



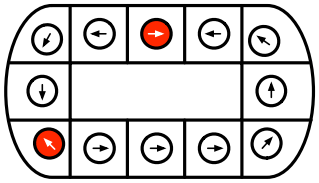
- Reverse latency of a stage

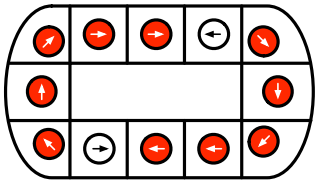


- Cycle time of a stage: Lf+Lr
- Forward latency of linear FIFO: n*Lf
- Reverse latency of linear FIFO: n*Lr
- Throughput of FIFO: 1/(Lf+Lr)

Measuring Speed

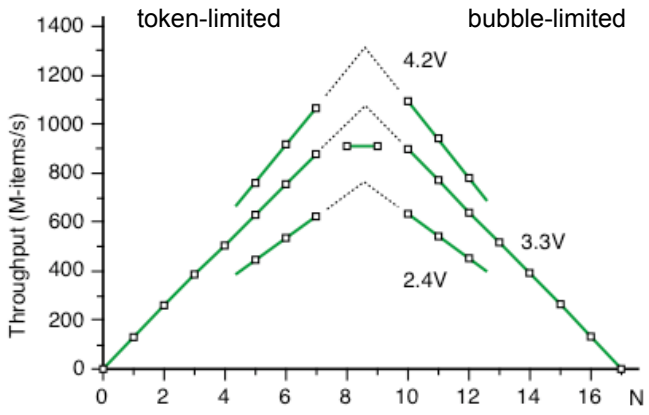
- Throughput of ring FIFOs
- "Token-limited"
  - Red plates never have to wait
  - Throughput determined by Lf
  - And # red plates
- "Bubble-limited"
  - White plates never have to wait
  - Throughput determined by Lr
  - And # white plates
- Limited by an interface stage
  - Interface stage acts as bottle neck
  - Throughput = throughput interface stage





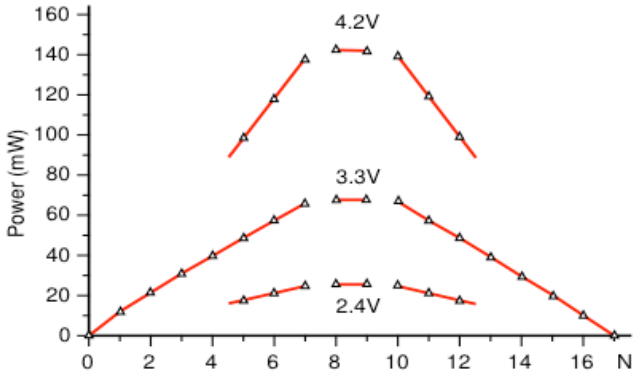
Speed Example

- Measurement from chip, asP* FIFO (600nm, 3.3V)



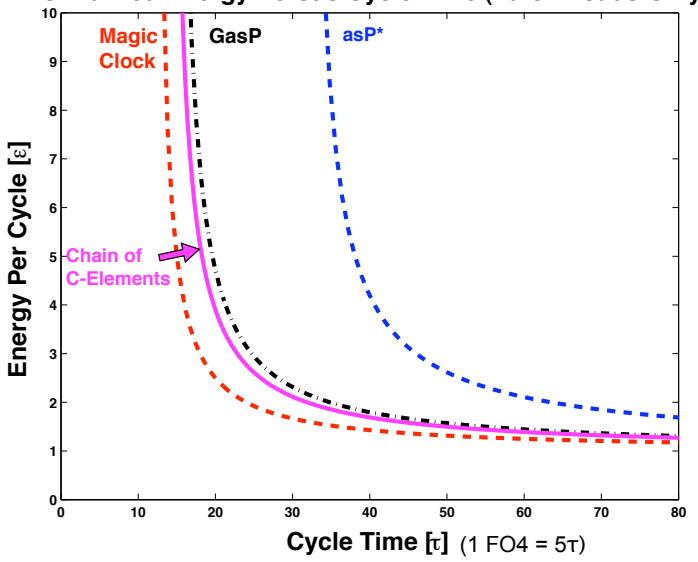
Power Example

- Measured from chip, asP* FIFO (600nm, 3.3V)
- Power proportional to throughput!



A Energy/Performance Comparison

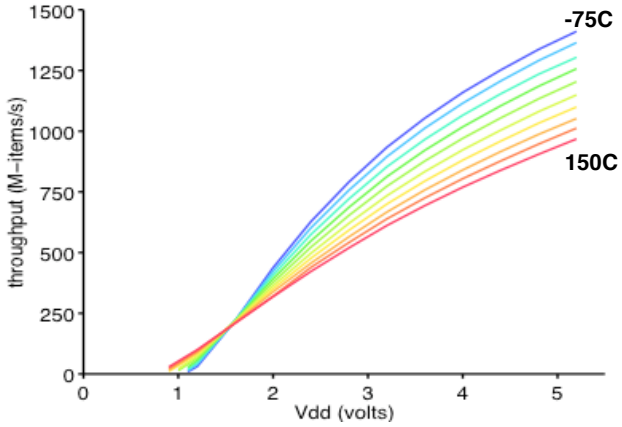
Normalized Energy Versus Cycle Time (Latch Loads Only)



1. Assumption: All gates have equal delay
2. Magic clock is infinite string of inverters with min cycle time of 6 gate delays
3. C-element FIFO has cycle time of 3 gate delays

Robustness

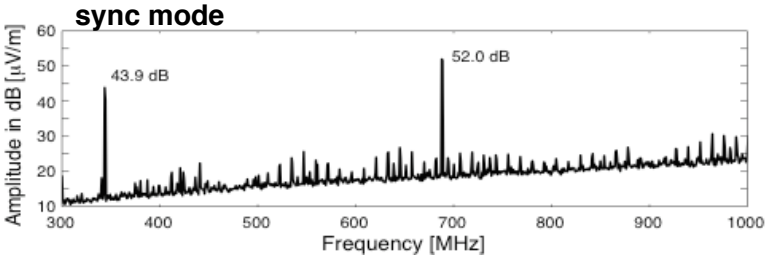
- Max throughput vs supply voltage and temperature
- C-element FIFO (600nm, 3.3V)



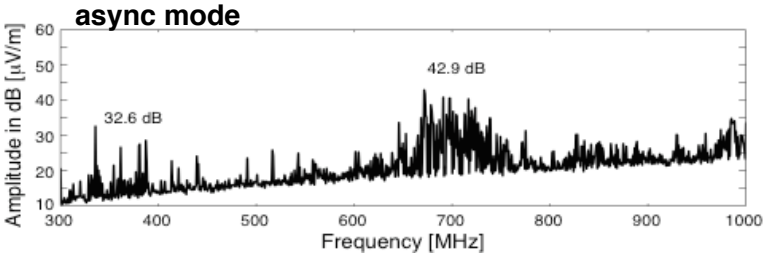
Reduced Radiation

- Frequency spectrum of the same chip operating in sync and async mode

sync mode

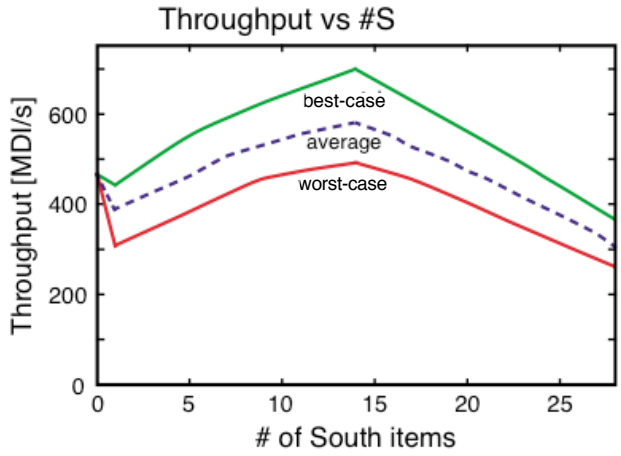


async mode



Average-Case Speed

- From asP* counterflow pipeline chip (600nm, 3.3V)
- Cycle time depends on data

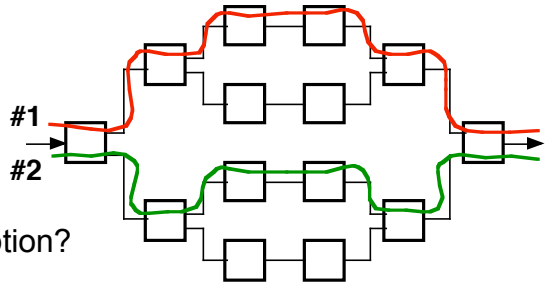


Architectural Freedom

- Can more freedom lead to better FIFOs?
- Better = lower power?
- Better = less latency?
- More concurrency?
- Different grain sizes (i.e. cycle times)?

A Non-Linear FIFO

- The idea:



- What is energy consumption?
- What is latency?
- What is cycle time of leaf cell (compared to root cell)?

Concluding Remarks

- Sun's UltraSPARCIIIi has async FIFOs
- Elsewhere..
  - Philips, IBM, Intel, DARPA, ..
  - Academia
  - Start-ups
- Many challenges remain
  - CAD tools
  - Education
  - Creating order in "chaos of concurrency"

Resources

- Async Design Group at Sun Labs
  - research.sun.com/async/
- Async Logic Home Page
  - www.cs.man.ac.edu/async/
- Ivan Sutherland's Turing Award lecture 1989
- "Computers Without Clocks" Scientific American, Aug. 2002
- ASYNC conference series