Review: Sequential Definitions

- Use two level sensitive latches of opposite type to build one master-slave flipflop that changes state on a clock edge (when the slave is transparent)
- Static storage
  - static uses a bistable element with feedback to store its state and thus preserves state as long as the power is on
  - Loading new data into the element: 1) cutting the feedback path (mux based); 2) overpowering the feedback path (SRAM based)
- Dynamic storage
  - dynamic stores state on parasitic capacitors so the state held for only a period of time (milliseconds); requires periodic refresh
  - dynamic is usually simpler (fewer transistors), higher speed, lower power but due to noise immunity issues always modify the circuit so that it is pseudostatic

Timing Classifications

- Synchronous systems
  - All memory elements in the system are simultaneously updated using a globally distributed periodic synchronization signal (i.e., a global clock signal)
  - Functionality is ensured by strict constraints on the clock signal generation and distribution to minimize
    - Clock skew (spatial variations in clock edges)
    - Clock jitter (temporal variations in clock edges)
- Asynchronous systems
  - Self-timed (controlled) systems
    - No need for a globally distributed clock, but have asynchronous circuit overheads (handshaking logic, etc.)
- Hybrid systems
  - Synchronization between different clock domains
  - Interfacing between asynchronous and synchronous domains
Review: Synchronous Timing Basics

- Under ideal conditions (i.e., when $t_{clk1} = t_{clk2}$):
  \[ T \geq t_{c-q} + t_{plogic} + t_{su} = t_{cdlogic} + t_{cdreg} \]

- Under real conditions, the clock signal can have both spatial (clock skew) and temporal (clock jitter) variations:
  - skew is constant from cycle to cycle (by definition); skew can be positive (clock and data flowing in the same direction) or negative (clock and data flowing in opposite directions)
  - jitter causes $T$ to change on a cycle-by-cycle basis

Sources of Clock Skew and Jitter in Clock Network

- Skew
  - manufacturing device variations in clock drivers
  - interconnect variations
  - environmental variations (power supply and temperature)

- Jitter
  - clock generation
  - capacitive loading and coupling
  - environmental variations (power supply and temperature)

Positive Clock Skew

- Clock and data flow in the same direction

\[ T = t_{\text{hold}} \]
Positive Clock Skew

- Clock and data flow in the same direction

\[ T + \delta \geq t_{c-q} + t_{\text{logic}} + t_{\text{pu}} \]
\[ t_{\text{hold}} = \delta = t_{\text{clogic}} + t_{\text{dreg}} \]
\[ \delta > 0: \text{Improves performance, but makes } t_{\text{hold}} \text{ harder to meet. If } t_{\text{hold}} \text{ is not met (race conditions), the circuit malfunctions independent of the clock period.} \]

Negative Clock Skew

- Clock and data flow in opposite directions

\[ T: t_{\text{hold}} = t_{\text{clogic}} + t_{\text{dreg}} - \delta \]
\[ \delta < 0: \text{Degrades performance, but } t_{\text{hold}} \text{ is easier to meet (eliminating race conditions).} \]
Clock Jitter

Jitter causes $T$ to vary on a cycle-by-cycle basis.

$T : T - 2t_{jitter} \geq t_{c-q} + t_{p_{logic}} + t_{s_{u}}$ 

Jitter directly reduces the performance of a sequential circuit.

Combined Impact of Skew and Jitter

Constraints on the minimum clock period ($\delta > 0$)

$T \geq t_{c-q} + t_{p_{logic}} + t_{s_{u}} - \delta + 2t_{jitter}$ 

$t_{hold} = t_{d_{logic}} + t_{d_{reg}} - \delta - 2t_{jitter}$ 

$\delta > 0$ with jitter: Degrades performance, and makes $t_{hold}$ even harder to meet. (The acceptable skew is reduced by jitter.)
Clock Distribution Networks
- Clock skew and jitter can ultimately limit the performance of a digital system, so designing a clock network that minimizes both is important
  - In many high-speed processors, a majority of the dynamic power is dissipated in the clock network.
  - To reduce dynamic power, the clock network must support clock gating (shutting down (disabling the clock) units)
- Clock distribution techniques
  - Balanced paths (H-tree network, matched RC trees)
    - In the ideal case, can eliminate skew
    - Could take multiple cycles for the clock signal to propagate to the leaves of the tree
  - Clock grids
    - Typically used in the final stage of the clock distribution network
    - Minimizes absolute delay (not relative delay)

H-Tree Clock Network
- If the paths are perfectly balanced, clock skew is zero

DEC Alpha 21164 (EV5)
- 300 MHz clock (9.3 million transistors on a 16.5x18.1 mm die in 0.5 micron CMOS technology)
  - single phase clock
- 3.75 nF total clock load
  - Extensive use of dynamic logic
- 20 W (out of 50) in clock distribution network
- Two level clock distribution
  - Single 6 stage driver at the center of the chip
  - Secondary buffers drive the left and right sides of the clock grid in m3 and m4
  - Total equivalent driver size of 58 cm!!
Clock Skew in Alpha Processor

- Absolute skew smaller than 90 ps
- The critical instruction and execution units all see the clock within 65 ps

Dealing with Clock Skew and Jitter

- To minimize skew, balance clock paths using H-tree or matched-tree clock distribution structures.
- If possible, route data and clock in opposite directions; eliminates races at the cost of performance.
- The use of gated clocks to help with dynamic power consumption make jitter worse.
- Shield clock wires (route power lines – VDD or GND – next to clock lines) to minimize/eliminate coupling with neighboring signal nets.
- Use dummy fills to reduce skew by reducing variations in interconnect capacitances due to interlayer dielectric thickness variations.
- Beware of temperature and supply rail variations and their effects on skew and jitter. Power supply noise fundamentally limits the performance of clock networks.
Major Components of a Computer

- Modern processor architecture styles
  - Pipelined, single issue (e.g., ARM)
  - Pipelined, hardware controlled multiple issue – superscalar
  - Pipelined, software controlled multiple issue – VLIW
  - Pipelined, multiple issue from multiple process threads - multithreaded

Basic Building Blocks

- Datapath
  - Execution units
    - Adder, multiplier, divider, shifter, etc.
    - Register file and pipeline registers
    - Multiplexers, decoders
- Control
  - Finite state machines (PLA, ROM, random logic)
- Interconnect
  - Switches, arbiters, buses
- Memory
  - Caches, TLBs, DRAM, buffers

MIPS 5-Stage Pipelined (Single Issue)
Datapath Bit-Sliced Organization

Tile identical bit-slice elements