Designers of digital circuits often want fastest performance. Means want high clock frequency. Frequency limited by longest register-to-register delay. Known as critical path. If clock is any faster, incorrect data may be stored into register. Longest path on right is 2 ns. Ignoring wire delays, and register setup and hold times, for simplicity:

$$\frac{1}{7 \text{ ns}} = 142 \text{ MHz}$$

Real wires have delay too. Must include in critical path. Example shows two paths. Each is $0.5 + 2 + 0.5 = 3$ ns. Trend: 1980s/1990s: Wire delays were tiny compared to logic delays. But wire delays not shrinking as fast as logic delays. Wire delays may even be greater than logic delays! Must also consider register setup and hold times, also add to path. Then add some time to the computed path, just to be safe. e.g., if path is 3 ns, say 4 ns instead.

A Circuit Made Have Numerous Paths

Paths can exist: In the datapath, In the controller, Between the controller and datapath. May be hundreds or thousands of paths. Timing analysis tools that evaluate all possible paths automatically very helpful.

Earlier sum-of-absolute-differences example. Started with high-level state machine. C code is even better starting point -- easier to understand.
Behavioral-Level Design: Start with C (or Similar Language)

- Replace first step of RTL design method by two steps
  - Capture in C, then convert C to high-level state machine
  - How convert from C to high-level state machine?

  Step 1A: Capture in C
  Step 1B: Convert to high-level state machine

Converting from C to High-Level State Machine

- Convert each C construct to equivalent states and transitions
  - Assignment statement
    - Becomes one state with assignment
  - If-then statement
    - Becomes state with condition check, transitioning to "then" statements if condition true, otherwise to ending state
  - "then" statements would also be converted to states

Simple Example of Converting from C to High-Level State Machine

- Convert if-then-else statement to states
- Then convert assignment statements to states

Example: Converting C code to High-Level State Machine

- Convert each construct to states
- From high-level state machine, follow RTL design method to create circuit
- Thus, can convert C to gates using straightforward automatable process
- Not all C constructs can be efficiently converted
- Use C subset if intended for circuit
- Can use languages other than C, of course

Design Challenge

- Convert the following C code, which calculates the number of times the value b is found within an array A consisting of 256 8-bit values, into a high-level state machine.
  - Due:
    - Next Lecture (Monday, October 31)
  - Extra Credit (Homework)
    - 2 points