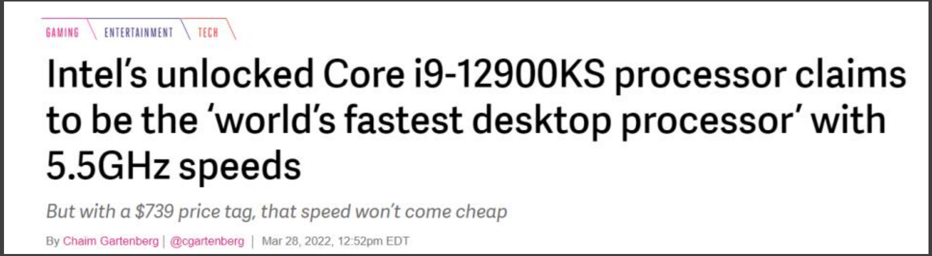
Hands on with FPGA's: Module 3

Venkat Rangan

Topics

- Pre-class: Open floor for questions
- Clocks:
  - What is a clock?
  - Why do we need a clock?
  - Interesting things about clocks
  - How are clocks used?
- Reset
  - What is a reset
  - Why do we need a Reset?
  - Variety of Reset's
- Intro to Verilog/Simulations/Synthesis
- Open discussion

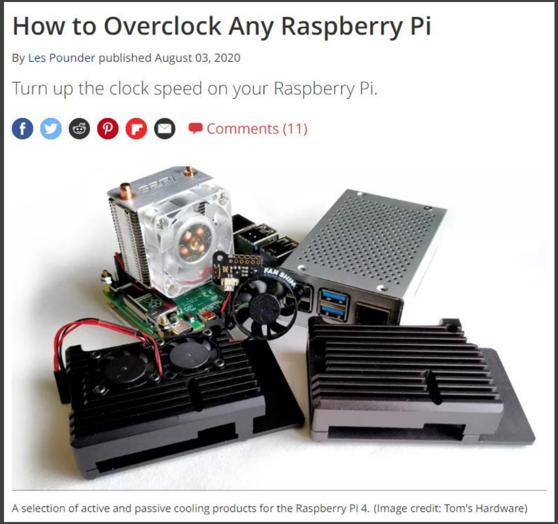
Clocks in Digital Logic circuits


GAMING ENTERTAINMENT TECH

Intel's unlocked Core i9-12900KS processor claims to be the 'world's fastest desktop processor' with 5.5GHz speeds

But with a $739 price tag, that speed won't come cheap

By Chaim Gartenberg | @cgartenberg | Mar 28, 2022, 12:52pm EDT


How to Overclock Any Raspberry Pi

By Les Pounder published August 03, 2020

Turn up the clock speed on your Raspberry Pi.

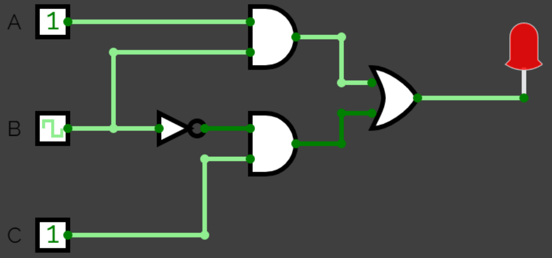
Comments (11)

A selection of active and passive cooling products for the Raspberry Pi 4. (Image credit: Tom's Hardware)

- What's the meaning of it all?
- Synchronous vs. Asynchronous design

Why clock?

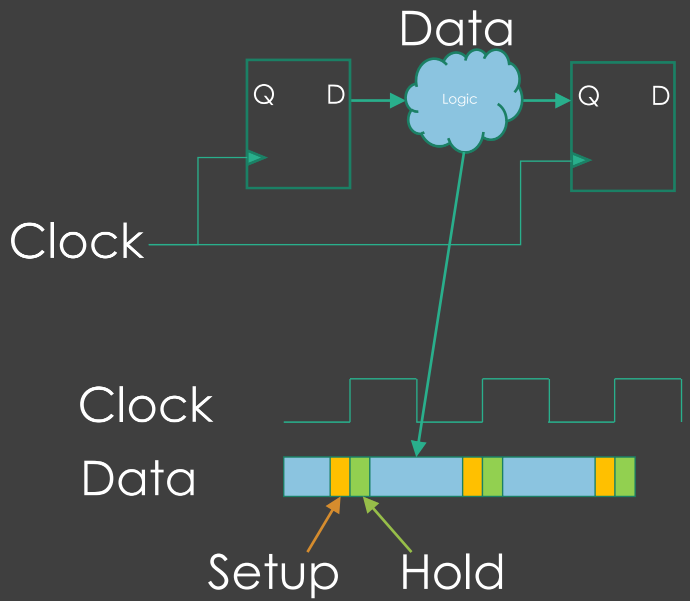
- Clocks are one way for circuits to have memory
  - SR Latch & D Flip Flop (DFF) in Circuitverse
  - DFF forms the basis of most digital logic
- Digital Heartbeat
  - Synchronization between different parts of a design
  - Every circuit knows when its time to do something
- Glitches



Important properties of a Clock

- Frequency
- Duty Cycle
- Jitter

- Setup and Hold
- Max frequency of operation



Clocks in Verilog

- How does HW actually work?
  - Everything is running in parallel
  - All changes happen on the positive edge of a clock
    - Less commonly also on negative edge or both edges (DDR DRAM)
  - Between clock edges, signals propagate between registers

Clock Sources

- RC Oscillator
- Quartz Oscillator
- Exotic Sources
    - Temperature compensated, Oven Controlled
    - GPS/Atomic Clocks

Clocks: Good design practice

- Minimize number of clocks in your design
- Have a single clock unless absolutely necessary
- Poor design vs. Good design practice

```
// simple ripple clock divider

always @(posedge clk)
  clk_div2 <= ~clk_div2;

always @(posedge clk_div2)
  clk_div4 <= ~clk_div4;

always @(posedge clk_div4)
  clk_div8 <= ~clk_div8;

always @(posedge clk_div8)
  clk_div16 <= ~clk_div16;
```

```
// simple ripple clock divider

always @(posedge clk)
  clk_div2 <= reset ? 0 : ~clk_div2;

always @(posedge clk)
  if (clk_div2)
  clk_div4 <= ~clk_div4;

always @(posedge clk)
  if (clk_div4 & clk_div2) clk_div8 <= ~clk_div8;

always @(posedge clk)
  if ( &{clk_div2, clk_div4, clk_div8}) clk_div16 <= ~clk_div16;
```

Reset

- What state does a system start in?
  - Whats the state of wires at power up?
- “x”, “0”, “1” states
- Define using an external “Reset”

Verilog Deep Dive

- Pages 3-17 of This presentation

Module 3: Clocks and Verilog basics

Open Discussion