EEC 180

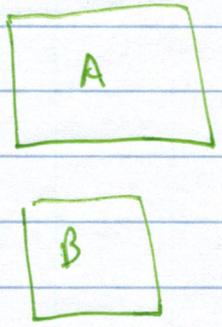
May 11, 2021

Variable-frequency clocking HW
Multi-frequency " "

Reason 1: to reduce power dissipation

Active power for CMOS = $\alpha C V^2 f$

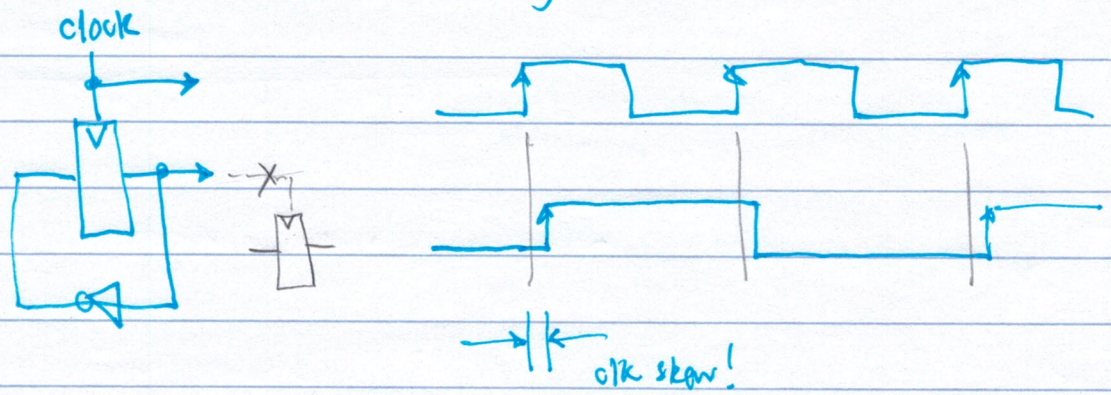
α = activity
C = capacitance
V = Vdd
f = clock frequency



Reason 2: A sub-module requires a clock freq. that is different than the main clk frequency.

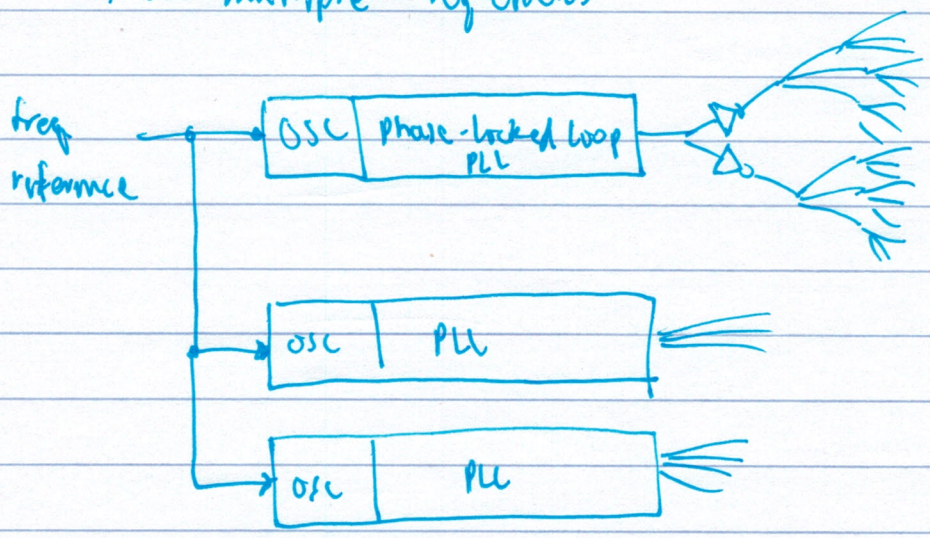
Ex. DDR4-3200 operates at 1.60 GHz
will need a clock at 1.60 GHz (or 0.80 GHz)

1) Build slower divided clocks using FFs



Never use this!

2) Use multiple-freq clocks



- How real systems are commonly built

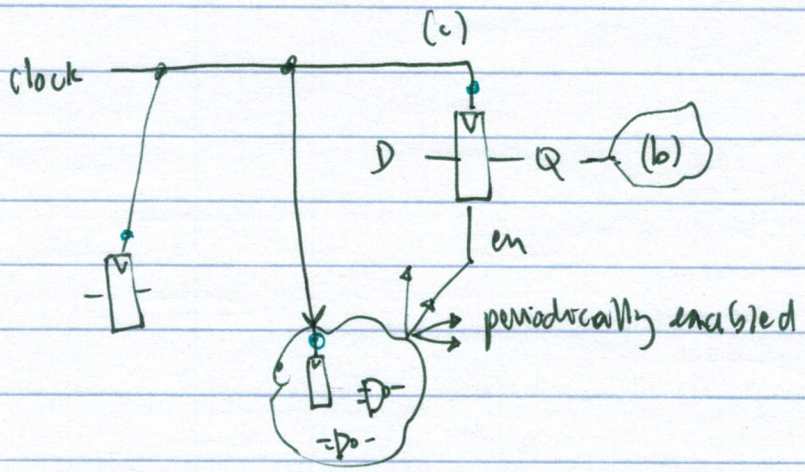
+ Likely to save power in large systems
- Requires multiple independent clocking systems
- PLLs require significant power

3) Pseudo multi rate

- Clock all FFs logic with the highest clock rate clock
- Utilize counters that load registers, or route signals on only certain clock edges

+ Simplest and most robust

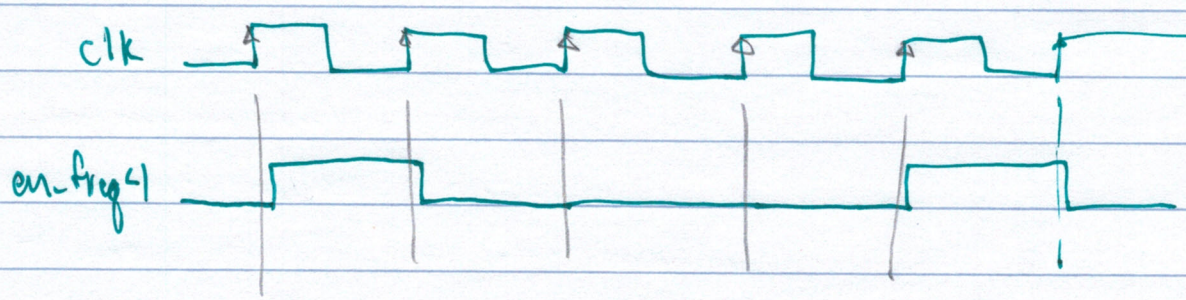
- Counters must be reset simultaneously
- Design in this way for 180



- possible issue for en signal if many FFs
  - delay of en fanout tree
  - en could be pipelined

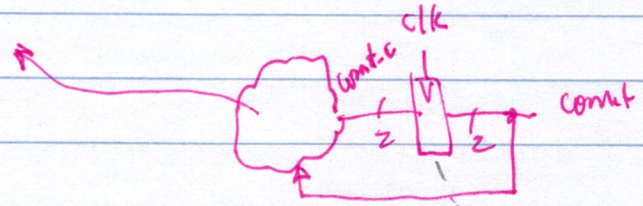
- Effectiveness:
  a) Logical operation - same as if duty was reduced ✓
  b) Power reduction of logic - " " " ✓
  c) Power reduction of clock - no reduction

Example 1a - Imitate freq/4 (e.g., 1 GHz, 250 MHz)



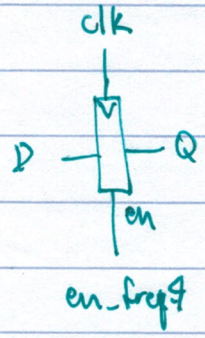

```
reg [1:0] cnt, cnt_c;
reg Q;

always @(*) begin
    cnt_c = cnt + 2'b01;
end
```

// 00 → 01 → 10 → 11 → 00 (wrap)




```
reg en_freq4;
always @(*) begin
    if (cnt == 2'b00) begin  en_freq4 = 1'b1; end
    else begin               n        = 1'b0; end
end
```

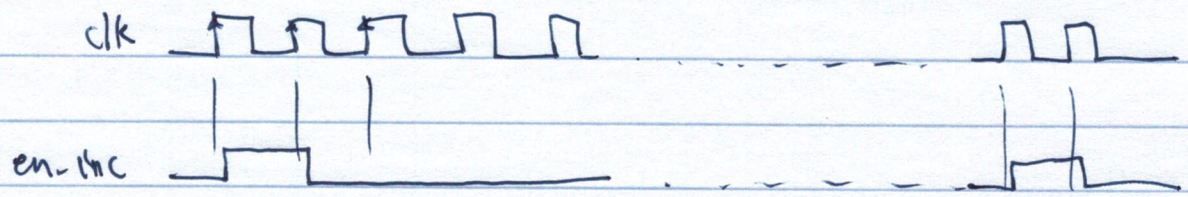
```
always @(posedge clk) begin
    cnt <= #1 cnt_c;
    if (en_freq4 == 1'b1) begin
        Q <= #1 D;
        // more registers
    end
end
```



Ex 1b:

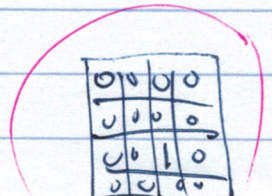
```
wire en_freq4;

assign en_freq4 = (count == 2'b00);
```

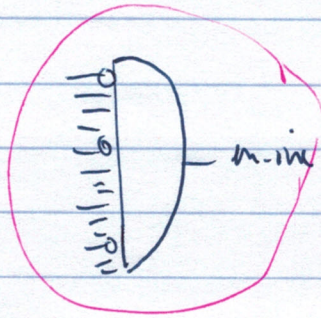
Ex 2: Imitate 1 Hz with 500 MHz clock

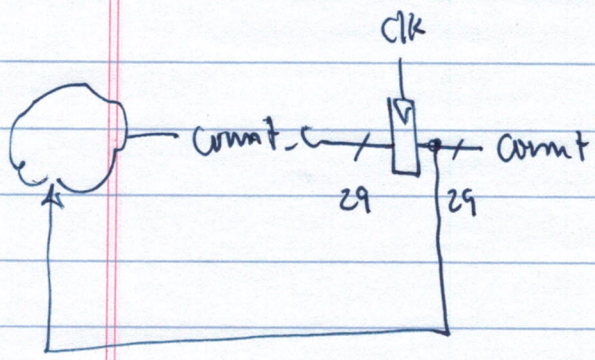


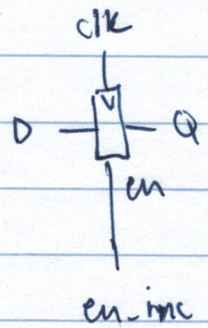
Divide by 500 million → 29 bits → 536million

```
reg [28:0]  count, count_c;
reg         en_inc;
reg         a;

always @(*) begin
    // default
    count_c  = count + 29'h0000_0001;
    en_inc   = 1'b0;

    if (count == 29'h499_999_999) begin
        count_c = 29'h0000_0000;
        en_inc  = 1'b1;
    end
end
```

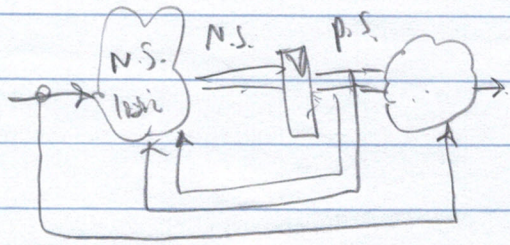









```
always @ (posedge clk) begin

    count <= #1 count_c;

    if (en_inc == 1'b1) begin
        Q <= #1 D;
    end

end
```



Next states (a b) (Moore)

Present state	00	01	10	11	output(s)
init					
a					
a0					
ab					
ab0					
aba					
aba0					