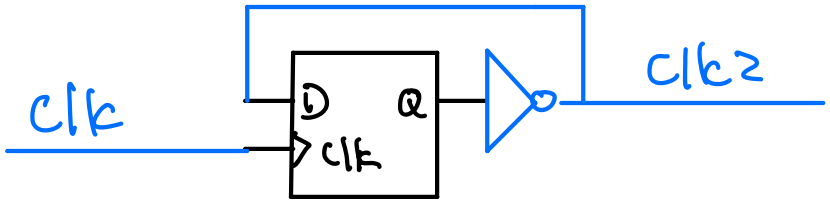
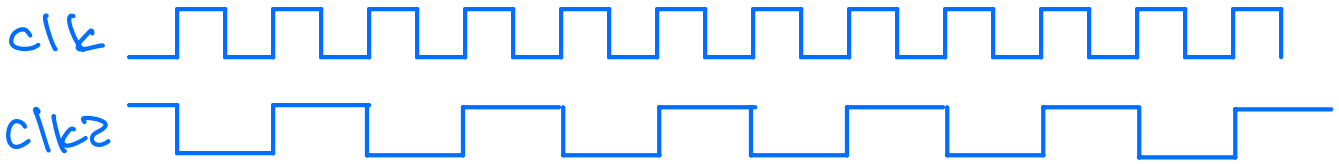
Clock divider

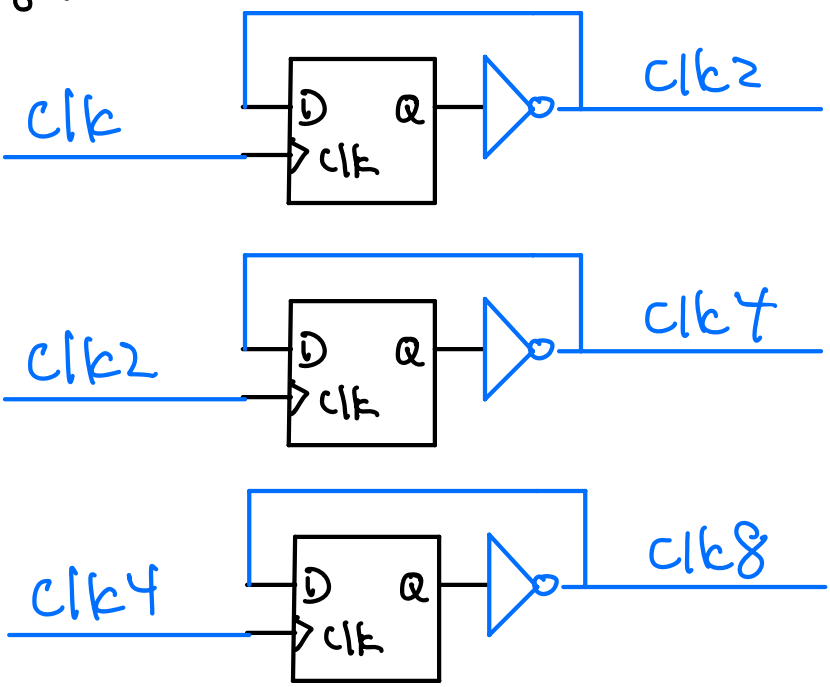


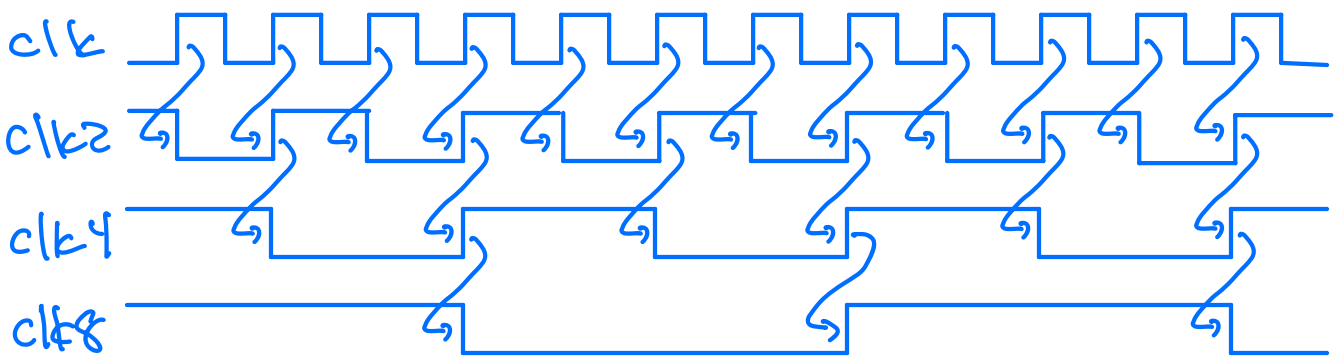


clk2 has a period $T_2 = 2T_1$ so $f_2 = f_1/2$ clock divider!

next add another divider onto clk2 → clk4,

and one on clk4 → clk8:

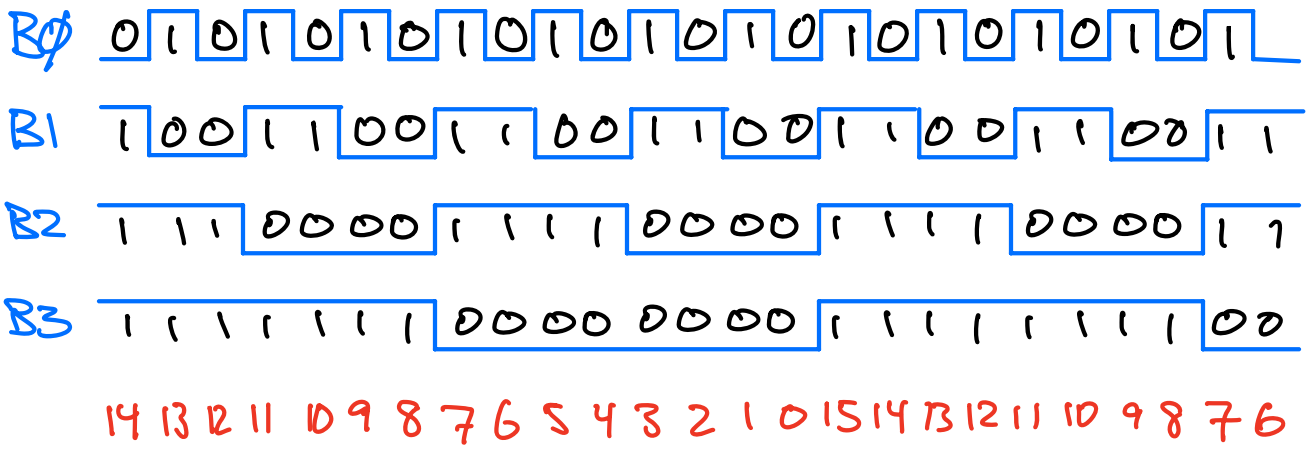




each "wire" is either 0 or 1, so look at the values of the Bus that is made from the wires

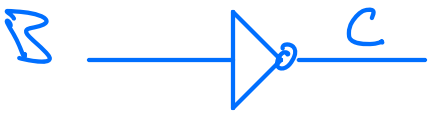
B[3:0] = {clk8, clk4, clk2, clk}

{} called a "concatenation"



B[3:0] is a "count-down" timer

if we want a "count-up" time we invert B



c[3:0] counts up

ex: we have a 1 MHz clock and want to use it to generate a pulse at ~1 Hz (not critical if it's ±10% off 1 Hz)

1 MHz → 1 ms period

1 Hz → 1000 ms period so need 1,000 "ticks" of the clock between pulses

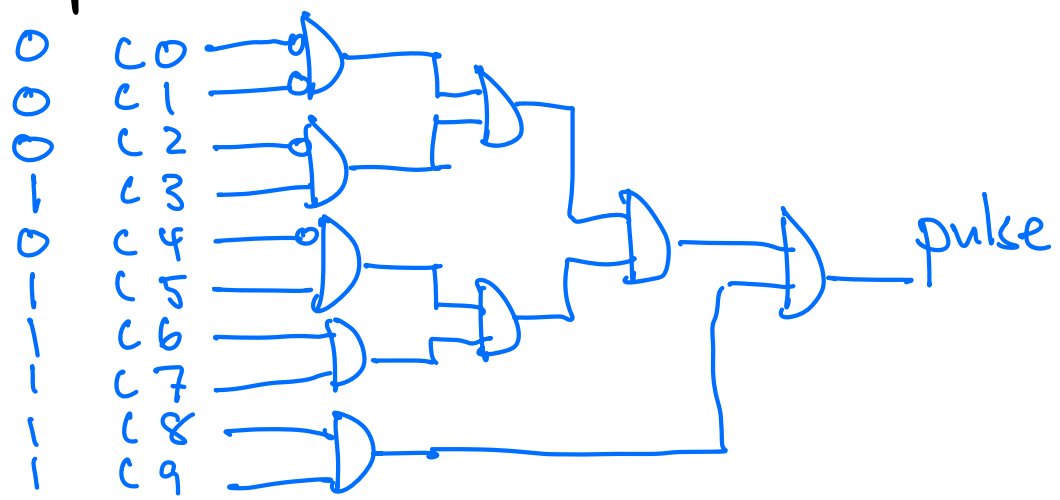
if we make a 10-bit count-up counter, max would be $2^{10} = 1024$ ticks before counter "rolls over"

decimal 1000 = 3E8 hex

= 1111101000 binary

10-bit counter C[9:0]

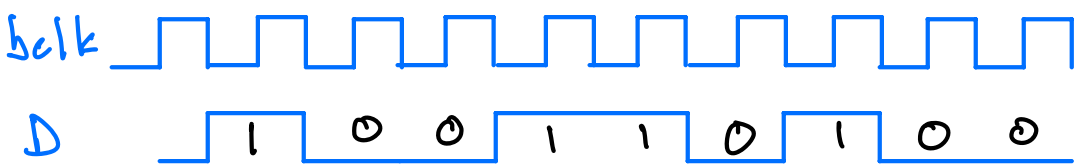
generate pulse when C = 3E8 hex



Serial data

say you want to send data on a line serially, so 1 bit of info followed by another and .... infinitum

define a "bit clock" → transition between bits (between posedge of bit clock)



"data stream" of bits, grouping into 4-bit groups "nibbles"

need a way to take these serial bits and form 4-bit "words": Q[3:0]

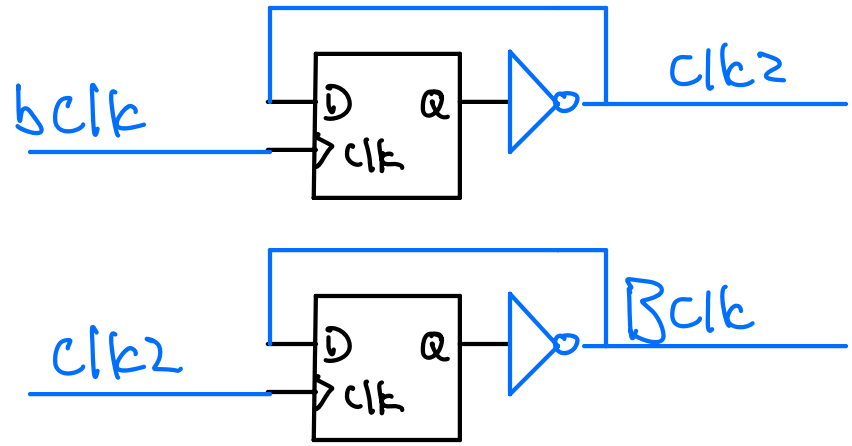
⇒ it takes 4 bit clock ticks to get 1 4-bit word

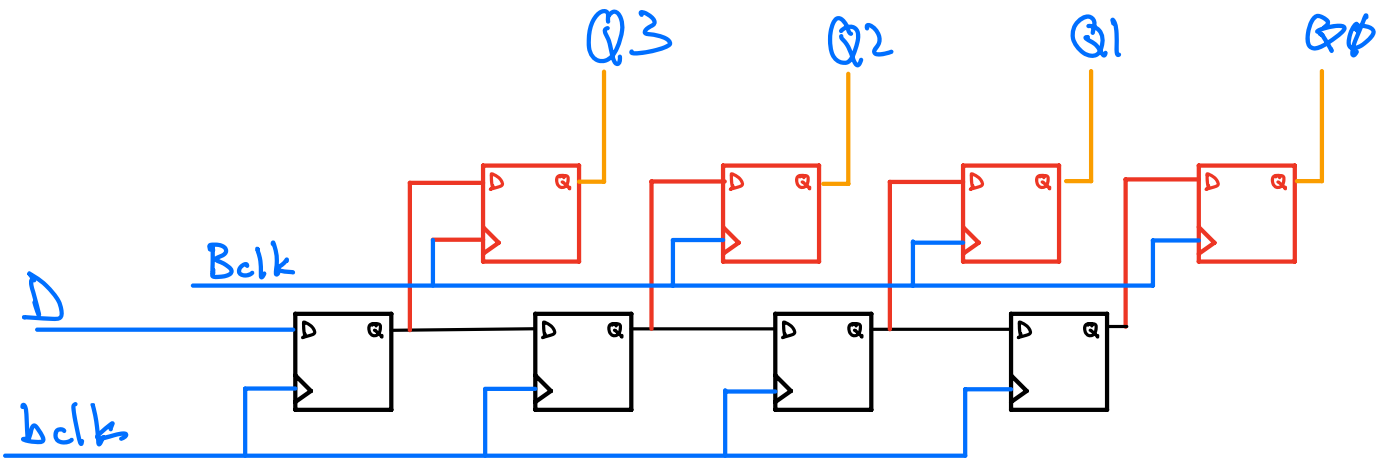
use a "shift register"

bclk = bit clock

Bclk = byte clock

(here 1 byte = 4 bits)





black DFF's are called a "shift register"

we shift data into SR w/ bclk

⇒ add red DFF's w/ Bclk to make

"serial-in parallel-out" shift register (SIPO)