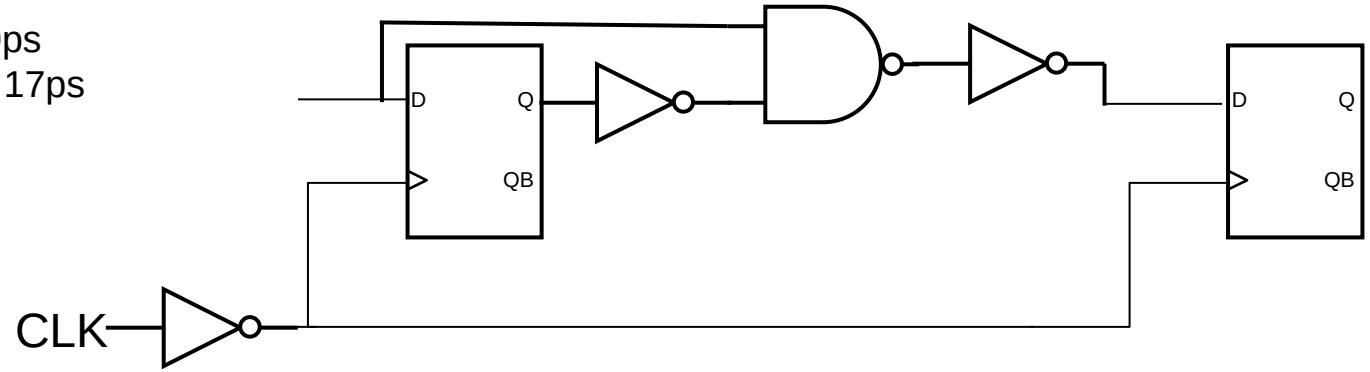
# EE 3921

Dr. Johnson

# Homework 1

1 – Calculate the fastest possible clock frequency 20pts

$t_{PD}$ INV = 10ps
$t_{PD}$ NAND = 17ps
$t_{CQ}$ = 22ps
$t_{setup}$ = 5ps
$t_{hold}$ = 2ps



2 – Word Match, Identify the best matching number on the left for each item on the right. (no duplicates) 10pts

| | | | |
|---|---|---|---|
| 1 | Analysis and Elaboration | | View Schematic |
| 2 | Analyze Current File | | State machine schematic/logic |
| 3 | Analysis and Synthesis | | Creates the physical design |
| 4 | Start Fitter | | Creates logic |
| 5 | Pin Planner | | Allows access to the physical design |
| 6 | RTL Viewer | | Checks for syntax errors |
| 7 | Signal Tap II | | Debug Tool |
| 8 | Model Sim | | Simulates RTL |
| 9 | State Machine Viewer | | Creates RTL |
| 10 | Chip Planner | | Assign/view pins |

3) Given the VHDL code below and the indicated inputs, provide the expected signal values

30pts

a=1, b=0, c=0, d=1
e=0110, f=1001
g=0111, h=1110

i=

aa=

j=

bb=

p=

cc=

q=

ee=

r=

ff=

s=

gg=

u=

hh=

v=

```
--------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.all;

-- IGNORING NAMING CONVENTION --

entity arithHW is
generic( NNN:  INTEGER  := 4);
   port  (  a,b,c,d:     in STD_LOGIC;
            e,f,g,h:     in STD_LOGIC_VECTOR(NNN-1 downto 0);
            i,j,k,l:     out STD_LOGIC;
            m,n:         out STD_LOGIC_VECTOR(NNN-1 downto 0);
            o,p,z:       out STD_LOGIC_VECTOR(2*NNN-1 downto 0);

            foo:         out std_logic_vector(65 downto 0)

         );
end entity;

Architecture behavioral of arithHW is

   signal q,r,s,t,u,v:        SIGNED((NNN-1) downto 0);
   signal aa,bb,cc,dd,ee,ff:  UNSIGNED((NNN-1) downto 0);
   signal gg:                 SIGNED((2*NNN-1) downto 0);
   signal hh:                 UNSIGNED((2*NNN-1) downto 0);

   begin
      q <= SIGNED(f);
      r <= SIGNED(g);
      aa <= UNSIGNED(f);
      bb <= UNSIGNED(g);

      i <= ((a or b) and c) xor d;
      j <= a and (e(1) or e(2));

      s <= q + r;
      gg <= q * r;
      u <= q / r;
      v <= q mod r;

      cc <= aa + bb;
      hh <= aa * bb;
      ee <= aa / bb;
      ff <= aa mod bb;

      p <= ('1' & h(2) & "00" & g(3 downto 1) & '1');

end architecture;
```

4 – Write your own behavioral VHDL code for a JK Flip-Flop
Provide code and a simulation 40pts