Tentti: 17.11.2009

Tehtävä 1

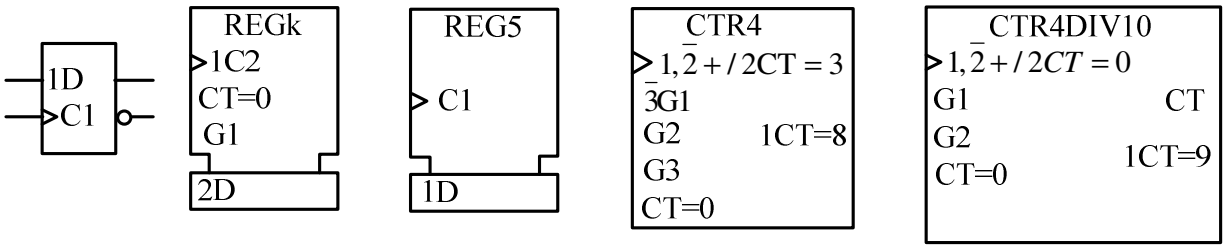
Suunnittele digitaalilogiikka, joka määrittää loogisen tulosignaalin INPUT jaksonajan ja tallettaa määritetyn jaksonajan etumerkittömänä binäärilukuna rekisteriin signaalin INPUT jakson tahtiin. Signaali INPUT on asynkroninen (tahdistamaton) pääkellosignaalin CLK suhteen, ja signaalin INPUT pulssisuhde on noin 50/50, eli signaali on kanttiaaltoa. Käytössäsi on CTRnDIVm-laskuri, jossa on kellotulon lisäksi synkroninen alustustulo, ja k-bittinen rekisteri REGk, jossa on latauksensallintatulo. Lisäksi voit käyttää kuvan mukaisia D-kiikkuja, jossa on sekä ei-invertoitu että invertoitu lähtö, ja yhden muun 2-tuloisen logiikkaportin (AND,OR,NAND,NOR,XOR,XNOR). Käytä logiikkasymboleille oheisten esimerkkien kaltaisia merkintöjä.

a) miten varmistat, että signaalin INPUT asynkronisuus ei aiheuta toiminnan epävarmuutta? (1 p)

b) esitä logiikan RTL-tason kuvaus (2 p)

c) esitä toiminnan ajoituskaavio/pöytäsimulointi, kun signaalin INPUT jaksonaika on neljä pääkellon CLK jaksoa. Esitä ajoituskaaviossa ainakin signaalit CLK, INPUT, laskurin lähtö, rekisterin lähtö, laskurin sallintasignaali ja rekisterin sallintasignaali. Ajoituskaavion on oltava vähintään 10 CLK:n jaksoa (1 p)

d) määritä parametrit m, n ja k, sekä pääkellon taajuus f_{CLK}, kun signaalin INPUT jaksonaika pitää pystyä määrittämään 10 mikrosekunnin erottelutarkkuudella, ja signaalin INPUT maksimi jaksonaika voi olla 1 sekunti. (1 p)

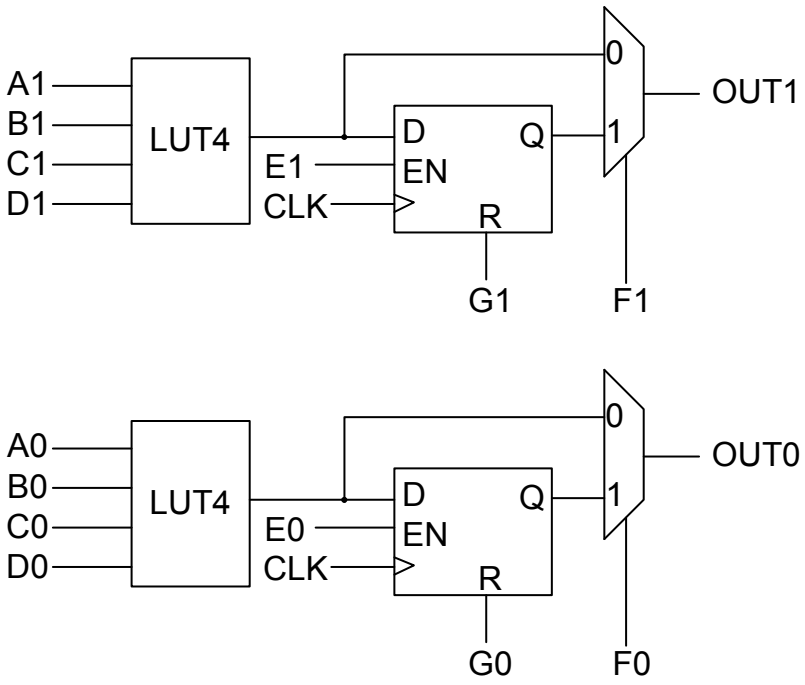


Tehtävä 2

Alla on RTL geneerisistä FPGA:n peruskomponenteista, logiikkaelementeistä (LE), joita FPGA:t ovat täynnä. Kuvassa kaikki A:ista G:ihin, OUT:it sekä FPGA-piirin I/O-portit voidaan kytkeä halutulla tavalla toisiinsa kytkentämatriisin kautta. CLK on globaali kellotulo tässä esimerkissä. Hakutaulukot (LUT) voidaan ohjelmoida sisältämään haluttu sisältö uudelleenkonfiguroinnin yhteydessä.

Selitä vastauksesi riittävällä tarkkuudella: Selitä miten lohkot liittyvät toisiinsä (mistä minne) ja mahdolliset LUT:ien sisältö. Kerro myös myös kaikkien käytössä olevien LE:iden kaikki kytkennät.

a) Suunnittele 2-bittinen siirtorekisteri, jossa on asynkroninen reset ja rinnakkainen ulostulo. Ohjaus G1 mahdollistaa siirron. Piirrä myös RTL kyseisestä SRG:stä.
b) Suunnittele 2-in-1 MUX FPGA arkkitehtuuriin.



Kuva 1. Kaksi geneeristä LE:tä.

Tentti: 17.11.2009

Tehtävä 3

a) Luettele digilogiikan synkronisuuden tunnusmerkit.
b) Mitä CMOS-teknologialla toteutetun integroidun piirin piirikuvioinnin viivanleveydellä tarkoitetaan?
c) Mitä integroidun digitaalipiirin piipinta-alan arvioinnissa käytetty *Cell Factor* –tarkoittaa?
d) Selitä minkälainen FPGA-piirin hakutaulukko on loogiselta rakenteeltaan?

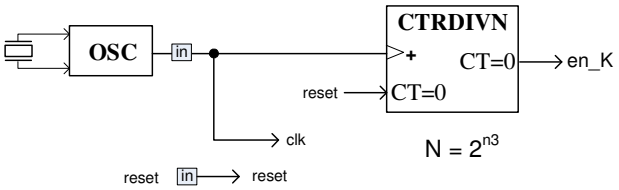
Tehtävä 4

Liitteessä 1 näet RTL-arkkitehtuurin ja liitteessä 2 sitä vastaavan VHDL-mallin. Kysymyksessä on Multiply-Cumulate-Add (MAC) operaation toteuttava digilogiikka.

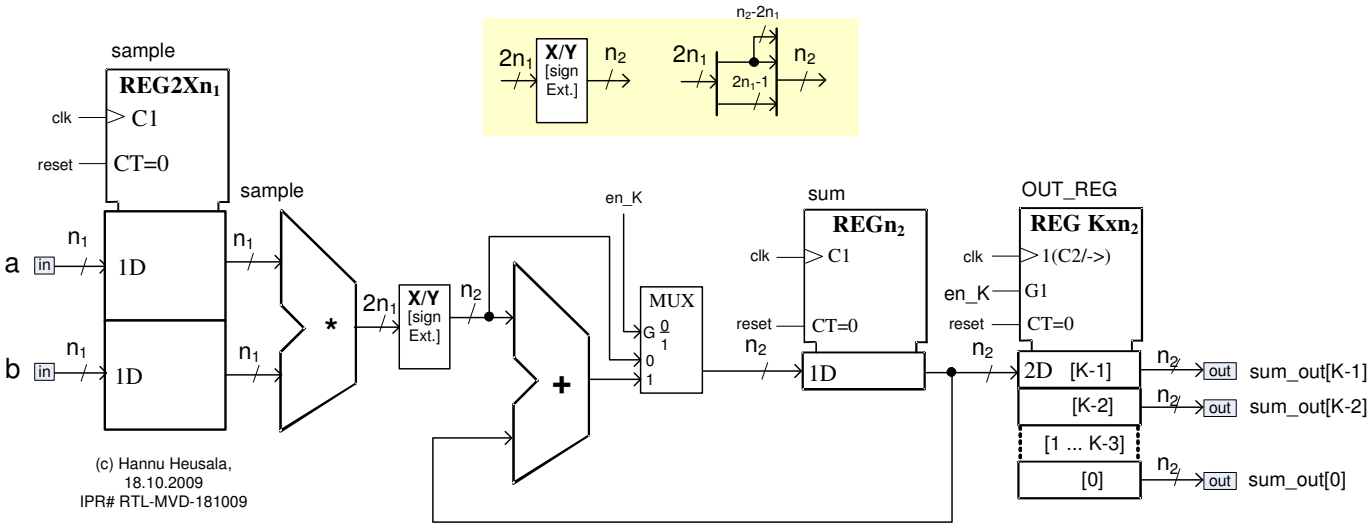
Arkkitehtuurista saadaan sanan- ja laskentasekvenssin pituuksiltaan erilaisia versioita muuttamalla parametrien (n_1, n_2, n_3 ja K) arvoja. Oletetaan, että K = N/2.

a) Päättele arkkitehtuurin ja VHDL-mallin perusteella mikä on parametrin n_2 optimaalinen arvo, jos n_1 on annettu.
b) Oletetaan, että liitteissä 1 ja 2 kuvattu arkkitehtuuri konfiguroidaan FPGA-piiriin siten, että kertojana käytetään valmista 18x18-bitin lohkokertojaa ja summaimena 4-tuloisilla hakutaulukoilla (4-LUT) toteutettua *ripple-carry* –tyyppistä rakennetta. Jos parametri $n_1 = 8$, n_2:lla on optimaalinen arvonsa ja $n_3 = 4$, niin montako D-kiikkua ja hakutaulukkoa (4-LUT) toteuttamiseen tarvitaan?
c) Minkä komponenttien kautta arkkitehtuurin kriittinen polku kulkee? Mitkä parametrit (n_1, n_2, n_3 ja K) vaikuttavat kriittisen polun pituuteen ja miten?
d) Kirjoita parametrinen (n_1, n_2, n_3 ja K) avulla kaksi matemaattista lauseketta: toinen D-kiikkujen ja toinen hakutaulukkojen lukumäärän arvioimista varten.

LIITE 1: Tehtävään 4 liittyvä RTL-arkkitehtuuri:



$$S_k = \sum_{n=0}^{n=N-1} a(k,n) \cdot b(k,n)$$



LIITE 2:Tehtävän 4 RTL-arkkitehtuuria vastaava VHDL-malli:

```
library ieee; use ieee.std_logic_1164.all; use ieee.std_logic_signed.all;
--
-- (c) Hannu Heusala 18-10-2009
-- MAC design for DT cources
-- Verification: 18-10-2009, RTL-VHDL-hhh-181009
--
entity MAC is
  generic(n1:natural:=8; n2:natural:=20;n3:natural:=4;K:natural:=8);
  port(clk,reset:in std_logic;
  sample_in_a,sample_in_b:in std_logic_vector(n1-1 downto 0);
  sum_out:out std_logic_vector(n2-1 downto 0));
end entity MAC;
--
architecture RTL1 of MAC is
type OUT_REG_type is array(K-1 downto 0) of std_logic_vector(n2-1 downto 0);
signal sample_a, sample_b:std_logic_vector(n1-1 downto 0);
signal product: std_logic_vector(2*n1-1 downto 0);
signal sum,P:std_logic_vector(n2-1 downto 0);
signal OUT_REG:OUT_REG_type;
signal CTR: std_logic_vector(n3-1 downto 0);
--
begin
  sync:process(clk,reset)
    variable en_K:std_logic:='0';
  begin
    if CTR=0 then en_K:='1'; else en_K:='0';end if;
    if reset='1' then
      sample_a<=(others=>'0');sample_b<=(others=>'0');
      sum<=(others=>'0');OUT_REG<=(others=>(others=>'0'));
      CTR <= (others=>'0');
     elsif rising_edge(clk)then
       sample_a<=sample_in_a;  sample_b<=sample_in_b;
       sum<=P+sum;  CTR<=CTR+'1';
      if en_K='1' then
        sum<=P;
        OUT_REG(K-1)<=sum;
        OUT_REG(K-2 downto 0)<=OUT_REG(K-1 downto 1);
      else OUT_REG<=OUT_REG;
      end if;
    end if;
   end process sync;
   --
   product<=sample_a*sample_b;
   --
    --sign extend:
    P(n2-1 downto 2*n1-1)<=(others => product(2*n1-1));
    P(2*n1-2 downto 0)<= product(2*n1-2 downto 0) ;
    --
    sum_out<=OUT_REG(0);
--
  end architecture RTL1;
```