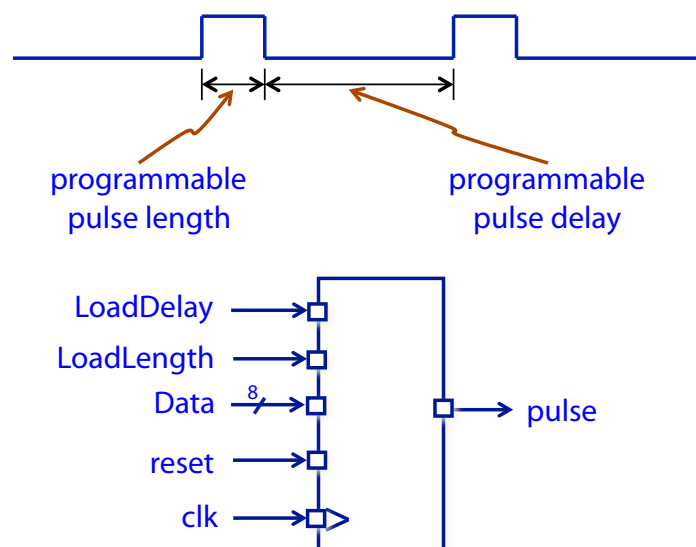
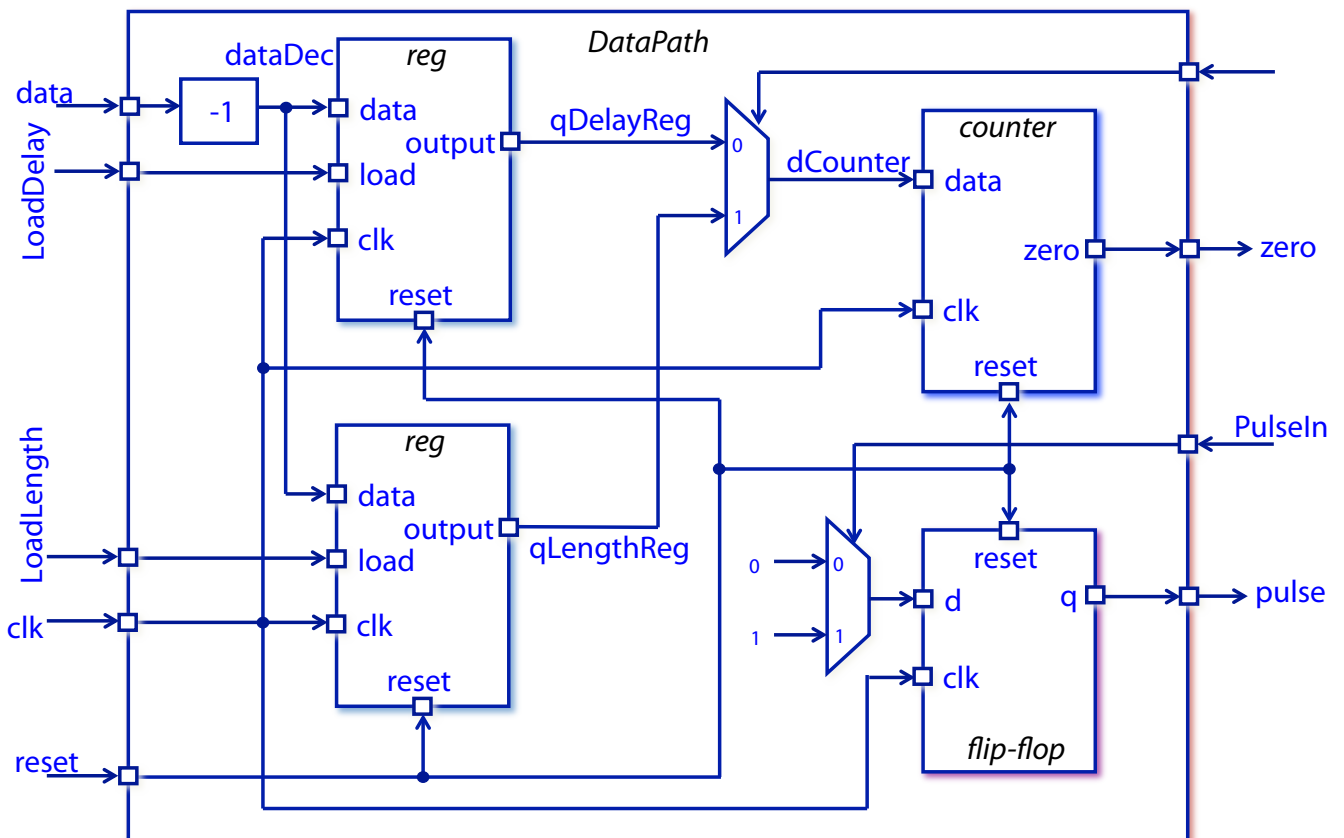
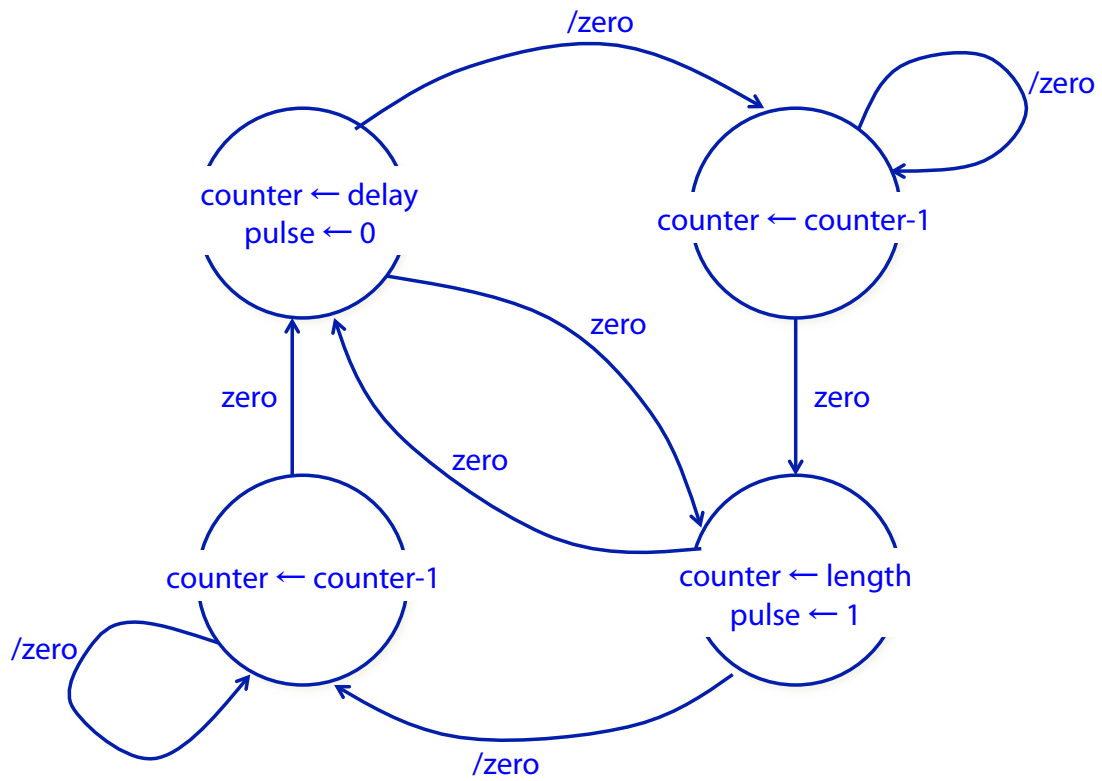


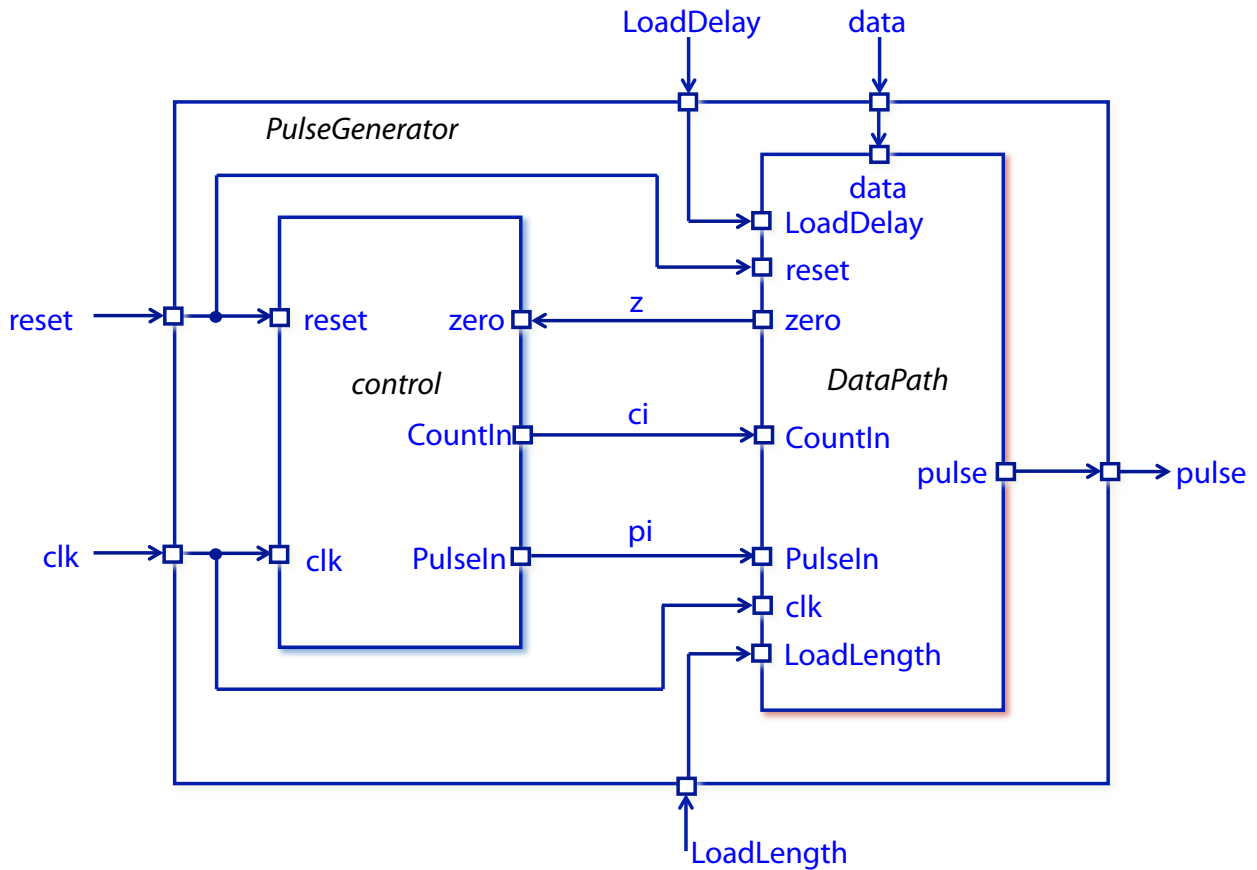
# A Programmable Pulse Generator

Eduardo Sanchez  
EPFL – HEIG-VD

- To design a programmable pulse generator. The delay between pulses, and the length of duration of each pulse is programmable







Eduardo Sanchez

5

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

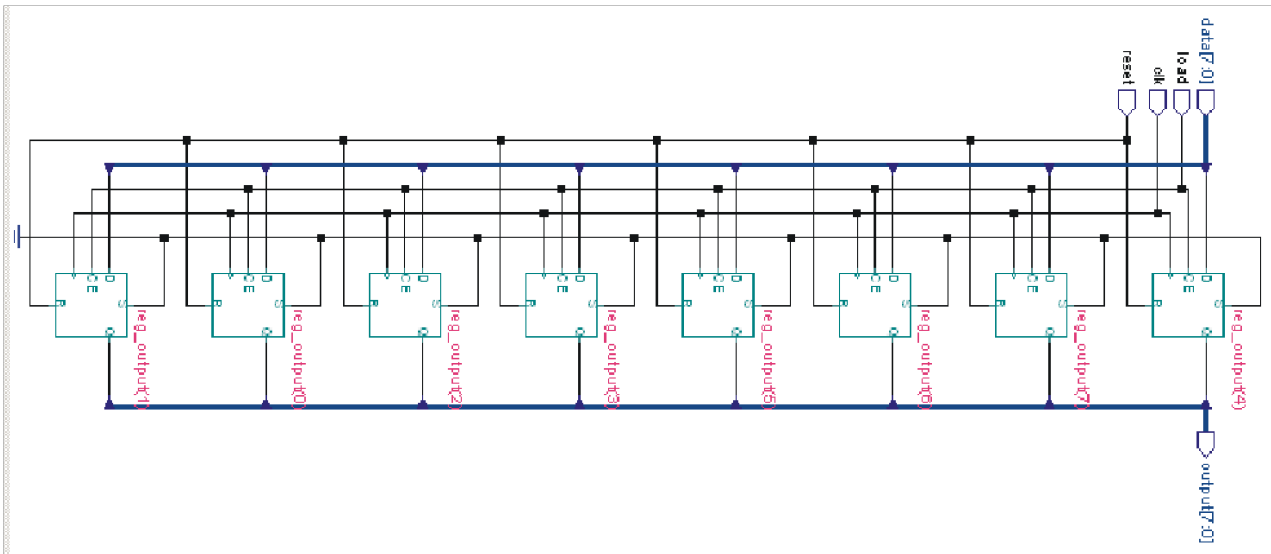
entity reg is
  port (clk      : in std_logic;
        reset    : in std_logic;
        data     : in std_logic_vector (7 downto 0);
        load     : in std_logic;
        output   : out std_logic_vector (7 downto 0));
end reg;

architecture course of reg is
begin
  process (clk, reset)
  begin
    if reset='1'
    then output <= (others => '0');
    else if (clk'event and clk='1')
    then if load='1'
    then output <= data;
    end if;
    end if;
  end if;
end process;
end course;

```

Eduardo Sanchez

6



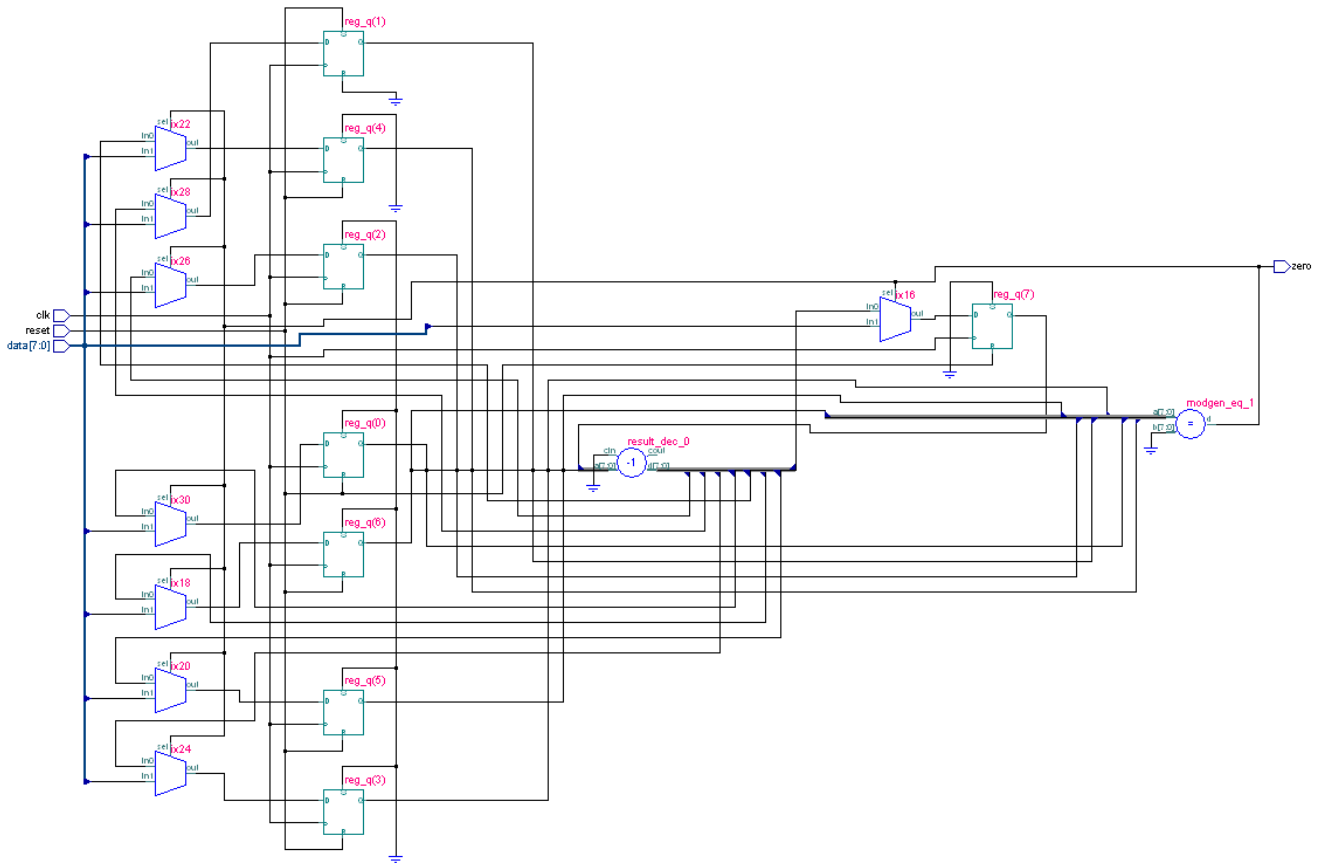
```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity counter is
    port (clk      : in std_logic;
          reset    : in std_logic;
          data     : in std_logic_vector (7 downto 0);
          zero     : out std_logic);
end counter;

architecture course of counter is
    signal z : std_logic;
    signal q : std_logic_vector (7 downto 0);
begin
    process (clk, reset)
    begin
        if reset='1'
            then q <= ("00000010");
        else if (clk'event and clk='1')
            then if z='1'
                    then q <= data;
                else q <= q - 1;
                end if;
            end if;
        end if;
    end process;
    z <= '1' when q="00000000"
        else '0';
    zero <= z;
end course;

```



Eduardo Sanchez

9

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use work.PulsePackage.all;

entity DataPath is
  port (clk          : in std_logic;
        reset       : in std_logic;
        LoadDelay   : in std_logic;
        LoadLength  : in std_logic;
        data        : in std_logic_vector (7 downto 0);
        pulse       : out std_logic;
        CountIn     : in std_logic;
        PulseIn     : in std_logic;
        zero        : out std_logic);
end DataPath;

```

Eduardo Sanchez

10

```

architecture course of DataPath is
    signal dataDec, dCounter, qDelayReg, qLengthReg : std_logic_vector (7 downto 0);

begin

    dataDec <= data - 1;
    dReg: reg port map (clk => clk,
                       reset => reset,
                       data => dataDec,
                       load => LoadDelay,
                       output => qDelayReg);
    lreg: reg port map (clk => clk,
                       reset => reset,
                       data => dataDec,
                       load => LoadLength,
                       output => qLengthReg);
    dCounter <= qLengthReg when CountIn='1'
                else qDelayReg;
    cnt: counter port map (clk => clk,
                           reset => reset,
                           data => dCounter,
                           zero => zero);

```

Eduardo Sanchez

11

```

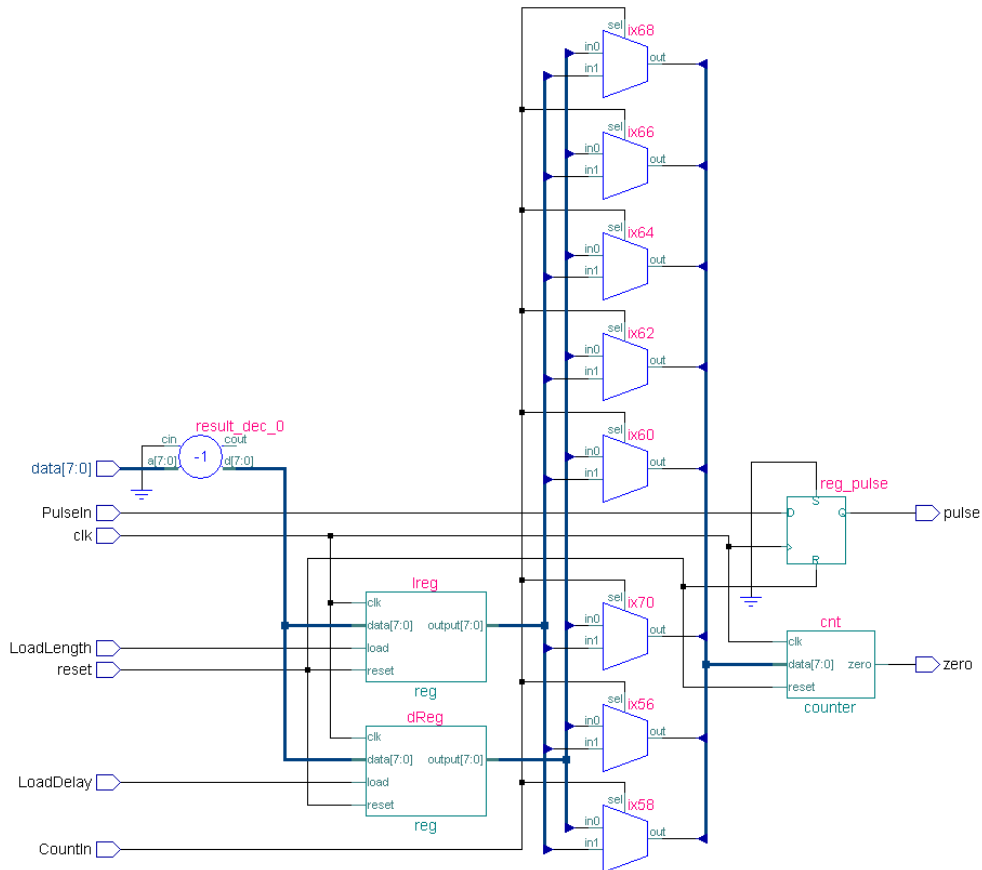
process (clk, reset)
begin
    if reset='1'
    then pulse <= '0';
    else if (clk'event and clk='1')
    then if PulseIn='1'
    then pulse <= '1';
    else pulse <= '0';
    end if;
    end if;
    end if;
end process;

end course;

```

Eduardo Sanchez

12



Eduardo Sanchez

13

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity control is
  port (clk      : in std_logic;
         reset    : in std_logic;
         zero     : in std_logic;
         CountIn  : out std_logic;
         PulseIn  : out std_logic);
end control;

```

Eduardo Sanchez

14

```

architecture course of control is
    type state is (init, delayDec, lengthLd, lengthDec);
    signal CurrentState, NextState : state;

begin

process (CurrentState, zero)
begin
    CountIn <= '0';
    PulseIn <= '0';
    NextState <= init;
    case CurrentState is
        when init => if zero='1'
            then NextState <= lengthLd;
            else NextState <= delayDec;
            end if;
        when delayDec => CountIn <= '1';
            if zero='1'
                then NextState <= lengthLd;
                else NextState <= delayDec;
                end if;

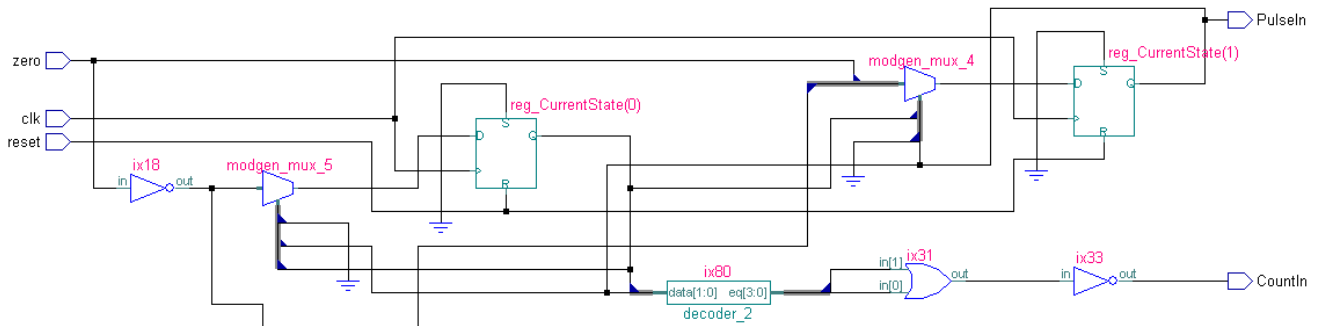
        when lengthLd => PulseIn <= '1';
            CountIn <= '1';
            if zero='0'
                then NextState <= lengthDec;
                end if;
        when lengthDec => PulseIn <= '1';
            if zero='0'
                then NextState <= lengthDec;
                end if;

    end case;
end process;

process (clk, reset)
begin
    if reset='1'
        then CurrentState <= init;
        elsif (clk'event and clk='1')
            then CurrentState <= NextState;
        end if;
end process;

end course;

```



```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

package PulsePackage is

component reg
    port (clk      : in std_logic;
          reset   : in std_logic;
          data    : in std_logic_vector (7 downto 0);
          load    : in std_logic;
          output  : out std_logic_vector (7 downto 0));
end component;

component counter
    port (clk      : in std_logic;
          reset   : in std_logic;
          data    : in std_logic_vector (7 downto 0);
          zero    : out std_logic);
end component;

```

```

component DataPath
  port (clk      : in std_logic;
        reset    : in std_logic;
        LoadDelay : in std_logic;
        LoadLength : in std_logic;
        data      : in std_logic_vector (7 downto 0);
        pulse     : out std_logic;
        CountIn   : in std_logic;
        PulseIn   : in std_logic;
        zero      : out std_logic);
end component;

component control
  port (clk      : in std_logic;
        reset    : in std_logic;
        zero     : in std_logic;
        CountIn  : out std_logic;
        PulseIn  : out std_logic);
end component;

end PulsePackage;

```

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use work.PulsePackage.all;

entity PulseGenerator is
  port (clk      : in std_logic;
        reset    : in std_logic;
        LoadDelay : in std_logic;
        LoadLength : in std_logic;
        data      : in std_logic_vector (7 downto 0);
        pulse     : out std_logic);
end PulseGenerator;

```

```

architecture course of PulseGenerator is
    signal z, ci, pi : std_logic;

begin
    u1: control port map (clk => clk,
        reset => reset,
        zero => z,
        CountIn => ci,
        PulseIn => pi);

    u2: DataPath port map (clk => clk,
        reset => reset,
        LoadDelay => LoadDelay,
        LoadLength => LoadLength,
        data => data,
        pulse => pulse,
        CountIn => ci,
        PulseIn => pi,
        zero => z);

end course;

```

