<http://electronics.stackexchange.com/questions/61422/how-to-divide-50mhz-down-to-2hz-in-vhdl-on-xilinx-fpga>

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.all;

USE IEEE.STD\_LOGIC\_unsigned.all;

ENTITY trafficlightwithcrosswalk IS

PORT (CLOCK\_50: IN STD\_LOGIC;

clr: IN STD\_LOGIC;

LEDR: OUT STD\_LOGIC\_VECTOR (5 DOWNTO 0);

LEDG: OUT STD\_LOGIC\_VECTOR (5 DOWNTO 0));

END trafficlightwithcrosswalk;

ARCHITECTURE stateMachine OF trafficlightwithcrosswalk IS

TYPE state\_type IS (s0, s1, s2, s3, s4, s5);

SIGNAL state: state\_type;

SIGNAL count: STD\_LOGIC\_VECTOR (3 DOWNTO 0);

CONSTANT SEC5: STD\_LOGIC\_VECTOR (3 DOWNTO 0) := "1111";

CONSTANT SEC1: STD\_LOGIC\_VECTOR (3 DOWNTO 0) := "0011";

BEGIN

PROCESS (CLOCK\_50, clr)

BEGIN

IF clr = '1' THEN state <= s0;

count <= X"0";

ELSIF (CLOCK\_50'EVENT AND CLOCK\_50 = '1') THEN

CASE state IS

WHEN s0 =>

IF count < SEC5 THEN

state <= s0;

count <= count +1;

ELSE

state <=s1;

count <= X"0";

END IF;

WHEN s1 =>

IF count < SEC1 THEN

state <= s1;

count <= count + 1;

ELSE

state <= s2;

count <= X"0";

END IF;

WHEN s2 =>

IF count < SEC1 THEN

state <= s2;

count <= count + 1;

ELSE

state <= s3;

count <= X"0";

END IF;

WHEN s3 =>

IF count < SEC5 THEN

state <= s3;

count <= count + 1;

ELSE

state <= s4;

count <= X"0";

END IF;

WHEN s4 =>

IF count < SEC1 THEN

state <= s4;

count <= count + 1;

ELSE

state <= s5;

count <= X"0";

END IF;

WHEN s5 =>

IF count < SEC1 THEN

state <= s5;

count <= count + 1;

ELSE

state <= s0;

count <= X"0";

END IF;

WHEN OTHERS => STATE <= s0;

END CASE;

END IF;

END PROCESS;

C2: PROCESS(state)

BEGIN

CASE state is

when s0 => LEDR <= "000001"; LEDG <= "001000";

when s1 => LEDR <= "000001"; LEDG <= "110000";

when s2 => LEDR <= "001001"; LEDG <= "000000";

when s3 => LEDR <= "001000"; LEDG <= "000001";

when s4 => LEDR <= "001000"; LEDG <= "000011";

when s5 => LEDR <= "001001"; LEDG <= "000000";

when others => LEDR <= "000000";

END CASE;

END PROCESS;

END stateMachine;

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.all;

ENTITY trafficlightwithcrosswalk\_lights\_top IS

PORT(mclk: IN STD\_LOGIC;

btn: IN STD\_LOGIC\_VECTOR(3 DOWNTO 2);

ld : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 2));

END trafficlightwithcrosswalk\_lights\_top;

ARCHITECTURE trafficlightwithcrosswalk\_lights\_top OF trafficlightwithcrosswalk\_lights\_top IS

COMPONENT clkdiv IS

PORT(mclk : IN STD\_LOGIC;

clr : IN STD\_LOGIC;

clk3: OUT STD\_LOGIC);

END COMPONENT;

COMPONENT trafficlightwithcrosswalk IS

PORT(CLOCK\_50: IN STD\_LOGIC;

clr: IN STD\_LOGIC;

LEDR: OUT STD\_LOGIC\_VECTOR(5 DOWNTO 0));

END COMPONENT;

SIGNAL clr, clk3: STD\_LOGIC;

BEGIN

clr <= btn(3);

U1: clkdiv

PORT MAP (mclk => mclk,

clr => clr,

clk3 => clk3);

U2: trafficlightwithcrosswalk

PORT MAP (CLOCK\_50 => clk3,

clr => clr,

LEDR => ld);

END trafficlightwithcrosswalk\_lights\_top;

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.all;

ENTITY traffic\_light\_system IS

PORT(mclk: IN STD\_LOGIC;

btn: IN STD\_LOGIC\_VECTOR(3 DOWNTO 2);

ld : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 2));

END traffic\_light\_system;

ARCHITECTURE traffic\_light\_system OF traffic\_light\_system IS

COMPONENT clkdiv IS

PORT(mclk : IN STD\_LOGIC;

clr : IN STD\_LOGIC;

clk3: OUT STD\_LOGIC);

END COMPONENT;

COMPONENT trafficlightwithcrosswalk IS

PORT(CLOCK\_50: IN STD\_LOGIC;

clr: IN STD\_LOGIC;

LEDR: OUT STD\_LOGIC\_VECTOR(5 DOWNTO 0));

END COMPONENT;

SIGNAL clr, clk3: STD\_LOGIC;

BEGIN

clr <= btn(3);

U1: clkdiv

PORT MAP (mclk => mclk,

clr => clr,

clk3 => clk3);

U2: trafficlightwithcrosswalk

PORT MAP (CLOCK\_50 => clk3,

clr => clr,

LEDR => ld);

END traffic\_light\_system;

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.all;

USE IEEE.STD\_LOGIC\_unsigned.all;

ENTITY trafficlightwithcrosswalk IS

PORT (CLOCK\_50: IN STD\_LOGIC;

clr: IN STD\_LOGIC;

LEDR: OUT STD\_LOGIC\_VECTOR (5 DOWNTO 0);

LEDG: OUT STD\_LOGIC\_VECTOR (5 DOWNTO 0));

END trafficlightwithcrosswalk;

ARCHITECTURE stateMachine OF trafficlightwithcrosswalk IS

TYPE state\_type IS (s0, s1, s2, s3, s4, s5);

SIGNAL state: state\_type;

SIGNAL count: STD\_LOGIC\_VECTOR (3 DOWNTO 0);

CONSTANT SEC5: STD\_LOGIC\_VECTOR (3 DOWNTO 0) := "1111";

CONSTANT SEC1: STD\_LOGIC\_VECTOR (3 DOWNTO 0) := "0011";

BEGIN

PROCESS (CLOCK\_50, clr)

BEGIN

IF clr = '1' THEN state <= s0;

count <= X"0";

ELSIF (CLOCK\_50'EVENT AND CLOCK\_50 = '1') THEN

CASE state IS

WHEN s0 =>

IF count < SEC5 THEN

state <= s0;

count <= count +1;

ELSE

state <=s1;

count <= X"0";

END IF;

WHEN s1 =>

IF count < SEC1 THEN

state <= s1;

count <= count + 1;

ELSE

state <= s2;

count <= X"0";

END IF;

WHEN s2 =>

IF count < SEC1 THEN

state <= s2;

count <= count + 1;

ELSE

state <= s3;

count <= X"0";

END IF;

WHEN s3 =>

IF count < SEC5 THEN

state <= s3;

count <= count + 1;

ELSE

state <= s4;

count <= X"0";

END IF;

WHEN s4 =>

IF count < SEC1 THEN

state <= s4;

count <= count + 1;

ELSE

state <= s5;

count <= X"0";

END IF;

WHEN s5 =>

IF count < SEC1 THEN

state <= s5;

count <= count + 1;

ELSE

state <= s0;

count <= X"0";

END IF;

WHEN OTHERS => STATE <= s0;

END CASE;

END IF;

END PROCESS;

C2: PROCESS(state)

BEGIN

CASE state is

when s0 => LEDR <= "000001"; LEDG <= "001000";

when s1 => LEDR <= "000001"; LEDG <= "110000";

when s2 => LEDR <= "001001"; LEDG <= "000000";

when s3 => LEDR <= "001000"; LEDG <= "000001";

when s4 => LEDR <= "001000"; LEDG <= "000011";

when s5 => LEDR <= "001001"; LEDG <= "000000";

when others => LEDR <= "000000";

END CASE;

END PROCESS;

END stateMachine;

--clock divider

library IEEE;

use IEEE.STD\_LOGIC\_1164.all;

use IEEE.STD\_LOGIC\_unsigned.all;

entity clkdiv is

port(

mclk : in STD\_LOGIC;

clr : in STD\_LOGIC;

clk190 : out STD\_LOGIC;

clk3 : out STD\_LOGIC

);

end clkdiv;

architecture clkdiv of clkdiv is

signal q:STD\_LOGIC\_VECTOR(24 downto 0);

begin

-- clock divider

process(mclk, clr)

begin

if clr = '1' then

q <= X"000000" & '0';

elsif mclk'event and mclk='1' then

q <= q + 1;

end if;

end process;

clk3 <= q(24);

clk190 <= q(18);

end clkdiv;