**servo**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity servo is

Port ( clk : in STD\_LOGIC;

s : in STD\_LOGIC;

Qpwm : out STD\_LOGIC);

end servo;

architecture Behavioral of servo is

signal COUNT : integer range 0 to 2048 ;

signal sq : std\_logic;

signal spwm : STD\_LOGIC\_VECTOR(1 DOWNTO 0);

component DIVIDER is

port (CLK : in std\_logic;

Q : out std\_logic);

end component;

begin

process (sq)

begin

if sq'event and sq = '1' then

if (COUNT >= 2000) then

COUNT <= 0;

else

COUNT <= COUNT +1;

end if;

end if;

end process;

process (COUNT)

begin

if (COUNT <= 0) then

spwm(0) <= '1';

else

spwm(0) <= '0';

end if;

if (COUNT <= 130) then

spwm(1) <= '1';

else

spwm(1) <= '0';

end if;

end process;

Qpwm <= spwm(0) when s = '0' else

spwm(1) ;

c1: DIVIDER port map(CLK, sq);

end Behavioral;

**DIVIDER**

library IEEE;

use IEEE.std\_logic\_1164.all;

entity DIVIDER is

generic (fin: integer := 20000000;

fout: integer := 100000);

port (CLK: in std\_logic;

Q : out std\_logic );

end DIVIDER;

architecture RTL of DIVIDER is

signal COUNT : integer range 0 to (fin/(2\*fout)) ;

signal qs : std\_logic := '0';

begin

process (CLK)

begin

if CLK'event and CLK = '1' then

if (COUNT >= (fin/(2\*fout)-1)) then

COUNT <= 0;

qs <= not(qs);

else

COUNT <= COUNT +1;

end if;

end if;

end process;

Q <= qs ;

end RTL;