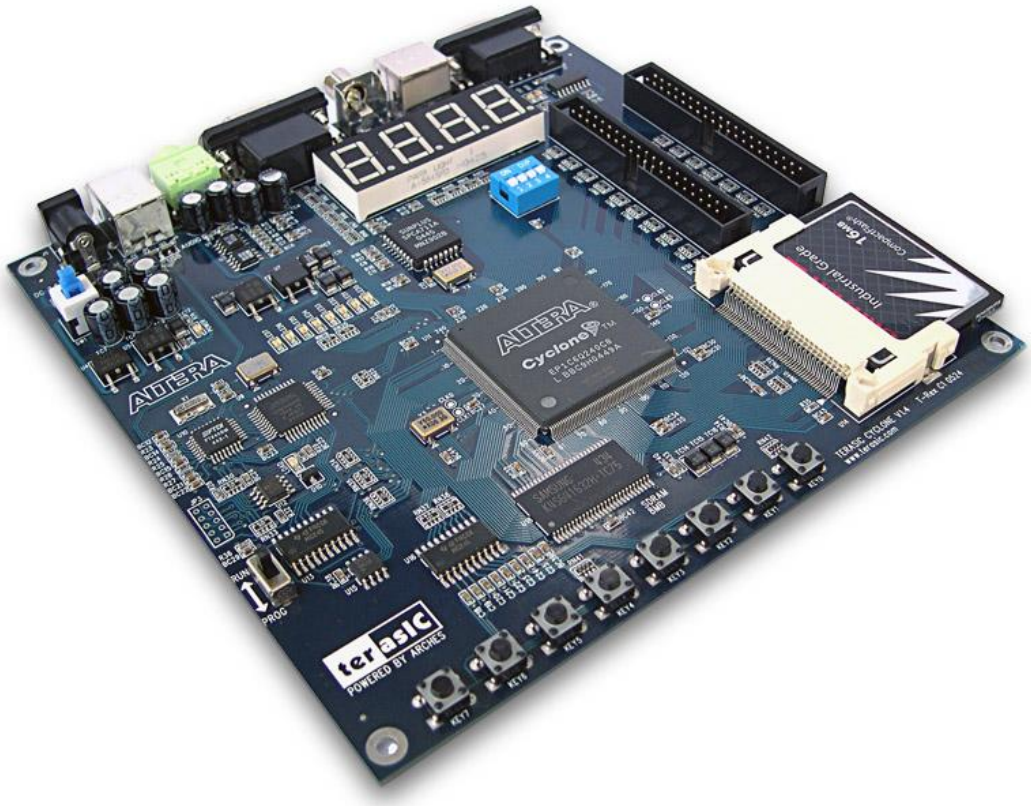
WOJSKOWA AKADEMIA TECHNICZNA
Wydział Elektroniki

Laboratorium układów programowalnych
Projekt w języku VHDL do Implementacji w układzie Altera Cyclone (EP1C6Q248C8) FPGA na płytce uruchomieniowej *Trex C1*
Temat: Prosty syntezator dźwięku (pianinko)

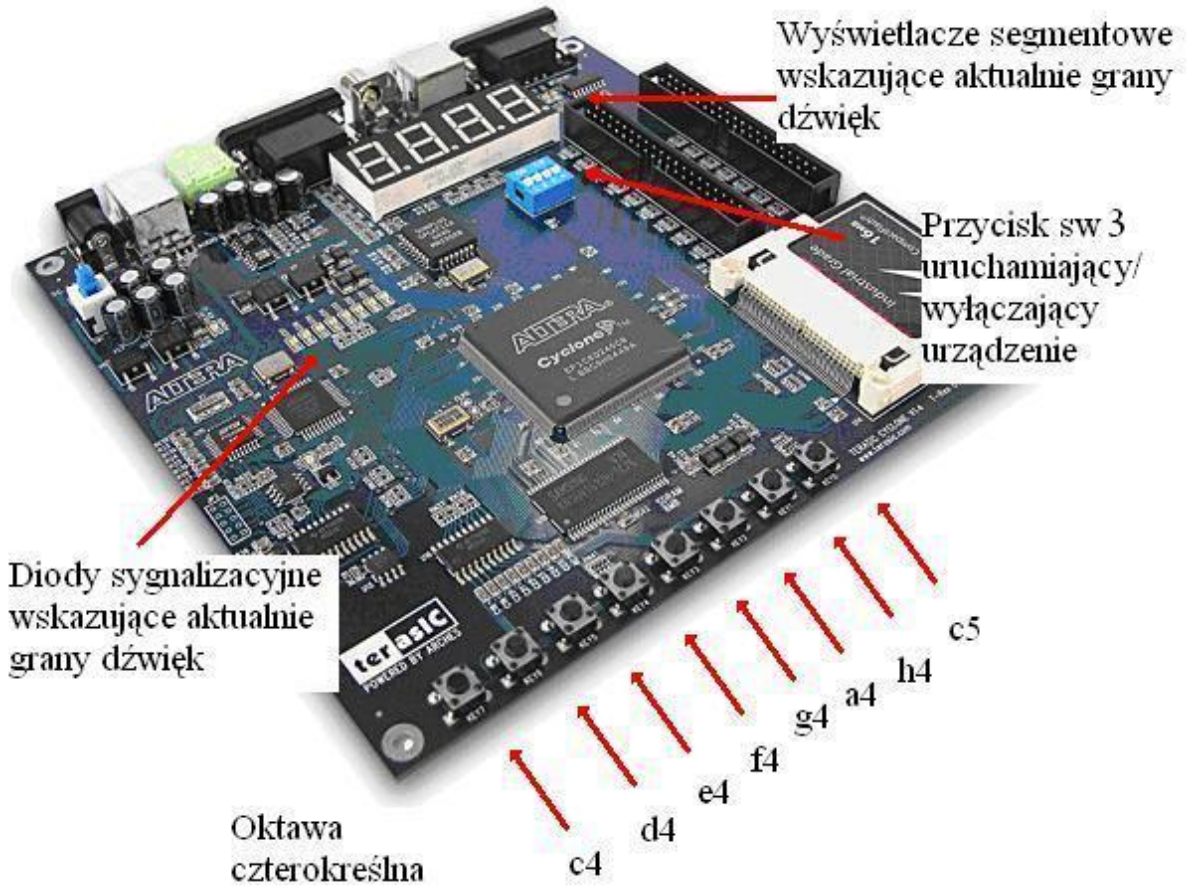
Grupa: **E4T2S0**	Data wykonania ćwiczenia: **11.12.2006**
Wykonał: **Mateusz Karolak**	Ocena:
	Prowadzący: dr inż. **Zbigniew JACHNA**



1. Założenia projektowe

Projekt realizowany jest na płytce TREX C1 zawierającej układ programowalny firmy Altera serii Cyclone EP1C6Q240C8.

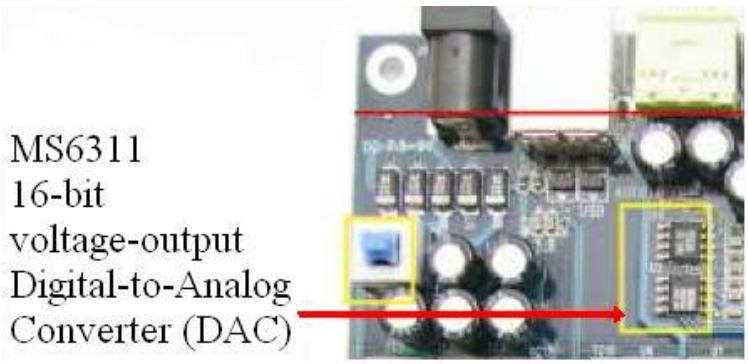
Projektowany układ ma spełniać funkcje prostego pianinka z możliwością generacji dźwięków z jednej oktawy. Poszczególne dźwięki wyzwalane są po wciśnięciu jednego z ośmiu przycisków. Dodatkowo aktualnie grany dźwięk jest sygnalizowany literą dźwięku na wyświetlaczu siedmiosegmentowym oraz zapaleniem się odpowiedniej diody sygnalizacyjnej. Całe urządzenie uruchamiane jest po przełączeniu przełącznika sw3. Bezpośrednio po uruchomieniu jako wprowadzenie załącza się kilkusekundowa sekwencja efektów świetlnych.

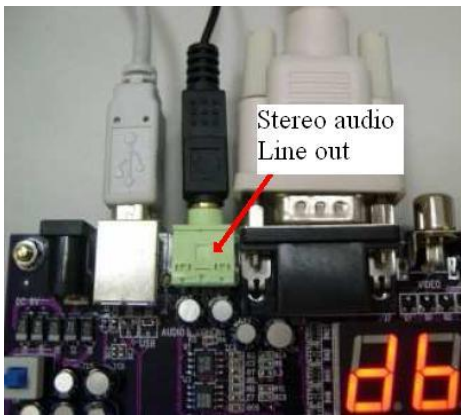


Zastosowana została oktawa czterokreślna o rozpiętości częstotliwości od 2093Hz do 4186Hz:
c4 ->2093 Hz
d4 ->2349 Hz
e4 ->2637 Hz
f4 ->2794 Hz
g4 ->3136 Hz
a4 ->3520 Hz
h4 ->3951 Hz
c5 ->4186 Hz

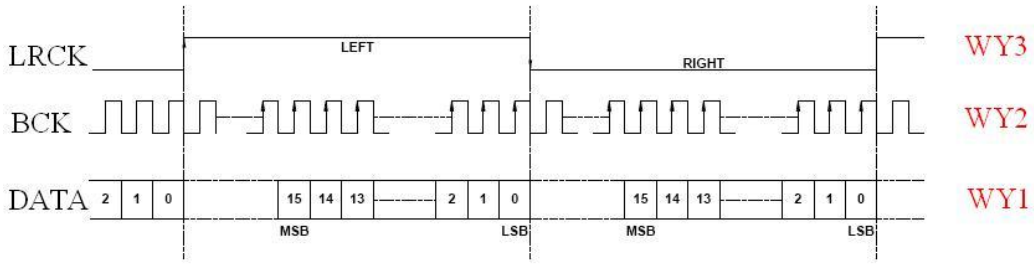
2. Realizacja projektu

Płytka TREX C1 zawiera szesnastobitowy przetwornik cyfrowo-analogowy(DAC) i wyjście Line Out.





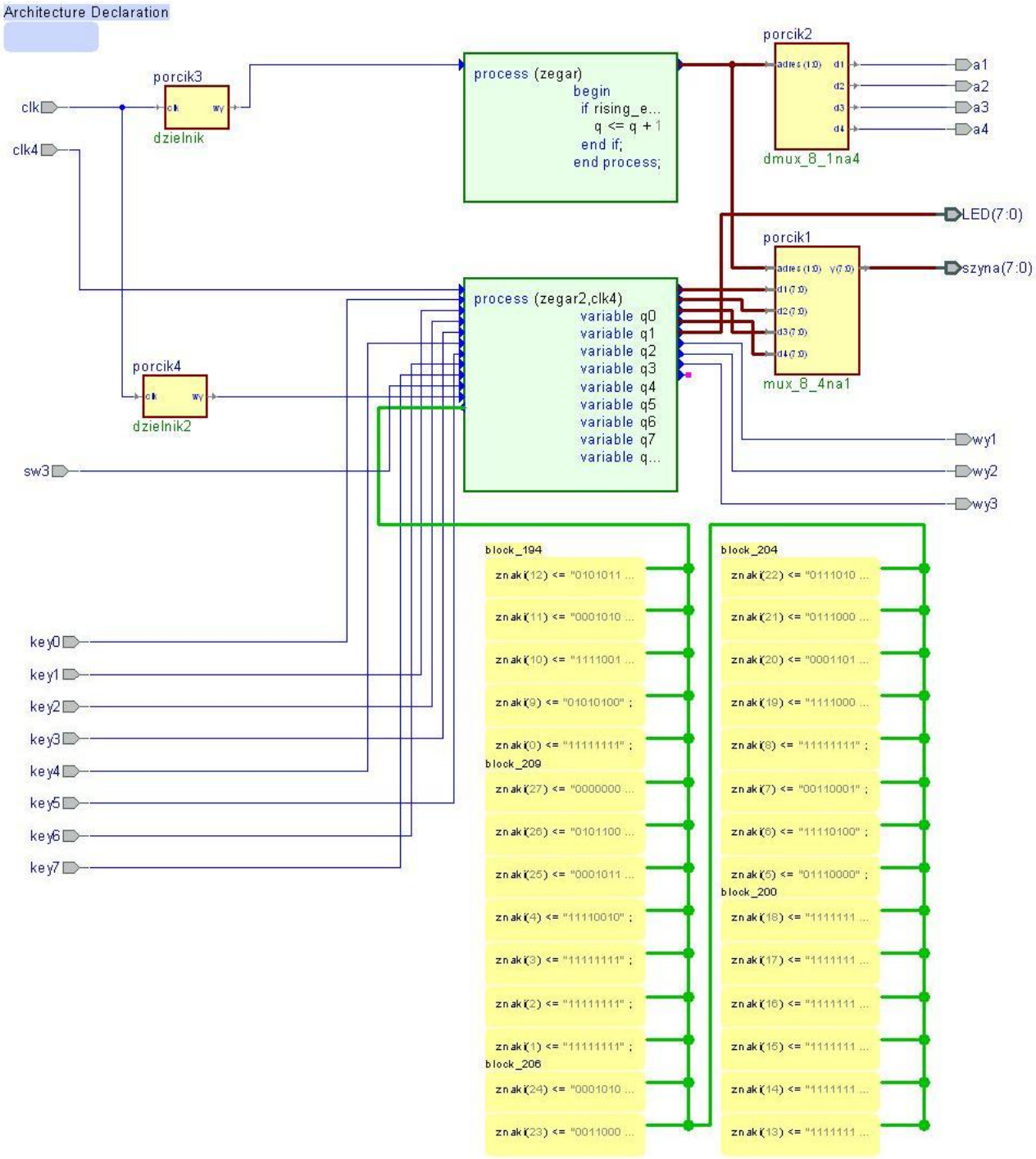
Sygnały częstotliwości dźwięków podawane są na wejście DATA przetwornika(WY1 w kodzie). Wejście przełączania kanałów LRCK przetwornika sterowane jest częstotliwością 44,1 kHz (WY3), natomiast wejście sygnału próbkującego BCK 32 razy większą częstotliwością 1411,2 kHz (WY2). Zegary zastosowane w programie to clk(27MHz) i clk4(50MHz).



Format of input signals.

3. Wnioski i spostrzeżenia

Program można usprawnić na wiele sposobów. Wprowadzając przełączanie oktaw na 3-ech wolnych przełącznikach układu bądź te same przełączniki wykorzystać do zapisu i odczytu zagranych melodii do pamięci. Zapis można prowadzić do pamięci SDRAM na płytce bądź do rejestrów wewnętrznych. Funkcje te jednocześnie najlepiej zastosować podłączając specjalnie skonstruowaną klawiaturę. Dodatkowo wtedy można by wprowadzić cyfrowy potencjometr do regulacji amplitudy.



Created:	2007-01-22
Title:	Mini Syntezator(Pianinko)
Author:	Mateusz Karolak
Source:	proj1234.vhd

KOD

```
-- MULTIPLEKSER DO OBSŁUGI SEGMENTÓW

library ieee;
use ieee.std_logic_1164.all;

entity mux_8_4na1 is
	port(adres: in std_logic_vector(1 downto 0);
		d1,d2,d3,d4: in std_logic_vector(7 downto 0);
		y: out std_logic_vector(7 downto 0));
end mux_8_4na1;

architecture a1 of mux_8_4na1 is
begin
	with adres select
-------------------------------
-------------------------------
	"XXXXXXXX" when others;
end a1;

-- DEMULTIPLEKSER DO OBSŁUGI CYFR

library ieee;
use ieee.std_logic_1164.all;

entity dmux_8_1na4 is
	port(adres: in std_logic_vector(1 downto 0);
		d1,d2,d3,d4: out std_logic);
end dmux_8_1na4;

architecture a2 of dmux_8_1na4 is
begin
-------------------------------
-------------------------------
end a2;

-- DZIELNIK 27 MHZ na 430 Hz


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


entity dzielnik is
	port(clk: in std_logic;
		wy:out std_logic);
end dzielnik;

architecture a4 of dzielnik is



begin

	process(clk)
	-------------------------------
```

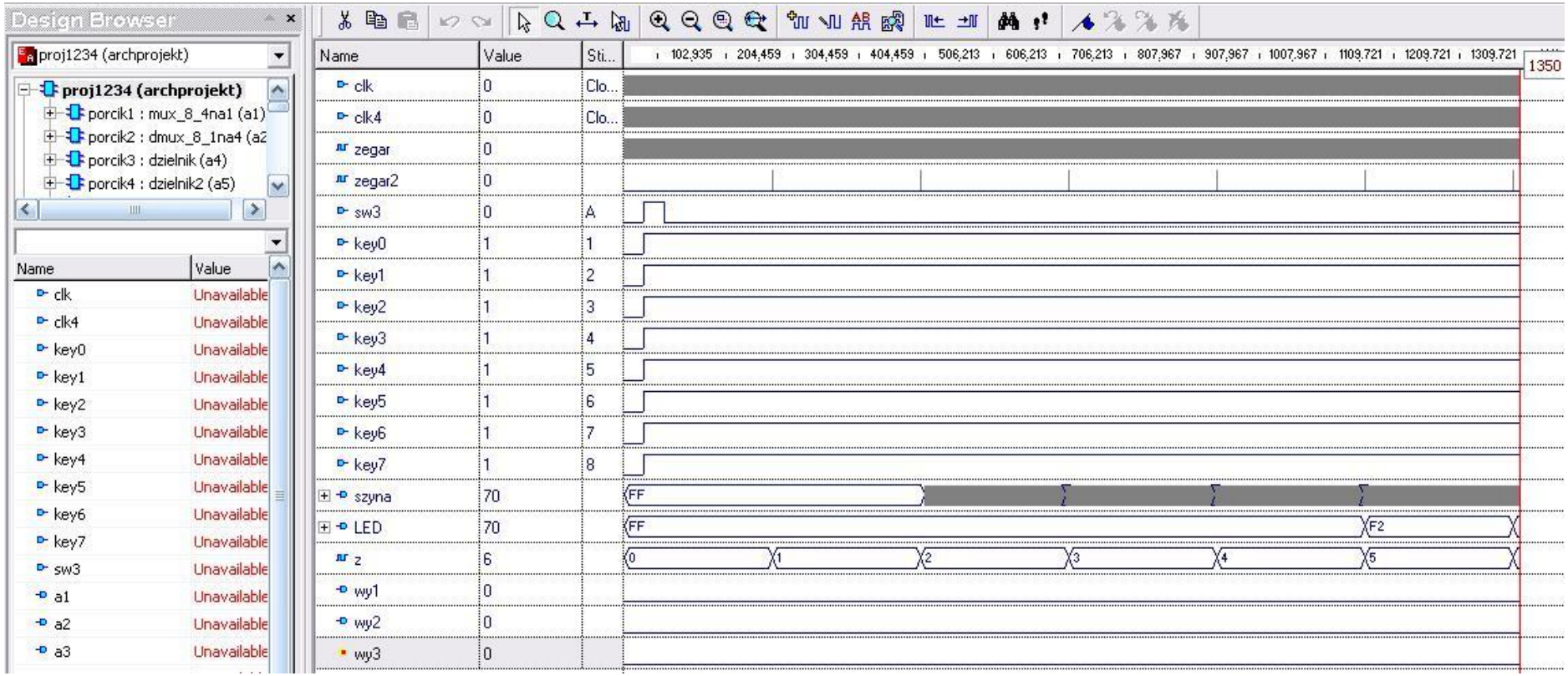
```
--------------------------------

        end process;

end a4;

-- DZIELNIK 2

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

entity dzielnik2 is
        port(clk: in std_logic;
                wy:out std_logic:='0');
end dzielnik2;

architecture a5 of dzielnik2 is

begin

        process(clk)
--------------------------------
--------------------------------
        end process;

end a5;




-- GŁÓWNY PROGRAM


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

entity proj1234 is
        port(
                clk,clk4,key0,key1,key2,key3,key4,key5,key6,key7,sw3: in std_logic;
                a1,a2,a3,a4: out std_logic;
                wy1,wy3,wy2: out std_logic :='0';
                szyna: out std_logic_vector(7 downto 0);
                LED:   out std_logic_vector(7 downto 0):="11111111");
end proj1234;

architecture archprojekt of proj1234 is

-- DEKLARACJA KOMPONENTU MULTIPLEKSERA

component mux_8_4na1
--------------------------------
--------------------------------
end component;

-- DEKLARACJA KOMPONENTU DEMULTIPLEKSERA
```

```
component dmux_8_1na4
------------------------------
------------------------------
end component;

-- DEKLARACJA KOMPONENTU DZIELNIKA

component dzielnik
------------------------------
------------------------------
end component;

-- DEKLARACJA KOMPONENTU DZIELNIKA2

component dzielnik2
------------------------------
------------------------------
end component;


-- DEKLARACJA POMOCNICZYCH ZMIENNYCH/SYGNAŁÓW

signal zegar,zegar2: std_logic;

signal q: std_logic_vector(1 downto 0);

signal b1,b2,b3,b4: std_logic_vector(7 downto 0):="11111111";

type slowo is array(27 downto 0) of std_logic_vector(7 downto 0);

signal znaki: slowo;
signal z: integer :=0;

begin
znaki(0)<="11111111";
znaki(1)<="11111111";
znaki(2)<="11111111";
znaki(3)<="11111111";
znaki(4)<="11110010";      --L
znaki(5)<="01110000";      --E
znaki(6)<="11110100";      --T
znaki(7)<="00110001";      --S
znaki(8)<="11111111";
znaki(9)<="01010100";      --P
znaki(10)<="11110010";     --L
znaki(11)<="00010100";     --A
znaki(12)<="01010110";     --Y
znaki(13)<="11111111";
znaki(14)<="11111111";
znaki(15)<="11111111";
znaki(16)<="11111111";
znaki(17)<="11111111";
znaki(18)<="11111111";
znaki(19)<="11110000";    --c
znaki(20)<="00011010";    --d
znaki(21)<="01110000";    --e
znaki(22)<="01110100";    --f
znaki(23)<="00110000";    --g
znaki(24)<="00010100";    --a
znaki(25)<="00010110";    --h
```

```
znaki(26)<="01011000";    --2
znaki(27)<="00000000";



porcik1: mux_8_4na1 port map(q,b1,b2,b3,b4,szyna);
porcik2: dmux_8_1na4 port map(q,a1,a2,a3,a4);
porcik3: dzielnik port map(clk,zegar);
porcik4: dzielnik2 port map(clk,zegar2);

process(zegar)
begin
--------------------------------
--------------------------------
end process;



process (zegar2, clk4)
--------------------------------
--------------------------------



begin
if sw3='1' then                    --urządzenie wyłączone
z<=0;



else                               --urządzenie włączone

if z<13  then
--------------------------------
--------------------------------              --graficzna sekwencja startowa
--------------------------------
--------------------------------              --błyskajace diody
--------------------------------
--------------------------------
--------------------------------
--------------------------------              --przelatujący napis na wyświetlaczu
--------------------------------
--------------------------------

        end if;
else



        if rising_edge(clk4) then              --oktawa czterokreślna
--------------------------------
--------------------------------
                if key7='0' then                                --dźwięk c 2093  Hz
                        b1<=znaki(19);
                        LED(7)<='1';
                        if  c7='0' then
                                if q7="10111010101000" then
                                        q7:="00000000000000";
                                        wy1<='1';
```

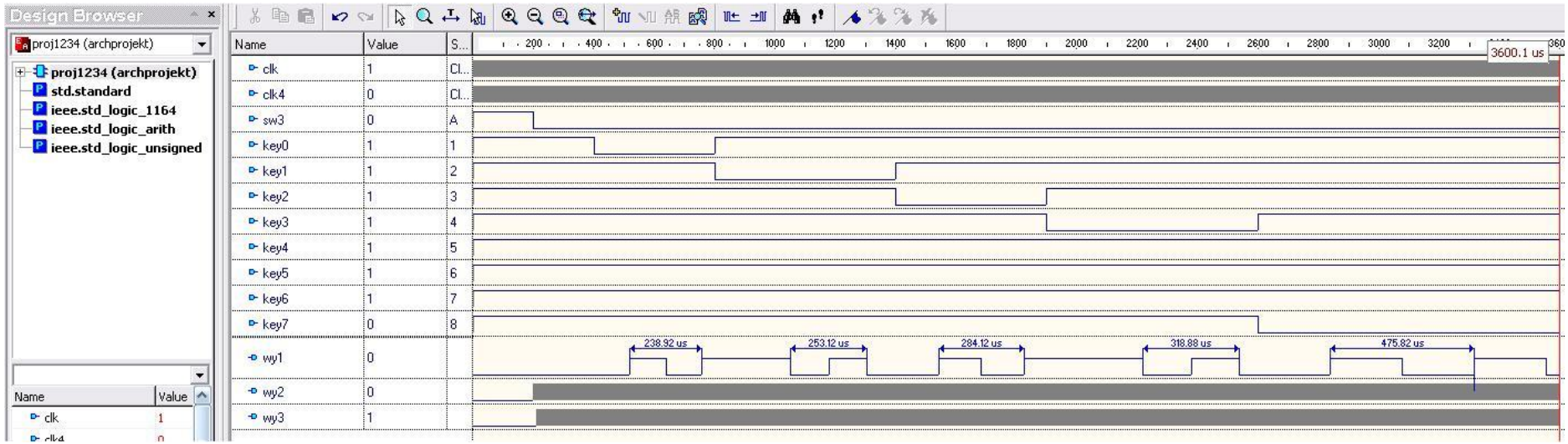
```
                                        c7:='1';
                                else
                                        q7:=q7+1;
                                end if;
                        else
                                if q7="10111010101000" then
                                        q7:="00000000000000";
                                        wy1<='0';
                                        c7:='0';
                                else
                                        q7:=q7+1;
                                end if;
                        end if;

                elsif key6='0' then                                     --dźwięk d  2349 Hz
-------------------------------
-------------------------------
                elsif key5='0' then                                       -- dźwięk e  2637 Hz
-------------------------------
-------------------------------
                elsif key4='0' then                                --       dźwięk f    2794 Hz
-------------------------------
-------------------------------
                elsif key3='0' then                                     --  dźwięk g    3136 Hz
-------------------------------
-------------------------------
                elsif key2='0' then                                     --    dźwięk a  3520 Hz
-------------------------------
-------------------------------
                elsif key1='0' then                                     --    dźwięk h  3951 Hz
-------------------------------
-------------------------------
                elsif key0='0' then                                       --  dźwięk c2    4186   Hz
                        b1<=znaki(19);
                        b2<=znaki(26);
                        LED(0)<='1';
                        if  c0='0' then
                                if q0="1011101010100" then
                                        q0:="0000000000000";
                                        wy1<='1';
                                        c0:='1';
                                else

                                        q0:=q0+1;
                                end if;
                        else
                                if q0="1011101010100" then
                                        q0:="0000000000000";
                                        wy1<='0';
                                        c0:='0';
                                else
                                        q0:=q0+1;
                                end if;
                        end if;




                end if;
```

```
                                                    -- 1411,2 kHz          BCK     wy2
if  cbck='0' then
-------------------------------
-------------------------------
end if;

if  clrck='0' then                                   -- 44,1 kHz           LRCK     wy3
        if qlrck="1000110110" then
                qlrck:="0000000000";
                wy3<='1';
                clrck:='1';
        else
                qlrck:=qlrck+1;
        end if;
else
        if qlrck="1000110110" then
                qlrck:="0000000000";
                wy3<='0';
                clrck:='0';
        else
                qlrck:=qlrck+1;
        end if;
        end if;
end if;
end if;

end if;

end process;


end archprojekt;
```



Symulacja sekwencji startowej urządzenia.

Ze względu na sterowanie dużymi zegarami clk4(50Mhz) i clk(27Mhz) symulacja nie może trwać długo(tu 1350ms) gdyż potrzebna jest coraz większa przestrzeń dyskowa. Symulacja wskazuje uruchomienie sekwencji przelatującego napisu na wyświetlaczu 7-seg. I błyskających diod. Zmiany stanu wyświetlacza i diod sterowane są „zegarem2”(~2,5Hz) wywoływaniem kolejnych elementów tablicy. Na wyjściu szyna widać jednak dużo dynamiczniejsze zmiany. Wynika to z faktu, ze wyświetlacz składa się z 4-kostek i jest multipleksowany z częstotliwością „zegara”(430Hz). Dopiero po zliczeniu przez „z” 13 zmian sekwencja jest przerywana i można zacząć sterować przetwornikiem DAC.



Symulacja dźwięków.

key0=’0’ -> wy1= 238,92us (c5->4185,5Hz)
key1=’0’ -> wy1= 253,12us (h4->3950,7Hz)
key2=’0’ -> wy1= 284,12us (a4->3519,6Hz)
key3=’0’ -> wy1= 318,88us (g4->3135,9Hz)
key7=’0’ -> wy1= 475,82us (c4->2101,6Hz)