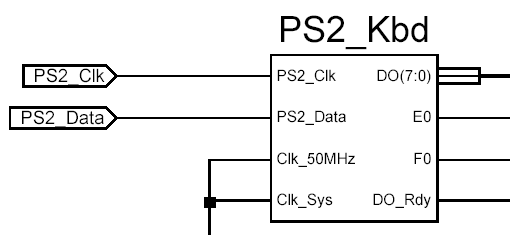
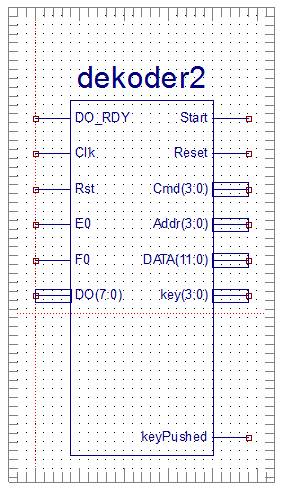
Dekoder + klawiatura

**Klawiatura** – moduł zaimportowany ze strony ***Zakładu Systemów Komputerowych[[1]](#footnote-1)*** . Jest to moduł realizujący operację wejścia w projektowanym systemie. Obsługuje on klawiaturę podłączoną przez port *PS2*. Jego obecność jest na tyle ważna, iż dekoduje on kod naciśniętego klawisza i przesyła tą sekwencję do modułu autorskiego ***dekoder***. Moduł ***PS2\_Kbd*** pokazuje *Rysunek 1.*



Rysunek 1. Odbiornik kodów wysyłanych przez klawiaturę PS/2

**Dekoder** – odrębny moduł napisany przez autorów niniejszego dokumentu. Moduł ten korzysta z zewnętrznych bibliotek:

* library IEEE;
* use IEEE.STD\_LOGIC\_1164.ALL;
* use IEEE.STD\_LOGIC\_unsigned.ALL;
* use ieee.numeric\_std.all;
* use IEEE.Std\_Logic\_arith.all;

Moduł ten posiada 6 sygnałów wejściowych (DO\_RDY, Clk, Rst, E0, F0, DO) oraz 7 sygnałów wyjściowych (Start, Cmd, Addr, DATA, Reset, key, keyPushed).

1. type state\_type is (s12,s0,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11);
2. signal state, next\_state: state\_type;
3. signal tmp:STD\_LOGIC\_VECTOR (7 downto 0) := "00000000";
4. signal i: STD\_LOGIC\_VECTOR (8 downto 0) := "000000000";
5. signal n: STD\_LOGIC\_VECTOR (8 downto 0) := "000000000";
6. signal licznik: STD\_LOGIC\_VECTOR (26 downto 0 ) := "000000000000000000000000000";
7. signal czy\_koniec : STD\_LOGIC :='0';
8. signal czy\_wyswietlic : STD\_LOGIC :='0';

Opis architektury modułu VHDL został po krótce przedstawiony powyżej. W linii oznaczonej ***(1)*** oznaczone zostały stany w których będzie znajdywał się układ, na podstawie maszyny stanów automatu Moore’a. ***(3)*** jest zmienna odpowiedzialną za 8 bitowy licznik, wykorzystywany przy przesyłaniu sygnału ***DATA*** na wejście przetwornika **DAC.** Linia ***(4)*** reprezentuje deklarację 8 bitowego iteratora wykorzystywanego przy generacji fali piłokształtnej. 9 bitowa wartość częstotliwości dźwięku odpowiedniego dla danej oktawy deklarowana jest w linii nr ***(5)***. Licznik zliczający takty zegara ***(6)*** potrzebne do odpowiedniej modulacji dźwięku. W liniach ***(7)*** i ***(8)*** są ustawiane sygnały odpowiednio dotyczące zakończenia odgrywania sygnału, oraz wysyłania danych na moduł VGA. Sygnały te używane są przy wyciszaniu dźwięku, które ma za zadanie imitować realistyczne wybrzmiewanie dźwięku odgrywanego na faktycznych organach.

**process1: process (Clk, E0, F0)**

**begin**

**if(rising\_edge (Clk)) then**

**if (DO\_RDY = '1') and (E0 ='0' and F0='0') then**

**state <=next\_state;**

**end if;**

**end if;**

**end process process1;**

***Process1*** przedstawia tutaj, przechodzenie do następnego stanu maszyny w zależności od narastającego zbocza zegara oraz wartości specjalnych ***F0*** i ***E0*** dostarczonych przez moduł PS2\_Kbd.

**process2: process (state,DO)**

**begin**

**next\_state <= state;**

**case state is**

**(1) when s12 => if DO = "00100010" then --x**

**(2) next\_state <= s0;**

**(3) n <= "101110101"; --373**

**(4) key <= "0000"; -- informacja dla wizualizacji VGA**

**elsif DO = "00100001" then --c**

**next\_state <= s1;**

**n <= "101001100"; --333**

**key <= "0001"; -- informacja dla wizualizacji VGA**

**elsif DO = "00101010" then --v**

**next\_state <= s2;**

**n <= "100101000"; --296**

**key <= "0010"; -- informacja dla wizualizacji VGA**

**elsif DO = "00110010" then --b**

**next\_state <= s3;**

**n <= "100010111"; --280**

**key <= "0011"; -- informacja dla wizualizacji VGA**

**elsif DO = "00110001" then --n**

**next\_state <= s4;**

**n <= "011111001"; --249**

**key <= "0100"; -- informacja dla wizualizacji VGA**

**elsif DO = "00111010" then --m**

**next\_state <= s5;**

**n <= "011011101"; --222**

**key <= "0101"; -- informacja dla wizualizacji VGA**

**elsif DO = "01000001" then --,**

**next\_state <= s6;**

**n <= "011000101"; --198**

**key <= "0110"; -- informacja dla wizualizacji VGA**

**elsif DO = "00100011" then --d**

**next\_state <= s7;**

**n <= "101100000"; --352**

**key <= "0111"; -- informacja dla wizualizacji VGA**

**elsif DO = "00101011" then --f**

**next\_state <= s8;**

**n <= "100111001"; --314**

**key <= "1000"; -- informacja dla wizualizacji VGA**

**elsif DO = "00110011" then --h**

**next\_state <= s9;**

**n <= "100000111"; --264**

**key <= "1001"; -- informacja dla wizualizacji VGA**

**elsif DO = "00111011" then --j**

**next\_state <= s10;**

**n <= "011101011"; --235**

**key <= "1010"; -- informacja dla wizualizacji VGA**

**elsif DO = "01000010" then --k**

**next\_state <= s11;**

**n <= "011010001"; --209**

**key <= "1011"; -- informacja dla wizualizacji VGA**

**else**

**next\_state <= s12;**

**n <= "000000000";**

**key <= "1111"; -- informacja dla wizualizacji VGA**

**end if;**

**(5) when s0 => if DO = "11110000" then --puszczono klawisz**

**(6) next\_state <= s12;**

**(7) n <= "000000000";**

**(8) key <= "1111"; -- informacja dla wizualizacji VGA**

**(9) elsif DO = "00100010" then --x**

**next\_state <= s0;**

**n <= "101110101"; --373**

**key <= "0000"; -- informacja dla wizualizacji VGA**

**else**

**next\_state <= s12;**

**n <= "000000000";**

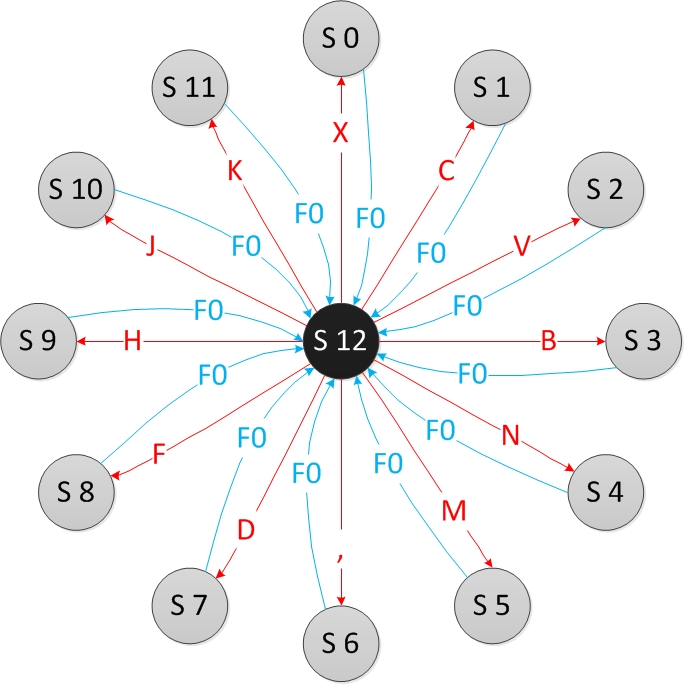
**key <= "1111"; -- informacja dla wizualizacji VGA**

**end if;**

**end case;**

**end process process2;**

W procesie następnym na podstawie stanu w którym się znajdujemy, oraz odpowiedniej wartości sygnału DO dostarczonym przez moduł klawiatury postępujemy zgodnie z zamieszczoną poniżej maszyną stanów.



Rysunek 2. Maszyna stanów

W linii oznaczonej przez znacznik **(1)** jesteśmy w jałowym stanie oznaczonym u nas jako ***s12***. W stanie tym czekamy na naciśnięcie jednego z określonych klawiszy. Jeśli został naciśnięty w tym przypadku klawisz „x” o kodzie binarnym: **00100010,** wtedy ustawiamy odpowiednio maszynę do stanu ***s0*** **(2),** liczbę ***n*** określającą częstotliwość dźwięku **C** odgrywanego z oktawy dwukreślnej.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | klawisz | hex | bin | f | f in program  - dec | f in program  - bin |
| 1 | c2 | **x** | 22 | 00100010 | 523,25 | 373 | 101110101 |
| 2 | cis2 | **c** | 21 | 00100001 | 554,4 | 352 | 101100000 |
| 3 | d2 | **v** | 2a | 00101010 | 587,3 | 333 | 101001100 |
| 4 | dis2 | **b** | 32 | 00110010 | 622,3 | 314 | 100111001 |
| 5 | e2 | **n** | 31 | 00110001 | 659,3 | 296 | 100101000 |
| 6 | f2 | **m** | 3a | 00111010 | 698,5 | 280 | 100010111 |
| 7 | fis2 | **,** | 41 | 01000001 | 740 | 264 | 100000111 |
| 8 | g2 | **d** | 23 | 00100011 | 784 | 249 | 011111001 |
| 9 | gis2 | **f** | 2b | 00101011 | 830,6 | 235 | 011101011 |
| 10 | a2 | **h** | 33 | 00110011 | 880 | 222 | 011011101 |
| 11 | b2 | **j** | 3b | 00111011 | 932,3 | 209 | 011010001 |
| 12 | h2 | **k** | 42 | 01000010 | 987,8 | 198 | 011000101 |

**Tabela 1 Dźwięki oktawy dwukreślnej z przeliczonymi wartościami zmiennych**

Powyższa tabela przedstawia wszystkie wartości które pojawiają się w module ***dekoder2***. Obliczenia te zostały wykonane na podstawie prostych wzorów fizycznych, poniższe obliczenia przedstawiają proces powstawania odpowiednich wartości dla dźwięku ***c2.***

W ostatniej linii obliczeń wartość ***n*** została podzielona przez 256, ponieważ wysyłamy 28 sampli tego dźwięku. Format wyjściowy **DATA** jest 12 bitowy, dlatego dopełniać będziemy ten sygnał czterema zerami (*0000*).

**Process 3 –** modułodpowiedzialny za wysyłanie fali piłokształtnej na przetwornik ***DAC*** a w momencie puszczenia klawisza uruchamiana jest procedura ściszania dźwięku. Przez pierwsze 0.3 sekundy odgrywamy ten sam sygnał, następnie w okresie czasu 0.3 – 0.5 dzięki zastosowaniu konkatenacji czterech zer, z sygnałem ***tmp*** przechowującym wartość aktualnie wysyłaną na przetwornik cyfrowo-analogowy, uzyskujemy obniżenie dźwięku. Następnym przedziałem czasowym jest 0.5 – 0.7, 0.7 – 0.9, 0.9 – 1.1, a jeśli odliczymy już okres powyżej 1.1 sekundy, odgrywanie dźwięku zostaje zatrzymane.

**DATA <=tmp&"0000"; -- 0.0 – 0.3**

**DATA <="0"&tmp&"000"; -- 0.3 – 0.5**

**DATA <="00"&tmp&"00"; -- 0.5 – 0.7**

**DATA <="000"&tmp&"0"; -- 0.7 – 0.9**

**DATA <="0000"&tmp; -- 0.9 – 1.1**

**DATA <="000000000000"; -- > 1.1**

Architektura **dekoder2** kończy się 3 poleceniami, które definiują odpowiednio kanał **D** przetwornika **DAC** *(1)***,** standardowe wejście tegoż przetwornika *(2)*, oraz informację dla modułu ***VGA*** dotyczącą zmiany wyświetlania na monitorze ***CRT*** *(3).*

1. **Addr <= "0011"; -- wybieramy kanal D**
2. **Cmd <= "0011"; -- standardowe wejście**
3. **keyPushed <= czy\_wyswietlic;**

1. http://www.zsk.ict.pwr.wroc.pl/zsk\_ftp/fpga [↑](#footnote-ref-1)