

**Max frequency signal generation**

Embedded Systems

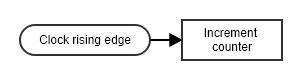
Intermediate Exam 1

Done by: EKSfmu-16 gr. st. Arūnas Butkus

Checked by: doc. dr. Andrius Ušinskas

**Work goal:** using VHDL in DE0 development board generate maximum possible frequency periodical signal.

Algorithms used in tasks.



**Fig. 1.** “Algorithm” for signal generation.

Results analysis.

All in all it is simple operation: on each clock rising edge a counter is increased (Fig. 1) and that counter value is directly tied to output – when counter is an even number output is high, when an odd number output is low. Thus generating a periodic square-ish wave signal. In theory maximum frequency of a generated square wave is 25 MHz because FPGA clock is 50 MHz so due to there being two distinct states in the signal that needs to be switched between we get 25 MHz maximum. That is what the oscilloscope shows as well (Fig. 2).

**Fig. 3.** Example audio signal.

Conclusions.

Simple VHDL script that switches the output voltage level on a single pin. Though the signal generated has significant rising and dropping edge noises and the end signal doesn’t really seem like a square wave signal, which it should be. Though the frequency is correct.

Source code.

01 library ieee;

02 use ieee.std\_logic\_1164.all;

03

04 entity IESigGen is

05 Port (

06 CLOCK\_50 : IN STD\_LOGIC;

07 SW : IN STD\_LOGIC\_VECTOR(9 DOWNTO 0);

08 KEY : IN STD\_LOGIC\_VECTOR(2 DOWNTO 0);

09 HEX0 : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0);

10 HEX1 : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0);

11 HEX2 : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0);

12 HEX3 : OUT STD\_LOGIC\_VECTOR(7 DOWNTO 0);

13 LEDG : OUT STD\_LOGIC\_VECTOR(9 downto 0);

14 GPIO\_0 : OUT STD\_LOGIC\_VECTOR(2 DOWNTO 0)

15 );

16 end IESigGen;

17

18 architecture Whatever of IESigGen is

19

20 signal timer\_count : INTEGER := 0;

21 signal counter : INTEGER := 0;

22

23 begin

24

25 WITH timer\_count MOD 2 SELECT GPIO\_0 <=

26 "001" WHEN 0,

27 "000" WHEN OTHERS;

28

29 process(CLOCK\_50)

30 begin

31 if rising\_edge(CLOCK\_50) then

32 timer\_count <= timer\_count + 1;

33 if (timer\_count = TIMER\_MAX) then

34 counter <= counter + 1;

35 timer\_count <= 0;

36 end if;

37 end if;

38 end process;

39

40 end Whatever;

Electrical schemes:

