Experiment 3 - Function Generator

Pasha Barahimi, 810199385
Misagh Mohaghegh, 810199484

***Abstract*— This document is a student report for the 3rd experiment of the Digital Logic Laboratory course (ECE 045) at University of Tehran, Department of Electrical and Computer Engineering.**

***Keywords*— Function Generator, DAC, PWM, RTL Design, RTL Simulation, FPGA Implementation, Altera Cyclone II, Quartus, ModelSim**

I. Digital to Analog conversion using PWM

A. Hardware Description

```
`include "Counter.v"
`include "Register.v"

module PWM(
    input [7:0] in,
    input clk, rst,
    output reg out
);
    wire [7:0] inOut, cntOut;
    wire cntCo;
    reg ldIn;

    Counter cnt(.load(1'b0), .en(1'b1), .clr(1'b0), .clk(clk), .rst(rst), .out(cntOut), .co(cntCo));
    Register inReg(.loadData(in), .load(ldIn), .clr(1'b0), .clk(clk), .rst(rst), .out(inOut));

    always @(inOut, cntOut, cntCo) begin
        ldIn = 1'b0;

        if (cntCo) ldIn = 1'b1;
        if (cntOut <= inOut) out = 1'b1;
        else out = 1'b0;
    end

endmodule
```

Fig. 1 DAC (Digital to Analog) Verilog description

B. Simulation

A 6ns clk is used for the simulation. As visible in Fig. 2, the waveform output (PWM's input) is 218. So the PWM's output should be on for 218 clock cycles, and should be off for the next 256 – 218 = 38 clock cycles.

Since the clock duration is 6ns, $T_{on} = 218 \times 6 = 1308$ns and $T_{off} = 38 \times 6 = 228$ns which are shown by the waveform cursors as expected.

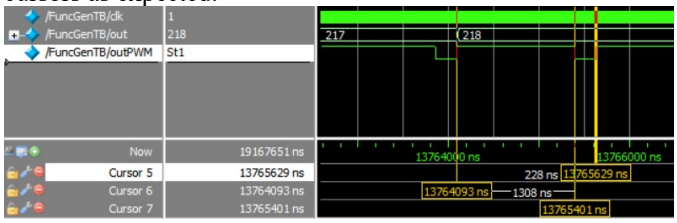

Fig. 2 PWM (Pulse Width Modulation) simulation waveform

II. Waveform Generator

A. Different Waveforms

The waveform generator is the main part of the function generator. It outputs an 8-bit number representing the amplitude of the analog signal.

For all functions, each full period takes 256 clock cycles and a counter is used for this.

The following functions are implemented:

1) *Square:* For 128 cycles, the output is 255 and for the next 128 cycles the output is 0, thus creating a square.

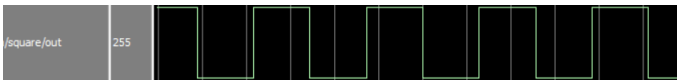

Fig. 3 Square wave

2) *Triangle:* For 128 cycles, the output is the counter's output which creates a line on the (y = x) axis. For the next 128 cycles, 255 - counter is used so the line is mirrored and creates a triangle with 127 as its max point.

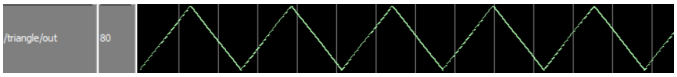

Fig. 4 Triangle wave

3) *Reciprocal:* (255 / 255-x) would create a reciprocal shape but to make the shape more clear, (255 / 63-(x/4)) is used.

The denominator of the fraction is zero for x=252 to x=255. At these points, the number 4 is put on the output to connect to the beginning of the shape on its next period.

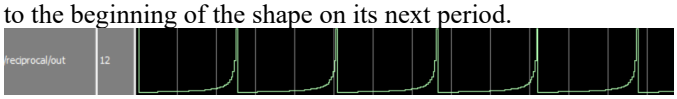

Fig. 5 Reciprocal wave

4) *Rhomboid:* A triangle is shifted up 128 bits which will have a low point of 127 and a max of 255 bits.

Now every other cycle, the triangle is flipped horizontally. To achieve this, the parity (first bit) of the counter output is used and (255 – triangle) will flip around the 127 point.

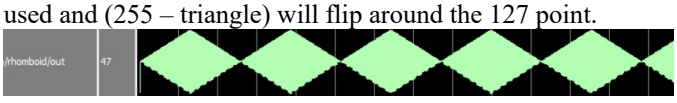

Fig. 6 Rhomboid wave

Fig. 7 Zoomed-in rhomboid wave

5) *Sine:* The sine wave is generated according to the instructions on the laboratory manual.

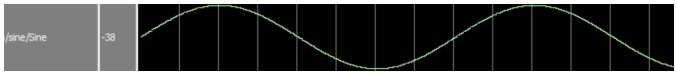

Fig. 8 Sine wave

6) *Full-wave rectified:* To generate full-wave, the sine's output is flipped for points below 127.

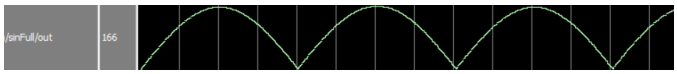

Fig. 9 Full-wave rectified

7) *Half-wave rectified:* To generate half-wave, the sine's output is a constant 127 when the points are below that which is the midpoint.

Fig. 10 Half-wave rectified

8) *Modulated square wave:* The sine wave and a counter is used. Every 16 clocks, the sine is flipped which generates the modulated shape.

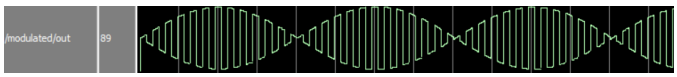

Fig. 11 Modulated square wave

B. *Waveform Generator Processor*

The processor handles all the aforementioned functions.

C. *DDS*

The DDS (Direct Digital Synthesis) uses a ROM to store its output. The period of the signal is controlled by Phase_cntrl's value.

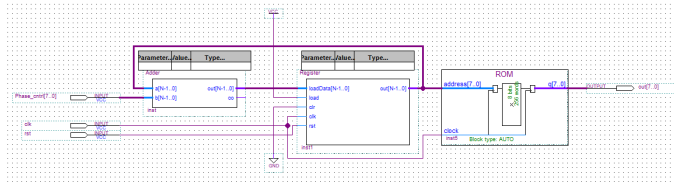

Fig. 12 DDS block diagram

The ROM is initialized using values for a sine wave.

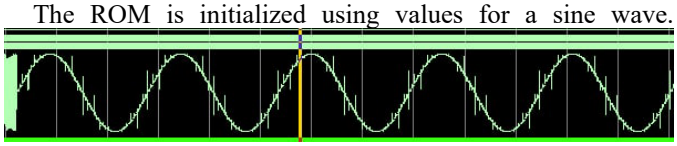

Fig. 13 DDS sine.mif with Phase_cntrl = 1

D. *Block Diagram*

The waveform generator processor is on the top and the DDS is on the bottom. The output is selected from either one using a multiplexer.

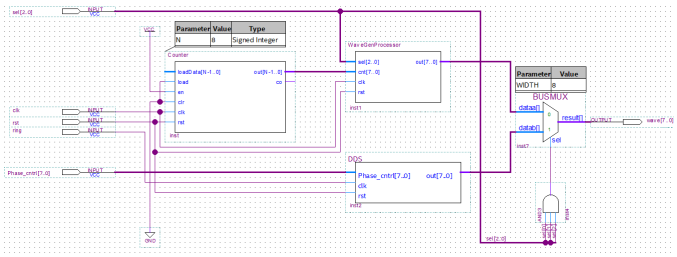

Fig. 14 Waveform generator block diagram

Flow Summary

<<Filter>>

Flow Status	Successful - Sat May 28 11:13:30 2022
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	FuncGen
Top-level Entity Name	WaveGen
Family	Cyclone IV GX
Device	EP4CGX22BF14C6
Timing Models	Final
Total logic elements	213 / 21,280 (1 %)
Total registers	80
Total pins	22 / 81 (27 %)
Total virtual pins	0
Total memory bits	2,048 / 774,144 (< 1 %)

Fig. 15 Synthesis report

III. Frequency Selector

A. *Hardware Description*

```
`include "Counter.v"
`include "jkFlipFlop.v"

module FreqSelector #(
    parameter N = 8
)(
    input [N-1:0] sel,
    input clr, clk, rst,
    output co
);
    wire [N-1:0] comp;
    assign comp = {1'b1, {N{1'b0}}} - sel;

    wire coLsb, coMsb;
    wire [N/2-1:0] outLsb, outMsb;

    Counter #(N/2) lsb(.loadData(comp[N/2-1:0]), .load(coMsb), .en(1'b1), .clr(clr), .clk(clk), .rst(rst), .out(outLsb), .co(coLsb));
    Counter #(N/2) msb(.loadData(comp[N-1:N/2]), .load(coMsb), .en(coLsb), .clr(clr), .clk(clk), .rst(rst), .out(outMsb), .co(coMsb));

    jkFlipFlop duty(.j(1'b1), .k(1'b1), .clk(coMsb), .rst(rst), .q(co));
endmodule
```

Fig. 16 Frequency selector Verilog description

B. *Testing*

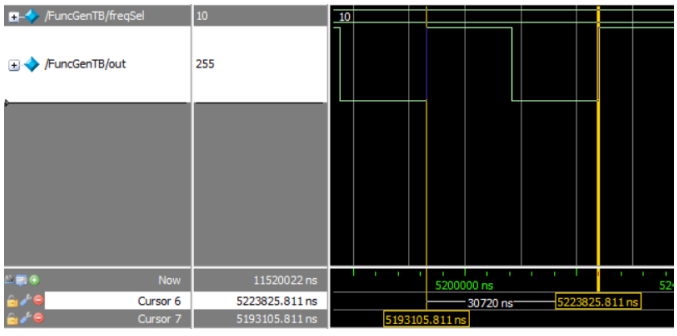

Fig. 17 Freq select is on 10

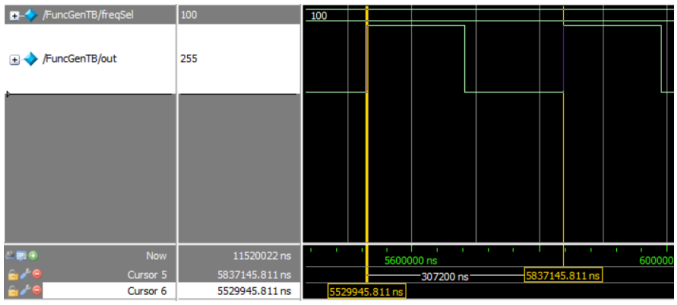

Fig. 18 Freq select is on 100

As we can see, in Fig. 18, the frequency is divided 10 times more than Fig. 17 and the time period is 10 times more.

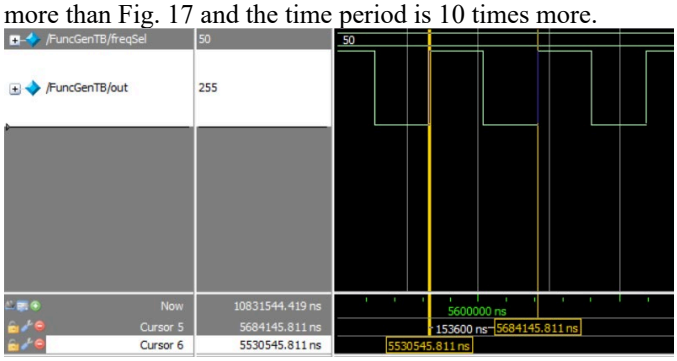

Fig. 19 Freq select is on 50

C. Phase Control

The DDS module generates the ROM's content with different phases using its Phase_cntrl input.

The phase gets added to the ROM pointer at every turn. The value of 1 will show every block of the ROM. Higher values will skip some blocks thus changing the frequency.

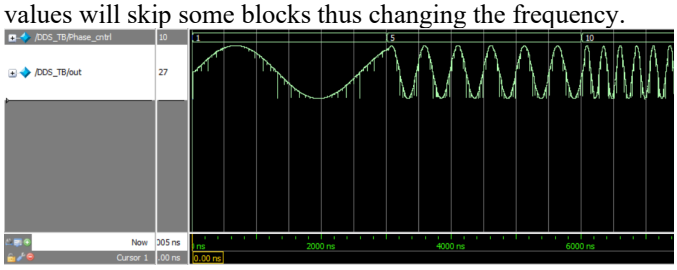

Fig. 20 DDS output with different phase values

IV. Amplitude Selector

Since the analog signal's amplitude is the value of the digital output, we can divide the amplitude by dividing the digital output. A simple shifting is done in order to divide the output by multiples of 2 and have selections between 1, 1/2, 1/4, and 1/8.

```
1  module AmplitudeSelector(
2      input [1:0] select,
3      input [7:0] in,
4      output [7:0] out
5  );
6      assign out = in >> select;
7  endmodule
```

Fig. 21 Amplitude selector Verilog description

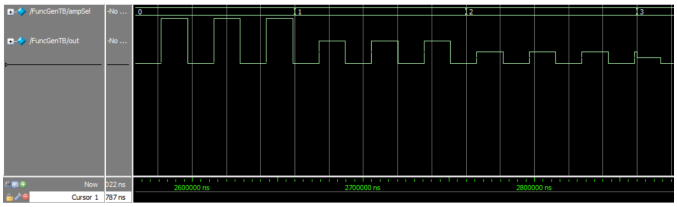

Fig. 22 Testing the amplitude selector with different values, dividing by 1, 1/2, 1/4, and 1/8

V. Total Design

A. Block Diagram

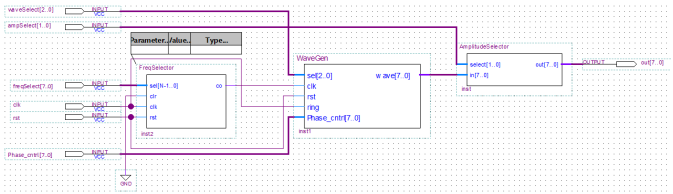

Fig. 23 The function generator block diagram

B. FPGA Implementation

1) *Pin Planner:* The 10 physical Cyclone II toggle buttons were assigned: 1 for reset, 3 for wave selection, 2 for amplitude selection, and 4 for frequency selection.

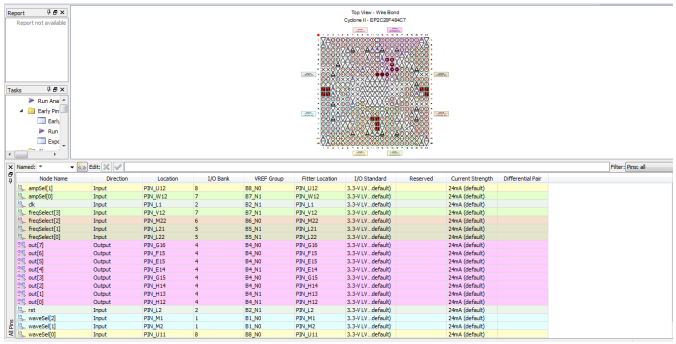

Fig. 24 Quartus pin planner

2) *FPGA:*

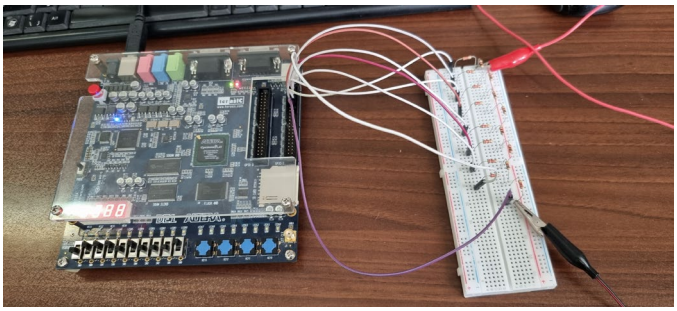

Fig. 25 The FPGA with its 8bit output connected to the amplitude selector circuit and the oscilloscope probes

3) *Results:* The results are shown on an oscilloscope.

All of the waveform generator functions are shown in two forms. The first is the normal output and in the second one, the amplitude selector divides the amplitude by 2.

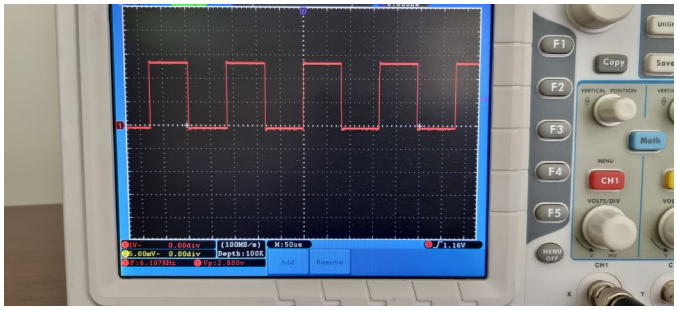

Fig. 26 Square wave (1)

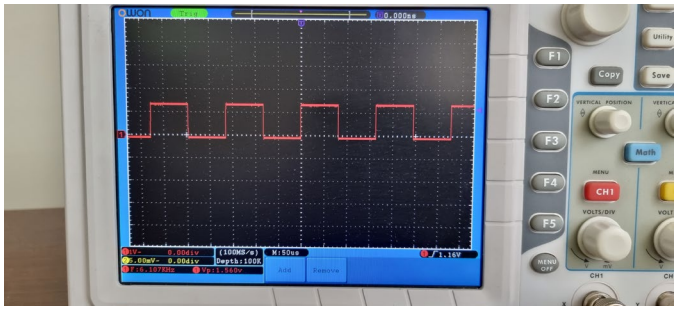

Fig. 27 Square wave (2)

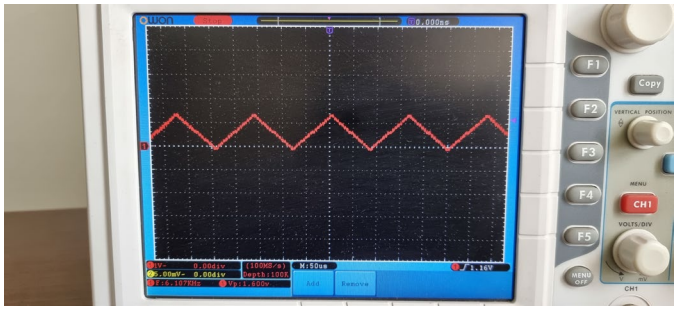

Fig. 28 Triangle wave (1)

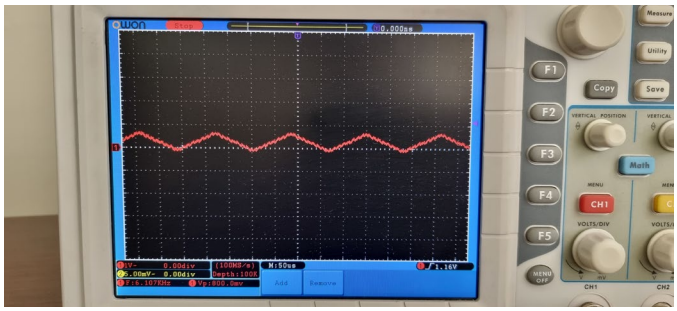

Fig. 29 Triangle wave (2)

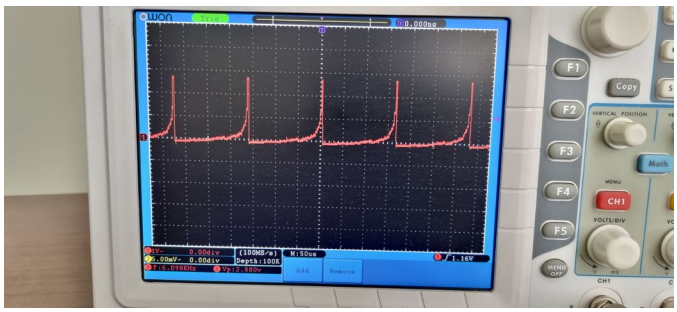

Fig. 30 Reciprocal wave (1)

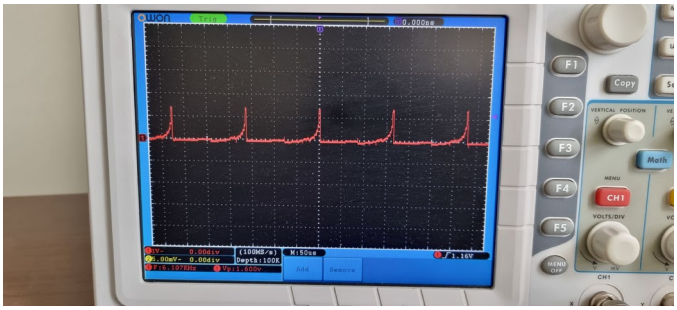

Fig. 31 Reciprocal wave (2)

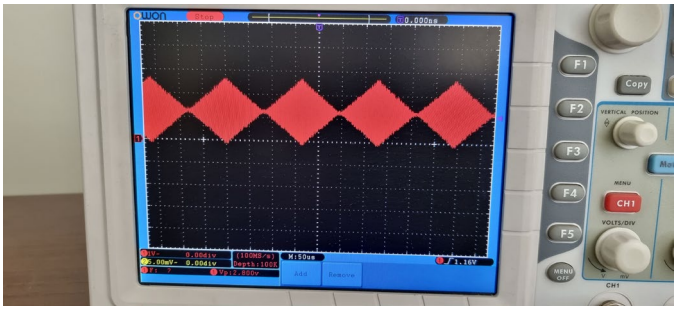

Fig. 32 Rhomboid wave (1)

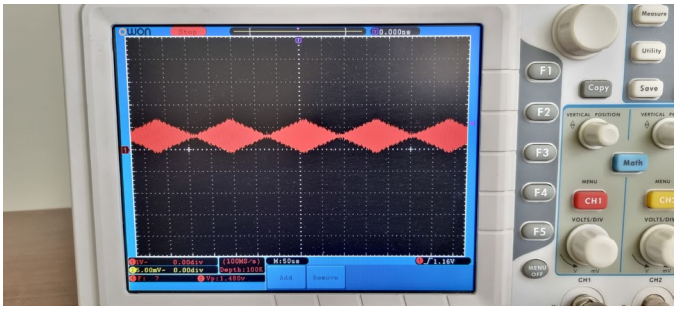

Fig. 33 Rhomboid wave (2)

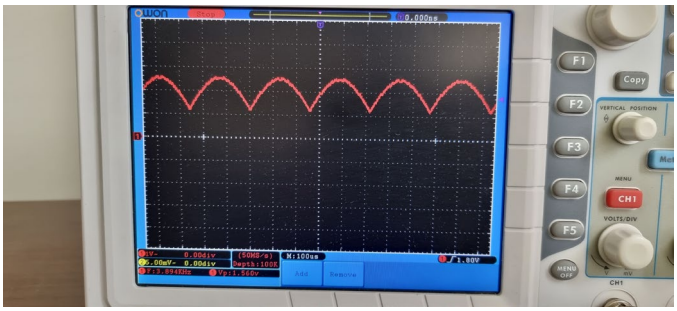

Fig. 34 Full-wave rectified (1)

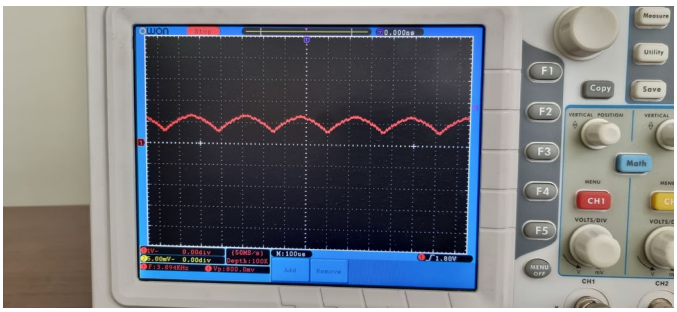

Fig. 35 Full-wave rectified (2)

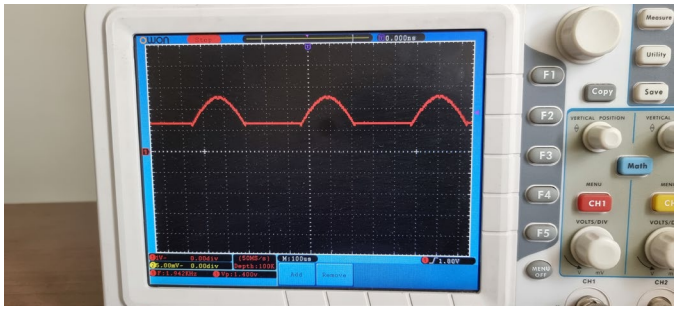

Fig. 36 Half-wave rectified (1)

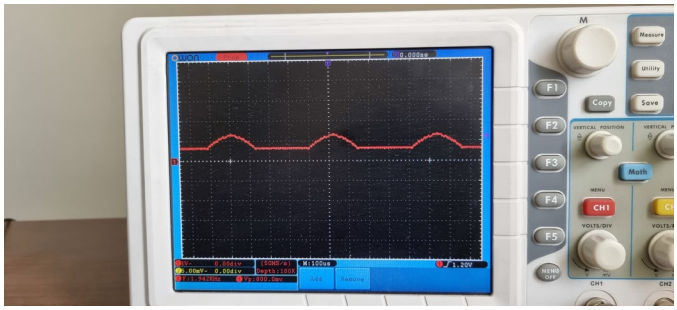

Fig. 37 Half-wave rectified (2)

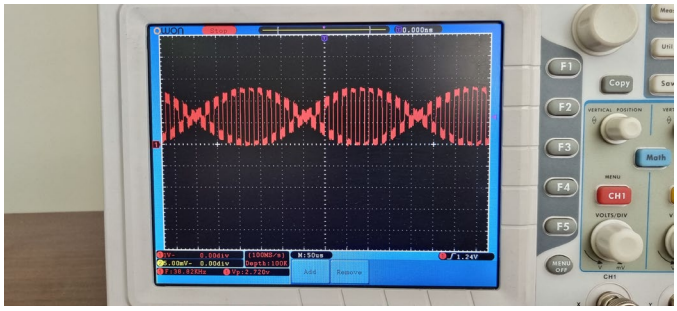

Fig. 38 Modulated square wave (1)

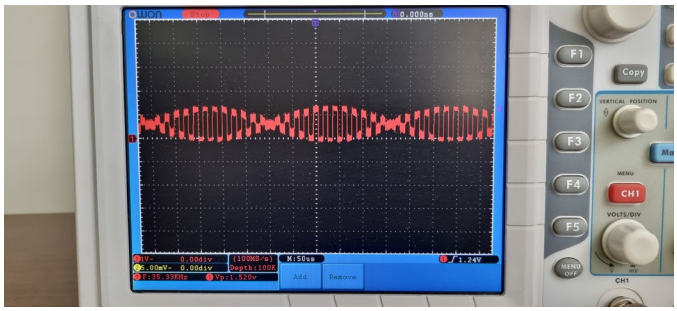

Fig. 39 Modulated square wave (2)

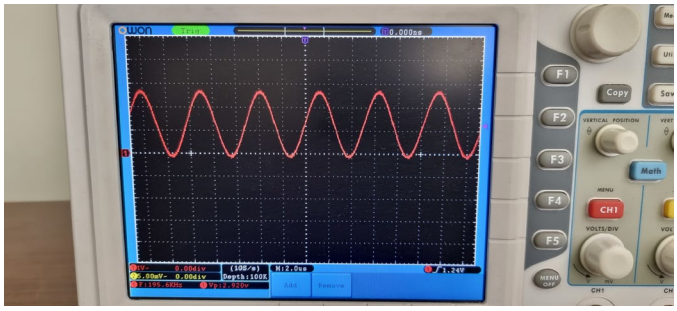

Fig. 40 DDS sine.mif (1)

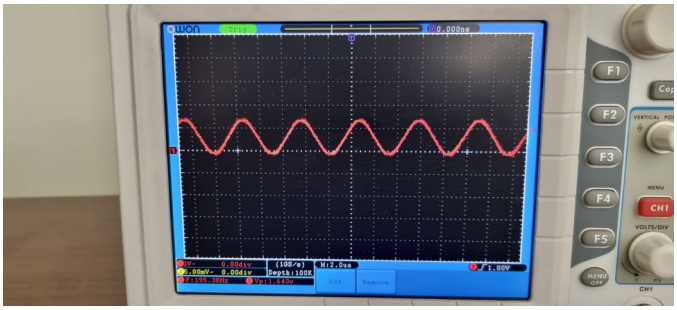

Fig. 41 DDS sine.mif (2)

VI. R-2R DAC Ladder

After implementing the design on FPGA, an R-2R DAC ladder is used in order to prepare an analog signal for the oscilloscope. The mentioned ladder is shown in Fig. 42.

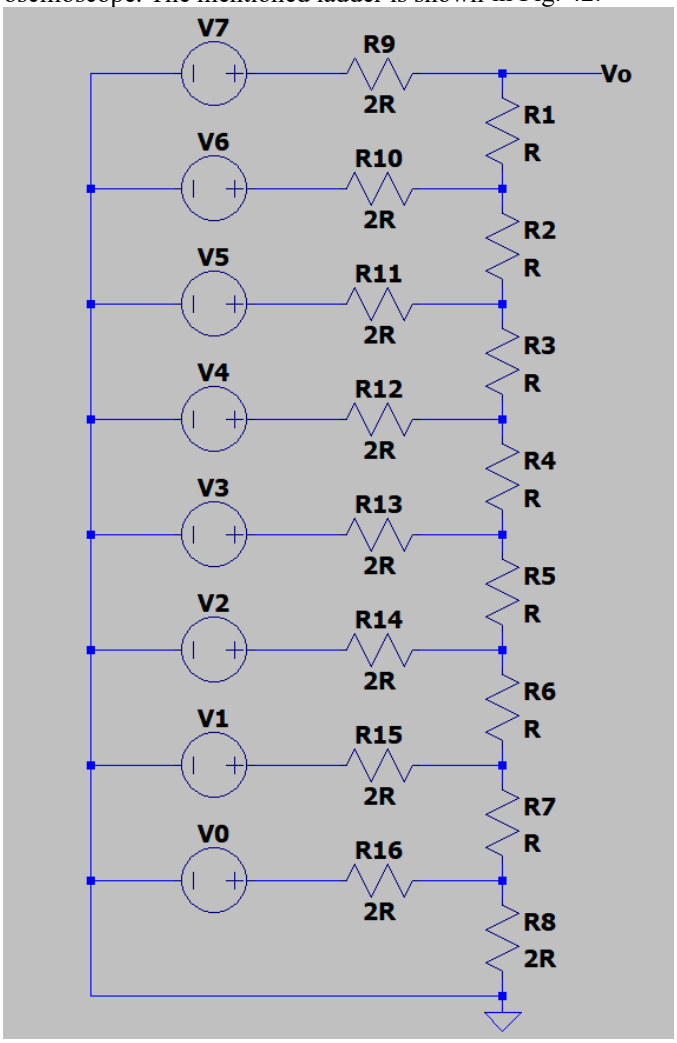

Fig. 42 The R-2R DAC Ladder

Assuming that only D6 is active, the circuit will be as presented in Fig. 43 and the simplified circuit is shown in Fig. 44.

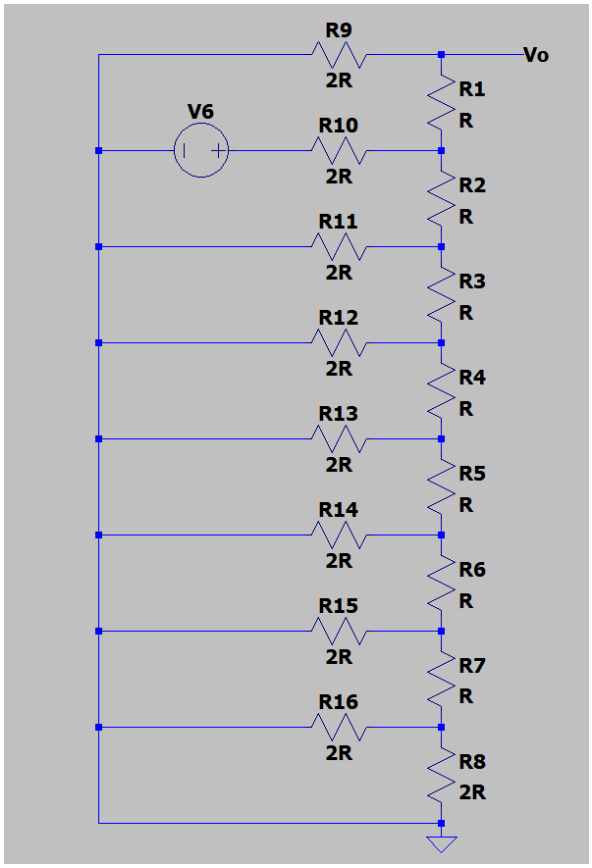

Fig. 43 The ladder with only D6 active

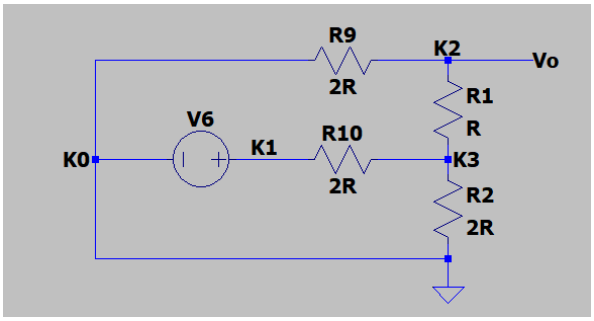

Fig. 44 Simplified circuit

KCL is used to calculate the output voltage (V_o). The calculation process is presented below:

$$KCL:\ V_{K0} = 0, \qquad V_{K1} = V_6$$

$$K2:\ \frac{V_{K2} - 0}{2R} + \frac{V_{K2} - V_{K3}}{R} = 0$$

$$K3:\ \frac{V_{K3} - V_{K2}}{R} + \frac{V_{K3} - 0}{2R} + \frac{V_{K3} - V_6}{2R} = 0$$

$$\rightarrow \begin{cases} 6V_{K2} - 4V_{K3} = 0 \\ 4V_{K3} - 2V_{K2} = V_6 \end{cases}$$

$$\rightarrow \boldsymbol{V_o = V_{K2} = \frac{V_6}{4}}$$

Based on induction and the superposition principle, the output voltage (V_o) is calculated as follows:

$$V_o = \frac{V_7}{2^1} \qquad V_o = \frac{V_3}{2^5}$$

$$V_o = \frac{V_6}{2^2} \qquad V_o = \frac{V_2}{2^6}$$

$$V_o = \frac{V_5}{2^3} \qquad V_o = \frac{V_1}{2^7}$$

$$V_o = \frac{V_4}{2^4} \qquad V_o = \frac{V_0}{2^8}$$

$$V_o = \frac{V_7}{2^1} + \frac{V_6}{2^2} + \frac{V_5}{2^3} + \frac{V_4}{2^4} + \frac{V_3}{2^5} + \frac{V_2}{2^6} + \frac{V_1}{2^7} + \frac{V_0}{2^8}$$