Experiment #3 – Function Generator

Daneshvar Amrollahi

Student ID: 810197685

***Abstract*— This document is a student report to experiment #3 of Digital Logic Laboratory course at ECE Department, University of Tehran. The goal of this experiment is to design an arbitrary function generator which is capable of generating many waveforms such as Rhomboid, Sine, Square, Reciprocal, and etc with wide range for frequency selection.**

***Keywords*—Function Generator, Waveforms, Frequency Selector, Amplitude Selector, Cyclone IV E**

I. Introduction

An overall view of the function generator system is showed in figure 1. It consists of 3 main units:

1. Frequency Selector (#Experiment #2)
2. Waveform Generator (main processing unit)
3. Amplitude Selector

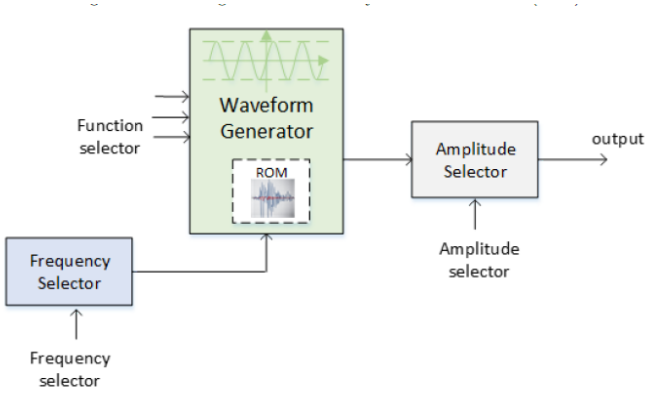

Fig. 1 Overall view of function generator

II. Waveform Generator

This module is the heart of the project. It inputs 3 bits indicating which function (waveform) needs to be generated by this unit. There are 8 choices:

func[2..0]	Waveform
3'b000	Rhomboid
3'b001	Sine
3'b010	Square
3'b011	Reciprocal
3'b100	Saw-Tooth
3'b101	Full-Wave Rectified
3'b110	Modulated Square Wave
3'b111	Arbitrary

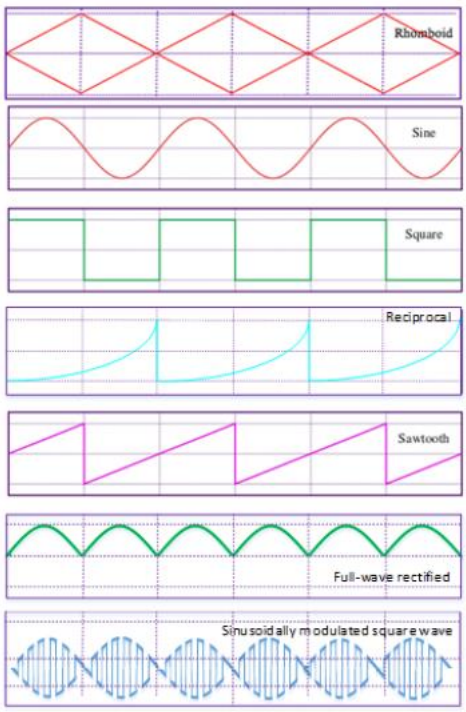

Fig. 2 Supported waveforms

The last case (3'b111) will generate a waveform stored in a ROM. The ROM is initialized using a .mif file with Quartus MegaWizard.

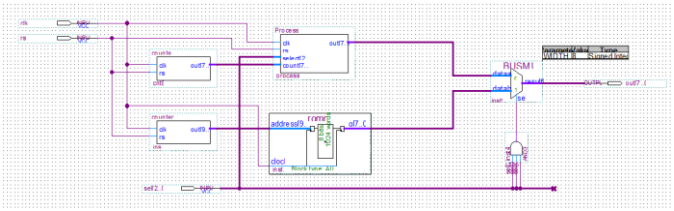

Fig. 3 Block Diagram of WaveForm Generator unit

Processor unit in Fig. 3 outputs the desired function (waveforms) based on input select[2..0]. 256 points are generated for each function. Each wave is represented by a function *f(x)* which *x* is the output of counters. An 8 bit counter is used for Processor unit and a 10 bit counter is used for the ROM since it stores 1024 words (8 bit values). The multiplexer puts the desired wave (generated by processor or ROM) on the output. This output will later be used as an input for Amplitude Selector which we discuss later in this report. Details on generating the functions can be found in processor.v file.

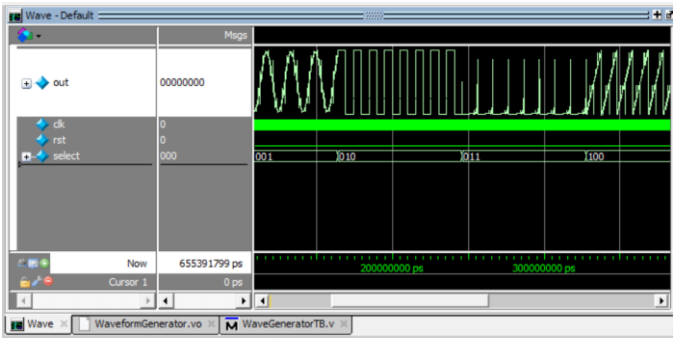

Fig. 4: Synthesize result of WaveForm Generator: Sine, Square, Reciprocal, Saw-tooth are illustrated

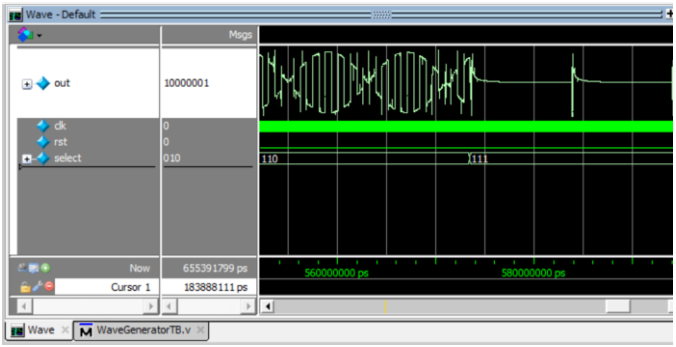

Fig. 5: Synthesize result of WaveForm Generator: Sinusodially Modulated Square Wave and ROM waves are illustrated

III. FREQUENCY SELECTOR

This section divides the FPGA clock (50MHz) and the divided lock is used for the WaveGenerator unit clock. The division is done by building a 256-counter with parallel load inputs. Counting starts from PL[7..0] and ends at 255 = $(11111111)_2$. When the counter reaches 255, the carry out bar bit of a 74193 IC becomes high, therefore there is a toggle on the T Flip Flop (Built using a 7476 JK Flip Flop). The output of the T Flip Flop is the divided clock. The divided clock has a duty cycle of 50%. It toggles whenever the counter reaches 255, therefore $T_{divided\_clock} = T_{FPGA} \times (255 - PL[7..0])$ where T indicates the period of a clock signal. The desired frequency can be set using the suitable parallel load values using the formula above.

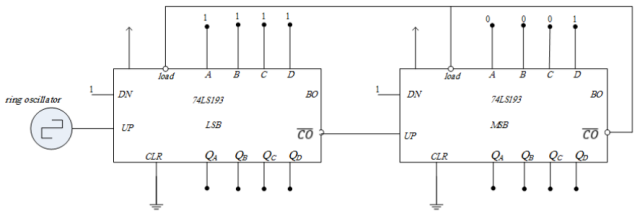

Fig. 6: Cascading 74193s to build a 256 counter

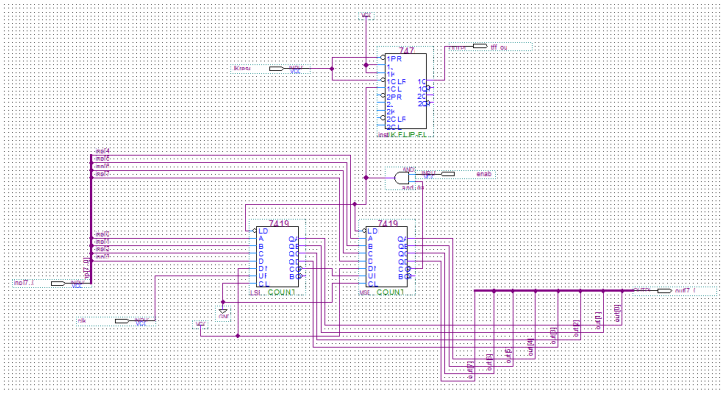

Fig. 7: Block Diagram of Clock Divider

A block diagram of the clock divider (Fig. 7) is built and added to the WaveForm generator explained in section II and synthesis is done.

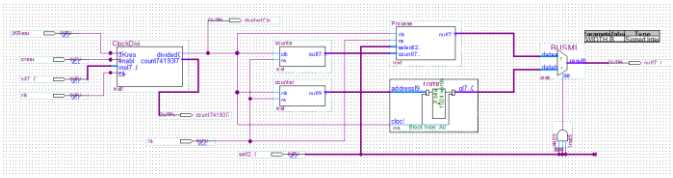

Fig. 8: Block Diagram of Wave Generator with a Frequency Selector (Clock divider)

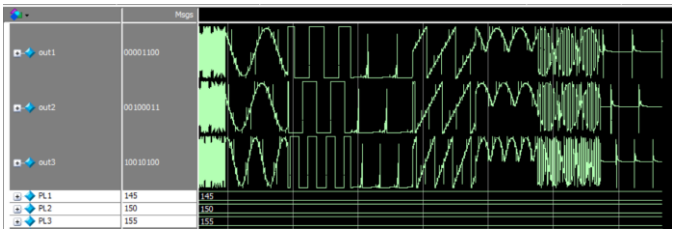

Fig. 9: Simulation result of 3 different units with different parallel loads causing 3 different frequencies.

The unit with least parallel load (145) will have the highest clock period (It takes longer for the counters to reach 255 and for the T Flip Flop to toggle), therefore less number of positive edges in a fixed length of time leading to a lower frequency wave. The unit with highest parallel load (155) will have the lowest clock period, therefore higher number of positive edges in a fixed length of time leading to a higher frequency wave. This is can be validated in Figure 9.

$T_{divided\_clock} = T_{FPGA} \times (255 - PL[7..0])$
$T_{FPGA} = 1 / f_{FPGA} = 1 / 50MHz = 20ns$

$T_1 = 20 \times (255 - 145) = 2200ns \rightarrow f_1 = 0.45MHz$
$T_2 = 20 \times (255 - 150) = 2100ns \rightarrow f_2 = 0.47MHz$
$T_3 = 20 \times (255 - 155) = 2000ns \rightarrow f_3 = 0.50MHz$

IV. AMPLITUDE SELECTOR

This is an option in the Function Generator to scale down the generated waveform. The 8 bit output of the WaveForm Generator unit is passed to this unit and scaling is executed based on a 2 bit input indicating the scaling intensity. Based on the 2 bit input, the output is divided by a number shown in the following table:

sw[12..11]	Amplitude
2'b00	1
2'b01	2
2'b10	4
2'b11	8

Dividing each waveform by 2^i is equivalent to shifting *i* units to the right, therefore sign extension needs to be applied to each signal. Further details about this unit can be found in AmpSelector.v file.

A block symbol of the Amplitude Selector section is added to the previous block diagrams. The final (complete) block diagram is as followed:

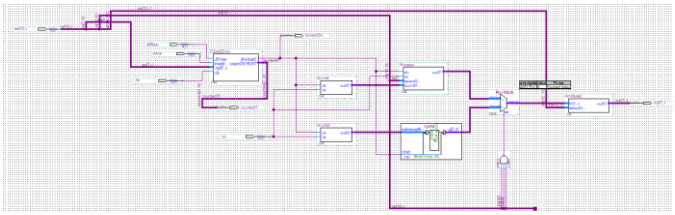

Fig. 10: Complete Block Diagram of Function Generator. For further details check FuncGenWithFreqAndAmp.BDF

The final module has 13 bit inputs (SW[12..0]). 3 bits are used for function selection. 2 bits are used to amplitude selection. The remaining 8 bits are used for parallel load of counters.

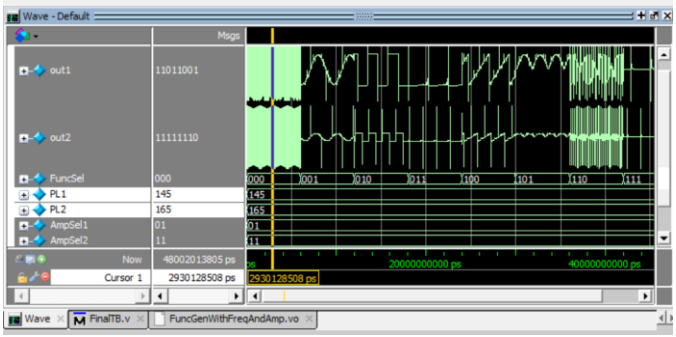

Fig. 11: Simulation results of two units with different Amplitudes. Unit 2, has Amplitude select 11 which divides the wave by 8. Meanwhile Unit 1, has Amplitude select 01 which divides the wave be 2. It can be seen that out2 has a smaller output range in compare to out1.

REFERENCES

[1] SN7407 Datasheet, Provided by Texas Instruments
[2] DM74LS191 Datasheet, Provided by Fairchild Semiconductor