Experiment #3 - Function Generator

Mohammad.heydari
810197494

Introduction

In this experiment we try to make a block to made different waveforms such as Rhomboid , Sine , Square , Reciprocal , Saw-tooth , Full-wave rectified , Modulated square wave and Arbitrary. This block can also change the amplitude of wave with deviding scnarios which you can see in the following. In this experiment we can also change the frequency of clk using scnarios in the previous part of this course . all of details have been described in the following.

1 Waveform Generator

This module is the heart of this project. It produces desired functions. Output of this module is an 8-bit digital representing the amplitude of signal.

<u>Note:</u> *<u>In order to do the mathematical operations with reasonable accuracy, operations are done in 16-bit fixed point. Also, considering the period of about 256 clock cycles from frequency selector, the equations turn to:</u>*

$\sin(n) = \sin(n-1) + 1/64.\cos(n-1)$
$\cos(n) = \cos(n-1) - 1/64\ .\sin(n)$

in this part using some knowledge we are going to write Verilog discribtions of each waveforms. After that I write a Tb to check the functionality of those waves. All of result have been attachad by Screenshots in folder & here with inShots.

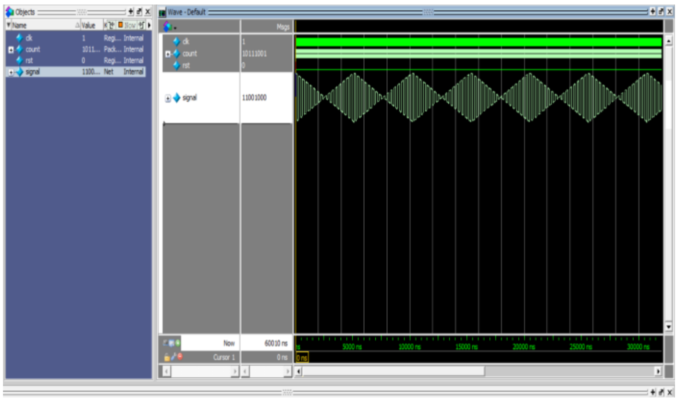


Fig. 1 Rhomboid waveform

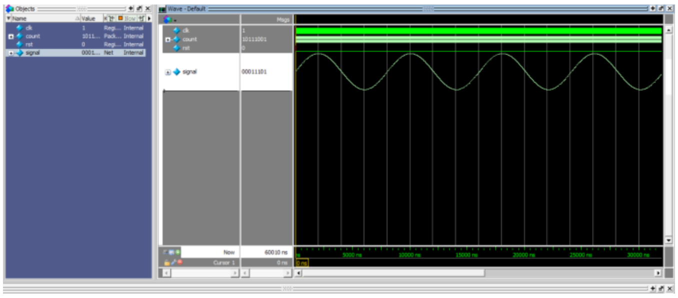


Fig. 2 Sin waveform

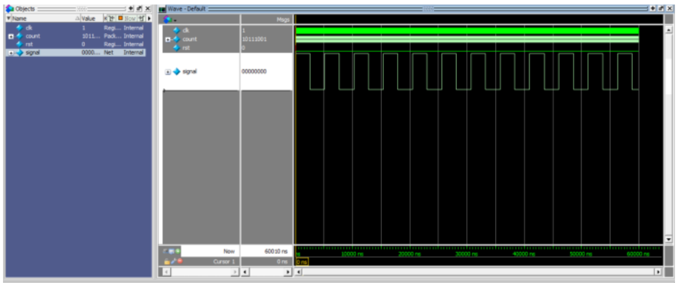


Fig. 3 Square waveform

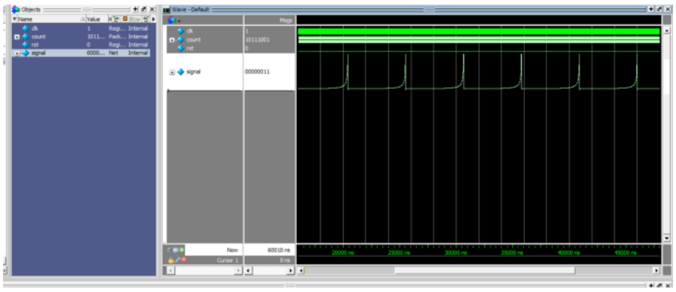


Fig. 4 reciprocal waveform

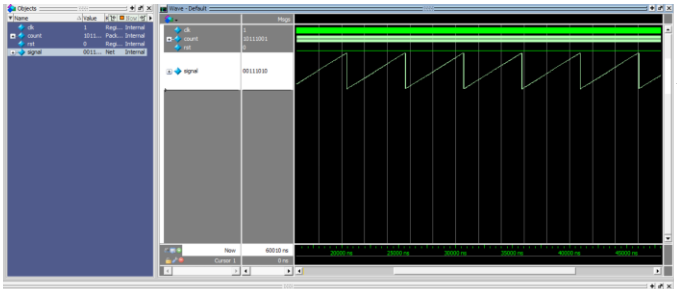


Fig. 5 Saw-tooth waveform

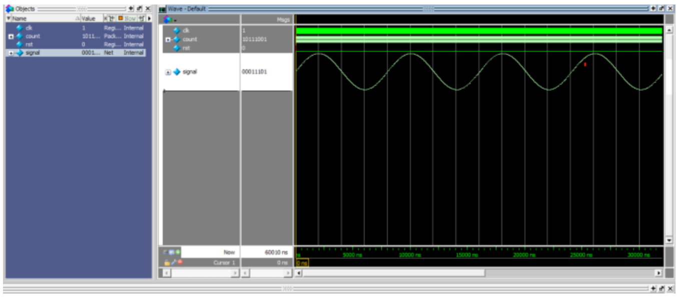


Fig. 6 Full-wave rectifiedwaveform

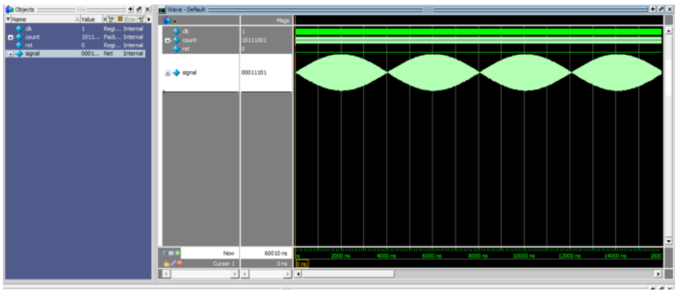


Fig. 7 Modulated square wave waveform

***Note:* we assume values are between -32768 to 32767 for sin and cos.**

Then according the below table we make the Wavegenerator.

Table 1: Function selection

func[2:0]	Function
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

According to the upper table we use a case statement to initial each waveforms equal to their values.

After that we open a schematic file in qrtus and using mux & and gates & Rom_1port and two counters start to design our block. After wiring we get a .vo file from this part called Waveform_Generator.

For our total design we consider a 12-bit input named SW that recalls the switch inputs in FPGA. We Dedicate 3 bits of this 12-bit input to the selectors of the waveform generator (SW[10:8]). The remaining bits will be used for the frequency and amplitude selectors in the Following.

After completing the waveform generator block diagram in Quartus,we synthesize the final design.

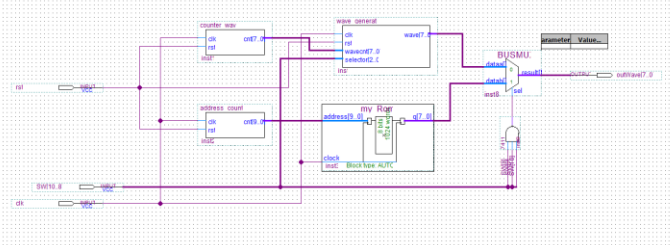


Fig. 7 schem of wave_generator in quartus

After passing all of steps we are going to write a TB to check the usage of Block .

<u>Note : For this part we use the 50-MHz clock frequency accordings to discribtion of lab .</u>

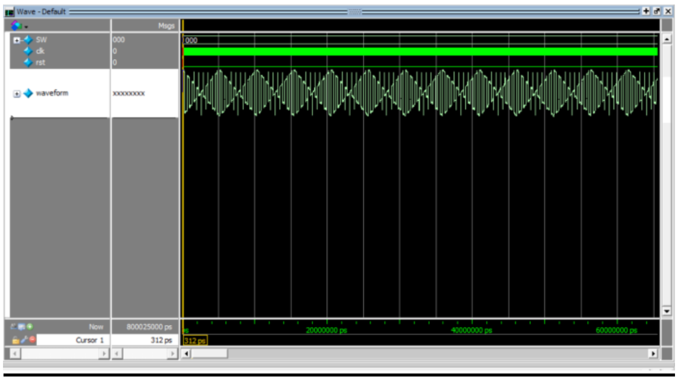


Fig. 8 state1

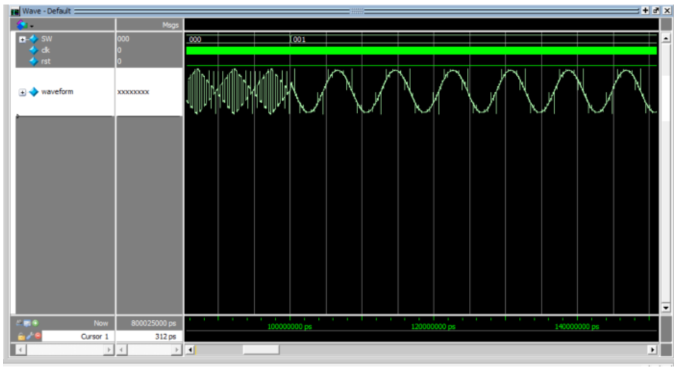


Fig. 9 state2

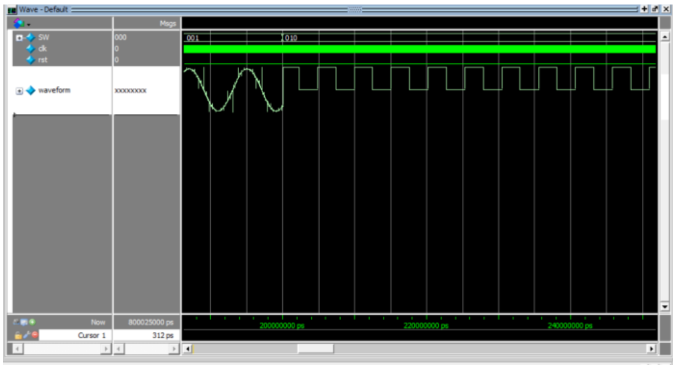


Fig 10 state3

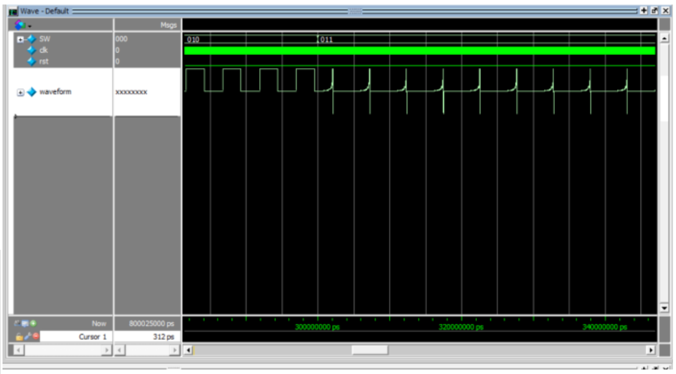


Fig 11 state4

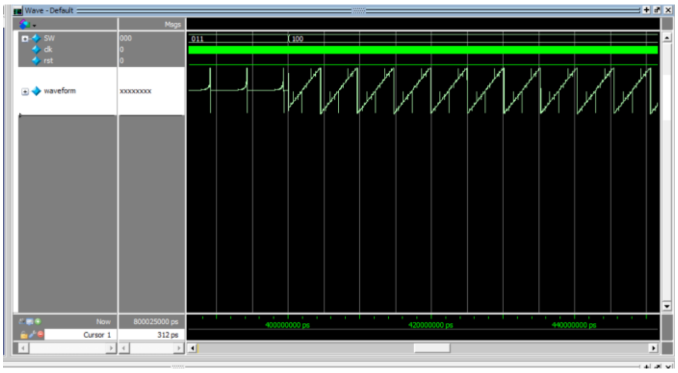


Fig 12 state5

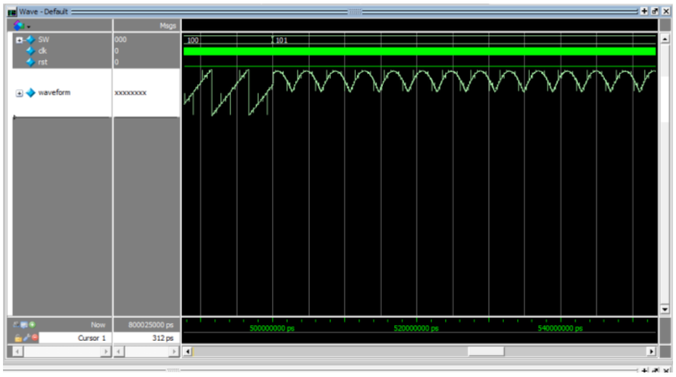


Fig 13 state6

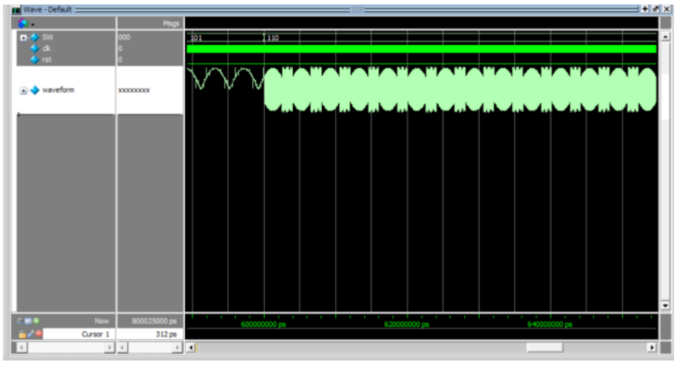


Fig 14 state7

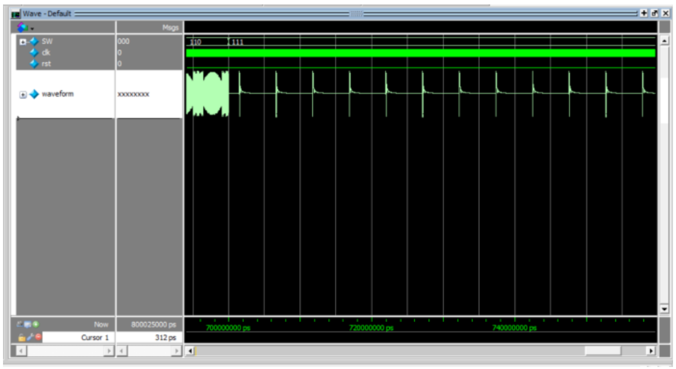


Fig 15 state8

Note:

all of results with high quality have attached in zip file both Screenshots & my_Codes

2 Frequency Selector

In order to set the frequency of the output signal a frequency selector is required. The frequency selector consists of a counter that divides a high source input signal to the desired value. You can take advantage of your previous design of the clock divider.

In this part we use divider of experiment 1 and do the wiring of the whole block. As you see in below figure.

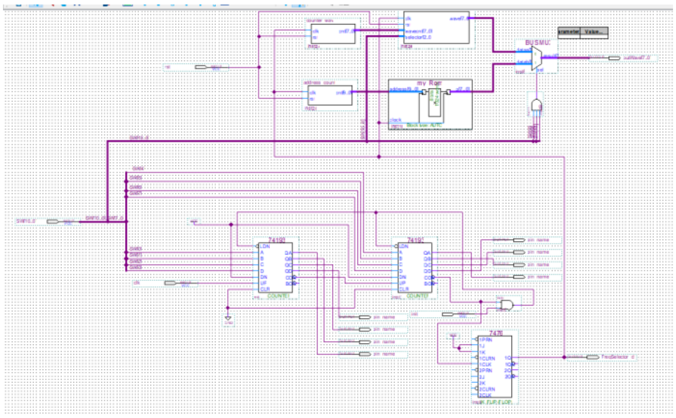


Fig 16   schem of frequency selector in quartus

In this part we Use the clock divider of Experiment 1 that uses a ring oscillator as the source signal.we Use the output of the clock divider as the desired clock for the function generator.

After that we synthesis the design unit in qrtus and get a .vo result we use SW[7:0] for the parallel loads of the divider.

At the final step we are going to modify the Tb that we wrote in previous part .

We show the result for frequency of 50,100 and 200 MHz in my TB module.

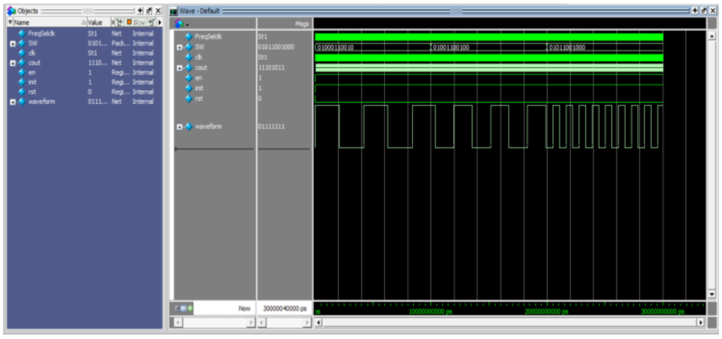


Fig 17 frequency_selector result

As we know to having easier than process to find duty cycle and see the various desired frequency we choose the **square** wave. Then to achive this goal give the SW[10:8]=010 then we have square wave in output.

The results have been attached

3 Amplitude Selector

One option in function generator is the amplitude of generated wave. The task of this module is to scale down the amplitude of the waveforms This can be done by dividing the output amplitude by a number. Value of divisor is chosen by a 2-bit input. Dedicate the last two bits of input SW (SW[12:11]) to this selector inputs.

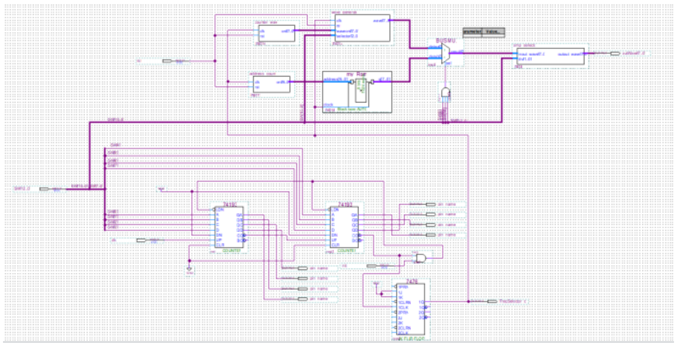


Fig 18 schem of amplitude selector in quartus

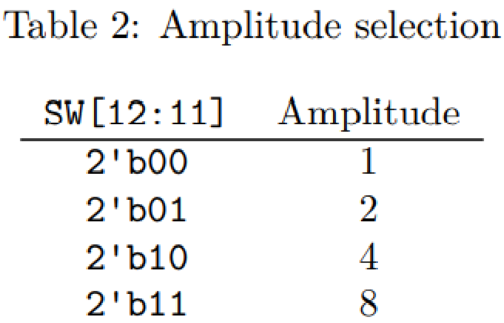


Table 2: Amplitude selection

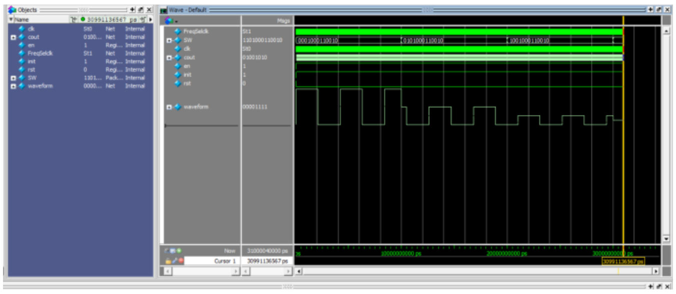
SW[12:11]	Amplitude
2'b00	1
2'b01	2
2'b10	4
2'b11	8

According to upper table we write the amplitude selector block via a case statement . then our design prepare to wiring and synthesizing.

After adding the amplitude selector module to the previous design,we synthesize our design and get a .vo file as our result .

In the last step we are going to modify the previous testbench and prepare that to work correctly in our case.

You can see the results of my works in below figures.



Name	30991136567 ps		
clk	St0	Net	Internal
cout	0100...	Net	Internal
en	1	Regi...	Internal
FreqSelclk	St1	Net	Internal
init	1	Regi...	Internal
rst	0	Regi...	Internal
SW	1101...	Pack...	Internal
waveform	0000...	Net	Internal

	Msgs
FreqSelclk	St1
SW	1101000110010
clk	St0
cout	01001010
en	1
init	1
rst	0
waveform	00001111
Now	31000040000 ps
Cursor 1	30991136567 ps

Fig 19 schem of amplitude selector in quartus

Conclusions

in this experiment we are going to design an function generator that have the abilities such as changing frequency & amplitude changing and also 8 waveform that stored in out block memory via 1port ROM. Also function generator is a specific form of signal generator that is able to generate waveforms with common shapes. Unlike RF generators and some others that only create sine waves, the function generator is able to create repetitive waveforms with a number of common shapes. And as we know this is very useful in basic of logical circuits.

References

[1] Some entries at github
[2] Some entries at stackoverflow.com
[3] Some entries to debug my codes
[4] A Review of below site

https://www.electronics-notes.com/articles/test-methods/signal-generators/function-generator.php