

FPGA BASED FUNCTION GENERATOR

Mrs Aarti H. Tirmare¹, Ms.Sangita R .Mohite², Ms.Priyadarshni S. Mali³, Mrs Varsha.A.Suryavanshi⁴

^{1,2,3} Assistant professor, E&TC Department, Bharati Vidyapeeth College of Engg.Kolhapur, India.

⁴ Assistant professor, E&TC Department, KIT,Kolhapur, India.

Abstract - This paper presents an implementation of Function Generator using Field Programmable Gate Array (FPGA).. The proposed Function Generator is first described in Very High Speed Integrated Circuit Hardware Description language (VHDL) and the description is verified using ModelSim SE 10.1 E from Model Tech. This function generator is built with same facilities which are available in function generator for practical implementation in lab. Which provide signals like Sine, square, triangular and ramp waveforms. This modified Function Generator is compact in size and re-configurable. This paper focuses on advantage of soft core over hard core by providing flexibility in modifying variation in frequency and waveform selection by using rotary switch button .

Key Words: IP core, HDL, ASIC.

1 INTRODUCTION:

Function Generator is a device that can produce various patterns of waveforms at a variety of frequencies and amplitudes. Basically, a function generator is used to generate signal with precise controlled frequency and amplitude characteristics to mimic the input signal of the circuit being tested. It is generally used in designing, testing and troubleshooting electronic devices. Signal generators generally fall into one of the two categories: function generators and arbitrary waveform generators.

Function generators are the simpler of the two types. They produce simple repetitive signals in waveforms such as a square waves, sine waves and triangular waves. Field-Programmable Gate Array (FPGA) provides an attractive platform for these Function generators in-terms of performance, power consumption and flexibility in configuration.

FPGA is a semiconductor device that can be configured by the customer or designer after manufacturing. FPGAs are programmed using a logic

circuit diagram or a source code in a Hardware Description Language (HDL) to specify how the chip will work. The most common HDL used to program FPGA is Very high speed integrated circuit Hardware Description Language (VHDL) and Verilog. The main objective is to design a Function generator using Field-Programmable Gate Array (FPGA) to generate a few types of waveforms - square waves, triangular waves and sine waves are the main objective of this project. As technologies are fast changing, a modifiable tool is essential and comparing to those high-priced signal generation instruments, an FPGA-based signal generator fits the bill. By modifying the soft coded VHDL or Verilog, we can develop a Function generator catering to our needs. Through research, design, programming and analysis, the goal of this project could be attainable. Function generators deliver a sinusoidal output of accurately calibrated frequency. The output signals are usually frequency or amplitude modulated. Signal generators are typically used to measure the output in simple electronic repair and design, therefore output accuracy is critical. Accuracy is the way in which the output level of a function generator is controlled. When generating the signal, an attenuator is used to maintain this output. The stability and intensity of the signal is maintained and made stronger by the inbuilt amplifier. An uncommon alternative expansion is "integrated processor block". IP cores can be used as building blocks within ASIC chip designs or FPGA logic designs. In digital-logic applications, IP cores are typically offered as generic gate net lists. The net list is a Boolean-algebra representation (gates, standard cells) of the IP's logical-function, analogous to an assembly-code listing for a high-level program application. The net list protects the vendor against reverse-engineering, while maintaining portability to multiple foundry targets. Some vendors also offer synthesizable versions of their IP cores. Synthesizable cores are delivered in a hardware description language such as Verilog or VHDL, permitting customer modification (at the functional level). Both net list and synthesizable cores are called 'soft cores', as both

follow the SPR design-flow (synthesis, placement and route.)

For digital applications, soft cores and hard cores serve different roles. Soft-cores offer greater customer flexibility, while hard-cores, by the nature of their low-level representation, offer better predictability in terms of timing-performance and area. IP cores in the electronic design industry have a profound impact on the design of SOCs. The paper is organized as follows. Section 2 describes Design of proposed Function Generator in brief. Section 3 focuses on efficient FPGA implementation of Function Generator. The experimental results are given in Section 4. Finally Section 5 ends up with conclusion and future work.

2. DESIGN OF IMPLEMENTED FUNCTION GENERATOR:

Figure 1 shows the block diagram of a Function generator. The Waveforms Generator Engine will output waveform from the waveforms sequence stored in Onboard Memory. This output waveform will go through Digital Gain for amplification or attenuation before heading for the Digital Filter to be interpolated. The interpolated waveform will go through Digital-to-Analog Converter (DAC) to output analog waveform triggered by the Clock. This analog waveform will lastly go through the Analog Filter to have most or all of its unwanted signals removed before generating the ideal output.

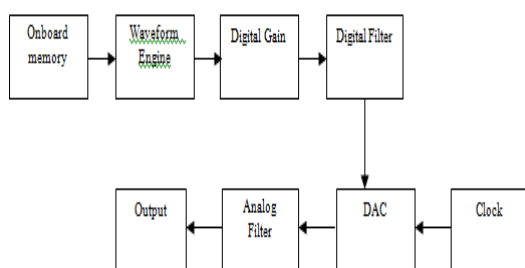


Fig. 2.1 Block Diagram of Implemented Function Generator

2.1 Onboard memory:

On the memory allocation for waveforms and sequence instructions stored on the Onboard Memory device of a function generator. The signal generator requires limited memory to store a single period of the

waveforms since its outputs are repetitive and with a standard library it is able to generate periodic waveforms.

2.2 Waveform generation engine:

Waveform Generator Engine is a program to link and loop waveform segments. Linking and looping can be divided into sequence generation mode and script generation mode. Outputting a predetermined series of waveforms with the sequence instructions stored in the onboard memory, script generation mode can have a waveform sequences that depends on an external or internal trigger to generate an output signal.

2.3 Digital gain:

The Amplifier and Attenuator are to maximize the digital signal's amplitude accuracy. When amplified signals are output as analog signal after DAC, users are able to adjust the amplitude of the signal without the need to reload a different waveform. DAC is to convert digital waveforms in the memory to analog waveforms.

2.4 Digital filter and Analog filter:

Both the digital and analog Filter is used to provide the best approximation of an ideal analog signal. During digital to analog conversion, digital filter are used to interpolate the signals to increase the effective sampling rate. But the digital filter might not be able to remove unwanted signals completely. The analog filter is able to attenuate these DAC signals and remove the unwanted signals through applying a low pass filter, high pass filter or a band pass filter.

2.5 Digital-to-analog Conversion:

Clocking of DAC is critical as it will affect the frequency accuracy and its effect is measurable. Referring to Figure 2.2, whenever the clock clocks on the rising edge, the DAC will generate the output signal with the sampled points of the single period waveforms sequence stored in the memory.

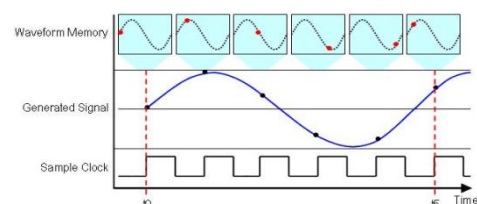


Fig. 2.2 Output Signal with Clocking

3. EXPERIMENTAL SETUP:

All the blocks shown above are tested separately on test bench & finally all are connected together to form Function Generator. To check the waveforms of the frequency range 0.1Hz to 100 KHz, in the simulation results. Then by using USB cable port personal computer is connected to Spartan3E FPGA kit. After this process Function Generator is implemented. Spartan-3E family and XC3s500e-5fg320 device. Following photographs shows generation of different signals.

Experimental setup1 shows generation of triangular wave. Setup2 shows generation of sine wave & setup 3 shows generation of square wave



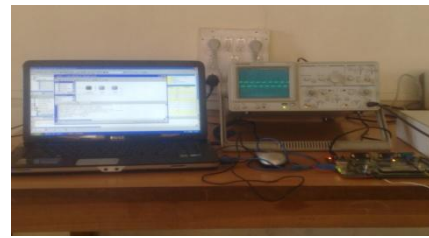
Experimental setup1



Experimental setup2



Experimental setup3



Experimental setup4

4. RESULTS:

All the results are based on simulations from the Modelsim SE 10.1e from ModelTech, using Test Bench Waveform Generator. Sin, square and triangular waveforms are formed in simulation results. All the waveforms are simulated in combine.

4.1 Simulation Results:

There are three results shown for three different frequencies and same amplitude. Results are shown for different three frequencies by keeping frame size= 8 bits.

Fig. 4.1 consists of sin, square and triangular waveforms with 1KHz frequency.

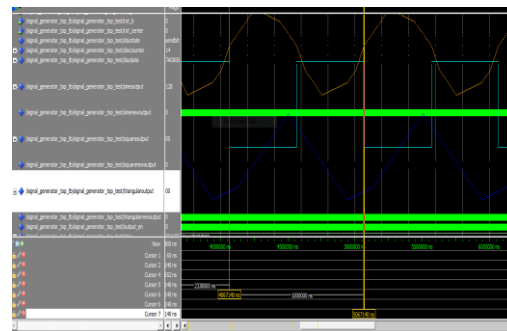


Fig. 4.1: With 1 KHz Frequency

Fig. 4.2 consists of sin, square and triangular waveforms with 10KHz frequency.

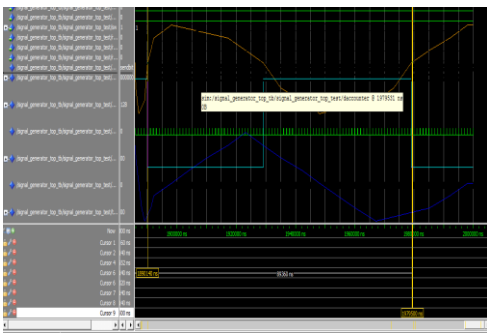


Fig. 4.2: With Frequency 10KHz

Fig. 4.3 consists of sin, square and triangular waveforms with 100KHz frequency.

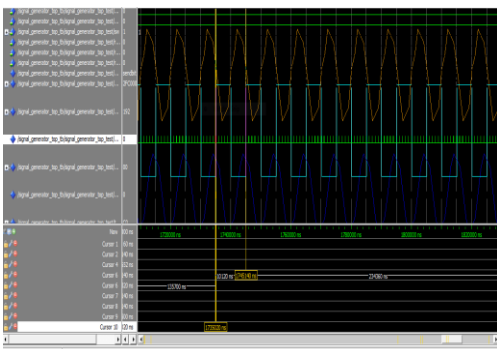


Fig. 4.3: With Frequency 100KHz

4.2 Synthesis report:

HDL Synthesis Report

Macro Statistics

- # ROMs : 3
- 16x3-bit ROM : 3
- # Adders/Subtractor : 7
- 4-bit adder : 3
- 5-bit subtractor : 1
- 6-bit subtractor : 3
- # Accumulator : 3
- 49-bit up accumulator : 3
- # Registers : 36

- 1-bit register : 14
 - 2-bit register : 1
 - 24-bit register : 1
 - 3-bit register : 3
 - 4-bit register : 3
 - 48-bit register : 1
 - 5-bit register : 1
 - 6-bit register : 3
 - 8-bit register : 6
 - 9-bit register : 3
 - # Multiplexers : 2
 - 1-bit 4-to-1 multiplexer : 2
- =====

Advanced HDL Synthesis Report

Macro Statistics

- # FSMs : 4
- # ROMs : 3
- 16x3-bit ROM : 3
- # Adders/Subtractors : 7
- 4-bit adder : 3
- 5-bit subtractor : 1
- 6-bit subtractor : 3
- # Accumulators : 3
- 49-bit up accumulator : 3
- # Registers : 159
- Flip-Flops : 159
- # Shift Registers : 2
- 2-bit shift registers : 2

Multiplexers : 2

1-bit 4-to-1 multiplexer : 2

Device utilization summary:

Selected Device : 3s500efg320-5

Number of Slices: 190 out of 4656 4%

Number of Slice Flip Flops: 291 out of 9312 3%

Number of 4 input LUTs: 360 out of 9312 3%

Number used as logic: 358

Number used as Shift registers: 2

Number of bonded IOBs: 21 out of 232 9%

Number of GCLKs: 1 out of 24 4%

Timing Summary:

Speed Grade: -5

Minimum period: 7.339ns (Maximum Frequency: 136.266MHz)

Minimum input arrival time before clock: 5.748ns

Maximum output required time after clock: 4.989ns

Maximum combinational path delay: No path found

5. CONCLUSIONS

FPGA based Function Generator is compact in size and re-configurable. Frequency can be varied by modifying VHDL code. Switches are used in conjunction with selectable parameters which allows to user to select desired output waveforms type. Frequency changes by using Rotary Push button.

REFERENCES

[1] S.M.Trimberger. "Field Programmable Gate Array Technology" Kunwedemic publisher 1999.

[2] Application specific instruction processor implementation on FPGA.Architecture IEEE Vol. 87 JAN 2007.

[3]Sangita Mohite, Aarti Tirmare, Priyadarshani Mali "OVERVIEW OF RISC PROCESSOR USING VHDL",Volume: 02 Issue: 07 | Oct-2015. e-ISSN: 2395-0056. p-ISSN: 2395-0072.

[4]Spartan-3A/3AN FPGA Starter Kit Board User Guide (Online)

http://www.xilinx.com/support/documentation/boards_and_kits/ug334.pdf

[5] How Does a Signal Generator Work? (By an eHow Contributing Writer, Online)

http://www.ehow.com/how-does_4970115_signal-generator-work.html

[6] Signal Generator (Online Encyclopedia)

http://en.wikipedia.org/wiki/Signal_generator

[7] Christopher Ziomek, Differences between Signal Generators, Function Generators and Arbitrary Waveform Generators, Jun 22 2009 (Online)

<http://blog.ztecinstruments.com/bid/22907/Differences-Between-Signal-Generators-Function-Generators-and-Arbitrary-Waveform-Generators>

[8] Signal Generator Architecture - Analog Output to Advanced Features (National Instruments, Online)

[9] DAC Data Sheet (Online)

<http://www.linear.com/pc/downloadDocument.do?navId=H0,C1,C1155,C1005,C1156,P2048,D2170>

[10] Spartan-3E FPGA Starter Kit Board User Guide (Online)

http://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf

[11] Rotary Encoder Interface for Spartan-3E Starter Kit (Online)

http://www.xilinx.com/products/boards/s3estarter/files/s3esk_rotary_encoder_interface.pdf

[12] Hardware Description Language (Online Encyclopedia)

http://en.wikipedia.org/wiki/Hardware_description_language (Online Encyclopedia)