EE314 Digital Laboratory Project Final Report An FPGA Based Oscilloscope

Bekir Buğra Cömert, Yüksel Mert Salar
Middle East Technical University, Department of Electrical and Electronics Engineering, Ankara, Turkey

This report includes all works related to EE314 Digital Electronics Laboratory Term Project namely an FPGA based oscilloscope which is designed by authors.

***Index Terms*—FPGA, Oscilloscope, Verilog, ADC, RAM, VGA**

I. Introduction

In this project, we are required to design an FPGA based oscilloscope with some specifications. Our oscilloscope must be able to visualize different waveforms via VGA screen. The screen must renew itself once for every second and oscilloscope must have two main modes: AC and DC. The signal will be shown using a reference line representing ground and the signal have to be shown with its' mean value first. Our design must also include peak to peak voltage measurement, autoscale button, period and frequency calculator. Thus, we made deep research about oscilloscope and FPGA. At the end of the research, we conclude that our design must include four main parts: Analog to digital converter, RAM, computation unit and VGA screen. Since we deal with different waveforms using FPGA, firstly the analog signal coming from generator must be turned into a digital signal. Then, we made brief research about ADC on the FPGA. We learned the basic properties of this module and how to use it. Now, the digital signal must be registered in a way to process to it later. We designed a RAM module for this purpose. After registering the signal, it is ready to process. We used different algorithms for different functions of computation unit. Finally, our signal is ready to display on a screen. For this purpose, we learned the details of VGA screen and program a simple algorithm using background data we found during our research.

II. System Components

- DE1-SoC development board (FPGA)
- VGA connector
- Monitor

III. ADC (Analog to Digital Converter)

Analog to digital converter is an embedded unit on FPGA. We read the datasheet of the FPGA and Intel tutorial to understand deeply how ADC works and how we can program it according to our purposes. This module can take input signal from 0V to 5V to convert them into digital signal. It has 8 pins for input signals. However, one channel is enough to receive signal. Therefore, we will set the address bits as 000 so that only channel 0 is used as a probe of oscilloscope. To set ADC for taking voltage and sending serial data to ADC_DOUT pin, we found the datasheet of ADC 7928 [3] and scanned it. According to datasheet, analog voltage value seen by ADC will be multiplied by 1000 and then turns into 12-bit digital signal. We set the parameters which must be given from ADC_DIN considering timing requirements as shown in Appendix C. In the timing part, write parameter must be logic 1 to take voltage from inputs. Seq and shadow parameters must be 0, PM0 and PM1 must be 1 to run in normal mode. Three address bits must be 0 to set channel 0. Finally, range must be 1 to take voltage in range 0-5V and coding must be 0 to use straight binary format. We arranged the frequency of ADC clock as 12.5 MHz. After all that, we have 12-bit serial output. It is obvious that 12-bit serial signal is hard to use for RAM and computational unit. Therefore, we programmed a 12-bit shift register by using arrays and turned serial input into parallel output. Now, our signal is ready to use for RAM module.

IV. RAM

We decided to take 40,000 samples in one second which is also the period of the duration of one frame in VGA screen. Every sample includes 12-bit data. Thus, we need at least 65536 cells which have 16-bit address bus defined as:

```
reg [11:0] ram [15:0];
```

In RAM module, we have created two states which are writing and fast reading modules. In the writing state, program writes a 12-bit sample to an address of RAM in every period of ADC reading which is 128 x CLK_50 period. After 40,000 addresses are written, module set fast reading state. In this mode, samples in addresses are read with 25 MHz by the computational unit to be processed instantaneously.

V. Computation Unit

Computation unit includes functions of desired oscilloscope such as peak to peak value, period and offset. To calculate peak to peak value and offset, first we find minimum and maximum values of the signal by comparing each value in the signal with current minimum and maximum values. If current value is greater than maximum value, then new maximum value will be the current value, vice versa for the minimum value. After calculated minimum and maximum values, we can find peak to peak value by subtracting minimum value from maximum value. Offset value is the average of minimum and maximum values. We can find period by calculating the number of cycles from one voltage value to same voltage value but the second time. To calculate frequency, we divided 100,000 into period to include maximum number of decimal digit of (1/period). If we assume one clock cycle is 80 ns (clock of ADC, 12.5 MHz), then the period is calculated as equation (1):

$$T_{real} \quad = \quad 40Tns \tag{1}$$

where T is the period shown in simulation. Then, the frequency is calculated as equation (2):

$$f_{real} \quad = \quad \frac{12.5f}{100,000}\ MHz \tag{2}$$

where f is the frequency shown in simulation. In Appendix II, the simulation of computation unit is shown. We have created a sine wave in MATLAB with 10 samples; that is period with 10. Then, in testbench we used these sampled data.

VI. VGA

In this part, we have used two modules to visualize processed signal data on the screen. The first module is 'vga640x480' which is used to set horizontal and vertical synchronization. To write this module, we have benefit from the VGA appendix on the project document on Odtuclass. In this module, we created two variables called VGA_HS_O and VGA_VS_O which is used for synchronization. They are directly connected to monitor by making pin assignment. There are also two variables called x and y which represents a pixel on screen. They are connected to another module called 'vga'. This module is the main module which decides the pixel and the color energized. Thus, processed signal data, timediv and voltdiv data comes from computation module to 'vga' module. Then, by using these three variables, it is decided to which pixel is energized. We only use one color and one tone which is green.

VII. Conclusion

In this project we basically made an oscilloscope by programming FPGA. Firstly, we learned the details of FPGA by reading User Manual and other online resources. We learned the operation principles of analog to digital converter. Then, we programmed it according to requirements. During, programming we considered timing issue of ADC. Timing is the most important part of ADC. Then, we tried to create a RAM module to store data. However, we could not operate it properly and the connection between ADC and computational unit is missing in our design. However, we discovered another way to test our oscilloscope's functions. We used MATLAB to create a sample signal and used it for computation unit by using test bench codes. Computational unit simulations were successful. Finally, for visualization we learned the basic principles of VGA screen. We created a 640x480 screen but since we could not operate RAM module we put three squares in different places with red, blue and green colors. We learned how to create a VGA screen via FPGA. To conclude, due to this project we found a chance to test our Verilog and logic information and we used our theoretical information to create a cheap and easy-to-use oscilloscope. Most significantly, we experienced on making research project during this project.

Appendix A

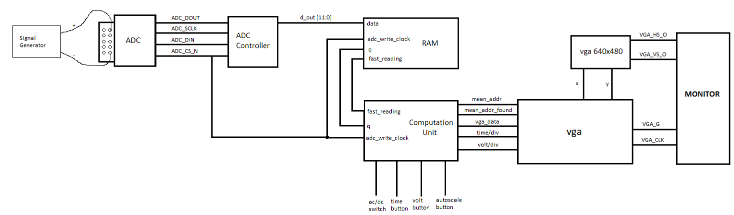

Fig. 1 The block diagram of all system

Appendix B
Simulation Results

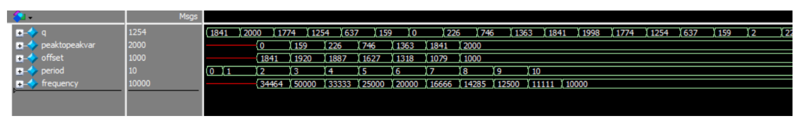

Fig. 2 The simulation result of computation unit

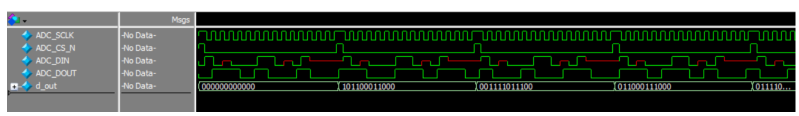

Fig. 3 The simulation result of ADC

Appendix C

MSB / LSB

WRITE	SEQ	DON'TCARE	ADD2	ADD1	ADD0	PM1	PM0	SHADOW	DON'TCARE	RANGE	CODING

Fig. 4 [3] The parameters of ADC must be set

References

[1] DE1-SoC User Manual, Terasic Technologies Inc., Hsinchu, Taiwan, 2016
[2] Using the DE-Series ADC Controller, Intel Corporation, Santa Clara, CA, 2017
[3] AD7928 Datasheet, Analog Devices, Norwood, Massachusetts, 2014