University of Tehran
College of Engineering
School of Electrical & Computer Engineering

Experiment 3
Sessions 7,8,9

Function Generator

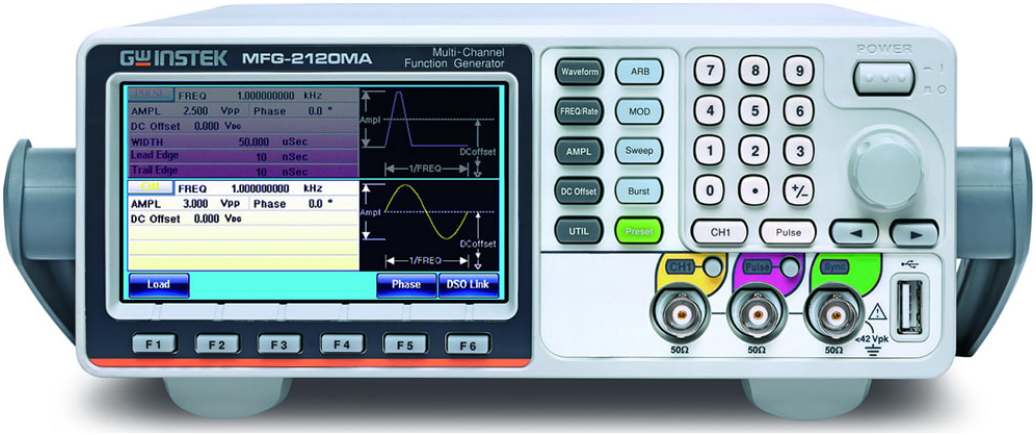
Digital Logic Laboratory
ECE 045
Laboratory Manual

Spring 1400

Contents

Prepared and developed by Katayoon Basharkhah under supervision of Professor Z. Navabi

Figure 1: A single channel AFG from Instek company



Introduction

An Arbitrary Function Generator (AFG) is an electronic test instrument that generates a wide variety of waveforms with different amplitude and frequency. Among them are sine, square, rhomboid, saw-tooth and any arbitrary waveform. You can use AFGs to simply generate a series of basic test signals, replicate real-world signals, or create signals that are not otherwise available. These signals can then be used to learn more about how a circuit works, to characterize an electronic component, and to verify electronic theories.Figure 1 shows a real AFG from Instek Company.

In this experiment, you are to design an Arbitrary Generator that is capable of generating each of the aforementioned waveforms with wide range for frequency selection. You will make use of the clock divider that you have designed in the Experiment 1.

By the end of this experiment, you should have learned:

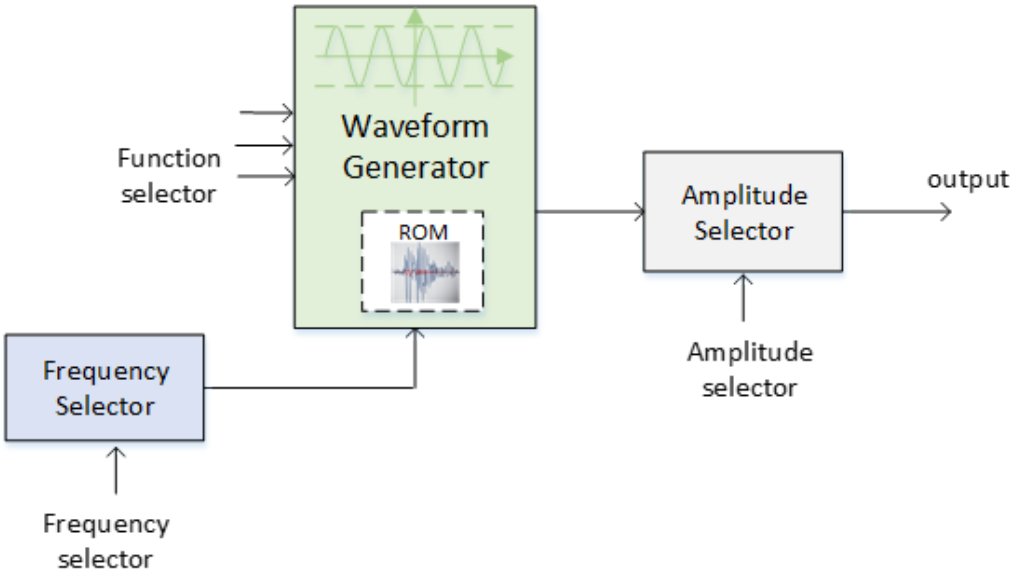
- The principle of function generators
- Using ROM memories in your design
- Schematic design in Quartus II

Figure 2 shows the simplified block diagram of an arbitrary generator. Based on these specifications there is a main component that generates one of the desired waveform based on the function selectors' value, a frequency selector that sets the output signal frequency, an amplitude selector. In an AFG a DAC is also used to convert the digital output to an analog signal, which can be observed and evaluated via an oscilloscope. Since in this experiment you don't have access to real analog elements like resistor and capacitor, this part is not considered in this experiment. A ROM memory is usually embedded to store any other arbitrary waveform.

Accordingly, Below is the topics that are explained in the following of this experiment in details:

- Waveform Generator
- Frequency Selector
- Amplitude Selector

Figure 2: Block diagram of an Arbitrary Function Generator (AFG)



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. The supported functions, shown in figure 3, are rhomboid, sine, square, reciprocal, triangle, full-wave and half-wave rectified signals and sinusoidally modulated square wave.

- Waveforms rhomboid, square, reciprocal and triangle are based on a counter that counts up or down with each clock for the period of the waveform. The output of the frequency selector is the input for this module that determines the discrete incremental values of this signal (resolution).
- The full-wave and half-wave rectified and sinusoidally modulated square wave are all generated based on the sine wave. So before generating these waves, you first need to generate a sine wave. Generation of sine function can be somewhat different. The following second order differential equation can be used to generate the sine function:

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

  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

Figure 3: Different waveforms of function generator

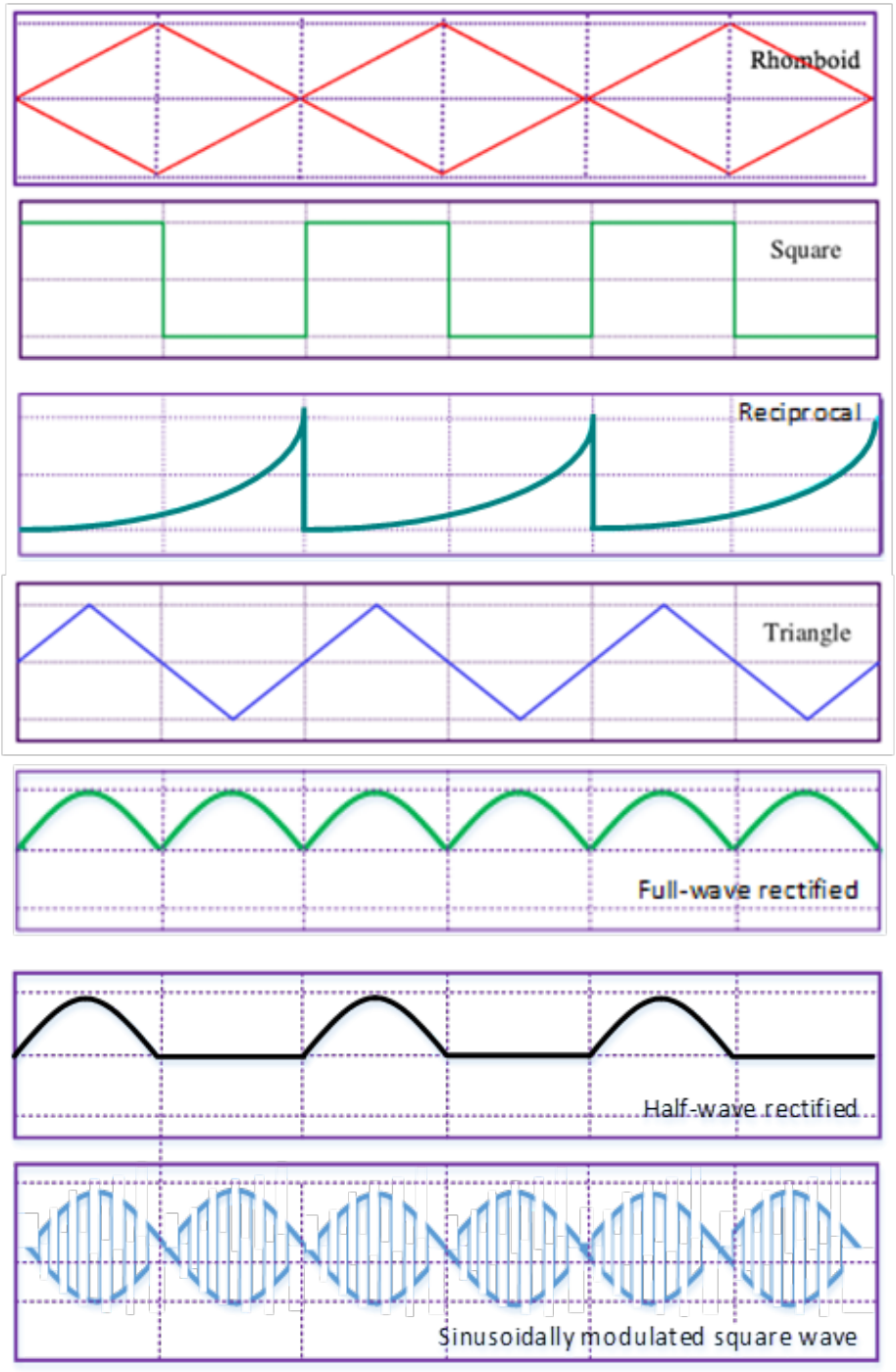
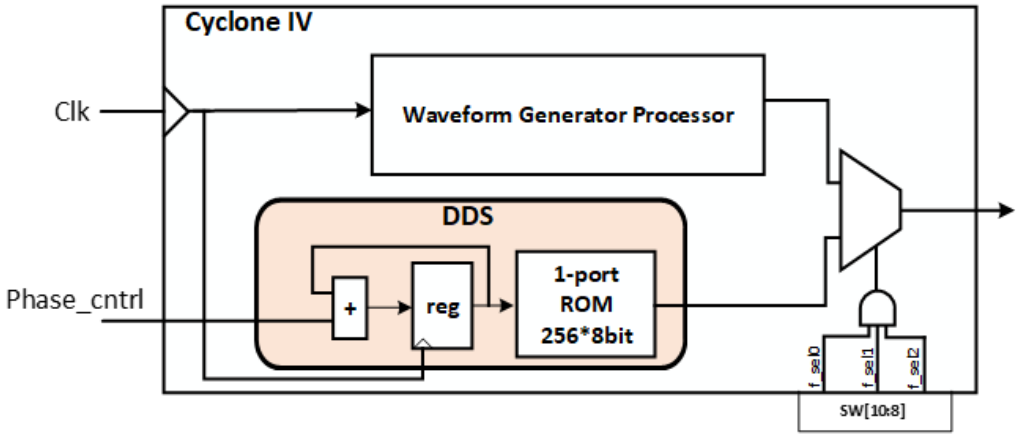


Figure 4: Block diagram of waveform generator



selector, the equations turn to:

$$\sin(n) = \sin(n-1) + \frac{1}{64}.\cos(n-1)$$
$$\cos(n) = \cos(n-1) - \frac{1}{64}.\sin(n)$$

Assuming values are between -32768 to 32767 for sin and cos.

Initialization of first values in differential equations is necessary. Use 0 for sin(0) and 30000 for cos(0). The results of sine and cosine operations are signed and between -127 to +128. However, for simplification and compatibility with other parts of this experiment, we add an offset of 127, making the range of our signal between 0 and 256.

- Now use the sine wave and the counters in a way that generates waveforms similar to figure 3.
- Most modern function generators use Direct Digital Synthesis (DDS) for generating their output waveforms. DDS is used for generating arbitary phase tunable output from a single fixed-frequency reference clock (like an oscillator). The output of DDS module is a quantized version of the output files (usually a sinusoid). The period of this signal is controlled by a Phase control value. To generate the DDS signal, you should use a 1-port ROM memory to store the value of a sine wave for several clock cycles. You will receive a file named `sine.mif` that is used for ROM initialization. A register and adder can generate the address location of this memory. This is done by incrementing the phase of the signal each time by the value of the `Phase_cntrl`. This module is usually called phase accumulator. The clock signal for DDS module should be a stable clock frequency. In this part the 50-MHz clock frequency of FPGA is used but in the total design this clock will be changed to the ring oscillator divided clock in next sections.

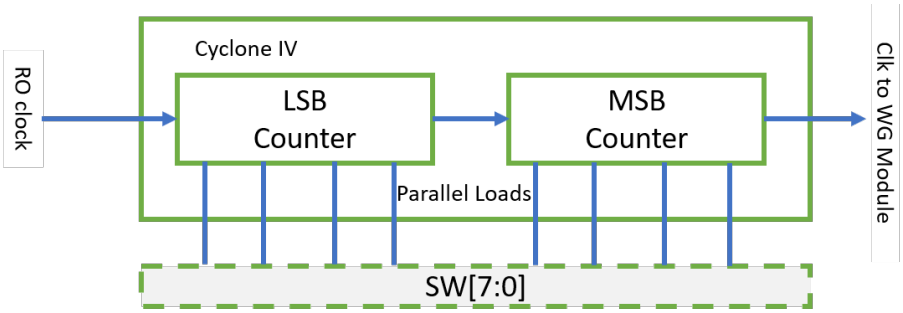
  Figure 4 shows the block diagram of the waveform generator module.

Table 1 shows the order of function selection. These are the control signals for the waveform generator module.

Table 1: Function selection

`func[2:0]`	Function
`3'b000`	Rhomboid
`3'b001`	Square
`3'b010`	Reciprocal
`3'b011`	Triangle
`3'b100`	Full-wave rectified
`3'b101`	Half-wave rectified
`3'b110`	Modulated square wave
`3'b111`	DDS

Figure 5: Block diagram of frequency selector



1. Write the Verilog code for the Waveform Generator Processor in figure 4 and use LPM or Megawizard functions for ROM memory, Multiplexer and other required gates.You will finally get all the components together in a schematic design. For your total design consider a 12-bit input named SW that recalls the switch inputs in FPGA. 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.

2. After completing the waveform generator block diagram in Quartus, synthesize the final design and include the synthesis summary in your report.

3. Before adding the frequency and amplitude selector, it is better to test your design in Modelsim. Therefore, write a simple testbench and verify the functionality of your design. For this part use the 50-MHz clock frequency. Include the Modelsim results for all the waveforms mentioned in Table 1. Set the value of `Phase_cntrl` to 1 for this part.

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. For this purpose, you can either write the Verilog description of a simple 8 bit counter like figure 5 or you can take advantage of your previous design of the clock divider in LAB1.

1. After adding the clock divider to the waveform generator design, synthesize your design.
2. Use SW[7:0] for the parallel loads of the divider.
3. Modify the testbench of section one so that you can verify the design for different desired frequency. To do this, change the Setperiod input (SW[7:0]) for at least three different frequencies.
4. Include the waveforms of each desired frequency in your report with mentioning the achieved frequencie, the input parallel loads and expected frequencies.
5. As mentioned the DDS module can generate sine wave with different phases based on the `Phase_cntrl` input. As a different way of frequency selection verify your design for at least 3 values of `Phase_cntrl` and observe the frequency change in the waveforms. Include the waveforms in your report and explain about the expected and achieved frequency.

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.

Type of amplitude is selected by SW[12:11] according to table 2.

1. After adding the amplitude selector module to the previous design, synthesize your design.
2. Modify the testbench of section two so that you can verify the design for different amplitude. To do this, change (SW[12:11])in your testbench.
3. Include the waveforms of each desired amplitude in your report.

Finally your top-level design must look like the block diagram of figure 6.

Acknowledgment

This lab manual was prepared and developed by Katayoon Basharkhah, PHD student of Digital Systems at University of Tehran, under the supervision of professor Zain Navabi.

This manual has been revised and edited by Maryam Rajabalipanah, PHD student of Digital Systems at University of Tehran.

Table 2: Amplitude selection

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

Figure 6: Final block diagram in Quartus II