Quadrature Oscillator

Davi Carvalho Moreno de Almeida
davicmorenoa@gmail.com

June 29, 2021

Abstract

This document contains the design of a low cost quadrature oscillator developed for FPGA.

Contents

1 Description

A simple quadrature oscillator that generates two quadrature waves (a sine and a cosine) with a period equal to 100 samples.

The implementation is designed for systems with hardware constrains, so that no multiplier is used and the sine and cosine samples are generated at each clock cycle with only 6 additions. The generated waves have a resolution of 8 bits (can be manually adjusted to up to 16 bits).

The oscillation frequency can be controlled by the clock frequency divided by 100 (eg 10 kHz clock frequency generates 100 Hz waves).

Repositories:

- Link for the SVN repository.
- Link for the GitHub repository.

2 Design

2.1 Mathematical development

The mathematics behind the functioning of the system lies in the design of marginally stable discrete systems (that is, with simple poles present in the unit circle of the Z plane).

To obtain the desired outputs, it is then necessary to design two systems with impulse response equal to sine and cosine. One can find the Z transform of such systems in books like [OS09], and they are given by:

$$\begin{aligned} \sin(\omega_o n)u[n] &\overset{\mathcal{Z}}{\longleftrightarrow} \frac{\sin(\omega_o)z^{-1}}{1-2\cos(\omega_o)z^{-1}+z^{-2}} \\ \cos(\omega_o n)u[n] &\overset{\mathcal{Z}}{\longleftrightarrow} \frac{1-\cos(\omega_o)z^{-1}}{1-2\cos(\omega_o)z^{-1}+z^{-2}}, \end{aligned} \tag{1}$$

where ω_o is the discrete oscillation frequency in radians. Note that the recursive part (the denominator) of the systems is the same, so it is possible to use only one recursive module for both systems.

The block diagram used to implement the system is given in Figure 1.

2.2 Calculation of coefficients

For an oscillation period of 100 samples, the corresponding discrete frequency is calculated as:

$$\omega_o = 2\pi\left(\frac{1}{100}\right) = \frac{\pi}{50} \text{ rad.} \tag{2}$$

Thus, using a signed fixed point numeric representation with 8 fractional bits, the approximation of each coefficient is calculated as:

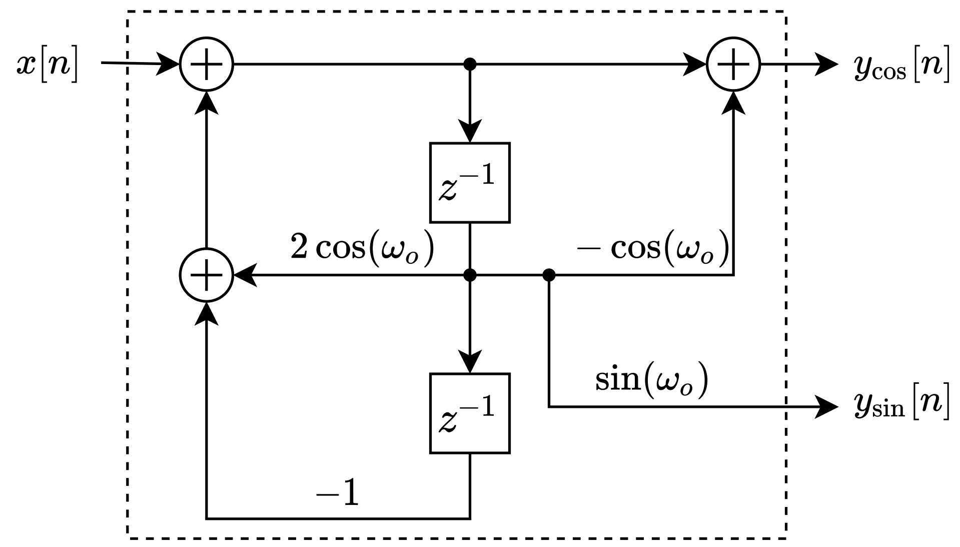

Figure 1: Quadrature oscillator system block diagram.

- $-\cos\left(\frac{\pi}{50}\right) \approx -\frac{255}{256}$
- $\sin\left(\frac{\pi}{50}\right) \approx \frac{16}{256}$
- $2\cos\left(\frac{\pi}{50}\right) \approx \frac{511}{256}$

In order to not need any multiplier, it was used multiplier adder graphs (like the ones shown in [MB14]). In this way it was possible to reduce the complexity so that no multiplier is used and the sine and cosine samples are generated at each clock cycle with only 6 additions.

Therefore, multiplications by 511, 16 and 255 are done simply using that:

- $511 = 2^9 - 1$
- $16 = 2^4$
- $255 = 2^8 - 1$

and the multiplications by powers of 2 are done only with shift left operations.

3 Usage

The program consist of two verilog files: quad_oscillator and auto_oscillator.The first one is the implementation of the design presented in Figure 1. The last one is a simple state machine that makes the system oscillate automatically, you just need to keep the start input at 1.

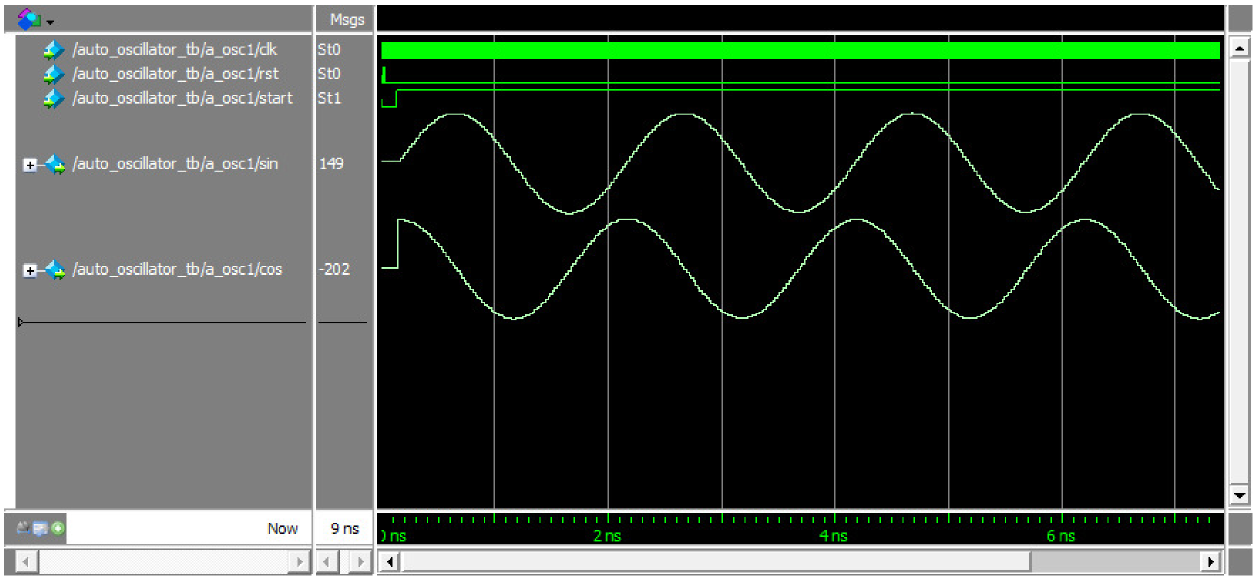
Msgs
/auto_oscillator_tb/a_osc1/clk
/auto_oscillator_tb/a_osc1/rst
/auto_oscillator_tb/a_osc1/start
/auto_oscillator_tb/a_osc1/sin
/auto_oscillator_tb/a_osc1/cos
St0
St0
St1
149
-202
Now
9 ns
0 ns
2 ns
4 ns
6 ns

Figure 2: Quadrature oscillator simulation results.

4 Simulation

The simulation done in ModelSim is shown in Figure 2.

5 FPGA test

Here is the oscillator frequency test video. A clock frequency of 48 kHz was used, so the oscillator generates a wave of approximately 48 kHz/100=480 Hz. The video shows that a 475 Hz frequency wave was generated, very close to what was expected (this small difference is due to small numerical precision errors in the calculation of the filter coefficients).

6 Synthesis result

Analysis and synthesis result (from Quartus Prime Lite Edition 18.1.0) of auto_oscillator.v file is given in Table 1.

7 License

Table 1: Analysis and synthesis result.

Family / Device	MAX 10 / 10M50DAF484C6GES
Total logic elements	97
Total registers	34
Total pins	35
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total PLLs	0

References

[MB14] Uwe Meyer-Baese. *Digital Signal Processing with Field Programmable Gate Arrays.* Springer Publishing Company, Incorporated, 4th edition, 2014.

[OS09] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing.* Prentice Hall Press, USA, 3rd edition, 2009.