EE214- Course Projects

Project Title – Arbitrary Function Generator (AFG)

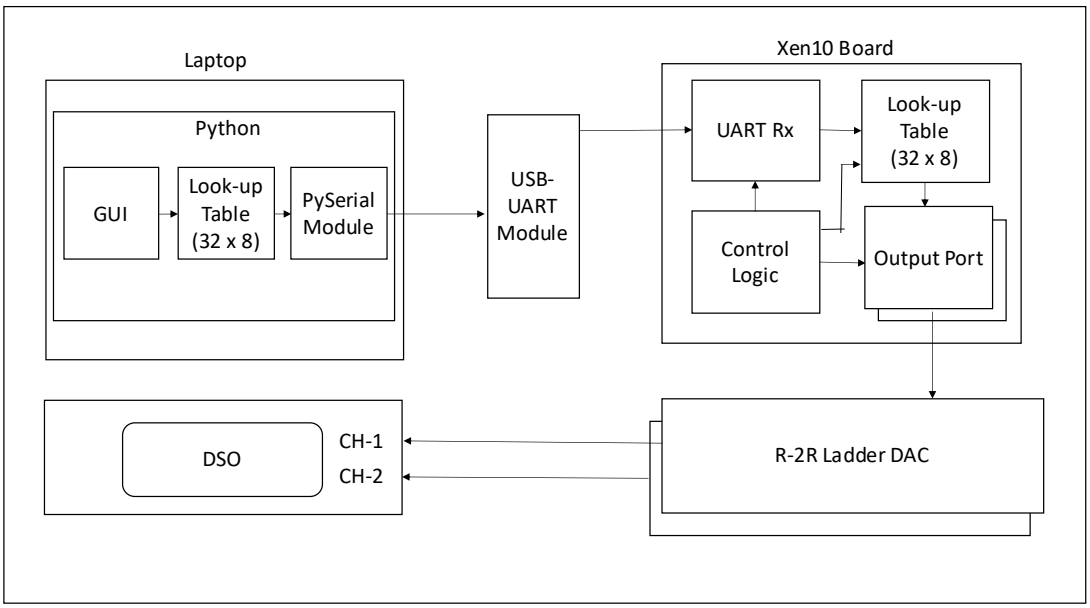
Introduction

In Electrical/ Electronics Engineering different types or arbitrary waveforms are required to analyse system performance, characterise devices, to control the system/(s) or system parameters, etc.

AFG generates different types of waveforms like Sine, Square, Ramp, Triangular or user defined type with different frequency, phase and amplitude parameters (arbitrary waveforms). You must have used or seen the AFG in the lab.

Different waveform can we generated by using Analog or digital circuits. Digital to analog convertors (DAC) are used to get analog output from digital circuits. In the analog waveforms generators by changing the value of R, L or C waveform parameters can be varied. Advantage of using Digital waveform generator is that different waveforms with different parameters can be generated without changing the pumped elements and user-friendly interface can be provided to select the required waveform parameters.

AFG block diagram connected to 2-channel DSO



Above block diagram suggested one of the possible solutions for implementing AFG on the Xen10 board. Design your system for the given AFG specifications in the abstract of the project.

Project Abstract

This project is to be build your own Arbitrary Function Generator (AFG) which will generate different waveform types with selectable frequency and amplitude using Python GUI (you can use any other software for GUI).

Our suggested way is to start with generating the Look-up table in python using the parameters selected through Python GUI. Look-up table must contain digital values of waveform points (which is our amplitude of the signal) and delay information to get required frequency. Use python serial module to send the values from generated look-up table to the hardware implemented on the FPGA.

USB to UART module can be used for the serial data transfer from the laptop USB port to the FPGA hardware. UART module gets detected as a serial COM-port in the laptop. This com-port is accessible in Python for serial data transfer.

In the FPGA, you will be implementing UART Rx (receiver) module to receive the 8-bits data serially. Once 8-bits are received, write these bits as one byte in the hardware look-up table implemented in the FPGA. Repeat this to get all bytes in the look-up table which will be stored in the memory in the FPGA. Once all bytes are received, send this received data to the output port of the FPGA. R-2R ladder type DAC (Digital-to-analog converter) connected at the output port of the FPGA which will convert digital value to analog signal. R-2R ladder is the DAC circuit consisting of few resistors to get the analog waveform. This DAC circuit needs to be implemented on the breadboard which will be interfaced to the FPGA. You need to generate appropriate control and clock signals to get the desired waveform on the DSO.

Arbitrary Waveform Generator Specifications for the project:

1) Waveform types – Sinusoidal, Square, Ramp, Triangular waveforms
2) Frequency range – 1 Hz to 1 MHz
3) Amplitude – 0 to 3.3 Vp-p
4) Simultaneous two output waveforms of same or different types of desired frequency and amplitude should be generated

Weekly Milestones:

Week-1:

1. Generate the look-up table for the Sine wave by using software or doing manual calculations.
2. Implement & test 2-2R ladder DAC circuit on the bread board with look-up table values. ***First two points are just to understand the working of AFG.***
3. Write a code to send one single byte serially from laptop to Xen10 board and display the output on 8 LEDs.
4. Write VHDL for look-up table with initial hard coded values and verify it using switches and LEDs.
5. Pen paper design of FSM for control signal logic and signal flow.
6. Write VHDL to generate variable clock signal. Use on board switches to select the frequency.
7. Make only GUI part to select waveform type and its different parameters.

Week-2:

1. Write VHDL to generate UART receiver module for receiving multiple bytes.
2. Write received data to look-up table when all bytes are received.
3. Write FSM for control logic to generate appropriate control signals.
4. Design a test output port structure.
5. Complete python code to generate look-up table and send the values serially to FPGA.
6. Combine all VHDL modules. Integrate GUI with the hardware implemented on the FPGA.

Week-3 (Final Demonstration):

- Interface the output of FPGA to R-2R ladder and show the output waveforms on the DSO for the parameters set through the GUI on the laptop.

Contact Emails for the project discussion:

1) Ms. Pragati Mali- pragatimali339@gmail.com
2) Vineesh Modi- vineeshmodi@iitb.ac.in (same on MS Teams)- WEL_RA
3) Mr. Amit Shetye- amits@ee.iitb.ac.in, i13069@iitb.ac.in (For MS Teams)