The University of Texas at Arlington


**EE 5313**

**System On Chip**

**Waveform Generator**

**Submitted to: Dr. Jason Losh**

**Submitted by: Rao Waqas Ali (1002035720)**

**& Padmanathan Kannan (1002011190)**

## Introduction:

The primary objective of this project is to develop a dual-channel waveform generator, controllable and configurable through an AXI4-lite interface from the processor subsystem. The system utilizes an MCP4822 DAC with a 2.048V internal reference, enabling the creation of two distinct waveforms within a voltage range of +/-2.5V at frequencies up to 50 kHz. The DAC outputs are processed through a reconstruction filter and level-shifting/gain stages, achieving a +/-2.5V range. The entire setup is powered by a +/-3.3V supply derived from the Blackboard, incorporating an ICL7660 charge pump.

The MCP4822 DAC, with connections to ~CS, SDI, SCK, and ~LDAC, is complemented by an op amp (TLV2372) for signal level shifting and gain. Bypassed with 0.1uF capacitors, the DAC outputs undergo further processing through an op amp, ensuring compatibility with the desired voltage range.

The project employs a -3.3V power rail generated by an ICL7660 charge pump, facilitated by 10uF capacitors on the switched energy element, input, and output rails.

## Registers:

The IP module interfaces with the processor subsystem through a memory-mapped AXI4-lite interface.

The master base address is 0x43C10000.

The register space includes:

MODE (R/W) Register (Offset 0): Configurable waveform modes for Channel A and B.

RUN Register (Offset 4): Controls the operation state of Channel A and B.

FREQ_A (R/W) Register (Offset 8): Sets the frequency of Channel A.

FREQ_B (R/W) Register (Offset 12): Sets the frequency of Channel B.

OFFSET (R/W) Register (Offset 16): Sets the offset voltage for Channel A and B.

AMPLITUDE (R/W) Register (Offset 20): Sets the amplitude for Channel A and B.

DTYCYC (R/W) Register (Offset 24): Sets the duty cycle for Channel A and B.

CYCLES (R/W) Register (Offset 28): Sets the number of cycles for Channel A and B.

## Waves:

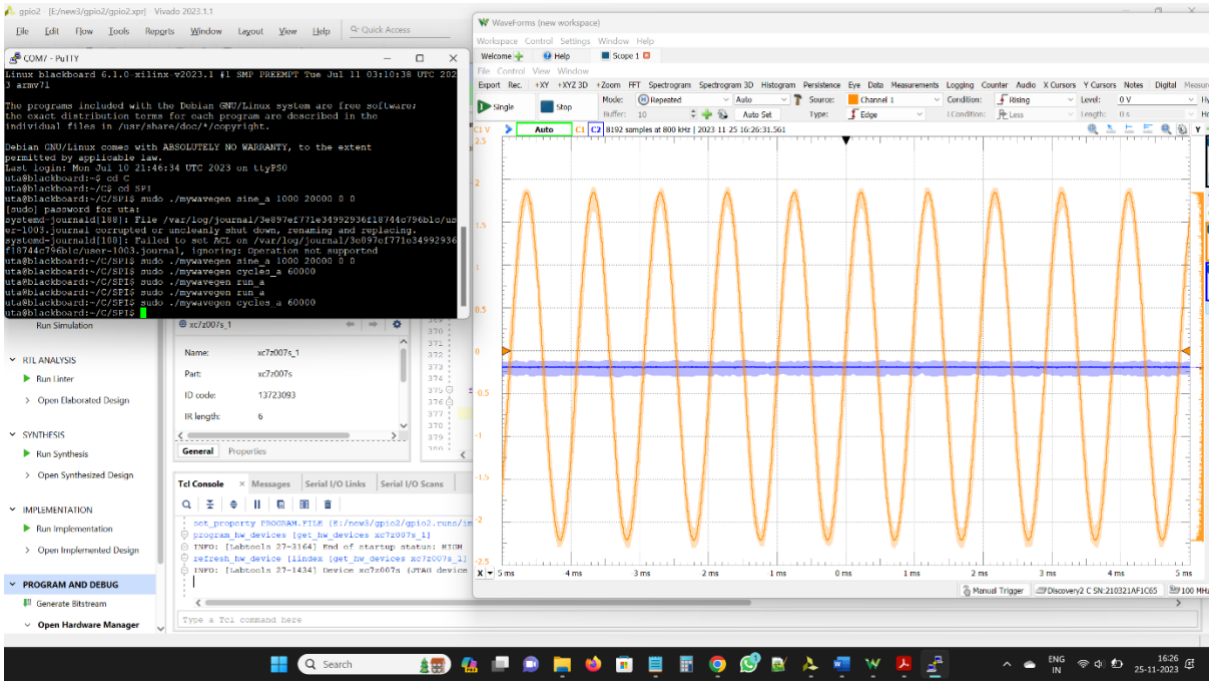
All the waveforms below were generated on putty using the wavegen.c, wavegen_ip.c and wavegen.h files present in C/SPI/ directory. The putty command nomenclature is the wave name followed by 4 arguments (except for dc wave) which are frequency, amplitude, offset and duty cycle. The duty cycle is only for square wave.

### 1. Sinusoidal:

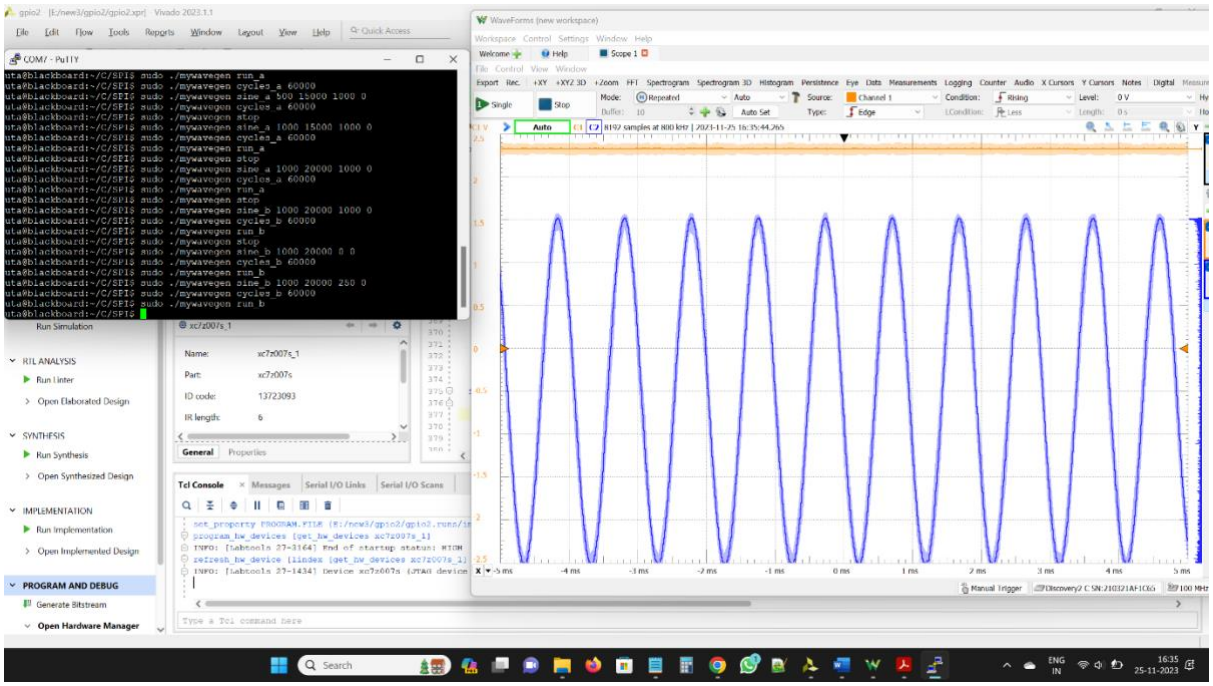
To generate sin waves, two modules are used which are phase accumulator and BRAM. Phase accumulator module with two independent accumulators, denoted as A and B. The module takes inputs such as operating mode, run signals, clock, enable, cycle counts, and frequencies for both accumulators. It calculates intermediate values based on the input frequencies and accumulates phase increments accordingly. The module operates in different modes and accumulates phase increments for the specified number of cycles. The output provides the most significant bits (MSBs) of the accumulated phase increments for each accumulator.

**BRAM** module (which have look up table .coe file, 16 bit width, 4096 depth) includes two clock inputs, clka and clkb, along with corresponding address and data ports, addra, douta, addrb, and doutb. The two 12-bit outputs of phase accumulator are the inputs of BRAM module.

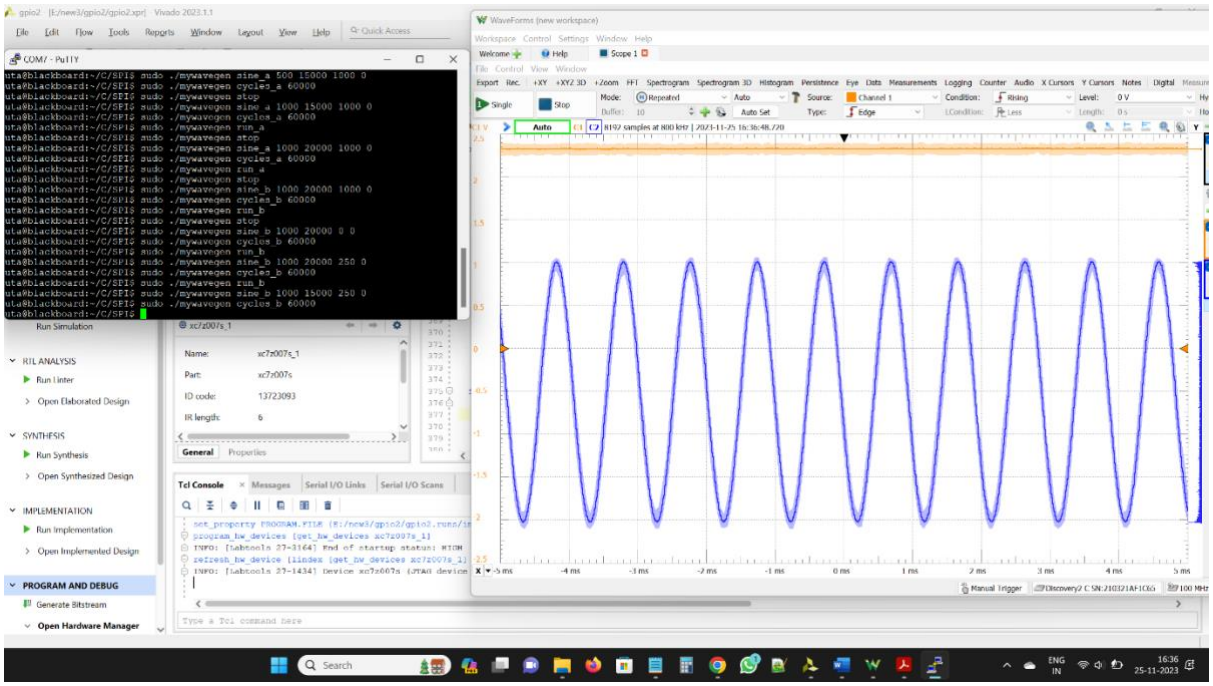
The below sine waveform was generated in the channel A with the putty command “sine_a 1000 20000 0 0”.



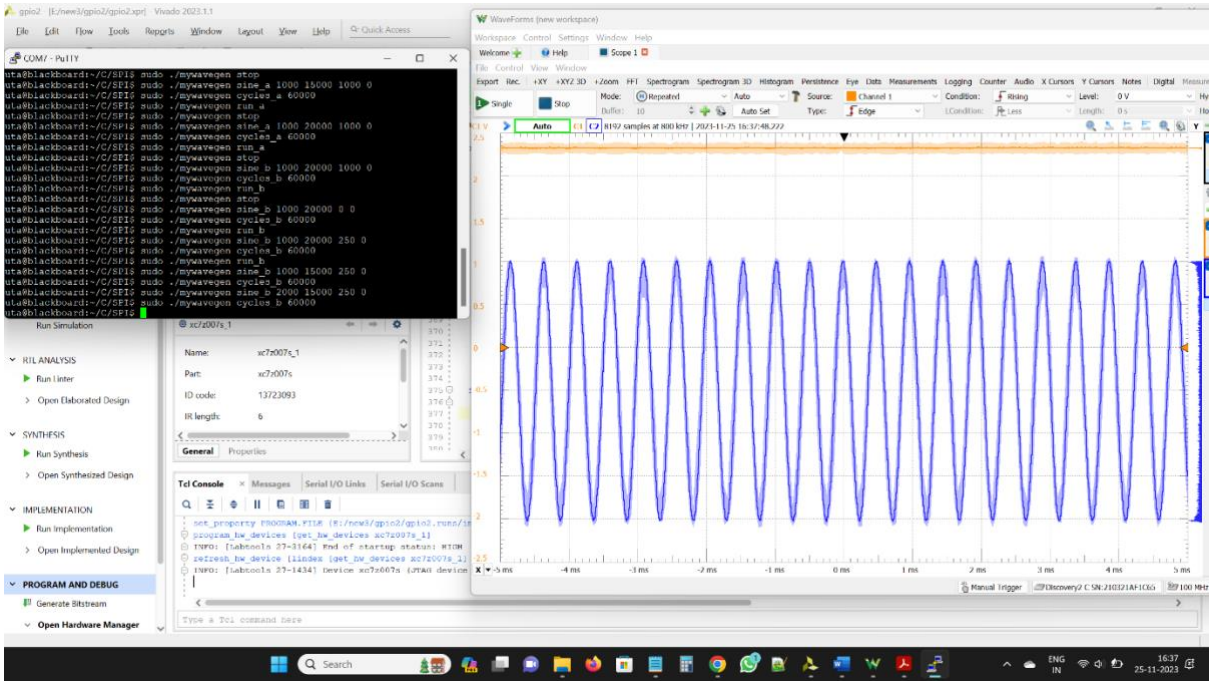
The below sine waveform was generated in the channel B with the putty command “sine_b 1000 20000 250 0”. The sinewave has an offset of 250 and the wave was shifted down.



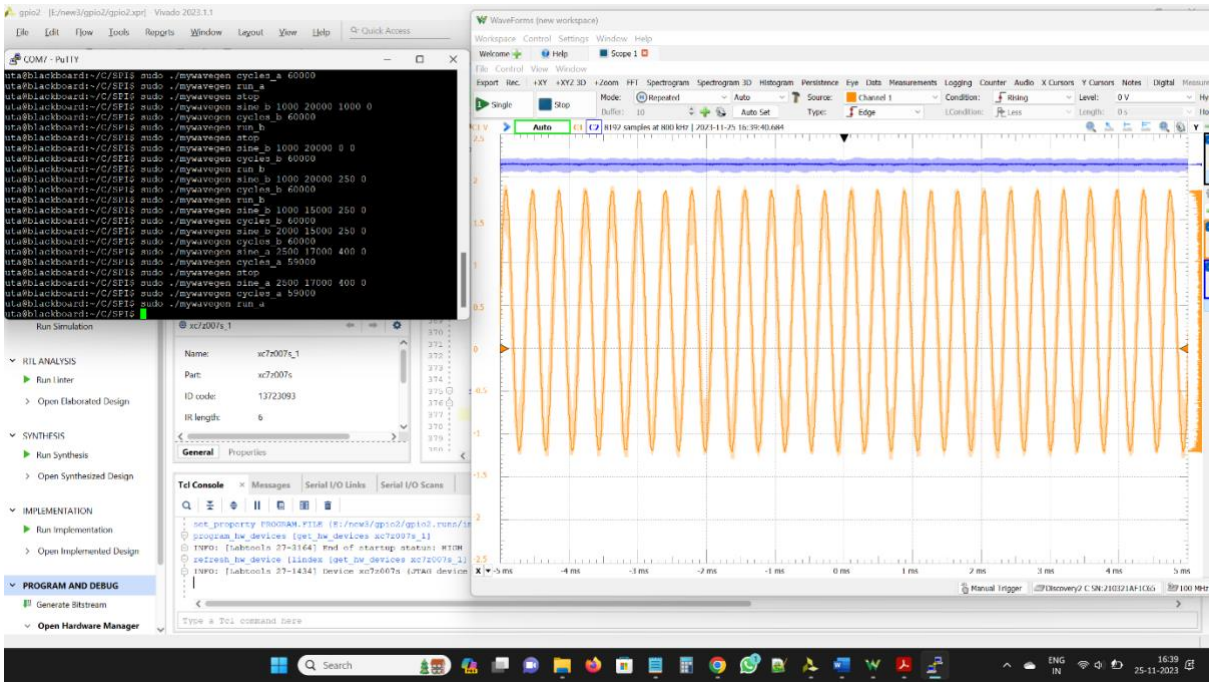
The below sine waveform was generated in the channel B with the putty command “sine_b 1000 15000 250 0”. The sinewave has an offset of 250 and the wave was shifted down.



The below sine waveform was generated in the channel B with the putty command “sine_b 2000 15000 250 0”. The sinewave has an offset of 250 and the wave was shifted down.



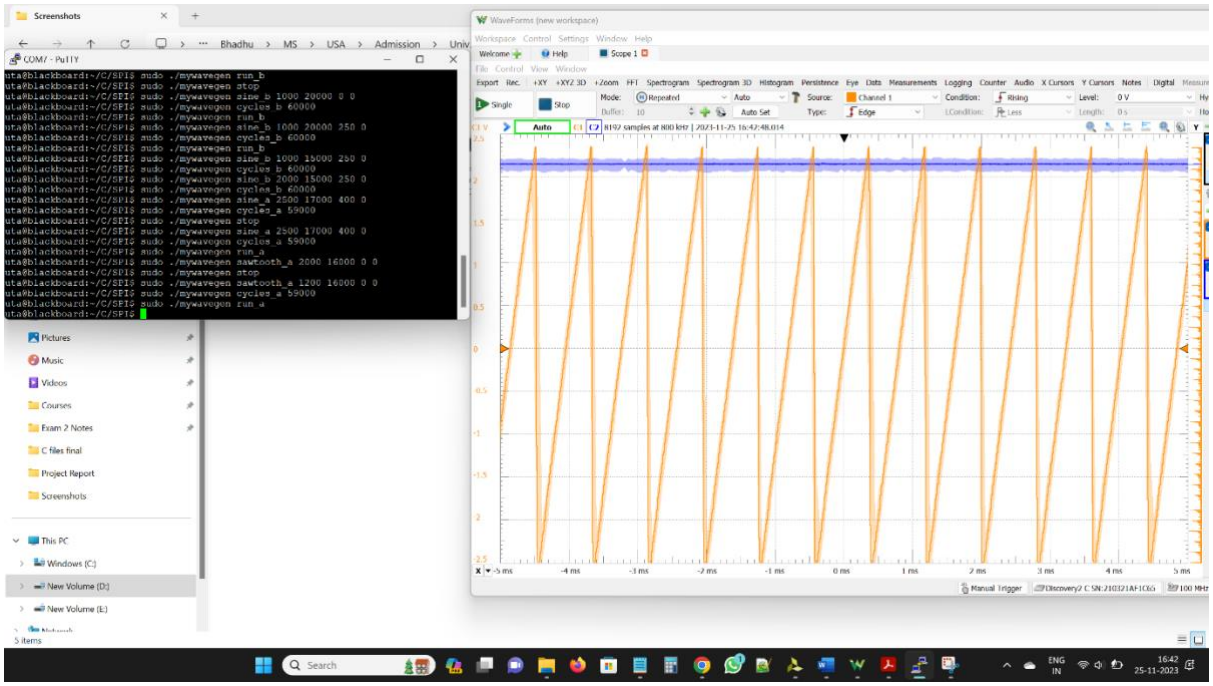
The below sine waveform was generated in the channel B with the putty command “sine_b 2000 15000 250 0”. The sinewave has an offset of 250 and the wave was shifted down.



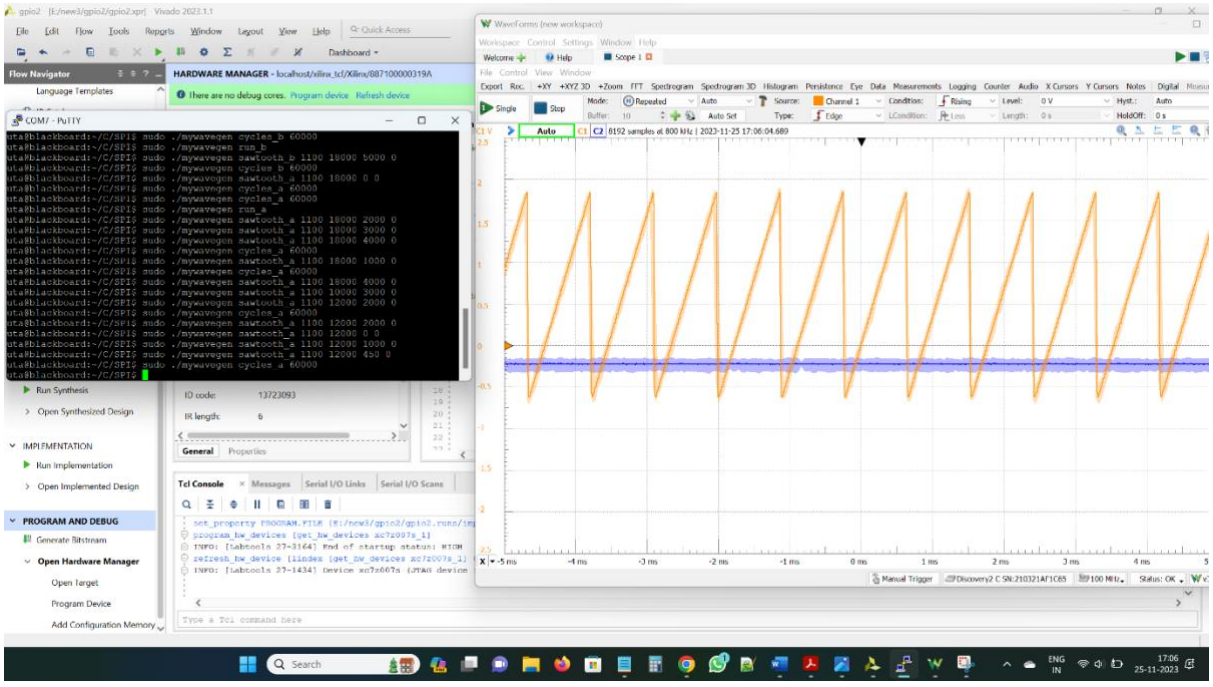

## 2. Sawtooth:

Sawtooth waveform generator module has two independent generators output denoted as countt and countt_2. The module takes inputs such as mode, run signals, clock, enable, frequencies, cycles, amplitudes, and offsets for each generator. It calculates parameters such as amplitude scaling factors, increments, and duty cycles based on the input values. The generators produce sawtooth waveforms with specified characteristics, and their operation is controlled by mode and run signals. The output consists of two signed N-bit counters representing the generated waveforms. The module utilizes counters, registers, and arithmetic operations to manage the generation process, updating the counters on each rising edge of the clock when the enable signal is active and the specified conditions are met. The resulting waveforms are influenced by the provided frequency, amplitude, and offset settings, producing sawtooth patterns with controlled parameters.

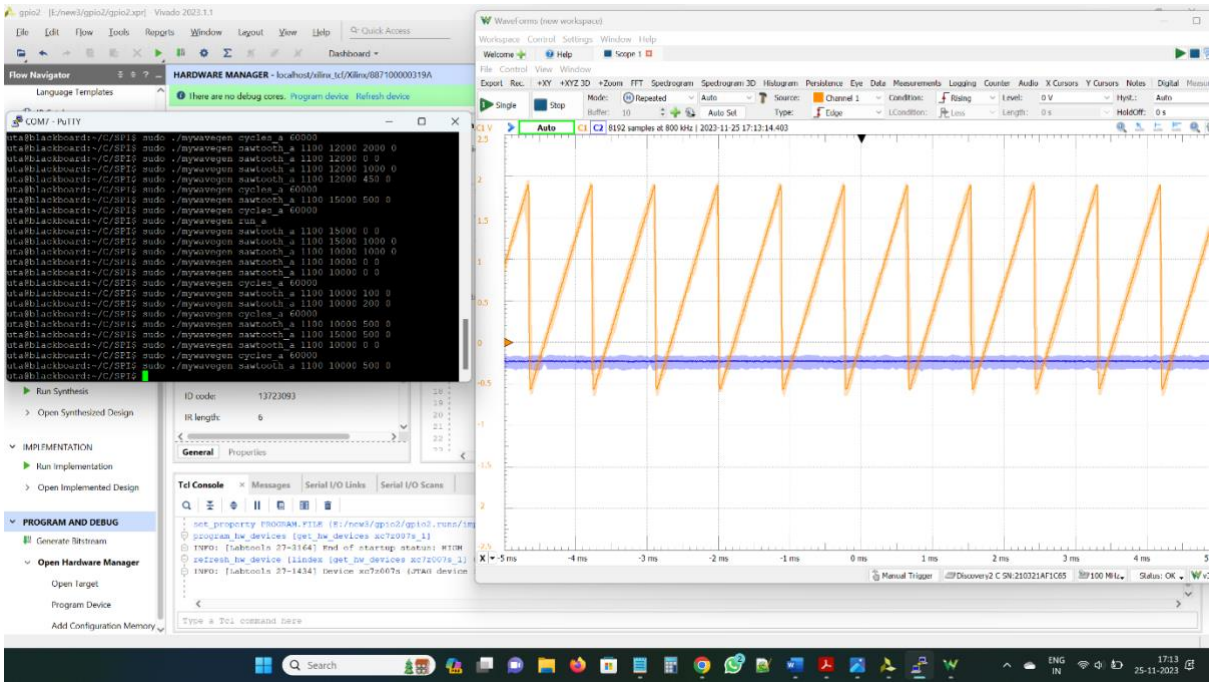
The below sawtooth waveform was generated in the channel B with the putty command “sawtooth_b 1100 18000 5000 0”. The wave has an offset of 5000.



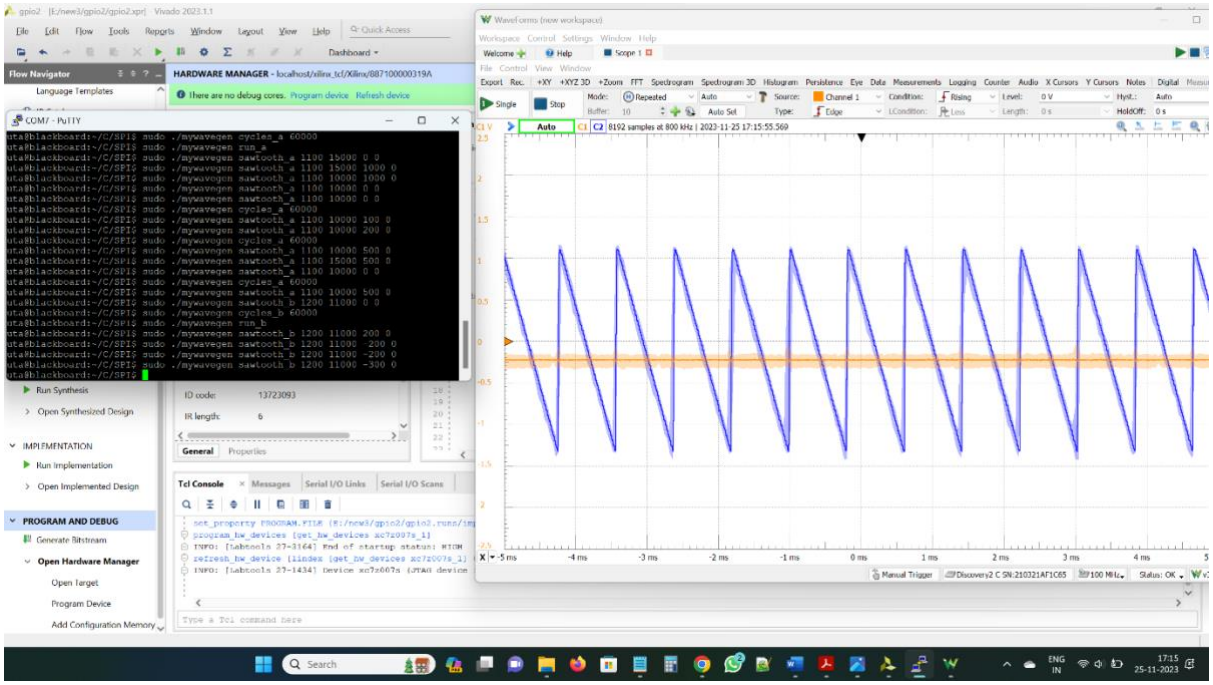
The below sawtooth waveform was generated in the channel A with the putty command “sawtooth_a 1100 12000 450 0”. The wave has an offset of 450.



The below sawtooth waveform was generated in the channel A with the putty command “sawtooth_a 1100 10000 500 0”. The wave has an offset of 500.



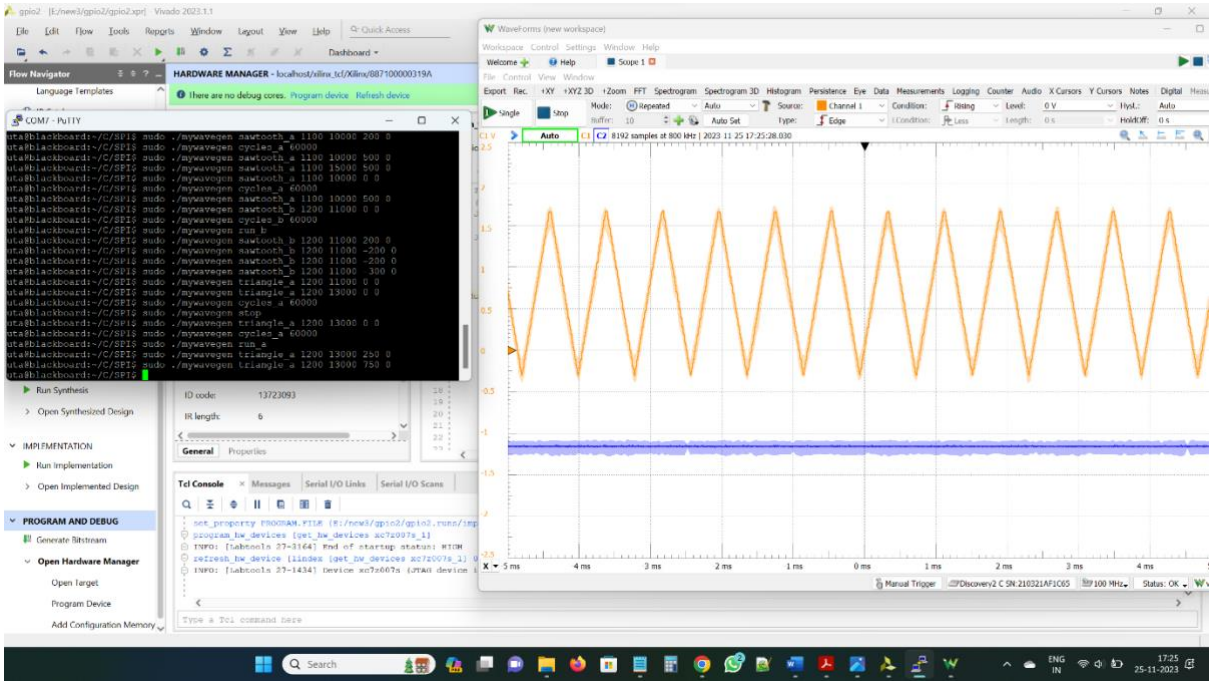
The below sawtooth waveform was generated in the channel B with the putty command “sawtooth_b 1200 11000 -300 0”. The wave has an offset of -300.



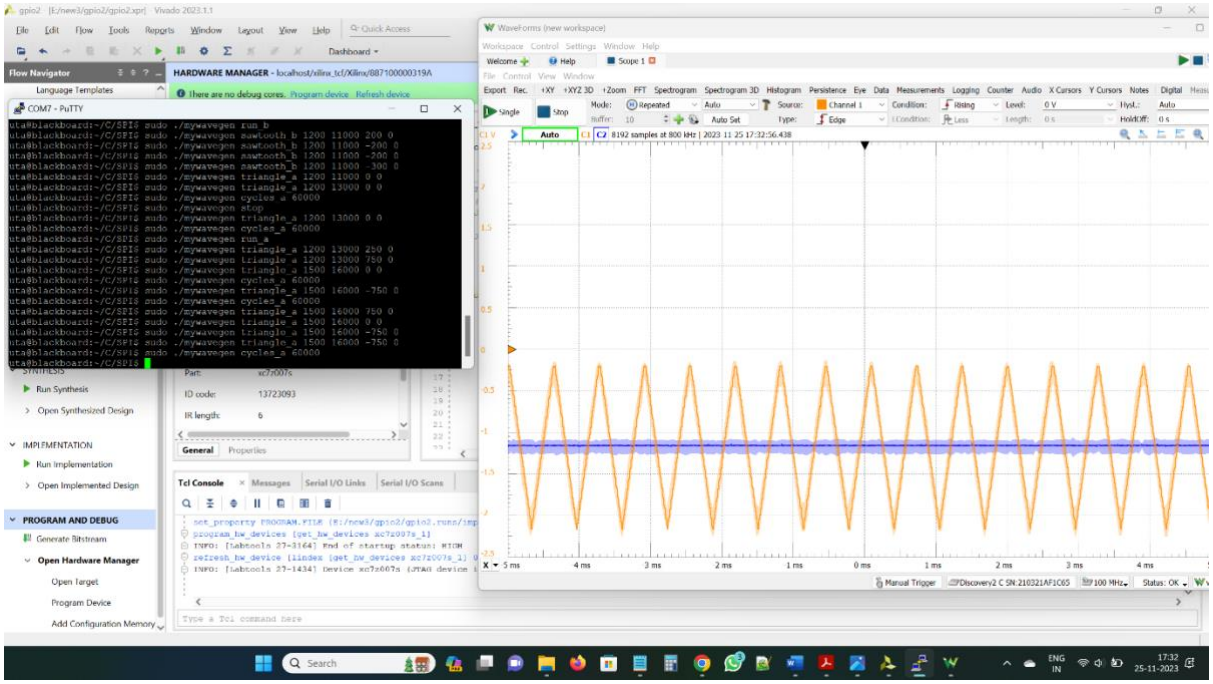
## 3. Triangle:

The triangle wave generator has two independent channels, A and B, each characterized by mode, run control, clock input, enable signal, and various waveform parameters. The module utilizes a clocked always block to generate triangular waveforms based on specified frequencies, amplitudes, cycles, and offsets. The triangular waveforms are produced by incrementing and decrementing counters within the always block, with control logic to manage direction, amplitude scaling, and waveform repetition. The triangular wave output of each channel, scaled and adjusted by amplitude and offset, is stored in countt and countt_2, respectively. The generator operates based on mode and run signals, allowing each channel to be individually controlled.

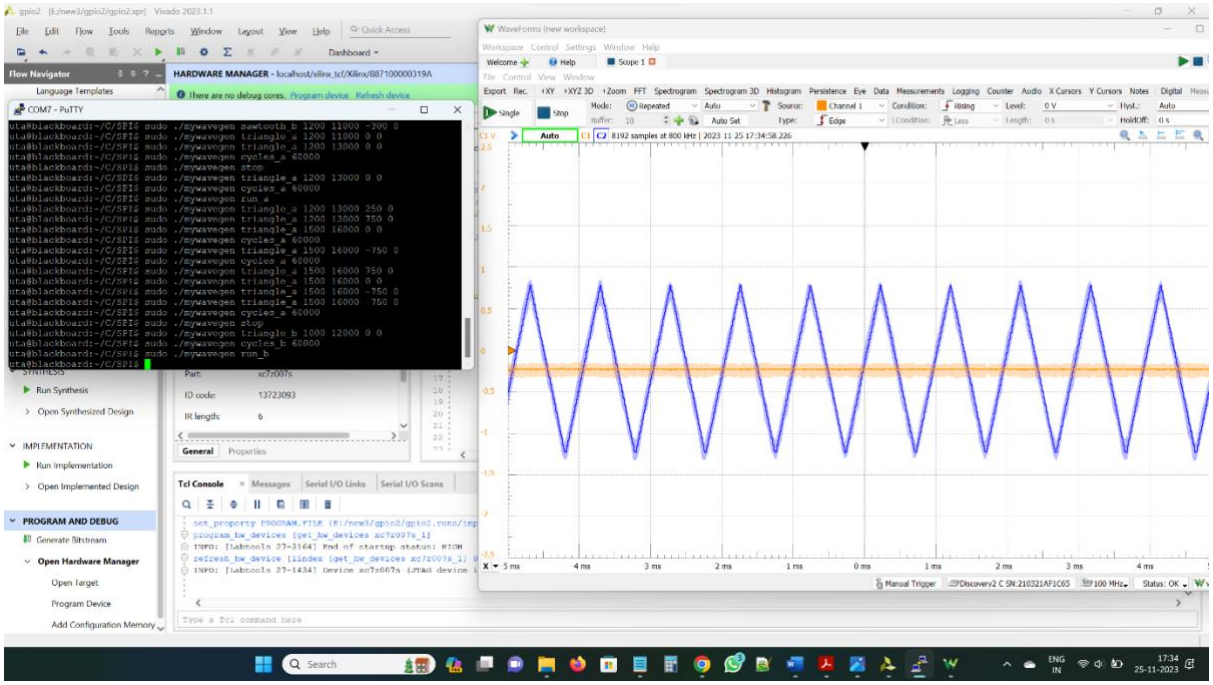
The below triangle waveform was generated in the channel B with the putty command “triangle_a 1200 13000 750 0”. The wave has an offset of 750.



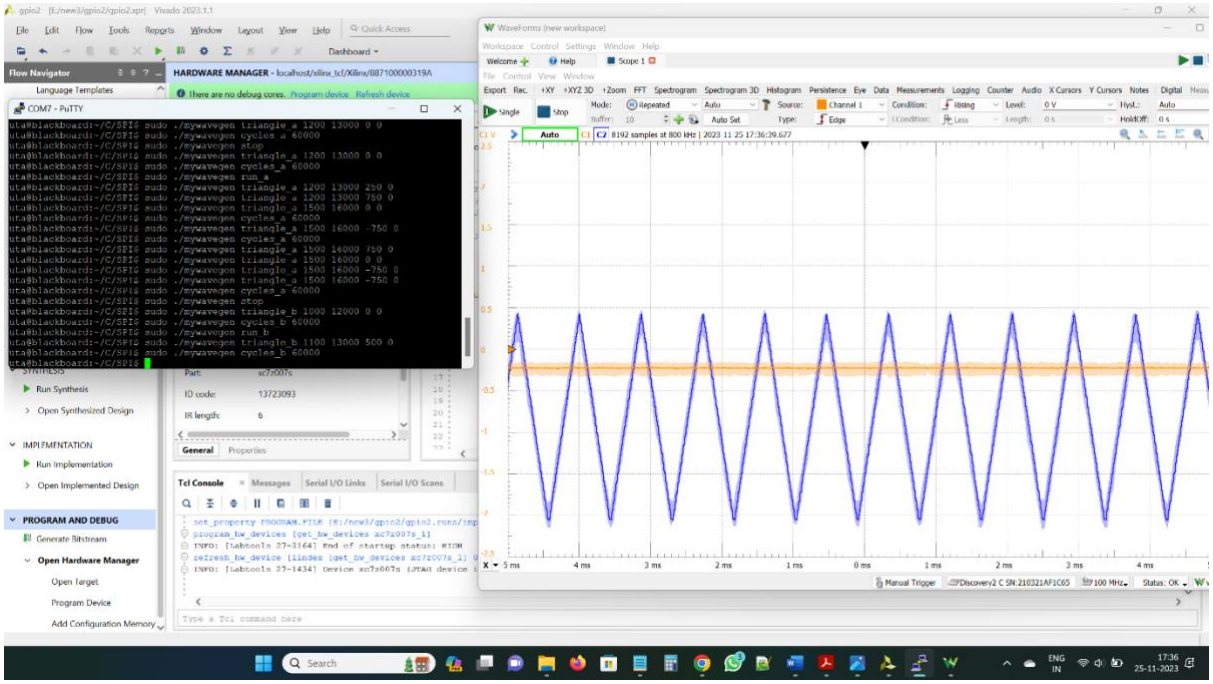
The below triangle waveform was generated in the channel B with the putty command “triangle_a 1500 16000 -750 0”. The wave has an offset of -750.



The below triangle waveform was generated in the channel B with the putty command “triangle_b 1000 12000 750 0”. The wave has an offset of 750.



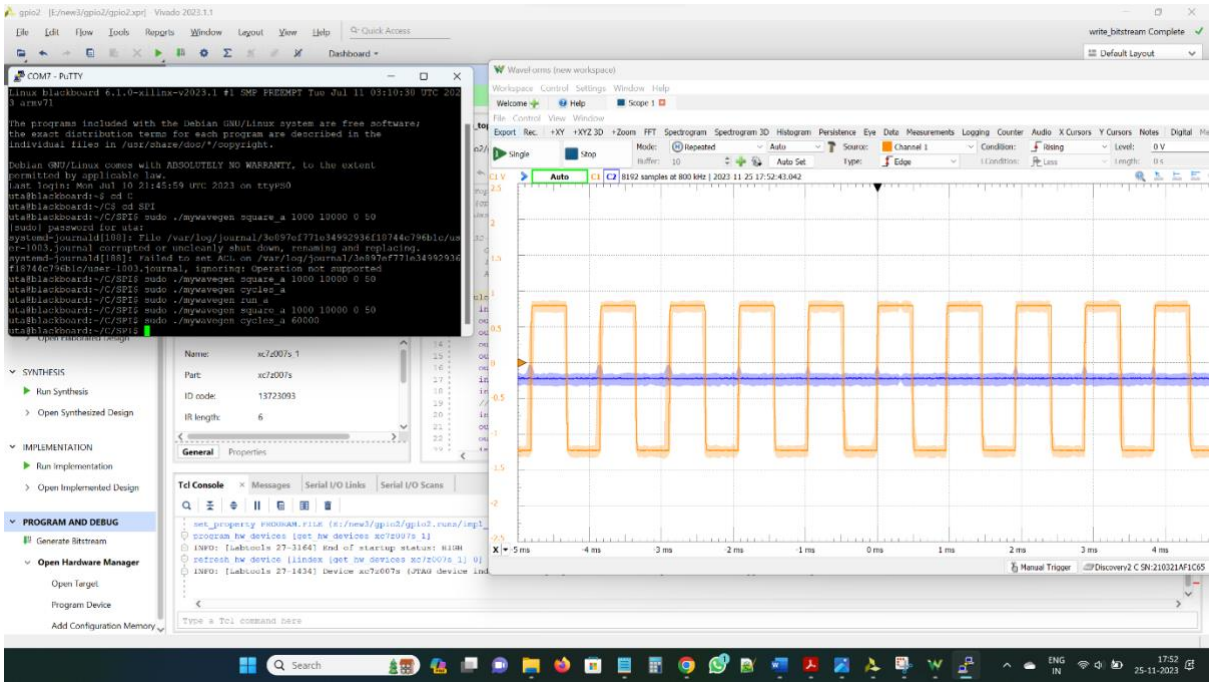
The below triangle waveform was generated in the channel B with the putty command “triangle_b 1000 12000 750 0”. The wave has an offset of 750.



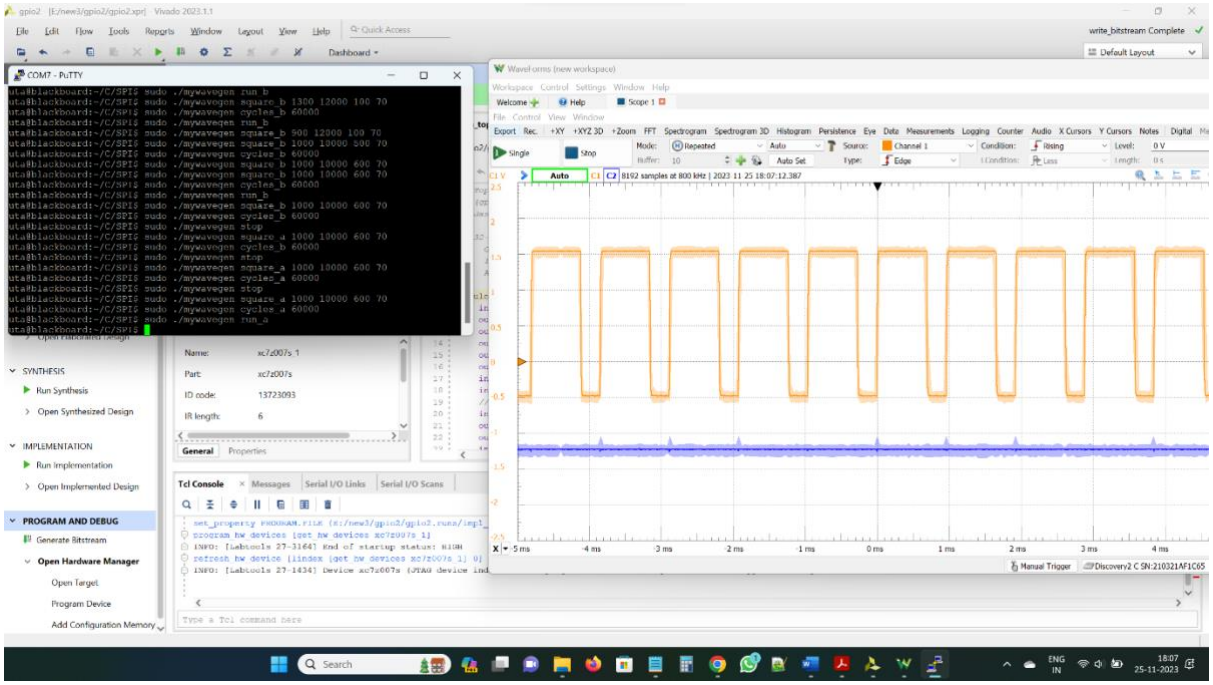
## 4. Square:

The square wave generator has two independent channels (A and B). The module generates square wave signals based on specified parameters such as mode, run control, clock input, enable signal, input frequencies, cycle durations, amplitudes, offsets, and duty cycles for each channel. The waveform parameters are used to compute the on-time duration and initialize various internal registers. The module operates on a positive clock edge when enabled and produces output signals "countt" and "countt_2" for channels A and B, respectively. The square wave generation includes amplitude scaling, offset addition, and duty cycle modulation. The code is structured to handle different operating modes and includes a counter mechanism to control the waveform generation based on the specified cycle durations.

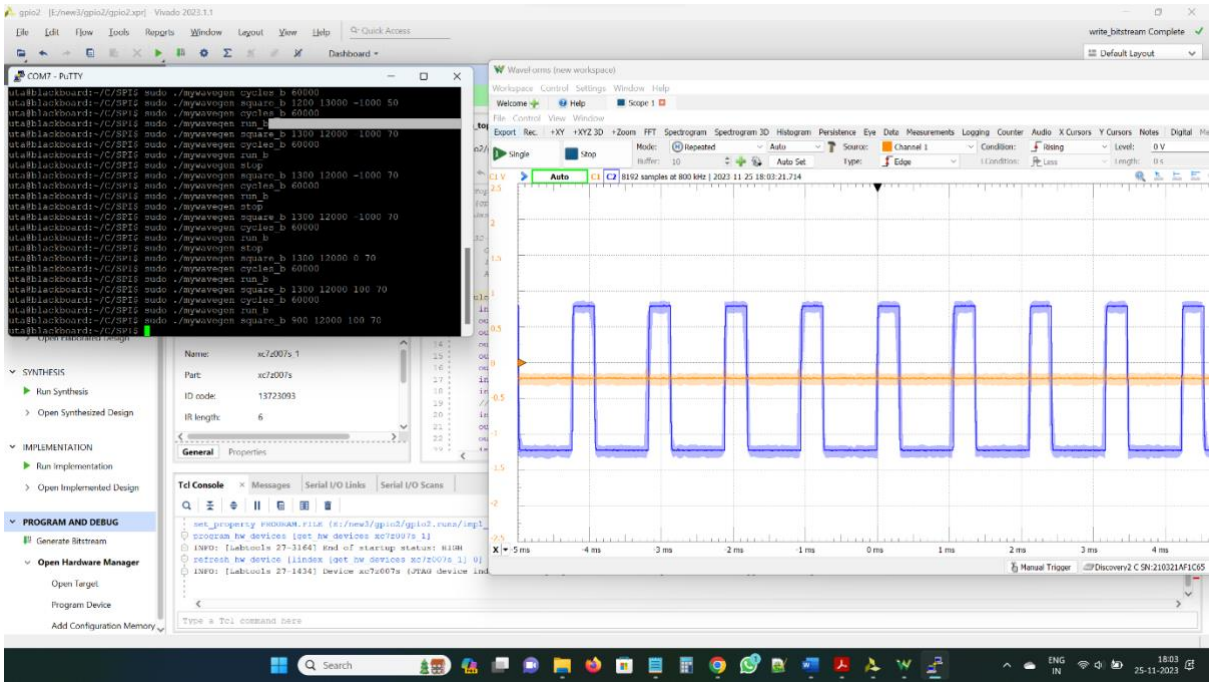
The below square waveform was generated in the channel A with the putty command “square_a 1000 10000 0 50”. The wave has an offset of 0 and a duty cycle of 50.



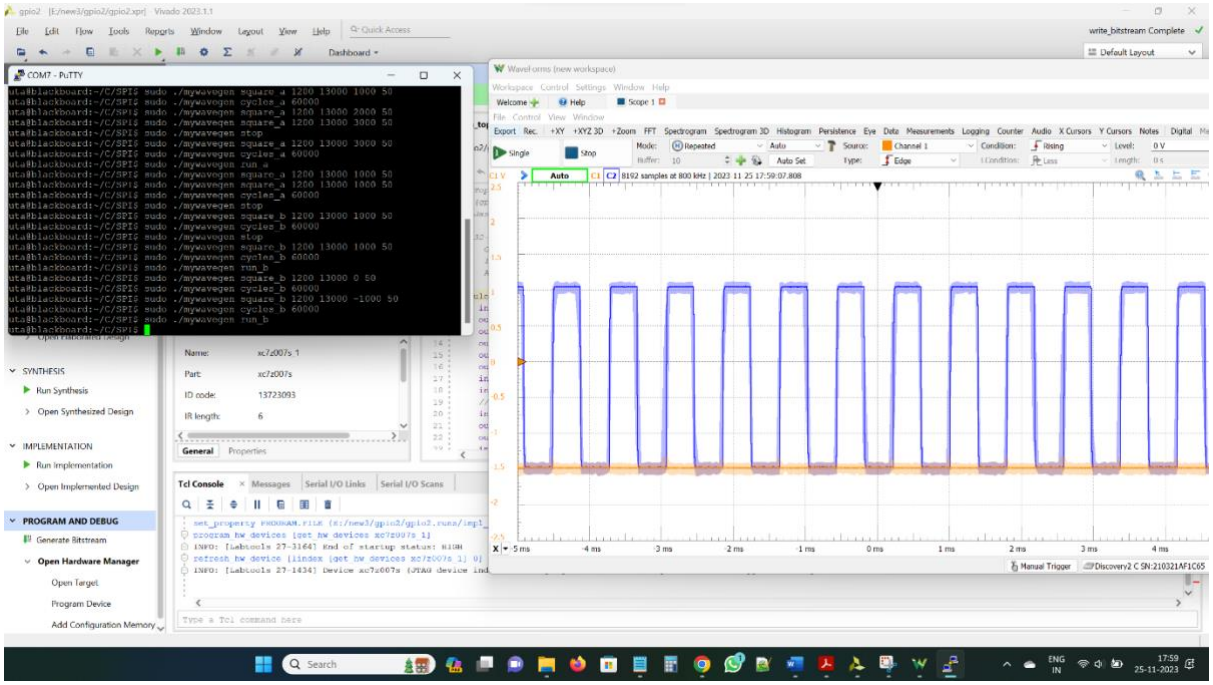
The below square waveform was generated in the channel A with the putty command “square_a 1000 10000 600 50”. The wave has an offset of 600 and a duty cycle of 50.



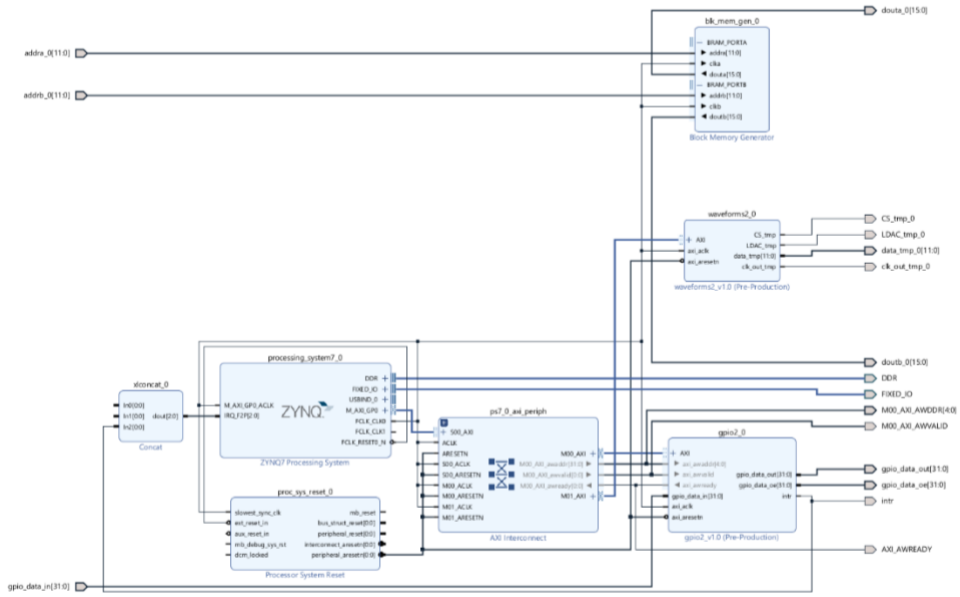
The below square waveform was generated in the channel B with the putty command “square_b 900 12000 100 70”. The wave has an offset of 600 and a duty cycle of 50.



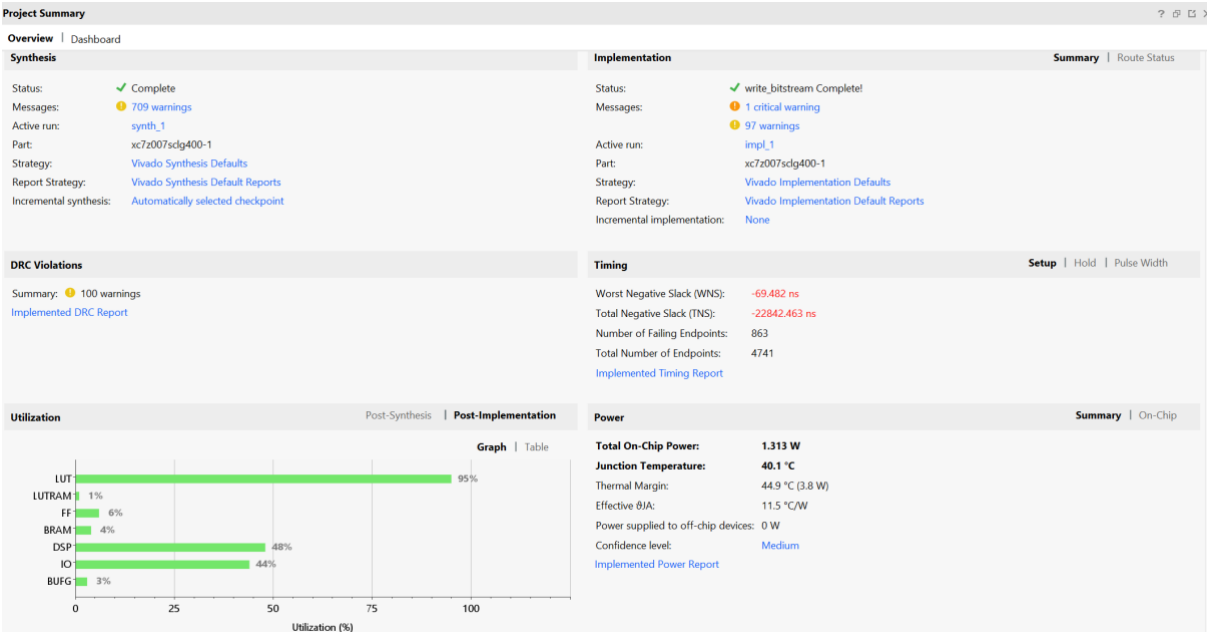
The below square waveform was generated in the channel B with the putty command “square_b 1200 13000 -1000 70”. The wave has an offset of -1000 and a duty cycle of 70.



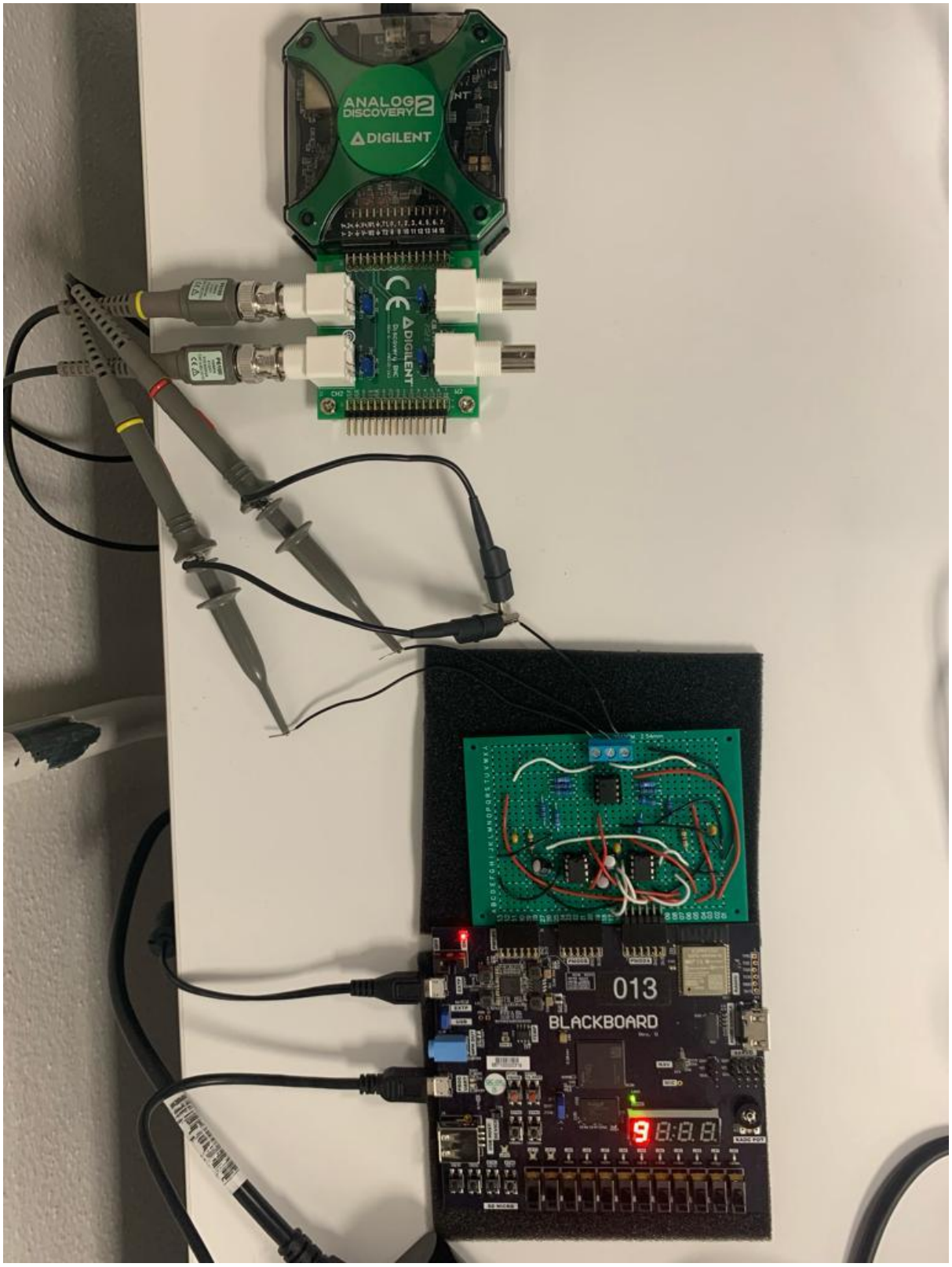
## Block Diagram:




## LUT Report:



**Setup:**



## Conclusion

The successful implementation of the dual-channel waveform generator project demonstrates a comprehensive understanding and application of System-on-Chip (SoC) design principles. This project involved the integration of hardware components, including the MCP4822 DAC, TLV2372 Op Amp, and ICL7660 Charge Pump, to create a versatile waveform generation system.

The Analog Output Processing section outlined the meticulous design considerations, including level shifting, gain adjustment, and filtering, ensuring accurate and reliable waveform generation within the specified voltage range. The incorporation of a 3.3V supply derived from the Blackboard, along with the ICL7660 charge pump for the -3.3V power rail, showcased thoughtful power management strategies.

The Wavegen IP module played a central role in providing a user-friendly interface for configuring and controlling the waveform generator. It offered flexibility in generating various waveforms, including sine, sawtooth, triangle, square, and even arbitrary waveforms. The use of block RAM for sine and arbitrary signal generation demonstrated efficient memory utilization.

The hardware description provided a comprehensive list of components, specifying quantities and values for resistors, capacitors, and other elements. The integration of these components into the printed circuit board (PCB) showcased proficiency in hardware design and layout.

The PS to IP Interface and PS Memory Mapped Control sections highlighted the seamless communication between the processor subsystem (PS) and the IP module. The memory-mapped AXI4-lite interface facilitated the configuration of waveform parameters, such as mode, run status, frequency, offset, amplitude, duty cycle, and cycles.

The PS Virtual File System (sysfs) Control section extended the project's usability by implementing a kernel module for controlling the waveform generator through sysfs. This additional interface enhances the ease of use and integration with other Linux-based systems.

In the final testing phase, the project successfully interfaced with a scope to validate the correct operation of the waveform generator. The ability to configure and control the generator using user-space commands showcased the practical application of the implemented design.

The project's completion involved a meticulous calibration process to ensure accurate amplitude and offset outputs under various conditions. This attention to detail enhances the reliability and precision of the waveform generator.

In conclusion, the project achieves its goals of implementing a dual-channel waveform generator, controlled from the processor subsystem, and serves as a valuable educational tool for courses like Wireless Communications Systems. The successful integration of hardware components, IP modules, and software interfaces reflects a well-rounded understanding of SoC design principles.