Cal Poly Pomona

College of Engineering

California Polytechnic State University, Pomona

ECE3300L

Experiment #8

GROUP K
Hoang, Dalton - 016062800
Siu, Andy - 016205137

August 12th, 2025

Introduction:

This lab focuses on designing and implementing an RGB LED controller on the Nexys A7 FPGA using Pulse Width Modulation (PWM) to achieve adjustable brightness and color mixing. The PWM core module converts stored resolution and duty cycle values into output signals for each color channel. At the same time, the RGB LED driver ensures correct signal polarity for the board's active-low LEDs.

Design:

Clock_divider_fixed.v: Generates two slower clock signals from the FPGA's high-frequency system clock, one at around 1 kHz for button debouncing and FSM logic, and another at around 20 kHz for flicker-free PWM. It uses counters to toggle each output clock after a calculated number of input clock cycles.

Debounce_onepulse.v: Cleans up button inputs by implementing a debounce mechanism, ensuring stable detection of presses. It also produces a one-cycle pulse for each valid press, regardless of how long the button is held. This prevents multiple unintended triggers from a single press.

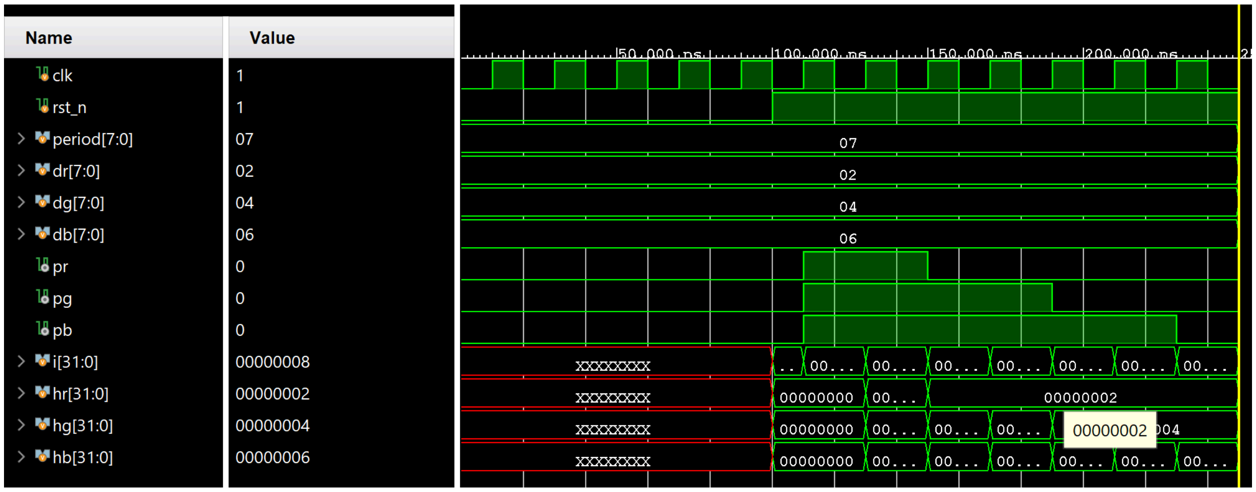
Load_fsm.v: Implements a state machine that cycles through four "slots" each time the load button is pressed. These slots correspond to loading the PWM period, and the red, green, and blue duty cycles.

Pwm_core.v: Generates independent PWM signals for the red, green, and blue channels using the stored period and duty cycle values. The module counts clock cycles and compares them to the duty cycle to decide whether each output is high or low.

Rgb_led_driver.v: Interfaces the PWM outputs with the physical RGB LEDs on the Nexys A7 board. Since the LEDs are active-low, it inverts the PWM signals if necessary. The parameter ACTIVE_LOW allows the same driver to be used for boards with different LED polarities.

Top_lab8.v: Integrates all the submodules into a single top-level design for the RGB LED PWM controller. It handles clock generation, button debouncing, slot selection, register storage for resolution and duty cycles, and signal crossing between clock domains. Finally, it connects the PWM outputs to the RGB LED driver for color control. Changes made included changing btnc_n and the rgb_led_driver to active low and using "slot" instead of "wr_reg", "wr_r", "wr_g", "wr_b" to make the inputs visible before loading them by pressing btnr.

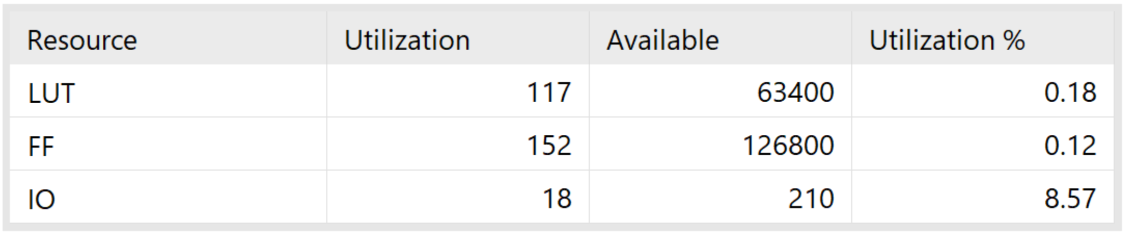
Simulation:



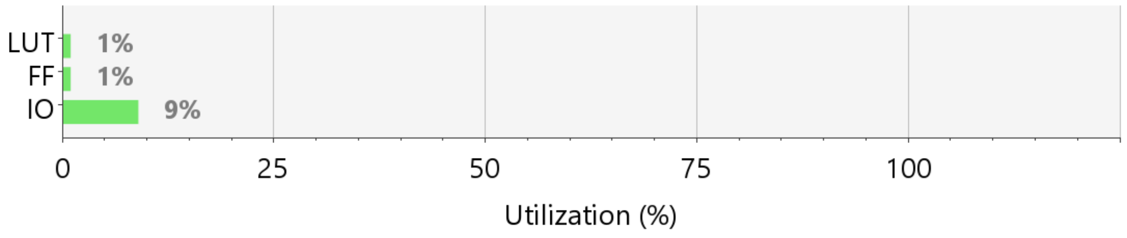
Name	Value
clk	1
rst_n	1
period[7:0]	07
dr[7:0]	02
dg[7:0]	04
db[7:0]	06
pr	0
pg	0
pb	0
i[31:0]	00000008
hr[31:0]	00000002
hg[31:0]	00000004
hb[31:0]	00000006

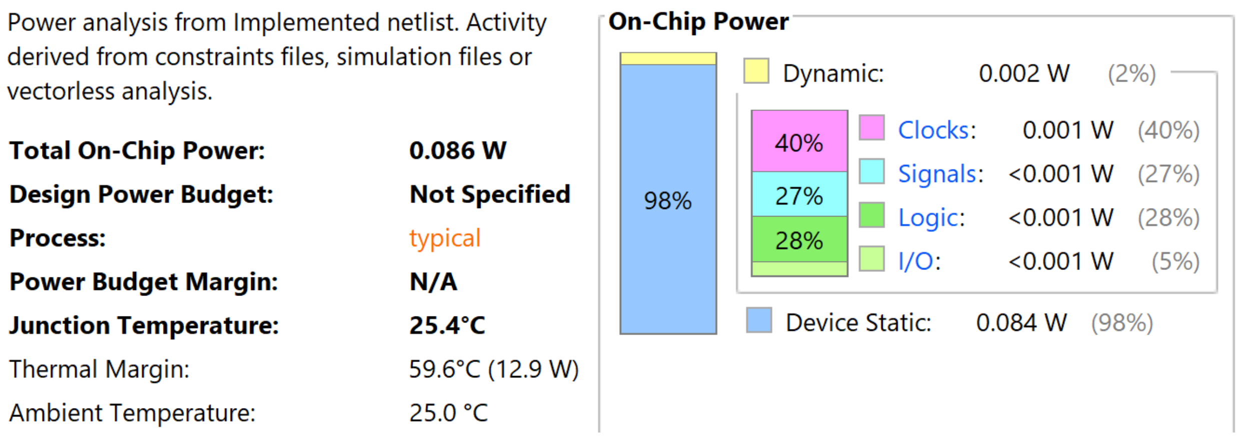
The waveform looks correct for an 8-step PWM. After reset is released, the counter runs a cycle of 8 clocks with period=0x07. The duty inputs are dr=0x02, dg=0x04, and db=0x06, and you can see the three PWM outputs (pr, pg, pb) go high at the start of each period and fall at counts 2, 4, and 6 respectively—so pr has the shortest high time (2/8), pg is mid (4/8), and pb is longest (6/8). The buses hr, hg, and hb capture those thresholds (2, 4, 6), and the wide bus shows valid values once reset is switched. The ordering of the falling edges matches the programmed duties, confirming the comparator/accumulator logic and period handling are behaving as intended.

Implementation:



Resource	Utilization	Available	Utilization %
LUT	117	63400	0.18
FF	152	126800	0.12
IO	18	210	8.57





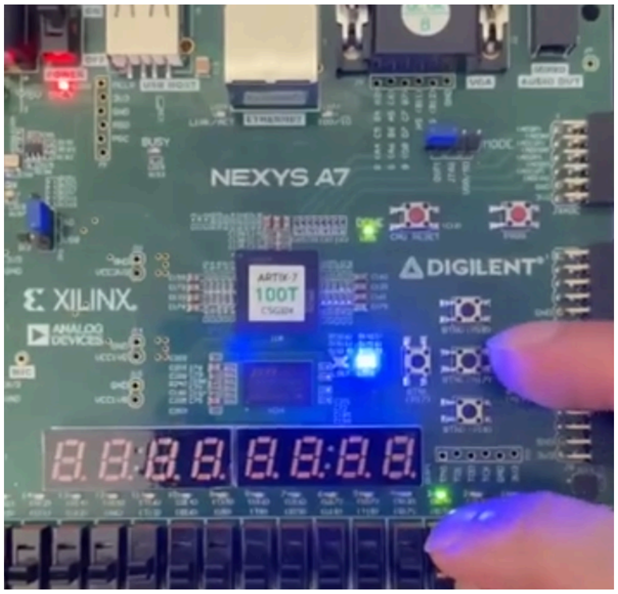
Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

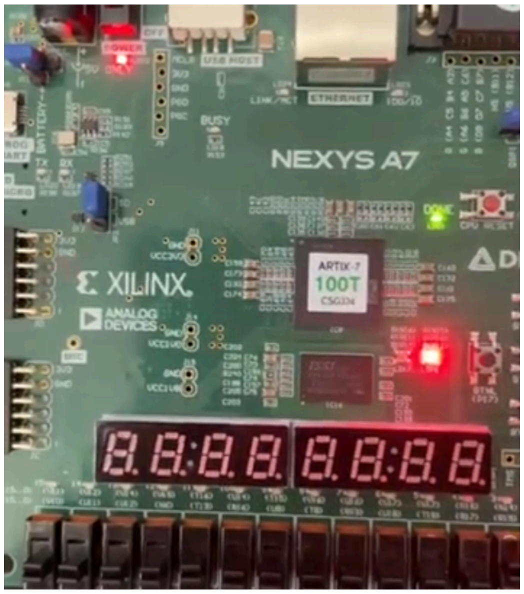
Total On-Chip Power:	**0.086 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**25.4°C**
Thermal Margin:	59.6°C (12.9 W)
Ambient Temperature:	25.0 °C

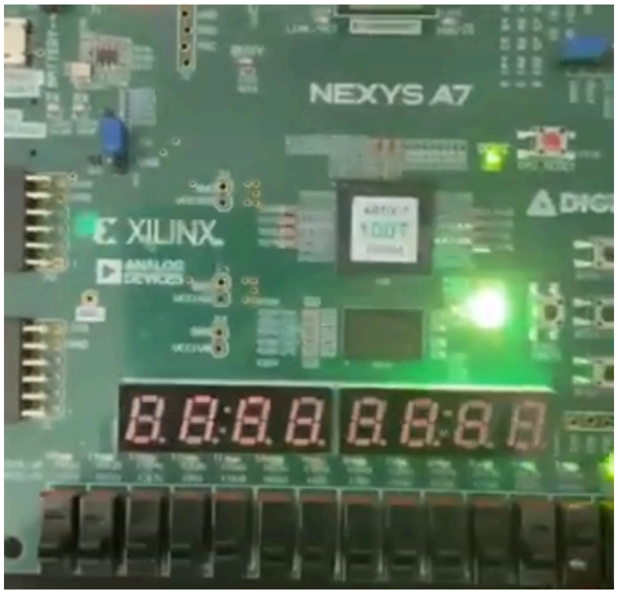
On-Chip Power

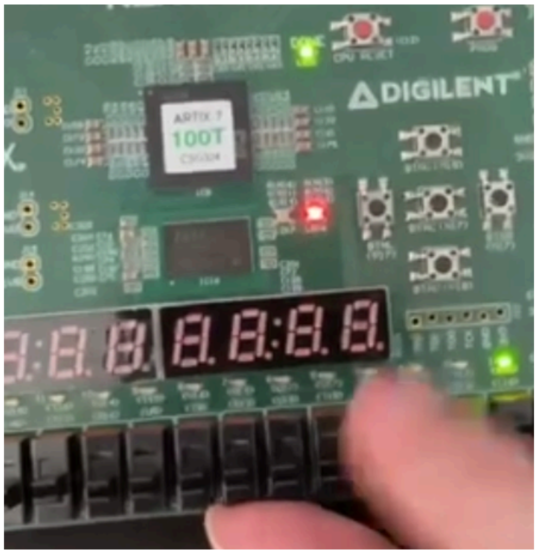
🟨	Dynamic:	0.002 W	(2%)
🟪	Clocks:	0.001 W	(40%)
🟦	Signals:	<0.001 W	(27%)
🟩	Logic:	<0.001 W	(28%)
🟩	I/O:	<0.001 W	(5%)
🟦	Device Static:	0.084 W	(98%)

Board Pictures:









Video Link: https://youtube.com/shorts/9r8k8MAB8WU

Contributions:

Andy Siu: 50% source files, testbench files, demo, report

Dalton Hoang: 50% source files, testbench files, simulation, report, implementation