Nexys4IO Peripheral and PWM Analyzer Integration

ECE 544 Project 1

Author: Phil Nevins

Date: February 2025

https://github.com/Dawgburt/ECE544-Project1

# **1. Introduction**

This report details the design, implementation, and testing of a MicroBlaze-based embedded system that interfaces with the Nexys4IO peripheral and an external Digilent PWM Analyzer. The system operates on a Nexys A7 FPGA board and integrates hardware and software components to control RGB LEDs using pulse-width modulation (PWM) signals. The project involves real-time data acquisition from the PWM Analyzer and visualization on the 7-segment display.

# **2. System Overview**

The system consists of the following components:

**MicroBlaze Soft Processor**: Executes embedded C code and manages peripheral interactions.

**Nexys4IO Peripheral**: Controls RGB LEDs, switches, and buttons.

**AXI Timer**: Generates a PWM clock signal.

**Interrupt Controller (XIntc)**: Manages periodic updates via the Fixed Interval Timer (FIT).

**Digilent PWM Analyzer**: Reads duty cycle values of externally generated PWM signals.

**GPIO Interface**: Enables communication between MicroBlaze and peripheral components.

# **3. Application Code Description**

The main application, implemented in test\_nexys4ioV2.c, performs the following operations

### **System Initialization**

Configures and initializes the AXI Timer, GPIO, and PWM Analyzer.

Enables interrupts for periodic updates.

### **Self-Test and Debugging**

Runs initial tests to verify LED, 7-segment display, and PWM output functionality.

### **Main Loop**

Reads button and switch states.

Adjusts the duty cycle of RGB LEDs based on button presses.

Updates the RGB2 LED based on measured duty cycle values from the PWM Analyzer.

Displays PWM values on the 7-segment display.

### **Fixed Interval Timer (FIT) Handler**

Periodically reads button and switch states.

Toggles decimal point on the 7-segment display to indicate system activity.

Reads PWM duty cycle values and updates the display.

# **4. Problems Encountered and Solutions**

### **4.1 AXI Register Reads Returning Zero**

**Problem**: Initial attempts to read from the PWM Analyzer registers returned zero values. **Solution**: Verified correct base addresses from the Vivado Address Map and ensured that the PWM Analyzer was enabled before reading.

### **4.2 Incorrect PWM Values Displayed on 7-Segment Display**

**Problem**: PWM duty cycle values displayed on the 7-segment display were inconsistent. **Solution**: Implemented a scaling factor to convert the 8-bit (0–255) duty cycle values to a percentage format (0–99%).

### **4.3 GPIO Configuration Issues**

**Problem**: GPIO readback values did not match the written values.   
**Solution**: Forced the GPIO to output mode using XGpio\_WriteReg() and confirmed correct configuration via debug prints. GPIO was not initialized properly as output, 32-bits.

### **4.4 RGB2 LED Constantly “On A Little”**

**Problem:** The RGB2 LED stays continuously lit even after resetting the PWM duty cycle (DC) variables for each color.

**Solution:** The initial approach involved reading the PWM duty cycle values on reset and subtracting them from each color in the RGB2 write function. However, this caused incorrect LED brightness levels, leading to display issues. During the demonstration, the actual issue was identified: the pwmColorDC values were being initialized and reset to 0 instead of 1. Since the PWM reader detects rising edges, a 0-value resulted in no rising edge detection, causing unexpected behavior.

# **5. Suggested Improvements**

**Increase PWM Resolution**: Modify PWM control to support a higher bit-depth for smoother transitions.

**Enhance User Interface**: Implement an OLED or LCD display for better visualization of PWM values.

**Implement Non-Blocking I/O**: Modify the main loop to use event-driven programming instead of polling.

**Add UART Debugging Support**: Integrate a serial interface for remote debugging and logging.

**Optimize Code Structure**: Separate core functionality into modular files to improve maintainability.

# **6.** Conclusion

This project successfully demonstrated the integration of an external Digilent PWM Analyzer with the Nexys4IO peripheral on a MicroBlaze-based FPGA system. The application code efficiently manages real-time PWM measurements, user interactions, and visual outputs. Despite initial challenges with hardware communication and register access, troubleshooting techniques such as debugging prints and address verification helped resolve the issues. The project provides a solid foundation for further enhancements, including improved user interfaces and increased system efficiency.

# 7. Appendix

## 7.1 Embsys Schematic

### 7.1a Elaboration Schematic

A screenshot of a computer

AI-generated content may be incorrect.

### 7.1b Block Design Schematic

A diagram of a computer

AI-generated content may be incorrect.

7.2 Implemented Design  
A screen shot of a computer

AI-generated content may be incorrect.

## 7.3 Included Libraries

The following libraries were utilized in the development of this project:

### 7.3a Standard Libraries

#include <stdio.h> – Standard Input/Output library.

#include <stdlib.h> – Standard library for memory management, conversion, etc.

### 7.3b Xilinx Platform-Specific Libraries

#include "platform.h" – Platform initialization and management.

#include "xil\_printf.h" – Xilinx-specific formatted printing functions.

#include "xparameters.h" – Hardware parameters and configuration definitions.

#include "xstatus.h" – Xilinx standard return status codes.

#include "microblaze\_sleep.h" – MicroBlaze sleep functions.

### 7.3c AXI and I/O Control Libraries

#include "xil\_io.h" – Functions for reading and writing AXI registers.

#include "xgpio.h" – Xilinx GPIO library for digital I/O control.

### 7.3d Timer and Interrupt Handling

#include "xtmrctr.h" – Xilinx AXI Timer driver.

#include "xintc.h" – Xilinx Interrupt Controller driver.

### 7.3e Peripheral and Custom Driver Libraries

#include "nexys4IO.h" – Custom driver for Nexys4IO peripheral.

#include "PWM\_Analyzer.h" – Custom driver for the PWM Analyzer.

### 7.3f Xilinx Utility Libraries

#include "xil\_types.h" – Xilinx-defined types for portability.

#include "xil\_assert.h" – Xilinx assertion utilities.

# 8. References

Digilent Nexys A7 Reference Manual

Xilinx MicroBlaze User Guide

Digilent PWM Analyzer Documentation  
RK IP Documentation

ChatGPT – Used to help debug issues and format design report