ECE – 544

PORTLAND STATE UNIVERSITY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Embedded System Design

Project-1

Deepen M Parmar
parmar@pdx.edu

Date: 24th April 2020

- ## Introduction

This project makes use of the Pulse Width Modulation capability to set the intensity of the RGB led on FPGA board. In this project we are passing the Hue Saturation and Value as the input with the pushbuttons and the rotary encoder and we will get the equivalent Duty cycle of the Red, Green and Blue signal.
HSV is the most common cylindrical-coordinate representation of points in an RGB color model. It rearranges the geometry of RGB in an attempt to be more intuitive and perceptually relevant than the cartesian representation.
This project also provides us with the experience of the Pulse Width Detection which is implemented in the Hardware and the Software side

- ## Functional Specifications

In this project HSV values are the input to the system which are given by the Pushbuttons and the rotary encoder and this value are displayed on the PMOD OLEDRGB. UP and DOWN pushbuttons are used to alter the values of brightness (Value), RIGHT and LEFT buttons are used to alter the values of the Saturation and Encoder is used to alter the value of Hue.
PMOD OLEDRGB also displays the color wheel which the color obtained by passing the equivalent value of the HSV. This values of HSV are also given to the RGB LED on the board in the form of the duty cycle which also glows the equivalent color obtained from the HSV values. This project also Calculates the Duty cycle of the RGB LED which are displayed on the Seven Segment and Switches selects which channel is getting displayed i.e. Duty cycle of Red Green or Blue. The calculation of the Duty cycle is implemented in Hardware and Software.

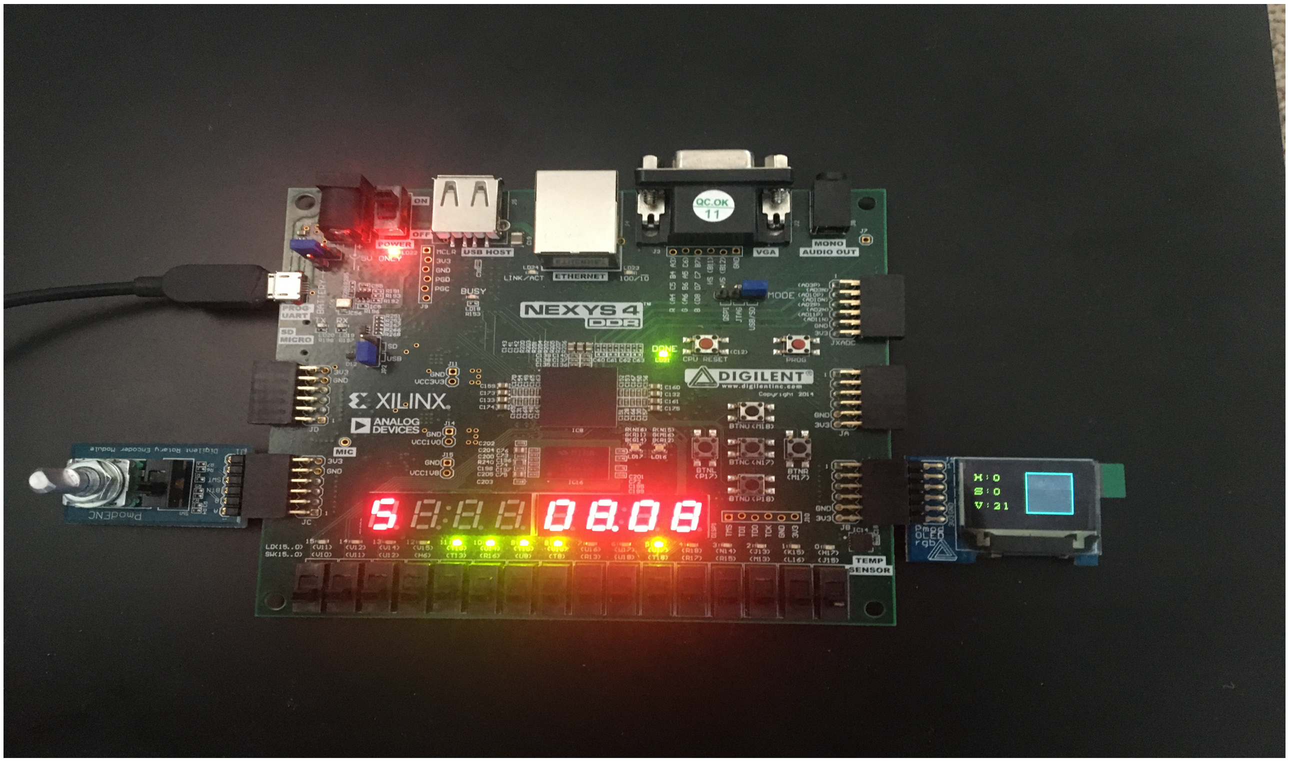

Figure-1

- ## Pulse Width Modulation and Detection

PWM Modulation is the process which uses the width of the pulse to convey the information in a digital signal. It is widely used to control the control the brightness of the LED and controlling the direction of the servo motor. It works well with the digital controls because of its ON/OFF nature which is used to set the needed duty cycle.
The term duty cycle describes the ON time to the regular period of time

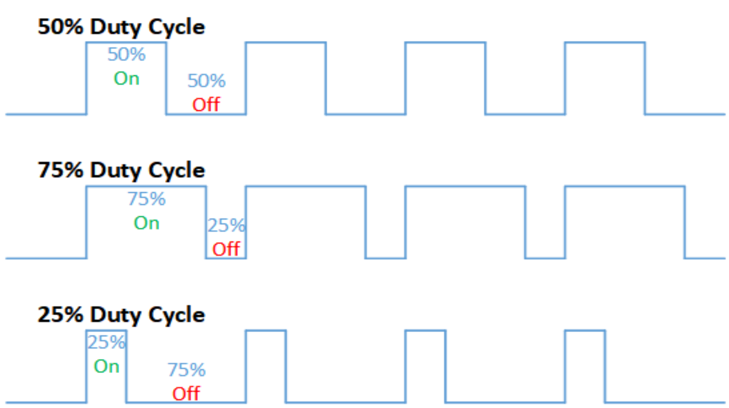

Figure-2

Pulse width Detection is the process of measuring the Duty cycle of the unknown PWM signal which can be easily measured by knowing the ON time and the OFF time of the signal which calculates the Duty cycle of the cycle.
In this Project we are measuring the duty cycle of the RGB LED in two ways

1. Hardware Pulse Width Detection
2. Software Pulse Width Detection

- **Hardware Pulse Width Detection Implementation**

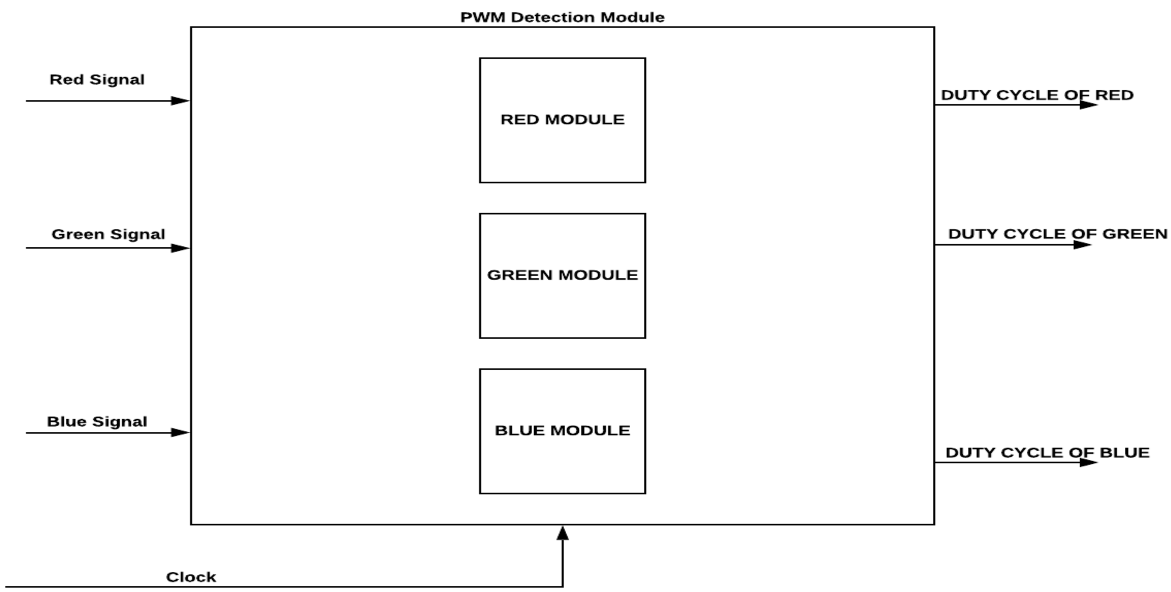

Figure-3

The above Figure Shows the Hierarchy of my module in which I have Implemented the logic of each signal differently in the module and then I am Instantiating the Three modules in to 1 module which is the PWM Detection module. Which is further getting instantiated into the Top level.

In this approach I am detecting the duty cycle of the PWM output of RGB1. To implement the Hardware based approach I am writing the System Verilog Code which calculates the duty of the PWM.
So, I am passing the RGB output to my detection module which uses the reference clock of the 6MHZ to calculate the duty cycle of the signal.

At the positive edge of the reference clock I am monitoring the level of the PWM output if it is High then starting the counter to measure the on time of the signal and if it is low it will measure the off time of the signal with different counter.
At the positive edge of the PWM output I am obtaining the value of the both high and the low period of the output thus, I am measuring the duty cycle at this point and passing the value to the SDK via GPIO port to display on the Seven Segment.

- ## Software Pulse Width Detection Implementation

In this approach the application needs to be able to read the duty cycle of the output of the RGB LED. The top-level module wraps the PWM output of RGB1 into the GPIO channel to make it available for the software. I implemented the Pulse Width Detection logic in the FIT Interrupt Handler. In which I am reading the values from the GPIO pin using API into FIT Handler I am also reading the value of the GPIO pins outside of the FIT Handler which will help me to detect the Edge of the signal.

```
    // Read the GPIO port to read back the generated PWM signal for RGB led's
    gpio_in = XGpio_DiscreteRead(&GPIOInst0, GPIO_0_INPUT_0_CHANNEL) & g_mask;

    if((previous_state != gpio_in) && (gpio_in))
    {
        g_duty_cycle = ((on * 2 * 100) / (on + off) );
        on = 0;
        off = 0;
    }
    if (gpio_in)
    {
        on++;
    }
    else
    {
        off++;
    }

    previous_state = gpio_in ;
}
```

Figure-4

The above snippet shows the same where I am incrementing the on and off counter at the level of the signal and when I am detecting the edge of the signal, I am calculating the duty cycle of the signal and making the counter 0. The edge detection is implemented by comparing the previous value which is the value of the signal out of the FIT interrupt handler with the value of the signal in the FIT interrupt handler.

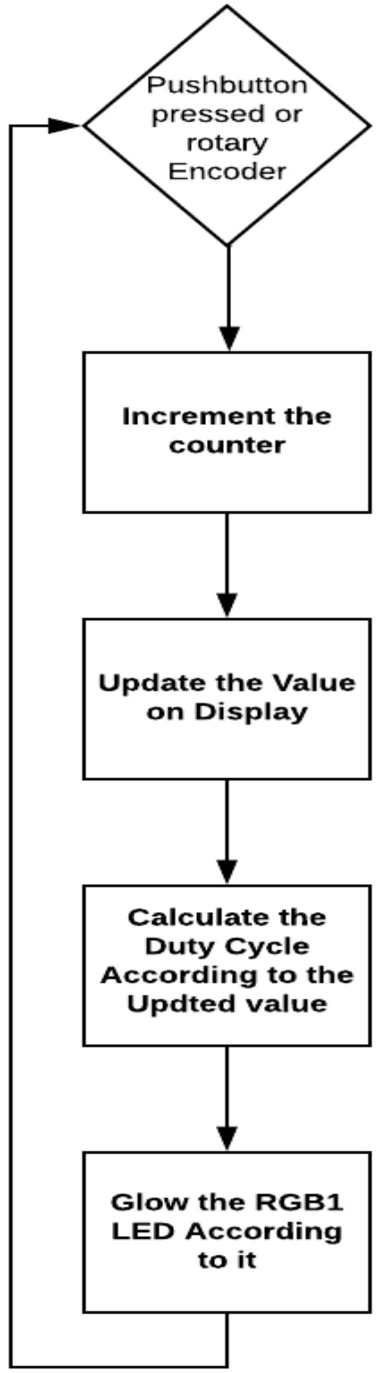

Figure-5

This above figure is the flow of the while loop in the software which I have implemented. It provides the basic understanding of how my while loop is getting executed.

Challenges Faced

I was not able to meet the timing specifications for the Hardware module due to which I added the synchronizer at the input side also buffered the signal in the Pipeline to add some flip-flops to meet the timing specifications.

Extra Features

- I have implemented the blinking of the unused LED’s to indicate how frequently my while loop was getting Execute
- Also, I am Displaying the “H” on Seven Segment when I am in the Hardware mode and “S” when I am in the software mode.

References

- Digilent Nexys A7 Reference Manual and schematics. Copyright Digilent, Inc.
- Digilent PmodOLEDrgb Reference Manual. Copyright Digilent, Inc.
- Digilent PmodENC Reference Manual. Copyright Digilent, Inc.
- https://www.xilinx.com/support/documentation/ip_documentation/axi_timer/v2_0/pg079-axi-timer.pdf