Portland State University

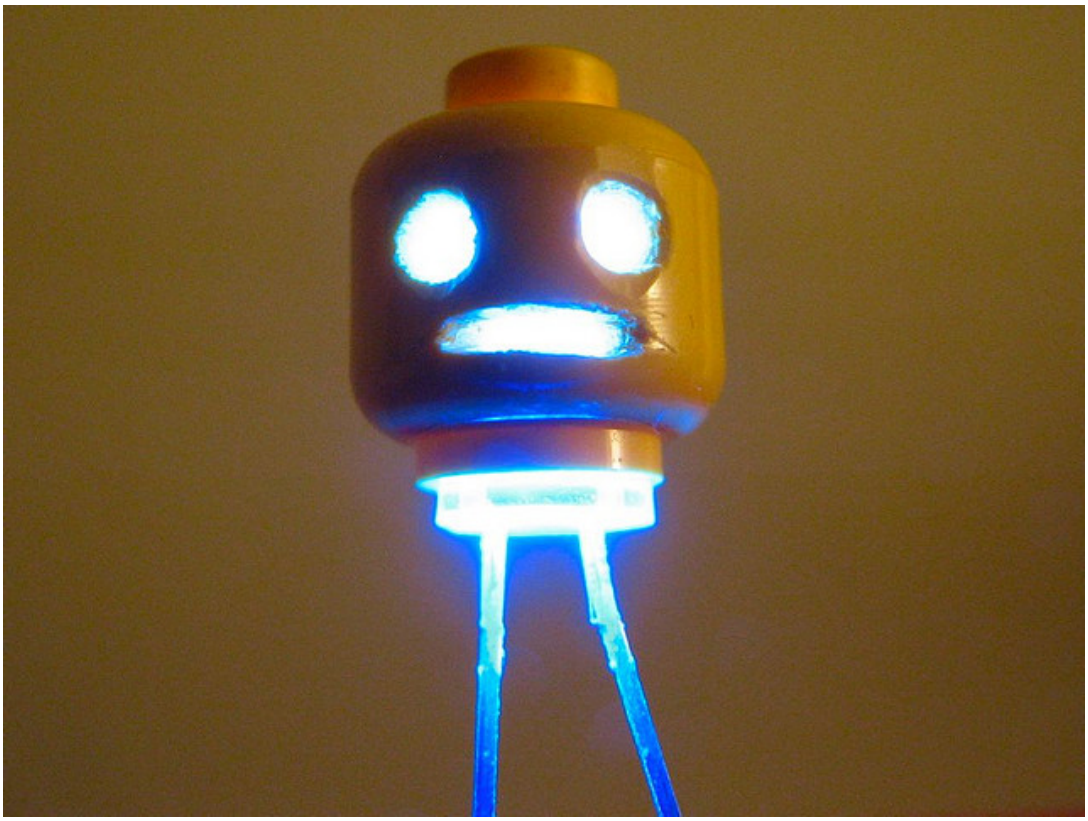
Embedded Systems on FPGAs

ECE544

Closed-Loop Control With PWM

Erik Rhodes Caren Zgheib

May 11, 2014



1 Introduction

~~Webster's dictionary defines~~ This project demonstrates how closed-loop control can be used in modern applications. Our embedded system monitored the output (brightness) of an LED with a light sensor, and calculated the corresponding PWM output to stabilize the brightness at the desired setpoint.

2 System Design

Embedded system includes: Micron RAM, lightsensor peripheral, GPIO pins for debug, N3eif peripheral to interface the PMOD rotary encoder and LCD display. Our circuit board was connected through the other ports. The PWM in and frequency out pins were...

3 Implementation

3.1 Control Methodology

Being able to control a dynamic system is an integral part of many everyday systems. Accomplishing this requires a feedback loop with an external measurement of the output being fed back into the system. These systems can be controlled in a number of ways. Two popular methods are **Bang Bang** and **PID**.

3.1.1 Bang Bang

Bang bang is quite straightforward: if the output is lower than desired, the controller puts the control signal at its highest amount. Likewise, if the output is higher than desired, the signal is set to the lowest level. While sometimes effective, this is a crude and cheap method for controlling systems, and is usually only used in devices where accuracy is not extremely important.

3.1.2 PID

The more prevalent method, PID, involves making multiple calculations to predict the accurately control the behavior of the output. The proportional (P) method involves using the previous error margin to calculate the next appropriate one. The integral (I) method calculates how the system behaves over time, and the derivative (D) calculates how fast the output is changing at that point in time. Using a specific arrangement of these parameters will yield a large increase in accuracy, so tuning the application is desirable. When tuning, we first attempted to follow the Ziegler-Nichols method. After understanding how each method changes the output, we modified the parameters as we saw fit to acquire the most accurate configuration for our control system.

3.2 Peripheral Interface

The TSL237-S-LF-ND light to frequency converter outputs a period that directly corresponds to the intensity of the light emitted from the LED. The PWM detection module receives this information and converts it to a "count", which is then scaled to fit within the parameters (between 0 and 4095) needed for our control calculations. These functions are handled in the light sensor driver. The driver also converts the scaled counts to a voltage, the unit of measure required to drive the PWM output.

Listing 1: Bang Bang Implementation

```
for (smpl_idx = 1; smpl_idx < NUM_FRQ_SAMPLES; smpl_idx++)
    {
        sample[smpl_idx] = LIGHTSENSOR_Capture(LIGHTSENSOR_BASEADDR, slope, offset
            , is_scaled, freq_min_cnt);
        volt_out = (-3.3 / 4095.0) * (sample[smpl_idx]) + 3.3;
        if (volt_out < setpoint)
        {
            Status = PWM_SetParams(&PWMTimerInst, pwm_freq, MAX_DUTY);
            delay_msecs(1);
            if (Status == XST_SUCCESS)
            {
                PWM_Start(&PWMTimerInst);
            }
        }
        else
        {
            Status = PWM_SetParams(&PWMTimerInst, pwm_freq, MIN_DUTY);
            delay_msecs(1);
            if (Status == XST_SUCCESS)
            {
                PWM_Start(&PWMTimerInst);
            }
        }
        delay_msecs(100);
    }
```

3.3 User Controls

The program starts by characterizing the system. This allows the initial scaling to be perform, which varies by the amount of light detected by the light sensor. After this is done, the default control parameter values are displayed and the user is given a chance to change them. The type of test and starting input voltage can also be selected. Once these have been configured, the user starts the test with the rotary button. When it has finished, long pressing on the rotary button sends the data to the computer connected serially via the UART port. If the user wants to change the test, they can modify the switch values and update the LCD by pressing on the rotary button. While he user interface remained close to the recommended specifications, it also included a few features that made it amazingly incredible.

4 Results

4.1 Characterization

Fairly linear

4.2 Bang Bang

Oscillates back and forth with an error of ...?

4.3 PID

Show a few different examples of some good results and list the parameter values.

Name	Value	Function
Switch[1:0]	00	Bang Bang
Switch[1:0]	01	PID
Switch[1:0]	10	Unused
Switch[1:0]	11	Characterization
Switch[2]	0	Vin Low
Switch[2]	1	Vin High
Pushbutton	North	Move Cursor
Pushbutton	East	Increase Value
Pushbutton	West	Decrease Value
LCD Display	Row 1	PID Values
LCD Display	Row 2	Setpoint and Offset
Rotary Encoder	Clockwise	Increase Setpoint
Rotary Encoder	Counter-Clockwise	Decrease Setpoint
Rotary Encoder	Press Button	Next section
Rotary Encoder	Long Press Button	Initiate test

Table 1: Nexys3 Controls

5 Conclusion

Difficult project, good results, hours spent...
Our work was split up as follows:

	Erik	Caren
Peripheral		✓
Lightsensor Driver		✓
User application	✓	
Integration	✓	✓
Documentation	✓	✓

Table 2: Division of Tasks

5.1 Challenges

- **Xilinx**: This "tool" screwed us over a lot.
- **Hardware interfacing**: PWMDET errors
- **Debugging**: Incorrect readings,
- **Typecasting**: Gave values converted incorrectly or with lost precision
- **Integration**: We wrote them separately very early on, but it was hard to integrate and fix all the different issues