Learning Objectives:

- Gain experience with PID control and peripheral driver development using I2C
- Gain experience FreeRTOS Implementation
- Lay the groundwork for your final project

Notes:

Revisions to support both the Digilent Nexys A7 (Nexys4 DDR) and the RealDigital Boolean Board:

- The Boolean board only has 4 pushbuttons instead of the 5 buttons on the Nexys A7. We will not use `btnC` on the Nexys A7. Buttons are mapped as follows (`Nexys A7(Boolean)`):

```
{btnC(N/A), btnu(BTN0), btnd(BTN3), btnl(BTN2), btnr(BTN1)}
```

- Both FPGA target boards have 2 RGB LEDs but they are numbered differently (`Nexys A7(Boolean)`):

```
{RGB1(RGB0), RGB2(RGB1)}
```

- There is no `btnCpuReset` on the Boolean Board. We will implement this on Boolean with:

```
btnCpuReset = ~(BTN0 & BTN1)
```

- Both FGPA target boards have 16 LEDs and 16 switches and an 8-digit 7-segment display.

The text in this document uses the Nexys A7 names to be consistent with past releases.

Additional tools to install

Terminal Software – `Putty` (`https://putty.org/`) or `TeraTerm` (https://tera-term.en.softonic.com/) (Allows us to save the console output)

Project Overview

In this project you will gain practical experience working with FreeRTOS and interfacing to an external sensor. Project #2 uses a MicroBlaze AXI based system which includes an AXI I2C IP. The AXI I2C module is used to get gyroscope measurements from an MPU-6050 sensor.

Project #2 is a team-based project (teams of 2). You may self-enroll in teams. One team member should pick an unassigned team, assign themselves to it, and let their partner know which group to join.

Functional Specification

Summary

Functionally, Project #2 implements firmware using FreeRTOS on a Microblaze-based system to measure the ROLL of an MPU-6050 Accelerometer/Gyroscope module. The angle reported by the sensor should match the target angle given to the application by the user. A closed-loop PID controller guides you in how much more you need to rotate the MPU-6050 to reach the target angle. Follow the given angle recommendation and try to match it over time. Capture all the angles and generate a graph. Try to get the best suggestions by tuning the K_p (proportional), K_i (integral), and K_d (differential) gain parameters in the PID loop.

You will be creating the MPU6050 driver to support the I2C interface to MPU-6050 and then build your application using the driver to:

- Start a `Menu` task which queries the user for a run mode request
- Start a `Run` task. You will be resuming three other tasks in the `Run` task – the `Data` Task, the `PID` Task, and the `Exit` Task.
    - Data Task - Gets the GyroX data from the MPU-6050 sensor using the AXI I2C driver functions and compute the GyroX angle.
    - PID Task - Get the GyroX angle and implement a closed loop PID Controller using the data. The less "distance" between the target angle and the GyroX angle, the smaller the correction.
    - Exit Task - monitor if any of the switches on the target FGPA board have been switched ON. If so, suspend the `Data` Task, `PID` Task and `Exit` Task and resume the `RUN` Task.
- Tune the PID Kx parameters to get good suggestions of angle
- Plot the graphs based on the responses.

Hardware, Firmware and Embedded System Setup:

Please refer to the Project outline for better understanding.

Deliverables (in checklist format):

- A live demo to either Omkar or myself. We will announce lab hours for demos at the Capstone Lab in FAB and, if there is demand, at WCC before or after class. We recommend including a video demo in your deliverables, too.
- A 3–5 page design report explaining the operation of your design, most notably a description of your application code, problems encountered and how you solved them, and suggestions for how the project could be improved.
- Source code for your project #2 application. The Getting Started and Project #1 applications will not be the best starting point – the functionality, OS , and application are different. Please make the application code your own. For example, my name appears in the comments at the top of the program… but this is your program, not mine. Make sure the headers and comments are up to date and that there are no "artifacts" (like commented-out lines) in the code. The readability of

your code counts. We will look kindly at code that is organized and easy to follow. We will not look kindly at code we have to slog through to make sense of it.

- Source code for any HDL that you write or modify. Be sure to include the code for your top-level module since there is a possibility that you will make changes to it. Personalize any code you copy/modify.
- Constraint file(s) if you made any changes to the provided `.xdc` file(s) or any additional constraint files you added to your project.
- A schematic for your embedded system. You can generate this from your block design by right- clicking in the diagram pane and selecting *Save as PDF File…*

Grading:

You will be graded on the following:

- (60 pts) The functionality of your demo.
- (20 pts) The quality of your design expressed in your C and SystemVerilog source code. Please add comments to your code to help us understand how it works. The better we understand it the better grade we can give it.
- (20 pts) The quality of your design report. Keep the report concise, but remember, the overarching goal is to help us understand your implementation and what went well, or not.

Project Tasks:

1. Create Project in Vivado

2. Follow the steps in Hardware Outline and Create the Project #2 embedded system and generate the bitstream

3. Follow the steps in Firmware Outline and create the application

4. Interface the MPU-6050 sensor

Connect the I2C Pins to the JC Connector of the Nexys A7 FPGA.
SDA to JB[4]
SCL to JB[3]
You need to use I2C Address as 0x68 and hence you need to ground the AD pin.

5. Test Your application

To test your application, follow the steps accordingly:

a. If you are waiting for an MPU-6050 sensor get the FreeRTOS Application by testing the basic working of all the other peripherals such as
   - UART debugging.
   - Task switching (i.e. do the tasks resume and/or suspend correctly.
   - Are you able to pass user inputs in `Run` mode?
   - Are you able to send the sensor into sleep mode?
   - Are you able to switch to test mode?
   - Are you able to exit the Run mode and switch back to the menu?

- Implement a workaround near the PID controller by providing constant values to the controller and getting the angle suggestions accordingly.

b. If you have an MPU-6050
   - Check if the I2C initialization is successfully done before starting the scheduler. You should be able to read the address of the MPU-6050 Sensor.
   - Are you able to set the Gyroscope configuration?
   - Can you get measurements from the Gyroscope when idle (without rotation)
   - Can you observe the readings and rotate accordingly!

6. Schedule a live demo for your project.

Once you have your Project #2 hardware and application working to your satisfaction you will demonstrate it to either Omkar or me. We will publish a demo schedule a few days before the due date for the project

7. **Upload your code (along with a working video) and report (including your "embsys" schematic) to Canvas.**

Bundle your deliverables into a single .zip (or .rar or .tgz or .7zip) file and upload it to your Project #2 dropbox. You may submit more than once – we will grade your latest submission so make sure that each upload is complete.

References:

1. *Digilent Nexys A7 Reference Manual* and schematics. Copyright Digilent, Inc.
2. *RealDigital Boolean Board Reference manual and schematics. Copyright RealDigital* 3. *Getting Started in ECE 544 (Vivado/Nexys A7)* by Roy Kravitz.
4. *Hardware and Firmware Outline for Creating your Project #2 embedded system setup*.
5. MPU-6050 Sensor Datasheet - https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
6. MPU-6050 Sensor Register Map - https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf
7. MPU-6050 Sensor Purchase Link: https://www.amazon.com/gp/product/B00LP25V1A/ref=ppx_yo_dt_b_asin_title_o00_s00?ie=UTF8&psc=1
8. Project #2 Embedded System Hardware Design
9. FreeRTOS API - https://www.freertos.org/a00106.html
10. *Previous documentation* by Roy Kravitz.
11. MPU-6050 Basics: https://howtomechatronics.com/tutorials/arduino/arduino-and-mpu6050-accelerometer-and-gyroscope-tutorial/
12. Calibration and angle measurement: https://www.youtube.com/watch?v=7VW_XVbtu9k

https://www.youtube.com/watch?v=Yh6mYF3VdFQ