

Project 2

(optional) “Live” demos: Mon, 24-May-2021
Deliverables: by 10:00 PM on Wed 26-May-2021

You will do this project in teams of two

Project 2 Assignment

- This project is designed to give you practical experience with closed loop control...and it does so with real hardware
 - You will use the PmodHB3 H-bridge to drive a DC motor with a gearbox
 - You will sample/read an encoder to calculate rotational velocity
 - You will vary the load on the motor by applying pressure to a wheel attached to the driver motor
 - The PID control algorithm in your application should try to maintain the target speed of the motor under varying loads

Project 2 Assignment (cont'd)

- You will control the H-bridge (and thus, the drive motor) from a Microblaze application
 - Includes outputting a PWM signal and a 1-bit logic signal to the H-bridge to control the speed and the direction of rotation of the drive motor
- You will monitor the speed of the motor from a Microblaze application
 - There is an integrated quadrature encoder in the motor that generates a pulse waveform that you can use to detect the speed (and rotational direction if desired) of the motor

Project #2 Assignment (cont'd)

- You will create a driver for the H-bridge custom peripheral using the Vivado Create and Package IP wizard
- You will implement closed loop control in your application to maintain drive motor speed in the face of a variable load
 - Closed control is implemented w/ a Proportional-Integral-Derivative (PID) controller done in software
 - You will vary the control parameters of the control system (K_p , K_i , K_d) in your application using the buttons and switches on the FPGA board
- Once you have “tuned” the control loop you will create a few “interesting” (no load, w/ load, ...) graphs on the performance of your control system by uploading the raw data to a PC and graphing the results
- You will demonstrate your working project with a video or you can request at time slot for a “live” (via Zoom) demo

Project #2 Assignment (cont'd)

- You will submit the following deliverables:
 - A block diagram of your system
 - All of the source code that you wrote. Please take ownership of your code. We want to see your comments not ours
 - Five to seven (5 to 7) page project report:
 - Explain the operation of your design, most notably your control algorithm and user interface
 - Include at least two “interesting” graphs showing the results from the control algorithm
 - List the work done by each team member. Be sure to note any work that you borrowed from somebody else
 - List issues you ran into and how you resolved them
 - Offer suggestions about how to improve this project

Project #2 Assignment (cont'd)

- ...and did I forget to mention:
 - The Microblaze should be configured with 128KB of memory and hardware floating point
 - You will have to modify the top-level module and the constraints file for your target FPGA board to match your new embedded system
 - Final target is a multitasking application implemented under FreeRTOS

Functional Specification

- After board reset, the system should end up in a mode with the motor off (PWM=0), and control constants (K_p , K_I , K_D) 0
- The rotary encoder on the PmodENC controls the motor, like a volume control
 - clockwise increases speed (RPM)
 - counter-clockwise decreases motor speed (RPM)
- Pushing the center button should set the motor speed and control constants back to zero, this should be a high priority event, possibly interrupt controlled and not polled
- The desired RPM should be displayed on the Seven Segment display
- The RPM and control parameters (K_p , K_I , K_D) should be displayed on the PmodOLEDr gb. you will have to choose scales and a display format that works best for you.
 - Rotational velocity at the output shaft should be RPM and it should be in decimal. Note that the integrated quadrature encoder is connected to the input shaft (before the gearbox) of the motor.

Hardware

- Use Vivado and the IP Integrator to create an embedded system with this minimum specification, you can add additional hardware:
 - Microblaze, mdm, interrupt controller, hardware floating point, etc.
 - Program/Data memory (board-dependent):
 - Local (BRAM) memory of 128KB for program/data memory
 - FIT timer set to generate 5KHz interrupts
 - Digilent PmodOLEDrgb and PmodEnc IP
 - UartLite peripheral, configure the Uart Baud Rate to 115200
 - Dedicated AXI Timer to provide “systick” to FreeRTOS. Note that AXI Timer/Counter 0 is the default in FreeRTOS.
 - AXI Timer or some other mechanism to provide PWM signal to the PmodHB3
 - AXI Timebase/Watchdog Timer
 - Two additional GPIOs. GPIO_0 should be 16-bit output only and connected to the LEDs. GPIO_1 should be configured for interrupts and connected to the pushbuttons and switches. Hint: You can use an x1concat block to combine all of the pushbuttons into a single input to the GPIO
- ...and whatever other hardware you need for your implementation

Control Circuit Schematic

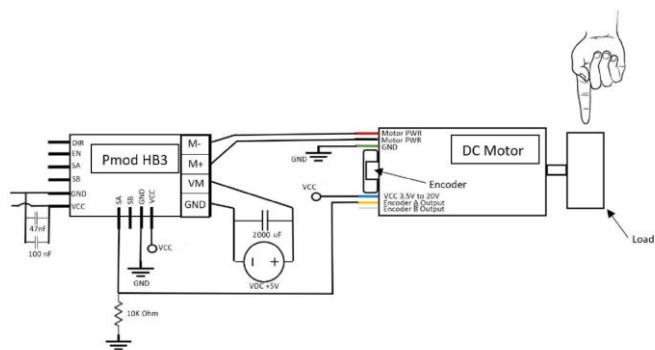


Figure 1. Control Circuit Schematic

BOM: [BOM_Project2.pdf](#)

Control Algorithm Characterization & Debug

Characterization:

- Step Response – Switch PWM from 0x00 to 0x255. Record the change in RPM over time
- Sweep – Sweep PWM from 0x00 to 0x255 with a delay between the steps. Record the RPM at each PWM setting

Debug:

- One issue is that things may happen so quickly that you will not be able to easily detect how well your control algorithm is operating
- One thought is to keep the accumulated sum of squared error after each speed change and display that on the PmodOLEDRgb

Project 2 Tasks Summary

- Select a partner
- Download the Project 2 release package
- Connect the motor to your FPGA development board.
- Write drivers to control and monitor the drive motor speed
- Build the embedded system and top-level module for the project
- Implement the control loop
- Build your application to control the motor
- Tune your control system response
- Demonstrate your project and submit the deliverables