Results module 8

Step 5.

|  |  |  |  |
| --- | --- | --- | --- |
| Duty | AVG actual RPS  (module8a) | Calc\_rps | Percent Difference |
| 25 | 2.52 | 2.47 | 1.98% |
| 50 | 3.89 | 3.94 | 1.28% |
| 75 | 4.59 | 4.67 | 1.74% |
| 90 | 4.75 | 4.8 | 1.05% |
| 100 | 4.72 | 4.75 | .63% |

Average percent difference: 1.43%

A graph with blue lines

Description automatically generated

Our FFT is a double of the value. So it was acting quite weird.

Step 6. Fair warning. Our motor does have slower RPS for each duty cycle compared to the expected RPS for other motors. We don’t know why this is and have talked with Prof. Vaughn and says it is okay.

Step 7

A graph with red lines

Description automatically generated

A graph with red lines

Description automatically generated

we reran the 50% duty cycle and the RPS value was 4.23 from module8a.py and the calc\_rps value was 4.25, so this graph is very accurate. At .5 seconds it reaches

Step 10.

Testing with 6 rps

1. kp = open loop gain
   1. 5.76
2. kp = ½\* open loop gain
   1. 5.56
3. kp = 2\*open loop gain
   1. 5.34
4. kp = 10\*open loop gain
   1. 5.1

In this section messing around with the size of Kp we found that when it was equal to open loop gain it was closest to the desired RPS and when it was equal to half of the open loop gain was a little less than the desired RPS. However, for 2x and 10x the performance of the Kp values became worse and worse, seeing lots of large oscillations throughout the graph. Overall from this procedure we came to the conclusion that we want to stick close to the open loop gain for the Kp, or less than it to reach the desired RPM.

Step 11.

The velocity of 2 was very inaccurate and the velocity of 6 was unobtainable to reach due to our motor. Professor Vaughn said we could do a velocity of 3 and 5 for the repo and results.txt.

Velocity of 3:

Rise time: 0.4 seconds

Overshoot: 1.89 RPS

Steady State Error: .31 RPS

Velocity of 5:

Rise time: 2.31 seconds

Overshoot: .52 RPS

Steady State Error: .27 RPS