Response Time and Data Comparison Between Implementations of Quadrature Decoder

Task: 105

Assigned: Jason Guest

This document is designed to serve as a reference in comparing how each of the three implementations of the quadrature decoder perform against each other, as well as illustrate the differences between them. A description as well as any relevant specifications are provided along with recorded performance metrics.

**Software Implementation:**

This implementation handles all output from the quadrature encoder in software only, without using any peripherals on the Curiosity Nano board to process any of it. This implementation is intended to serve as a baseline in performance expectations for the interrupt and CCL based implementations.

*Recorded data:*

Following results collected via software timer in the code, catching only the execution of the logic required by the project specifications.

**Interrupt-driven Implementation:**

In this implementation of the quadrature decoder, hardware interrupts are enabled, as well as a couple other on-board peripherals, such as timer counters in order to process the information coming from the encoder. The purpose of the data from this implementation is to show that the use of hardware peripherals improves performance when compared to a purely software-based approach.

*Recorded data:*

Data from this implementation was collected from a script written by team member Tanner Lisonbee in Python, and the data was automatically collected by the script with simulated input.

**CCL Implementation:**

This third implementation is intended to showcase the new CCL or Configurable Custom Logic block or peripheral of the ATmega 4809 Curiosity Nano board. By using the CCL peripheral, we intend to show that the performance of a Quadrature Encoder can be improved over an interrupt driven or software-based approach.

*Recorded data:*