Peter Lodato

17 April 2018

**VHDL Motor Encoder Reader Tech Memo**

1. Intro

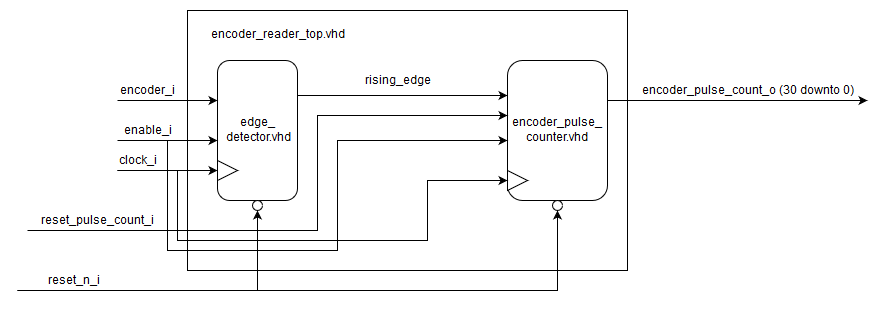
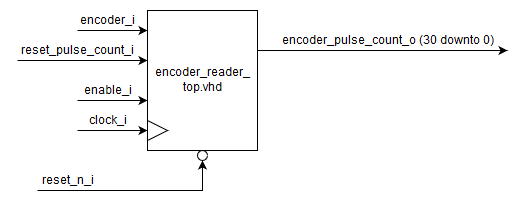
It is crucial that our rover knows the distance that it has been travelling over time. This is done by reading the encoder signals from the DC motors that turn the rover wheels. By reading these signals and measuring how rapidly they change over time, the speed and distance can be calculated through either hardware or software means.

2. Working Principle

As the motor moves, the encoder reads a black-and-white disk and sends a signal HIGH or LOW depending on whether a non-moving light sensor is reading a bright (white) or dark (black) portion of the disk. As a result, the faster the motor moves, the faster the disk moves along with it and the faster the output signal changes from HIGH to LOW. The hardware must read this signal and decide what to do with it. For simplicity’s sake, it was decided that the design would simply count the number of times the encoder output signal goes HIGH over time before a reset signal was sent to the design. It accomplishes this by using a rising-edge detector for the encoder signal with a tick counter being used to count the number of times that the resulting signal for this component goes HIGH. Originally, there would be another VHDL subcomponent that would convert tick count to a distance in centimeters or millimeters. However, it was later decided that this task would be offloaded to software instead to keep the hardware simple.

To count the number of times the encoder signal goes HIGH, the signal is first fed into a rising edge detector. This subcomponent outputs a signal that is LOW until there is a rising edge on the encoder signal, at which point the resulting output signal goes HIGH for one clock cycle. This pulsing signal is then fed into a subcomponent that increments an output vector by one for every clock cycle that the input signal is HIGH. Since the input signal is only ever HIGH for only one clock cycle, this subcomponent essentially counts the amount of pulses on this signal. The pulse count resets to zero when the reset counter signal is activated or when enable is deactivated.

3. Block Diagrams



4. Simulation and Results

The design was simulated using Modelsim for 4.05 milliseconds and according to the results, the design worked as intended with the proper requirements and within specifications.

