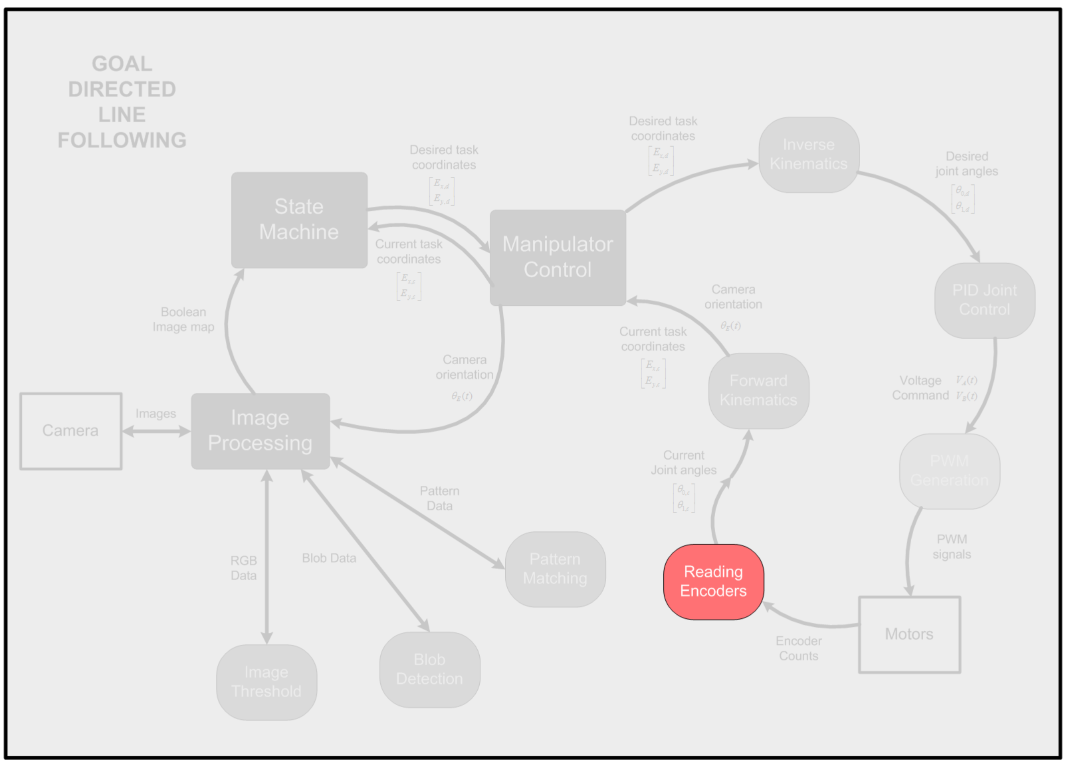
Reading Encoders

Topics Covered

- Decoding encoder data in Quadrature mode.
- Implementing a LabVIEW™ VI to decode in quadrature mode.
- Adding direction sense and calibration functionality.

Prerequisites

- The QNET Mechatronic Systems is set up according to the Quick Start Guide.



1 Background

1.1 Encoders

The QNET Mechatronic Systems motors use rotary incremental optical encoders, shown in Figure 1.1. Similar to rotary potentiometers, encoders can also be used to measure angular position. Unlike potentiometers, encoders are relative. The angle they measure depends on the last position and when it was last powered. It should be noted, however, that absolute encoders are available.

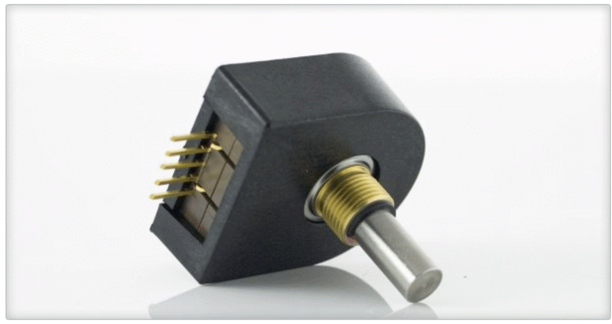


Figure 1.1: US Digital incremental rotary optical shaft encoder

The encoder has a coded disc that is marked with a radial pattern. This disc is connected to the shaft of the DC motor. As the shaft rotates, a light from a LED shines through a pattern and is picked up by two photo sensors, A and B. This effectively generates the A and B signals shown in Figure 1.2. An index pulse is triggered once for every full rotation of the disc, which can be used for calibration or homing a system. Note that the QNET Mechatronic Systems encoders do not use an indexing pulse.

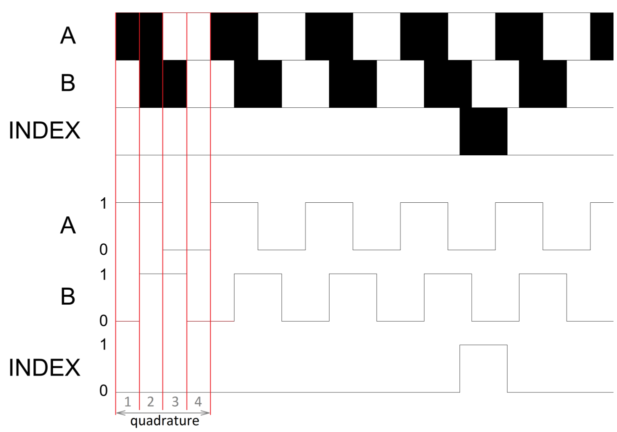


Figure 1.2: Optical incremental encoder signals

The A and B signals that are generated as the shaft rotates are used in a decoder algorithm to generate a count. The resolution of the encoder depends on the coding of the disc and the decoder. As the QNET Mechatronic Systems motor encoders have 512 lines on the disc, they can generate a total of 512 counts for every rotation of the encoder shaft. However, in a quadrature decoder as depicted in Figure 1.2, the number of counts (and thus its resolution) quadruples for the same line patterns and generates 2048 counts per revolution. This can be explained by the offset between the A and B patterns: Instead of a single strip being either on or off, now there are two strips that can go through a variety of on/off states before the cycle repeats. This allows the encoder to detect the directionality of the rotation, as the sequence of on/off states differs for a clockwise and counter-clockwise rotation.

1.2 Calibration

The Encoder counts should ideally be set to zero whenever an experiment begins. However, this encoder ‘zero’ state would then refer to whatever state the QNET Mechatronic Systems was initially in. It is ideal to calibrate the QNET Mechatronic Systems to a ‘home’ position to maintain consistency during all the experiments. This also helps develop functions or models that are independent of the initial QNET Mechatronic Systems starting position.

This can be done by resetting the encoder counts to 0, when the manipulator reaches it’s ‘home’ position.

2 In-Lab Exercises

2.1 Quadrature decoding in LabVIEW™

In your system's explorer, browse to the your project folder. Under the folders `Subsystems | Investigation Controllers | Reading Encoders`, open `Reading Encoders.vi`. Browse to the block diagram, and open `Unidirectional Quadrature Decoder.vi`. Complete the block diagram according to the model shown in Figure 2.1.

Note: Ensure that the file is opened through your system's explorer, and not `Mechatronic Systems.lvproj`. This will make sure that the processor from your CPU is used, and not that on the NI ELVIS RIO Control Module.

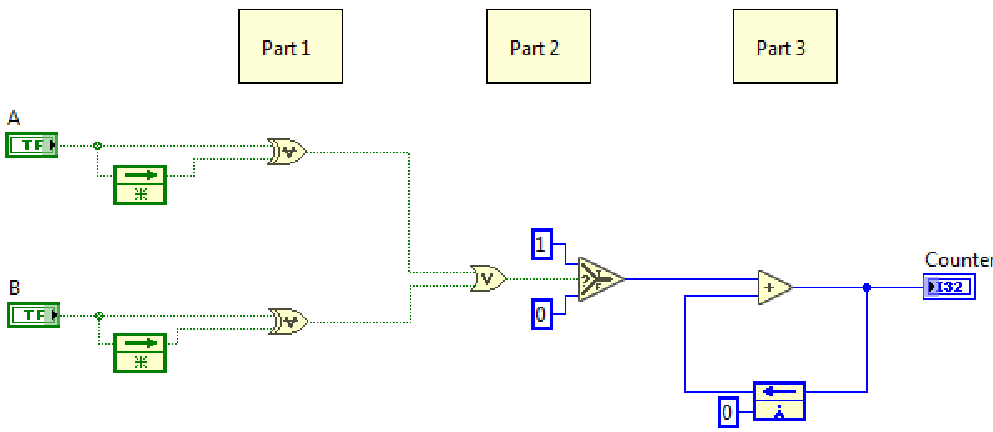


Figure 2.1: A quadrature decoder to generate counts

As the motor spins, A and B receive pulsing waveforms with 'high (1)' and 'low (0)' signals. The first step involves detecting a rise or drop, which corresponds to a count. This is done in Part 1 of Figure 2.1 using the XOR function block. Part 2 adds 1 to the counter if it detects a rise or drop, and 0 otherwise. Part 3 updates the counter value. Save and close this VI. `Reading Encoders.vi` generates a sample signal for A and B, and displays the encoder counts on a graph.

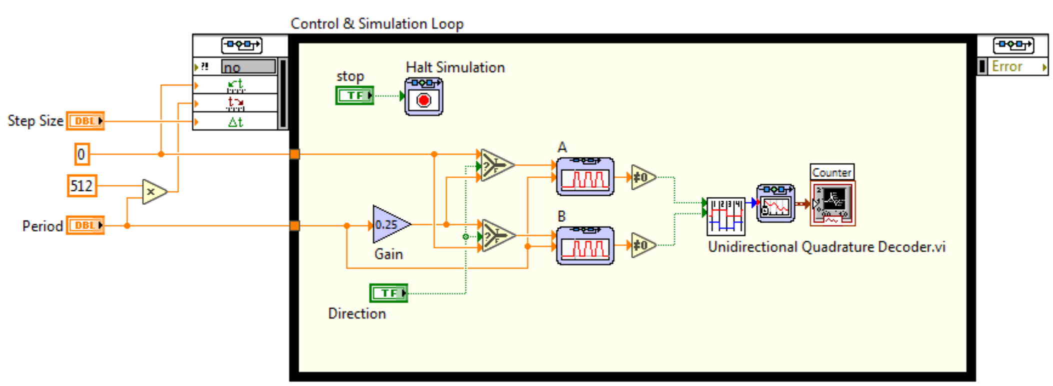


Figure 2.2: Main VI to validate the Quadrature Decoder Sub-VI

1. Back in `Reading Encoders.vi`, set the Step Size as 0.1 s and the Period as 1 s. Keep the Direction switch ON

(that is, A leads B). Run `Reading Encoders.vi`. Are the number of counts generated in the simulation time expected?

2. Turn the Direction switch OFF (that is, B leads A) and run the VI. Are the counts accurate?

2.2 Quadrature decoding with Direction sensing and Encoder Reset

To sense direction, the decoder must add to the current count as A leads B, and subtract from the count as B leads A. Modify `Unidirectional Quadrature Decoder.vi` according to the model shown in Figure 2.3 and save it as `Bidirectional Quadrature Decoder.vi`. This new VI checks the current A signal compared to the previous B signal, and identifies if A is leading B or vice versa. It adds 1 or -1 accordingly. Update `Reading Encoders.vi` to use this new sub-VI.

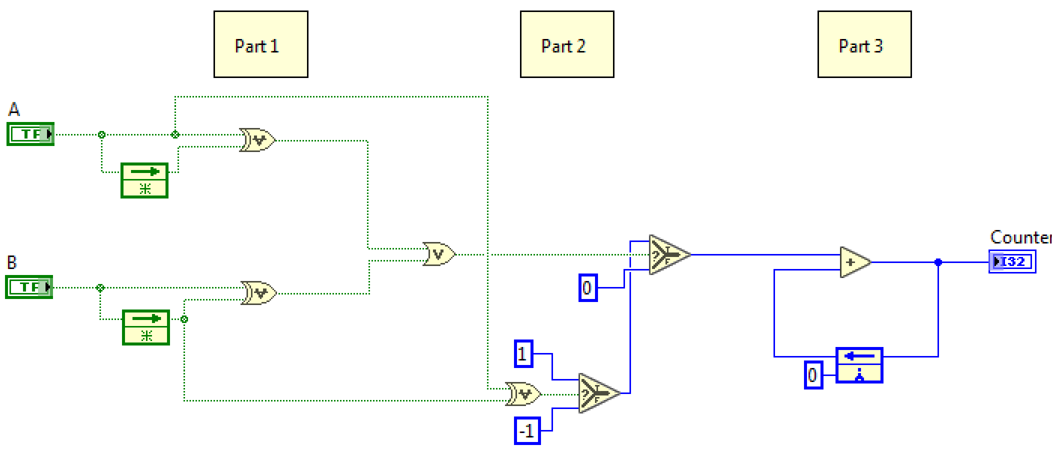


Figure 2.3: A quadrature decoder to generate counts with direction sensing

1. Run `Reading Encoders.vi` with the Direction switch ON and OFF, and attach the resulting graphs. Does the decoder work as expected?

2. Modify the Sub-VI with bidirectional sensing as shown in Figure 2.4. The addition in Part 3 allows the user to programmatically reset encoder counts back to 0, which aids calibration. Run `Reading Encoders.vi`. Does the reset button works as expected?

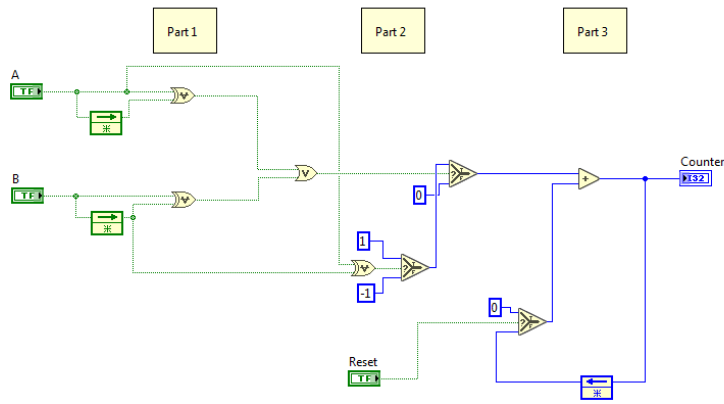


Figure 2.4: Modified Quadrature Decoding with direction sensing and reset

2.3 Quadrature decoding on the FPGA

While running the decoder, the simulation time step must be at least 25% of the Period, because the A and B pulses iterate through 4 different states within a single period, and must all be read by the system. If the simulation time step is larger than this, some of the states won't be read. In such a scenario, the decoder will skip counts as it won't be able to process information as fast as the counts are changing.

1. Keeping the Period at 1s, what should the maximum simulation step size be? Set the Step Size to a value larger than this threshold, say 0.3s with a 1s period. Enable the Reset switch. Run `Reading Encoders.vi`. Attach the resulting graph, and explain the result.

A typical speed of 1 rot/s corresponds to 2048 counts per second. To prevent anti-aliasing and missing counts, the sampling frequency must be at least twice as fast, that is, 4096 counts per second, or roughly 5000 Hz. Single point sampling cannot be done in LabVIEW™ this fast. This requires the use of an FPGA, which can prove advantageous when manipulating the QNET Mechatronic Systems.

Open the project `Mechatronic Systems.lvproj`. In the project menu, under `Quanser ELVIS RIO | Chassis | FPGA Target` , open `ELVIS-RIO Customized FPGA.vi`. In the block diagram, under the encoder section, open `Simple Encoder.vi`. This is the VI used on the FPGA to decode A and B signals from the QNET Mechatronic Systems motor encoders, as seen in Figure 2.5. As you can see, the sub-VI is very similar to the one developed during this investigation.

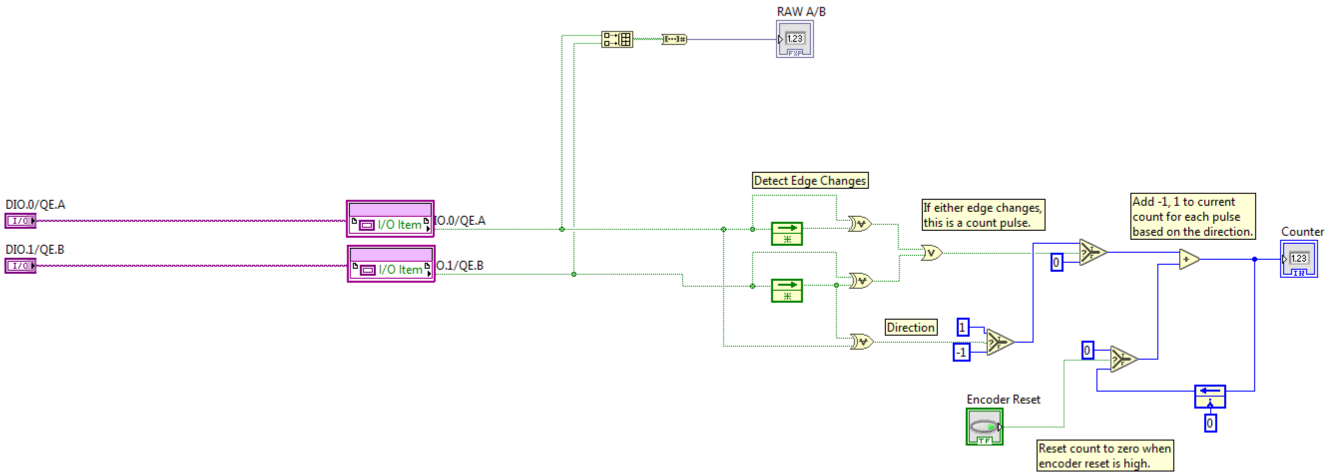


Figure 2.5: Encoder used by the FPGA for PWM

Quanser Inc.
119 Spy Court
Markham, Ontario
L3R 5H6
Canada
info@quanser.com
Phone: 1-905-940-3575
Fax: 1-905-940-3576

Printed in Markham, Ontario.

For more information on the solutions Quanser Inc. offers, please visit the web site at:
http://www.quanser.com