1. The QEI module interprets the two-bit gray code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel.

2. The position integrator and velocity capture can be independently enabled, though the position integrator must be enabled before the velocity capture can be enabled.

3. The robot will accept parameters from the programmer such as current position and final destination. These parameters along with parameters measured from the qei will develop the wheel speeds and direction to reach a destination. Additionally, the programmer will need to input dimensions for the robot.

4. Input parameters for robot dimensions.

getRobotDimensions(uint32\_t L, theta, r);

5. Input parameters for current destination and final destination.

getCurrentAndFinal(uint32\_t xCurrent, yCurrent, thetaCurrent, xFinal, yFinal, thetaFinal);

6. Adjust direction, turn first.

setTurn(uint32\_t phi, direction)

For direction, a value of 1 will indicate cw rotation and 0 is ccw.

7. Once the robot is directed in the correct path, proceed with movement in the x.

setDrive(uint32\_t x2-x1)

8. Add the PWM code from lab 2 to provide electrical power and controllability to the HUBee wheels.

9. Proceeding to the rest of the implementation:

//initialize system clock source

void SysCtlClockSet (uint32\_t ui32Config);

//initialize GPIO ports as needed

//Pins PD3 and PC4 will read INDEX values into the QEI modules IDX0 and IDX1 respectively.

//Pins PD6, PC5, PF1, and PC6 will read the pulses produced from the wheel encoders.

//Respectively they are PhA0, PhA1 and PhB0, PhB1

//We will need to enable these ports: C, D and F

void SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOC | SYSCTL\_PERIPH\_GPIOD | SYSCTL\_PERIPH\_GPIOF);

//Enable the PWM and QEI peripherals

void SysCtlPeripheralEnable(QEI0);

//Poll, wait for peripheral to accept clock setting

//initialize the QEI modules

void QEIConfigure (uint32\_t ui32Base, uint32\_t ui32Config, uint32\_t ui32MaxPosition);

//Enable the

QEIEnable(QEI\_BASE);

//API example suggests delaying before reading encoder position, may need a for loop

//Read encoder position

QEIPositionGet(QEI\_BASE);

//Copy over PWM code from lab2 here

PWM code…

//Once the peripherals have been initialize the main body of the code will call for the parameters listed above

getRobotDimensions(uint32\_t L, theta, r);

getCurrentAndFinal(uint32\_t xCurrent, yCurrent, thetaCurrent, xFinal, yFinal, thetaFinal);

setTurn(uint32\_t phi, direction)

setDrive(uint32\_t x2-x1)