The following tasks will be created:

* Thrust Input task
  + Awakened by a semaphore that is posted by the button ISRs when they change state. Responsible for setting the current thrust values (small or large) in the craft thrust data struct, based on the current status of the buttons.
* Direction Input task
  + Awakened by a periodic timer interrupt, which posts a semaphore. Responsible for updating the craft direction data struct based on the current position of the capacitive sensor.
* Physics calculation task
  + Awakened by a periodic timer interrupt, which posts to a semaphore. Responsible for calculating the next tick data for our craft, based on our physical constants, and updating the craft direction data and craft position data accordingly. It also sends messages through message queues to the LED0 and LED1 control tasks to adjust their duty cycle and frequency. At the end of all calculations, it notifies the LCD of a change in position and direction data through a semaphore.
* LED0 control task
  + Awakened by a message in its queue from the physics calculation task. Responsible for deleting the previous LED0 On task timer and reinstantiating it with new values, as well as updating the LED0 clock data structure.
* LED1 control task
  + Awakened by a message in its queue from the physics calculation task. Responsible for deleting the previous LED1 On task timer and reinstantiating it with new values, as well as updating the LED1 clock data structure.
* LCD Display task
  + Awakened by a semaphore that is posted by the physics calculation task. Responsible for displaying the craft correctly on the LCD screen based on the craft position & direction data.
* LED0 On task
  + Awakened by a semaphore post from a periodic timer interrupt. Responsible for turning on LED0, reading the expected duty cycle and frequency from the LED0 clock data structure, and starting a one-shot timer based on that duty cycle and frequency which will call back to an LED0 off callback function.
* LED1 On task
  + Awakened by a semaphore post from a periodic timer interrupt. Responsible for turning on LED1, reading the expected duty cycle and frequency from the LED1 clock data structure, and starting a one-shot timer based on that duty cycle and frequency which will call back to an LED1 off callback function.

The following shared data structures will be created:

* Craft Thrust Data
  + Will contain data about the current fuel burn rate (high or low).
* Craft Direction Data
  + Will contain data about the current angle of the craft, as well as the current angular velocity of the craft.
* Craft Position Data
  + Will contain data about the current X, Y position of the craft.
* LED0 Clock Data
  + Will contain data about the current desired duty cycle and frequency of LED0.
* LED1 Clock Data
  + Will contain data about the current desired duty cycle and frequency of LED0.

The following ITC constructs will be utilized:

* Semaphore
  + A semaphore will be used to signal the LCD display to update after each time the physics update task has completed
* Mutex
  + Used to guarantee exclusive access to aspects of the craft direction data. Separate mutexes will be created for:
    - Craft direction data
    - Craft thrust data
    - Craft position data
  + The LED clock data structures, which would normally be mutexed, will be left without any mutex protection. This is because the potential damage of unlucky timing with mutexes could severely impact the output of the desired duty cycle and frequency, while leaving the data without mutex protection will at worst cause strange effects for a single LED on/off cycle after the data should be changed.
* Message Queue
  + A message queue will be used to signal the LED0 and LED1 control tasks when the duty cycle and frequency of flashes need to be updated.

Unit Testing plan

1. Initialize the direction input task and the craft direction data struct. Create stub versions of all of the capacitive sensor functions that return known values. Verify that with the capacitive sensor function stubs reading expected data into the direction input task, expected data is in the craft direction struct at the end of the test.
2. Initialize the LED0 control task, the LED0 data struct and the LED0 control task message queue. Create a mock task that runs on a periodic timer and sends known messages to the control task. Create a spy callback function that will be passed into the LED0 On timer callback. Verify that after each message, the clock data struct holds correct values for the period and duty cycle.
3. Initialize the LED1 control task, the LED1 data struct and the LED1 control task message queue. Create a mock task that runs on a periodic timer and sends known messages to the control task. Create a spy callback function that will be passed into the LED1 On timer callback. Verify that after each message, the clock data struct holds correct values for the period and duty cycle.