Code Narrative:

1. Configuration
   1. Reset peripheral registers
   2. Enable System Clock
   3. Enable peripherals RCC
   4. Configure peripherals (USART, Timers, GPIO pins)
2. Interrupt Handlers
   1. SysTick to increment 1ms system clock (used on buzzers, watchdog/heartbeat monitor)
   2. Timers (Get Motor Speed)
   3. Encoders (Get position)
   4. Proxy Sensor
   5. USART (HMI commands)
3. Sequences
   1. USART Read Buffer
      1. Read in characters
      2. Compare characters to list of commands (switch logic)
      3. Execute command
   2. USART Write
      1. Send full string for each command
   3. PID Controller
      1. Check position error
      2. Calculate PID correction
      3. Convert to PWM duty cycle
      4. Check max speed requirement
      5. Set duty cycle
   4. HMI request data
      1. HMI will request updates on position every 10-250ms in order to refresh monitored data
      2. MCU will send the data in packets
   5. Calibration/Homing
      1. HMI sends request to calibrate position
      2. Desk lowers until proxy sensors are detected (no commands available during calibration)
      3. MCU sends signal to HMI saying calibration is complete
      4. A buzzer also sounds to alert that calibration has completed
   6. Manual Control
      1. While holding down Up or Down buttons on HMI, the Desk will move in selected direction. Once released, the HMI will send a stop command to the MCU.
      2. If the Heartbeat is lost, the MCU immediately stops moving the desk.
   7. Maintenance Mode
      1. In advanced settings, each leg can be moved independently.
4. Faults (indications and sounds, all motion stops until fault is cleared, faults latch)
   1. Reverse Polarity (encoder counting wrong way meaning motor leads are backwards)
   2. Desk level too large (one leg is too high/low)
   3. Comms fault (HMI lost signal)
   4. No motion fault (watchdog timer for motors)
   5. Proxy Fault (unexpected loss of proxy sensor when not calibrating)

// Initialize Peripherals

SysTick\_Init(); // Enable System Clock and set priority

GPIO\_Init(); // Enables GPIO RCC

LED\_RED\_Init(); // Enable Red LED for debugging

LED\_BLU\_Init(); // Enable Blue LED for debugging

LED\_ORG\_Init(); // Enable Orange LED for debugging

LED\_GRN\_Init(); // Enable Green LED for debugging

PWM\_Init(); // Initialize PWM

Encoder\_Init(); // Initialize Encoder Pins

Buzzer\_Init(); // Initialize Buzzer Pin

TIM1\_Init(); // Timer 1 Configure

TIM2\_Init(); // Timer 2 Configure

USART1\_Init(); // Configure/enable USART1

// Each motor has PWM control (with limited time, only doing a PID controller)

PWM\_SetDutyCycle(int16\_t pwm\_dutycycle); // Update the duty cycle

PID\_Update(); // Position Controller

// General Commands

Set\_Tone(int16\_t tone); // Set buzzer tone using PWM

Buzzer\_On(); // Set Buzzer Off

Buzzer\_Off(); // Set Buzzer On

UART\_Read();

UART\_Write();

UART\_WriteString();

// Interrupts

SysTick\_Handler() // System Clock increment 1ms