**Barcode**

|  |  |
| --- | --- |
| **Function** | **Use** |
| code\_39\_decoder | Takes in the odd and even bits to decode the barcode, allocate the checksum and determine char based on predefined values, then return the char |
| determine\_odd\_bit | Takes in 5 odd bit timings to find the 2 highest and convert into ‘1’s while other bits will be ‘0’s |
| determine\_even\_bit | Takes in 4 even bit timings to find the highest and convert into ‘1’ while other bits will be ‘0’s |
| determine\_char | Gets one char at a time and keep track of the index of character being received. If the first char is the flipped delimiter, it will inform determine\_odd\_bit and determine\_even\_bit to flip the bits in the future conversion. |
| barcode\_task | Read the time taken for each bits within the 9 bits, always starting with black bit. On first black, will start count until it turns white and vice versa, until it reaches 9 bits.  Upon reaching 9 bits, 5 odd bits timings will be passed to determine\_odd\_bit and 4 even bits timings will be passed to determine\_even\_bit.  Thereafter, the converted bit will be passed to code\_39\_decoder which will decode the value and lastly, the decoded value will be passed to determine\_char. |

**initialize\_pins**

|  |  |
| --- | --- |
| **Function** | **Use** |
| motor\_control\_init | Initialize the GP pins required for motor control and set default PWM values of the motor |
| encoder\_init | Initialize the GP pins required for wheel syncing (speed) and pre-set amount of wheel turns |
| infrared\_init | Initialize the GP pins required for barcode and wall detection (infrared) |
| ultrasonic\_init | Initialize the GP pins required for ultrasonic to generate sound and receive sound |

**ultrasonic**

|  |  |
| --- | --- |
| **Function** | **Use** |
| calculate\_distance | Based on time provided, calculate the distance travelled |
| generate\_ultrasonic\_task | Generate Ultrasonic sound every 100 milliseconds for interrupt callback to trigger |

**Interrupts\_tasks**

|  |  |
| --- | --- |
| **Function** | **Use** |
| interrupt\_callback | Handles GPIO interrupts for various sensors (right and left encoders, infrared sensors, barcode infrared sensor, ultrasonic sensor).  Left and right encoder count increases based on encoder interrupts.  Infrared rising edges will determine if there is a wall on the left, right or front.  Ultrasonic rising and falling edge will determine the time and check if there is an object in front of the car |
| disable\_all\_interrupts | Disables interrupts for all sensors by calling gpio\_set\_irq\_enabled\_with\_callback for each sensor with the callback function interrupt\_callback. |
| enable\_all\_interrupts | Enables interrupts for all sensors by calling gpio\_set\_irq\_enabled\_with\_callback for each sensor with the callback function interrupt\_callback. |
| enable\_encoder\_interrupts | Enables interrupts for the left and right encoders using gpio\_set\_irq\_enabled\_with\_callback with the callback function interrupt\_callback. |
| enable\_infrared\_interrupts | Enables interrupts for left and right infrared sensors using gpio\_set\_irq\_enabled\_with\_callback with the callback function interrupt\_callback. |
| enable\_ultrasonic\_interrupts | Enables interrupts for the ultrasonic sensor using gpio\_set\_irq\_enabled\_with\_callback with the callback function interrupt\_callback. |
| start\_tasks | Initializes and starts various FreeRTOS tasks responsible for controlling the car, including main tasks, a web server task, tasks for handling duty cycle selection, synchronization of wheel tasks, straight path execution, left and right wheel control, ultrasonic sensor simulation, and barcode scanning. |

**motor\_control**

|  |  |
| --- | --- |
| **Function** | **Use** |
| left\_wheel\_forward | Move left wheel forward |
| right\_wheel\_forward | Move right wheel forward |
| left\_wheel\_backward | Move left wheel backward |
| right\_wheel\_backward | Move right wheel backward |
| left\_wheel\_stop | Stop left wheel |
| right\_wheel\_stop | Stop right wheel |
| set\_wheel\_duty\_cycle | Set the duty cycle of the motor |
| sync\_wheel\_task | Runs the wheel for 10 seconds and get the ratio to adjust wheel speed to accordingly to ratio |
| choose\_duty\_cycle | Prompt user for the duty cycle they want the car to move at |
| left\_wheel\_task | Move left wheel accordingly to direction received based on left encoder count before stopping |
| right\_wheel\_task | Move right wheel accordingly to direction received based on right encoder count before stopping |
| send\_buffer\_to\_wheels | Send the buffer message to the wheels on direction to move |
| straight\_path\_task | Move the vehicle forward until a wall, object or 5 seconds passed, whichever comes first. Then stop the vehicle. |