| Test Protocol | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | **Test Case Description** | **Test Case Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| **MCAL Module** | | | | | |
| *GPIO Driver* | | | | | |
| *TC_GPIO_001* | Test GPIO_init | create struct that holds all pin configs and send it to DIO_init_pin(&structct) | the pin initializes correctly | *Matches Expected Result* | Pass |
| *TC_GPIO_002* | Test GPIO_write | send port and pin and level to GPIO_write(port,pin,level) | the level is set on pin correctly | *Matches Expected Result* | Pass |
| *TC_GPIO_003* | Test GPIO_toggle | send port and pin to GPIO_toggle(port,pin) | the status of pin is toggled correcty | *Matches Expected Result* | Pass |
| *TC_GPIO_004* | Test GPIO_read | send port and pin and address of variable to GPIO_read(port,pin,&value) | the status stored in variable correctly | *Matches Expected Result* | Pass |
| *TC_GPIO_005* | Test GPIO_enable_interrupt | send port and pin to GPIO_enable_interrupt(port,pin) | the interrupt enabled correctly | *Matches Expected Result* | Pass |
| *TC_GPIO_006* | Test GPIO_disable_interrupt | send port and pin to GPIO_disable_interrupt(port,pin) | the interrupt ddisabled correctly | *Matches Expected Result* | Pass |
| *GPT Driver* | | | | | |
| *TC_GPT_001* | Test GPT_init | calling the GPT_init() | All Configration Initialize Succesful | *Matches Expected Result* | Pass |
| *TC_GPT_002* | Test start_time_ms | calling the GPT_start_time_ms() and required delay | provide the required delay in milliseconds. | *Matches Expected Result* | Pass |
| *TC_GPT_003* | Test start_time_us | calling the GPT_start_time_us() and passing the required delay | provide the required delay in microseconds. | *Matches Expected Result* | Pass |
| *TC_GPT_004* | Test elabsed_time | calling the GPT_elapsed_time(), send pointer to var to store the elapsed time. | store the elapsed timer | *Matches Expected Result* | Pass |
| *TC_GPT_005* | Test remaining_time | calling the GPT_remaining_time(), send pointer to var to store the remaining ti | store the remaining time | *Matches Expected Result* | Pass |
| *TC_GPT_006* | Test enable_interrupt | calling the GPT_enable_interrupt(), and send the timer channel id | enable the interrupt. | *Matches Expected Result* | Pass |
| *TC_GPT_007* | Test disable_interrupt | calling the GPT_disable_interrupt(), and send the timer channel id | disable the interrupt. | *Matches Expected Result* | Pass |
| **HAL Module** | | | | | |
| *Button Driver* | | | | | |
| *TC_BTN_001* | Intialize Push Button | Call BUTTON_init To Initialize Push Button | All Configration Initialize Succesful | *Matches Expected Result* | Pass |
| *TC_BTN_002* | Get Push Button Status | Call BUTTON_read To Get Its Status Pressed Or Relased | Push Button Status Returned Succesful | *Matches Expected Result* | Pass |
| *LED Driver* | | | | | |
| *TC_LED_001* | Test LED_init | call LED_init | all LEDS initialized correctly | *Matches Expected Result* | Pass |
| *TC_LED_002* | Test LED_on | call LED_on and pass led id | the led turned on | *Matches Expected Result* | Pass |
| *TC_LED_003* | Test LED_off | Call LED_off and pass led id | the led turned off | *Matches Expected Result* | Pass |
| *TC_LED_004* | Test LED_toggle | Call LED_toggle and pass led id | the led toggled | *Matches Expected Result* | Pass |
| *T_Handler Driver* | | | | | |
| *TC_Handler_001* | Test Handler_init | calling the Handler_init() | All Configration Initialize Succesful | *Matches Expected Result* | Pass |
| *TC_Handler_002* | Test Handler_start_time_ms | calling the Handler_start_time_ms() and required delay | provide the required delay in milliseconds. | *Matches Expected Result* | Pass |
| *TC_Handler_003* | Test Handler_start_time_us | calling the Handler_start_time_us() and passing the required delay | provide the required delay in microseconds. | *Matches Expected Result* | Pass |
| *TC_Handler_004* | Test Handler_elabsed_time | calling the Handler_elapsed_time(), send pointer to var to store the elapsed tir | store the elapsed timer | *Matches Expected Result* | Pass |
| *TC_Handler_005* | Test Handler_remaining_time | calling the Handler_remaining_time(), send pointer to var to store the remaini | store the remaining time | *Matches Expected Result* | Pass |
| *TC_Handler_006* | Test Handler_enable_interrupt | calling the Handler_enable_interrupt(), and send the timer channel id | enable the interrupt. | *Matches Expected Result* | Pass |
| *TC_Handler_007* | Test Handler_disable_interrupt | calling the Handler_disable_interrupt(), and send the timer channel id | disable the interrupt. | *Matches Expected Result* | Pass |
| *DCM Driver* | | | | | |
| *TC_DCM_001* | Test DCM_motorinit | 1. Call DCM_motorinit | The motor pins are initialized to be output | *Matches Expected Result* | Pass |
| *TC_DCM_002* | Test DCM_changeDiresction | 1. Call DCM_changeDirection | The motor direction is changed | *Matches Expected Result* | Pass |
| *TC_DCM_003* | Test DCM_vdStop | 1. Call DCM_vdStop | The motors are stopped | *Matches Expected Result* | Pass |
| *TC_DCM_004* | Test DCM_setDutyCycle | 1. Call DCM_setDutyCycle | The duty Cycle is passed to the PWM function | *Matches Expected Result* | Pass |
| *TC_DCM_005* | Test DCM_rotate | 1. Call DCM_rotate | The robot will rotate | *Matches Expected Result* | Pass |
| *PWM_ Driver* | | | | | |
| *TC_PWM_001* | Test PWM_init | calling the PWM_init() | All Configration Initialize Succesful | *Matches Expected Result* | Pass |
| *TC_PWM_002* | Test PWM_set_duty_cycle | calling the PWM_set_duty_cycle() | provide the required duty cycle for motors. | *Matches Expected Result* | Pass |
| **Application** | | | | | |
| *APP* | | | | | |
| *TC_APP_001* | initialize all Hal Modules | Call led_init and button_init | all modules initializes correctly | *Matches Expected Result* | Pass |
| *TC_APP_002* | Run main Logic of application | implement main logic in super loop | app works fine and covered all known cases | *Matches Expected Result* | Pass |
| *User Stories* | | | | | |
| *TC_4WD_APP_001* | Intializing all the modules | power up the system | All modules are initialized correctly | *Matches Expected Result* | Pass |
| *TC_4WD_APP_002* | press stop before start | first press of sw1 | Nothing happens | *Matches Expected Result* | Pass |
| *TC_4WD_APP_003* | press stop after start | second press of sw1 | Sudden break will occure | *Matches Expected Result* | Pass |