Robot Static design

(Heba Ramadan Taha)

Project description:

1- Initially Robot will be in Power up mode waiting for select one of movement modes based on Mode_Buttons:

- Button_1 make Robot in Autonomous mode.
- Button_2 make Robot in PC mode.

2- In Autonomous Mode:

Robot move Forward and Monitor module using Ultrasonic to measure distance between Robot and objects, Based on measured distance Robot control decide next direction to move and speed of motor:

- no object -> Move in Normal direction (Forward) with Normal speed (80%)
- object detected (distance > 50cm) ->Move in Normal direction (Forward) with slow speed(30%)
- object detected (50 >= distance > 30) -> Change direction (Stop Robot and Turn right) with slow speed(30%)
- object detected (distance <= 30cm) -> Move back till distance is equal to 30cm and Change direction with slow speed (30%)

3- In PC Mode:

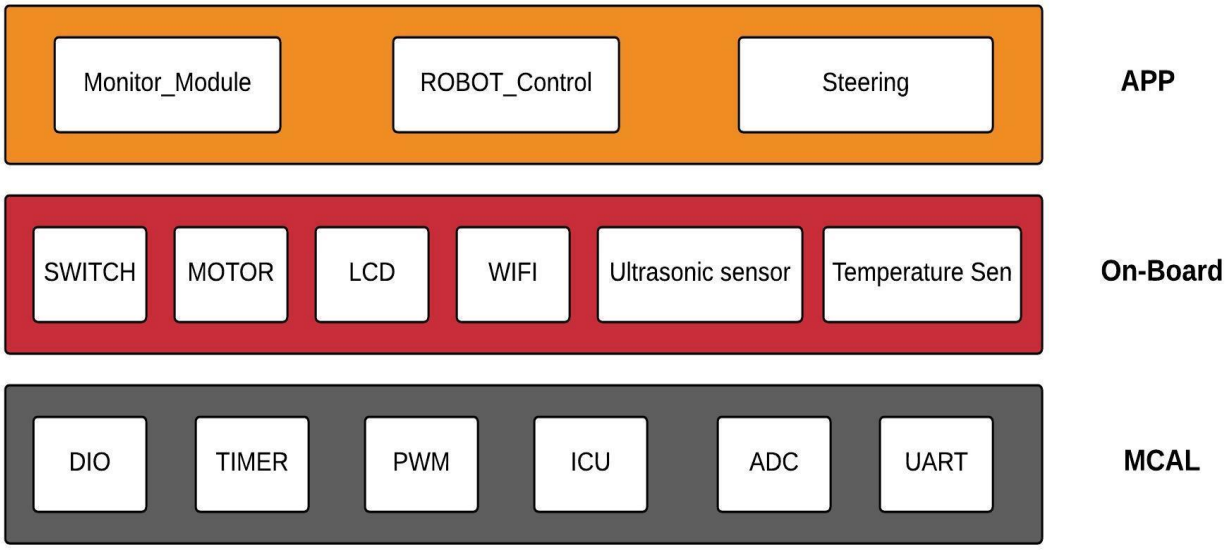
PC will control robot and send directions CMD using wifi and Robot receive data through UART, in this case Monitor module will take data and check cmd, based in CMD received Robot control decide which direction to move:

- F -> Forward_CMD
- B ->Back_CMD
- R ->Right_CMD
- L ->Left_CMD
- S ->Stop_CMD

4- In Autonomous or PC mode Robot send Current Temperature periodically every 5sec to PC.

5- Movement mode (Autonomous or PC) not change in run time, to change it restart Robot.

Layered Architecture:



1- MCAL Layer

A. DIO APIs:

Function Name	DIO_eSetPinDirection(PinId_t PinIdCpy , PinDir_t PinDirCpy)		
Arguments	Inputs	PinIdCpy	enumeration
		description: The pin number to set direction	
		PinDirCpy	enumeration
		description: The direction of pin as input or output	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to set pin direction as input or output		

Name	PinIdCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . . DIO_PIN31	0 1 . . . 31	For pin 0 For pin 1 . . . For pin 31
Description	These values are to determine which pin in MC to be affected by the function		

Name	PinDirCpy		
Type	enumeration		
Rang	DIO_OUTPUT	1	To be output
	DIO_INPUT	0	To be input
Description	These values are to determine the direction of pin as output or input		

Function Name	DIO_eSetPinValue(PinId_t PinIdCpy , PinVal_t PinValCpy)		
Arguments	Inputs	PinIdCpy	enumeration
		description: The pin number to set value	
		PinValCpy	enumeration
		description: The direction of pin as high or low	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to set pin value high or low		

Name	PinIdCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . . DIO_PIN31	0 1 . . . 31	For pin 0 For pin 1 . . . For pin 31
Description	These values are to determine which pin in MC to be affected by the function		

Name	PinValCpy		
Type	enumeration		
Rang	DIO_HIGH	1	To make pin high
	DIO_LOW	0	To make pin low
Description	These values are to determine the value of pin as high or low		

Function Name	DIO_eGetPinValue(PinId_t PinIdCpy , u8 * pPinVal)		
Arguments	Inputs	PinIdCpy	enumeration
		description: The pin number to get value	
	Outputs	pPinVal	u8 *
		description: pointer to location which save value	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get pin value high or low		

Name	PinIdCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . . DIO_PIN31	0 1 . . . 31	For pin 0 For pin 1 . . . For pin 31
Description	These values are to determine which pin in MC to be affected by the function		

B. Timer APIs:

Function Name	TIMER_eInit(ChannelId_t ChIdCpy)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID to initialize as Timer	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Initialize Timer		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Function Name	TIMER_eStart(ChannelId_t ChIdCpy , u16 TimeCountCpy)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID to Start Timer	
		TimeCountCpy	u16
		description: Number of counts in ms	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Start timer and count from 0 till number of counts in msec		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Name	TimeCountCpy
Type	u16
Rang	{1 , …. , 50000}
Description	To determine which timer to be affected by the function

Function Name	TIMER_eStop(ChannelId_t ChIdCpy)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID of Timer	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to stop Timer		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Function Name	TIMER_eGetStatus(ChannelId_t ChIdCpy , TimStat_t* pTimerState)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID to initialize as Timer	
	Outputs	pTimerState	TimStat_t*
		description: return state of timer	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get current state of timer		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Name	pTimerState		
Type	enumeration		
Rang	TIMER_RUNNIG	0	
	TIMER_STOP	1	
	TIMER_EXPIRED	2	
Description	To describe current state of timer		

C. PWM APIs:

Function Name	PWM_eInit(ChannelId_t ChIdCpy)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID of timer to initialize as PWM	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Initialize PWM module		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Function Name	PWM_eStart (ChannelId_t ChIdCpy ,u8 DutyCycleCpy , u32 FreqCpy)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID to start PWM	
		DutyCycleCpy	u8
		description: the duty cycle of the signal	
		FreqCpy	u32
		description: the frequency of the signal	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Start PWM module		

Name	DutyCycleCpy	
Type	u8	
Rang	0	Min value of duty
	100	Max value of duty
Description	Duty cycle of pwm signal	

Name	FreqCpy	
Type	u32	
Rang	0	Min value of Freq
	100000	Max(10usec period)
Description	Frequency of pwm signal	

Name	ChIdCpy		
Type	enumeration		
Range	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Function Name	PWM_eStop(ChannelId_t ChIdCpy)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID of timer	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Stop PWM module		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

D. ICU APIs:

Function Name	ICU_eInit(ChannelId_t ChIdCpy)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID of timer to initialize as ICU	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Initialize ICU module		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Function Name	ICU_eGetTime(ChannelId_t ChIdCpy , u32 * pMeasureTime)		
Arguments	Inputs	ChIdCpy	enumeration
		description: The Channel ID of timer to initialize as ICU	
	Outputs	pMeasureTime	u32 *
		description: return Measured time	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get Measured time between rising edge and falling edge which detected be ICU		

Name	ChIdCpy		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

E. ADC APIs:

Function Name	ADC_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Initialize ADC module		

Function Name	ADC_eGetResult(ChannelId_t ChIdCpy , u16 * pResult)		
Arguments	Inputs	ChIdCpy	
		description: ADC hardware channel	
	Outputs	pResult	
		description: return digital result	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Initialize ADC module		

Name	ChIdCpy		
Type	enumeration		
Range	ADC_CHANNEL_0	0	For chID0
	ADC _CHANNEL_1	1	For chID1
	ADC _CHANNEL_2	2	For chID2
	ADC _CHANNEL_3	3	For chID3
	ADC _CHANNEL_4	4	For chID4
	ADC _CHANNEL_5	5	For chID5
	ADC _CHANNEL_6	6	For chID6
	ADC _CHANNEL_7	7	For chID7
Description	To determine which ADC hardware channel used to be affected by the function		

Name	pResult	
Type	u16	
Rang	0	Min value of digital result
	1023	Max(10 bit resolution adc)
Description	ADC digital value	

F. UART APIs:

Function Name	UART_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Initialize UART module		

Function Name	UART_eSendByte(u8 ByteCpy)		
Arguments	Inputs	ByteCpy	u8
		description: Byte which sent by uart	
	Outputs	N/A	
		description: return digital result	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to send byte using module		

Name	ByteCpy	
Type	u8	
Rang	0	Min value to send
	255	Max value to send
Description	Byte which sent by uart	

Function Name	UART_eReceiveByte(u8 * pByteReceived)		
Arguments	Inputs	N/A	
		description:	
	Outputs	pByteReceived	u8 *
		description: return digital result	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Receive byte using uart module		

2- On-Board Layer

A. LCD:

Function Name	LCD_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize LCD module		

Function Name	LCD_eSendCommand(Cmd_t u8cmdcpy)		
Arguments	Inputs	u8cmdcpy	enumeration
		description: a copy of the command to send to the lcd	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to send command to set LCD		

Name	u8cmdcpy	
Type	enumeration	
Rang	LCD_8BIT	0
	LCD_DISPLAY_ON	1
	LCD_DISPLAY_OFF	2
	LCD_CLEAR	3
	LCD_ENTRY_MODE	4
Description	These values are the commands to be sent to lcd .	

Function Name	LCD_eSendChar(u8 u8charcpy)		
Arguments	Inputs	u8charcpy	u8
		description: a copy of the Data to send on the lcd	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to send Data on LCD		

Name	u8charcpy
Type	u8
Rang	{0 ,….,127}
Description	The decimal representation of ASCII code.

Function Name	LCD_eSetPosition(u8 u8ColCpy , u8 u8RowCpy)		
Arguments	Inputs	u8ColCpy	u8
		description: the horizontal position starting from 0:15 for 2x16 lcd	
		u8RowCpy	u8
		description: the vertical position (0:1) for 2x16 lcd	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to go to specific position on the lcd		

Name	u8ColCpy	
Type	u8	
Rang	0	The first position in the screen starting from left
	15	The last position in the screen starting from left
Description	These values are the horizontal positions in a 2x16 LCD.	

Name	u8RowCpy	
Type	u8	
Rang	0	The first row in the screen starting from upper row
	1	The second row in the screen starting from upper row
Description	These values are the vertical positions in a 2x16 LCD	

B. Switch:

Function Name	Switch_eInit(SwitchID_t SwitchIdcpy)		
Arguments	Inputs	SwitchIdcpy	enumeration
		description: Id of switch	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize switch module by initialize dio pin as input pull up		

Name	SwitchIdcpy		
Type	enumeration		
Rage	SWITCH_1	1	For Switch 1 which mapped to dio pin as configured
	SWITCH_2	2	For Switch 2 which mapped to dio pin as configured
Description	These values are to determine which pin in MC to be affected by the function		

Function Name	Switch_eGetStatus(SwitchID _t SwitchIdcpy, Switch_State* pSwitchState)		
Arguments	Inputs	SwitchIdcpy	enumeration
		description: Dio pin for switch	
	Outputs	pSwitchState	Switch_State *
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get current status of specific switch to know it pressed or not		

Name	SwitchIdcpy		
Type	enumeration		
Rage	SWITCH_1	1	For Switch 1 which mapped to dio pin as configured
	SWITCH_2	2	For Switch 2 which mapped to dio pin as configured
Description	These values are to determine which pin in MC to be affected by the function		

Name	Switch_State	
Type	enumeration	
Rang	0	PRESSED
	1	NOT_PRESSED
Description	These values are Status of switch to indicate that switch pressed or not	

C. MOTOR:

Function Name	MOTOR_eInit(PinId_t MotorIdCpy , PinId_t SpeedPincpy, ChId_t PWM_Ch)		
Arguments	Inputs	MotorIdCpy	enumeration
		description: The motor existence dio pin	
		SpeedPincpy	enumeration
		description: The motors peed dio pin	
		PWM_Ch	enumeration
		description: The PWM channel dio pin	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize Motor module (initialize dio pin as output and pwm_Init() with specific channel of timer)		

Name	MotorIdCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . . DIO_PIN31	0 1 . . . 31	For pin 0 For pin 1 . . . For pin 31
Description	These values are to determine which pin in MC to be affected by the function		

Function Name	MOTOR_eStartt(PinId_t MotorIdCpy , u8 MotorSpeed, ChId_t PWM_Ch)		
Arguments	Inputs	MotorIdCpy	enumeration
		description: The motor existence dio pin	
		MotorSpeed	u8
		description: The motors peed dio pin	
		PWM_Ch	enumeration
		description: The PWM channel dio pin	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to start Motor with specific speed		

Name	MotorIdCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . DIO_PIN31	0 1 . . 31	For pin 0 For pin 1 . . For pin 31
Description	Specific dio pin for motor		

Name	MotorSpeed	
Type	u8	
Rang	0	Min speed
	80	Max speed for robot
Description	Speed of motor	

Function Name	MOTOR_eStop(PinId_t MotorIdCpy)		
Arguments	Inputs	MotorIdCpy	enumeration
		description: The motor existence dio pin	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to stop Motor module		

Name	MotorIdCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . . DIO_PIN31	0 1 . . . 31	For pin 0 For pin 1 . . . For pin 31
Description	These values are to determine which pin in MC to be affected by the function		

D. Temperature Sensor :

Function Name	Temp_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize Temperature sensor module		

Function Name	Temp_eGeTempValue(u8* pTempValue)		
Arguments	Inputs	N/A	
		description:	
	Outputs	pTempValue	u8*
		description: return measured value by temp sensor	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get Temperature value		

E. Ultrasonic Sensor :

Function Name	Ultrasonic_eInit(PinId_t TriggerCpy, PinId_t EchoCpy, ChId_t ICU_Ch)		
Arguments	Inputs	TriggerCpy	enumeration
		description: Trigger DIO pins as output	
		EchoCpy	enumeration
		description: Echo DIO pins as input	
		ICU_Ch	enumeration
		description: specific channel of timer as ICU	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize Ultrasonic sensor module		

Name	TriggerCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . DIO_PIN31	0 1 . . 31	For pin 0 For pin 1 . . For pin 31
Description	Specific dio pin for Trigger		

Name	EchoCpy		
Type	enumeration		
Rang	DIO_PIN0 DIO_PIN1 . . DIO_PIN31	0 1 . . 31	For pin 0 For pin 1 . . For pin 31
Description	Specific dio pin for Echo		

Name	ICU_Ch		
Type	enumeration		
Rang	TIMER_CHANNEL_0	0	For Timer0
	TIMER_CHANNEL_1	1	For Timer1
	TIMER_CHANNEL_2	2	For Timer2
Description	To determine which timer to be affected by the function		

Function Name	Ultrasonic_eSendTrigger(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	pTempValue	u8*
		description: return measured value by temp sensor	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get Temperature value		

Function Name	Ultrasonic_eGeDistanceValue(u16* pDistanceValue)		
Arguments	Inputs	N/A	
		description:	
	Outputs	pDistanceValue	u16*
		description: return measured value by temp sensor	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get distance value in cm		

F. Wifi Module:

Function Name	ESP_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize ESP module		

Function Name	ESP_eConnectToWifi(u8* NetworkName , u8* NetworkPass)		
Arguments	Inputs	NetworkName	u8*
		description:	
		NetworkPass	u8*
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to connect ESP module with wifi network		

Function Name	ESP_eConnectToServer(u8* ServerIP , u8* ServerPort)		
Arguments	Inputs	ServerIP	u8*
		description:	
		ServerPort	u8*
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to connect ESP module with Server		

Function Name	ESP_eSendByteToPC(u8 ByteCpy)		
Arguments	Inputs	ByteCpy	u8
		description: data to send	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to Send data from MC to ESP through uart		

Function Name	ESP_eGetReceivedByteFromPC(u8* ByteCpy)		
Arguments	Inputs	N/A	
		description: data to send	
	Outputs	ByteCpy	u8*
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get data which received from PC to ESP through uart		

3- APP Layer

A. Monitor Module:

Function Name	Monitor_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize the monitor Robot module and the needed other modules to observe Robot status in Autonomous and PC mode		

Function Name	Monitor_eSendTemp(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API periodically and check if Timer count 5sec to start sending current measured temperature by temp_sensor to PC. This function called in Autonomous and PC mode.		

Function Name	Monitor_eMainFunction(ModState_t RobotModeCpy)		
Arguments	Inputs	RobotModeCpy	emumeration
		description: State of Robot mode	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to periodically to update Robot status in Autonomous or PC mode based on mode status -in Autonomous mode detect object using ultrasonic module -in PC mode get CMD which received from Wifi module		

Name	RobotModeCpy		
Type	enumeration		
Rang	AUTONOMOUS_MODE	0	For Robot mode 0
	PC_MODE	1	For Robot mode 1
Description	To determine which mode to be affected by the function		

Function Name	Monitor_eGetStatus(MonitorState_t * pMonitorState)		
Arguments	Inputs	N/A	
		description:	
	Outputs	pMonitorState	MonitorState_t *
		description: pointer to Status of monitor module	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get current monitor status, this function called by Robot_eMainFunction Api.		

Name	pMonitorState		
Type	enumeration		
Rang	NO_OBJECT	0	For Autonomous_Mode and no object detected
	FAR_OBJECT	1	for Autonomous _Mode and distance between Robot and object > 50cm
	NEAR_OBJECT	2	for Autonomous _Mode and distance between Robot and object < 50cm and > 30
	VERY_NEAR_OBJECT	3	for Autonomous _Mode and distance between Robot and object < 30cm
	F_CMD	4	for PC_Mode forward cmd
	B_CMD	5	for PC_Mode back cmd
	R_CMD	6	for PC_Mode right cmd
	L_CMD	7	for PC_Mode left cmd
	S_CMD	8	for PC_Mode stop cmd
Description	To determine which monitor state to be returned by the function		

B. Robot Module:

Function Name	Robot_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize the Robot Control module and the needed other modules		

Function Name	Robot_eGetMode(RobotMode_t * pRobotMode)		
Arguments	Inputs	N/A	
		description:	
	Outputs	pRobotMode	RobotMode_t *
		description:	
	Input/Output	N/A	
		description:	
Return	POWER_UP	0	
	RUNNING	1	
	E_NOK		
Description	Call this API get Robot Mode		

Name	pRobotMode		
Type	Pointer to enum		
Rang	AUTONOMOUS_MODE	0	For Robot mode 0
	PC_MODE	1	For Robot mode 1
Description	To determine which mode to be affected by the function		

Function Name	Robot_eMainFunction(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to periodically to get Robot status in Autonomous or PC mode from monitor module.		

Function Name	Robot_eGetMovement(Dir_t * pDirection, Speed_t * pSpeed)		
Arguments	Inputs	N/A	
		description:	
	Outputs	pDirection	Dir_t *
		description:	
		pSpeed	Speed_t *
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to get Robot movement status in Autonomous or PC mode.		

Name	pDirection		
Type	Pointer to enum		
Rang	NORMAL_DIR	0	For forward
	CHANGE_DIR	1	For Stop then move right
	BACK_DIR	2	For back
Description	To determine which direction to be returned by the function		

Name	pSpeed		
Type	Pointer to enum		
Rang	NORMAL_SPEED	0	For 80% of speed
	SLOW_SPEED	1	For 30% of speed
Description	To determine which speed to be returned by the function		

C. Steering Module:

Function Name	Steer_eInit(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API to initialize the steer Control module and the needed other modules		

Function Name	Steer_eUpdateMovement(void)		
Arguments	Inputs	N/A	
		description:	
	Outputs	N/A	
		description:	
	Input/Output	N/A	
		description:	
Return	E_OK	0	
	E_NOK	1	
Description	Call this API periodically and call Robot_eGetMovement API to update movement direction and speed of robot.		