Rain Sensing Automatic Car Wiper Using STM32 Microcontroller

Implementation…

1. Code

```
#include <stdint.h>
#include "stm324xx.h"
#include "gpio.h"
#include "rcc.h"

#if !defined(__SOFT_FP__) && defined(__ARM_FP)
  #warning "FPU is not initialized, but the project is compiling for an FPU. Please initialize the FPU before use."
#endif

/* RCC (Reset and clock Control) Registers */
#define RCC_BASE_ADDR              0x40023800
#define RCC_AHB1ENR_OFFSET  0x30
#define RCC_AHB1ENR                *(volatile unsigned int *)(RCC_BASE_ADDR + RCC_AHB1ENR_OFFSET)

/* GPIO Registers */
#define GPIOD_BASE_ADDR            0x40020C00
#define GPIOD_MODER_OFFSET  0x00
#define GPIOD_MODER                *(volatile unsigned int *)(GPIOD_BASE_ADDR + GPIOD_MODER_OFFSET)
#define GPIOD_ODR_OFFSET    0x14
#define GPIOD_ODR                  *(volatile unsigned int *)(GPIOD_BASE_ADDR + GPIOD_ODR_OFFSET)

static volatile int flag;
static volatile int SCount = 0, count=0;        //Global variables

        //Scount is number of times the switch is pressed

        //count is simple variable to count the loop

//Interrupt handler
void exti0_irqhandler(void)
{
       if(EXTI->PR & (1 << PIN_0))
       {
              flag = 1;

              EXTI->PR = EXTI->PR | (1 << PIN_0);
       }
}

//Code to Turn on and off Red LED
long delay = 0xFFFFFFF;

static void on_red_led(void)
{
       volatile long i;
       GPIOD_ODR = GPIOD_ODR | 0x00001000;

}
static void off_red_led(void)
{
       volatile long i;
       GPIOD_ODR = GPIOD_ODR & ~(0x00001000);
}
```

```
53.
54. //GPIO Configuration
55. static void init_config(void)
56. {
57.         config_rcc(GPIOD);
58.         config_gpiox(GPIOD, PIN_12, GPIO_OUTPUT_PP, GPIO_SPEED_VERY_HIGH);
          //Green LED
59.
60.         config_rcc(GPIOD);
61.         config_gpiox(GPIOD, PIN_14, GPIO_OUTPUT_PP, GPIO_SPEED_VERY_HIGH);
          //Red LED
62.
63.         config_rcc(GPIOD);
64.         config_gpiox(GPIOD, PIN_13, GPIO_OUTPUT_PP, GPIO_SPEED_VERY_HIGH);
          //Orange LED
65.
66.         config_rcc(GPIOD);
67.         config_gpiox(GPIOD, PIN_15, GPIO_OUTPUT_PP, GPIO_SPEED_VERY_HIGH);
          //Blue LED
68.
69.         config_rcc(GPIOA);
70.         config_gpiox(GPIOA, PIN_0, GPIO_INPUT, GPIO_SPEED_VERY_HIGH);
          //Push Button
71.
72.         config_gpio_irq_priority(IRQ_NO_EXTI0, NVIC_IRQ_PR_LVL_0);
          //Interrupt Handler
73.         config_gpio_interrupt(IRQ_NO_EXTI0, ENABLE);
74. }
75.
76. // Code to Turn on and off Blue LED
77. static void LED_1_ON()
78. {
79.         /* Setting PD15 (Pin 15 of PORTD) as General Purpose Output */
80.         GPIOD_MODER = GPIOD_MODER | 0x01000000000;
81.
82.         GPIOD_ODR = GPIOD_ODR | 0x00001000;
83. }
84.
85. static void LED_1_OFF()
86. {
87.         /* Setting PD15 (Pin 15 of PORTD) as General Purpose Output */
88.         GPIOD_MODER = GPIOD_MODER | 0x01000000000;
89.
90.         GPIOD_ODR = GPIOD_ODR & ~0x00001000;
91. }
92. //Code to Turn on and off Green LED
93.
94. static void LED_2_ON()
95. {
96.         /* Setting PD12 (Pin 12 of PORTD) as General Purpose Output */
97.         GPIOD_MODER = GPIOD_MODER | 0x01000000;
98.
99.         GPIOD_ODR = GPIOD_ODR | 0x00001000;
100.        }
101.
102.        static void LED_2_OFF()
103.        {
104.                /* Setting PD12 (Pin 12 of PORTD) as General Purpose Output */
105.                GPIOD_MODER = GPIOD_MODER | 0x01000000;
106.
107.                GPIOD_ODR = GPIOD_ODR & ~0x00001000;
108.        }
109.
110.        //Code to Turn on and off Orange LED
111.        static void LED_3_ON()
112.        {
113.                /* Setting PD13 (Pin 13 of PORTD) as General Purpose Output */
```

```
114.              GPIOD_MODER = GPIOD_MODER | 0x010000000;
115.
116.              GPIOD_ODR = GPIOD_ODR | 0x00001000;
117.       }
118.
119.       static void LED_3_OFF()
120.       {
121.              /* Setting PD13 (Pin 13 of PORTD) as General Purpose Output */
122.              GPIOD_MODER = GPIOD_MODER | 0x010000000;
123.
124.              GPIOD_ODR = GPIOD_ODR & ~0x00001000;
125.       }
126.
127.       static void Delay(int x)                        // Delay function for
   LED
128.       {
129.              for(int i=0;i<x;i++);
130.       }
131.
132.       //Main Function
133.       int main(void)
134.       {
135.
136.
137.              int key;
138.              long int A;                                                    //
   variable to check for button pressed duration
139.
140.              init_config();                                                 //
   Configuration Function for GPIO and interrupt
141.
142.
143.       //Logic for Wiper System
144.              while (1)
145.              {
146.                     key = gpiox_read_pin(GPIOA, PIN_0);                     //
   Polling
147.
148.                     if(key == 1)
149.                     {
150.                                   A = 0;
151.                            long int j;
152.                            for(long int j=0;j<=5000000;j++)
153.                            {
154.                                   A++;
155.                            }
156.
157.       if(A>2000000)               //1uSec = 1 clock  hence 2 Sec = 2000000
   Cycle
158.       {
159.
160.                     on_red_led();                           //Turn on Red LED
161.
162.                     if (flag == 1) // Set in ISR
163.                     {
164.                            flag = 0;
165.                            count++;
166.                            SCount=count % 4;
167.                     }
168.
169.
170.       //Wiper Moving at 1 Hz
171.                     while(SCount==1)
172.                     {
173.                     LED_1_ON();
174.                     Delay(10000);
175.                     LED_1_OFF();
```

```
                    LED_2_ON();
                    Delay(10000);
                    LED_2_OFF();
                    LED_3_ON();
                    Delay(10000);
                    LED_3_OFF();
                    LED_2_ON();
                    Delay(10000);
                    LED_2_OFF();
                    LED_1_ON();
                    Delay(10000);
                    LED_1_OFF();
                    }

      /* Wiper Moving at 4 Hz*/
       while(SCount==2)
                    {
                    LED_1_ON();
                    Delay(2500);
                    LED_1_OFF();
                    LED_2_ON();
                    Delay(2500);
                    LED_2_OFF();
                    LED_3_ON();
                    Delay(2500);
                    LED_3_OFF();
                    LED_2_ON();
                    Delay(2500);
                    LED_2_OFF();
                    LED_1_ON();
                    Delay(2500);
                    LED_1_OFF();
                    }

      /*Wiper Moving at 8 Hz */

                    while(SCount==3)
                    {
                    LED_1_ON();
                    Delay(1250);
                    LED_1_OFF();
                    LED_2_ON();
                    Delay(1250);
                    LED_2_OFF();
                    LED_3_ON();
                    Delay(1250);
                    LED_3_OFF();
                    LED_2_ON();
                    Delay(1250);
                    LED_2_OFF();
                    LED_1_ON();
                    Delay(1250);
                    LED_1_OFF();
                    }
      /*Wiper Movement stop*/
                    while(SCount==0)
                    {
                           LED_1_OFF();
                           LED_2_OFF();
                           LED_3_OFF();
                    }

             }

      }

      /*Wiper System turn Off when switch pressed for less than 2 sec*/
```

```
243.                    else if(A<2000000)
244.                    {
245.                           off_red_led();
246.                           LED_1_OFF();
247.                           LED_2_OFF();
248.                           LED_3_OFF();
249.                    }
250.                    }
251.       return 0;
252.       }
```