TEST-2

ESE – 3025

EMBEDDED REAL TIME OPERATING SYSTEMS

Submitted to :

Takis Zourntos

Submitted by:

Shweta(747925)

A. Design a code for task function for blinking red, green and blue LED in sequence with no overlapping colors with output as red LED on and then pause, Green led on and then pause, Blue led on and then pause. Consider only one task function while writing the code.

<u>Solution:</u>

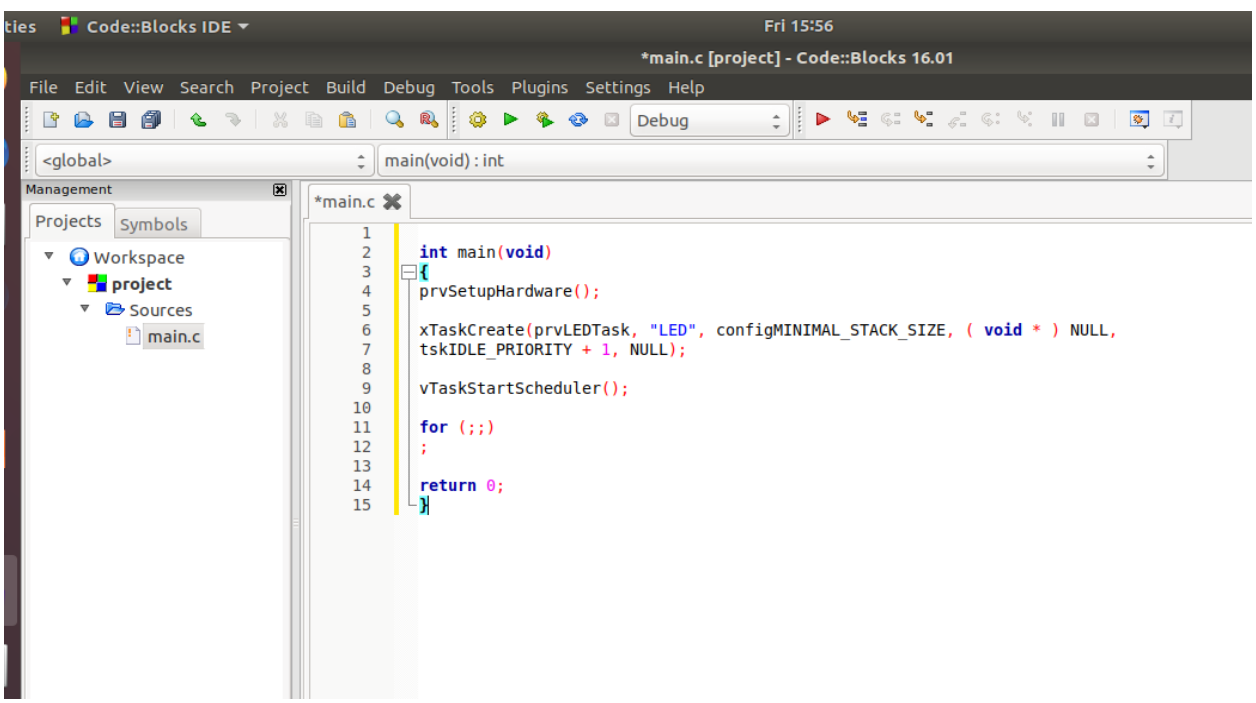
```c
#include "board.h"
#include "FreeRTOS.h"
#include "task.h"

/*
 P0.22-red_LED
 P3.26-blue_led
 P3.25-green_led
 */

#define red_led ( 1 << 22 )
#define blue_led ( 1 << 26 )
#define green_led ( 1 << 25 )

#define red_led_ON      GPIO0->FIOSET = red_led
#define red_led_OFF     GPIO0->FIOCLR = red_led

#define blue_led_ON     GPIO3->FIOSET = blue_led
#define blue_led_OFF    GPIO3->FIOCLR = blue_led

```

```c
#define green_led_ON    GPIO3->FIOSET = green_led
#define green_led_OFF   GPIO3->FIOCLR = green_led

red_led_DIR_OUTPUT GPIO0->FIODIR = red_led
blue_led_DIR_OUTPUT GPIO3->FIODIR = blue_led
green_led_DIR_OUTPUT GPIO3->FIODIR = green_led

/* This is for the blinking rate for the LED. */
#define mainLed_blinking_Rate ( ( portTickType ) 1000 / portTICK_PERIOD_MS )

static void prvLEDTask( void *pvParameters )
{
portTickType xLastRunTime;

xLastRunTime = xTaskGetTickCount();

/*This is to set LEDs to output*/
red_led_DIR_OUTPUT;
blue_led_DIR_OUTPUT;
green_led_DIR_OUTPUT;
```

```c

/*This is to off all leds in beginning*/
red_led_OFF;
blue_led_OFF;
green_led_OFF;

for(;;)
{
red_led_ON;
vTaskDelayUntil( &xLastRunTime, mainLed_blinking_Rate );
red_led_OFF;
vTaskDelayUntil( &xLastRunTime, mainLed_blinking_Rate );

blue_led_ON;
vTaskDelayUntil( &xLastRunTime, mainLed_blinking_Rate );
blue_led_OFF;
vTaskDelayUntil( &xLastRunTime, mainLed_blinking_Rate );

green_led_ON;
vTaskDelayUntil( &xLastRunTime, mainLed_blinking_Rate );
green_led_OFF;
vTaskDelayUntil( &xLastRunTime, mainLed_blinking_Rate );
}
}
```

B. Write a code showing main function of your code.

<u>Solution-</u>



C. Describe the operation of your code in terms of the scheduler. What sort of policy is used by the scheduler?

Solution- Here, the scheduler is using preemptive policy which is based on priority. This is the policy which is used by freeRTOS by default. So, at first higher priority task will complete its execution .After that lower priority task will be executed. A task can be blocked if a task having priority more than that comes in ready state. In this code, LED task's priority is more than that of idle task ,so it gets executed whenever it comes to ready state.