Compose a FreeRTOS application that flashes the LPC1769 RGB LED as follows:

**SOLUTION**

/\*  
 \* @brief FreeRTOS Blinky example  
 \*  
 \* @note  
 \* Copyright(C) NXP Semiconductors, 2014  
 \* All rights reserved.  
 \*  
 \* @par  
 \* Software that is described herein is for illustrative purposes only  
 \* which provides customers with programming information regarding the  
 \* LPC products.  This software is supplied "AS IS" without any warranties of  
 \* any kind, and NXP Semiconductors and its licensor disclaim any and  
 \* all warranties, express or implied, including all implied warranties of  
 \* merchantability, fitness for a particular purpose and non-infringement of  
 \* intellectual property rights.  NXP Semiconductors assumes no responsibility  
 \* or liability for the use of the software, conveys no license or rights under any  
 \* patent, copyright, mask work right, or any other intellectual property rights in  
 \* or to any products. NXP Semiconductors reserves the right to make changes  
 \* in the software without notification. NXP Semiconductors also makes no  
 \* representation or warranty that such application will be suitable for the  
 \* specified use without further testing or modification.  
 \*  
 \* @par  
 \* Permission to use, copy, modify, and distribute this software and its  
 \* documentation is hereby granted, under NXP Semiconductors' and its  
 \* licensor's relevant copyrights in the software, without fee, provided that it  
 \* is used in conjunction with NXP Semiconductors microcontrollers.  This  
 \* copyright, permission, and disclaimer notice must appear in all copies of  
 \* this code.  
 \*/  
  
#include "board.h"  
#include "FreeRTOS.h"  
#include "task.h"  
  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Private types/enumerations/variables  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Public types/enumerations/variables  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Private functions  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
  
/\* Sets up system hardware \*/  
static void prvSetupHardware(void)  
{  
SystemCoreClockUpdate();  
Board\_Init();  
  
/\* Initial LED0 state is off \*/  
Board\_LED\_Set(0, false);  
Board\_LED\_Set(1, false);  
Board\_LED\_Set(2, false);  
  
}  
  
/\* LED1 toggle thread \*/  
static void vLEDTask1(void \*pvParameters)  
{  
//bool LedState = false;  
  
while (1)  
{  
Board\_LED\_Set(0, 0);  
vTaskDelay(configTICK\_RATE\_HZ);  
Board\_LED\_Set(0, 1);  
  
/\* About a 3Hz on/off toggle rate \*/  
vTaskDelay(3 \* configTICK\_RATE\_HZ + configTICK\_RATE\_HZ / 2);  
}  
}  
  
/\* LED2 toggle thread \*/  
static void vLEDTask2(void \*pvParameters)  
{  
vTaskDelay(configTICK\_RATE\_HZ + configTICK\_RATE\_HZ / 2);  
  
while (1)  
{  
  
Board\_LED\_Set(1, 0);  
vTaskDelay(configTICK\_RATE\_HZ);  
Board\_LED\_Set(1, 1);  
  
/\* About a 7Hz on/off toggle rate\*/  
vTaskDelay(3 \* configTICK\_RATE\_HZ + configTICK\_RATE\_HZ / 2);  
}  
}  
static void vLEDTask3(void \*pvParameters)  
{  
vTaskDelay(3 \* configTICK\_RATE\_HZ);  
while (1)  
{  
  
Board\_LED\_Set(2, 0);  
vTaskDelay(configTICK\_RATE\_HZ);  
Board\_LED\_Set(2, 1);  
  
/\* About a 7Hz on/off toggle rate\*/  
vTaskDelay(3 \* configTICK\_RATE\_HZ + configTICK\_RATE\_HZ / 2);  
}  
}  
  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Public functions  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
  
/\*\*  
 \* @brief main routine for FreeRTOS blinky example  
 \* @return Nothing, function should not exit  
 \*/  
int main(void)  
{  
prvSetupHardware();  
  
/\* LED1 toggle thread \*/  
xTaskCreate(vLEDTask1, (signed char\* ) "vTaskLed1",  
configMINIMAL\_STACK\_SIZE, NULL, (tskIDLE\_PRIORITY + 3UL),  
(xTaskHandle \*) NULL);  
  
/\* LED2 toggle thread \*/  
xTaskCreate(vLEDTask2, (signed char\* ) "vTaskLed2",  
configMINIMAL\_STACK\_SIZE, NULL, (tskIDLE\_PRIORITY + 2UL),  
(xTaskHandle \*) NULL);  
  
xTaskCreate(vLEDTask3, (signed char\* ) "vTaskLed3",  
configMINIMAL\_STACK\_SIZE, NULL, (tskIDLE\_PRIORITY + 1UL),  
(xTaskHandle \*) NULL);  
  
/\* Start the scheduler \*/  
vTaskStartScheduler();  
  
/\* Should never arrive here \*/  
return 1;  
}  
  
/\*\*  
 \* @}  
 \*/