```
 1/*
 2 * @brief FreeRTOS Blinky example
 3 *
 4 * @note
 5 * Copyright(C) NXP Semiconductors, 2014
 6 * All rights reserved.
 7 *
 8 * @par
 9 * Software that is described herein is for
   illustrative purposes only
10 * which provides customers with programming
   information regarding the
11 * LPC products.  This software is supplied "AS IS"
   without any warranties of
12 * any kind, and NXP Semiconductors and its licensor
   disclaim any and
13 * all warranties, express or implied, including all
   implied warranties of
14 * merchantability, fitness for a particular purpose
   and non-infringement of
15 * intellectual property rights.  NXP Semiconductors
   assumes no responsibility
16 * or liability for the use of the software, conveys
   no license or rights under any
17 * patent, copyright, mask work right, or any other
   intellectual property rights in
18 * or to any products. NXP Semiconductors reserves the
   right to make changes
19 * in the software without notification. NXP
   Semiconductors also makes no
20 * representation or warranty that such application
   will be suitable for the
21 * specified use without further testing or
```

```
  modification.
22 *
23 * @par
24 * Permission to use, copy, modify, and distribute
  this software and its
25 * documentation is hereby granted, under NXP
  Semiconductors' and its
26 * licensor's relevant copyrights in the software,
  without fee, provided that it
27 * is used in conjunction with NXP Semiconductors
  microcontrollers.  This
28 * copyright, permission, and disclaimer notice must
  appear in all copies of
29 * this code.
30 */
31
32#include "board.h"
33#include "FreeRTOS.h"
34#include "task.h"
35
36/
  ************************************************
  ***********************
37 * Private types/enumerations/variables
38
  ************************************************
  ***********************/
39
40/
  ************************************************
  ***********************
41 * Public types/enumerations/variables
42
```

```
  ************************************************************
  **********************/
43
44/
  ************************************************************
  ************************
45 * Private functions
46
  ************************************************************
  **********************/
47
48/* Sets up system hardware */
49static void prvSetupHardware(void)
50{
51    SystemCoreClockUpdate();
52    Board_Init();
53
54    /* Initial LED0 state is off */
55    Board_LED_Set(0, false);
56}
57
58/* LED1 toggle thread */
59static void vLEDTask1(void *pvParameters) {
60    bool LedState = false;
61
62    while (1) {
63        Board_LED_Set(0, LedState);
64        LedState = (bool) !LedState;
65
66        /* About a 3Hz on/off toggle rate */
67        vTaskDelay(configTICK_RATE_HZ / 6);
68    }
69}
```

```
70
71/* LED2 toggle thread */
72static void vLEDTask2(void *pvParameters) {
73    bool LedState = false;
74
75    while (1) {
76        Board_LED_Set(1, LedState);
77        LedState = (bool) !LedState;
78
79        /* About a 7Hz on/off toggle rate */
80        vTaskDelay(configTICK_RATE_HZ / 14);
81    }
82}
83
84/* UART (or output) thread */
85static void vUARTTask(void *pvParameters) {
86    int tickCnt = 0;
87
88    while (1) {
89        DEBUGOUT("Tick: %d\r\n", tickCnt);
90        tickCnt++;
91
92        /* About a 1s delay here */
93        vTaskDelay(configTICK_RATE_HZ);
94    }
95}
96
97/
  ************************************************************
  ************************
98 * Public functions
99
  ************************************************************
```

```
   **********************/
100
101/**
102 * @brief   main routine for FreeRTOS blinky example
103 * @return  Nothing, function should not exit
104 */
105int main(void)
106{
107    prvSetupHardware();
108
109    /* LED1 toggle thread */
110    xTaskCreate(vLEDTask1, (signed char *)
   "vTaskLed1",
111                configMINIMAL_STACK_SIZE, NULL,
   (tskIDLE_PRIORITY + 1UL),
112                (xTaskHandle *) NULL);
113
114    /* LED2 toggle thread */
115    xTaskCreate(vLEDTask2, (signed char *)
   "vTaskLed2",
116                configMINIMAL_STACK_SIZE, NULL,
   (tskIDLE_PRIORITY + 1UL),
117                (xTaskHandle *) NULL);
118
119    /* UART output thread, simply counts seconds */
120    xTaskCreate(vUARTTask, (signed char *)
   "vTaskUart",
121                configMINIMAL_STACK_SIZE, NULL,
   (tskIDLE_PRIORITY + 1UL),
122                (xTaskHandle *) NULL);
123
124    /* Start the scheduler */
125    vTaskStartScheduler();
```

```
126
127     /* Should never arrive here */
128     return 1;
129}
130
131/**
132 * @}
133 */
134
```