example_freeRTOSBlinky.c

This is an example which descibes the steps to create an example application which toggles the High End Timer (HET) pin 17 (LED in USB & HDK) based on the FreeRTOS timer tick of one second.

Step 1:

Create a new project.

Navigate: -> File -> New -> Project

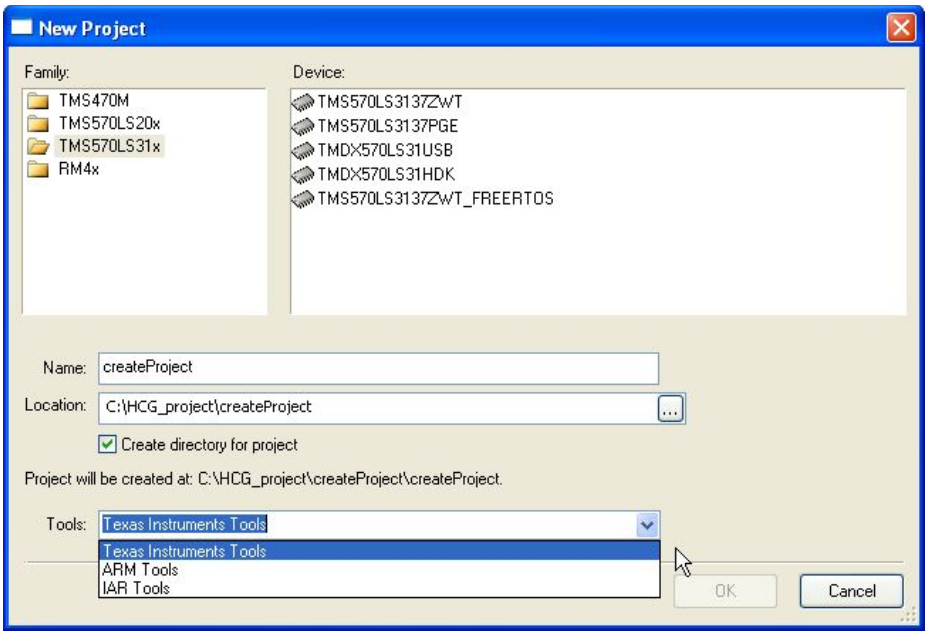

Figure: Create a new Project

Step 2:

Configure driver code generation:

- Enable GIO driver
- Disable others

Navigate: -> TMS570LSxx /RM4 -> Driver Enable

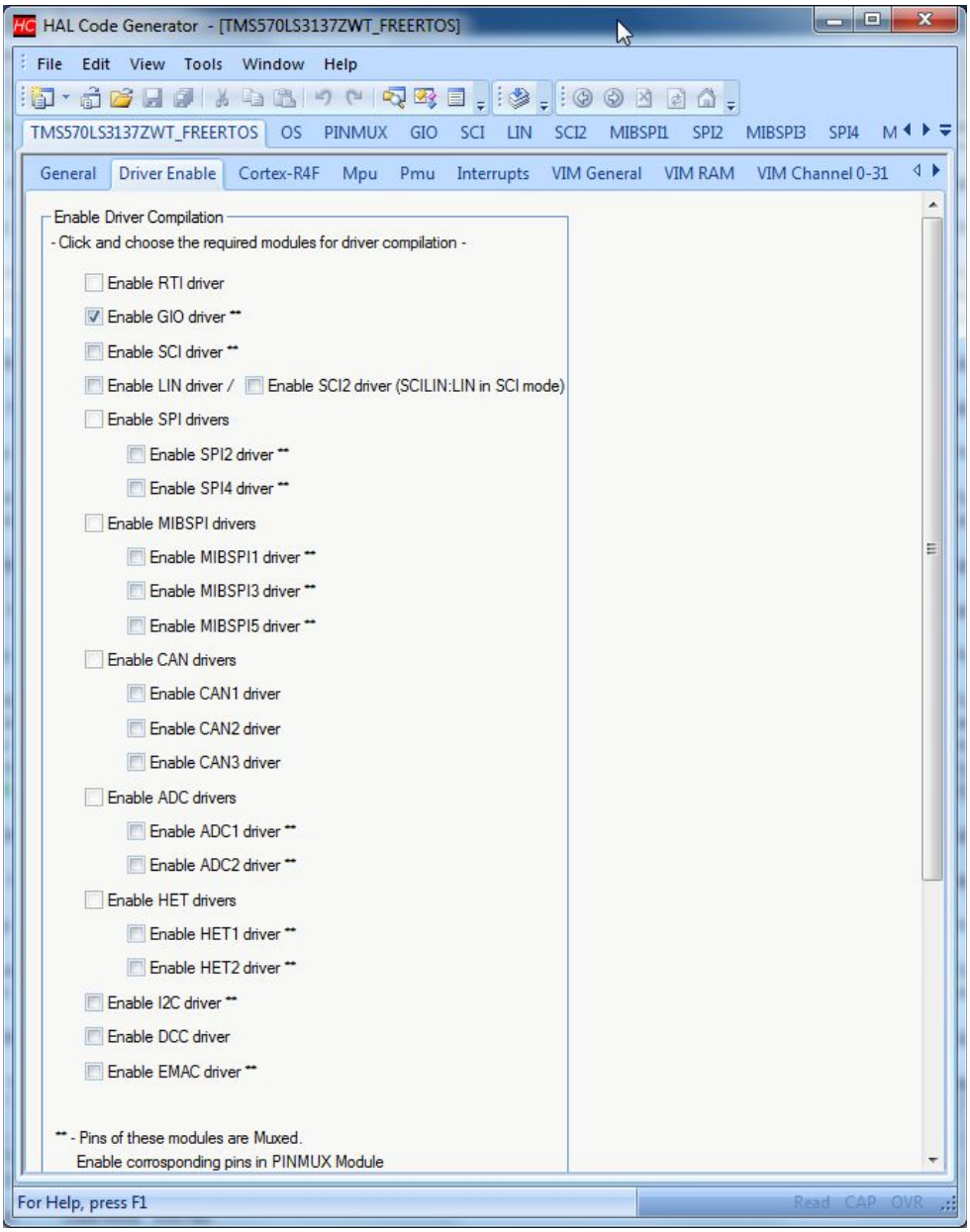

Figure: Driver Configuration

Step 3:

Configure Interrupt handling:

- Enable SVC
- Enter FreeRTOS SVC handler name 'vPortSWI'

Navigate: -> TMS570LSxx /RM4 -> Interrupts

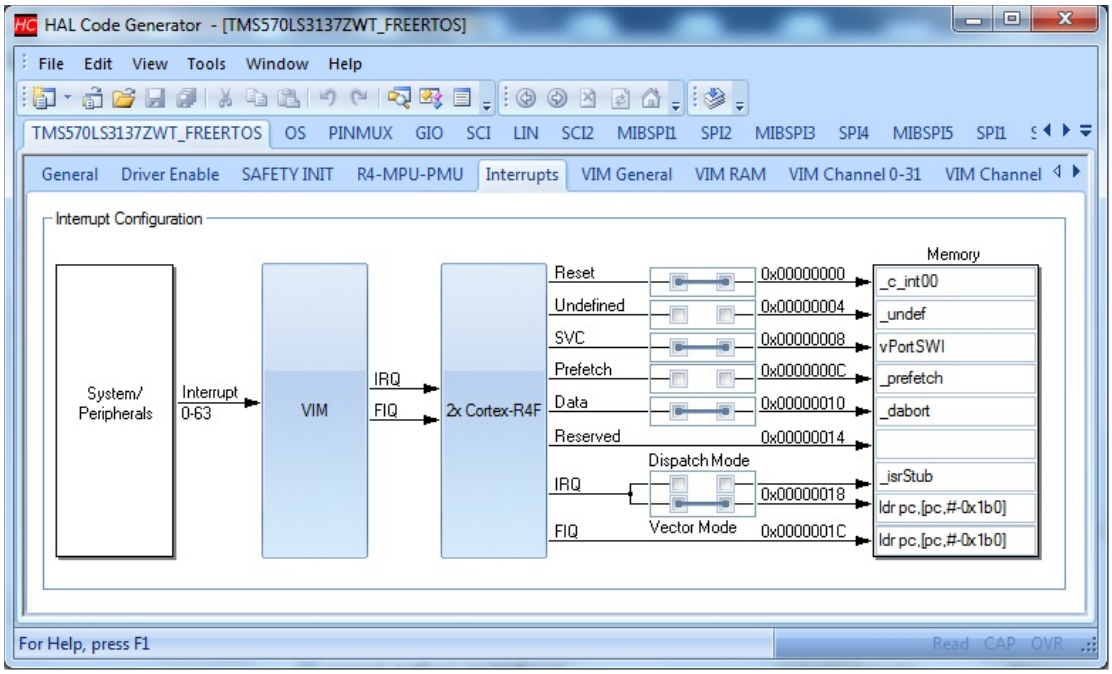

Figure: Interrupt Configuration

Step 4:

Configure VIM RAM:

- Enter FreeRTOS Timer Tick handler name 'vPortPreemptiveTick' at offset 0x0000000C
- Enter SSI handler name 'vPortYeildWithinAPI ' at offset 0x00000058

Navigate: -> TMS570LSxx /RM4 -> VIM RAM

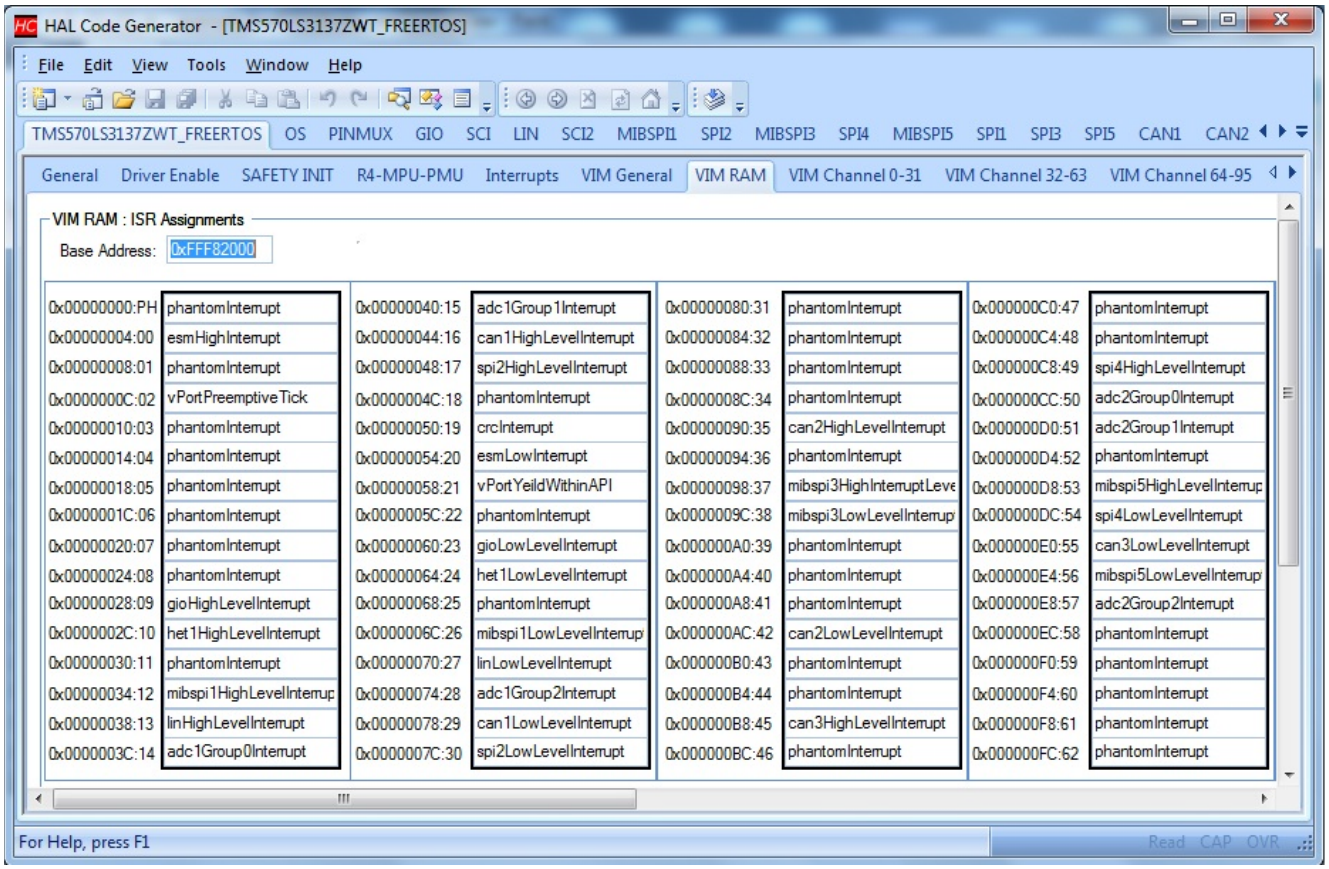

Figure: VIM RAM Configuration

Step 5:

Configure Vectored Interrupt Module Channels:

- Enable VIM Channel 2
- Map VIM Channel 2 to IRQ
- Enable VIM Channel 21
- Map VIM Channel 21 to IRQ

Navigate: -> TMS570LSxx /RM4 -> VIM Channel 0-31

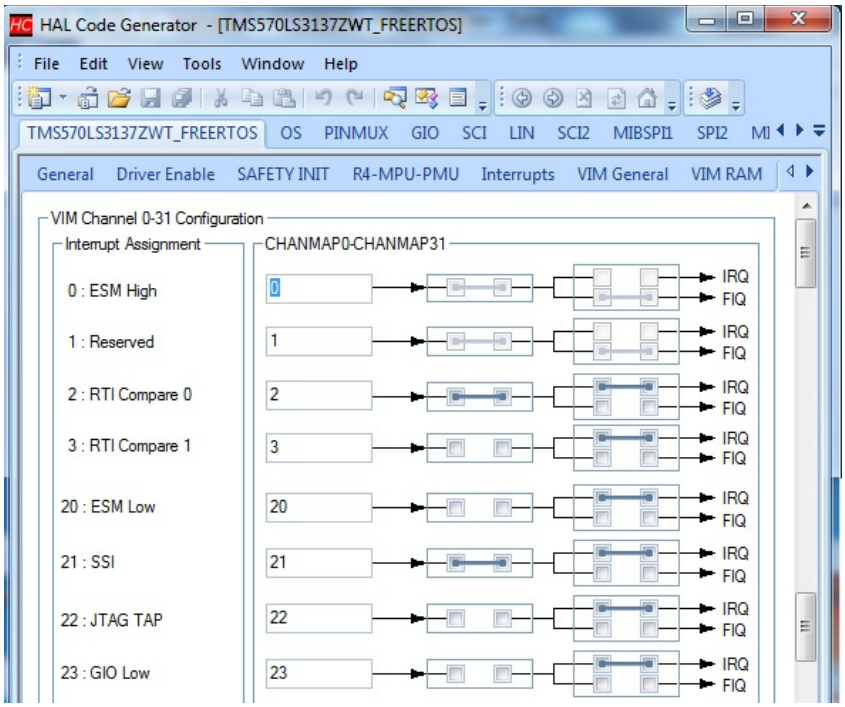

Figure: VIM Channel Configuration

Step 6:

Configure OS timer tick to 1 ms:

- Enter Tick Rate of 1000

Navigate: -> OS -> General

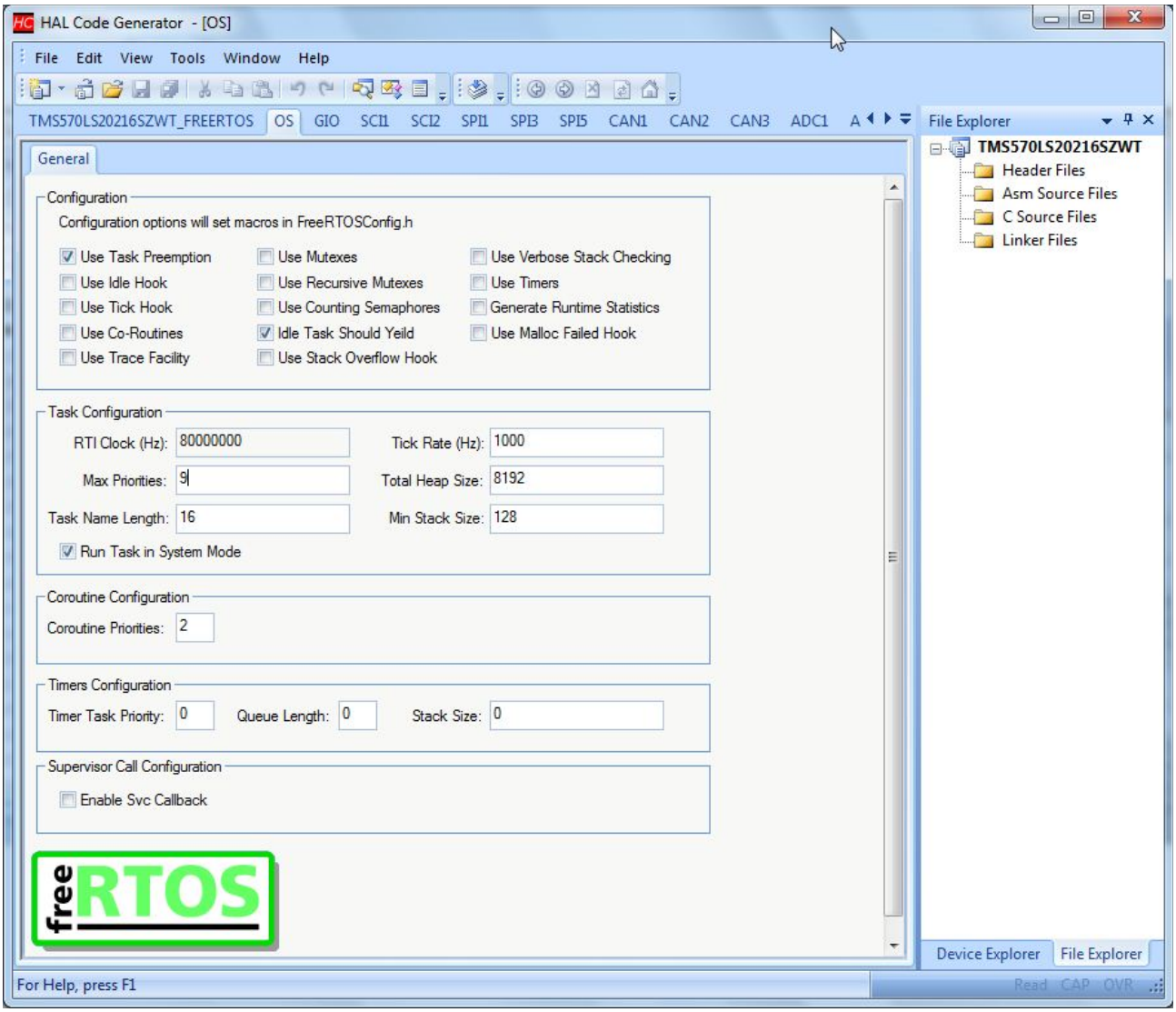

Figure: OS General Configuration

Step 7:

Generate code

Navigate: -> File -> Generate Code

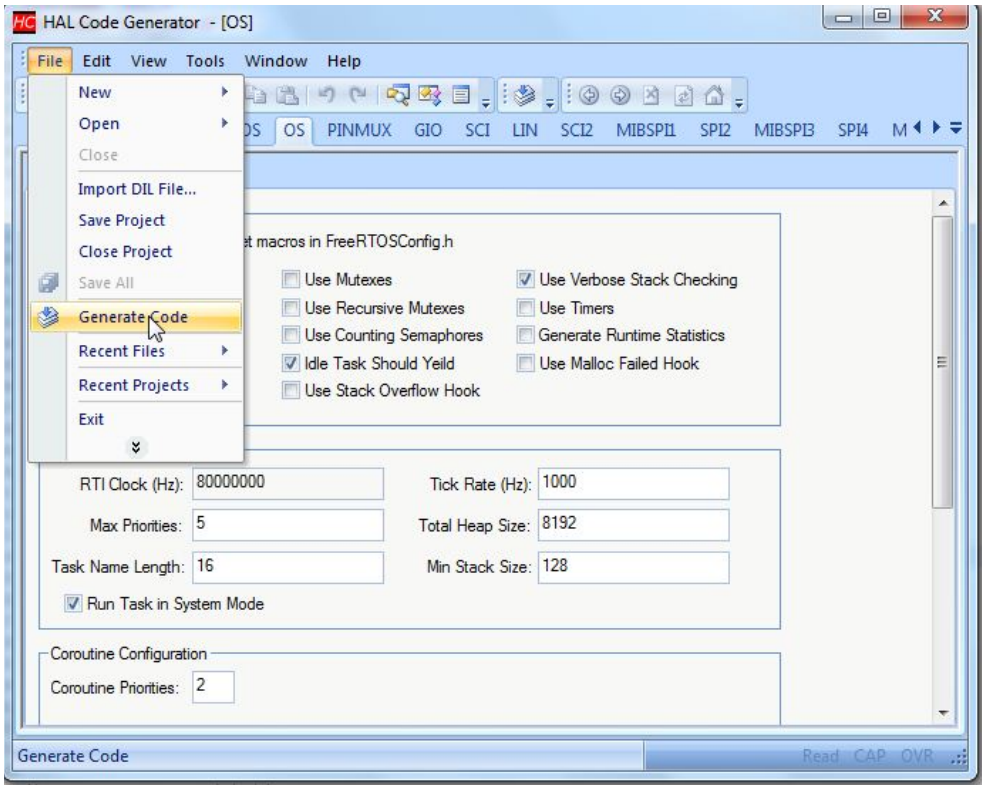

Figure: Generate Code

Step 8:

Copy source code below into your application.

The example file **example_freeRTOSBlinky.c** can also be found in the examples folder: ../HALCoGen/examples

Note

HALCoGen generates an enpty main function in **sys_main.c**, please make sure that you link in the right main function or copy the source into the user code sections of this file.

Enable GCC extension in the CCS project (Project properties -> Build -> ARM Compiler -> Advanced options -> Language options -> Enable support for GCC extensions)

```
/*
* Copyright (C) 2009-2015 Texas Instruments Incorporated - www.ti.com
*
*
*  Redistribution and use in source and binary forms, with or without
*  modification, are permitted provided that the following conditions
*  are met:
*
*    Redistributions of source code must retain the above copyright
*    notice, this list of conditions and the following disclaimer.
*
*    Redistributions in binary form must reproduce the above copyright
*    notice, this list of conditions and the following disclaimer in the
*    documentation and/or other materials provided with the
*    distribution.
*
*    Neither the name of Texas Instruments Incorporated nor the names of
*    its contributors may be used to endorse or promote products derived
*    from this software without specific prior written permission.
*
*  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
*  "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
*  LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
*  A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
*  OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
*  SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES INCLUDING, BUT NOT
*  LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
*  DATA, OR PROFITS; OR BUSINESS INTERRUPTION HOWEVER CAUSED AND ON ANY
*  THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
*  INCLUDING NEGLIGENCE OR OTHERWISE ARISING IN ANY WAY OUT OF THE USE
*  OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*
*/
```

```
/* USER CODE BEGIN (0) */
/* USER CODE END */

/* Include Files */

#include "sys_common.h"
#include "system.h"

/* USER CODE BEGIN (1) */
/* Include FreeRTOS scheduler files */
#include "FreeRTOS.h"
#include "os_task.h"

/* Include HET header file - types, definitions and function declarations for system driver */
#include "het.h"
#include "gio.h"

/* Define Task Handles */
xTaskHandle xTask1Handle;

/* Task1 */
void vTask1(void *pvParameters)
{
    for(;;)
    {
        /* Taggle HET[1] with timer tick */
        gioSetBit(hetPORT1, 17, gioGetBit(hetPORT1, 17) ^ 1);
        vTaskDelay(100);
    }
}
/* USER CODE END */


/* USER CODE BEGIN (2) */
/* USER CODE END */


void main(void)
{
/* USER CODE BEGIN (3) */

    /* Set high end timer GIO port hetPort pin direction to all output */
    gioSetDirection(hetPORT1, 0xFFFFFFFF);


    /* Create Task 1 */
    if (xTaskCreate(vTask1,"Task1", configMINIMAL_STACK_SIZE, NULL, 1, &xTask1Handle) != pdTRUE)
    {
        /* Task could not be created */
        while(1);
    }

    /* Start Scheduler */
    vTaskStartScheduler();

    /* Run forever */
    while(1);
/* USER CODE END */
}


/* USER CODE BEGIN (4) */
/* USER CODE END */
```