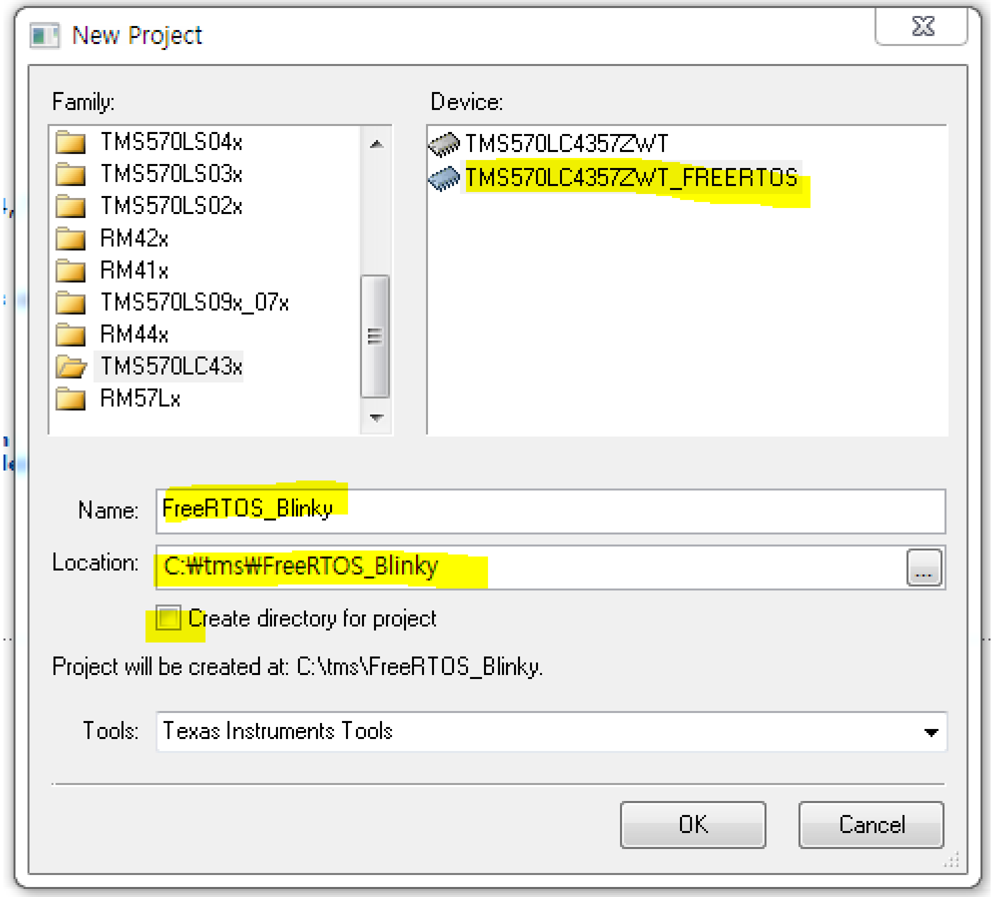
## 2018.05.15.Tue

| | | | |
|---|---|---|---|
| **노트북:** | SW | | |
| **만든 날짜:** | 2018-05-15 오후 3:18 | **수정한 날짜:** | 2018-05-15 오후 3:40 |
| **작성자:** | 정상요 | | |

RTOS



Name: FreeRTOS_Blinky

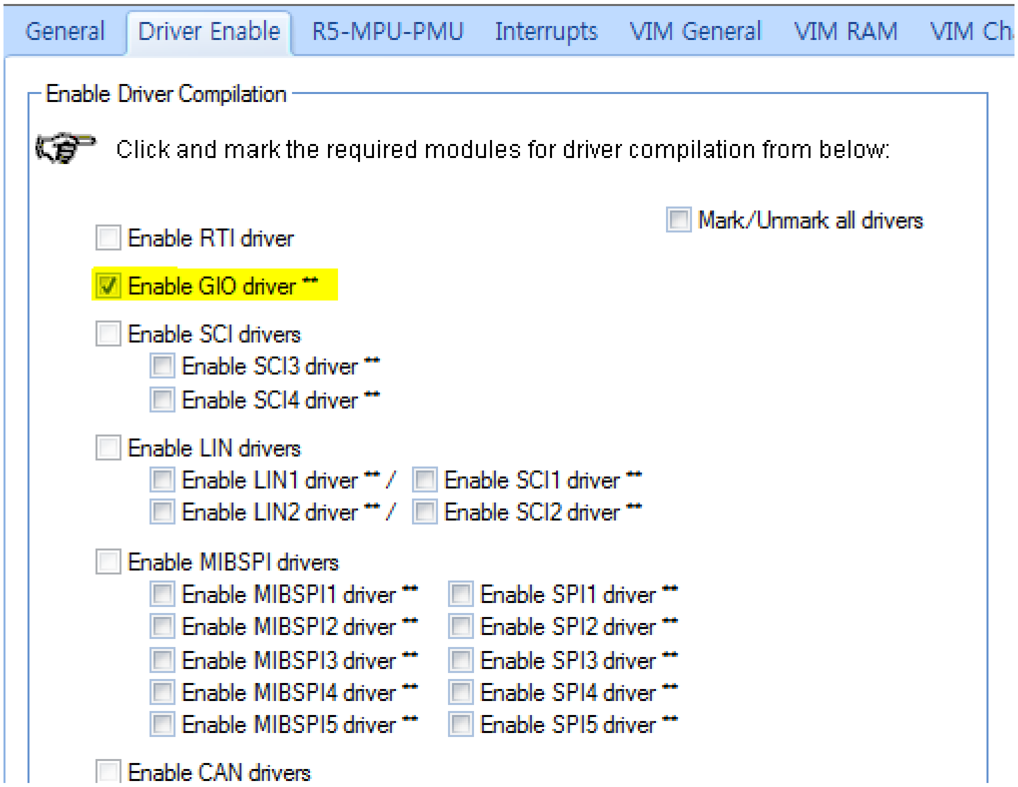
Location: C:₩tms₩FreeRTOS_Blinky

□ Create directory for project

Project will be created at: C:\tms\FreeRTOS_Blinky.

Tools: Texas Instruments Tools

FreeRTOS로 설정



General | Driver Enable | R5-MPU-PMU | Interrupts | VIM General | VIM RAM | VIM Ch

Enable Driver Compilation

Click and mark the required modules for driver compilation from below:

□ Mark/Unmark all drivers

□ Enable RTI driver

☑ Enable GIO driver **

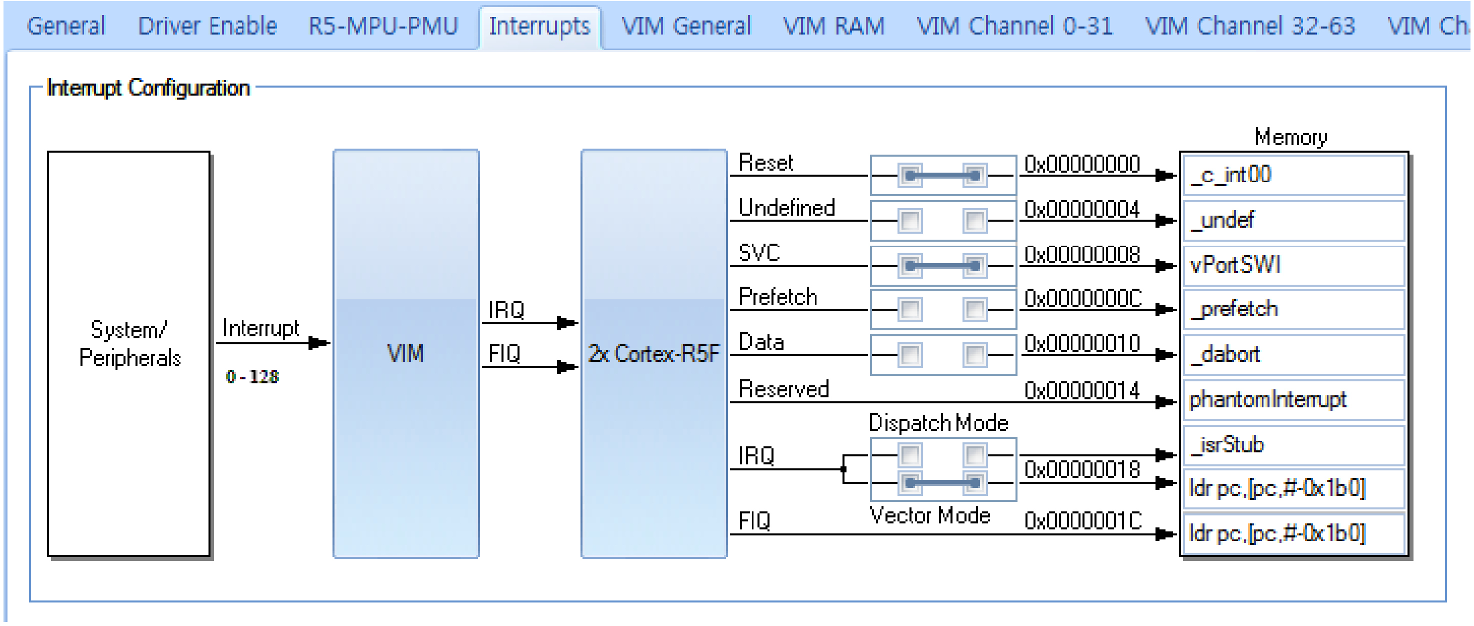
□ Enable SCI drivers
- □ Enable SCI3 driver **
- □ Enable SCI4 driver **

□ Enable LIN drivers
- □ Enable LIN1 driver ** / □ Enable SCI1 driver **
- □ Enable LIN2 driver ** / □ Enable SCI2 driver **

□ Enable MIBSPI drivers
- □ Enable MIBSPI1 driver ** □ Enable SPI1 driver **
- □ Enable MIBSPI2 driver ** □ Enable SPI2 driver **
- □ Enable MIBSPI3 driver ** □ Enable SPI3 driver **
- □ Enable MIBSPI4 driver ** □ Enable SPI4 driver **
- □ Enable MIBSPI5 driver ** □ Enable SPI5 driver **

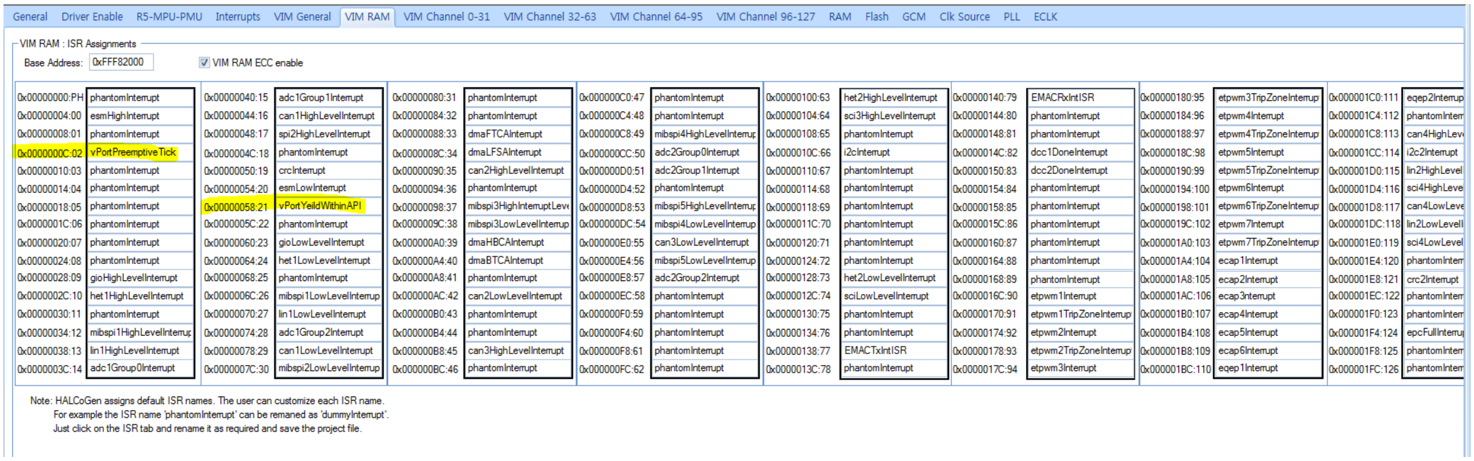
□ Enable CAN drivers

Enable GIO driver만 set



General | Driver Enable | R5-MPU-PMU | Interrupts | VIM General | VIM RAM | VIM Channel 0-31 | VIM Channel 32-63 | VIM Ch

Interrupt Configuration



General | Driver Enable | R5-MPU-PMU | Interrupts | VIM General | VIM RAM | VIM Channel 0-31 | VIM Channel 32-63 | VIM Channel 64-95 | VIM Channel 96-127 | RAM | Flash | GCM | Clk Source | PLL | ECLK

VIM RAM : ISR Assignments

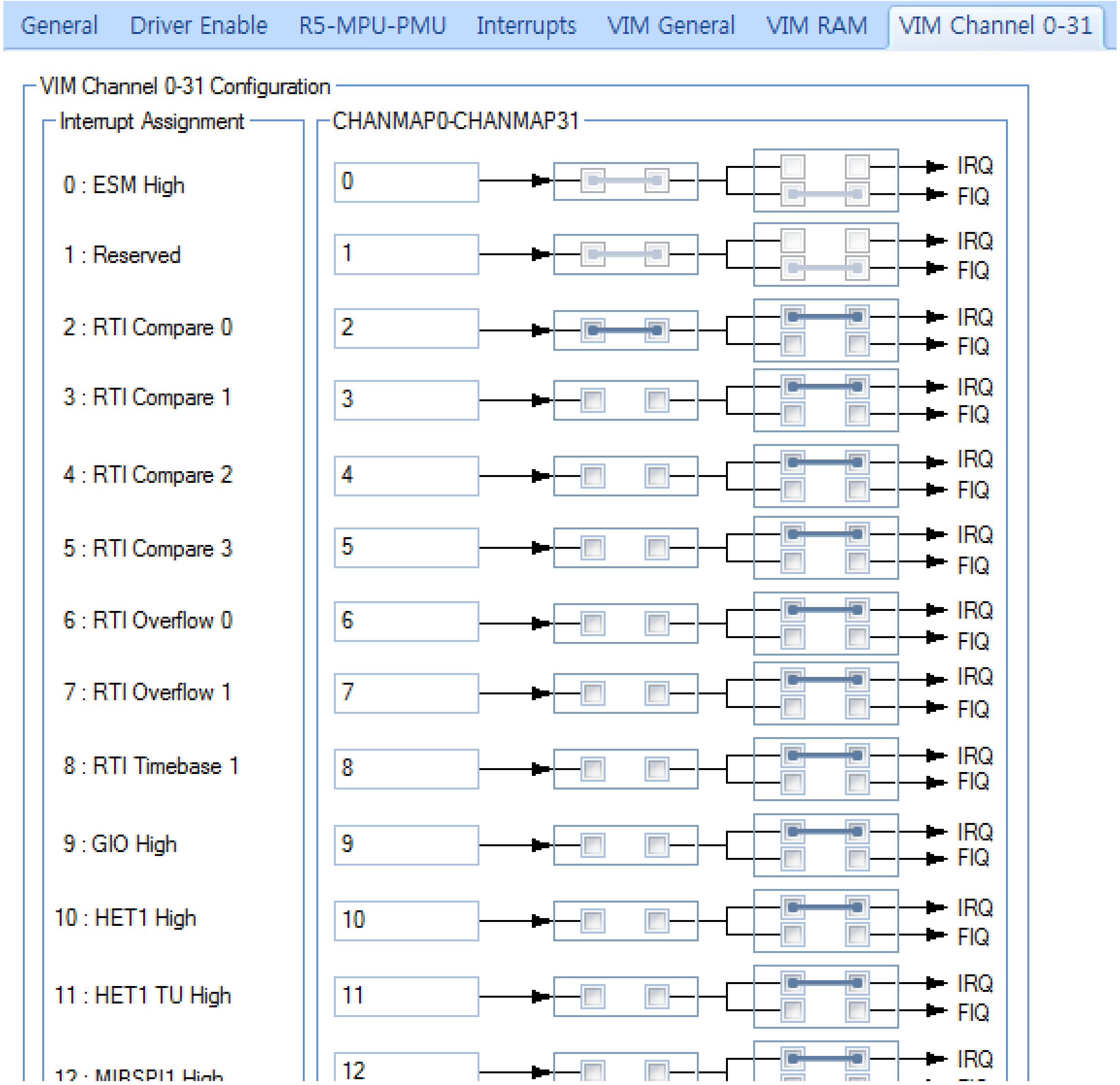
Base Address: 0xFFF82000   ☑ VIM RAM ECC enable

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x00000000:PH | phantomInterrupt | 0x00000040:15 | adc1Group1Interrupt | 0x00000080:31 | phantomInterrupt | 0x000000C0:47 | phantomInterrupt |
| 0x00000004:00 | esmHighInterrupt | 0x00000044:16 | can1HighLevelInterrupt | 0x00000084:32 | phantomInterrupt | 0x000000C4:48 | phantomInterrupt |
| 0x00000008:01 | phantomInterrupt | 0x00000048:17 | spi2HighLevelInterrupt | 0x00000088:33 | dmaFTCAInterrupt | 0x000000C8:49 | mibspi4HighLevelInterrup |
| 0x0000000C:02 | vPortPreemptiveTick | 0x0000004C:18 | phantomInterrupt | 0x0000008C:34 | dmaLFSAInterrupt | 0x000000CC:50 | adc2Group0Interrupt |
| 0x00000010:03 | phantomInterrupt | 0x00000050:19 | crcInterrupt | 0x00000090:35 | can2HighLevelInterrupt | 0x000000D0:51 | adc2Group1Interrupt |
| 0x00000014:04 | phantomInterrupt | 0x00000054:20 | esmLowInterrupt | 0x00000094:36 | phantomInterrupt | 0x000000D4:52 | phantomInterrupt |
| 0x00000018:05 | phantomInterrupt | 0x00000058:21 | vPortYeildWithinAPI | 0x00000098:37 | mibspi3HighInterruptLeve | 0x000000D8:53 | mibspi5HighLevelInterrup |
| 0x0000001C:06 | phantomInterrupt | 0x0000005C:22 | phantomInterrupt | 0x0000009C:38 | mibspi3LowLevelInterrup | 0x000000DC:54 | mibspi4LowLevelInterrup |
| 0x00000020:07 | phantomInterrupt | 0x00000060:23 | gioLowLevelInterrupt | 0x000000A0:39 | dmaHBCAInterrupt | 0x000000E0:55 | can3LowLevelInterrupt |
| 0x00000024:08 | phantomInterrupt | 0x00000064:24 | het1LowLevelInterrupt | 0x000000A4:40 | dmaBTCAInterrupt | 0x000000E4:56 | mibspi5LowLevelInterrup |
| 0x00000028:09 | gioHighLevelInterrupt | 0x00000068:25 | phantomInterrupt | 0x000000A8:41 | phantomInterrupt | 0x000000E8:57 | adc2Group2Interrupt |
| 0x0000002C:10 | het1HighLevelInterrupt | 0x0000006C:26 | mibspi1LowLevelInterrup | 0x000000AC:42 | can2LowLevelInterrupt | 0x000000EC:58 | phantomInterrupt |
| 0x00000030:11 | phantomInterrupt | 0x00000070:27 | lin1LowLevelInterrupt | 0x000000B0:43 | phantomInterrupt | 0x000000F0:59 | phantomInterrupt |
| 0x00000034:12 | mibspi1HighLevelInterrup | 0x00000074:28 | adc1Group2Interrupt | 0x000000B4:44 | phantomInterrupt | 0x000000F4:60 | phantomInterrupt |
| 0x00000038:13 | lin1HighLevelInterrupt | 0x00000078:29 | can1LowLevelInterrupt | 0x000000B8:45 | can3HighLevelInterrupt | 0x000000F8:61 | phantomInterrupt |
| 0x0000003C:14 | adc1Group0Interrupt | 0x0000007C:30 | mibspi2LowLevelInterrup | 0x000000BC:46 | phantomInterrupt | 0x000000FC:62 | phantomInterrupt |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x00000100:63 | het2HighLevelInterrupt | 0x00000140:79 | EMACRxIntISR | 0x00000180:95 | etpwm3TripZoneInterrupt | 0x000001C0:111 | eqep2Interrup |
| 0x00000104:64 | sci3HighLevelInterrupt | 0x00000144:80 | phantomInterrupt | 0x00000184:96 | etpwm4Interrupt | 0x000001C4:112 | phantomInterr |
| 0x00000108:65 | phantomInterrupt | 0x00000148:81 | phantomInterrupt | 0x00000188:97 | etpwm4TripZoneInterrupt | 0x000001C8:113 | can4HighLeve |
| 0x0000010C:66 | i2cInterrupt | 0x0000014C:82 | dcc1DoneInterrupt | 0x0000018C:98 | etpwm5Interrupt | 0x000001CC:114 | i2c2Interrupt |
| 0x00000110:67 | phantomInterrupt | 0x00000150:83 | dcc2DoneInterrupt | 0x00000190:99 | etpwm5TripZoneInterrupt | 0x000001D0:115 | lin2HighLevel |
| 0x00000114:68 | phantomInterrupt | 0x00000154:84 | phantomInterrupt | 0x00000194:100 | etpwm6Interrupt | 0x000001D4:116 | sci4HighLevel |
| 0x00000118:69 | phantomInterrupt | 0x00000158:85 | phantomInterrupt | 0x00000198:101 | etpwm6TripZoneInterrupt | 0x000001D8:117 | can4LowLeve |
| 0x0000011C:70 | phantomInterrupt | 0x0000015C:86 | phantomInterrupt | 0x0000019C:102 | etpwm7Interrupt | 0x000001DC:118 | lin2LowLevel |
| 0x00000120:71 | phantomInterrupt | 0x00000160:87 | phantomInterrupt | 0x000001A0:103 | etpwm7TripZoneInterrupt | 0x000001E0:119 | sci4LowLevel |
| 0x00000124:72 | phantomInterrupt | 0x00000164:88 | phantomInterrupt | 0x000001A4:104 | ecap1Interrupt | 0x000001E4:120 | phantomInterr |
| 0x00000128:73 | het2LowLevelInterrupt | 0x00000168:89 | phantomInterrupt | 0x000001A8:105 | ecap2Interrupt | 0x000001E8:121 | crc2Interrupt |
| 0x0000012C:74 | sciLowLevelInterrupt | 0x0000016C:90 | etpwm1Interrupt | 0x000001AC:106 | ecap3nterrupt | 0x000001EC:122 | phantomInterr |
| 0x00000130:75 | phantomInterrupt | 0x00000170:91 | etpwm1TripZoneInterrupt | 0x000001B0:107 | ecap4Interrupt | 0x000001F0:123 | phantomInterr |
| 0x00000134:76 | phantomInterrupt | 0x00000174:92 | etpwm2Interrupt | 0x000001B4:108 | ecap5Interrupt | 0x000001F4:124 | epcFullInterru |
| 0x00000138:77 | EMACTxIntISR | 0x00000178:93 | etpwm2TripZoneInterrupt | 0x000001B8:109 | ecap6Interrupt | 0x000001F8:125 | phantomInterr |
| 0x0000013C:78 | phantomInterrupt | 0x0000017C:94 | etpwm3Interrupt | 0x000001BC:110 | eqep1Interrupt | 0x000001FC:126 | phantomInterr |

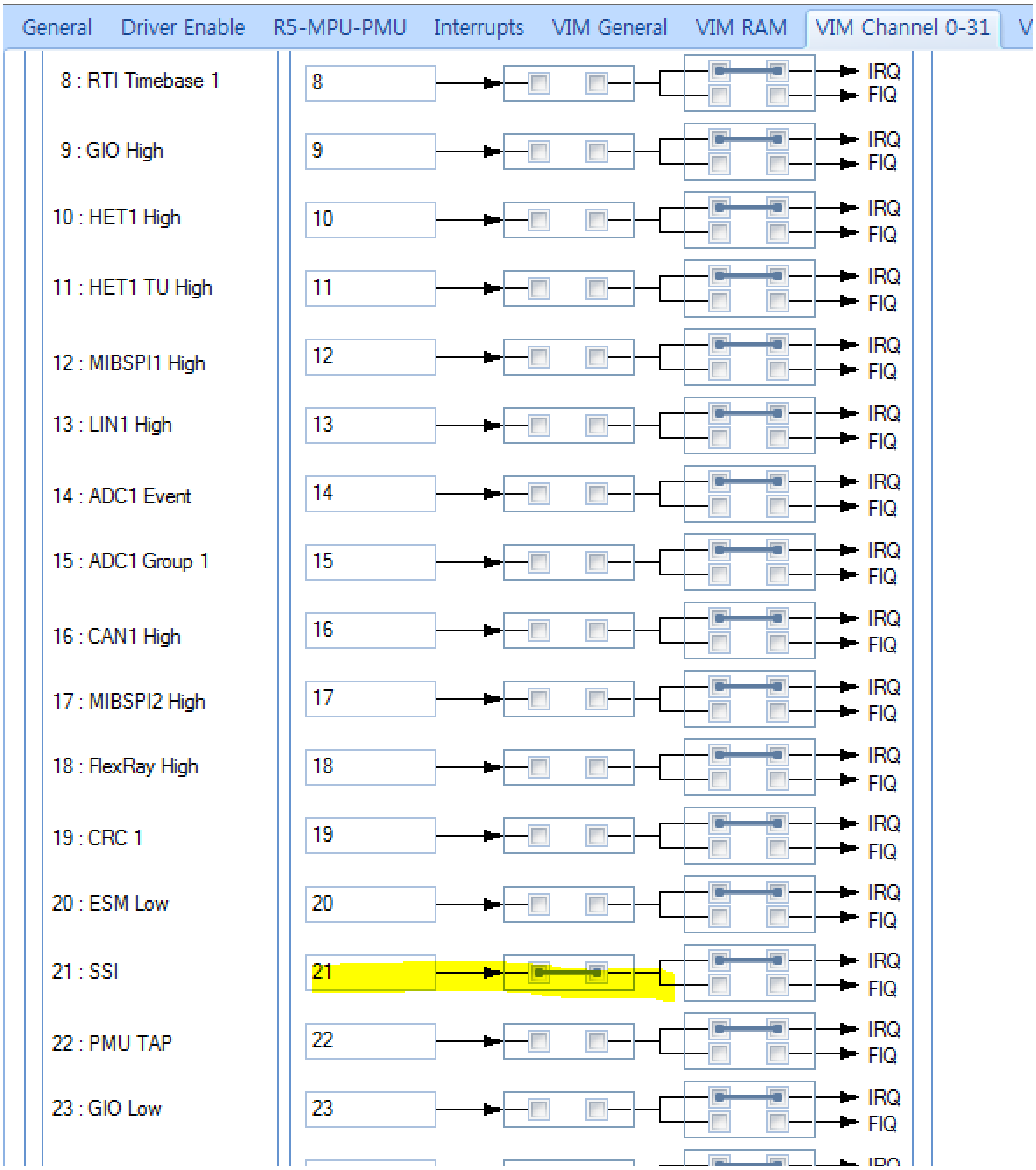
Note: HALCoGen assigns default ISR names. The user can customize each ISR name.
For example the ISR name 'phantomInterrupt' can be remaned as 'dummyInterrupt'.
Just click on the ISR tab and rename it as required and save the project file.

IVT(Interrupt Vector Table)라고 생각하면 된다.



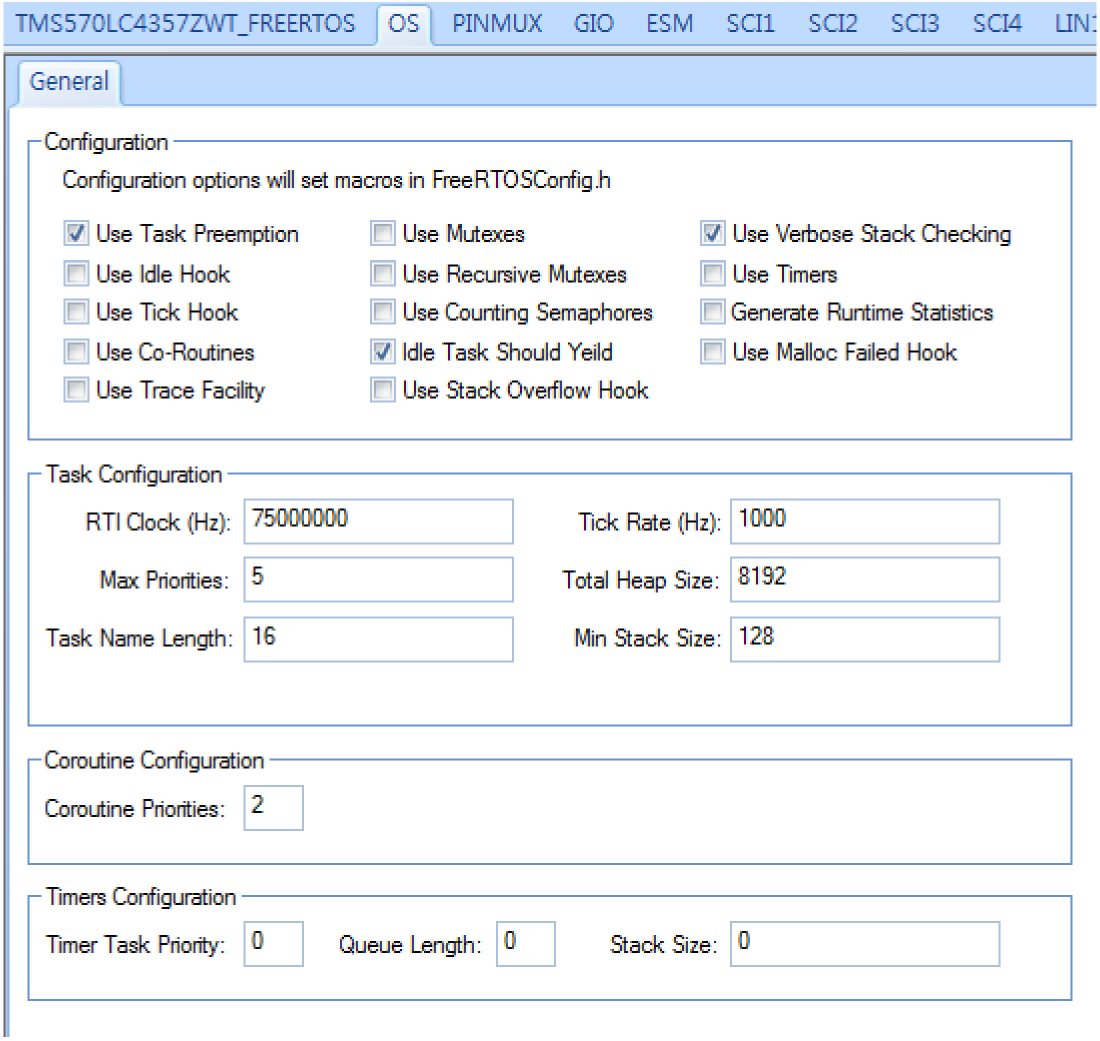
VIM Channel 0-31 Configuration

| Interrupt Assignment | CHANMAP0-CHANMAP31 |
|---|---|
| 0 : ESM High | 0 |
| 1 : Reserved | 1 |
| 2 : RTI Compare 0 | 2 |
| 3 : RTI Compare 1 | 3 |
| 4 : RTI Compare 2 | 4 |
| 5 : RTI Compare 3 | 5 |
| 6 : RTI Overflow 0 | 6 |
| 7 : RTI Overflow 1 | 7 |
| 8 : RTI Timebase 1 | 8 |
| 9 : GIO High | 9 |
| 10 : HET1 High | 10 |
| 11 : HET1 TU High | 11 |
| 12 : MIBSPI1 High | 12 |



| | | | |
|---|---|---|---|
| 8 : RTI Timebase 1 | 8 | IRQ | FIQ |
| 9 : GIO High | 9 | IRQ | FIQ |
| 10 : HET1 High | 10 | IRQ | FIQ |
| 11 : HET1 TU High | 11 | IRQ | FIQ |
| 12 : MIBSPI1 High | 12 | IRQ | FIQ |
| 13 : LIN1 High | 13 | IRQ | FIQ |
| 14 : ADC1 Event | 14 | IRQ | FIQ |
| 15 : ADC1 Group 1 | 15 | IRQ | FIQ |
| 16 : CAN1 High | 16 | IRQ | FIQ |
| 17 : MIBSPI2 High | 17 | IRQ | FIQ |
| 18 : FlexRay High | 18 | IRQ | FIQ |
| 19 : CRC 1 | 19 | IRQ | FIQ |
| 20 : ESM Low | 20 | IRQ | FIQ |
| 21 : SSI | 21 | IRQ | FIQ |
| 22 : PMU TAP | 22 | IRQ | FIQ |
| 23 : GIO Low | 23 | IRQ | FIQ |

2번, 21번 set 양보를 의미



Configuration

Configuration options will set macros in FreeRTOSConfig.h

☑ Use Task Preemption
□ Use Idle Hook
□ Use Tick Hook
□ Use Co-Routines
□ Use Trace Facility

□ Use Mutexes
□ Use Recursive Mutexes
□ Use Counting Semaphores
☑ Idle Task Should Yeild
□ Use Stack Overflow Hook

☑ Use Verbose Stack Checking
□ Use Timers
□ Generate Runtime Statistics
□ Use Malloc Failed Hook

Task Configuration

RTI Clock (Hz): 75000000
Tick Rate (Hz): 1000
Max Priorities: 5
Total Heap Size: 8192
Task Name Length: 16
Min Stack Size: 128

Coroutine Configuration

Coroutine Priorities: 2

Timers Configuration

Timer Task Priority: 0
Queue Length: 0
Stack Size: 0

Use Verbose Stack Checking -> Stack이 깨지는지 확인

이제 Generate Code

---

CCS에서 Source Code 작성

```
#include "HL_sys_common.h"
#include "FreeRTOS.h"
#include "os_task.h"
#include "HL_het.h"
#include "HL_gio.h"
xTaskHandle xTask1Handle;
void vTask1(void* pvParameters)
{
    for(;;)
    {
        /* Taggle HET[1] with timer tick */
        gioSetBit(hetPORT1, 17, gioGetBit(hetPORT1, 17) ^ 1);
        vTaskDelay(100);
    }
}
void main(void)
{
    /* Set High end timer GIO port hetPort pin direction to all output */
    gioSetDirection(hetPORT1, 0xFFFFFFFF);
    /* Create Task1 */
    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);
}
```