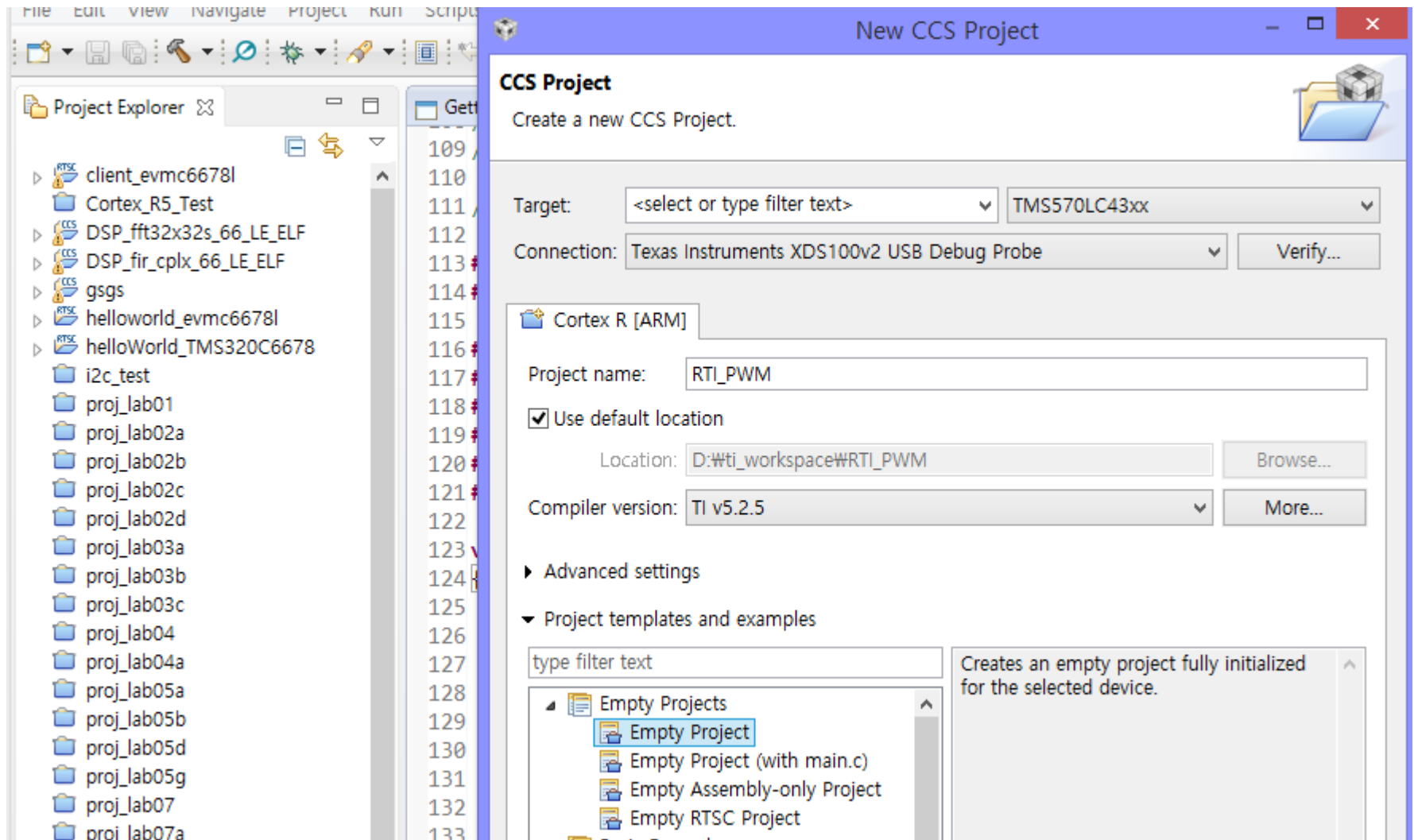


# Xilinx Zynq FPGA, TI DSP, MCU 기반의 회로 설계 및 임베디드 전문가 과정

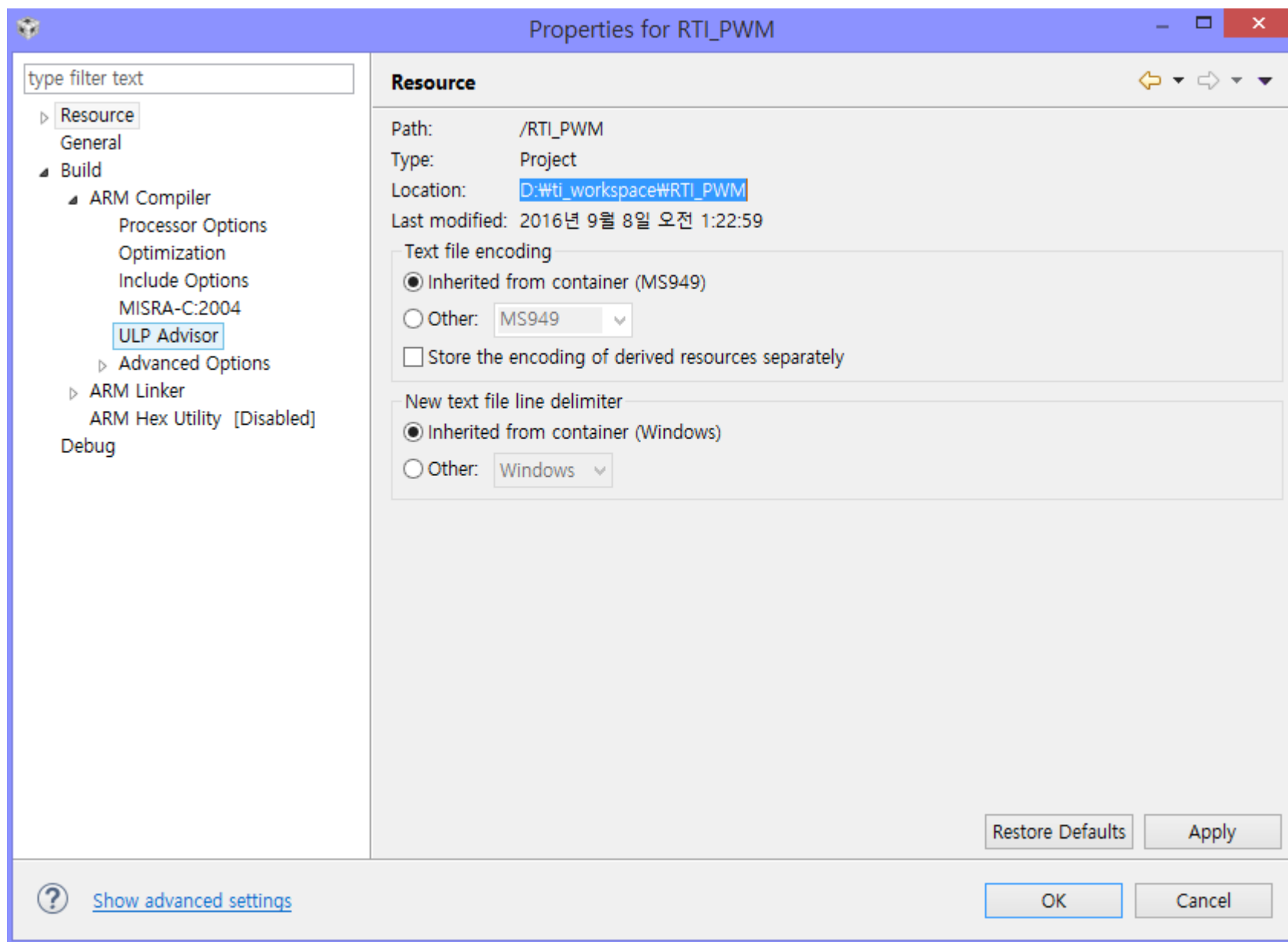
강사 – Innova Lee(이상훈)  
gcccompil3r@gmail.com

# **etPWM Control & Real Time Interrupt**

1. File – New – CCS Project -> Target, Connection 설정
2. Compiler Version – TI v5.2.6(이하 버전도 가능)
3. Empty Project 선택
4. 생성하기



CCS에서 생성한 Project의 주소를 복사한다.



복사한 상태로 HALCoGen을 동작시킨다.


(D:) > ti > Hercules > HALCoGen > v04.05.02

름	수정한 날짜	유
config	2016-04-24 오전...	파
Docs	2016-04-24 오전...	파
drivers	2016-04-24 오전...	파
edit	2016-04-24 오전...	파
examples	2016-04-24 오전...	파
help	2016-05-03 오전...	파
HTML	2016-04-24 오전...	파
styles	2016-04-24 오전...	파
HALCOGEN.exe	2015-04-07 오후...	응
HCG_updater.exe	2015-07-02 오전...	응
HCG_updater.ini	2016-04-24 오전...	구
mfc100.dll	2013-06-27 오후...	응
msvcr100.dll	2013-06-27 오후...	응
Production_License_Agreement_SRAS14...	2015-02-19 오후...	PC
readme.txt	2016-03-02 오후...	텍
TICGEN.dll	2015-04-07 오후...	응
TIDEVTMP.dll	2015-04-07 오후...	응
TIDILIO.dll	2015-04-07 오후...	응
TIDRVTMP.dll	2015-04-07 오후...	응
TIHCGIO.dll	2015-04-07 오후...	응
TIJS32.dll	2015-04-07 오후...	응
uninstall.dat	2016-04-24 오전...	D/
uninstall.exe	2016-04-24 오전...	응

File Edit View Tools Window Help

Start Page

My.TI | TI Home | Microcontrollers

 **TEXAS INSTRUMENTS**

**HALCoGen**

**INNOVATE. CREATE. MAKE THE DIFFERENCE.™**

**HALCoGen: 04.05.02 - Released 02.Mar.2016**

### Important Hercules Safety MCU Links:

Hercules product web pages provide access to device data sheets, technical reference manuals, application notes, videos, software downloads/updates, and online ordering of evaluation and development kits.

### HALCoGen Wiki Page

### Hercules Product Main Home Page

- [RM4 Product Home Page](#)
- [TMS570 Product Home Page](#)
- [TMS470M Product Home](#)

### Hercules Technical Support Forum

Search for topics or ask technical questions about all Hercules MCUs - RM4, TMS570 and TMS470M

### Hercules MCU Wiki Site

Download development kit schematics, software examples, training videos and information and much more on the Hercules WIKI pages.

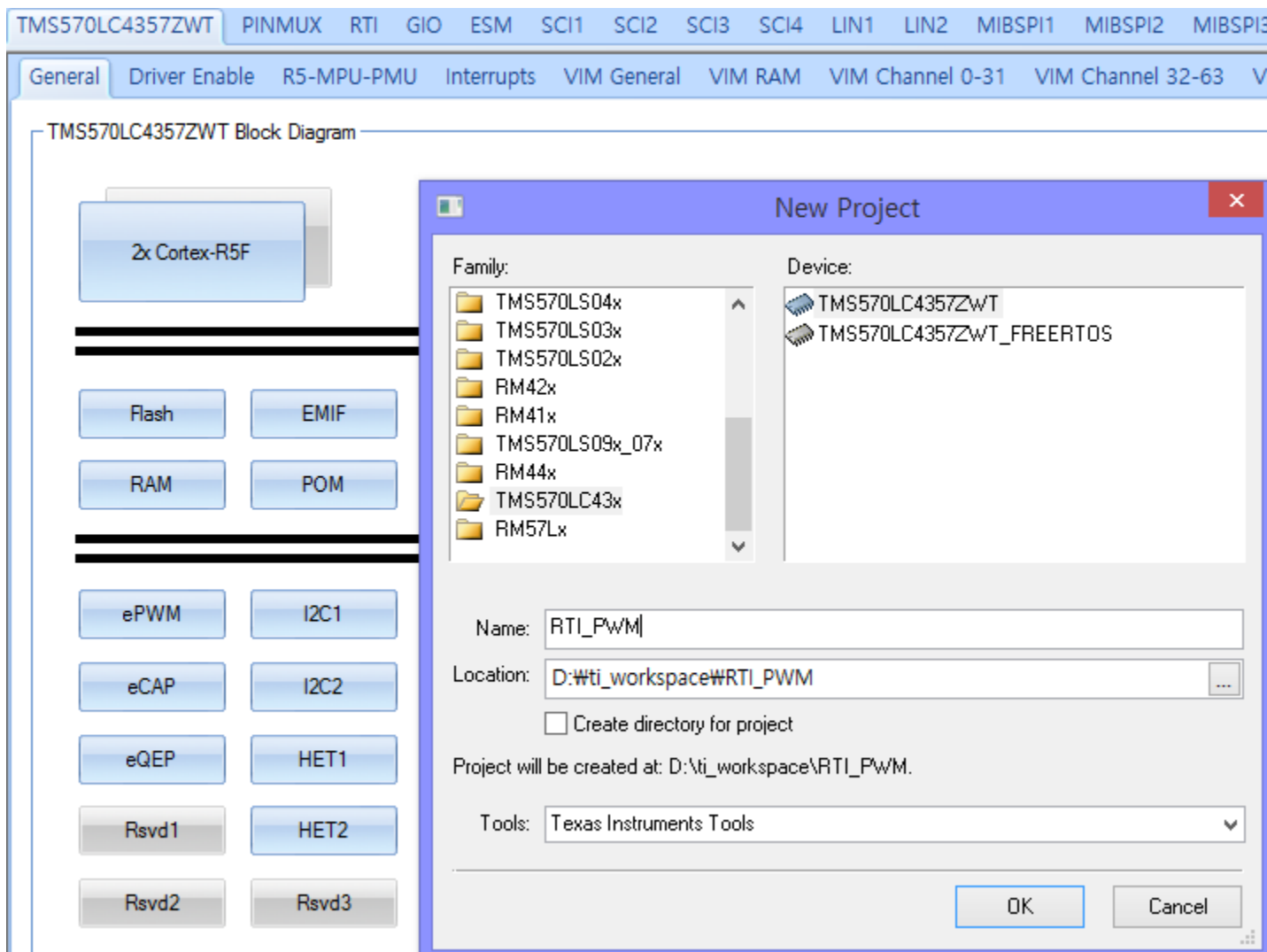
### 3rd Party Links

[FreeRTOS Home](#)  
[Keil Application Note on how use HALCoGen generated code in &microVision](#)  
[IAR Application Note on how use HALCoGen generated code in IAR Embedded Workbench](#)  
[ARM Cortex-R4F Technical Technical Reference Manual](#)

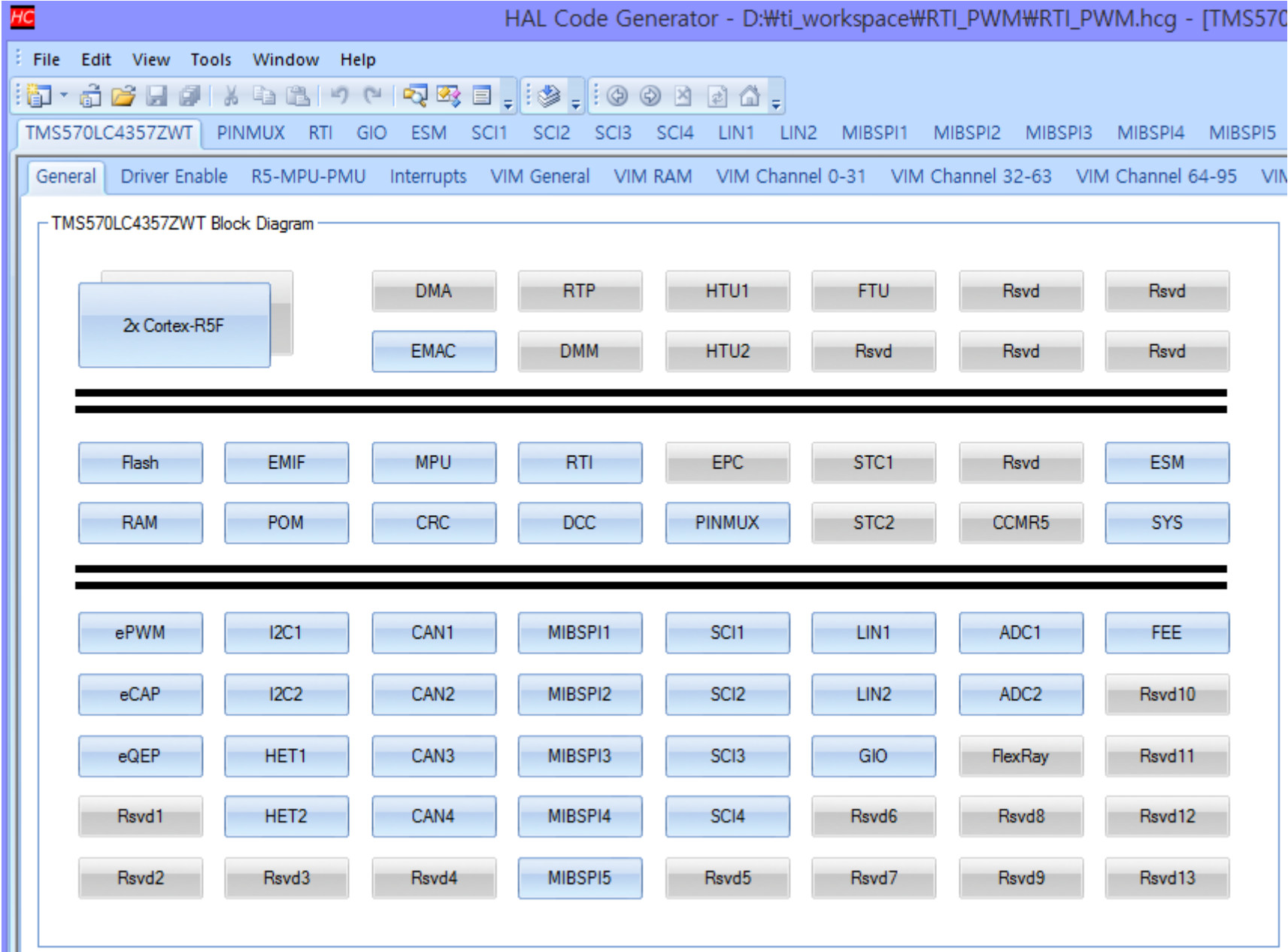
### Open Source

[HALCoGen Manifest](#)  
[Open Source Information and Download](#)

1. HALCoGen에서 File – New – Project
2. Device 선택(TMS570LC4357ZWT)
3. Location에 복사한 주소 붙여넣기(Create Directory for Project 해제)
4. Project 생성



이번엔 PWM 을 활용하여 LED 를 제어해보도록 하자!






우선 **Driver Enable** 을 누르고 RTI 를 활성화하도록 한다.

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

Enable Driver Compilation

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

☒ Enable RTI driver ☐ Mark/Unmark all drivers

☐ Enable GPIO 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 CAN1 driver

☐ Enable CAN2 driver

☐ Enable CAN3 driver

☐ Enable CAN4 driver \*\*

그리고 추가적으로 ETPWM 을 활성화 한다.

- ☐ Enable HET drivers
  - ☐ Enable HET1 driver \*\*
  - ☐ Enable HET2 driver \*\*
- ☐ Enable I2C driver \*\*
  - ☐ Enable I2C1 driver \*\*
  - ☐ Enable I2C2 driver \*\*
- ☐ Enable EMAC driver \*\*
- ☐ Enable DCC driver
- ☐ Enable EMIF driver \*\*
- ☐ Enable POM driver
- ☐ Enable CRC driver
  - ☐ Enable CRC1 driver
  - ☐ Enable CRC2 driver
- ☐ Enable EQEP driver
  - ☐ Enable EQEP1 driver \*\*
  - ☐ Enable EQEP2 driver \*\*
- ☒ Enable ETPWM driver
- ☐ Enable ECAP driver
- ☐ Enable FEE driver

Note :

\*\* - Pins of these modules are muxed. Enable the corresponding pins in PINMUX Module.

다음으로 PINMUX 에서 K18 행에 etPWM2B 를 클릭하여 활성화 하도록 한다.

TMS570LC4357ZWT   PINMUX   RTI   GIO   ESM   SCI1   SCI2   SCI3   SCI4   LIN1   LIN2   MIBSPI1   MIBSPI2   MIBSPI3   MIBSPI4   MIBSPI5   SPI1   SF

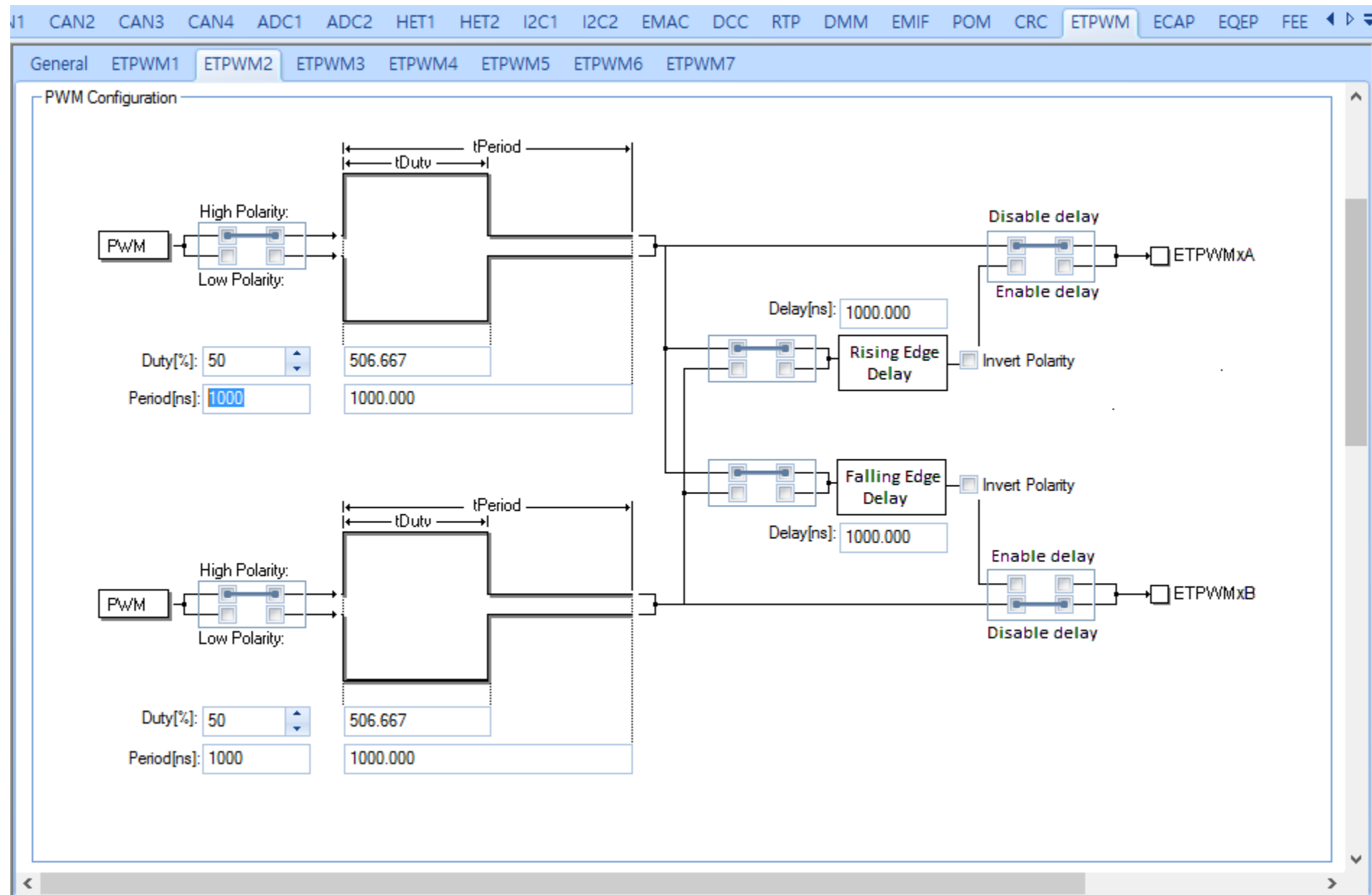
Pin Muxing   Input Pin Muxing   Special Pin Muxing

K3	RESERVED	EMIF_DATA[07]	EMIF_DATA[00]	NONE	NONE	NONE	
K5	ETMDATA[23]	EMIF_DATA[07]	NONE	NONE	NONE	NONE	
K15	ETMDATA[16]	EMIF_DATA[00]	NONE	NONE	NONE	NONE	
K17	EMIF_nCS[3]	RTP_DATA[14]	N2HET2[09]	NONE	NONE	NONE	
K18	N2HET1[00]	MIBSPI4CLK	NONE	NONE	NONE	eTPWM2B	
K19	N2HET1[28]	NONE	MII_RXCLK	RMII_REFCLK	MII_RX_AVCLK4	NONE	
L5	ETMDATA[24]	EMIF_DATA[08]	N2HET2[24]	MIBSPI5NCS[4]	NONE	NONE	
L15	ETMDATA[17]	EMIF_DATA[01]	NONE	NONE	NONE	NONE	
L17	EMIF_nCS[2]	NONE	GIOB[4]	NONE	NONE	NONE	
M1	GIOA[7]	NONE	N2HET2[06]	NONE	NONE	eTPWM2A	

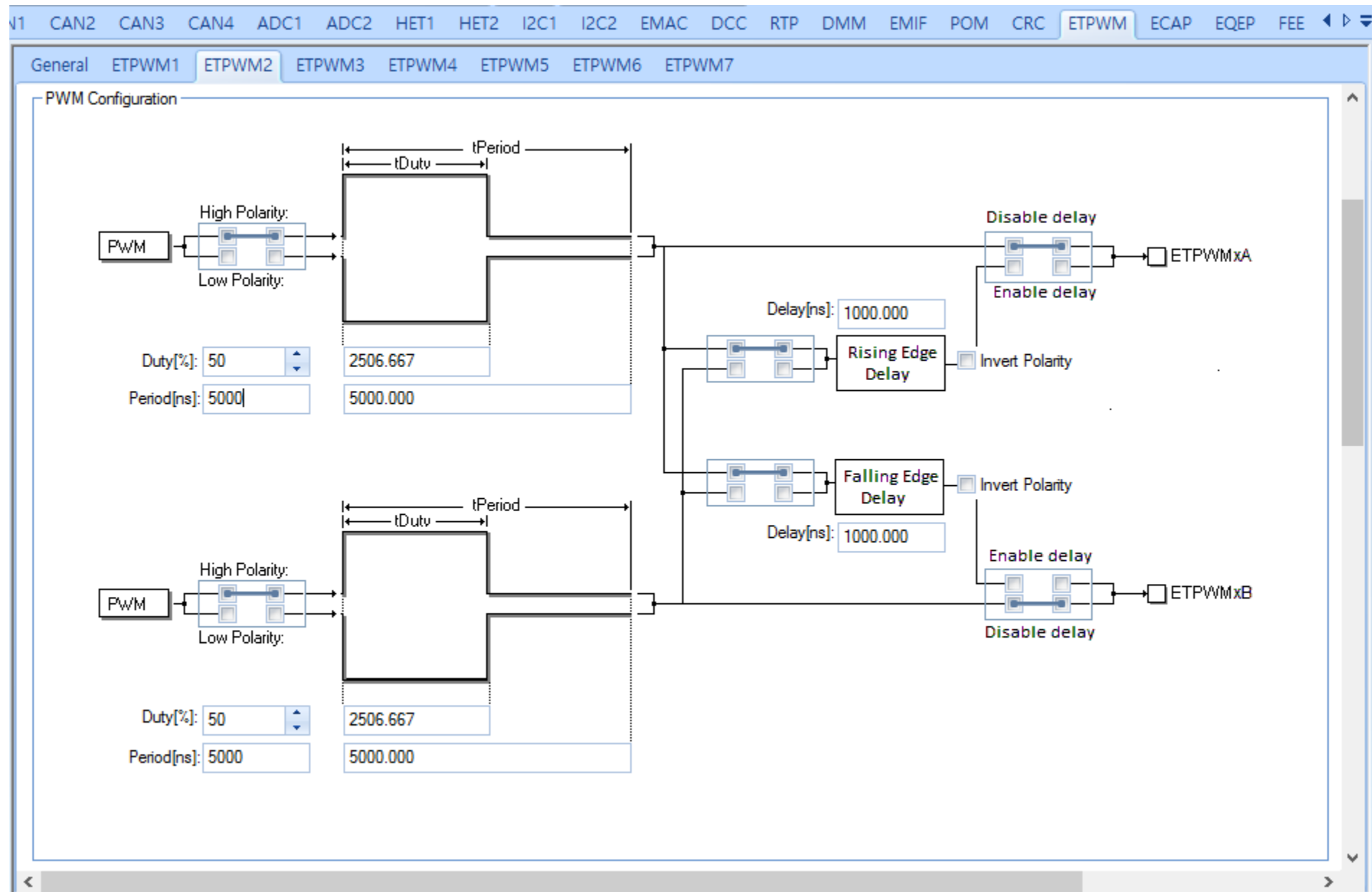
다음으로 ETPWM 탭으로 가서 ETPWM2 를 활성화 한다.

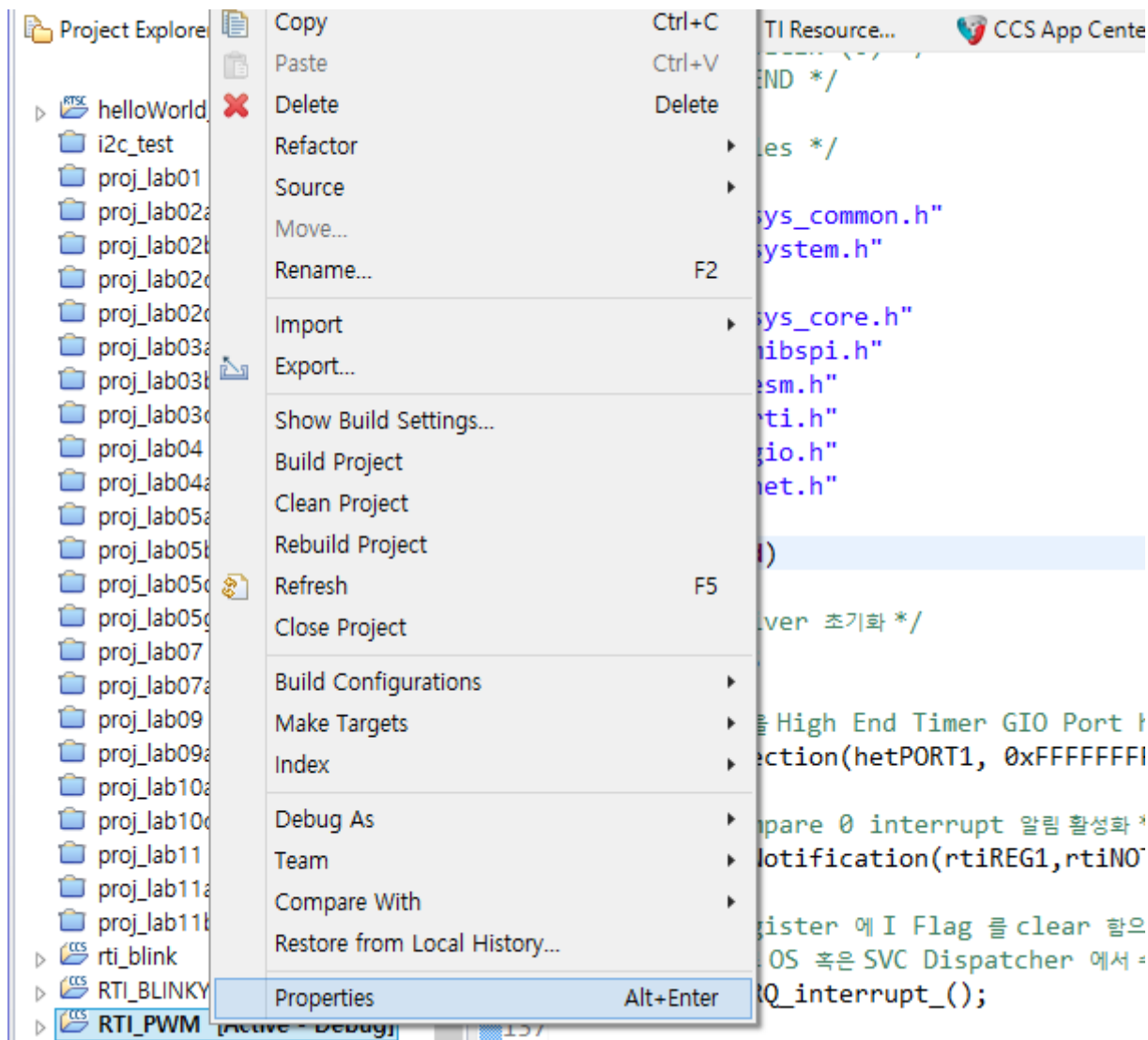


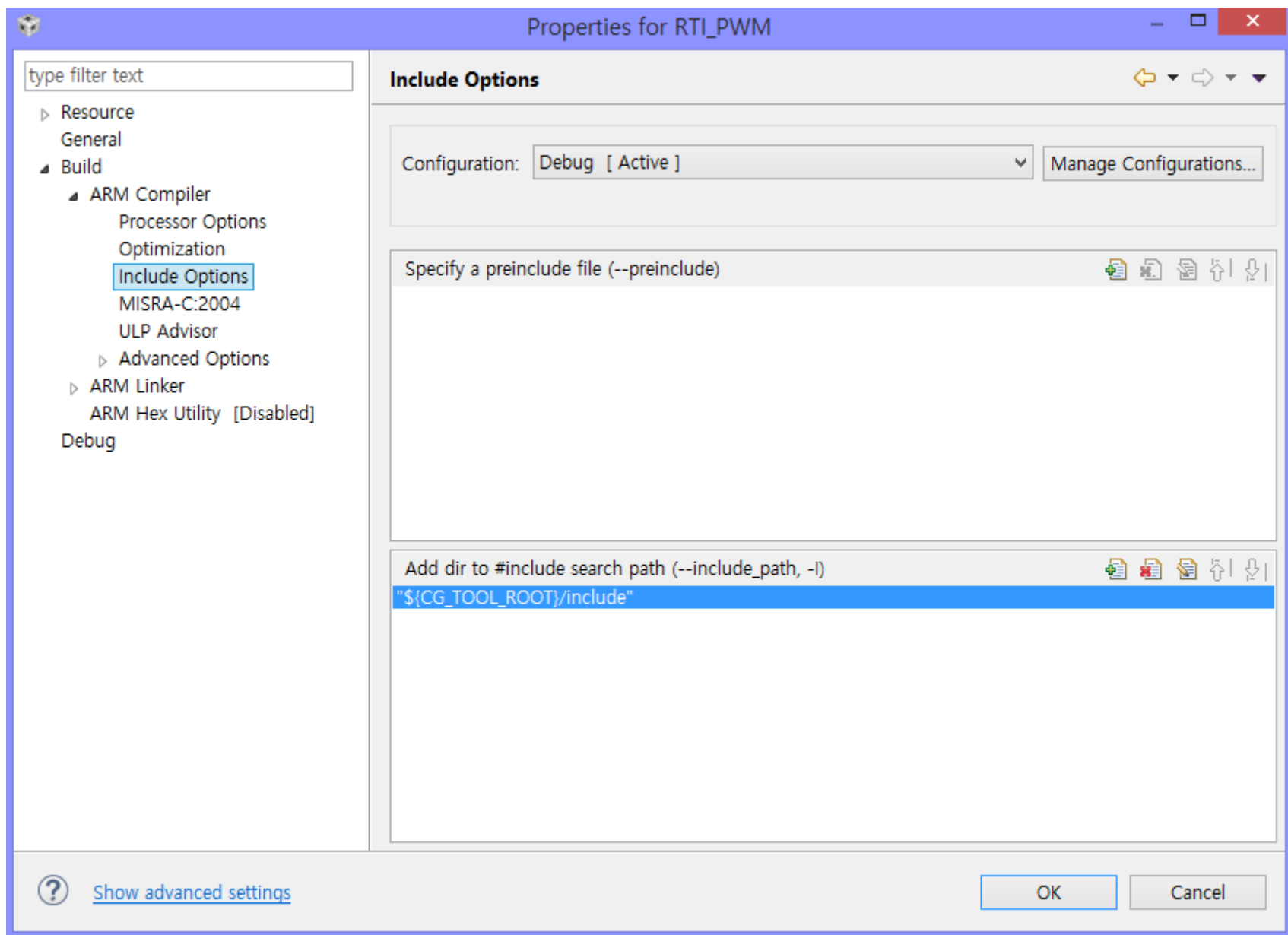
이제 ETPWM2 에 가서 Period 를 조정할 것인데 초기에는 아래와 같다.



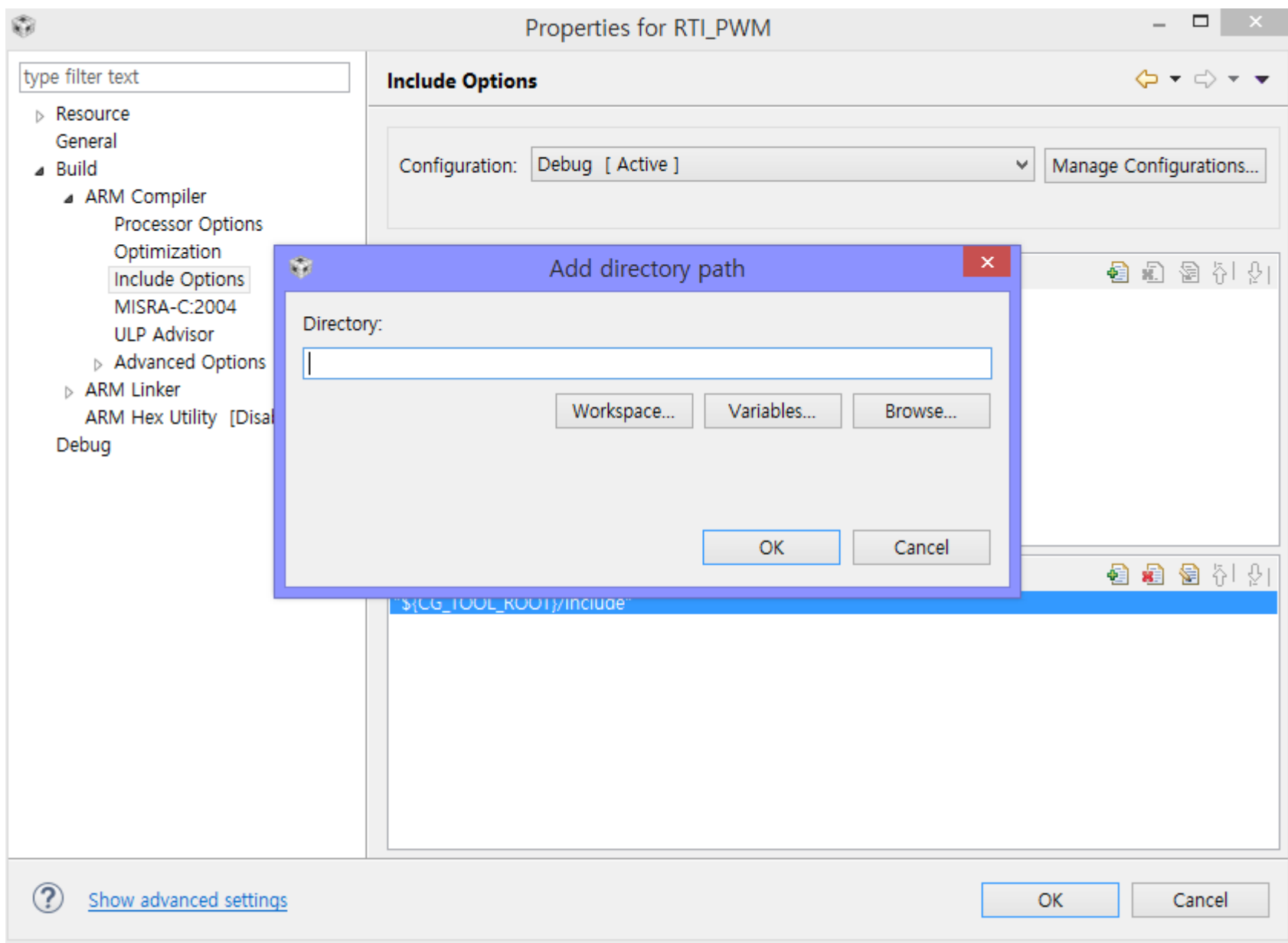
5000 을 설정해본다.



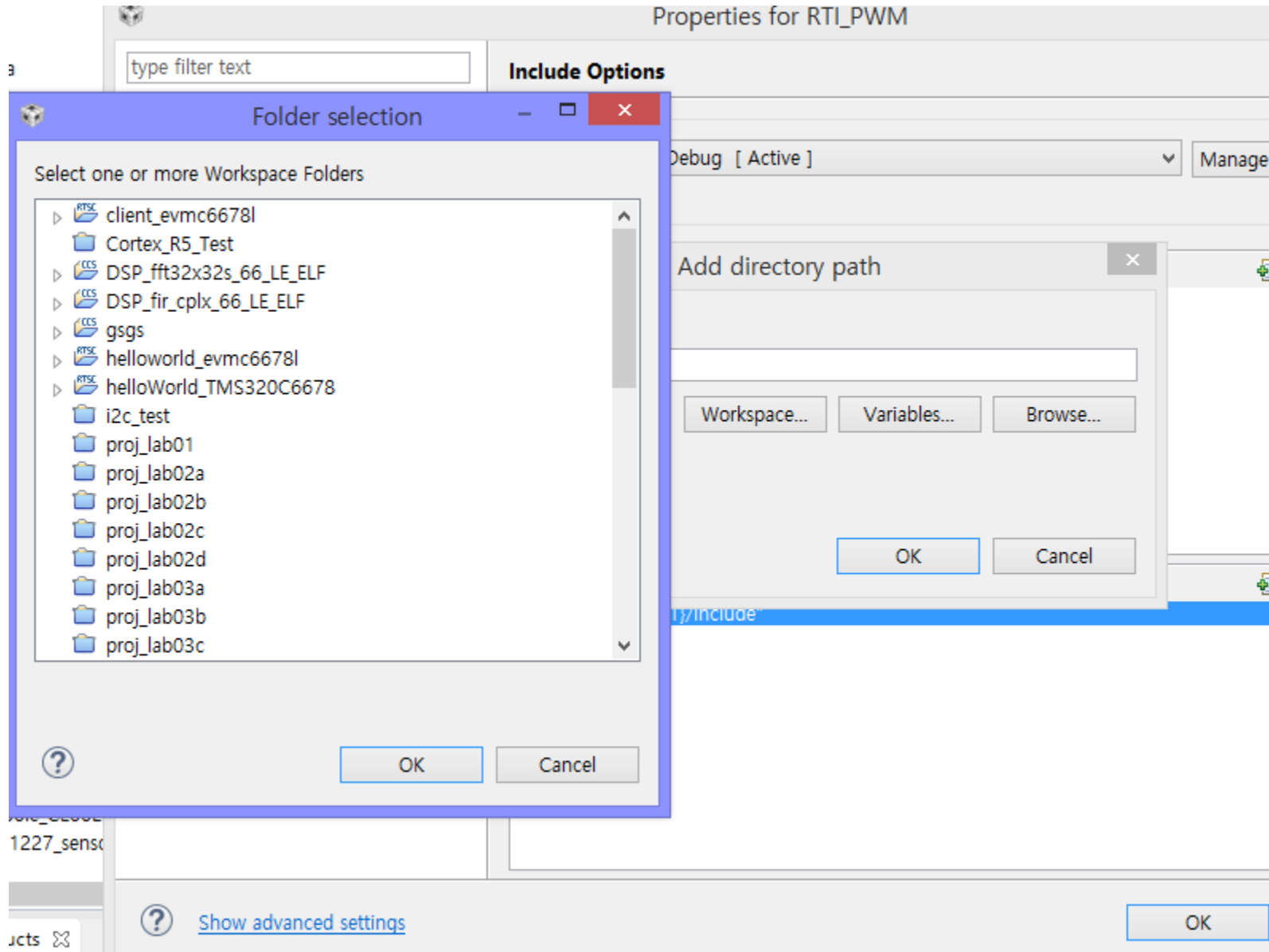








3



11  
12a  
12  
12  
13  
13  
14  
14  
15  
15  
15  
15  
17  
17  
19  
19  
10  
10  
11  
11  
11  
  
K  
M  
  
:book\_0000  
LS1227\_sens  
  
ducts ✕  
on the packa

Properties for RTI\_PWM

type filter text

Include Options

Debug [ Active ]

Mar

Add directory path

Workspace... Variables... Browse...

OK Cancel

ty/include

?

Show advanced settings

OK

Folder selection

Select one or more Workspace Folders

proj\_lab09a

proj\_lab10a

proj\_lab10c

proj\_lab11

proj\_lab11a

proj\_lab11b

RemoteSystemsTempFiles

rti\_blink

RTI\_BLINKY

RTI\_PWM

.settings

Debug

include

source

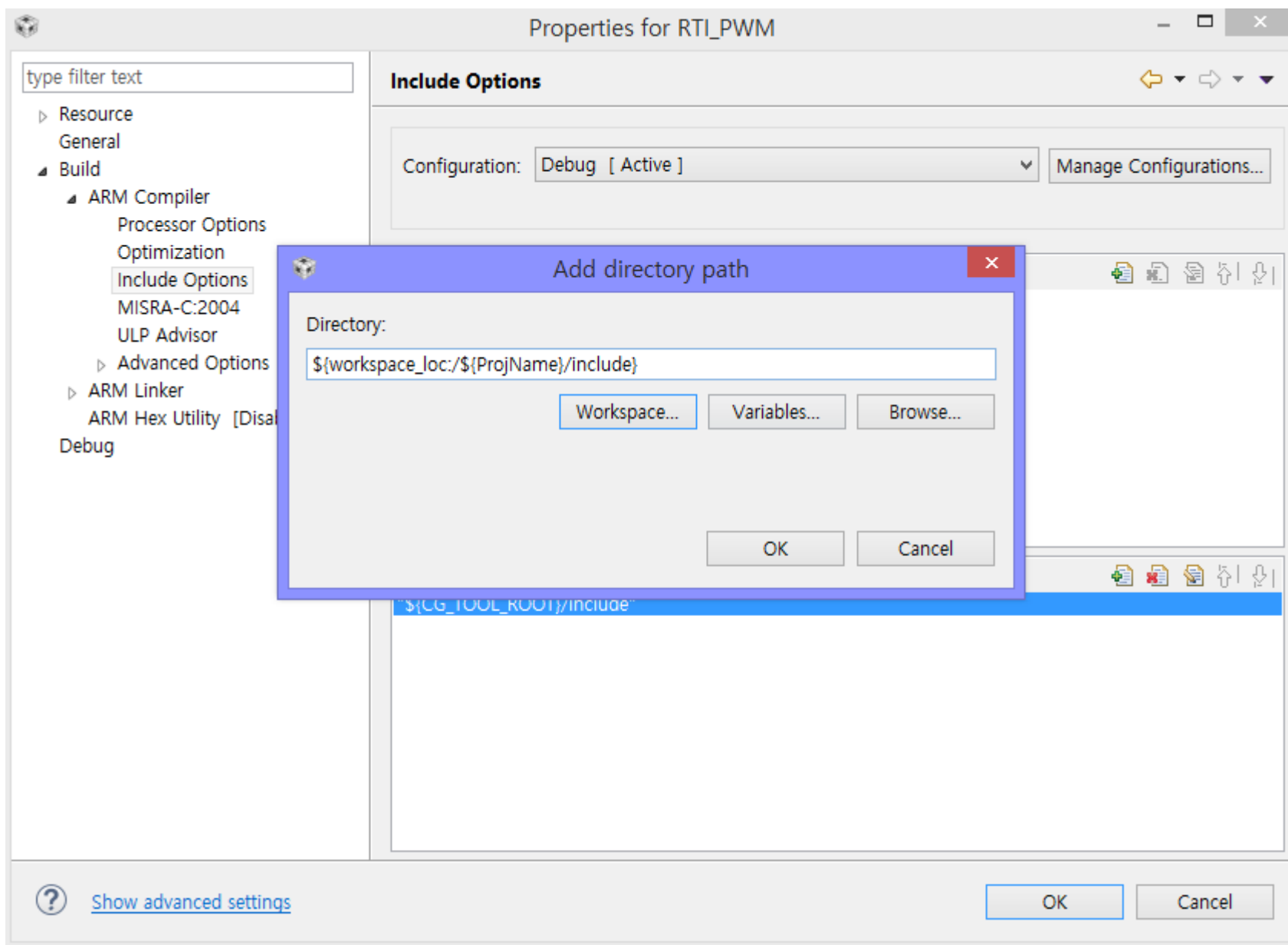
targetConfigs

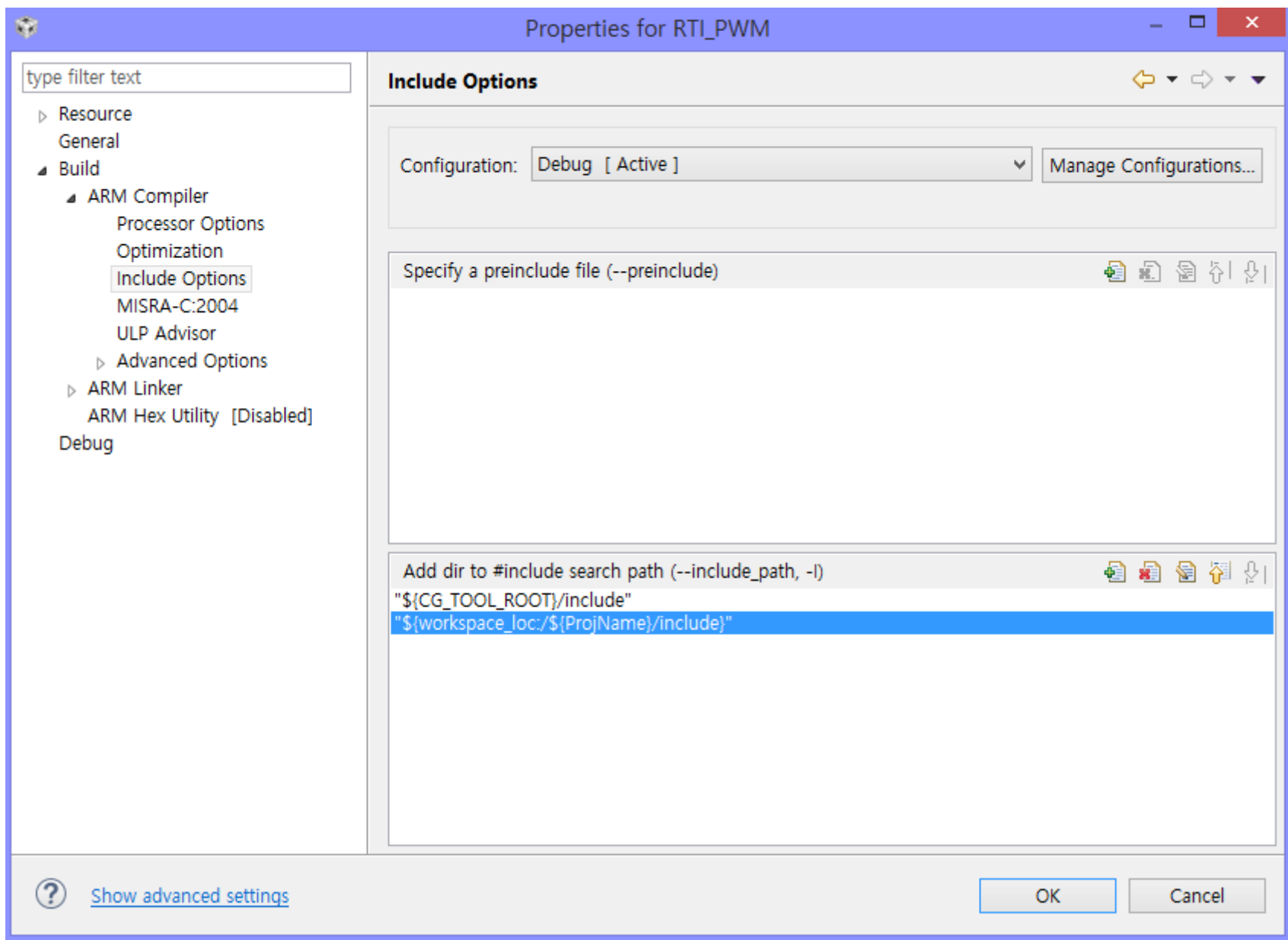
test

?

OK

Cancel





Project Explorer

- proj\_lab03a
- proj\_lab03b
- proj\_lab03c
- proj\_lab04
- proj\_lab04a
- proj\_lab05a
- proj\_lab05b
- proj\_lab05d
- proj\_lab05g
- proj\_lab07
- proj\_lab07a
- proj\_lab09
- proj\_lab09a
- proj\_lab10a
- proj\_lab10c
- proj\_lab11
- proj\_lab11a
- proj\_lab11b
- rti\_blink
- RTI\_BLINKY
- RTI\_PWM [Active - Debug]
  - Binaries
  - Includes
  - Debug
  - include
  - source

Getting Started

TI Resource...

CCS App Center

example\_rtiB...

HL\_sys\_main.c

```
1 #include "HL_sys_common.h"
2 #include "HL_etpwm.h"
3
4 void main(void)
5 {
6     int i;
7     unsigned short val = 0;
8
9     etpwmInit();
10
11     for(;;)
12     {
13         etpwmStartTBCLK();
14
15         for(i = 0; i < 100000; i++)
16             ;
17
18         etpwmSetCmpB(etpwmREG2, val);
19
20         val += 1;
21
22         etpwmStopTBCLK();
23
24         if(val == 500)
25             val = 0;
26     }
27 }
```



이제 코드를 해석해보도록 하는데 먼저 etpwmInit() 부터 살펴보자!

```
/** @fn void etpwmInit(void)
 * @brief eTPWM Driver 를 초기화한다.
 *
 * 이 함수는 eTPWM Module 을 초기화한다.
 *
 * @note 이 함수는 Time-Base Counter 를 Up-Count Mode 로 설정한다.
 * App 은 이 Driver 에서 다른 함수를 사용하여 다른 Mode 로 Module 을 구성할 수 있다.
 * 해당 케이스의 경우, App 은 etpwmInit 함수를 호출할 필요가 없다.
 * 이 함수 호출 이전에 pinmuxInit 이 호출되어야 한다.
 */
void etpwmInit(void)
{
    /** ETPWM2 를 초기화한다. */

    /** - High Speed Time-Base Clock 프리스케일 비트를 설정한다. */
    etpwmREG2->TBCTL = (uint16)0U << 7U;

    /** - Time-Base Clock 프리스케일 비트를 설정한다. */
    etpwmREG2->TBCTL |= (uint16)((uint16)0U << 10U);

    /** - PWMA 와 PWMB 모두 차단하는 ETPWM 을 위한 Time 주기 혹은 주파수를 설정한다. */
    etpwmREG2->TBPRD = 374U;

    /** - PWMA 에 대한 Duty Cycle 을 설정한다. */
    etpwmREG2->CMPA = 188U;

    /** - PWMB 에 대한 Duty Cycle 을 설정한다. */
    etpwmREG2->CMPB = 188U;
```



```

/** - Counter 가 0 에 도달할 때 EPWMxA 출력을 높이고 Counter 가 A 값에 도달할 때 EPWMxA 출력을 낮춘다. */
etpwmREG2->AQCTLA = ((uint16)((uint16)ActionQual_Set << 0U)
| (uint16)((uint16)ActionQual_Clear << 4U));

/** - Counter 가 0 에 도달할 때 EPWMxB 출력을 높이고 Counter 가 B 값에 도달할 때 EPWMxB 출력을 낮춘다. */
etpwmREG2->AQCTLB = ((uint16)((uint16)ActionQual_Set << 0U)
| (uint16)((uint16)ActionQual_Clear << 8U));

/** - Dead Band Module 의 Mode 를 설정한다.
* -Dead Band Module 을 위한 입력 모드를 선택한다.
* -Dead Band Module 을 위한 출력 모드를 선택한다.
* -출력 PWM 의 극성을 선택한다.
*/
etpwmREG2->DBCTL = ((uint16)((uint16)0U << 5U) /* 하강 에지 지연을 위한 Source(0-PWMA, 1-PWMB) */
| (uint16)((uint16)0U << 4U) /* 상승 에지 지연을 위한 Source(0-PWMA, 1-PWMB) */
| (uint16)((uint16)0U << 3U) /* EPWMxB 반전 활성화/비활성화 */
| (uint16)((uint16)0U << 2U) /* EPWMxA 반전 활성화/비활성화 */
| (uint16)((uint16)0U << 1U) /* 상승 에지 지연 활성화/비활성화 */
| (uint16)((uint16)0U << 0U)); /* 하강 에지 지연 활성화/비활성화 */

/** - 상승 에지 지연 설정 */
etpwmREG2->DBRED = 110U;

/** - 하강 에지 지연 설정 */
etpwmREG2->DBFED = 110U;

/** - ETPWMx 를 위한 Chopper Module 활성화
* -Chopper 변조된 Wave 로 원샷 펄스폭을 설정한다.
* -후속 펄스 열에 대한 Duty Cycle 을 설정한다.
* -후속 펄스 열에 대한 기간을 설정한다.
*/
etpwmREG2->PCCTL = ((uint16)((uint16)0U << 0U) /* Chopper Module 활성화/비활성화 */
| (uint16)((uint16)0U << 1U) /* One-Shot 펄스폭 */
| (uint16)((uint16)3U << 8U) /* Chopping Clock Duty Cycle */
| (uint16)((uint16)0U << 5U)); /* Chopping Clock Frequency */

```

```

/** - Trip Source 활성화 설정 */
etpwmREG2->TZSEL = 0x0000U    /** - One-Shot Trip Source 로 TZ1 활성화/비활성화 */
    | 0x0000U    /** - One-Shot Trip Source 로 TZ2 활성화/비활성화 */
    | 0x0000U    /** - One-Shot Trip Source 로 TZ3 활성화/비활성화 */
    | 0x0000U    /** - One-Shot Trip Source 로 TZ4 활성화/비활성화 */
    | 0x0000U    /** - One-Shot Trip Source 로 TZ5 활성화/비활성화 */
    | 0x0000U    /** - One-Shot Trip Source 로 TZ6 활성화/비활성화 */
    | 0x0000U    /** - CBC Trip Source 로 TZ1 활성화/비활성화 */
    | 0x0000U    /** - CBC Trip Source 로 TZ1 활성화/비활성화 */
    | 0x0000U    /** - CBC Trip Source 로 TZ1 활성화/비활성화 */
    | 0x0000U    /** - CBC Trip Source 로 TZ1 활성화/비활성화 */
    | 0x0000U    /** - CBC Trip Source 로 TZ1 활성화/비활성화 */
    | 0x0000U;    /** - CBC Trip Source 로 TZ1 활성화/비활성화 */

/** - 인터럽트 활성화 설정 */
etpwmREG2->TZEINT = 0x0000U    /** - Digital 비교기 출력 A 이벤트 1 활성화/비활성화 */
    | 0x0000U    /** - Digital 비교기 출력 A 이벤트 2 활성화/비활성화 */
    | 0x0000U    /** - Digital 비교기 출력 A 이벤트 1 활성화/비활성화 */
    | 0x0000U    /** - Digital 비교기 출력 A 이벤트 2 활성화/비활성화 */
    | 0x0000U    /** - One-Shot 인터럽트 생성 활성화/비활성화 */
    | 0x0000U;    /** - Cycle-by-Cycle 인터럽트 생성 활성화/비활성화 */

```

```

/** - 인터럽트를 위한 Event 설정 */
etpwmREG2->ETSEL = (uint16)NO_EVENT;

if ((etpwmREG2->ETSEL & 0x0007U) != 0U)
{
    etpwmREG2->ETSEL |= 0x0008U;
}
/** - 인터럽트 생성 빈도 설정 */
etpwmREG2->ETPS = 1U;

/** - ADC SOC 인터럽트 설정 */
etpwmREG2->ETSEL |= ((uint16)(0x0000U)
                    | (uint16)(0x0000U)
                    | (uint16)((uint16)DCAEVT1 << 8U)
                    | (uint16)((uint16)DCBEVT1 << 12U));

/** - ADC SOC 기간 설정 */
etpwmREG2->ETPS |= ((uint16)((uint16)1U << 8U)
                  | (uint16)((uint16)1U << 12U));
}

```

먼저 etpwmREG2 에 대해 알아보도록 하자 ~

```
184 #define etpwmREG2 ((etpwmBASE_t *)0xFCF78D00U)
```

다음으로 etpwmBASE\_t 에 대해 알아보자!

```
typedef volatile struct etpwmBASE
{
    uint16 TBSTS;           /**< 0x0000 Time-Base Status Register*/
    uint16 TBCTL;           /**< 0x0002 Time-Base Control Register*/
    uint16 TBPHS;           /**< 0x0004 Time-Base Phase Register*/
    uint16 rsvd1;           /**< 0x0006 Reserved*/
    uint16 TBPRD;           /**< 0x0008 Time-Base Period Register*/
    uint16 TBCTR;           /**< 0x000A Time-Base Counter Register*/
    uint16 CMPCTL;          /**< 0x000C Counter-Compare Control Register*/
    uint16 rsvd2;           /**< 0x000E Reserved*/
    uint16 CMPA;            /**< 0x0010 Counter-Compare A Register*/
    uint16 rsvd3;           /**< 0x0012 Reserved*/
    uint16 AQCTLA;          /**< 0x0014 Action-Qualifier Control Register for Output A (ETPWMxA)*/
    uint16 CMPB;            /**< 0x0016 Counter-Compare B Register*/
    uint16 AQSFRC;          /**< 0x0018 Action-Qualifier Software Force Register*/
    uint16 AQCTLB;          /**< 0x001A Action-Qualifier Control Register for Output B (ETPWMxB)*/
    uint16 DBCTL;           /**< 0x001C Dead-Band Generator Control Register*/
    uint16 AQCSFRC;         /**< 0x001E Action-Qualifier Continuous S/W Force Register Set*/
    uint16 DBFED;           /**< 0x0020 Dead-Band Generator Falling Edge Delay Count Register*/
    uint16 DBRED;           /**< 0x0022 Dead-Band Generator Rising Edge Delay Count Register*/
    uint16 TZDCSEL;         /**< 0x0024 Trip Zone Digital Compare Select Register*/
    uint16 TZSEL;           /**< 0x0026 Trip-Zone Select Register*/
    uint16 TZEINT;          /**< 0x0028 Trip-Zone Enable Interrupt Register*/
    uint16 TZCTL;           /**< 0x002A Trip-Zone Control Register*/
    uint16 TZCLR;           /**< 0x002C Trip-Zone Clear Register*/
    uint16 TZFLG;           /**< 0x002E Trip-Zone Flag Register*/
}
```

```

uint16 ETSEL;          /**< 0x0030 Event-Trigger Selection Register*/
uint16 TZFRC;          /**< 0x0032 Trip-Zone Force Register*/
uint16 ETFLG;          /**< 0x0034 Event-Trigger Flag Register*/
uint16 ETPS;           /**< 0x0036 Event-Trigger Pre-Scale Register*/
uint16 ETFRC;          /**< 0x0038 Event-Trigger Force Register*/
uint16 ETCLR;          /**< 0x003A Event-Trigger Clear Register*/
uint16  rsvd4;          /**< 0x003C Reserved*/
uint16 PCCTL;          /**< 0x003E PWM-Chopper Control Register*/
uint16  rsvd5[16U];     /**< 0x0040 Reserved*/
uint16 DCACTL;         /**< 0x0060 Digital Compare A Control Register*/
uint16 DCTRIPSEL;      /**< 0x0062 Digital Compare Trip Select Register*/
uint16 DCFCTL;         /**< 0x0064 Digital Compare Filter Control Register*/
uint16 DCBCTL;         /**< 0x0066 Digital Compare B Control Register*/
uint16 DCFOFFSET;      /**< 0x0068 Digital Compare Filter Offset Register*/
uint16 DCCAPCTL;       /**< 0x006A Digital Compare Capture Control Register*/
uint16 DCFWINDOW;      /**< 0x006C Digital Compare Filter Window Register*/
uint16 DCFOFFSETCNT;   /**< 0x006E Digital Compare Filter Offset Counter Register*/
uint16 DCCAP;          /**< 0x0070 Digital Compare Counter Capture Register*/
uint16 DCFWINDOWCNT;   /**< 0x0072 Digital Compare Filter Window Counter Register*/
} etpwmBASE_t;

```

Figure 2-3. Memory Map

0xFFFFFFFF	SYSTEM Peripherals - Frame 1
0xFFFF80000	
0xFFFF7FFFF	
0xFF000000	Peripherals - Frame 3
0xFEFFFFFF	CRC1
0xFE000000	
	RESERVED
0xFCFFFFFF	Peripherals - Frame 2
0xFC000000	
0xFBFFFFFF	
0xFB000000	CRC2
	RESERVED
0xF047FFFF	Flash (Flash ECC, OTP and EEPROM accesses)
0xF0000000	
	RESERVED
0x9FFFFFFF	EMIF (128MB) SDRAM
0x80000000	

	RESERVED
0x6FFFFFFF	reserved0x6C000000
	CS4 0x68000000
	CS3 0x64000000
0x60000000	CS2
	EMIF (16MB * 3)
	Async RAM
	RESERVED
0x37FFFFFF	R5F-1 Cache
0x34000000	
0x33FFFFFF	R5F-0 Cache
0x30000000	
	RESERVED
0x0847FFFF	RAM - ECC
0x08400000	
	RESERVED
0x0807FFFF	RAM (512kB)
0x08000000	
	RESERVED
0x003FFFFF	Flash (4MB)
0x00000000	

Registers/Memories under PCR2 (Peripheral Segment 2)						
CPPI Memory Slave (Ethernet RAM)	PCS[41]	0xFC52_0000	0xFC52_1FFF	8kB	8kB	Abort
CPGMAC Slave (Ethernet Slave)	PS[30]-PS[31]	0xFCF7_8000	0xFCF7_87FF	2kB	2kB	No Error
CPGMACSS Wrapper (Ethernet Wrapper)	PS[29]	0xFCF7_8800	0xFCF7_88FF	256B	256B	No Error
Ethernet MDIO Interface	PS[29]	0xFCF7_8900	0xFCF7_89FF	256B	256B	No Error
ePWM1	PS[28]	0xFCF7_8C00	0xFCF7_8CFF	256B	256B	Abort
ePWM2		0xFCF7_8D00	0xFCF7_8DFF	256B	256B	Abort
ePWM3		0xFCF7_8E00	0xFCF7_8EFF	256B	256B	Abort
ePWM4		0xFCF7_8F00	0xFCF7_8FFF	256B	256B	Abort
ePWM5	PS[27]	0xFCF7_9000	0xFCF7_90FF	256B	256B	Abort
ePWM6		0xFCF7_9100	0xFCF7_91FF	256B	256B	Abort
ePWM7		0xFCF7_9200	0xFCF7_92FF	256B	256B	Abort
eCAP1		0xFCF7_9300	0xFCF7_93FF	256B	256B	Abort
eCAP2	PS[26]	0xFCF7_9400	0xFCF7_94FF	256B	256B	Abort
eCAP3		0xFCF7_9500	0xFCF7_95FF	256B	256B	Abort
eCAP4		0xFCF7_9600	0xFCF7_96FF	256B	256B	Abort
eCAP5		0xFCF7_9700	0xFCF7_97FF	256B	256B	Abort
eCAP6	PS[25]	0xFCF7_9800	0xFCF7_98FF	256B	256B	Abort
eQEP1		0xFCF7_9900	0xFCF7_99FF	256B	256B	Abort
eQEP2		0xFCF7_9A00	0xFCF7_9AFF	256B	256B	Abort
PCR2 registers	PPSE[4]-PPSE[5]	0xFCFF_1000	0xFCFF_17FF	2kB	2kB	
NMPU (CPGMAC)	PPSE[6]	0xFCFF_1800	0xFCFF_18FF	512B	512B	Abort
EMIF Registers	PPS[2]	0xFCFF_E800	0xFCFF_E8FF	256B	256B	Abort

Address Offset	Name	Description
<b>Time-Base Submodule Registers</b>		
00h	TBSTS	Time-Base Status Register
02h	TBCTL	Time-Base Control Register
04h	TBPHS	Time-Base Phase Register
08h	TBPRD	Time-Base Period Register
0Ah	TBCTR	Time-Base Counter Register
<b>Counter-Compare Submodule Registers</b>		
0Ch	CMPCTL	Counter-Compare Control Register
10h	CMPA	Counter-Compare A Register
16h	CMPB	Counter-Compare B Register
<b>Action-Qualifier Submodule Registers</b>		
14h	AQCTLA	Action-Qualifier Control Register for Output A (EPWMxA)
18h	AQSFRC	Action-Qualifier Software Force Register
1Ah	AQCTLB	Action-Qualifier Control Register for Output B (EPWMxB)
1Eh	AQCSFRC	Action-Qualifier Continuous S/W Force Register Set
<b>Dead-Band Generator Submodule Registers</b>		
1Ch	DBCTL	Dead-Band Generator Control Register
20h	DBFED	Dead-Band Generator Falling Edge Delay Count Register
22h	DBRED	Dead-Band Generator Rising Edge Delay Count Register
<b>Trip-Zone Submodule Registers</b>		
24h	TZDCSEL	Trip Zone Digital Compare Event Select Register
26h	TZSEL	Trip-Zone Select Register
28h	TZEINT	Trip-Zone Enable Interrupt Register
2Ah	TZCTL	Trip-Zone Control Register
2Ch	TZCLR	Trip-Zone Clear Register
2Eh	TZFLG	Trip-Zone Flag Register
32h	TZFRC	Trip-Zone Force Register



### 35.4.1.2 Time-Base Control Register (TBCTL)

**Figure 35-64. Time-Base Control Register (TBCTL) [offset = 02h]**

15	14	13	12	10	9	8	
FREE	SOFT	PHSDIR	CLKDIV		HSPCLKDIV		
R/W-0	R/W-0	R/W-0	R/W-0		R/W-0		
7	6	5	4	3	2	1	0
HSPCLKDIV	SWFSYNC	SYNCOSSEL		PRDLD	PHSEN	CTRMODE	
R/W-1	R/W-0	R/W-0		R/W-0	R/W-0	R/W-3h	

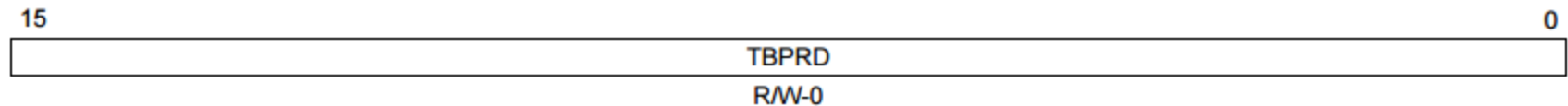
LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

즉 High Speed Time-Base Clock 과 Time-Base Clock 을 분주하지 않고 그대로 사용한다.

12-10	CLKDIV	<div> <div></div> <div>0</div> <div>1h</div> <div>2h</div> <div>3h</div> <div>4h</div> <div>5h</div> <div>6h</div> <div>7h</div> </div>	<p>Time-base Clock Prescale Bits</p> <p>These bits determine part of the time-base clock prescale value.  <math>TBCLK = VCLK3 / (HSPCLKDIV \times CLKDIV)</math></p> <p>/1 (default on reset)</p> <p>/2</p> <p>/4</p> <p>/8</p> <p>/16</p> <p>/32</p> <p>/64</p> <p>/128</p>
9-7	HSPCLKDIV	<div> <div></div> <div>0</div> <div>1h</div> <div>2h</div> <div>3h</div> <div>4h</div> <div>5h</div> <div>6h</div> <div>7h</div> </div>	<p>High Speed Time-base Clock Prescale Bits.</p> <p>These bits determine part of the time-base clock prescale value:  <math>TBCLK = VCLK3 / (HSPCLKDIV \times CLKDIV)</math></p> <p>/1</p> <p>/2 (default on reset)</p> <p>/4</p> <p>/6</p> <p>/8</p> <p>/10</p> <p>/12</p> <p>/14</p>

#### 35.4.1.4 Time-Base Period Register (TBPRD)

**Figure 35-66. Time-Base Period Register (TBPRD) [offset = 08h]**



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-26. Time-Base Period Register (TBPRD) Field Descriptions**

Bits	Name	Description
15-0	TBPRD	<p>These bits determine the period of the time-base counter. This sets the PWM frequency.</p> <p>Shadowing of this register is enabled and disabled by the TBCTL[PRDLD] bit. By default this register is shadowed.</p> <ul style="list-style-type: none"><li>• If TBCTL[PRDLD] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the active register will be loaded from the shadow register when the time-base counter equals zero.</li><li>• If TBCTL[PRDLD] = 1, then the shadow is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware.</li><li>• The active and shadow registers share the same memory map address.</li></ul>

TBPRD 주파수를 374 로 설정한다.

### 35.4.2.2 Counter-Compare A Register (CMPA)

**Figure 35-69. Counter-Compare A Register (CMPA) [offset = 10h]**

15	CMPA	0
	RW-0	

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-29. Counter-Compare A Register (CMPA) Field Descriptions**

Bits	Name	Description
15-0	CMPA	<p>The value in the active CMPA register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare A" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include:</p> <ul style="list-style-type: none"> <li>• Do nothing; the event is ignored.</li> <li>• Clear: Pull the EPWMxA and/or EPWMxB signal low</li> <li>• Set: Pull the EPWMxA and/or EPWMxB signal high</li> <li>• Toggle the EPWMxA and/or EPWMxB signal</li> </ul> <p>Shadowing of this register is enabled and disabled by the CMPCTL[SHDWAMODE] bit. By default this register is shadowed.</p> <ul style="list-style-type: none"> <li>• If CMPCTL[SHDWAMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADAMODE] bit field determines which event will load the active register from the shadow register.</li> <li>• Before a write, the CMPCTL[SHDWAFULL] bit can be read to determine if the shadow register is currently full.</li> <li>• If CMPCTL[SHDWAMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware.</li> <li>• In either mode, the active and shadow registers share the same memory map address.</li> </ul>

Time Base Counter 와 지속적으로 비교하는 값을 188 로 설정하며  
값이 같아졌을때 Counter Compare 모듈에서 Time Base Counter 가 Compare A 와 같다는 이벤트를 발생시킴

### 35.4.2.3 Counter-Compare B Register (CMPB)

**Figure 35-70. Counter-Compare B Register (CMPB) [offset = 16h]**

15	CMPB	0
R/W-0		

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-30. Counter-Compare B Register (CMPB) Field Descriptions**

Bits	Name	Description
15-0	CMPB	<p>The value in the active CMPB register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include:</p> <ul style="list-style-type: none"><li>• Do nothing. event is ignored.</li><li>• Clear: Pull the EPWMxA and/or EPWMxB signal low</li><li>• Set: Pull the EPWMxA and/or EPWMxB signal high</li><li>• Toggle the EPWMxA and/or EPWMxB signal</li></ul> <p>Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed.</p> <ul style="list-style-type: none"><li>• If CMPCTL[SHDWBMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADBMODE] bit field determines which event will load the active register from the shadow register:</li><li>• Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full.</li><li>• If CMPCTL[SHDWBMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware.</li><li>• In either mode, the active and shadow registers share the same memory map address.</li></ul>

### 35.4.3.1 Action-Qualifier Output A Control Register (AQCTLA)

**Figure 35-71. Action-Qualifier Output A Control Register (AQCTLA) [offset = 14h]**

15	12	11	10	9	8
Reserved				CBD	CBU
R-0				R/W-0	R/W-0
7	6	5	4	3	2
CAD		CAU		PRD	ZRO
R/W-0		R/W-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-31. Action-Qualifier Output A Control Register (AQCTLA) Field Descriptions**

Bits	Name	Value	Description
15-12	Reserved	0	Reserved
11-10	CBD	0 Do nothing (action disabled) 1h Clear: force EPWMxA output low. 2h Set: force EPWMxA output high. 3h Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.	Action when the time-base counter equals the active CMPB register and the counter is decrementing.
9-8	CBU	0 Do nothing (action disabled) 1h Clear: force EPWMxA output low. 2h Set: force EPWMxA output high. 3h Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.	Action when the counter equals the active CMPB register and the counter is incrementing.

7-6	CAD		Action when the counter equals the active CMPA register and the counter is decrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
5-4	CAU		Action when the counter equals the active CMPA register and the counter is incrementing.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
3-2	PRD		Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
1-0	ZRO		Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up.
		0	Do nothing (action disabled)
		1h	Clear: force EPWMxA output low.
		2h	Set: force EPWMxA output high.
		3h	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.



```
typedef enum
{
    ActionQual_Disabled = 0U, /** Do nothing (action disabled)
    ActionQual_Clear     = 1U, /** Clear: force ETPWMA/ETPWB output low
    ActionQual_Set       = 2U, /** Set: force ETPWMA/ETPWB output high
    ActionQual_Toggle    = 3U, /** Toggle ETPWMA/ETPWB output

    ForceSize_ActionQual = 0xFFFFU /** Do not use (Makes sure that etpwm
} etpwmActionQual_t;
```

ZRO 가 2 가 됨(1번 비트 활성화)으로써 EPWMxA 출력이 High 가 된다.  
Counter 가 0 이 되면 발생한다.

그리고 ActionQual\_Clear 부분은 4번 비트가 활성화됨으로써  
Counter 가 CMPA 와 같아지면 발생함  
설정값에 의해 EPWMxA 출력을 Low 로 강제함

AQCTLB 부분에서 ZRO 가 2 로 활성화되는 부분은 동일하며

Counter 가 CMPB 와 같아지면 발생하게 된다.  
설정값에 의해 EPWMxB 출력을 Low 로 강제함



### 35.4.4.1 Dead-Band Generator Control Register (DBCTL)

**Figure 35-75. Dead-Band Generator Control Register (DBCTL) [offset = 1Ch]**

15	14						8
HALFCYCLE	Reserved						
R/W-0	R-0						
7	6	5	4	3	2	1	0
Reserved	IN_MODE		POLSEL		OUT_MODE		
R-0	R/W-0		R/W-0		R/W-0		

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-35. Dead-Band Generator Control Register (DBCTL) Field Descriptions**

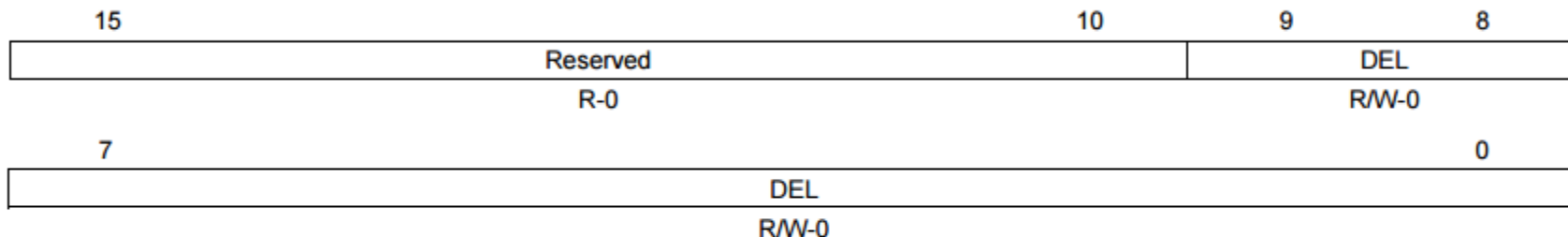
Bits	Name	Value	Description
15	HALFCYCLE	0 1	Half Cycle Clocking Enable Bit: Full cycle clocking enabled. The dead-band counters are clocked at the TBCLK rate. Half cycle clocking enabled. The dead-band counters are clocked at TBCLK × 2.
14-6	Reserved	0	Reserved
5-4	IN_MODE	0 1h 2h 3h	Dead Band Input Mode Control Bit 5 controls the S5 switch and bit 4 controls the S4 switch shown in <a href="#">Figure 35-28</a> . This allows you to select the input source to the falling-edge and rising-edge delay. To produce classical dead-band waveforms the default is EPWMxA In is the source for both falling and rising-edge delays. 0 EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay. 1h EPWMxB In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxA In (from the action-qualifier) is the source for falling-edge delayed signal. 2h EPWMxA In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxB In (from the action-qualifier) is the source for falling-edge delayed signal. 3h EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal.

3-2	POLSEL		<p>Polarity Select Control</p> <p>Bit 3 controls the S3 switch and bit 2 controls the S2 switch shown in <a href="#">Figure 35-28</a>.</p> <p>This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule.</p> <p>The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter.</p> <p>These assume that DBCTL[OUT_MODE] = 1,1 and DBCTL[IN_MODE] = 0,0. Other enhanced modes are also possible, but not regarded as typical usage modes.</p> <p>0 Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default).</p> <p>1h Active low complementary (ALC) mode. EPWMxA is inverted.</p> <p>2h Active high complementary (AHC). EPWMxB is inverted.</p> <p>3h Active low (AL) mode. Both EPWMxA and EPWMxB are inverted.</p>
Bits	Name	Value	Description
1-0	OUT_MODE		<p>Dead-band Output Mode Control</p> <p>Bit 1 controls the S1 switch and bit 0 controls the S0 switch shown in <a href="#">Figure 35-28</a>.</p> <p>This allows you to selectively enable or bypass the dead-band generation for the falling-edge and rising-edge delay.</p> <p>0 Dead-band generation is bypassed for both output signals. In this mode, both the EPWMxA and EPWMxB output signals from the action-qualifier are passed directly to the PWM-chopper submodule.</p> <p>In this mode, the POLSEL and IN_MODE bits have no effect.</p> <p>1h Disable rising-edge delay. The EPWMxA signal from the action-qualifier is passed straight through to the EPWMxA input of the PWM-chopper submodule.</p> <p>The falling-edge delayed signal is seen on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].</p> <p>2h The rising-edge delayed signal is seen on output EPWMxA. The input signal for the delay is determined by DBCTL[IN_MODE].</p> <p>Disable falling-edge delay. The EPWMxB signal from the action-qualifier is passed straight through to the EPWMxB input of the PWM-chopper submodule.</p> <p>3h Dead-band is fully enabled for both rising-edge delay on output EPWMxA and falling-edge delay on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].</p>

Full Cycle 로 구동하며 EPWMxA In(동작 한정자로부터)은 하강 에지와 상승 에지 지연의 Source 이다.  
또한 Active High 모드이며 EPWMxA 와 B 가 반전되지 않고 어떠한 지연도 발생하지 않는다.

### 35.4.4.3 Dead-Band Generator Rising Edge Delay Register (DBRED)

**Figure 35-77. Dead-Band Generator Rising Edge Delay Register (DBRED) [offset = 22h]**



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

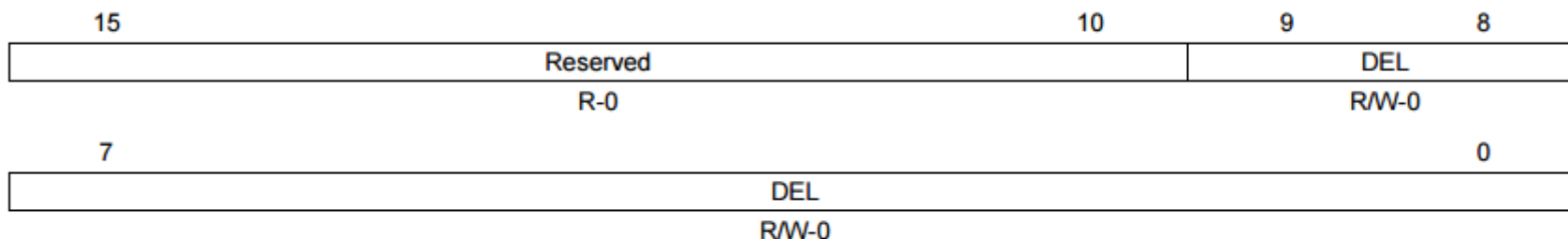
상승 및 하강 에지 지연을 110

**Table 35-37. Dead-Band Generator Rising Edge Delay Register (DBRED) Field Descriptions**

Bits	Name	Description
15-10	Reserved	Reserved
9-0	DEL	Rising Edge Delay Count. 10-bit counter.

### 35.4.4.2 Dead-Band Generator Falling Edge Delay Register (DBFED)

**Figure 35-76. Dead-Band Generator Falling Edge Delay Register (DBFED) [offset = 20h]**



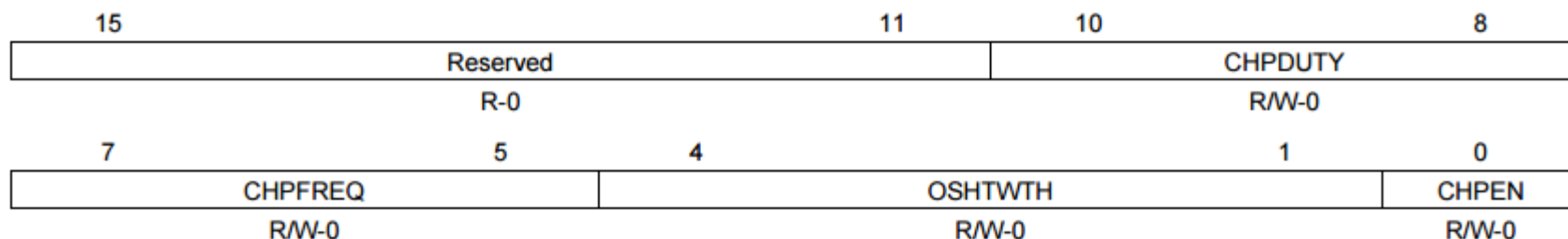
LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-36. Dead-Band Generator Falling Edge Delay Register (DBFED) Field Descriptions**

Bits	Name	Description
15-10	Reserved	Reserved
9-0	DEL	Falling Edge Delay Count. 10-bit counter

### 35.4.7.1 PWM-Chopper Control Register (PCCTL)

**Figure 35-90. PWM-Chopper Control Register (PCCTL) [offset = 3Eh]**



LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-50. PWM-Chopper Control Register (PCCTL) Bit Descriptions**

Bits	Name	Value	Description
15-11	Reserved	0	Reserved
10-8	CHPDUTY	0	Chopping Clock Duty Cycle Duty = 1/8 (12.5%)
		1h	Duty = 2/8 (25.0%)
		2h	Duty = 3/8 (37.5%)
		3h	Duty = 4/8 (50.0%)
		4h	Duty = 5/8 (62.5%)
		5h	Duty = 6/8 (75.0%)
		6h	Duty = 7/8 (87.5%)
		7h	Reserved

7-5	CHPFREQ		Chopping Clock Frequency
		0	Divide by 1 (no prescale, = 12.5 MHz at 100 MHz VCLK3)
		1h	Divide by 2 (6.25 MHz at 100 MHz VCLK3)
		2h	Divide by 3 (4.16 MHz at 100 MHz VCLK3)
		3h	Divide by 4 (3.12 MHz at 100 MHz VCLK3)
		4h	Divide by 5 (2.50 MHz at 100 MHz VCLK3)
		5h	Divide by 6 (2.08 MHz at 100 MHz VCLK3)
		6h	Divide by 7 (1.78 MHz at 100 MHz VCLK3)
		7h	Divide by 8 (1.56 MHz at 100 MHz VCLK3)
4-1	OSHTWTH		One-Shot Pulse Width
		0	1 x VCLK3 / 8 wide ( = 80 nS at 100 MHz VCLK3)
		1h	2 x VCLK3 / 8 wide ( = 160 nS at 100 MHz VCLK3)
		2h	3 x VCLK3 / 8 wide ( = 240 nS at 100 MHz VCLK3)
		3h	4 x VCLK3 / 8 wide ( = 320 nS at 100 MHz VCLK3)
		4h	5 x VCLK3 / 8 wide ( = 400 nS at 100 MHz VCLK3)
		5h	6 x VCLK3 / 8 wide ( = 480 nS at 100 MHz VCLK3)
		6h	7 x VCLK3 / 8 wide ( = 560 nS at 100 MHz VCLK3)
		7h	8 x VCLK3 / 8 wide ( = 640 nS at 100 MHz VCLK3)
		8h	9 x VCLK3 / 8 wide ( = 720 nS at 100 MHz VCLK3)
		9h	10 x VCLK3 / 8 wide ( = 800 nS at 100 MHz VCLK3)
		Ah	11 x VCLK3 / 8 wide ( = 880 nS at 100 MHz VCLK3)
		Bh	12 x VCLK3 / 8 wide ( = 960 nS at 100 MHz VCLK3)
		Ch	13 x VCLK3 / 8 wide ( = 1040 nS at 100 MHz VCLK3)
		Dh	14 x VCLK3 / 8 wide ( = 1120 nS at 100 MHz VCLK3)
		Eh	15 x VCLK3 / 8 wide ( = 1200 nS at 100 MHz VCLK3)
		Fh	16 x VCLK3 / 8 wide ( = 1280 nS at 100 MHz VCLK3)
0	CHPEN		PWM-chopping Enable
		0	Disable (bypass) PWM chopping function
		1	Enable chopping function

PWM Chopping 을 활성화하고 One Shot Pulse 폭을 80ns 로 지정함  
Chopping Clock 주파수를 프리스케일 없이 12.5 MHz 로 지정하고 Duty 는 50%

### 35.4.5.2 Trip-Zone Select Register (TZSEL)

**Figure 35-79. Trip-Zone Select Register (TZSEL) [offset = 26h]**

15	14	13	12	11	10	9	8
DCBEVT1	DCAEVT1	OSHT6	OSHT5	OSHT4	OSHT3	OSHT2	OSHT1
R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 35-39. Trip-Zone Submodule Select Register (TZSEL) Field Descriptions**

Bits	Name	Value	Description
<b>One-Shot (OSHT) Trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until the user clears the condition via the TZCLR register.</b>			
15	DCBEVT1	0 1	Digital Compare Output B Event 1 Select Disable DCBEVT1 as one-shot-trip source for this ePWM module. Enable DCBEVT1 as one-shot-trip source for this ePWM module.
14	DCAEVT1	0 1	Digital Compare Output A Event 1 Select Disable DCAEVT1 as one-shot-trip source for this ePWM module. Enable DCAEVT1 as one-shot-trip source for this ePWM module.
13	OSHT6	0 1	Trip-zone 6 ( $TZ6$ ) Select Disable $TZ6$ as a one-shot trip source for this ePWM module. Enable $TZ6$ as a one-shot trip source for this ePWM module.
12	OSHT5	0 1	Trip-zone 5 ( $TZ5$ ) Select Disable $TZ5$ as a one-shot trip source for this ePWM module Enable $TZ5$ as a one-shot trip source for this ePWM module



11	OSHT4	0 1	Trip-zone 4 (TZ4) Select Disable TZ4 as a one-shot trip source for this ePWM module Enable TZ4 as a one-shot trip source for this ePWM module
10	OSHT3	0 1	Trip-zone 3 (TZ3) Select Disable TZ3 as a one-shot trip source for this ePWM module Enable TZ3 as a one-shot trip source for this ePWM module
9	OSHT2	0 1	Trip-zone 2 (TZ2) Select Disable TZ2 as a one-shot trip source for this ePWM module Enable TZ2 as a one-shot trip source for this ePWM module
8	OSHT1	0 1	Trip-zone 1 (TZ1) Select Disable TZ1 as a one-shot trip source for this ePWM module Enable TZ1 as a one-shot trip source for this ePWM module
<b>Cycle-by-Cycle (CBC) Trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero.</b>			
7	DCBEVT2	0 1	Digital Compare Output B Event 2 Select Disable DCBEVT2 as a CBC trip source for this ePWM module Enable DCBEVT2 as a CBC trip source for this ePWM module
6	DCAEVT2	0 1	Digital Compare Output A Event 2 Select Disable DCAEVT2 as a CBC trip source for this ePWM module Enable DCAEVT2 as a CBC trip source for this ePWM module



Bits	Name	Value	Description
5	CBC6	0 1	Trip-zone 6 (TZ6) Select Disable TZ6 as a CBC trip source for this ePWM module Enable TZ6 as a CBC trip source for this ePWM module
4	CBC5	0 1	Trip-zone 5 (TZ5) Select Disable TZ5 as a CBC trip source for this ePWM module Enable TZ5 as a CBC trip source for this ePWM module
3	CBC4	0 1	Trip-zone 4 (TZ4) Select Disable TZ4 as a CBC trip source for this ePWM module Enable TZ4 as a CBC trip source for this ePWM module
2	CBC3	0 1	Trip-zone 3 (TZ3) Select Disable TZ3 as a CBC trip source for this ePWM module Enable TZ3 as a CBC trip source for this ePWM module
1	CBC2	0 1	Trip-zone 2 (TZ2) Select Disable TZ2 as a CBC trip source for this ePWM module Enable TZ2 as a CBC trip source for this ePWM module
0	CBC1	0 1	Trip-zone 1 (TZ1) Select Disable TZ1 as a CBC trip source for this ePWM module Enable TZ1 as a CBC trip source for this ePWM module

ePWM Module 을 위한 One-Shot-Trip 소스로 DCAEVT1 와 DCBEVT1 을 비활성화한다.  
One-Shot Trip Source 로 TZ1 ~ TZ6 을 비활성화하고  
One-Shot-Trip 소스로 DCAEVT2 와 DCBEVT2 을 비활성화한다.  
CBC Trip Source 로서 TZ1 ~ TZ6 을 비활성화한다.



### 34.4.5.3 Trip-Zone Enable Interrupt Register (TZEINT)

**Figure 34-80. Trip-Zone Enable Interrupt Register (TZEINT) [offset = 2Ah]**

15							8	
Reserved								
R -0								
7	6	5	4	3	2	1	0	
Reserved	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	Reserved	
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 34-40. Trip-Zone Enable Interrupt Register (TZEINT) Field Descriptions**

Bits	Name	Value	Description
15-3	Reserved	0	Reserved
6	DCBEVT2	0 1	Digital Comparator Output B Event 2 Interrupt Enable Disabled Enabled
5	DCBEVT1	0 1	Digital Comparator Output B Event 1 Interrupt Enable Disabled Enabled
DCBEVT1 ~ 2 가 비활성화됨			
4	DCAEVT2	0 1	Digital Comparator Output A Event 2 Interrupt Enable Disabled Enabled
3	DCAEVT1	0 1	Digital Comparator Output A Event 1 Interrupt Enable Disabled Enabled
DCAEVT1 ~ 2 가 비활성화됨			
2	OST	0 1	Trip-zone One-Shot Interrupt Enable Disable one-shot interrupt generation Enable Interrupt generation; a one-shot trip event will cause a EPWMx_TZINT VIM interrupt.
One-Shot 인터럽트 생성 비활성화			
1	CBC	0 1	Trip-zone Cycle-by-Cycle Interrupt Enable Disable cycle-by-cycle interrupt generation. Enable interrupt generation; a cycle-by-cycle trip event will cause an EPWMx_TZINT VIM interrupt.
Cycle-by-Cycle 인터럽트 생성 비활성화			
0	Reserved	0	Reserved

### 34.4.6.1 Event-Trigger Selection Register (ETSEL)

**Figure 34-85. Event-Trigger Selection Register (ETSEL) [offset = 32h]**

15	14	12	11	10	8
SOCBEN	SOCBSEL		SOCAEN	SOCASEL	
R/W-0	R/W-0		R/W-0	R/W-0	
7	4	3	2	0	
Reserved			INTEN	INTSEL	
R-0			R/W-0	R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 34-45. Event-Trigger Selection Register (ETSEL) Field Descriptions**

Bits	Name	Value	Description
15	SOCBEN	0 1	Enable the ADC Start of Conversion B (EPWMxSOCB) Pulse Disable EPWMxSOCB. Enable EPWMxSOCB pulse.
14-12	SOCBSEL	0 1h 2h 3h 4h 5h 6h 7h	EPWMxSOCB Selection Options These bits determine when a EPWMxSOCB pulse will be generated. Enable DCBEVT1.soc event Enable event time-base counter equal to zero. (TBCTR = 0x0000) Enable event time-base counter equal to period (TBCTR = TBPRD) Enable event time-base counter equal to zero or period (TBCTR = 0x0000 or TBCTR = TBPRD). This mode is useful in up-down count mode. Enable event time-base counter equal to CMPA when the timer is incrementing. Enable event time-base counter equal to CMPA when the timer is decrementing. Enable event: time-base counter equal to CMPB when the timer is incrementing. Enable event: time-base counter equal to CMPB when the timer is decrementing.

11	SOCAEN	0	Enable the ADC Start of Conversion A (EPWMxSOCA) Pulse
		1	Disable EPWMxSOCA.
10-8	SOCASEL	1	Enable EPWMxSOCA pulse.
			EPWMxSOCA Selection Options
			These bits determine when a EPWMxSOCA pulse will be generated.
		0	Enable DCAEVT1.soc event
		1h	Enable event time-base counter equal to zero. (TBCTR = 0x0000)
		2h	Enable event time-base counter equal to period (TBCTR = TBPRD)
		3h	Enable event time-base counter equal to zero or period (TBCTR = 0x0000 or TBCTR = TBPRD). This mode is useful in up-down count mode.
		4h	Enable event time-base counter equal to CMPA when the timer is incrementing.
		5h	Enable event time-base counter equal to CMPA when the timer is decrementing.
		6h	Enable event: time-base counter equal to CMPB when the timer is incrementing.
		7h	Enable event: time-base counter equal to CMPB when the timer is decrementing.
7-4	Reserved	0	Reserved
3	INTEN		Enable ePWM Interrupt (EPWMx_INT) Generation
		0	Disable EPWMx_INT generation
		1	Enable EPWMx_INT generation
2-0	INTSEL		ePWM Interrupt (EPWMx_INT) Selection Options
		0	Reserved
		1h	Enable event time-base counter equal to zero. (TBCTR = 0x0000)
		2h	Enable event time-base counter equal to period (TBCTR = TBPRD)
		3h	Enable event time-base counter equal to zero or period (TBCTR = 0x0000 or TBCTR = TBPRD). This mode is useful in up-down count mode.
		4h	Enable event time-base counter equal to CMPA when the timer is incrementing.
		5h	Enable event time-base counter equal to CMPA when the timer is decrementing.
		6h	Enable event: time-base counter equal to CMPB when the timer is incrementing.
		7h	Enable event: time-base counter equal to CMPB when the timer is decrementing.

```

typedef enum
{
    NO_EVENT      = 0U, /** Reserved */
    DCAEVT1       = 0U, /** DCAEVT1.soc event */
    DCBEVT1       = 0U, /** DCBEVT1.soc event */
    CTR_ZERO      = 1U, /** Event CTR = Zero */
    CTR_PRD       = 2U, /** Event CTR = PRD */
    CTR_ZERO_PRD  = 3U, /** Event CTR = Zero or CTR = PRD */
    CTR_UP_CMPA   = 4U, /** Event CTR = CMPA when when the timer is incrementing */
    CTR_D0WM_CMPA = 5U, /** Event CTR = CMPA when when the timer is decrementing */
    CTR_UP_CMPB   = 6U, /** Event CTR = CMPB when when the timer is incrementing */
    CTR_D0WM_CMPB = 7U /** Event CTR = CMPB when when the timer is decrementing */
} etpwmEventSrc_t;

```

Conversion B(EPWMxSOCB) Pulse 의 ADC 를 비활성화하고  
 Conversion A(EPWMxSOCA) Pulse 의 ADC 를 비활성화한다.  
 그리고 EPWMx\_INT 생성을 비활성화한다.

다음으로 DCAEVT1.soc 와 DCBEVT1.soc 이벤트를 활성화한다.

다음으로 첫 번째 이벤트에서 EPWMxSOCA Pulse 를 생성하게 한다.  
 마찬가지로 EPWMxSOCB Pulse 가 생성된다.

### 34.4.6.3 Event-Trigger Prescale Register (ETPS)

**Figure 34-87. Event-Trigger Prescale Register (ETPS) [offset = 34h]**

15	14	13	12	11	10	9	8
SOCBCNT		SOCBPRD		SOCACNT		SOCAPRD	
R-0		R/W-0		R-0		R/W-0	
7		4	3	2	1	0	
Reserved				INTCNT		INTPRD	
R-0				R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 34-47. Event-Trigger Prescale Register (ETPS) Field Descriptions**

Bits	Name	Value	Description
15-14	SOCBCNT	0 1h 2h 3h	ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Counter Register These bits indicate how many selected ETSEL[SOCBSEL] events have occurred: No events have occurred. 1 event has occurred. 2 events have occurred. 3 events have occurred.
13-12	SOCBPRD	0 1h 2h 3h	ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Period Select These bits determine how many selected ETSEL[SOCBSEL] events need to occur before an EPWMxSOCB pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCBEN] = 1). The SOCB pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCB] = 1). Once the SOCB pulse is generated, the ETPS[SOCBCNT] bits will automatically be cleared. 0 Disable the SOCB event counter. No EPWMxSOCB pulse will be generated 1h Generate the EPWMxSOCB pulse on the first event: ETPS[SOCBCNT] = 0,1 2h Generate the EPWMxSOCB pulse on the second event: ETPS[SOCBCNT] = 1,0 3h Generate the EPWMxSOCB pulse on the third event: ETPS[SOCBCNT] = 1,1
11-10	SOCACNT	0 1h 2h 3h	ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Counter Register These bits indicate how many selected ETSEL[SOCASEL] events have occurred: No events have occurred. 1 event has occurred. 2 events have occurred. 3 events have occurred.

9-8	SOCAPRD		<p>ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Period Select</p> <p>These bits determine how many selected ETSEL[SOCASEL] events need to occur before an EPWMxSOCA pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCASEN] = 1). The SOCA pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCACNT] = 1). Once the SOCA pulse is generated, the ETPS[SOCACNT] bits will automatically be cleared.</p> <p>0 Disable the SOCA event counter. No EPWMxSOCA pulse will be generated</p> <p>1h Generate the EPWMxSOCA pulse on the first event: ETPS[SOCACNT] = 0,1</p> <p>2h Generate the EPWMxSOCA pulse on the second event: ETPS[SOCACNT] = 1,0</p> <p>3h Generate the EPWMxSOCA pulse on the third event: ETPS[SOCACNT] = 1,1</p>
7-4	Reserved	0	Reserved
3-2	INTCNT		<p>ePWM Interrupt Event (EPWMx_INT) Counter Register</p> <p>These bits indicate how many selected ETSEL[INTSEL] events have occurred. These bits are automatically cleared when an interrupt pulse is generated. If interrupts are disabled, ETSEL[INT] = 0 or the interrupt flag is set, ETFLG[INT] = 1, the counter will stop counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].</p> <p>0 No events have occurred.</p> <p>1h 1 event has occurred.</p> <p>2h 2 events have occurred.</p> <p>3h 3 events have occurred.</p>
1-0	INTPRD		<p>ePWM Interrupt (EPWMx_INT) Period Select</p> <p>These bits determine how many selected ETSEL[INTSEL] events need to occur before an interrupt is generated. To be generated, the interrupt must be enabled (ETSEL[INT] = 1). If the interrupt status flag is set from a previous interrupt (ETFLG[INT] = 1) then no interrupt will be generated until the flag is cleared via the ETCLR[INT] bit. This allows for one interrupt to be pending while another is still being serviced. Once the interrupt is generated, the ETPS[INTCNT] bits will automatically be cleared.</p> <p>Writing a INTPRD value that is the same as the current counter value will trigger an interrupt if it is enabled and the status flag is clear.</p> <p>Writing a INTPRD value that is less than the current counter value will result in an undefined state.</p> <p>If a counter event occurs at the same instant as a new zero or non-zero INTPRD value is written, the counter is incremented.</p> <p>0 Disable the interrupt event counter. No interrupt will be generated and ETFRC[INT] is ignored.</p> <p>1h Generate an interrupt on the first event INTCNT = 01 (first event)</p> <p>2h Generate interrupt on ETPS[INTCNT] = 1,0 (second event)</p> <p>3h Generate interrupt on ETPS[INTCNT] = 1,1 (third event)</p>

첫번째 이벤트 INTCNT = 01 에서 인터럽트를 생성한다.



```

/** @fn void etpwmStartTBCLK()
 * @brief 모든 eTPWMx Modules 의 Time-Base Clock 을 시작한다.
 *
 * 이 함수는 모든 eTPWMx Modules 의 Time-Base Clock 을 시작한다.
 */
void etpwmStartTBCLK(void)
{
    /* Pin Muxing 활성화 */
    pinMuxReg->KICKER0 = 0x83E70B13U;
    pinMuxReg->KICKER1 = 0x95A4F1E0U;

    pinMuxReg->PINMUX[166U] = (pinMuxReg->PINMUX[166U] & PINMUX_ETPWM_TBCLK_SYNC_MASK) |
                              (PINMUX_ETPWM_TBCLK_SYNC_ON);

    /* Pin Muxing 비활성화 */
    pinMuxReg->KICKER0 = 0x00000000U;
    pinMuxReg->KICKER1 = 0x00000000U;
}

```

먼저 pinMuxReg 부터 살펴보도록 하자

```

/** @def pinMuxReg
 * @brief Pin Muxing Control Register Frame Pointer
 *
 * 이 포인터는 PINMUX Module 레지스터에 접근하기 위해 사용된다.
 */
#define pinMuxReg ((pinMuxBASE_t *) 0xFFFF1C00U)

```

실제 이 Register 는 IOMM 에 있다고 보면 된다.

```
typedef volatile struct pinMuxBase
{
    uint32 REVISION_REG; /**< 0x00: Revision Register */
    uint32 rsvd1[7];      /**<Reserved */
    uint32 BOOT_REG;      /**< 0x20: Boot Mode Register */
    uint32 rsvd2[5];      /**<Reserved */
    uint32 KICKER0;        /**< 0x38: Kicker Register 0 */
    uint32 KICKER1;        /**< 0x3C: Kicker Register 1 */
    uint32 rsvd3[40];     /**<Reserved */
    uint32 ERR_RAW_STATUS_REG; /**< 0xE0: Error Raw Status / Set Register */
    uint32 ERR_ENABLED_STATUS_REG; /**< 0xE4: Error Enabled Status / Clear Register */
    uint32 ERR_ENABLE_REG; /**< 0xE8: Error Signaling Enable Register */
    uint32 ERR_ENABLE_CLR_REG; /**< 0xEC: Error Signaling Enable Clear Register*/
    uint32 rsvd4;         /**<Reserved */
    uint32 FAULT_ADDRESS_REG; /**< 0xF4: Fault Address Register */
    uint32 FAULT_STATUS_REG; /**< 0xF8: Fault Status Register */
    uint32 FAULT_CLEAR_REG; /**< 0xFC: Fault Clear Register */
    uint32 rsvd5[4];      /**< Reserved*/
    uint32 PINMUX[180];   /**< 0x110 - 1A4 :
                           Output Pin Multiplexing Control Registers (38 registers);
                           0x250 - 0x29C : Input Pin Multiplexing Control Registers (20);
                           0X390 - 3DC : Special Functionality Control Registers (20) */
}pinMuxBASE_t;
```

Output Pin MUX 가 148 byte : 38 개

Input Pin MUX 가 76 byte : 20 개

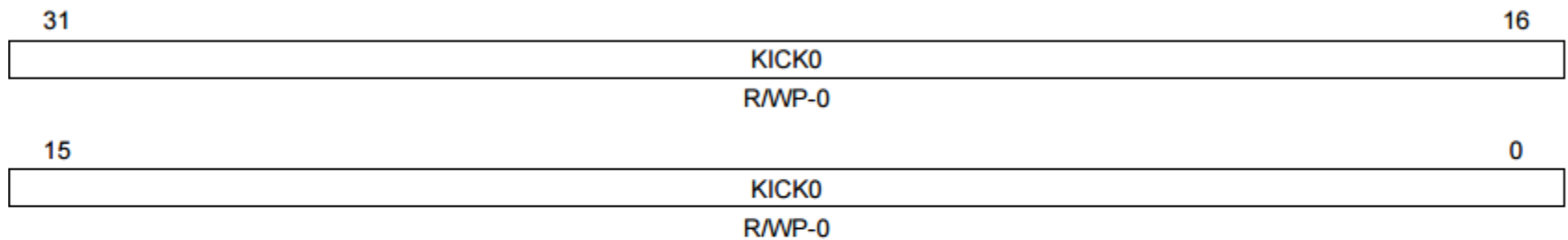
Special Functionality Control Register 76 byte : 20 개



PINMMRnn 레지스터에 대한 CPU write 접근을 unlock 하기 위한 절차의 일부로 0x87E70B13 을 KICK0 에 기록해야 한다.

### 6.7.3 KICK\_REG0: Kicker Register 0

Figure 6-12. KICK\_REG0: Kicker Register 0 (Offset = 38h)



LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -n = value after reset

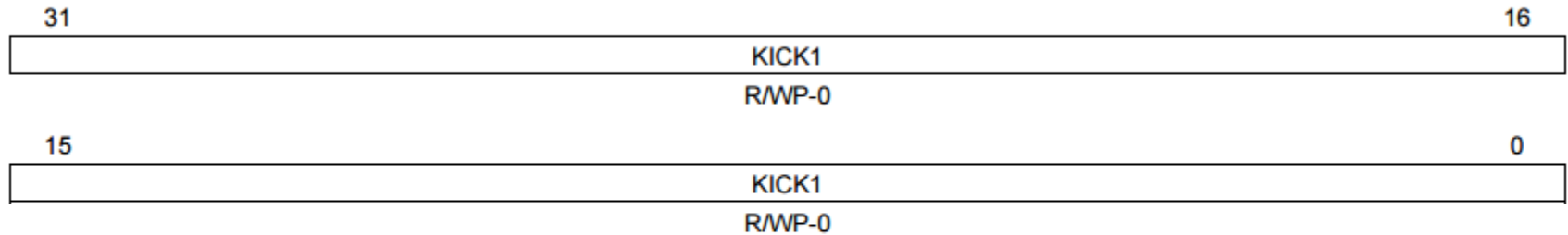
Table 6-15. Kicker Register 0 Field Descriptions

Bit	Field	Description
31-0	KICK0	Kicker 0 Register. The value 83E7 0B13h must be written to KICK0 as part of the process to unlock the CPU write access to the PINMMRnn registers.

PINMMRnn 레지스터에 대한 CPU write 접근을 unlock 하기 위한 절차의 일부로 0x95A4F1E0 을 KICK1 에 기록해야 한다.

#### 6.7.4 KICK\_REG1: Kicker Register 1

Figure 6-13. KICK\_REG1: Kicker Register 1 (Offset = 3Ch)



LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -n = value after reset

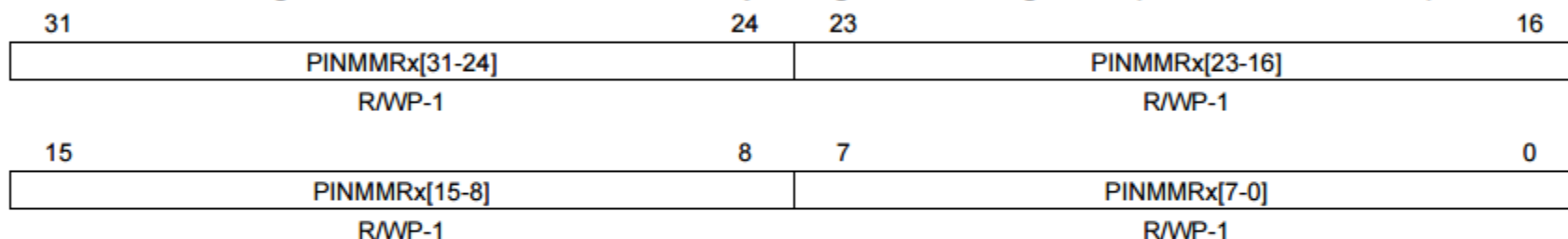
Table 6-16. Kicker Register 1 Field Descriptions

Bit	Field	Description
31-0	KICK1	Kicker 1 Register. The value 95A4 F1E0h must be written to the KICK1 as part of the process to unlock the CPU write access to the PINMMRnn registers.

### 6.7.12 PINMMRnn: Output Pin Multiplexing Control Registers

These registers control the output multiplexing of the functionality available on each pad on the microcontroller. There are 38 such registers – PINMMR0 through PINMMR37. Each 8-bit field of a PINMMR register controls the functionality of a single ball/pin. The mapping between the PINMMRx control registers and the functionality selected on a given terminal is defined in [Table 6-1](#).

**Figure 6-21. PINMMRnn: Pin Multiplexing Control Registers (Offset = 110h-1A4h)**



LEGEND: R/W = Read/Write; R = Read only; WP = Write in privileged mode only; -n = value after reset

**Table 6-24. Pin Multiplexing Control Registers Field Descriptions**

Bit	Field	Value	Description
31-24	PINMMRx[31-24]	1h	Each of these byte-fields control the functionality on a given ball/pin. Please refer to <a href="#">Table 6-1</a> for a list of multiplexed signals.
23-16	PINMMRx[23-16]	1h	
15-8	PINMMRx[15-8]	1h	
7-0	PINMMRx[7-0]	1h	

```
#define PINMUX_ETPWM_TBCLK_SYNC_MASK  
    (~(uint32)((uint32)0xFFU << PINMUX_ETPWM_TBCLK_SYNC_SHIFT))  
  
#define PINMUX_ETPWM_TBCLK_SYNC_SHIFT    0U  
  
#define PINMUX_ETPWM_TBCLK_SYNC_ON  
    ((uint32)((uint32)0x2U << PINMUX_ETPWM_TBCLK_SYNC_SHIFT))
```

166 에 해당하는 부분을 찾아보면 아래와 같은 정보를 IOMM 에서 볼 수 있다.

ePWMx TBCLKSYNC Enable	3A8h	PINMMR166[1]			See <a href="#">Section 6.5.7</a>
------------------------	------	--------------	--	--	-----------------------------------

그리고 다시 Kick 레지스터를 설정하여 write 를 막는다.

### **6.5.7 Control for Synchronizing Time Bases for All ePWMx Modules**

The ePWMx modules implement a mechanism that allows their time bases to be synchronized. This is done by using a signal called TBCLKSYNC, which is a common input to all the ePWMx modules. This TBCLKSYNC is generated by a register bit in the I/O multiplexing module. PINMMR166[1] is the TBCLKSYNC signal. This bit is clear ('0') by default.

When TBCLKSYNC = 0, the time-base clock of all ePWMx modules is stopped. This is the default condition.

When TBCLKSYNC = 1, the time-base clocks of all ePWMx modules are started aligned to the rising edge of the TBCLKSYNC signal.

The correct procedure for enabling and synchronizing the time-base clocks of all the ePWMx modules is:

1. Enable the clocks to the desired individual ePWMx modules if they have been disabled
2. Set TBCLKSYNC = 0. This will stop the time-base clocks of any enabled ePWMx module.
3. Configure the time-base clock prescaler values and desired ePWM modes.
4. Set TBCLKSYNC = 1.

```

/** @def etpwmREG2
 * @brief ETPWM2 Register Frame Pointer
 *
 * 이 포인터는 ETPWM2 레지스터에 접근하기 위한 ETPWM Driver 로 사용된다.
 */
#define etpwmREG2 ((etpwmBASE_t *)0xFCF78D00U)

```

ePWM2 에 해당하는 포인터가 etpwm 으로 넘어온다.  
그리고 val 값은 0 에 해당한다.

```

/** @fn void etpwmSetCmpB(etpwmBASE_t *etpwm, uint16 value)
 * @brief Compare B 값을 설정한다.
 *
 * @param etpwm    PWM(ETPWM) Object Handle(etpwmREG1 ~ 7)
 * @param value    16 비트 Compare B 값
 *
 * 이 함수는 compare B 레지스터를 설정한다.
 */
void etpwmSetCmpB(etpwmBASE_t *etpwm, uint16 value)
{
    etpwm->CMPB = value;
}

```

### 34.4.2.3 Counter-Compare B Register (CMPB)

**Figure 34-70. Counter-Compare B Register (CMPB) [offset = 14h]**

15	CMPB	0
R/W-0		

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

**Table 34-30. Counter-Compare B Register (CMPB) Field Descriptions**

Bits	Name	Description
15-0	CMPB	<p>The value in the active CMPB register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include:</p> <ul style="list-style-type: none"> <li>• Do nothing. event is ignored.</li> <li>• Clear: Pull the EPWMxA and/or EPWMxB signal low</li> <li>• Set: Pull the EPWMxA and/or EPWMxB signal high</li> <li>• Toggle the EPWMxA and/or EPWMxB signal</li> </ul> <p>Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed.</p> <ul style="list-style-type: none"> <li>• If CMPCTL[SHDWBMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADBMODE] bit field determines which event will load the active register from the shadow register:</li> <li>• Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full.</li> <li>• If CMPCTL[SHDWBMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware.</li> <li>• In either mode, the active and shadow registers share the same memory map address.</li> </ul>

```

/** @fn void etpwmStopTBCLK()
 * @brief 모든 eTPWMx Modules 의 Time-Base Clock 을 정지한다.
 *
 * 이 함수는 모든 eTPWMx Modules 의 Time-Base Clock 을 정지한다.
 */
void etpwmStopTBCLK(void)
{
    /* Pin Muxing 활성화 */
    pinMuxReg->KICKER0 = 0x83E70B13U;
    pinMuxReg->KICKER1 = 0x95A4F1E0U;

    pinMuxReg->PINMUX[166U] = (pinMuxReg->PINMUX[166U] & PINMUX_ETPWM_TBCLK_SYNC_MASK) |
                              (PINMUX_ETPWM_TBCLK_SYNC_OFF);

    /* Pin Muxing 비활성화 */
    pinMuxReg->KICKER0 = 0x00000000U;
    pinMuxReg->KICKER1 = 0x00000000U;
}

```