```
1 #include "DSP28x_Project.h"     /* Device Headerfile and Examples Include File */
2
3 #define   SYSTEM_CLOCK      150E6   /* 150MHz */
4 #define   TBCLK             150E6   /* 150MHz */
5 #define   PWM_CARRIER         20E3   /* 20kHz */
```

• **SYSTEM_CLOCK** : 시스템 클럭을 150MHz로 설정, 주기는 1/150MHz=0.0067μs이다.
• **DSP28x_Project.h** : 헤더 파일에는 F2802x_Device.h와 F2802x_Examples.h가 포함되어 있다.
왜냐하면 이름이 devicegeneric하기 때문에, 예제와 사용자 정의 프로젝트는 쉽게 서로 다른 장치 헤더 파일 사이에서 쉽게 port 될 수 있다.
• **TBCLK** : 시스템 클럭의 사전 정의 된 버전(prescaled version of the system clock)으로, 시스템 클럭 내의 모든 하위 항목에 사용된다. 카운터 주파수 결정 변수. 150MHz로 설정.
• **PWM_CARRIER** : PWM 삼각파의 주파수는 20kHz로 설정.

```
8 /* Prototype statements for functions found within this Example */
9 void InitEPwm5Module(void);
10 interrupt void EPwm5Isr(void);
11 interrupt void adc_isr(void);                // ADC 인터럽트 함수 선언
12
13 /* Global variables used in this Example */
14 Uint16   BackTicker;
15 Uint16   EPwm5IsrTicker;
```

- 프로젝트는 EPWM5A, 5B만을 사용하기에, 그에 맞는 함수와 PWM 인터럽트, ADC 인터럽트에 대한 함수를 선언한다.

```
17 interrupt void Xint3_isr(void);
18 interrupt void Xint4_isr(void);
19 interrupt void Xint5_isr(void);
20 interrupt void Xint6_isr(void);
21
22 float32   PwmCarrierFrequency;
23 float32   PwmDutyRatio;
24 float32   FallingEdgeDelay1;
25 float32   RisingEdgeDelay1;
26 float32   PWM_CARRIER2, PWM_CARRIER1;
27
28
29 Uint16 ADC_value01, ADC_value02, deadt = 0;
30 Uint16 Loop_cnt, ADC_cnt;
31 Uint16 SW1_cnt, SW2_cnt, SW3_cnt, SW4_cnt;
```

- 전연변수 설정.
• **FallingEdgeDelay1, RisingEdgeDelay1** : Dead band 모듈에서 사용되는 상승, 하강 지연시간.
• **PWM_CARRIER1, PWM_CARRIER2** : 가변저항 A, B를 사용하는 PWM 주파수 가변 실험에서 사용할 PWM 주파수변수.
• **ADC_value01, ADC_value02** : 가변저항 A, B의 변화에 대한 전압 변동의 상대적인 디지털 값.

```
70 void main(void)
71 {
72
73     /*-------------------------------------------------------------------------------
74        Step 1
75        Disable Global Interrupt & Interrupt Flag Clear
76     -------------------------------------------------------------------------------*/
77     DINT;
78     IER = 0x0000;
79     IFR = 0x0000;
80
81     /*-------------------------------------------------------------------------------
82        Step 2
83        2.1 InitSysCtrl()
84        2.1.1 Disables the watchdog
85        2.1.2 Set the PLLCR for proper SYSCLKOUT frequency
86        2.1.3 Set the pre-scaler for the high and low frequency peripheral clocks
87        2.1.4 Enable the clocks to the peripherals
88        2.2 Initialize GPIO MUX
89     -------------------------------------------------------------------------------*/
90     InitSysCtrl();
91
92     EALLOW;
93     GpioCtrlRegs.GPAPUD.bit.GPIO8 = 0;    /* Enable pull-up on GPIO6 (EPWM4A) */
94     GpioCtrlRegs.GPAPUD.bit.GPIO9 = 0;    /* Enable pull-up on GPIO7 (EPWM4B) */
95     GpioCtrlRegs.GPAMUX1.bit.GPIO8 = 1;    /* Configure GPIO6 as EPWM4A */
96     GpioCtrlRegs.GPAMUX1.bit.GPIO9 = 1;    /* Configure GPIO7 as EPWM4B */
97     SysCtrlRegs.HISPCP.bit.HSPCLK = 1;      // HSPCLK = SYSCLKOUT/(HISPCP*2)
98     EDIS;
```

• **DINT** : 칩 동작에 악영향을 주는 인터럽트 발생을 방지하기 위해서, 시스템 초기화 과정에서 전역 인터럽트 스위치를 끄기 위해, DINT 명령어를 사용한다.

• **IER, IFR** : 인터럽트가 없는 레지스터의 상태로 설정하여, 오류방지.

- IFR(CPU interrupt flag register) : 16비트 CPU 레지스터로, 보류 중인 인터럽트를 식별하고 지우는 데 사용한다. IRF에는 CPU 레벨(INT1 ~ INT14, DLOGINT, RTOSINT)의 모든 maskable한 인터럽트에 대한 flag bits를 포함되어 있다. PIE가 활성화 되면, PIE 모듈은 INT1 ~ INT12의 인터럽트 소스를 멀티 플렉싱한다.

-

• **InitSysCtrl** : 시스템을 초기화한다.

Table 65. GPIO Port A Internal Pullup Disable (GPAPUD) Register Field Descriptions

Bits	Field	Value	Description (1)
31-0	GPIO31-GPIO0		Configure the internal pullup resister on the selected GPIO Port A pin. Each GPIO pin corresponds to one bit in this register.
		0	Enable the internal pullup on the specified pin. (default for GPIO12-GPIO31)
		1	Disable the internal pullup on the specified pin. (default for GPIO0-GPIO11)

• **GpioCtrlReges.GPAPUD.bit.GPIO8 = 0** ; : PWM의 출력신호를 사용하기 위해서, GPAPUD에 0의 값이 들어오면, 지정된 핀(GPIO8)에서 내부 풀업(Pull up)을 활성화한다.

Table 50. GPIO Port A Multiplexing 1 (GPAMUX1) Register Field Descriptions (continued)

17-16	GPIO8		Configure the GPIO8 pin as:
		00	GPIO8 - General purpose I/O 8 (default) (I/O)
		01	EPWM5A - ePWM5 output A (O)
		10	CANTXB - eCAN-B transmit (O)
		11	$\overline{\text{ADCSOCAO}}$ - ADC Start of conversion A

• **GpioCtrlRegs.GPAMUX1.bit.GPIO8 = 1;** : GPIO8의 output을 EPWM5A로 설정하기 위해, 1의 값을 입력한다.

Table 18. High-Speed Peripheral Clock Prescaler (HISPCP) Field Descriptions

Bits	Field	Value	Description (1)
15-3	Reserved		Reserved
2-0	HSPCLK		These bits configure the high-speed peripheral clock (HSPCLK) rate relative to SYSCLKOUT: If HISPCP(2) ≠ 0, HSPCLK = SYSCLKOUT/(HISPCP X 2) If HISPCP = 0, HSPCLK = SYSCLKOUT
		000	High speed clock = SYSCLKOUT/1
		001	High speed clock = SYSCLKOUT/2 (reset default)
		010	High speed clock = SYSCLKOUT/4
		011	High speed clock = SYSCLKOUT/6
		100	High speed clock = SYSCLKOUT/8
		101	High speed clock = SYSCLKOUT/10
		110	High speed clock = SYSCLKOUT/12
		111	High speed clock = SYSCLKOUT/14

(1) This register is EALLOW protected. See Section 7.2 for more information.
(2) HISPCP in this equation denotes the value of bits 2:0 in the HISPCP register.

SysCtrlRegs.HISPCP.bit.HSPCLK = 1; : 1의 값이 들어왔기에, HSPCLK = SYSCLKOUT/(HISPCP*2)로 설정한다.

```
    Step 3
    3.1 Initialize Peripheral Interrupt Expansion circuit
-------------------------------------------------------------------------------*/
InitPieCtrl();
IER = 0x0000;
IFR = 0x0000;
/*-------------------------------------------------------------------------------
    Step 4
    4.1 Pie Vector Table Re-allocation
-------------------------------------------------------------------------------*/
InitPieVectTable();

/*-------------------------------------------------------------------------------
    Step 5
    5.1 Interrupt Service routine re-mapping and Interrupt vector enable
-------------------------------------------------------------------------------*/

/* Interrupt Service Routine Re-mapping */
EALLOW;
PieVectTable.EPWM5_INT = &EPwm5Isr;
PieVectTable.ADCINT = &adc_isr;
EDIS;

EALLOW;
PieVectTable.XINT3 = &Xint3_isr;
PieVectTable.XINT4 = &Xint4_isr;
PieVectTable.XINT5 = &Xint5_isr;
PieVectTable.XINT6 = &Xint6_isr;
EDIS;
```

• InitPieCtrl() : PIE 회로 초기화.

Table 112. PIE Vector Table (continued)

Name	VECTOR ID [1]	Address [2]	Size (x16)	Description [3]		CPU Priority	PIE Group Priority
INT11.6	117	0x0000 0DEA	2	Reserved		15	6
INT11.7	118	0x0000 0DEC	2	Reserved		15	7
INT11.8	119	0x0000 0DEE	2	Reserved		15	8 (lowest)
PIE Group 12 Vectors - Muxed into CPU INT12							
INT12.1	120	0x0000 0DF0	2	XINT3		16	1 (highest)
INT12.2	121	0x0000 0DF2	2	XINT4		16	2
INT12.3	122	0x0000 0DF4	2	XINT5		16	3
INT12.4	123	0x0000 0DF6	2	XINT6		16	4
INT12.5	124	0x0000 0DF8	2	XINT7		16	5
INT12.6	125	0x0000 0DFA	2	Reserved		16	6
INT12.7	126	0x0000 0DFC	2	LVF	FPU	16	7
INT12.8	127	0x0000 0DFE	2	LUF	FPU	16	8 (lowest)

• InitPieVectTable() : 0 ~ 127의 Vector ID를 가지는 PIE vector의 함수 사용.

Table 111. PIE MUXed Peripheral Interrupt Vector Table

INTx.7	INTx.6	INTx.5	INTx.4	INTx.3
TINT0	*ADCINT*	*XINT2*	*XINT1*	Reserved
(TIMER 0)	(ADC)	Ext. int. 2	Ext. int. 1	-
0xD4C	0xD4A	0xD48	0xD46	0xD44
Reserved	*EPWM6_ TZINT*	*EPWM5_TZINT*	*EPWM4_TZINT*	*EPWM3_TZINT*
-	(ePWM6)	(ePWM5)	(ePWM4)	(ePWM3)
0xD5C	0xD5A	0xD58	0xD56	0xD54
Reserved	*EPWM6_ INT*	*EPWM5_INT*	*EPWM4_INT*	*EPWM3_INT*
-	(ePWM6)	(ePWM5)	(ePWM4)	(ePWM3)
0xD6C	0xD6A	0xD68	0xD66	0xD64
Reserved	*ECAP6_INT*	*ECAP5_INT*	*ECAP4_INT*	*ECAP3_INT*
-	(eCAP6)	(eCAP5)	(eCAP4)	(eCAP3)
0xD7C	0xD7A	0xD78	0xD76	0xD74

• PieVectTable.EPWM5_INT, PieVectTable.ADCINT : 각각의 함수(&EPwm5Isr, &adc_isr)의 값으로 vector table을 재설정.

```
//---------------------------------------------------------------------------
// Step 4. GPIO 초기화
//---------------------------------------------------------------------------
EALLOW;
GpioCtrlRegs.GPBMUX1.bit.GPIO44 = 0;        // 핀 기능선택: GPIO44
GpioCtrlRegs.GPBMUX1.bit.GPIO45 = 0;        // 핀 기능선택: GPIO45
GpioCtrlRegs.GPBMUX1.bit.GPIO46 = 0;        // 핀 기능선택: GPIO46
GpioCtrlRegs.GPBMUX1.bit.GPIO47 = 0;        // 핀 기능선택: GPIO47
GpioCtrlRegs.GPBDIR.bit.GPIO44 = 0;           // GPIO44 입출력 선택: Input
GpioCtrlRegs.GPBDIR.bit.GPIO45 = 0;           // GPIO45 입출력 선택: Input
GpioCtrlRegs.GPBDIR.bit.GPIO46 = 0;           // GPIO46 입출력 선택: Input
GpioCtrlRegs.GPBDIR.bit.GPIO47 = 0;           // GPIO47 입출력 선택: Input
EDIS;
//---------------------------------------------------------------------------
```

• GpioCtrlRegs.GPBDIR.bit.GPIO44 = 0; : "0"의 값이 입력되었기에, 해당 GPIO 핀을 입력으로 구성 (기본값).

Table 63. GPIO Port B Direction (GPBDIR) Register Field Descriptions

Bits	Field	Value	Description [1]
31-0	GPIO63-GPIO32		Controls direction of GPIO pin when GPIO mode is selected. Reading the register returns the current value of the register setting
		0	Configures the GPIO pin as an input. (default)
		1	Configures the GPIO pin as an output

(1) This register is EALLOW protected. See Section 7.2 for more information.

```
// Step 5. Qualification 초기화
//-----------------------------------------------------------------------------------
EALLOW;
GpioCtrlRegs.GPBCTRL.bit.QUALPRD1 = 0xFF;    // (GPIO40~GPIO47) Qual period 설정
GpioCtrlRegs.GPBQSEL1.bit.GPIO44 = 2;       // Qualification using 6 samples
GpioCtrlRegs.GPBQSEL1.bit.GPIO45 = 2;       // Qualification using 6 samples
GpioCtrlRegs.GPBQSEL1.bit.GPIO46 = 2;       // Qualification using 6 samples
GpioCtrlRegs.GPBQSEL1.bit.GPIO47 = 2;       // Qualification using 6 samples
EDIS;
```

Table 60. GPIO Port B Qualification Select 1 (GPBQSEL1) Register Field Descriptions

Bits	Field	Value	Description (1)
31-0	GPIO47-GPIO32		Select input qualification type for GPIO32 to GPIO47. The input qualification of each GPIO input is controlled by two bits as shown in Figure 55.
		00	Synchronize to SYSCLKOUT only. Valid for both peripheral and GPIO pins.
		01	Qualification using 3 samples. Valid for pins configured as GPIO or a peripheral function. The time between samples is specified in the GPACTRL register.
		10	Qualification using 6 samples. Valid for pins configured as GPIO or a peripheral function. The time between samples is specified in the GPACTRL register.
		11	Asynchronous. (no synchronization or qualification). This option applies to pins configured as peripherals only. If the pin is configured as a GPIO input, then this option is the same as 0,0 or synchronize to SYSCLKOUT.

• GpioCtrlRegs.GPBQSEL1.bit.GPIO44 = 2; : GPIO32 ~ GPIO47까지 입력 자격 유형을 선택한다. 각 GPIO의 입력자격은 2비트로 제어된다. "2"가 입력되었기에, "Qualification using 6 samples." . GPIO 혹은 주변 기능으로 구성된 핀을 유효화 한다.

```
// Step 6. XINT 초기화
//-----------------------------------------------------------------------------------
EALLOW;
GpioIntRegs.GPIOXINT3SEL.bit.GPIOSEL = 47;    // 외부 인터럽트 XINT3로 사용할 핀 선택: GPIO47
GpioIntRegs.GPIOXINT4SEL.bit.GPIOSEL = 46;    // 외부 인터럽트 XINT4로 사용할 핀 선택: GPIO46
GpioIntRegs.GPIOXINT5SEL.bit.GPIOSEL = 45;    // 외부 인터럽트 XINT5로 사용할 핀 선택: GPIO45
GpioIntRegs.GPIOXINT6SEL.bit.GPIOSEL = 44;    // 외부 인터럽트 XINT6로 사용할 핀 선택: GPIO44
EDIS;

XIntruptRegs.XINT3CR.bit.POLARITY = 0;       // XINT3 인터럽트 발생 조건 설정: 입력 신호의 하강 엣지
XIntruptRegs.XINT4CR.bit.POLARITY = 0;       // XINT4 인터럽트 발생 조건 설정: 입력 신호의 상승 엣지
XIntruptRegs.XINT5CR.bit.POLARITY = 0;       // XINT5 인터럽트 발생 조건 설정: 입력 신호의 하강 엣지
XIntruptRegs.XINT6CR.bit.POLARITY = 0;       // XINT6 인터럽트 발생 조건 설정: 입력 신호의 하강 & 상승 엣지

XIntruptRegs.XINT3CR.bit.ENABLE = 1;         // XINT3 인터럽트 : Enable
XIntruptRegs.XINT4CR.bit.ENABLE = 1;         // XINT4 인터럽트 : Enable
XIntruptRegs.XINT5CR.bit.ENABLE = 1;         // XINT5 인터럽트 : Enable
XIntruptRegs.XINT6CR.bit.ENABLE = 1;         // XINT6 인터럽트 : Enable

// 외부 인터터트 포함된 백터 활성화
PieCtrlRegs.PIEIER12.bit.INTx1 = 1;          // PIE 인터럽트(XINT3) : Enable
PieCtrlRegs.PIEIER12.bit.INTx2 = 1;          // PIE 인터럽트(XINT4) : Enable
PieCtrlRegs.PIEIER12.bit.INTx3 = 1;          // PIE 인터럽트(XINT5) : Enable
PieCtrlRegs.PIEIER12.bit.INTx4 = 1;          // PIE 인터럽트(XINT6) : Enable
IER |= M_INT12;                          // CPU 인터럽트(INT12) : Enable
//-----------------------------------------------------------------------------------
```

Table 80. GPIO XINTn Interrupt Select (GPIOXINTnSEL)(1) Register Field Descriptions

Bits	Field	Value	Description (2)
15-5	Reserved		Reserved
4-0	GPIOXINTnSEL		Select the port A GPIO signal (GPIO0 - GPIO31) that will be used as the XINT1 or XINT2 interrupt source. In addition, you can configure the interrupt in the XINT1CR or XINT2CR registers described in Section 8.6. To use XINT2 as ADC start of conversion, enable it in the ADCTRL2 register. The ~~ADCSOC~~ signal is always rising edge sensitive.
		00000	Select the GPIO0 pin as the XINTn interrupt source (default)
		00001	Select the GPIO1 pin as the XINTn interrupt source
		. . .	. . .
		11110	Select the GPIO30 pin as the XINTn interrupt source
		11111	Select the GPIO31 pin as the XINTn interrupt source

• GpioIntRegs.GPIOXINT3SEL.bit.GPIOSEL = 47; : XINT1 ~ XINT2의 인터럽트 소스로 사용될 XINT1 또는 XINT2로 사용할 포트 A GPIO 신호(GPIO0 ~ GPIO31)를 선택한다.

Table 83. XINT3 - XINT7 Interrupt Select and Configuration Registers

n	Interrupt	Interrupt Select Register	Configuration Register
3	XINT3	GPIOXINT3SEL	XINT3CR
4	XINT4	GPIOXINT4SEL	XINT4CR
5	XINT5	GPIOXINT5SEL	XINT5CR
6	XINT6	GPIOXINT6SEL	XINT6CR
7	XINT7	GPIOXINT7SEL	XINT7CR

Table 121. External Interrupt *n* Control Register (XINT*n*CR) Field Descriptions

Bits	Field	Value	Description
15-4	Reserved		Reads return zero; writes have no effect.
3-2	Polarity		This read/write bit determines whether interrupts are generated on the rising edge or the falling edge of a signal on the pin.
		00	Interrupt generated on a falling edge (high-to-low transition)
		01	Interrupt generated on a rising edge (low-to-high transition)
		10	Interrupt is generated on a falling edge (high-to-low transition)
		11	Interrupt generated on both a falling edge and a rising edge (high-to-low and low-to-high transition)
1	Reserved		Reads return zero; writes have no effect
0	Enable		This read/write bit enables or disables external interrupt XINT*n*.
		0	Disable interrupt
		1	Enable interrupt

• XIntruptRegs.XINT3CR.bit.POLARITY = 0; : 읽기, 쓰기 비트는 핀에 있는 신호의 상승엣지 또는 하강 엣지에서 인터럽트가 생성되는 여부를 결정한다.
- "0"을 입력함으로써, 하강 엣지에서 인터럽트가 생성 된다.(high에서 low로 전환)

• XIntruptRegs.XINT3CR.bit.ENABLE = 1; : 읽기, 쓰기 비트는 외부 인터럽트 XINTn을 활성화하거나, 비활성화 한다.
- "1"을 입력함으로써, interrupt를 활성화한다.

Table 117. PIEIERx Register (x = 1 to 12) Field Descriptions

Bits	Field	Description
15-8	Reserved	Reserved
7	INTx.8	These register bits individually enable an interrupt within a group and behave very much like the core interrupt enable register. Setting a bit to 1 enables the servicing of the respective interrupt. Setting a bit to 0 disables the servicing of the interrupt. x = 1 to 12. INTx means CPU INT1 to INT12
6	INTx.7	
5	INTx.6	
4	INTx.5	
3	INTx.4	
2	INTx.3	
1	INTx.2	
0	INTx.1	

• PieCtrlRegs.PIEIER12.bit.INTx1 = 1; : 레지스터 비트는 개별적으로 그룹 내에서 인터럽트를 활성화하고, 코어 인터럽트 활성화 레지스터와 비슷하게 동작한다. 비트를 1로 설정하면, 각각의 인터럽트를 할 수 있게 한다. 0을 설정하면, 인터럽트의 동작이 비활성화된다. (INTx : CPU INT1 ~ INT12)

```
//---------------------------------------------------------------------------------------
// Step 4. ADC 초기화
//---------------------------------------------------------------------------------------
   InitAdc();

   // ADC 설정
   AdcRegs.ADCTRL3.bit.ADCCLKPS = 3;            // ADCCLK = HSPCLK/(ADCCLKPS*2)/(CPS+1)
   AdcRegs.ADCTRL1.bit.CPS = 1;              // ADCCLK = 75MHz/(3*2)/(1+1) = 6.25MHz
   AdcRegs.ADCTRL1.bit.ACQ_PS = 3;             // 샘플/홀드 사이클 = ACQ_PS + 1 = 4 (ADCCLK기준)
   AdcRegs.ADCTRL1.bit.SEQ_CASC = 1;           // 시퀀스 모드 설정: 직렬 시퀀스 모드 (0:병렬 모드, 1:직렬 모드)
   AdcRegs.ADCMAXCONV.bit.MAX_CONV1 = 1;       // ADC 채널수 설정: 1개(=MAX_CONV+1)채널을 ADC
   AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0;        // ADC 순서 설정: 첫번째로 ADCINA2 채널을 ADC
   AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 8;        // ADC 순서 설정: 2번째로 ADCINB0 채널을 ADC

   AdcRegs.ADCTRL2.bit.EPWM_SOCB_SEQ = 1;        // ePWM_SOCB로 ADC 시퀀스 시동
   AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1 = 1;       // ADC 시퀀스 완료시 인터럽트 발생 설정

   //ePWM_SOCB 이벤트 트리거 설정
   EPwm5Regs.ETSEL.bit.SOCBEN = 1;             // SOCB 이벤트 트리거 Enable
   EPwm5Regs.ETSEL.bit.SOCBSEL = 2;           // SCCB 트리거 조건 : 카운터 주기 일치 시
   EPwm5Regs.ETPS.bit.SOCBPRD = 1;             // SOCB 이벤트 분주 설정 : 트리거 조건 한번 마다
   EPwm5Regs.TBCTL.bit.CTRMODE = 0;           // 카운트 모드 설정: Up-conut 모드
   EPwm5Regs.TBCTL.bit.HSPCLKDIV = 1;          // TBCLK = [SYSCLKOUT / ((HSPCLKDIV*2) * 2^(CLKDIV))]
   EPwm5Regs.TBCTL.bit.CLKDIV = 1;             // TBCLK = [150MHz / (2*2)] = 37.5MHz
   EPwm5Regs.TBPRD = (TBCLK / PwmCarrierFrequency) - 1;                  // TB주기= (TBPRD+1)/TBCLK = 1875/37.5MHz = 50us(20KHz)
   EPwm5Regs.TBCTR = 0x0000;                 // TB 카운터 초기화
```

7	CPS		Core clock prescaler. The prescaler is applied to divided device peripheral clock, HSPCLK.
		0	ADCCLK = F_{clk}/1
		1	ADCCLK = F_{clk}/2
			Note: F_{clk} = Prescaled HSPCLK (ADCCLKPS[3:0])

• AdcRegs.ADCTRL1.bit.CPS = 1; : Core clock prescaler. Prescaler는 분할된 장치 주변기기의 시계인 HSPCLK에 적용된다.

11-8	ACQ_PS[3:0]		Acquisition window size. This bit field controls the width of SOC pulse, which, in turn, determines for what time duration the sampling switch is closed. The width of SOC pulse is ADCTRL1[11:8] + 1 times the ADCLK period.

• AdcRegs.ADCTRL1.bit.ACQ_PS = 3; : Acquisition window size. 비트 필드는 SOC 펄스의 폭을 제어하며, 이는 샘플링 스위치가 닫히는 시간을 결정한다. SOC 펄스폭은 ADCLK 주기의 ADCTRL1[11:8]+1배이다.

4	SEQ_CASC		Cascaded sequencer operation. This bit determines whether SEQ1 and SEQ2 operate as two 8-state sequencers or as a single 16-state sequencer (SEQ).
		0	Dual-sequencer mode. SEQ1 and SEQ2 operate as two 8-state sequencers.
		1	Cascaded mode. SEQ1 and SEQ2 operate as a single 16-state sequencer (SEQ).

• **AdcRegs.ADCTRL1.bit.SEQ_CASC = 1;** : Cascaded sequencer operation. 이 비트는 SEQ1과 SEQ2가 두 개의 8개 상태 시퀀서로 작동하는지 또는 단일 16개 상태 시퀀서(SEQ)로 작동하는지 여부를 결정한다.

Table 2-4. Maximum Conversion Channels Register (ADCMAXCONV) Field Descriptions

Bit(s)	Name	Description
15-7	Reserved	Reads return a zero. Writes have no effect.
6-0	MAX_CONVn	MAX_CONVn bit field defines the maximum number of conversions executed in an autoconversion session. The bit fields and their operation vary according to the sequencer modes (dual/cascaded). For SEQ1 operation, bits MAX_CONV1[2:0] are used. For SEQ2 operation, bits MAX_CONV2[2:0] are used. For SEQ operation, bits MAX_CONV1[3:0] are used. An autoconversion session always starts with the initial state and continues sequentially until the end state if allowed. The result buffer is filled in a sequential order. Any number of conversions between 1 and (MAX_CONVn +1) can be programmed for a session.

• **AdcRegs.ADCMAXCONV.bit.MAX_CONV1 = 1;** : MAX_CONVN 비트 필드는 자동 변환 세션에서 실행된 최대 변환 수를 정의한다. 비트 필드와 그 작동은 시퀀서 모드(dual/cascaded)에 따라 달라진다.

- SEQ 작동에는 비트 MAX_CONV1[3:0]이 사용된다. 자동 변환 세션(autoconversion session)은 항상 초기 상태로 시작하고 허용되는 경우 종료 상태까지 순차적으로 계속된다. 결과 버퍼가 순차적으로 채워진다. 세션에 대해 1과 (MAX_CONVN +1) 사이의 임의 개수의 변환을 프로그래밍할 수 있다.

Bits	Name	Value	Description
15	SOCBEN		Enable the ADC Start of Conversion B (EPWMxSOCB) Pulse
		0	Disable EPWMxSOCB.
		1	Enable EPWMxSOCB pulse.

• **EPwm5Regs.ETSEL.bit.SOCBEN=1;** : 변환 B펄스의 adc 시작을 활성화한다.

- "1"을 입력함으로써, EPWMxSOCB pulse를 활성화한다.

14-12	SOCBSEL		EPWMxSOCB Selection Options These bits determine when a EPWMxSOCB pulse will be generated.
		000	Reserved
		001	Enable event time-base counter equal to zero. (TBCTR = 0x0000)
		010	Enable event time-base counter equal to period (TBCTR = TBPRD)
		011	Reserved
		100	Enable event time-base counter equal to CMPA when the timer is incrementing.
		101	Enable event time-base counter equal to CMPA when the timer is decrementing.
		110	Enable event: time-base counter equal to CMPB when the timer is incrementing.
		111	Enable event: time-base counter equal to CMPB when the timer is decrementing.

13-12	SOCBPRD		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.
		00	Disable the SOCB event counter. No EPWMxSOCB pulse will be generated
		01	Generate the EPWMxSOCB pulse on the first event: ETPS[SOCBCNT] = 0,1
		10	Generate the EPWMxSOCB pulse on the second event: ETPS[SOCBCNT] = 1,0
		11	Generate the EPWMxSOCB pulse on the third event: ETPS[SOCBCNT] = 1,1

• **EPwm5Regs.ETPS.bit.SOCBPRD=1;** : EPWMxSOCB 펄스가 생성되기 전에 몇 개의 선택된 ETSEL 이벤트가 발생해야 하는 지를 결정한다. 생성하려면 펄스를 활성화 해야 한다.(ETFLG[SOCB]=1). 이전 변환 시작(ETFLG[SOCB]=1)에서 상태 플래그를 설정해도 SOCV펄스가 생성된다. SOCB 펄스가 생성되면 ETPS[SOCBCNT]비트가 자동으로 지워진다.

- "1"을 입력함으로써, 1번째 이벤트에서 EPWMxSOCB펄스가 생성된다. ETPS[SOCBCNT]=0, 1.

1:0	CTRMODE		Counter Mode The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change. These bits set the time-base counter mode of operation as follows:
		00	Up-count mode
		01	Down-count mode
		10	Up-down-count mode
		11	Stop-freeze counter operation (default on reset)

• **EPwm5Regs.TBCTL.bit.CTRMODE=0;** : counter mode에 "0"을 입력함으로써, "Up-Count Mode"가 된다.

9:7	HSPCLKDIV		High Speed Time-base Clock Prescale Bits These bits determine part of the time-base clock prescale value. TBCLK = SYSCLKOUT / (HSPCLKDIV × CLKDIV) This divisor emulates the HSPCLK in the TMS320x281x system as used on the Event Manager (EV) peripheral.
		000	/1
		001	/2 (default on reset)
		010	/4
		011	/6
		100	/8
		101	/10
		110	/12
		111	/14

• **EPwm5Regs.TBCTL.bit.HSPCLKDIV=1;** : High Speed Time-base Clock Prescale Bits.

- Time-base clock prescale value의 일부를 결정한다.
- TBCLK = SYSCLKOUT/(HSPCLKDIV*CLKDIV).
- 구분자(divisor)은 이벤트 관리자 주변장치에 사용되는 TMS320x281x 시스템의 HSPCLK를 에뮬레이트 한다.

12:10	CLKDIV		Time-base Clock Prescale Bits These bits determine part of the time-base clock prescale value. TBCLK = SYSCLKOUT / (HSPCLKDIV × CLKDIV)
		000	/1 (default on reset)
		001	/2
		010	/4
		011	/8
		100	/16
		101	/32
		110	/64
		111	/128

• **EPwm5Regs.TBCTL.bit.CLKDIV = 1;** : Time-base Clock Prescale Bits.

- 비트는 타임베이스 클럭 사전 설정 값의 일부를 결정한다. **(TBCLK = SYSCLKOUT / (HSPCLKDIV*CLKDIV))**

=> TBCLK = [SYSCLKOUT / (HSPCLKDIV*CLKDIV)]

$= \frac{150MHz}{2*2} = 37.5MHz$ [SYSTEM_CLOCK : 150E6, HSKPCLKDIV : 1, CLKDIV : 1]

SysCtrlRegs.HISPCP.bit.HSPCLK = 1; // HSPCLK = SYSCLKOUT/(HISPCP*2)

$\frac{150MHz}{2} = 75MHz$

```
/*---------------------------------------------------------
   Step 6
   6.1 Initialize Periphrals for User Application
---------------------------------------------------------*/
/* Initialize EPWM4 Module */
InitEPwm5Module();

/*---------------------------------------------------------
   Step 7
   7.1 Initialize S/W modules and Variables
---------------------------------------------------------*/
BackTicker = 0;
EPwm5IsrTicker = 0;

PwmCarrierFrequency = PWM_CARRIER;

FallingEdgeDelay1 = (1.0 / TBCLK) * EPwm5Regs.DBFED;
RisingEdgeDelay1 = (1.0 / TBCLK) * EPwm5Regs.DBRED;
```

• **InitEPwm5Module();** : 가변저항 변화에 대한 PWM 주파수 가변이 가능하도록 설정하는 ePWM 함수.

• **PwmCarrierFrequency = PWM_CARRIER;** : PwmCarrierFrequency을 20kHz로 설정한다.

- FallingEdgeDelay는 $DBFEB^* T_{BCLK}$, RisingEdgeDelay는 $DBREB^* T_{BCLK}$로 설정한다.

```
//----------------------
ADC_value01 = 0;
ADC_value02 = 0;
Loop_cnt = 0;
//======================
```

ADC_value01, 02를 0으로 초기화 한다.

```
for (;;)
{

    BackTicker++;
    if (mode == 1)
    {

        EPwm5Regs.TBPRD = (TBCLK / PwmCarrierFrequency) - 1;
        PwmDutyRatio = ADC_value01 * 0.475 / 65536;
        EPwm5Regs.CMPA.half.CMPA = (Uint16)((EPwm5Regs.TBPRD + 1) * PwmDutyRatio);   /* Set Compare A Value to 50% */
        EPwm5Regs.CMPB = (EPwm5Regs.TBPRD + 1) * (1 - PwmDutyRatio); //상보적
        /* Setup Counter Mode and Clock */
        EPwm5Regs.TBCTL.bit.CTRMODE = 2;//up-downcount
           /* Set Dead-time */
        EPwm5Regs.DBCTL.bit.IN_MODE = 2;      /* EPWMxA is the source for both falling-edge & rising-edge delay */
        EPwm5Regs.DBCTL.bit.OUT_MODE = 3;      /* Dead-band is fully enabled for both rising-edge delay on EPWMxA and falling-edge delay on EPWMxB */
        EPwm5Regs.DBCTL.bit.POLSEL = 2;         /* Active High Complementary (AHC). EPWMxB is inverted */
        EPwm5Regs.DBFED = 450;               /* 3usec, Falling Edge Delay */
        EPwm5Regs.DBRED = 450;               /* 3usec, Rising Edge Delay */

    }
```

Table 27. Counter-Compare B Register (CMPB) Field Descriptions

Bits	Name	Description
15-0	CMPB	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: • Do nothing. event is ignored. • Clear: Pull the EPWMxA and/or EPWMxB signal low • Set: Pull the EPWMxA and/or EPWMxB signal high • Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed. • 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: • Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full. • 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. • In either mode, the active and shadow registers share the same memory map address.

• **PWM_DUTY_RATIO_A = ADC_value01 * 0.475/65536;** : duty 비를 ADC 변수에 달라지는 변수로 설정한다.
- ADC_Value01 : 최소값인 0으로 설정하고, 3V에서 최댓값인 65536을 가진다. 따라서, 이와 비례하는 duty 비를 설정하였고, 최댓값이 PWM 주기 절반의 95%로 설정하기 위해, 0.475를 최대 duty 비로 설정하였다.

• **EPwm5Regs.CMPA.half.CMPA = (Uint16)((EPwm5Regs.TBPRD + 1) * PwmDutyRatio);**
- 비교 A값을 50%로 설정하기 위해서, PwmDutyRatio를 0.5로 곱한다.

• **EPwm5Regs.CMPB = (EPwm5Regs.TBPRD + 1) = (1 - PwmDutyRatio);** : EPWM5B의 PWM파형은 EPWM5A와 상보적으로 나타나야 하기에, 1에서 CMPA에서 설정한 duty값을 빼준 값이 CMPB로 설정된다.

- 값이 같을 때, Counter-compare module은 "카운트 비교 B와 동일한 타임 베이스 카운터" 이벤트를 생성한다.
- 이 이벤트는 자격이 주어지는 action-qualifier에게 전송되어, 하나 이상의 조치로 변환된다.
- 이런 동작은 AQCTLA 및 AQCTLB 레지스터의 구성에 따라 PWMxA 또는 EPWMxB 출력에 적용할 수 있다.

• **EPwm5Regs.DBFED, EPwm5Regs.DBRED** : 상승, 하강 지연시간을 결정하는 것으로, 450으로 설정함으로써, 3μs를 입력한다.

```
else if (mode == 2)
{

    EPwm5Regs.CMPCTL.bit.SHDWBMODE = 0;      /* Compare B Register is loaded from its shadow when CNTR=Zero */
    EPwm5Regs.CMPCTL.bit.LOADBMODE = 0;
    PWM_CARRIER2 = 100E3 + 200E3*ADC_value01 / 65536;
    EPwm5Regs.TBPRD = (TBCLK / PWM_CARRIER2) - 1;   /* Set Timer Period, (150MHz/100KHz)-1 = 1,499 (0x05DB) */
    EPwm5Regs.CMPA.half.CMPA = (Uint16)((EPwm5Regs.TBPRD + 1)*0.5);   /* Set Compare A Value to 50% */



    EPwm5Regs.TBCTL.bit.CTRMODE = 0;//upcont
    EPwm5Regs.AQCTLA.bit.ZRO = 2;      /* Set EPWM4A on CNTR=Zero */
     /* Set Dead-time */
    EPwm5Regs.DBCTL.bit.IN_MODE = 0;      /* EPWMxA is the source for both falling-edge & rising-edge delay */
    EPwm5Regs.DBCTL.bit.OUT_MODE = 3;      /* Dead-band is fully enabled for both rising-edge delay on EPWMxA and falling-edge delay on EPWMxB */
    EPwm5Regs.DBCTL.bit.POLSEL = 2;          /* Active High Complementary (AHC). EPWMxB is inverted */
    EPwm5Regs.DBFED = 75;               /* 2usec, Falling Edge Delay */
    EPwm5Regs.DBRED = 75;               /* 1usec, Rising Edge Delay */

}
```

6	SHDWBMODE		Counter-compare B (CMPB) Register Operating Mode
		0	Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register.
		1	Immediate mode. Only the active compare B register is used. All writes and reads directly access the active register for immediate compare action.

• **EPwm5Regs.CMPCTL.bit.SHDWBMODE = 0;** : Counter-compare B(CMPB) Register Operating Mode.
- "0"을 입력함으로써, Shadow Mode. 이중 버퍼로써 작동한다. 모든 write는 CPU를 통해, Shadow register에 접속한다.

1:0	CTRMODE		Counter Mode
			The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change.
			These bits set the time-base counter mode of operation as follows:
		00	Up-count mode
		01	Down-count mode
		10	Up-down-count mode
		11	Stop-freeze counter operation (default on reset)

• **EPwm5Regs.TBCTL.bit.CTRMODE = 0;** : 일반적으로 한 번 구성되며 정상 작동 중에는 변경되지 않는다. 카운터의 모드를 변경하면, 다음 TBCLK Edge에서 변경사항이 적용되며, 현재 카운터 값은 모드 변경 전 값에서 증가 또는 감소한다.
- "0"을 입력함으로써, "Up-count mode".

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.
		00	Do nothing (action disabled)
		01	Clear: force EPWMxA output low.
		10	Set: force EPWMxA output high.
		11	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.

• **EPwm5Regs.AQCTLA.bit.ZRO =2;** : 카운터가 0일 때 동작.
- "1"을 입력함으로써, "Set : Force EPWMxA output high."

Table 26. Counter-Compare A Register (CMPA) Field Descriptions

Bits	Name	Description
15-0	CMPA	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: • Do nothing; the event is ignored. • Clear: Pull the EPWMxA and/or EPWMxB signal low • Set: Pull the EPWMxA and/or EPWMxB signal high • Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWAMODE] bit. By default this register is shadowed. • 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. • Before a write, the CMPCTL[SHDWAFULL] bit can be read to determine if the shadow register is currently full. • 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. • In either mode, the active and shadow registers share the same memory map address.

• **EPwm5Regs.CMPA.half.CMPA = (Uint16)((EPwm5Regs.TBPRD + 1) *0.5);** : 비교 A 값을 50%로 설정.
- 활성 CMPA 레지스터의 값은 TBCTR와 지속적으로 비교된다. 값이 같을 때, counter-compare module은 "카운트 비교 A와 동일한 타임 베이스 카운터" 이벤트를 생성한다.
- 이런 이벤트는 자격이 주어지는 조치 인증자에게 전송되어 하나 이상의 조치로 변환되고, AQCTLA 및 AQCTLB 레지스터의 구성에 따라 EPWMxA 또는 EPWMxB 출력에 적용할 수 있다.

- 만약 CMPCTL[SHDWAMODE]=0일 때, shadow는 활성화되고 모든 쓰기 또는 읽기가 자동으로 shadow register로 이동한다.
- 이 경우 CMPCTL[LOADAMODE] 비트 필드는 shadow 레지스터에서 활성 레지스터를 로드 할 이벤트를 결정한다.

```
else if (mode == 3)
{
    EPwm5Regs.TBPRD = (TBCLK / PwmCarrierFrequency) - 1;
    EPwm5Regs.CMPA.half.CMPA = (Uint16)((EPwm5Regs.TBPRD + 1) * 0.5);
    deadt = (float32)75 * ADC_value01 / 65536;
    EPwm5Regs.TBCTL.bit.CTRMODE = 0;//upcont
    EPwm5Regs.AQCTLA.bit.ZRO = 2;      /* Set EPWM4A on CNTR=Zero */
       /* Set Dead-time */
    EPwm5Regs.DBCTL.bit.IN_MODE = 0;      /* EPWMxA is the source for both falling-edge & rising-edge delay */
    EPwm5Regs.DBCTL.bit.OUT_MODE = 3;      /* Dead-band is fully enabled for both rising-edge delay on EPWMxA and falling-edge delay on EPWMxB */
    EPwm5Regs.DBCTL.bit.POLSEL = 2;
    EPwm5Regs.DBFED = deadt;               /* 2usec, Falling Edge Delay */
    EPwm5Regs.DBRED = 0;               /* 1usec, Rising Edge Delay */
```

<DBCTL : Dead-Band Control Register>

Bits	Name	Value	Description
15-6	Reserved		Reserved
5-4	IN_MODE		Dead Band Input Mode Control Bit 5 controls the S5 switch and bit 4 controls the S4 switch shown in Figure 31. 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.
		00	EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay.
		01	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.
		10	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.
		11	EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal.

• EPwm5Regs.DBCTL.bit.IN_MODE = 0; : Dead Band Input Mode Control

- 하강, 상승 엣지에 대한 입력 소스를 선택할 수 있다.
- "0"을 입력함으로써, EPWMxA In(from the action – qualifier)은 하강 엣지와 라이징 엣지 지연 Source이다.

1-0	OUT_MODE		Dead-band Output Mode Control Bit 1 controls the S1 switch and bit 0 controls the S0 switch shown in Figure 31. This allows you to selectively enable or bypass the dead-band generation for the falling-edge and rising-edge delay.
		00	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. In this mode, the POLSEL and IN_MODE bits have no effect.
		01	Disable rising-edge delay. The EPWMxA signal from the action-qualifier is passed straight through to the EPWMxA input of the PWM-chopper submodule. The falling-edge delayed signal is seen on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].
		10	The rising-edge delayed signal is seen on output EPWMxA. The input signal for the delay is determined by DBCTL[IN_MODE]. Disable falling-edge delay. The EPWMxB signal from the action-qualifier is passed straight through to the EPWMxB input of the PWM-chopper submodule.
		11	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].

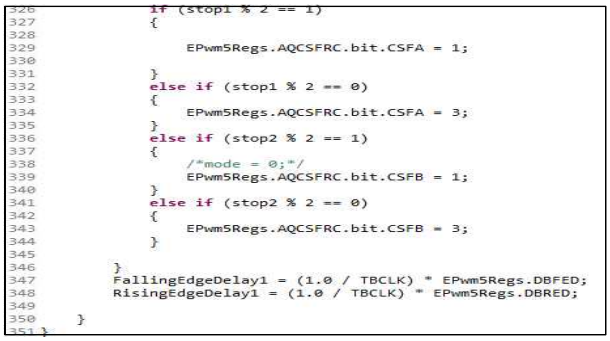
• EPwm5Regs.DBCTL.bit.IN_MODE = 0; : Dead Band Input Mode Control

- 하강 에지 및 상승 에지 지연에 대해 Dead-band 생성을 선택적으로 활성화하거나 우회할 수 있다.
- "3"을 입력함으로써, Dead-band는 출력 EPWMxA의 상승 에지 지연과 출력 EPWMxB의 하강 에지 지연 모두에 대해 완전히 활성화된다. 지연에 대한 입력 신호는 DBCTL[IN_MODE]에 의해 결정된다.

3-2	POLSEL		Polarity Select Control Bit 3 controls the S3 switch and bit 2 controls the S2 switch shown in Figure 31. This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule. The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter. 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.
		00	Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default).
		01	Active low complementary (ALC) mode. EPWMxA is inverted.
		10	Active high complementary (AHC). EPWMxB is inverted.
		11	Active low (AL) mode. Both EPWMxA and EPWMxB are inverted.

• EPwm5Regs.DBCTL.bit.POLSEL = 2; : Polarity Select Control

- 지연된 신호 중 하나가 dead-band 하위 모듈에서 전송되기 전에 선택적으로 반전시킬 수 있다.
- "2"가 입력함으로써, high complementary(AHC)가 활성화되고, EPWMxB는 반전된다.



```
            if (stop1 % 2 == 1)
            {

                EPwm5Regs.AQCSFRC.bit.CSFA = 1;

            }
            else if (stop1 % 2 == 0)
            {
                EPwm5Regs.AQCSFRC.bit.CSFA = 3;
            }
            else if (stop2 % 2 == 1)
            {
                /*mode = 0;*/
                EPwm5Regs.AQCSFRC.bit.CSFB = 1;
            }
            else if (stop2 % 2 == 0)
            {
                EPwm5Regs.AQCSFRC.bit.CSFB = 3;
            }

        }
        FallingEdgeDelay1 = (1.0 / TBCLK) * EPwm5Regs.DBFED;
        RisingEdgeDelay1 = (1.0 / TBCLK) * EPwm5Regs.DBRED;

    }
}
```

1-0	CSFA		Continuous Software Force on Output A In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register.
		00	Forcing disabled, i.e., has no effect
		01	Forces a continuous low on output A
		10	Forces a continuous high on output A
		11	Software forcing is disabled and has no effect

• **CSFA** : Continuous Software Force on Output A.

- "1"을 입력함으로써, "Forces a continuous low on output A.(출력 A에서 연속 low로 설정한다.)"
- "3"을 입력함으로써, 소프트웨어 강제력이 비활성화되어 어떤 효과도 없다.

3-2	CSFB		Continuous Software Force on Output B In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. To configure shadow mode, use AQSFRC[RLDCSF].
		00	Forcing disabled, i.e., has no effect
		01	Forces a continuous low on output B
		10	Forces a continuous high on output B
		11	Software forcing is disabled and has no effect

• **CSFB** : Continuous Software Force on Output B.

- 즉시 모드(Immediae mode)에서는 다음 TBCLK 가장자리에 연속적인 힘이 작용한다.
- Shadow 모드에서, 활성 레지스터에 Shadow가 로드된 후 다음 TBCLK 에지에서 연속적인 힘이 작용한다.
- Shadow 모드를 구성하려면 AQSFRC[RLDCSF]를 사용한다.

```
353 /*-------------------------------------------------------------------------------
354    Step 10
355    10.1 Local Interrupt Service Routines & Functions
356 -------------------------------------------------------------------------------*/
357 interrupt void EPwm5Isr(void)
358 {
359     EPwm5IsrTicker++;
360
361     /* Clear INT flag for this timer */
362     EPwm5Regs.ETCLR.bit.INT = 1;
363
364     /* Acknowledge this interrupt to receive more interrupts from group 3 */
365     PieCtrlRegs.PIEACK.bit.ACK3 = 1;
```

Figure 89. Event-Trigger Clear Register (ETCLR)

15					8
Reserved					
R = 0					
7	**4**	**3**	**2**	**1**	**0**
Reserved		SOCB	SOCA	Reserved	INT
R-0		R/W-0	R/W-0	R-0	R/W-0

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

Table 47. Event-Trigger Clear Register (ETCLR) Field Descriptions

Bits	Name	Value	Description
15-4	Reserved		Reserved
3	SOCB		ePWM ADC Start-of-Conversion B (EPWMxSOCB) Flag Clear Bit
		0	Writing a 0 has no effect. Always reads back a 0
		1	Clears the ETFLG[SOCB] flag bit
2	SOCA		ePWM ADC Start-of-Conversion A (EPWMxSOCA) Flag Clear Bit
		0	Writing a 0 has no effect. Always reads back a 0
		1	Clears the ETFLG[SOCA] flag bit
1	Reserved		Reserved
0	INT		ePWM Interrupt (EPWMx_INT) Flag Clear Bit
		0	Writing a 0 has no effect. Always reads back a 0
		1	Clears the ETFLG[INT] flag bit and enable further interrupts pulses to be generated

• **EPwm5Regs.ETCLR.bit.INT = 1;** : ePWM interrupt (EPWMx_INT) Flag Clear Bit.

- "1"을 입력함으로써, ETFLG[INT] 플래그 비트를 지우고, 추가 인터럽트 펄스가 생성한다.

Table 115. PIE Interrupt Acknowledge Register (PIEACK) Field Descriptions

Bits	Field	Value	Description
15-12	Reserved		Reserved
11-0	PIEACK		Each bit in PIEACK refers to a specific PIE group. Bit 0 refers to interrupts in PIE group 1 that are MUXed into INT1 up to Bit 11, which refers to PIE group 12 which is MUXed into CPU INT12
		bit x = 0 (1)	If a bit reads as a 0, it indicates that the PIE can send an interrupt from the respective group to the CPU.
			Writes of 0 are ignored.
		bit x = 1	Reading a 1 indicates if an interrupt from the respective group has been sent to the CPU and all other interrupts from the group are currently blocked.
			Writing a 1 to the respective interrupt bit clears the bit and enables the PIE block to drive a pulse into the CPU interrupt input if an interrupt is pending for that group.

(1) bit x = PIEACK bit 0 - PIEACK bit 11. Bit 0 refers to CPU INT1 up to Bit 11, which refers to CPU INT12

• **PieCtrlRegs.PIEACK.bit.ACK3 =1;** : ePWM interrupt (EPWMx_INT) Flag Clear Bit.

- PIEACK의 각 비트는 특정 PIE 그룹을 가리킨다. 비트 0은 INT1에서 비트 11까지 MUX로 MUX된 PIE 그룹 1의 인터럽트를 말하며, CPU INT12로 MUX된 PIE 그룹 12를 가리킨다.

-

- "1"의 값을 읽는 것은 해당 그룹의 인터럽트가 CPU로 전송되고 그룹의 다른 모든 인터럽트의 차단 여부를 나타낸다.
- "1"의 값을 쓰는 것은 각 인터럽트 비트에 1을 기록하면서, 비트가 지워지고 해당 그룹에 대한 인터럽트가 보류 중인 경우, PIE 블록이 CPU 인터럽트 입력으로 펄스를 구동할 수 있다는 것을 나타낸다.

```
370 void InitEPwm5Module(void)
371 {
372
373     EPwm5Regs.TBCTL.bit.HSPCLKDIV = 0;       /* TBCLK = SYSCLKOUT / (HSPCLKDIV * CLKDIV) = 150MHz */
374     EPwm5Regs.TBCTL.bit.CLKDIV = 0;
375
376     /* Setup Phase */
377     EPwm5Regs.TBPHS.half.TBPHS = 0;          /* Phase is 0 */
378     EPwm5Regs.TBCTL.bit.PHSEN = 0;          /* Disable phase loading */
379
380     /* Setup Period (Carrier Frequency) */
381     EPwm5Regs.TBPRD = (TBCLK / PWM_CARRIER) - 1;   /* Set Timer Period, (150MHz/100KHz)-1 = 1,499 (0x05DB) */
382     EPwm5Regs.TBCTR = 0;                    /* Clear Counter */
383
384
385     /* Setup shadowing */
386     EPwm5Regs.TBCTL.bit.PRDLD = 0;          /* Period Register is loaded from its shadow when CNTR=Zero */
387     EPwm5Regs.CMPCTL.bit.SHDWAMODE = 0;       /* Compare A Register is loaded from its shadow when CNTR=Zero */
388     EPwm5Regs.CMPCTL.bit.LOADAMODE = 0;
389
390     /* St actions */
391     EPwm5Regs.AQCTLA.bit.CAD = 2;       /* Set EPWM4A on CNTR=Zero */
392     EPwm5Regs.AQCTLA.bit.CAU = 1;       /* Clear EPWMBA on CNTR=CMPA, Up-Count */
393     EPwm5Regs.AQCTLB.bit.CBU = 2;       /* Set EPWM4A on CNTR=Zero */
394     EPwm5Regs.AQCTLB.bit.CBD = 1;       /* Clear EPWMBA on CNTR=CMPA, Up-Count */
395
396     /* Set Interrupts */
397     EPwm5Regs.ETSEL.bit.INTSEL = 1;       /* Select INT on CNTR=Zero */
398     EPwm5Regs.ETPS.bit.INTPRD = 1;       /* Generate INT on 1st event */
399     EPwm5Regs.ETSEL.bit.INTEN = 1;       /* Enable INT */
400 }
```

Table 22. Time-Base Phase Register (TBPHS) Field Descriptions

Bits	Name	Value	Description
15-0	TBPHS	0000-FFFF	These bits set time-base counter phase of the selected ePWM relative to the time-base that is supplying the synchronization input signal. • If TBCTL[PHSEN] = 0, then the synchronization event is ignored and the time-base counter is not loaded with the phase. • If TBCTL[PHSEN] = 1, then the time-base counter (TBCTR) will be loaded with the phase (TBPHS) when a synchronization event occurs. The synchronization event can be initiated by the input synchronization signal (EPWMxSYNCI) or by a software forced synchronization.

• **EPwm5Regs.TBPSH.Half.TBPSH = 0;** : 동기화 입력 신호를 공급하는 타임 베이스에 상대적인 선택된 ePWM의 타임베이스 카운터 위상을 설정한다.

- **TBCTL[PHSEN]**이 0이면 동기화 이벤트가 무시되고 타임베이스 카운터가 위상과 함께 로드되지 않는다.
- "0"으로 입력하였기에, 위상지연이 없다.

PHSEN		Counter Register Load From Phase Register Enable
	0	Do not load the time-base counter (TBCTR) from the time-base phase register (TBPHS)
	1	Load the time-base counter with the phase register when an EPWMxSYNCI input signal occurs or when a software synchronization is forced by the SWFSYNC bit

• **EPwm5Regs.TBCTL.bit.PHSEN = 0;** : counter Register Load From Phase Register Enable.

- "0"을 입력하였기에, time-base phase register(TBPHS)에서 time-base counter(TBCTR)를 로드하지 않는다.

```
404 interrupt void adc_isr(void)
405 {
406     ADC_value01 = AdcRegs.ADCRESULT0; //가변저항 a
407     ADC_value02 = AdcRegs.ADCRESULT1; //가변저항 b
408
409
410     AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;          // Reset SEQ1
411     AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1;        // Clear INT SEQ1 bit
412     PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;    // Acknowledge interrupt to PIE
413 }
414
```

Figure 2-13. ADC Conversion Result Buffer Registers (ADCRESULTn) - (Addresses 0x7108-0x7117)

15	14	13	12	11	10	9	8
D11	D10	D9	D8	D7	D6	D5	D4
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
7	6	5	4	3			0
D3	D2	D1	D0	Reserved			
R-0	R-0	R-0	R-0	R-0			

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

- 계단식 시퀀서 모드에서 ADCRESULT8 ~ 15를 통해 ADCRESULT8을 등록하면 9번째에서 16번째 변환 결과가 유지된다. ADCRESULTn 레지스터는 대기 상태가 2개인 주변 프레임 2(0x7108-0x7117)에서 읽을 때 왼쪽 정당화되고 대기 상태가 0인 주변 프레임 0(0x0B00-0x0F)에서 읽을 때 오른쪽 정당화된다.

- ADC 변환 된 값이 저장되는 ADCRESULT0, 1의 값을 전역변수 ADC_value01, 02에 저장하였다.

Table 2-2. ADC Control Register 2 (ADCTRL2) Field Descriptions

14	RST_SEQ1		Reset sequencer1 Writing a 1 to this bit resets SEQ1 or the cascaded sequencer immediately to an initial "pretriggered" state, i.e., waiting for a trigger at CONV00. A currently active conversion sequence will be aborted.
		0	No action
		1	Immediately reset sequencer to state CONV00

• AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1; : 이 비트에 1을 쓰는 리셋 시퀀서 1은 SEQ1 계단식 시퀀서가 즉시 초기 "예: CONB00에서 트리거를 기다리는" 상태로 재설정된다. 현재 활성 변환 시퀀스가 중단된다.
- "1"을 입력함으로써, 시퀀서를 즉시 상태 CONB00으로 재설정한다.

4	INT_SEQ1_CLR		Interrupt clear bit. Read of this bit always returns 0. The clear action is a one-shot event following a write of 1 to this bit.
		0	Writing a zero to this bit has no effect.
		1	Writing a 1 to this bit clears the SEQ1 interrupt flag bit, INT_SEQ1. This bit does not affect the EOS_BUF1 bit.

• AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1; : 클리어 비트를 중단한다. 이 비트를 읽으면 항상 0이 돌아온다. clear action은 1에서 이 비트까지의 쓰기에 이은 one-shot event이다.

Table 30. Action-Qualifier Output A Control Register (AQCTLA) Field Descriptions (continued)

Bits	Name	Value	Description
5-4	CAU		Action when the counter equals the active CMPA register and the counter is incrementing.
		00	Do nothing (action disabled)
		01	Clear: force EPWMxA output low.
		10	Set: force EPWMxA output high.
		11	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.
		00	Do nothing (action disabled)
		01	Clear: force EPWMxA output low.
		10	Set: force EPWMxA output high.
		11	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.
		00	Do nothing (action disabled)
		01	Clear: force EPWMxA output low.
		10	Set: force EPWMxA output high.
		11	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.

Bits	Name	Value	Description
15-12	Reserved		Reserved
11-10	CBD		Action when the time-base counter equals the active CMPB register and the counter is decrementing.
		00	Do nothing (action disabled)
		01	Clear: force EPWMxA output low.
		10	Set: force EPWMxA output high.
		11	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
9-8	CBU		Action when the counter equals the active CMPB register and the counter is incrementing.
		00	Do nothing (action disabled)
		01	Clear: force EPWMxA output low.
		10	Set: force EPWMxA output high.
		11	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.
7-6	CAD		Action when the counter equals the active CMPA register and the counter is decrementing.
		00	Do nothing (action disabled)
		01	Clear: force EPWMxA output low.
		10	Set: force EPWMxA output high.
		11	Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low.

• EPwm5Regs.AQCTLA.bit.CAD = 2; EPwm5Regs.AQCTLA.bit.CAU = 1;
EPwm5Regs.AQCTLB.bit.CBU = 2; EPwm5Regs.AQCTLB.bit.CBD = 1;

- "CAD : 2" : 카운터가 활성 CMPA 레지스터와 동일하고, 카운터가 감소할 때, 동작.
"2"를 입력함으로써, "Set : Force EPWMxA output high."

- "CAU : 1" : 카운터가 활성 CMPA 레지스터와 동일하고, 카운터가 증분 중일 때, 동작.
"1"를 입력함으로써, "Clear : Force EPWMxA output low."

- "CBU : 2" : 카운터가 활성 CMPB 레지스터와 동일하고, 카운터가 증분 중일 때, 동작.
"2"를 입력함으로써, "Set : force EPWMxA output high."

- "CBD : 1" : time-base counter가 활성 CMPB 레지스터와 동일하고 카운터가 감소할 때, 동작.
"1"를 입력함으로써, "Clear : force EPWMxA output low."