F28P55x如何实现EPWM配置

- Code Composer Studio
- C2000Ware
- LaunchXL-F28P55x

编程实现EPWM

功能需求

课程目的

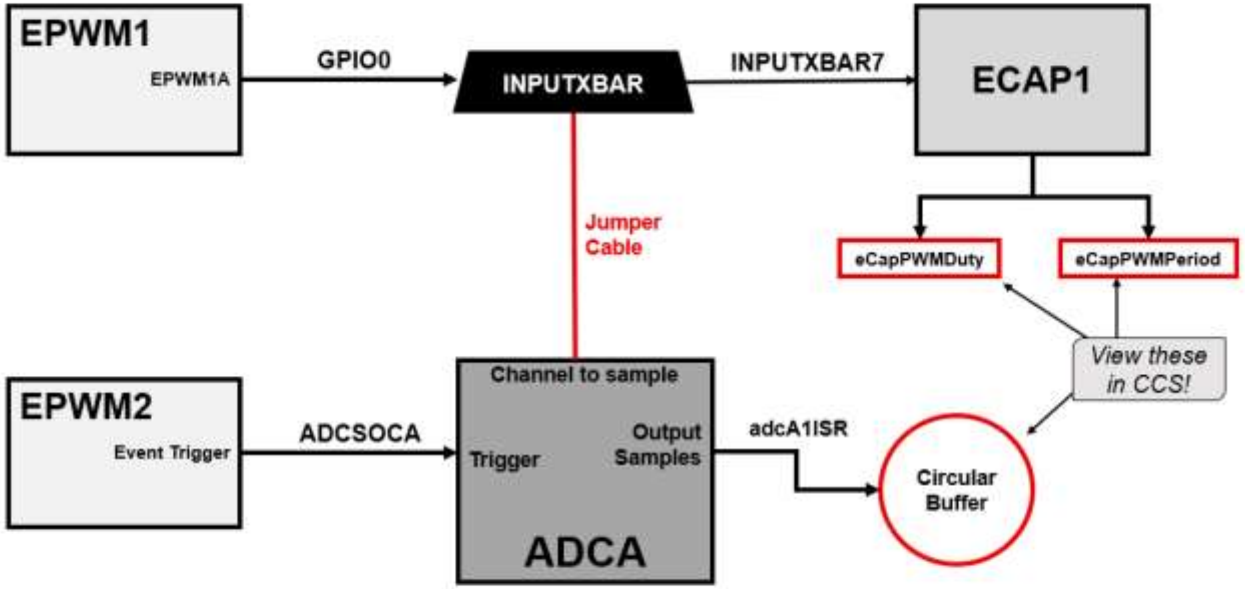
本课程的目的是学习EPWM、ECAP、ADC的配置

功能需求

EPWM1产生可变占空比的方波信号，ADC采集该PWM信号，ECAP采集EPWM1的的占空比，ADC的SOC由EPWM2触发。通过CCS的观察窗口Debug数据变化。

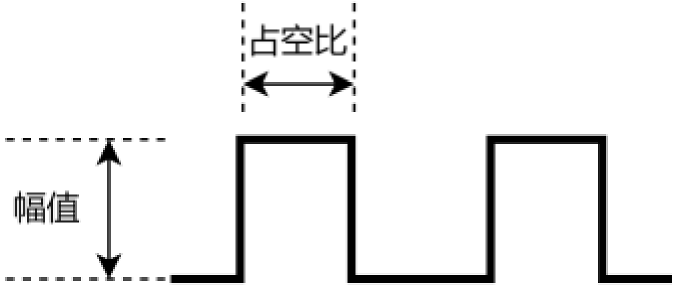
ADCINA0-A0/B15-Head 70

EPWM1-EPWMA-Head 40



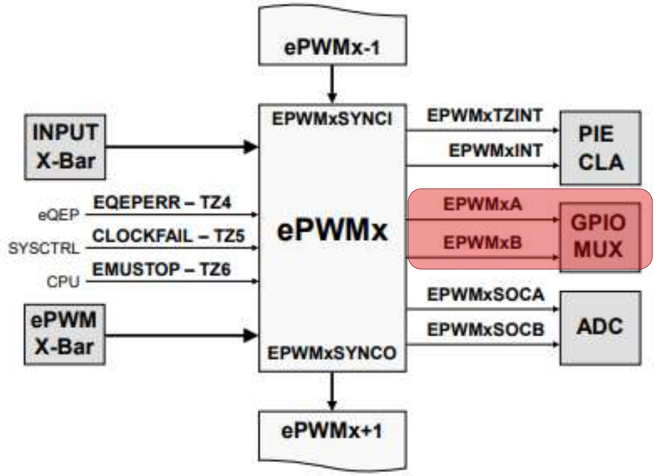
EPWM

Enhanced Pulse Width Modulation, 增强型脉冲宽度调制



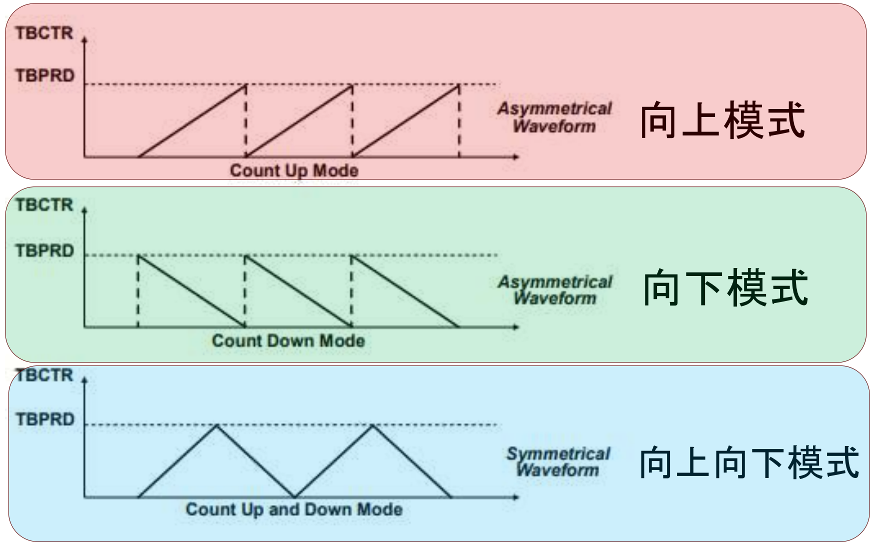
- DC/DC电源转换
- BLDC电机驱动
- 变频控制

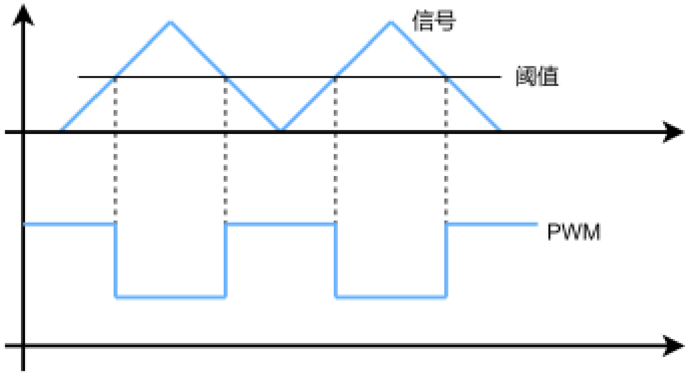
特征：1)生成复杂波形 2)生成死区3) 灵活同步



通道 A 上的 HR 占空比和死区控制
通道 B 上的 HR 占空比和死区控制

时基-Time Base　　比较器-Compare





向上向下模式

编程实现EPWM

功能需求

课程目的

本课程的目的是学习EPWM、ECAP、ADC的配置

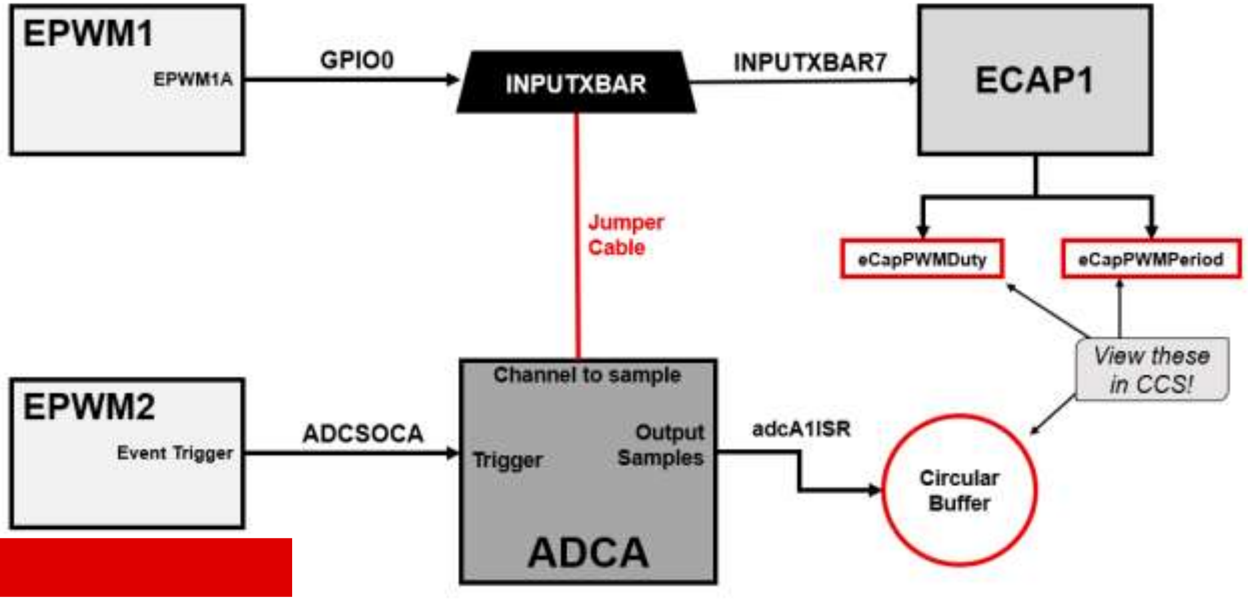
功能需求

EPWM1产生可变占空比的方波信号，ADC采集该PWM信号，ECAP采集EPWM1的的占空比， ADC的SOC由EPWM2触发 。通过CCS的观察窗口Debug数据变化 。

ADCINA0-A0/B15-Head 70

EPWM1-EPWMA-Head 40

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比



实验步骤

1. **复制空工程**

   路径: …C2000Ware_5_02_00_00\training\device\f28p55x\empty_lab

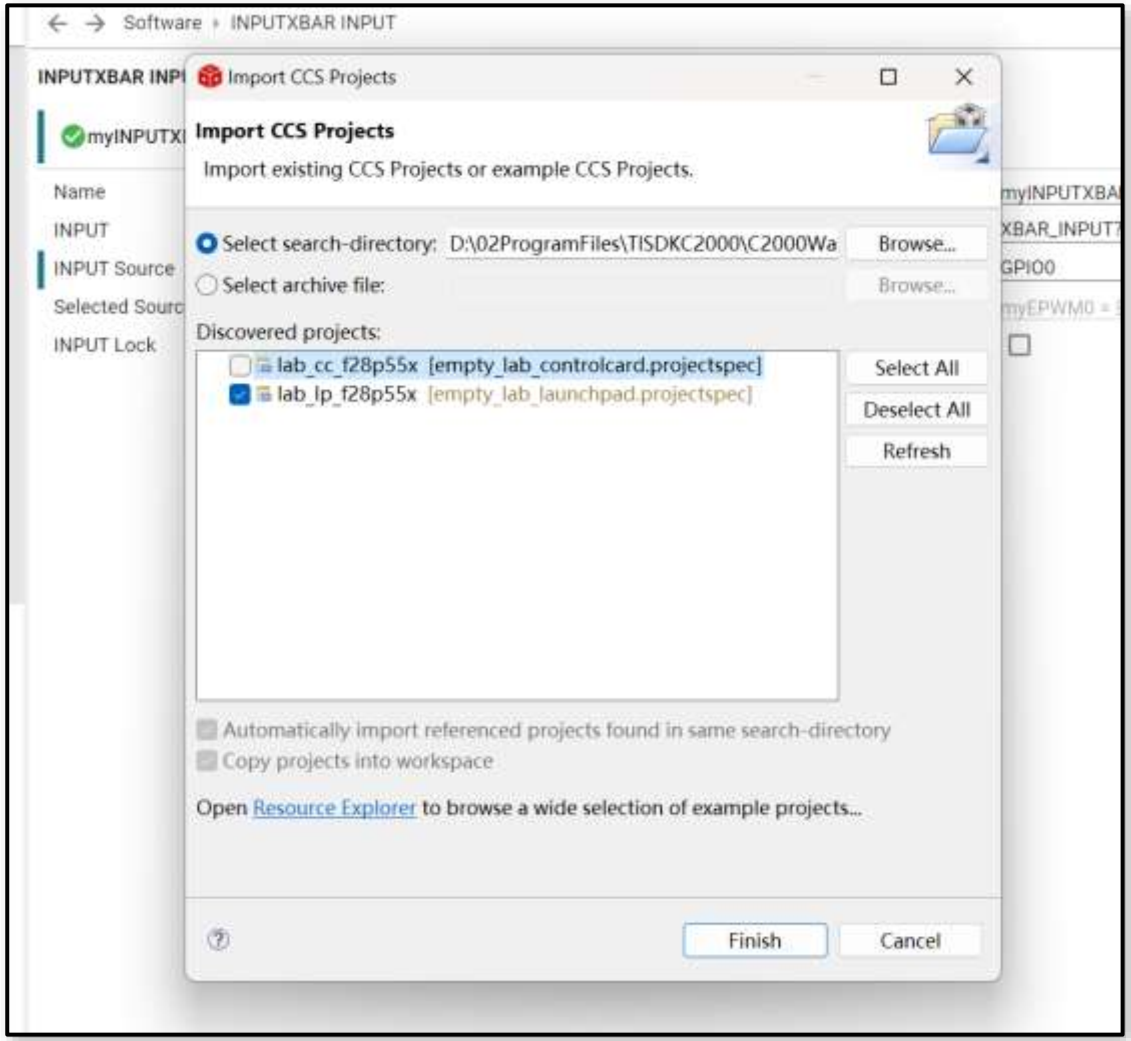
2. **配置LED5**

3. **配置EPWM1**

4. **配置EPWM2**

5. **配置ADC**

6. **配置ECAP**



Import CCS Projects

Import existing CCS Projects or example CCS Projects.

● Select search-directory: D:\02ProgramFiles\TISDKC2000\C2000Wa [Browse...]

○ Select archive file: [Browse...]

Discovered projects:

- □ lab_cc_f28p55x [empty_lab_controlcard.projectspec]
- ☑ lab_lp_f28p55x [empty_lab_launchpad.projectspec]

[Select All] [Deselect All] [Refresh]

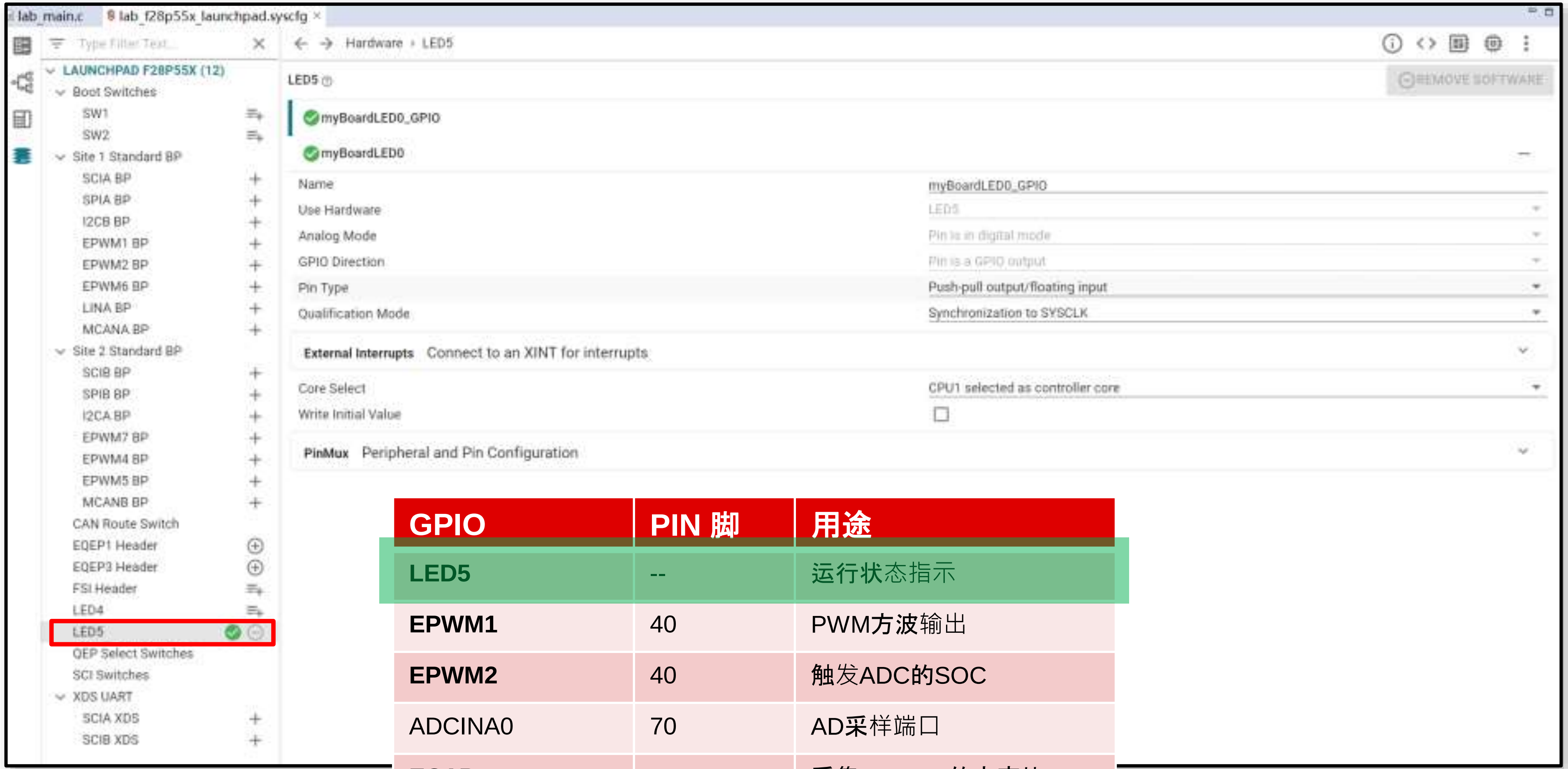
□ Automatically import referenced projects found in same search-directory

□ Copy projects into workspace

Open Resource Explorer to browse a wide selection of example projects...

[Finish] [Cancel]

配置LED5



lab_main.c | lab_f28p55x_launchpad.syscfg ×

Type Filter Text...

- LAUNCHPAD F28P55X (12)
  - Boot Switches
    - SW1
    - SW2
  - Site 1 Standard BP
    - SCIA BP
    - SPIA BP
    - I2CB BP
    - EPWM1 BP
    - EPWM2 BP
    - EPWM6 BP
    - LINA BP
    - MCANA BP
  - Site 2 Standard BP
    - SCIB BP
    - SPIB BP
    - I2CA BP
    - EPWM7 BP
    - EPWM4 BP
    - EPWM5 BP
    - MCANB BP
  - CAN Route Switch
  - EQEP1 Header
  - EQEP3 Header
  - FSI Header
  - LED4
  - LED5 ✓
  - QEP Select Switches
  - SCI Switches
  - XDS UART
    - SCIA XDS
    - SCIB XDS

Hardware › LED5

LED5 — REMOVE SOFTWARE

✓ myBoardLED0_GPIO

✓ myBoardLED0

Name	myBoardLED0_GPIO
Use Hardware	LED5
Analog Mode	Pin is in digital mode
GPIO Direction	Pin is a GPIO output
Pin Type	Push-pull output/floating input
Qualification Mode	Synchronization to SYSCLK

External Interrupts Connect to an XINT for interrupts

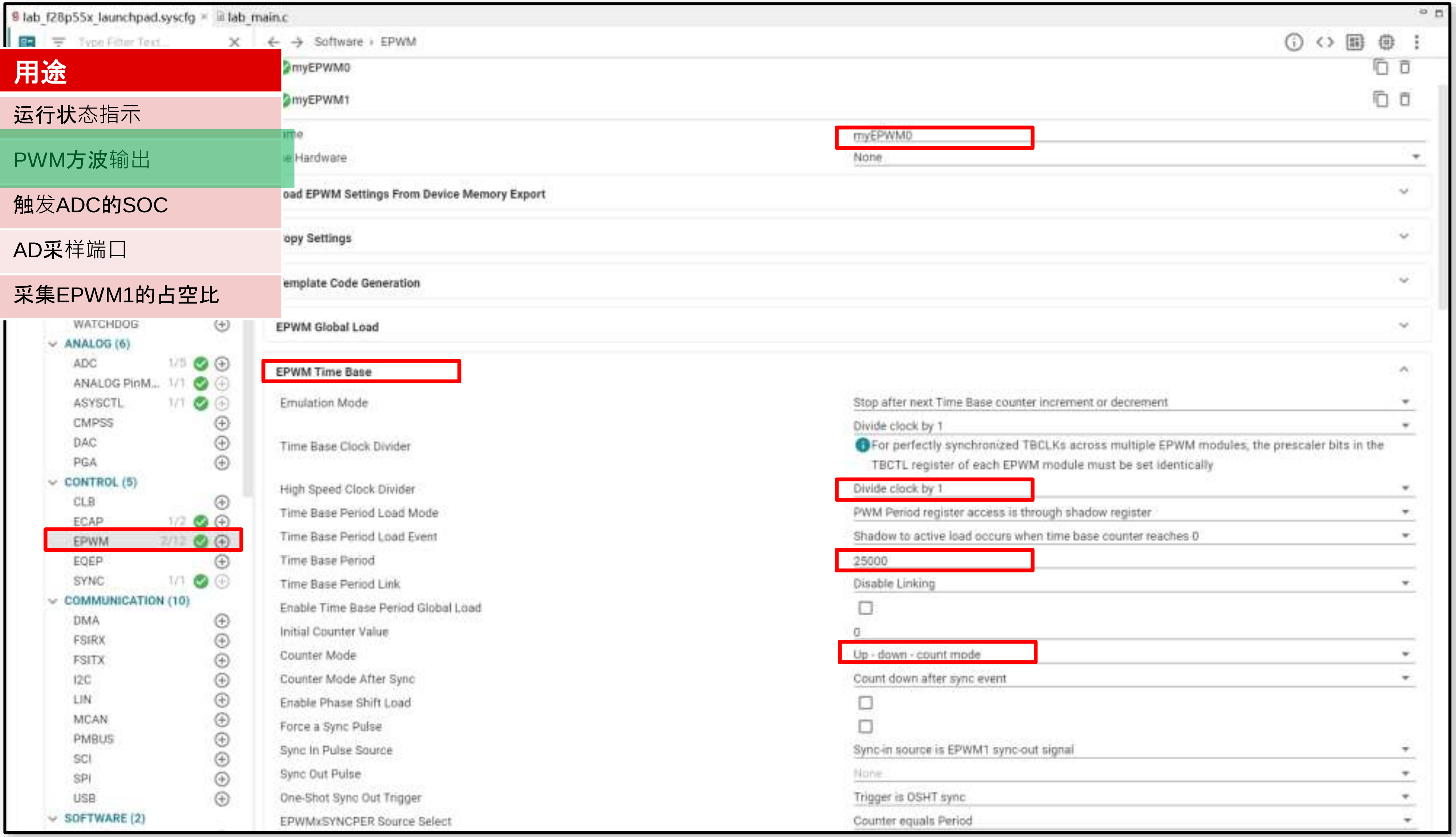
Core Select	CPU1 selected as controller core
Write Initial Value	□

PinMux Peripheral and Pin Configuration

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

配置PWM1

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比



myEPWM0

myEPWM1

me: myEPWM0

e Hardware: None

oad EPWM Settings From Device Memory Export

opy Settings

emplate Code Generation

EPWM Global Load

EPWM Time Base

Emulation Mode: Stop after next Time Base counter increment or decrement

Time Base Clock Divider: Divide clock by 1

For perfectly synchronized TBCLKs across multiple EPWM modules, the prescaler bits in the TBCTL register of each EPWM module must be set identically

High Speed Clock Divider: Divide clock by 1

Time Base Period Load Mode: PWM Period register access is through shadow register

Time Base Period Load Event: Shadow to active load occurs when time base counter reaches 0

Time Base Period: 25000

Time Base Period Link: Disable Linking

Enable Time Base Period Global Load: □

Initial Counter Value: 0

Counter Mode: Up - down - count mode

Counter Mode After Sync: Count down after sync event

Enable Phase Shift Load: □

Force a Sync Pulse: □

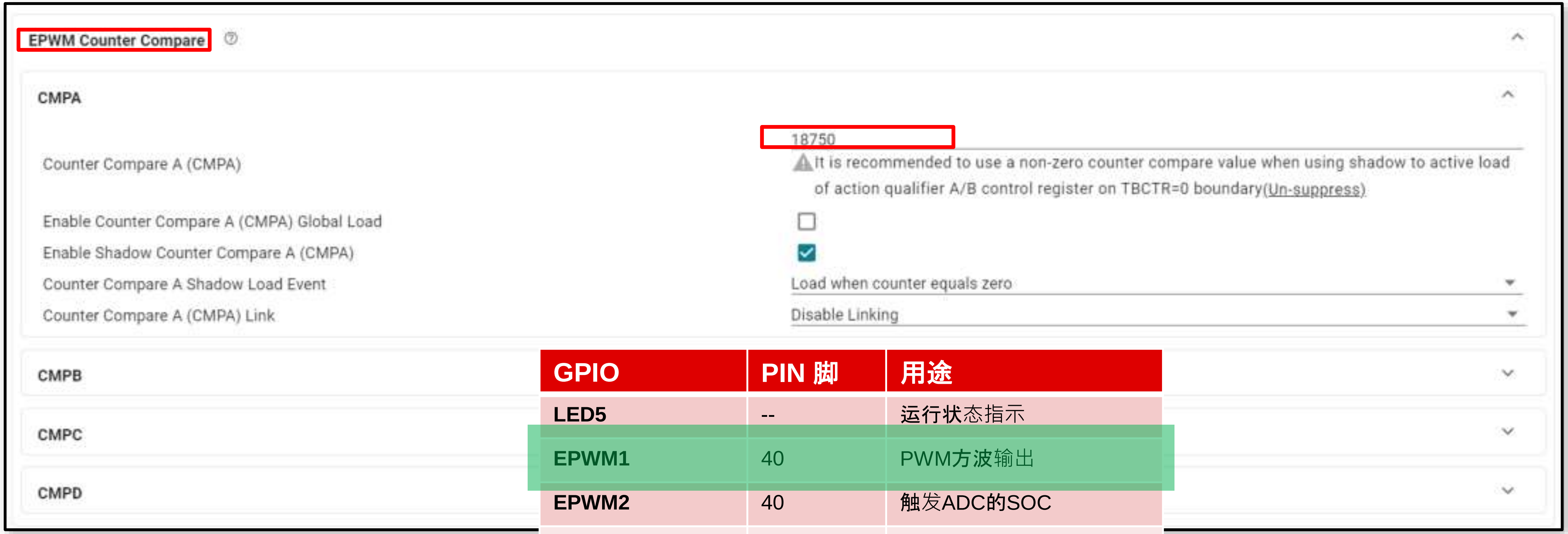
Sync In Pulse Source: Sync-in source is EPWM1 sync-out signal

Sync Out Pulse: None

One-Shot Sync Out Trigger: Trigger is OSHT sync

EPWMxSYNCPER Source Select: Counter equals Period

配置PWM1



EPWM Counter Compare

CMPA

Counter Compare A (CMPA)	18750 It is recommended to use a non-zero counter compare value when using shadow to active load of action qualifier A/B control register on TBCTR=0 boundary(Un-suppress)
Enable Counter Compare A (CMPA) Global Load	□
Enable Shadow Counter Compare A (CMPA)	☑
Counter Compare A Shadow Load Event	Load when counter equals zero
Counter Compare A (CMPA) Link	Disable Linking

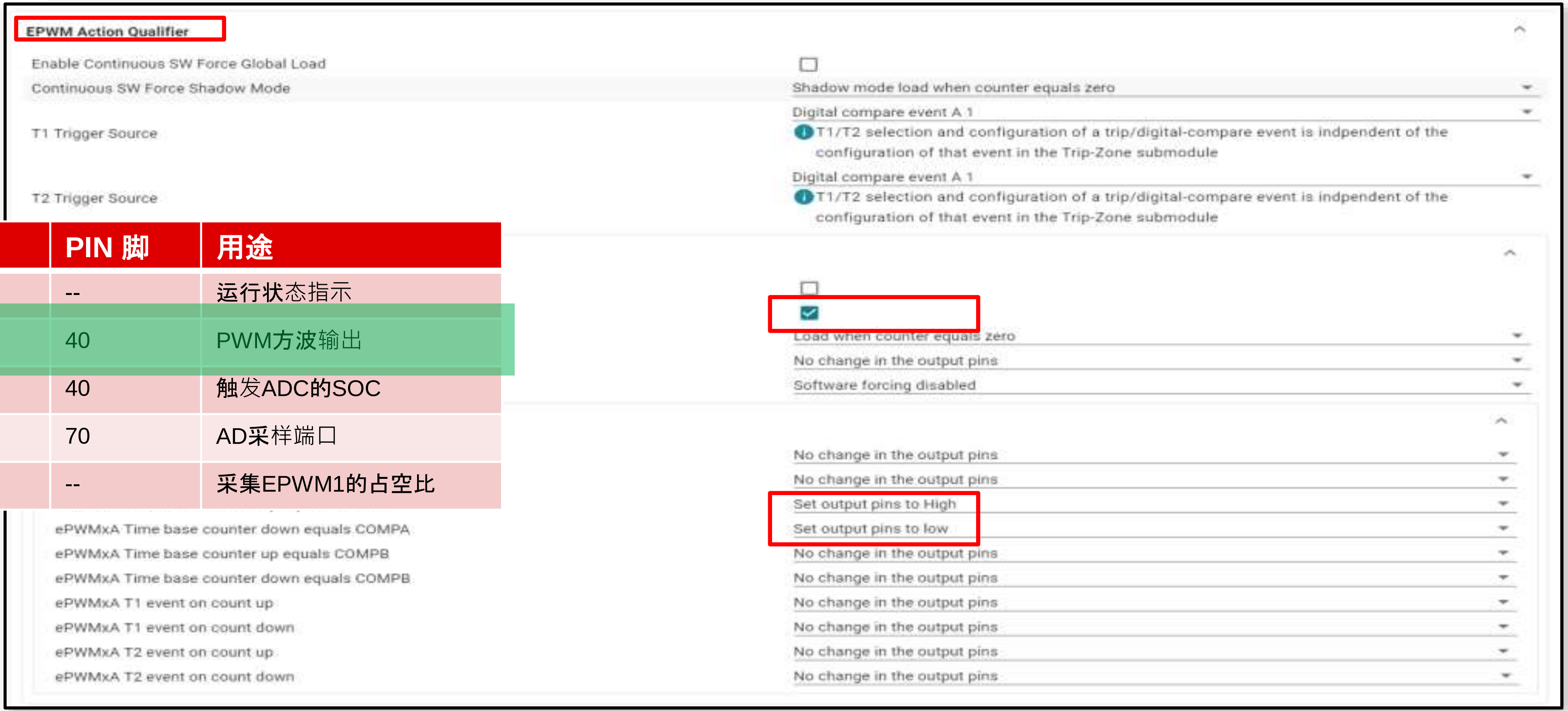
CMPB

CMPC

CMPD

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

配置PWM1



EPWM Action Qualifier

Enable Continuous SW Force Global Load □

Continuous SW Force Shadow Mode: Shadow mode load when counter equals zero

T1 Trigger Source: Digital compare event A 1

T1/T2 selection and configuration of a trip/digital-compare event is indpendent of the configuration of that event in the Trip-Zone submodule

T2 Trigger Source: Digital compare event A 1

T1/T2 selection and configuration of a trip/digital-compare event is indpendent of the configuration of that event in the Trip-Zone submodule

□

☑

Load when counter equals zero

No change in the output pins

Software forcing disabled

No change in the output pins

No change in the output pins

Set output pins to High

Set output pins to low

ePWMxA Time base counter down equals COMPA: No change in the output pins

ePWMxA Time base counter up equals COMPB: No change in the output pins

ePWMxA Time base counter down equals COMPB: No change in the output pins

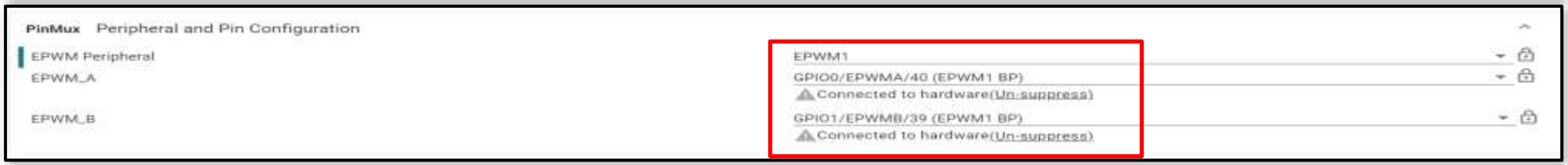
ePWMxA T1 event on count up: No change in the output pins

ePWMxA T1 event on count down: No change in the output pins

ePWMxA T2 event on count up: No change in the output pins

ePWMxA T2 event on count down: No change in the output pins

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比



PinMux Peripheral and Pin Configuration

EPWM Peripheral: EPWM1

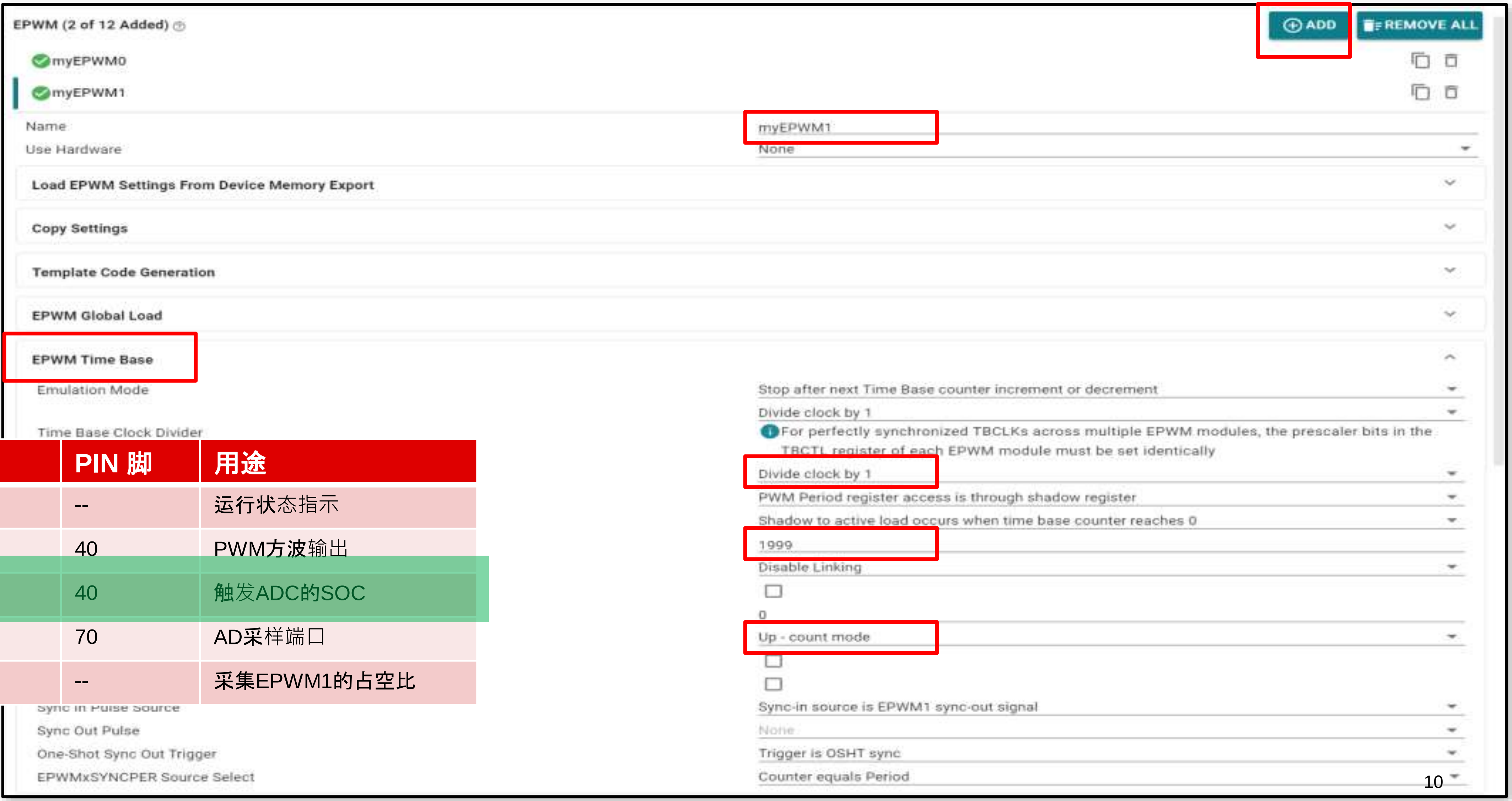
EPWM_A: GPIO0/EPWMA/40 (EPWM1 BP)

Connected to hardware(Un-suppress)

EPWM_B: GPIO1/EPWMB/39 (EPWM1 BP)

Connected to hardware(Un-suppress)

配置PWM2



myEPWM0

myEPWM1

Name: myEPWM1

Use Hardware: None

Emulation Mode: Stop after next Time Base counter increment or decrement

Divide clock by 1

Time Base Clock Divider

For perfectly synchronized TBCLKs across multiple EPWM modules, the prescaler bits in the TBCTL register of each EPWM module must be set identically

Divide clock by 1

PWM Period register access is through shadow register

Shadow to active load occurs when time base counter reaches 0

1999

Disable Linking

☐

0

Up - count mode

☐

☐

Sync In Pulse Source: Sync-in source is EPWM1 sync-out signal

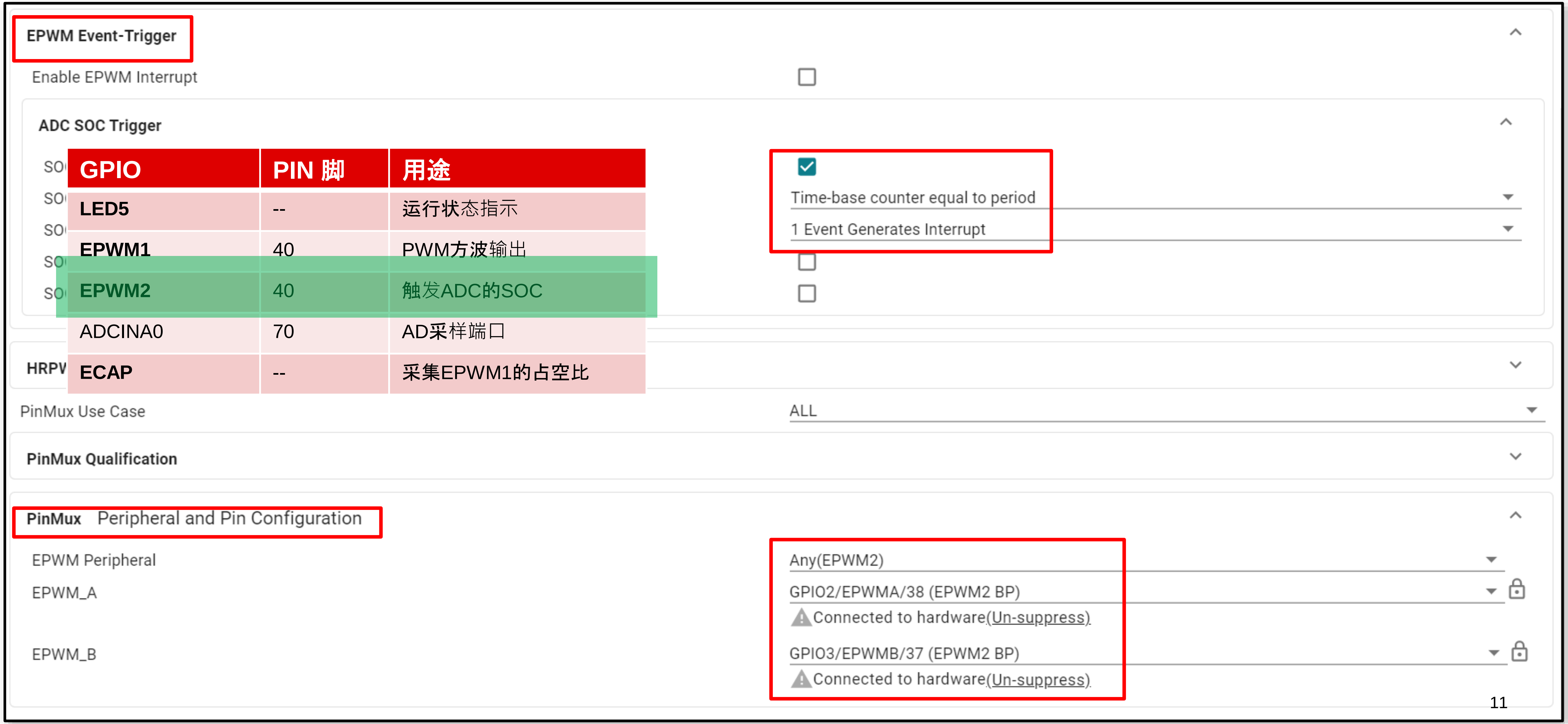
Sync Out Pulse: None

One-Shot Sync Out Trigger: Trigger is OSHT sync

EPWMxSYNCPER Source Select: Counter equals Period

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

配置PWM2



EPWM Event-Trigger

Enable EPWM Interrupt □

ADC SOC Trigger

SO ☑

SO Time-base counter equal to period

SO 1 Event Generates Interrupt

SO □

SO □

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

HRPW

PinMux Use Case: ALL

PinMux Qualification

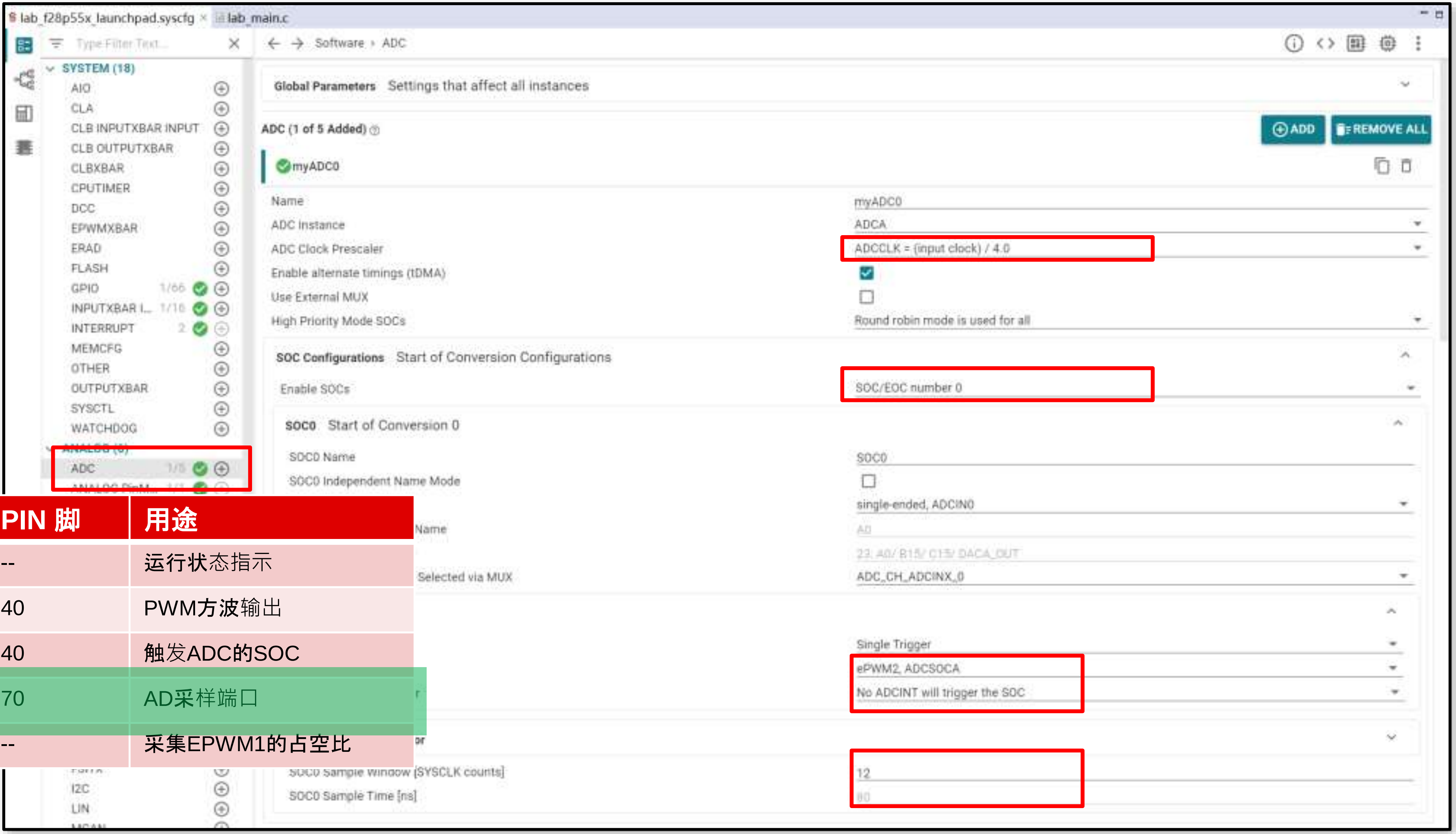
PinMux Peripheral and Pin Configuration

EPWM Peripheral: Any(EPWM2)

EPWM_A: GPIO2/EPWMA/38 (EPWM2 BP)
⚠Connected to hardware(Un-suppress)

EPWM_B: GPIO3/EPWMB/37 (EPWM2 BP)
⚠Connected to hardware(Un-suppress)

配置ADC



Global Parameters Settings that affect all instances

ADC (1 of 5 Added)

myADC0

Name	myADC0
ADC Instance	ADCA
ADC Clock Prescaler	ADCCLK = (input clock) / 4.0
Enable alternate timings (tDMA)	☑
Use External MUX	□
High Priority Mode SOCs	Round robin mode is used for all

SOC Configurations Start of Conversion Configurations

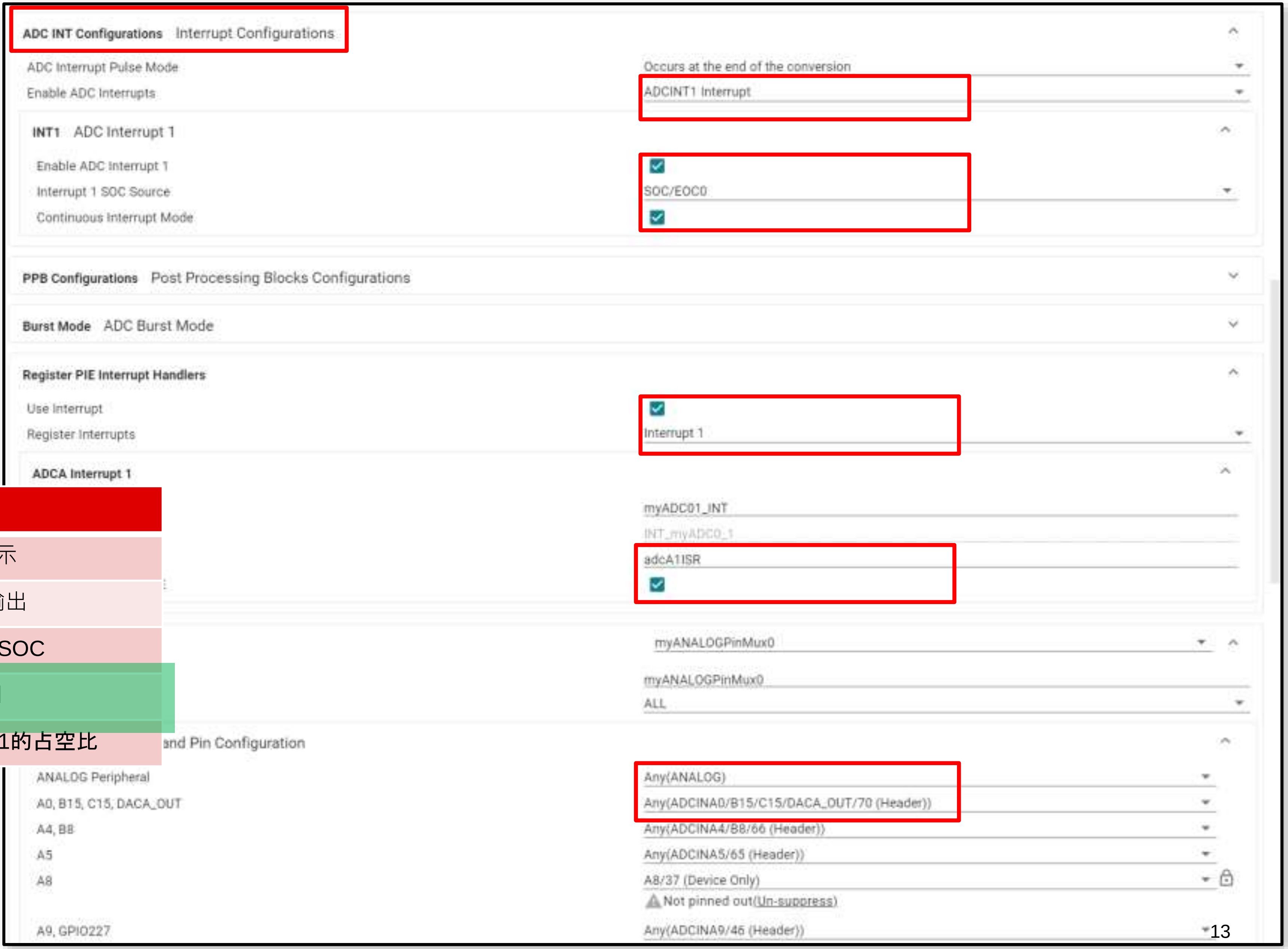
Enable SOCs	SOC/EOC number 0

SOC0 Start of Conversion 0

SOC0 Name	SOC0
SOC0 Independent Name Mode	□
	single-ended, ADCIN0
Name	A0
	23: A0/ B15/ C15/ DACA_OUT
Selected via MUX	ADC_CH_ADCINX_0
	Single Trigger
	ePWM2, ADCSOCA
	No ADCINT will trigger the SOC
SOC0 Sample Window [SYSCLK counts]	12
SOC0 Sample Time [ns]	80

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

配置ADC



ADC INT Configurations Interrupt Configurations

ADC Interrupt Pulse Mode: Occurs at the end of the conversion

Enable ADC Interrupts: ADCINT1 Interrupt

INT1 ADC Interrupt 1

Enable ADC Interrupt 1: ☑

Interrupt 1 SOC Source: SOC/EOC0

Continuous Interrupt Mode: ☑

PPB Configurations Post Processing Blocks Configurations

Burst Mode ADC Burst Mode

Register PIE Interrupt Handlers

Use Interrupt: ☑

Register Interrupts: Interrupt 1

ADCA Interrupt 1

myADC01_INT

INT_myADC0_1

adcA1ISR

☑

myANALOGPinMux0

myANALOGPinMux0

ALL

and Pin Configuration

ANALOG Peripheral: Any(ANALOG)

A0, B15, C15, DACA_OUT: Any(ADCINA0/B15/C15/DACA_OUT/70 (Header))

A4, B8: Any(ADCINA4/B8/66 (Header))

A5: Any(ADCINA5/65 (Header))

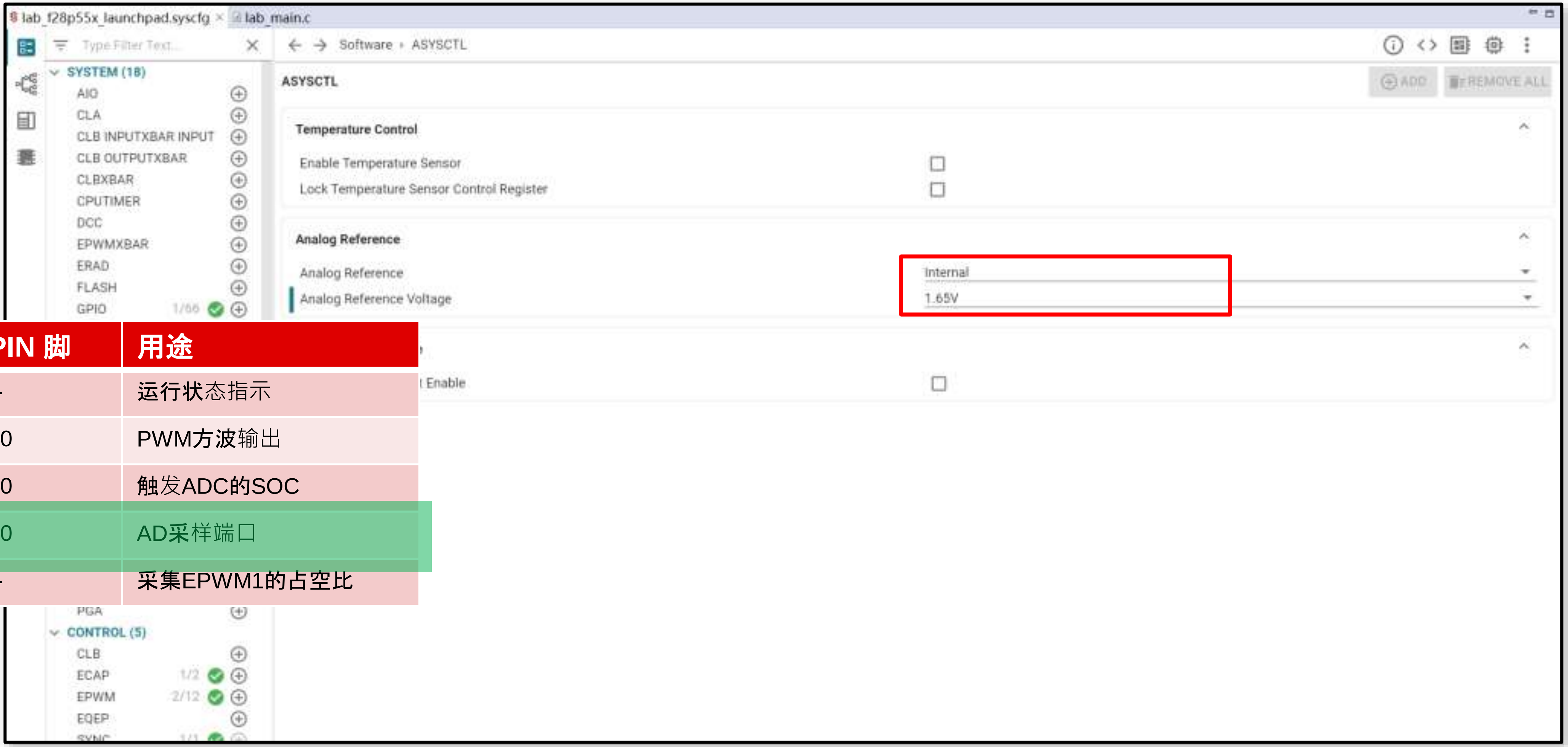
A8: A8/37 (Device Only)

⚠ Not pinned out(Un-suppress)

A9, GPIO227: Any(ADCINA9/46 (Header))

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

配置ADC



ASYSCTL

Temperature Control

Enable Temperature Sensor □

Lock Temperature Sensor Control Register □

Analog Reference

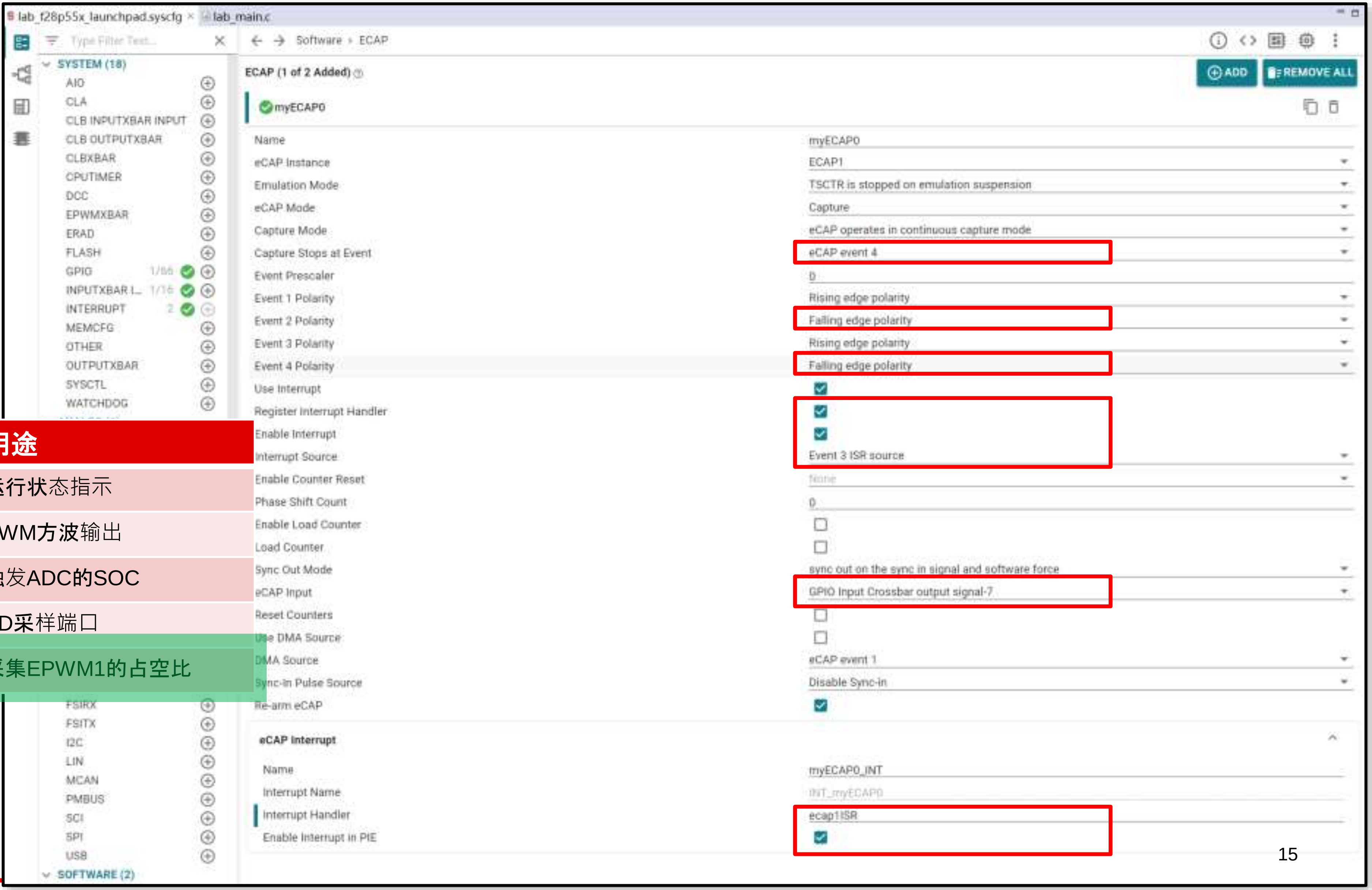
Analog Reference: Internal

Analog Reference Voltage: 1.65V

Enable □

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

配置ECAP



ECAP (1 of 2 Added)

myECAP0

Name	myECAP0
eCAP Instance	ECAP1
Emulation Mode	TSCTR is stopped on emulation suspension
eCAP Mode	Capture
Capture Mode	eCAP operates in continuous capture mode
Capture Stops at Event	eCAP event 4
Event Prescaler	0
Event 1 Polarity	Rising edge polarity
Event 2 Polarity	Falling edge polarity
Event 3 Polarity	Rising edge polarity
Event 4 Polarity	Falling edge polarity
Use Interrupt	☑
Register Interrupt Handler	☑
Enable Interrupt	☑
Interrupt Source	Event 3 ISR source
Enable Counter Reset	None
Phase Shift Count	0
Enable Load Counter	□
Load Counter	□
Sync Out Mode	sync out on the sync in signal and software force
eCAP Input	GPIO Input Crossbar output signal-7
Reset Counters	□
Use DMA Source	□
DMA Source	eCAP event 1
Sync-in Pulse Source	Disable Sync-in
Re-arm eCAP	☑

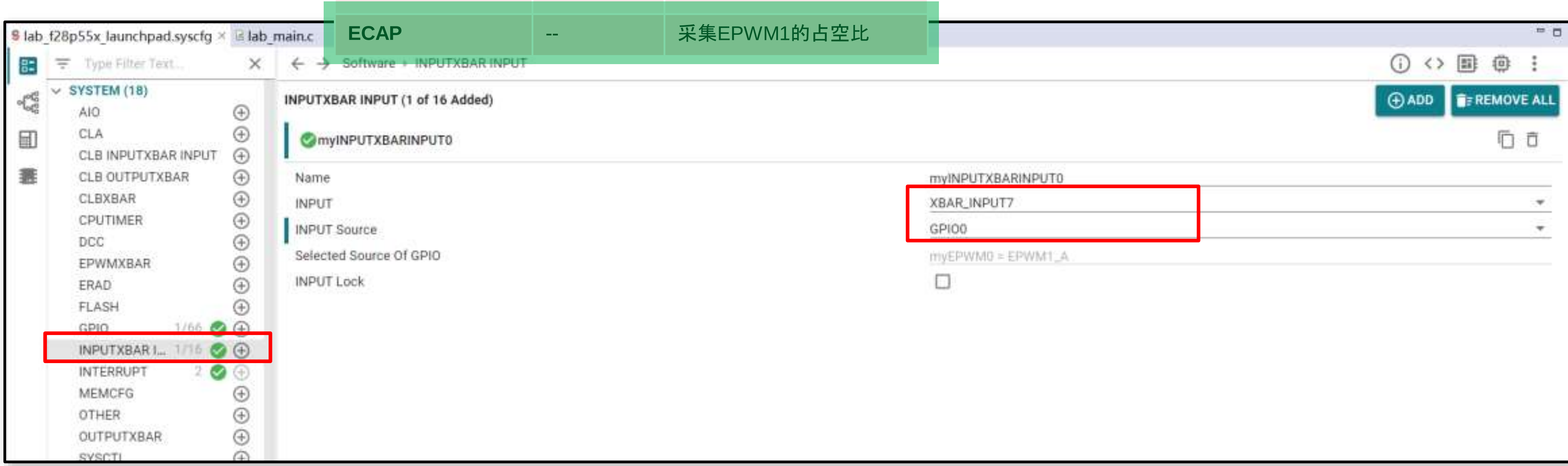
eCAP Interrupt

Name	myECAP0_INT
Interrupt Name	INT_myECAP0
Interrupt Handler	ecap1ISR
Enable Interrupt in PIE	☑

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比

配置ECAP

GPIO	PIN 脚	用途
LED5	--	运行状态指示
EPWM1	40	PWM方波输出
EPWM2	40	触发ADC的SOC
ADCINA0	70	AD采样端口
ECAP	--	采集EPWM1的占空比



INPUTXBAR INPUT (1 of 16 Added)

myINPUTXBARINPUT0

Name	myINPUTXBARINPUT0
INPUT	XBAR_INPUT7
INPUT Source	GPIO0
Selected Source Of GPIO	myEPWM0 = EPWM1_A
INPUT Lock	□

代码

```
__interrupt void adcA1ISR(void)
{
    // Clear interrupt flags.
    Interrupt_clearACKGroup(INT_myADC0_1_INTERRUPT_ACK_GROUP);
    ADC_clearInterruptStatus(myADC0_BASE, ADC_INT_NUMBER1);
    // Write contents of the ADC register to a circular buffer.
    *AdcBufPtr = ADC_readResult(myADC0_RESULT_BASE, myADC0_SOC0);
    if (AdcBufPtr == (AdcBuf + 49))
    {
        // Force buffer to wrap around.
        AdcBufPtr = AdcBuf;
    } else {
        AdcBufPtr += 1;
    }
    if (LedCtr >= 49999) {
        // Divide 50kHz sample rate by 50e3 to toggle LED at a rate of 1Hz.
        GPIO_togglePin(myBoardLED0_GPIO);
        LedCtr = 0;
    } else {
        LedCtr += 1;
    }
    if (DutyModOn) {
        // Divide 50kHz sample rate by 16 to slow down duty modulation.
        if (DutyModCtr >= 15) {
            if (DutyModDir == 0) {
                // Increment State => Decrease Duty Cycle.
                if (ePwm_curDuty >= ePwm_MinDuty) {
                    DutyModDir = 1;
                } else {
                    ePwm_curDuty += 1;
                }
            } else {
                // Decrement State => Increase Duty Cycle.
                if (ePwm_curDuty <= ePwm_MaxDuty) {
                    DutyModDir = 0;
                } else {
                    ePwm_curDuty -= 1;
                }
            }
            DutyModCtr = 0;
        } else {
            DutyModCtr += 1;
        }
    }
    // Set the counter compare value.
    EPWM_setCounterCompareValue(myEPWM0_BASE, EPWM_COUNTER_COMPARE_A, ePwm_curDuty);
}
```

```
__interrupt void ecap1ISR(void)
{
    Interrupt_clearACKGroup(INT_myECAP0_INTERRUPT_ACK_GROUP);
    ECAP_clearGlobalInterrupt(myECAP0_BASE);
    ECAP_clearInterrupt(myECAP0_BASE, ECAP_ISR_SOURCE_CAPTURE_EVENT_3);
    eCapPwmDuty = (int32_t)ECAP_getEventTimeStamp(myECAP0_BASE, ECAP_EVENT_2) -
                  (int32_t)ECAP_getEventTimeStamp(myECAP0_BASE, ECAP_EVENT_1);
    eCapPwmPeriod = (int32_t)ECAP_getEventTimeStamp(myECAP0_BASE, ECAP_EVENT_3) -
                    (int32_t)ECAP_getEventTimeStamp(myECAP0_BASE, ECAP_EVENT_1);
}
```

Board.h

```
uint32_t ePwm_TimeBase;
uint32_t ePwm_MinDuty;
uint32_t ePwm_MaxDuty;
uint32_t ePwm_curDuty;
uint16_t AdcBuf[50];           // Buffer to store ADC samples.
uint16_t *AdcBufPtr = AdcBuf;  // Pointer to ADC buffer samples.
uint16_t LedCtr = 0;           // Counter to slow down LED toggle in ADC ISR.
uint16_t DutyModOn = 0;        // Flag to turn on/off duty cycle modulation.
uint16_t DutyModDir = 0;       // Flag to control duty mod direction up/down.
uint16_t DutyModCtr = 0;       // Counter to slow down rate of modulation.
int32_t eCapPwmDuty;           // Percent = (eCapPwmDuty/eCapPwmPeriod)*100.
int32_t eCapPwmPeriod;         // Frequency = DEVICE_SYSCLK_FREQ/eCapPwmPeriod.
```

```
void main(void)
{
    Device_init();
    Interrupt_initModule();
    Interrupt_initVectorTable();
    Board_init();
    // Initialize variables for EPWM Duty Cycle
    ePwm_TimeBase = EPWM_getTimeBasePeriod(myEPWM0_BASE);
    ePwm_MinDuty = (uint32_t)(0.95f * (float)ePwm_TimeBase);
    ePwm_MaxDuty = (uint32_t)(0.05f * (float)ePwm_TimeBase);
    ePwm_curDuty = EPWM_getCounterCompareValue(myEPWM0_BASE, EPWM_COUNTER_COMPARE_A);
    EINT;
    ERTM;
    for (;;) {
        NOP;
    }
}
```