

Timer

Timer

1. Learning Objectives
 - Timer
 - Basic parameters of the timer
2. Hardware construction
3. Experimental steps
 1. Open the SYSCONFIG configuration tool
 2. Timer parameter configuration
 3. Write the program
 4. Compile
4. Program Analysis
5. Experimental phenomenon

1. Learning Objectives

1. Learn the basic use of the MSPM0G3507 motherboard timer.
2. Realize the timed flipping of LED status.

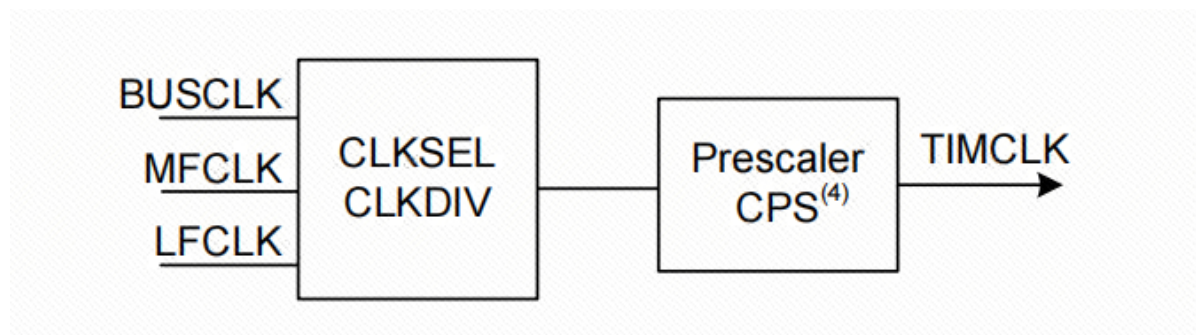
Timer

The timer is a very important peripheral in the microcontroller, used for timing and generating timed interrupts. The core principle of the timer is to record time through an internal counter. The counter is incremented by one per machine cycle. When the counter reaches a preset value, it triggers an interrupt. The program can perform certain operations in the interrupt service program, such as flipping LEDs, reading sensor data, controlling motors, etc.

Basic parameters of the timer

1. Prescaler

The prescaler can divide the timer clock (TIMCLK) frequency by any value between 1 and 256 (1 to 256 is based on the 8-bit timer divider). TIMG can choose BUSCLK, MFCLK, LFCLK as the clock source, and can divide the clock by a maximum of 8. After passing through an 8-bit prescaler, it is finally called the timer count clock.



2. Up or down counting mode

Up counting means that the counter starts counting from 0 to the auto-load value. Once the counter counts to the auto-load value, it will start counting from 0 again and generate an overflow event. Down counting means that the counter starts counting from the auto-load value to 0. Once the counter counts to 0, it will start counting from the auto-load value again and generate an underflow event.

3. Update event

The update event is triggered when the counter overflows or underflows and starts a new counting cycle. The update event can trigger a DMA request to update the timer's operating parameters in time at the beginning of the next counting cycle, which is particularly suitable for real-time control.

2. Hardware construction

The MSPM0G series has a total of 7 timers, which can be divided into 2 types, general timers (TIMG) and advanced control timers (TIMA). Different types of timers have different numbers of functions. Generally, advanced timers have the most functions, followed by general timers.

Table 25-1. TIMx Instance Configuration

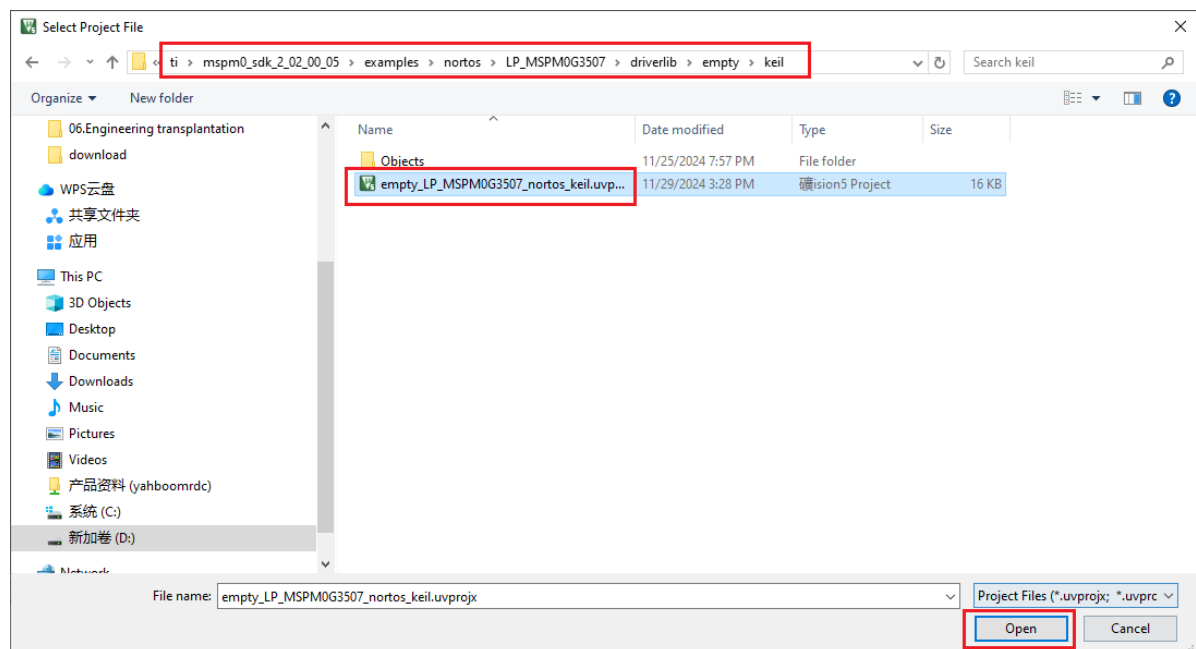
Instance	Power Domain	Counter Resolution	Prescaler	Repeat Counter	CCP Channels (External/Internal)	External PWM Channels	Phase Load	Shadow Load	Shadow CCs	Deadband	Fault Handler	QEI / Hall Input Mode
TIMG0	PD0	16-bit	8-bit	-	2	2	-	-	-	-	-	-
TIMG6	PD1	16-bit	8-bit	-	2	2	-	Yes	Yes	-	-	-
TIMG7	PD1	16-bit	8-bit	-	2	2	-	Yes	Yes	-	-	-
TIMG8	PD0	16-bit	8-bit	-	2	2	-	-	-	-	-	Yes
TIMG12	PD1	32-bit	-	-	2	2	-	-	Yes	-	-	-
TIMA0	PD1	16-bit	8-bit	Yes	4/2	8	Yes	Yes	Yes	Yes	Yes	-
TIMA1	PD1	16-bit	8-bit	Yes	2/2	4	Yes	Yes	Yes	Yes	Yes	-

3. Experimental steps

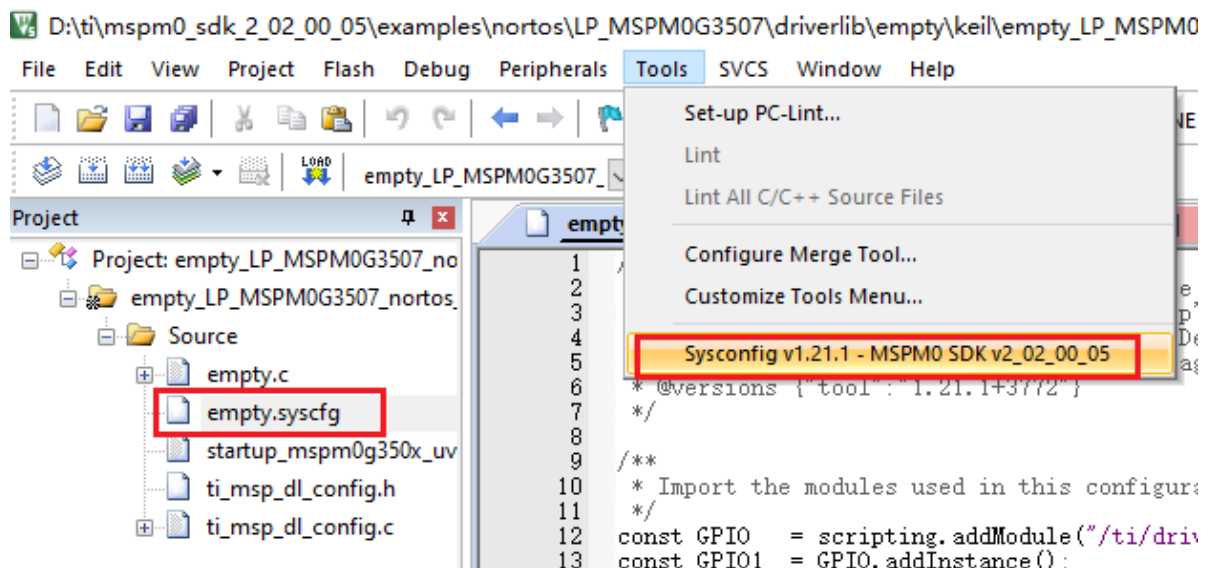
If the project needs to execute or repeat specific tasks at a fixed time, the configuration of the timer is particularly important. The following will introduce how to set the timer to trigger an interrupt every 1 second, and implement the LED state flip in the interrupt service function, so as to achieve the effect of LED on for 1 second and off for 1 second.

1. Open the SYSCONFIG configuration tool

Open the blank project empty in the SDK in KEIL.



Select and open, open the empty.syscfg file in the keil interface, **with the empty.syscfg file open**, then select to open the SYSCONFIG GUI interface in the Tools column.



In SYSCONFIG, select MCU peripherals on the left, find the TIMER option and click to enter. Click ADD in TIMER to add a timer peripheral.

FILEABOUT

RESTART

Type Filter Text...

Software > TIMER

PROJECT CONFIGURATION...

Project Config... 1/1

MSPM0 DRIVER LIBRARY ...

SYSTEM (9)

Board 1/1

Configuration NVM

DMA

GPIO 2

MATHACL

RTC

SYSTICK 1/1

SYSTICK 1/1

WWDTP

ANALOG (6)

ADC12

COMP

DAC12

GPAMP

OPA

VREF

COMMUNICATIONS (6)

I2C

I2C - SMBUS

MCAN

SPI

UART 1/4

UART - LIN

TIMERS (6)

TIMER

TIMER - CAPTURE

TIMER - COMPARE

TIMER - PWM

TIMER - QEI

Timer Fault

TIMER (0 of 7 Added)

ADDREMOVE ALL

Description

The **Timer module** allows you to configure timers in one-shot or periodic mode.

Under *Basic Configuration* users can:

Select Timer Clock configuration

Configure:

One-shot or periodic mode

Enable repeat mode (if supported)

Under *Advanced configuration* users can:

Configure a timer instance to trigger other timer instances under cross-

This module supports the different types of Timers included in MSPM0 includi

The following table includes a description of the features included in **MSPM0G**

Check the datasheet of your selected device for more detailed information.

TIM Name	Power Domain	Resolution	Prescaler	Repeat	CC Ch	Phase Load
TIMG0	PD0	16-bit	8-bit	-	2	-
TIMG6	PD1	16-bit	8-bit	-	2	-
TIMG7	PD1	16-bit	8-bit	-	2	-
TIMG8	PD0	16-bit	8-bit	-	2	-

Click the **Add** button to add a TIMER to your design

Name

TIMER_0

Selected Peripheral

Quick Profiles

Timer Profiles

Custom

Problems

There are no problems in the current design.

Generated Files

Filter: all

File name	Category	Include in build
ti_msp_dl_config.c	MSPM0 Driver Library	<input checked="" type="checkbox"/>
ti_msp_dl_config.h	MSPM0 Driver Library	<input checked="" type="checkbox"/>
Event.dot	MSPM0 Driver Library	<input checked="" type="checkbox"/>
empty.syscfg	Configuration Script	<input type="checkbox"/>

4 Total Files

MSPM0G3507 (Device)

LQFP-64 (PM) (Package)

2. Timer parameter configuration

In this case, select BUSCLK-32MHz for the Timer Clock Source, select 8 for the Timer Clock Divider, and configure 100 for the Timer Clock Prescaler, so the timer frequency is 40KHz. Desired Timer Period sets the timer period to 1S, Timer Mode to period down counting mode, Enable Interrupts turns on 0 overflow interrupt. The GPIO configuration of LED is carried out according to the LED light course.

TIMER (1 of 7 Added) ?

+

ADD

≡

REMOVE ALL

✓

TIMER_0

📄

🗑

Name

TIMER_0

Selected Peripheral

TIMG0

Quick Profiles

^

Timer Profiles

Custom

▼

Basic Configuration

^

Clock Configuration

^

Timer Clock Source

BUSCLK

▼

Timer Clock Divider

Divided by 8

▼

Calculated Timer Clock Source

4000000

Timer Clock Prescaler

100

Calculated Timer Clock Values

^

Timer Clock Frequency

40.00 kHz

Timer Period Range And Resolution

25.00 μs to 1.64 s w/ resolution of 25.00 μs

Timer Mode

Periodic Down Counting

▼

Desired Timer Period

1000 ms

Actual Timer Period

1.00 s

Start Timer ?

✓

Advanced Configuration

▼

Interrupts Configuration

^

Enable Interrupts

Zero event

▼

Interrupt Priority

Default

▼

Event Configuration

▼

PinMux Peripheral and Pin Configuration

^

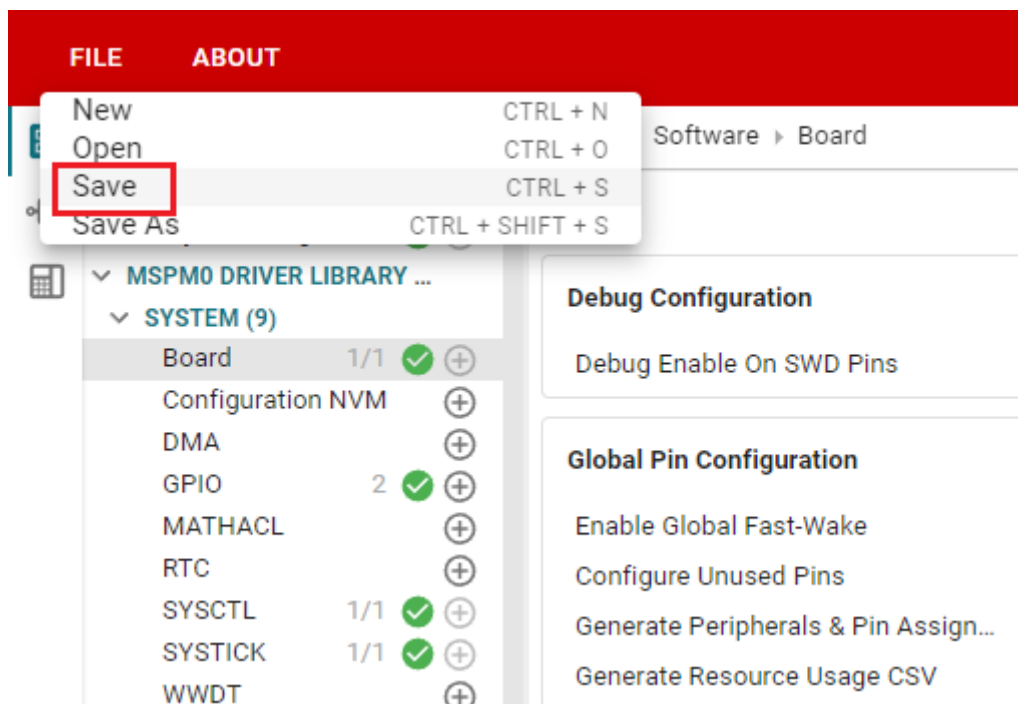
Timer Peripheral

TIMG0

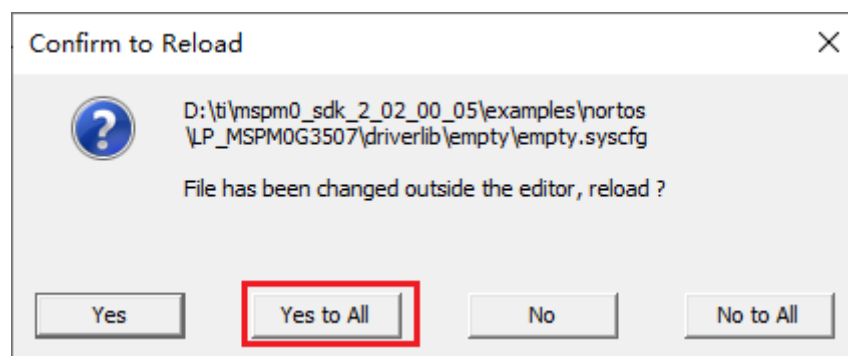
▼

🔒

Click SAVE to save the configuration in SYSCONFIG, then turn off SYSCONFIG and return to keil.



Select **Yes to All** in the pop-up window



Similarly, we also need to confirm whether the `ti_msp_dl_config.c` and `ti_msp_dl_config.h` files are updated. Compile directly, and the compilation will automatically update to keil. If there is no update, we can also copy the file content in `SYSCONFIG`.

3. Write the program

In the `empty.c` file, write the following code

```
#include "ti_msp_dl_config.h"

int main(void)
{
    SYSCFG_DL_init();
    //清除定时器中断标志 Clear the timer interrupt flag
    NVIC_ClearPendingIRQ(TIMER_0_INST_INT_IRQN);
    //使能定时器中断 Enable timer interrupt
    NVIC_EnableIRQ(TIMER_0_INST_INT_IRQN);

    while (1)
    {

    }
}

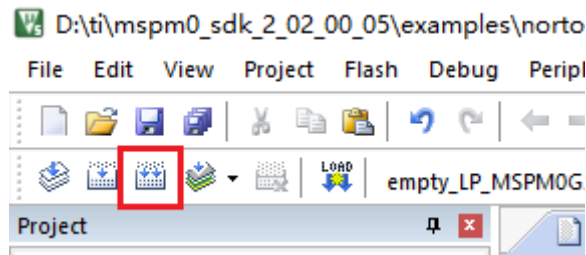
//定时器的中断服务函数 已配置为1秒的周期
```

```
//The timer interrupt service function has been configured to a period of 1
second
void TIMER_0_INST_IRQHandler(void)
{
    //如果产生了定时器中断 If a timer interrupt occurs
    switch( DL_TimerG_getPendingInterrupt(TIMER_0_INST) )
    {
        case DL_TIMER_IIDX_ZERO://如果是0溢出中断 If it is 0 overflow interrupt
            //将LED灯的状态翻转 Flip the LED status
            DL_GPIO_togglePins(LED1_PORT, LED1_PIN_2_PIN);
            break;

        default://其他的定时器中断 other timer interrupts
            break;
    }
}
```

4. Compile

Click the Rebuild icon. The following prompt appears, indicating that the compilation is complete and there are no errors.



```
Generating Code (empty.syscfg)...
Unchanged D:\ti\mspm0_sdk_2_02_00_05\examples\nortos\LP_MSPM0G3507\driverlib\empty\ti_msp_dl_config.c...
Unchanged D:\ti\mspm0_sdk_2_02_00_05\examples\nortos\LP_MSPM0G3507\driverlib\empty\ti_msp_dl_config.h...
Unchanged D:\ti\mspm0_sdk_2_02_00_05\examples\nortos\LP_MSPM0G3507\driverlib\empty\Event.dot...
assembling startup_mspm0g350x_uvision.s...
compiling empty.c...
compiling ti_msp_dl_config.c...
linking...
Program Size: Code=544 RO-data=208 RW-data=0 ZI-data=352
FromELF: creating hex file...
".\Objects\empty_LP_MSPM0G3507_nortos_keil.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:06
```

4. Program Analysis

- empty.c

```
//定时器的中断服务函数 已配置为1秒的周期
//The timer interrupt service function has been configured to a period of 1
second
void TIMER_0_INST_IRQHandler(void)
{
    //如果产生了定时器中断 If a timer interrupt occurs
    switch( DL_TimerG_getPendingInterrupt(TIMER_0_INST) )
    {
        case DL_TIMER_IIDX_ZERO://如果是0溢出中断 If it is 0 overflow interrupt
            //将LED灯的状态翻转 Flip the LED status
            DL_GPIO_togglePins(LED1_PORT, LED1_PIN_2_PIN);
            break;

        default://其他的定时器中断 other timer interrupts
```

```
        break;  
    }  
}
```

This is a timer interrupt service program, which is used to handle timer overflow interrupts and perform corresponding operations. Every time the timer overflows, the state of an LED is flipped.

5. Experimental phenomenon

After the program is downloaded, you can see that the LED light is flashing according to the configuration of 1 second period.