Light up the LED

Light up the LED

- 1. Learning objectives
- 2. Hardware construction
- 3. Experimental steps
 - 1. Open the SYSCONFIG configuration tool
 - 2. Parameter settings
 - 2.1 Set port parameters
 - 2.2 Set pin parameters
 - 2.3 Save and update the configuration file
 - 3. Write the program
 - 4. Compile
- 4. Program Analysis
- 5. Experimental phenomenon

1. Learning objectives

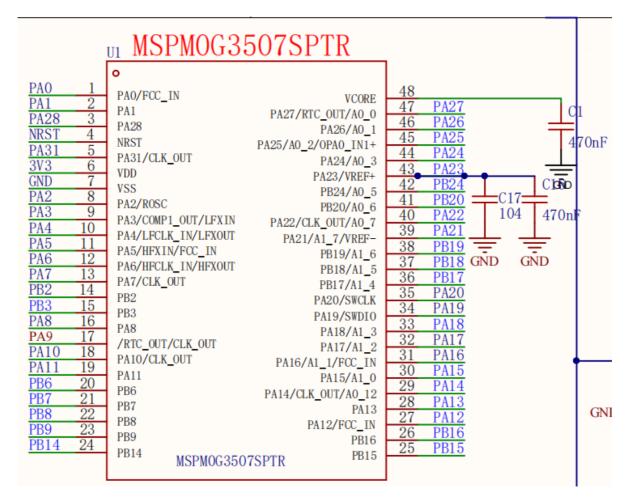
- 1. Learn the basic use of the pins of the MSPM0G3507 motherboard.
- 2. Understand how to control the onboard LED lights.

2. Hardware construction

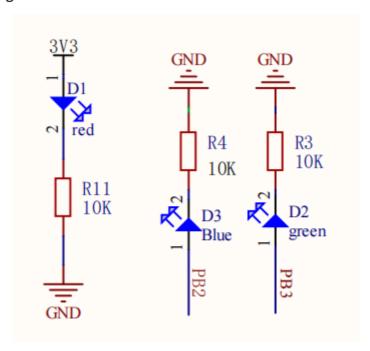
This course does not require additional hardware equipment, and can directly use the onboard LED lights on the MSPM0G3507 motherboard.

We set up two LED user lights on the motherboard, and users can DIY their functions. Take **LED1** as an example below.

MSPM0G3507 main control diagram



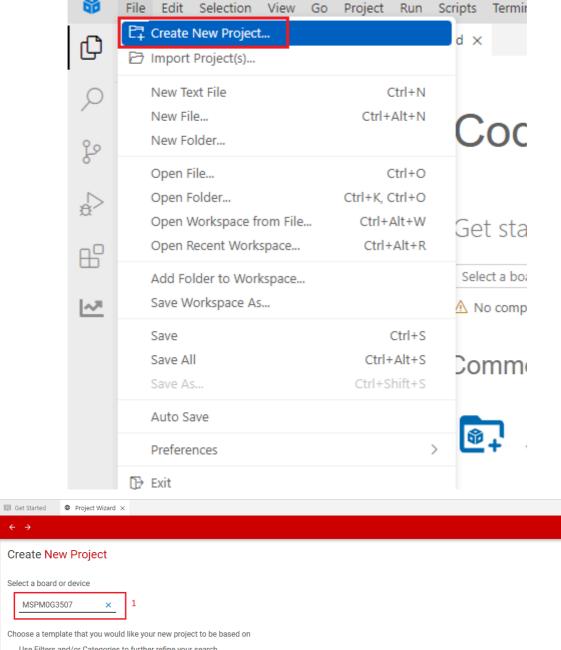
LED schematic diagram

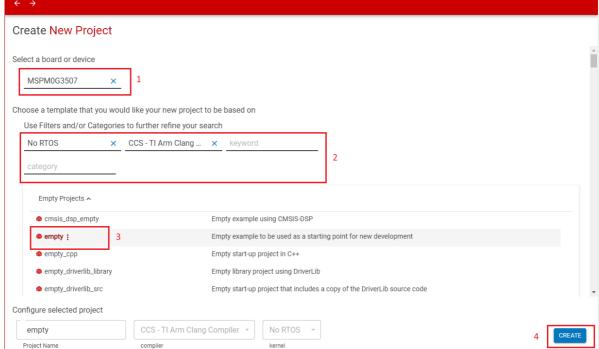


3. Experimental steps

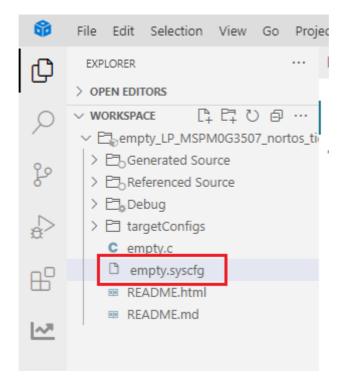
1. Open the SYSCONFIG configuration tool

Create a blank project empty in CCS.

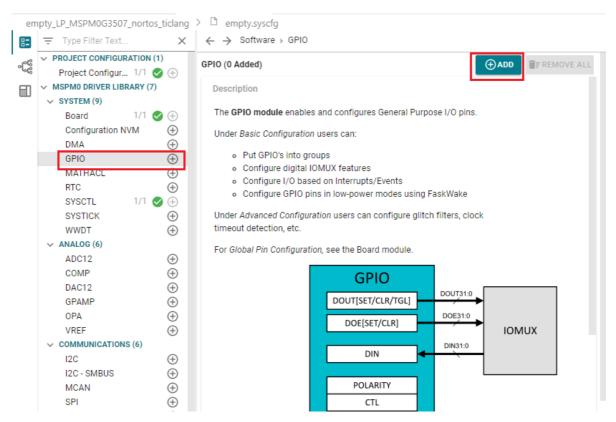




Find and open the empty.syscfg file in the left workspace of CCS.



In SYSCONFIG, select MCU peripherals on the left, find the GPIO option and click to enter. Click ADD in GPIO to add a group of GPIO.



2. Parameter settings

2.1 Set port parameters

Set GPIO parameters and name the pin LED1

From the LED schematic, we can see that the port where LED1 is located is PB2, so set the Port to PORTB.

GPIO (1 Added) ②		⊕ ADD
⊘ LED1		ā d
Name	LED1	
Port	PORTB	,
Port Segment	Any	,
	Ally	

Parameter description:

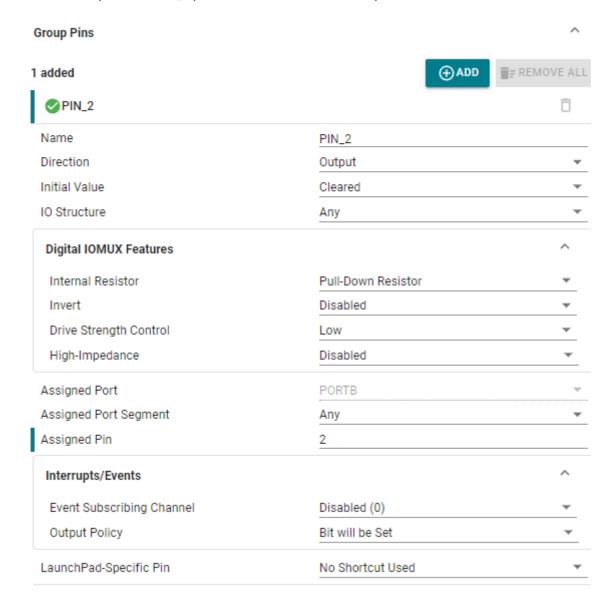
Name: Custom name of the GPIO instance. By default, the name starts with a numeric suffix "0"; we can customize the name to reflect the purpose of the module (for example, name the GPIO "LED1" so that we know that this pin is specifically used to control LED1).

Port: The port where the GPIO instance is located. The LED is connected to the GPIOB2 pin, so only PORTB can be selected.

Port Segment: Set the pull-up and pull-down resistors on the port. Note that it is the pull-up and pull-down of the port, which sets the entire GPIOB port.

2.2 Set pin parameters

Set the pin to output mode, set the pin to output low level by default, set the pin to pull-down mode, set the pin to GPIOB_2 pin, and do not enable interrupt events.



Parameter description:

Name: User-defined pin name, set to PIN_2.

Direction: Set the pin mode, input and output. Here we control the LED light and choose the output mode.

Initial Value: Set the initial state of the pin, which can only be set when configured as output mode. There are two options, clear and set. Clear means output low level, and set means output high level.

IO Structure: Set the IO structure. There are multiple options, including Any, Standard, Standard with Wake, High-Speed, and 5V Tolerant Open Drain.

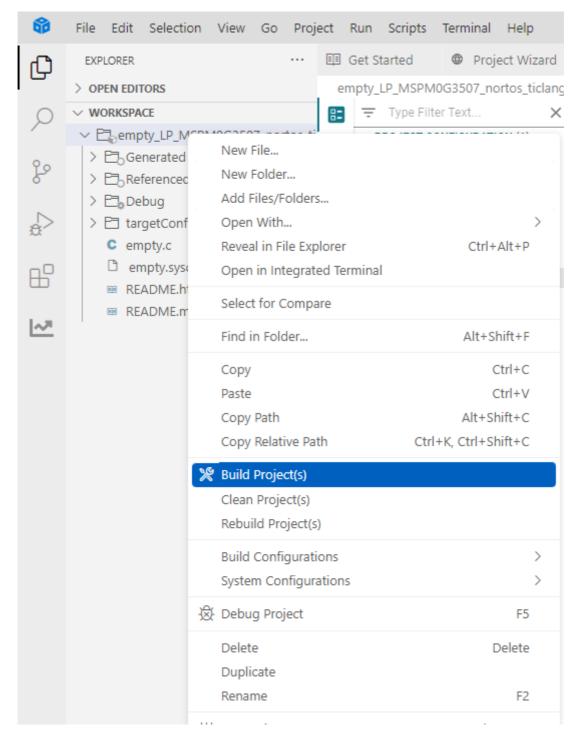
Internal Resistor: Set the pull-up and pull-down resistors of the pin. There are three options, no pull-up, pull-up, and pull-down. Here, the pull-down resistor is selected according to the connection method of the LED light.

Assigned Pin: Set the pin number. Fill in the corresponding pin number for the pin to be controlled. For example, if the LED in the development board is connected to GPIOB2, fill in 2.

2.3 Save and update the configuration file

Save the configuration in the .syscfg file using the shortcut key Ctrl+S.

Compile after saving



After compiling, we can see that there are two more files in the Debug folder, ti_msp_dl_config.c and ti_msp_dl_config.h. These two files are the peripheral low-level driver configuration codes generated based on the .sysfig file we just saved.

```
File Edit Selection View Go Project Run Scripts Terminal Help
                                                                                                  Code Composer Studio Theia
                                                     ··· 🕮 Get Started 🏶 Project Wizard 🔀 empty.syscfg 🕻 ti_msp_dl_config.c 🕻 ti_msp_dl_config.h 🗙
Ф
                                                            empty\_LP\_MSPM0G3507\_nortos\_ticlang \ \gt \ Debug \ \gt \ \ \textbf{c} \ \ ti\_msp\_dl\_config.h \ \gt \dots

∨ WORKSPACE

∨ □ empty_LP_MSPM0G3507_nortos_ticlang [Debug] 1

       > Ph Generated Source
                                                                  /* clang-format off */
       > En Referenced Source
       ∨ ⊟₀ Debug
                                                             73 #define POWER_STARTUP_DELAY
           ccsObjs.opt
          device_linker.cmd
                                                                 #define CPUCLK FREO
                                                                                                                                             32000000
           device.cmd.genlibs
          device.opt
           o empty_LP_MSPM0G3507_nortos_ticlang_linkInfo....
~7
          □ empty_LP_MSPM0G3507_nortos_ticlang.map
                                                                    /* Port definition for Pin Group LED1 */
          mempty LP MSPM0G3507 nortos ticlang.out
                                                                   #define LED1_PORT
                                                                                                                                              (GPIOB)
          empty.d
          m empty.o
                                                                  /\ast Defines for PIN_2: GPIOB.2 with pinCMx 15 on package pin 50 ^\ast/
                                                                                                                                     (DL_GPIO_PIN_2)
          (IOMUX PINCM15
           makefile
           objects.mk
                                                                  /* clang-format on */
           sources.mk
          startup_mspm0g350x_ticlang.d
                                                             91
                                                                  void SYSCFG_DL_initPower(void);
          startup_mspm0g350x_ticlang.o
           subdir_rules.mk
                                                           ..., costnession, cost, compiler, to ego on milym_4.0.0.Els/ozin, con melong.cxc
Finished building: "ti_msp_dl_config.c"
            subdir_vars.mk
          C ti msp dl config.c
                                                            Building file: "D:/ti/mspm0 sdk 2 03 00 07/source/ti/devices/msp/m0p/startup system files/ticlang/
            ti_msp_dl_config.d
          C ti_msp_dl_config.h
                                                             "D:/ti/ccstheia151/ccs/tools/compiler/ti-cgt-armllvm 4.0.0.LTS/bin/tiarmclang.exe" -c @"device.opt
            ti msp dl config.o
                                                           Finished building: "D:/ti/mspm0_sdk_2_03_00_07/source/ti/devices/msp/m0p/startup_system_files/ticl
        > 🖹 targetConfigs
          C empty.c
                                                            Building target: "empty_LP_MSPM0G3507_nortos_ticlang.out"
                                                            Invoking: Arm Linker
"D:/ti/ccstheia151/ccs/tools/compiler/ti-cgt-armllvm_4.0.0.LTS/bin/tiarmclang.exe" @"device.opt"
          🗅 empty.syscfg
          ■ README.htm
                                                            Finished building target: "empty_LP_MSPM0G3507_nortos_ticlang.out"
          ■ README.md
                                                            **** Build Finished ****
503
```

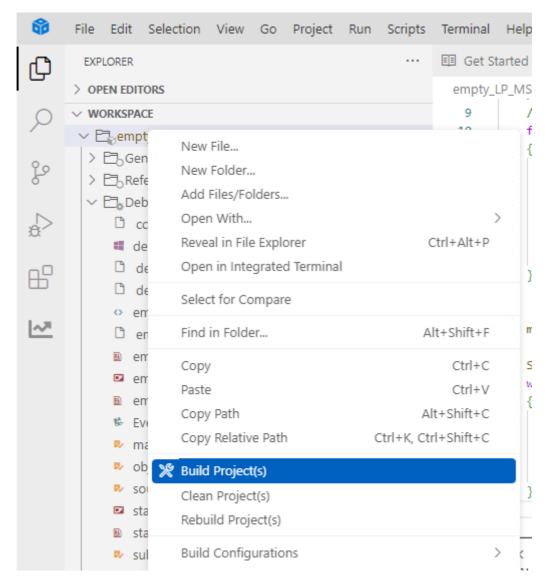
3. Write the program

In the empty.c file, write the following code

```
#include "ti_msp_dl_config.h"
//自定义延时(不精确) Custom delay (not precise)
void delay_ms(unsigned int ms)
{
   unsigned int i, j;
   // 下面的嵌套循环的次数是根据主控频率和编译器生成的指令周期大致计算出来的,
   // 需要通过实际测试调整来达到所需的延时。
   // The number of nested loops below is roughly calculated based on the master
control frequency and the instruction cycle generated by the compiler, and needs
to be adjusted through actual testing to achieve the required delay.
   for (i = 0; i < ms; i++)
   {
       for (j = 0; j < 8000; j++)
           // 仅执行一个足够简单以致于可以预测其执行时间的操作
           // Perform only one operation that is simple enough to predict its
           __asm__("nop"); // "nop" 代表"无操作",在大多数架构中,这会消耗一个或几个时钟
周期 "nop" stands for "no operation", which on most architectures consumes one or
a few clock cycles
       }
   }
}
int main(void)
{
   SYSCFG_DL_init();
```

```
while (1) {
    DL_GPIO_clearPins(LED1_PORT,LED1_PIN_2_PIN);//输出低电平 Output low level delay_ms(1000);//延时大概1S Delay about 1S
    DL_GPIO_setPins(LED1_PORT,LED1_PIN_2_PIN); //输出高电平 Output high level delay_ms(1000);//延时大概1S Delay about 1S
}
```

4. Compile



It shows that the compilation is successful here, and the program can be downloaded to the development board.

```
File Edit Selection View Go Project Run Scripts Terminal Help
                                                                      <sub>C</sub>
                                                                                            // The number of nested loops below is roughly calculated based on the master control frequency and the instruction cycle g Tokenen (i = 0; i < ms; i++)
                                                                                               for (j = 0; j < 8000; j++)
 go
                                                                                                    // 仅执行一个足够简单以数于可以预测其执行时间的操作
// Perform only one operation that is simple enough to predict its execution time
__asm_("nop"); // "nopo" (**死漢師*"、在子家聚集的中、这会網長一个威几个时時開闢 "nop" stands for "no operation", which
 ₽
 B
                    empty LP MSPM0G3507 nortos ticlang linkInfo....
 ~71
                empty_LP_MSPM0G3507_nortos_ticlar
empty_LP_MSPM0G3507_nortos_ticlar
                                                                                               DL_GPIO_clearPins(LED1_PORT,LED1_PIN_2_PIN);//輸出低电平 Output low level delay_ms(1000);//提时大概15 Delay about 15 DL_GPIO_setPins(LED1_PORT,LED1_PIN_2_PIN); //输出离电平 Output high level delay_ms(1000);//提射大概15 Delay about 15
                                                                              26
27
28
29
30
31
                 startup_mspm0g350x_ticlang.d
startup_mspm0g350x_ticlang.o
subdir_rules.mk
                                                                              Finished building: "ti_msp_dl_config.c"
                 C ti msp dl config.o
                 ti_msp_dl_config.d

ti_msp_dl_config.d

ti_msp_dl_config.o
                                                                               "0:/ti/ccstheia151/ccs/tools/compiler/ti-cgt-armllvm_4.0.0.LTS/bin/tiarmclang.exe" -c @"device.opt" -march=thumbv6m -mcpu=cortex-m0plus -mfloat-abi=sor
Finished building: "0:/ti/mspm0_sdk_2_03_00_07/source/ti/devices/msp/m0p/startup_system_files/ticlang/startup_mspm0g350x_ticlang.c"
              targetConfigs
                                                                                TO:/ti/cstheiai51/cs/tools/compiler/ti-cgt-armllvm_4.0.0.LTS/bin/tiarmclang.exe" @"device.opt" -march=thumbv6m -mcpu=cortex-m0plus -mfloat-abi=soft
                                                                              Finished building target: "empty_LP_MSPM0G3507_nortos_ticlang.out
               README.md
```

4. Program Analysis

• dl_gpio.h

```
ti > mspm0 sdk 2 02 00 05 > source > ti > driverlib > C dl gpio.h
           gpio->DOUT31_0 = doutVal;
2201
2202
2203
2204
          @brief
                     Set a group of GPIO pins
2205
2206
2207
          @param[in] gpio Pointer to the register overlay for the peripheral
2208
           @param[in] pins Pins to set high. Bitwise OR of @ref DL_GPIO_PIN.
2209
       __STATIC_INLINE void DL_GPIO_setPins(GPIO_Regs* gpio, uint32_t pins)
2210
2211
           gpio->DOUTSET31_0 = pins;
2212
2213
2214
2215
       * @brief Clear a group of GPIO pins
2216
2217
        * @param[in] gpio Pointer to the register overlay for the peripheral
2218
          @param[in] pins Pins to clear. Bitwise OR of @ref DL_GPIO_PIN.
2219
2220
       __STATIC_INLINE void DL_GPIO_clearPins(GPIO_Regs* gpio, uint32_t pins)
2221
2222
2223
           gpio->DOUTCLR31_0 = pins;
2224
```

_STATIC_INLINE void DL_GPIO_setPins(GPIO_Regs* gpio, uint32_t pins): This function is to control the pin to output a high level.

__STATIC_INLINE void DL_GPIO_clearPins(GPIO_Regs* gpio, uint32_t pins): This function is used to control the pin to output a low level.

· empty.c

```
empty_LP_MSPM0G3507_nortos_ticlang > C empty.c >
    #include "ti_msp_dl_config.h"
     //自定义延时 (不精确) Custom delay (not precise)
     void delay_ms(unsigned int ms)
         unsigned int i, j;
         // 下面的嵌套循环的次数是根据主控频率和编译器生成的指令周期大致计算出来的,
         // 需要通过实际测试调整来达到所需的延时。
         // The number of nested loops below is roughly calculated based on the master control frequency and the instruction cycle gener
 9
         for (i = 0; i < ms; i++)
10
11
             for (j = 0; j < 8000; j++)
13
                // 仅执行一个足够简单以致于可以预测其执行时间的操作
                // Perform only one operation that is simple enough to predict its execution time __asm__("nop"); // "nop" 代表"无操作",在大多数架构中,这会消耗一个或几个时钟周期 "nop" stands for "no operation",which on
15
16
17
18
19
     int main(void)
22
23
        SYSCFG_DL_init();
24
        while (1)
25
            DL_GPIO_clearPins(LED1_PORT,LED1_PIN_2_PIN);//输出低电平 Output low level
           delay_ms(1000);//延时大概1S Delay about 1S
            DL_GPIO_setPins(LED1_PORT,LED1_PIN_2_PIN); //輸出高电平 Output high level
29
            delay_ms(1000);//延时大概1S Delay about 1S
30
31
```

Using an imprecise delay, set PB2 to output a low level, and then output a high level after a delay of about 1s, and then delay for about 1s. This process is repeated.

5. Experimental phenomenon

After the program is downloaded, you can see that the LED1 on the MSPM0 development board is lit and flashes every about 1s.

