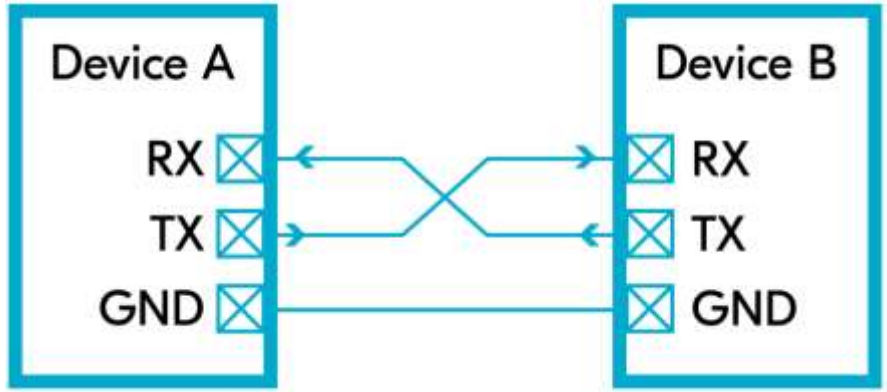
F28P55x编程实例Labs-SCI

- Code Composer Studio
- C2000Ware
- LaunchXL-F28P55x

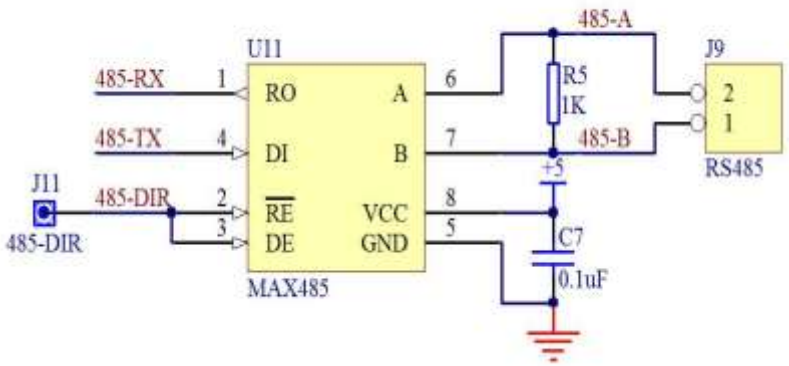
SCI/UART

Serial Communication Interface, 串行通讯接口，UART

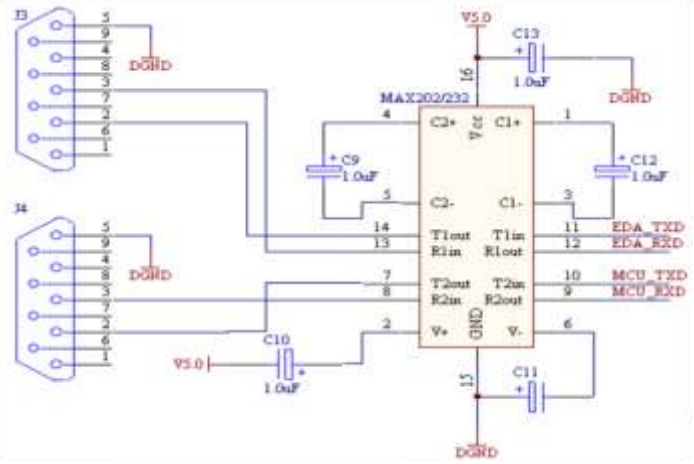


- 异步通信
- 双线制，TXD-发送数据口，RXD-接收数据口
- 波特率可编程，最大64k
- 数据格式，1个起始位，数据长度1-8字节可编程，偶校验/奇校验/无校验
- 停止位1/2个
- 半双工/全双工模式

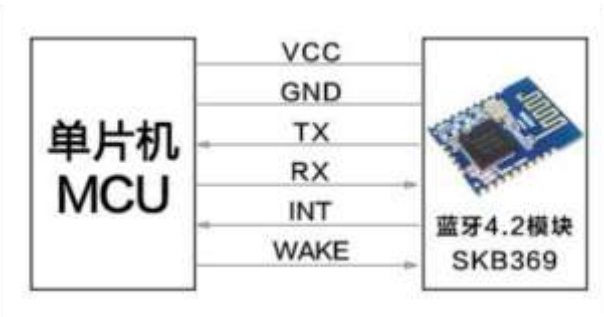
RS485接口



RS232接口



模组/AT指令



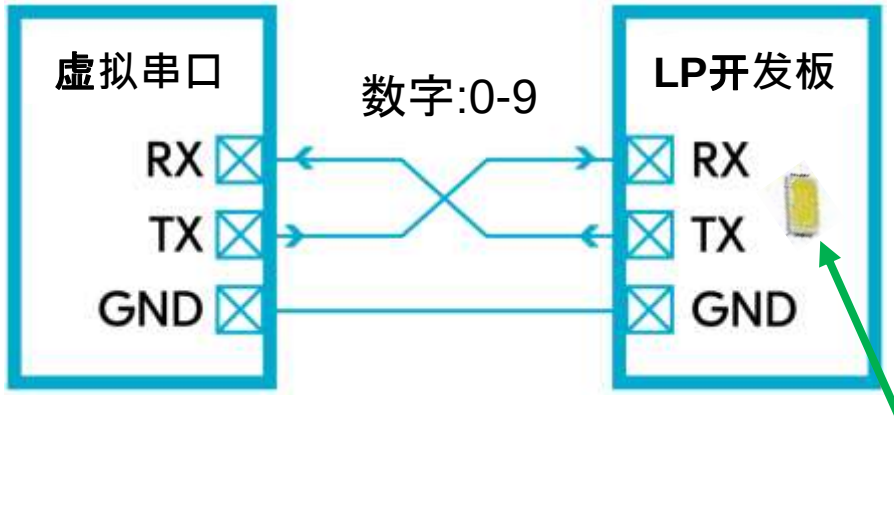
SCI/UART

功能实现

用CCS自带的串口虚拟工具，实现和LP开发板的串口通讯，并实现串口调节LED的闪烁频率

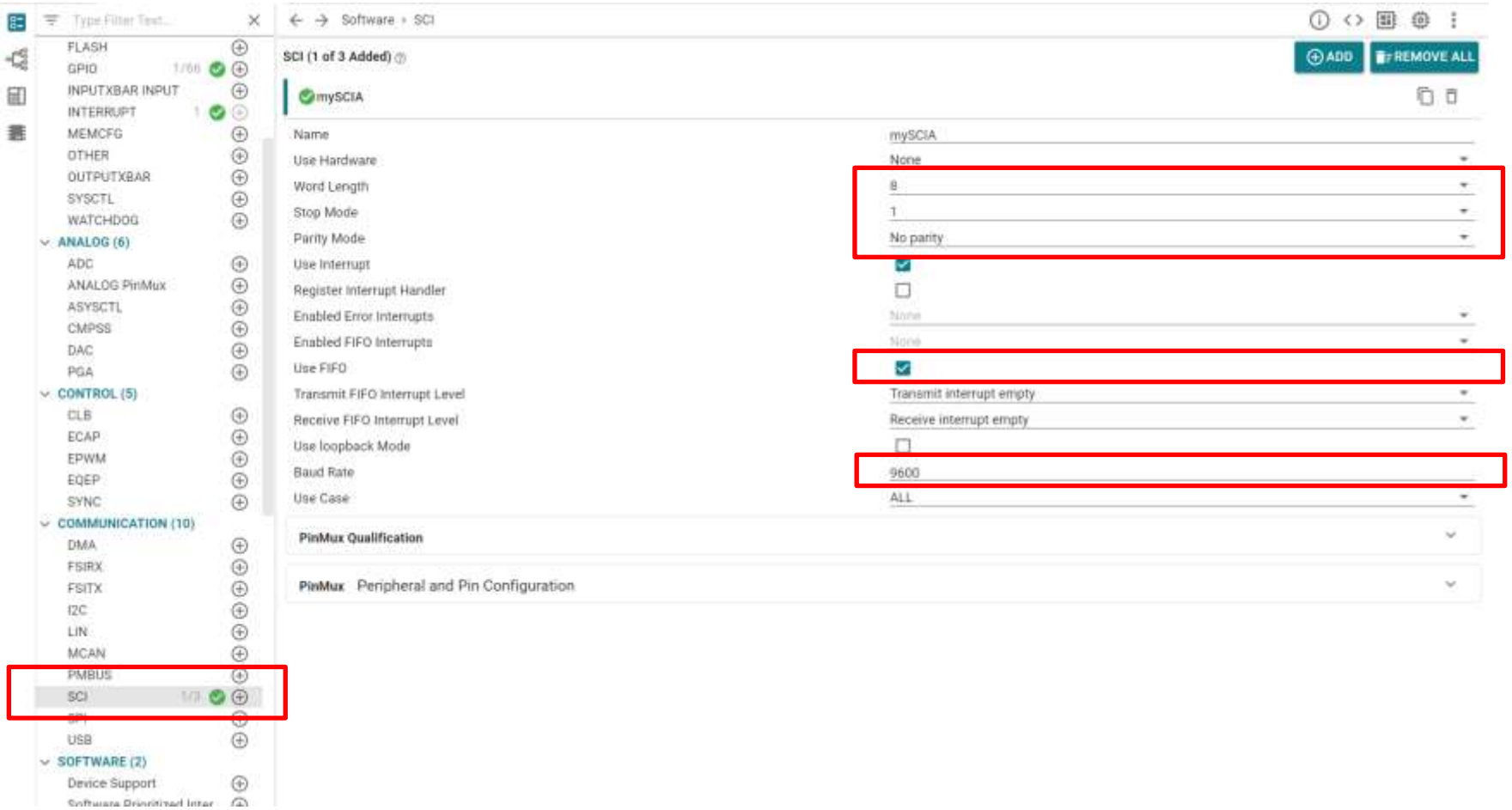
实现步骤

- 复制空白工程
- Sysconfig配置SCI
- Sysconfig配置定时器
- Sysconfig配置GPIO驱动LED
- 编写应用代码



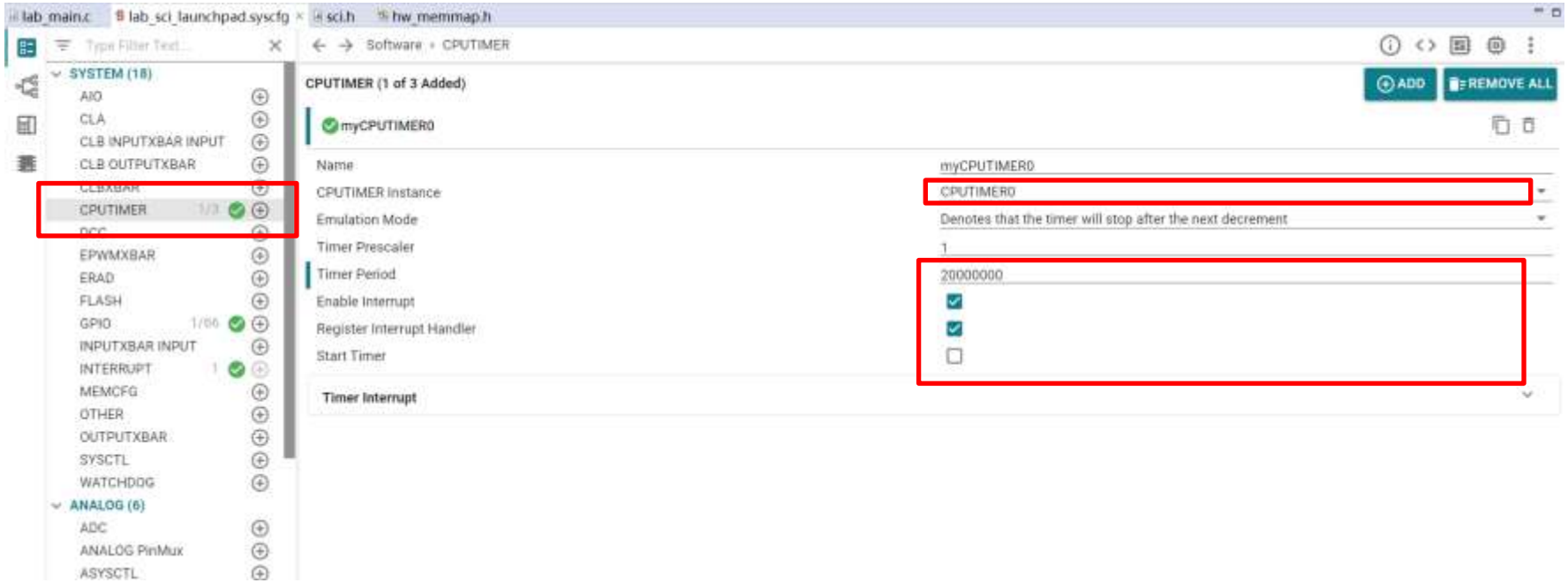
数字0-9控制不同的闪烁频率

SCI配置



- Word Length of 8 bits
- Stop Mode of 1
- No Parity
- FIFO enabled
- Baud Rate of 9600

定时器配置



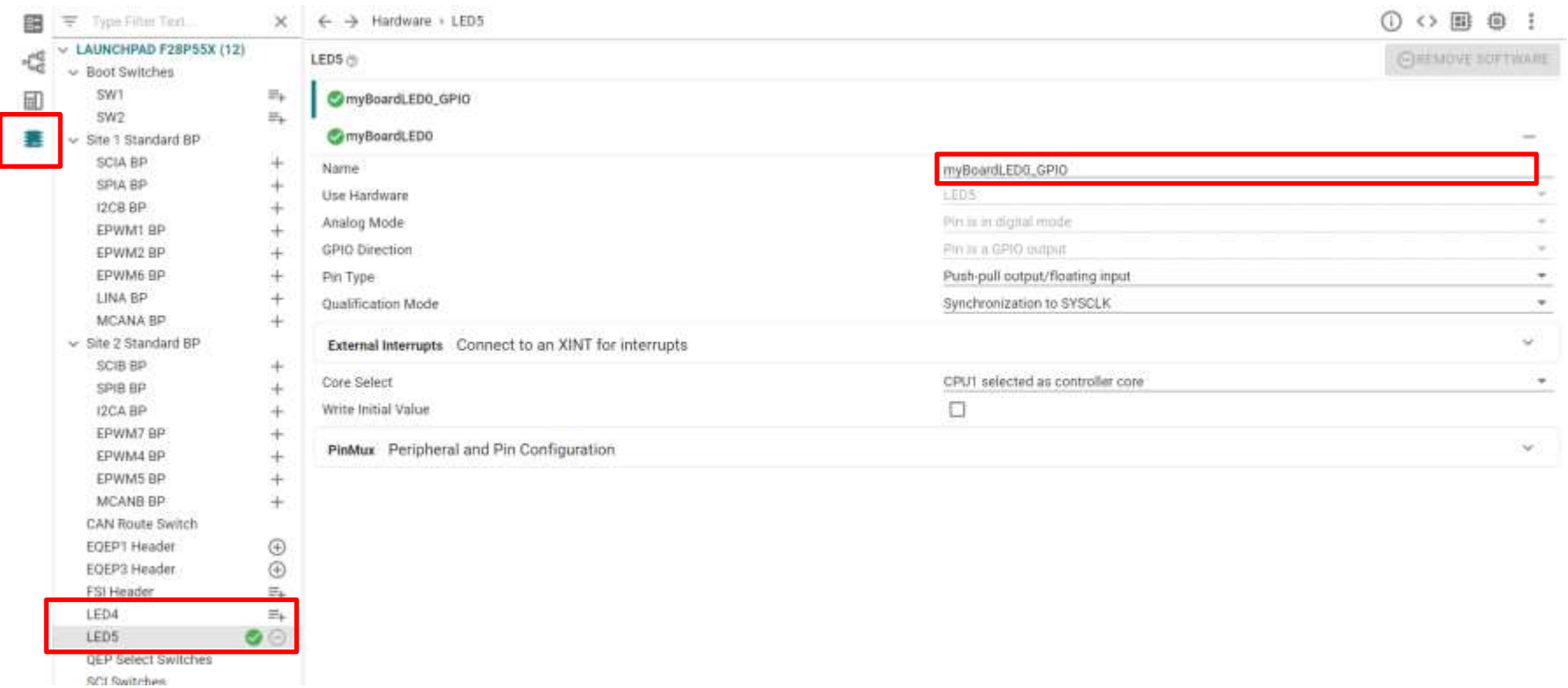
- Use CPU Timer 0
- Prescaler of 1 (SYSCLK)
- Timer Period: 20000000us
- Enable/generate interrupt when the timer has stopped counting

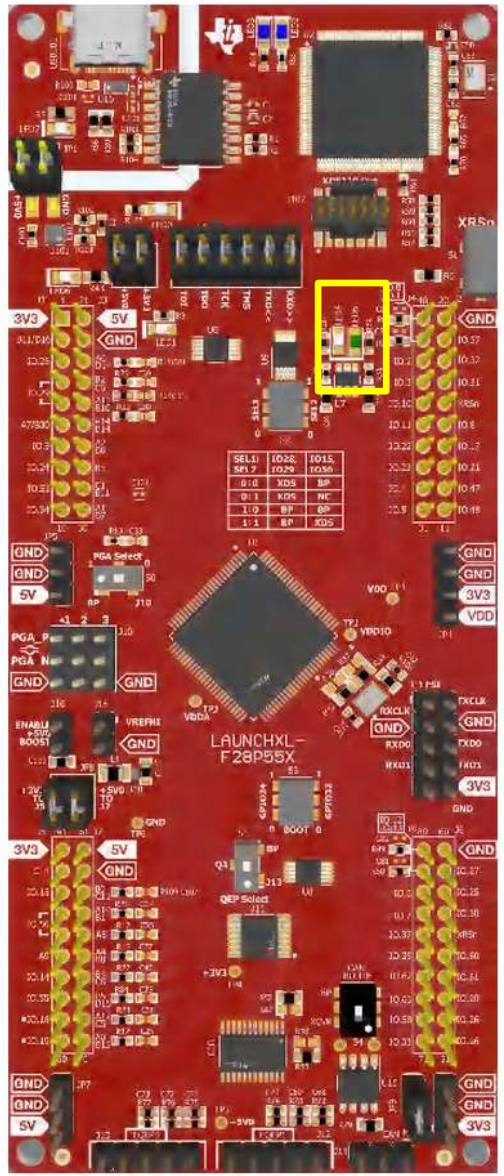
$$\text{timer period} = (\text{uint32\_t}) * ((\frac{\text{freq}}{1000000}) * \text{period})$$

Freq: 系统时钟，单位Hz
Period：所要设置的时间，单位us

系统时钟为150MHz，当timer period设置为20000000时，所计算出来的时间周期是133ms。

LED配置





应用代码

```
//
// Included Files
//
#include "driverlib.h"
#include "device.h"
#include "board.h"

//
// Globals
//
uint16_t cpuTimer0IntCount; //number of times TIMER 0 ISR is triggered
uint16_t delayCount;        //number (0-9) to scale the LED frequency

//
// ISR for CPUTIMER0 to change LED blink rate based on input to
delayCount
//
__interrupt void INT_myCPUTIMER0_ISR(void)
{
    cpuTimer0IntCount++;
    if (cpuTimer0IntCount >= delayCount){
        cpuTimer0IntCount = 0;
        GPIO_togglePin(myBoardLED0_GPIO);
    }

    //
    // Acknowledge this interrupt to receive more interrupts from
group 1
    //
    Interrupt_clearACKGroup(INT_myCPUTIMER0_INTERRUPT_ACK_GROUP);
}
```

```
//
// Function Prototypes
//
__interrupt void INT_myCPUTIMER0_ISR(void);
```

应用代码

```
for(;;)
        {
            msg = "\r\nEnter a number 0-9: \0";
            SCI_writeCharArray(mySCIA_BASE, (uint16_t*)msg, 24);

            //
            // Read a character from the FIFO.
            //
            receivedChar = SCI_readCharBlockingFIFO(mySCIA_BASE);


            //Turns character to digit
            delayCount = receivedChar - '0';
            if(delayCount >= 9) delayCount = 9;

            rxStatus = SCI_getRxStatus(mySCIA_BASE);
            if((rxStatus & SCI_RXSTATUS_ERROR) != 0)
            {
                //
                //If Execution stops here there is some error
                //Analyze SCI_getRxStatus() API return value
                //
                ESTOP0;
            }

            //
            // Echo back the character.
            //
            msg = "\r\nLED set to blink rate \0";
            SCI_writeCharArray(mySCIA_BASE, (uint16_t*)msg, 25);
            SCI_writeCharBlockingNonFIFO(mySCIA_BASE, receivedChar);
        }
}
```