

# LED Demo

You will learn how to use GPIO as the output by this demo. You can write code to control the eight LEDs LED1 (D1) ~ LED8 (D8) on the board to blink one by one.

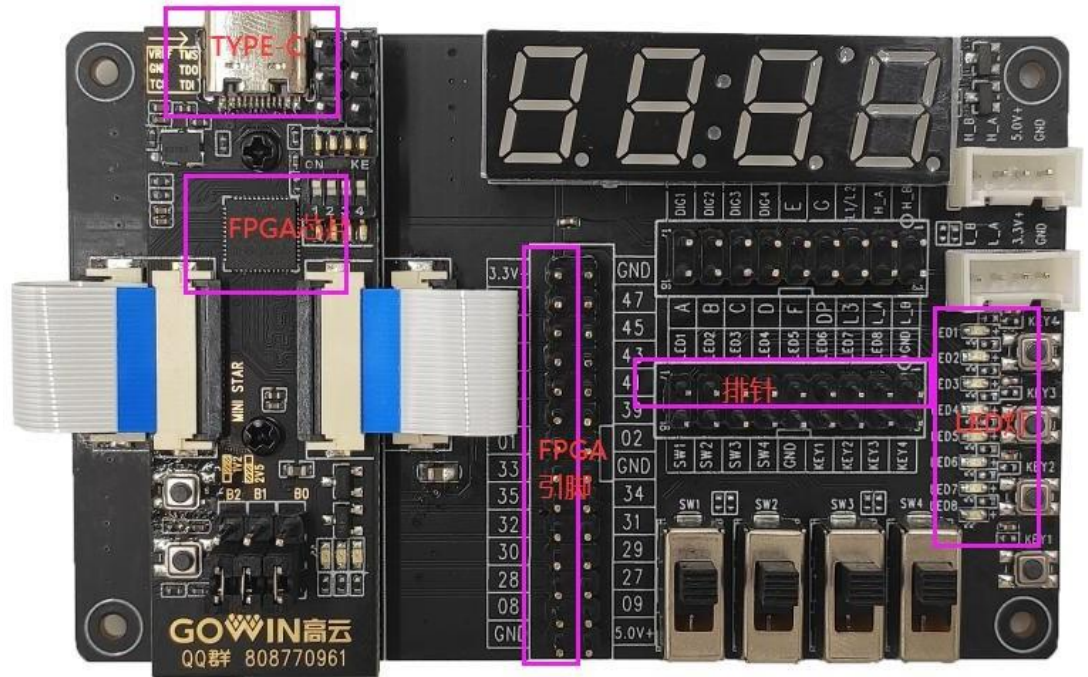


Figure 1 Mini-Star Development Board (GW1NSR-LV4CQN48P) and Extension

This demo introduction includes four parts:

1. GPIO Introduction
2. Hardware Design
3. Software Design
4. Download and Verification

## GPIO Introduction

Gowin GW1NSR-LV4CQN48P is used as the platform in this demo. From [DS861, GW1NSR series of FPGA Products Datasheet](#), you can learn the GPIO module and know there is a Cortex-M3 RISC embedded in this chip. The key to this demo is how to control the GPIO as the output. The design information listed below is excerpted from the datasheet.

The SoC microprocessor system communicates with the GPIO block through the AHB bus. The GPIO block interconnects with the FPGA. GPIO provides a 16 bits I/O interface with the following properties:

- Programmable interrupt generation capability. You can configure each bit of the I/O pins to generate interrupts;
- Bit masking support using address values;
- Registers for alternate function switching with pin multiplexing support;
- Thread safe operation by providing separate set and clear addresses for control registers.

The GPIO register is as shown in Table 3-21. The GPIO base address is 0x40010000.

The following table lists GPIO registers.

Table 3-21 GPIO Register

Name	Base Offset	Type	Data Width	Reset Value	Description
DATA	0x0000	Read/Write	16	0x-----	Data value [15:0]
DATAOUT	0x0004	Read/Write	16	0x0000	Data output register value [15:0]
OUTENSET	0x0010	Read/Write	16	0x0000	Output enable set [15:0] Write 1: Set the output enable bit. Write 0: No effect. Read 1: Indicates the signal direction as output. Read 0: Indicates the signal direction as input.
OUTENCLR	0x0014	Read/Write	16	0x0000	Output enable clear [15:0]
ALTFUNCSET	0x0018	Read/Write	16	0x0000	Alternative function set [15:0] Write 1: Sets the ALTFUNC bit. Write 0: No effect. Read 0: GPIO as I/O. Read 1: ALTFUNC Function
ALTFUNCCLR	0x001C	Read/Write	16	0x0000	Alternative function clear [15:0]
INTENSET	0x0020	Read/Write	16	0x0000	Interrupt enable set [15:0] Write 1: Sets the enable bit. Write 0: No effect. Read 0: Interrupt disabled. Read 1: Interrupt enabled.
INTENCLR	0x0024	Read/Write	16	0x0000	Interrupt enable clear [15:0] Write 1: Clear the enable bit. Write 0: No effect. Read 0: Interrupt disabled. Read 1: Interrupt enabled.
INTTYPESET	0x0028	Read/Write	16	0x0000	Interrupt type set [15:0]
INTTYPECLR	0x002C	Read/Write	16	0x0000	Interrupt type clear [15:0]
INTPOLSET	0x0030	Read/Write	16	0x0000	Polarity-level, edge interrupt request configuration [15:0]
INTPOLCLR	0x0034	Read/Write	16	0x0000	Polarity-level, edge interrupt request configuration [15:0]
INTSTATUS/ INTCLEAR	0x0038	Read/Write	16	0x0000	Read interrupt status register. Write 1: Clear the interrupt request

**Note !**

For the details, you can see 3.11.10 GPIO in [DS861, GW1NSR series of FPGA Products Datasheet](#).

# Hardware Design

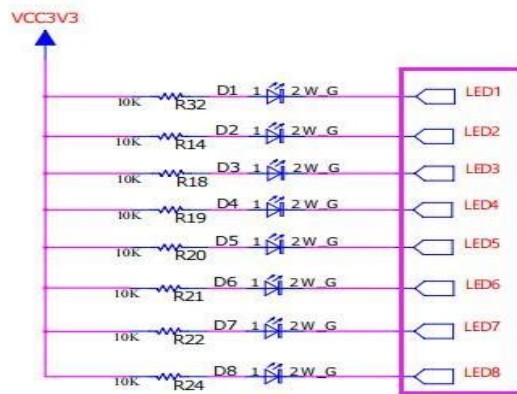


Figure 2 LED Schematic

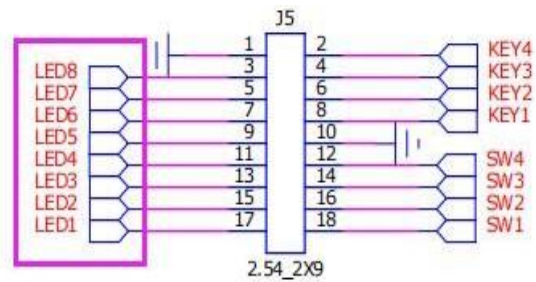


Figure 3 LED Signal Pins

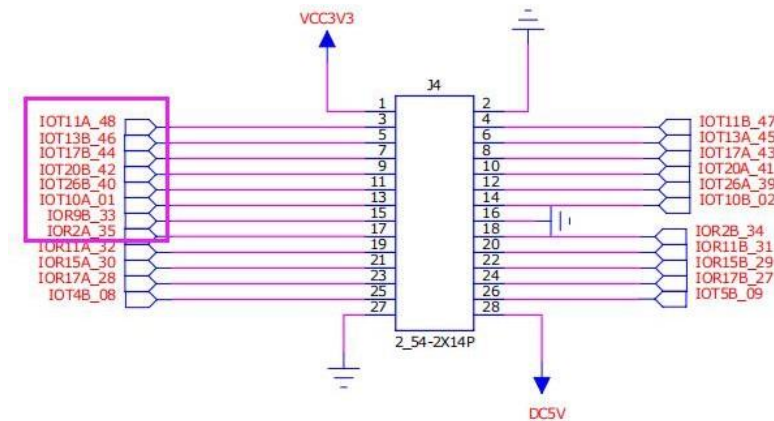


Figure 4 FPGA Pins

In Figure 2, one end of the LED is connected to the power supply via a current-limiting resistor, and the other end is connected to the pin in Figure 3 (low level, on; high level, off). Use DuPont wire to connect the LED1~LED8 signals to the FPGA pins such as pins 48~35 in Figure 4 (you can also choose other pins). The connection is as shown in Figure 5.

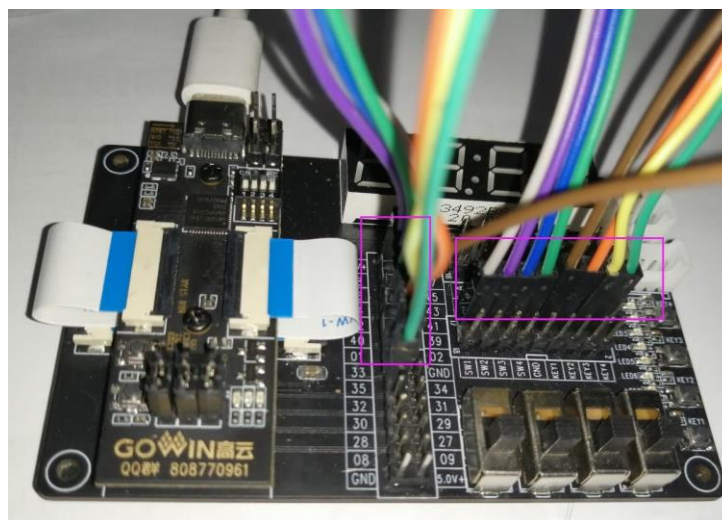


Figure 5 Connection of LED and FPGA

# Software Design

The software design includes two parts: FPGA internal hardware logic and Cortex-M3 software control code, and you can refer to [IPUG930, Gowin EMPU \(GW1NS-4C\) Quick Design Reference Manual](#).

## FPGA Internal Logic Design

Software: Gowin\_V1.9.7.02Beta and above. The reference template is the fpga\_led folder as shown in Figure 6.

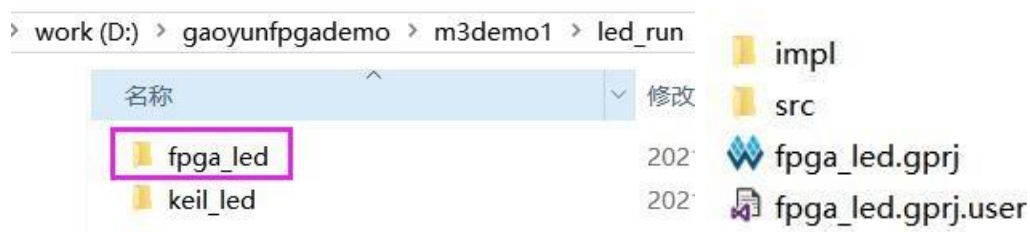


Figure 6 FPGA Example Folder

Use Gowin Software to open fpga\_led.gprj as shown in Figure 7.

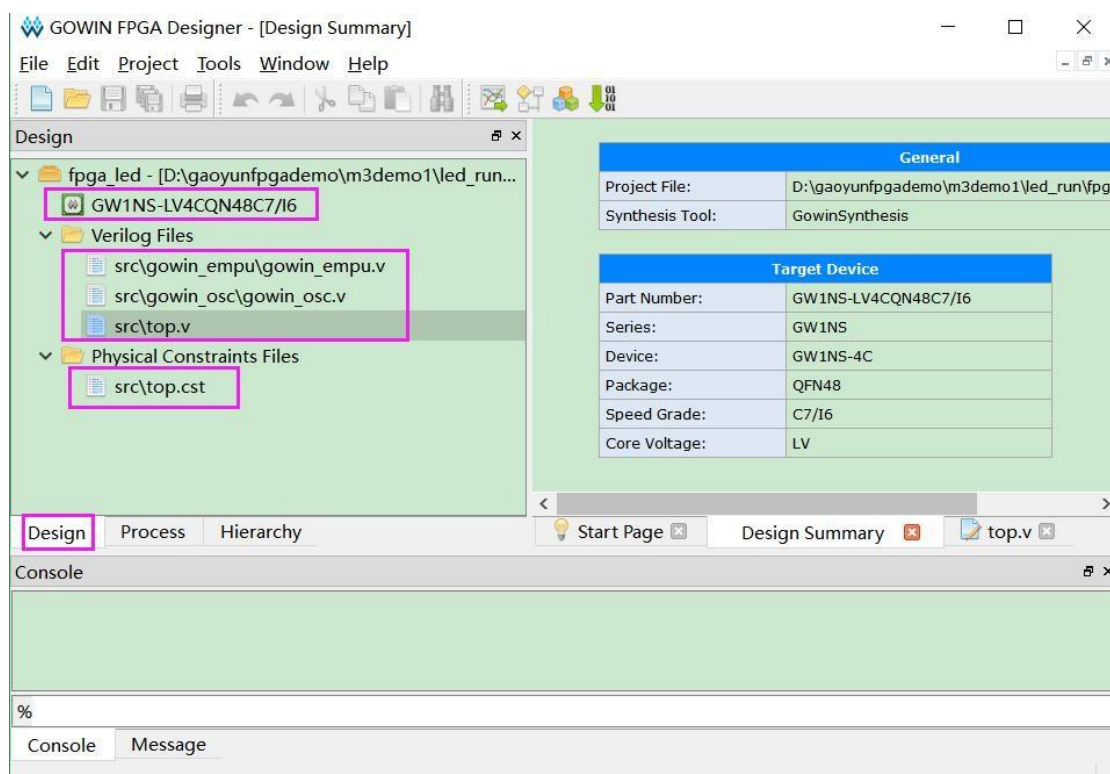


Figure 7 Project View

Click top.v and the interface is as shown in Figure 8; the embedded Cortex-M3 core provides a 16-bit GPIO interface.



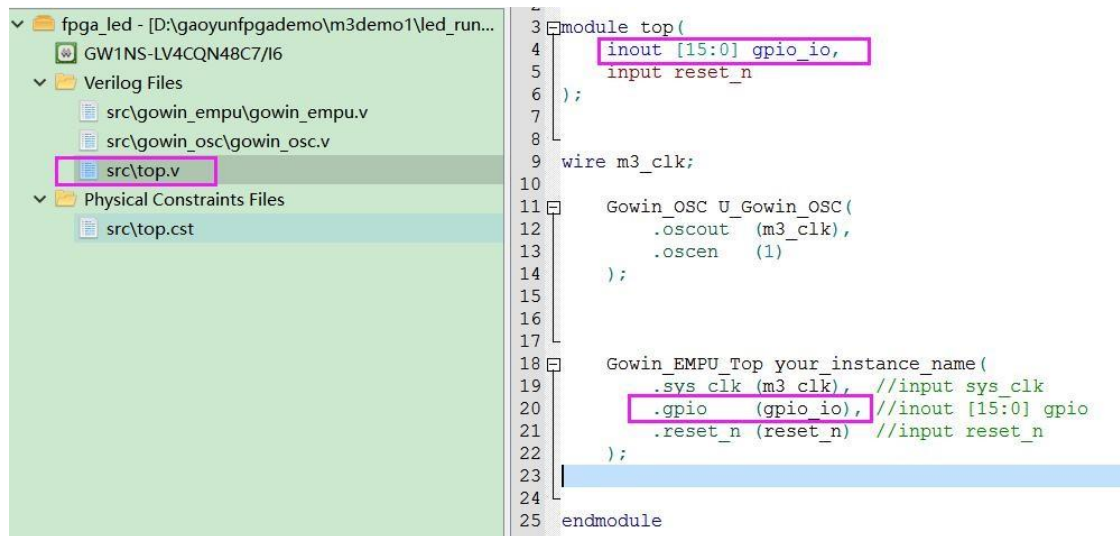


Figure 8 FPGA Top Code

Click "Process" in Figure 9 and then select "FloorPlanner" to open I/O constraints as shown in Figure 10.

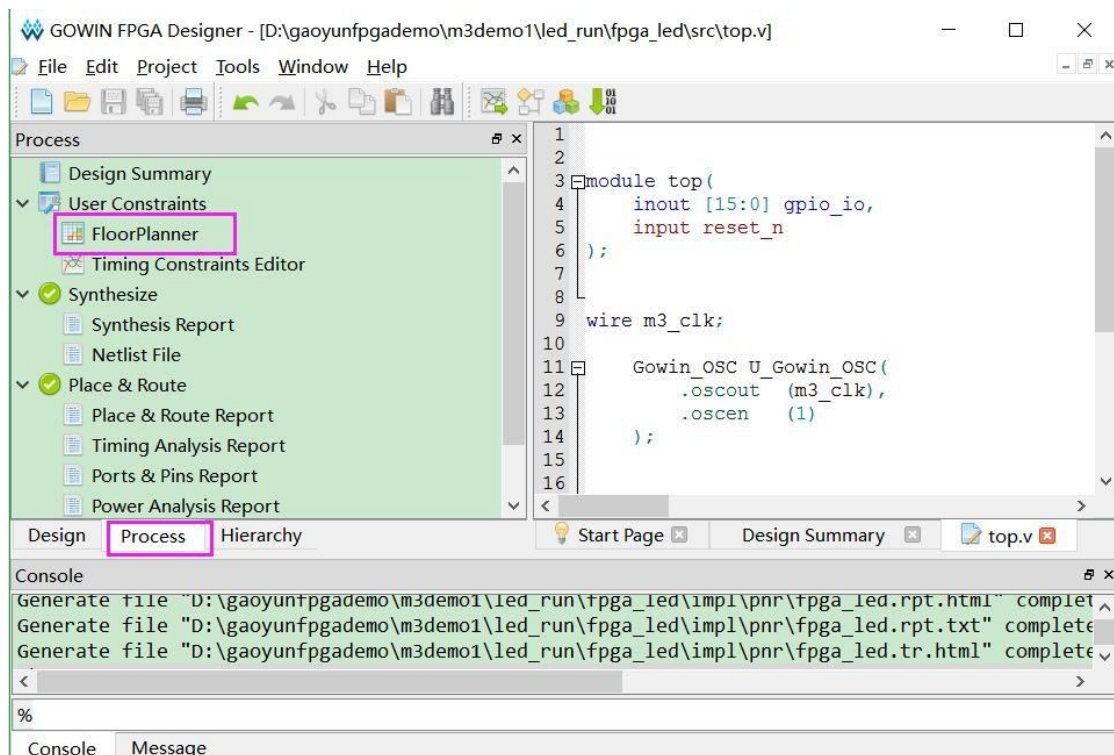


Figure 9 Process View

Click steps 1-5 in turn on the interface in Figure 10 to set the FPGA pin number and power supply voltage of LED according to the connection in Figure 3 and Figure 4, and save the settings when finished.

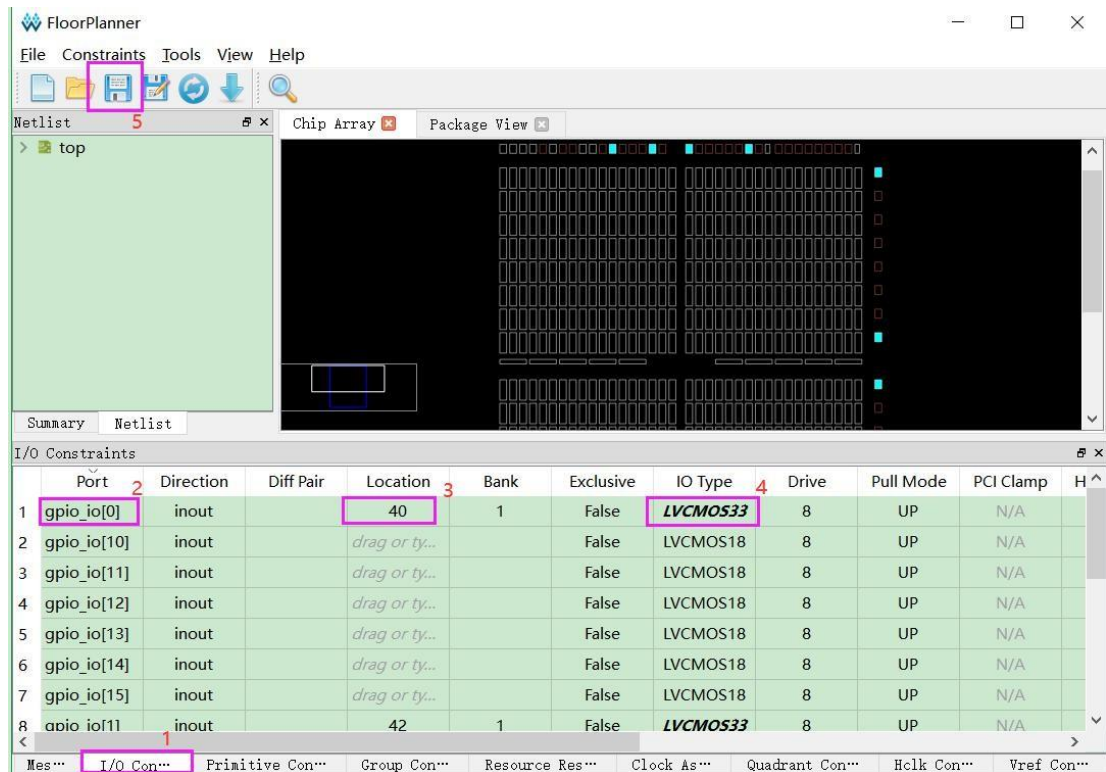


Figure 10 I/O Constraints

Then click "Synthesize" and "Place & Route" in Figure 11 to generate the logic file.

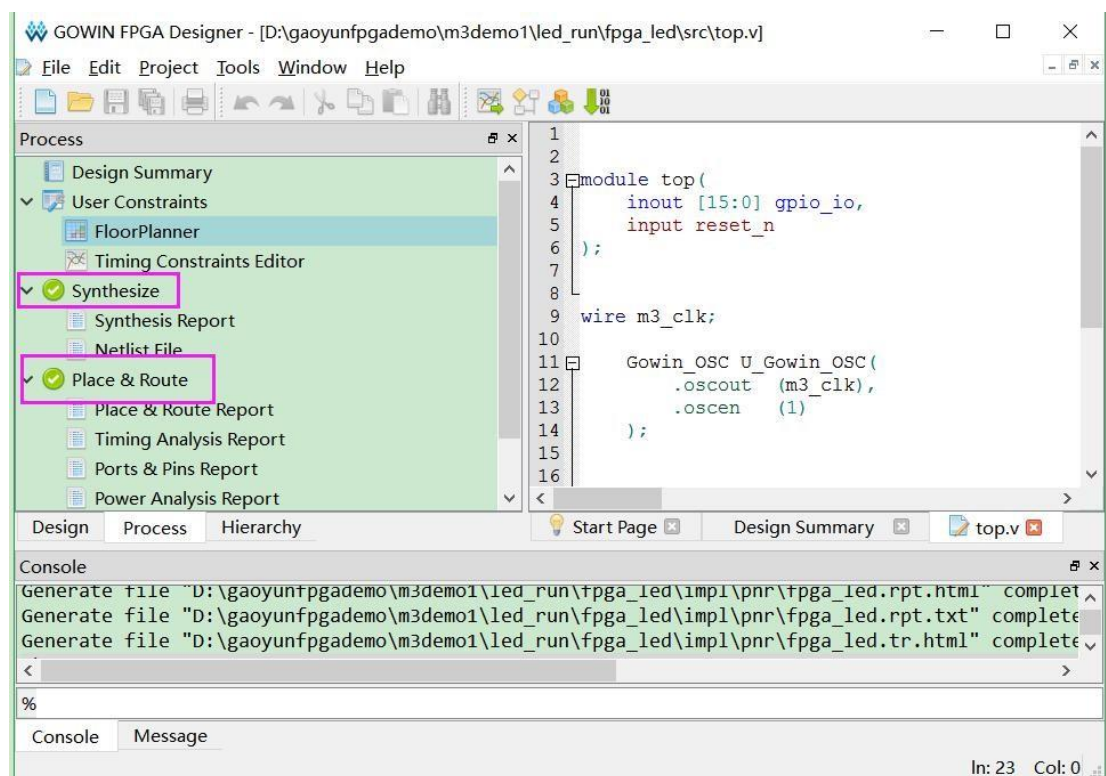


Figure 11 Synthesize and Place & Route

rk (D:) > gaoyunfpgademo > m3demo1 > led\_run > fpga\_led > impl > pnr

名称	修改日期	类型	大小
cmd.do	2021/5/13 星期...	DO 文件	1 KB
device.cfg	2021/5/13 星期...	CFG 文件	1 KB
fpga_led.bin	2021/5/13 星期...	BIN 文件	217 KB
fpga_led.binx	2021/5/13 星期...	BINX 文件	217 KB
fpga_led.db	2021/5/13 星期...	Data Base File	5 KB
fpga_led.fs	2021/5/13 星期...	Visual F# Source...	1,732 KB
fpga_led.log	2021/5/13 星期...	文本文档	2 KB
fpga_led.pin.html	2021/5/13 星期...	HTML 文件	18 KB
fpga_led.power.html	2021/5/13 星期...	HTML 文件	9 KB
fpga_led.rpt.html	2021/5/13 星期...	HTML 文件	23 KB
fpga_led.rpt.txt	2021/5/13 星期...	文本文档	16 KB
fpga_led.timing_paths	2021/5/13 星期...	TIMING_PATHS ...	19 KB
fpga_led.tr.html	2021/5/13 星期...	HTML 文件	1 KB
fpga_led_tr_cata.html	2021/5/13 星期...	HTML 文件	8 KB
fpga_led_tr_content.html	2021/5/13 星期...	HTML 文件	483 KB

Figure 12 Logic File Directory

## Cortex-M3 Software Control Design

Software: ARM Keil MDK V5.26 and above. The reference template is the Keil\_led folder as shown in Figure 13.

rk (D:) > gaoyunfpgademo > m3demo1 > led\_run >

名称	修改日期
fpga_led	2021/5/13
keil_led	2021/5/13

- CORE
- PERIPHERAL
- PROJECT
- STARTUP
- SYSTEM
- USER

Figure 13 Keil Project Folder

Open led.uvprojx in the PROJECT folder as shown in Figure 14.

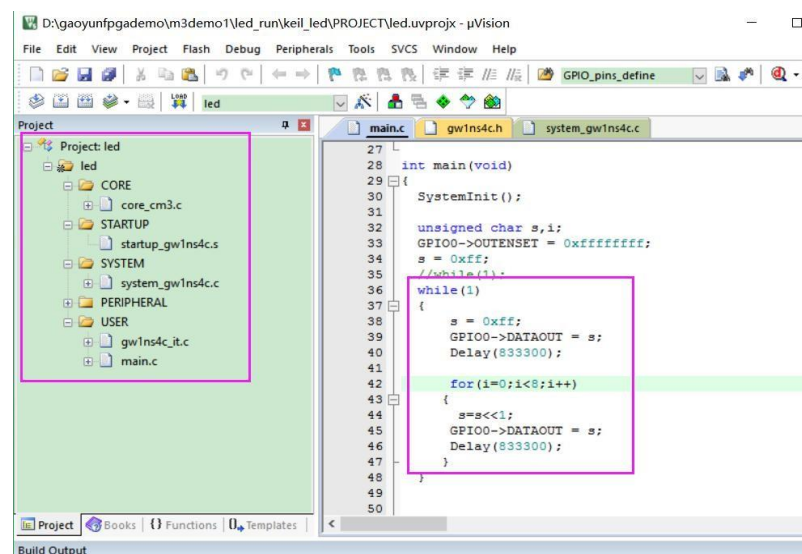


Figure 14 Keil Example Project

Set GPIO-> OUTENSET register to realize the output function.

```
GPIO0->OUTENSET = 0xffffffff;
```

Set GPIO-> DATAOUT register to implement LED blinking one by one.

```
while(1)
{
    s = 0xff;
    GPIO0->DATAOUT = s;
    Delay(833300);

    for(i=0;i<8;i++)
    {
        s=s<<1;
        GPIO0->DATAOUT = s;
        Delay(833300);
    }
}
```

After build, the download file led.bin is generated as shown in Figure 15.

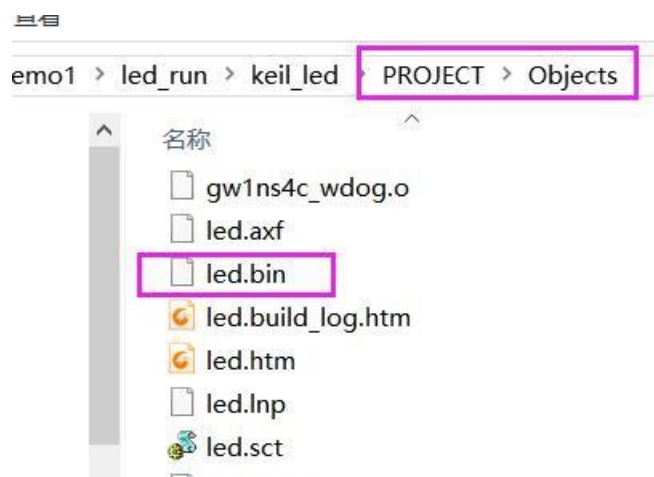


Figure 15 led.bin Directory

## Download and Verification

Use Gowin Software to download as shown in Figure 16. The FPGA hardware platform file is fpga\_led.fs, and the Cortex-M3 software file is led.bin, so be careful to choose the correct file path and build file.



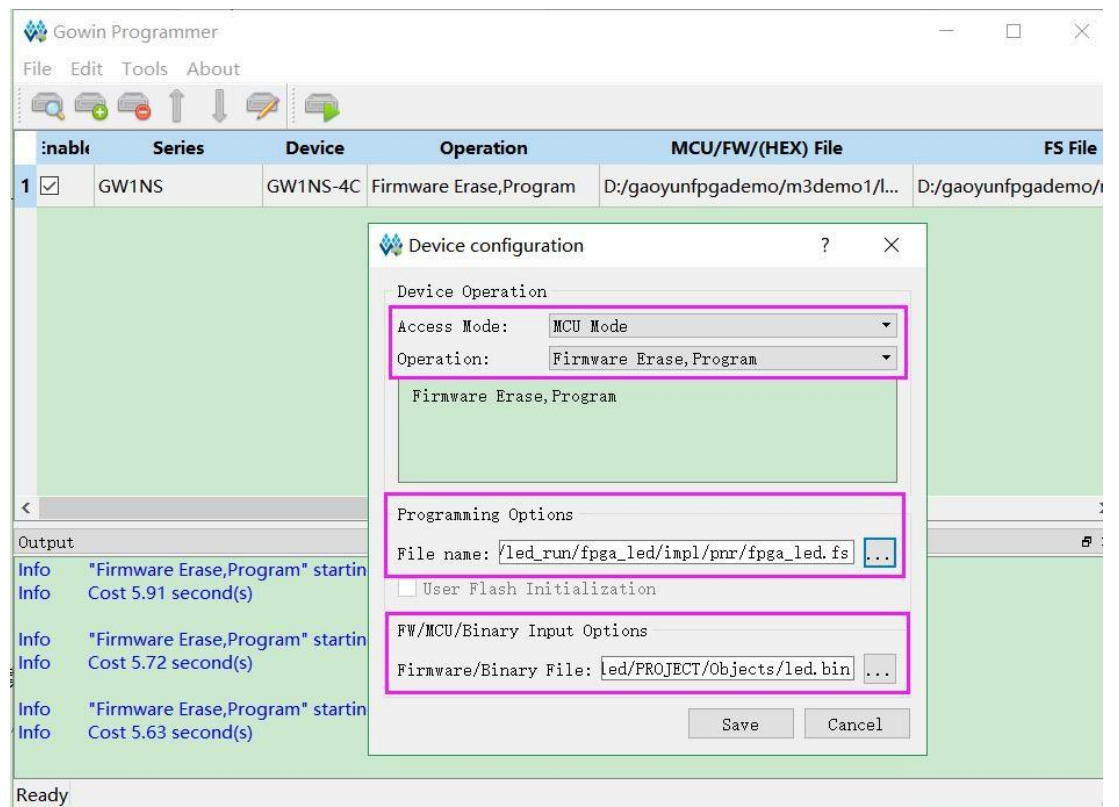


Figure 16 Download Interface

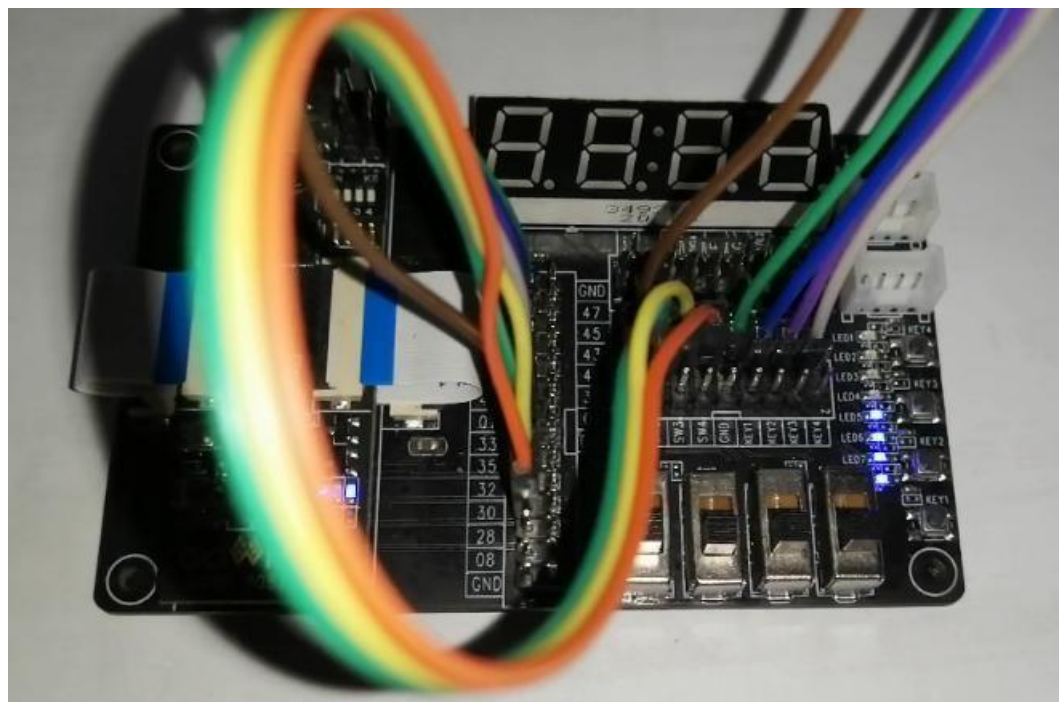


Figure 17 Demo Running