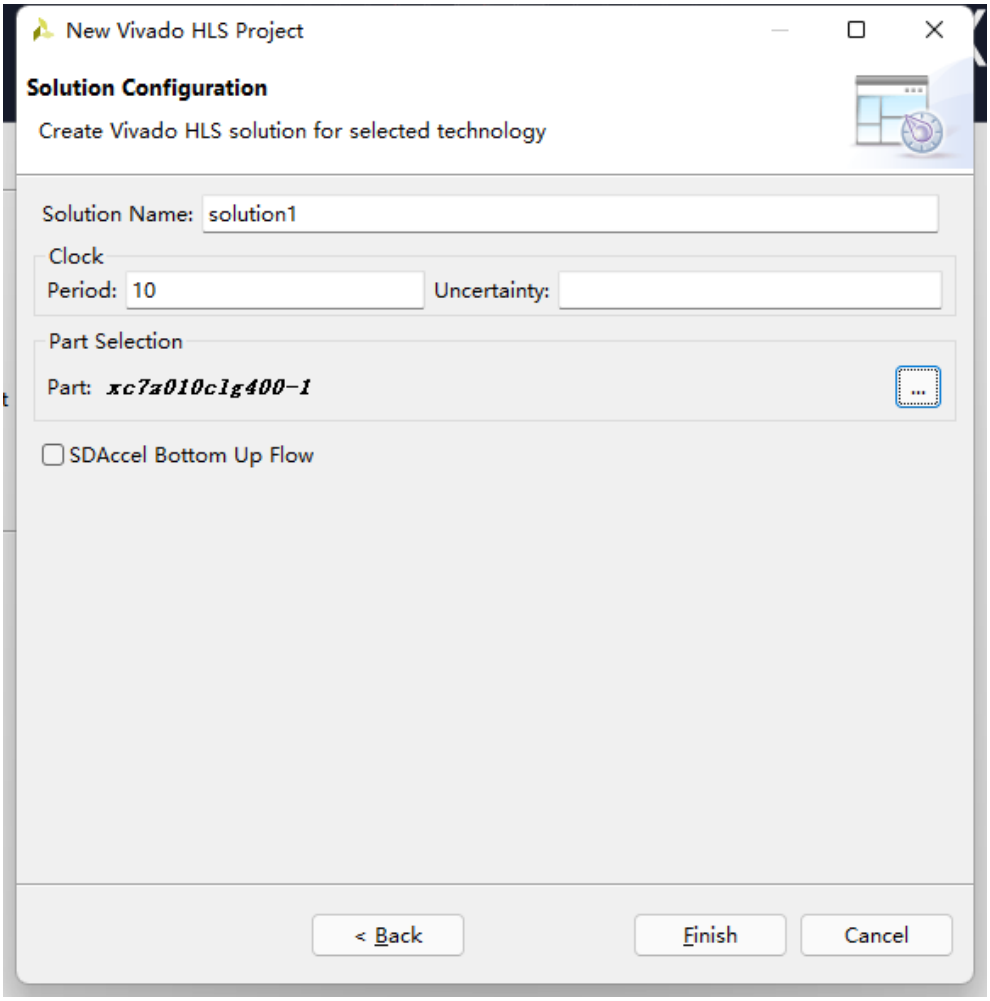
[L12]HLS基础学习

学习本节推荐配合UG871和UG902,HLS目前依然很多BUG,不过比Vitis还是好很多,所以做一个了解其实就还好了,一开始Xilinx总是说HLS替代传统Verilog,Vits替代HLS各种,这就如ARM能替代51一样,短时间应该还是比较难.

新建工程并记得选择正确的器件.



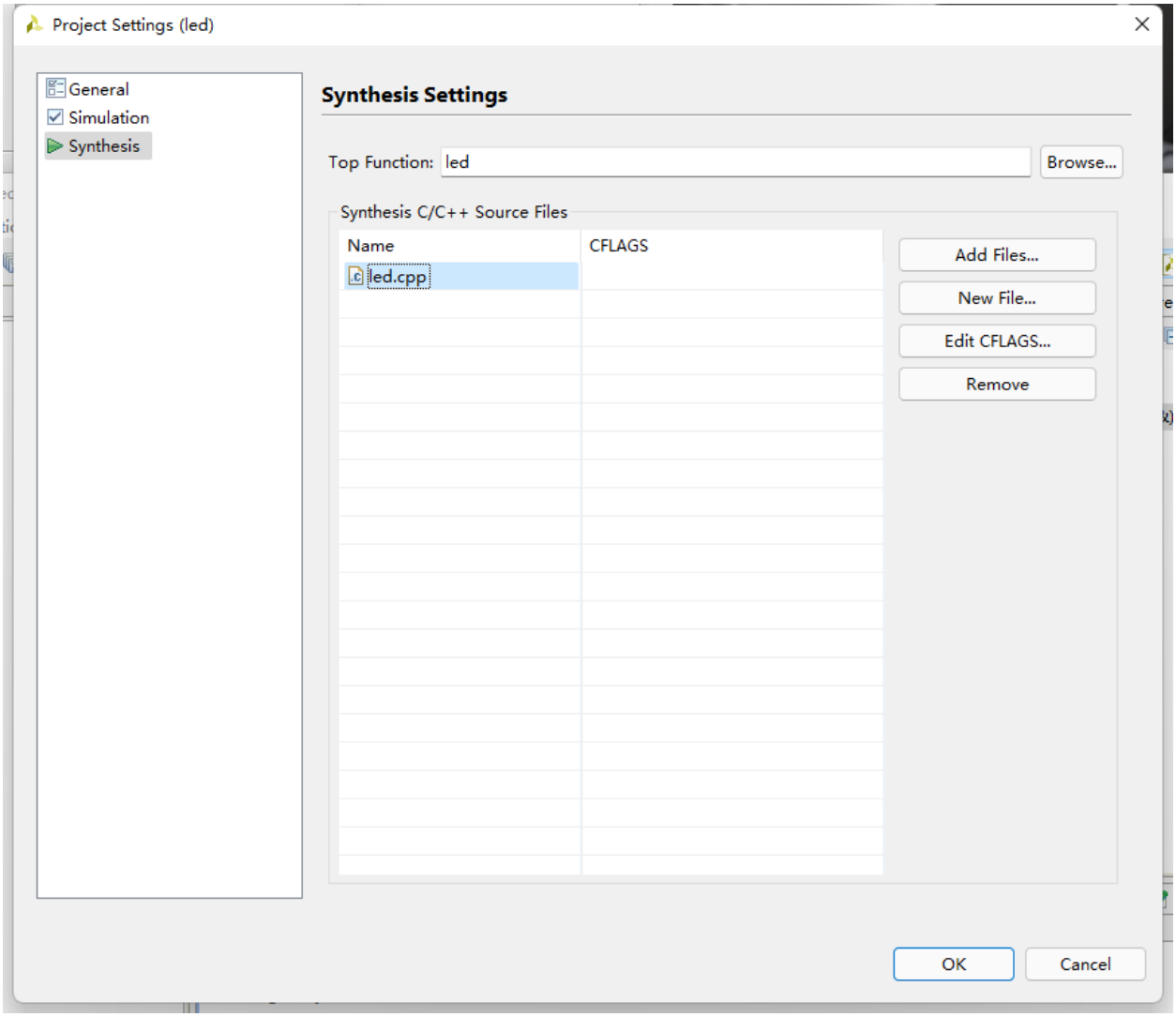
我写了一个led.cpp文件,作为入口,代码如下.

注:为什么不用switch,因为我们还是为了综合结果考虑.

```
#include <stdio.h>
#include <ap_int.h>

void led(ap_int<4> &led){
  #pragma HLS interface ap_none port=led
  #pragma HLS interface ap_ctrl_none port=return

  static int ledc = 0x01;
  static long cnt = 0;

  if(cnt < 50000000){
    cnt++;
  }
  else{
    cnt = 0;

    if(ledc == 0x01){
      led = 0x02;
      ledc = 0x02;
    }else if(ledc == 0x02){
```

```
      led = 0x04;
      ledc = 0x04;
    }else if(ledc == 0x04){
      led = 0x08;
      ledc = 0x08;
    }else if(ledc == 0x08){
      led = 0x01;
      ledc = 0x01;
    }
  }
}
```

否则会有这个错误.



然后编译(综合)看看.

led.cpp | Synthesis(solution1)

Synthesis Report for 'led'

General Information

Date: Mon Feb 28 15:00:05 2022
Version: 2018.3 (Build 2405991 on Thu Dec 06 23:56:15 MST 2018)
Project: led
Solution: solution1
Product family: zynq
Target device: xc7z010clg400-1

Performance Estimates

Timing (ns)

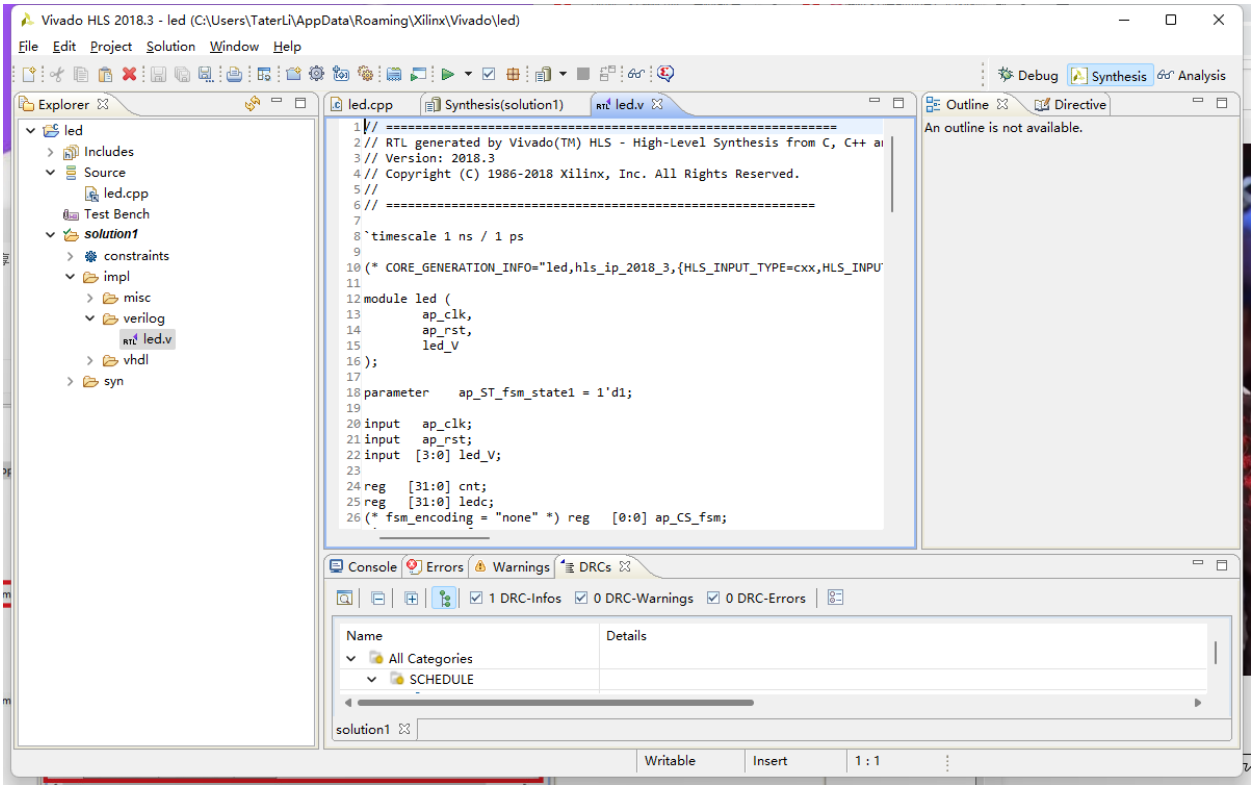
Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	4.321	1.25

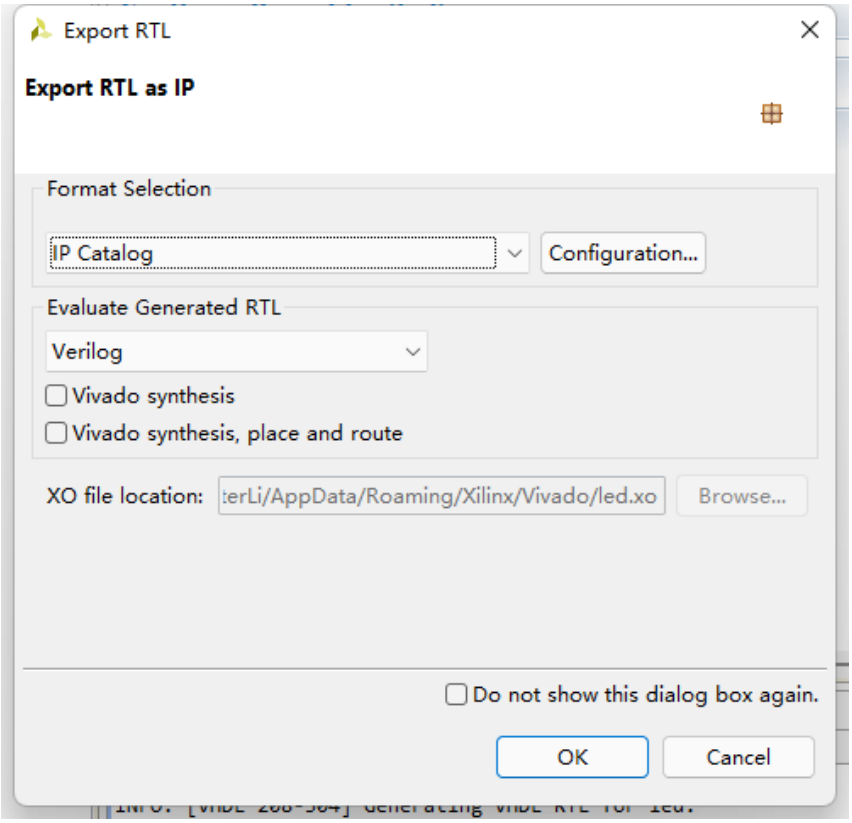
Latency (clock cycles)

Summary

也可以看到综合结果代码.

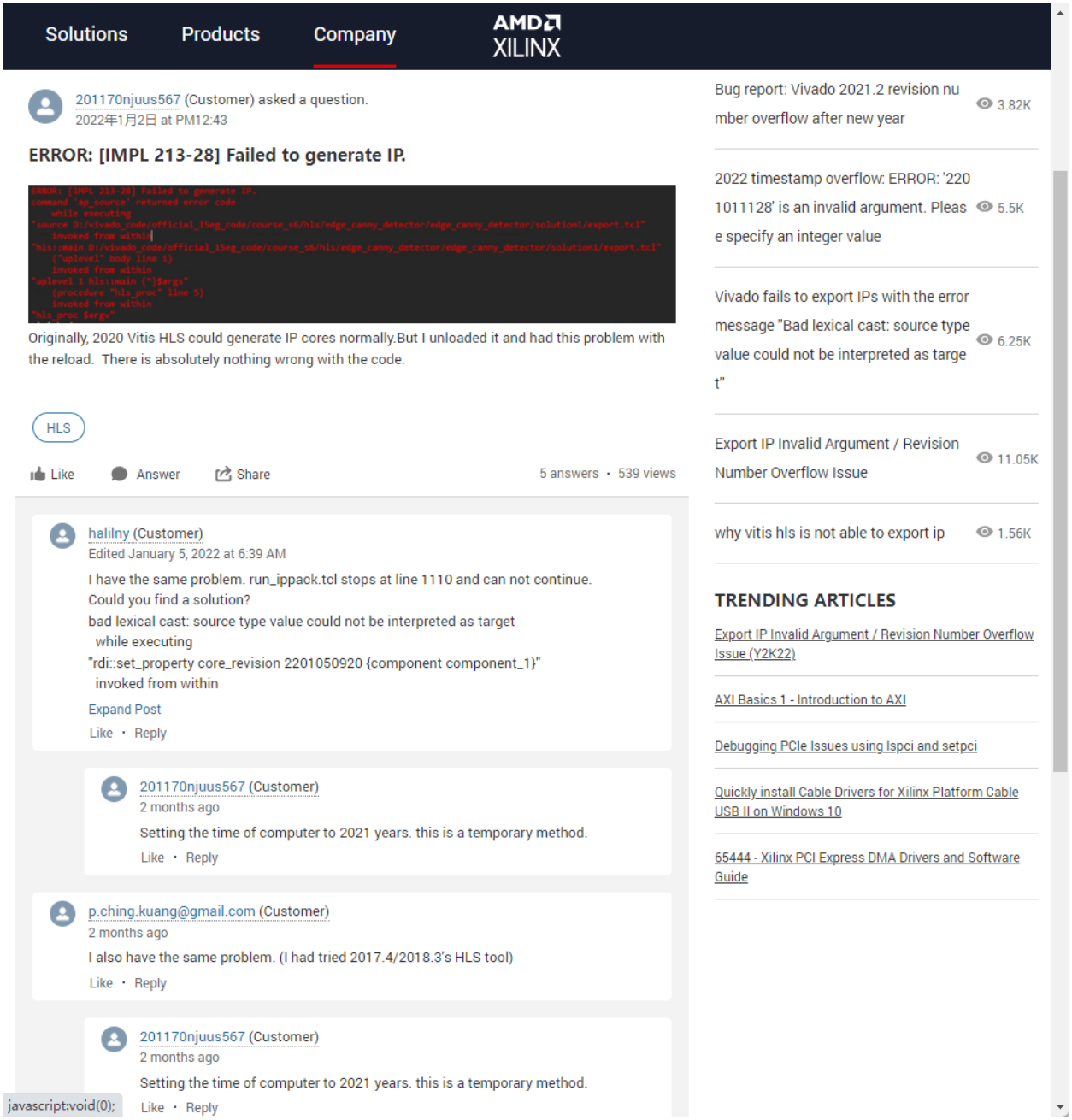


点击Export RTL as IP按钮(橙色捆包图标),打开导出.

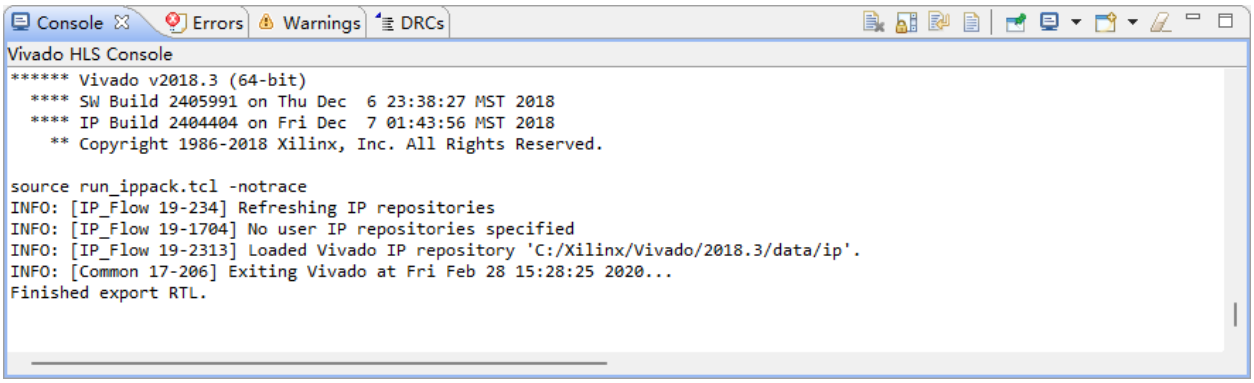


不过喜闻乐见失败,这里需要修改电脑时间,请查看官方论坛的吐槽:

https://support.xilinx.com/s/question/0D52E00006uyTmwSAE/error-impl-21328-failed-to-generate-ip?language=en_US



终于可以导出了,就这个BUG,我就可以认为毫无生产力?每次调试还得换时间,很多软件又依赖时间.



可以从代码中看到,语法稍微有点不同,

```
#pragma HLS interface ap_none port=led
#pragma HLS interface ap_ctrl_none port=return
```

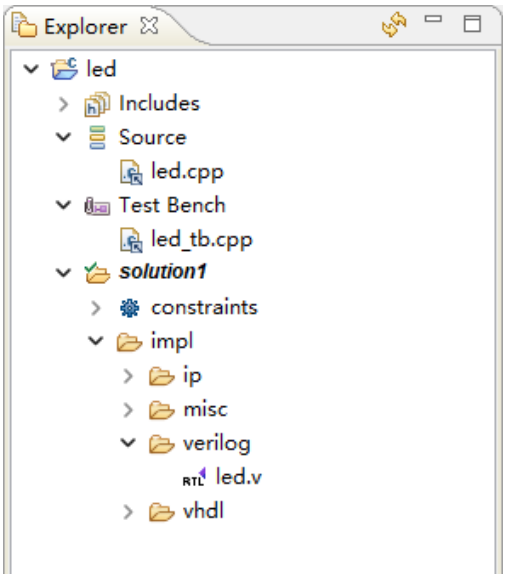
这些是综合行为指示,比如第一句,告诉综合器LED是输出信号.第二句告诉综合器这没有端口控制信号,比如下面这句是说返回值无,输出ap_int类型,这个是什么类型呢?就是HLS类型,指定4位宽的LED,就是4只LED嘛.

```
void led(ap_int<4> &led)
```

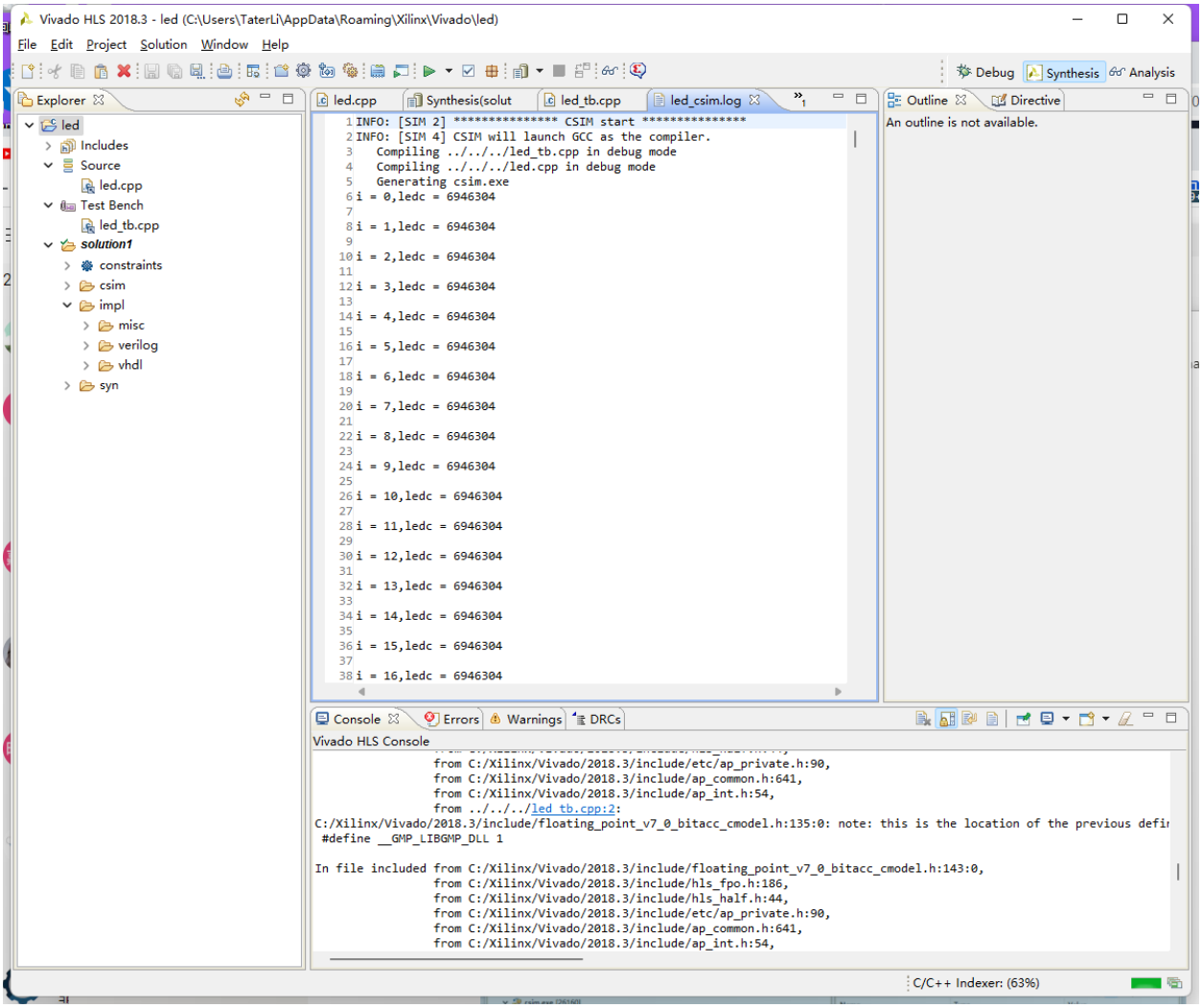
那能不能仿真呢?当然可以,不过也要自己编写仿真激励,风格就非常C语言了,文件名要以一定格式,比如led.cpp对应led_tb.cpp才能正确识别,当然位置也要放对.

```
#include "stdio.h"
#include "ap_int.h"

extern void led(ap_int<4> &led);

int main(void)
{
  ap_int<4> ledc=7;
  for(int i=0;i<1000;i++)
  {
    led(ledc);
    printf("i = %d,ledc = %d \r\n",i,ledc & 0xf);
  }
  return 0;
}
```

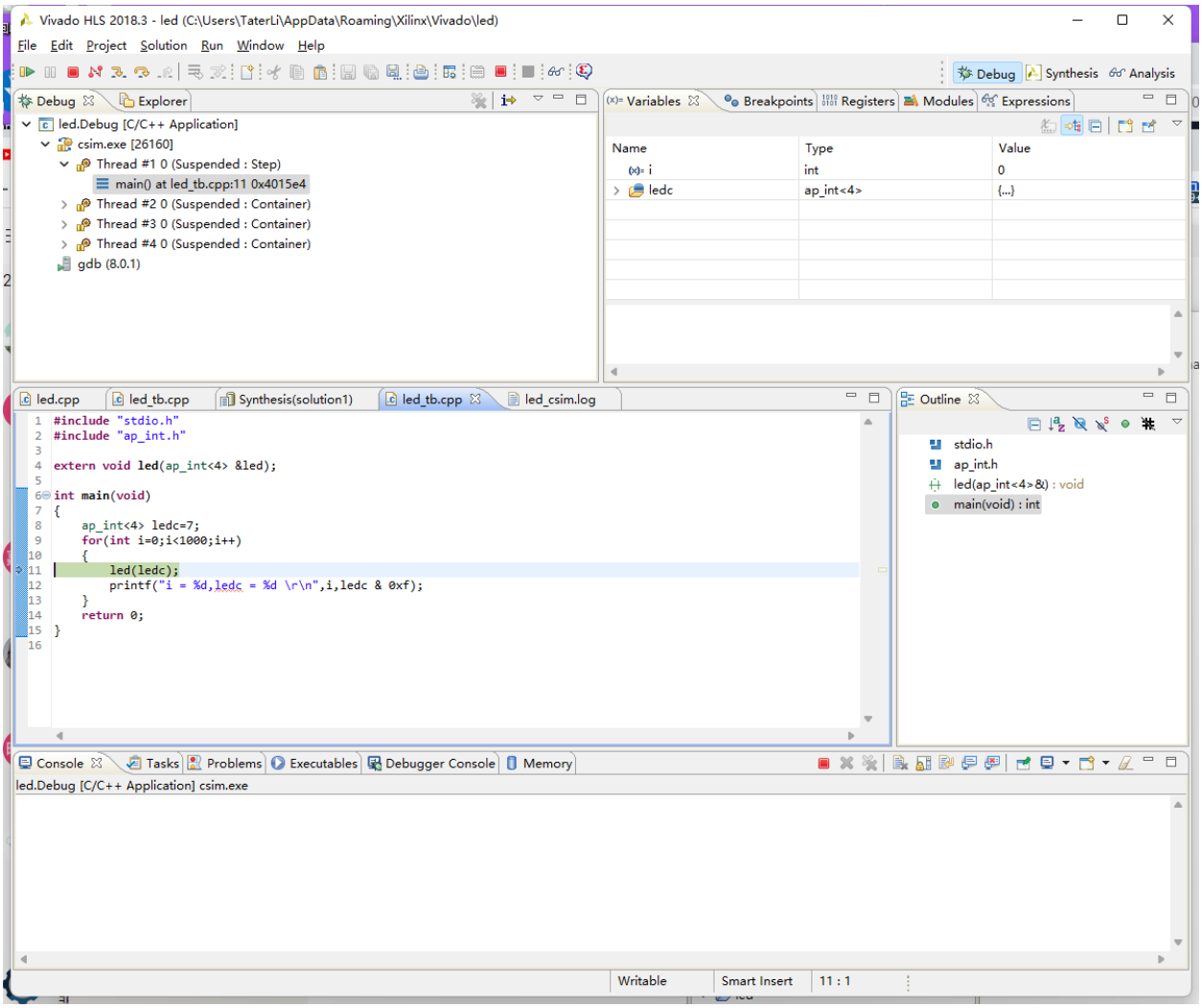
工程整体.



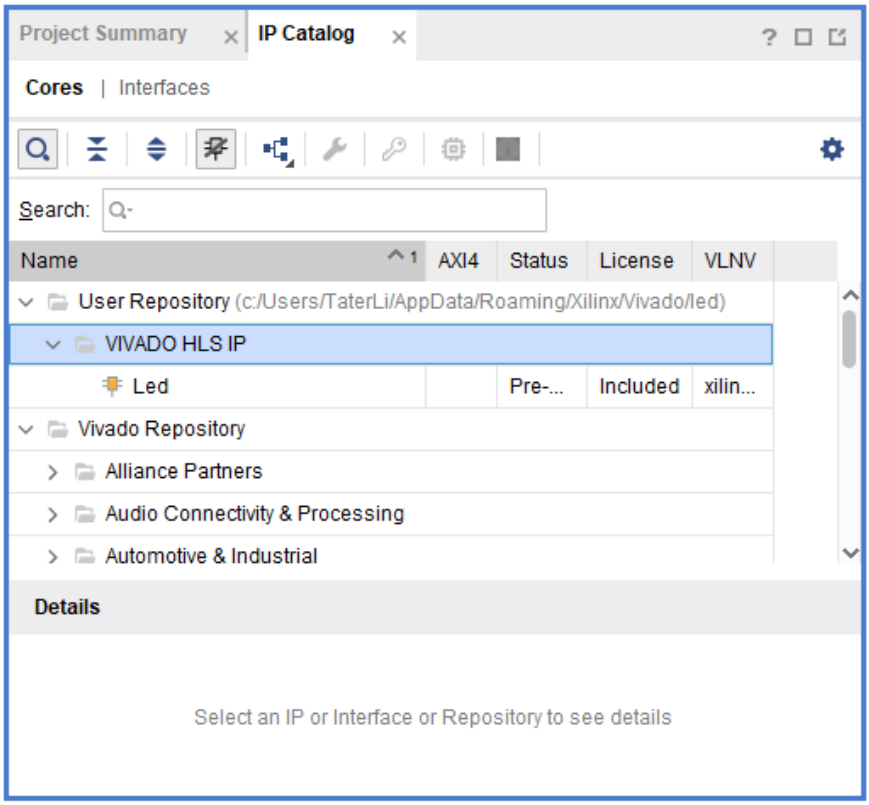
直接执行仿真会输出所有结果.



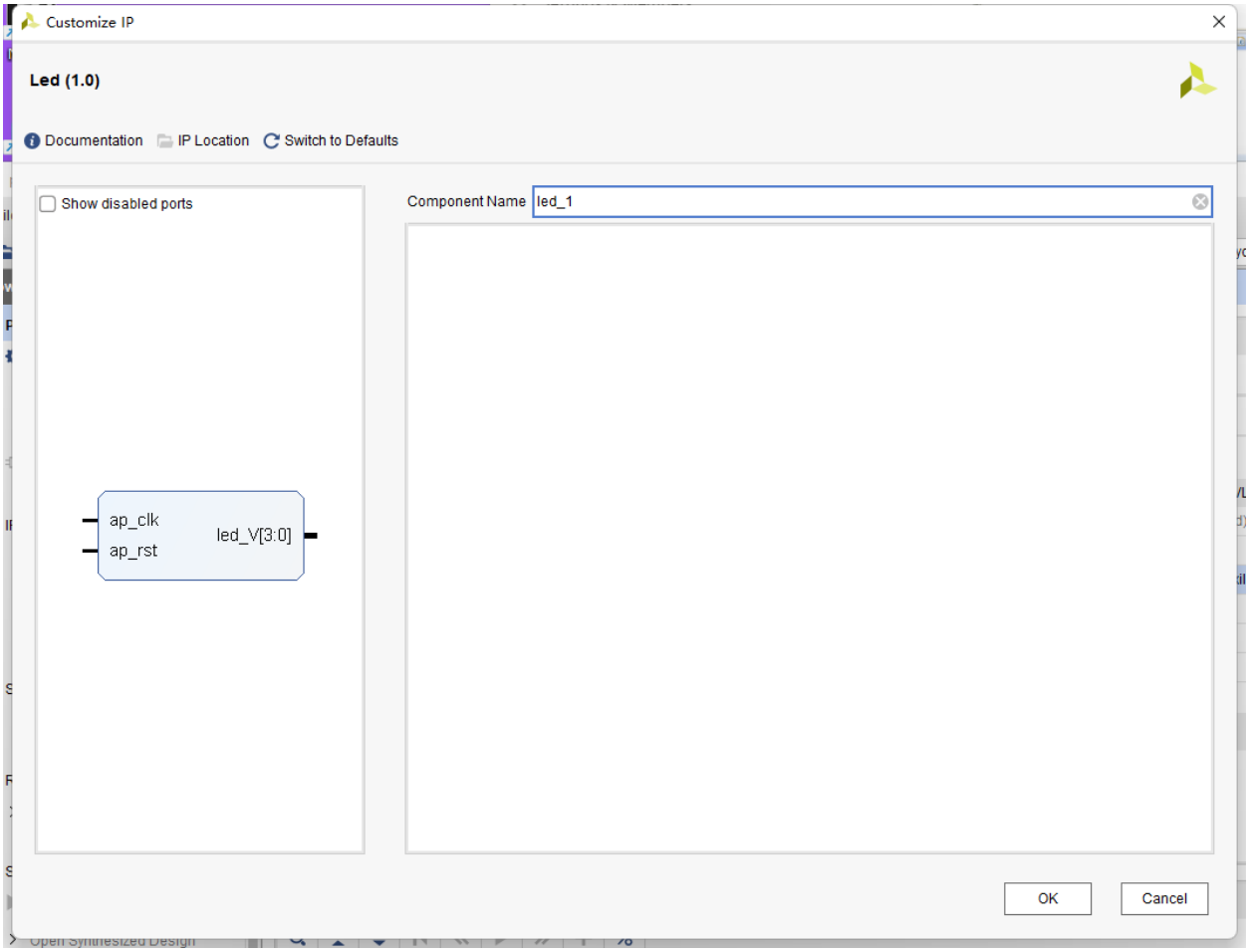
也可以尝试进入单步调试器.



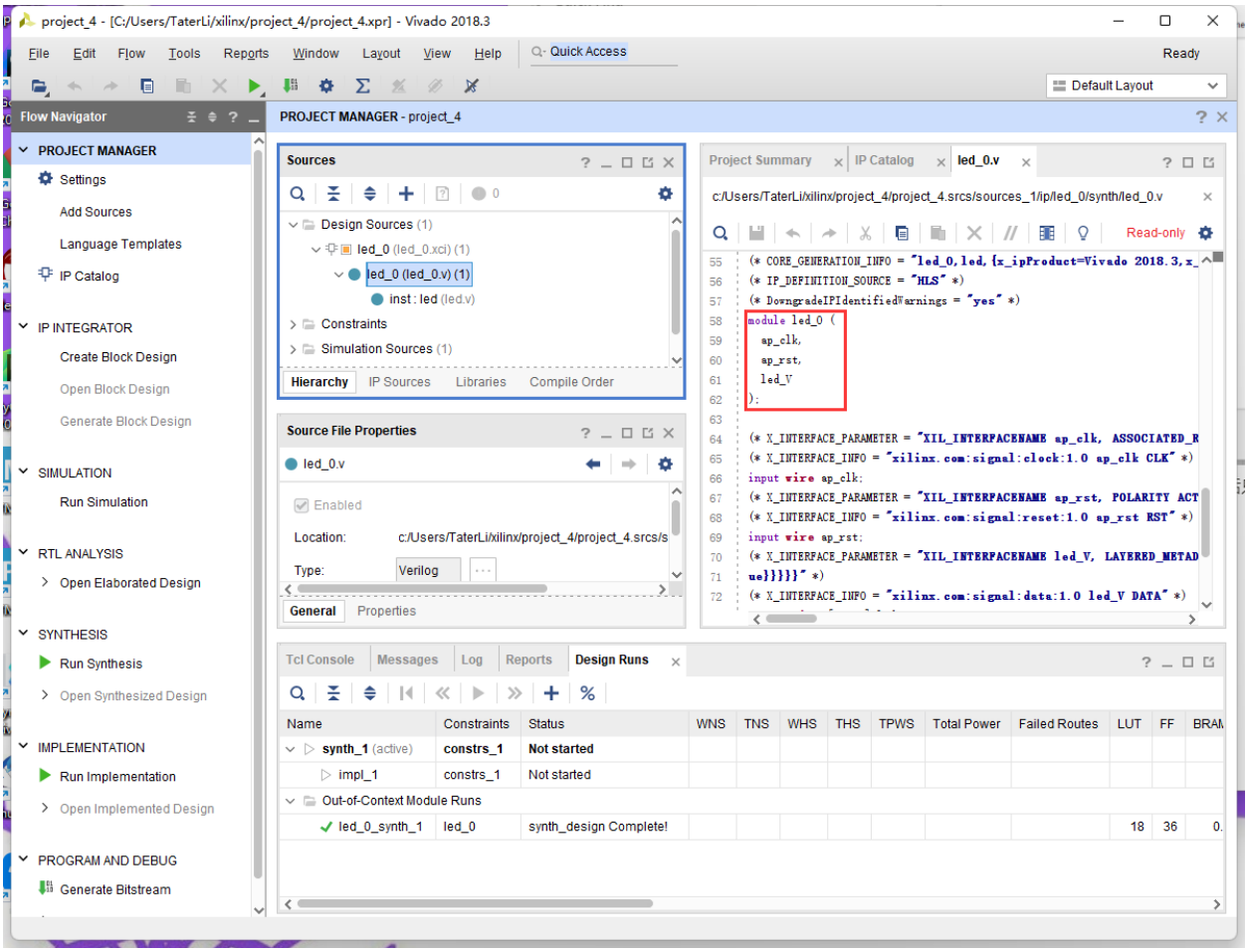
可是怎样才能到FPGA里,那就要用到刚才导出的IP,IP是位于HLS工程根目录的,因此导入即可.


Project Summary
IP Catalog
Cores | Interfaces
Search:
Name
AXI4
Status
License
VLNV
User Repository (c:/Users/TaterLi/AppData/Roaming/Xilinx/Vivado/led)
VIVADO HLS IP
Led
Pre-...
Included
xilin...
Vivado Repository
Alliance Partners
Audio Connectivity & Processing
Automotive & Industrial
Details
Select an IP or Interface or Repository to see details

现在看到IP了.



双击IP随意配置了一下,然后生成文件,然后只需要Generate Output Product,然后展开并准备实例化信号.



不过,HLS综合后的复位信号是高有效,和正常思路不同,特别要注意,可是坑了我很久,当然HLS可以以C风格写很复杂的程序,还可以自动调用一些所需IP,甚至是OpenCV之类高级库都能轻松引入,下面贴一个求均值代码,因为他用了数组,寄存器必然不够用,所以他还会自动调用一个PL核的RAM IP,并且有输入输出.

```
#include "stdio.h"
#include "ap_int.h"

#define MAX_FIR_TIME 200 //允许设置的最大平滑均值滤波次数 20

void filter_ip(unsigned int input, unsigned char fir_time, unsigned int *output, unsigned char *result_valid)
{
  #pragma HLS INTERFACE ap_ctrl_none port = return
  #pragma HLS INTERFACE ap_none port = output
  #pragma HLS INTERFACE ap_none port = result_valid

  static int filter_ptr = 0;
  static unsigned int buffer_in[256] = {0}; //缓存，存储输入进来的数据
  static unsigned char result_valid_flag = 0;
  unsigned long long tmp_add_value = 0;
  int i = 0;
  int filter_time = 0;

  filter_time = fir_time;
```

```
  if (filter_time == 0)
  {
    *output = 0;
    return;
  }
  else
  {
    if (filter_time > MAX_FIR_TIME)
      filter_time = MAX_FIR_TIME;
    buffer_in[filter_ptr] = input;
    for (i = 0; i < filter_time; i++)
      tmp_add_value += buffer_in[i]; // 累加最近滤波次数的所有数据
    filter_ptr++;
    if (filter_ptr > (filter_time - 1))
    {
      filter_ptr = 0;
      result_valid_flag = 1; // 滤波数据开始有效
    }
    *result_valid = result_valid_flag;
    *output = tmp_add_value / filter_time; // 求均值
    return;
  }
}
```