

# 新建 UartTx 样例工程的过程

此工程是直接在 example 工程中修改的。

1. VE 文件结尾 增加串口输出引脚定义：

```
59
60 # GPIO4_1 PIN_34 # LED1
61 # GPIO4_2 PIN_33 # LED2
62 # GPIO4_3 PIN_32 # LED3
63 # GPIO4_4 PIN_31 # LED4
64
65 GPTIMER1_CH0 PIN_7
66
67 # add for uart-tx
68 UART_TX PIN_17:OUTPUT
69
70 # add test led (to debug)
71 TEST_LED PIN_32:OUTPUT
72
```

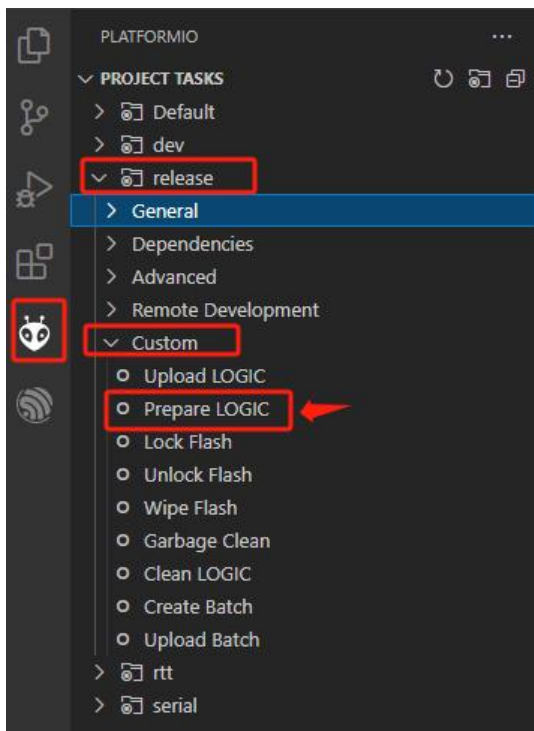
修改后，保存文件。

2. Platformio.ini 里打开自定义 logic:

```
7 #ips_dir = ../ips
8 ip_name = analog_ip
9 logic_dir = logic
10
```

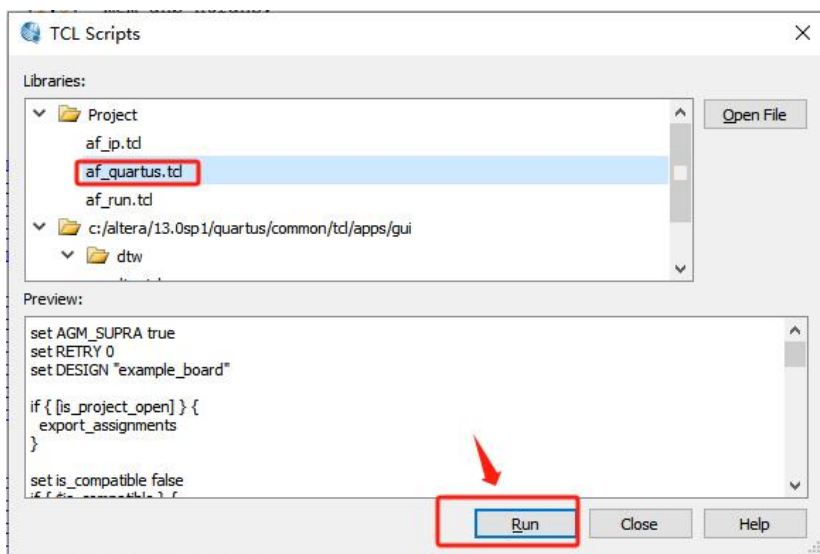
修改后，保存文件。

3. 在 VSCODE 左边栏里，生成 cpld 工程：



点完后，在 example 路径下产生 logic 文件夹。就是 cpld 的工程。

4. 用 Quartus 打开 logic 下的工程，先转换一次：



此时是个空的工程。

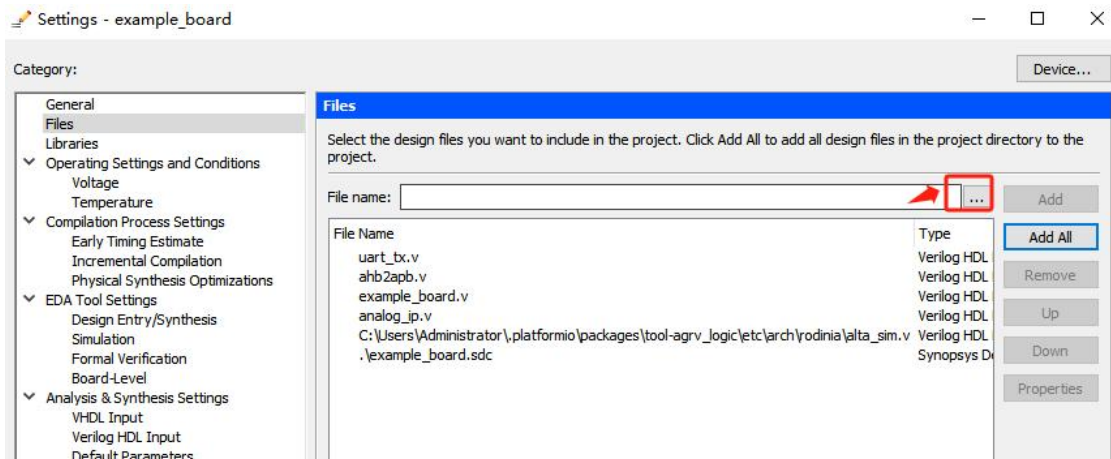
5. 添加 uart 部分的代码：

先把样例中的两个文件 copy 到这个 logic 路径下。

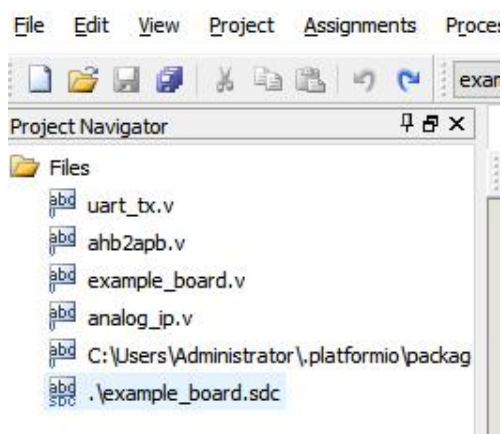
ahb2apb.v 和 uart\_tx.v。

其中，ahb2apb.v 是通用的转换模块；uart\_tx.v 是自定义逻辑，实现模拟往外发送串口数据的功能。

然后，把这两个文件加入到工程：



添加完后如图：



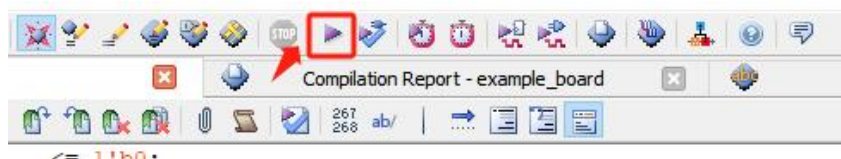
#### 6. 修改 analog\_ip.v 的代码：

原 analog\_ip.v 中只有一个空的接口：module analog\_ip

在里边增加模拟 ADC 部分的功能。

（该部分代码，可从样例中对比 copy 过来）

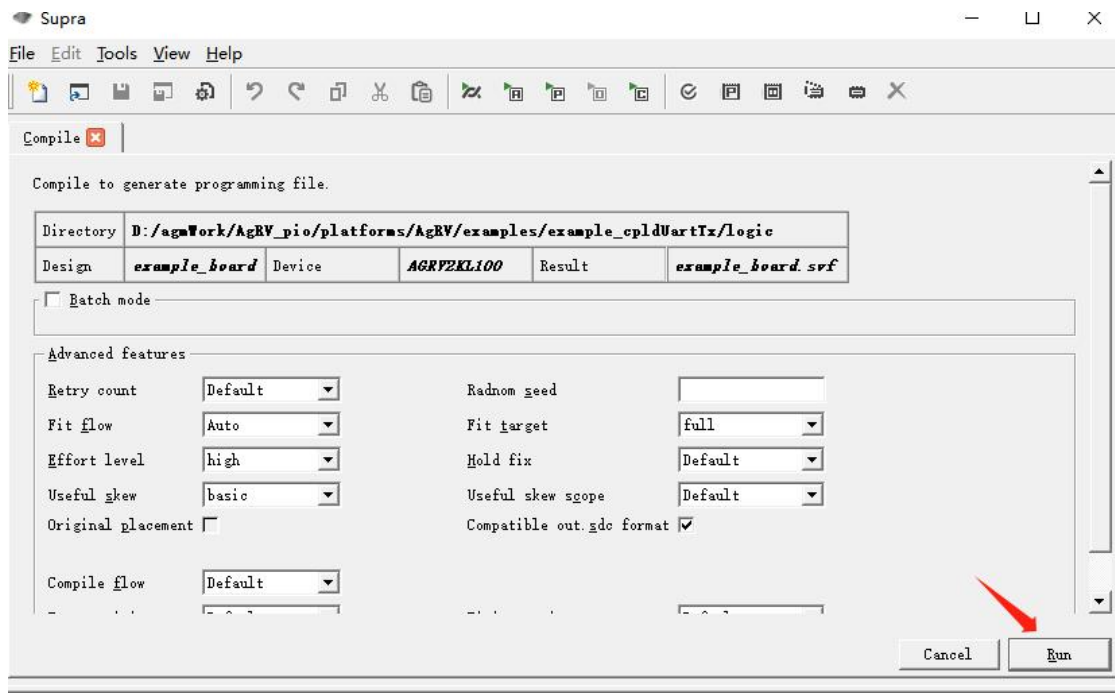
#### 7. 然后编译：



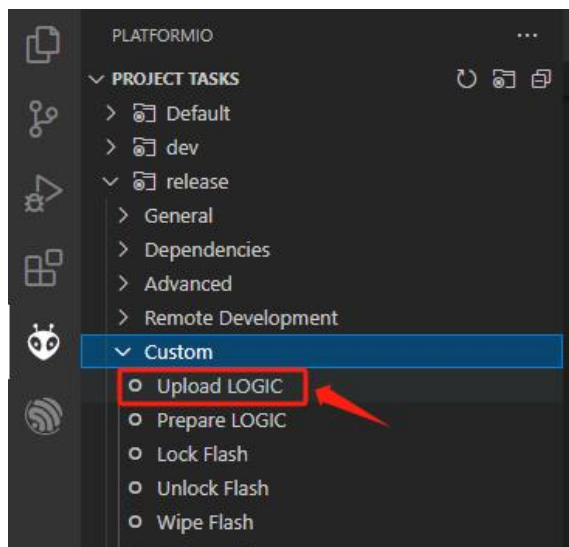
编译通过。

#### 8. 用 supra 编译：

在 SDK 根目录下打开 supra 工具，打开该工程，编译。



9. VSCODE 下烧录刚编译好的 logic:



10. 修改 mcu 的代码，来访问 cp1d:

修改如下:

```
#define RD_STATE_BUSY 0x01
#define RD_STATE_TC 0x02
typedef struct
{
    __IO uint32_t DAT; // 0x00 要写的数据
    __IO uint32_t STA; // 0x04 读取的状态
} UARTOWN_TypeDef;

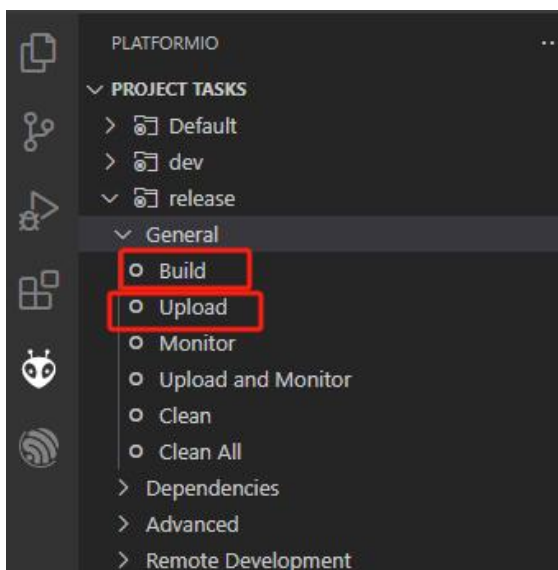
#define UART ((UARTOWN_TypeDef *) 0x60006000) //cp1d中该“外设”的地址
```

```

volatile int regState = 0;
while(1)
{
    for (int i = 1; i < 255; i++)
    {
        while((UART->STA) & RD_STATE_BUSY);
        UART->DAT = i; //发送一个字节
        while(!(UART->STA) & RD_STATE_TC)); //等待发送完成
        UTIL_IdleUs(100e3);
    }
}

```

11. 编译并烧录 mcu 代码:



在编译时，会报错找不到“analog.h”，是因为错误打开了宏 IPS\_ANALOG\_IP。

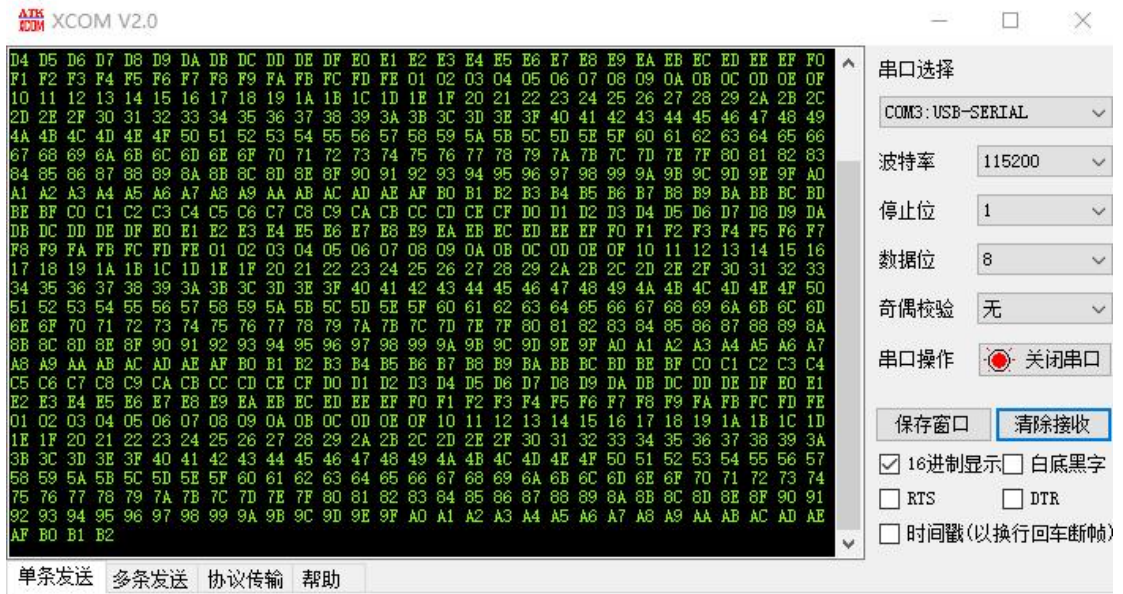
---- 打开 example\_analog.c，将宏 IPS\_ANALOG\_IP 改为 IPS\_ANALOG\_IP\_1，继续屏蔽整个文件的编译 即可。

再次编译。

最后下载。

12. 用串口查看:

然后，在 PC 上打开串口工具，用串口线的 RX 接开发板的 PIN\_17，设置 PC 串口工具的波特率为 115200，就可以看到数据的输出了。



cpld 中代码及说明 请参考 analog\_ip.v。