12 板载 PLL 芯片 MS5351 原理与应用

12.1. 背景介绍

本章实验将使用 I2C 驱动 AC208 核心板上的 PLL 芯片 MS5351M，并将该芯片输出的时钟频率给 FPGA 侧使用。

12.1.1. MS5351 简介

MS5351M 是一款通用频率综合器芯片，通过 I2C 配置，可产生从 2.5KHz 至 200MHz 的任意时钟输出。可替代晶体、晶体振荡器、锁相环、输出缓冲器。该芯片具有以下特点：

1. 3 通道输出从 2.5KHz 至 200MHz 时钟；
2. 输出频率误差 0ppm；
3. 高分辨率、低输出抖动；
4. 可工作在 25MHz 或者 27MHz 石英晶体；
5. 输出时钟相位可调；
6. 输出延时可调；
7. 输出时钟上升/下降时间可控；
8. 频率切换无毛刺；
9. 相互独立的电源供电引脚：内部核心电路电源 VDD：2.5V 或 3.3V；输出级电源 VDDO：1.8V 或 2.5V 或 3.3V；
10. 内部高电源抑制比；
11. 兼容 HCSL 和 PCIE Gen1

MS5351 的内部框图如下图 12. 1 所示。

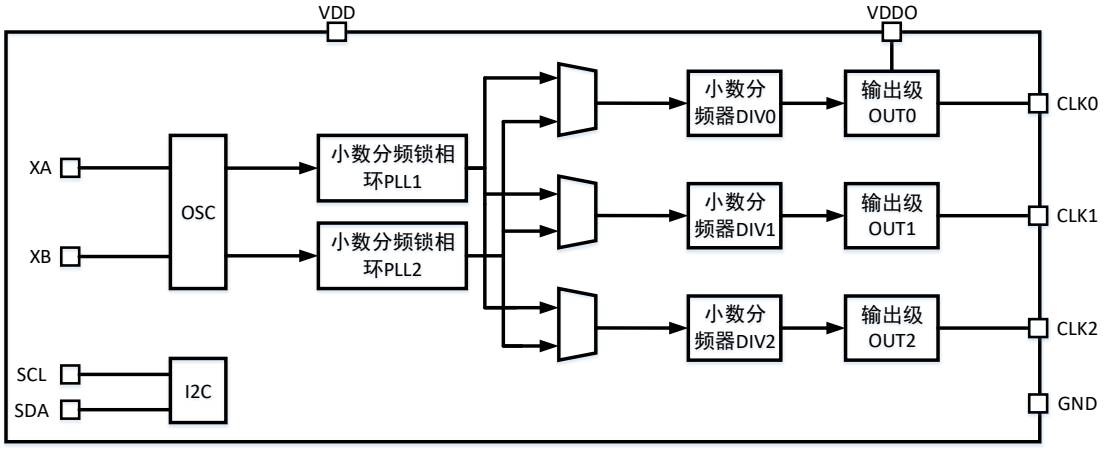


图 12. 1MS5351 的内部框图

由上图可以看出，VDD 和 VDDO 是两个供电口，VDD 为内部电路的供电口，VDDO 为三个输出级供电。（注：VDD 和 VDDO 可以分开供电，如果分开供电要求 VDDO 上电时间要早于 VDD）。XA 和 XB 是无源晶振（25M 或者 27M）的两端，通过 I2C 的配置，可以生成两组不同的 VCO 频率（PLL1，PLL2），两组 VCO 通过小数分频器和输出级共同作用下确定三组时钟的输出频率。在 AC208 核心板上对应的电路图如下图 12. 2 所示。

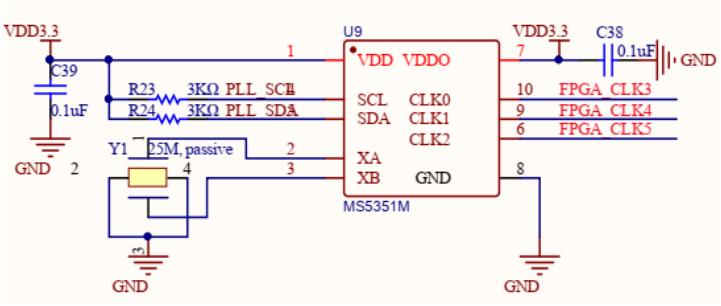


图 12. 2 MS5351M 电路原理图

12.1.2. 寄存器简介

MS5351M 与 SI5351 是完全兼容的，在进行软件编程的时候可以参考 SI5351 的寄存器进行配置。下面将对使用到的比较重要的寄存器进行介绍，更详细的寄存器请参考对应的技术手册。SI5351 的可编程时钟 IC 框图如下图 12. 3 所示。

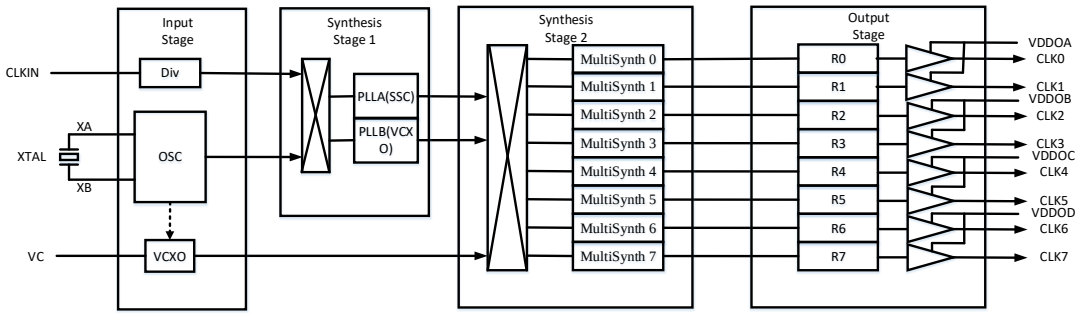


图 12. 3 SI5351 框图

12.1.2.1. Register 16(CLK0 Control)

输出通道 CLK0 的控制寄存器，对于该寄存器的每一位的介绍如下表 12- 1 所示。

表 12- 1 CLK0 Control 寄存器介绍表

对应位	名称	类型	描述
7	CLK0_PDN	R/W	CLK0 掉电设置位。该位允许在不需要使用 CLK0 输出时，关闭 CLK0 的输出驱动器以节省电力。 0：CLK0 上电；1：CLK0 掉电
6	MS0_INT	R/W	MultiSynth 0 整数模式。该位可以强制 MS0 为整数模式，以改善抖动性能。注意，当为 CLK0 指定延迟偏移量时，必须设置为小

			数模式。 0：MS0 以小数模式运行；1：MS0 以整数模式运行
5	MS0_SRC	R/W	为 CLK0 选择 MultiSynth 来源。 0：选择 PLLA 作为 MultiSynth 的时钟来源； 1：选择 PLLB 作为 MultiSynth 的时钟来源；
4	CLK0_INV	R/W	输出时钟 CLK0 反向位。 0：输出时钟 CLK0 不反向；1：输出时钟 CLK0 反向
3:2	CLK0_SRC[1:0]	R/W	输出时钟 CLK0 的输入源。 00：选择 XTAL 作为 CLK0 的时钟源； 01：选择 CLKIN 作为 CLK0 的时钟源； 10：未被定义； 11：选择 MultiSynth 0 作为 CLK0 的时钟源
1:0	CLK0_IDRV[1:0]	R/W	CLK0 输出上升和下降时间/驱动强度控制。 00：2mA； 01：4mA； 00：6mA； 01：8mA；

CLK1 Control（17）和 CLK2 Control（18）寄存器的每一位的意义和上诉表所示内容一致。

12.1.2.2. Register26~ Register33(Multisynth NA Parameters)

PLLA 输出频率参数设置寄存器，对于寄存器的说明如下表 12- 2 所示。

表 12- 2 PLLA 输出频率参数设置寄存器

寄存器编号	对应位	类型	名字	描述
26	7:0	R/W	MSNA_P3[15:8]	Multisynth NA 参数 3。PLLA 反馈合成分频器小数部分分母的编码表示。
27	7:0	R/W	MSNA_P3[7:0]	
28	7:4	R/W	Unused	未使用
	3:2	R/W	Reserved	保留默认值 0
	1:0	R/W	MSNA_P1[17:16]	Multisynth NA 参数 1。PLLA 反馈合成分频器整数部分的编码表示。
29	7:0	R/W	MSNA_P1[15:8]	
30	7:0	R/W	MSNA_P1[7:0]	
31	7:4	R/W	MSNA_P3[19:16]	Multisynth NA 参数 3。
	3:0	R/W	MSNA_P2[19:16]	Multisynth NA 参数 2。PLLA 反馈合成分频器小数部分分子的编码表示。
32	7:0	R/W	MSNA_P2[15:18]	
33	7:0	R/W	MSNA_P2[7:0]	

Register34~ Register41 表示 PLLB 输出频率参数设置寄存器。相关寄存器的描述请查看对应的技术手册。

12.1.2.3. Register42~ Register49(Multisynth0 Parameters)

MultiSynth 0 Divider 参数设置寄存器说明如下表 12- 3 所示。

表 12- 3 MultiSynth 0 Divider 参数设置寄存器说明表

寄存器编号	对应位	类型	名字	描述
42	7:0	R/W	MS0_P3[15:8]	

43	7:0	R/W	MS0_P3[7:0]	Multisynth 0 参数 3。MultiSynth 0 Divider 小数部分的分母的编码表示。
44	7	R/W	Unused	未使用
	6:4	R/W	R0_DIV[2:0]	R0 输出分频器。 000b: Divide by 1 001b: Divide by 2 010b: Divide by 4 011b: Divide by 8 100b: Divide by 16 101b: Divide by 32 110b: Divide by 64 111b: Divide by 128
	3:2	R/W	MS0_DIVBY4[1:0]	MS0 除以 4 启用。 11: Divide by 4 enabled. 00: Divide by a value other than 4.
	1:0	R/W	MS0_P1[17:16]	Multisynth 0 参数 1。MultiSynth 0 分频器整数部分的编码表示
45	7:0	R/W	MS0_P1[15:8]	
46	7:0	R/W	MS0_P1[7:0]	
47	7:4	R/W	MS0_P3[19:16]	Multisynth 0 参数 3。
	3:0	R/W	MS0_P2[19:16]	Multisynth 0 参数 2。MultiSynth 0 Divider 小数部分的分子的编码表示。
48	7:0	R/W	MS0_P2[15:8]	
49	7:0	R/W	MS0_P2[7:0]	

Register50~ Register57、Register58~ Register65 分别是 MultiSynth 1 Divider、MultiSynth 2 Divider 相关参数设置寄存器，对于寄存器的每一位介绍和上表所述一致。

12.1.2.4. Register 177(PLL Reset)

复位寄存器，对于寄存器的每一位介绍如下表 12- 4 所示。

表 12- 4 复位寄存器说明表

Bit	名称	功能
7	PLLB_RST	PLLB 复位。向该位写入 1 将会复位 PLLLB。这是一个自清除位。
6	Reserved	保留为默认值
5	PLLA_RST	PLLA 复位。向该位写入 1 将会复位 PLLLA。这是一个自清除位。
4:0	Reserved	保留为默认值

12.1.3. 输出频率参数设置

接下来我们将介绍如何配置相关参数得到所需的频率。

12.1.3.1. 配置输入和 PLL 寄存器参数

通过配置控制寄存器的 MS0_SRC 位来确定使用 PLLA 还是 PLLB，为 0 则是 PLLA；1 为 PLLB。接下来以 PLLA 为例：

PLLA 输入参考源来自于 25M 的晶振 XTAL，PLLA 的输出频率计算公式如下所示：

$$f_{pllout} = f_{XTAL} \times (a + \frac{b}{c})$$

上述公式中的$a + \frac{b}{c}$的有效取值范围为 15+0/1048575 到 90+0/1048575。这里以 PLLA 为例，对应需要配置的寄存器为 Register26~ Register33 中的 MSNA_P3~ MSNA_P1。其对应需要配置的值计算如下：

$$MSNA\_P1[17:0] = 128 \times a + Floor\left(128 \times \frac{b}{c}\right) - 512$$

$$MSNA\_P2[19:0] = 128 \times b - c \times Floor(128 \times \frac{b}{c})$$

$$MSNA\_P3[19:0] = c$$

12.1.3.2. 配置输出寄存器相关参数

将得到的 PLL 的输出频率通过输出寄存器得到最终的频率输出。以 PLLA 为例，需要配置的寄存器为 Register42~ Register49 的 MS0_P3~ MS0_P1。

当需要输出的频率 fout≤150Mhz 时，其对应计算公式如下所示：

$$MS0\_P1[17:0] = 128 \times a + Floor\left(128 \times \frac{b}{c}\right) - 512$$

$$MS0\_P2[19:0] = 128 \times b - c \times Floor(128 \times \frac{b}{c})$$

$$MS0\_P3[19:0] = c$$

当 150Mhz <fout≤200Mhz 时，需要设置以下位：

- MS0_P1=0；
- MS0_P2=0；
- MS0_P3=1；
- MS0_INT=1；
- MS0_DIVBY4[1:0]=11；

在某些情况下，用户可能需要相对于其他输出反转一个或多个输出的极性(即 180°相位偏移)，可以通过设置 CLKx _ INV = 1 实现。

12.2. 实验介绍

本次实验将使用 I2C 驱动核心板上的板载 PLL 芯片输出频率，并将得到的频率提供给 FPGA 侧使用，最终通过 LED 灯的闪烁判断频率是否设置成功。

12.3. 建立 HQFPGA 工程

将已有的 FPGA 工程复制至本次工程的实验目录之下，对其进行工程名的修改，修改方式参考实验二中 2.3 节的内容。

12.3.1. 编写 FPGA 侧应用程序

本次实验提供的 FPGA 侧程序位于：CM3_System/XIST_MS5351/rtl/led_drtiver。需要实现的功能就是根据时钟频率的不同，LED 闪烁频率不同，以此来判断板载 PLL 芯片的频率是否设置成功。代码设计思路：定义 4 个计数器，计数值都是一样的，然后通过 4 个不同的时钟进行计数，四个时钟分别是：MAIN_CLK（25M）、板载 PLL 芯片的三个时钟输出[2:0]pll_clk，具体的实验代码请自行查看对应的.v 文件。添加.v 文件的方式参考实验二中 2.4.2 节的内容。

12.4. 物理管脚约束

本次实验需要分配的引脚有：I2C 引脚分配给板载 PLL 芯片，板载 PLL 芯片的 3 个时钟信号引脚，四个 LED 引脚。本次实验需要新加的引脚分配文件如下所示（注意：在之前的实验中是通过 GPIO 控制的 LED 灯，在进行本次实验需要将其引脚分配语句删除）：

```
#板载 PLL 芯片时钟信号接口
phycst.pin.set {pll_clk[0]} N9
phycst.pin.set {pll_clk[1]} N11
phycst.pin.set {pll_clk[2]} P10

#----IIC 接口-----------
#IIC_SCL
phycst.pin.set {GPIO[15]} E3
#IIC_SDA
phycst.pin.set {GPIO[16]} D3

#----LED 接口-----------
#开发板上的 LED0
phycst.pin.set {led[0]} D15
#开发板上的 LED1
phycst.pin.set {led[1]} G15
#开发板上的 LED2
phycst.pin.set {led[2]} F14
#开发板上的 LED3
phycst.pin.set {led[3]} E14
```

12.5. 编译设计

保存完修改后的顶层文件和物理约束文件之后，点击全部运行，生成本次实验需要的 FPGA 侧的 bin 文件，操作如下图 12. 4 所示：

图 12. 4 编译设计

12.6. 建立 MDK 工程

首先将已有的 MDK 工程复制至存放本次实验代码的文件夹下，然后对其进行修改，修改方式参考实验二中 2.8 节中的内容。实验中需要使用到的应用库分别是“io_i2c”和“i2c”，进行实验时需要将这两个库文件添加至工程中。

12.7. 软件设计

本次实验通过两种方式实现 I2C 通信：IO 模拟和硬件 I2C。在“ms5351.c”文件中有关于这两种实现方式的条件编译语句，如下所示，使用的时候选择其中一种即可：

```
#define IIC_HAL 0 //硬件 I2C
#define IIC_IO 1//选择 IO 模拟
```

12.7.1. MS5351 库

12.7.1.1. MS5351_IIC_Init

初始化 I2C，在使用 MS5351 的时候，都需要先使用该函数初始化 I2C，函数代码如下所示：

```
void MS5351_IIC_Init()
{
  #if IIC_IO
  io_i2c_init();//IO_IIC 初始化
  #endif

  #if IIC_HAL
  IIC_Init(CM3DS_MPS2_I2C, I2C_Mode_Master, I2C_Speed_Standard, CM3_I2C_OWN_ADDR);
  #endif
}
```

12.7.1.2. i2cSendRegister

写 MS5351 寄存器函数。通过该函数向 MS5351 寄存器中写入需要配置的

值。函数代码如下所示：

```
uint8_t i2cSendRegister (uint8_t reg, uint8_t data)
{
    #if IIC_HAL
    I2CWriteOneReg(MS5351_Address,0,reg,data);
    #endif

    #if IIC_IO
    io_i2c_write_reg(MS5351_Address,0,reg,data);
    #endif
}
```

可以看出，代码中主要调用了 I2CwriteOneReg 函数和 io_i2c_write_reg 函数，这两个函数都是通过 I2C 写单字节的数据，函数变量意义都是一样的，这里以 IO 模拟为例，对其变量说明如下表 12- 5 所示：

表 12- 5 io_i2c_write_reg 函数变量说明表

变量名称	变量意义
dev_id	器件地址
addr_mode	寄存器地址模式选择，0：代表寄存器的地址为 8 位；1：寄存器地址为 16 位
reg_addr	寄存器地址
data	需要写入的数据

由上表可以看出，首先需要知道的就是 MS5351 的器件地址，通过查询相关的器件手册得知对应的器件地址：0xC0（写数据）；0xC1（读数据）。MS5351 的寄存器都是 8 位的，故模式选择位为 0。

12.7.1.3. SofterCB_FreSet

首先通过 ClockBuilder 软件生成对应的寄存器配置表，然后通过该函数依次将寄存器配置表中的内容写入 PLL 芯片。函数代码如下所示：

```
void SofterCB_FreSet()
{
    uint32_t i =0;
    for(i=0;i<sizeof(ms5351_cfg_reg)/sizeof(ms5351_cfg_reg[0]);i++)
    {
        i2cSendRegister(ms5351_cfg_reg[i][0],ms5351_cfg_reg[i][1]);
    }
}
```

函数是通过 i2cSendRegister 函数实现寄存器的写操作。函数中的 ms5351_cfg_reg 就是对应的寄存器配置表，用户可以根据自己的需求修改寄存器配置表中的内容，得到自己需要的频率值。例程中提供的是三路输出，分别是 50M、80M 和 125M。

12.7.2. 添加用户代码

在主函数中，我们只需要初始化 I2C，然后调用 SofterCB_FreSet 函数实现寄存器的配置，main 函数内容如下所示：

```
#include "CM3DS_rcc.h"
#include "CM3DS_MPS2.h"
#include "CM3DS_i2c.h"
#include "io_i2c.h"
#include "i2c.h"
#include "ms5351.h"
int main(void)
{
    MS5351_IIC_Init();
    SofterCB_FreSet();
    while(1){

    }
}
```

12.7.3. ClockBuilder 软件的使用

软件可以通过官网连接下载 www.silabs.com/ClockBuilder ，如果官网查询不到，可以联系我们。

12.7.3.1. ClockBuilder 安装步骤说明

1. 解压“clockbuilder-pro-installer.rar”文件夹，找到 ClockBuilder-Pro-4.4 进行安装，如下图 12. 5 所示。

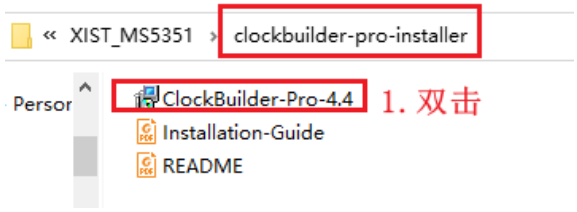


图 12. 5 找到 clockbuilder 软件安装

2. 双击之后，弹出如下图 12. 6 所示的界面，直接点击 Next。

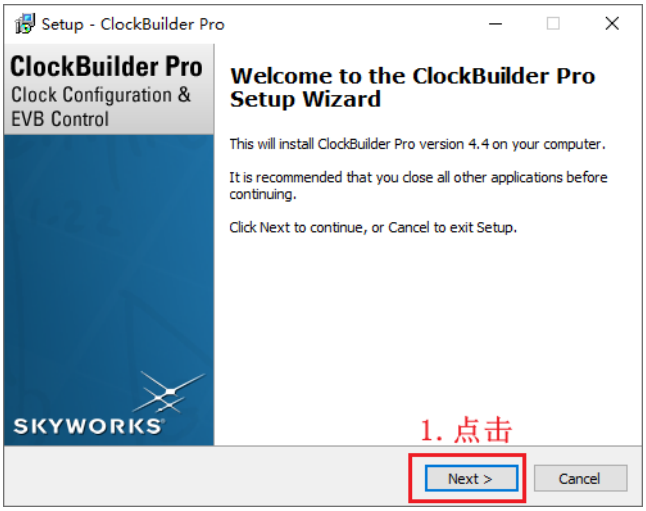


图 12. 6 软件安装欢迎界面

3. 进入软件许可协议界面，选择“I accept the agreement”，操作如下图 12. 7 所示。

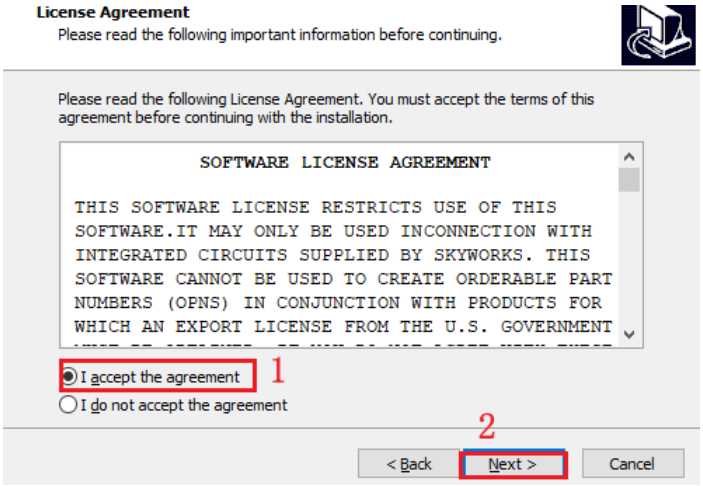


图 12. 7 软件许可协议界面

4. 继续授权安装，操作步骤如下图 12. 8 所示。

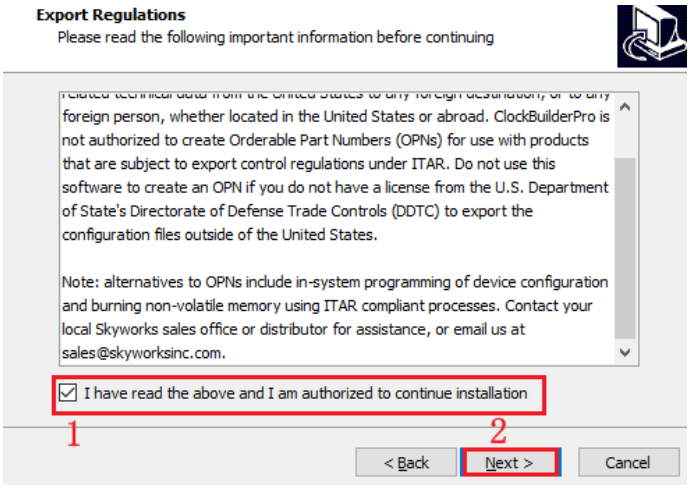


图 12. 8 授权安装

5. 选择软件安装路径，操作如下图 12. 9 所示。

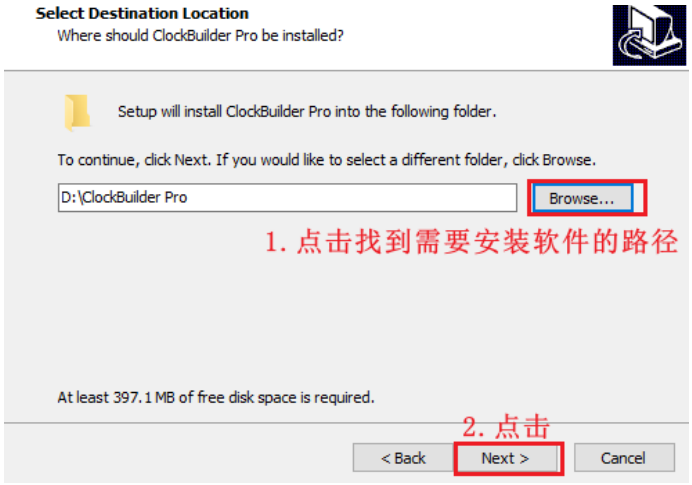


图 12. 9 选择软件路径

6. 软件快捷方式选择，直接点击 Next 即可，操作如下图 12. 10 所示。

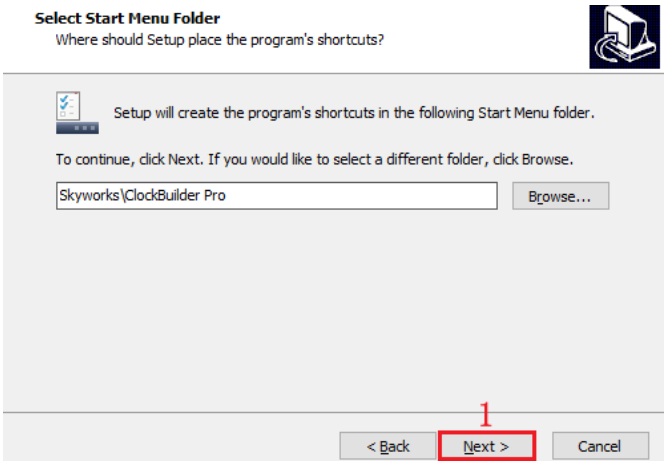


图 12. 10 选择软件快捷方式路径

7. 选择需要安装的附加任务，直接点击 Next，操作如下图 12. 11 所示。

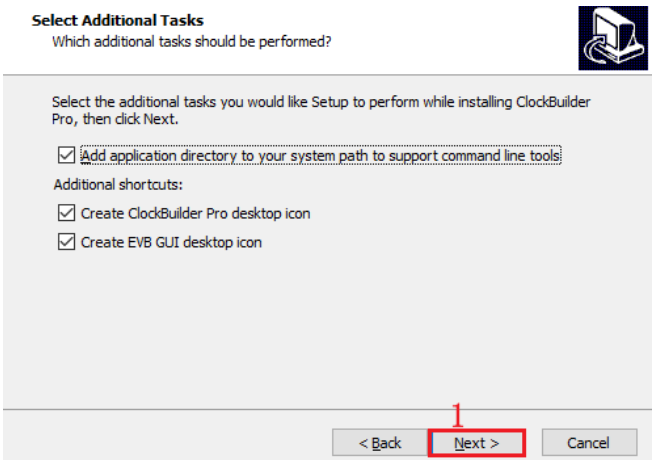


图 12. 11 选择需要安装的附加任务

8. 选择 Install，开始安装，操作如下图 12. 12 所示。

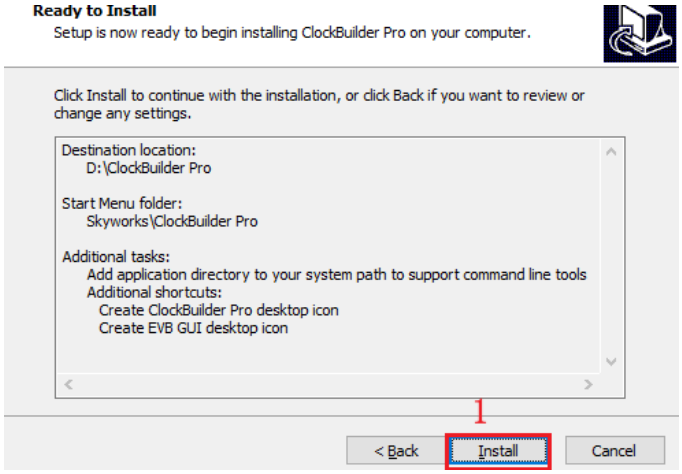


图 12. 12 进入安装

9. 等待安装完成，安装完成之后，显示如下图 12. 13 所示的界面。

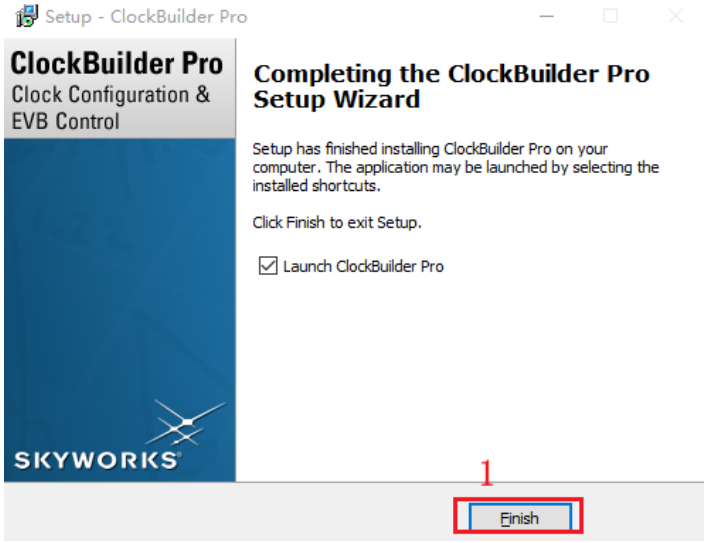


图 12. 13 软件安装完成示意界面

12.7.3.2. 使用 ClockBuilder 软件生成寄存器配置表

1. 打开 ClockBuilder Pro Wizard 软件，建立新工程，操作步骤如下图 12. 14 所示。

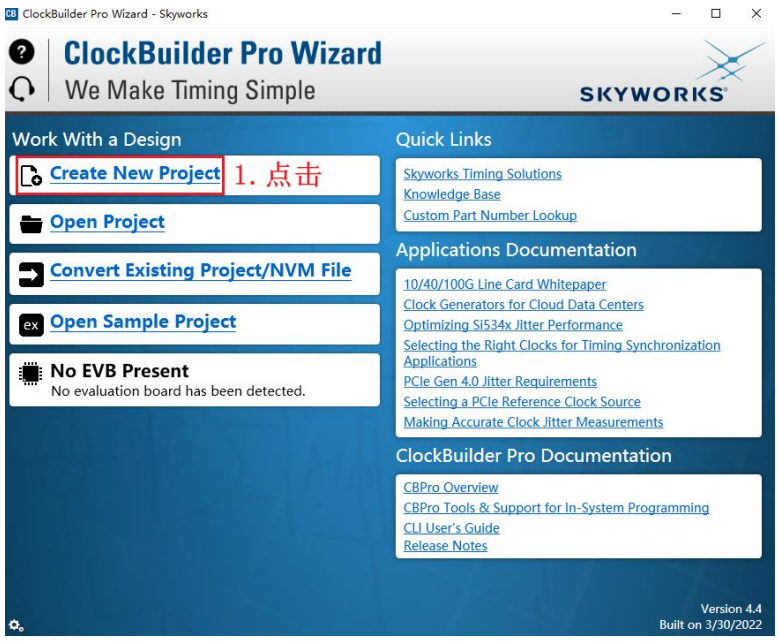


图 12. 14 建立新工程

2. 选择 Show All Devices，展示所有器件，操作如下图 12. 15 所示。

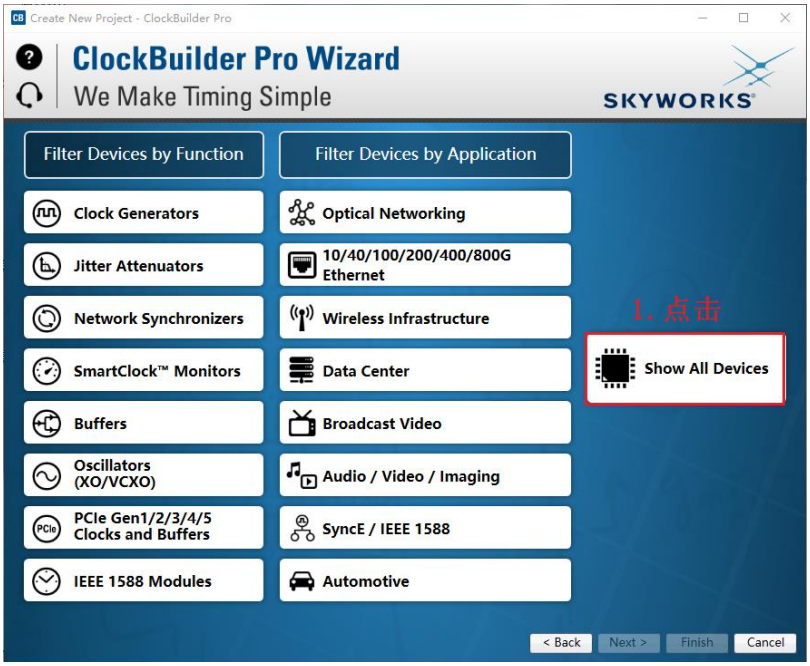


图 12. 15 选择展示所有器件

3. 找到器件 SI5351A 进行选择，操作如下图 12. 16 所示。

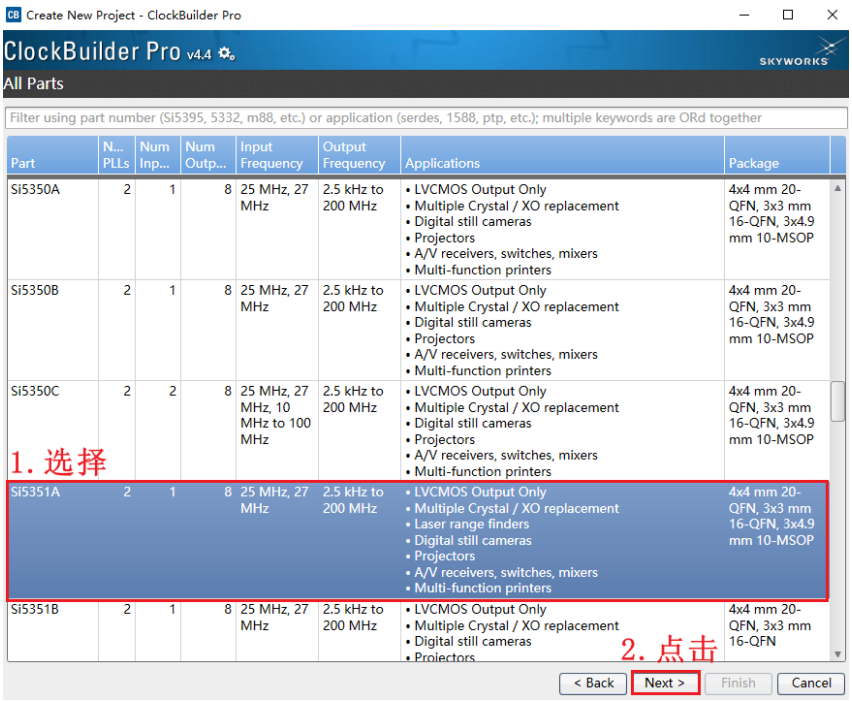


图 12. 16 选择器件

4. 选择器件之后，弹出如下图 12. 17 界面直接点击 Next。

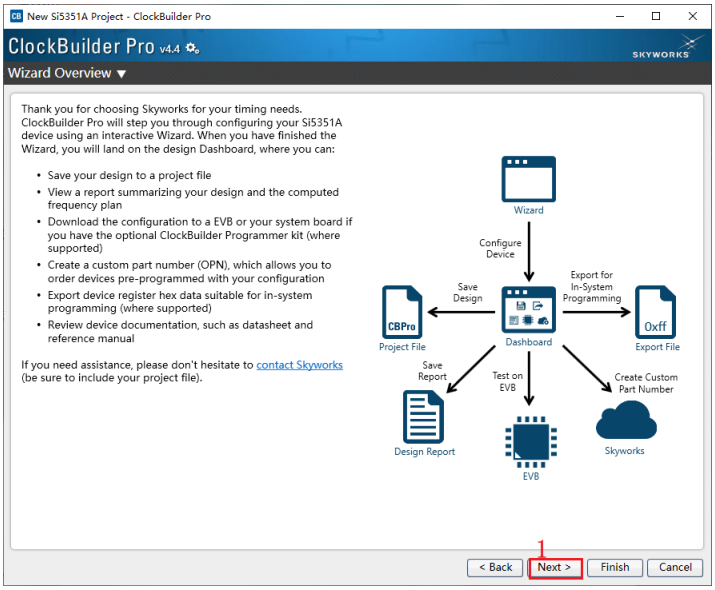


图 12. 17

5. 出现如下图 12. 18 所示界面，直接点击 Next。

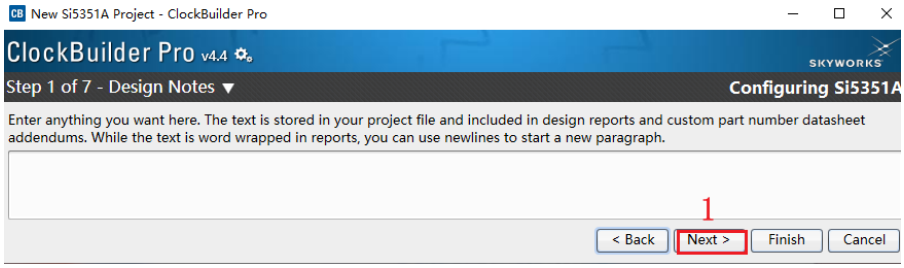


图 12. 18

6. 选择器件类型，我们使用到的 3 路输出，操作步骤如下图 12. 19 所示。

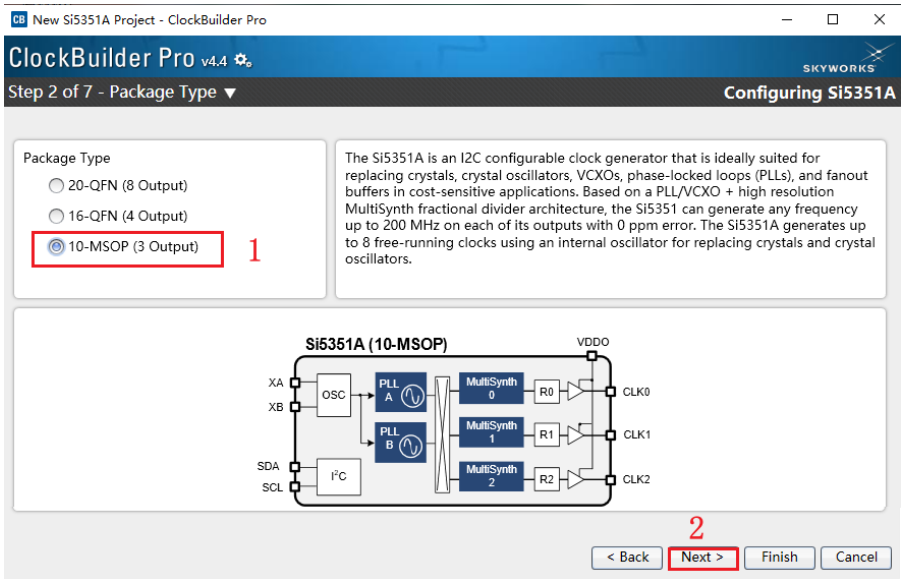


图 12. 19 选择器件类型

7. 选择 I2C 的器件地址，在实验中器件地址是通过 MDK 软件设置的，在这里不需要修改，直接点击 Next，操作如下图 12. 20 所示。

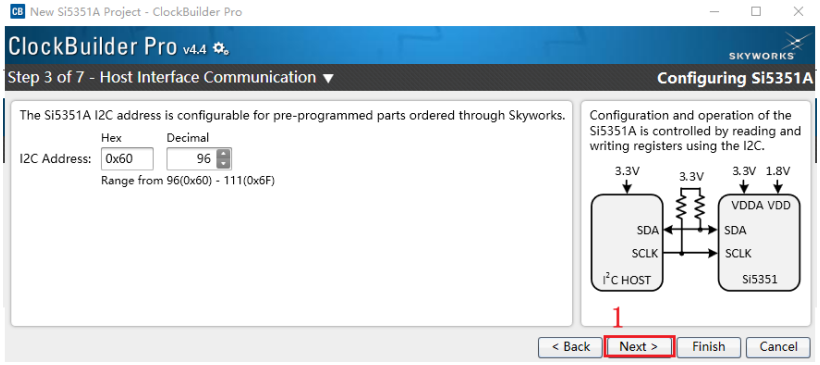


图 12. 20 设置器件地址

8. 选择 XA/XB 的时钟频率输入，在 12.1.1 节中，我们介绍过，使用的是 25M 的晶振，操作步骤如下图 12. 21 所示。

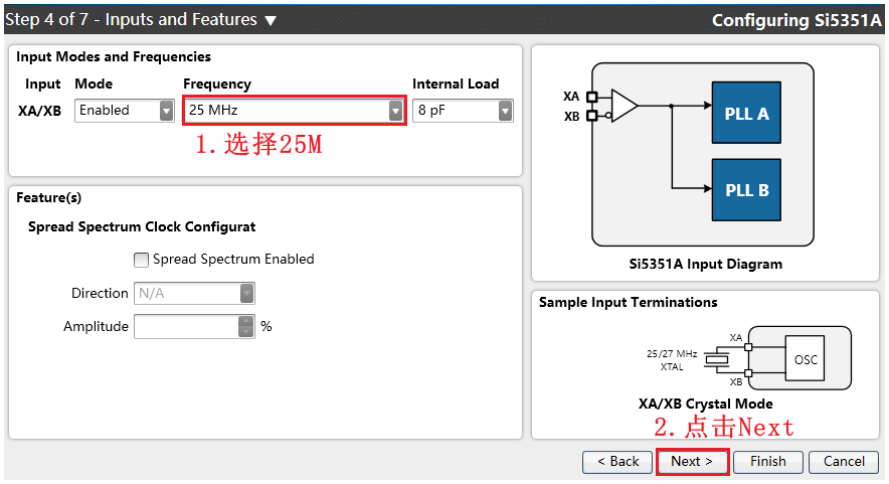


图 12. 21 选择时钟输入频率

9. 选择需要的输出频率，这里举例：使用三路输出，分别是 50M、80M 和 125M（设置出现警告可忽略），设置步骤如下图 12. 22 所示。

图 12. 22 设置需要的输出频率

10. 设置相位偏移，用户根据自己需求进行设置，本次实验暂未使用，直接点击 Next 即可，操作如下图 12. 23 所示。

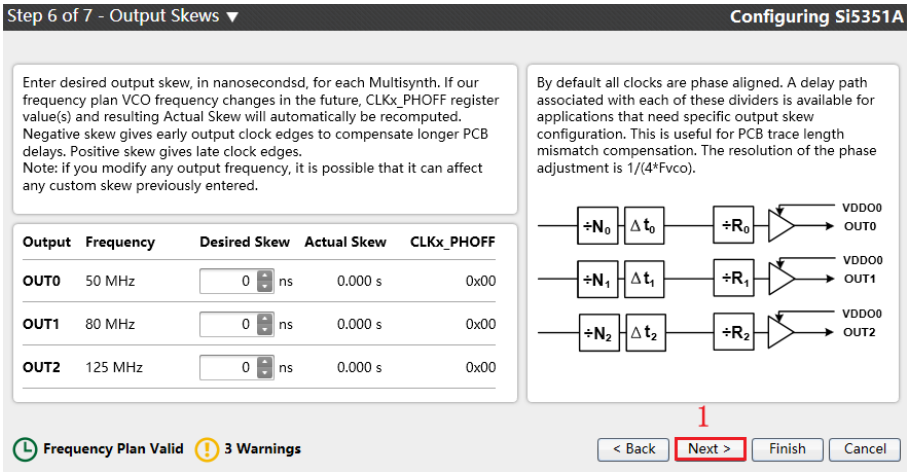


图 12. 23 设置相位偏移

11. 设置跟踪特性，直接点击 Next，操作如下图 12. 24 所示。

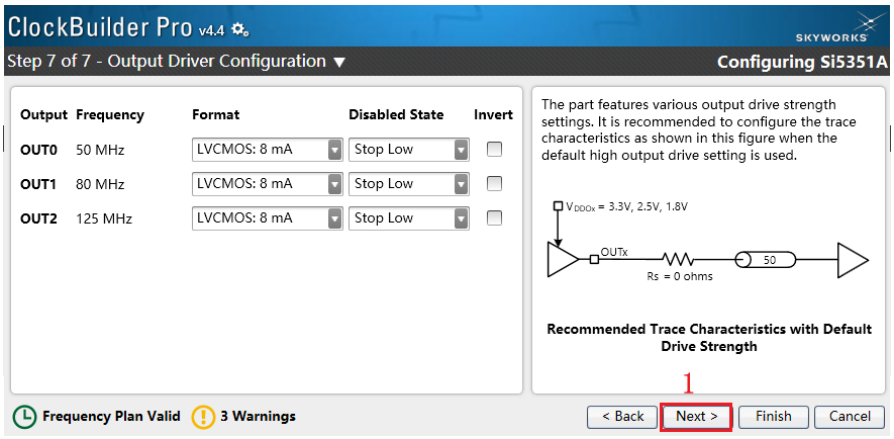


图 12. 24 设置跟踪特性

12. 点击 Export，导出所需文件，操作如下图 12. 25 所示。

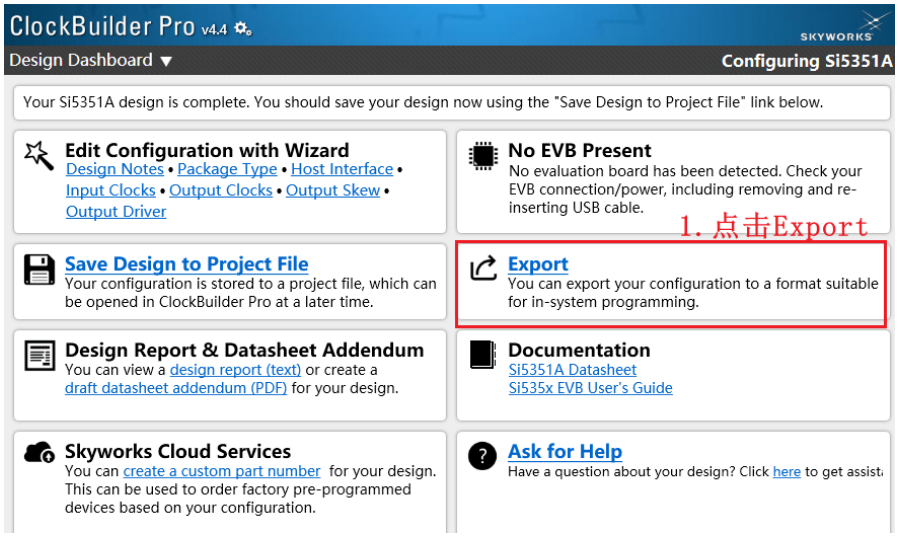


图 12. 25 选择 Export 导出所需文件

13. 在弹出的 Export 界面中，依次选择 Register File→C Code Header File→Save to File，操作步骤如下图 12. 26 所示。

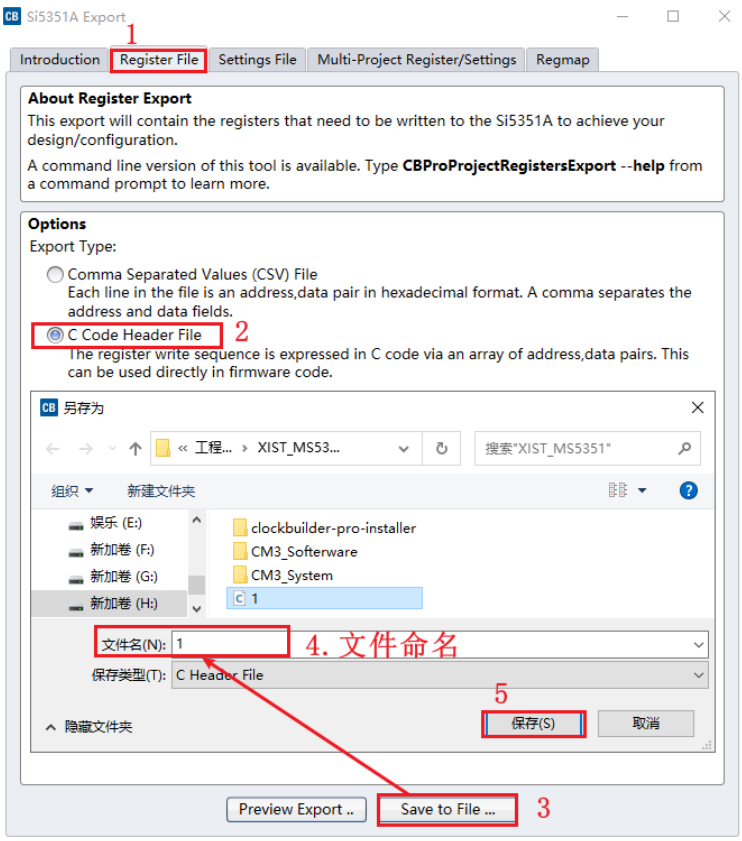


图 12. 26 保存导出的文件

14. 替换代码中寄存器配置，找到导出的文件，可以看到对应的寄存器配置，将得到的文件中的寄存器表复制至 ms5351cfg.h 文件中，操作步骤如下图 12. 27 所示。

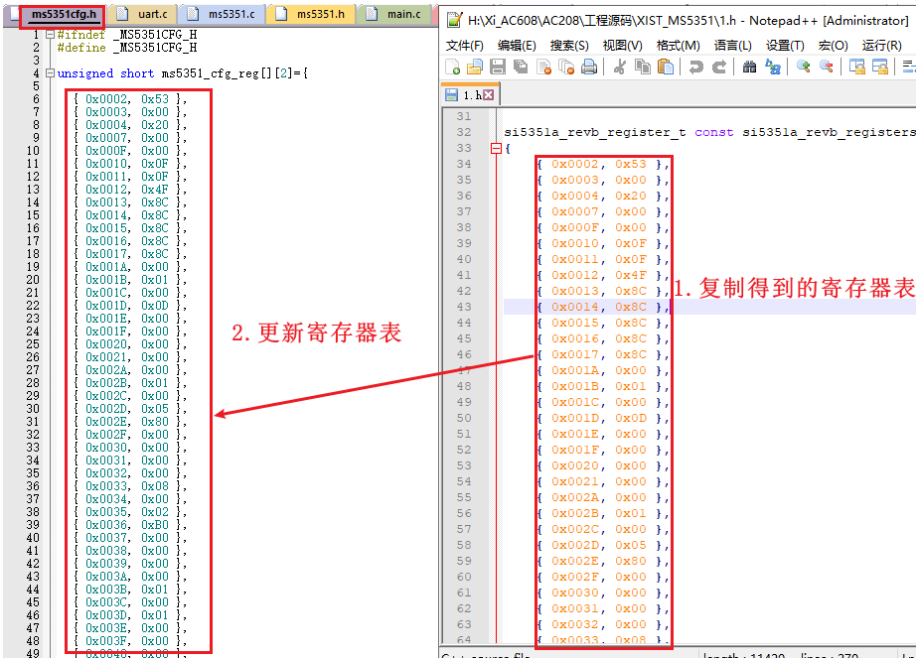


图 12. 27 替换寄存器配置表

12.8. 板级验证

12.8.1. 实验所需硬件

(1) AC208 开发板
(2) FPGA 下载器：XIST USB Cable
(3) CM3 仿真器：DAP Link
(4) 电源线一根

12.8.2. 硬件连接

硬件连接参考实验一。

12.8.3. 下载文件至目标板

下载文件的方式参考实验一。

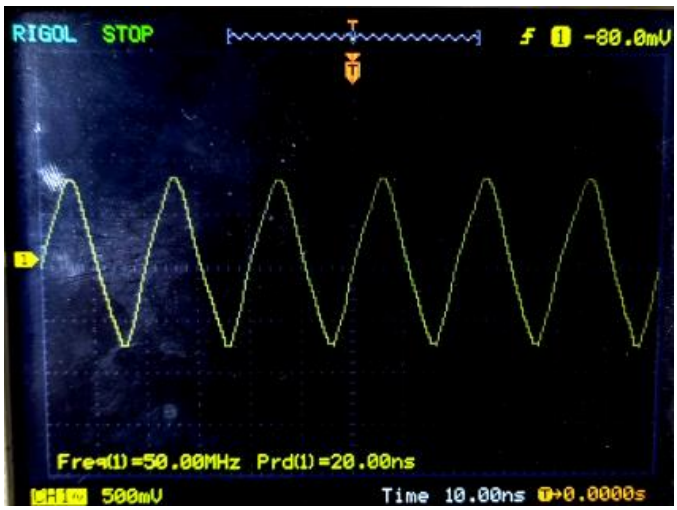
12.8.4. 功能演示

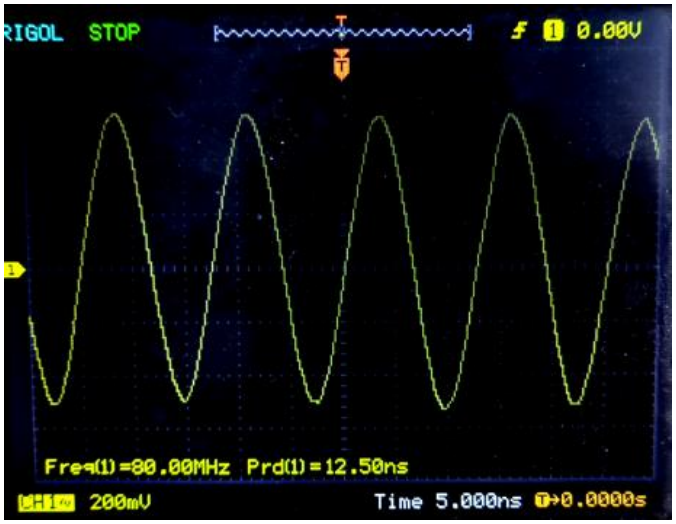
开发板上可以看到的现象是：LED0~LED3 闪烁速度依次变快，这表示频率设置成功。

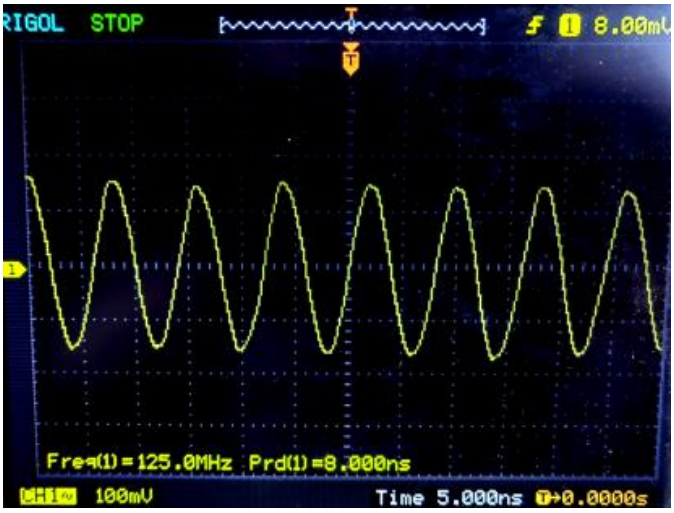
12.9. 思考与总结

本次实验通过 I2C 成功驱动了 AC208 核心板上的 PLL 芯片 MS5351， 并最终将得到的时钟频率提供给 FPGA 侧使用。关于 PLL 芯片的寄存器配置是通过 ClockBuilder 软件生成的，此种方法方便用户在不了解寄存器的情况下配置。

最终输出的频率用户可以通过示波器进行测量，在我们提供的例程用户可以通过通用 GPIO[1]（CLK0）、GPIO[3]（CLK1）、GPIO[5]（CLK2）进行测量。




RIGOL STOP
1 0.00V
Freq(1)=80.00MHz Prd(1)=12.50ns
CH1 200mV
Time 5.000ns T→0.0000s


RIGOL STOP
1 8.00mV
Freq(1)=125.0MHz Prd(1)=8.000ns
CH1 100mV
Time 5.000ns T→0.0000s