蓝牙通信

实验介绍：

本实验利用 **EGo 数模混合口袋实验平台**上的蓝牙模块与板卡进行无线通信。使用支持蓝牙 4.0 的手机与板卡上的蓝牙模块建立连接，并且通过手机 APP 发送命令，控制 FPGA 板卡上的硬件外设。

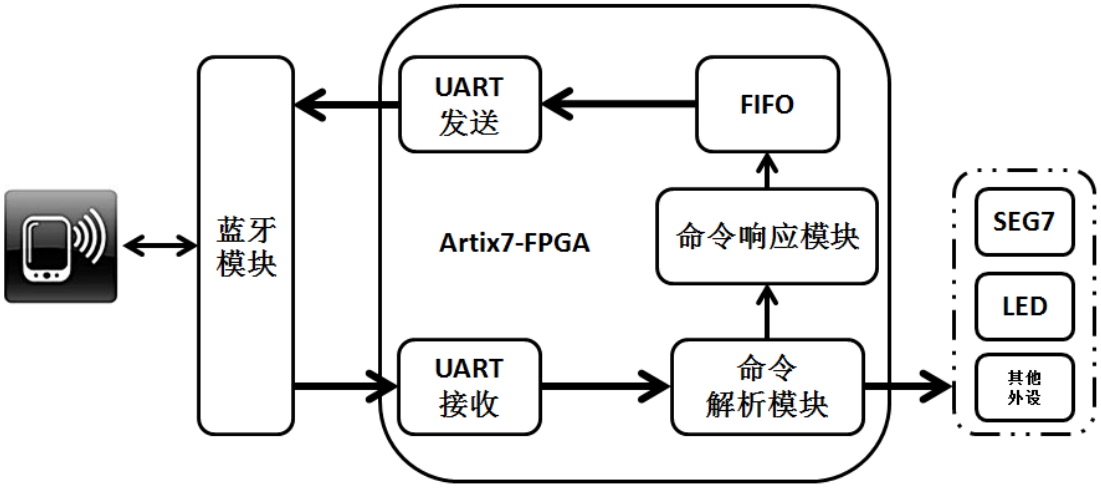
实验目的：

1. 学习使用 Xilinx Vivado 开发工具完成 FPGA 设计。
2. 学习蓝牙无线传输技术的原理。
3. 熟悉 **EGo 数模混合口袋实验平台**上的蓝牙模块、数码管等接口外设的使用。

实验原理：

蓝牙无线技术是使用范围最广泛的全球短距离无线标准之一。**EGo 数模混合口袋实验平台**搭载的蓝牙模块是基于 TI 公司 CC2541 芯片的蓝牙 4.0 模块，具有 256Kb 配置空间，遵循 V4.0 BLE 蓝牙规范。

本实验中利用板卡上的蓝牙模块与外界支持蓝牙 4.0 标准的设备（如手机）进行交互。该蓝牙模块出厂默认配置为通过串口协议与 FPGA 进行通信，用户无需研究蓝牙相关协议与标准，只需要按照 UART 串口通信协议来处理发送与接收的数据即可。实验框图如下：



本实验通过**串口发送**与**串口接收**模块来完成与蓝牙模块的数据传输。通过**命令解析模块**及**命令响应模块**来实现简单的串口命令的解析控制以及命令的执行。FPGA 在接收到蓝牙模块传输进来的串口数据后，会将相应数据以及命令响应通过蓝牙模块发送给与之连接的通信设备，在这个过程中采用 FIFO 来存储我们需要发送的数据。

在这个设计中，我们在串口协议基础上自定义了若干控制命令，以便于在远端设备上可以通过蓝牙对 FPGA 平台上的逻辑、外设以及接口进行控制。例如在这里我们自定义了如下命令：

命令 1：*Naaaa

命令 2：*Waaaaaaaa

其中，星号（*）用作命令的起始位，N、W 等大写字母作为命令的名称，小写字母 a 表示任意一个十六进制数。命令 N 后接四位十六进制数即 16bit 数据，命令 W 后接八位十

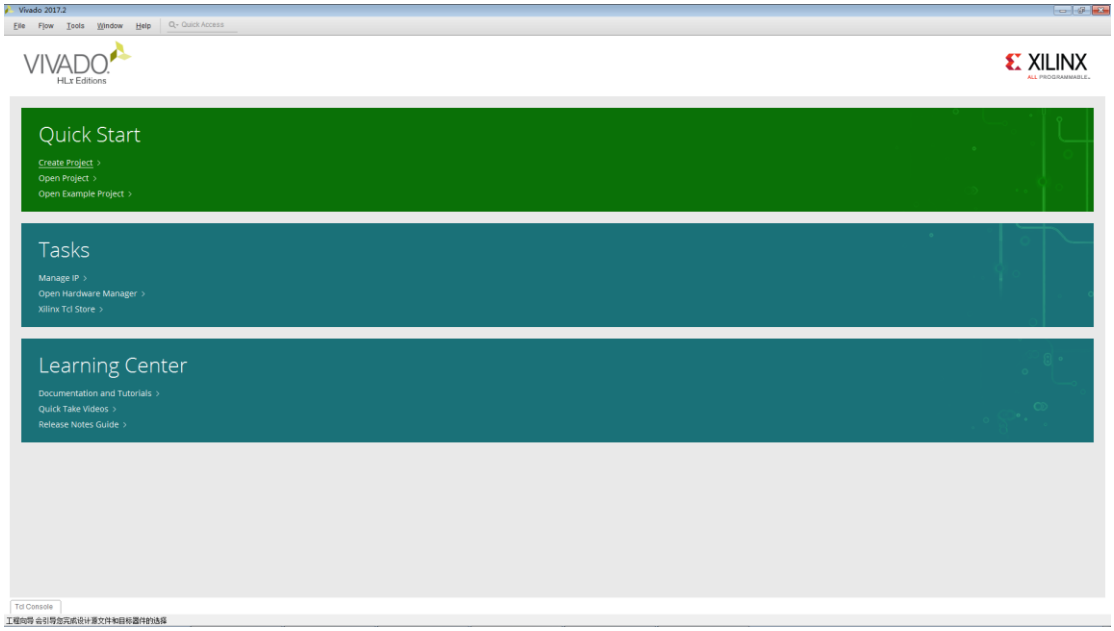
六进制数即 32bit 数据。

在实验中，我们定义命令 N 用于点亮板卡的 LED，其后接 16bit 数据分别对应板卡上 16 位 LED 灯；定义命令 W 用于控制七段数码管显示，其后接 32bit 数据分别对应板卡上 8 位七段数码管显示器的数值。用户可以自定义命令的功能，或将命令后接的数据用作其它用途。此外，用户还可以在此基础上定义新的命令。

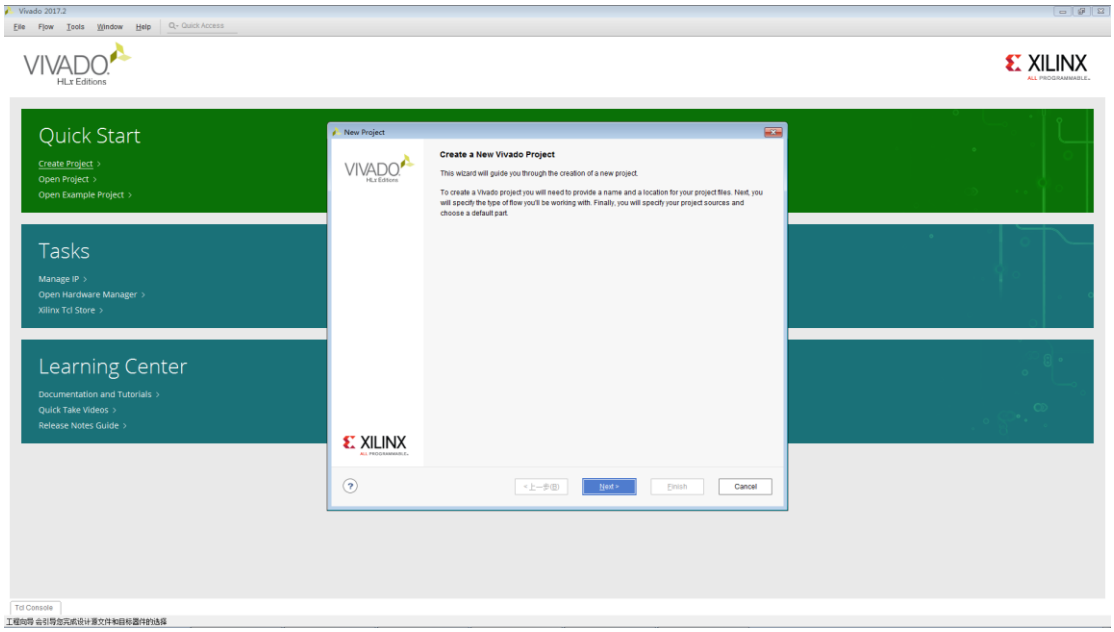
实验步骤：

一、创建工程

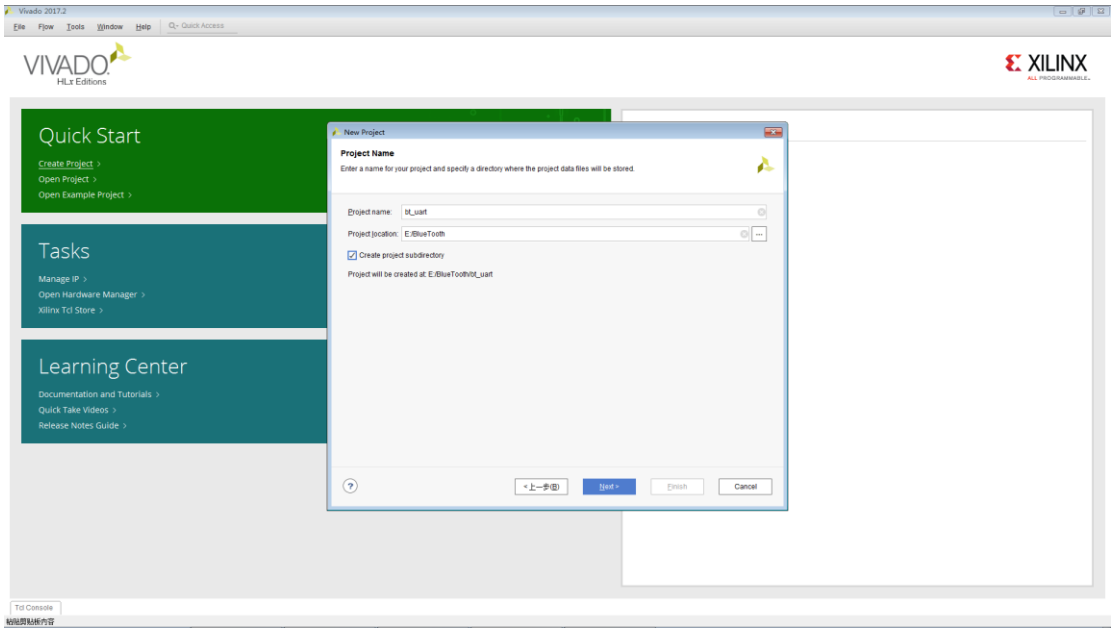
1、打开 Vivado 2017.2，选择“Create New Project”，创建一个新的工程。



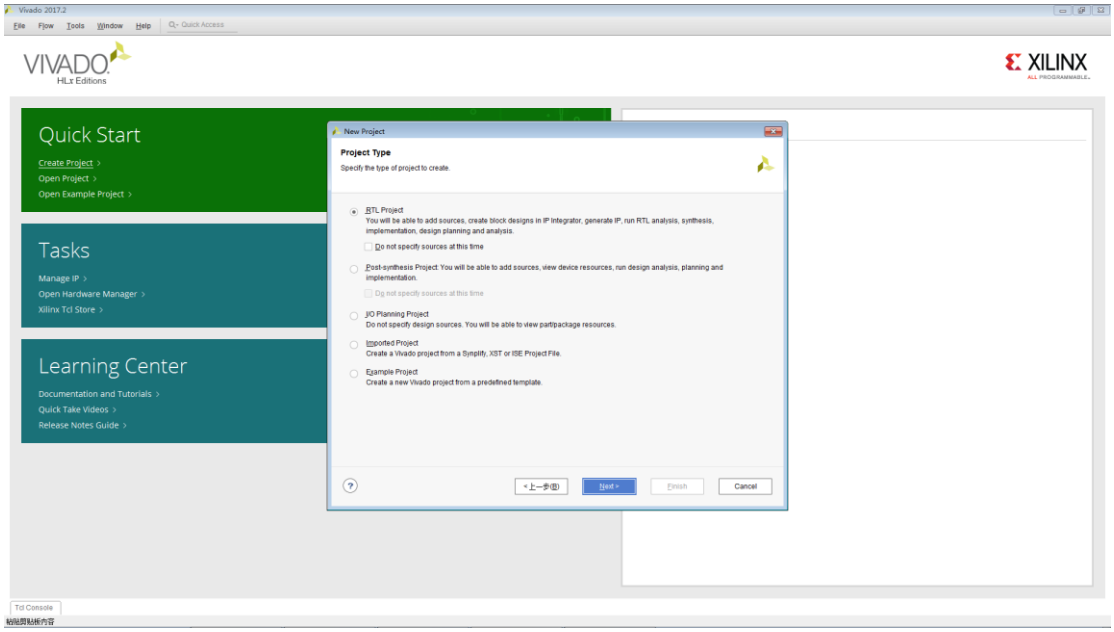
2、进入新建工程向导。



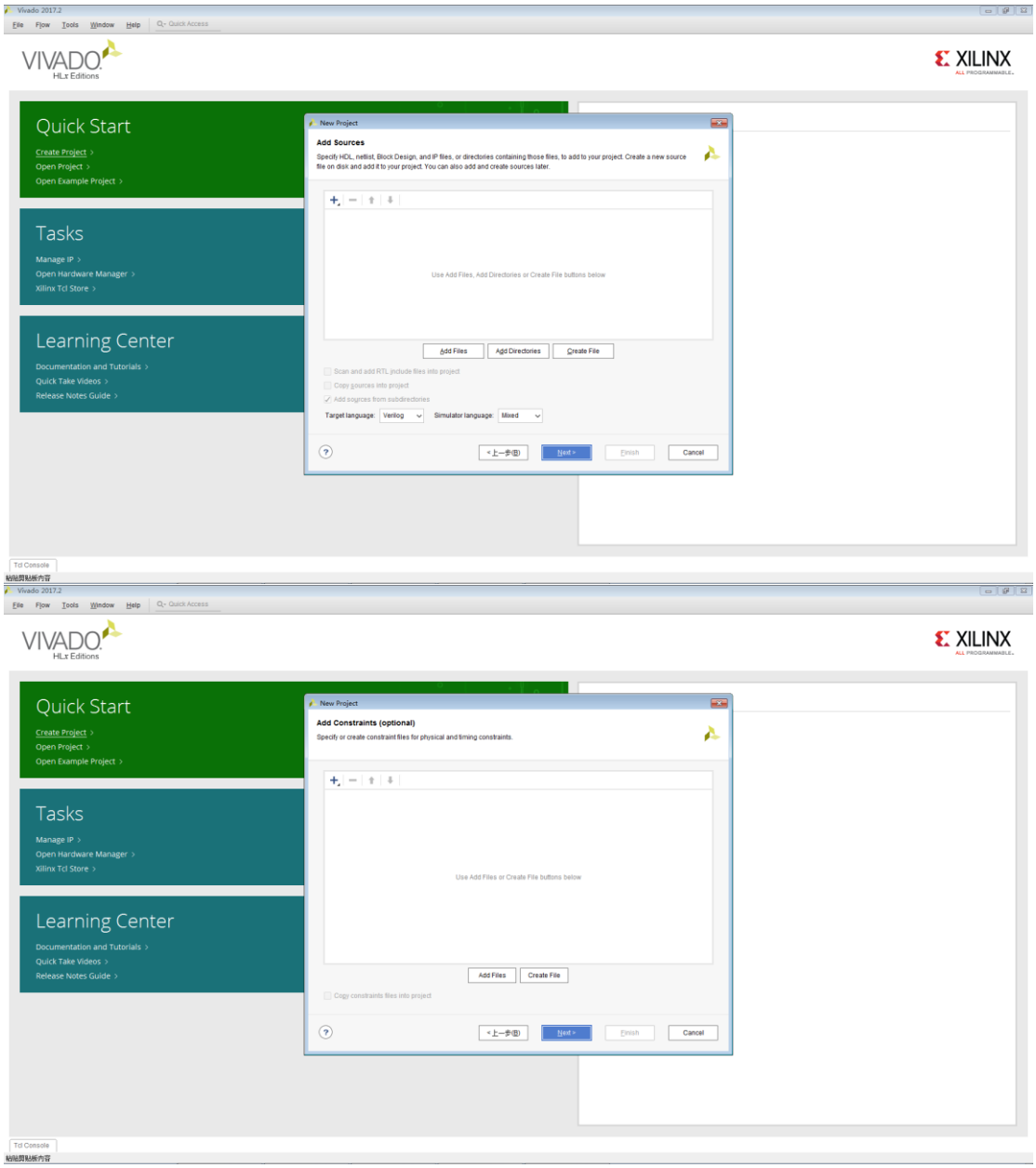
3、点击“Next”，输入工程名并指定工程所在的目录，确认勾选“Create project subdirectory”。注意工程名及工程所在的路径中只能包括数字、字母及下划线，不允许出现空格、汉字以及特殊字符等。这里我们将工程名命名为 bt_uart，工程目录选择 E:/BlueTooth。



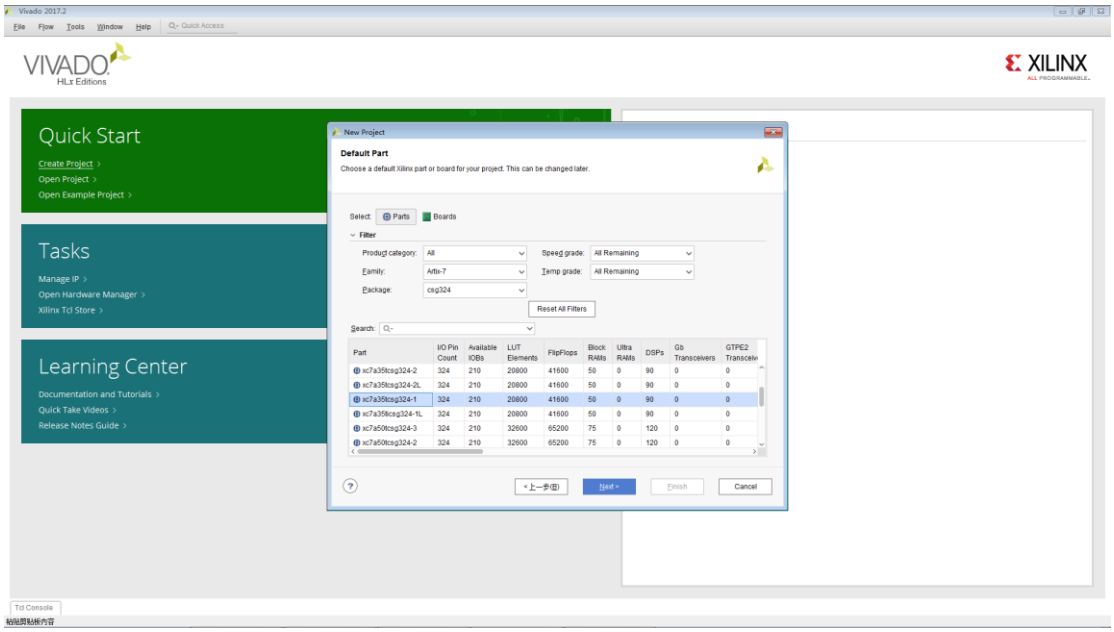
4、点击“Next”，指定创建的工程类型，选择“RTL Project”。



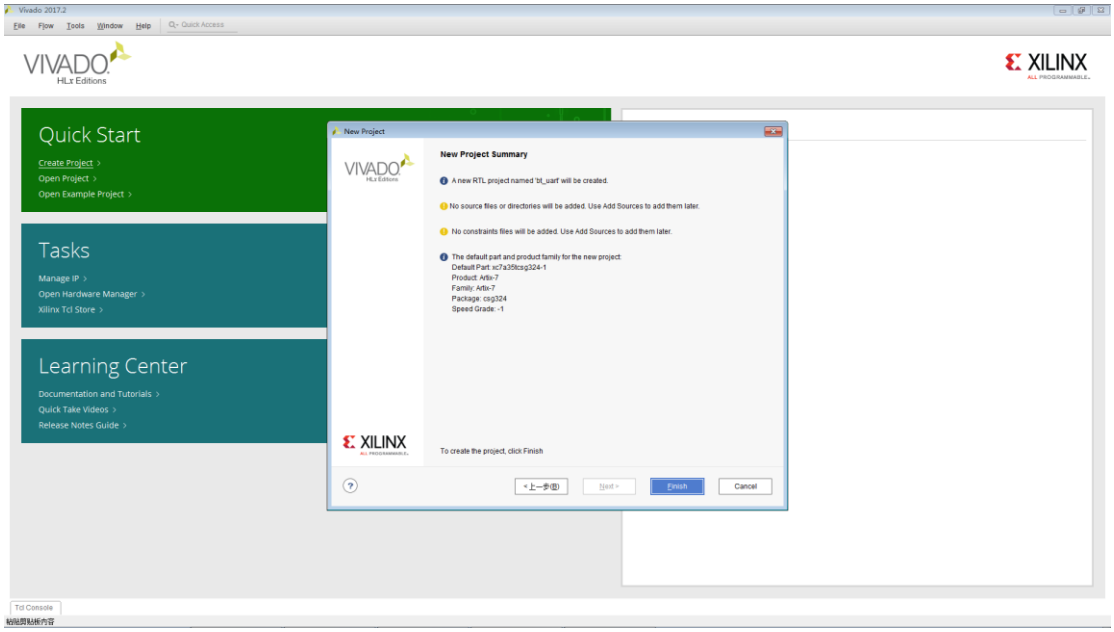
5、连续点击“Next”。



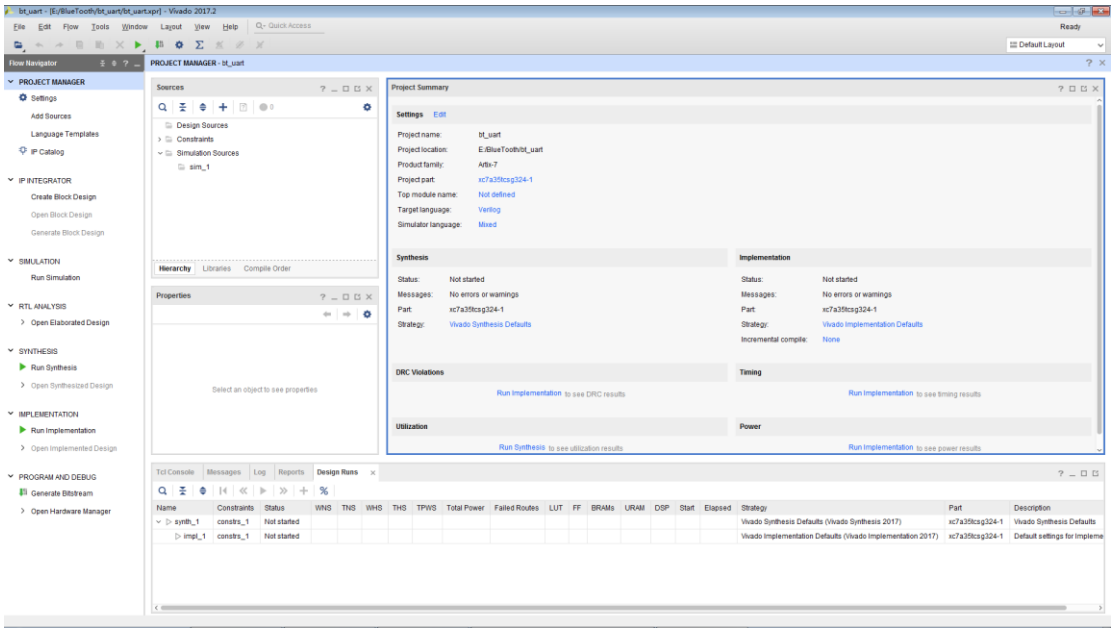
6、进入器件选择界面，通过下拉按钮选择器件的系列、封装形式，速度等级和温度等级，在符合条件的器件中选中板卡对应的 FPGA 芯片。EGo1 上的 FPGA 芯片型号为 XC7A35TCSG324-1。



7、点击“Next”，进入 Summary 界面查看所创建工程的相关信息。

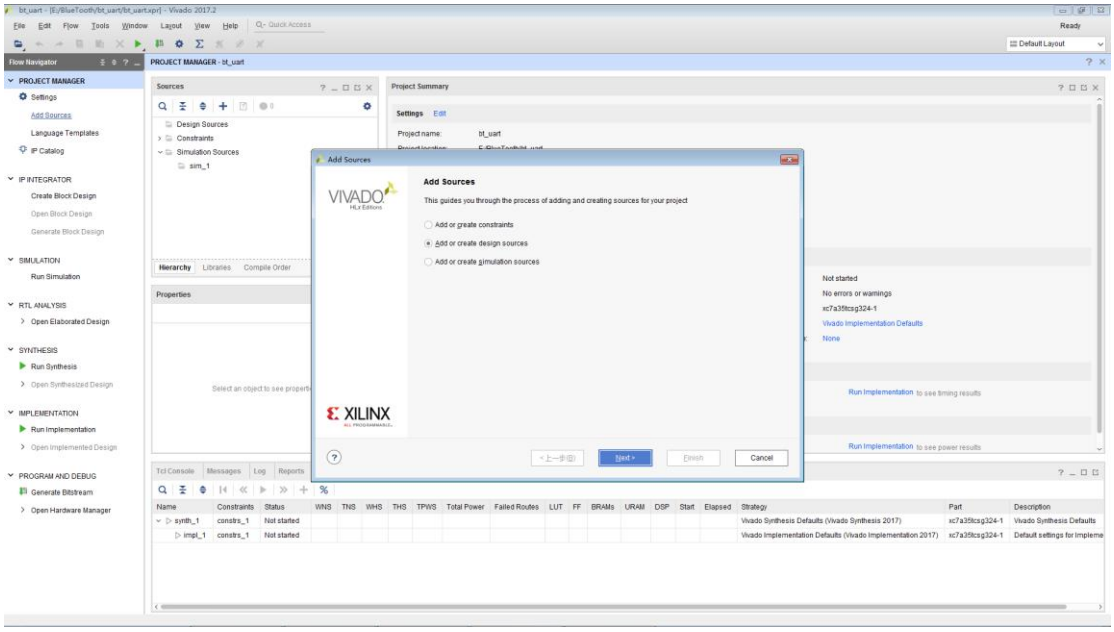


8、点击“Finish”，打开创建的工程。

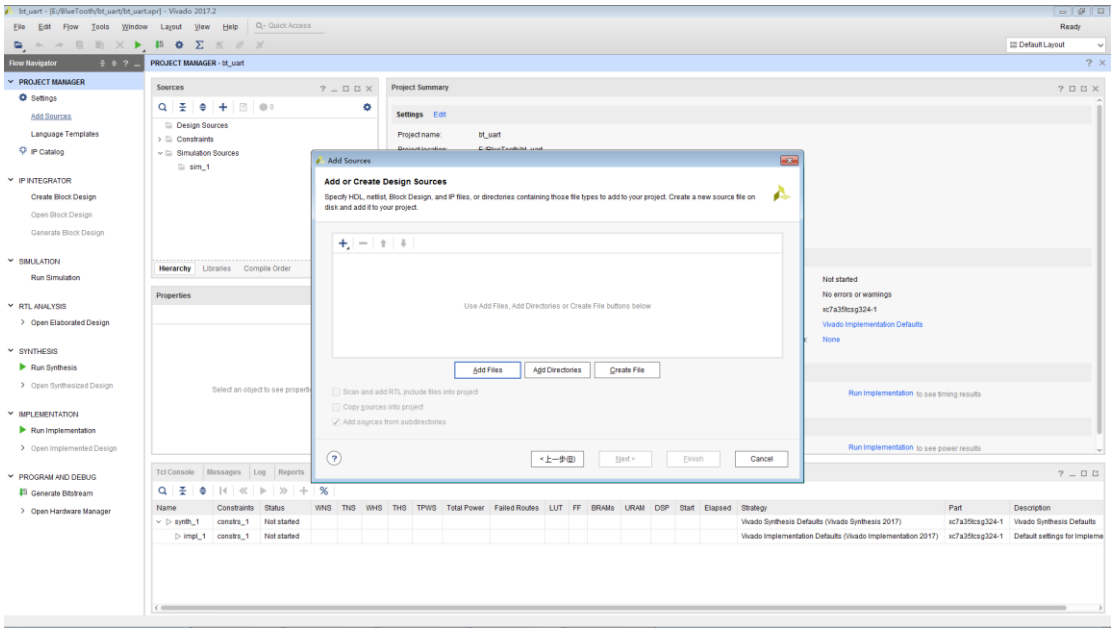


二、添加源文件

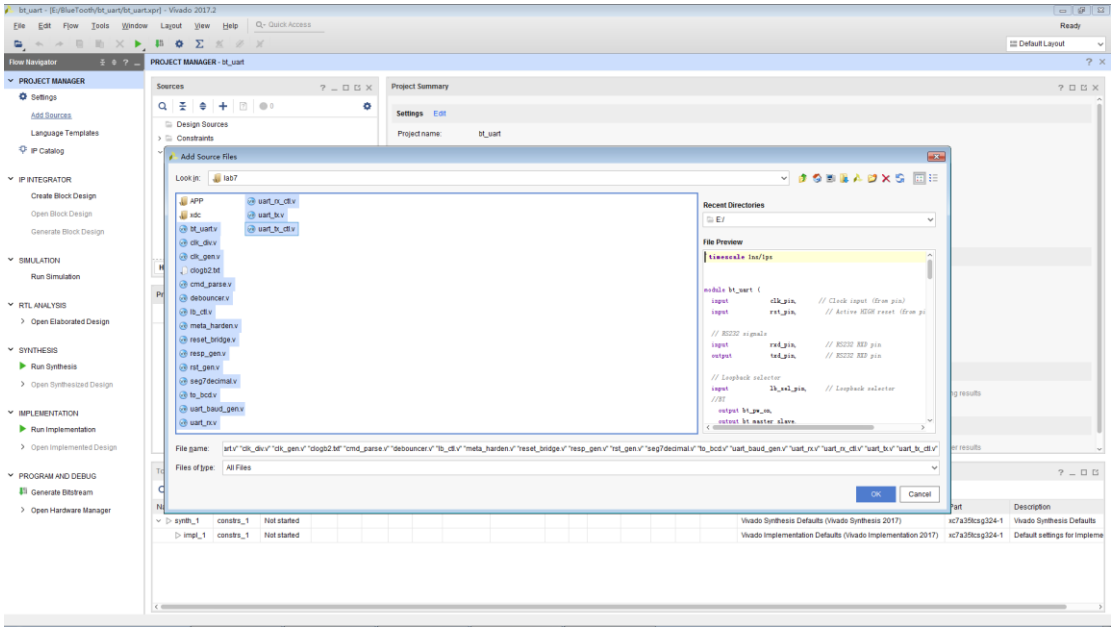
9、在左侧“Flow Navigator”栏中的“Project Manager”下点击“Add Sources”，在弹出的窗口中选择“Add or create design sources”，点击“Next”。



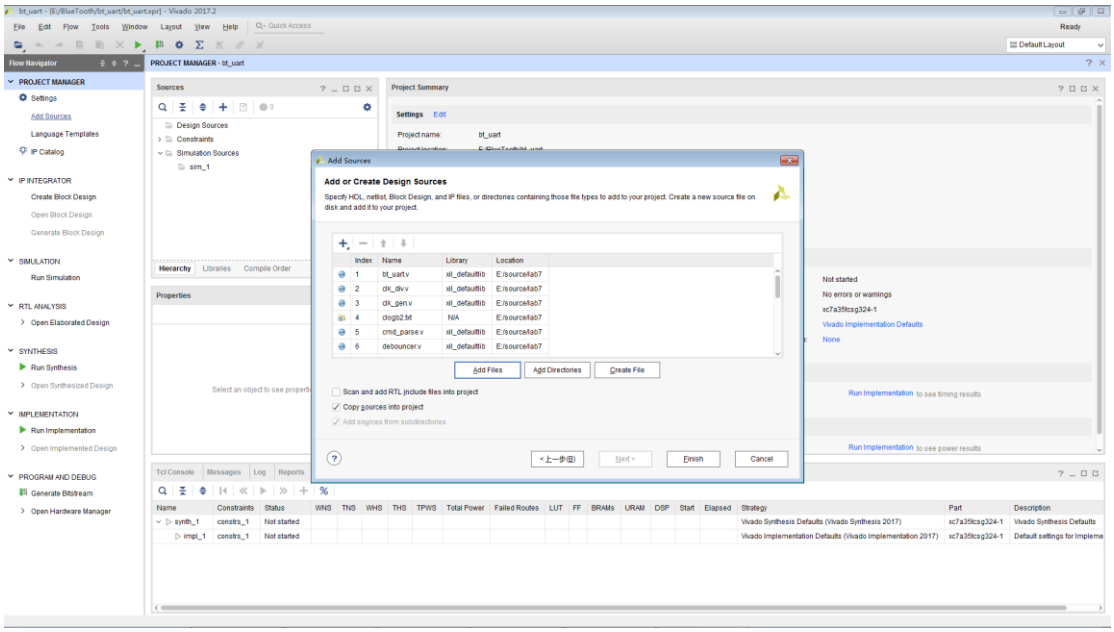
10、选择“Add Files”。



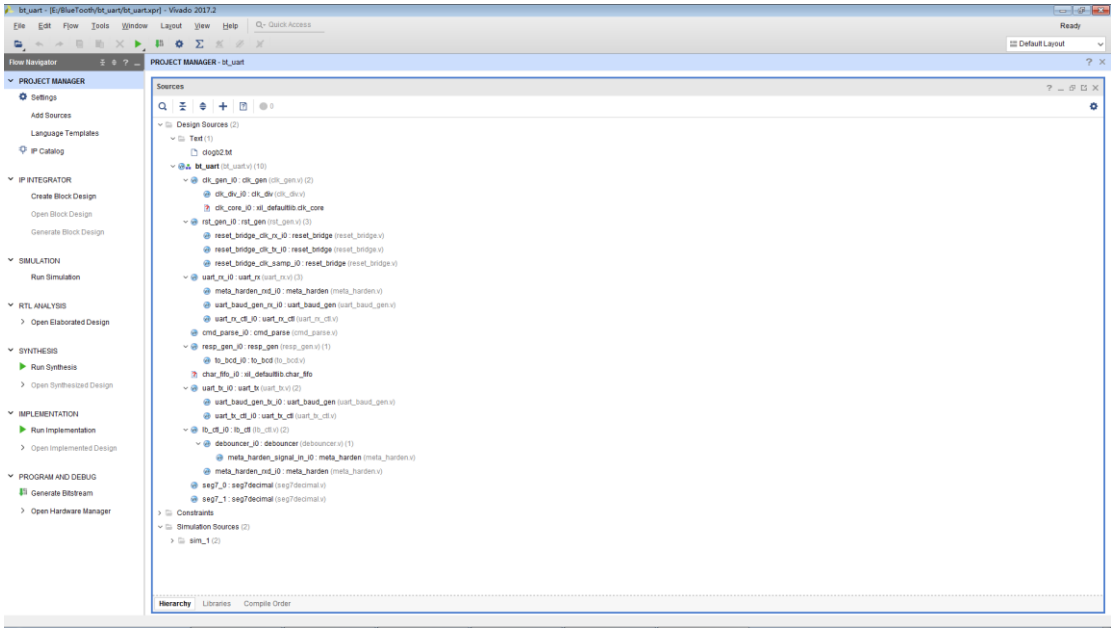
11、进入本实验源文件目录（E:\source\lab7），将 Files of type 设置为 All Files，选中列表中所有的文件，点击“OK”。



12、点击“Finish”。



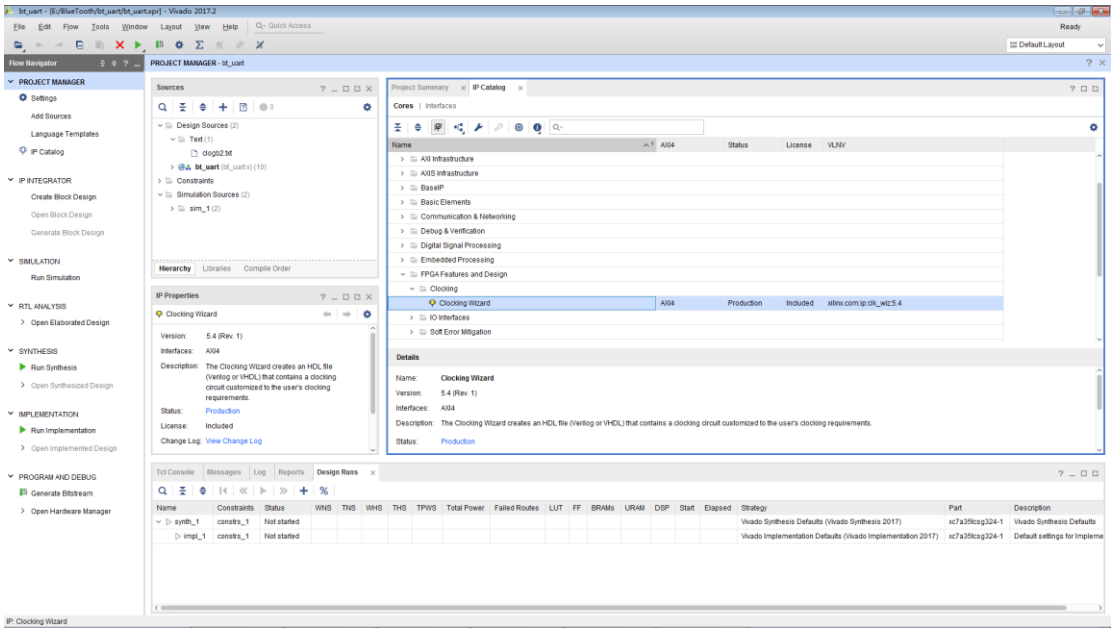
13、在 Sources 窗口可查看添加的源文件。



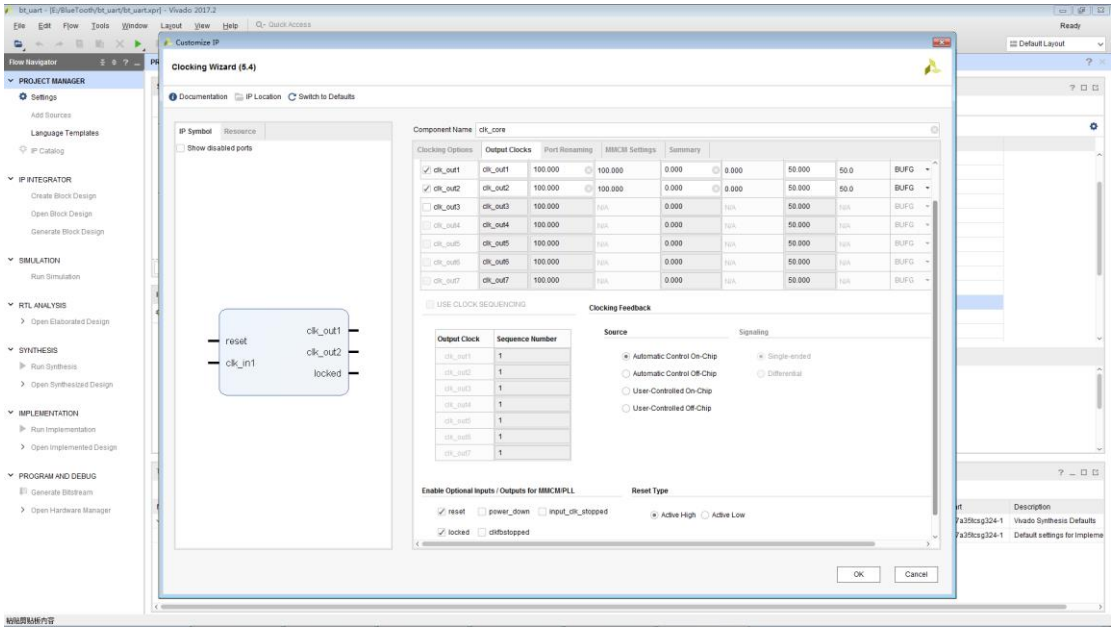
三、例化 IP 核

14、在上图可以看到有两个模块缺失，一个是时钟模块，另一个是 FIFO 模块。我们将通过例化 IP 核来完成这两个模块的设计。

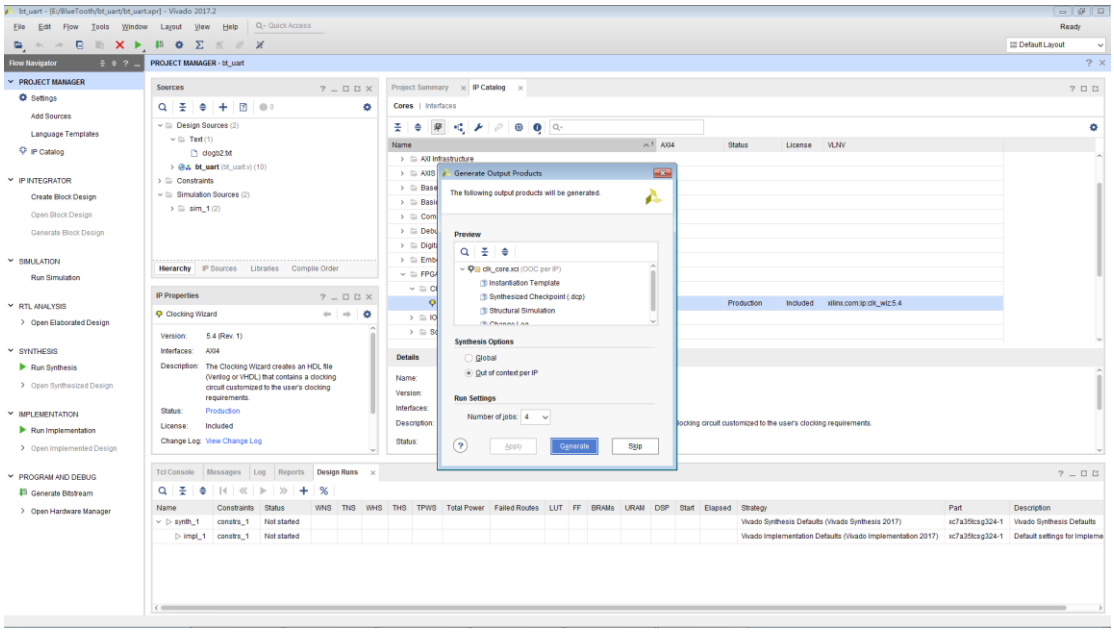
在左侧“Flow Navigator”栏中的“Project Manager”下点击“IP Catalog”。在界面右侧弹出的“IP Catalog”窗口中选择“FPGA Features and Design→Clocking→Clocking Wizard”。



15、双击打开“Clocking Wizard”，在“Customize IP”窗口中将 Clocking Wizard 的 Component Name 改为“clk_core”。在“Output Clocks”标签页下勾选 clk_out1、clk_out2，输出频率均为 100MHz。然后下拉到界面的底部，确认勾选 reset 和 locked 选项。

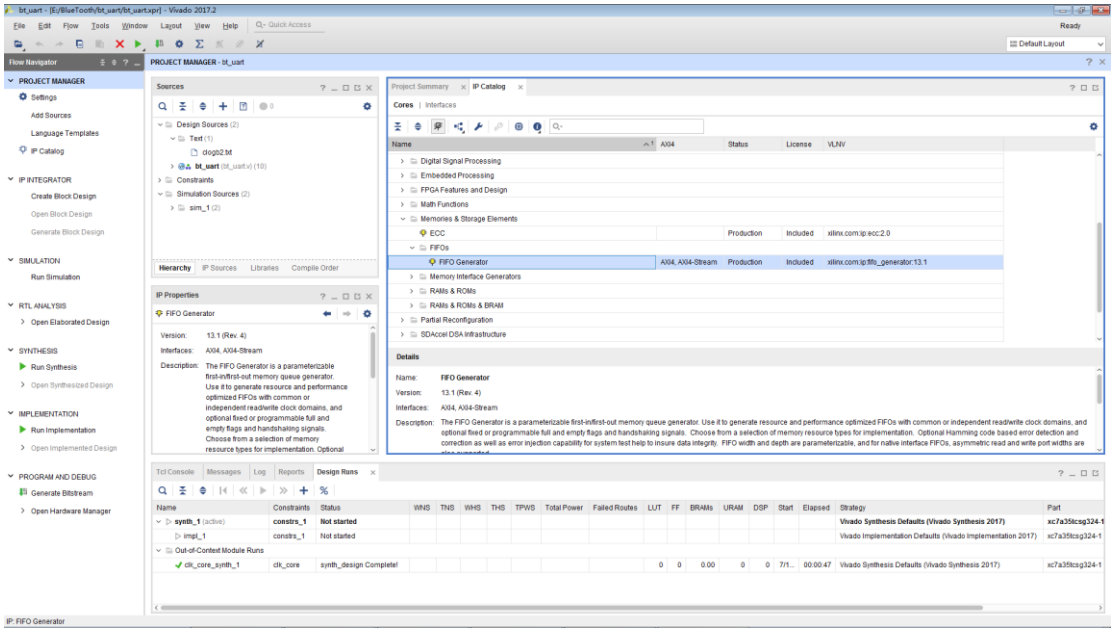


16、点击 OK，在弹出的窗口中选择 Out of context per IP，然后点击“Generate”。该过程结束后在弹出的窗口中点击 OK 确认。

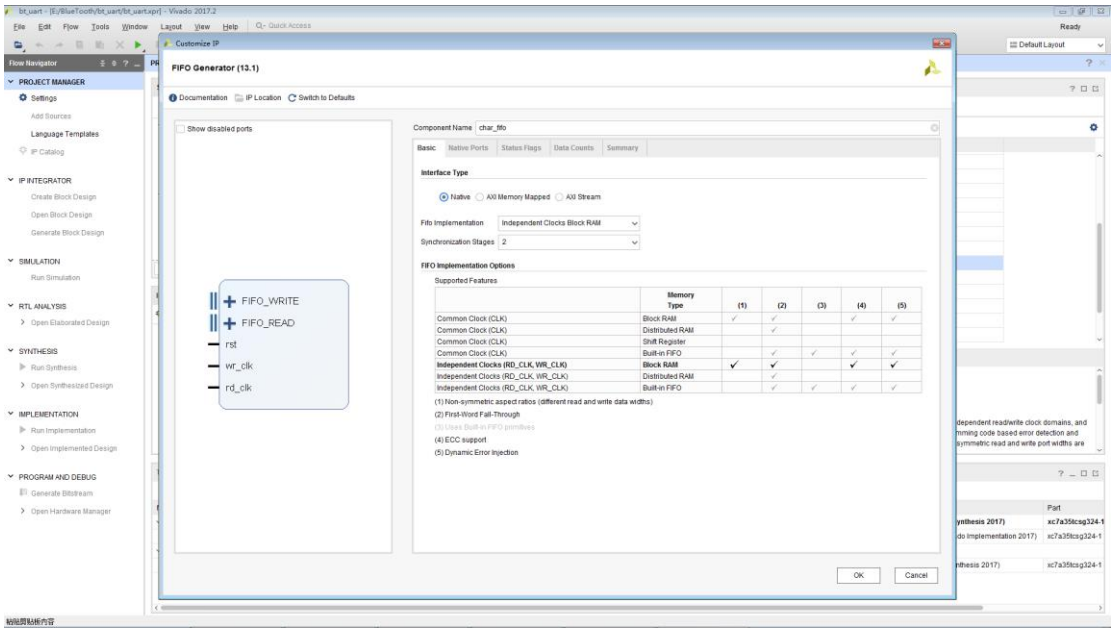


17、至此已经完成时钟 IP 核的例化，接下来我们通过例化 FIFO 的 IP 核并对其进行配置，用来作为串口发送数据的缓存。

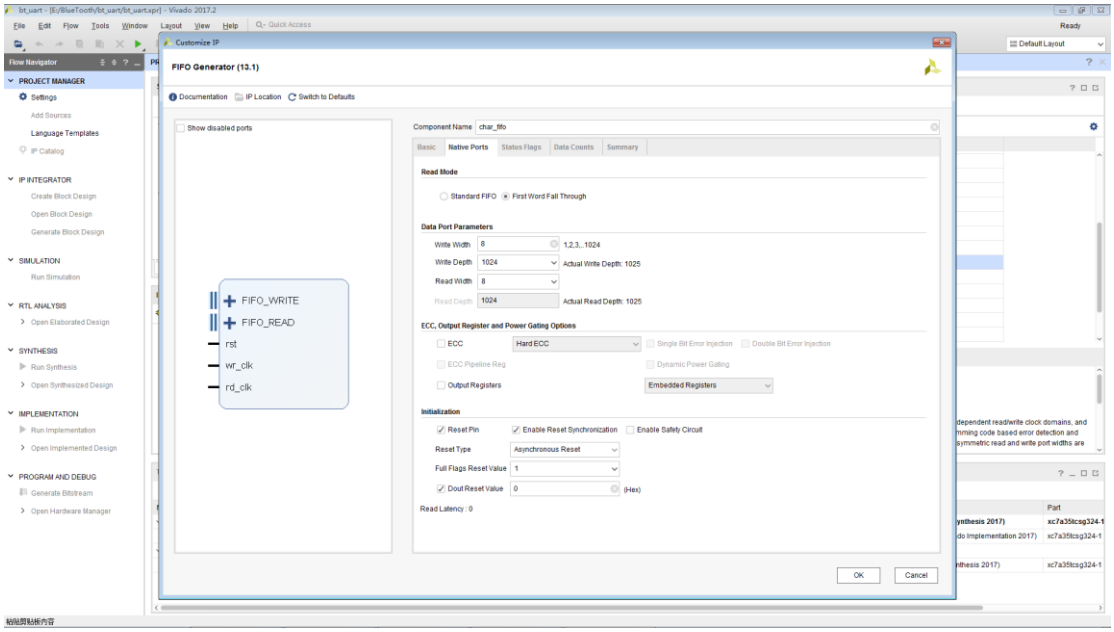
同样，在“IP Catalog”窗口中选择“Memories & Storage Elements→FIFOs→FIFO Generator”。



18、双击打开 FIFO Generator，在 Customize IP 窗口中，将 Component Name 改为 char_fifo。Basic 选项卡界面我们选择 Native 接口形式的 FIFO；FIFO 实现形式选择为 Independent Clocks Block RAM，即读写侧时钟分别为独立时钟，且使用 FPGA 内部的 BRAM 资源来构建的 FIFO；通常使用 Artix-7 以及 ZYNQ 系列器件时，系统时钟频率在 200MHz 以下时 Synchronization stages 选项推荐选择为“2”。

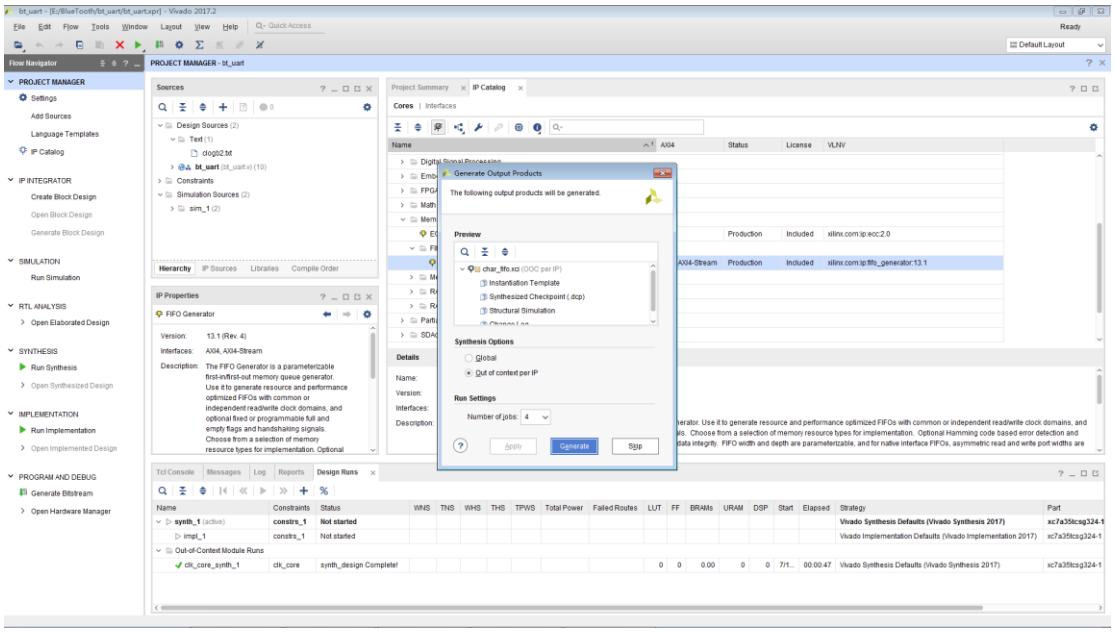


19、在 Native Ports 标签页中我们选择使用 First Word Fall Through 模式的 FIFO，即具有输出寄存功能的 FIFO 模块；FIFO 读写侧数据位宽均为 8bit，读写侧数据深度均为 1024；不使用 ECC 功能；勾选 FIFO 复位引脚 Reset Pin 以及复位同步 Reset Synchronization 选项；最后我们将复位情况下 FIFO 标志位以及 FIFO 数据输出默认值分别设置为“1”和“0”。



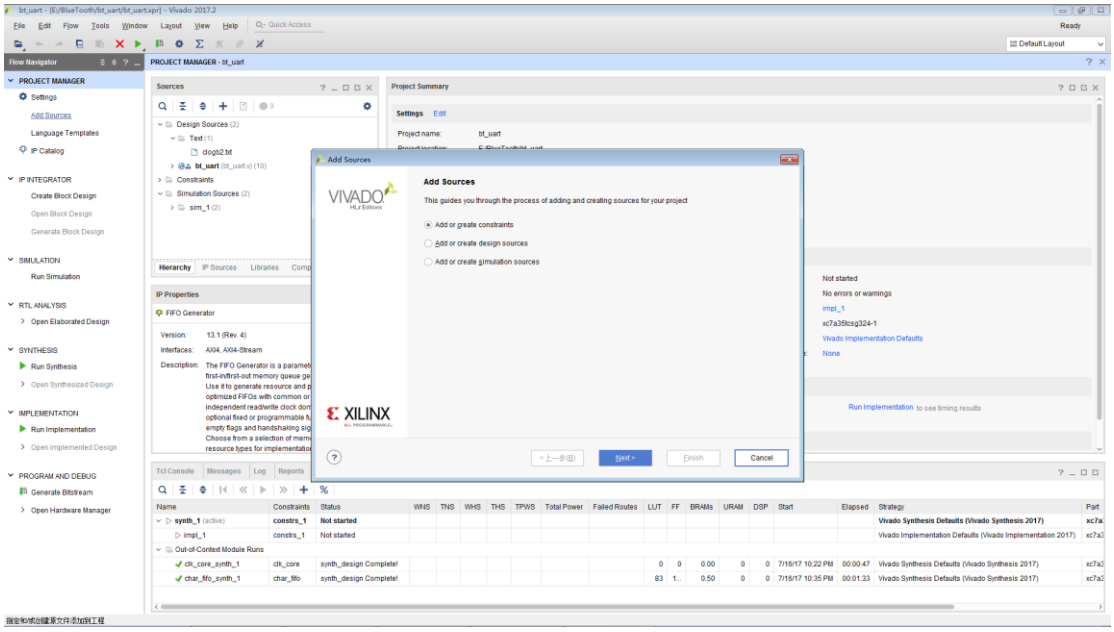
20、在 FIFO 配置界面的状态标志位 Status Flags 选项卡以及数据计数 Data Counts 选项卡中不勾选任何功能，在设计中不会使用到相关功能。

21、点击 OK，在弹出的窗口中选择 Out of context per IP，然后点击“Generate”。该过程结束后在弹出的窗口中点击 OK 确认。

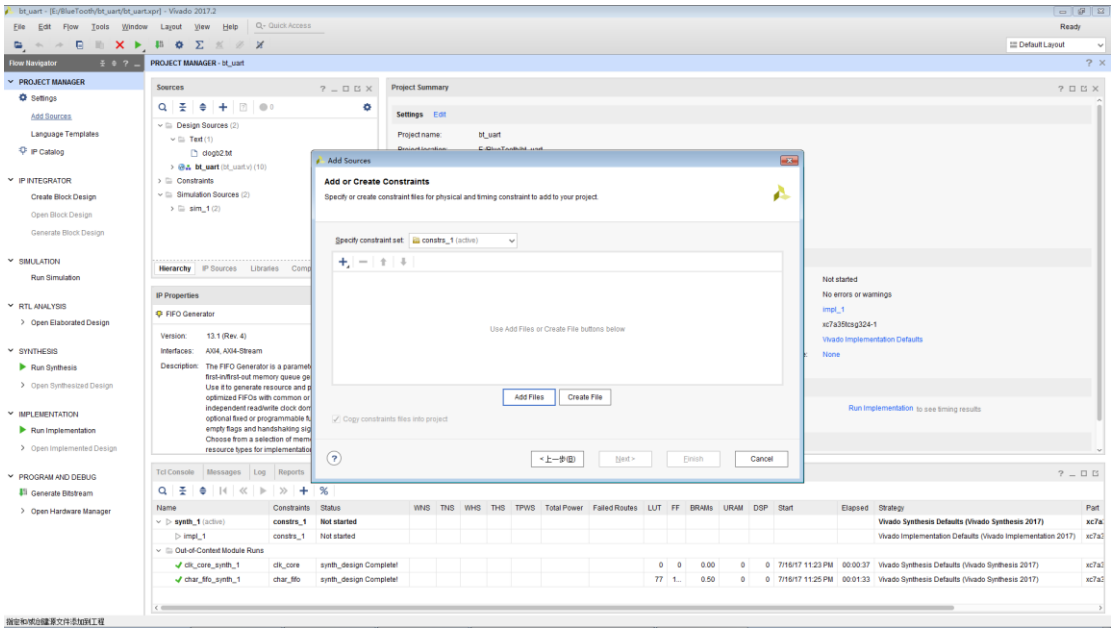


四、添加约束文件

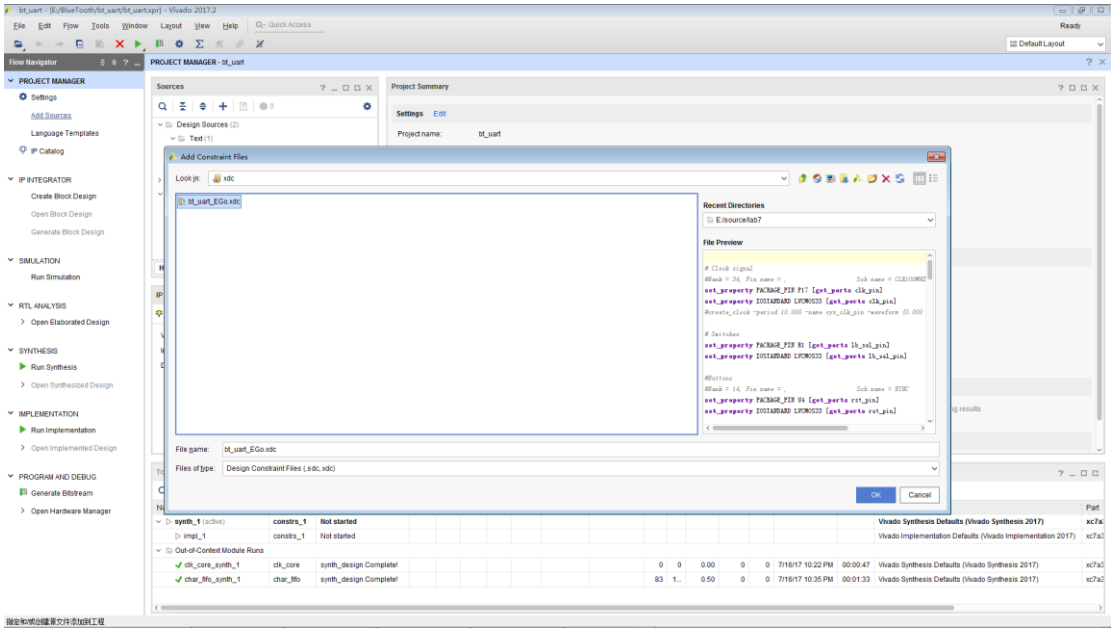
22、添加约束文件。在左侧“Flow Navigator”栏中的“Project Manager”下点击“Add Sources”，在弹出的窗口中选择“Add or create constraints”，点击“Next”。



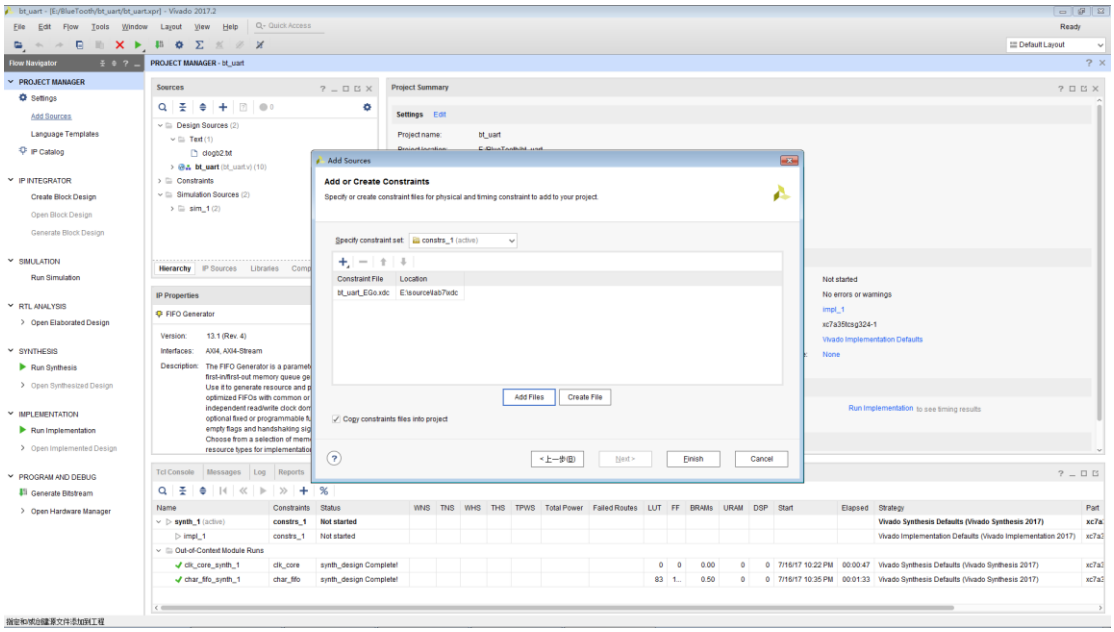
23、选择“Add Files”。



24、进入本实验约束文件目录（E:\source\lab7\xdc），选中 bt_uart_EGo.xdc 后点击 OK。

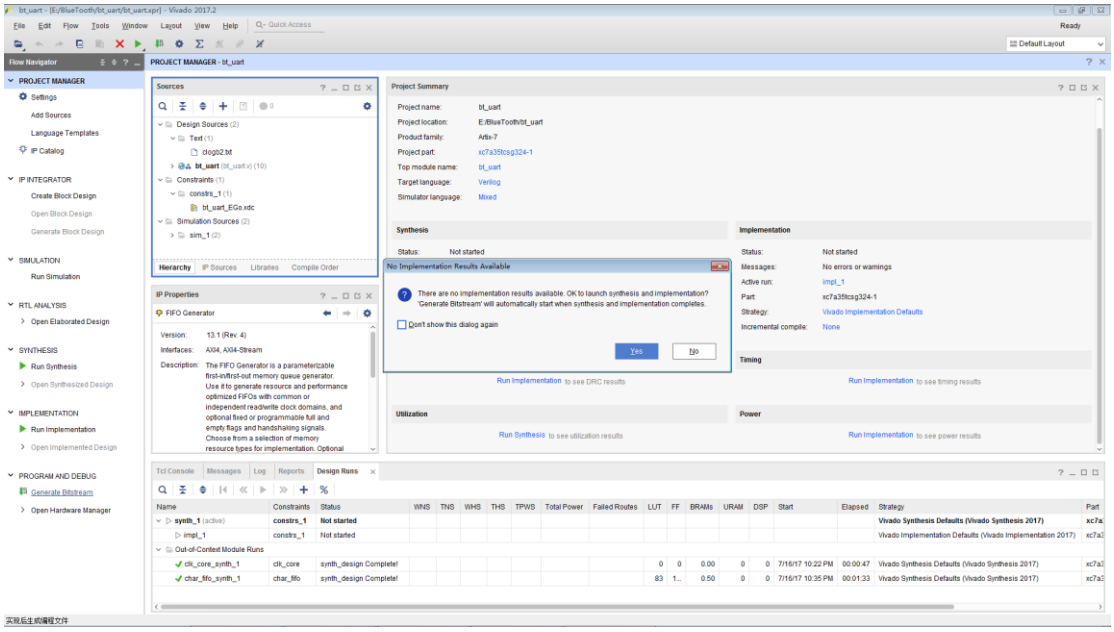


25、点击 Finish。

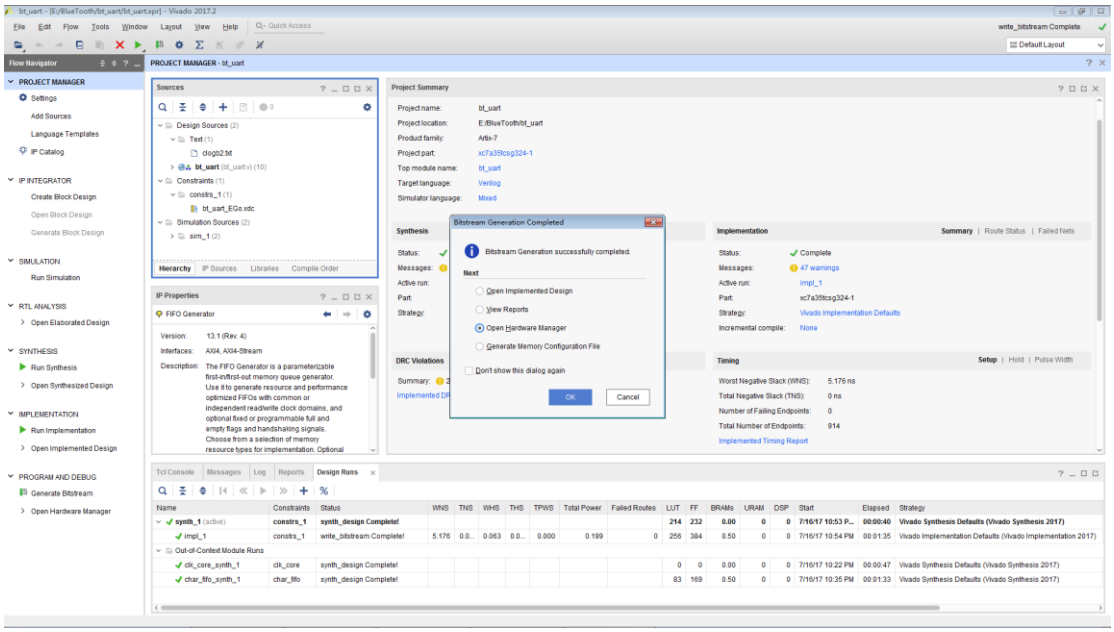


五、生成 bit 文件

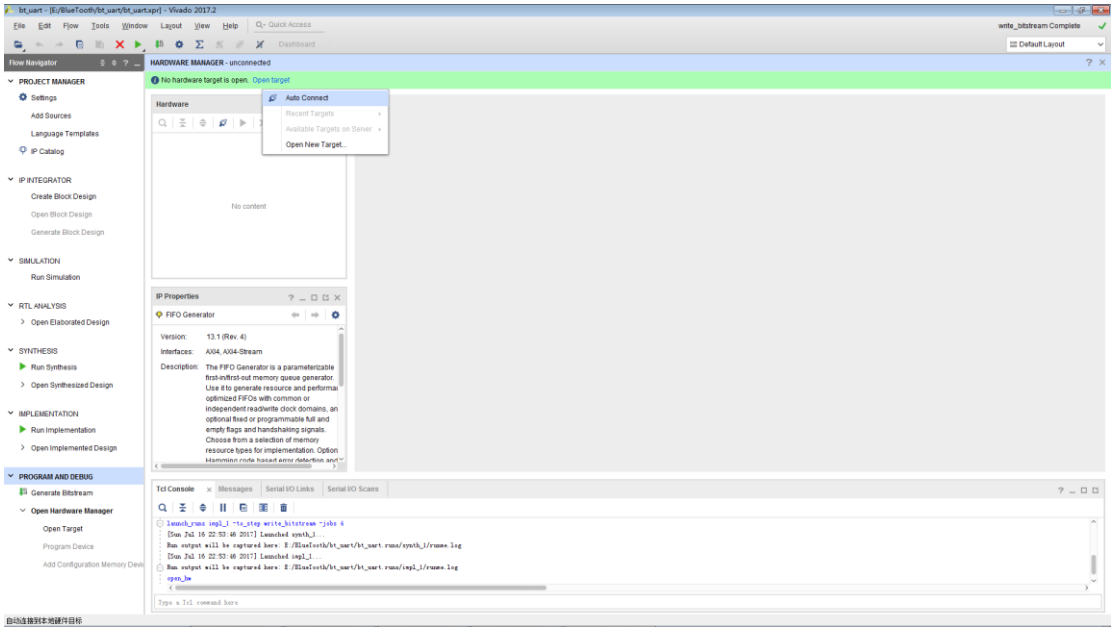
26、在“Flow Navigator”一栏中的“Program and Debug”下点击“Generate Bitstream”，此时会提示工程没有实现，点击“Yes”，会自动执行综合及实现过程。



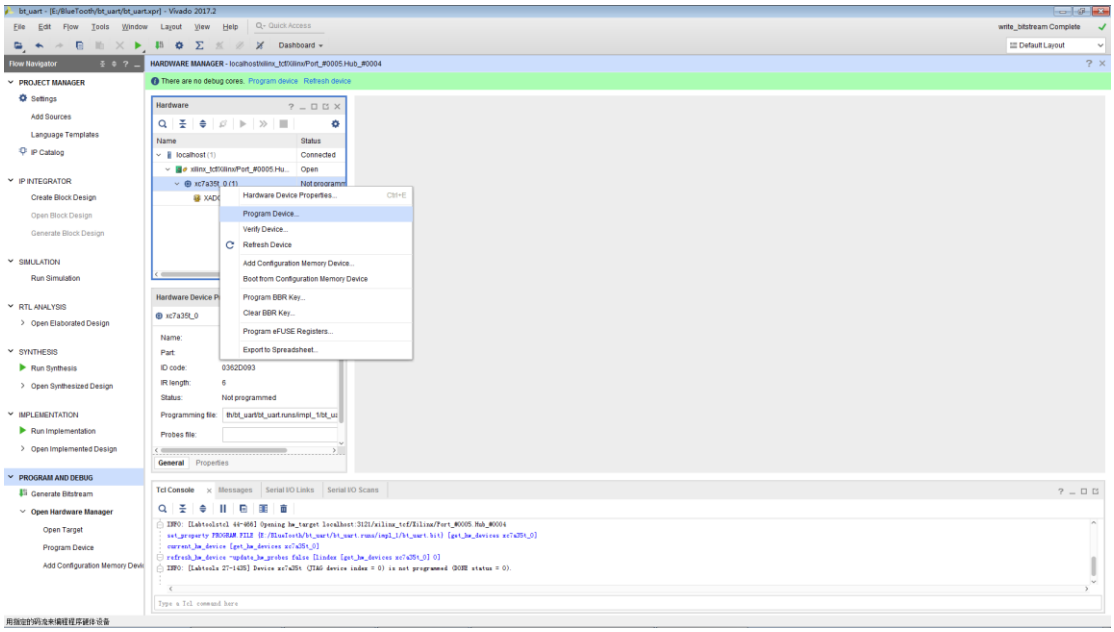
27、生成比特流文件完成后，选择“Open Hardware Manager”并点击“OK”。



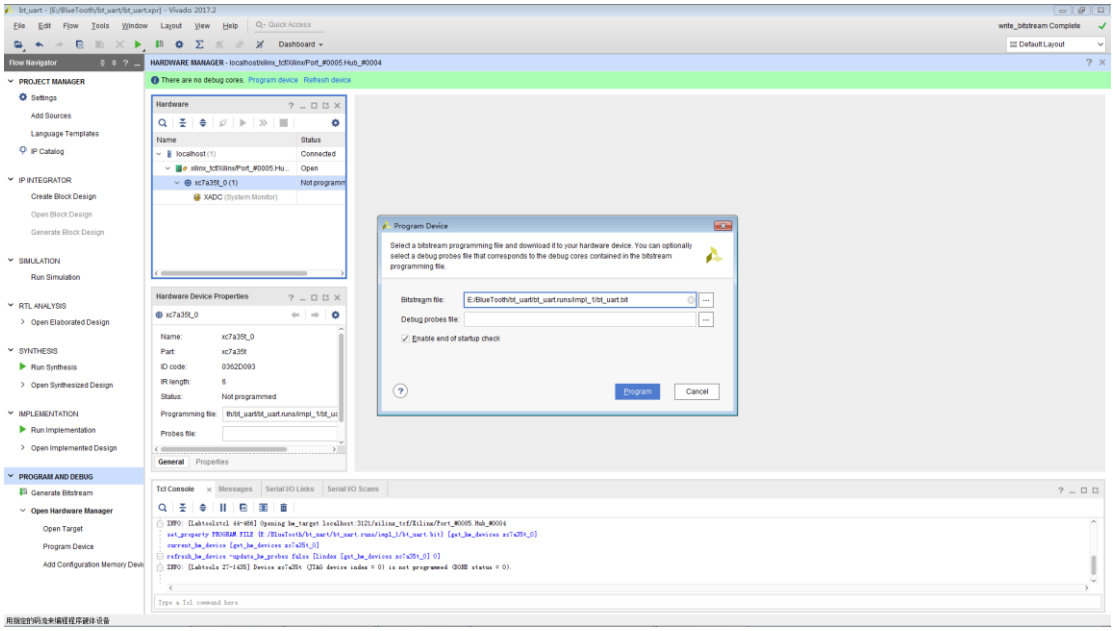
28、用 Micro USB 线连接电脑与板卡上的 JTAG 端口，并打开电源开关。在“Hardware Manager”界面点击“Open target”，选择“Auto Connect”。



29、连接成功后，在目标芯片上右击，选择“Program Device”。



30、在弹出的对话框中“Bitstream File”一栏中已经自动加载本工程生成的比特流文件，点击“Program”对 FPGA 芯片进行编程。



六、验证实验结果

31、比特流文件下载完成后，先将 SW1 设为低，SW0、SW2、SW3、SW4 设为高。通过 SW2 将蓝牙模块进行复位（拉低再拉高），此时蓝牙处于 slave 模式，蓝牙状态指示灯 LED2 闪烁较慢。

32、在安卓环境下安装 BLE 蓝牙串口终端 APP，并打开 APP，连接实验平台上的蓝牙模块，当 LED2 常亮时，表示连接成功。

33、通过在 APP 中输入对应的命令来完成与实验平台的交互，如图所示。

安卓端蓝牙：EGo1_BT_Tool

APP 命令：

*Nxxxx:控制 LED 灯

*Wxxxx_xxxx:控制七段数码管