实验三 分秒数字钟的设计与实现

一． 实验目的

1. 掌握基于 SystemVerilog HDL 的时序逻辑电路建模方法；
2. 掌握计数器设计方法，并能够使用计数器设计使能时钟（用于时钟分频）；
3. 掌握移位寄存器设计方法，并能够利用移位寄存器设计边沿检测电路；
4. 掌握 7 段数码管的动态显示。

二． 实验环境

1. 操作系统：Windows 10 或 Ubuntu 16.04
2. 开发环境：Xilinx Vivado 2018.2
3. 硬件平台：Nexys4 DDR FPGA 开发板

三． 实验原理

1. 使能时钟（时钟分频）

在时序逻辑电路中，时间的计算都要以时钟作为基本的单元。一般而言，在数字系统设计中只有一个基准的系统时钟，故经常需要对基准时钟进行处理而得到各模块所需的时钟频率。获取比基准时钟频率更慢的时钟称为时钟分频，例如将 100MHz 时钟转换为 25MHz 时钟。通常，时钟分频采用**计数器**对基准高频时钟进行计数，每隔固定计数值将输出时钟进行翻转从而形成周期性变化的低频时钟。图 3-1 给出了 4 分频的波形图。

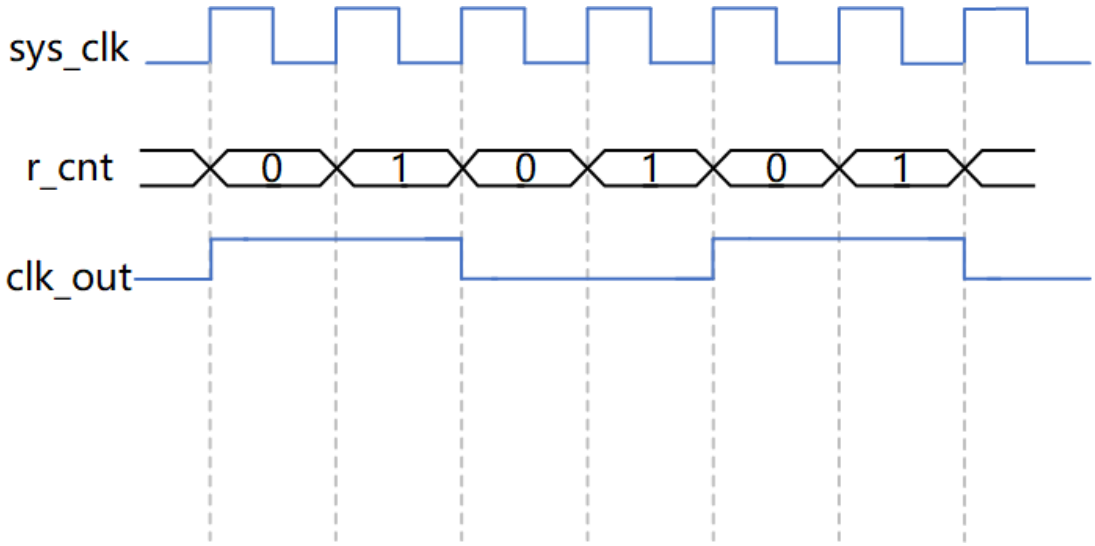


图 3-1 四分频波形图

假设输入的高频时钟信号 clk_in 的频率为 f_i（周期 $T_i = 1/f_i$），计数值为 N，则输出的低频时钟信号 clk_out 的频率 $f_o = f_i/N$。由于输出频率是输入频率的 1/N 倍，所以称为 **N 分频器**。例如，输入时钟信号 clk_in 的频率 f_i = 25MHz，如果要得到 2.5MHz 的输出信号，则计数值 N 为 10。

但是，所生成的分频时钟如果频率仍然很高，或作为芯片内部其它子模块的时钟信号使用，是不合适的。因为，时序逻辑电路通常是同步设计，这就要芯片内部（顶层模块）中所有寄存器均采用同一个时钟信号，显然在芯片内部生成多个分频时钟会造成“时钟漫天飞”的情况，不符合同步时序设计的理念，不利于 EDA 工具进行时序分析，影响系统性能和稳定性，更严重的可能导致系统错误。

因此，为了实现分频效果，同时满足同步设计要求，可以使用使能时钟代替分频时钟。所谓“**使能时钟**”，本质上还是时钟分频器，**它的工作原理是根据计数值按固定时间间隔产生一个周期的高电平使能信号**。图 3-2 给出了满足 4 分频需求的使能时钟波形图。从图中可以看出，每隔 4 个系统时钟（sys_clk）周期，即计数值 r_cnt 达到 3，则使能时钟信号 clk_flag 被拉高一个时钟周期，满足四分频的需求。同时，整个系统中只有一个时钟 sys_clk，也满足同步设计要求。**因此**

在实际工程设计中，均使用使能时钟代替时钟分频器。

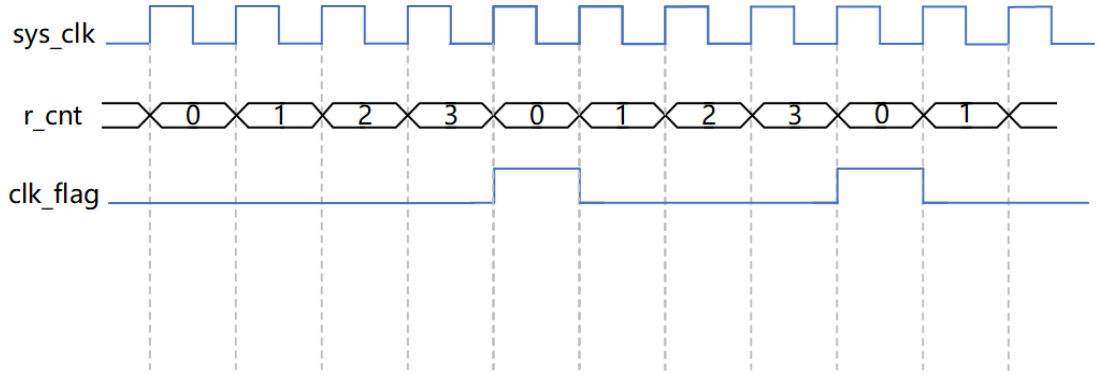
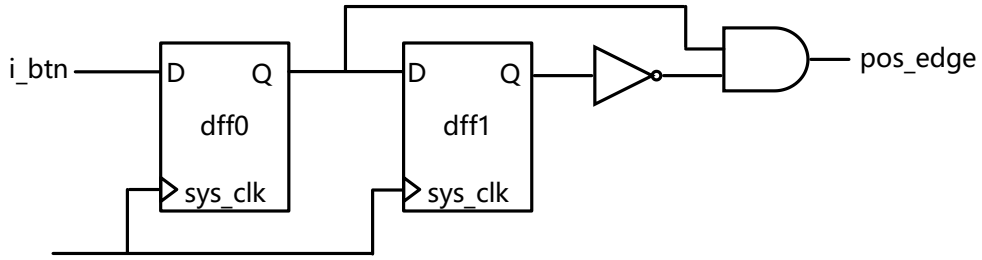


图 3-2 使能时钟波形图（4 分频）

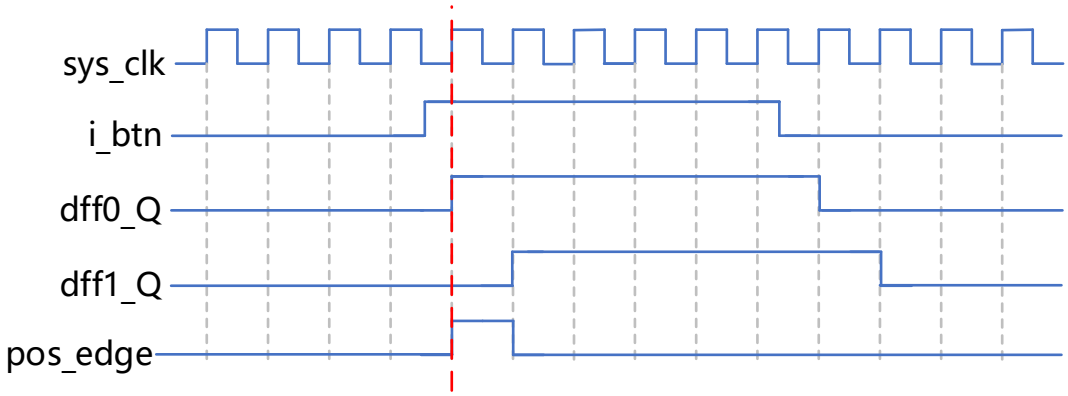
2. 基于移位寄存器的边沿检测电路

边沿检测电路是非常常用的一种数字电路。所谓**边沿检测**，就是检测外部输入信号或内部逻辑信号的跳变，即实现上升沿或下降沿的检测，捕获得到边沿使能（通常为 1 个周期高电平）来作为其他时许逻辑的触发信号。边沿检测是数字逻辑电路设计常用的方法。

边沿检测电路的实现原理十分简单，**如果判断某信号在上一个时刻为低电平，而当前时刻为高电平，则显而易见待检测信号出现了上升沿**；**反之，当判断某信号在上一个时刻为高电平，而当前时刻为低电平，则待检测信号出现了下降沿**。因此，边沿检测电路通常使用**移位寄存器**来实现。图 3-3 给出了基于 2 位移位寄存器的上升沿检测电路和波形图。



(a) 原理图



(b) 波形图

图 3-3 边沿检测（上升沿检测）电路

从上图可以看出，经过两级触发器缓存后，当输入信号 i_btn 出现时钟上升沿，则输出信号 pos_edge 出现 1 个周期的高电平。从原理上来看 1 级触发器就可以实现边沿检测了，而之所以使用两级是因为如果输入信号来自于模块外部，则该信号与模块内部不在同一个时钟域，属于异步信号，容易造成亚稳态。因此，使用两级移位寄存器不仅能够实现边沿检测功能，同时还起到了**同步器**的作用，提高了系统的稳定性。

3. 七段数码管的动态显示

一个系统中通常会同时使用多个七段数码管，图 3-4 所示。为了节省端口数，所有七段数码管的数据段 CA~CG 都是共享的，通过使能端口 AN0~AN7 控制将哪个数码管点亮。因此，当 CA~CG 给定一组输入编码时，8 个七段都显示同一个数字，这种显示方式称为七段数码管的静态显示。但更为常见的应用是需要七段数码管显示不同的数字，如显示“12345678”，那应该如何实现呢？这就需要使用**七段数码管的动态显示**了。

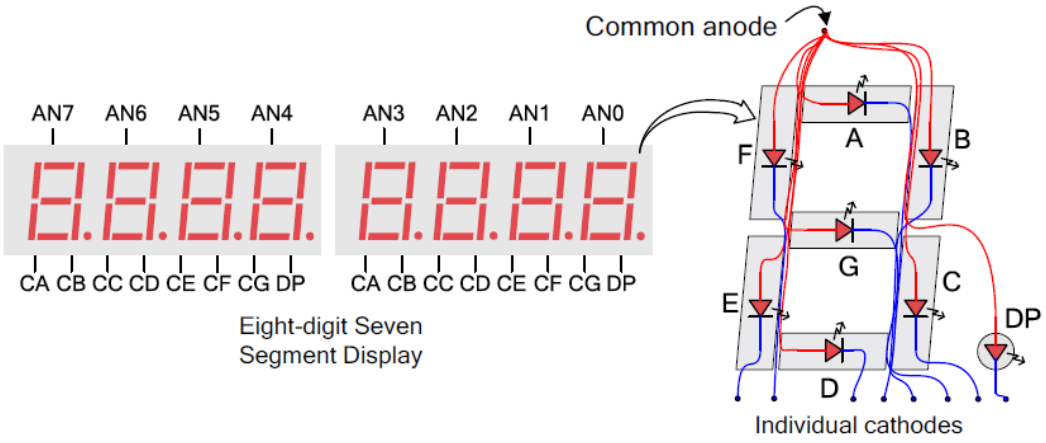


图 3-4 八个七段数码管

七段数码管的动态显示需要利用其“余晖效应”和人眼的视觉暂留现象。即七段数码管的亮/灭不是瞬时完成的，而是需要经过一段时间。利用这一特性，我们可以设计一个电路，分时轮流控制各个数码管的使能端，使各个数码管轮流显示不同数字。在轮流显示的过程中，首先，通过 CA~CG 送入一组编码，并点亮第一个数码管，然后就灭掉，接着送入一组新的编码，并点亮第二个数码管，然后再灭掉，依次循环下去，如图 3-5 所示。每个数码管的点亮时间可控制在几个毫秒。由于余晖效应，尽管各个数码管并非同时点亮，但只要扫描的速度足够快，给人的感觉就是一组稳定的显示数字，不会有闪烁感，实现不同数字的显示。这种用于控制七段数码管动态显示的电路称为**动态扫描电路**。

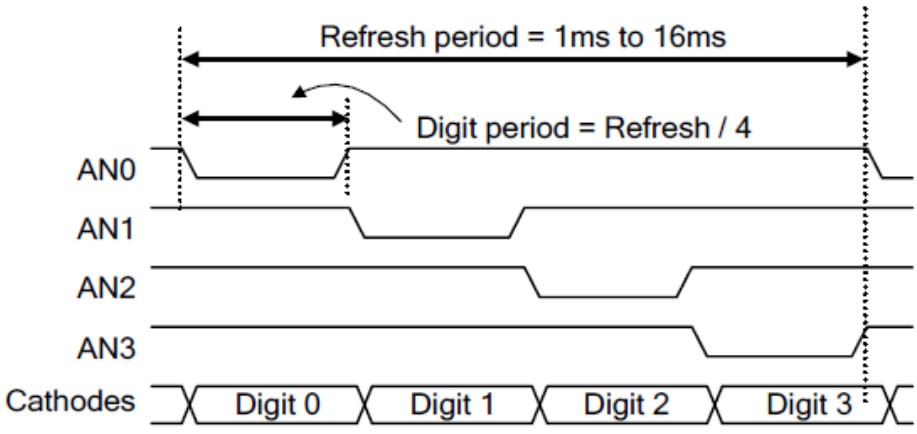


图 3-5 七段数码管动态显示示意图（每个数码管显示 1~4ms）

四． 实验内容

基于 SystemVerilog HDL 设计并实现一个分秒数字钟。整个工程的顶层模块如图 3-6 所示，输入/输出端口如表 3-1 所示。使用 **4 个七段数码**管显示当前的计时。其中，两个数码管对应“分”，另两个数码管对应“秒”。通过 **1 个拨动开关**对数字钟进行复位控制。使用 **1 个按键**对数字中进行“暂停/计时”控制，**按键每按下一次，进行暂停和计时的切换**，即暂停时，按下按键启动计时；计时过程中，按下按键暂停计时。

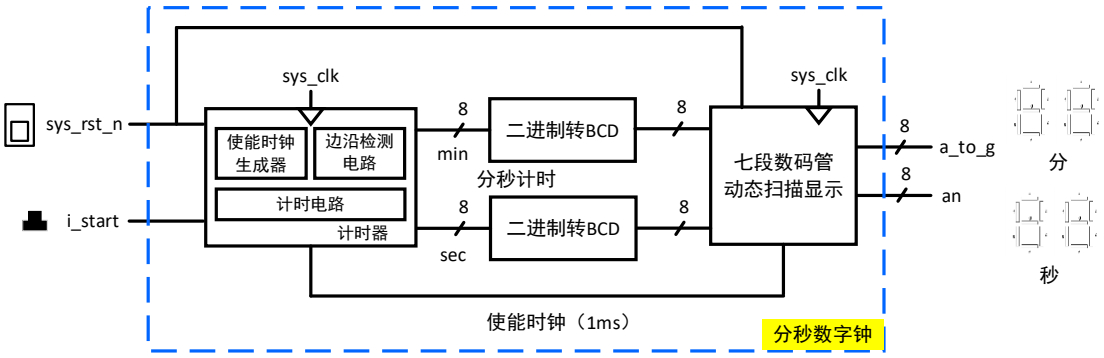


图 3-6 分秒数字钟顶层模块

分秒数字钟由 3 部分构成。

- 第一部分是数字钟的核心——**计时器模块**，该模块又由 3 个子模块构成，分别是计时电路、使能时钟生成电路和边沿检测电路。
  - ✧ 计时电路通过计数器实现计时功能，产生**二进制**的“分”（min）和“秒”（sec）输出。
  - ✧ 使能时钟生成电路用于产生控制七段数码管动态显示的使能时钟，使能时钟高电平出现的周期为 1ms。
  - ✧ 边沿检测电路模块对按键输入进行上升沿检测，产生控制计时器暂停和启动的信号。
- 第二部分是 **8 位二进制转 BCD 模块**，它将二进制的“分”、“秒”值转化为

BCD 编码，即 10 进制数。其中，分、秒各使用一个。**本实验中，该模块不需要实现，由教师直接提供 IP 使用**。

- 第三部分是 **7 段数码管动态扫描显示模块**，它实现“分”、“秒”值的最终显示。“分”、“秒”值各使用两个 7 段数码管进行显示。

表 3-1 输入/输出端口

端口名	方向	宽度（位）	作用
sys_clk	输入	1	系统基准时钟，主频 25MHz。
sys_rst_n	输入	1	连接拨动开关，对数字中进行复位。复位信号低电平有效。
i_start	输入	1	连接按键，用于控制暂停/计时。每按下按键进行暂停和计时切换。按键按下为高电平。
a_to_g	输出	8	连接七段数码管的数据输入端 CA~CG 和 DP，用于显示秒表的分和秒。采用共阳极控制，数码管低电平点亮。DP 段永远不点亮。
an	输出	4	连接 4 个七段数码管的使能端 AN0~AN3，其中 AN0 和 AN1 显示秒，AN3 和 AN4 显示分。使能信号高电平有效。

完成上述分秒数字钟的设计，需要有以下几点需要注意：

1. 7 段数码管动态扫描必须采用**使能时钟**实现，扫描频率为 **1KHz**（1ms）。
2. 必须通过边沿检测电路识别**“暂停/计时”**按键按下产生的上升沿，以边进行后续处理。
3. 用于计时的时钟频率为 **25MHz**（40ns）。

4. 本实验在 xdc 约束文件中添加**时钟约束**，所对应的语句如下：

```
set_property -dict { PACKAGE_PIN F5    IOSTANDARD LVCMOS33 } [get_ports { sys_clk }];
create_clock -add -name sys_clk_pin -period 40.00 [get_ports {sys_clk}];
```

5. 由于分秒数字钟计时单位为 1 秒，7 段数码管扫描周期是 1ms，从而造成仿真时间过长。**为了加快仿真速度，可以在仿真的时候使用较大的计时单位和扫描速度**。

五． 实验步骤（建议）

1. 解压缩 dig_clock_stu.rar，打开教师提供的工程文件 dig_clock.xpr，如图 3-7 所示。目前工程中已经提供了顶层文件 dig_clock.sv 和约束文件 dig_clock.xdc，并根据表 3-1 定义了输入/输出接口。**同学对顶层模块的端口定义和约束文件不要进行任何修改**，只需要添加必要模块，并在顶层文件中通过结构化建模完成数字钟的最终设计。

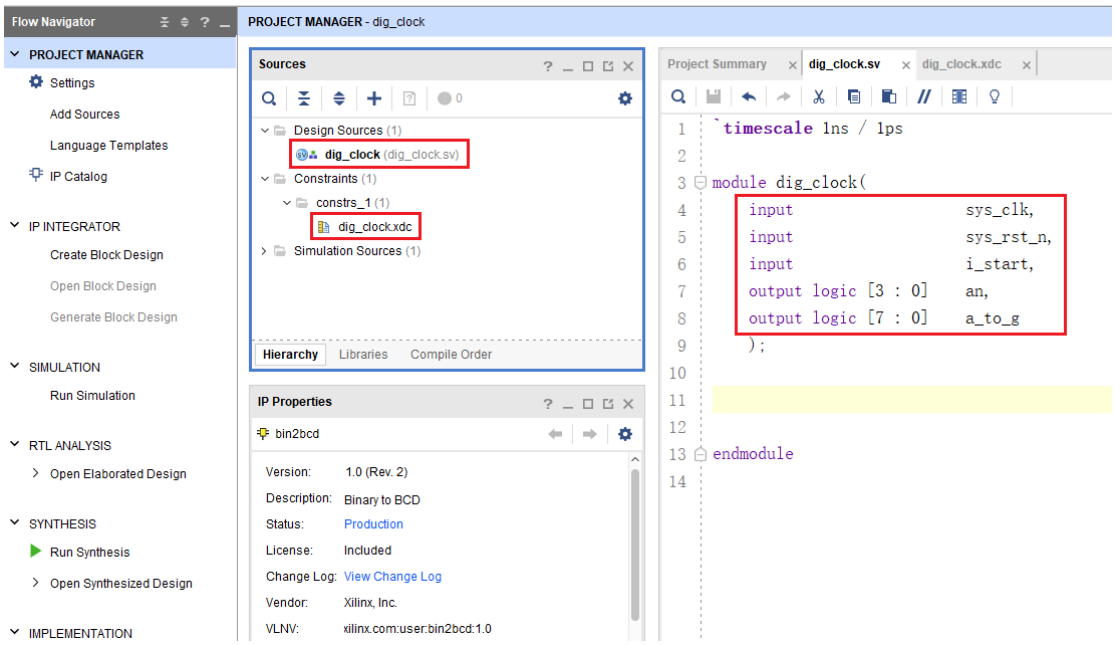


图 3-7 工程 dig_clock

2. 添加二进制转 BCD 码的 IP——bin2bcd。

- 在 Flow Navigator 中，单击 PROJECT MANAGER 下的 IP Catalog 选项，打开 Vivado 的 IP 目录窗口，如图 3-8 所示。

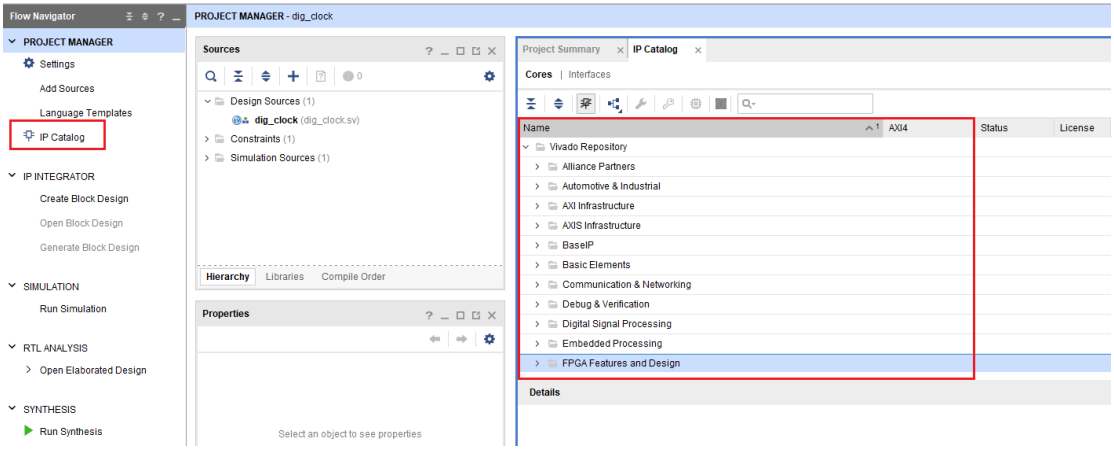


图 3-8 打开 IP Catalog 窗口

- 在 IP Catalog 窗口的空白处，右键单击，从快捷菜单中选择 Add Repository 选项，进行 IP 核目录的添加，如图 3-9 所示。

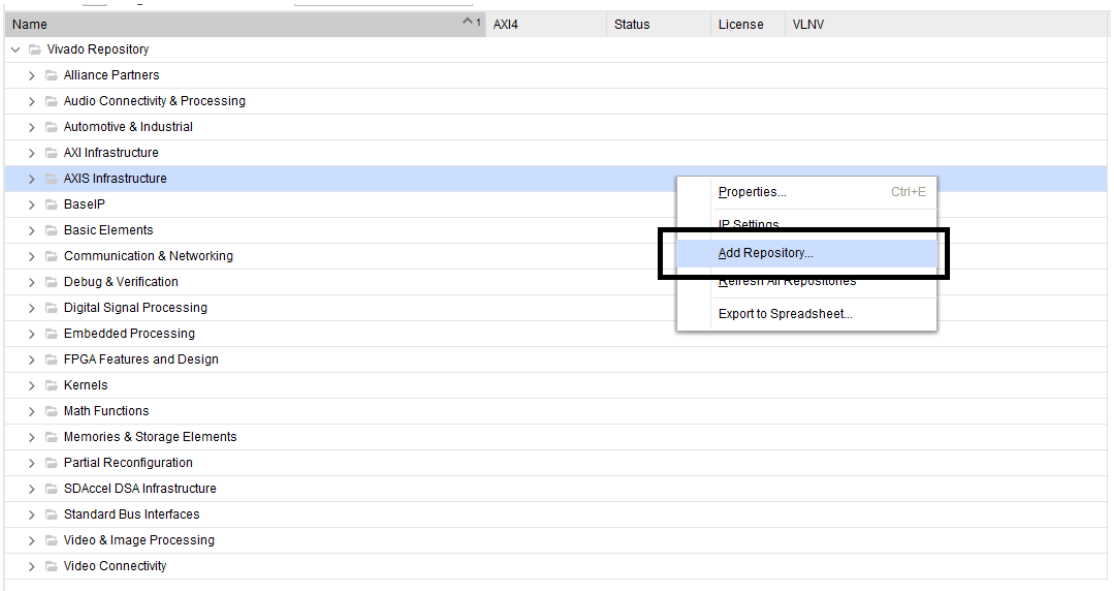


图 3-9 Add Repository 选项

- 在弹出的窗口中，如图 3-10 所示，选择 bin2bcd IP 核所在的目录（“工程路径/bin2bcd”），单击 Select 按钮。然后，弹出一个窗口，如图 3-11 所示，提示为该工程添加一个 IP 核目录，单击 OK，完成 IP 核目录的添

加。所添加的 IP 核目录仅对本工程有效，若创建新工程，需要重新添加。

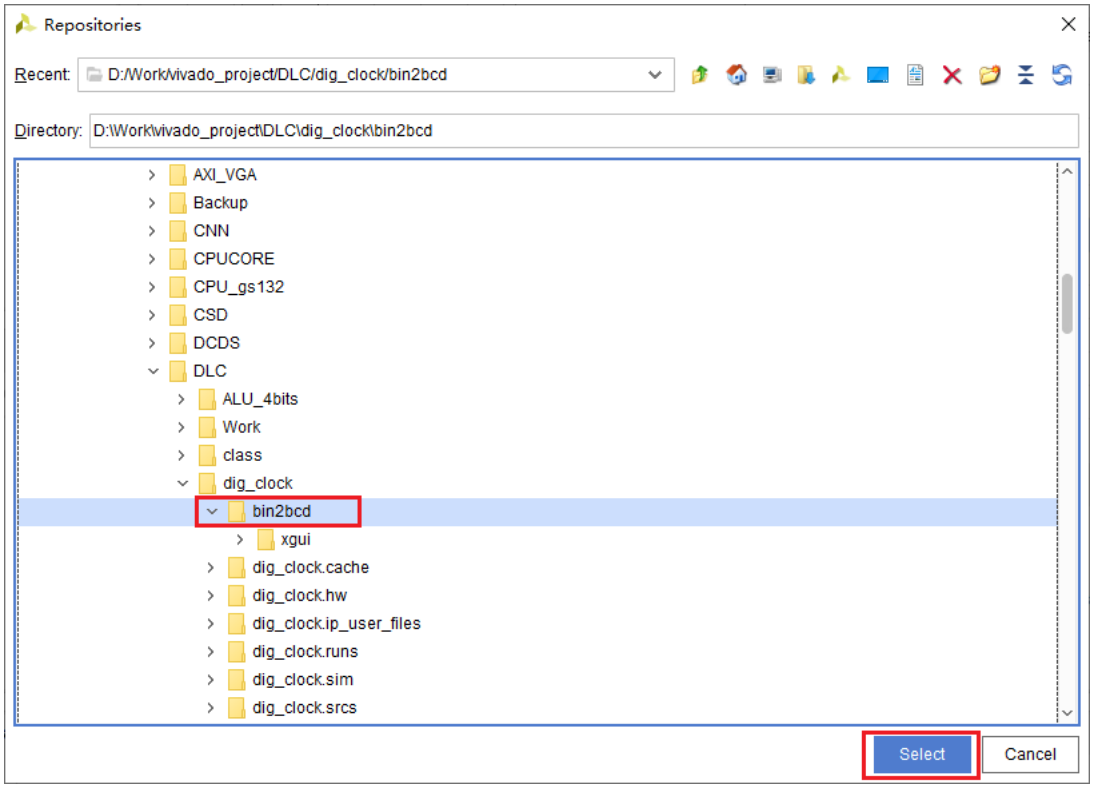


图 3-10 选择 IP 核目录窗口

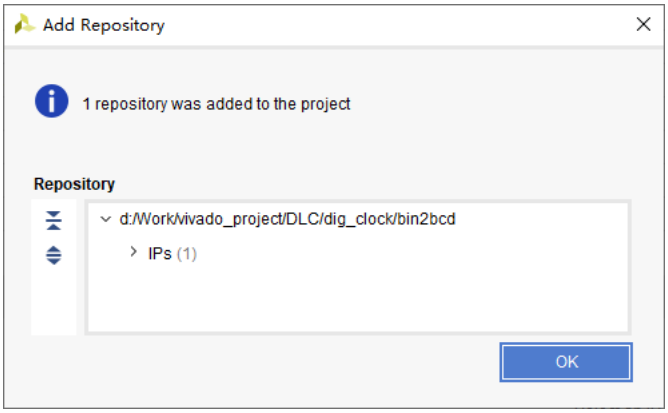


图 3-11 提示为该工程添加 IP 核目录

- 此时，在 IP Catalog 窗口下可以看到新添加的 IP 核目录“User Repository → UserIP”，该目录下有一个 IP 核 bin2bcd，如图 3-12 所示。

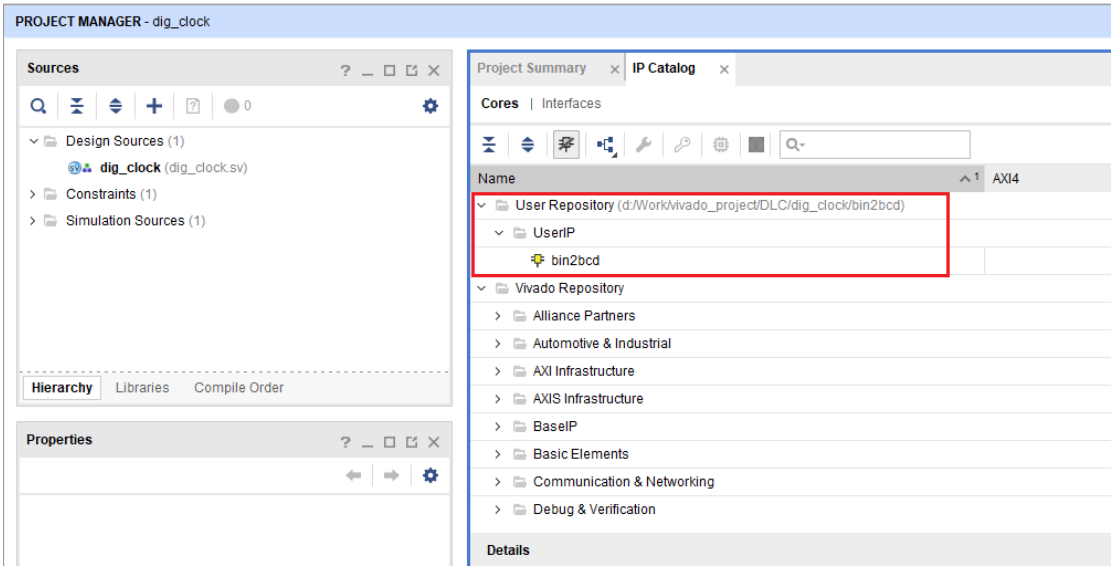
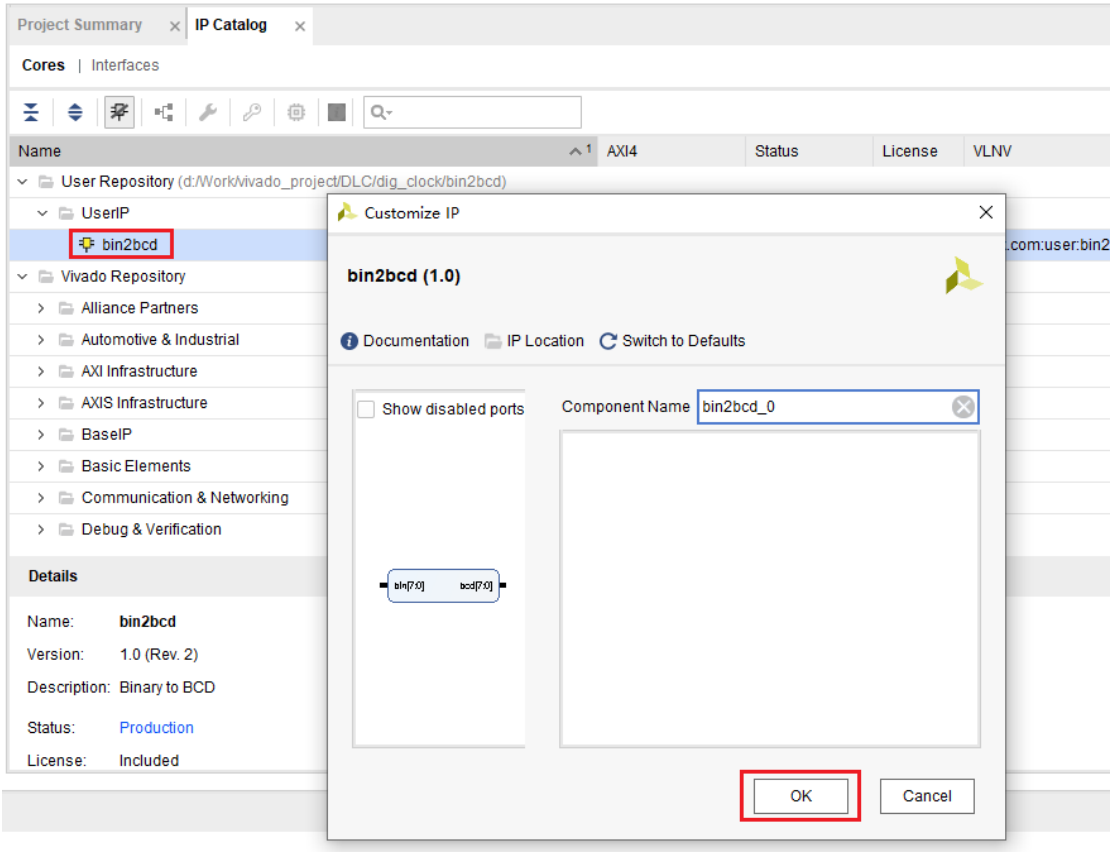
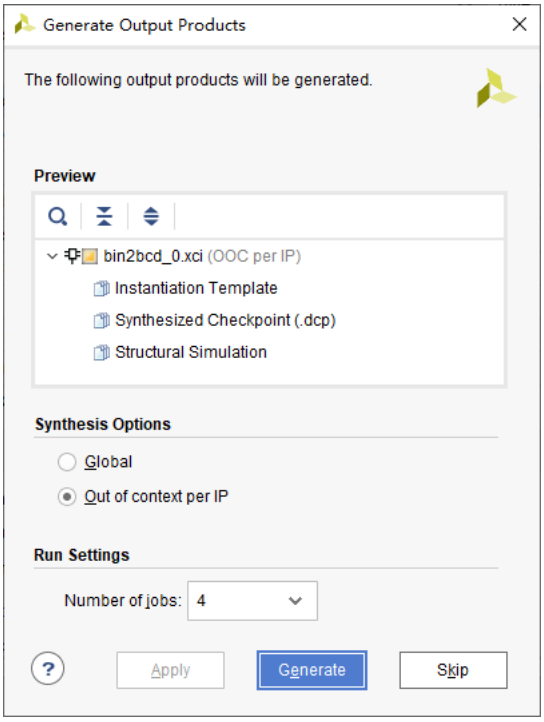


图 3-12 新添加的 IP 核目录

- 双击 bin2bcd IP 核，在弹出的窗口中，如图 3-13(a)所示，点击 OK 按钮。然后，又弹出一个 Generate Output Products 窗口，如图 3-13(b)所示，无需任何修改，直接点击 generate 按钮完成 IP 核的添加。



(a)



(b)

图 3-13 添加 bin2bcd IP 核

- 最终在 Sources 窗口中可以看到所添加的 bin2bcd IP 核，其模块名为“bin2bcd_0”，如图 3-14 所示。

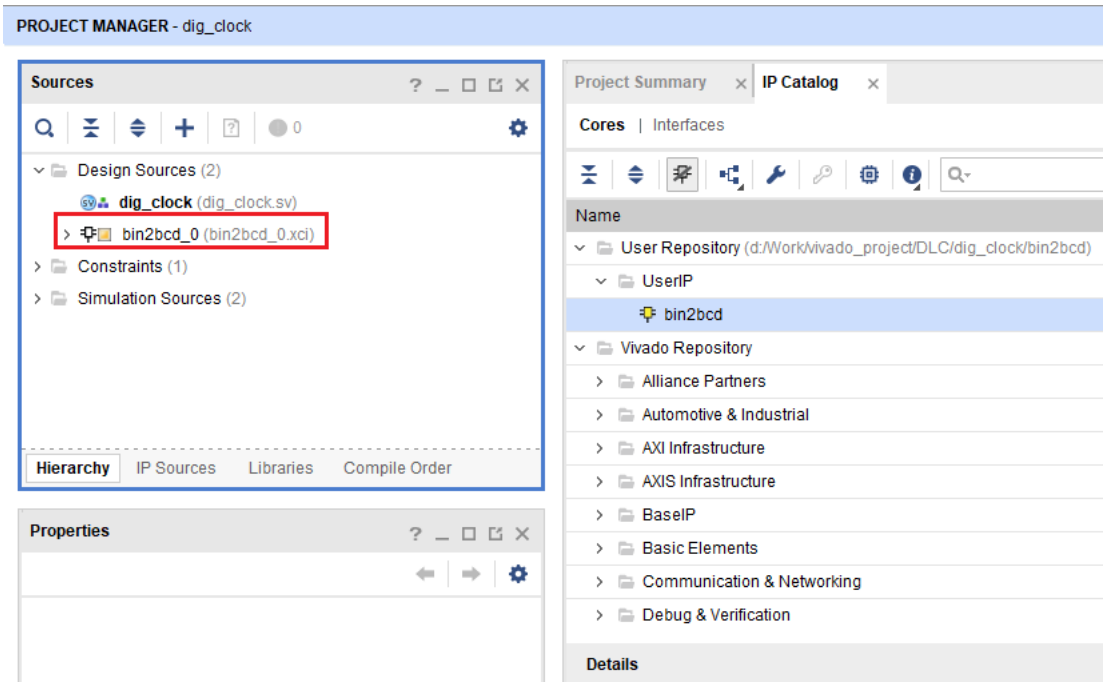


图 3-14 成功添加 bin2bcd IP 核

3. 在 Sources 窗口中，点击“IP Sources”标签，然后展开“Instantiation Template”，双击 bin2bcd_0.veo 文件，给出了 bin2bcd IP 核进行例化的模板，如图 3-15 所示。同学们可以按照该模板在顶层文件 dig_clock.sv 中例化 bin2bcd 模块。

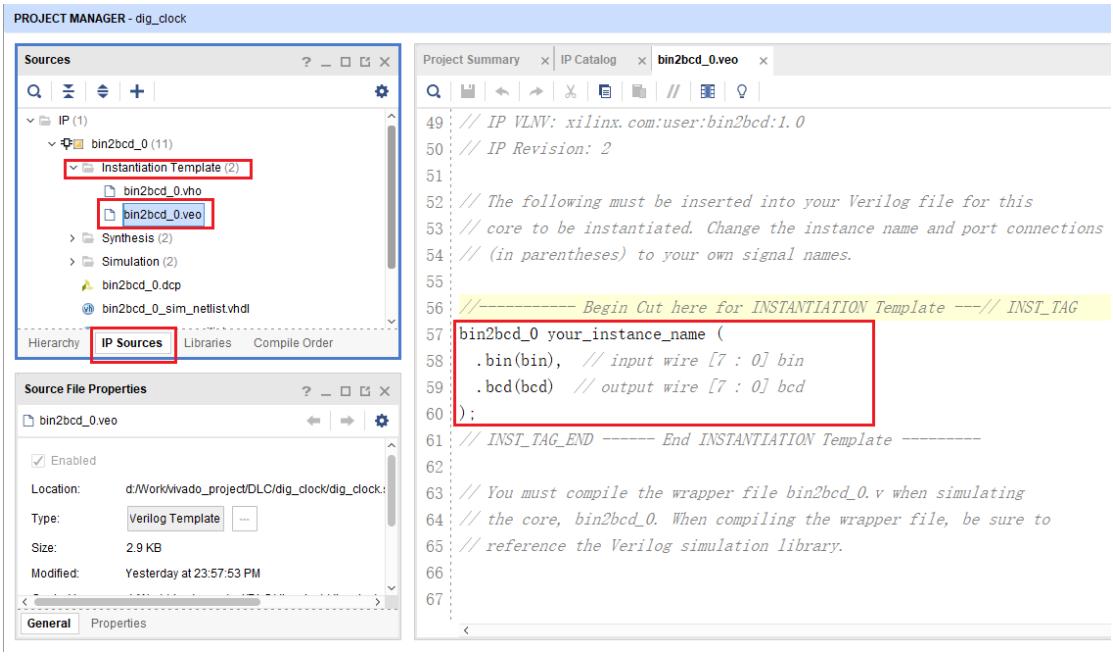


图 3-15 bin2bcd IP 核例化模板

4. 同学们接着根据图 3-6 及设计要求，完成其它子模块的设计，并在顶层文件 dig_clock.sv 中例化这些子模块，进行连接，完成分秒数字中的最终设计。
5. 添加测试文件 **dig_clock_tb.sv**，对所实现的数字钟进行行为仿真。注意，为了提高仿真速度，应对使能时钟频率和计时频率进行适当放大。
6. 如果行为仿真通过，则对工程进行综合、实现、生成 bin 文件。**注意，约束文件已在工程中提供，不需要再添加。**
7. 登录远程 FPGA 硬件云平台，直接导入教师提供的验证平台文件（**分秒数字钟的设计与实现.epl**），无需进行绘制，验证平台如图 3-16 所示。烧写生成的 bin 文件，然后运行实验，通过按键和拨动开关验证所设计数字钟的正确性。

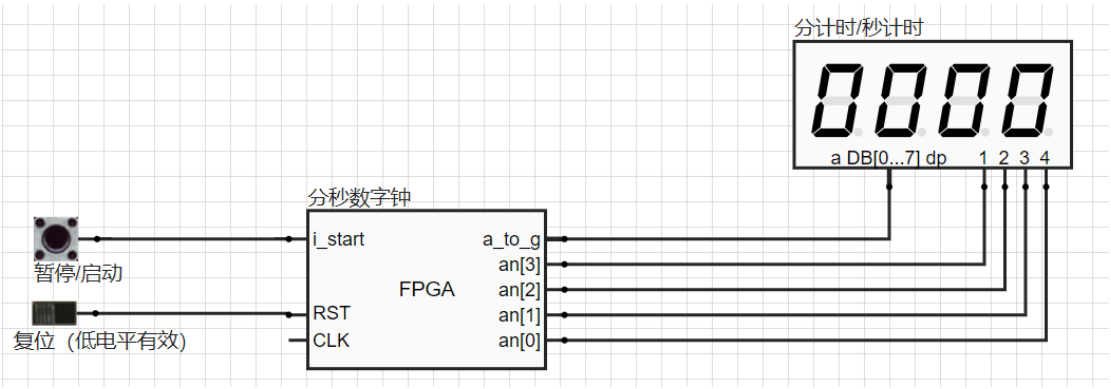


图 3-16 验证平台

六． 实验方式

每位同学独立上机编程实验，实验指导教师现场指导。

七． 参考内容

1. 教材内容和课件

八． 实验报告

1. 画出分秒数字钟电路的原理图（模块级别即可，如使能时钟模块、边沿检测模块等）。
2. 分秒数字钟电路中一共使用了几个计数器，作用分别是什么？
3. 给出分秒数字钟的 SystemVerilog 代码。
4. 给出仿真的波形图和板级验证的截图。

九. 附加题

无