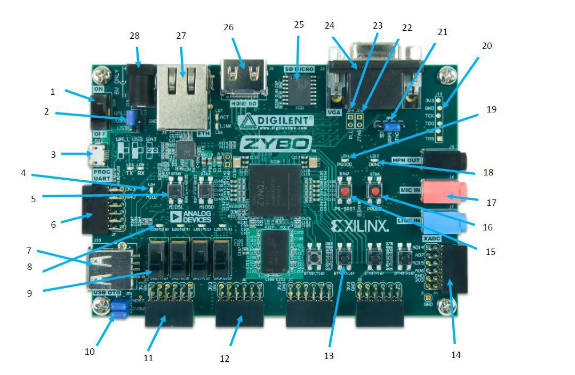
**华东师范大学软件学院课程项目报告**

|  |  |  |
| --- | --- | --- |
| **课程名称**：嵌入式Linux编程与应用 |  | **成绩**： |
| **项目名称**：软硬件协同实现对外部器件的控制——LED灯点亮实验 | | |
| **姓名：**金华俊 **学号：**10122510357 | **班级：** 3 | **同组成员：**王中一 |

1. **项目简介:**

ZYBO(Zynq Board)(如下图)是一款基于Z-7010而构建的功能丰富的入门级嵌入型软件和数字电路开发平台。具有板上储存器，视频与音频I/O,双模USB，以太网和SD插槽等丰富配置，还提供5个Pmod连接器，便于对任何设计进行升级。





ZYBO还可用于设计各种复杂程度的系统，例如本项目中要运用到的控制LED的简单程序。而且Xilinx工具和Linux解决方案都无需附加费用即可与ZYBO搭配使用。

本嵌入式编程与应用项目通过ZYBO板和Xilinx工具的综合使用，实现了软硬件协同的目标。最终实现通过ZYBO板上的按钮实现对ZYBO板上Pmod接口处外接放光二极管灯泡的发光控制

本项目中我们组实现了2个功能，1个功能是完成zybo板上的开关对于Pmod接口处二极管灯泡的发光控制，还有1个是完成一个Pmod接口处二极管灯泡一闪一闪的闪烁效果。

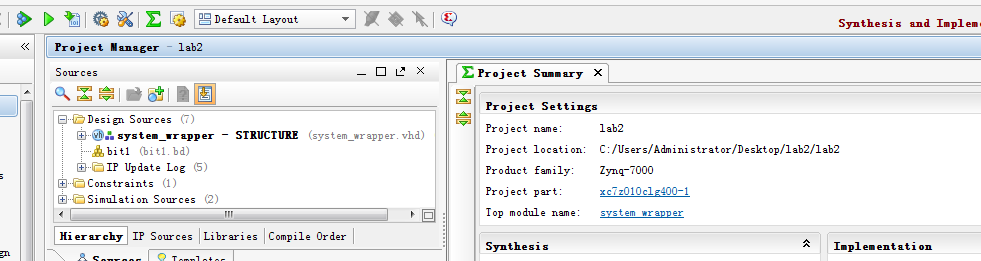
1. **技术原理：**

ZYBO板上的ZYBO芯片通过AXI连接FPGA可编程模块，然后由FPGA可编程模块连接BUTTONS模块和发光二极管模块。之后分别将BUTTONS模块和发光二极管模块的GPIO接口和ZYBO板上相应的BUTTONS引脚和Pmod引脚对应配置好。最后，将编译好的程序下载到ZYBO上，将二极管插入到相应的Pmod引脚里，控制BUTTONS开关就可以将发光二极管点亮。

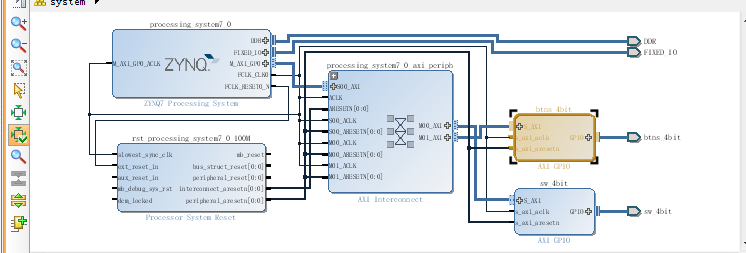
FPGA

AXI

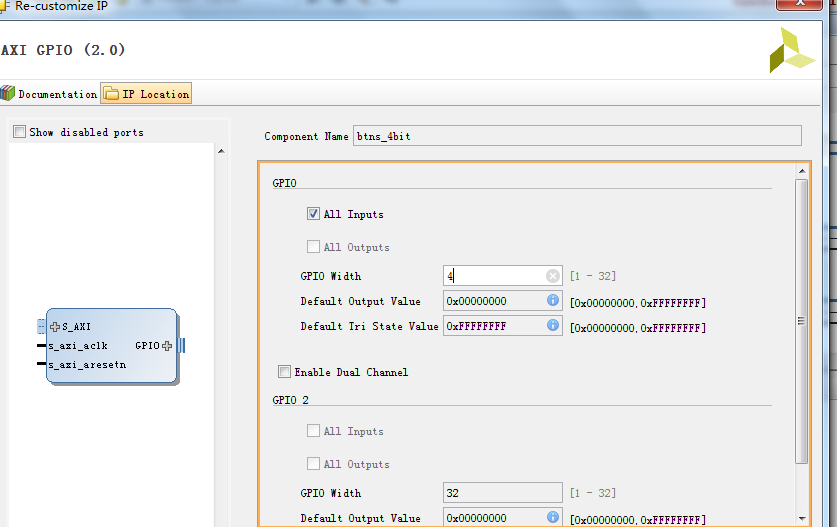
1. **实现方法与步骤：**
2. 打开LAB2工程，因为我们组是在LAB2的基础上完成此项目的。



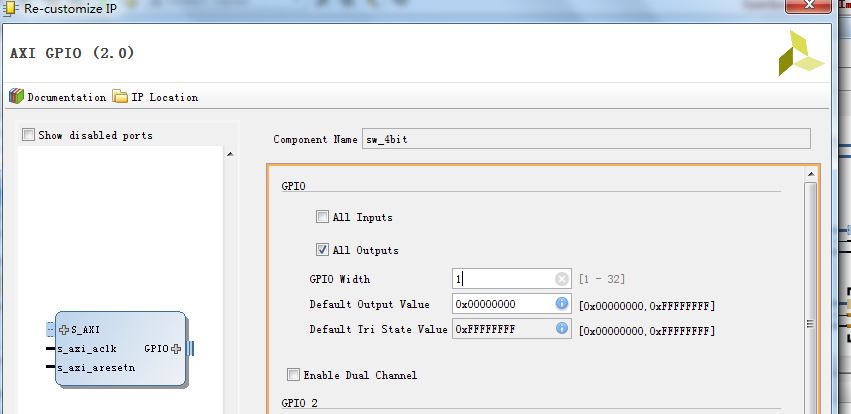
1. 根据本项目需要更改IP相应设置，设置一个发光二极管模块



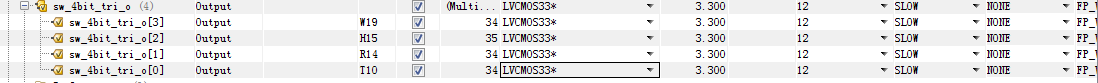
3.双击Buttons模块，弹出设置窗口，在GPIO width中设置宽度4，因为需要有4个Buttons按钮，每个按钮对应其中一位。

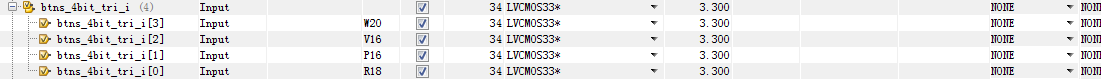


4.设置发光二极管的GPIO宽度。因为我们只用1个发光二极管，发光二极管的长脚接JDP口，短脚接GND端口，所以我们需要设置一位宽度用于输出电压将发光二极管点亮

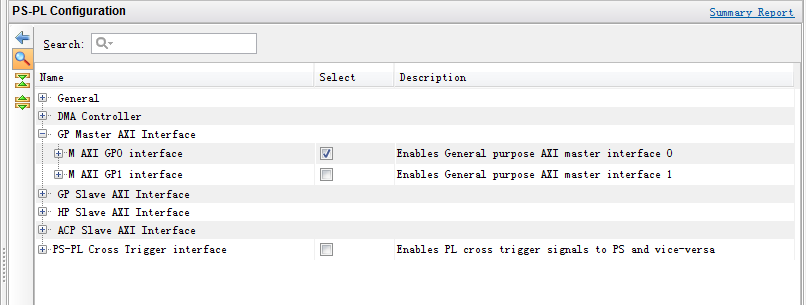


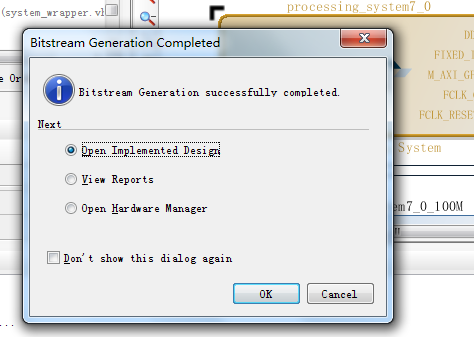
5.设置GPIO的外围连接，将BUTTONS和发光二极管模块的GPIO端口分别设置到相应的引脚上，并设置电压为3.3V，这是点亮二极管的临界电压。





6.设置PS-PL，生成bitsream并且输出到Sdk里面





7.编辑SDK里TestApp，在sdk中进行测试，代码如下：

最基本的控制单个灯管亮暗的实验代码：

#include "xparameters.h"

#include "xgpio.h"

#include "xutil.h"

//====================================================

int main (void)

{

/\*源代码

XGpio dip,push;

int i,psb\_check,dip\_check;

xil\_printf("-- Start of the Program --\r\n");

XGpio\_Initialize(&dip, XPAR\_SW\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&dip, 1, 0xffffffff);

XGpio\_Initialize(&push, XPAR\_BTNS\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&push, 1, 0xffffffff);

while (1)

{

psb\_check = XGpio\_DiscreteRead(&push, 1);

xil\_printf("Push Buttons Status %x\r\n", psb\_check);

dip\_check = XGpio\_DiscreteRead(&dip, 1);

xil\_printf("DIP Switch Status %x\r\n", dip\_check);

for (i=0; i<9999999; i++);

}

\*/

//----------变量定义-------------

XGpio led,push; //2个GPIO led灯 和 button(push)

int i,psb\_check;//psb\_check为push的返回值

xil\_printf("-- Start of the Program --\r\n");

printf("result");

//GPIO口的初始化 及 输入输出设置

XGpio\_Initialize(&led,XPAR\_SW\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&led,1,0x00000000);//led 0是输出

XGpio\_Initialize(&push, XPAR\_BTNS\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&push, 1, 0xffffffff);//button 1是输入

while (1)

{

//按下button,得到按下哪个按钮

psb\_check = XGpio\_DiscreteRead(&push,1);

xil\_printf("psb\_check %x\r\n",psb\_check);

if(psb\_check==1)

{

//写入led

XGpio\_DiscreteWrite(&led,1,0xffffffff);

}

else

{

XGpio\_DiscreteWrite(&led,1,0x00000000);

}

for (i=0; i<9999999; i++);

}

}

灯泡闪烁效果的代码

#include "xparameters.h"

#include "xgpio.h"

#include "xutil.h"

//====================================================

int main (void)

{

/\*源代码

XGpio dip,push;

int i,psb\_check,dip\_check;

xil\_printf("-- Start of the Program --\r\n");

XGpio\_Initialize(&dip, XPAR\_SW\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&dip, 1, 0xffffffff);

XGpio\_Initialize(&push, XPAR\_BTNS\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&push, 1, 0xffffffff);

while (1)

{

psb\_check = XGpio\_DiscreteRead(&push, 1);

xil\_printf("Push Buttons Status %x\r\n", psb\_check);

dip\_check = XGpio\_DiscreteRead(&dip, 1);

xil\_printf("DIP Switch Status %x\r\n", dip\_check);

for (i=0; i<9999999; i++);

}

\*/

//----------变量定义-------------

XGpio led,push; //2个GPIO led灯 和 button(push)

int i,psb\_check;//psb\_check为push的返回值

xil\_printf("-- Start of the Program --\r\n");

printf("result");

//GPIO口的初始化 及 输入输出设置

XGpio\_Initialize(&led,XPAR\_SW\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&led,1,0x00000000);//led 0是输出

XGpio\_Initialize(&push, XPAR\_BTNS\_4BIT\_DEVICE\_ID);

XGpio\_SetDataDirection(&push, 1, 0xffffffff);//button 1是输入

try2 一闪一闪

int count = 0;

while(1)

{

//按下button,得到按下哪个按钮

psb\_check = XGpio\_DiscreteRead(&push,1);

count++;

xil\_printf("psb\_check %x\r\n",psb\_check);

if(count%2!=0)

{

xil\_printf("count %x\r\n", count);

XGpio\_DiscreteWrite(&led,1,0xffffffff);

}

else

{

xil\_printf("count %x\r\n", count);

XGpio\_DiscreteWrite(&led,1,0x00000000);

}

for (i=0; i<9999999; i++);

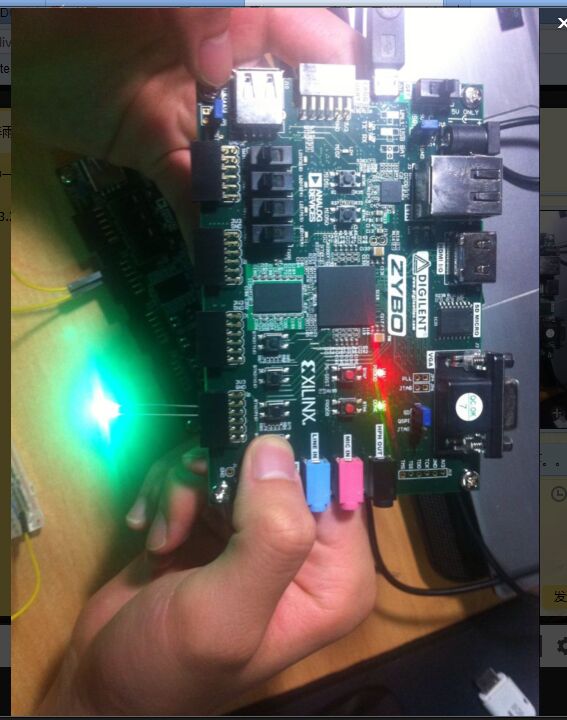
}

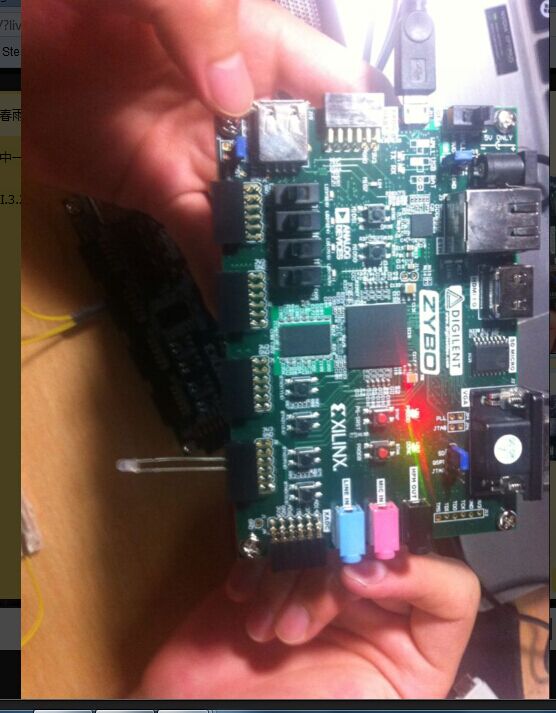
\*/

}

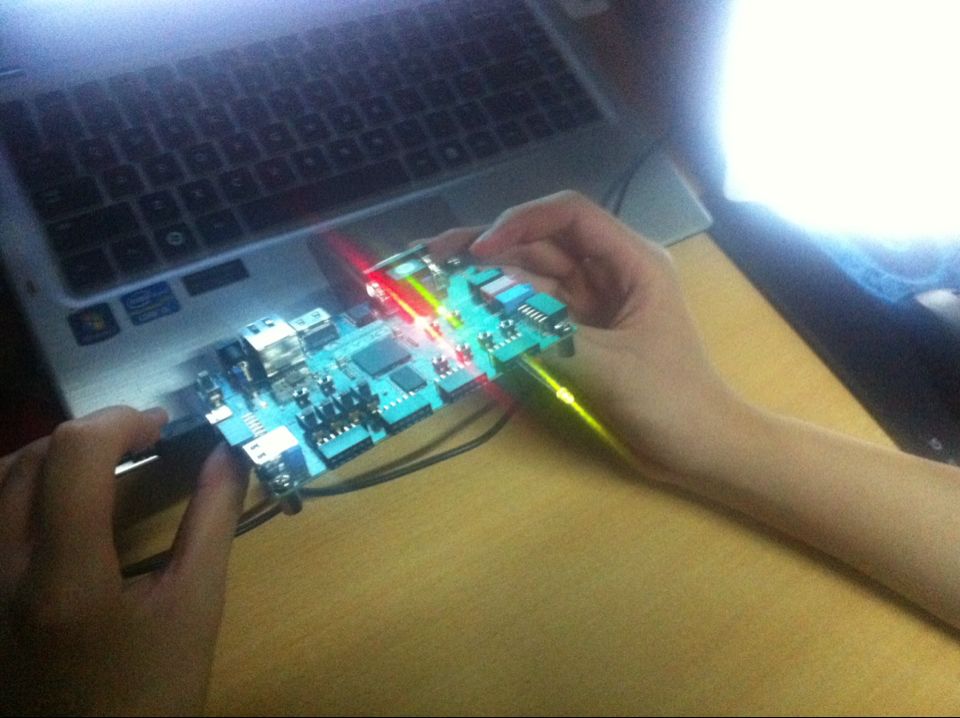
8.输出结果如下图：

结果1：单开关控制单灯泡的亮暗





结果2：小灯泡的闪烁实现



1. **个人贡献**

在本项目中，我主要负责后期sdk部分代码的编写，设计，测试，以及最后文档的编写，结果的录入。个人贡献在本项目中约占50%。

1. **项目总结**

通过此次实验，我加深了对vivado开发实验过程的理解，对于软硬件协同增加了认识。更加熟悉了vivado里面模块的配置，设计，sdk里面代码的编写。也遇到了一些说vivado报错,SDK中无法加载elf文件等等问题，通过尝试查找资料，重建项目，重启应用等等方法解决了其中较多的问题。对于嵌入式编程积累了经验，以后也会努力的做一名优秀的嵌入式开发人员