zedboard的裸机中断实验（一）

原创

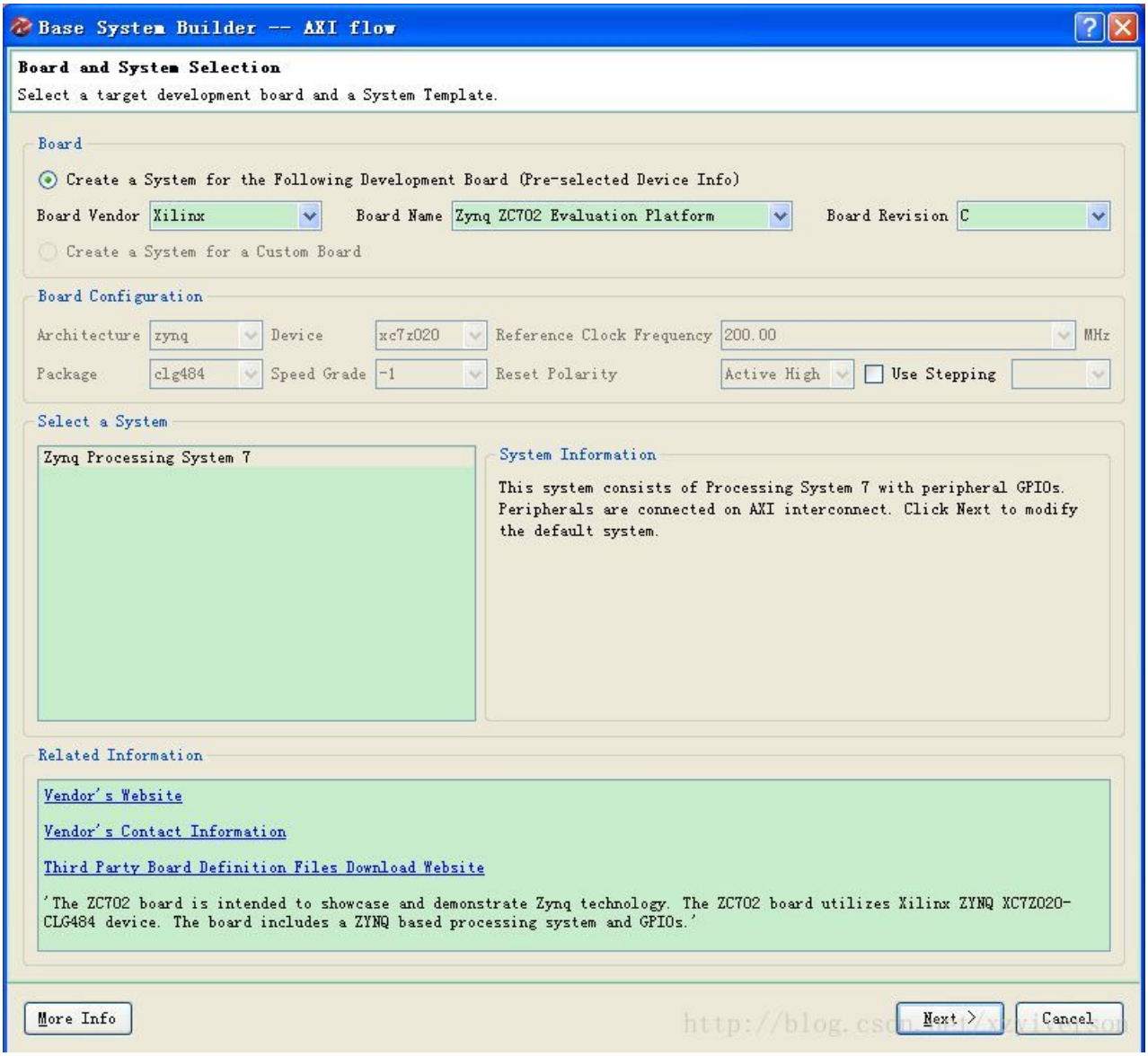
iverson1991 于 2014-03-03 21:06:08 发布 6669 收藏
分类专栏： zedboard学习 文章标签： zedboard中断 zynq gpio 中断实验 zynq裸机中断

学习了zynq的中断系统后，这里做一个简单的中断实验，第一个中断的实验是一个简单的按键中断实验。

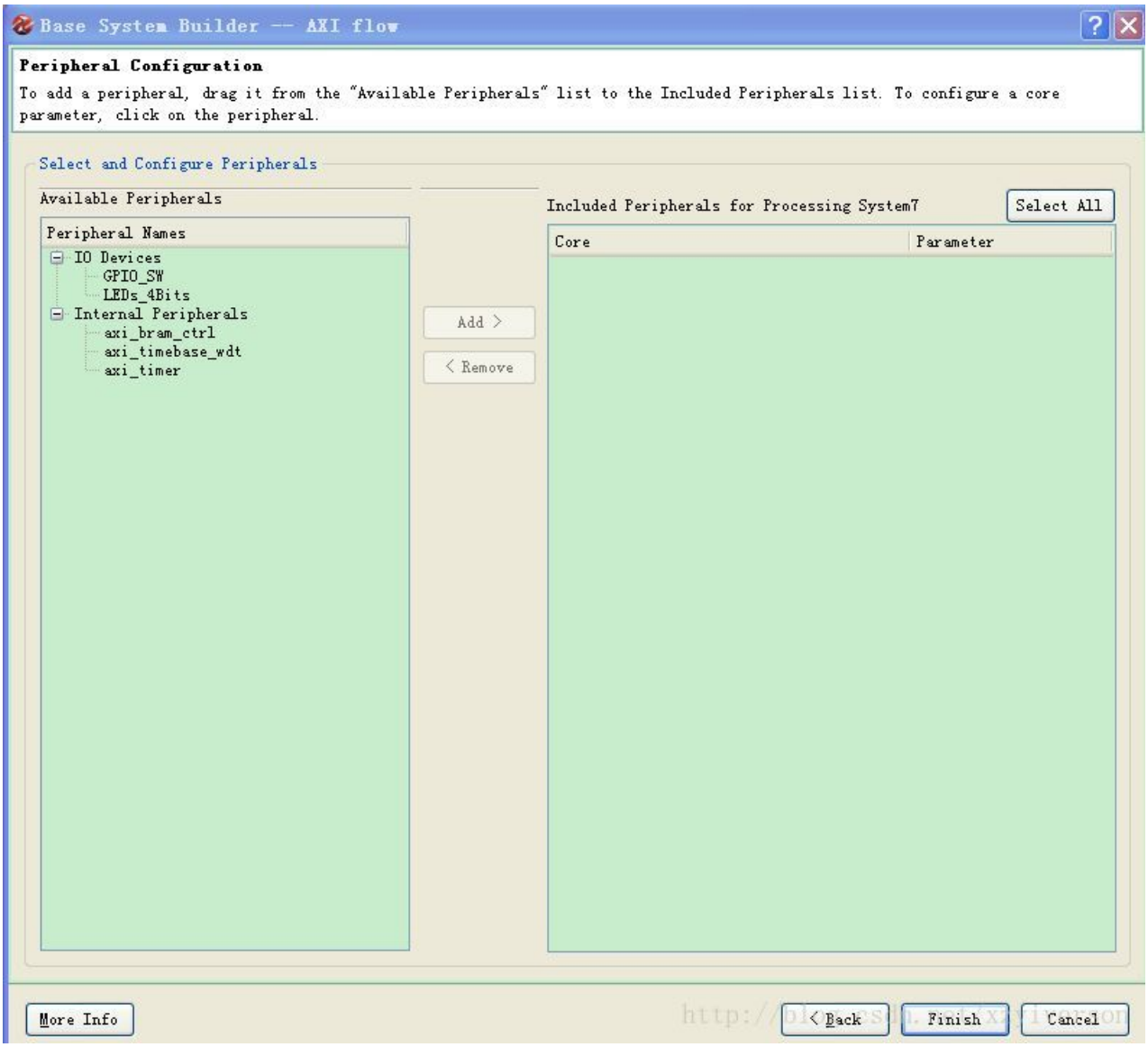
开发环境：XPS14.6+SDK14.6

一：硬件配置

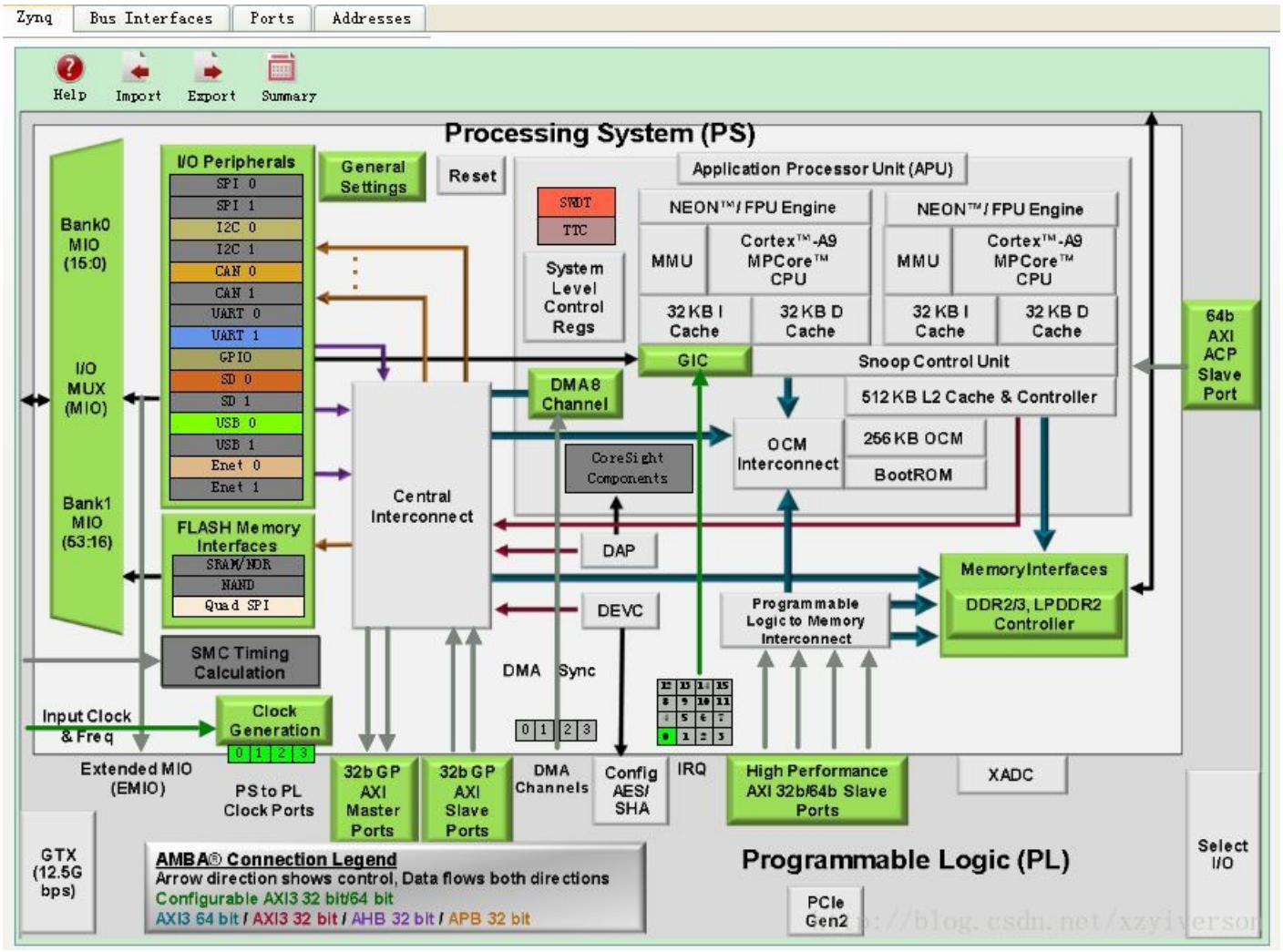
1. 启动xps，创建工程，选择好平台。



2.移除外设

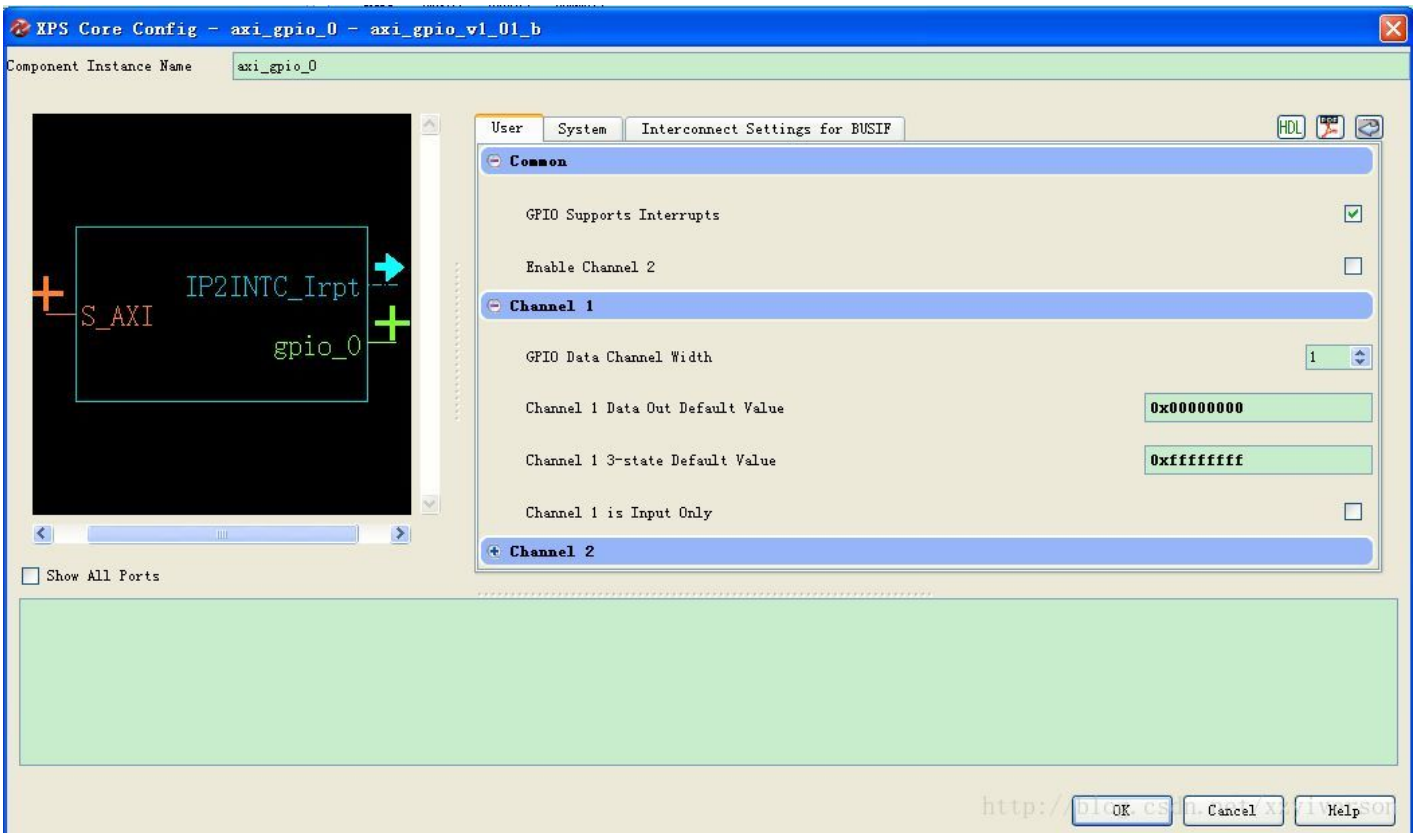


3.进入工程，已经配置好了（PlanAhead的话好像是不是自动配置好的）

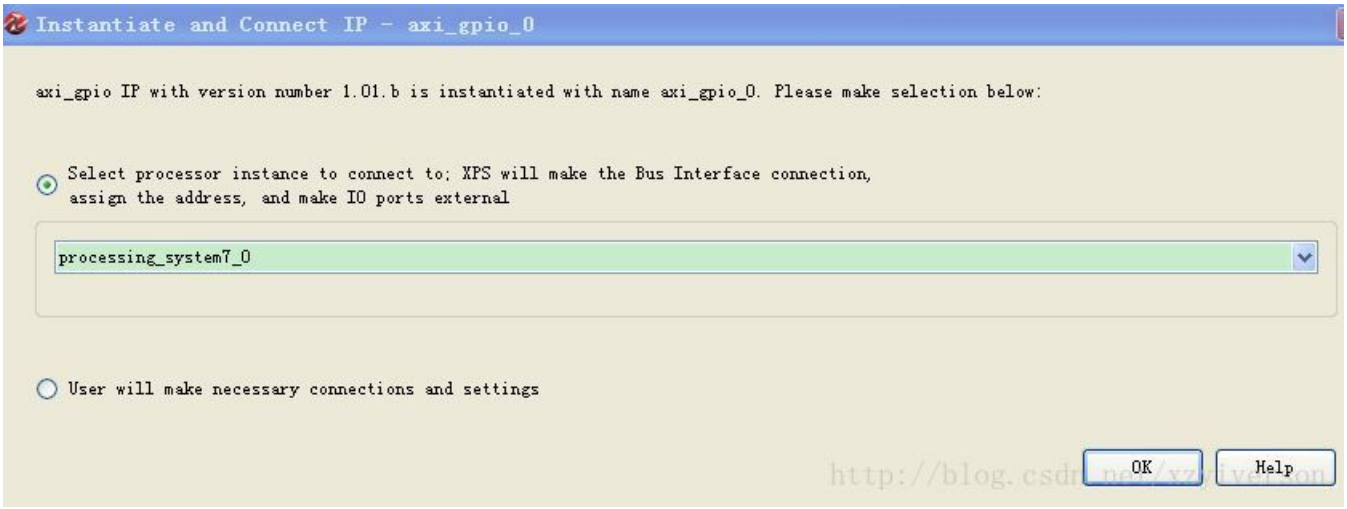


注：建议自己手动添加一次zedboard配置文件，我这前面没添加后面不行，后面添加后就可以了。

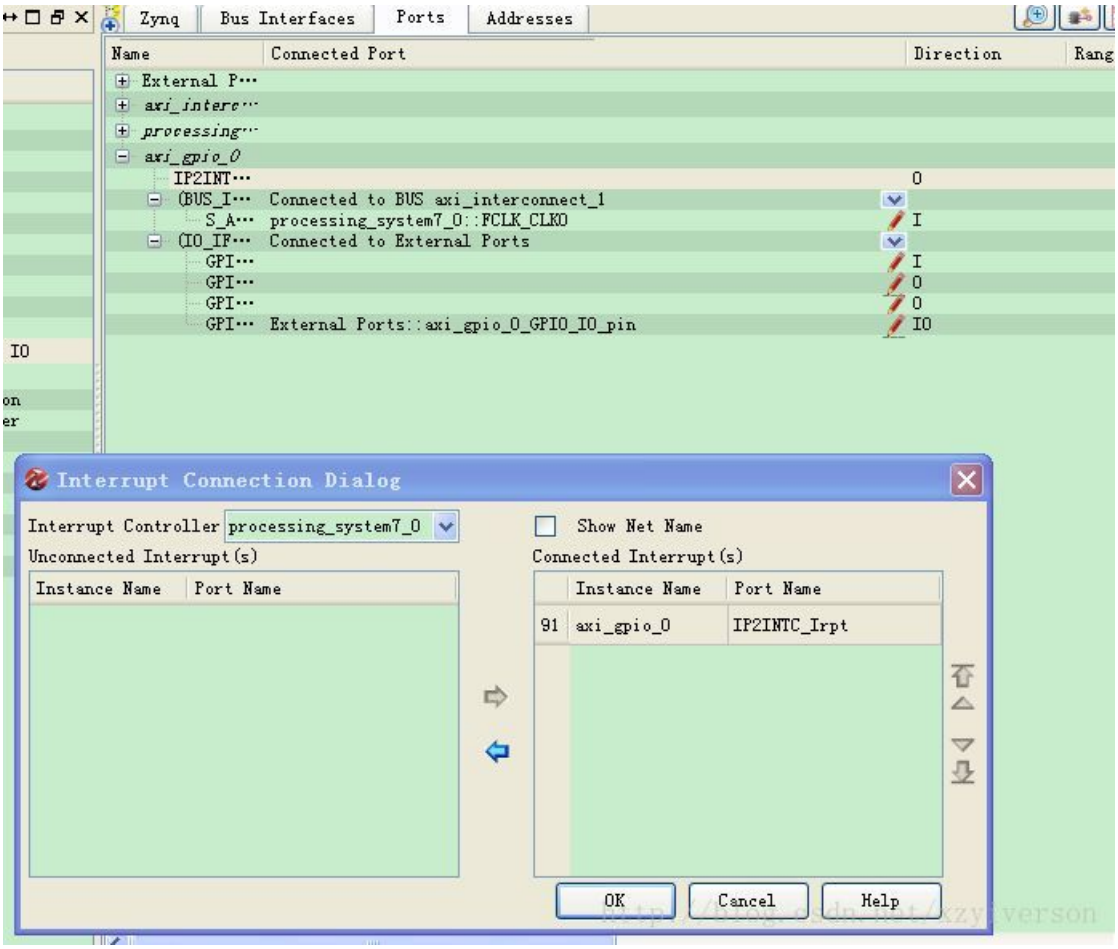
4.添加axi GPIO的IP核，注意这里勾上GPIO Support Interrupts 和设置好为1位。



5.点击ok



6.这里需要连接中断，如下图



7.然后就是添加约束了，生成bit流文件了，在导入到SDK就可以了。硬件配置就结束了。

二.应用软件

和以前一样，新建一个helloworld工程。

这里我对比了一下又添加中断后的axi gpio和没有添加中的gpio的区别

这是添加中断后的工程（这是在xparameter.h中找到的信息）

```
/******************************************************************/

/* Definitions for Fabric interrupts connected to ps7_scugic_0 */
#define XPAR_FABRIC_AXI_GPIO_0_IP2INTC_IRPT_INTR 91

/******************************************************************/

/* Canonical definitions for Fabric interrupts connected to ps7_scugic_0 */
#define XPAR_FABRIC_GPIO_0_VEC_ID XPAR_FABRIC_AXI_GPIO_0_IP2INTC_IRPT_INTR

/******************************************************************/
```


这是以前博客zedboard--zynq使用自带外设IP让ARM PS访问FPGA（八）

```
/******************************************************************/

/* Definitions for Fabric interrupts connected to ps7_scugic_0 */

/******************************************************************/

/* Canonical definitions for Fabric interrupts connected to ps7_scugic_0 */

/******************************************************************/
```


也就是在这里查找中断的ID号了，因为我们在应用程序中要使用ID号了。

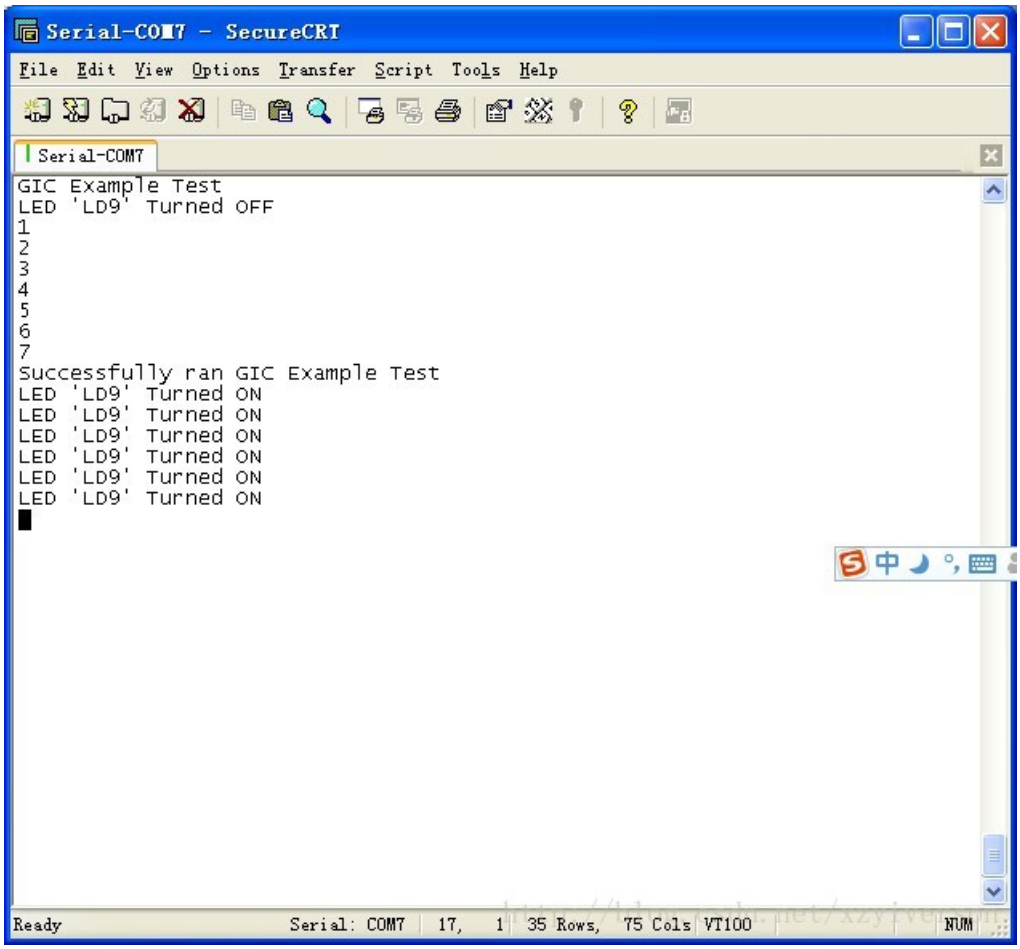
软件代码：

```
#include <stdio.h>
#include <stdlib.h>
#include "xil_io.h"
#include "xil_exception.h"
#include "xparameters.h"
#include "xil_cache.h"
#include "xil_printf.h"

#include "xscugic.h"
#include "xgpio.h"
#include "xgpiops.h"

#define INTC_DEVICE_ID      XPAR_SCUGIC_SINGLE_DEVICE_ID              //设备ID
#define INTC_DEVICE_INT_ID  XPAR_FABRIC_AXI_GPIO_0_IP2INTC_IRPT_INTR   //中断号

int ScuGicExample(u16 DeviceId);
void DeviceDriverHandler(void *CallbackRef);

unsigned int led_state;
volatile static int InterruptProcessed = FALSE;
static int iPinNumber = 7; /*Led LD9 is connected to MIO pin 7*/

XScuGic InterruptController;         /*Instance of the Interrupt Controller */
static XScuGic_Config *GicConfig;    /* The configuration parameters of the controller */

static XGpioPs psGpioInstancePtr;
XGpioPs_Config *GpioConfigPtr;

static XGpio GPIOInstance_Ptr;
```

```
static XGpio GPIOInstance_Ptr;

int main(void)
{
    int Status;
    int xStatus;

    u32 uPinDirection = 0x1;
    led_state = 0;
    init_platform();
    print("GIC ExampleTest\r\n");
    //axi gpio的初始化，这里比之前的GPIO多了中断的使能
    xStatus =XGpio_Initialize(&GPIOInstance_Ptr,XPAR_AXI_GPIO_0_DEVICE_ID);
    if(XST_SUCCESS != xStatus)
       print("GPIO INITFAILED\n\r");
    XGpio_SetDataDirection(&GPIOInstance_Ptr, 1,1);
    XGpio_InterruptGlobalEnable(&GPIOInstance_Ptr);
    XGpio_InterruptEnable(&GPIOInstance_Ptr, XGPIO_IR_CH1_MASK);
    //LD9，即MIO7的初始化
      GpioConfigPtr =XGpioPs_LookupConfig(XPAR_PS7_GPIO_0_DEVICE_ID);
      if(GpioConfigPtr ==NULL)
        return XST_FAILURE;
      xStatus =XGpioPs_CfgInitialize(&psGpioInstancePtr,
          GpioConfigPtr,
          GpioConfigPtr->BaseAddr);
      if(XST_SUCCESS !=xStatus)
        print(" PS GPIO INIT FAILED \n\r");
     XGpioPs_SetDirectionPin(&psGpioInstancePtr, iPinNumber,uPinDirection);
     XGpioPs_SetOutputEnablePin(&psGpioInstancePtr, iPinNumber,1);
      print("LED 'LD9' Turned OFF \r\n");
      XGpioPs_WritePin(&psGpioInstancePtr,iPinNumber,0);
      //调用中断控制器的初始化
    Status = ScuGicExample(INTC_DEVICE_ID);
    if (Status != XST_SUCCESS) {
       print("GIC ExampleTest Failed\r\n");
       return XST_FAILURE;
    }
    print("Successfully ran GICExample Test\r\n");
    while(1)
    {
       usleep(1);
       if(TRUE == InterruptProcessed)
       {
           InterruptProcessed = FALSE;
           sleep(1);
           XScuGic_Enable(&InterruptController,INTC_DEVICE_INT_ID);
       }
    }
    cleanup_platform();
    return XST_SUCCESS;
}
//中断初始化，
int ScuGicExample(u16 DeviceId)
{
    int Status;
    print("1 \r\n");
    GicConfig = XScuGic_LookupConfig(DeviceId);
    if (NULL == GicConfig) {
       return XST_FAILURE;
    }
```

```
    print("2 \r\n");
    Status = XScuGic_CfgInitialize(&InterruptController,GicConfig,
                  GicConfig->CpuBaseAddress);
    if (Status != XST_SUCCESS) {
       return XST_FAILURE;
    }

    print("3 \r\n");

    XScuGic_SetPriorityTriggerType(&InterruptController,INTC_DEVICE_INT_ID,
                     0x05, 0x01);

    Status = XScuGic_SelfTest(&InterruptController);
    if (Status != XST_SUCCESS) {
       return XST_FAILURE;
    }

    print("4 \r\n");

    Xil_ExceptionRegisterHandler(XIL_EXCEPTION_ID_INT,
           (Xil_ExceptionHandler) XScuGic_InterruptHandler,
           &InterruptController);

    print("5 \r\n");

    Status = XScuGic_Connect(&InterruptController,INTC_DEVICE_INT_ID,
              (Xil_ExceptionHandler)DeviceDriverHandler,
              (void*)&InterruptController);
    Xil_ExceptionEnable();
    if (Status != XST_SUCCESS) {
       return XST_FAILURE;
    }
    print("6 \r\n");

    XScuGic_Enable(&InterruptController, INTC_DEVICE_INT_ID);
    print("7 \r\n");
    return XST_SUCCESS;
}

void DeviceDriverHandler(void *CallbackRef)
{
     print("LED 'LD9' Turned ON \r\n");
     XScuGic_Disable(&InterruptController,INTC_DEVICE_INT_ID);
     led_state = (0 ==led_state)?1:0;
     XGpioPs_WritePin(&psGpioInstancePtr,iPinNumber,led_state);
     XGpio_InterruptClear(&GPIOInstance_Ptr,XGPIO_IR_CH1_MASK);
     XScuGic_Enable(&InterruptController,INTC_DEVICE_INT_ID);
     InterruptProcessed = TRUE;
}
```

实验结果：板子的图就不贴了，没多大的意思，看串口的信息吧。



代码中其实是按一下就亮，再按一下就熄灭，但是这里存在边沿触发中断的情况（似乎与实际的不一样，这个还有待查证），但不妨碍本实验了练习zedboard的中断，下次继续学习zynq的中断。

PS：中途出现的问题，之前其实就可以了的，但是后面自己新建一个xps工程后又不可以了，之前的工程是用PlanAhead来弄的，是自己添加的配置文件，这次xps工程一开始没有添加配置文件，后面添加配置文件后重新生成bit文件，导入到sdk中，就可以了。还有就是ucf文件这里（不知道有没有编写ucf比较好的方法，反正我是很害怕这里了。），这里真的是要非常当心。