

Lab9: “板载加速计”

——Xilinx EDK 设计基本流程

基于 Nexys 4 FPGA 平台

Lab 9: “板载加速计”

实验简介

本实验旨在使读者学会 Xilinx 的 XPS 和 SDK 工具的使用，同时完成一个在串口上显示板卡 xyz 三轴加速度的数据简单例程。

实验目标

在完成本实验后，您将学会：

- XPS 工具的使用流程，从新建工程到导入到 SDK。
- SDK 工具的使用流程，从导入到 SDK 到在板卡上运行 C 语言程序。
- 学会使用板载加速计

实验过程

本实验旨在指导读者使用 Xilinx 的 XPS 工具，调用串口的 IP 核，并将导入到 SDK，调用这个 IP 核，在串口上显示板卡 xyz 三轴加速度的数据，然后在 Nexys 4 平台上进行测试验证。

实验由以下步骤组成：

1. 在 XPS 中建立工程
2. 添加 IP 核并调整相关设置
3. 进行端口的互连
4. 将工程导入到 SDK
5. 在 SDK 中添加 c 语言源程序
6. 在 Nexys 4 上进行测试验证

实验环境

◆ 硬件环境

1. PC 机
2. Nexys 4 FPGA 平台

◆ 软件环境

Xilinx ISE Design Suite 14.3（FPGA 开发工具）

第一步 创建工程

- 1-1. 运行 **Xilinx Platform Studio**, 创建一个空的新工程, 基于 **xc6slx45csg484-3** 芯片和 **VHDL** 语言.
- 1-1-1. 选择 开始菜单 > 所有程序 > **Xilinx Design Tools > ISE Design Suite 14.3 > EDK > Xilinx Platform Studio**. 点击运行 **Xilinx Platform Studio(XPS)** (Xilinx Platform Studio 是 ISE 嵌入式版本 Design Suite 的关键组件, 可帮助硬件设计人员方便地构建、连接和配置嵌入式处理器系统, 能充分满足从简单状态机到成熟的 32 位 RISC 微处理器系统的需求。).
- 1-1-2. 点击 **Create New Project Using Base System Builder** 来打开新工程建立向导。会出现一个 **Create New XPS Project Using BSB Wizard** 对话框, 如图 3.



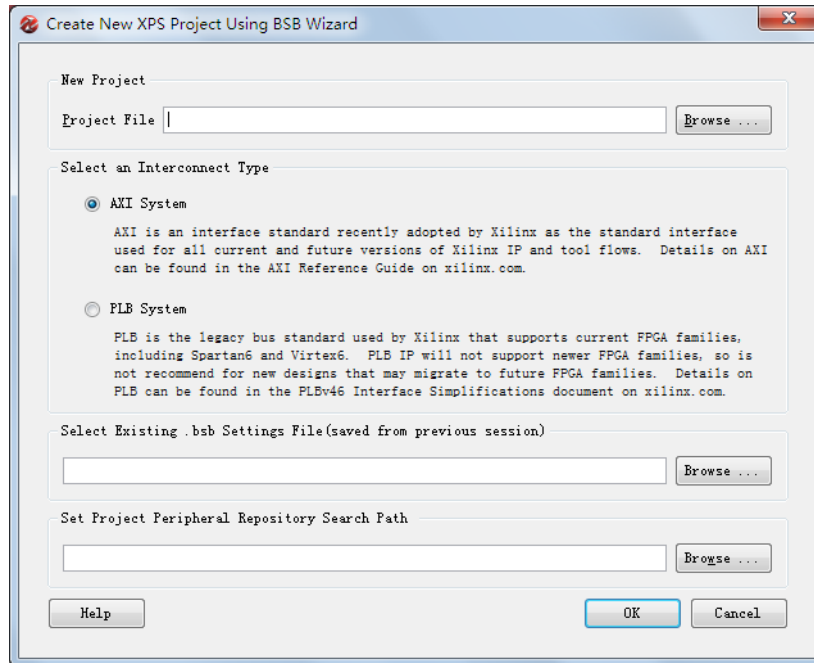


图 3：新工程建立向导

1-1-3.如图 3，在新工程建立向导对话框的 Project File 栏选择工程建立后存放的路径，这里可以选择 I:\Jungle\verilog_projects\Digilent\accelerate，于是 Project File 栏中的路径变为 I:\Jungle\verilog_projects\Digilent\accelerate。点击 OK。

1-1-4. 新出现的是关于工程的一些参数设置的对话框，设置如下的参数后，点击 **Next**，如图 4。

architecture: artix 7
Device: XC7a1007
Package: CSG324
Speed: -3

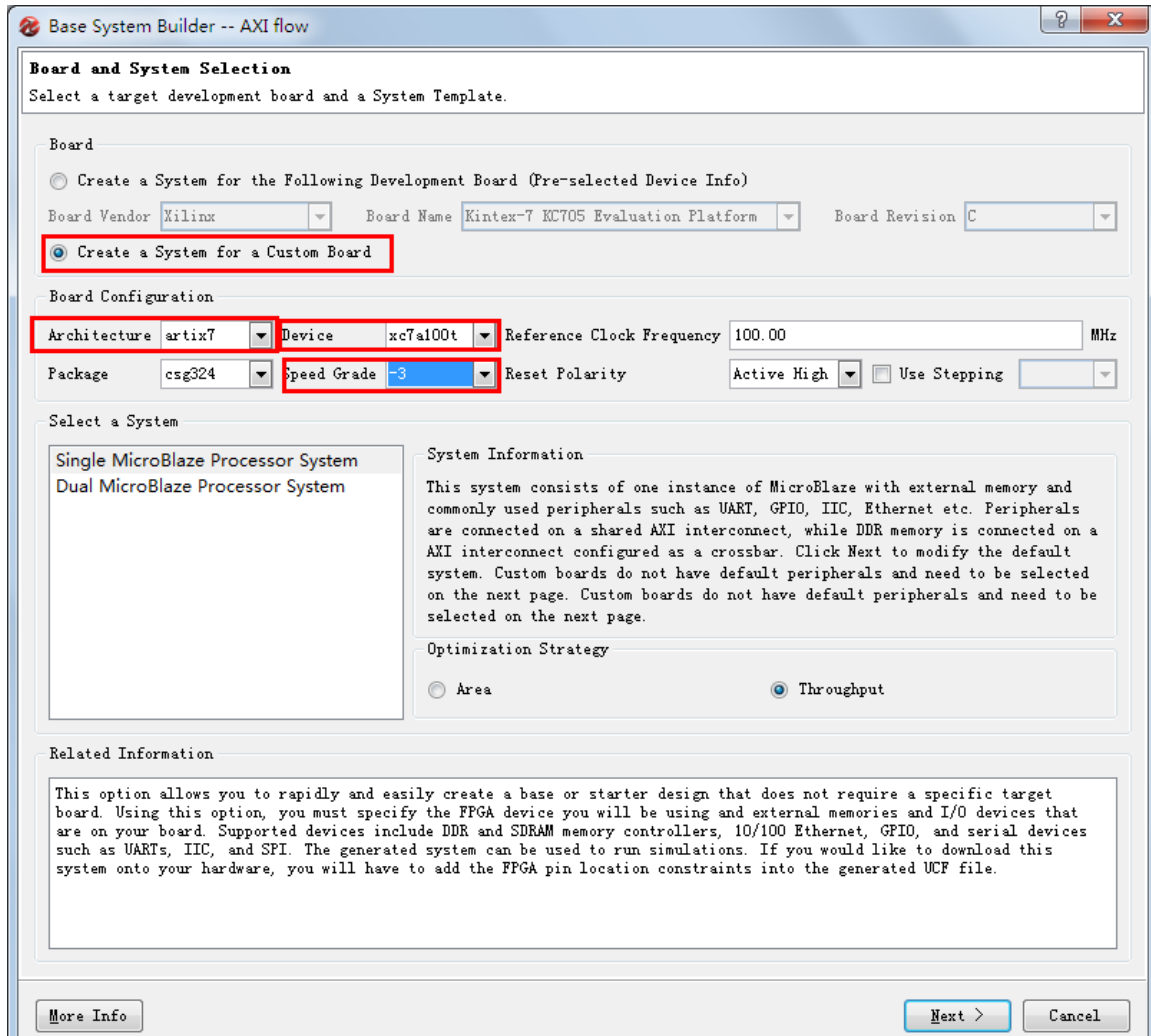
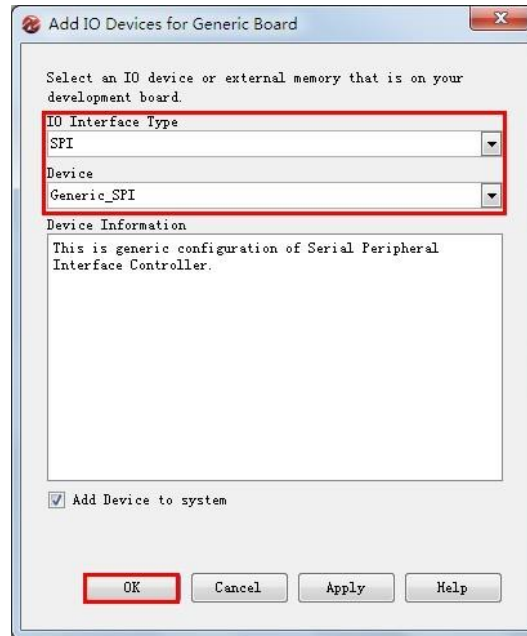


图 4：新工程参数设置

1-1-5. 在接下来出现的页面中选择要添加的 IP 核，并设置 IP 核的参数：

单击 Select and configure Peripherals 下的 Add Device...

然后按照下图所示，选择红色框内的选项，最后点击 OK



1-1-6. 单击 Select and configure Peripherals 下的 Add Device...

出现图 5 中的蓝色对话框。

在 IO Interface Type 中的下拉菜单中选择 UART。

在 Device 的下拉菜单中选择 RS232。

单击 OK

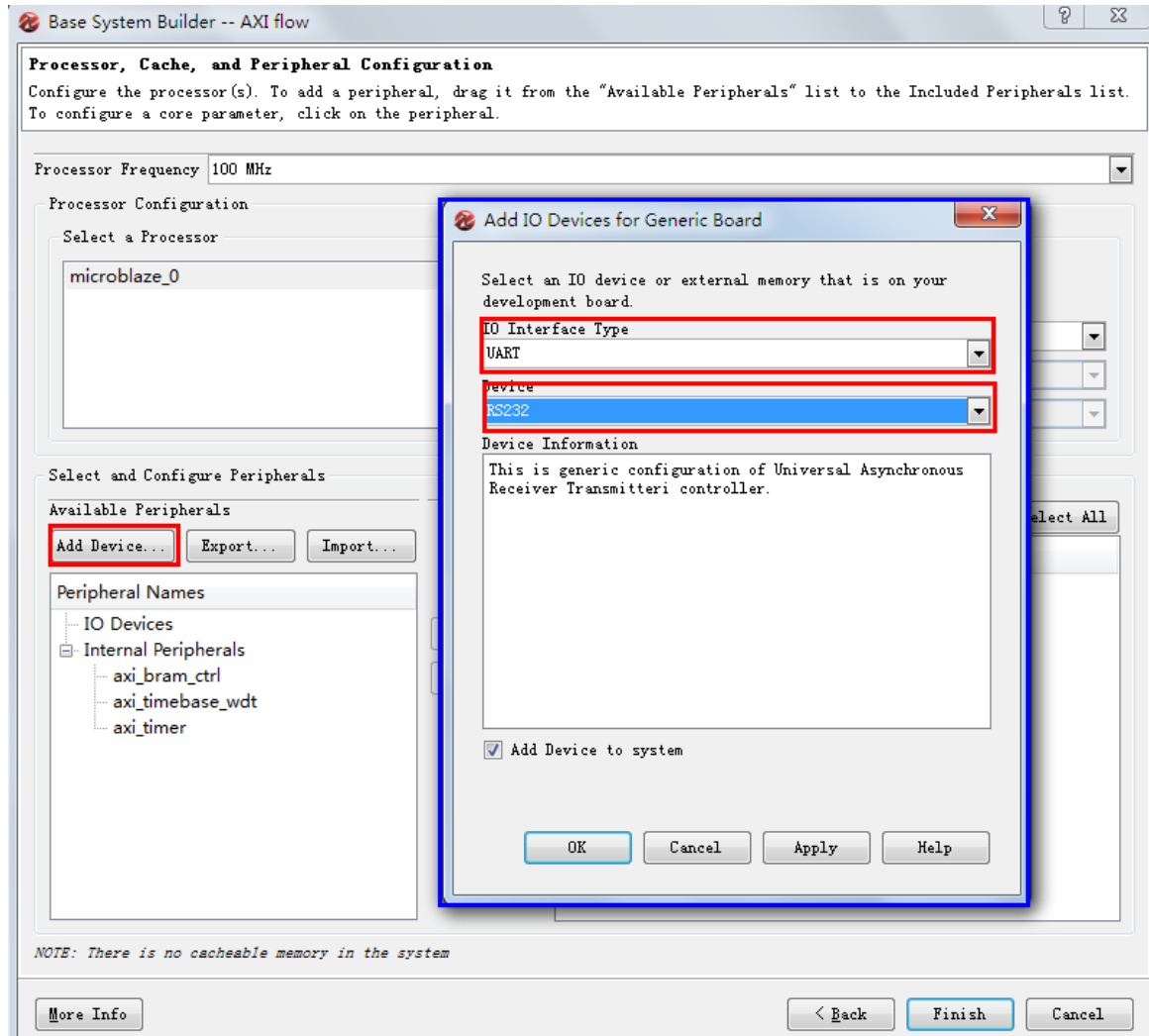


图 5. 添加串口的 IP

- 1-1-7.** 注意,串口的默认波特率设置为 9600。在这个试验中不做修改。以后的设计中根据需要进行调整。但是为了确保串口的正常通讯, SDK 工程中的 Terminal 的波特率以及串口的其他设置必须与之保持一致。

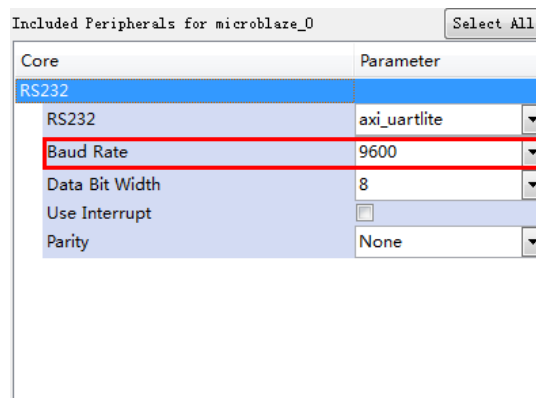


图 6. 串口的设置

第二步 进行端口的互连

2-1. 在 PORT 选项卡中修改时钟的相关设置

2-1-1. Port 选项卡（展开 External Port），如图 7.

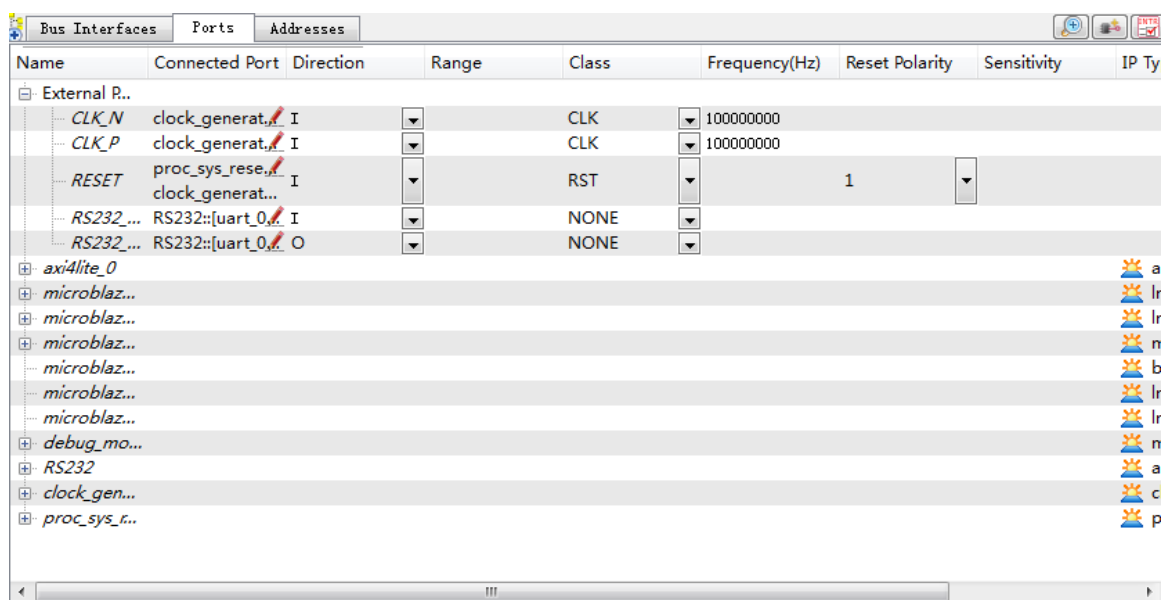


图 7: PORT 选项卡的初始状况

2-1-2. 将 External Port 中的 CLK_N 和 CLK_P 都去掉。

右键选中该端口，然后点击 Delete External Port，如图 8.

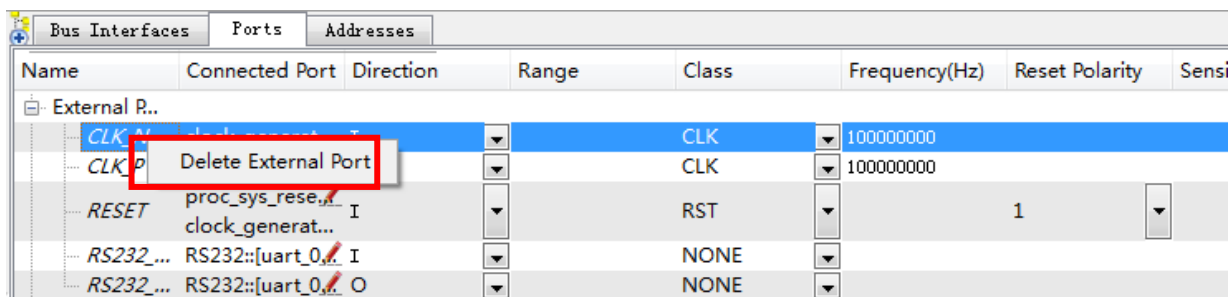


图 8: 源文件添加后的 ISE 用户界面

2-1-3. 将 Clock_generator_0 作为新的时钟，加入外部端口。

找到 Clock_generator_0 中的 CLKIN，右键选中，在菜单中点击 Make external

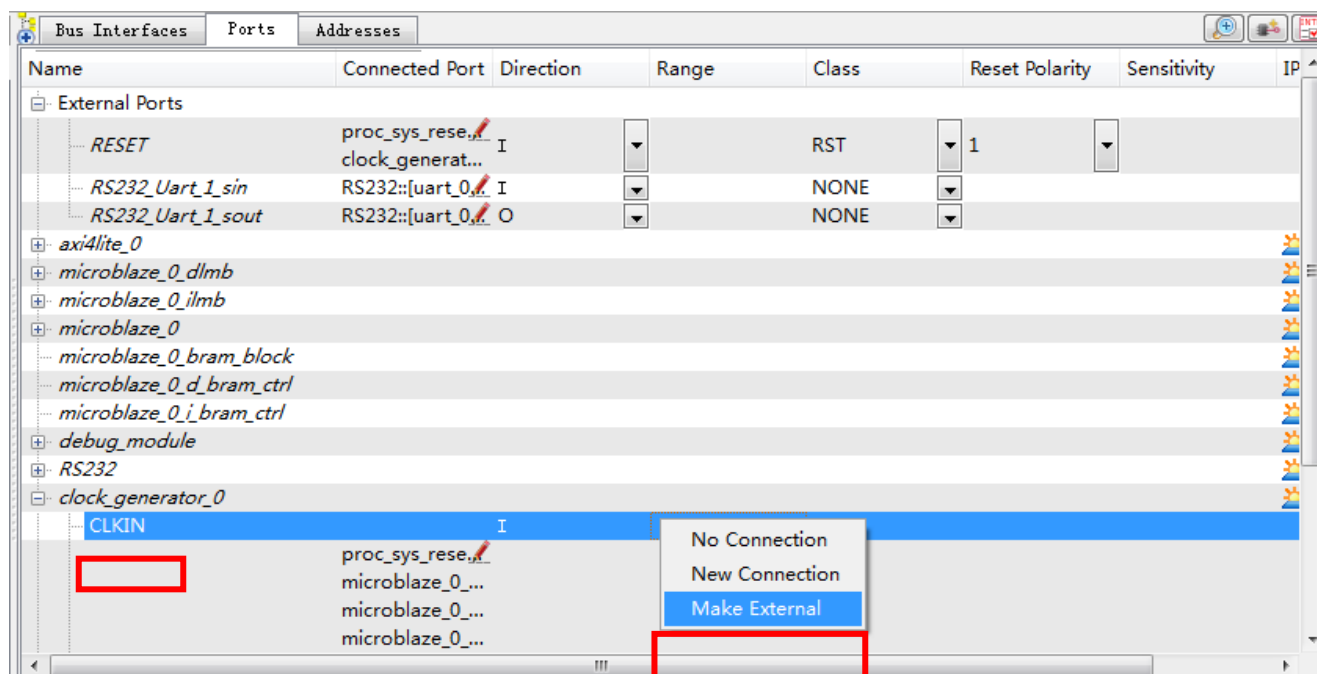


图 9: Clock_generator_0 中的 CLKIN

注意 External 中的 Name 一项，这是我们添加用户约束文件（UCF）的依据。

Name	Connected Port	Direction	Range	Class	Reset Polarity	Se
External Ports						
RESET	proc_sys_res...	I		RST	1	
RS232_Uart_1_sin	RS232::[uart_0...	I		NONE		
RS232_Uart_1_sout	RS232::[uart_0...	O		NONE		
clock_generator_0_CLKIN_pin	clock_generat...	I		CLK		

图 10: 修改后的 External PORT

第三步 添加用户约束文件

3-1. 打开初始 UCF 文件，根据需求进行修改

- 3-1-1. 在页面偏左找到 IP catalogue / Project 选项卡，双击 UCF File: DATA\lab1.ucf，ucf 文件在右侧打开

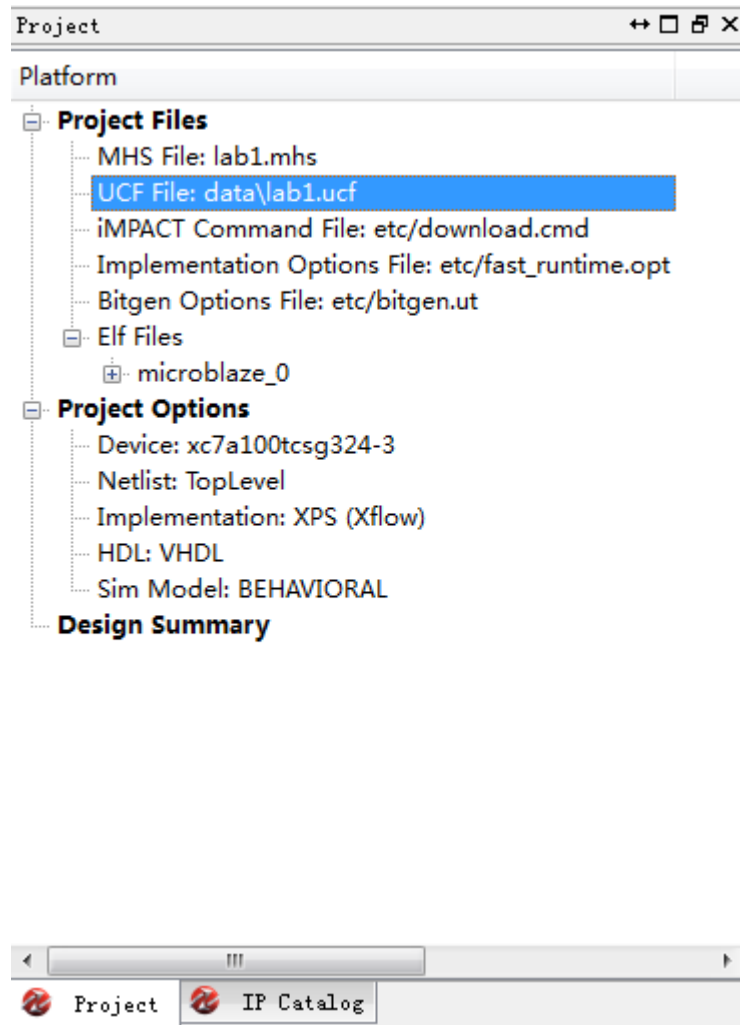


图 11: UCF 文件的位置

3-1-2. 这里我们手动输入 LOC（引脚位置）约束代码，如图 12。点击保存。

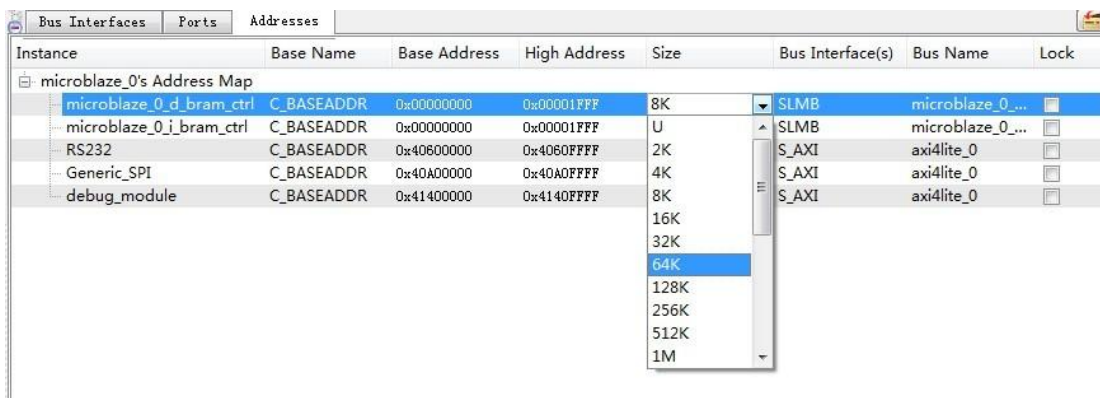
```

1
2  ## Clock signal
3  NET "clock_generator_0_CLKIN_pin" LOC = "E3" | IOSTANDARD = "LVCMOS33";
4  NET "clock_generator_0_CLKIN_pin" TNM_NET = sys_clk_pin;
5  TIMESPEC TS_sys_clk_pin = PERIOD sys_clk_pin 100 MHz HIGH 50%;
6
7  ## Switches
8  NET "RESET" LOC = "U9" | IOSTANDARD = "LVCMOS33"; #Bank = 34, Pin
9
10 ## Accelerometer
11 NET "SPI_FLASH_MISO" LOC = "D13" | IOSTANDARD = "LVCMOS33"; #Bank =
12 NET "SPI_FLASH_MOSI" LOC = "B14" | IOSTANDARD = "LVCMOS33"; #Bank =
13 NET "SPI_FLASH_SCLK" LOC = "D15" | IOSTANDARD = "LVCMOS33"; #Bank =
14 NET "SPI_FLASH_SS" LOC = "C15" | IOSTANDARD = "LVCMOS33"; #Bank =
15
16 ## USB-RS232 Interface
17 NET "RS232_Uart_1_sin" LOC = "C4" | IOSTANDARD = "LVCMOS33"; #
18 NET "RS232_Uart_1_sout" LOC = "D4" | IOSTANDARD = "LVCMOS33"; #
19

```

图 12: UCF 文件

3-1-3. 接下来要扩大板卡的存储空间，选择“Address”选项卡，然后选择 **microblaze_0_bram_ctrl** 的 **Size** 属性，从默认的 8K 改到 64K 对于这个实验来讲就足矣了，如下图所示。



3-1-4. 保存之后将工程导入到 SDK

在页面左边，点击 **Export Design**，如图 13。

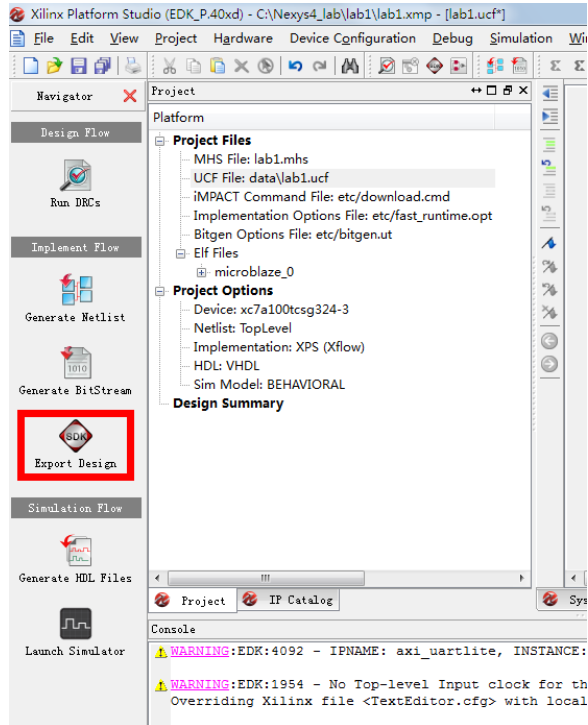


图 13: export design

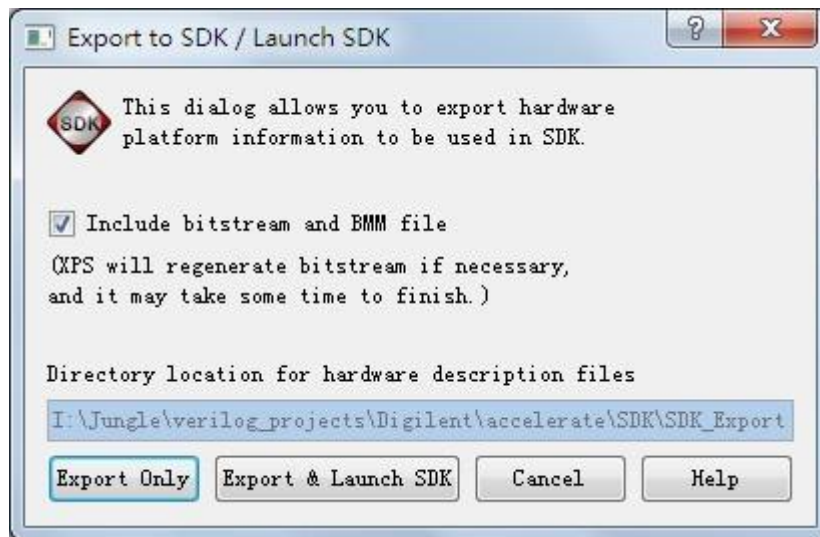


图 14: 在弹出的对话框中选择 Export & launch sdk

3-1-4. 选择 SDK 导入路径

注意要具体到..lsdk\sdk_export, 如图 14

点击 ok

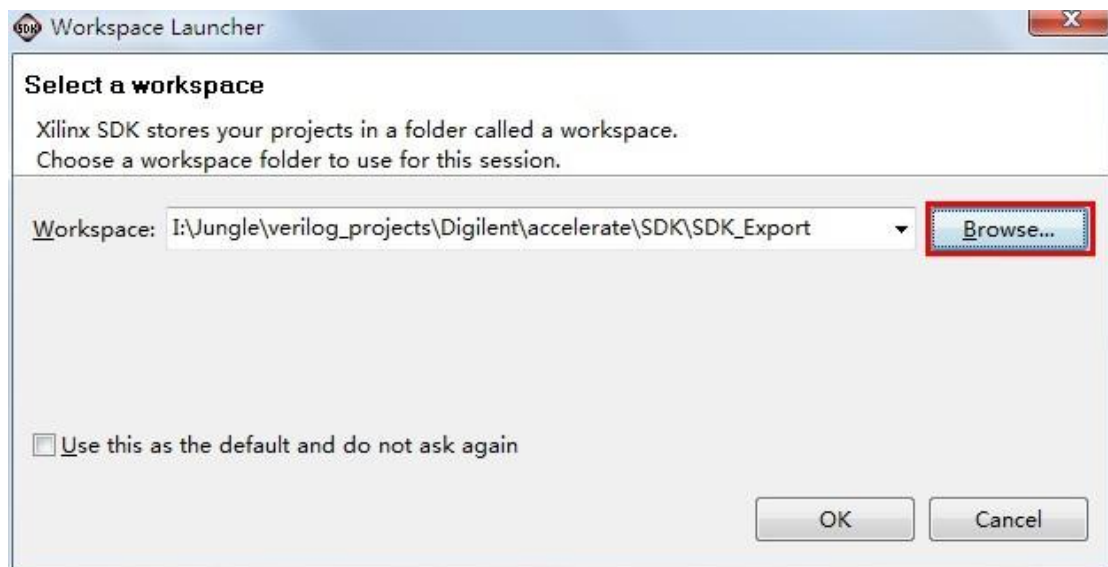


图 14: 导入 sdk 的工作空间选择

第四步 添加 app

4-1. 添加软件应用。

4-1-1. 在 SDK 的用户界面中，选择 **file—new—application project**，如图 15。

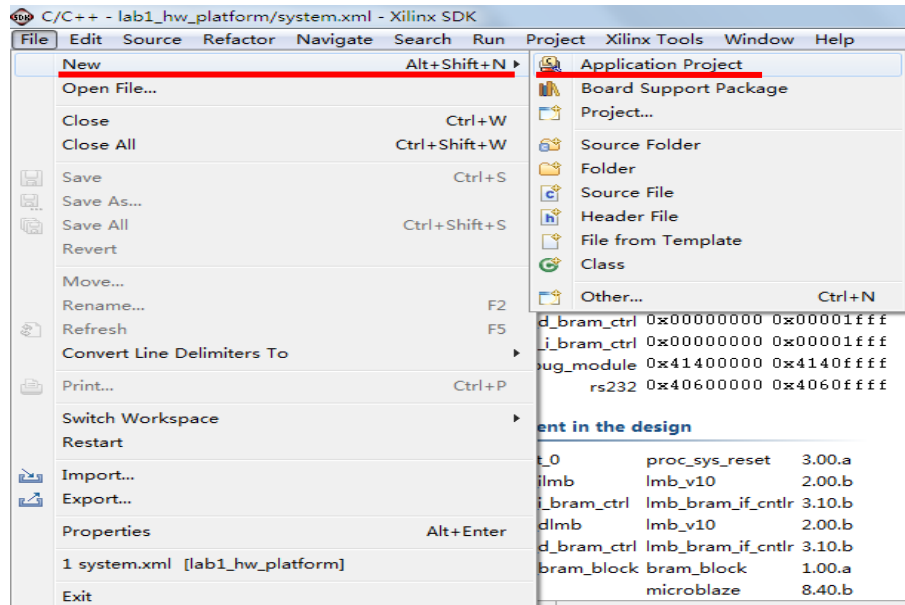


图 15: 新建软件应用

4-1-2. 输入工程的名称，这里使用 **acc**，同样不要包含空格和中文，点击 **next**

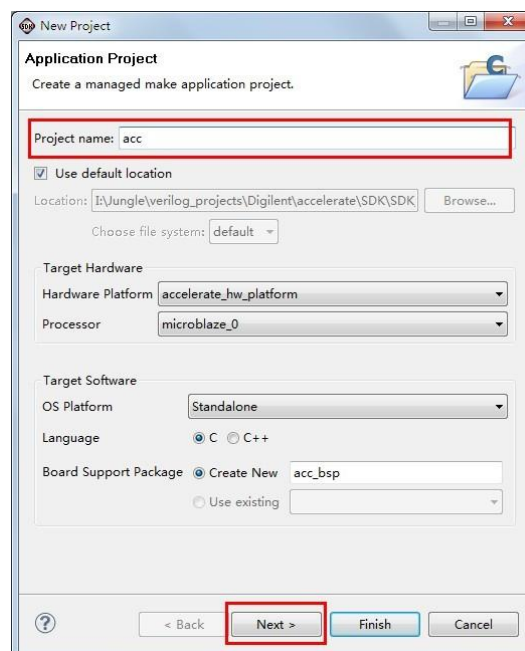


图 16: 新建工程---命名

4-1-3. 在下一步弹出的对话框中选择 **hello world**，然后点击 **finish**

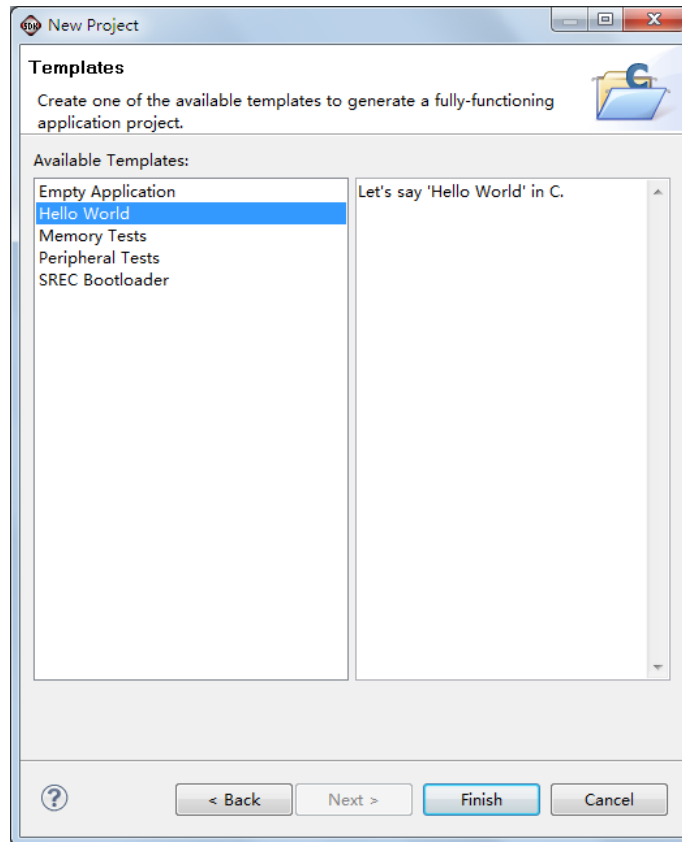


图 16: 选择 Helloworld 示例代码

4-1-4. 添加完毕以后可以在左侧双击源文件，查看这段代码：

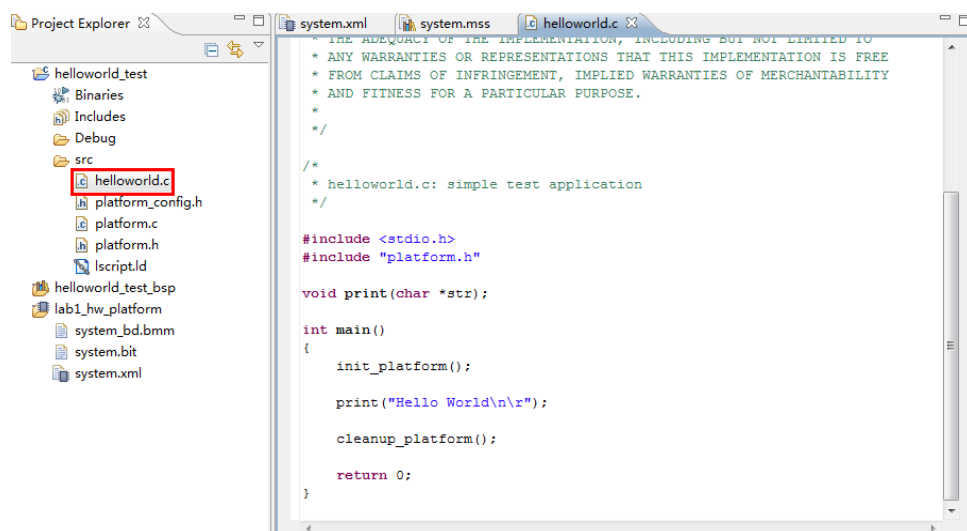


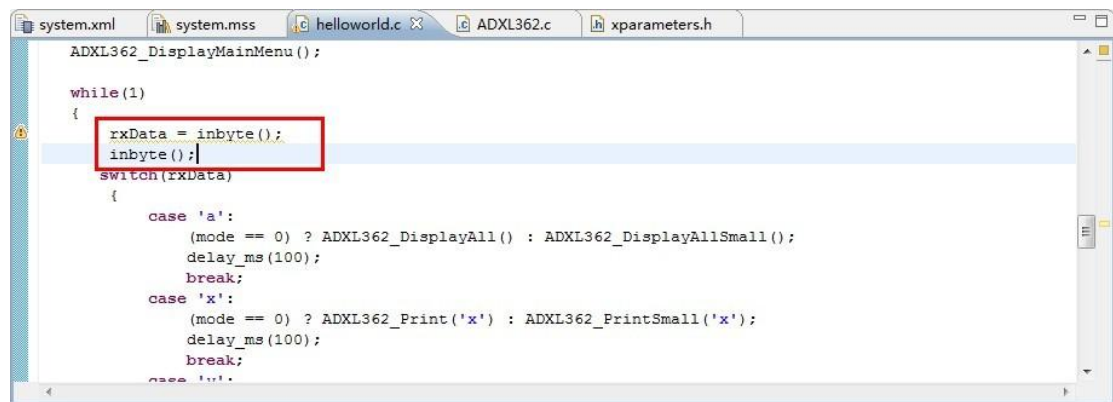
图17: helloworld源代码

4-1-5.将 **src** 目录中的文件解压，解压后的文件中的 **sw** 目录中的代码文件复制到 **acc** 工程下的 **src** 目录中（**testperiph.c** 不要复制）

将 **testperiph.c** 的代码复制到 **helloworld.c** 中（覆盖原有的 **helloworld.c** 中的代码）

名称	修改日期	类型	大小
ADXL362.c	2013/1/29 23:19	C Source	17 KB
ADXL362.h	2013/1/29 23:06	C/C++ Header	8 KB
lscript.ld	2013/11/26 10:15	LD 文件	5 KB
platform.c	2013/11/26 10:15	C Source	4 KB
platform.h	2013/11/26 10:15	C/C++ Header	1 KB
platform_config.h	2013/11/26 10:15	C/C++ Header	1 KB
spi.c	2013/1/29 23:23	C Source	9 KB
spi.h	2013/1/29 23:06	C/C++ Header	5 KB

然后打开更改后的 **helloworld.c** 文件，按照下图添加两行代码，然后保存所有文件（ctrl+s）：



```
ADXL362_DisplayMainMenu();

while(1)
{
    rxData = inbyte();
    inbyte();
    switch(rxData)
    {
        case 'a':
            (mode == 0) ? ADXL362_DisplayAll() : ADXL362_DisplayAllSmall();
            delay_ms(100);
            break;
        case 'x':
            (mode == 0) ? ADXL362_Print('x') : ADXL362_PrintSmall('x');
            delay_ms(100);
            break;
        case '!':
```


4-1-6. 代码 main 函数简述

```

□ /*****
 * @brief Main function.
 *
 * @return Always returns 0.
 *****/
□ int main()
{
    Xil_ICacheEnable();
    Xil_DCacheEnable();
    // Enable Interrupts
    microblaze_register_handler((XInterruptHandler)uartIntHandler, (void *)0);
    Xil_Out32(XPAR_RS232_UART_1_BASEADDR+0xC, (1 << 4));
    microblaze_enable_interrupts();
    // Initialize SPI
    SPI_Init(SPI_BASEADDR, 0, 0, 0);
    // Software Reset
    ADXL362_WriteReg(ADXL362_SOFT_RESET, ADXL362_RESET_CMD);
    delay_ms(10);
    ADXL362_WriteReg(ADXL362_SOFT_RESET, 0x00);
    // Enable Measurement
    ADXL362_WriteReg(ADXL362_POWER_CTL, (2 << ADXL362_MEASURE));
    ADXL362_DisplayMainMenu();
}
```

以上这段代码主要功能是开启 **cache** 和设置处理器等中断，初始化 **spi** 以及其他的预配置项

```

while(1)
{
    switch(rxData)
    {
        case 'a':
            (mode == 0) ? ADXL362_DisplayAll() : ADXL362_DisplayAllSmall();
            delay_ms(100); break;
        case 'x':
            (mode == 0) ? ADXL362_Print('x') : ADXL362_PrintSmall('x');
            delay_ms(100); break;
        case 'y':
            (mode == 0) ? ADXL362_Print('y') : ADXL362_PrintSmall('y');
            delay_ms(100); break;
        case 'z':
            (mode == 0) ? ADXL362_Print('z') : ADXL362_PrintSmall('z');
            delay_ms(100); break;
        case 't':
            ADXL362_PrintTemp(); break;
        case 'r':
            ADXL362_SetRange(); break;
        case 's':
            ADXL362_SwitchRes(); break;
        case 'm':
            ADXL362_DisplayMainMenu(); break;
        case 'i':
            ADXL362_PrintID(); break;
        case 0:
            break;
        default:
            xil_printf("\n\r> Wrong option! Please select one of the options below");
            ADXL362_DisplayMenu(); break;
    }
}
Xil_DCacheDisable();
Xil_ICacheDisable();
return 0;
}

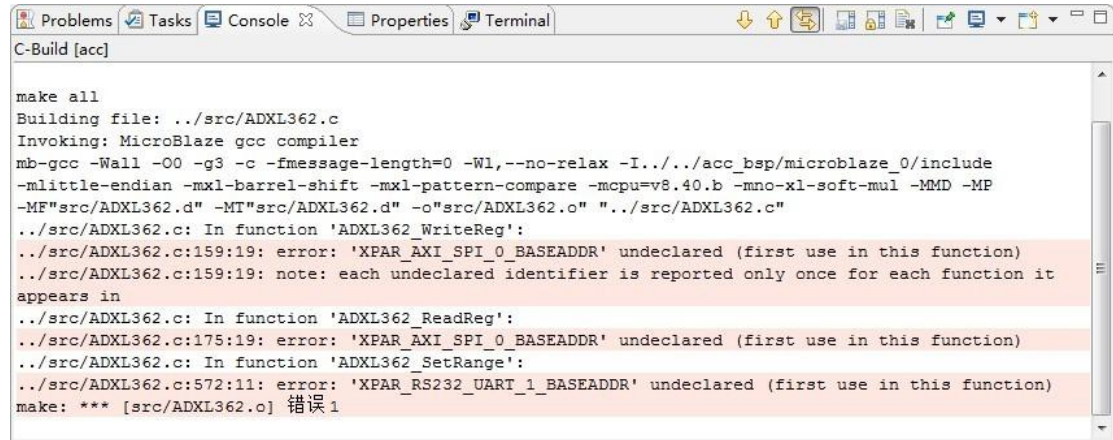
```

串口一直接收命令，从 switch 中可以看出：

命令	A	X	Y	Z	T	R	S	I	M
作用	显示三轴加速度	显示 x 轴加速度	显示 y 轴加速度	显示 z 轴加速度				显示设备 ID	显示主菜单

4-1-7. 保存后可能会有如下错误，如果有则一一按照如下过程改过，若没有则忽略之：

问题 1：

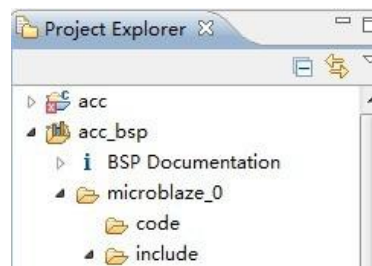


```
C-Build [acc]

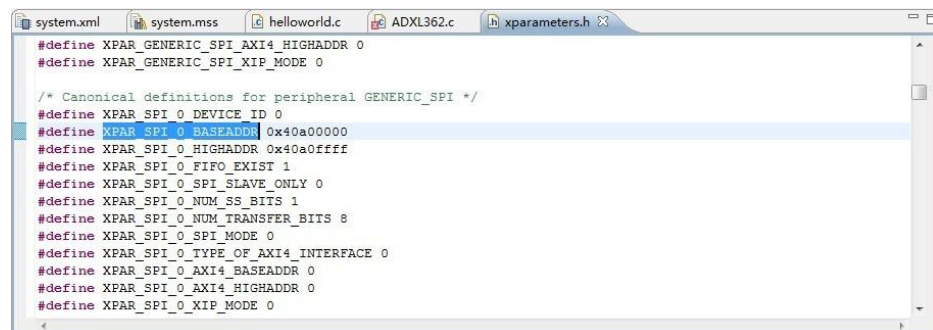
make all
Building file: ../src/ADXL362.c
Invoking: MicroBlaze gcc compiler
mb-gcc -Wall -O0 -g3 -c -fmessage-length=0 -Wl,--no-relax -I../acc_bsp/microblaze_0/include
-mlittle-endian -mxl-barrel-shift -mxl-pattern-compare -mcpu=v8.40.b -mno-xl-soft-mul -MMD -MP
-MF"src/ADXL362.d" -MT"src/ADXL362.d" -o"src/ADXL362.o" "../src/ADXL362.c"
../src/ADXL362.c: In function 'ADXL362_WriteReg':
../src/ADXL362.c:159:19: error: 'XPAR_AXI_SPI_0_BASEADDR' undeclared (first use in this function)
../src/ADXL362.c:159:19: note: each undeclared identifier is reported only once for each function it
appears in
../src/ADXL362.c: In function 'ADXL362_ReadReg':
../src/ADXL362.c:175:19: error: 'XPAR_AXI_SPI_0_BASEADDR' undeclared (first use in this function)
../src/ADXL362.c: In function 'ADXL362_SetRange':
../src/ADXL362.c:572:11: error: 'XPAR_RS232_UART_1_BASEADDR' undeclared (first use in this function)
make: *** [src/ADXL362.o] 错误 1
```

解决方案 1：

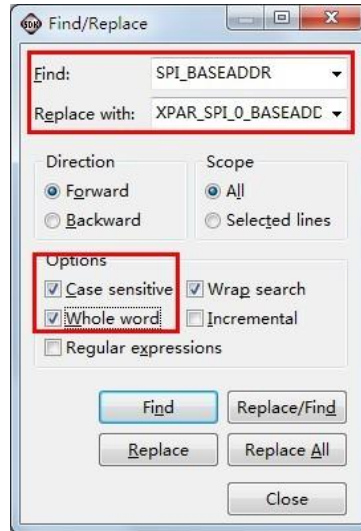
展开左侧的树状目录如下图所示，在#include 目录中找到 xparameters.h 文件，打开。



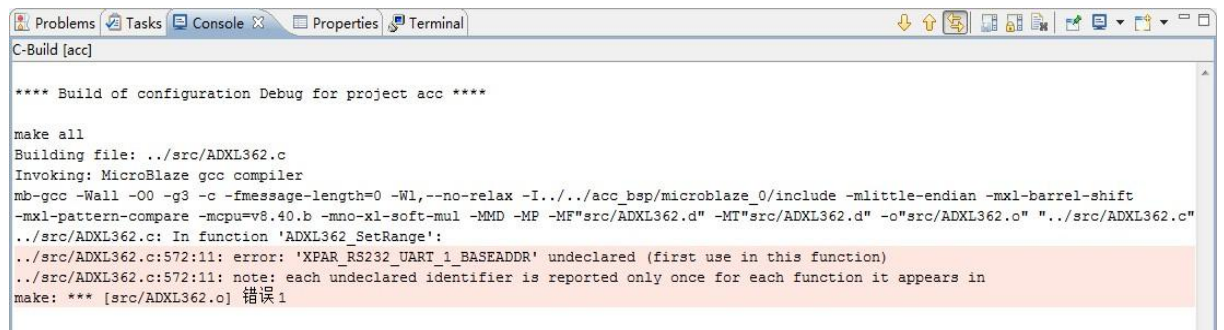
将选项卡切换到 xparameters.h 找到 xpara_spi_0_baseaddr（具体如下图所示）



将选项卡切换回 helloworld.c，按住 ctrl+f，按照下图所示配置，最后选择 replace all



问题 2:

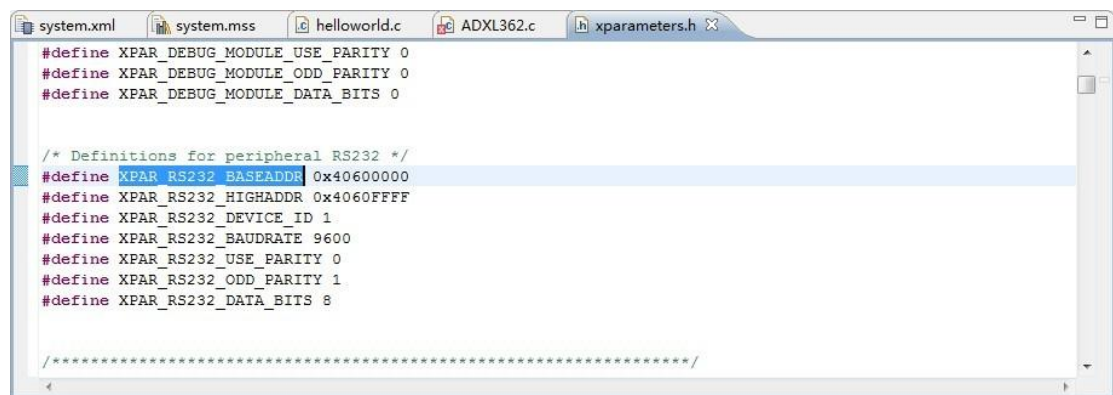


```
**** Build of configuration Debug for project acc ****

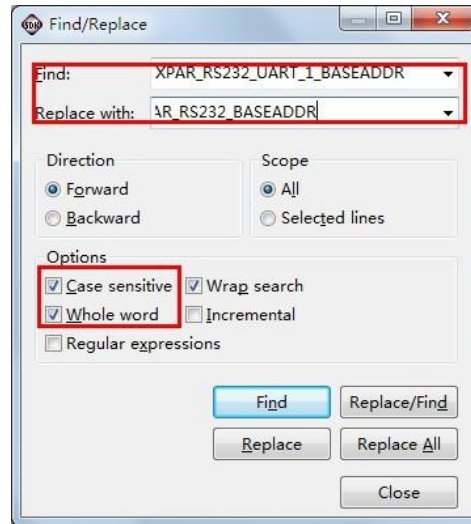
make all
Building file: ../src/ADXL362.c
Invoking: MicroBlaze gcc compiler
mb-gcc -Wall -O0 -g3 -c -fmessage-length=0 -Wl,--no-relax -I../acc_bsp/microblaze_0/include -mlittle-endian -mxl-barrel-shift
-mxl-pattern-compare -mcpu=v8.40.b -mno-xl-soft-mul -MMD -MP -MF"src/ADXL362.d" -MT"src/ADXL362.o" -o"src/ADXL362.o" "../src/ADXL362.c"
../src/ADXL362.c: In function 'ADXL362_SetRange':
../src/ADXL362.c:572:11: error: 'XPAR_RS232_UART_1_BASEADDR' undeclared (first use in this function)
../src/ADXL362.c:572:11: note: each undeclared identifier is reported only once for each function it appears in
make: *** [src/ADXL362.o] 错误 1
```

解决方案 2:

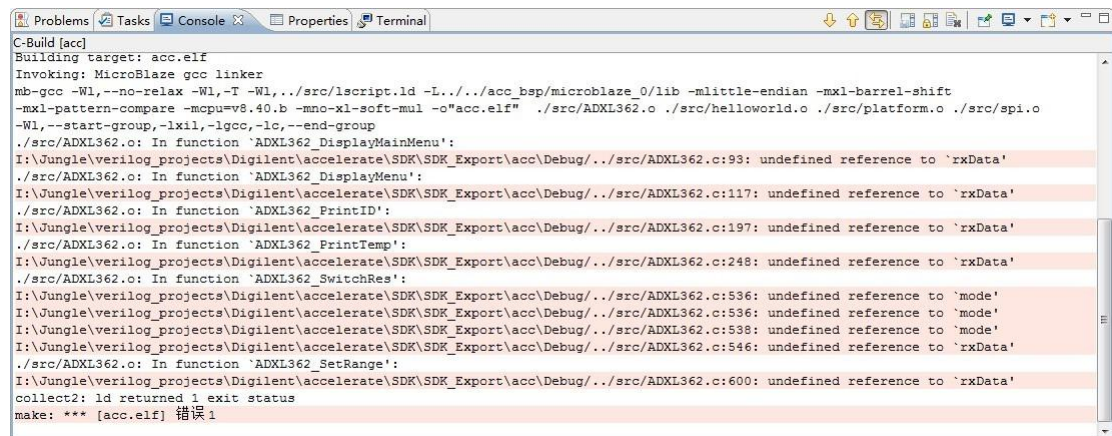
将选项卡切换到 xparameters.h 找到 xpar_rs232_baseaddr（具体如下图所示）



将选项卡切换回 helloworld.c，按住 ctrl+f，按照下图所示配置，最后选择 replace all



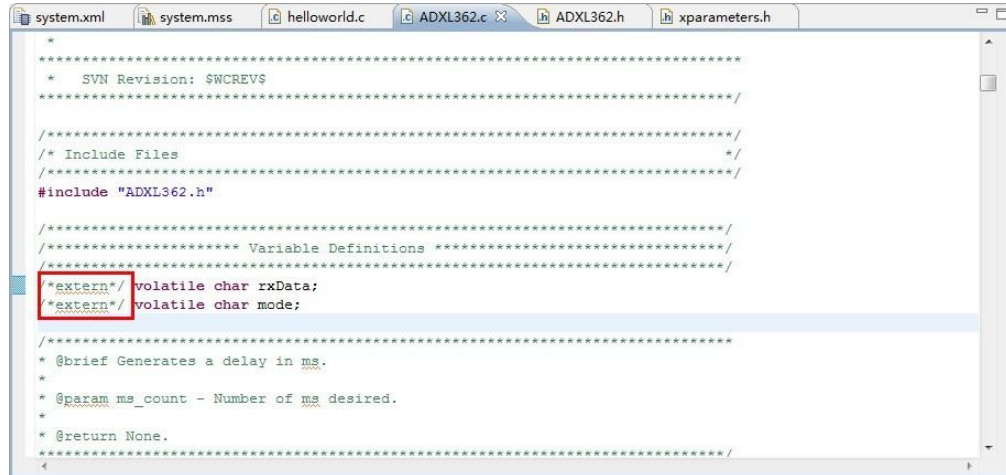
问题 3:



```
C-Build [acc]
Building target: acc.elf
Invoking: MicroBlaze gcc linker
mb-gcc -Wl,--no-relax -Wl,-T -Wl,../src/ldscript.ld -L../acc_bsp/microblaze_0/lib -mlittle-endian -mxl-barrel-shift
-mxl-pattern-compare -mcpu=v8.40.b -mno-xl-soft-mul -o"acc.elf" ./src/ADXL362.o ./src/helloworld.o ./src/platform.o ./src/spi.o
./src/ADXL362.o: In function 'ADXL362_DisplayMainMenu':
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:93: undefined reference to 'rxData'
./src/ADXL362.o: In function 'ADXL362_DisplayMenu':
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:117: undefined reference to 'rxData'
./src/ADXL362.o: In function 'ADXL362_PrintID':
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:197: undefined reference to 'rxData'
./src/ADXL362.o: In function 'ADXL362_PrintTemp':
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:248: undefined reference to 'rxData'
./src/ADXL362.o: In function 'ADXL362_SwitchRes':
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:536: undefined reference to 'mode'
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:536: undefined reference to 'mode'
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:538: undefined reference to 'mode'
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:546: undefined reference to 'rxData'
./src/ADXL362.o: In function 'ADXL362_SetRange':
I:\Jungle\verilog_projects\Diligent\accelerate\SDK\SDK_Export\acc\Debug\../src/ADXL362.c:600: undefined reference to 'rxData'
collect2: ld returned 1 exit status
make: *** [acc.elf] 错误 1
```

解决方案 3:

将选项卡切换到 `adxl362.c`, 将下图所示的 `extern` 关键字注释



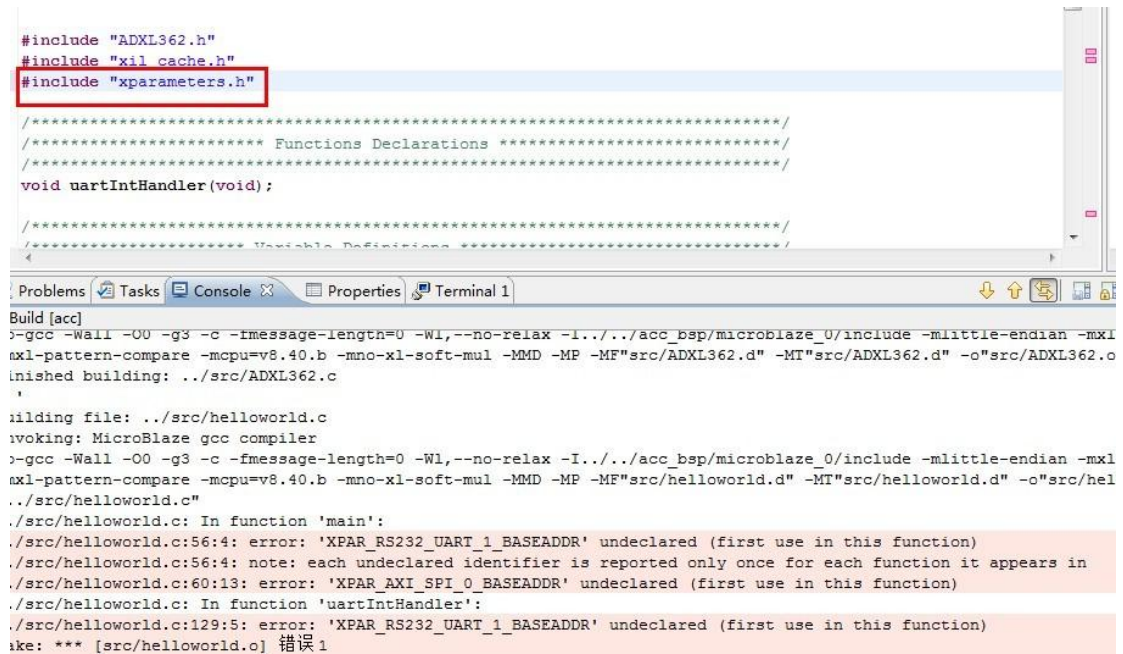
```
system.xml system.mss helloworld.c ADXL362.c ADXL362.h xparameters.h
*
* SVN Revision: $WCREV$
*
/*****
/* Include Files
/*****
#include "ADXL362.h"

/*****
/***** Variable Definitions *****/
/*****

extern volatile char rxData;
extern volatile char mode;

/*****
* @brief Generates a delay in ms.
*
* @param ms_count - Number of ms desired.
*
* @return None.
*****/
```

问题 4:



```
#include "ADXL362.h"
#include "xil_cache.h"
#include "xparameters.h"

/*****
/***** Functions Declarations *****/
/*****

void uartIntHandler(void);

/*****
/***** Variable Definitions *****/
/*****
```

Problems Tasks Console Properties Terminal 1

Build [acc]
\$-gcc -Wall -O0 -g3 -c -fmessage-length=0 -Wl,--no-relax -I../acc_bsp/microblaze_0/include -mlittle-endian -mxl
-mxl-pattern-compare -mcpu=v8.40.b -mno-xl-soft-mul -MMD -MP -MF"src/ADXL362.d" -MT"src/ADXL362.d" -o"src/ADXL362.o"
inished building: ../src/ADXL362.c

Building file: ../src/helloworld.c
Invoking: MicroBlaze gcc compiler
\$-gcc -Wall -O0 -g3 -c -fmessage-length=0 -Wl,--no-relax -I../acc_bsp/microblaze_0/include -mlittle-endian -mxl
-mxl-pattern-compare -mcpu=v8.40.b -mno-xl-soft-mul -MMD -MP -MF"src/helloworld.d" -MT"src/helloworld.d" -o"src/hel
../src/helloworld.c"

../src/helloworld.c: In function 'main':
../src/helloworld.c:56:4: error: 'XPAR_RS232_UART_1_BASEADDR' undeclared (first use in this function)
../src/helloworld.c:56:4: note: each undeclared identifier is reported only once for each function it appears in
../src/helloworld.c:60:13: error: 'XPAR_AXI_SPI_0_BASEADDR' undeclared (first use in this function)
../src/helloworld.c: In function 'uartIntHandler':
../src/helloworld.c:129:5: error: 'XPAR_RS232_UART_1_BASEADDR' undeclared (first use in this function)
make: *** [src/helloworld.o] 错误 1

解决方案 4:

按照上图中，将 xparameters.h 加入（在 helloworld.c 中加入）

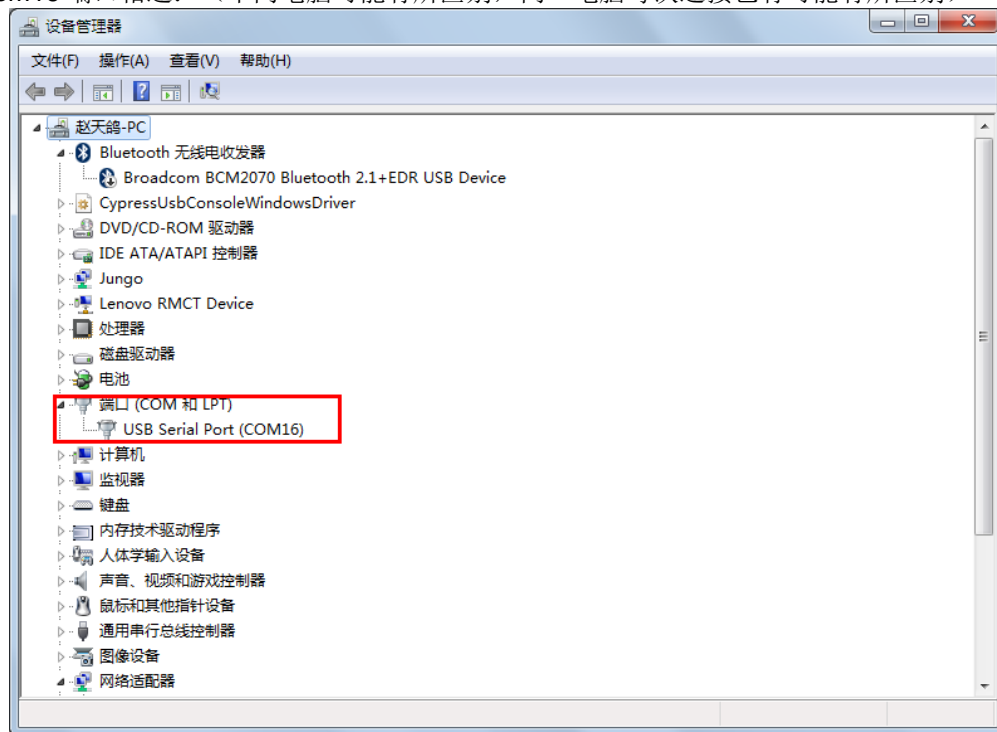
第五步 上板验证

5-1. 将 Nexys4 与 Pc 的 USB 接口连接

5-2. 查看端口号：

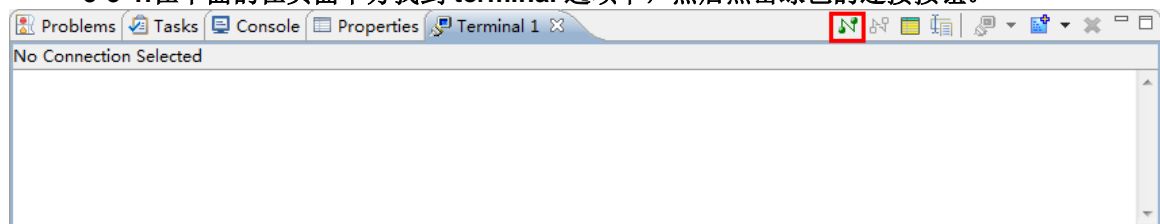
右键“我的电脑”——“属性”——在页面左侧选择“设备管理器”

发现与 com16 端口相连：（不同电脑可能有所区别，同一电脑每次连接也有可能有所区别）

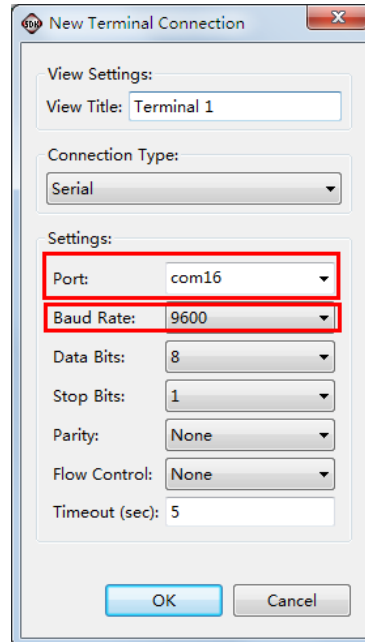


5-3. 在 SDK 中打开串口：

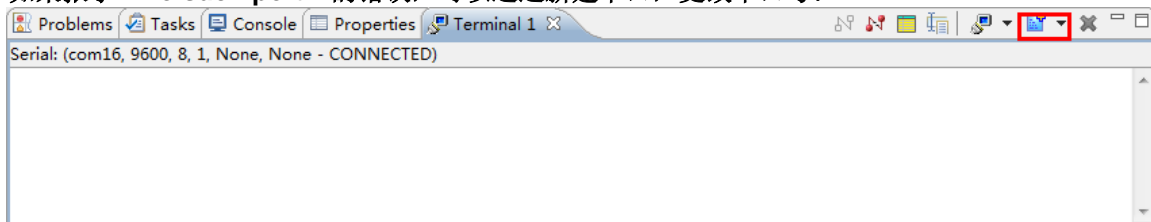
5-3-1. 在下面的在页面下方找到 **terminal** 选项卡，然后点击绿色的连接按钮。



5-3-2. 按照端口号和 XPS 中的波特率（**baud rate**）进行如下设置：

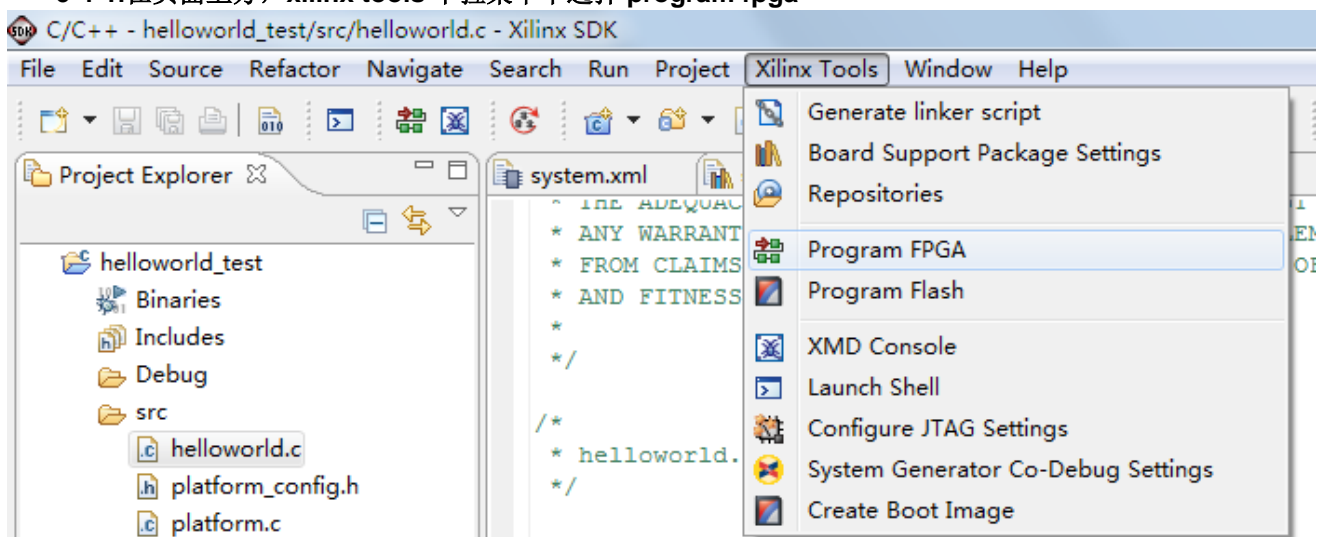


如果报了“no such port”的错误，可以通过新建串口，更改串口号：

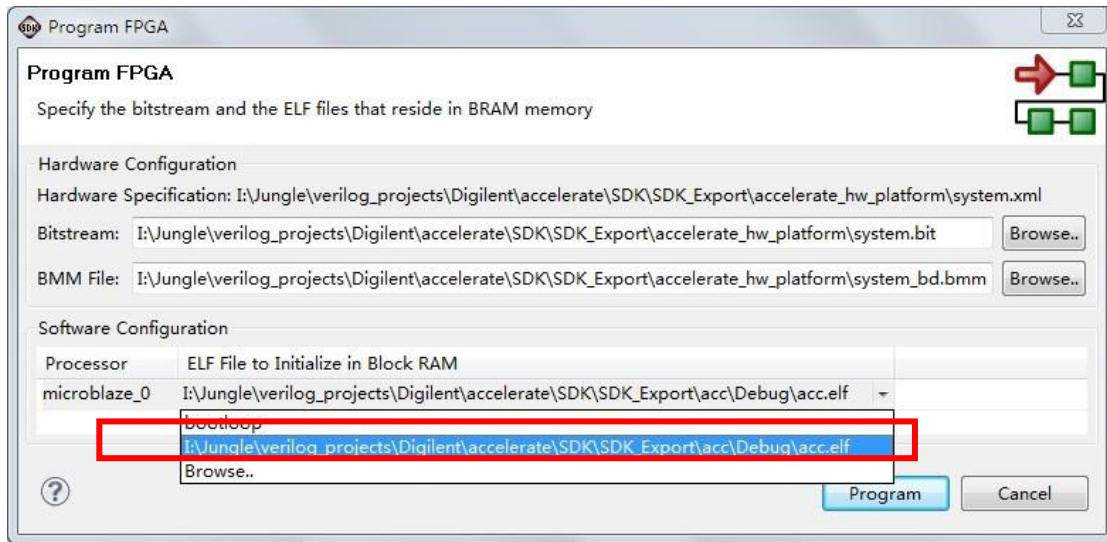


5-4. 将程序下载到板子上并运行

5-4-1. 在页面上方，xilinx tools 下拉菜单中选择 program fpga



5-4-2. 注意要选择正确的 elf 文件:



点击 program

5-5. 在串口中看到结果:



上图为在串口中输入 a 时，串口会相应显示出 x, y, z 三轴的加速度数据，其他的命令可参看右侧相应的菜单解释进行选择。