**《计算机网络》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | |  | | | **姓名** |  |
| **实验题目** | **Ethernet多端口交换机设计与开发** | | | | | |
| **实验时间** | **2024.10.21** | | **实验地点** | **DS3 304** | | |
| **实验成绩** |  | | **实验性质** | **□验证性 □设计性 综合性** | | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | | | | |
| 1. 实验目的  * 了解硬件系统开发流程。 * 熟悉Vivado开发平台。 * 深入了解交换机的工作原理，以及如何设计与实现一个真实的交换机系统， * 同时验证交换机的基础功能。 | | | | | | |
| 二、实验项目内容  利用电脑和Xilinx ZBOX板卡实现Ethernet交换机核心功能。具体功能包括如下：   * ARM处理器模块功能实现，实现CPU处理功能 * MAC模块功能实现，实现网络数据的接收发送基本功能 * DMA模块功能实现，实现MAC模块与存储器间的数据交换 * AXI互联模块功能实现，实现不同Ethernet端口间通信互联 * 交换机功能模块实现，实现交换机端口的状态管理维护、地址转发表的查询与管理维护、基于源地址学习的转发表学、以及帧收发等功能 * 交换机配置软件实现。实现交换机的管理配置和操作命令 | | | | | | |
| 三、实验过程或算法（源程序）  1.交换功能IP核完善封装  打开并解压文件夹Lab\src\_zynq\_switch\src\_znet\znet工程。  在zwitch\_i - zswitch模块中打开packet\_pipeline\_v1\_0的ip核编辑工程界面  查看代码并理解后再次封装IP  eth\_parser.v: 解析数据包，从数据包中提取出源端口、源mac地址和目的mac地址等信息。  mac\_cam\_lut.v: 根据源MAC地址实现地址自学习，根据目的MAC地址获取输出端口。  2.实验工程中更新交换功能IP核  在IP status窗口选中packet\_pipeline\_v1\_0\_0，点击“Upgrade Selected”对工程中的交换功能IP核进行更新  生成Output Products  3.重新生成顶层文件，综合实现生成bit文件  生成bit流  导出bit文件至linux文件夹   1. 创建Boot.bin启动文件   先打开Xilinx SDK工具，点击“Xilinx Tools”菜单选项，选择“Create Boot Image” ，再在弹出的“Create Boot Image”界面，在Output BIF file path中选择生成文件存放的路径名  然后按顺序添加三个文件：zynq\_fsbl.elf，zynq\_switch.bit，u-boot.elf  在Boot image partitions里点击Add，在弹出的Add partition对话框里，看到Partition type为bootloader时，点击File path按钮，选中linux文件夹下的zynq\_fsbl.elf  再次点击Add，看到Partition type为datafile，选择FPGA镜像文件：zynq\_switch.bit  再次点击Add，看到Partition type为datafile，选择u-boot文件  点击“Create Image”按钮，生成Boot.bin文件，在linux文件夹下面可以看到BOOT.bin文件  5.交换机配置软件开发  创建编译petalinux工程  在虚拟机新建文件夹zynq\_switch，并进入到该文件夹目录下  在虚拟机新建文件夹zynq\_switch，并进入到该文件夹目录下  mkdir zynq\_switch  cd zynq\_switch  在使用Petalinux工具之前应该首先指明相应的环境变量，在Linux终端中执行以下命令：**source /opt/pkg/petalinux/settings.sh**    将“\*\*\*\_hw\_platform\_0”这个文件夹，复制到安装Petalinux的Linux 系统中的zynq\_switch文件夹中    创建一个petalinux工程：petalinux-create --type project --template zynq --name zbox  lcd到zbox    给petalinux工程导入硬件工程描述：进入znet工程文件夹，通过petalinux-config --get-hw-description=”../zynq\_nic\_wrapper\_hw\_platform\_0”文件夹路径，命令cd导入HDF文件。  进入zbox工程文件夹，配置内核：petalinux-config -c kernel  选择Device Drivers ->  选择<\*> Userspace I/O drivers  <\*> Userspace I/O platform driver with generic IRQ handling(Y)  <\*> Userspace platform driver with generic irq and dynamic memory(Y)  编译Linux系统：petalinux-build -v  修改Device tree文件： project-spec/meta-user/recipes-bsp/device-tree/files/system-user.dtsi  完成如下两个配置  1.设备每个网络端口的MAC地址，物理PHY器件类型。   1. 指定交换模块是UIO驱动。   编译完成后，在zbox工程路径下创建switch-config应用：  petalinux-create -t apps --template c --name switch-config --enable  打开自动生成的应用源文件：  zbox/project-spec/meta-user/recipes-apps/switch-config/files/switch-config.c，把提供的switch-config.c文件中的代码复制到上述文件，保存。  在zbox工程路径下创建router-config应用：petalinux-create -t apps --template c --name router-config --enable  打开自动生成的应用源文件:  znet/project-spec/meta-user/recipes-apps/router-config/files/router-config.c，把提供的router-config.c文件中的代码复制到上述文件，保存  在zbox工程路径下，编译应用：petalinux-build -v  在zbox/images/linux路径下，可以看到：image.ub，上述两个应用已经封装进image.ub文件，将该文件拷贝到linux文件夹下  交换机配置软件开发这个部分，按部就班地按照实验指导书来便可，但有大量的时间用于等待软件执行程序上，比如下面的一步可能要执行20分钟之久，类似的操作还有数个，耗时比较长。  正如软件提示那样：“This may take time!” | | | | | | |
| 1. 实验结果及分析和（或）源程序调试过程   **交换机功能验证**   1. **验证开发板能否正常启动系统**   为了验证交换机的功能，我们需要TypeC连接线连接电脑，将指定文件通过SD卡读卡器来进行烧录。  首先我们确保SD卡是FAT分区格式，方便烧录。利用读卡器在电脑上读取SD卡内容，然后替换SD卡FAT分区的BOOT.bin文件。  确认BOOT.bin文件烧录成功后，进行线路连接，连接好对应的线路后进行启动。发现两个LED灯点亮，绿色灯指示电源正常，蓝色灯指示Linux系统启动正常。  开发板能够正常启动系统。   1. **验证能否成功连接开发板**   打开串口调试工具putty，选择对应端口号、比特率、连接类型后进行登录。  端口号选择：COM4，比特率设置：115200，连接类型选择：Serial。  可以看到成功登录，与开发板连接成功。   1. **验证交换机软件配置**   使用switch-config来读写交换机寄存器，进行表的查找。  依次使用命令  查询交换机寄存器：switch-config -r 5  查询交换机查找表：switch-config -r reg\_num  配置交换机查找表：switch-config -a reg\_num -w reg\_data    交换机寄存器与交换机查找表查询成功。   1. 最终验证   在进行最后的验证阶段时，我们尝试配置本机IP，却发现所有主机的IP地址均无法更改，因为IPv4协议因某种原因不能进行配置。因此本次实验仅对实验要求的前三个功能进行了实现与验证。后续经过助教和老师的复盘分析，学生端无法进行IP地址配置是因为学生端受到了教学管理平台的控制限制，因此本次实验的最终验证并未完成，略有遗憾。 | | | | | | |