# 骑驴玩儿漂移-FPGA之PYNQ - Z2 入门学习笔记1

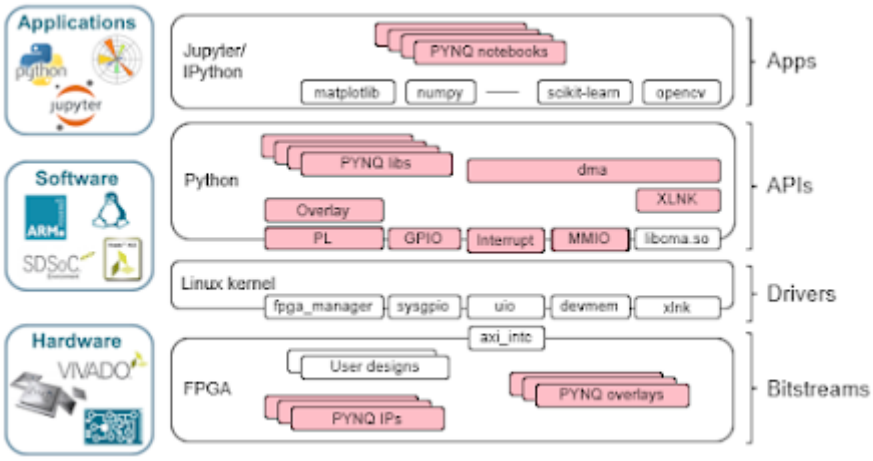
今天收到了PYNQ的板子，感谢赛灵思大佬的赞助，我有幸尝试一下FPGA的板子，之前拿到一块来自苏老师的小脚丫的板子，那是非常精致啊，但是没有入门，对FPGA很有兴趣，听大家说PYNQ支持python而且非常强大，我就想试试，正好遇到群里的赛灵思大佬，有幸得以尝试一下FPGA，不过我感觉我根本没用到fpga，都是在linux和python的边缘游走。哈哈。
好了，先做个介绍：

# PYNQ的来历

> PYNQ是Xilinx公司的开源项目® ，可以很容易地设计与赛灵思ZYNQ嵌入式系统® 系统上级芯片（SoC）。 使用Python语言和库，设计人员可以利用Zynq中可编程逻辑和微处理器的优势来构建更强大，更令人兴奋的嵌入式系统。PYNQ用户现在可以创建高性能的嵌入式应用程序。

## 特色：

> 并行硬件执行
> 高帧率视频处理
> 硬件加速算法
> 实时信号处理
> 高带宽IO
> 低延迟控制




## PYNQ旨在供各种设计人员和开发人员使用，包括：

希望利用Zynq和可编程硬件功能而无需使用ASIC式设计工具来设计硬件的软件开发人员。
系统架构师需要一个简单的软件界面和框架来快速进行原型设计和Zynq设计的开发。
希望他们的设计被最广泛的受众使用的硬件设计师。

## 关键技术：

Jupyter Notebook 是一个基于浏览器的交互式计算环境。可以创建Jupyter笔记本文档，包括实时代码，交互式小部件，绘图，说明文本，方程式，图像和视频。 可以使用Python在Jupyter Notebook中轻松编程支持PYNQ的Zynq板。
使用Python，开发人员可以 在可编程逻辑上使用硬件库和覆盖。 硬件库或覆盖可以加速Zynq板上运行的软件，并自定义硬件平台和接口。
PYNQ映像是可引导的Linux映像，包括 pynq Python包和其他开源包。

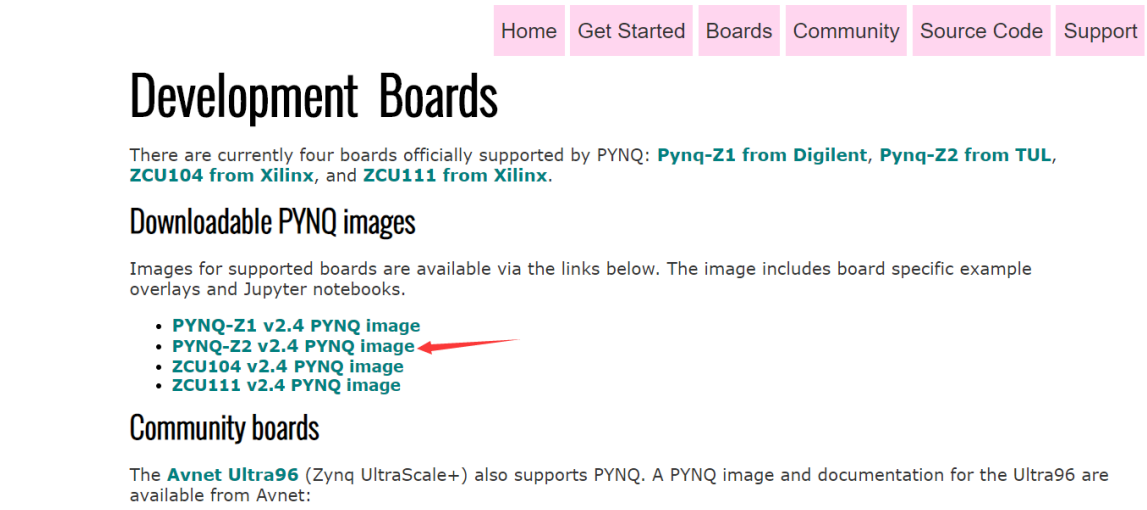
## 软件需求

> Jupyter笔记本 界面基于浏览器。 只需一个 兼容的Web浏览器 即可开始使用Python 编程 PYNQ。 为了获得更高的性能，您还可以将C / C ++与PYNQ结合使用。赛灵思SDK软件开发环境是免费提供的。您还可以使用第三方软件开发工具。 可以使用标准Xilinx和第三方硬件设计工具创建新的 硬件库和叠加层。Xilinx Vivado的免费WebPACK版本完全支持许多Zynq主板。

# 上面都是官方说明

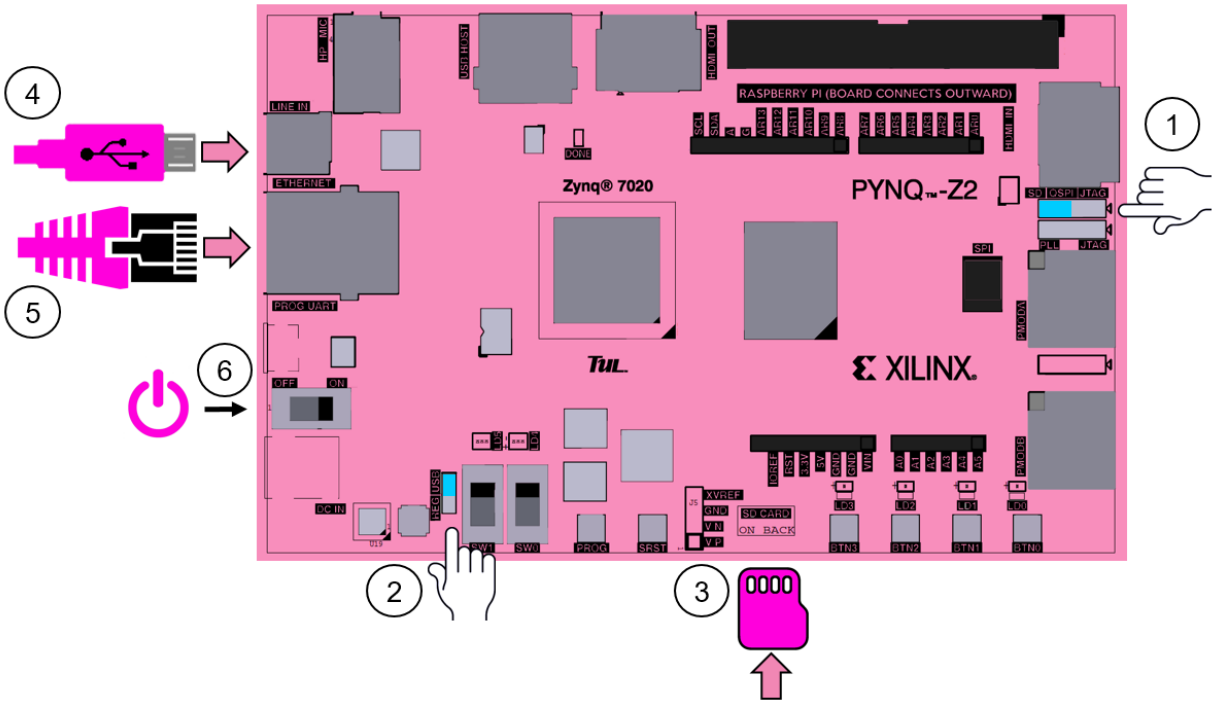
我来写一下我的操作步骤吧：

1. 访问：https://github.com/Xilinx/Pynq
2. 得到信息说明需要在官方下载PYNQ的镜像文件，然后下载镜像文件需要注册账号。
3. http://www.pynq.io/board.html 这里有下载文件的路径：


4. 


5. 注册账号
6. https://www.xilinx.com/registration/create-account.html
7. https://pynq.readthedocs.io/en/latest/getting_started.html

烧录镜像到卡上，然后按照下图调试好Jumper的位置

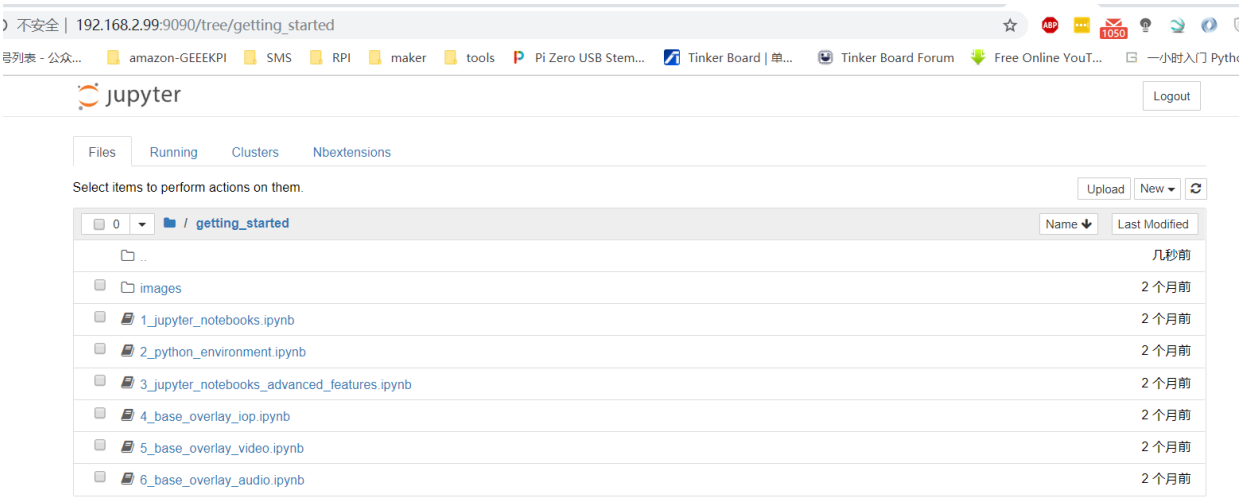



然后上电就启动了。将电源开关滑动到ON 。红色 LED会立刻上来就确认板上有电源。几秒钟后，黄色/绿色/完成 LED将亮起，表示设备正常运行。

一会儿你就会看到两个蓝色LED，就是板子上的LD4，LD5亮一下，和四个黄绿色 LED同时闪烁。系统现已启动并可供使用。

刚开始我选择的方式是：一根网线连着PYNQ和我的电脑，然后电脑本地网卡IP地址设置为192.168.2.1/24，因为xilinx的板子默认的地址和端口号是：192.168.2.99:9090

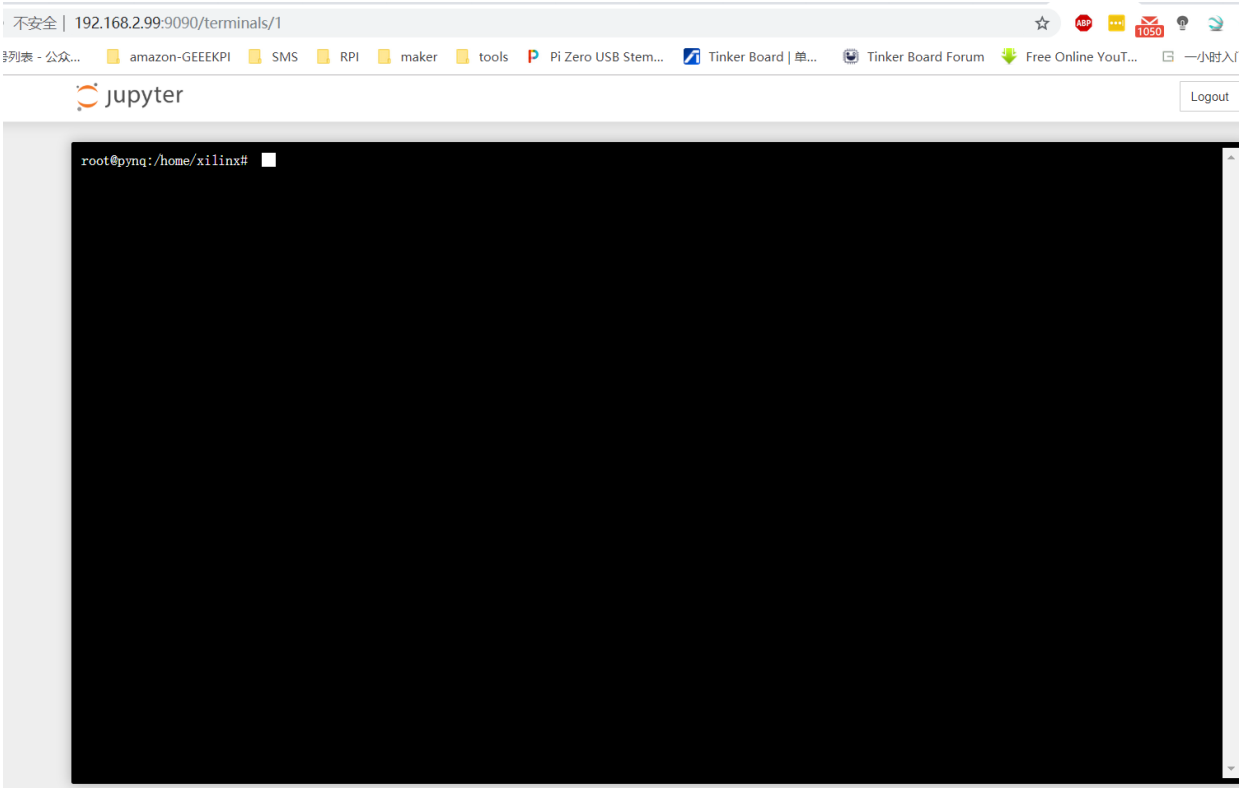
通过谷歌浏览器可以顺利登陆并看到下图：

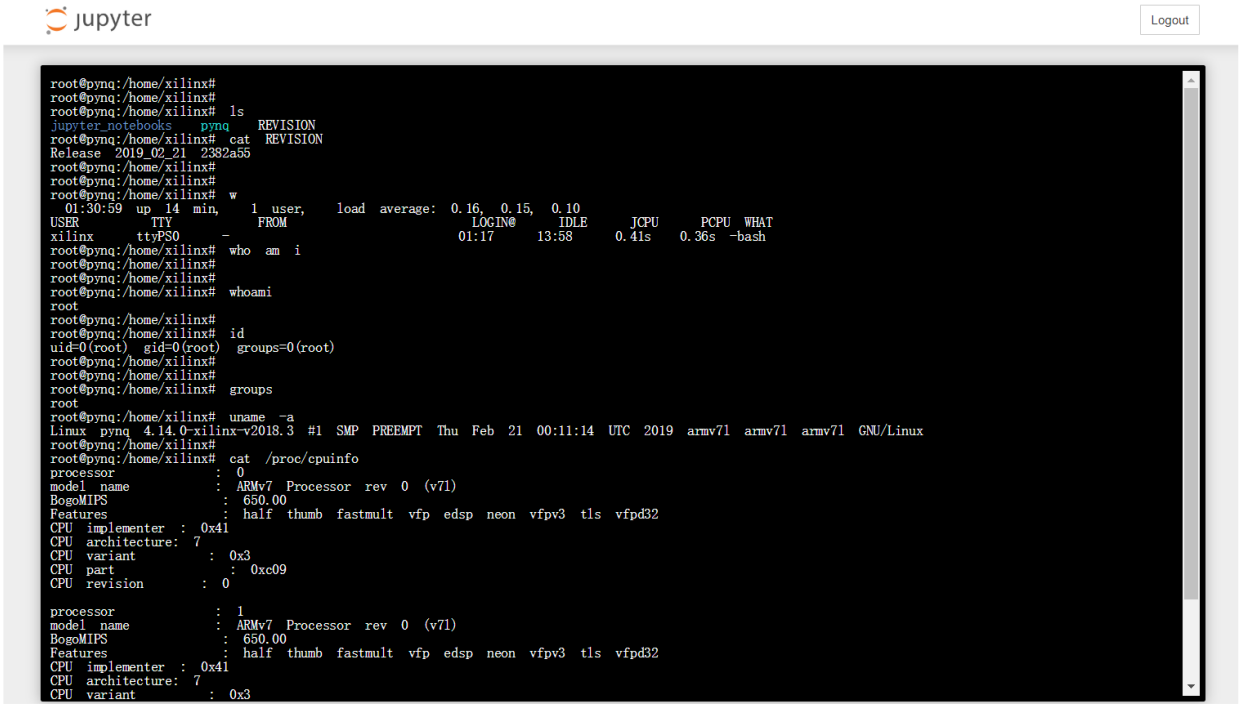



我们尝试一下开个新的终端：

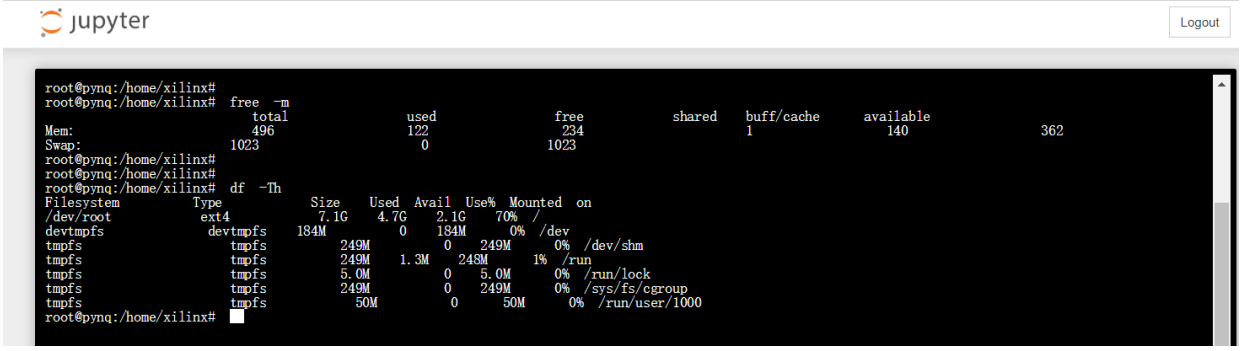

jupyter
Logout
Files
Running
Clusters
Nbextensions
Select items to perform actions on them.
Upload
New
Notebook:
Python 3
Other:
Text File
Folder
Terminal
0
/ getting_started
..
images
1_jupyter_notebooks.ipynb
2_python_environment.ipynb
3_jupyter_notebooks_advanced_features.ipynb
2 个月前
4_base_overlay_iop.ipynb
2 个月前
5_base_overlay_video.ipynb
2 个月前
6_base_overlay_audio.ipynb
2 个月前

然后看到下面的效果：

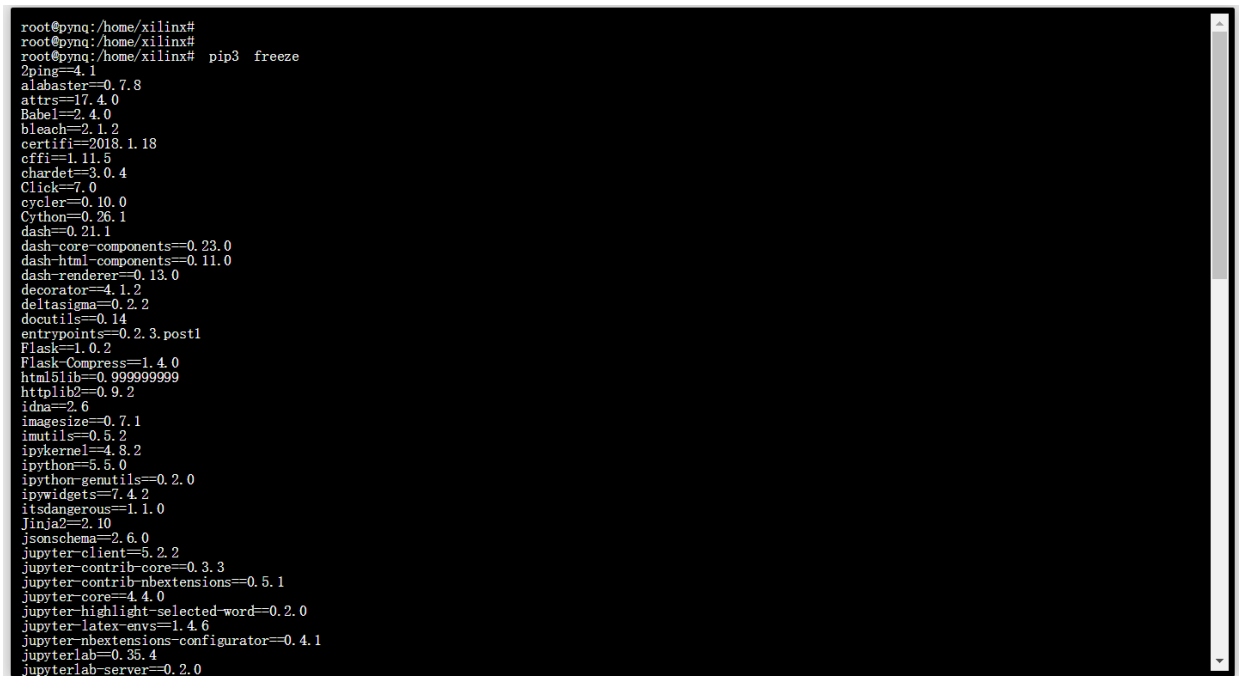

不安全 | 192.168.2.99:9090/terminals/1
amazon-GEEEKPI
SMS
RPI
maker
tools
Pi Zero USB Stem...
Tinker Board | 单...
Tinker Board Forum
Free Online YouT...
jupyter
Logout
root@pynq:/home/xilinx#




简单操作了一下，还是很爽滑的。




检查一下支持的python库有哪些：



```
ipython-genutils==0.2.0
ipywidgets==7.4.2
itsdangerous==1.1.0
Jinja2==2.10
jsonschema==2.6.0
jupyter-client==5.2.2
jupyter-contrib-core==0.3.3
jupyter-contrib-nbextensions==0.5.1
jupyter-core==4.4.0
jupyter-highlight-selected-word==0.2.0
jupyter-latex-envs==1.4.6
jupyter-nbextensions-configurator==0.4.1
jupyterlab==0.35.4
jupyterlab-server==0.2.0
lxml==4.2.1
MarkupSafe==1.0
matplotlib==2.1.1
mistune==0.8.3
mpmath==1.0.0
nbconvert==5.3.1
nbformat==4.4.0
nbsphinx==0.3.1
nbwavedrom==0.2.0
netifaces==0.10.4
networkx==1.11
notebook==5.2.2
numexpr==2.6.4
numpy==1.13.3
pandas==0.24.1
pandocfilters==1.4.2
pexpect==4.2.1
pickleshare==0.7.4
Pillow==5.1.0
plotly==3.6.1
pluggy==0.6.0
ply==3.11
prompt-toolkit==1.0.15
psutil==5.4.2
py==1.5.2
pycparser==2.18
pycurl==7.43.0.1
pyeda==0.28.0
Pygments==2.2.0
pygobject==3.26.1
pygraphviz==1.4rc1
```

```
psutil==5.4.2
py==1.5.2
pycparser==2.18
pycurl==7.43.0.1
pyeda==0.28.0
Pygments==2.2.0
pygobject==3.26.1
pygraphviz==1.4rc1
pynq==2.4
pyparsing==2.2.0
pytest==3.3.2
pytest-sourceorder==0.5
python-apt==1.6.0
python-dateutil==2.6.1
pytz==2018.3
PyWavelets==0.5.1
PyYAML==3.12
pyzmq==16.0.2
requests==2.18.4
requests-unixsocket==0.1.5
retrying==1.3.3
rise==5.2.0
roman==2.0.0
scikit-image==0.13.1
scipy==0.19.1
simplegeneric==0.8.1
six==1.11.0
Sphinx==1.6.7
sphinx-rtd-theme==0.4.3
SQLAlchemy==1.1.11
ssh-import-id==5.7
sympy==1.1.1
terminado==0.7
testpath==0.3.1
tornado==4.5.3
traitlets==4.3.2
transitions==0.5.3
unattended-upgrades==0.1
urllib3==1.22
uvloop==0.8.1
wcwidth==0.1.7
webencodings==0.5
Werkzeug==0.14.1
widgetsnbextension==3.4.2
root@pynq:/home/xilinx#
```

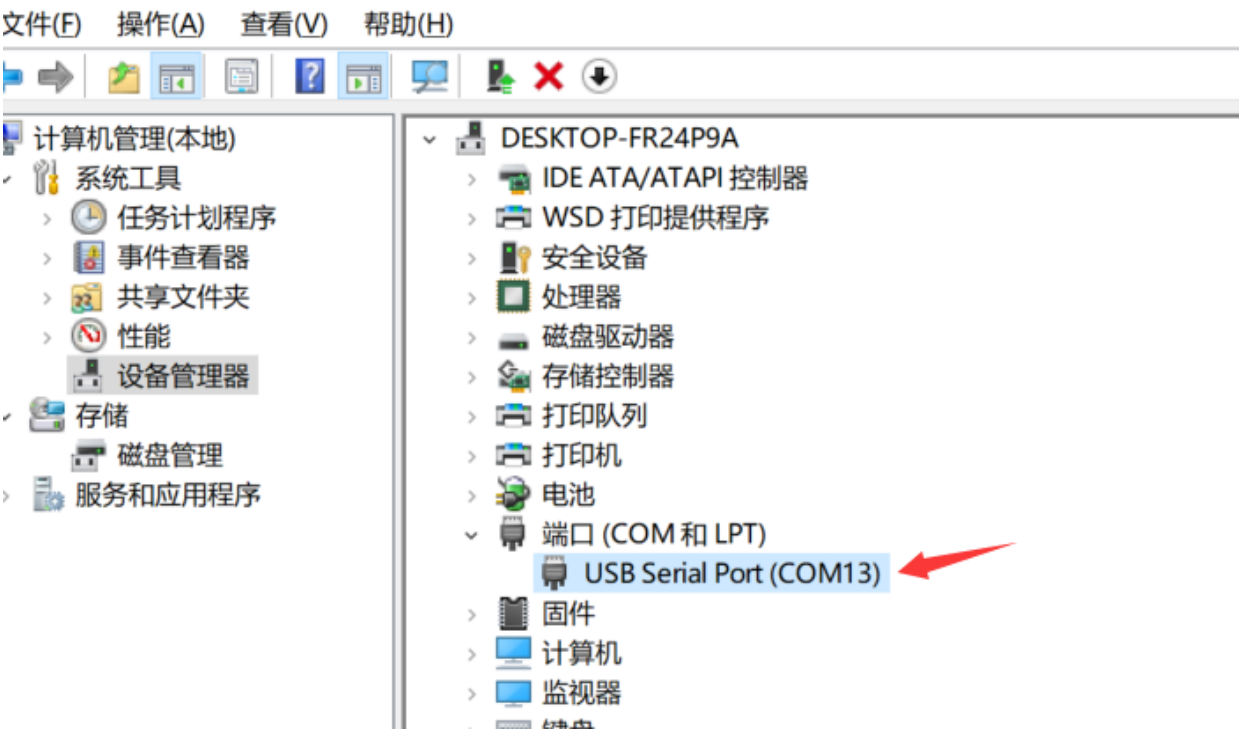
简单尝试写了个伪随机代码：

```
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#  cat  test.py
#!/usr/bin/env  python3
from  random  import  randint
import  time  as  t

for  i  in  range(0,5):
        s  =  i*randint(0,99999999)/  t.time()
        print(s)
        t.sleep(1)

root@pynq:/home/xilinx#  python3  test.py
0.0
0.03760030628714377
0.005912715598996392
0.15408853706991532
0.09322528039786672
root@pynq:/home/xilinx#
```

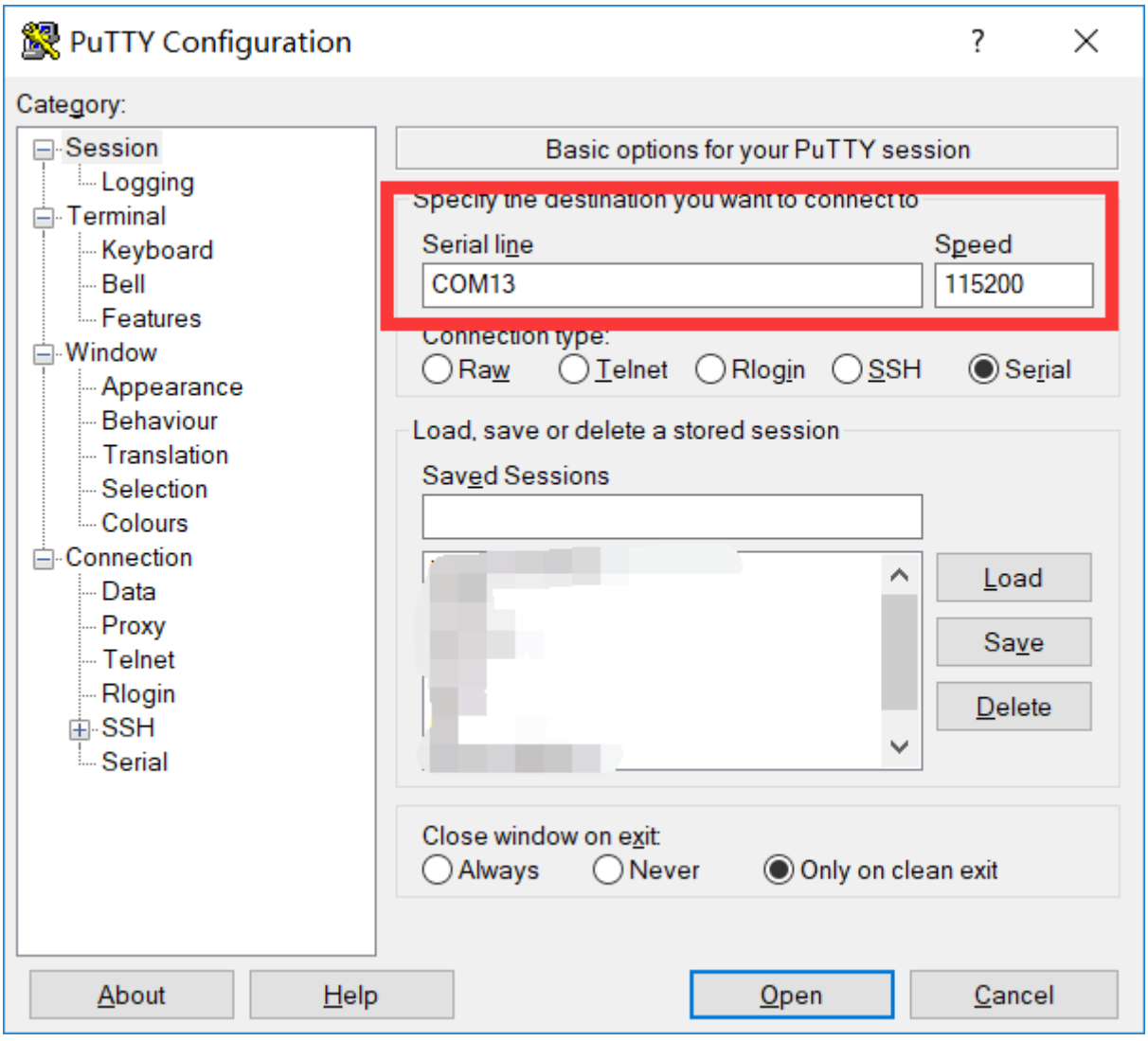
测试一下看看IP地址，路由信息啥的。

```
root@pynq:/home/xilinx#  route  -n
Kernel  IP  routing  table
Destination         Gateway                 Genmask                 Flags  Metric  Ref       Use  Iface
192.168.2.0         0.0.0.0                 255.255.255.0     U          0            0               0  eth0
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#  ifconfig
eth0:  flags=4163<UP,BROADCAST,RUNNING,MULTICAST>    mtu  1500
               inet6  fe80::205:6bff:fe00:c870    prefixlen  64    scopeid  0x20<link>
               ether  00:05:6b:00:c8:70    txqueuelen  1000    (Ethernet)
               RX  packets  3082    bytes  215665  (215.6  KB)
               RX  errors  0    dropped  0    overruns  0    frame  0
               TX  packets  2569    bytes  1083186  (1.0  MB)
               TX  errors  0    dropped  0  overruns  0    carrier  0    collisions  0
               device  interrupt  29    base  0xb000

eth0:1:  flags=4163<UP,BROADCAST,RUNNING,MULTICAST>    mtu  1500
               inet  192.168.2.99    netmask  255.255.255.0    broadcast  192.168.2.255
               ether  00:05:6b:00:c8:70    txqueuelen  1000    (Ethernet)
               device  interrupt  29    base  0xb000

lo:  flags=73<UP,LOOPBACK,RUNNING>    mtu  65536
               inet  127.0.0.1    netmask  255.0.0.0
               inet6  ::1    prefixlen  128    scopeid  0x10<host>
               loop    txqueuelen  1000    (Local  Loopback)
               RX  packets  2704    bytes  173192  (173.1  KB)
               RX  errors  0    dropped  0    overruns  0    frame  0
               TX  packets  2704    bytes  173192  (173.1  KB)
               TX  errors  0    dropped  0  overruns  0    carrier  0    collisions  0

root@pynq:/home/xilinx#  ip  addr
1:  lo:  <LOOPBACK,UP,LOWER_UP>  mtu  65536  qdisc  noqueue  state  UNKNOWN  group  default  qlen  1000
       link/loopback  00:00:00:00:00:00  brd  00:00:00:00:00:00
       inet  127.0.0.1/8  scope  host  lo
              valid_lft  forever  preferred_lft  forever
       inet6  ::1/128  scope  host
              valid_lft  forever  preferred_lft  forever
2:  eth0:  <BROADCAST,MULTICAST,UP,LOWER_UP>  mtu  1500  qdisc  pfifo_fast  state  UP  group  default  qlen  1000
       link/ether  00:05:6b:00:c8:70  brd  ff:ff:ff:ff:ff:ff
       inet  192.168.2.99/24  brd  192.168.2.255  scope  global  eth0:1
              valid_lft  forever  preferred_lft  forever
       inet6  fe80::205:6bff:fe00:c870/64  scope  link
              valid_lft  forever  preferred_lft  forever
3:  sit0@NONE:  <NOARP>  mtu  1480  qdisc  noop  state  DOWN  group  default  qlen  1000
       link/sit  0.0.0.0  brd  0.0.0.0
root@pynq:/home/xilinx#
```

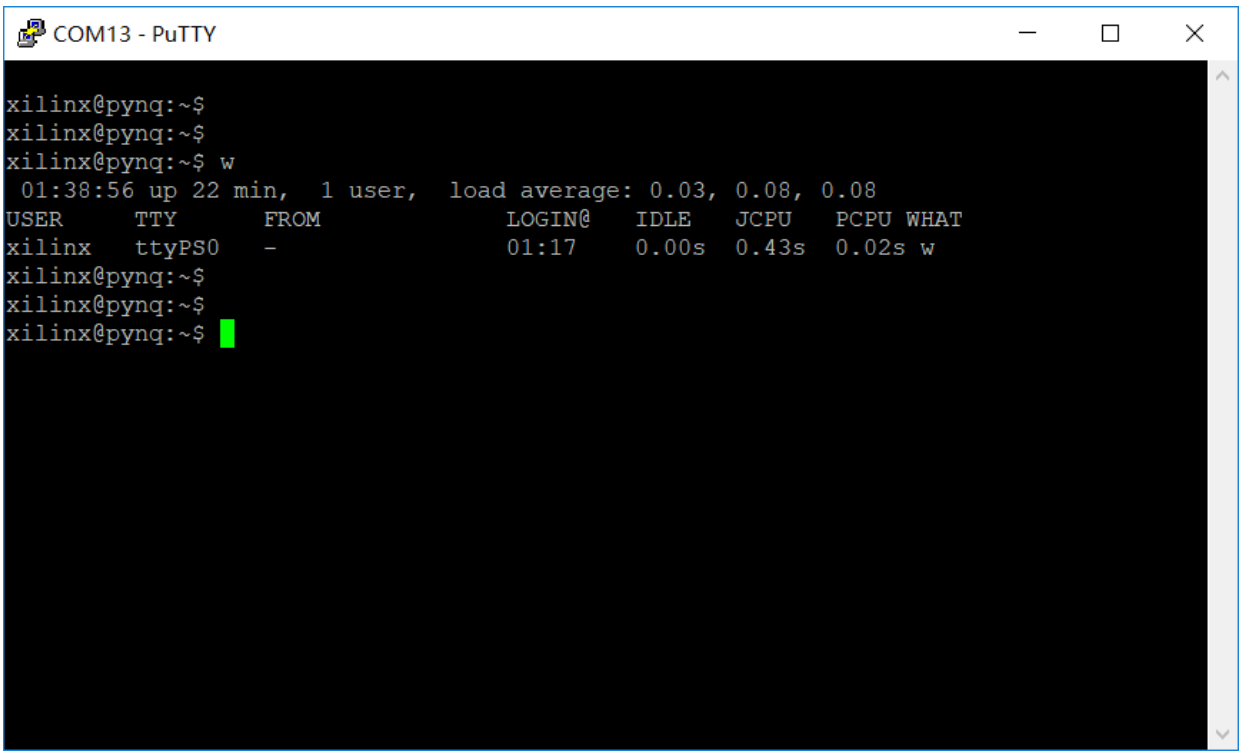
都正常。



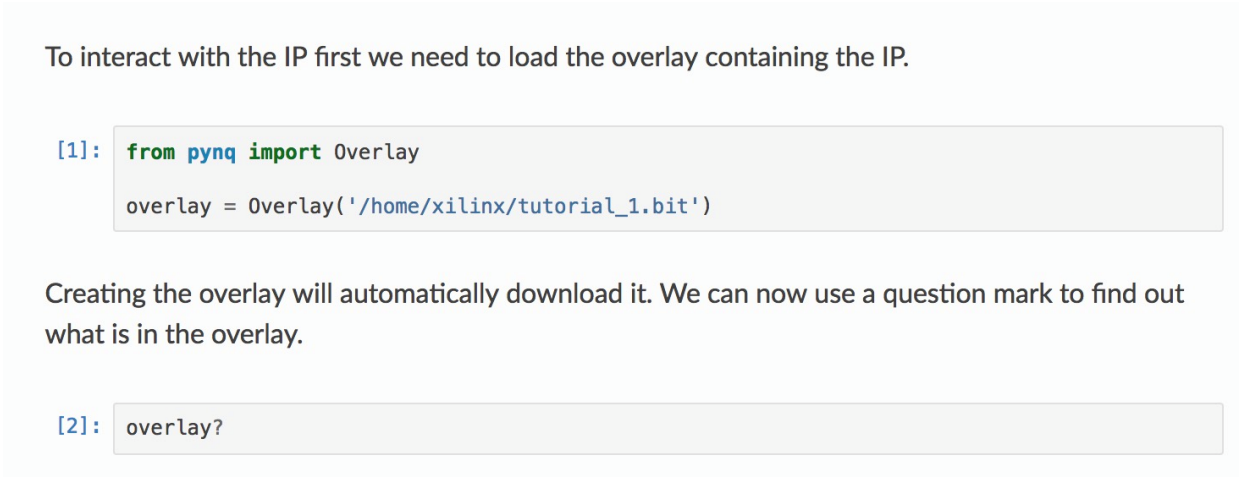

因为目前PYNQ的板子是通过USB串口连接到电脑的，也就意味着我应该可以通过串口也能连上来。简单配置putty，使用串口，刚才看到的是com13，波特率我猜115200，尝试一下先。



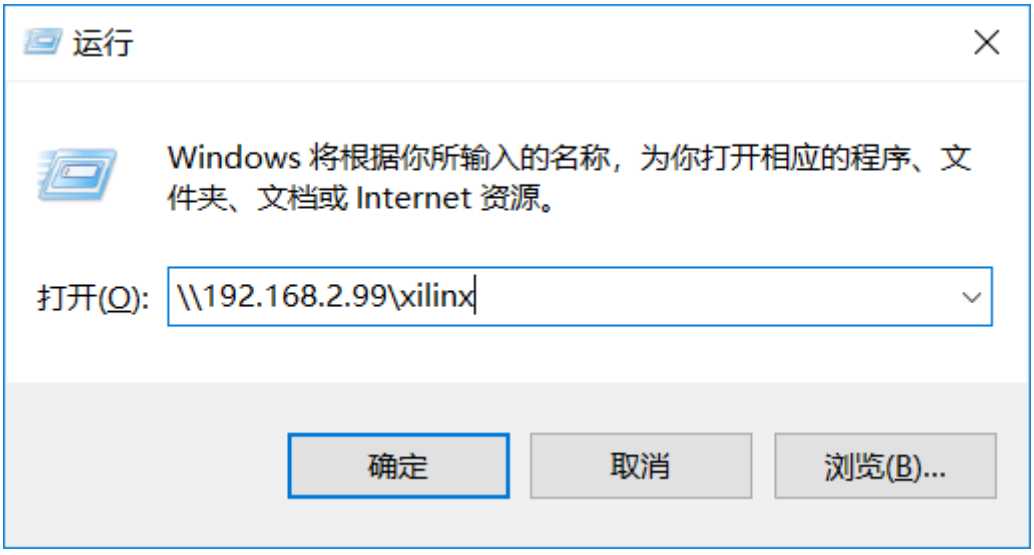

发现是可以的，而且很顺畅，没有卡顿，没有乱码，哈利路亚。

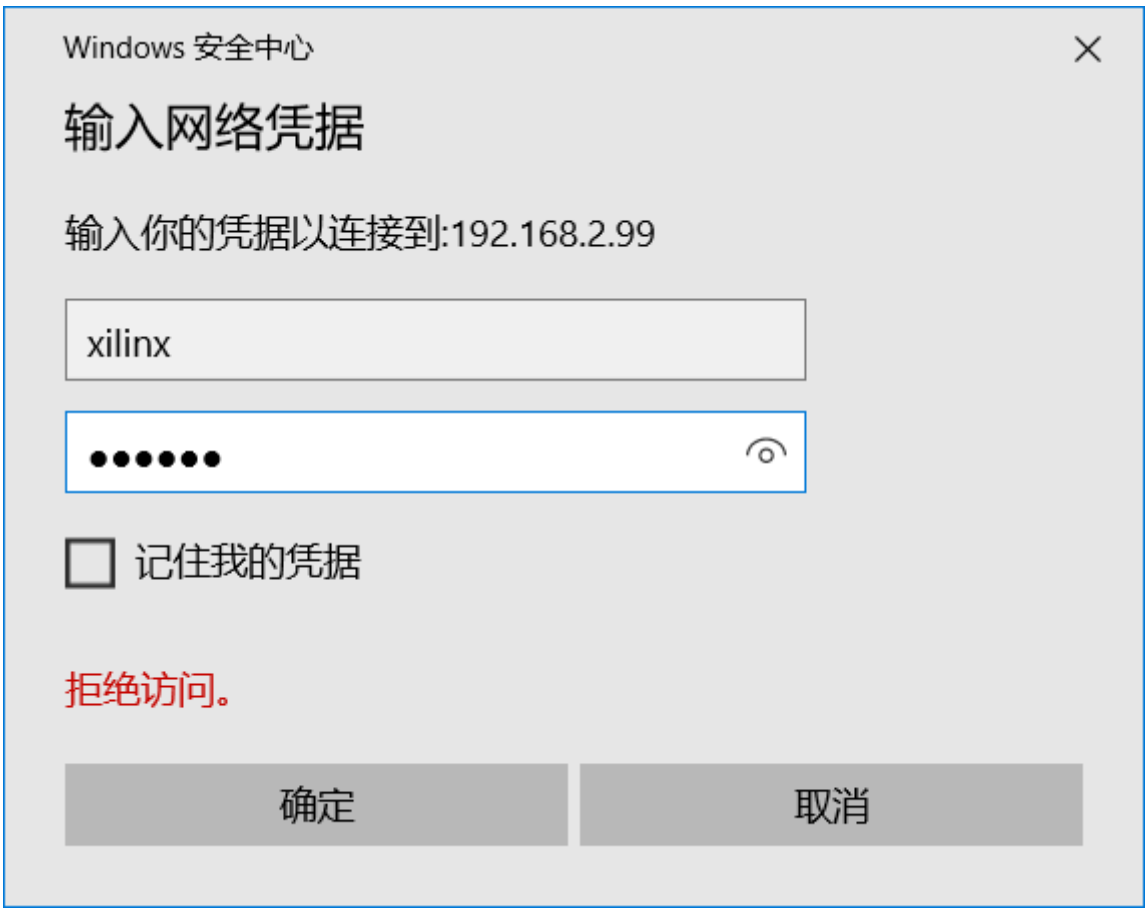



August说看了下gameboy代码，应该跟audio无关，感觉可以在PYNQ-Z2上跑。
https://github.com/kastnerkyle/gb/tree/master/verilog 这个链接

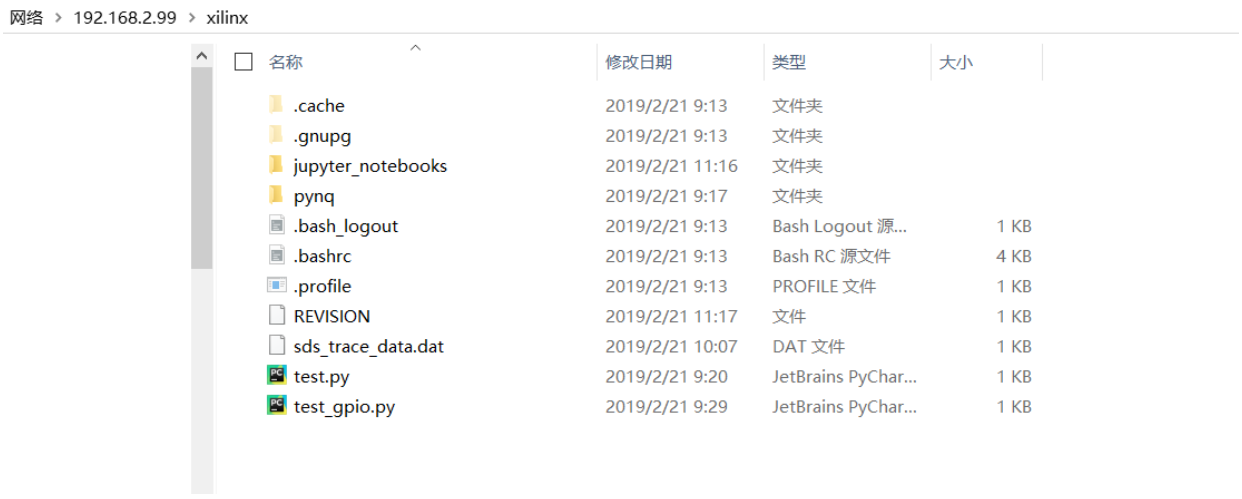



我突然想起刚才尝试的时候发现有samba服务，尝试了一下果然发现有共享：

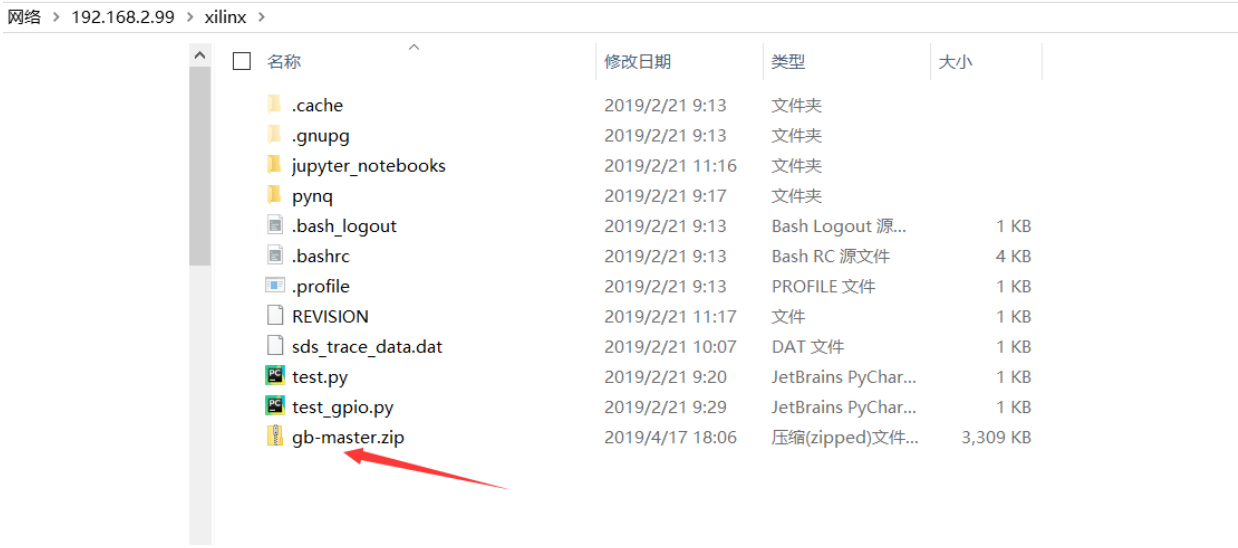





用户名密码都应该是xilinx,今天好幸运，果然都猜对了。




从github上下载下来直接传上去就好了，非常方便。

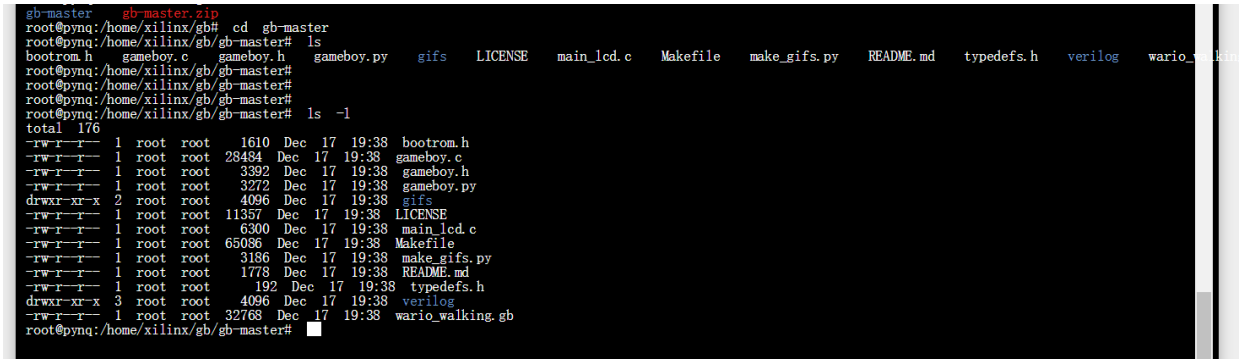



```
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#  ls
gb-master.zip    jupyter_notebooks    pynq    REVISION    sds_trace_data.dat    test_gpio.py    test.py
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
```

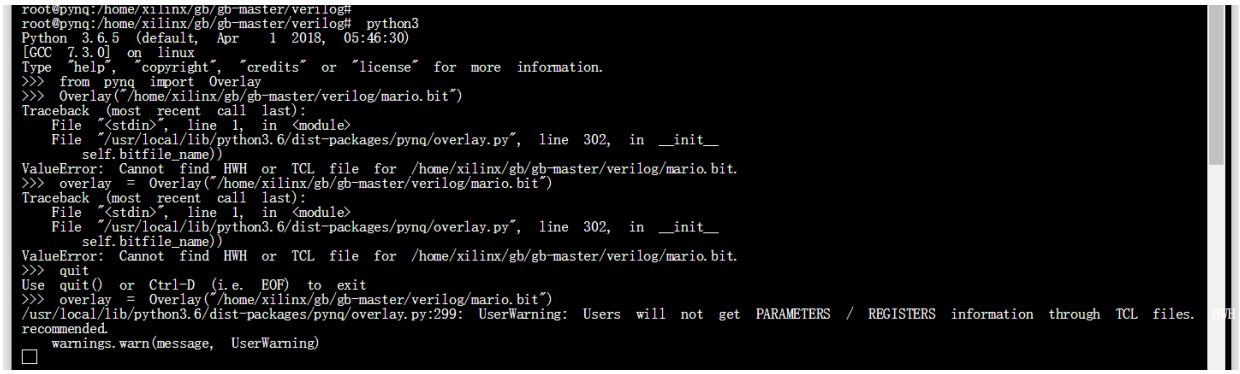
下面开始解压，然后测试一下

```
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#
root@pynq:/home/xilinx#  mkdir  gb
root@pynq:/home/xilinx#  cd  gb
root@pynq:/home/xilinx/gb#  mv  ../gb-master.zip  .
root@pynq:/home/xilinx/gb#
root@pynq:/home/xilinx/gb#  ls
gb-master.zip
root@pynq:/home/xilinx/gb#  unzip  gb-master.zip
Archive:    gb-master.zip
93c4502b4003771f8743a62f3f66b899013c83da
      creating:  gb-master/
    inflating:  gb-master/LICENSE
    inflating:  gb-master/Makefile
    inflating:  gb-master/README.md
    inflating:  gb-master/bootrom.h
    inflating:  gb-master/gameboy.c
    inflating:  gb-master/gameboy.h
    inflating:  gb-master/gameboy.py
      creating:  gb-master/gifs/
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_124856.gif
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_184680.gif
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_244488.gif
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_304912.gif
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_364336.gif
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_423184.gif
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_484080.gif
    inflating:  gb-master/gifs/Micro_Machines_(USA,_Europe).gb_63216.gif
    inflating:  gb-master/gifs/wario_walking.gif
    inflating:  gb-master/main_lcd.c
    inflating:  gb-master/make_gifs.py
    inflating:  gb-master/typedefs.h
      creating:  gb-master/verilog/
      creating:  gb-master/verilog/.Xil/
    inflating:  gb-master/verilog/.Xil/top_propImpl.xdc
    inflating:  gb-master/verilog/.top.pynq.v.swp
```

用unzip解压。




然后调用python3 的IDLE执行了一下，发现报错。




然后根据august提供的建议，将build.tcl 拷贝了一份改名为mario.tcl就好了。。

但是仍然在加载mario.bit时候死机，灯都灭了，可能是bitstream不正确吧，回家慢慢研究。

今天先到这里，给大家发学习心得哈~