Panoseti_tftp PythonPackage User Manual V1.0.1

Wei

12-12-2019

This document is about how to use the python script--panoseti_tftp. If you just want to use it, read part2--"How to use panoseti_tftp"; if you are a designer, or you want to know more about it, read part3--"Memory Map".

1. Python package requirement

This python package is based on another open source python package--tftpy, so we need to install tftpy first.

```
wei@Wei-Berkeley:~$ pip3 install tftpy
Collecting tftpy
Installing collected packages: tftpy
Successfully installed tftpy-0.8.0
```

I tested it on python3.x, so it's better to use python3.x.

```
wei@Wei-Berkeley:~/tftp_data$ python3 -V
Python 3.6.9
```

2. How to use panoseti_tftp

You should import panoseti_tftp to your python script first, and then make a connection to tftp server. The ip address here is the ip of tftp server.

```
>>> from panoseti_tftp import tftpw
>>> client=tftpw('192.168.1.11')
```

Then there are 8 methods can be used.

2.1 help()

By using help(), you can get some help information about the methods.

```
>>> client.help()
Help Information:
  get_flashuid()          : get flash uid from flash chip, and the flash uid are 8 bytes.
  get_wrpc_filesys()      : get wrpc file system from flash chip. [ default space : 0x00E00000--0x00F0FFFF]
  get_mb_file()           : get mb file from flash chip. [default space : 0x00F10000--0x0100FFFF]
  put_wrpc_filesys(file)  : put file to wrpc filesys space. [default space : 0x00E00000--0x00F0FFFF]
  put_mb_file(file)       : put file to mb file space. [default space : 0x00F10000--0x0100FFFF]
  put_bin_file(file)      : put fpga bin file to flash chip. [default from 0x01010000]
  reboot()                : reboot fpga. [default from 0x01010000]
```

2.2 get_flashuid(filename='flashuid')

You can get an 8-BYTE flash device UID here, which is unique.

The default finename is flashuid, and you can change it to what you want.

```
>>> client.get_flashuid()
Get flash Device UID successfully!
```

2.3 get_wrpc_filesys(filename='wprc_filesys')

This is used to get "wrpc file system" from flash chip, which contains sfp calibration data for WR, mac address of WR and so on.

Maybe we need to have a backup of the "wrpc file system", so you can download it by using this method.

The default filename is "wrpc_filesys", and you can change it to what you want.

```
>>> client.get_wrpc_filesys()
Download wrpc file system successfully!
```

2.4 get_mb_file(filename='mb_file')

This is used to get microbalze file, which contains the information about focus motor position and so on.

The default filename is 'mb_file', and you can change it to what you want.

```
>>> client.get_mb_file()
Download mb file successfully!
```

2.5 put_wrpc_filesys(filename)

You can put a "wrpc file system" to the flash chip by using this method, and make sure **the "file system" is got from quabo by using get_wrpc_filesys()**.

```
>>> client.put_wrpc_filesys()
Upload wrpc_filesys to panoseti wrpc_filesys space successfully!
```

If the file you want to put to wrpc_filesys space is not got by using get_wrpc_filesys(), you will not do it successfully.

```
>>> client.put_wrpc_filesys('mb_file')
The size of wrpc_filesys is incorrect, please check it!
```

2.6 put_mb_file(filename)

You can put a microbalze file to the flash chip by using this method, but make sure the size of the file is **not more than 1MB**, or it will not write to the flash successfully.

```
>>> client.put_mb_file()
Upload mb_file to panoseti mb_file space successfully!
```

If you try to write a file larger than 1MB:

```
>>> client.put_mb_file('quabo_000000.bin')
The size of mb file is too large, and it will mess up other parts on the flash chip!
```

2.7 put_bin_file(filename)

You can put a bin file to the flash chip by using this method, which is the “silver” firmware on the flash chip.

```
>>> client.put_bin_file('quabo_016000.bin')
Upload quabo_016000.bin to panoseti bin file space successfully!
```

2.8 reboot()

This method is to make the fpga reboot from the “silver” firmware.

Because the fpga is reboot, tftp will get no response from the fpga, it will try to send cmds several times, but that’s ok, just ignore it. If it works well, you can get housekeeping data successfully 30s later.

```
>>> client.reboot()
*******************************************************
FPGA is rebooting, just ignore the timeout information
Wait for 30s, and then check housekeeping data!
*******************************************************
Timeout waiting for traffic, retrying...
Timed-out waiting for traffic
resending last packet
Resending packet DAT packet: block 1
    data: 4 bytes on sessions <tftpy.TftpStates.TftpStateExpectACK object at 0x7f8d50a6efd0>
Timeout waiting for traffic, retrying...
Timed-out waiting for traffic
resending last packet
Resending packet DAT packet: block 1
    data: 4 bytes on sessions <tftpy.TftpStates.TftpStateExpectACK object at 0x7f8d50a6efd0>
Timeout waiting for traffic, retrying...
Timed-out waiting for traffic
resending last packet
Resending packet DAT packet: block 1
    data: 4 bytes on sessions <tftpy.TftpStates.TftpStateExpectACK object at 0x7f8d50a6efd0>
Timeout waiting for traffic, retrying...
Timed-out waiting for traffic
resending last packet
Resending packet DAT packet: block 1
    data: 4 bytes on sessions <tftpy.TftpStates.TftpStateExpectACK object at 0x7f8d50a6efd0>
Timeout waiting for traffic, retrying...
Timed-out waiting for traffic
```

If you can’t get the housekeeping data, just re-power the system, and call put the bin file again, and then call reboot() again.

3. Memory Map

In our design, the memory map is as follow:

Address	Size(MB)	Content
0x00000000----0x006FFFFF	7	Golden Firmware
0x00700000----0x00DFFFFF	7	Empty
0x00E00000----0x00F0FFFF	1.1	WRPC File System
0x00F10000----0x0100FFFF	1	MicroBlaze File
After 0x01010000	-	Silver Firmware

So each method has it’s default address, which means the related files starts from this “addr”.

(You should shift 8 bits to right on the bin file addr, and then set it as reboot addr)

```
#get wrpc_filesys
#space → : 0x00E00000--0x00F0FFFF
#size → : 1MB + 64K BYTES = 1114112 BYTES
def get_wrpc_filesys(self, filename='wrpc_filesys',addr=0x00e00000):
  → 
#get mb_file 
#space → : 0x00F10000--0x0100FFFF
#size → : 1MB = 1048576 BYTES
def get_mb_file(self, filename='mb_file',addr=0x00F10000):
  → 
#put wprc_filesys, starting from 0x00E00000
def put_wrpc_filesys(self,filename='wrpc_filesys', addr=0x00E00000):
  → 
#put mb file, starting from 0x00F10000
def put_mb_file(self,filename='mb_file', addr=0x00F10000):
  → 
#put bin file,starting from 0x01010000
def put_bin_file(self,filename,addr=0x01010000):
  → 
def reboot(self,addr=0x00010100):
```

If the memory map changes in the future, don’t forget to change the default addr value in the panoseti_tftp package.