

Infocube

에뮬레이터 설명서

DU-EMULATOR, 계측기

목차

Revision History	3
에뮬레이터 개요	4
1. 시작하기.....	5
1.1 외형소개.....	5
1.2 시스템 사용준비 (OS : Windows).....	6
2. PCAP 파일 재생.....	7
2.1. 사용법 요약;.....	7
2.2 사용법 상세.....	8
2.3 ./emul_pcap_play 사용법	12
2.4 명령어 요약.....	13
3. Capture Data & PCAP으로 저장	14
3.1 사용법-Capture	14
3.2 저장된 패킷을 'PCAP'으로 저장.....	15
4. Terminal 명령어	16
4.1 MAC 변경	16
● 4.1.1 MAC 변경 활성화	16
● 4.1.1 MAC 변경 비활성.....	16
4.2 Ping	17
4.3 Fan Control	17
4.4 진단 기능 명령어.....	18
● 4.4.1 diag_rst	18
● 4.4.2 diag	19
● 4.4.3 diag_hubsetvl.....	19
● 4.4.4 diag_pkinfo	20
4.5 Link Up Check	21
4.6 클럭동기 & 타이밍 조절	22
● 외부 PPS에 동기 방법1.....	22
● 외부 PPS에 동기 방법2.....	22

● TX Packet 지연방법	22
● 캡처 타이밍 조절방법	23
5. LUA Script 명령	24
● 5.2. POE 파워상태모니터	24
6. 사양 & 특성	25
5.1 Player(TX)	25
5.2 TX Packet Timing	25
5.3 Capture(RX)	26
5.4 RX Packet Timing	26
7. Register Map	27
7.1MAC Control Register	27
7.2 Pcap_player Register	27
7.3 Pcap_Capture Register	29
8..... TBD	30

Revision History

Date	Version	Description of Revisions
2020/07/03	1.0	
2020/09/09	1.1	5.3 VRAN_TAG 변경방법(200909업데이트) 외부동기방법(200909 업데이트) TX Packet 지연방법 (200909 업데이트)
		이후 버전부터는 실행 명령이 변경되었습니다. 새 10월 이후 납품된 보드에 해당되는 매뉴얼입니다.
2020/10/07	2.0	

에뮬레이터 개요

이 에뮬레이터는 다음과 같은 기능을 갖추고 있습니다.

- 10G Ethernet 3+1 Port (1: 광포트)
- PCAP 파일을 4개 포트로 출력
 - Source MAC 설정가능
 - 각 포트 별로 Destination MAC 설정가능
- Packet Capture
 - 4개 각 포트로 수신되는 Uplan Packet을 저장
 - 각 포트 당 1GB 저장 공간
 - 저장된 Uplan Packet을 PCAP 파일로 저장
- Ping
 - eth1
- 외부 입력 10MHz, 1PPS
- 외부 출력 10MHz, 1PPS. 외부로부터 받은 신호를 그대로 내보낸다.
- 외부 출력 EVT0, EVT0 (TBD)
- 디버깅 포트 UART
- 디버깅 포트 1000M Ethernet

1. 시작하기

1.1 외형소개

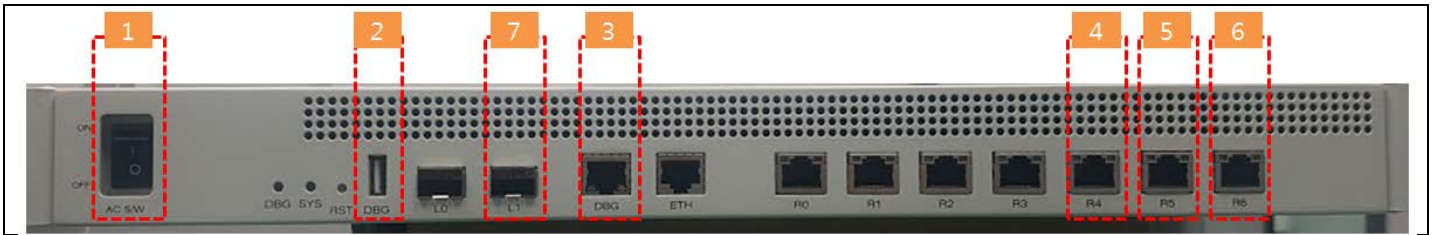


그림1. 전면

1. 전원스위치
2. 디버깅포트 UART
3. 디버깅포트 Ethernet 1G
4. Copper 10G 포트, Port-0
5. Copper 10G 포트, Port-1
6. Copper 10G 포트, Port-2
7. Optical 10G 포트, Port-3

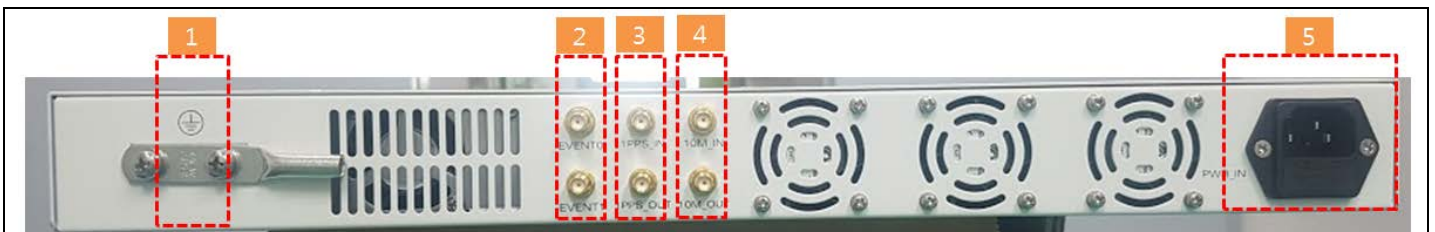
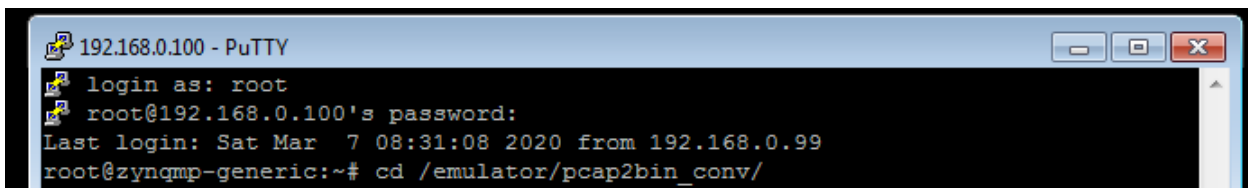
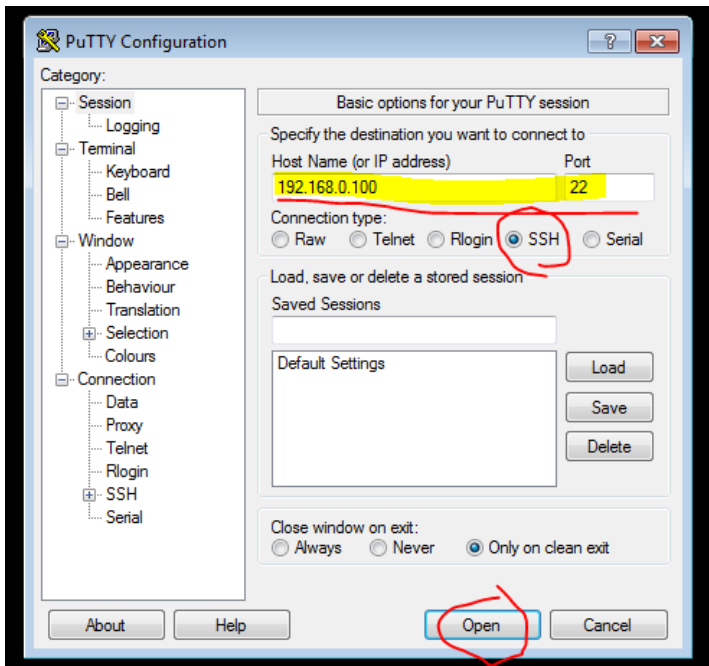


그림2. 후면

1. 새시 그라운드
2. Event Output 2개 포트 (TBD)
3. 1-PPS Input/output Port
4. 10MHz Input/output Port,
 - A. 시스템 클럭 으로 사용되는 10M-input. 외부와 동기된다.
 - B. 입력 받은 10M 클럭이 그대로 나간다.
5. 220VAC 전원

1.2 시스템 사용준비 (OS : Windows)

전원을 올리고 부팅이 완료되면, 'PuTTY'와 같은 프로그램으로 아래 그림과 같이 에뮬레이터에 접속한다.
보드 IP는 일반적으로 192.168.0.100~102으로 설정되어 있다.
로그인에 필요한 정보는, id: root passwd : root.



터미널에 'cd /emulator'를 쳐서 이동한다. 에뮬레이터에 '/emulator' 폴더 아래서 모든 작업이 이루어진다.

2. PCAP 파일 재생

2.1. 사용법 요약;

1. 'Putty', 'Filezilla', 'Wireshark'를 PC(windows)에 설치한다.
2. Wireshark를 사용하여, 'pcap' 파일을 'pcapng' 파일로 변환한다.
3. Filezilla를 사용하여, 'pcapng' 파일을 계측기로 복사한다.
4. Putty를 사용하여 계측기에 접속한다.
 - A. 'pcapng' 파일을 bin파일로 변경한다.

B. bin파일을 실행시킨다.

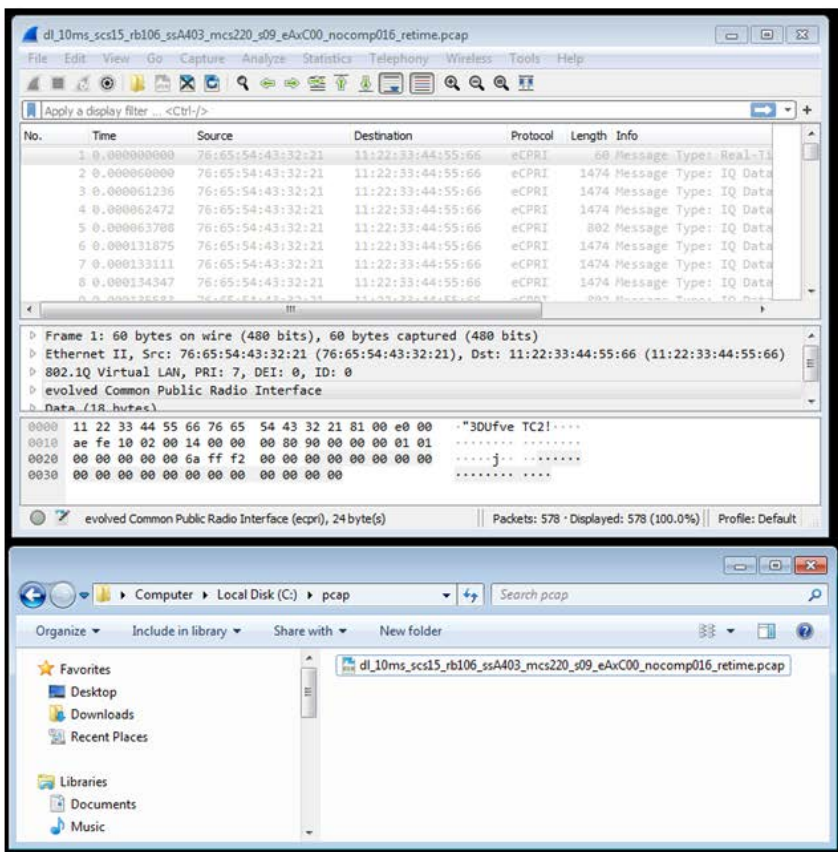
→ 4개 포트에서 데이터가 출력된다.

putty로 보드에 접속하여 pcap 파일을 재생하는데 필요한 실행하는 명령어 요약

```
cd /emulator/util
./conv-pcapng-axi128nsec.py ./full_band_test_20ms.pcapng ./full_band_test_20ms.bin 0 0
cp -rf ./full_band_test_20ms.bin /emulator
cd /emulator
./emul_pcap_play 1 full_band_test_20ms.bin 20 0 0 0
```

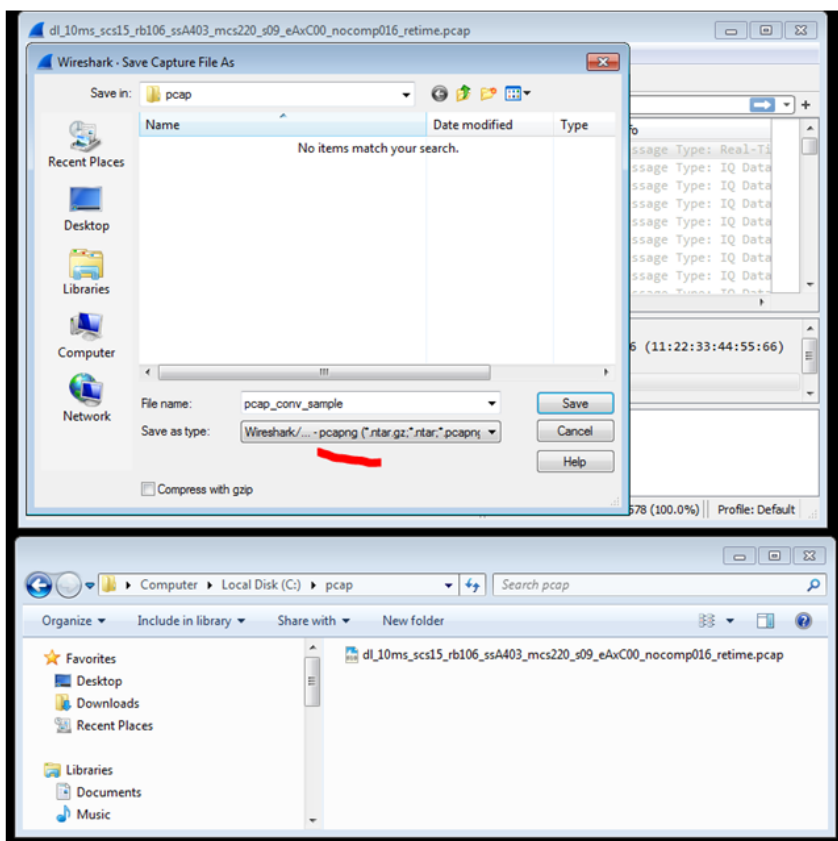

2.2 사용법 상세

PCAP 파일을 PCAPNG로 변경



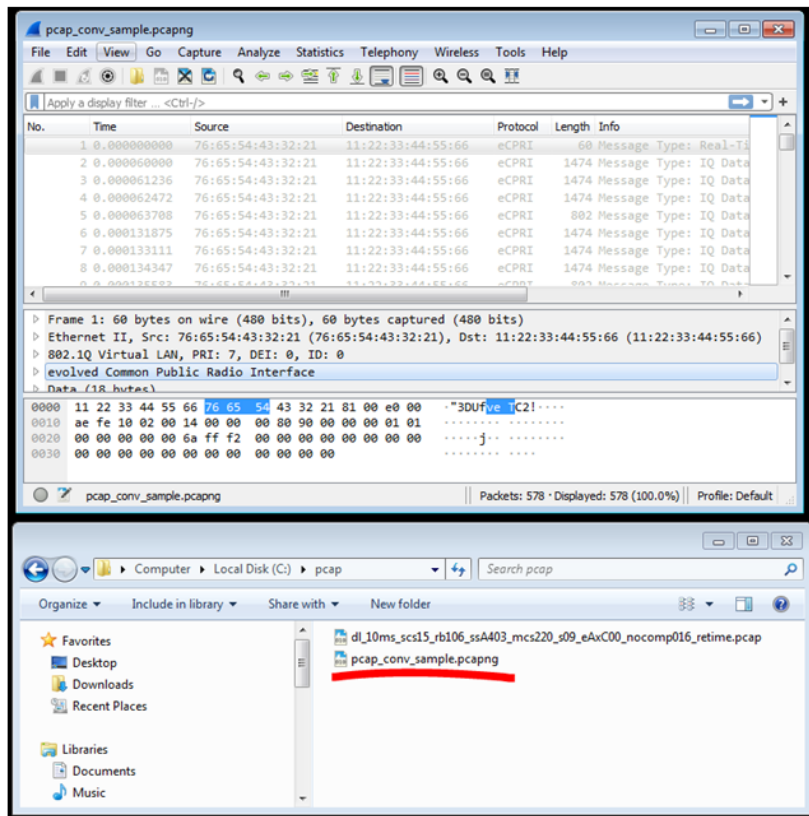
Wireshark로 'PCAP'파일을 연다.

PCAP 파일을 PCAPNG로 변경



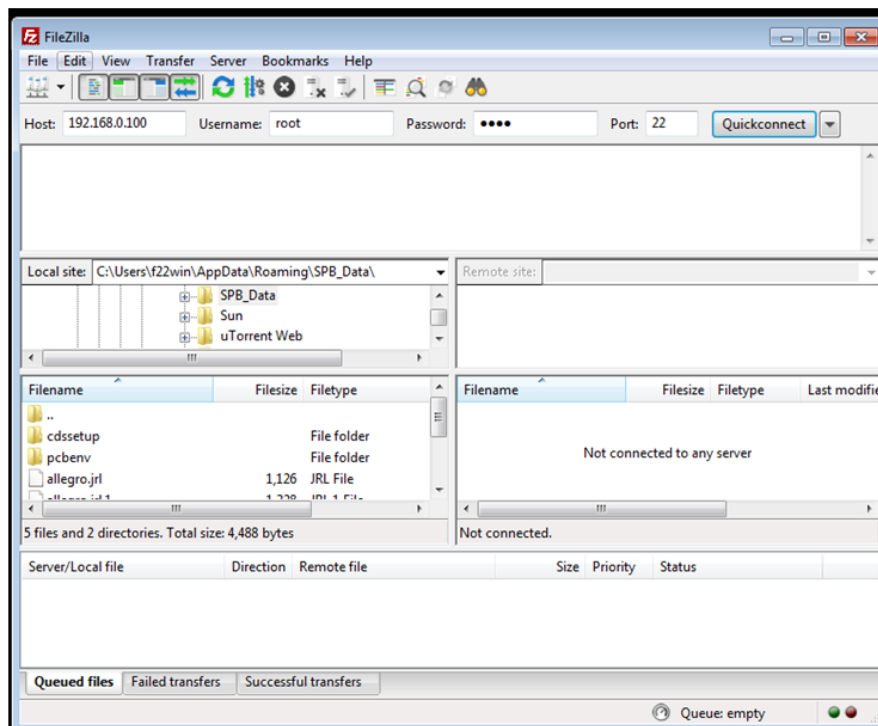
Wireshark로 'PCAP'파일을
'pcapng'파일로 다시 저장한다.

PCAP 파일을 PCAPNG로 변경



그림과 같이 'pcapng' 파일이 생긴다.

pcap_conv_sample.pcapng 파일을 보드에 복사 보드에 원격 로그인(ssh)

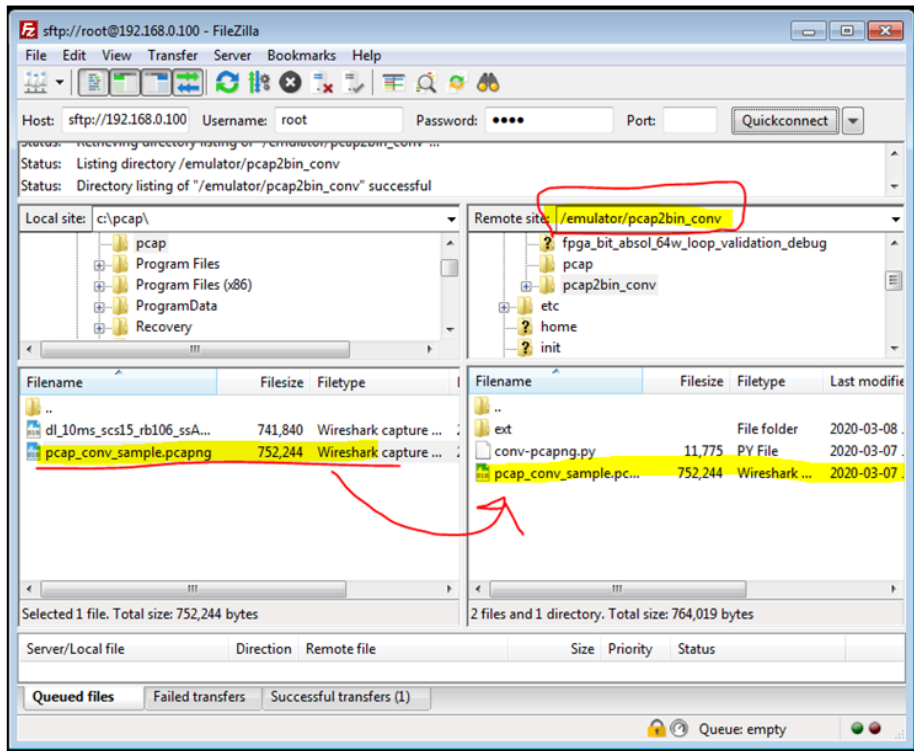


FileZilla를 사용하여,
그림과 같이 아래 정보를 타이핑하고,
'Quickconnect' 버튼을 눌러 접속한다.

Host: 192.168.0.100
Id: root
Pass: root
Port: 22

pcap_conv_sample.pcapng 파일을 보드에 복사

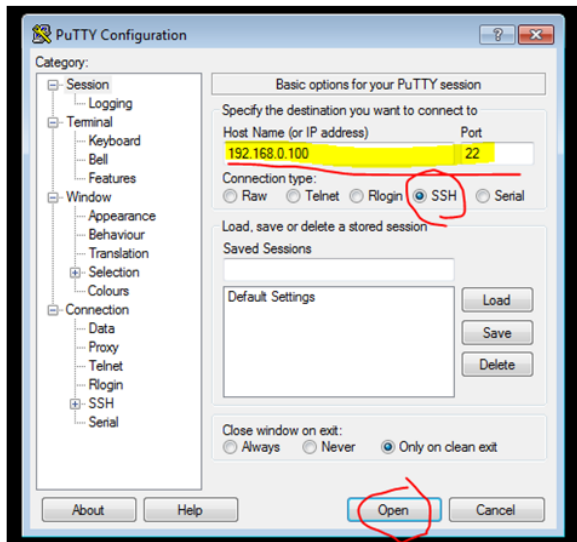
.pcapng 파일을 보드로 복사 (PC폴더에서 -> 보드폴더 / emulator/util)



그림과 같이 FileZilla를 사용하여 ,
'pcap'에서 변환한 'pcapng'파일을
보드-폴더에 복사한다.

'pcapng'파일이 복사되는 폴더는,
/emulator/util

Putty로 보드에 접속



그림과 같이 Putty접속하면, 아래그림과 같이 터미널이 올라온다.

Login : root

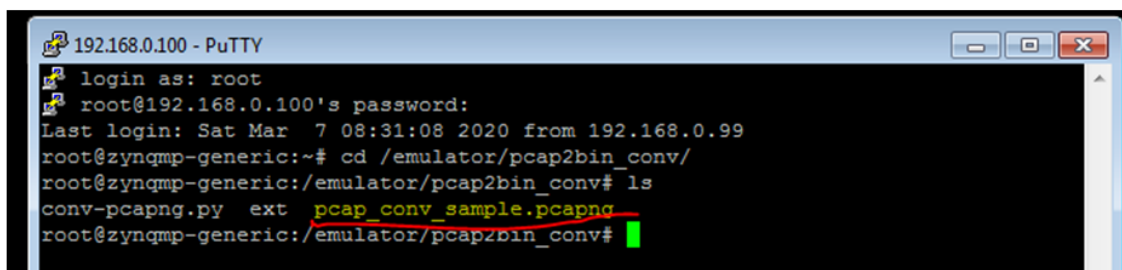
Passwd : root

→ 로그인 하고, pcapng 파일이 있는 곳으로 이동한다.

로그인 완료하고,

cd /emulator/ util

그림과 같이 보드에 복사가 잘됐다면,
'pcap_conv_sample.pcapng' 파일이 있다.



Pcapng 파일을 .bin 파일로 변환

Pcap_player는 pcapng파일을 bin으로 변환해서 사용한다.

```
cd /emulator/util
```

```
./ conv-pcapng-axi128nsec.py ./pcap_conv_sample.pcapng ./tx.bin 0 0
```

```
cp -rf ./tx.bin /emulator
```

```
192.168.0.100 - PuTTY
TIMIG/SIZE 563 9774583 9773403 -1180 detail 1474 92 2
TIMIG/SIZE 564 9775819 9774639 -1180 detail 1474 92 2
TIMIG/SIZE 565 9777055 9775875 -1180 detail 1474 92 2
TIMIG/SIZE 566 9778291 9777649 -642 detail 802 50 2
TIMIG/SIZE 567 9845938 9844758 -1180 detail 1474 92 2
TIMIG/SIZE 568 9847174 9845994 -1180 detail 1474 92 2
TIMIG/SIZE 569 9848410 9847230 -1180 detail 1474 92 2
TIMIG/SIZE 570 9849646 9849004 -642 detail 802 50 2
TIMIG/SIZE 571 9917292 9916112 -1180 detail 1474 92 2
TIMIG/SIZE 572 9918528 9917348 -1180 detail 1474 92 2
TIMIG/SIZE 573 9919764 9918584 -1180 detail 1474 92 2
TIMIG/SIZE 574 9921000 9920358 -642 detail 802 50 2
TIMIG/SIZE 575 9988646 9987466 -1180 detail 1474 92 2
TIMIG/SIZE 576 9989882 9988702 -1180 detail 1474 92 2
TIMIG/SIZE 577 9991118 9989938 -1180 detail 1474 92 2
TIMIG/SIZE 578 9992354 9991712 -642 detail 802 50 2
-----
total packet # = 578
error packet # = 0
root@zynqmp-generic:/emulator/pcap2bin_conv# cp -rf ./tx.bin ../
root@zynqmp-generic:/emulator/pcap2bin_conv#
```

Tx_pat.bin 파일 실행

```
./emul_pcap_play 1 tx_pat.bin 20 0 1 0
```

여기까지 실행하면 bin파일이 재생된다.

```
root@zynqmp-generic:/emulator# emul_pcap_play 1 tx_pat.bin 20 0 1 0
nbyte autoset to file size
mmap phyaddr(0x40000000) page_size(0xb16000) copy_size(0xb15300)
to mem movesize(8)
0 ( 0x7f9bd82000): 110203040560000
1 ( 0x7f9bd82008): 40000000000007fc
2 ( 0x7f9bd82010): 000000008100e000
3 ( 0x7f9bd82018): 40000000000007fc
4 ( 0x7f9bd82020): aeef100005b00002
5 ( 0x7f9bd82028): 40000000000007fc
6 ( 0x7f9bd82030): 0000000000000000
7 ( 0x7f9bd82038): 40000000000007fc
8 ( 0x7f9bd82040): 0003276e276ed892
9 ( 0x7f9bd82048): 40000000000007fc
a ( 0x7f9bd82050): d892276e276e276e
b ( 0x7f9bd82058): 40000000000007fc
c ( 0x7f9bd82060): 276ed892276ed892
d ( 0x7f9bd82068): 40000000000007fc
e ( 0x7f9bd82070): 276ed892276e276e
f ( 0x7f9bd82078): 40000000000007fc
10 ( 0x7f9bd82080): 276e276e276e276e
11 ( 0x7f9bd82088): 40000000000007fc
12 ( 0x7f9bd82090): d892d892d892276e
13 ( 0x7f9bd82098): 40000000000007fc
14 ( 0x7f9bd820a0): 276e276e276e276e
15 ( 0x7f9bd820a8): 40000000000007fc
16 ( 0x7f9bd820b0): 276e276e276ed892
17 ( 0x7f9bd820b8): 40000000000007fc
18 ( 0x7f9bd820c0): d892276ed892d892
19 ( 0x7f9bd820c8): 40000000000007fc
1a ( 0x7f9bd820d0): 276ed892d892276e
1b ( 0x7f9bd820d8): 40000000000007fc
1c ( 0x7f9bd820e0): d892276e276ed892
1d ( 0x7f9bd820e8): 40000000000007fc
1e ( 0x7f9bd820f0): 276e276e276ed892
1f ( 0x7f9bd820f8): 40000000000007fc
done
filename = tx_pat.bin
file_size = 11021120 Byte
tx_period = 20.000000 ns
tx_count = 0, Infinite play
pps_src = external pps
tx_delay = 0 ns
status 'ING' 00000002
root@zynqmp-generic:/emulator#
```

2.3 ./emul_pcap_play 사용법

그림과 같이 하시면 사용방법이 나옵니다.

“./emul_pcap_play 1 tx_pat.bin 20 0 1 0” 에 의미

1 : enable. 0인경우 패턴재생이 리셋됩니다.

tx_pat.bin : 재생 bin파일이름

20 : 20ms 간격으로 tx_pat.bin파일을 재생하라는 의미

0 : 무제한 출력, 1이면 한번 재생, 2이면 두번재생.....

1 : '1'인 경우, 외부 PPS를 사용, 외부와 동기 하는 경우 1로 설정함. '0' 내부 pps를 사용.

0 : pps가 들어오고 0ns 부터 패킷을 보낸다.

```
root@zynqmp-generic:/emulator# emul_pcap_play -h
usage: emul_pcap_play [-h] enable filename period_ms play_count pps_src delay

./emul_pcap_play <enable> <filename> <period_ms> <play_count> <pps_src> <delay>
./emul_pcap_play 1 tx_pat.bin 20 0 0 0
-> enable      : 1; player enable 0; disable
-> filename    : tx_pat.bin
-> period_ms   : 20 ms
-> play_count  : 0=Infinite, 1=1, 2=2, 3=3.....
-> pps_src     : 0=Internal pps 1=external pps
-> delay       : Unit nanosec.

positional arguments:
  enable      1(enable) or 0(disable)
  filename    bin file name
  period_ms   10~100ms
  play_count  0=Infinite, 1=1, 2=2, 3=3...
  pps_src     0=internal pps, 1=external pps...
  delay       Time delay from pps to tx-start, unit is ns

optional arguments:
  -h, --help  show this help message and exit

InfoCube
root@zynqmp-generic:/emulator#
```

2.4 명령어 요약

```
1  #!/bin/sh
2
3
4  #####
5  ##  와이어나샤크 PCAP을 du-emulator로 출력하는 방법
6  ##  1. 와이어나샤크를 사용하요 '.pcap' 파일을 '.pcapng'
7  #####
8
9
10 #####
11 ##  2. 'pcapng' 파일을 bin 파일로 변환. bin 파일은 du-emulator용 출력 파일.
12 #####
13 telnet이나 ssh로 (윈도우에서는 푸티나, 파일질라를 이용) 192.168.0.101 또는 192.168.0.102 로 접속한다.
14 터미널이 열리면 아래와 같이
15
16 # 변환라이브러리가 있는 폴더로 이동
17 # pcapng 파일을 아래와 같이 bin으로 변환
18 cd /emulator/util
19 ./conv-pcapng-axil28nsec.py ./Pattern_1_full_band_test_20msec.pcapng ./p1_full_band_test_20m.bin 0 0
20 ./conv-pcapng-axil28nsec.py ./Pattern_2_full_band_test_20msec.pcapng ./p2_full_band_test_20m.bin 0 0
21 ./conv-pcapng-axil28nsec.py ./Pattern_3_full_band_test_20msec.pcapng ./p3_full_band_test_20m.bin 0 0
22 ./conv-pcapng-axil28nsec.py ./Pattern_4_full_band_test_20msec.pcapng ./p4_full_band_test_20m.bin 0 0
23 ./conv-pcapng-axil28nsec.py ./Pattern_5_full_band_test_20msec.pcapng ./p5_full_band_test_20m.bin 0 0
24 ./conv-pcapng-axil28nsec.py ./Pattern_6_full_band_test_20msec.pcapng ./p6_full_band_test_20m.bin 0 0
25 ./conv-pcapng-axil28nsec.py ./Pattern_7_full_band_test_20msec.pcapng ./p7_full_band_test_20m.bin 0 0
26 ./conv-pcapng-axil28nsec.py ./Pattern_8_full_band_test_20msec.pcapng ./p8_full_band_test_20m.bin 0 0
27
28 # pcap v플레이어가 있는 폴더로 bin파일 복사
29 cd /emulator/util
30 cp -f ./p1_full_band_test_20m.bin /emulator
31 cp -f ./p2_full_band_test_20m.bin /emulator
32 cp -f ./p3_full_band_test_20m.bin /emulator
33 cp -f ./p4_full_band_test_20m.bin /emulator
34 cp -f ./p5_full_band_test_20m.bin /emulator
35 cp -f ./p6_full_band_test_20m.bin /emulator
36 cp -f ./p7_full_band_test_20m.bin /emulator
37 cp -f ./p8_full_band_test_20m.bin /emulator
38
39
40 #####
41 ##  3. bin 파일출력
42 #####
```

```
root@zynqmp-generic:/emulator# emul_pcap_play 1 tx_pat.bin 20 0 1 0
```

3. Capture Data & PCAP으로 저장

3.1 사용법-Capture

아래 그림을 참조

cd /emulator/

./emul_capture 4 0 → 1pps 이후 0ns부터 들어오는 40ms동안에 데이터를 저장.

./emul_capture 2 0 → 1pps 이후 0ns부터 들어오는 20ms동안에 데이터를 저장.

```
root@zynqmp-generic:/emulator# ./emul_capture -h
usage: emul_capture [-h] number delay

    ./emul_capture <number> <delay>
    ./emul_capture 10 1000
        -> number : Number of frames in units of 10ms
                   Capture for 100ms( 10ms x10 )
        -> delay  : Time delay from pps to start of capture, unit is ns.
                   1000 ns delay

positional arguments:
  number      bin file name
  delay      10~100ms

optional arguments:
  -h, --help  show this help message and exit

InfoCube
root@zynqmp-generic:/emulator# ./emul_capture 4 0

##Capture information
  Period   = 40 ms
  Number   = 4
  Delay    = 0 ns
  .
  .
  .
Capture done
capture frame scatter-gather list
  R4  0x1000_0000 ~ 10023000
  R5  0x2000_0000 ~ 20023000
  R6  0x3000_0000 ~ 30023000
  CC  0x4000_0000 ~ 40023000

root@zynqmp-generic:/emulator#
```


3.2 저장된 패킷을 '.PCAP'으로 저장

메모리에 저장된 데이터를 아래 명령으로 PCAP파일로 저장한다.

```
cd /emulator/util
```

```
./save_port0      <- R4 저장
```

```
./save_port1      <- R5 저장
```

```
./save_port2      <- R6 저장
```

```
./save_port3      <- CC 저장
```

```
root@zynqmp-generic:/# cd /emulator/util/
root@zynqmp-generic:/emulator/util# ./save_port
save_port0 save_port1 save_port2 save_port3
root@zynqmp-generic:/emulator/util# ./save_port0
error / file exist (ultest.bin2k)
file(ultest.bin2k) branch(0) nframe(0)
show all controller info
CTRL : 0x(      0)
STATUS: 0x(      1)
BR(0) START : 0x(    1000) END : 0x(10025800)
BR(1) START : 0x(    2000) END : 0x(20025800)
BR(2) START : 0x(    3000) END : 0x(30000000)
BR(3) START : 0x(    4000) END : 0x(40000000)
-----
LIST BR(0)
PA64_CTRL : 0x(      80a20000)
PA64_START : 0x(    410000000)
PA64_END : 0x(    410025800)
list : (9600)
-----
auto size is (9600)
final nframe :(9600)
N( 0) offset(266c6000) blk64( 1536) ==> PA64(    5266c6000)
N( 1) offset(266c6800) blk64( 1152) ==> PA64(    5266c6800)
N( 2) offset(266c7000) blk64( 128) ==> PA64(    5266c7000)
N( 3) offset(266c7800) blk64( 128) ==> PA64(    5266c7800)
N( 4) offset(266c8000) blk64( 128) ==> PA64(    5266c8000)
N( 5) offset(266c8800) blk64( 128) ==> PA64(    5266c8800)
N( 6) offset(266c9000) blk64( 128) ==> PA64(    5266c9000)
```

중략.....

```
n(8) pa64( 5266ca000) ts(      0 : 0.000000000) ncopy( 128)
n(9) pa64( 5266ca800) ts(      0 : 0.000000000) ncopy( 128)
n(10) pa64( 5266cb000) ts(      0 : 0.000000000) ncopy( 1536)
n(11) pa64( 5266cb800) ts(      0 : 0.000000000) ncopy( 1152)
n(12) pa64( 5266cc000) ts(      0 : 0.000000000) ncopy( 1536)
n(13) pa64( 5266cc800) ts(      0 : 0.000000000) ncopy( 1152)
n(14) pa64( 5266cd000) ts(      0 : 0.000000000) ncopy( 1536)
n(15) pa64( 5266cd800) ts(      0 : 0.000000000) ncopy( 1152)
n(16) pa64( 5266ce000) ts(      0 : 0.000000000) ncopy( 1536)
n(17) pa64( 5266ce800) ts(      0 : 0.000000000) ncopy( 1152)
n(18) pa64( 5266cf000) ts(      0 : 0.000000000) ncopy( 1536)
n(19) pa64( 5266cf800) ts(      0 : 0.000000000) ncopy( 1152)
done
local pcapng
main
branch1 ./ultest.bin2k ./ultest_102_br0.pcapng 0 9600
-----
Opening output file
PKT_SIZE+FCS 1 1478
PKT_SIZE+FCS 2 1090
PKT_SIZE+FCS 3 1478
PKT_SIZE+FCS 4 1090
PKT_SIZE+FCS 5 1478
PKT_SIZE+FCS 6 1090
PKT_SIZE+FCS 7 1478
PKT_SIZE+FCS 8 1090
PKT_SIZE+FCS 9 1478
PKT_SIZE+FCS 10 1090
stop / file end
-----
total packet # = 9601
error packet # = 0
root@zynqmp-generic:/emulator/util#
```


4. Terminal 명령어

4.1 MAC 변경

PCAP 파일 재생하여 출력할 때, DST, SRC MAC주소를 변경하는 기능

4.1.1 MAC 변경 활성화

```
##mac modify enable
```

```
devmem 0x80000020 32 0xff
```

```
##src-mac 66:55:44:33:22:11
```

```
devmem 0x80000070 6 0x0000665544332211
```

```
##R7 dst-mac a7:22:33:44:55:66
```

```
##R6 dst-mac a6:22:33:44:55:66
```

```
##R5 dst-mac a5:22:33:44:55:66
```

```
##R4 dst-mac a4:22:33:44:55:66
```

```
##R3 dst-mac a3:22:33:44:55:66
```

```
##R2 dst-mac a2:22:33:44:55:66
```

```
##R1 dst-mac a1:22:33:44:55:66
```

```
##R0 dst-mac a0:22:33:44:55:66
```

```
devmem 0x80000030 64 0x00006655443322a7
```

```
devmem 0x80000038 64 0x00006655443322a6
```

```
devmem 0x80000040 64 0x00006655443322a5
```

```
devmem 0x80000048 64 0x00006655443322a4
```

```
devmem 0x80000050 64 0x00006655443322a3
```

```
devmem 0x80000058 64 0x00006655443322a2
```

```
devmem 0x80000060 64 0x00006655443322a1
```

```
devmem 0x80000068 64 0x00006655443322a0
```

4.1.1 MAC 변경 비활성

```
devmem 0x80000020 32 0x00
```

4.2 Ping

Loopback으로 케이블을 연결하면 ping이 안됨

```
ifconfig eth1 xxx.xxx.xxx.xxx(Port IP)
```

```
ping xxx.xxx.xxx.xxx(상대IP)
```

```
#####
```

'Mac Table Reset'이 필요한 경우..... 노랑박스를 실행한다.

```
#####
```

```
#args 1 : Aging Time(Aging Time = time + time/2)
```

```
#args 2 : Aging Enable
```

```
#args 3 : Entry Clear
```

```
#hub_mactable(200, 1, 1)
```

```
devmem 0x80900008 32 (1<<25 | 1<<24 | 200)
```

```
sleep(2)
```

```
devmem 0x80900008 32 (0<<25 | 1<<24 | 200)
```

4.3 Fan Control

```
echo 10 > /sys/class/hwmon/hwmon1/pwm1
```

```
echo 10 > /sys/class/hwmon/hwmon1/pwm2
```

```
echo 10 > /sys/class/hwmon/hwmon1/pwm3
```

```
echo 10 > /sys/class/hwmon/hwmon1/pwm4
```

최대 RPM

```
echo 250 > /sys/class/hwmon/hwmon1/pwm1
```

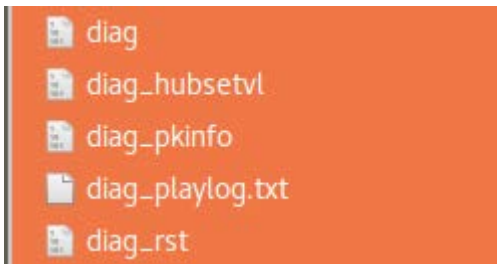
```
echo 250 > /sys/class/hwmon/hwmon1/pwm2
```

```
echo 250 > /sys/class/hwmon/hwmon1/pwm3
```

```
echo 250 > /sys/class/hwmon/hwmon1/pwm4
```

4.4 진단 기능 명령어

/emulator 폴더에 다음 그림과 같은 진단 기능이 있다.



/emulator/diag_rst : 진단정보를 리셋한다.

/emulator/diag :

/emulator/diag_hubsetvl :

/emulator/diag_play_log.txt :

/emulator/diag_pkinf :

diag_rst	모든 진단정보가 '0'으로 리셋된다.
diag	계측기버전, 클락소스체크, pps소스체크, 링크업상태, MAC 수정여부를 보여준다.
diag_hubsetvl	이더넷타입과 vlanid를 설정한다. 설정이 안된 패킷은 드랍된다.
diag_play_log.txt	Pcap_playe가 실행한 파일이 기록된다.
diag_pkinf	송수신바이트 정보가 표시된다.

4.4.1 diag_rst

다음과 같이 실행한다.

```
root@zynqmp-generic:/emulator# ./diag_rst

RESET 10M & 1-PPS Clock Souce Check
Emulator tx packet counter
Emulator rx link status
Emulator Packet Counter

RUN ./diag

root@zynqmp-generic:/emulator#
```

4.4.2 diag

##Version

생성날짜 : 2020.01.05

type: 100(계측기)

Version: 2.0

##Clock Souce Check

클럭상태: 외부 10M, 내부시스템클럭, 내부 pps, 외부 pps, 모두 정상인 상태.

##1-PPS Source

pps소스 : 외부 PPS

##1-PPS Source

fpga - phy - ext

모든 링크가 다 붙은 상태를 나타낸다.

```
##Version
  FPGA Bitstream Date       : 20201005
  FPGA Bitstream Version    : 10020001

##Clock Souce Check
0x8000_0028 : 0x00000037
0x28[0]exit_ext10m : 1, [16] 1->0 change : 0
0x28[1]locked_sysclk : 1, [17] 1->0 change : 0
0x28[2]exit_gen1pps : 1, [18] 1->0 change : 0
0x28[4]exit_ext1pps : 1, [20] 1->0 change : 0
0x28[5]exit_int1pps : 1, [21] 1->0 change : 0

##1-PPS Source
0x8000_0024 : 0x00001000
0x8000_0024[13:12] : 1
  1-PPS_SRC is external 1-pps reference

L0 : 0x0a0000
L1 : 0x200011

##Link Up Check
=====
| fpga-phy-linkUp | phy-ext-linkUp |
=====
L1 | 1 | 1 |
R4 | 1 | 1 |
R5 | 1 | 1 |
R6 | 1 | 1 |
=====

##SRC MAC, DST MAC Modify : Disable
```

4.4.3 diag_hubsetvl

이 명령을 이용해서 vlan-id를 잡아준다. 그렇지 않은 경우 패킷이 드롭된다.

```
root@zynqmp-generic:/emulator# ./diag_hubsetvl -h
main
usage: diag_hubsetvl [-h] vlanslot vlanid_en vlantype_en vlanid vlantype

Setting vlan
./hubsetvl <slot 0~3> <vlan.id_en> <vlan.type_en> <vlan.id> <vlan.type>
./hubsetvl 0 1 0 0x03E8 0xAEFE
-> slot=0
-> vlan.id_enable = 1, vlan.id = 0x03E8
-> vlan.type_en = 0, vlan.type = 0xAEFE

positional arguments:
  vlanslot      0, 1, 2, 3
  vlanid_en     0 or 1
  vlantype_en   0 or 1
  vlanid        0x3E8...
  vlantype      0xAEFE...

optional arguments:
  -h, --help  show this help message and exit

InfoCube
root@zynqmp-generic:/emulator#
```

4.4.4 diag_pkinfo

위 명령을 실행하면 다음과 같은 정보를 보여준다.

```
##Version
  FPGA Bitstream Date      : 20201005
  FPGA Bitstream Version   : 10020001

##Clock Souce Check
  0x8000_0028              : 0x00000037
  0x28[0]exit_ext10m       : 1, [16] 1->0 change : 0
  0x28[1]locked_sysclk     : 1, [17] 1->0 change : 0
  0x28[2]exit_genipps      : 1, [18] 1->0 change : 0
  0x28[4]exit_extipps      : 1, [20] 1->0 change : 0
  0x28[5] exit_intipps     : 1, [21] 1->0 change : 0

##1-PPS Source
  0x8000_0024              : 0x00001000
  0x8000_0024[13:12]      : 1
  1-PPS_SRC is external 1-pps reference

## Pcap-player Tx frame number per tx-tick
  Accumulative frame       : 1540700
  Minimum frame            : 0
  Maximum frame            : 224000

##Vlan Check
=====
|  | vlan.etype | | etype.en ||  | vlan.id | | id.en | |
=====
R4:0| 0xaefe | | 1 || 0x000 | | 1 | |
1| 0xaefe | | 1 || 0x3e8 | | 1 | |
2| 0xaefe | | 1 || 0x1a8 | | 1 | |
3| 0x0000 | | 0 || 0x000 | | 0 | |
-----
R5:0| 0xaefe | | 1 || 0x000 | | 1 | |
1| 0xaefe | | 1 || 0x3e8 | | 1 | |
2| 0xaefe | | 1 || 0x1a8 | | 1 | |
3| 0x0000 | | 0 || 0x000 | | 0 | |
-----
R6:0| 0xaefe | | 1 || 0x000 | | 1 | |
1| 0xaefe | | 1 || 0x3e8 | | 1 | |
2| 0xaefe | | 1 || 0x1a8 | | 1 | |
3| 0x0000 | | 0 || 0x000 | | 0 | |
-----
CC:0| 0xaefe | | 1 || 0x000 | | 1 | |
1| 0xaefe | | 1 || 0x3e8 | | 1 | |
2| 0xaefe | | 1 || 0x1a8 | | 1 | |
3| 0x0000 | | 0 || 0x000 | | 0 | |
-----

##Packet Count Check
=====
|  | Packet Count | | TX | | RX | |
=====
R4: | Byte | 10563579978 | 98084174094 |
   | Broadcast | 0 | 0 |
   | Multicast | 0 | 0 |
   | Unicast | 5516356 | 76402477 |
   | VlanidMatch | 2790707 | 74996816 |
-----
R5: | Byte | 10550608442 | 98100275574 |
   | Broadcast | 0 | 0 |
   | Multicast | 0 | 0 |
   | Unicast | 5506062 | 76415019 |
   | VlanidMatch | 2790768 | 74996960 |
-----
R6: | Byte | 10520817808 | 98144923260 |
   | Broadcast | 0 | 0 |
   | Multicast | 0 | 0 |
   | Unicast | 5482450 | 76449808 |
   | VlanidMatch | 2790816 | 75003312 |
-----
CC: | Byte | 8027752032 | 101192072322 |
   | Broadcast | 0 | 0 |
   | Multicast | 0 | 0 |
   | Unicast | 3511942 | 78823371 |
   | VlanidMatch | 2790867 | 75003360 |
-----
```

4.5 Link Up Check

모든 포크가 링크업 상태일 때, 'diag' 실행 시, 다음과 같이 나타난다.

```
##RX Optic Link status : 0x20_0011, linkup, rx valid code, rx block lock
  L0 : 0x0a002e
  L1 : 0x200011
##Link Up Check
=====
      | fpga-phy-linkUp | phy-ext-linkUp |
=====
L1 |          1          |          1          |
R4 |          1          |          1          |
R5 |          1          |          1          |
R6 |          1          |          1          |
-----
```

L0, L1 광포트 링크 상태.

21	Rxlink-up
20	stat_rx_received_local_fault
19	stat_rx_internal_local_fault
18	stat_rx_remote_fault
17	stat_rx_local_fault
16	stat_rx_bad_sfd
15	stat_rx_bad_preamble
14	stat_rx_stomped_fcs
13	stat_rx_packet_bad_fcs
12	stat_rx_bad_fcs
11	stat_rx_toolong
10	stat_rx_undersize
9	stat_rx_oversize
8	stat_rx_packet_large
7	stat_rx_jabber
6	stat_rx_packet_small
5	stat_rx_bad_code
4	stat_rx_valid_ctrl_code
3	stat_rx_hi_ber
2	stat_rx_framing_err
1	stat_rx_framing_err_valid
0	Always '1'

4.6 클럭동기 & 타이밍 조절

외부 PPS에 동기 방법1

계측기전체가 리셋되면서 변경된다.

```
root@zynqmp-generic:/emulator# ./diag_pps_sel -h
main
usage: diag_pps_sel [-h] pps_sel

    Setting pps_sel
    ./diag_pps_sel <pps_sel>
    ./diag_pps_sel 0
        -> pps_sel= 0: int_pps 1: ext_pps

positional arguments:
  pps_sel      0: int_pps 1: ext_pps

optional arguments:
  -h, --help  show this help message and exit

InfoCube
root@zynqmp-generic:/emulator#
```

외부 PPS에 동기 방법2

pcap_play를 실행할 때, 외부 PPS로 설정하고 사용하면, 캡처기능도 외부 PPS에 동기된다.

```
root@zynqmp-generic:/emulator# ./emul_pcap_play 1 tx_pat.bin 20 0 1 0
```

TX Packet 지연방법

외부동기신호 1PPS Edge에 맞춰서 패킷을 전송한다. 이 1PPS를 기준으로 패킷이 나가는 시간 지연을 줄 수 있다.

```
root@zynqmp-generic:/emulator# emul_pcap_play -h

usage: emul_pcap_play [-h] enable filename period_ms play_count pps_src

    ./emul_pcap_play <enable> <filename> <period_ms> <play_count> <pps_src>
    ./emul_pcap_play 1 tx_pat.bin 20 0 0
        -> enable      : 1; player enable 0; disable
        -> filename   : tx_pat.bin
        -> period_ms  : 20 ms
        -> play_count : 0=Infinite, 1=1, 2=2, 3=3.....
        -> pps_src    : 0=Internal pps 1=external pps

positional arguments:
  enable      1(enable) or 0(disable)
  filename    bin file name
  period_ms   10~100ms
  play_count  0=Infinite, 1=1, 2=2, 3=3...
  pps_src     0=internal pps, 1=external pps...

optional arguments:
  -h, --help  show this help message and exit

InfoCube
root@zynqmp-generic:/emulator#
```

캡처 타이밍 조절방법

```
root@zynqmp-generic:/emulator# ./emul_capture -h
usage: emul_capture [-h] number delay

./emul_capture <number> <delay>
./emul_capture 10 1000
-> number : Number of frames in units of 10ms
             Capture for 100ms( 10ms x10 )
-> delay   : Time delay from pps to start of capture, unit is ns.
             1000 ns delay

positional arguments:
  number      capture frame number
  delay       Unit nanosec

optional arguments:
  -h, --help  show this help message and exit

InfoCube
root@zynqmp-generic:/emulator#
```


5. LUA Script 명령

5.2. POE 파워상태모니터

//args 1 : channel-mask(1:Enable, 0:Disable, lsb:channel0)

//args 2 : print time : 횟수

//args 3 : print interval 초

POE-R4: poe_pwrprint(0x10, 1, 1)

POE-R5: poe_pwrprint(0x20, 1, 1)

POE-R6: poe_pwrprint(0x40, 1, 1)

ex) # cd /emulator/util/

./Api

```
root@zynqmp-generic:/emulator#
root@zynqmp-generic:/emulator# cd /emulator/util/
root@zynqmp-generic:/emulator/util# ./Api

*****
**Lua version : Lua 5.3
**Lua release version : Lua 5.3.3
**Lua 5.3.3 Copyright (C) 1994-2016 Lua.org, PUC-Rio [R. Ierusalimschy, L. H. d
e Figueiredo, W. Celes]
*****

Note::Lua Do-file(ApiHub.lua) absence cannot open ApiHub.lua: No such file or di
rectory

pleth-lua> poe_pwrprint(0x10, 1, 1)
Device opened successfully
Open session was called with API Version 1.4 and result is 0

Code 81 Get Total Power, result PoE_RC_SUCSESS
-->powerConsumption.....29.000W
-->calculatedPower.....29.000W
-->availablePower.....730.000W
-->powerLimit.....760.000W
-->powerBank.....0xF
-->vmainVoltage.....56.200V

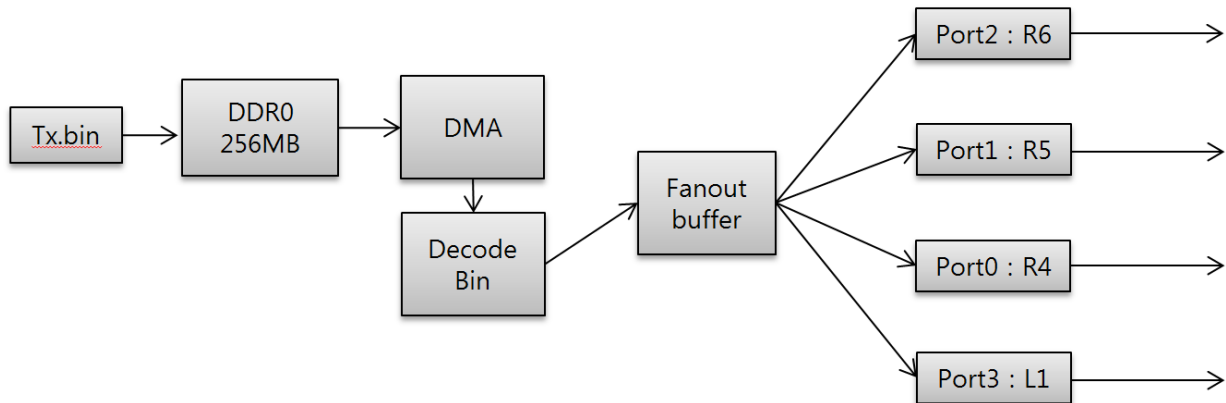
Port-4::Code 87 Get Power Port, result PoE_RC_SUCSESS
-->vmainVoltage.....56.200V
-->calculatedCurrent.....0.526A
-->measuredPortPower.....29.500W
-->portVoltage.....56.400V

Code 81 Get Total Power, result PoE_RC_SUCSESS
-->powerConsumption.....29.000W
-->calculatedPower.....29.000W
-->availablePower.....730.000W
-->powerLimit.....760.000W
-->powerBank.....0xF
-->vmainVoltage.....56.200V

Port-4::Code 87 Get Power Port, result PoE_RC_SUCSESS
-->vmainVoltage.....56.200V
-->calculatedCurrent.....0.526A
-->measuredPortPower.....29.500W
-->portVoltage.....56.400V
pleth-lua> 
```

6. 사양 & 특성

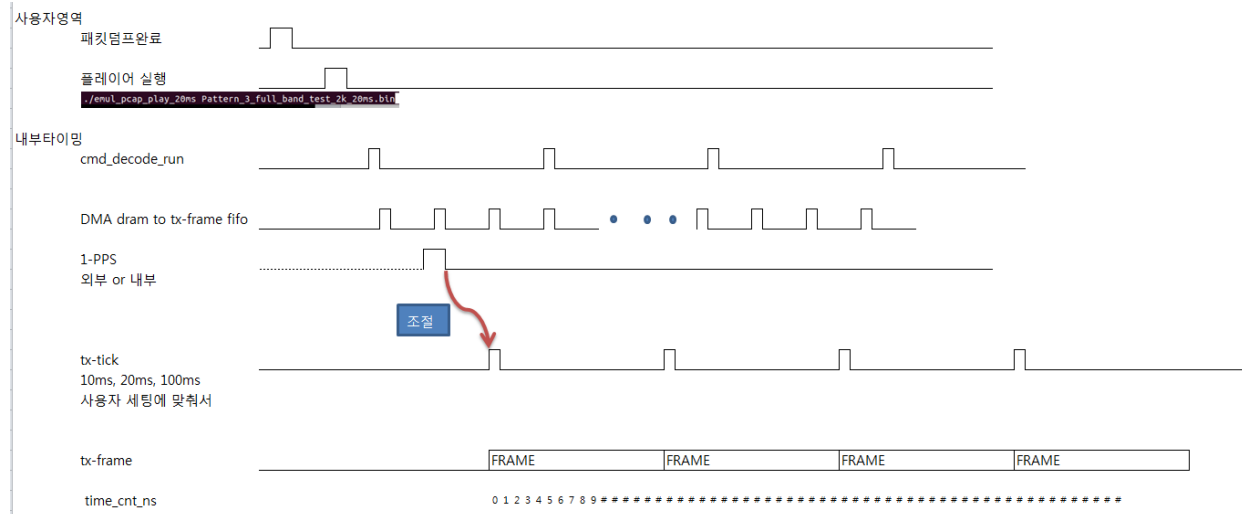
5.1 Player(TX)



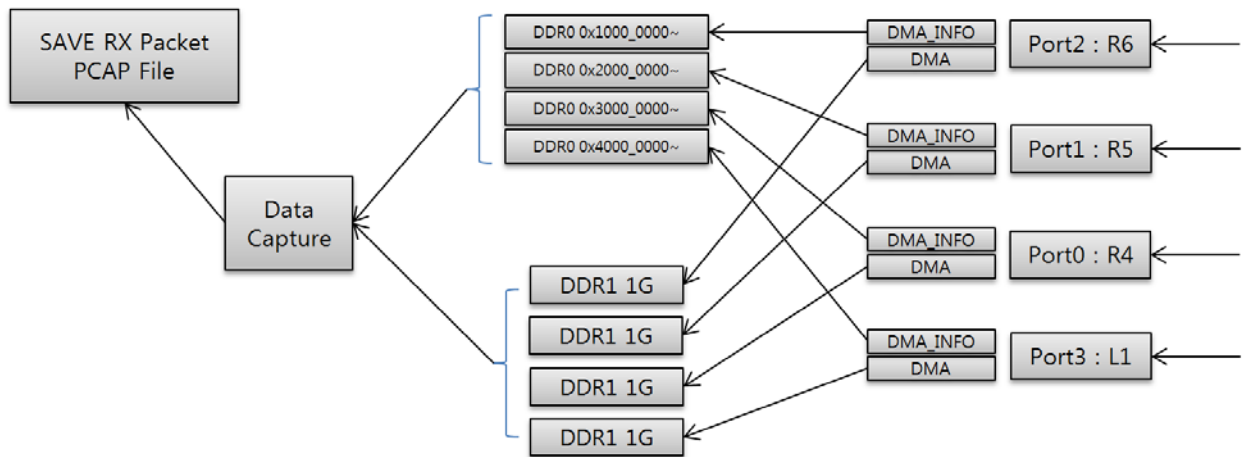
.bin 파일을 메모리 올려 놓고, 사용자가 설정한 10ms? 20ms? 주기로 읽어서 출력한다.

pcap 파일이 20ms 데이터 세트라면, 사용자는 20ms 이상으로 tx-tick을 설정해야 한다.

5.2 TX Packet Timing



5.3 Capture(RX)

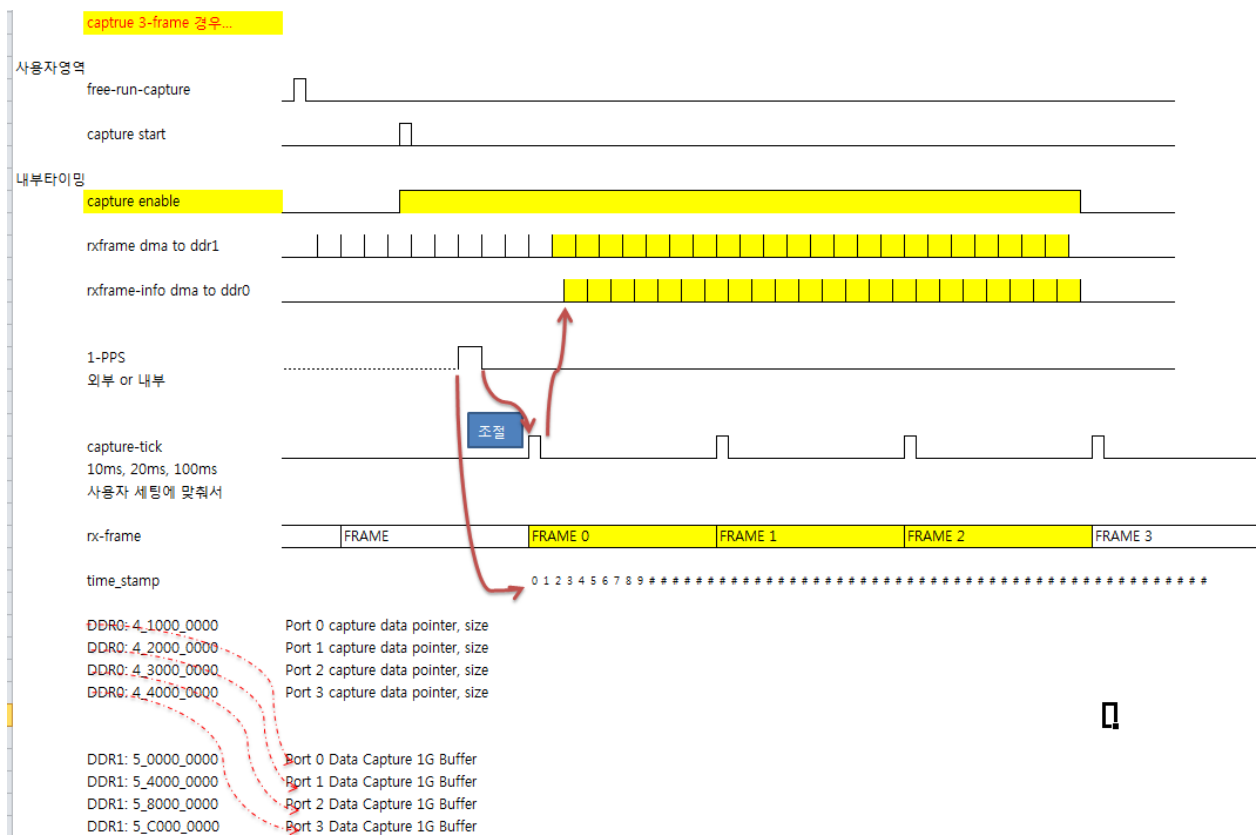


Capture로직은 U-Plane Data만 캡처가 가능하다. 최대 용량은 Port당 1GB만큼 저장 가능하다.

'cap_re-run==1'이 되면 'DDR1'으로 패킷이 계속 저장된다. 'cap_run==1'이 되면, 1초 시스템 동기 신호에 cap_tick을 동기 시키고, 바로 이어오는 cap_tick에 맞춰서, 캡처를 시작한다. 캡처가 시작되면, DDR1에는 패킷이 저장되고, DDR0에는 DDR1에 저장되는 패킷의 저장주소가 저장된다. 캡처는 사용자가 설정한 cap_tick의 개수만큼 저장된다.

저장이 완료되면, DDR0에 저장된 DDR1의 주소를 참조해서 .PCAP 파일로 저장한다.

5.4 RX Packet Timing



7. Register Map

7.1 MAC Control Register

MAC & Control

address	register	bit	type	Description
0x80000020	MAC overriding ENABLE	[7:0]	R/W	0xff: 재생하는 원본 pcap에 MAC을 변경하여 재생 0x00: 재생하는 원본 pcap 파일에 MAC을 그대로 재생
0x80000024	DlyAlign_En HUB_EmulatorMode HUB_DlyMode_output sel_1sec_pulse	[0] [8] [9] [13:12]	R/W R/W R/W R/W	항상 '1' 항상 '1' 항상 '0' 1초를 펄스만드는 소스를 선택한다. 이 1초 펄스는 내부 로직을 동기시키는 신호이다. 내부 로직은 pcap-플레이어, pcap-캡처로직이다. 이 동기 신호에 맞춰서 pcap을 재생하고, 캡처한다. 0: 시스템클럭 156.25Mz를 소스로 1초펄스를 만들어 동기 신호로 사용한다 1: IDT Clock 소스에서 나오는 1초펄스를 사용한다 2: 외부 EXT 1PPS를 1초펄스로 사용한다.
0x80000028	CFG_PCID_Cmd CFG_PCID_CmdIdx CFG_PCID_Mu CFG_PCID_PktPerSym CFG_PCID_Id	[1:0] [5:2] [10:6] [15:12] [31:16]	R/W R/W R/W R/W R/W	
0x8000002C	CFG_PCID_Timeout CFG_PCID_OperFlag	[23:0] [31:24]	R/W R/W	
0x80000030	R0 destination MAC[63:0]		R/W	
0x80000038	R1 destination MAC[63:0]		R/W	
0x80000040	R2 destination MAC[63:0]		R/W	
0x80000048	R3 destination MAC[63:0]		R/W	
0x80000050	R4 destination MAC	[63:0]	R/W	MAC_Overriding_Enable == 1 인 경우 이 값으로 나간다.
0x80000058	R5 destination MAC	[63:0]	R/W	MAC_Overriding_Enable == 1 인 경우 이 값으로 나간다.
0x80000060	R6 destination MAC	[63:0]	R/W	MAC_Overriding_Enable == 1 인 경우 이 값으로 나간다.
0x80000068	CC destination MAC	[63:0]	R/W	MAC_Overriding_Enable == 1 인 경우 이 값으로 나간다.
0x80000070	source MAC	[63:0]	R/W	MAC_Overriding_Enable == 1 인 경우 이 값으로 나간다.

7.2 Pcap_player Register

Pcap_player

address	register	bit	type	Description
0x80A00000	reg_vector_address	[26:0]	R/W	tx.bin' 파일이 올라가는 메모리 주소 0x5_0000_0000 >> 9 = 항상 '0x02800000'
0x80A00004	reg_vector_length	[31:0]	R/W	패턴파일 'X' / 4096 +1' 값이 들어간다. 패턴파일 크기가 245472KB인 경우, 245472KB / 4096KB + 1 = '60'
0x80A00008	play_sts_err_cmdvector_seq play_sts_err_unknown play_sts_sts_ing play_sts_sts_cpl	[31] [30] [1] [0]	RO RO RO RO	tx.bin 파일에 문제가 있는 경우, 플레이어 재생중 예측되지 않은 문제가 있는 경우, 플레이어 재생중 플레이어가 'reg_player_lp_cnt' 만큼 재생되면 올라온다.
0x80A0000C	pcap_player_reset pcap_player_statistic_reset pcap_plaer_run	[31] [30] [0]	R/W R/W R/W	pcap 파일을 재생하기 전에 리셋이 필요함 pcap 플레이어에 tx-frame-counte를 리셋한다. pcap 파일 재생을 시작한다.
0x80A00010	reg_player_lp_cnt	[31:0]	R/W	패턴파일이 재생되는 횟수. 0: 무한반복 1: 1 회 2: 2 회.....
0x80A00014	reg_tx_tick_gen_cnt	[31:0]	R/W	tx_tick의 주기를 설정한다. 기본값 (156,250,000-1) / 100 = 10ms 1,562,500 : 10ms 3,124,999 : 20ms
0x80A00024	tx_tick_delaycnt	[31:0]	R/W	1sec_pulse 기준으로 tx_tick이 발생하는 지연을 줄수 있다. pcap 플레이어는 tx_tick에 맞추어서 재생한다. 6.4ns/1-clk

7.3 Pcap_Capture Register

Pcap_capture

address	register	bit	type	Description
0x80A20000	R4 reg_cap_start_addr	[31:0]	R/W	항상 '0x1000' 캡처데이터가 저장된, 메모리 시작 주소
0x80A20004	R5 reg_cap_start_addr	[31:0]	R/W	항상 '0x2000' 캡처데이터가 저장된, 메모리 시작 주소
0x80A20008	R6 reg_cap_start_addr	[31:0]	R/W	항상 '0x3000' 캡처데이터가 저장된, 메모리 시작 주소
0x80A2000C	CC reg_cap_start_addr	[31:0]	R/W	항상 '0x4000' 캡처데이터가 저장된, 메모리 시작 주소
0x80A20010	reg_delayblock_reset	[31]	R/W	delay block reset... capture에 상위 로직 리셋
	reg_cap_rst	[30]	R/W	capture 로직 리셋
	reg_cap_prerun	[1]	R/W	dram에 패킷을 저장을 시작한다. Start에 앞서 '1'을 준다
	reg_cap_start	[0]	R/W	start가 '1'이 되면, capture_tick에 맞춰서 캡처 주소를 저장한다. reg_capframe_count 만큼 캡처를 하고 정지한다.
0x80A20014	capture_cpl	[0]	RO	캡처 완료
	capture_ing	[1]	RO	캡처 진행중
	capture_err	[2]	RO	캡처 에러
0x80A20018	R4 reg_cap_end_addr	[31:0]	RO	캡처데이터가 저장된, 메모리 마지막 주소
0x80A2001C	R5 reg_cap_end_addr	[31:0]	RO	캡처데이터가 저장된, 메모리 마지막 주소
0x80A20020	R6 reg_cap_end_addr	[31:0]	RO	캡처데이터가 저장된, 메모리 마지막 주소
0x80A20024	CC reg_cap_end_addr	[31:0]	RO	캡처데이터가 저장된, 메모리 마지막 주소
0x80A20028	reg_cap_tick_gen_cnt	[31:0]	R/W	capture_tick의 주기를 설정한다. 기본값 (156,250,000-1) / 100 = 10ms 1,562,500 : 10ms 3,124,999 : 20ms
0x80A2002C	reg_capframe_delay	[31:0]	R/W	1sec_pulse 기준으로 capture_tick이 발생하는 지연을 줄수 있다. capture는 capture_tick에 맞추어서 저장한다. 6.4ns/1-clk
0x80A20030	reg_capframe_count	[31:0]	R/W	capture frame 수를 설정한다. 현재는 1~4 Frame 저장가능

8..... TBD