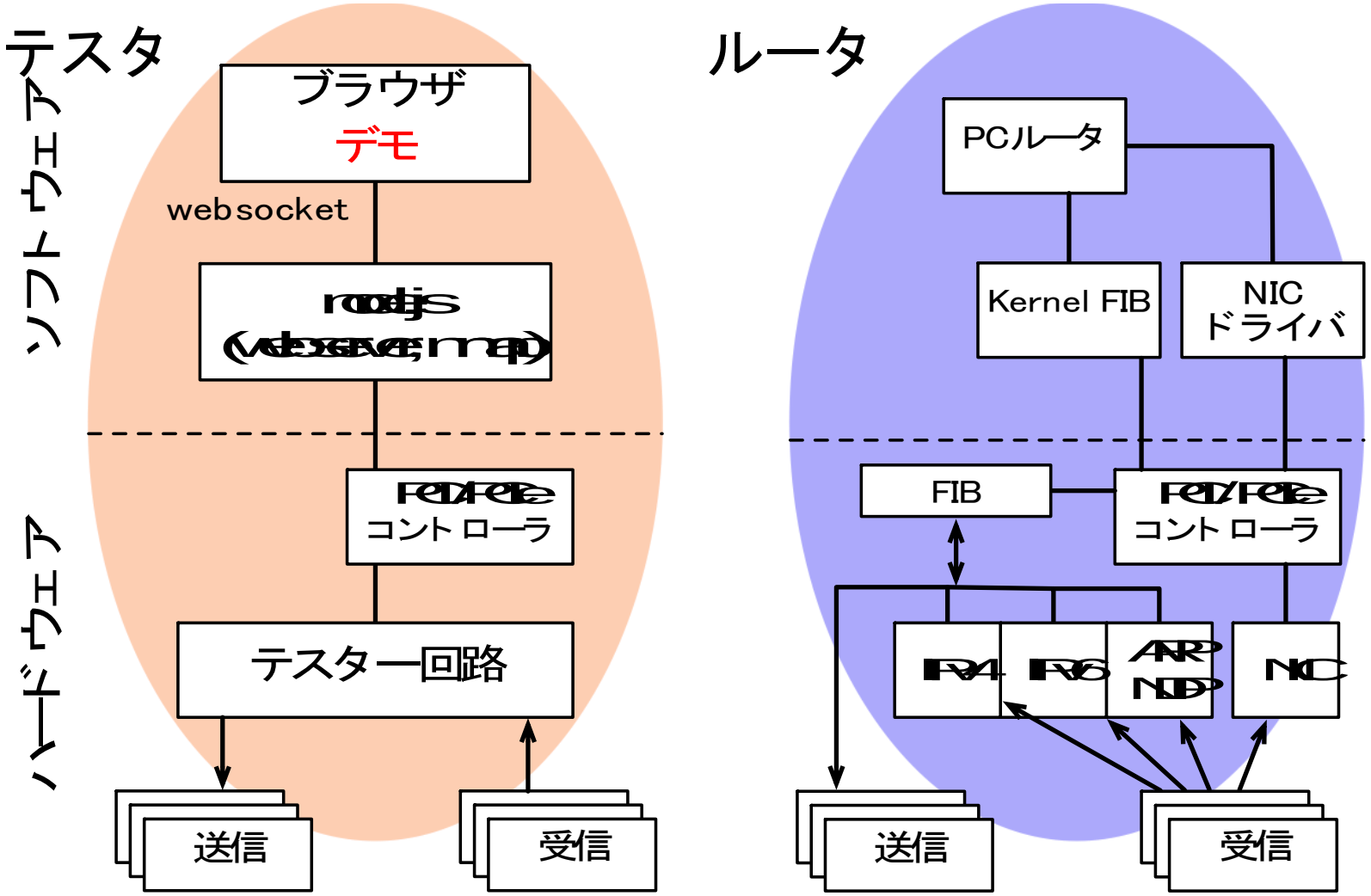
チームまるたか
~コマンドラインでお手軽SDN+α~

慶應義塾大学 空閑洋平

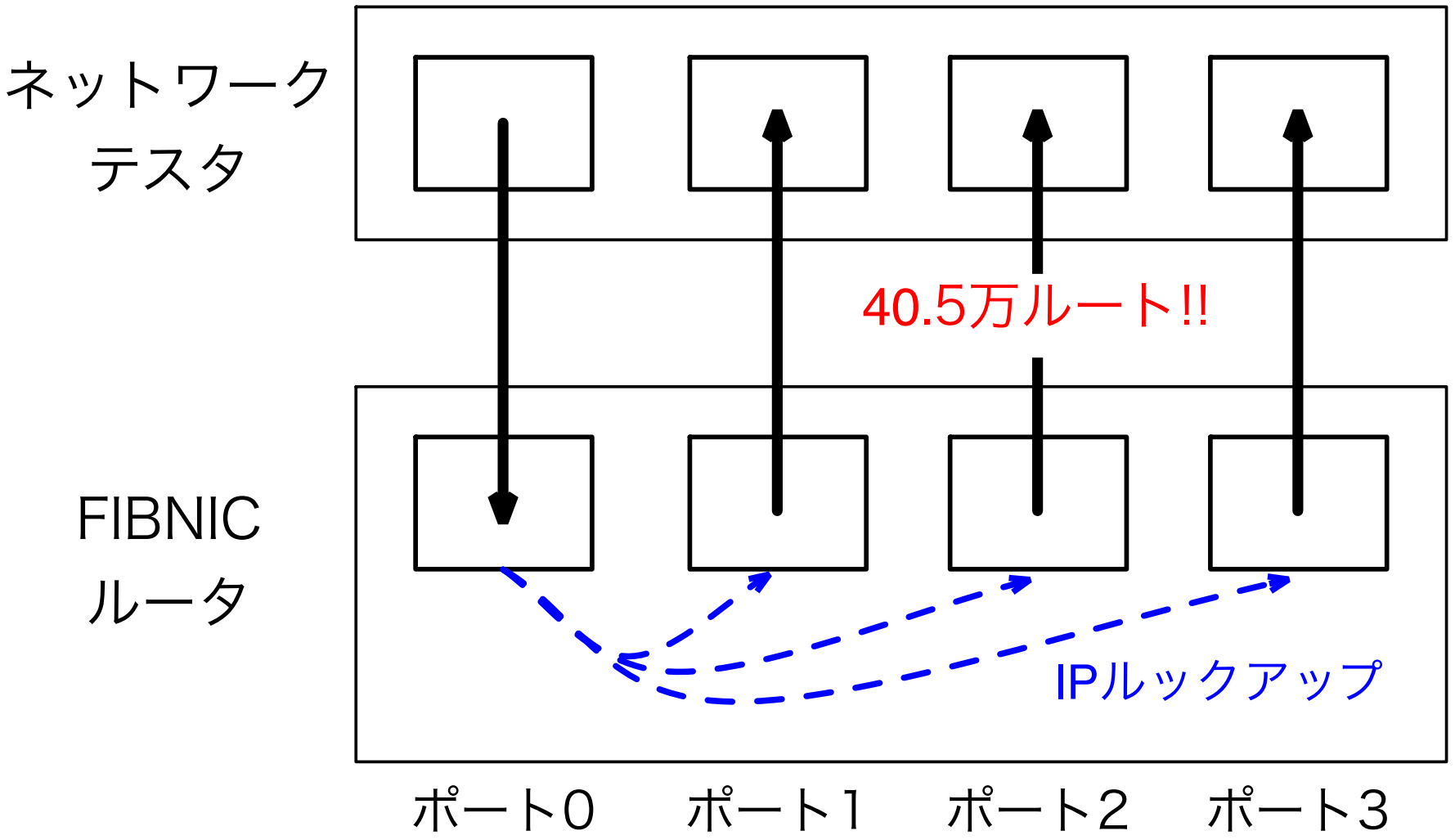
sora@haeena.net

ORC night #1 (2013/1/28)

FIBNIC概要


テスタ
ルータ
ソフトウェア
ハードウェア
ブラウザ
デモ
websocket
PCルータ
Kernel FIB
NIC
ドライバ
FIB
コントローラ
コントローラ
テスター回路
IPv4
IPv6
ARP
NDP
NIC
送信
受信
送信
受信

FIBNIC概要



賞金の使い道

- 主にFPGA+10G開発環境の整備
- Altera DE0 Board
- Xilinx Kintex-7 FPGA KC705 Evaluation Kit
  - Kintex-7 FPGA, SFP+ゲージ x1
- Xilinx Kintex-7 FPGA Connectivity Kit
  - Kintex-7 FPGA, SFP+ゲージ x5, SFP+モジュール x2
- Mellanox ConnectX-3 10G Adapter
- 各種ソフトウェア・ライセンス, etc.

FIB NICからの進展

- FIBNIC (cut-throughのL3エンジン, QoS)
- FPGA HUB
  - 2万円くらいの自作筐体
  - 5k円FPGA + 1000BASE-T 8ポートだけの箱
- **EtherPIPE Adapter**
  - Linuxのデータプレーン用インタフェースを検討
  - 今日の話

元々は…

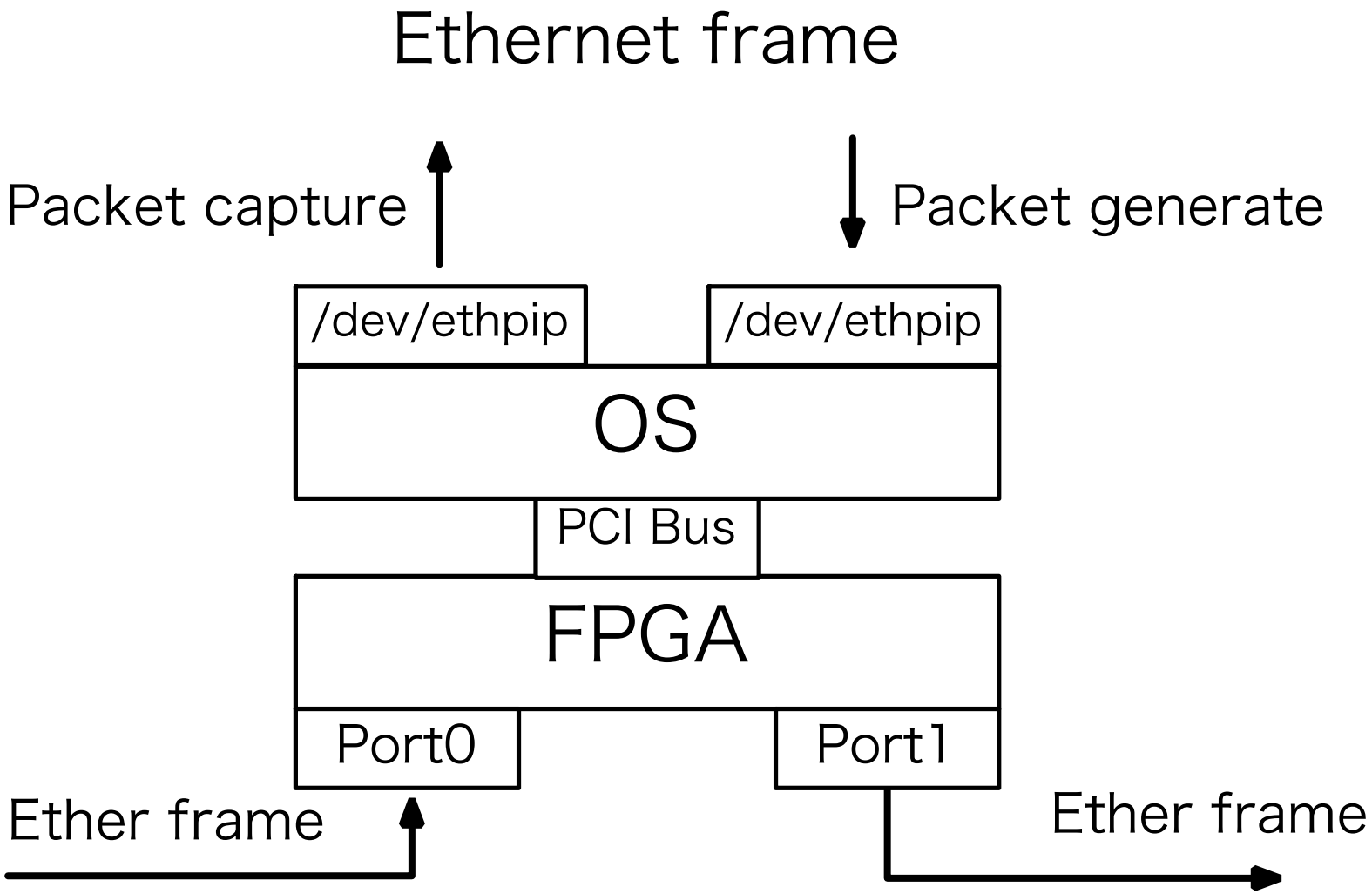
- 自作デバイス検証用にネットワークテスタがほしい
  - 個人で買うには高い + すぐに必要だった

  ⇛ RFC2544を参考にしたFPGA回路
    - Throughput, PPS, Latency (8ns単位)の計測
    - 64 byte + line-rateなパケット送受信機能

  - キャプチャしたトラフィックデータの送受信機能
    - Pcapを**簡単に**FPGAから送受信したい

    个を作っていました

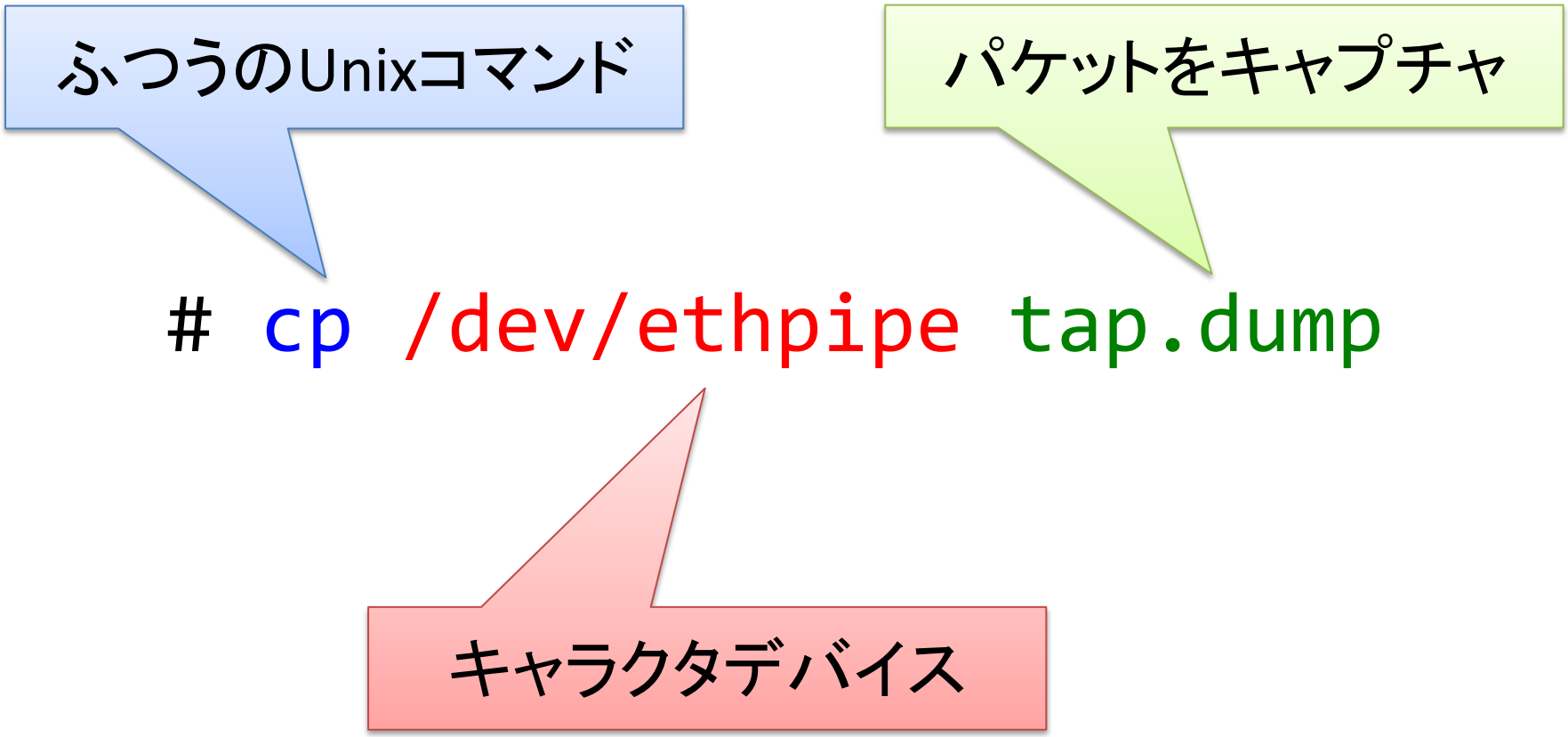
できたもの


Ethernet frame
Packet capture
Packet generate
/dev/ethpip
/dev/ethpip
OS
PCI Bus
FPGA
Port0
Port1
Ether frame
Ether frame

使い方

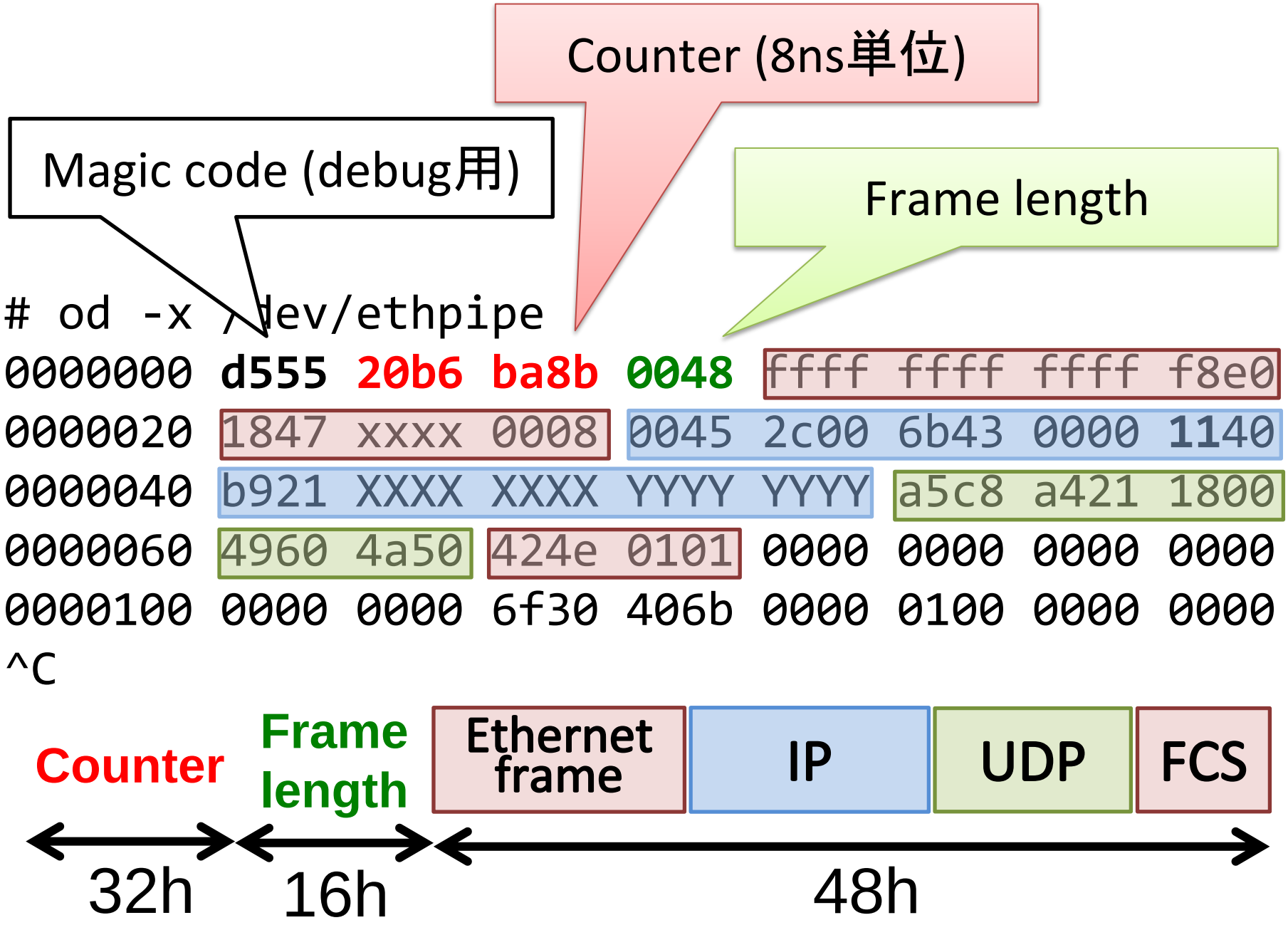
```
# cp /dev/ethpipe tap.dump
```

使い方

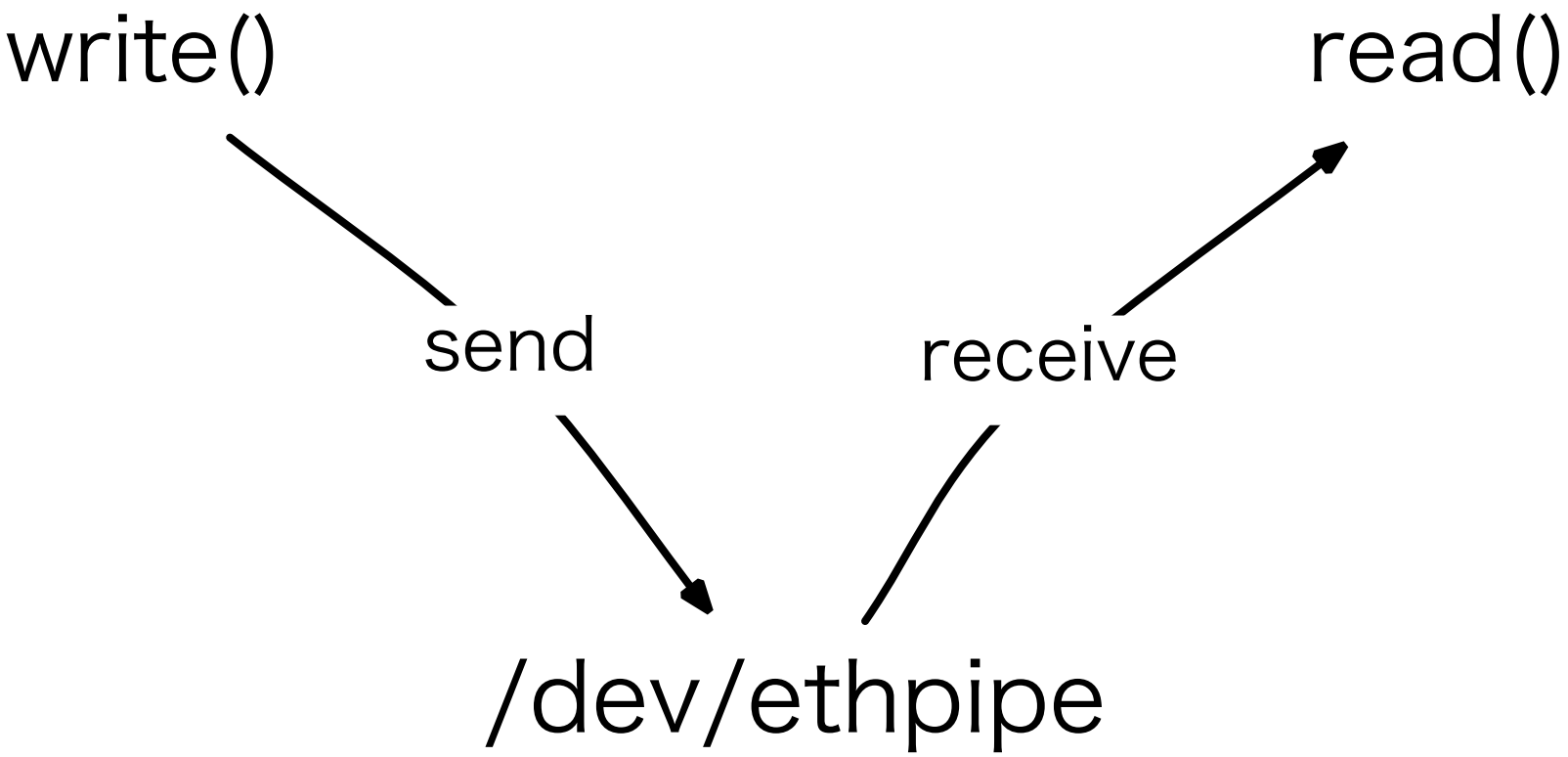

ふつうのUnixコマンド
パケットをキャプチャ
cp /dev/ethpipe tap.dump
キャラクタデバイス

フォーマットを見てみる

```
# od -x /dev/ethpipe
0000000 d555 20b6 ba8b 0048 ffff ffff ffff f8e0
0000020 1847 xxxx 0008 0045 2c00 6b43 0000 1140
0000040 b921 XXXX XXXX YYYY YYYY a5c8 a421 1800
0000060 4960 4a50 424e 0101 0000 0000 0000 0000
0000100 0000 0000 6f30 406b 0000 0100 0000 0000
^C
```


Counter (8ns単位)
Magic code (debug用)
Frame length
od -x /dev/ethpipe
0000000 d555 20b6 ba8b 0048 ffff ffff ffff f8e0
0000020 1847 xxxx 0008 0045 2c00 6b43 0000 1140
0000040 b921 XXXX XXXX YYYY YYYY a5c8 a421 1800
0000060 4960 4a50 424e 0101 0000 0000 0000 0000
0000100 0000 0000 6f30 406b 0000 0100 0000 0000
^C
Counter
Frame length
Ethernet frame
IP
UDP
FCS
32h
16h
48h

キャラクタデバイス型ネットワークIO



パケットのキャプチャと生成

Unixフレンドリなトラフィック操作

```
Packet capture
 # dd if=/dev/ethpipe of=pkt.dump

Packet generator
 # dd if=pkt.dump of=/dev/ethpipe
```

パケット解析

PcapNg

これで保存できれば，色々解析ツールで使える

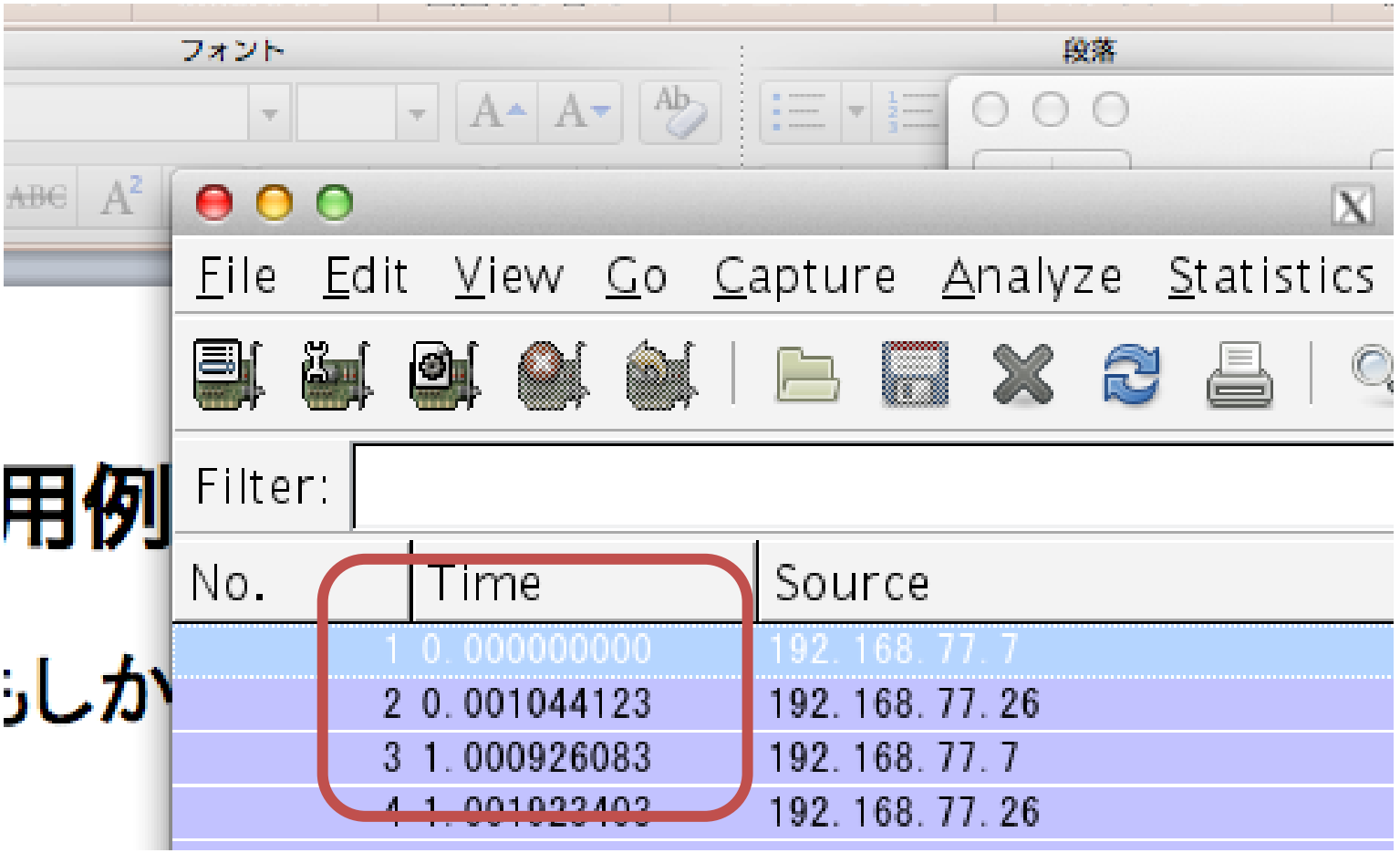
- Wireshark, libpcapなどでほぼ対応

Optionを使用 { Ethernet FCS, nsec timestamp }

```
Packet capture
   # ./ethdump < /dev/ethpipe > of=tap.ntar

Packet generator
   # ./ethgen < tap.ntar > /dev/ethpipe
```

Wireshark画面



http://wiki.wireshark.org/Development/PcapNg

応用例

- もしかして: 計測用途以外でも便利かも

- 別の視点からみたら
  - PacketIn/PacketOut ぽい
  - **パイプ**とコマンドでトラフィック操作できて超楽しい
  - ポートごとにデバイス化したらブリッジできる

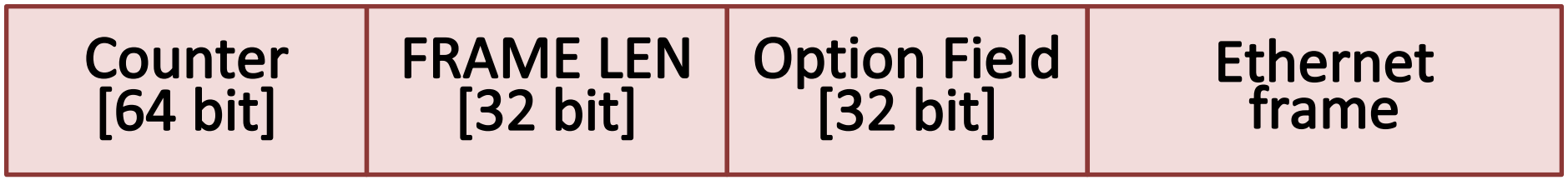
```
EX) ポート0からポート1にブリッジ
 # dd if=/dev/ethpipe/0 of=/dev/ethpipe/1
```

EthPIPE adapter (仮)

- PCI Expressの”**PIPE IF**”単純なNetwrok IO
- Unix“**パイプ**”によるトラフィックのFiltering操作

- Linuxフレンドリなデータプレーン実装
  - Ether frameをデバドラ経由で直接open/close
  - キャラクタデバイスによるread/write
  - 密なハードウェア連携

データフォーマット (案)



Counter[64bit]: IEEE1588v2互換

Option Field

- ユーザ指定可能な32bit固定のフィールド
  - ソフトウェアの一部処理をOffloading

- たとえば
  - 5-tupleハッシュ値
    - LB，IDS，解析ツールのフロー識別を補助
  - 任意のヘッダフィールドの処理+外出し
    - L2, L3, L4, L7の特定フィールド

コマンドラインSDN

DstMACの書き換え

```
dd if=/dev/ethpipe/0 | ethtr "xx:xx:xx:xx:xx"
  "yy:yy:yy:yy:yy" | dd of=/dev/ethpipe/1
```

Command line filtering

```
dd if=/dev/ethpipe/0 | ethgrep --multicast | dd
  of=/dev/ethpipe/1
```

※ もしかして: MTUとかCheck sumがおかしくなる
⇛ そこはHWが得意なので，HW側がよしなに対応予定

こんなことも

sshオーバーレイトンネル

(IPマルチキャストパケットだけ転送とか)

```
# dd if=/dev/ethpipe/0 |ssh
haeena.net dd of=/dev/ethpipe/0
```

Ethernet frame over SMTP　　# DTN的なサムシング

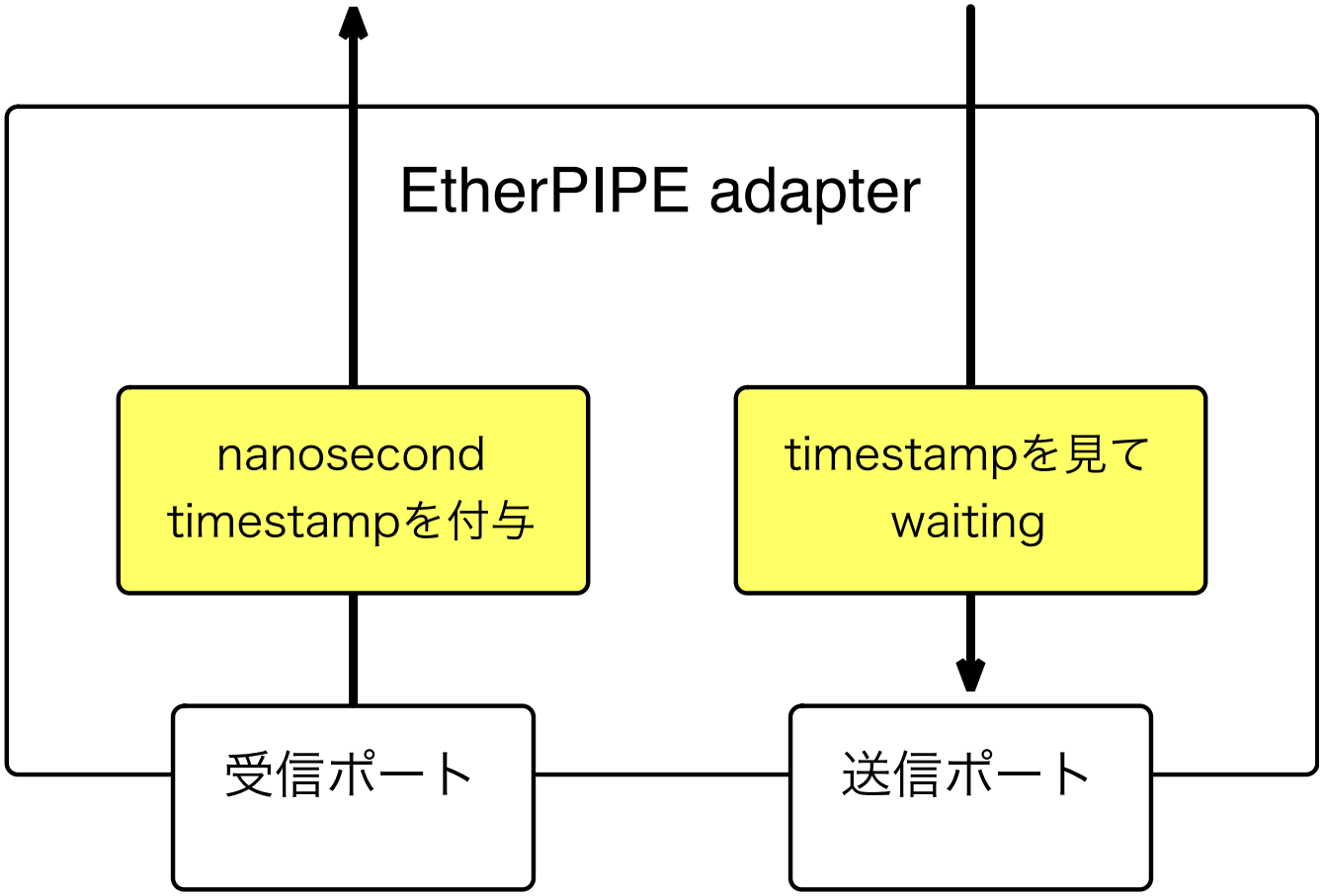
```
# dd if=/dev/ethpipe/0 bs=1MB | sendmail
```

その他の機能: 送受信タイミング調整

- 受信: 各フレームのハードウェアカウンタを64 bitに拡張
  - Intel i350/82580などで可能
- 送信: 受信フレーム到達タイミングを再生
  - Linux tc用途を想定

- パケット受信タイミングの再現や遅延エミュレーションに利用

その他の機能: 送受信タイミング調整



まとめ

- Linuxフレンドリなデータプレーンの設計
  - コマンド1行でトラフィックを色々いじって遊べます
  - 使ってて楽しい

- 今後の予定
  - ポートステータス，テスタ機能のマージ
  - HW Ingress filter + ポート間直接Forwarding
    - Application switching
    - Software switchのHW D-plane

さいごに

- 開発FPGAボード
  - Lattice ECP3 versa kit
    - ~~$99~~ -> $299, PCIe1.1 x1, 1000BASE-T x2
  - NetFPGA-1G (サポート予定)
    - $699 (academic), PCI-X, 1000BASE-T x4
- Repository
  - テスタ回路 (あとで一部マージ)
    - https://github.com/Murailab-arch/magukara
  - EtherPIPE adapter
- HW側のスライド: http://www.slideshare.net/ykuga/ovshw

補助スライド

IO memory address: global status

Address	Size (Byte)	Permission	Port	
00	1	rw		Slot status
04-07	4	rw		Global counter
08-0B	4	rw		Global counter

ポートごとのPPS, Throughtput, etcを導入
ケーブルテスタ，ネットワークテスタ機能 (see RFC2544)

IO memory address: Frame slots

Address	Size (Byte)	Permission	Port	
4000-5FFF	8K	r	RX0	Recv frame
6000-7FFF	8K	w	TX0	Send frame
8000-9FFF	8K	r	RX1	Recv frame
A000-BFFF	8K	w	TX1	Send frame

※ 現在はスロット1つでやってます