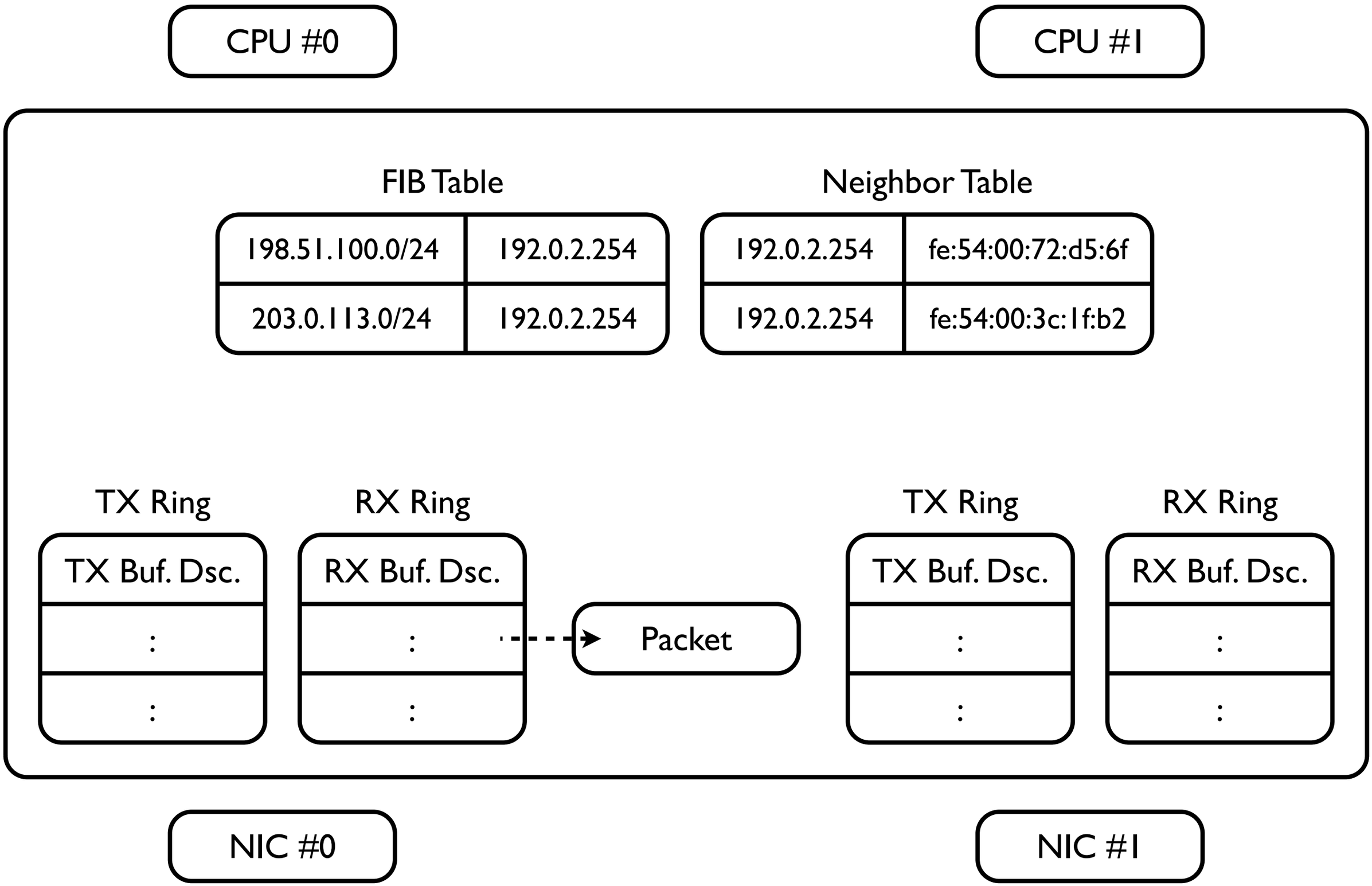
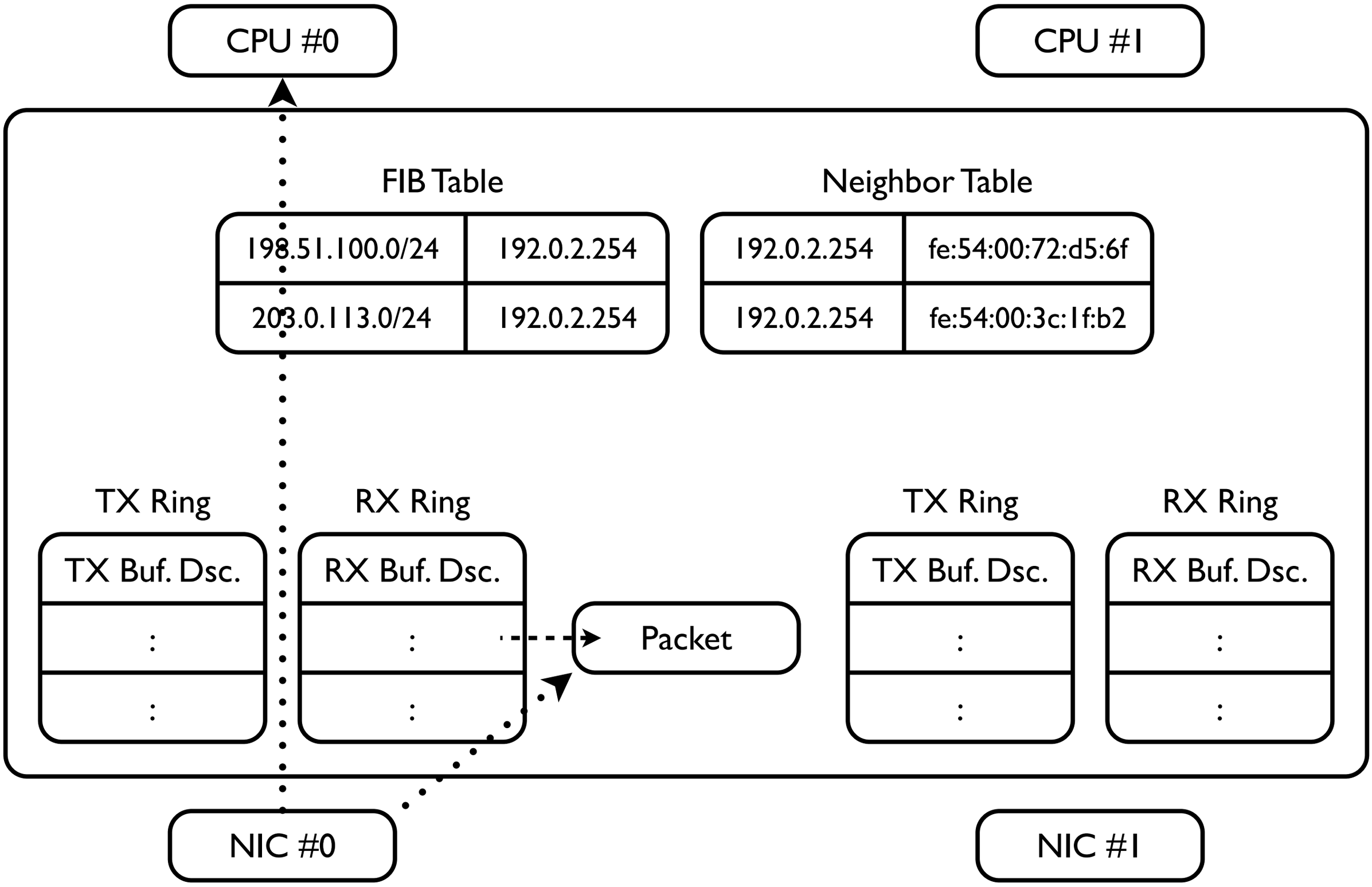
Linux システムにおける
パケットフォワーディング概要

浅間 正和 @ 有限会社 銀座堂

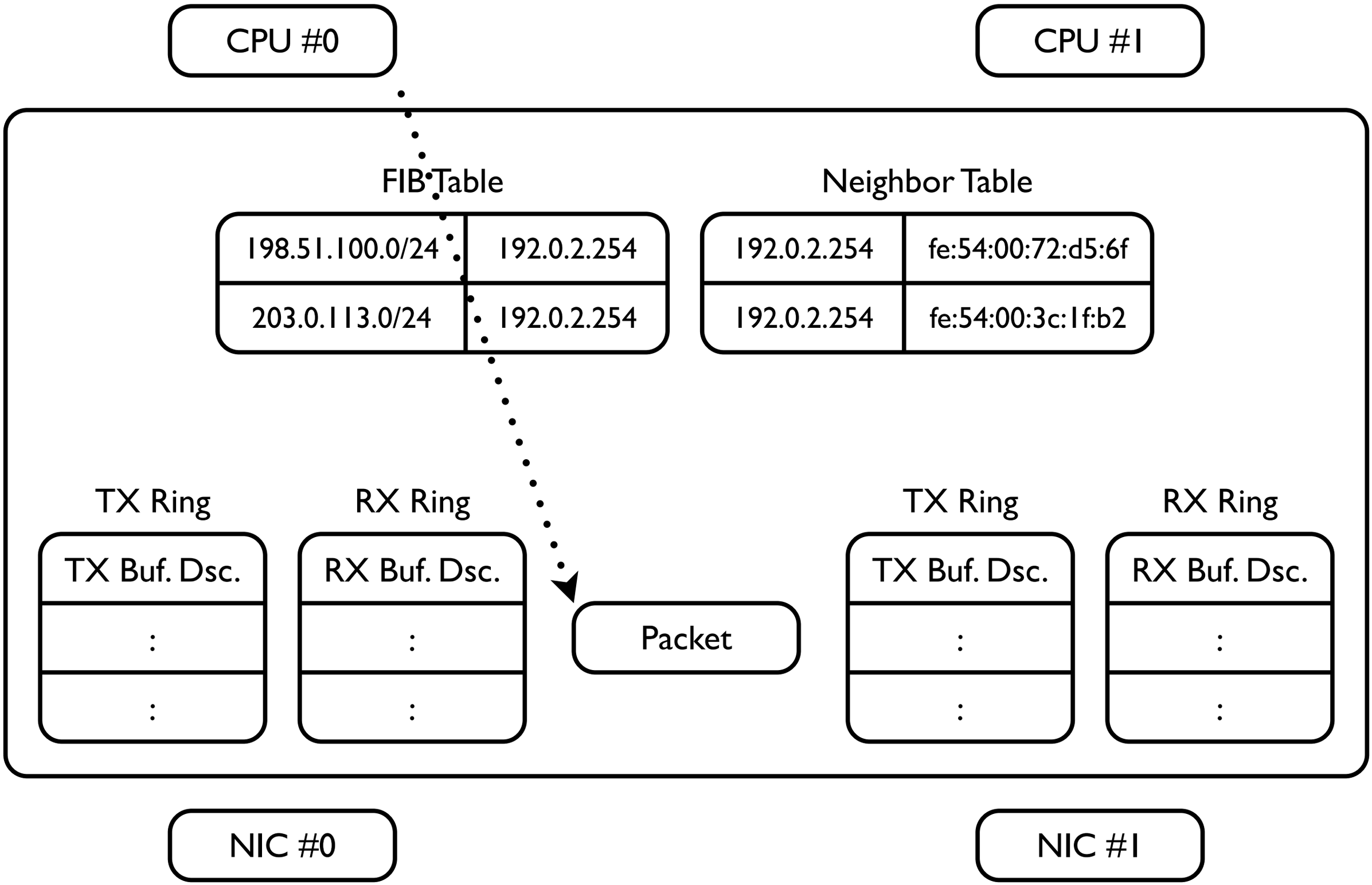
おおまかなフォワーディングの流れ



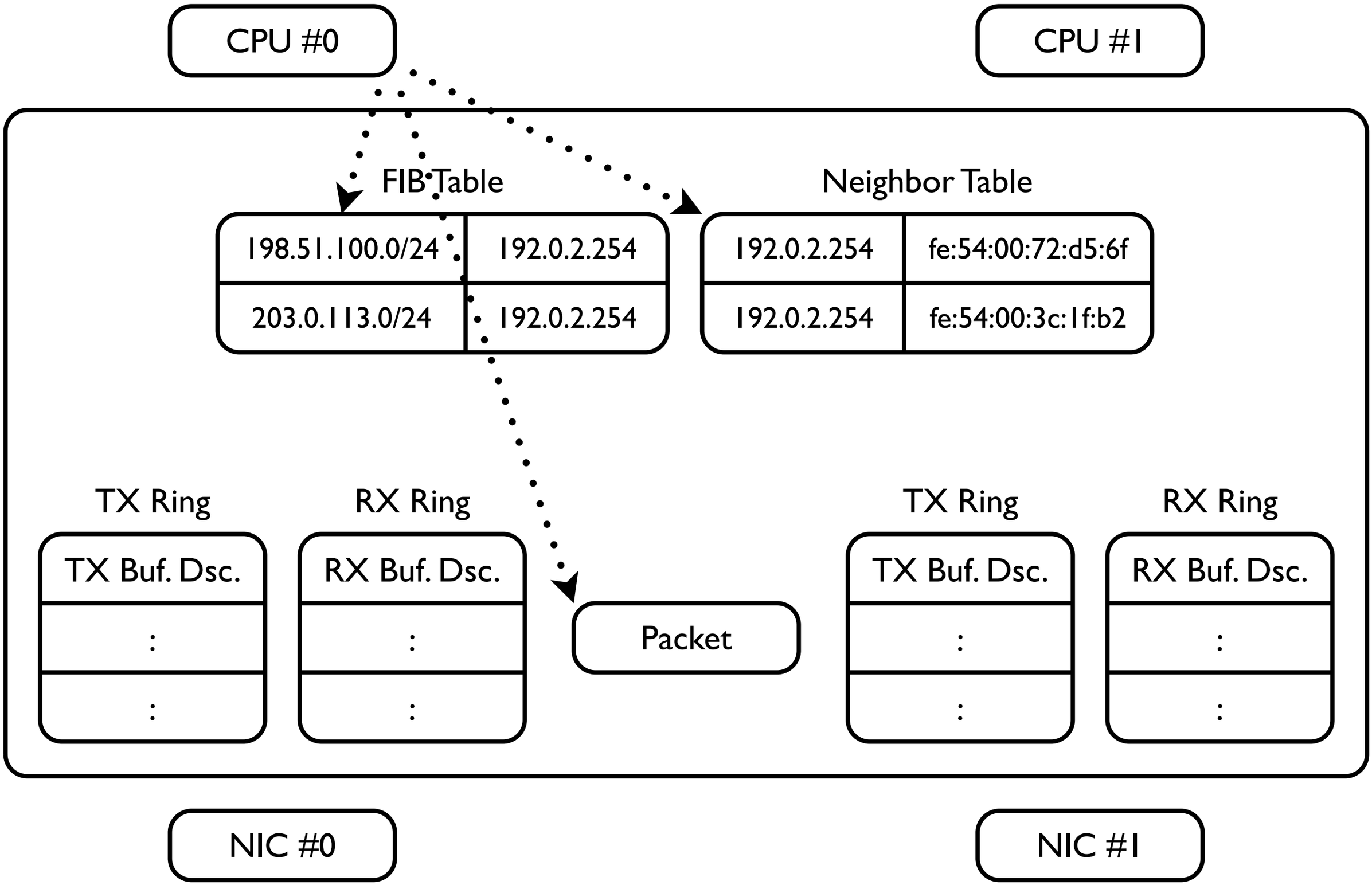
受信処理（１）


CPU #0
CPU #1
FIB Table
198.51.100.0/24
192.0.2.254
203.0.113.0/24
192.0.2.254
Neighbor Table
192.0.2.254
fe:54:00:72:d5:6f
192.0.2.254
fe:54:00:3c:1f:b2
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
Packet
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
NIC #0
NIC #1

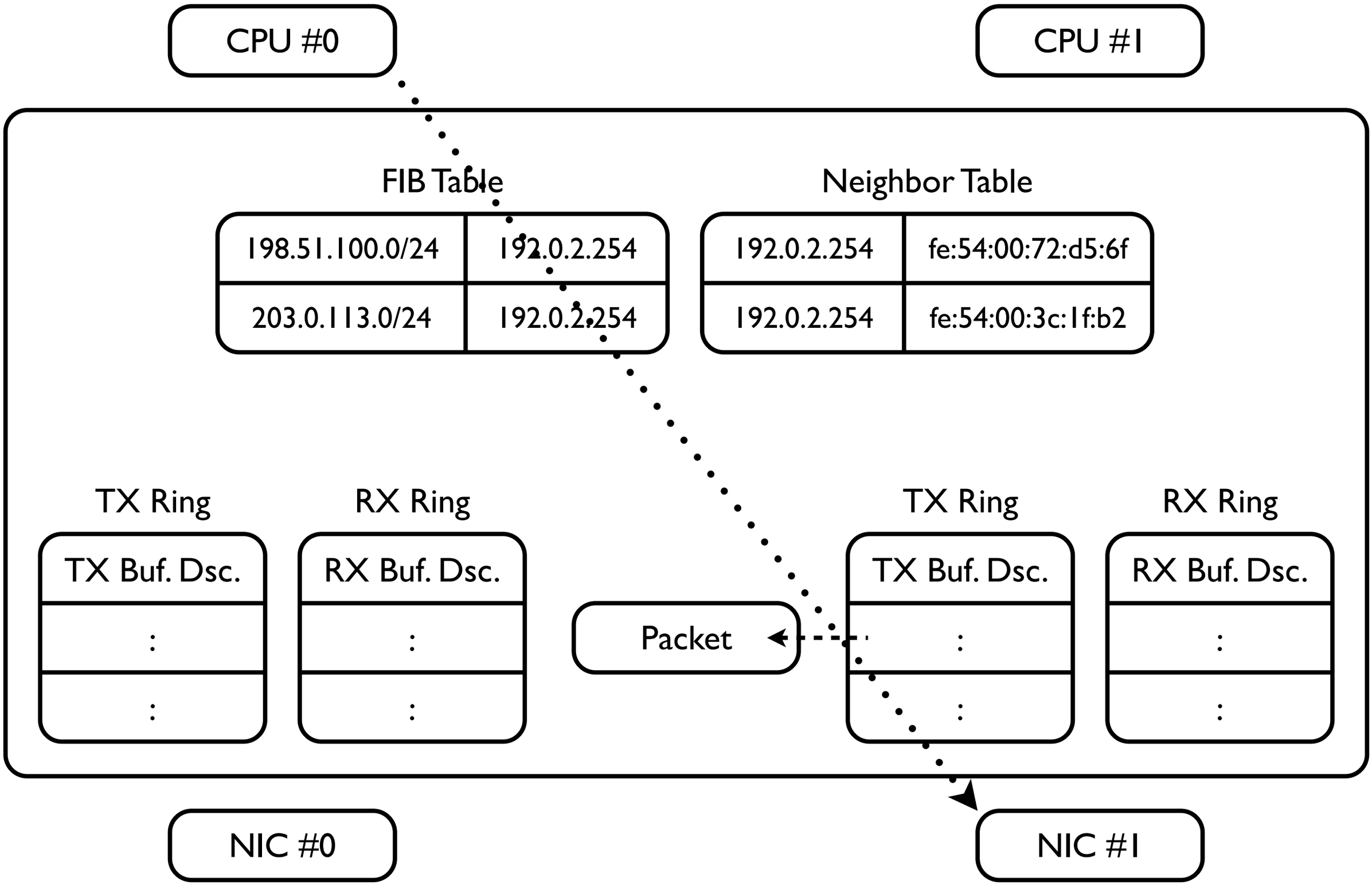
受信処理（2）


CPU #0
CPU #1
FIB Table
198.51.100.0/24
192.0.2.254
203.0.113.0/24
192.0.2.254
Neighbor Table
192.0.2.254
fe:54:00:72:d5:6f
192.0.2.254
fe:54:00:3c:1f:b2
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
Packet
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
NIC #0
NIC #1

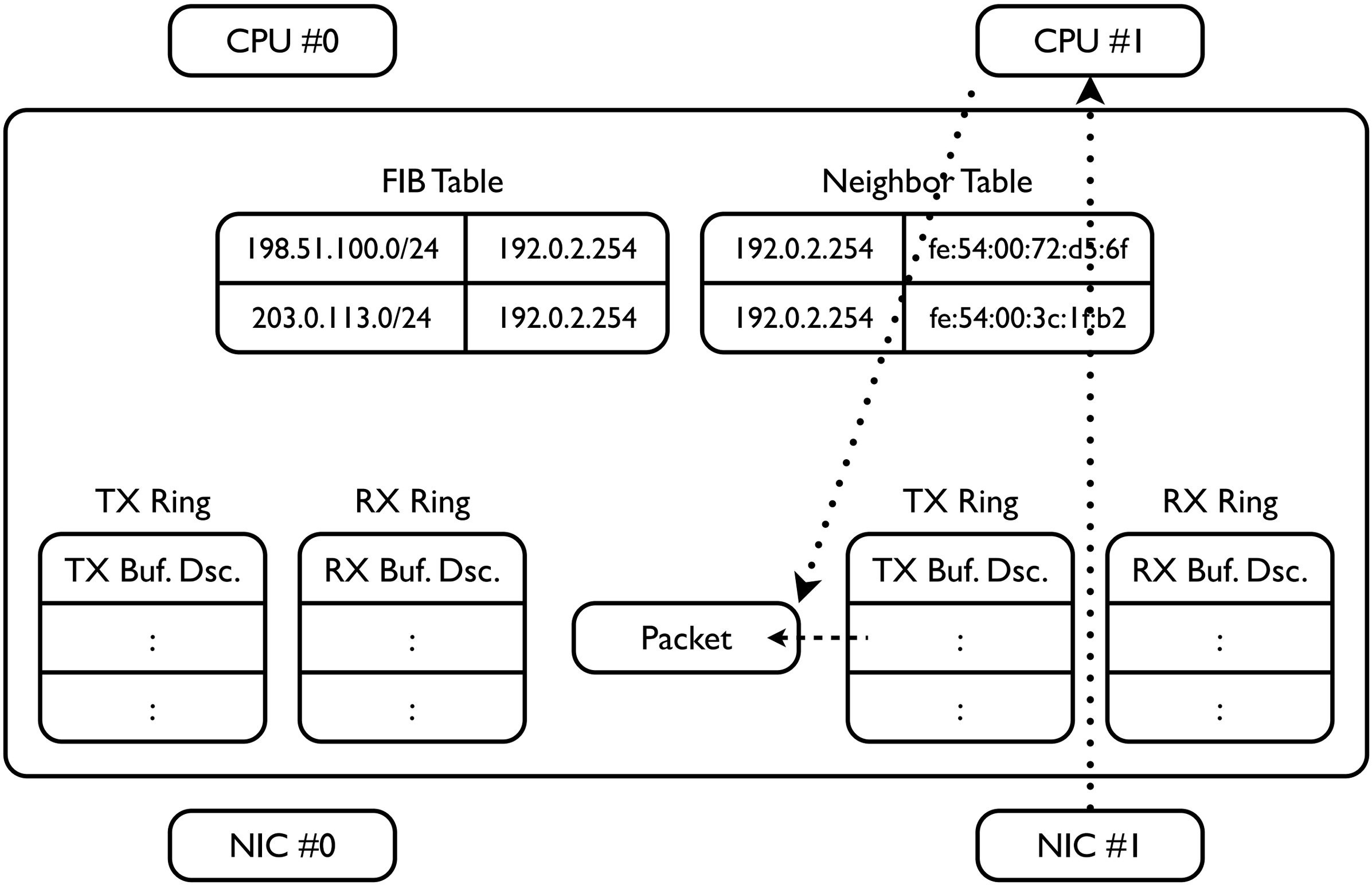
FIB Table と Neighbor Table の探索



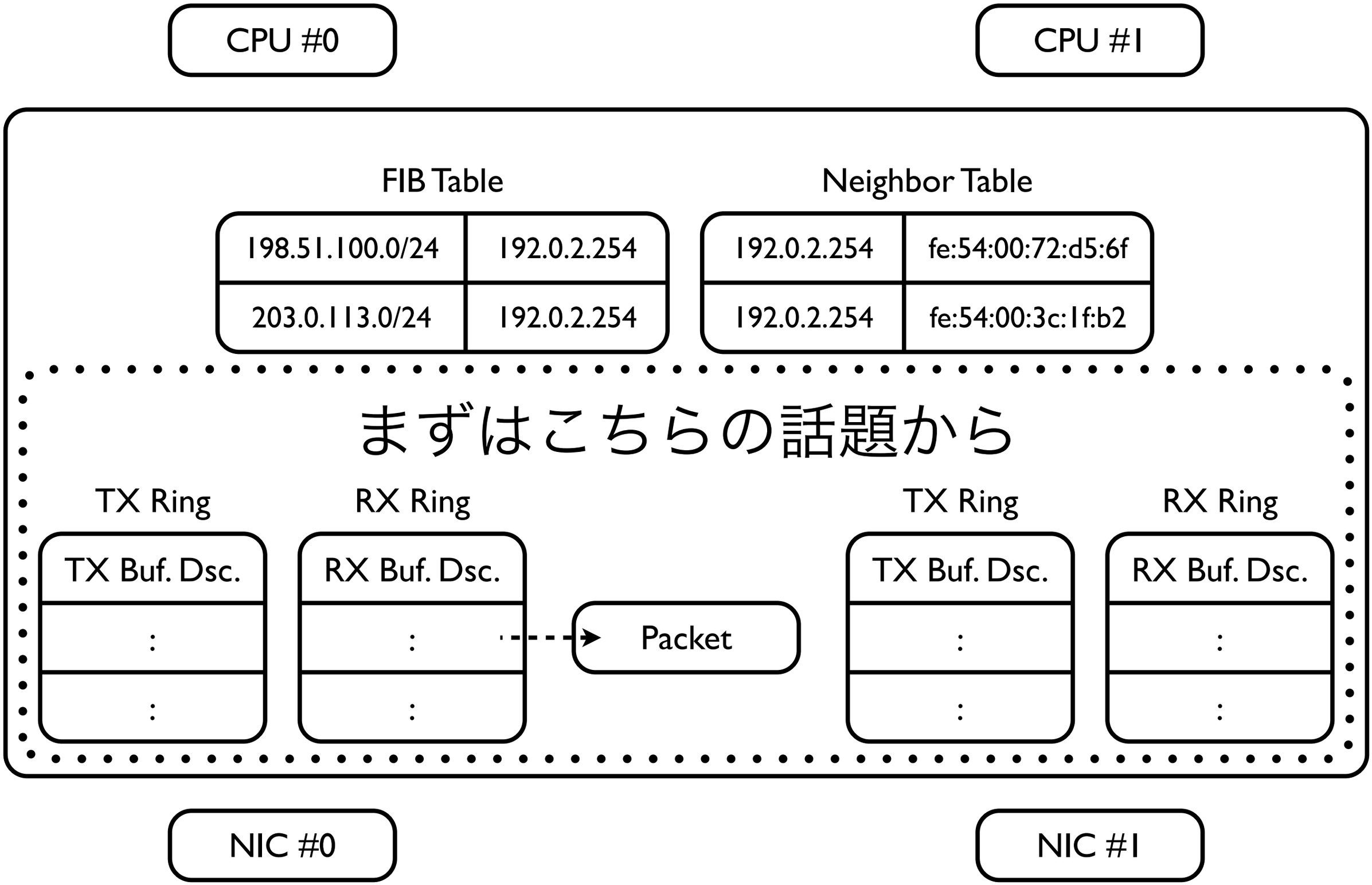
送信処理（１）


CPU #0
CPU #1
FIB Table
198.51.100.0/24
192.0.2.254
203.0.113.0/24
192.0.2.254
Neighbor Table
192.0.2.254
fe:54:00:72:d5:6f
192.0.2.254
fe:54:00:3c:1f:b2
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
Packet
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
NIC #0
NIC #1

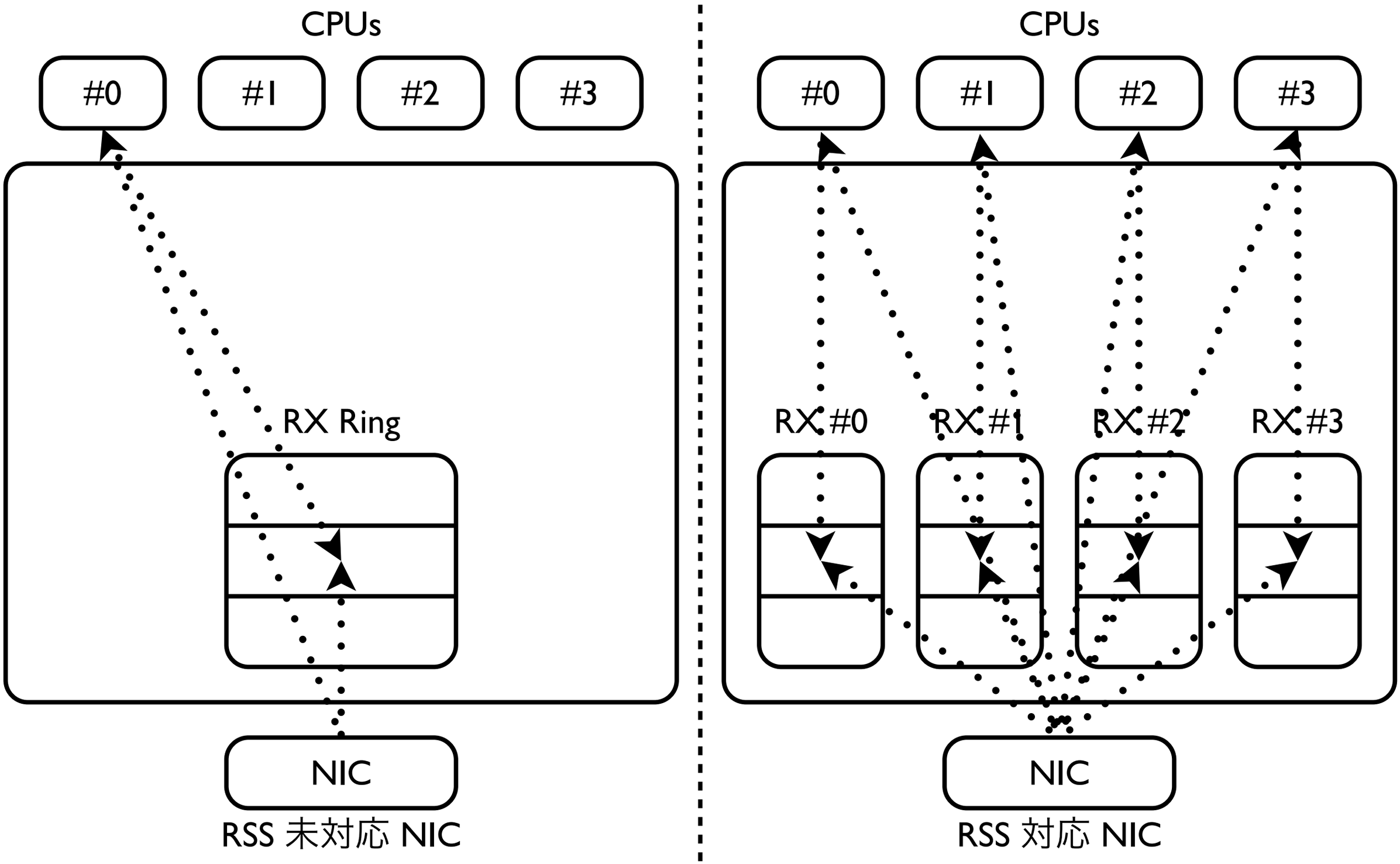
送信処理（2）


CPU #0
CPU #1
FIB Table
198.51.100.0/24
192.0.2.254
203.0.113.0/24
192.0.2.254
Neighbor Table
192.0.2.254
fe:54:00:72:d5:6f
192.0.2.254
fe:54:00:3c:1f:b2
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
Packet
TX Ring
TX Buf. Dsc.
RX Ring
RX Buf. Dsc.
NIC #0
NIC #1

CPU Multi-Core と転送性能



Receive Side Scaling: H/W による分散処理



Ethernet Controller の対応が必要

Product Brief
Intel® 82599 10 Gigabit Ethernet Controller
Network Connectivity

intel

Intel® 82599 10 Gigabit Ethernet Controller
Transforming the data center through a reliable and unified 10GbE network

BROADCOM
Connecting everything®

BCM57710
PRODUCT Brief

10-Gbps DUAL-PORT TCP, RDMA, iSCSI CONTROLLER WITH x8 LANE PCI EXPRESS®

FEATURES

SUMMARY OF BENEFITS

Low latency interrupts

Receive Side Scaling for Windows environments and Scalable I/O for Linux* environments (IPv4, IPv6, TCP/UDP)

Intel® Ethernet Flow Director
Supports advanced filters that dir
their flows to different queues

An Intelligent Solution

A number of trends are driving change in the data center. Information growth continues unabated, server virtualization requires multi-core servers with higher bandwidth and networked storage, and power and cooling remains a significant challenge. These trends are driving IT managers to look for new, optimized solutions that support the dynamic data center. From a network perspective, IT managers recognize how the trend towards 10 Gigabit Ethernet (10GbE) will benefit them; they want to consolidate multiple GbE links into a single 10GbE network to lower cost and complexity.

Intel's third-generation 10GbE controller, the Intel® 82599 10 Gigabit Ethernet controller continues to build on the innovative trends set by its predecessor and pushes the envelope even further.

- **Simultaneous operation of Ethernet, TCP/IP, iSCSI, and RDMA m**
- **Other performance features**
  - Receive side scaling (RSS) for IPv4 and IPv6
  - Teaming for L2, L4, and L5
  - Giant send offload (GSO) support
  - Jumbo frame support (9600 bytes)
  - TCP, IPv4, IPv6 checksum offload
  - TCP segmentation offload

割り込みと受信キューの対応付け（手動）

- 受信キューと割り込み番号の対応の確認

```
[root@linux ~]# cat /proc/interrupts
           CPU0       CPU1
...
  52:          0          0   PCI-MSI-edge      p1p1-TxRx-0
  53:          0          0   PCI-MSI-edge      p1p1-TxRx-1
  54:          0          0   PCI-MSI-edge      p1p1
  55:          0          0   PCI-MSI-edge      p1p2-TxRx-0
  56:          0          0   PCI-MSI-edge      p1p2-TxRx-1
  57:          0          0   PCI-MSI-edge      p1p2
...
```

- 受信キューと割り込み先 CPU の対応付け

```
[root@linux ~]# echo 1 > /proc/irq/52/smp_affinity
[root@linux ~]# echo 2 > /proc/irq/53/smp_affinity
[root@linux ~]# echo 1 > /proc/irq/55/smp_affinity
[root@linux ~]# echo 2 > /proc/irq/56/smp_affinity
```

参考） CPU0=0x1, CPU1=0x2, CPU2=0x4, CPU3=0x8, CPU4=0x10, ...

割り込みと受信キューの対応付け（自動）

- CPU の個数等の情報から自動的に IRQ の割り込み先を設定してくれる irqbalance というコマンドがある
- デーモンとして起動する場合は以下のように実行

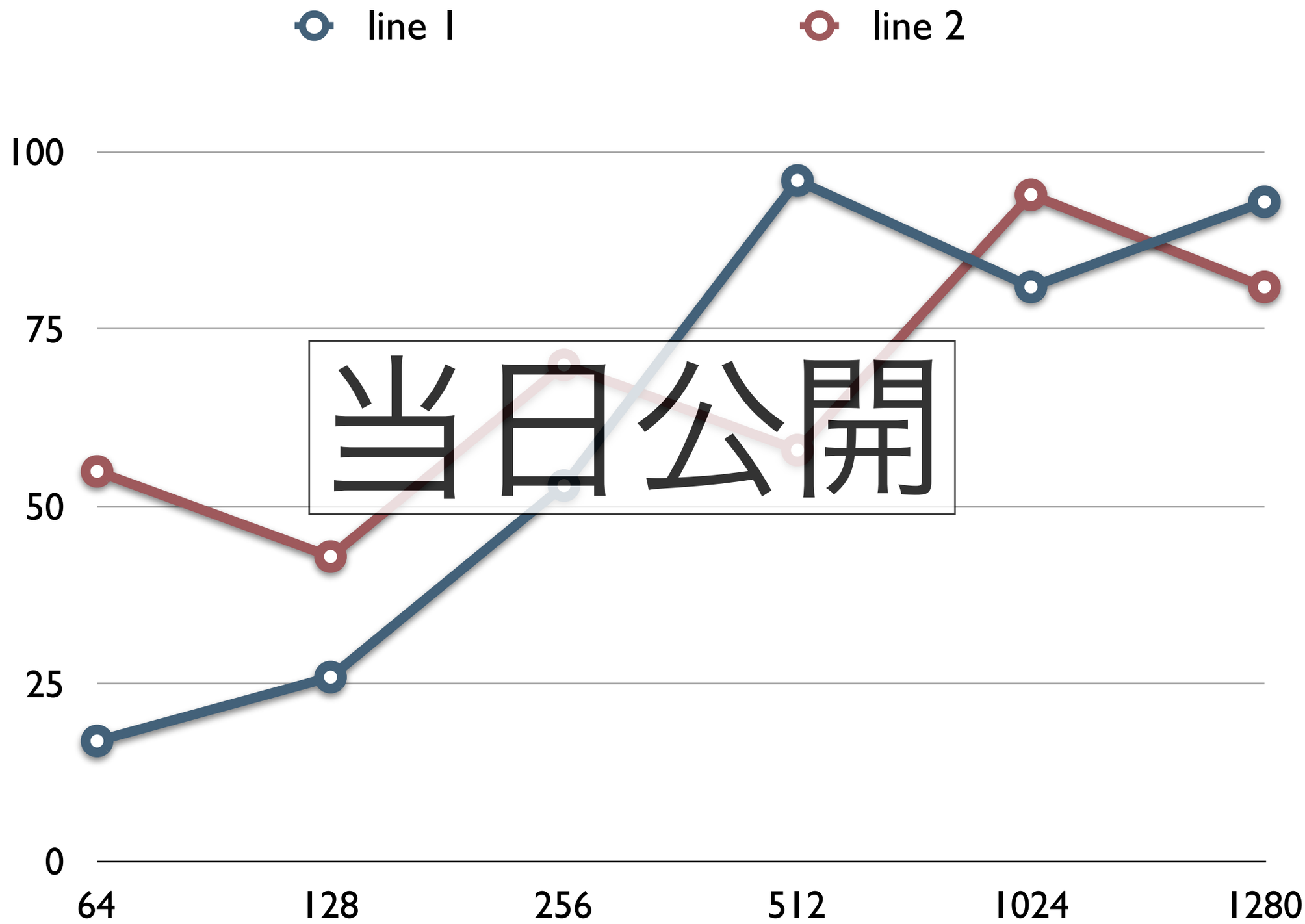
```
[root@linux ~]# irqbalance
```

- 一度だけ設定しデーモンとして起動させたくない場合は以下のように実行

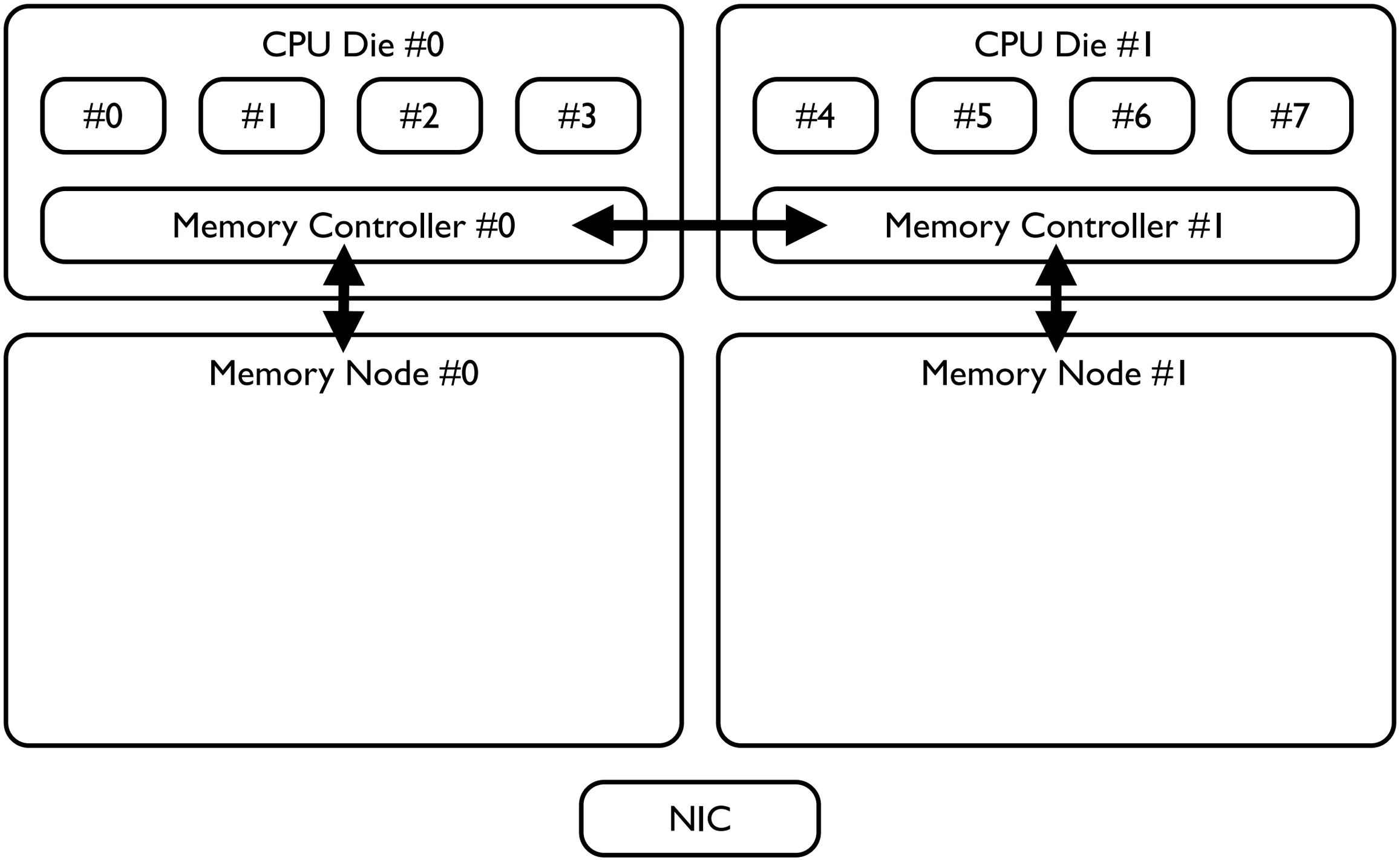
```
[root@linux ~]# irqbalance --oneshot
```

参考）ディストリビューションによっては再起動時に自動的に irqbalance を起動するための起動スクリプトが用意されている場合があります

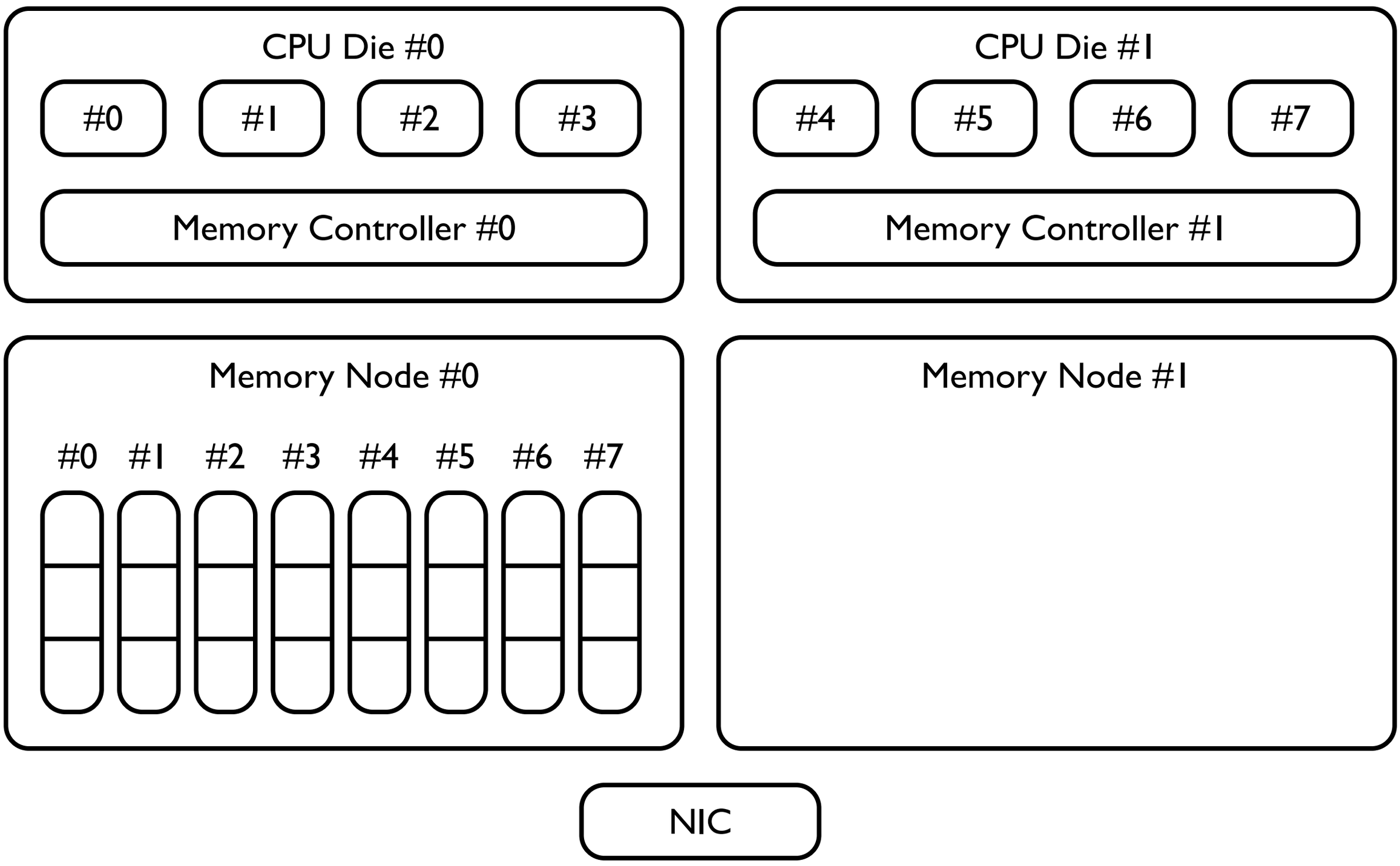
RSS による性能改善の評価



Non-Uniform Memory Access: メモリの距離



Non-Uniform Memory Access: メモリの距離



CPU と Node の対応の確認方法

```
[root@linux ~]# numactl --hardware
available: 2 nodes (0-1)
node 0 cpus: 0 2 4 6
node 0 size: 16384 MB
node 0 free: 15900 MB
node 1 cpus: 1 3 5 7
node 1 size: 16374 MB
node 1 free: 15858 MB
node distances:
node   0   1
  0:  10  20
  1:  20  10

[root@linux ~]# numactl --show
policy: default
preferred node: current
physcpubind: 0 1 2 3 4 5 6 7
cpubind: 0 1
nodebind: 0 1
membind: 0 1
```

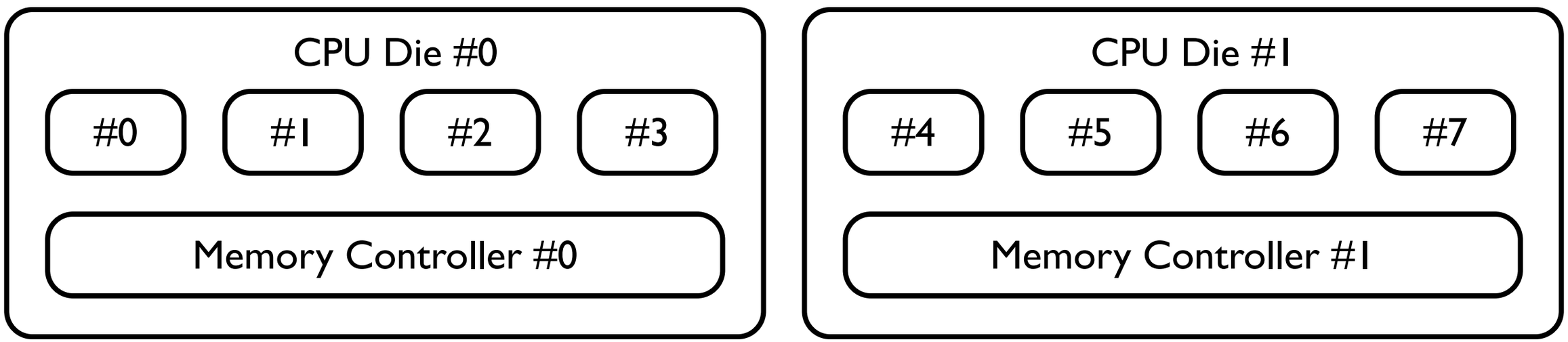
この PC が持つ Node の数

Node 0 にある CPU

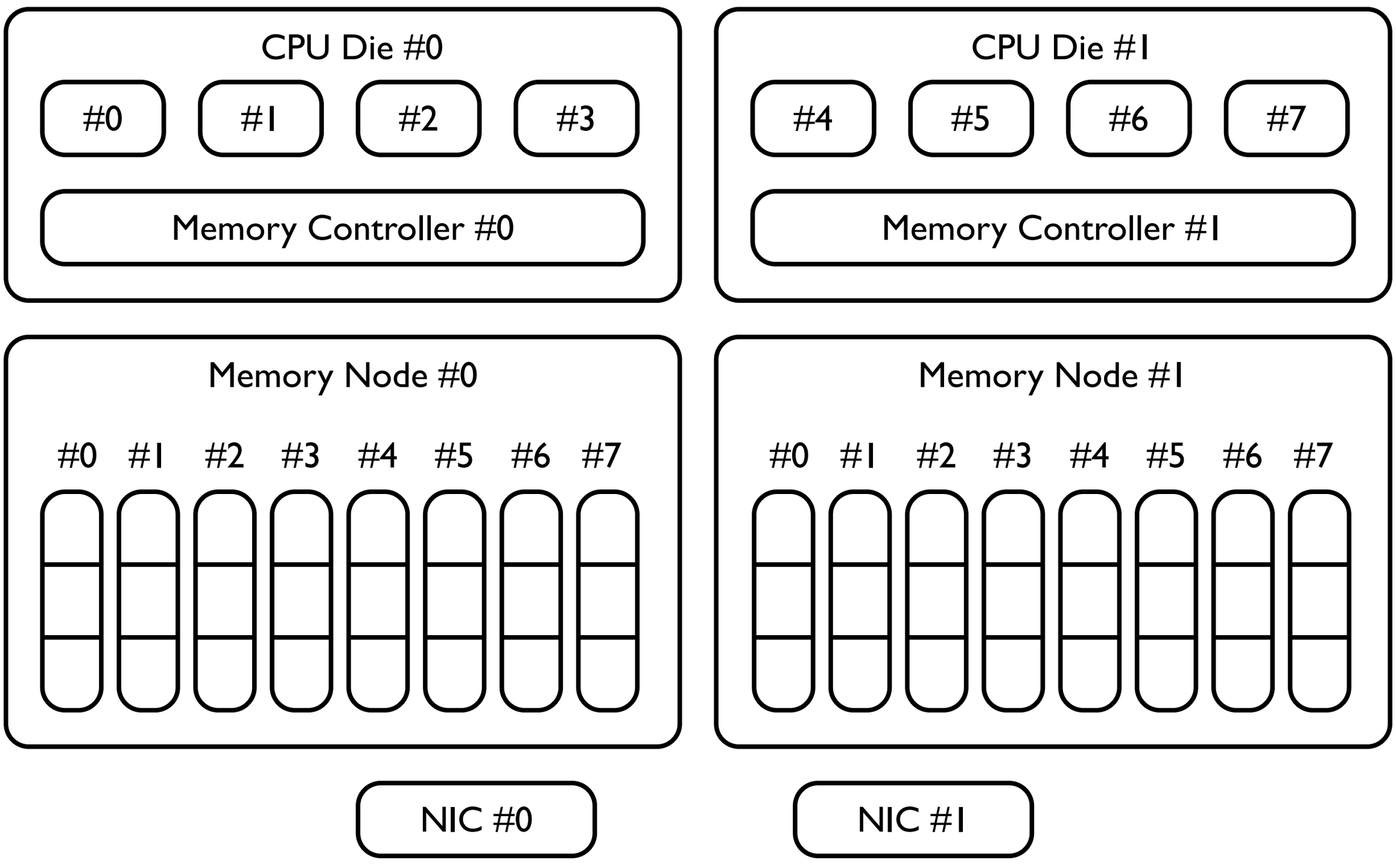
Node 1 にある CPU

※ 詳細は numactl のマニュアルを確認してください

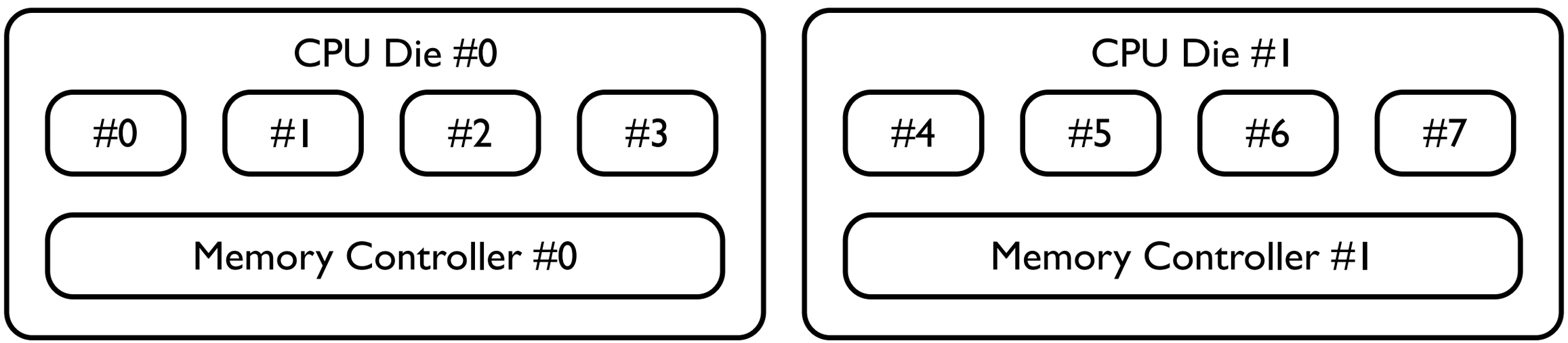
modprobe ixgbe RSS=8,8 Node=0,0



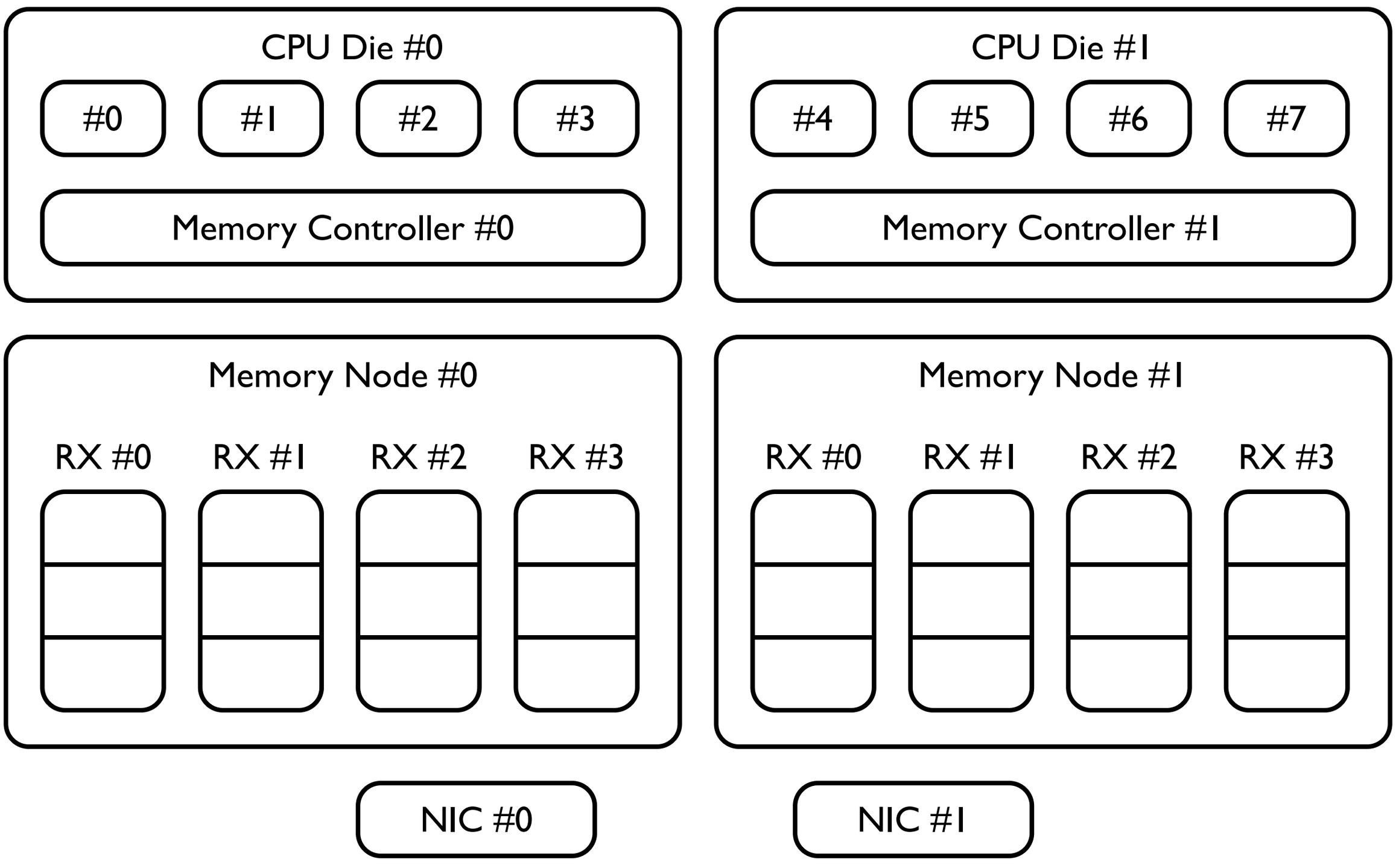
modprobe ixgbe RSS=8,8 Node=0,1



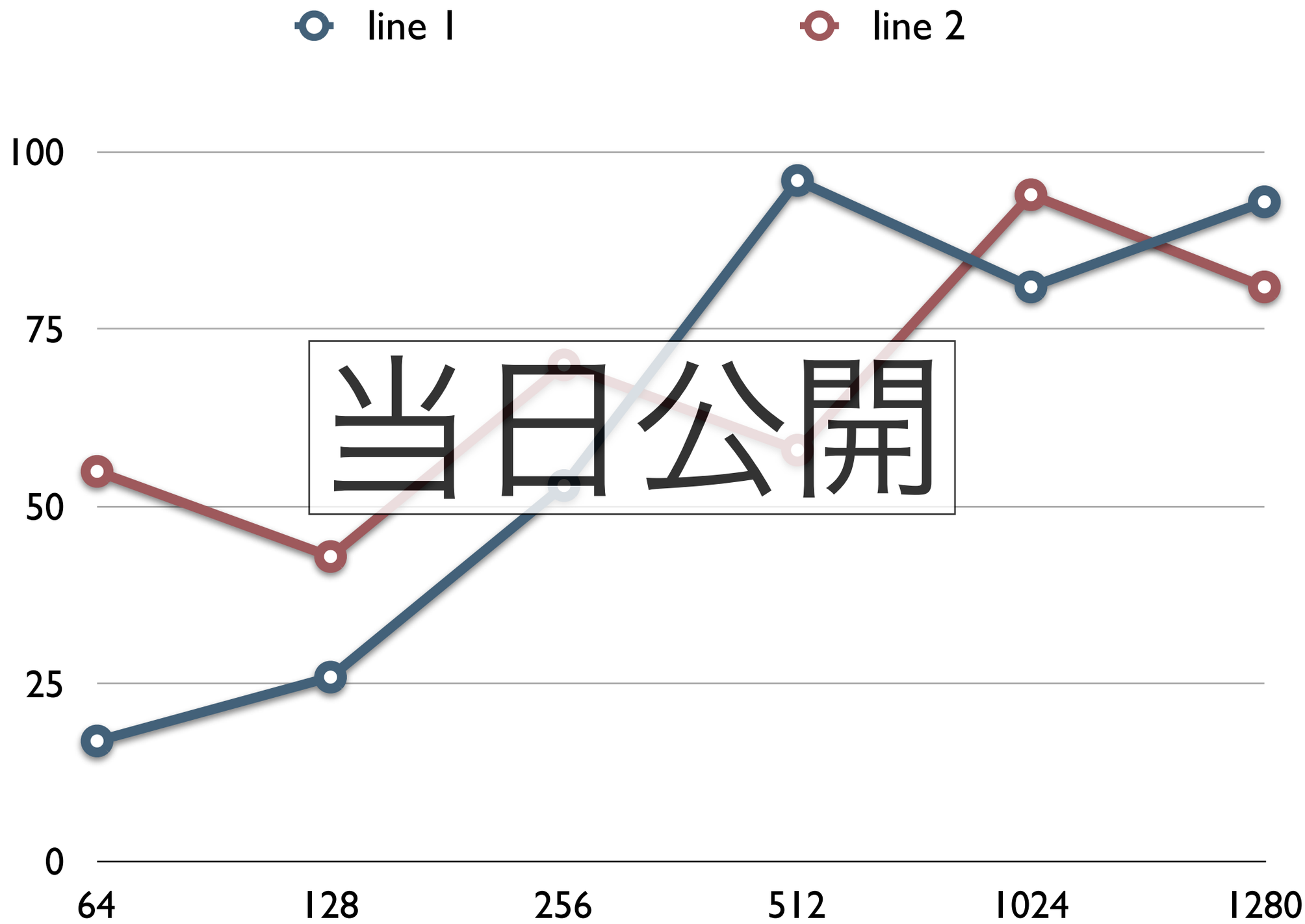
modprobe ixgbe RSS=4,4 Node=0,0



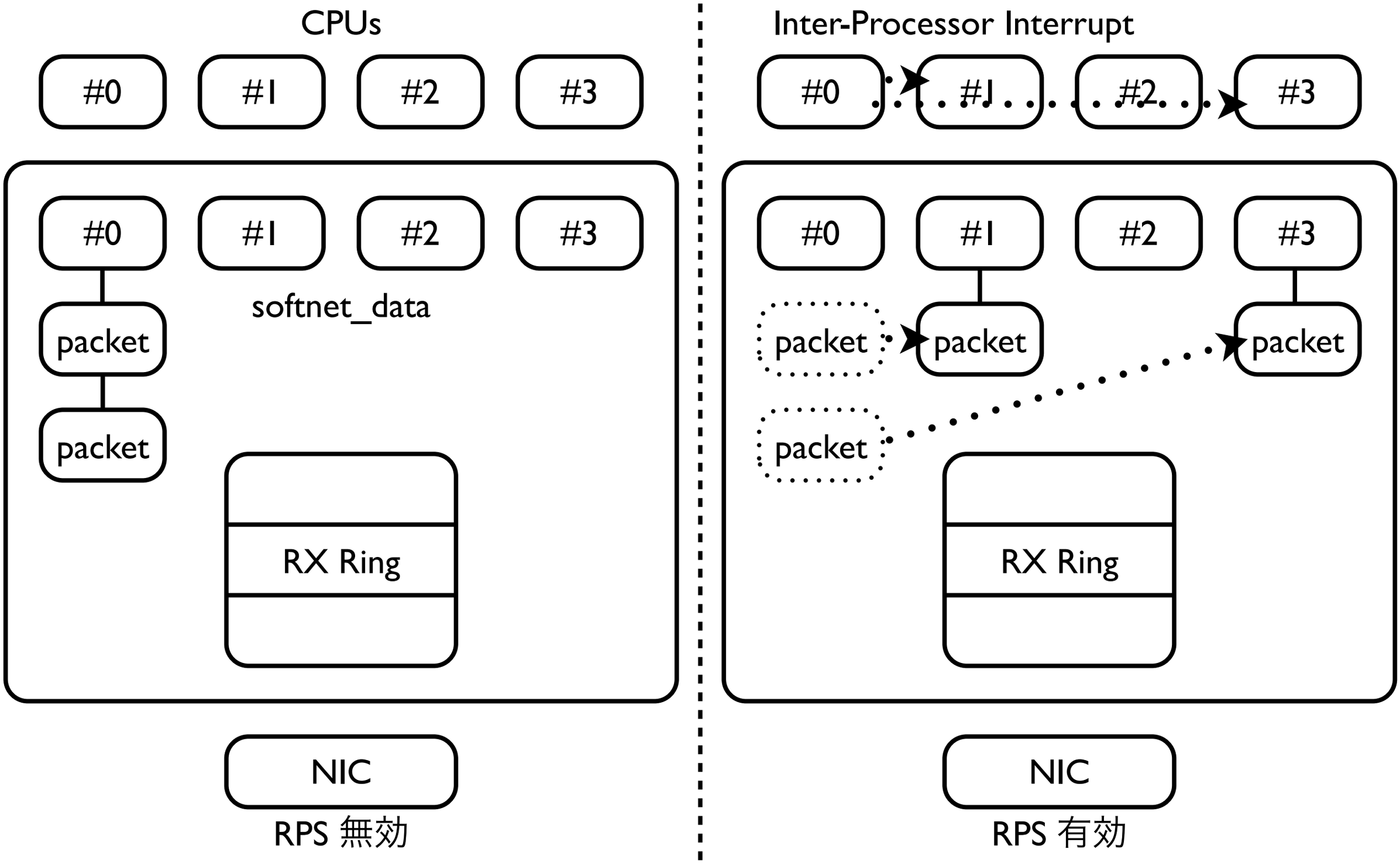
modprobe ixgbe RSS=4,4 Node=0,1



Node の差異による性能差



Receive Packet Steering: S/W による分散処理



受信キューと CPU の対応付け

- 設定したい NIC の sysfs ファイルに CPU マスクを書き込む

```
# echo f > /sys/class/net/p1p1/queues/rx-0/rps_cpus
```

NIC の受信キュー番号
通常は 0

設定したい NIC の名前

割り当てたい CPU のマスク

例）CPU0-3 があって CPU1-3 に割り当てたい場合

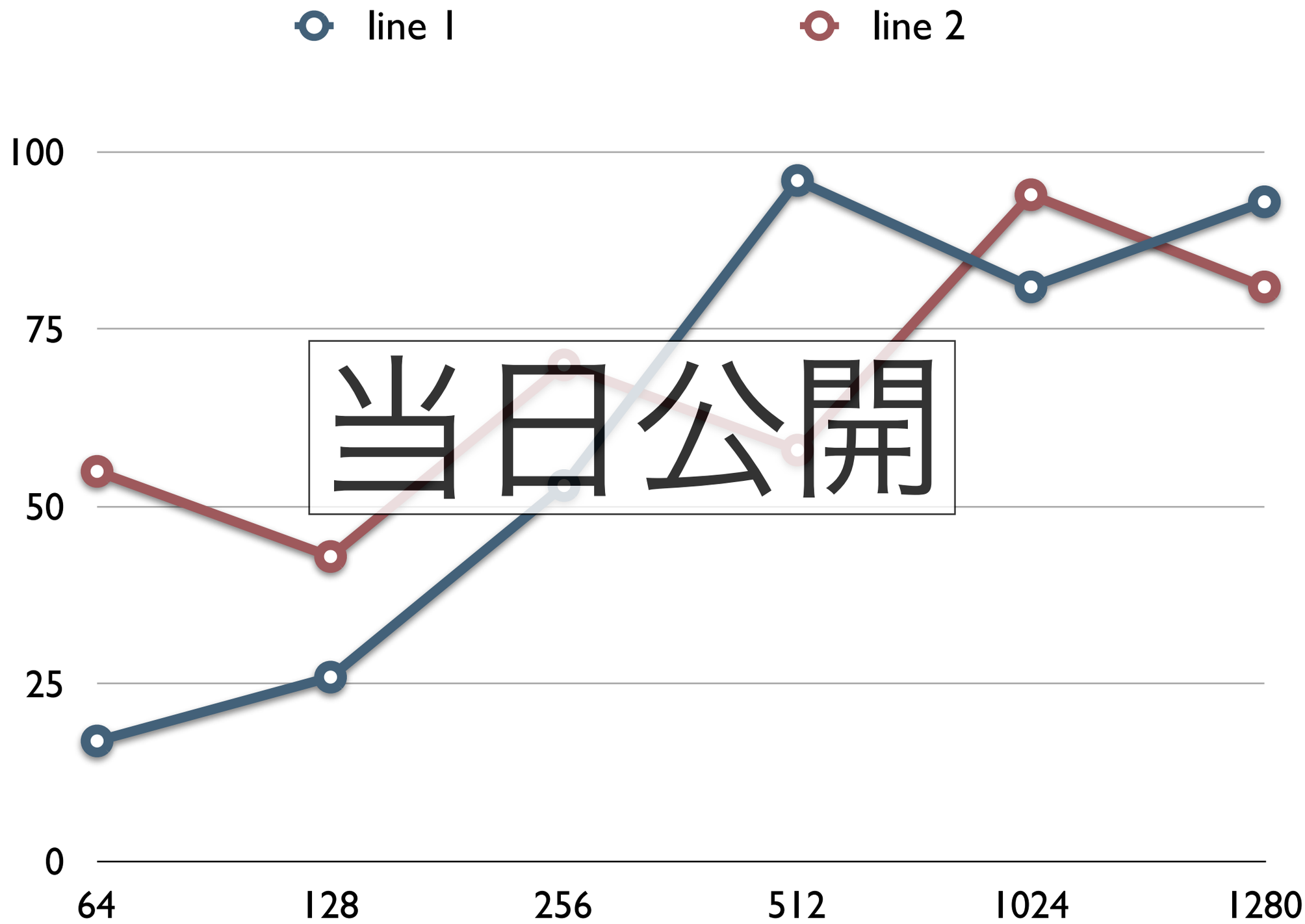
0x2(CPU1) + 0x4(CPU2) + 0x8(CPU03) = 0xe

なので e を書き込む

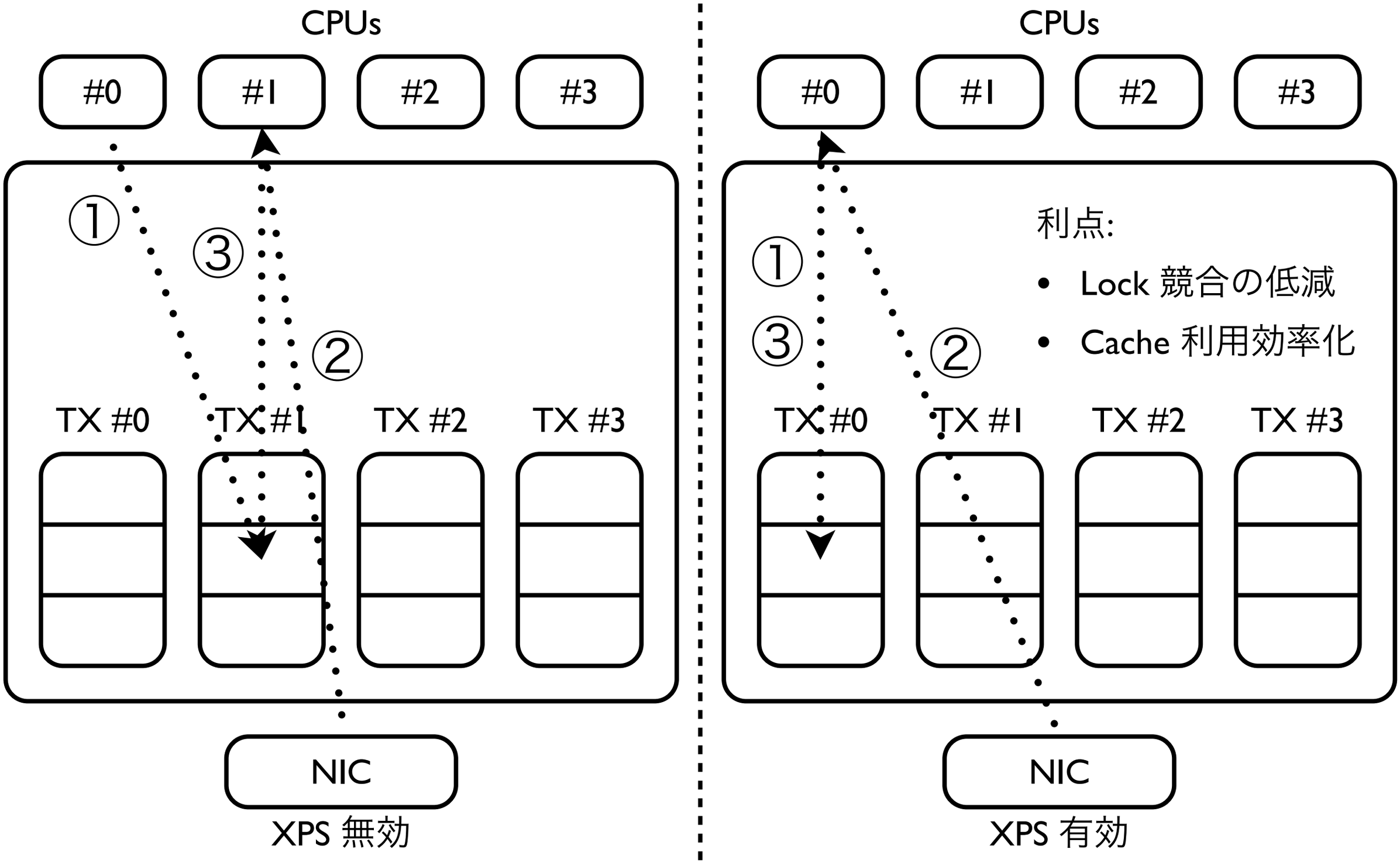
上記の例では f を書き込んでいるので CPU0-3 に
割り当たるようになる

※ Linux Kernel 2.6.35 以降でコンパイル時に CONFIG_RPS=y が設定されていること

RPS による性能改善の評価



Transmit Packet Steering



CPU と送信キューの対応付け

- 設定したい NIC の sysfs ファイルに CPU マスクを書き込む

```
# echo 1 > /sys/class/net/p1p1/queues/tx-0/xps_cpus
# echo 2 > /sys/class/net/p1p1/queues/tx-1/xps_cpus
# echo 4 > /sys/class/net/p1p1/queues/tx-2/xps_cpus
# echo 8 > /sys/class/net/p1p1/queues/tx-3/xps_cpus
```

NIC の送信キュー番号

設定したい NIC の名前

割り当てたい CPU のマスク

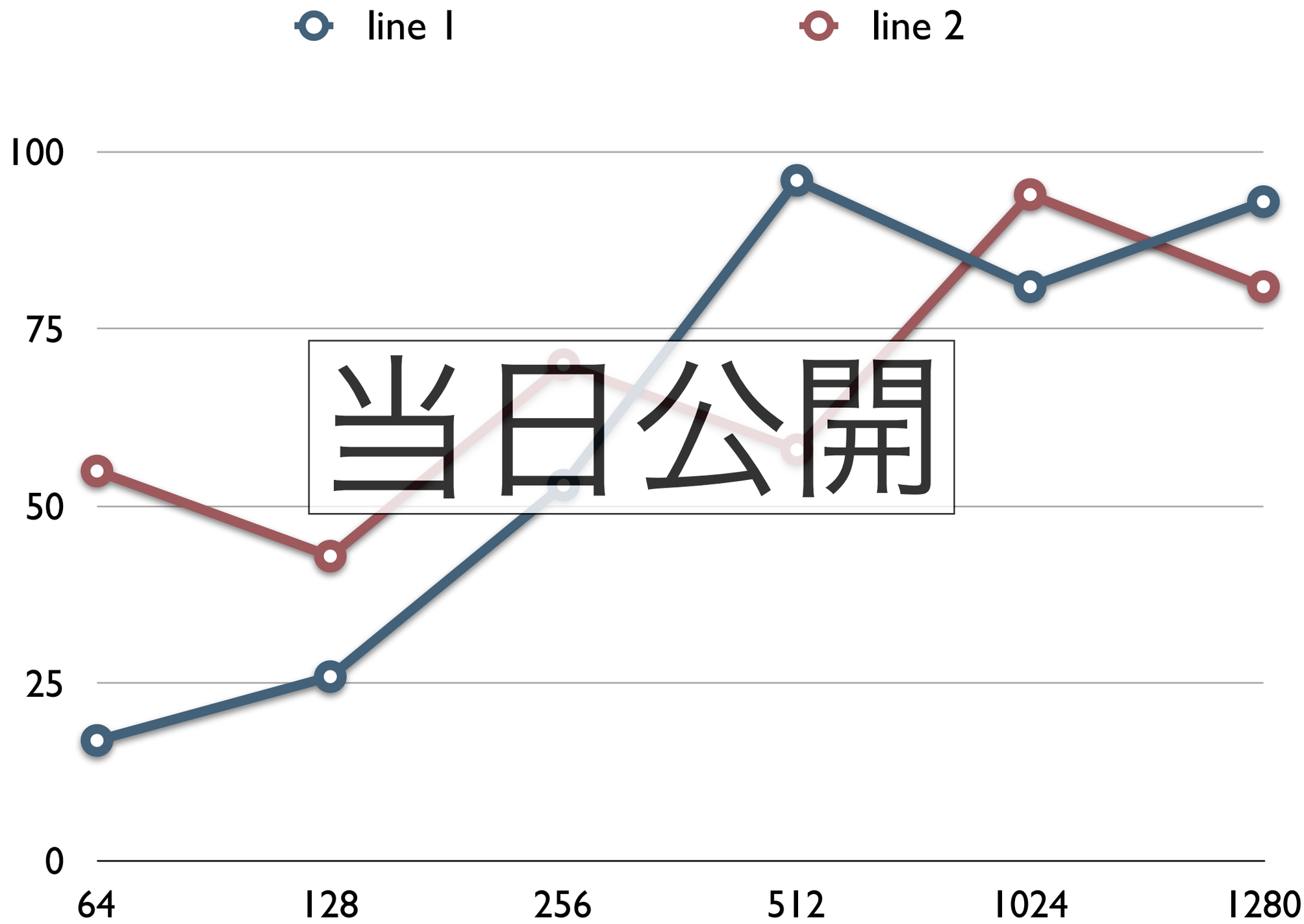
例）CPU0-3 があって CPU1,CPU3 に割り当てたい場合

0x2(CPU1) + 0x8(CPU03) = 0xa

（通常は 1 CPU → 1 送信キューとするが送信キュー
の数よりも CPU の数の方が多い場合など）

※ Linux Kernel 2.6.38 以降でコンパイル時に CONFIG_XPS=y が設定されていること

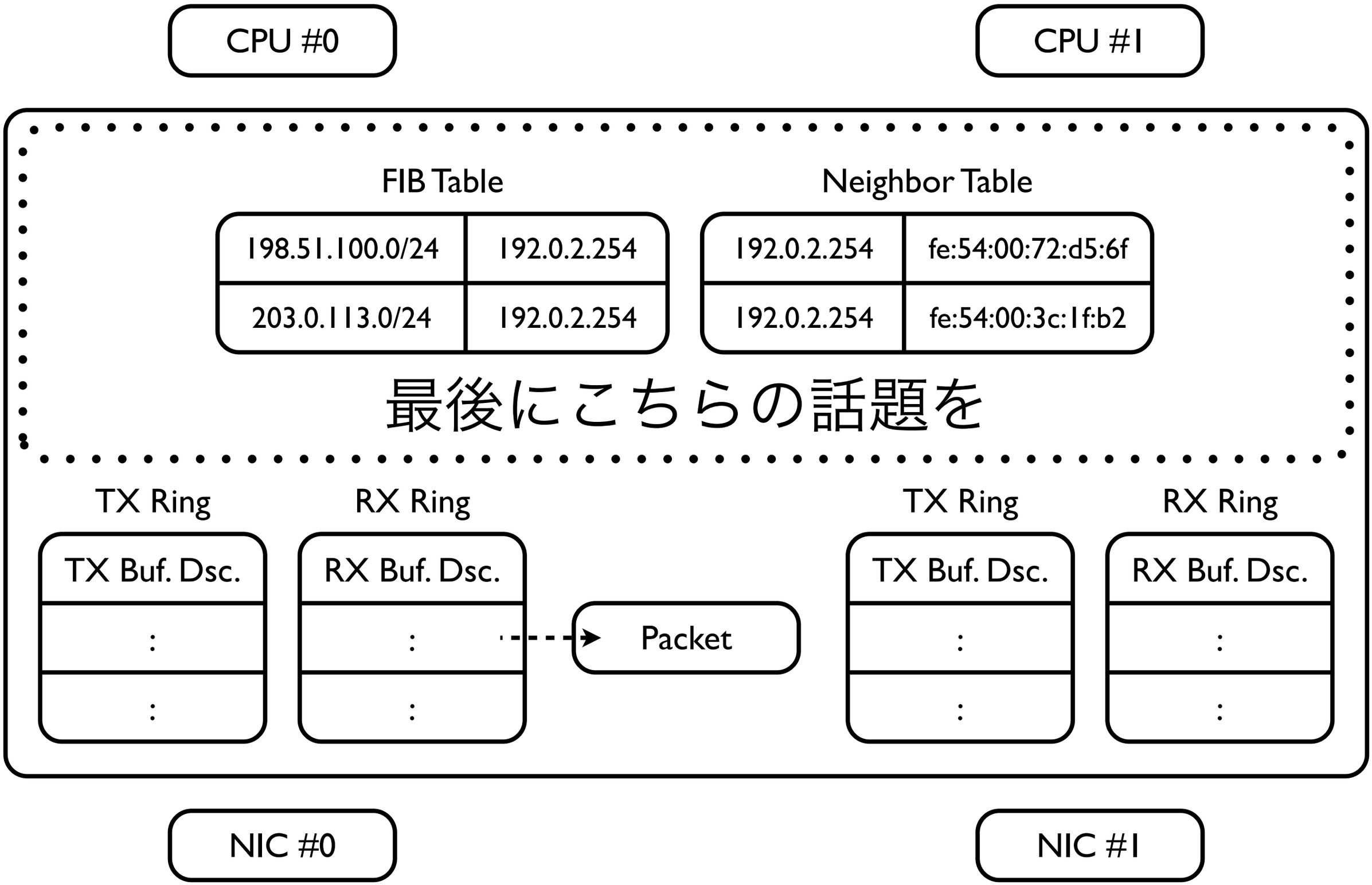
XPS による性能改善の評価



まとめ

- 可能であれば Receive Side Scaling に対応した NIC を選定しましょう
  - CPU の数に応じた転送性能が得られます
- NUMA の落とし穴に気をつけましょう
  - 構成によってはパケットロスが生じる可能性があります
  - パケットロスが許容されない環境の場合は安価な CPU x 2 ソケット構成より高価な CPU x 1 ソケット構成の方が良いかもしれません
- Receive Side Scaling に対応しない NIC の場合は RPS の利用を検討しましょう
  - 負荷に応じて CPU の割り振りを検討しましょう
  - 高負荷が見込まれるときは割り込み処理とフォワーディング処理とで CPU を分離したほうがよいかもしれません
- 送信 Multi-Queue 対応の NIC の場合は XPS の利用を検討しましょう
  - 送信時のロック競合が低減しスループットが向上するかもしれません
  - ただしドライバによっては XPS が有効に機能しない場合があります

経路数と Flow 数と転送性能



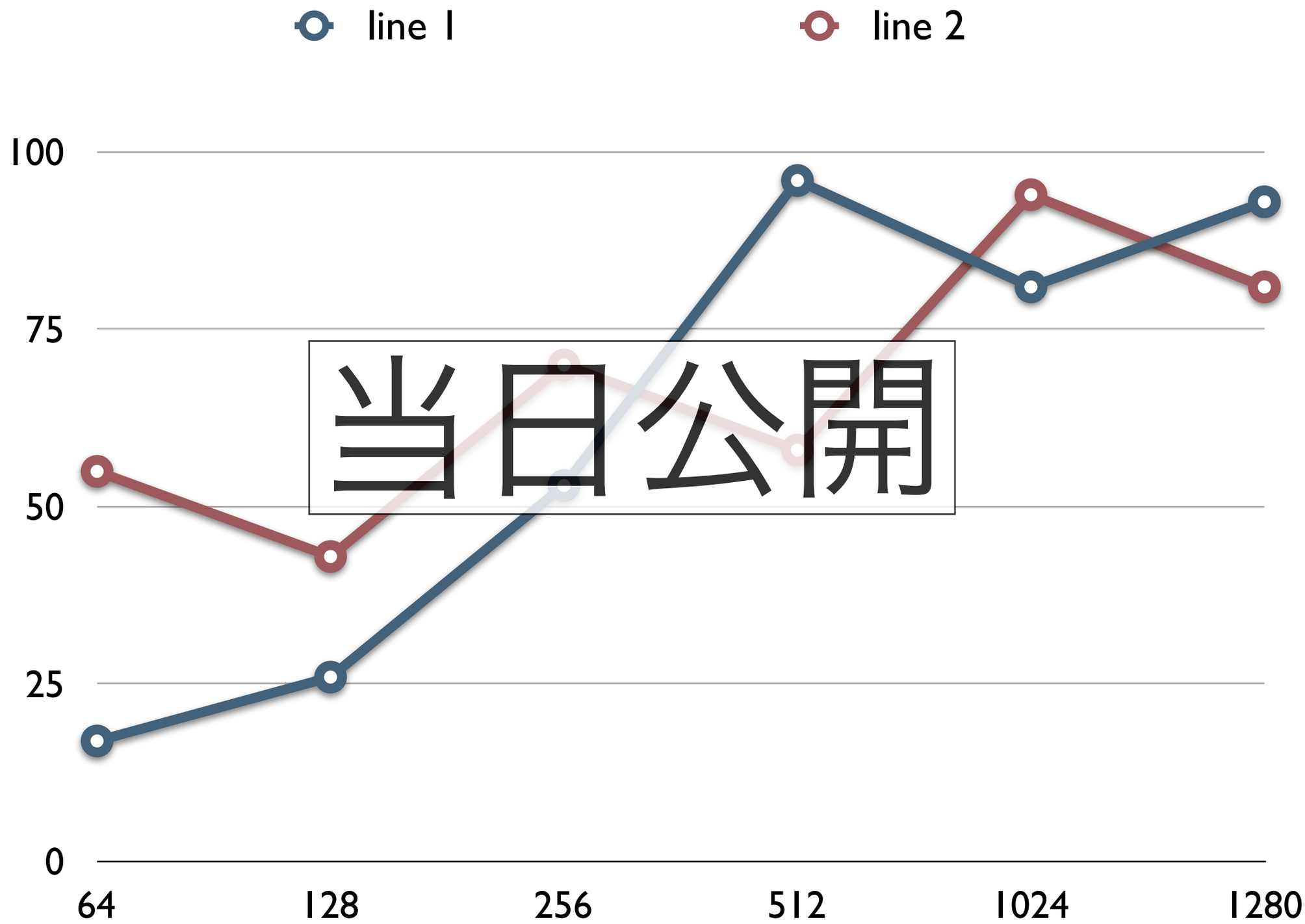
試験内容

経路数		Flow 数		方向
1		16		→
1,024	×	16,384	×	
1,048,576		524,288		⇄
		16,777,216		

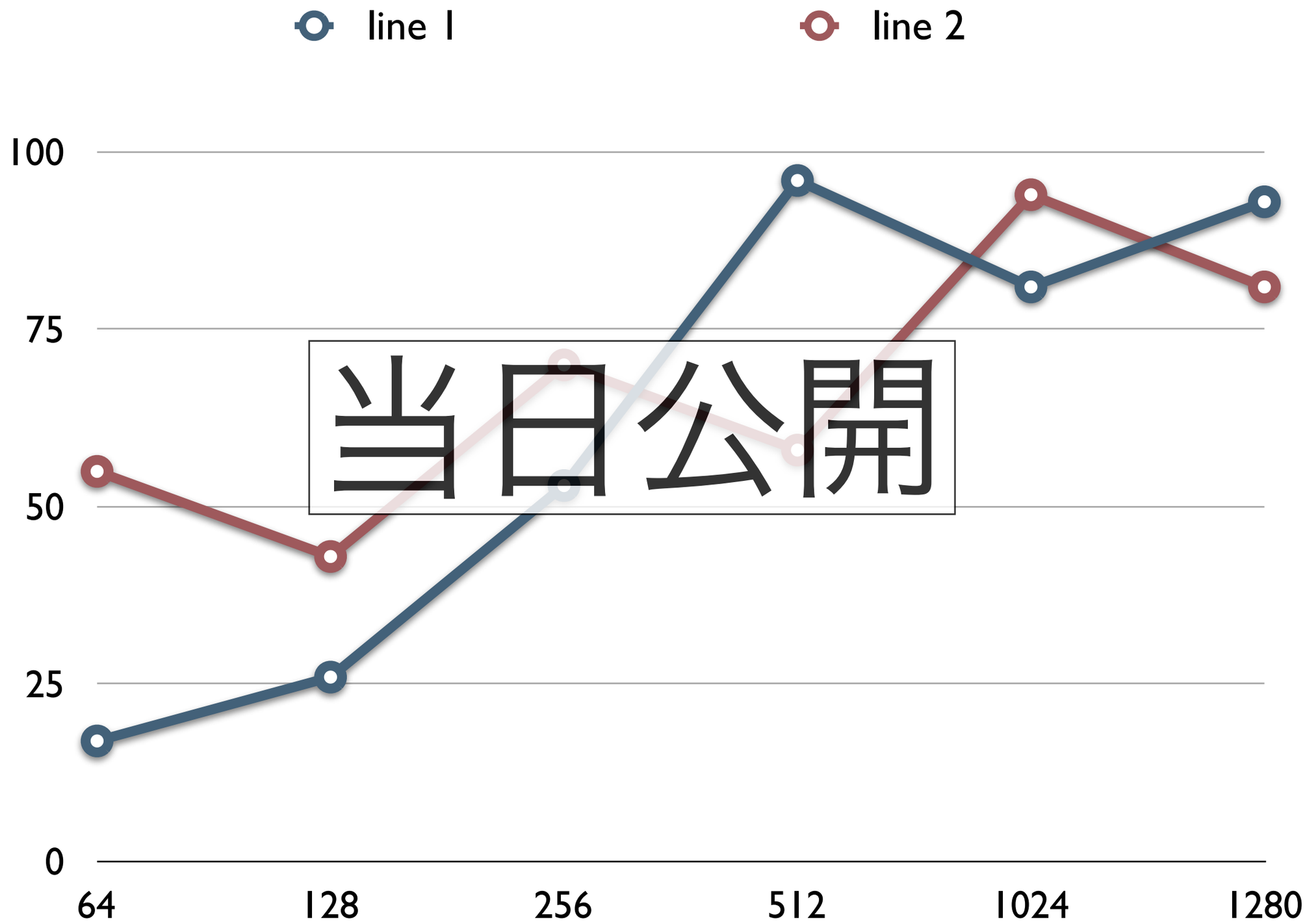
※ Packet Size はすべて IPv4=64byte/IPv6=78Byte

※ RSS=4,4 Node=0,0 で IRQ は全て Node0 の CPU に割当

経路数と Flow 数と転送性能（IPv4 の場合）



経路数と Flow 数と転送性能（IPv6 の場合）



FIB と Destination Cache（IPv4 の場合）

FIB Table(Trie)

Destination	Next Hop
198.51.100.0/24	192.0.2.254
203.0.113.0/24	192.0.2.253

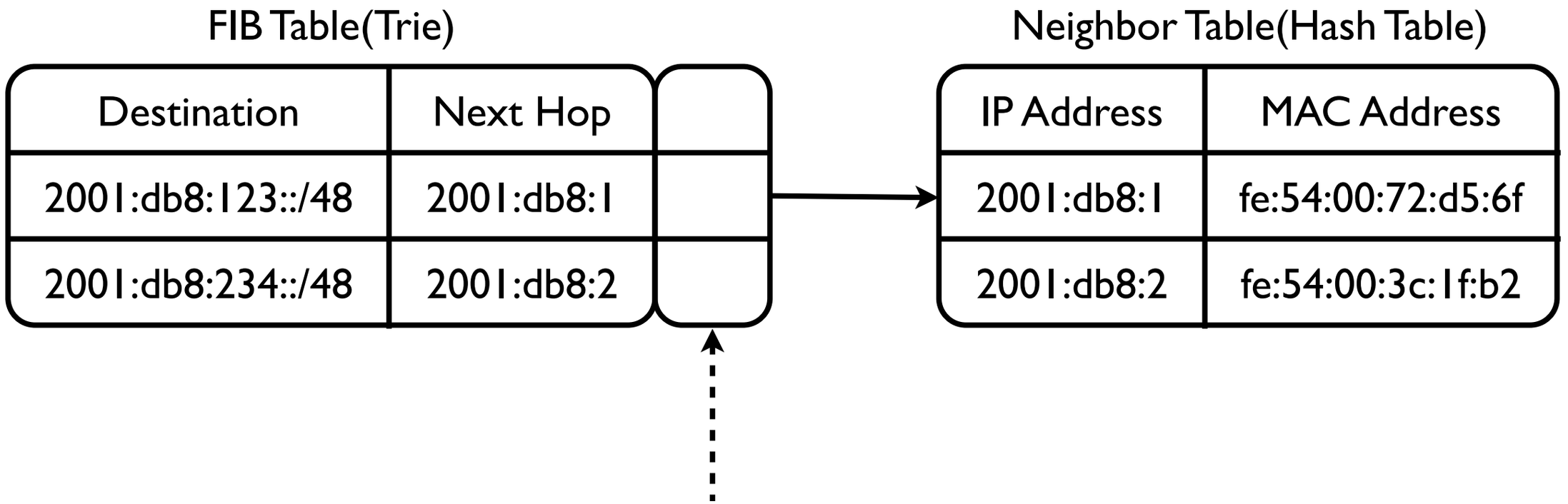
Neighbor Table(Hash Table)

IP Address	MAC Address
192.0.2.254	fe:54:00:72:d5:6f
192.0.2.253	fe:54:00:3c:1f:b2

Destination Cache(Hash Table)

Source	Destination	Next Hop
203.0.113.3	198.51.100.1	192.0.2.254
203.0.113.8	198.51.100.1	192.0.2.254
203.0.113.4	198.51.100.1	192.0.2.254
:	:	:

FIB と Destination Cache（IPv6 の場合）



IPv4 とは異なり FIB Entry の先にリンクされる

また Flow 毎に生成されることはない

参考）5 つの ISP に接続された BGP ルータ

```
vyatta@router01:~$ show system memory
             total       used       free     shared    buffers     cached
Mem:       3754164     974916    2779248          0      67088     169748
Swap:            0          0          0
Total:     3754164     974916    2779248

vyatta@router01:~$ show ip bgp memory
715516 RIB nodes, using 44 MiB of memory
1921181 BGP routes, using 59 MiB of memory
2 Static routes, using 64 bytes of memory
1921188 Adj-In entries, using 29 MiB of memory
25 Adj-Out entries, using 500 bytes of memory
338922 BGP attributes, using 12 MiB of memory
57365 BGP extra attributes, using 4033 KiB of memory
305452 BGP AS-PATH entries, using 3580 KiB of memory
305729 BGP AS-PATH segments, using 3583 KiB of memory
52 BGP community entries, using 832 bytes of memory
12 peers, using 30 KiB of memory
103 hash tables, using 2060 bytes of memory
644447 hash buckets, using 7552 KiB of memory
6 compiled regexes, using 192 bytes of memory
```

まとめ

- メモリは多めに搭載しましょう
  - PC 用のメモリなんてタダみたいなものです
- インターネットの経路数が増大しても困ることはないと思います
  - メモリを 4GB 搭載した PC でもいまの 4 倍までなら確実に余裕です
- 但し IPv4 に限り Flow 数に注意しましょう
  - Flow 数が多いとパケットロスが発生する可能性があります