Mintを用いたNICドライバの割り込みデバッグ支援環境の評価

2015/12/08

藤田将輝

1 はじめに

本資料では，Mint を用いた NIC ドライバの割り込みデバッグ支援環境の評価について述べる．まず，本デバッグ支援環境におけるパケット長による送信処理の時間についてを測定し，どの程度の間隔でパケットを送信できるかを評価する．次に，本デバッグ支援環境を使用した際，どの程度の通信量を実現できるかを測定し，評価する．最後に，本デバッグ支援環境を用いて連続でパケットを送信した際に，NIC ドライバと End-to-End で処理できるパケットの数に，どの程度の差があるかを示す．

2 評価環境

評価環境を表 1 に示す．

表 1　測定環境

項目名	環境
OS	Fedora14 x86_64(Mint 3.0.8)
CPU	Intel(R) Core(TM) Core i7-870 @ 2.93GHz
メモリ	2GB
Chipset	Intel(R) 5 Series/3400
NIC	RTL8111/8168B
NIC ドライバ	RTL8169
ソースコード	r8169.c

3 送信処理時間の測定

本環境を用いた際，どの程度の短い間隔ならばパケットを送信できるかを測定した．パケットの送信処理には，パケットをメモリにコピーする処理が含まれているため，送信処理時間は，パケット長が増加する毎に 1 次関数的に増加するものと考えられる．これを示すため，いくつかのパケット長の条件下で，送信処理時間を測定した．具体的には，3000 回の送信処理中の 1000 回〜2000 回目の処理時間の平均を求める．最初の処理と最後の処理には不確定要素が含まれると考えられるため，間の 1000 回の時間から平均を取る．結果を表 2 に示す．また，結果が 1 次関数的に増加していることを示すため，パ

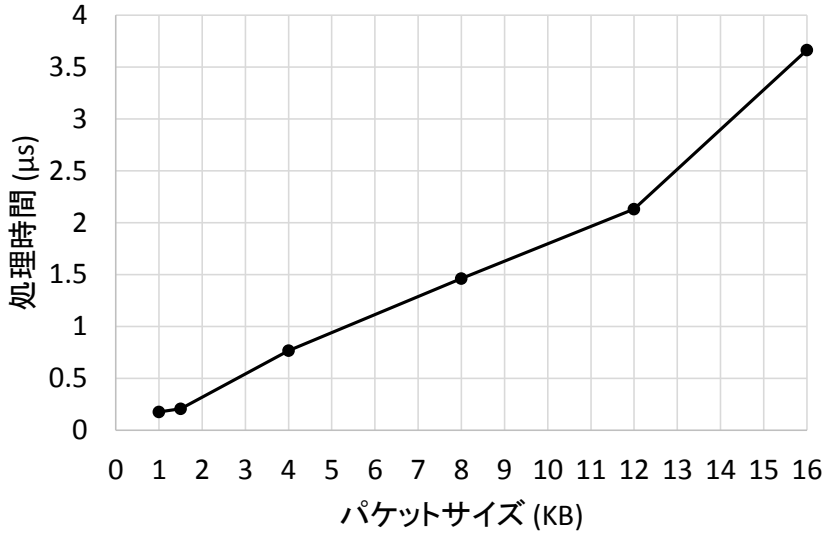

図 1　パケットサイズと送信処理時間の関係

ケット長と処理時間の関係を図 1 に示す．本デバッグ支援環境では，1 処理毎に表 2 に示している時間は必ずかかるため，表 2 に示した以上の間隔を指定する必要がある．

表 2　各パケットサイズにおける送信処理時間

パケットサイズ (KB)	処理時間 (μ s)
1	0.175
1.5	0.205
4	0.766
8	1.462
12	2.131
16	3.664

4　実現できる通信量の測定

本デバッグ支援環境で実現できる通信量について測定する．具体的には，いくつかのサイズの EthernetFrame(UDP パケット) を 5000 回連続で送信することを 1 サイクルとし，送信間隔を増加させながら何サイクルも行い，全てのパケットを処理できる最も短い間隔の際の時間から通信量を測定した．受信の成功は，NIC ドライバでパケットをソケットバッファに格納する処理を通ることとした．また，実 NIC を用いた場合と比較するため，実 NIC を用いた際の通信量も併せて測定する．実 NIC を用いて通信する際の，MTU(Maximum Transmission Unit) は 1.5KB であるため，パケット長は 1.5KB とした．結果を表 3 に示す．結果から，実 NIC を大きく超える通信量を実現できていることがわかる．また，最高で 30Gbps を実現できており，現在は開発されていない高速な NIC をシミュレートできる

と考えられる．

表 3　各パケットサイズにおける実現可能な通信量

パケットサイズ (KB)	通信量 (Gbps)
1	4.6
1.5(実 NIC)	0.92
1.5	6.3
4	17.4
8	22.7
12	29.6
16	30.6

5　NIC ドライバと End-to-End における処理可能なパケット数の測定

本環境を用いて，連続でパケットを処理させる際，以下の 2 つの条件でどれだけ差があるかを測定した．

(条件 1)　NIC ドライバで処理するパケットをカウント
(条件 2)　End-to-End で処理するパケットをカウント

デバッグ支援 OS からパケットを送信し，デバッグ対象 OS でパケットを受信する際，NIC ドライバ，プロトコルスタックを経由して，AP までパケットが配送される．本環境を用いると，NIC ドライバで処理できる限界の時間と End-to-End で処理できる限界の時間を測定し，プロトコルスタックの影響を調査できる．測定の構成を図 2 に示し，説明する．

(1) デバッグ支援 OS 上で動作する割り込みジェネレータにより，パケットの作成，割り込み回数の指定，および割り込み間隔の指定を行う．
(2) システムコールにより，デバッグ支援機構が動作し，送信処理を行う．
(3) NIC ドライバがパケットを受信する．条件 1 ではここで処理するパケットの数をカウントする．
(4) AP にパケットが届く，条件 2 ではここで処理するパケットの数をカウントする．

結果を図 3 と図 4 に示す．結果から，受信成功率が 100% になるまでの時間は，図 4 の方が長いことが分かる．この差は，プロトコルスタックによる影響であると考えられる．また，図 4 は受信成功率が大きく下がる部分がある．NIC ドライバ側では，受信成功率が 100% を超えると大きく落ち込むことはないため，これもプロトコルスタックに影響が大きいと考えられる．CPU 性能を高めたり，プロトコルスタックのアルゴリズムを高速なものにしたりして，プロトコルスタックの処理を早くするとこれらの差は小さくなると考えられる．このように，本環境を用いると，実 NIC を用いずにドライバやプロ

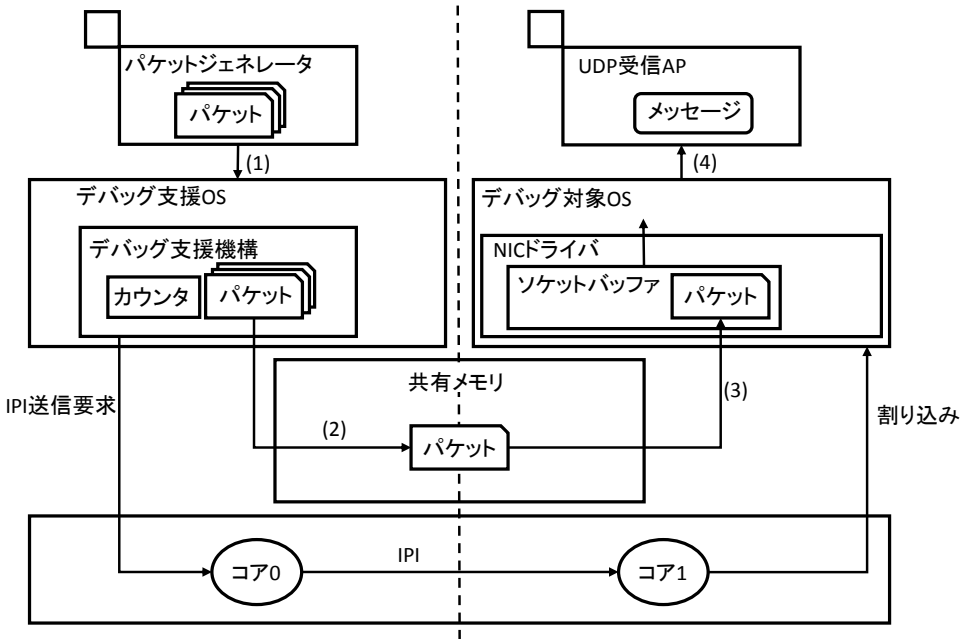

図 2　測定の構成図

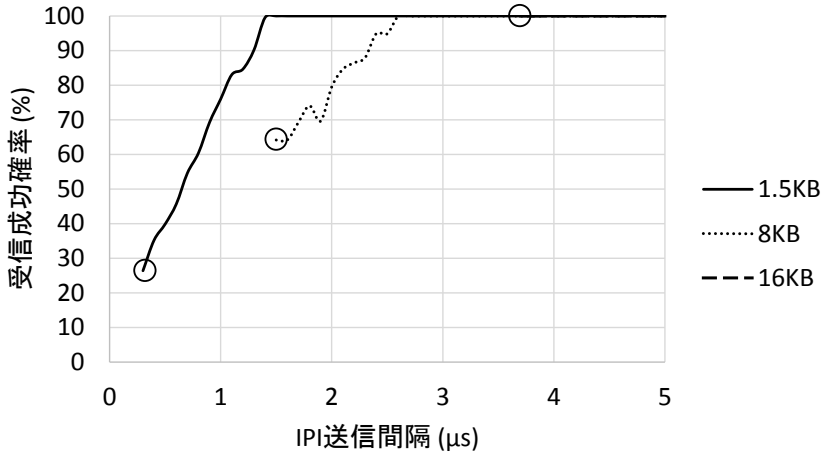

図 3　NIC ドライバにおける IPI 送信間隔と受信成功率の関係

トコルスタックの診断を行える．

6　おわりに

本資料では，Mint を用いた NIC ドライバのデバッグ支援環境の評価を行った．本デバッグ支援環境を用いることで，最速で約 200ns の割り込み間隔を実現できる．また，最高で約 30Gbps の通信量を実現できる．さらに，本デバッグ支援環境を用いることで，プロトコルスタックの影響がどのように現れるかを示した．

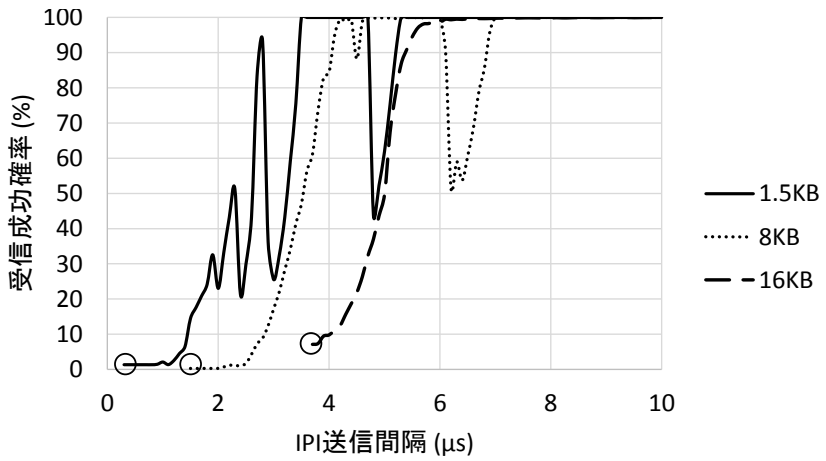

図 4　NIC ドライバにおける IPI 送信間隔と受信成功率の関係