デバッグ支援機構の処理流れと評価

2015/10/28

藤田将輝

1　はじめに

Mint を用いたデバッグ支援環境を評価する際に，各処理流れを明確にし，これらにかかる時間を測定した．測定した各処理の時間を用いて，パケットの成功率を求める測定を行った．各処理時間とパケットの成功率から本デバッグ支援環境について考察した．本資料では，デバッグ支援環境における各処理とこれにかかる時間，どの程度のパケットサイズとパケットの間隔ならばパケットの受信に成功するかを求め，これらについて考察したことを示す．

2　測定環境

測定を行った際の環境を表 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　デバッグ支援環境の処理流れ

3.1　目的

本デバッグ支援環境を用いて，どの程度のパケットサイズとパケットの間隔ならばパケット受信に成功するかを測定する際，どの程度短い間隔でパケットを送信できるかは，パケットのサイズが関係している．パケットのサイズ毎にパケットを送信，または受信する処理でどのくらいの時間がかかっているかを知る必要がある．このため，パケットの送信と受信処理の流れを示し，パケットのサイズによってどの程度の時間がかかるかを測定する．

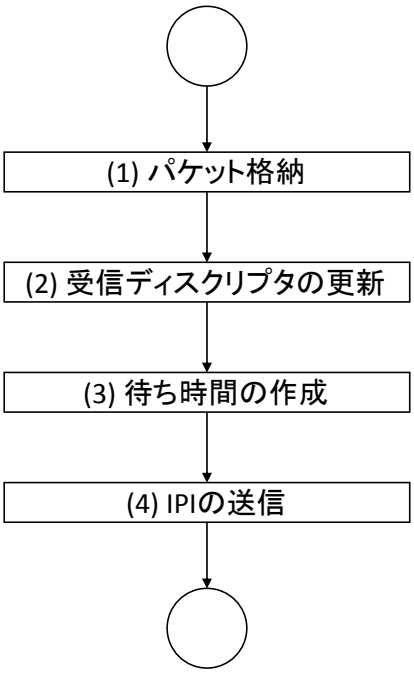

図 1　パケット送信処理フロー

3.2　送信処理

3.2.1　処理流れ

パケット送信処理は，デバッグ支援 OS のシステムコールとして実装しているデバッグ支援機構が行う．デバッグ支援機構がパケットを NIC の受信バッファに配置し，IPI を送信することでパケットの送信を擬似的に行う．デバッグ支援機構の処理中の IPI を送信するまでの処理を図 1 に示し，以下で説明する．

(1) パケットの格納
作成したパケットを Mint の共有メモリに作成した NIC の受信バッファに格納する．

(2) 受信ディスクリプタの更新
NIC ドライバはパケットの受信処理を行う際，受信ディスクリプタからパケットを受信しているかどうかを判断する．このため，受信ディスクリプタを更新し，受信状態にする．

(3) 待ち時間の作成
デバッグ支援機構は指定した間隔でパケットを送信する機能を持っているため，IPI を送信する前に指定した秒数になるまで処理を待つ．

(4) IPI の送信
デバッグ対象 OS の保持するコアに向けて IPI を送信する．これにより，デバッグ対象 OS の NIC ドライバが割り込みハンドラを動作させる．

3.2.2 測定

3.2.1 項における IPI を送信するまでの時間を測定した．つまり，(1) と (2) にかかる時間を測定した．RDTSC を用いて，(1) の処理の始めから，(2) の処理の最後までを 256 回計り，その平均値を処理時間とした．測定した結果を，表 2 に示す．

表 2　IPI の送信までにかかる時間

パケットサイズ	かかった時間
1KB	302ns
1.5KB	326ns
4KB	683ns
8KB	1287ns
12KB	2132ns
16KB	3314ns

結果から，パケットのサイズが大きいほど処理に時間がかかることが分かる．また，表 2 の間隔以下ではパケットを送信することができない．

3.3 受信処理

3.3.1 処理流れ

パケット受信処理は，デバッグ対象 OS の NIC ドライバが行う．デバッグ支援 OS が送信した IPI を契機にして，割り込みハンドラが動作する．この際，割り込みハンドラ内では，パケット受信割り込み処理を行わず，NAPI という機構を用いて，ポーリングでパケットの処理を行わせるように NIC ドライバのポーリング関数を NAPI が管理している poll_list に登録し，ソフトウェア割り込みを発生させる．ソフトウェア割り込みを受けて，poll_list から登録したポーリング関数を呼び出し，パケット受信割り込みを行う．パケット受信割り込み処理が終了すると，poll_list からポーリング関数を削除する．パケット受信処理を図 2 に示し，以下で説明する．

(1) 割り込みハンドラの動作
　デバッグ支援 OS から送信された IPI を契機にして，割り込みハンドラが動作する．これにより，NIC ドライバのポーリング関数を poll_list に登録し，ソフトウェア割り込みを発せさせる．
(2) ポーリング関数の呼び出し
　ネットワーク層に位置する net_rx_action() により，poll_list から (1) で登録したポーリング関数を呼び出す．
(3) ポーリング関数の実行
　NIC ドライバのポーリング関数が実行され，受信割り込み処理関数である rtl8169_rx_interrupt()

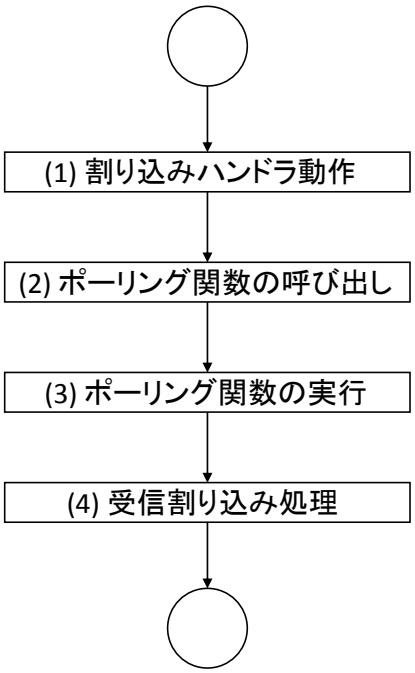

図 2　パケット受信処理フロー

を呼び出す．rtl8169_rx_interrupt() の処理が終了すると，poll_list からポーリング関数を削除する．

(4) 受信割り込み処理
受信割り込み処理を行う．受信バッファにあるパケットを処理すると終了する．

3.3.2　測定

3.3.1 項の処理流れの内，以下の 2 項目を測定した．

(1) 割り込みハンドラが開始してからポーリング関数が呼ばれるまでの時間の測定
割り込みハンドラではパケット受信処理を行なっていないため，割り込みを挿入してからパケットが受信されるまでの時間を知るには，割り込みハンドラが呼ばれてから，ポーリング関数が開始するまでの時間を知る必要がある．このため，割り込みハンドラが開始してからポーリング関数が呼ばれるまでの時間を測定する．具体的には，RDTSC を用いて，3.2.1 項の (1) の始めから，(3) の始めまでの時間を測定した．

(2) 受信割り込み処理にかかる時間の測定
どの程度の IPI 送信間隔ならばパケットを受信できるかを知るためには，受信処理にどの程度の時間がかかるかを知る必要がある．このため，受信割り込み処理にかかる時間を測定する．具体的には，RDTSC を用いて，3.3.1 項の (4) の始めから，(4) のおわりまでの時間を測定した．

(1) の測定結果は約 3us であった．(2) の測定結果を表 3 に示す．

表 3　パケットの受信にかかる時間

パケットサイズ	かかった時間
1KB	384ns
1.5KB	350ns
4KB	683ns
8KB	1518ns
12KB	2065ns
16KB	3331ns

結果から，割り込みハンドラの開始から約 3us までは処理が開始しないことが分かる．また，パケットの処理はパケットのサイズが大きいほど時間がかかることが分かる．表 2 の値と類似していることから，memcpy() にかかる時間が大半を占めているのではないかと考えている．

4　どの程度のパケットサイズと間隔ならば受信に成功するかの測定と考察

4.1　測定対象

パケットのサイズとパケットの送信間隔を変化させて，どの程度のパケットサイズ，間隔ならばパケットの受信に成功するかを測定する．この測定の対象として，以下の 2 つがある．

(1) NIC ドライバ
NIC ドライバでパケットを受信した際に，ソケットバッファに格納したことを成功と定義し，測定を行う．

(2) デバッグ対象 OS 上で動作する UDP 受信プログラム
デバッグ対象 OS 上で動作する UDP 受信プログラムでメッセージを取得できたことを成功と定義し，測定を行う．

4.2　NIC ドライバ

6 つのサイズのパケットを 5000 回連続で送信した際の受信成功率を計測する．5000 回は指定した一定の間隔で送信する．5000 回連続で送信することを 1 サイクルとし，間隔を 1 サイクル毎に増加させながら何サイクルも行う．これらの操作により，どの程度のパケットサイズ，間隔ならば全てのパケットが受信に成功するかが分かる．また，各パケットサイズにおける最短の間隔は，表 2 の値を超えるものとしている．結果を図 3 に示し，以下で説明する．図 3 からどのサイズでも 3 μ s まではほとんどパケットが受信できていないことが分かる．この詳細についてはまだわかっていない．3 μ s 以降は，パ

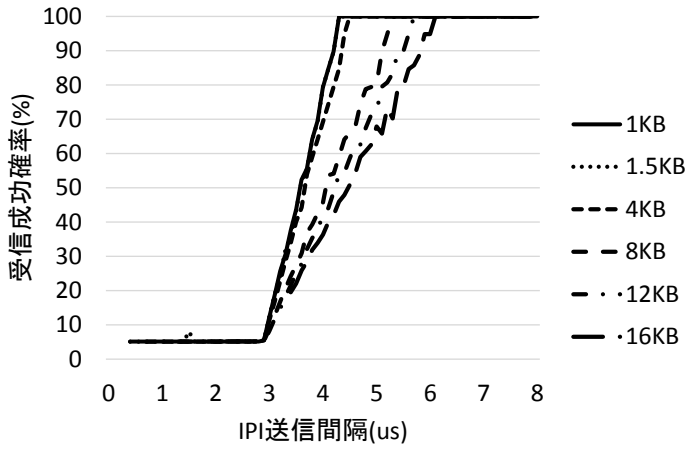

図 3　ドライバにおけるパケット受信成功確率

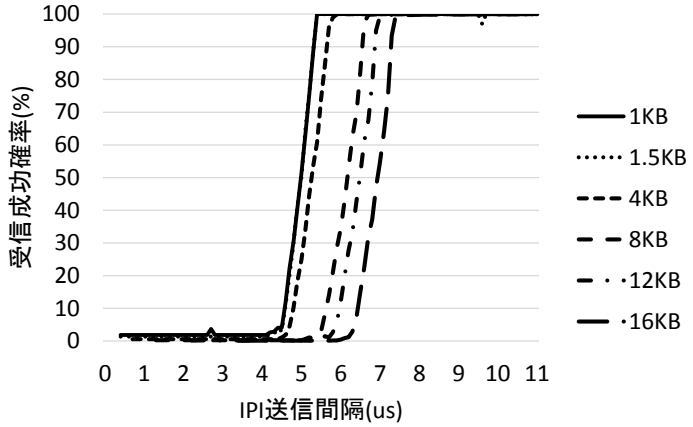

図 4　プログラムにおけるパケット受信成功確率

ケットが大きくなるほど傾きが緩やかになっていることが分かる．これは，パケットが大きくなるほどパケット受信処理に時間がかかるためである．

4.3　デバッグ対象 OS 上で動作する UDP 受信プログラム

4.2 節とほとんど同様の流れで測定を行う．結果を図 4 に示し，以下で説明する．図 4 から，パケットのサイズが増加するほど受信成功確率が上昇し始める時間が増加していることが分かる．これは，パケットのサイズが増加するほど NIC ドライバですべてのパケットを処理できる時間が増加することに起因していると考えられる．また，パケットのサイズによらず傾きは一定である．このことから，ドライバ以降の処理は，パケットのサイズによらない処理をしていると考えられる．

5　おわりに

本資料では本デバッグ支援環境における送信処理と受信処理の処理流れ，およびこれらにかかる時間を示した．また，この結果から，受信処理に関する考察を行った．