2003－HPC－95　(9)
2003／8／4

ハードウェアネットワークエミュレータを用いたTCP/IP通信の評価

児玉祐悦† 工藤知宏†
佐藤博之† 関口智嗣†

我々はネットワークの観測、エミュレーション、新規プロトコルの試験等を目的としたFPGAを用いたネットワークテストベッドGNET-1を開発した。GNET-1は、Gigabit Ethernetポート4組と高速SRAMなどがFPGAに接続された構成をとっている。本報告ではGNET-1を紹介すると共に、GNET-1を用いて広域網の遅延を模擬する環境を構築し、いくつかのOSでのTCP/IP通信についてその性能やフレーム破棄時の振る舞いを観測した結果を報告する。

Evaluation of TCP/IP communication using a hardware network emulator

Yuetsu Kodama ,† Tomohiro Kudoh ,† Hiroyuki Sato †
and Satoshi Sekiguchi†

We have developed a network test bed GNET-1 so as to observe network traffic, emulate networks and test communication protocols. GNET-1 is provided with four Gigabit Ethernet ports and four high speed RAM banks which are connected to a central FPGA. In this report, we introduce the organization of GNET-1 and show the evaluation results of TCP/IP communication on several operating systems. GNET-1 emulates a large delay, which is corresponds to that of a wide are network, and a single frame discard on full speed of Gigabit Ethernet traffic.

1. はじめに

近年、広域網でもGbpsクラスのネットワークが利用されるようになりつつある。しかし、広域網では、光の速度の制約などにより大きな遅延があり、単一のアプリケーションでGbpsクラスのバンド幅を有効利用するには様々な問題がある。特にインターネットで一般に用いられるTCPでは、フローコントロールなどにより実際のバンド幅が制限される。これらを解決するためのプロトコル開発などが行われている[1)]。我々は、ネットワークの観測、エミュレーション、新機能の評価などを行うことを目的にネットワーク実験装置GNET-1を開発した。本稿では、GNET-1の概要と、GNET-1を用いた既存通信プロトコルの予備評価結果について述べる。

2. GNET-1の概要

GNET-1のブロックダイアグラムと実装基板を、図1と図2にそれぞれ示す。全体を制御するFPGA(Field Programmable Gate Array)と4チャンネルのGigabit Ethernet(GbE)光入出力ポートを持つ。使用しているFPGAは、Xilinx社のXC2V6000で76K論理セルと2.5Mbitのブロックメモリを持ち、ユーザI/Oピン数は824pinである。これに、SERDESを介して1000BASE-SXの光トランシーバーが接続されている。

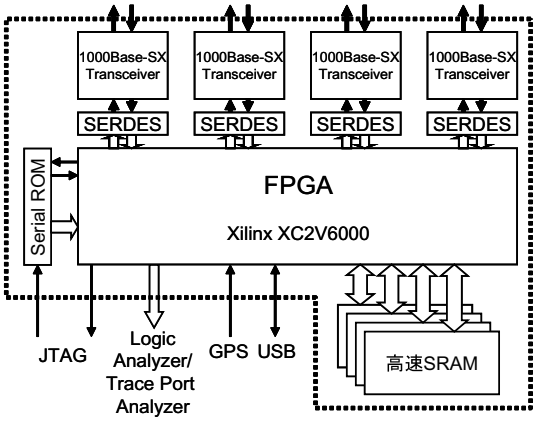

図1 GNET-1のブロックダイアグラム

FPGA外部に36bit × 4M word（144Mbit）の高速SRAMを4組搭載しており、それぞれが片方向あたり32bit/31.25MHz（125MB/s）以上のバンド幅で同時読み書き可能である。また、1チャンネルのUSB1.1

† 産業技術総合研究所グリッド研究センター
Grid Technology Research Center, AIST

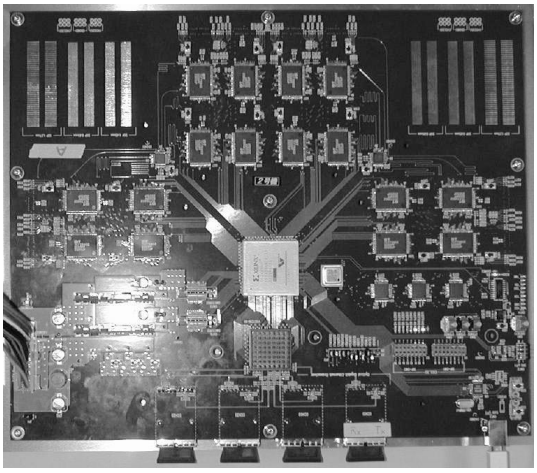

図 2 GNET-1 基板

インタフェイスを実装し、PC との通信により、実験装置内の制御情報の修正変更及び監視が出来る。さらにロジックアナライザ接続のための観測コネクタと正確な時刻を得るための GPS 接続端子を実装している。

GNET-1 は、FPGA コンフィグレーションの変更により様々な用途に用いることができる。現在想定している用途は以下のようなものである。

(1) Gigabit Ethernet 上のパケットの観測
GbE のリンクに GNET-1 を挿入することにより、リンクを流れるイーサネットフレームを観測しフレームそのものをロジックアナライザ等により観測したり、ハードウェアによる統計情報を得る。また、WAN をまたがる通信の送信端と受信端に GNET-1 を用意し、GPS から得られた時刻をイーサネットフレーム上の IP パケットに載せることにより片方向の通信レイテンシを高い精度 (μs 単位) で知ることができる。

(2) 広域網のエミュレーション
広域網を介したクラスタ間の通信にはインターネットプロトコルを用いることが普通で、信頼のある通信の実現には TCP が用いられる。広域網では通信遅延が大きくなる。ルーターなどのネットワーク機器において遅延が発生するほか、広域網で用いられる光ファイバーでは 1m 当たり 5ns 程度の遅延があり、直線距離で 1 万 km 程度離れている日米間では、遅延は片道あたり 100ms 程度になる。このような遅延がある環境では通信性能は著しく低下する。また、広域網ではネットワーク中に異なる性質を持つ物理層や種々のルータなどが介在する。広域網で安定して高バンド幅な通信を行うためには、これらのネットワークの性質に対応したプロトコルが必要である。
GNET-1 は高速 SRAM を FIFO として用いることにより、広域網の遅延を模擬することができる。各組は 144Mbit の容量を持つので、1Gbps の通信リンクに 100ms 強の遅延を与えることができる。GbE リンクの双方向にそれぞれ遅延を付加すれば、往復 200ms 強の遅延を模擬できることになる。また、通信誤りの模擬や、中間経路での ATM などを想定したパケットの分割や、ルータにおける輻輳や RED(Randum Early Detection) などのレートコントロール機能の模擬も可能である。
GNET-1 を用いて広域網のエミュレーションを行えば、再現性のある環境でプロトコルの振る舞いを確認しながら開発を効率的に進めることができる。

(3) 新規スイッチ機能の開発
GNET-1 は 4 組の GbE ポートを持っているので、FPGA の設定により 4 ポートスイッチとすることができる。我々はスイッチに通信の効率や信頼性を向上させるための新たな機能を組み込むことを検討しており、GNET-1 はそのためのテストベッドとなる。

これらは、例えば広域網をエミュレーションしながらパケットを観測するなど同時に組み合わせて行うことができる。

3. 遅延機能の確認

まず、GNET-1 上に単純なネットワーク遅延を模擬する機能を実装した。現在の実装では、入力ポートから到着したデータ (8bit, 125MHz) に対して MAC 入力処理等を行い、32bit にまとめてチップ内の非同期 FIFO に格納する。この時、フレーム毎に 32bit のタイムスタンプ (分解能 100ns) を追加するとともに、32bit 毎に 4bit のタグを追加する。このタグは、タイムスタンプワードを示す tv フラグ、フレームの最後を示す eop フラグ、ワード内の有効バイト数を示す bs からなる。tv フラグによりフレームの先頭を容易に検出できる。非同期 FIFO からのデータ (36bit, 31.25MHz) は外部の FIFO へいったん格納される。それと同時に、外部 FIFO の先頭からデータを読み出すため、外部 FIFO へのアクセスはデータ処理部の倍の 62.5MHz で処理される。外部 FIFO から読み出されたデータは、チップ内の非同期 FIFO に格納される。非同期 FIFO からのデータは、遅延処理および MAC 出力処理等を行いネットワークポートに出力 (8bit, 125MHz) される。遅延処理ではフレームの先頭のタイムスタンプに、指定された遅延時間を加えた値が、現在の時刻を越えるまで出力を待たせることにより遅延を制御している。

遅延機能が正常に働いていることを確認するために、パーソナルコンピュータを用いてソフトウェアによっ

遅延	NISTnet		GNET-1	
設定値	RTT	BW	RTT	BW
(ms)	(ms)	(Mbps)	(ms)	(Mbps)
0	0.09	796.46	0.05	918.66
1	2.17	225.46	2.05	244.41
2	4.13	119.47	4.05	123.50
5	10.26	48.22	10.05	49.45
10	20.24	24.08	20.05	24.39
20	40.27	12.09	40.05	11.89
50	100.33	4.71	100.05	4.30
100	200.19	2.24	200.04	1.75

表 1 NISTnet と GNET-1 による遅延とバンド幅

て遅延を実現する NISTnet[2] と本プロトタイプを用いてギガビットイーサネットによる 2 点間の通信に遅延を挿入し遅延と実効性能の関係を測定した。測定環境の諸元は以下のとおりである。

```
PC
  Intel(R)Xeon(TM)2.4GHz(FSB=533MHz)x2
  DDR-SDRAM 2GB
  NIC SysKonnect SK-9843SX(Driver Ver. V6.05)
  OS:RedHat8.0 (Kernel2.4.18-14smp)
  iperf Ver.1.1.1

NISTnet
  Intel(R)Xeon(TM)2.4GHz(FSB=400MHz)x2
  DDR-SDRAM 1GB
  NIC SysKonnect SK-9843SX(Driver Ver. V6.05)x2
  OS:RedHat8.0 (Kernel2.4.20-pre5)
```

表 1 に測定結果を示す。通信路の双方向にそれぞれ遅延を設定し、ping コマンドによる遅延の実測値と iperf による実効バンド幅を測定した。3 回の計測の平均値を示している。

この結果から、遅延が大きい場合は、ほぼ同等の実効ハンド幅が得られていることが分かる。2 つのノードをケーブルで直結したときのバンド幅は 917.79Mbps であった。GNET-1 での遅延設定 0 での測定では、直結とほぼ同等の性能が得られている。これに対し、NISTnet にはソフトウェアによるルーティングのオーバヘッドが存在するため、特に遅延が小さな領域で精密なエミュレーションができない。遅延を 0 に設定したときの実効バンド幅が 800Mbps 弱と小さくなっているのはこのためと考えられる。

また、ping による遅延の実測値は GNET-1 が設定値＋ 0.05ms 程度 (送受信ノードでのオーバヘッドと考えられる) で安定しているのに対し、NISTnet ではややばらつきがある。これは NISTnet はカーネル内でデータをバッファリングすることにより遅延を挿入しておりソフトウェアで割り込みを用いてタイミングを制御しているため、精密な制御が困難であるためと考えられる。GNET-1 ではクロック単位 (31.25MHz のクロックを用いているので 32ns 単位で遅延の設定が可能) でより細かく遅延を設定できる。

4. フレーム破棄のバンド幅への影響

本ネットワークエミュレータは制御部に大規模 FPGA を用いており、機能の追加が容易に行える。現在種々の機能拡張を行っているが、本節では、ビットエラー生成機能及びフレーム単位の破棄機能に関する実装について述べる。また、高い時間分解能でバンド幅を測定する方法についても述べる。これらの機能を用いて、パケットエラーによるバンド幅への影響を詳細に検討する。

4.1 フレーム破棄機能と高精度バンド幅測定機構の実装

遅延機能の実装でも述べたように、GNET-1 内では 31.25MHz, 62.5MHz, 125MHz の 3 種類のクロック (正確には 125MHz は各入力ポートに対応するクロックと出力ポートに対応するクロックの全部で 5 種類) が使われている。32bit データ処理部では 31.25MHz ともっとも遅いクロックで処理すれば良いため、タイミング制約の観点からは、追加機能はできるだけこの 32bit データについて行うようにするのがよい。したがって、ビットエラー生成機能等は外部 SRAM からの読み出しと遅延制御の間に挿入する形で実装している。

ビットエラー生成機能は、乱数生成部とビットエラー部からなる。乱数生成部は、64bit の線形帰還型シフトレジスタを用いて 32bit の乱数を生成している。この乱数が指定したレジスタよりも小さいときに 1 ビットのエラーをデータ部に生成する。また、フレーム破棄制御は、USB を介して制御レジスタがセットされてから最初に到着したフレームの先頭からフレームの最後までの間、出力非同期 FIFO の書き込みをマスクすることにより行っている。これらの処理は 2 段のパイプラインで処理されており、遅延 64ns が加わるだけで最大バンド幅を損なうことなく実装されている。

また、このビットエラー生成部にデータ転送サイズおよび転送フレーム数のカウンタを設けて、これを USB 経由で読み出すことにより 100m 秒程度の時間分解能での測定が可能である。さらに、このカウンタをハードウェアで直接サンプリングすることにより、任意の時間分解能で正確なスループットを測定することができる。このようなサンプリングには以下の複数の方法が可能である。

(1) 本エミュレータでは、各種測定用にロジックアナライザへの出力ポートへ種々のデータを選択して出力することも可能である。これには別途ロジックアナライザを用意する必要がある。

(2) 制御部が FPGA から構成されていることを用いて、専用回路を実装することは可能であるが、観測したい信号がその実験毎に変化する場合には、毎回専用回路を設計する手間は大きい。た

だし、自由度はもっとも大きい。

(3) 最近の FPGA には汎用の内部信号観測ツールが開発環境として用意されていることが多い。本装置で用いている Xilinx 社の VirtexII にも ChipScope Pro というツールが提供されている。これは、観測したい信号向けのロジックアナライザー回路を自動的に生成して、既存の回路に組込むもので、トレースデータはチップ内部のブロックメモリに格納され、あとから JTAG 経由で取り出す仕組みである。信号あたり最大 16K サンプルまでであるとともに、トレースする信号の数および深さは、チップ内部の空きブロックメモリにより制約を受ける。しかし、非常に手軽に内部信号をトレースできる。また、Agilent 社から内部メモリを用いずに最大サンプル数を 2M サンプルに拡張する Trace Port Analyzer (TPA) という製品がある。

以下では、ChipScope Pro および Agilent TPA を用いて上記の転送ワード数およびフレーム数のカウンタをサンプリングすることにより高精度なバンド幅計測を行った。サンプリングクロック周波数をレジスタで指定することにより任意精度での計測を可能にした。ただし、ChipScope Pro にはサンプリング周波数の下限に制限があり安定した動作が可能な 50KHz を最低サンプリング周波数としたため、2M サンプル時で最大 4 秒の測定が可能である。

4.2 単一フレーム廃棄のバンド幅への影響

上記の機能を用いて、ある PC から他の PC へ間で一方向のバースト転送時にパケット破棄を起こした時のバンド幅への影響を詳細に調べた。表 2 に示す同じ構成の 2 台の PC を用いて実験を行った☆。図 3 に本実験のデータの流れを示す。表に示すように、Linux, Solaris, FreeBSD の 3 種類の OS を用いて実験を行った。用いたベンチマークは TCP/IP により通信し、128M バイトのリングバッファ領域への 1M バイトの write を指定した時間繰り返すプログラムである。すべての OS においてソケットバッファは 2M バイトに設定して測定を行った。

図 4 に各 OS で遅延なしで接続した場合の 100ms 毎の平均バンド幅を示す。横軸がフレーム破棄を行った時刻を 0 としたときの経過時間 (秒) である。縦軸はヘッダを含んだデータ転送バンド幅 (MB/s) である。フレーム破棄よりも前の時点で、Linux と Solaris は 120MB/s を超えているのに対し、FreeBSD では 100MB/s 弱しかなく、また性能が安定していない。フレームを破棄すると、いずれの OS でも 100ms 後にはほぼ破棄前のバンド幅に戻っている。

☆ PC は 1000BASE-T のインタフェースを持っているため、メディアコンバータ NETGEAR GC102JP を介して GNET-1 と接続している

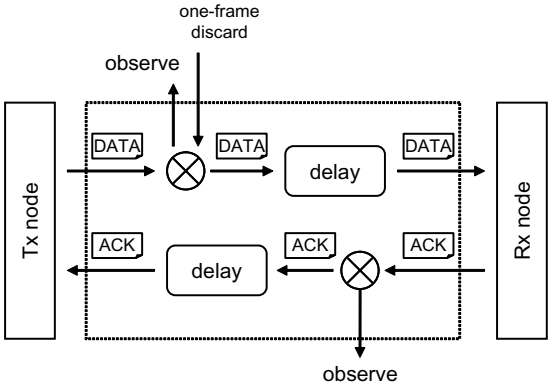

図 3 本実験におけるデータの流れ

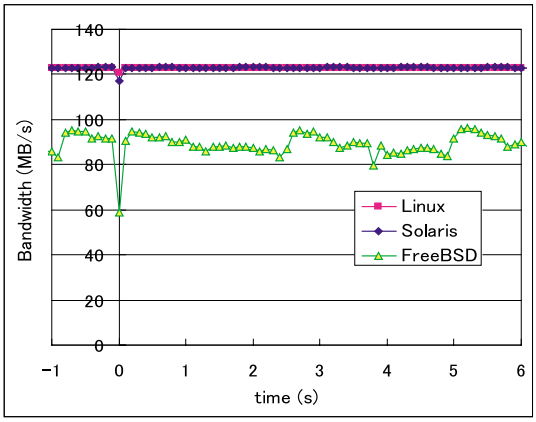

図 4 遅延 0ms 時にフレーム破棄がバンド幅へ与える影響

一方、図 5 は片道 4ms(往復で 8ms) の遅延を入れた場合のバンド幅である。Solaris はこの場合でも 100ms 後には回復しているのに対し、FreeBSD は 5 秒程度、Linux は 8 秒程度回復にかかっている。また、FreeBSD ではいったん回復したバンド幅がまた低下している。この原因は不明である。

Solaris と Linux でのバンド幅の回復にかかる時間と遅延の関係を図 6 に示す。フレーム破棄後 100ms 平均のバンド幅が 120MB/s を越えるまでの時間を、

表 2 実験に用いた PC の諸元

項目	詳細
CPU	Intel Xeon 2.4GHz (FSB:533MHz)x2
Memory	DDR-SDRAM 2GB
NIC	Intel PRO/1000
OS (1)	Linux kernel-2.4.20 + web100 patch
NIC driver	e1000 4.4.12
OS (2)	Solaris 9 (4/03)
NIC driver	ITNCGigaE 4.1.4
OS (3)	FreeBSD 5.1-Release (SMP-Generic)
NIC driver	em0 1.5.31
Socket Buffer	2MB

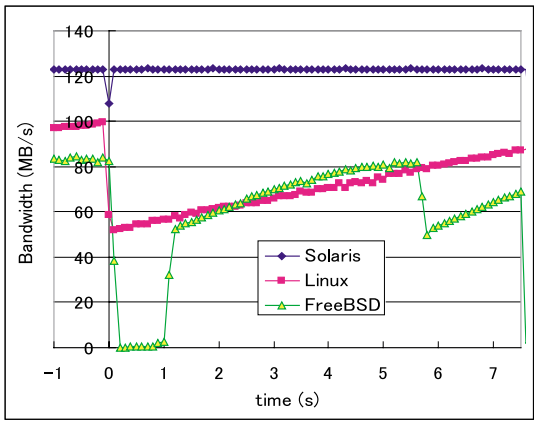

図 5 遅延 4ms 時にフレーム破棄がバンド幅へ与える影響

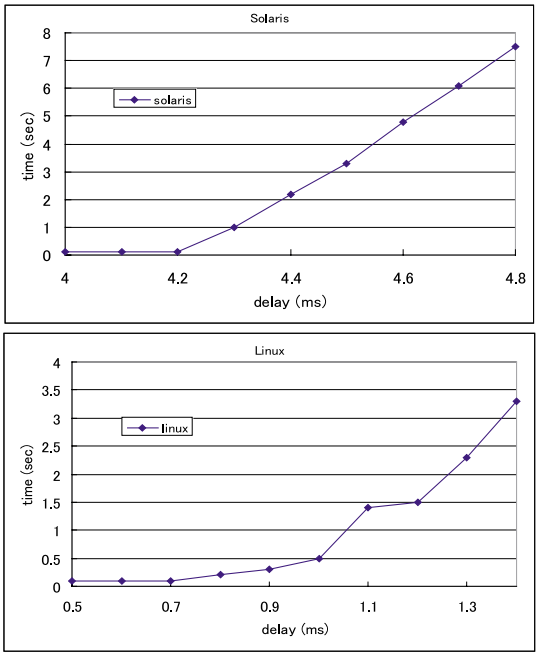

図 6 フレーム破棄からバンド幅が 120MB/s に復帰するまでの時間

遅延を変化させて示している。Solaris では、遅延が 4.2ms を越えると回復に時間がかかり始め、4.7ms の遅延では回復に 7 秒程度かかることがわかった。

GbE では、一つのフレームは連続して送られるので、バンド幅が低い場合はフレーム間の転送が行われていない時間が多いことになる。大容量のデータ転送では、ほとんどの場合最大フレームサイズ 1526B の転送が行われており、GbE 上では 1 フレームの転送に 12μs を要する。そこで、20μs 毎の平均バンド幅を測定した。この間隔で測定を行えば、1〜2 フレームの転送時間の解像度で観測が行えることになる。

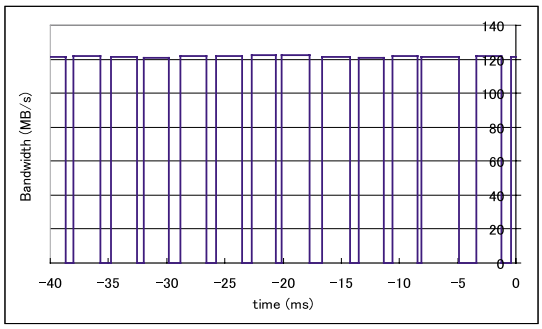

図 7 フレーム破棄を行わない場合の FreeBSD の転送バンド幅

遅延 0 でフレーム破棄を行わない状態での観測結果によると、Solaris や Linux ではほぼ最大の転送性能を示しており、フレーム間に大きな間隔が空くことはないことがわかった。これらに対して FreeBSD では転送が行われない期間が多く現れていた。これを示したのが図 7 である。これはフレーム破棄前の 20μ 毎のバンド幅を示したものである。20μs 単位で連続的にフレームが転送されている場合はまとめて、その平均バンド幅により示している。この図によると、平均 3.4ms の周期で転送とアイドルを繰り返し、アイドルはそのうち 28%を占めている。また、Ack フレームも同様に観測したところ、Solaris では 8 フレームに 1 フレーム程度の Ack フレームが帰ってきているのに対し、Linux では 5 フレームに 1Ack フレーム、FreeBSD では 2 フレームに 1Ack フレームが帰ってきていることも確認できた。

図 8(a) は遅延 0 で単一フレーム破棄を行った場合に、フレーム破棄時刻付近を観測した結果である。横軸はフレーム破棄が起こった時間を 0 としたときの経過時間、縦軸は 20μs 単位のヘッダを含むデータ転送バンド幅である。Solaris の場合はフレーム破棄した部分も含めてデータ転送が連続している。これは、破棄したフレームもデータ転送側ではカウントしているためであるが、重複 Ack を受け取ってもデータ転送バンド幅を落さずに転送を継続していることが分かる。Linux では平均 1600 フレームを 20ms で連続的に転送しており、その後平均 270μs のアイドルが続いていることが観測された。FreeBSD では平均 150 フレームを 1.8ms で連続的に転送し、その後平均 320μs のアイドルが続いていることが観測された。

図 8(b) は同様に片道遅延 4ms の場合の結果である。Solaris では 4.8ms 程度アイドル期間があるが、その後は連続転送が開始している。Linux ではフレーム破棄前は連続転送が行われているが、破棄が行われたあとは連続転送フレーム数は小さくなり、アイドル期間も見られる。これはフレーム破棄によりコンジェスションウィンドウが小さくなった結果と見られる。

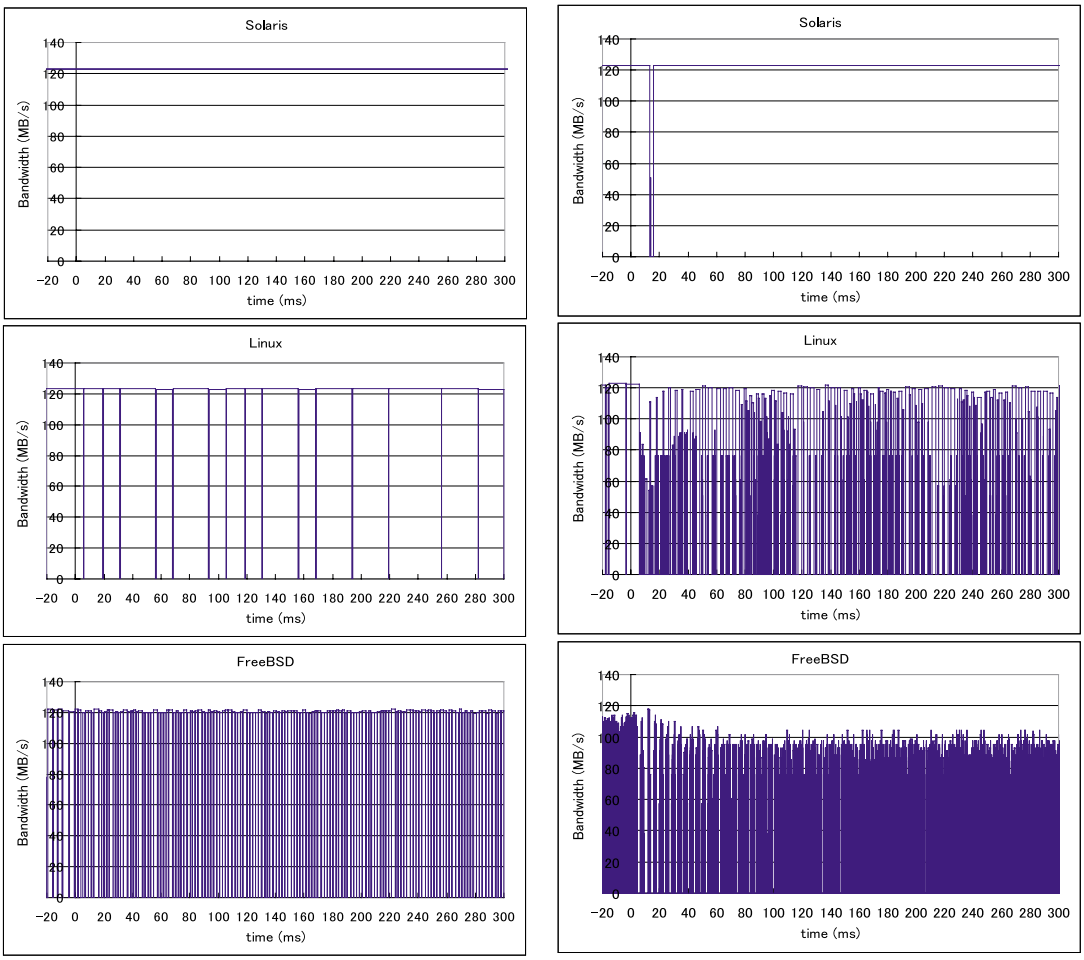

図 8 20μs ステップでの観測結果

5. おわりに

GNET-1 は、FPGA と高速メモリを搭載したシンプルな構造のネットワーク実験装置である。FPGA のプログラムにより様々な用途に用いることができる。PC などを用いてソフトウェアで処理する方式と比べ、ハードウェアで処理するために処理のタイミングが決定的であり、GNET-1 の処理能力によりリンクバンド幅が制限されることはない。

本稿では、GNET-1 の構成についてのべ、GNET-1 を用いて広域網に相当する遅延の挿入とイーサネットフレームの破棄を行う回路を実装し、3 種のオペレーティングシステムの TCP 通信性能の評価結果を紹介した。

今後 GNET-1 を用いてネットワークの観測、エミュレーション、新しいプロトコルのテストなどを行っていく予定である。

謝辞

本研究の一部は、新エネルギー・産業技術総合開発機構基盤技術研究促進事業（民間基盤技術研究支援制度）の一環として委託を受け実施している「大規模・高信頼サーバの研究」の成果である。GNET-1 の開発にご協力いただいた (株) シナジェテックの清水敏行氏、三精システム (株) の藤代行康氏、孫自敏氏に感謝する。

参 考 文 献

1) Eric Weigle and Wu-chun. A comparison of tcp automatic tuning techniques for distributed computing. In *11 th IEEE International Symposium on High Performance Distributed Computing HPDC-11 20002 (HPDC'02)*, July 2002.
2) National Institute of Standards and Technology. *http://snad.ncsl.nist.gov/itg/nistnet/*.