計算機アーキテクチャ　92－10
（１９９２．　１．　２４）

ドーナツ構造Loop Structured Computer(LSC)の提案とそのプロトタイプについて

吉岡　良雄　　石田　努　　成田　清人

弘前大学理学部情報科学科

ノイマン型コンピュータの高速化に対する問題点が数多く指摘されるようになり、これを解決するため各所で大規模なマルチプロセッサの研究が行われている。本報告ではこれらの問題を解決する一つの方策として提案したLoop Structured Computer(LSC)をドーナツ構造にした大規模マルチプロセッサ（ドーナツ型LSC）の構成、これに至るまでの提案動機、パケット構成や処理手順などについて述べている。次に、8層のLSCからなるプロトタイプの構成と実行結果を示し、その有用性を明らかにしている。

A Suggestion of a Doughnut Type Loop Structured Computer and it's Prototype

Yoshio Yoshioka,　Tsutomu Ishida and Kiyohito Narita

Dept. of Information Science Faculty of Science,
Hirosaki University

There has been much discussion of the algorithms of von Neumann machines, and several problems have been indicated. In order to clarify there problems, a large-scale multi-processors system has been investigated.This paper presents a doughnut type multi-processor system consists of several LSC's,its proposal motivation and system specification. Moreover, we show the prototype system of 8 LSC's and the execution results, and that the system is useful.

1 まえがき

ノイマン型計算機が１９４０年代に提言されてから、計算機は論理回路の高速化、集積化等のハードウェアや、OS・高級言語・多種応用ソフトウェアの開発によって、急速な成長を遂げた。この計算機は、１つのプログラム・カウンタで記憶装置から１命令づつを中央処理装置に取り込んで実行する為に、処理のほとんどがデータ転送に費やされている。現在、大規模数値解析において威力を発揮しているスーパーコンピュータも、パイプライン処理などを付加した形態のノイマン型計算機である。これらの計算機の処理能力向上の為には、中央処理装置と記憶装置との間のデータ転送を高速にする必要がある。しかし、導線上を信号が伝わる速さには限界があるので、回路の集積度を上げ配線の長さを短くしなければならない。

それに対して、生体の認識処理などは、信号伝搬速度は導線のそれとは比較にならないくらい遅いのにも関わらず、スーパーコンピュータの足元にも及ばないくらいの処理速度である。なぜなら、脳などでは、単純な処理を行う大量の神経細胞が立体的に結合され、大量のデータを分散的、かつ並列的に処理を行っている。

以上の事を考慮すると、現在、計算機が扱うべきデータが大量になってきていることから、ノイマン型計算機の処理能力向上にも限界が見られ、数万もの単純なプロセッサを組み合わせた超並列機構の開発が必要となる。そこで、我々は、Loop Structured Computer (LSC)[1]~[4]の構造を拡張したドーナツ型LSCの提案動機について示し、その構成とプロトタイプについて述べる。

2 ドーナツ型LSCの提案動機

ここではノイマン型計算機の高速化に対する問題点を各項目に分けて記述する。

クロックについて

計算機の処理速度を上げるためには、基本クロック周波数を上げることが必要である。しかし、信号の伝搬速度は光の速度を超えることが出来ない。このために、１クロックの間に信号が進む距離は表１のとおりである。又、導線上で信号は遅れる事を考慮しなければならない。ノイマン型計算機では１クロック内に中央演算装置からメモリにアクセスし、その内容が戻ってこなければならない。これより、単にクロックを上げるならば、中央演算装置とメモリ間の距離を小さくしなければならない。もし、クロックを２倍に上げたならば、回路は1/4 に縮小しなければならない。

また、クロック周波数を高くすると、伝送路の曲がり角や、コネクタの部分ではインピーダンスが異なることが多く、電波として外にでたり、隣接した信号線に影響を及ぼしたり、反射波として信号が逆戻りしたりしてバス上の信号が崩れる恐れがある。従って、高速化を実現するためには、同軸ケーブルや光ファイバ等を用いてこれらを極力抑える必要がある。

バス・ボトルネック

ノイマン型計算機は中央演算装置とメモリが分離しており、その間をアドレスバス、データバス、及び複数本の制御線で結んでいる。信号を受ける側では制御線の中で基本となるストローブ信号に対して、各信号は外れて到着するので、いかに素子が高速であってもクロック周波数を簡単に上げることが出来ない。これらは、バスドライバゲートの動作の不揃いや、各信号線の長さの不揃い、伝送インピーダンスの不揃いなどが原因として考えられる。また実装面積が小さくなればなるほど、バスの長さを揃えることは困難となり、簡単には回路を小さくできない。

一方、バスのデータ転送率を上げるためにバスの信号線数を増やすと、各信号線の長さを揃えることが困難になり、クロックを上げることが出来なくなる。

これらの解決策の１つとしては、バスの信号線の本数を減らして信号線のバスド

Table 1: 周波数と光の進む距離

1クロック(周波数)	進む光の距離	導線上で進む距離
1us(1MHz)	300m	200m - 250m
100ns(10MHz)	30m	20m - 25m
10ns(100MHz)	3m	2m - 250mm
1ns(1GHz)	300mm	200mm - 250mm
100ps(10GHz)	30mm	20mm - 25mm
10ps(100GHz)	3mm	2mm - 2.5mm
1ps(1THz)	0.3mm	0.2mm - 0.25mm

ライバゲートや長さを揃え、データ転送率を上げることが考えられる。極端に言えば、クロック信号と1本の信号線でデータ転送を行えば、ジョセフソン素子の様な高速素子の動作限界でデータ転送を行うことができ、バスによるデータ転送よりも計算機の高速化が期待できる。

一点動作

ノイマン型計算機は、プログラムカウンタで1つの命令機械語を中央演算装置に取り込んで実行する逐次処理である。この為、各部がどんなに速くても、一点しか実行されないことになる。これに対して、人間の脳などの個々の細胞の処理速度は電子回路とは比べようもないくらい遅いにも関わらず、数億もの脳細胞が立体的に並列に処理を行っているので、現在の計算機には比べようもないくらい高い処理能力である。現在の計算機は物理的に限界に達しており、それ故、脳細胞のように数千、数万ものプロセッサを立体的に組合せて並列的に処理を行う超並列処理機構を持つ計算機の開発が必要になる。

並列処理の問題点

逐次処理の限界を考えるとき、脳細胞の様なたくさんのプロセッサをもつ並列処理モデルが提案されてきた。しかし、プロセッサ数増加に伴い、各プロセッサ間の通信の複雑さが増加し、簡単にはプロセッサ数を増加することはできなくなる。また、各プロセッサ間の距離が様々であり、その為プロセッサにメッセージ到着のずれが起こり、同期をとる事も難しくなる。

以上の事を考慮すると、図1に示すように、複数のプロセッサをシフトレジスタでループ状に接続した構成とするLoop Structured Computer となる。それらのシフトレジスタ間のデータ転送を高速にするために、信号の崩れが起こらないように、同軸ケーブル、光ファイバ等で結合し、半導体素子の限界で直列に転送させる。この転送速度は10Gbps~100Gbpsを考えている。プロセッサ0(PE0)は制御部でありホスト計算機との通信の役割を果たすことになる。このプロセッサから各プロセッサへパケットが送られることになり、それらのパケットはシフトレジスタにのせられ1方向にのみ運ばれることになる。その為、プロセッサで受け取られなっかたパケットはLSC上を巡回することになり、再送する必要等がなく、パケット転送などが容易である。このLSCのプロセッサ数として128個、パケットが1周する時間としては約100ns程度のLSCを考えている。

さらに、このLSCを拡張して、図2の様なドーナツ状に結合することを考えると大規模な並列処理が可能となり、興味ある処理が期待できる。このドーナツ型LSCは、LSCを1つの層として立体的に重ね合わせている。LSCではパケットがループ上を一定方向に流れるのに対して、ドーナツ型LSCでは、下層のPEへのパケット転送が中心になる。また、この上層PEと下層PEのパケット転送は、シフトレジスタによる高速転送ではなく、調歩同期等

の通常のデータ転送で良いと考えている。このドーナツ型LSCのプロセッサ数として256層のLSCを用いて、128 * 256 = 32768個のプロセッサを考えている。

1PEの構成は図3の様になっており、各PEはシフトレジスタ上のパケットが自分のPEへのものであれば取り込む。そのPEで処理されたパケットは下層のLSCへ送られる。PEでは2値、または1値演算をデータフロー的に処理する為に、PEの負担は軽くなる。

3 ドーナツ型LSCの構成

LSCは、ループ状に接続されたシフトレジスタに複数のプロセッサを接続したモデルである。複数あるプロセッサの内の一つは制御部であり、ホスト計算機と結合されていて、各プロセッサにホスト計算機からの命令等を送る役目を果たす。各PEへのデータは、全てシフトレジスタを介して行われることになり、また、一方向へしか流れないので、通信が高速で簡単になる。

ドーナツ型LSCは、そのようなLSCを1層として、上下層の各プロセッサを結び、ドーナツ状にしたものである。1PEの構成は図3の様になっている。データの流れは上層と下層の方向が中心となり、データフロー的に演算を行う。各PEはホスト計算機によって機能が決定される。ホスト計算機と各PEの通信は、各層のLSCの通信用PE（LSCの場合制御部）が取り持つことになる。

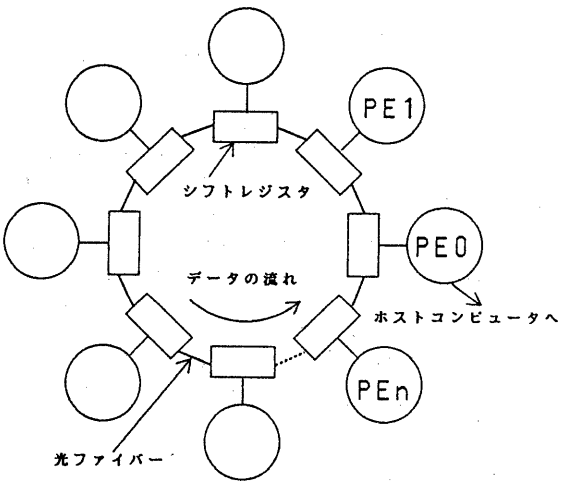


図1 LSCの構成

現在稼働中のプロトタイプは、8つのPE（この内一つは通信用PE）をもったLSCで、8層のLSCから成り立っている。各PEの機能を決定したり、データを送るのは制御部で行う。制御部はホスト計算機とつながっており、ホスト計算機から送られて来る命令によって、演算マッピングパケット、定数データパケット、変数データパケット、演算マッピング解消パケットを通信用PEに送る。各PEは、以下のパケットにもとずいて処理する。以下に、これらのパケットの説明を述べる。

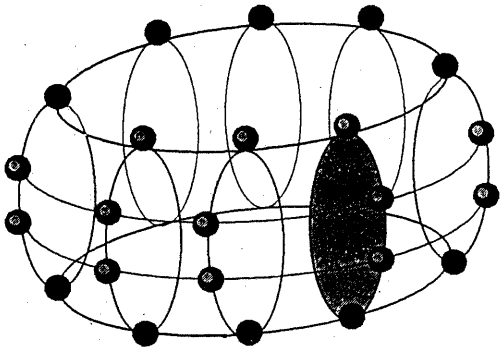


図2 ドーナツ型LSCの構造

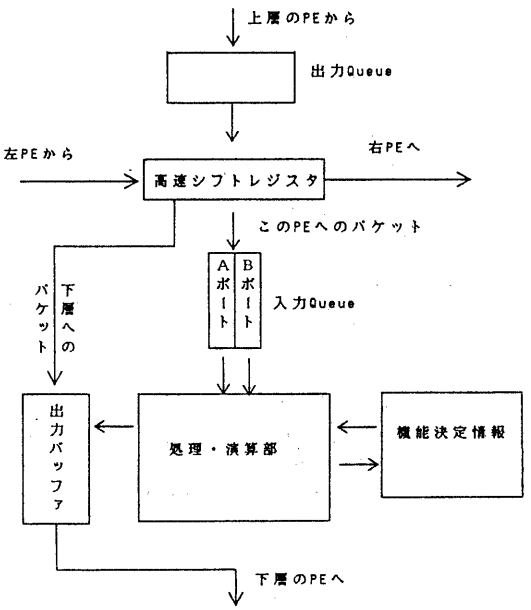


図3 PEの構成

- 演算マッピングパケット

  演算マッピングパケットは、制御部でホスト計算機の命令を受けて生成される、16バイトの情報をもったパケットである。これらの情報は図4に示すように、PEに対する演算の決定、発火条件、出力データ型、出力個数、出力PEアドレス等が記述されている。各PEではこの情報をもとに図5に示すような機能決定情報を作る。

- 演算マッピング解消パケット

  このパケットは図6に示すような構成で、目的PEにマッピングされた様々な情報を無効化するもので、この処理を施されたPEがはじめて空PEとなる。また、機能決定情報も消去される。

- 定数データパケット

  定数データパケットは制御部で生成される16バイトの情報を持つパケットで図7に示すような構成である。これは、演算マッピングパケットによって処理が決定したPEに演算データとしてaポート又はbポートに入る。この定数データはPEの演算を解消するか、新しい定数データが入って来るまでそのポートから消去されないので、大量なデータに同様な処理を施す場合に有効である。また、データナンバーにも左右されないので配列の処理にも適している。

- 変数データパケット

  変数データパケットは制御部や各PEで生成される16バイトの情報を持つパケットであり図8に示すような構成となっている。変数データは、PEで演算データとして処理されると消滅するようにしている。このパケットには目的PEアドレスやデータナンバー等の情報も含まれており配列などの処理にも対応している。また、定数データ、変数データ共に8バイト整数型と8バイト浮動小数点型が用意されている。

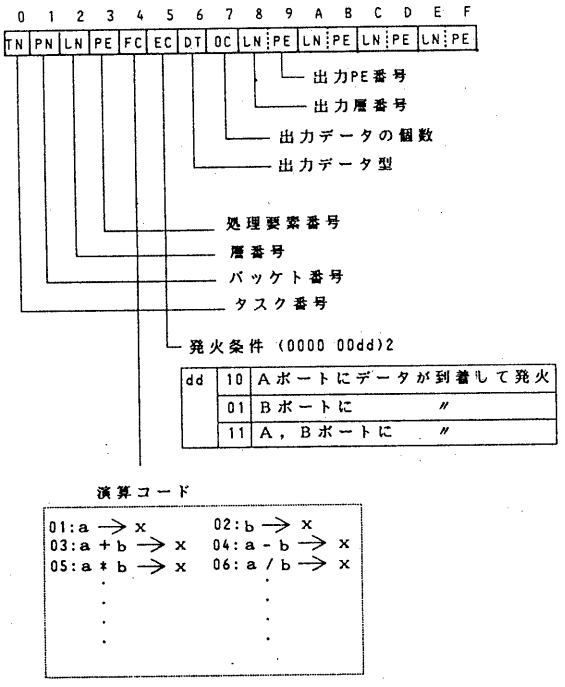


図4　演算マッピングデータ

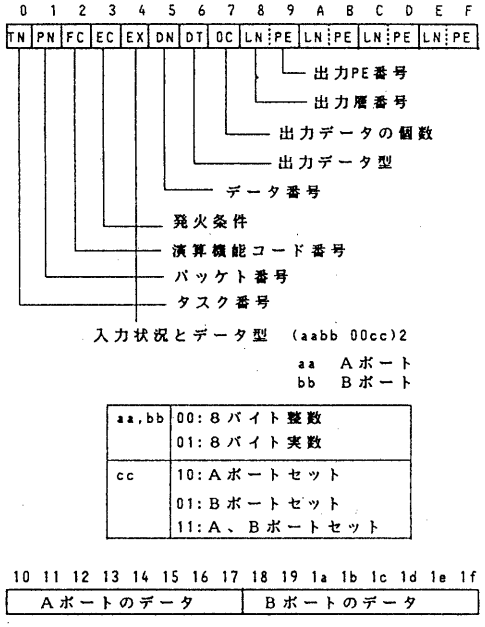


図5　機能決定情報

- 結果データパケット

  このパケットは、上の変数データパケットの内容と等しく、演算マッピングパケットで決定された目的PEアドレスを持ち、出力個数分（最大4つ）の結果を出力する。また、各層の通信用PEに転送することにより、制御部へ結果を返すことが出来る。

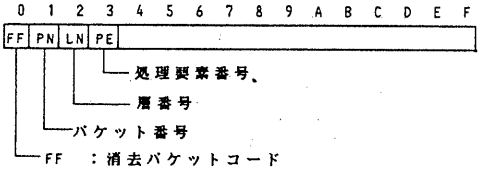


図６　演算マッピング解消パケット

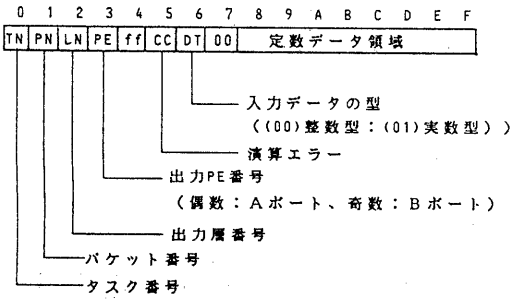


図７　定数データパケット

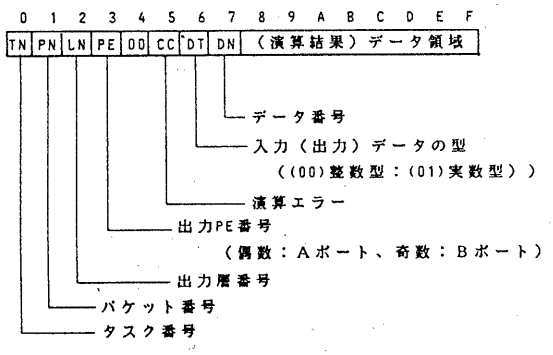


図８　変数データパケット

4　処理手順

ドーナツ型LSCの処理は大まかに言うと、ホスト計算機から制御部へ要求が出され、制御部がそれらの命令を各PEに伝える、と言うものである。(図9参照) そのために、制御部の働きが重要になる。制御部では各種パケットを作るために各PEの状態を把握しなければならない。また、マッピングされていないPEへデータを送るとデータが消滅してしまうので、各PEにマッピングして定めた目的PEアドレスも管理しなければならない。以下は処理手順について示す。

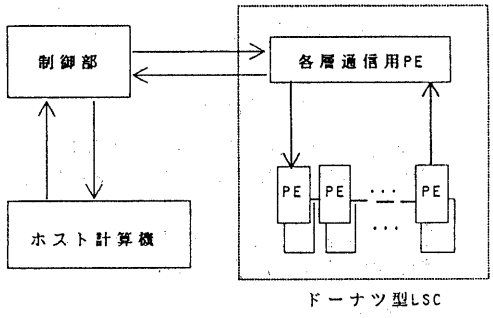


図９　通信の構造

1.　ホスト計算機⟹制御部

ホスト計算機は制御部に対して、プログラムに基づいた各種パケットの生成要求や、ベクトル演算命令等を出す。制御部では管理している各PEの状態を調べ、上の要求に対するパケットを生成する。

2.　制御部⟹各層通信用PE

制御部では、目的PEのある層の通信用PEに、演算マッピングパケットから順に送り出す。但し、出力結果がまだマッピングされていないPEに送られそうなパケットは準備が出来るまで保管するか、その結果を制御部に送り返すようにしなければならない。

3.　通信用PE⟹目的PEへ

通信用PEは受け取ったパケットをシフトレジスタに乗せ、目的PEは自分のパケットならばそれを取り込む。

4.　PE⟹PE

PEに取り込まれ処理されたパケットは消滅し、結果パケットが生成される。こ

のパケットは演算マッピングパケットで指定された目的PEアドレスを持っているので、それを頼りに下層のLSCに乗せられる。また、目的PEアドレスが各層の通信用PEならばそれを通して制御部に返される。

5 ドーナツ型LSCの機能

各PEでは2値、または1値演算を静的データフロー的に行い、非同期に処理する。ドーナツ型LSCの中心になるデータの流れは、上層から下層の方向であるので、図10の様なデータフローグラフのイメージがそのまま利用できる。ドーナツ型LSCの演算の特徴は、データフローグラフの様な、単純な演算過程や、ベクトル演算や、パイプライン的演算に効果的であると思われる。つまり、データパケットにはデータナンバーを利用することにより配列等にも対応できるので、ほんの数回しか使われないような演算をマッピングするよりは、大量にデータをもって、しかもパイプライン的に処理することが出来る命令をマッピングした方が、当然PEの利用率も向上する。

ここでは、プログラム例として図11の流れ図を用いる。この演算過程は各ステージで並列処理が可能であり、一つのステージの演算をそのままLSCの各プロセッサに割り当てることにより、イメージ的にも理解しやすい。さらに処理するデータが大量にあると、パイプライン的に処理することもでき、効率的に動作する。実は、図11の流れ図は高速フーリエ変換(N=8)の実部の演算フローであり、回転因子W^{mn}は$\cos(\frac{k\pi}{4})$を用いたものである。以下、パケット形式のプログラムと、結果データをいくつか図12にあげる。

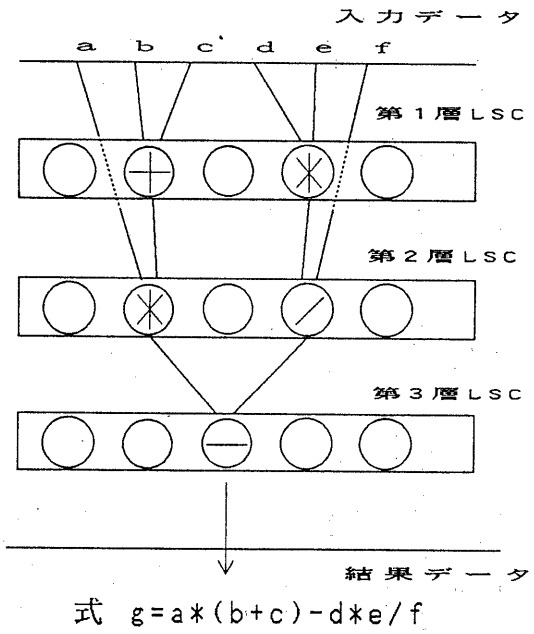


図10 データフローグラフ

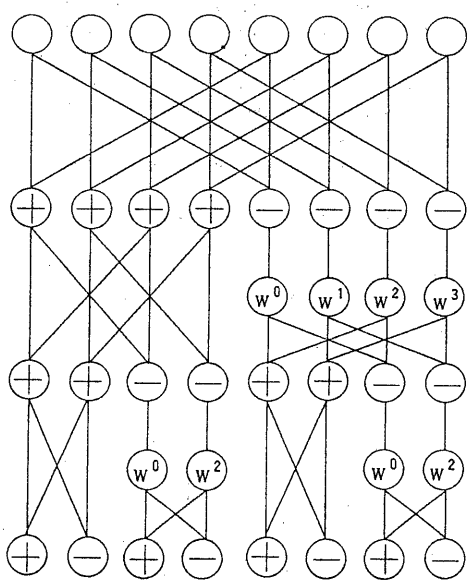


図11 演算流れ図

```
m 1000  0f 0f 00 02 01 02 01 02 02 02 03 02
m 1010  0f 0f 00 04 01 02 01 02 02 04 03 04
m 1020  0f 0f 00 06 01 02 01 02 02 06 03 06
m 1030  0f 0f 00 08 01 02 01 02 02 08 03 08
m 1040  0f 0f 01 02 01 02 01 02 02 03 03 03
m 1050  0f 0f 01 04 01 02 01 02 02 05 03 05
m 1060  0f 0f 01 06 01 02 01 02 02 07 03 07
m 1070  0f 0f 01 08 01 02 01 02 02 09 03 09
m 1080  0f 0f 02 02 03 03 01 02 04 02 04 06
m 1090  0f 0f 02 04 03 03 01 02 04 04 04 08

                . . . . . .

m 12b0  0f 0f 04 0d 00 00 01 00 40 00 00 00 00 00 00 00
m 12c0  0f 0f 05 0b 00 00 01 00 40 00 00 00 00 00 00 00
m 12d0  0f 0f 06 0b 00 00 01 00 40 00 00 00 00 00 00 00
```

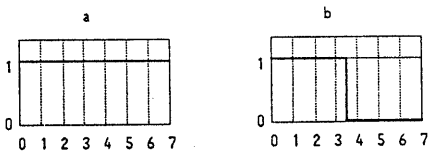
a　パケット形式プログラム

```
m 12e0  0f 0f 00 02 00 00 01 00 40 04 10 00 00 00 00 00
m 12f0  0f 0f 00 04 00 00 01 00 40 04 10 00 00 00 00 00
m 1300  0f 0f 00 06 00 00 01 00 40 04 10 00 00 00 00 00
m 1310  0f 0f 00 08 00 00 01 00 40 04 10 00 00 00 00 00
m 1320  0f 0f 01 02 00 00 01 00 40 04 10 00 00 00 00 00
m 1330  0f 0f 01 04 00 00 01 00 40 04 10 00 00 00 00 00
m 1340  0f 0f 01 06 00 00 01 00 40 04 10 00 00 00 00 00
m 1350  0f 0f 01 08 00 00 01 00 40 04 10 00 00 00 00 00
```

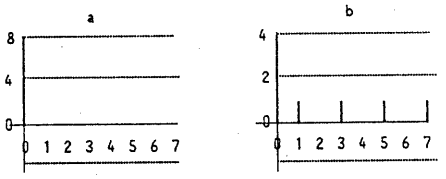
b　入力データ

```
0F0F0000000001004003800000000000 =====  8.000000e+000
0F0F0200000001004000000000000000 =====  0.000000e+000
0F0F0100000001004000000000000000 =====  0.000000e+000
0F0F0400000001004000000000000000 =====  0.000000e+000
0F0F0600000001004000000000000000 =====  0.000000e+000
0F0F0300000001004000000000000000 =====  0.000000e+000
0F0F0700000001004000000000000000 =====  0.000000e+000
0F0F0500000001004000000000000000 =====  0.000000e+000
```

c　結果データの１０進表示



d 入力データ



e 結果データ

図１２　演算プログラムと結果データ

6　おわりに

ノイマン型コンピュータの高速化に対する問題点が数多く指摘されるようになり、これを解決する一つの方策として提案した Loop Structured Computer(LSC) をドーナツ構造にした大規模マルチプロセッサ（ドーナツ型 LSC ）の構成について述べた。次に、これに至るまでの提案動機、このようなシステムのパケット構成や動作手順などのシステム仕様を示した。また、8 層の LSC からなるプロトタイプを作成し、その動作結果を示した。そして良好な結果を得た。

今後、このシステムについての性能評価を行うとともに、高級言語レベルから動作するコンパイラの開発を行う予定である。

参考文献

1 熊沢、吉岡 'Loop Structured Computer について'、情報処理学会・計算機アーキテクチャ研究会資料、56-1 、1985 年 1 月

2 吉岡　良雄 'Loop Structured Computer のトラヒック解析'、電子情報通信学会論文誌 D-I 、Vol.J72-D-I 、3 、pp.149-156 、1989 年 3 月

3 吉岡　良雄 ' 超並列処理機構（ドーナツ型 Loop Structured Computer ）の提案'、平成 3 年度第 1 回情報処理学会東北支部研究会、3-1-1 、1991 年 5 月

4 吉岡　良雄 'Loop Structured Computer の特性解析' 、1989 年　並列処理シンポジウム JSPP　論文集、1989 年 2 月

5 加川　幸雄 ' アナログ・デジタルフィルタ'、科学技術出版社