# 分散形データフロー計算機$D^3C$の構成と評価

浅田邦博，グエン・ニュット，斉藤忠夫，猪瀬博
（東京大学　工学部）

## 1．まえがき

データフロー計算機（以下ＤＦＣ）では，処理を表わすデータフロープログラム（以下ＤＦＰ）が，処理単位である演算子（actor）とそれらの間のデータフロー関係の表現（link）の集合として記述され，それぞれ必要な入力データ（入力トークン）と処理のためのリソースが整い次第，処理が相互に非同期的に実行される「データ駆動原則」に基づく処理形態を特徴としている。これにより自然な形での並列処理が実現されるが，そのため，ＤＦＣは一般に，複数の演算ユニット，命令を格納する複数の命令セル，及び，その両者を結合する結合ユニット等から構成されている。筆者らは先に，これらのＤＦＣに自然に存在する冗長性に着目し，いくつかの付加的機能を与えることにより自律的故障診断機能を有するシステムを構成する手法について提案したが[1]，本報告は，それに基づき構成した分散形データフロー計算機$D^3C$（Distributed Data Driven Computer）のシステム構成を，ハードウェアとソフトウェアの両面から概説したものである。ＤＦＣは，その処理単位であるactorの処理レベルの観点から，「低レベルＤＦＣ」（四則演算等，比較的プリミティブactor集合をもつもの）から「高レベルＤＦＣ」（手続き以上の比較的高度な処理単位のactor集合をもつもの）に分類することが，本システムは，後者に属する。以下，第２節では，システムの基本設計，第３節では，実験システムの実際について述べ，第４節では，システムの性能を表わすいくつかのパラメタの測定結果について報告する。

## 2．システムの基本設計

### 2.1　システム設計条件

ＤＦＣは，その本来の目的である高度の並列処理を達成するための「データ駆動原則」を実現する手段を含む必要があるが，システムの柔軟性・拡張性や，障害に対する予防対策，保守性についても考慮する必要がある。先に提案したＤＦＣ向きの故障診断方式は，図１に示すように有向グラフ（actor，linkをノードとしてもつＤＦＰの記述法の一種）で表現されたＤＦＰをまず二重化し，各入力，出力（中間出力）に「検証actor」を配した「冗長化プログラム」に変換し，ＤＦＣで実行するものである。障害は検証actorによる比較照合により検出されるが，本方式がハードウェアの故障検出に有効に作用するためには，

> 冗長化プログラムの二重化された両ＤＦＰの各対応するプログラム要素（actor，link）が，故障に関して独立なハードウェア・ユニットを用いて処理される。　------（1）

ことが前提条件となっている。（本条件は，両

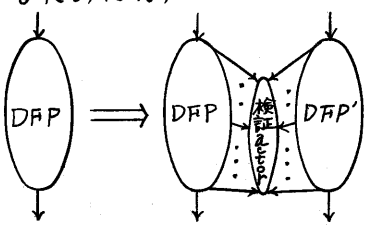


図1　冗長化プログラム変換

ＤＦＰを全く異なるハードウェアで固定的に処理する従来の二重系の形態と異なり，自由度の高いことに注意されたい。）

これらの各要請より，システムを設計する際に考慮すべき条件は，

（ｉ）「データ駆動原則」を効率的に実現可能なシステム構成とする。

（ii）故障に関し，独立性の高い構成要素の集合体として構成する。

（iii）システムの拡張性や，障害からの回復の柔軟性に富むシステム構成とする。

（iv）各プログラム要素とハードウェア・ユニットとの割付けが制御可能な構成とする。

等である。上述の条件の中で（iv）と（ii）は本診断方式に由来するものであるが，（ｉ）～（iii）は一般のＤＦＣ設計にも共通する事項である。また（iv）の「割付制御機能」は（iii）の障害からの回復とも関連する。

## 2.2　割付制御方式

本診断方式を実現するためにＤＦＣに要求される中心的な機能は，2.1 項で述べた「割付制御機能」である。従来のＤＦＣでは，実行可能なプログラム要素を空き状態となっている任意の演算ユニットで実行することにより並列性を高めることに主眼があり，「割付制御機能」を有しないのが普通である。本研究で提案する「割付制御機能」は多くの場合，システムの処理速度にマイナスの効果を与えるが，これによる性能の低下は，従来の二重系等と比較すべきものであり，本方式は，先に述べたように自由度に富むより柔軟な手法となっている。

割付制御機能には大別して次の２つの実現手法が考えられる。

（a）静的割付制御方式：ＤＦＰの実行の前において各プログラム要素を処理するハードウェア・ユニットの集合を定める制御方式。

（b）動的割付制御方式：ＤＦＰの実行とともに，１つのプログラム要素を処理したハードウェア・ユニットに基づき，他の対応するプログラム要素を実行するハードウェア・ユニット集合に制限を課す制御方式。

後者は前者に比較して割付の自由度が高く，2.1 項の（１）の条件を最小限の制約の下で実現するものであるが，実現のためには多くのコストを必要とする。それに対し，前者はやや割付の自由度が低いが，実現が容易な制御方式である。本研究では前者を採用しているが，静的割付制御方式では，各プログラム要素を固定的に１つのハードウェア・ユニットに割付けるのではないことに注意されたい。

図２は静的割付制御方式を表わす概念図であり，割付を表わす Ｚ はプログラム要素（空間）からハードウェア・ユニット（空間）への写像を表わしている。プログラム要素 α の実行に割付けられたハードウェア要素集合を Ｚ(α) とする時，割付に要求される条件は，Ｚ(α)∩Ｚ(α′)＝φ である。ただし，α と α′ は冗長化プログラムにおける対応するプログラム要素とする。

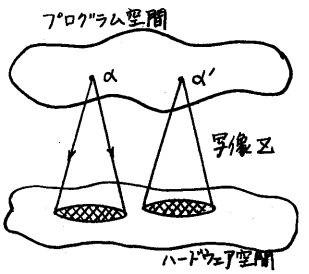



図２ 静的割付制御

## 2.3　分散形ＤＦＣ

本研究では，命令セルの位置により，そこに格納された命令の実行に関与できる演算ユニット集合に制限が加えられる形態のＤＦＣを「分散形ＤＦＣ」と呼ん

でおり，制限のないものを「集中形ＤＦＣ」と定義している。分散形ＤＦＣは，2.2項の静的割付制御方式を実現する１つの構成手法であるが，2.1項で述べた（ii），（iii）の条件とも合致するものである。分散形ＤＦＣを構成する手法には種々のものが考えられるが，本研究では，図３に示すようなネットワーク計算機としての実現形態を提案，検討している。本方式では，個々のＤＦＣ単位としての機能を有するノード計算機をネットワークを介して接続し，全体として１つのＤＦＣシステムを構成するものである。各ノード計算機は，複数の命令セルと複数の演算ユニットを有し，１つの処理プログラムは分割され各ノード計算機の命令セルに分配され実行される。１つの命令セルに格納された命令は，そのノード計算機内の演算ユニットによって処理され，ネットワーク上には，他ノード計算機内の命令セルへ転送すべきトークンが流れ，全体の処理が進行するものである。１つのノード計算機の命令セル数や演算ユニット数は，システムの対象とする処理内容や，処理モード（システムの負荷率等）によって決定される。

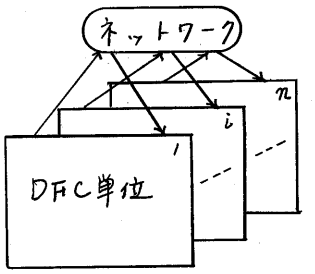



図３ネットワーク計算機としての分散形ＤＦＣ

## 3　実験システムの構成

### 3.1　実験システムの目的

本実験システムの目的は，図３に示すネットワーク計算機としての分散形ＤＦＣの構成法を検討し，その実現性を示し，故障診断方式を実証することであるが，システムの各部の処理速度等のパラメタを決定するためには，システムの対象とする処理内容を定める必要がある。本実験システムでは，手続きレベル，またはそれ以上の比較的高度な actor の集合をもつ中高レベルＤＦＣを目ざしたものである。低レベルＤＦＣをネットワーク計算機として実現することも可能であるが，ネットワーク遅延等の本質的オーバヘッドの比率を小さくする意味で，本方式はより高レベルＤＦＣに適していると考えられる。

### 3.2　実験システムのハードウェア構成

ネットワークシステム：　図４は本実験システムの全体構成を示したものである。本システムでは，拡張性や，中継機能のハードウェア化の容易性を確保するため，ループ状ネットワークを用いている。ループのアクセス制御には今日まで種々の方式が提案されているが（[2]～[4]），実験システムでは可変長メッセージの同期送出可能な Liu 等の提案になるＤＬＣＮ方式による制御を実現している。

システムは，ＤＦＣの単位であるノード計算機がループインタフェースを介しネットワークに接続される構成をとっているが，ノードの１つであるホスト計算機は，通常のミニコンピュータであり，シス

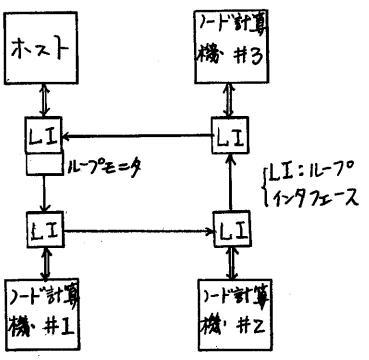



図４　実験システムの全体構成

テムのプログラムロードや，故障診断のための保守センタの働きをするものである。

図5はループインタフェースのブロック図である。ループインタフェースは，「レシーバ」，「トランスミッタ」，及び，「ノードスーパバイザ」回路等から構成され，DMAチャンネルを介してノード計算機の通信用メモリーと接続している。図6はループメッセージ形式を示している。レシーバは，メッセージの目的アドレスを判別し，自ノード宛ての場合には（ノード計算機によって指示される）通信用メモリー内の所定の領域に格納し，他ノード宛ての場合にはトランスミッタへ回送する。この際DLCN方式に従ってレシーバとトランスミッタ間にはFIFO（First In First Out）バッファを設ける必要があるが，本ループインタフェースでは通信用メモリー内の特定領域を用いてFIFOを実現している。トランスミッタは，自ノードからのメッセージと，FIFOを介し回送されてくるメッセージとの競合を制御し，FIFOがあふれる恐れのない場合（送出メッセージ長くFIFOの空き）にのみ自ノードのメッセージ送出を行っている。ノードスーパバイザはレシーバとは異なるアドレスが与えられており，ホスト計算機からの特殊メッセージに応動し，バイパススイッチSWの制御や，ノード計算機のリセット信号の制御を行うものである。ノード計算機の障害時には，この機能によりノード計算機をループネットワークから切り離すことができる。（電磁リレーは電源オフ時のバイパス）

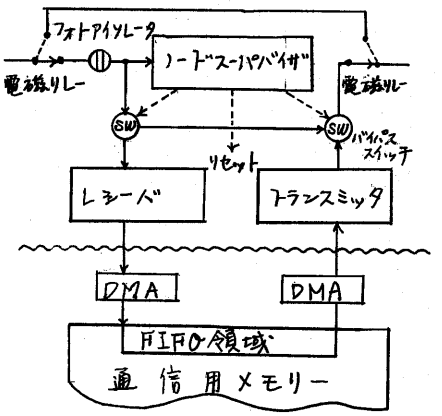
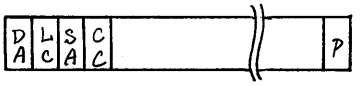



図5　ループインタフェースの構成




DA---目的アドレス
LC---ループカウント
SA---ソースアドレス
CC---制御コード
P---水平パリティ

図6　ループメッセージ形式

| | |
|---|---|
| 伝送速度 | 8Mbps |
| 伝送方式 | シリアル（データ，クロック） |
| アクセス制御 | DLCN方式 |
| メッセージ長 | 3～239バイト（可変長） |
| 制御方式 | マイクロプログラム＋ランダムロジック |

表1　ループインタフェース仕様

なお，ループネットワークに特有の問題として誤ったメッセージがループ上を際限なく巡回することに対処するため，図4に示すようにループモニタが用意されている。ループモニタでは，通過するメッセージの第2フィールドを1ずつ増加しオーバフローの生じたものをループ上からとり除くとともに，そのアドレスフィールドをホストへ通知する。

表1はループインタフェース仕様の主なものである。

ノード計算機システム：　図7はノード計算機の構成要素を示したブロック図である。ノード計算機はそれ自身，1つのDFC単位を構成しており，「汎用演算装置」と「結合制御装置」を中心に構成されている。汎用演算装置はDFCとしての各種プログラム要素（actor, link）の演算処理を行うものであり，本システムは中高レベルDFCとして設計されており，マイクロプロセッサを用いたプログラム制御方式を用いている。必要な演算処理を記述したプログラムをDFPの

実行に先だち，汎用演算装置内の「演算処理プログラムメモリー」へロードする必要がある。結合制御装置はＤＦＰに従いトークンの流れを監理し実行可能となった命令を２本の「命令キュー装置」を介して汎用演算装置へ与える。命令キュー装置が２本用意されているのは，ＤＦＰの実行優先度として２レベルを考えているからであり，詳細については[1]を参照されたい。ＤＦＰをコード化して格納する命令セル装置は，結合制御装置内のメモリーを用いている。結合制御装置もマイクロプロセッサを用いたプログラム制御方式を採用しているが，中高レベルＤＦＣシステムにおいてもハードウェア化，ファームウェア化可能な部分である。結合制御装置，汎用演算装置は共に通信用メモリーを共通メモリーとしてアクセスでき，他ノード計算機とのトークンは本メモリーを介し直接汎用演算装置がアクセスする。命令キュー装置を介して与えられる情報は，処理の種別（オペコード）とデータバッファのアドレス情報である。バッファメモリーはすべて結合制御装置によって監理される。

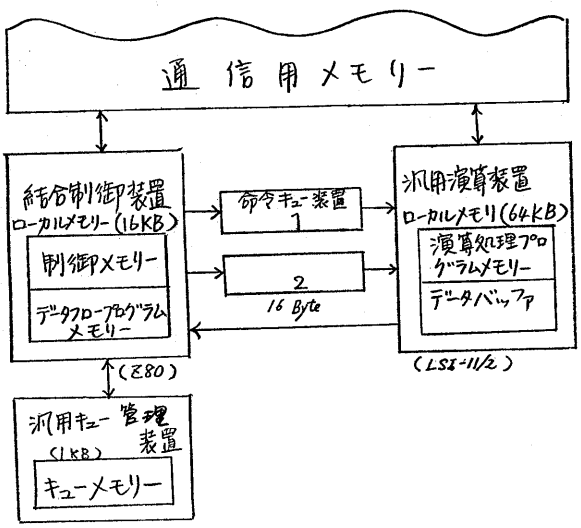



図７　ノード計算機システム構成図

図７の汎用キュー監理装置は，結合制御装置で用いる各種のキュー構造を監理する補助プロセッサである。次項で述べるように本結合制御装置では多数のキューを用いているが，それらはすべて本プロセッサが監理している。本プロセッサでは，複数の可変長キュー構造を適宜定義し使用することができるが，図８はそのための制御ブロック構造を表わしている。１個のキュー毎に１つの制御ブロックが生成される。汎用キュー監理装置はマイクロプログラム制御回路と加減算ハードウェア等から構成されており，結合制御装置からは，

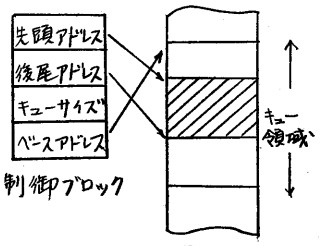



図８　キュー制御構造

（ｉ）キュー定義（初期化）：制御ブロックの作成．
（ii）キューへのセット：キュー後尾へのデータ項目の接続．
（iii）キューからのロード：キュー先頭のデータ項目の取り出し．
（iv）キューのセンス：キュー内のデータ項目数の読み取り．

の四種の命令を用いて使用される。本プロセッサはこれらの各命令を数μ秒で実行する。

| | |
|---|---|
| 汎用演算装置 | LSI 11/2，64KB |
| 結合制御装置 | Z80，20KB(2MHz) |
| 命令キュー装置 | 16B長 |
| 汎用キュー管理装置 | 1KB（最大256個 最大長256B） |
| 通信用メモリ | 4MB/sec，4KB |

B … byte

表２　ノード計算機仕様

表２は，ノード計算機の各構成要素の性能仕様の主なものである。なお本システムでは各ノードで１個の

汎用演算装置を用いているが構成上は複数個接続することが可能である。

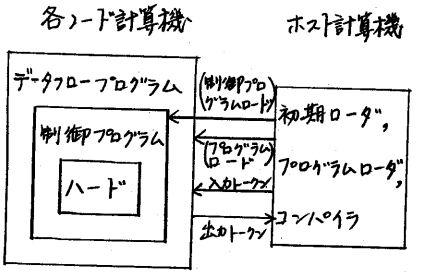



図9 実験システム始動手順

## 3.3 実験システムのソフトウェア構成

システム始動手順: 本実験システムでは，システムの設計目標（中高レベルDFCを分散形システムとして構成する）より，前項で述べたように汎用演算装置や，結合制御装置をプログラム制御方式で実現している。このため，DFPとともに制御プログラム（演算処理プログラム，結合制御プログラム）をロードする必要があるが，本システムではこれをループネットワークを介しホスト計算機よりラインロードしている。図9はこの始動手順を示したものである。各ノード計算機は入出力装置をもたないため，DFPの初期入力，及び，出力結果はホスト計算機で処理している。

演算処理プログラム: 本システムの処理単位は手続き，またはそれ以上のプログラムである。これを演算処理プログラムと呼んでいるが，図10は汎用演算装置内のプログラム構成を示したものであり，図11は本プログラムによって処理される「命令構造」を示したものである。図10の「Fetcher」は命令キュー装置から図11の形式の命令をとり込み，オペコードに従い適当な演算処理プログラムを起動する。演算処理プログラムは'bun code'と'seg code'[5]で記述されたものの2種類があり後者の場合はPASCALインタプリタによって処理される。各処理プログラムでは図11の3つのオペランド（データバッファアドレス）情報を用いて処理が行われるが，各オペランドの入出力属性はオペコードによって定まっている。

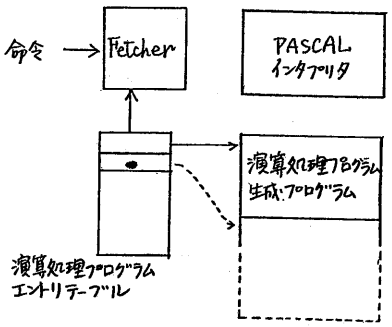



図10 汎用演算装置プログラム

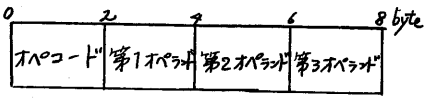



図11 命令構造（3オペランド形式）

演算処理プログラムの中には，他の演算処理プログラムを生成する機能をもつ「演算処理プログラム生成プログラム」が置かれており，演算処理プログラムのロードは，本プログラムへトークンを与えることにより行われる。

結合制御プログラム: 本制御プログラムはトークンの流れを制御し，必要な入力がすべて整い実行可能となった命令を上述の演算処理プログラムへ与えることによりデータ駆動原則による処理を実現するものである。本システムでは演算処理の単位が比較的大きいため，要求される制御速度は比較的低い。図12は本制御プログラムの構成図であるが，制御プログラムは複数のプロセスの集合体として構成されている。各プロセス間の通信は汎用キュー監理装置を用いて構成されるデータキューを用いて行っており，実行プロセス監理にもプロセスキューが用いられている。図13はこれらのプロセスによって処理されるDFPをコード化した命令を格納する「命令セルの構造」を表わしている。DFPはこのような構造を有する複数の命令の集合としてコード化されるが，本システムでは便宜上，そ

れらを小グループ（部分プログラム）に分割し，それを単位として各ノード計算機へ分配している。また制御プログラムは部分プログラム毎の所在情報を表わす「部分プログラムインデックス」を保持しており，部分プログラムの存在するノード計算機番号や，自ノードに存在する場合にはその先頭アドレスを知ることができる。

命令セルには，演算の種別を表わすオペコード，オペランドデータの到着状況や命令の実行プライオリティを表わす制御フィールド，及び，3つのオペランドフィールドが含まれている。各オペランドフィールドには入出力やデータ長の区別とともにそれと接続される命令セルのオペランドアドレス情報が含まれる。これにより，DFPは「両方向リンク構造のグラフ」としてコード化されるが，順方向リンクはデータ（トークン）の伝達に，逆方向リンクはトークン受信可能状態を示す「制御信号」の伝達に用いられる。

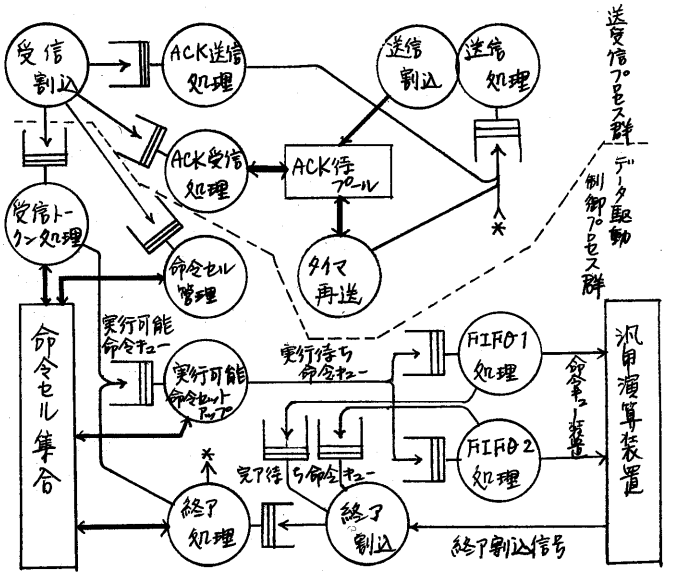



図12 制御プログラム構成図

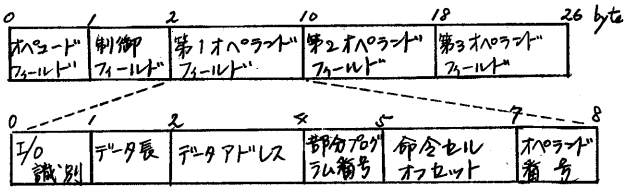



図13 命令セルデータ構造

図12の各プロセスはデータ送受信プロセス群とデータ駆動制御プロセス群の2つに大別できる。前者は通常のACK信号によるデータ伝送制御を行っており，後者はDFCとしての制御機能を実現するものである。主な動作は次の通りである。

（イ）受信トークン処理：他ノード計算機からのトークン，及び制御信号により，命令セルの制御フィールドとデータアドレスフィールドを更新し，実行可能となった命令を「実行可能命令キュー」へ接続する。

（ロ）実行可能命令セットアップ処理：実行可能となった命令に，出力バッファ等を割当てプライオリティに従い2つの「実行待ち命令キュー」へ接続する。

（ハ）FIFO1，FIFO2処理：2本のハードウェア命令キュー装置の延長である。

（ニ）終了割込，終了処理：汎用演算装置からの演算終了割込に応じて，出力トークン，及び，制御信号のルーティングを行い，自ノード宛ての場合には（イ）と同様の処理を行い，他ノード宛ての場合はメッセージを送出する。

（ホ）命令セル監理：本プロセスはデータ駆動制御には直接関係しないホストからの命令セルのロードを司どるプロセスである。

## 4. 性能測定

本節では，本実験システムの性能を決定するいくつかのパラメタの測定結果について報告する。

本システムの性能は主に各ノード計算機の性能によって決定されるが，図7に示されるようにそれは，汎用演算装置の処理速度と，結合制御装置の処理能力によって定まる。前者は処理内容に依存するが，表3は行列演算を例にとった場合の処理速度である。行列は8byteの実数から構成され，処理はPASCALコードで記述している。後者は図12に示される各プロセスの処理速度によって定まるが，表4は主要なプロセスのコードサイズ，処理時間を示したものである。あるプロセスでは他ノード計算機へトークンを送出する場合と，自ノード内の時とで処理時間が異なるため2つの場合について測定している。測定に使用したDFPは図14に示すものである。(各actorは，1入力1出力であるため，一般の3オペランドのactorでは「終了処理」は約1.5倍になると考えられる。) また，これらの総合的性能を表わす「1トークンの伝播時間」は図14のプログラムでは，約1.9msec(ノード内)，4.0msec(ノード間)である。これは本システムのトークン遅延時間の最小値を表わすが，表3の演算能力と比較して良いバランスを得ている。

| 行列演算 | 2×2 | 3×3 | 4×4 |
|---|---|---|---|
| 加算 | 8 | 14 | 26 |
| 乗算 | 38 | 68 | 220 |
| 逆行列 | 43 | 124 | 270 |

表3 汎用演算装置処理速度 (msec)

| プロセス名 | サイズ | 時間(ノード内) | 時間(ノード間) |
|---|---|---|---|
| 終了処理 | 628 | 0.85 | 1.19 |
| 受信トークン | 562 | | 0.38 |
| ACK受信 | 96 | | 0.19 |
| 実行可能命令 | 223 | 0.37 | 0.40 |
| FIFO1 | 64 | 0.14 | 0.14 |
| FIFO2 | 64 | 0.14 | 0.14 |
| ACK送信 | 67 | | 0.14 |
| 送信処理 | 54 | | 0.09 |
| 受信割込 | 108 | | 0.15 |
| 送信割込 | 51 | | 0.13 |
| 終了割込 | 62 | 0.08 | 0.08 |

表4 結合制御プロセス処理速度(msec)とプログラムサイズ (byte)

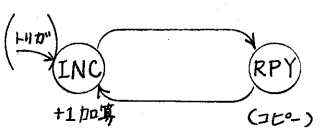



図14 性能測定用DFP

## 5. あとがき

本報告では，先の[1]で提案した故障診断方式を実現するためのいくつかの機能を有する分散形データフロー計算機(D³C)の基本設計方針，実際のシステムのハードウェア構成とソフトウェア構成，及び，いくつかの性能パラメタの測定値について概説した。本システムは，中高レベルのDFCを目ざしたものであるが，性能測定結果に示すように，各部処理能力のバランスの点で一応の目的を達成したものと考えている。なお本報告で触れなかった本システムを用いた故障診断実験については，稿を改めて報告する予定である。

(謝辞：各種制御プログラムの作成に協力戴いた当研究室の安達淳氏に感謝の意を表す。)

(参考文献)[1] 浅田，グエン他「データフロー計算機における故障検出の一方式」，情報処理学会，分散処理システム1-4

[2] Hafner,E.R.,"Digital Communication Loops - A Survey",Zurich Seminar (1974)

[3] Reames,C.C.,and Liu,M.T.,"A Loop Network for Simultaneous Transmission of Variable Length Messages",2nd Annual Symposium on Computer Architecture (1975)

[4] Oh,Y.,and Liu,M.T.,"Interface Design for Distributed Control Network",NTC (1977)

[5] Hansen,P.B.,"Concurrent PASCAL Implementation Notes",Information Science,California Institute of Technology (1976)