分散形データフロー計算機の構成と実験的評価†

大山敬三†† グェン・ニュット††
ビノッド・プラサッド・スレスタ††
斉藤忠夫†† 猪瀬博††

データフロー計算機は，従来のフォンノイマン型計算機にかわり，高度の並列性が要求される情報処理の面で，重要な役割を果たすものと期待されている．データフロー計算機の実現にあたっては，命令セルからの命令を処理エレメントへ転送する裁定回路網と，処理エレメントの出力を命令セルへ転送する分配回路網の，効率のよい構成を見いだす必要がある．本論文では，この点に着目して，分散形データフロー計算機の新しい構成を提案し，縮小モデルの試作を通じて，その実現性を実験的に確認している．提案しているシステムにおいては，裁定回路網と命令セルを処理エレメントに分散配置している．また分配回路網については，可能な諸構成を検討した結果，高速共通バスを採用することとし，処理能力 10 MFLOPS をもつ分散型データフロー計算機のシステム設計を行った．さらにこのシステムの実現性を確認する目的で，縮小実験モデルを試作し，評価した結果，提案システムの実現性の確認ができた．

1. まえがき

電子計算機の歴史において，いわゆるフォンノイマン型の構造は現在まで一貫した思想である．しかし，フォンノイマン型の構造に対しては，ハードウェアの点からも種々の制約が指摘されている．

最近の情報技術は，マイクロエレクトロニクス技術にもとづく大量生産によって急速な価格低減を続けている複雑な論理機械を中心として発展してきている．しかし，いわゆるフォンノイマン型の構造で大型計算機を構成しようとするとマイクロエレクトロニクス技術の量産性の特質を十分享受することはできないといえよう．

たんなる計算能力からすれば，安価なマイクロプロセッサを多数使用するほうが，単一の大型計算機よりも安いコストで計算を実現できることは一般的に理解されている．もちろん，小型計算機を多数集めただけでは，大規模な処理を一体として実行する大型機の能力は実現できない．このため，小型のプロセッサを多数有機的に結合して，大型機に匹敵する能力を実現する技術開発が求められている．

本論文では，小型のプロセッサを用いた演算処理装置を多数使用し，これらを有機的に結合することによって，高度の並列処理を行うことにより数値計算を主対象として 10 MFLOPS 程度の処理能力をもつ，データフロー計算機を実現することを目的として，その構成法を具体的に明らかにするとともに，縮小モデルを試作して，実現性の検証，性能の評価，問題点の抽出を行っている．

2. システムの基本的構成

データフロー計算機の構成方法は基本的に次のような視点から分類できよう．

（1） 集中形構成と分散形構成
（2） アクタの処理レベル
（3） 一様構成と階層構成

従来，データフロー計算機においては，データフロー原則の自然な実現形態として，おもに**図1**のような集中形構成が研究されてきたが[1]，性能に対する考察はあまり行われていなかったといってよい．筆者らは高性能データフロー計算機を実現するという立場でまず集中形構成の検討を行った結果，以下のようなボトルネックが明らかになった．

（1） 裁定回路網における処理装置の選択の高速化が困難である．
（2） 命令メモリへのアクセスの競合が生じ，性能低下の原因となる．
（3） 命令メモリの管理処理が大規模化する．

すなわち，裁定回路網においては，データフローグラフで記述されたプログラムの処理単位であるアクタが実行可能となるたびに動的に処理装置を選択せねば

† System Design of a Distributed Data Flow Computer and Its Experimental Evaluation by KEIZO OYAMA, NHUT NGUYEN, VINOD PRASAD SHRESTHA, TADAO SAITO and HIROSHI INOSE (Department of Electrical Engineering, Faculty of Engineering, University of Tokyo).

†† 東京大学工学部電気工学科

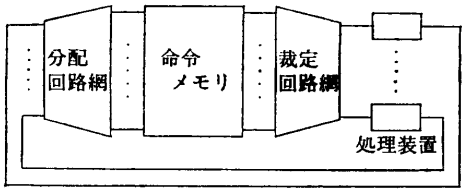

図 1 集中形データフロー計算機の構成
Fig. 1 Structure of a concentrated data flow computer.

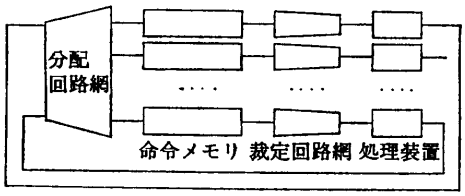

図 2 分散形データフロー計算機の構成
Fig. 2 Structure of a distributed data flow computer.

ならず，非常に多数の処理装置の管理を高速で行う必要があり，高性能化の障害となる．

命令メモリには，分配回路網と裁定回路網からのアクセスが頻繁にあり，競合が起こる．また，大規模なデータフロープログラムを実行するには，命令メモリの容量も非常に大きくなり，したがって，この管理が大規模化し，同種の回路を多数用いて実現することがむずかしくなる．

以上の問題を解決するため，筆者らは先に**図 2** に示すような分散形構成を提案した[2),3)]．すなわち，命令メモリを分割し，そこに記憶されているアクタを実行する処理装置に位置的な制限を加えることにより，裁定回路網の負荷を軽減し，同時に命令メモリの規模を小さく抑える構成である．

データフロー計算機は，対象とするデータフロープログラムの処理単位であるアクタに含まれる処理内容に応じて低，中，高レベルに分類できる．低レベルとは機械語のレベルをいい，高レベルとは通常の手続きやプロセスのレベルをいう．本システムは，マイクロプロセッサを用いるという立場から，四則演算にして数個から数十個程度の処理を単位とする中レベルデータフロー計算機とする．

本システムでは，分散形構成を採用したことによりアクタの割付問題が生ずる．筆者らは先に分配回路網に環状回路網を用いた予備実験システムを構成し[2),3)]，割付問題を検討したが，この場合には隣接した処理装置間では通信を短時間で実行できるが，遠い処理装置間の通信に長時間を要するため，データフロープログラムのアクタの割付が複雑になることが明らかになった．

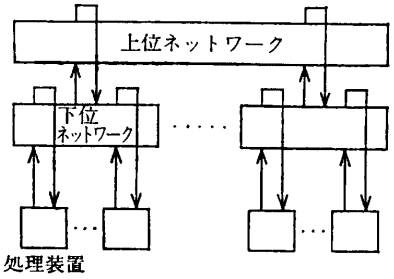

図 3 階層構成のデータフロー計算機
Fig. 3 Data flow computer having hierarchical architecture.

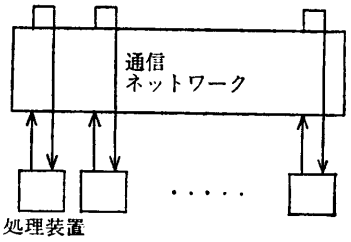

図 4 一様構成のデータフロー計算機
Fig. 4 Data flow computer having uniform architecture.

図 3 に示すような階層性のあるアーキテクチャでもプログラムの局所性を抽出する必要があるため，割付は複雑となる[3),4)]．これに対して**図 4** に示す一様構成をとれば，割付問題は単純な負荷配分の問題に帰着でき，階層構成のデータフロー計算機の場合より，はるかに容易となる．

以上の検討をまとめ，分散形データフロー計算機の利点と欠点を示せば，次のようになる．

利点としては，

（1） 裁定回路網，命令メモリの構成が容易である，

（2） システムの拡張性が得やすい，

（3） ハードウェアの融通性・柔軟性が高い，

（4） 同種の部品，回路を多数用いることで構成が可能である，

（5） 静的割付を行うため，入出力装置等の機能の追加が，全体的構造の変更なしに行える，

等が挙げられる．

一方，欠点としては，

（1） 割付により実行速度が大幅に変化し，割付が重要課題となる，

（2） 割付の動的変更が困難である，

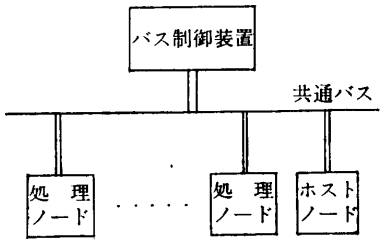

図 5 分散形データフロー計算機の全体構成
Fig. 5 Overall system configuration of the distributed data flow computer.

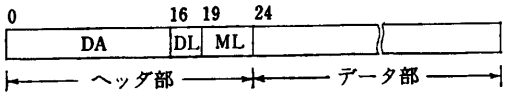

図 6 メッセージ構成
Fig. 6 Message format.

(3) 資源の利用効率が下がる,

等が挙げられる.

マイクロプロセッサのコストは急速に低減しているので,個々の利用効率がある程度低くとも,これらを多数使用した分散形構成をとることにより,システム全体としての性能価格比の向上が期待できる.

以上をまとめると,本システムの全体的構成は**図5**に示すようになる.分配回路網には,データ転送容量,遅延,一様性,拡張性,実現の容易さ等を考慮し,高速の共通バスを使用することとする.

本システムでは 50 kFOPS 程度の処理能力をもつ演算装置を含む処理ノードを 256 台程度用いて 10 MFLOPS の性能を実現することを想定している.そのほかに,データフロープログラムの作成,コンパイル,ロードおよびデータの入出力のために,ホスト計算機として1台のホストノードを置く.

3. システムの詳細な構成

各部のより詳細な構成は次のようになる.

3.1 共通バス構成

分配回路網には,同期式の共通バスを用いる.システムの規模によっては,他の結合回路網を用いたほうが適切な場合もあり,構成に応じて決める必要がある.ノード数 256,全体の性能が 10 MFLOPS 程度では共通バスが最も実現性がある.

バス幅は,データバス 128 本,アドレスバス 8 本とし,その他に必要な制御線数本を張る.

メッセージは**図6**に示すように,ヘッダ部とデータ部とから成る.ヘッダ部はメッセージに関する制御情報であり,DA は目的アクタ番号で 16 ビット,DL は目的リンク番号で 3 ビット,ML はメッセージ長で 5 ビットとする.データ部はオペランドデータを運ぶ部分で,13〜409 バイトである.

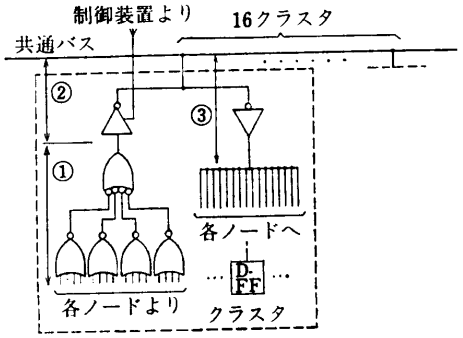

図 7 クラスタ化による共通バス接続
Fig. 7 Common bus connection by means of clustering.

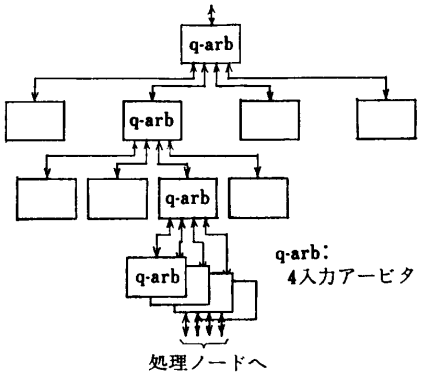

図 8 競合制御回路の構成例
Fig. 8 Example construction of competition control circuit.

共通バスに直接 256 ノードを接続するのは困難だが,たとえば 16 ノードごとにまとめてクラスタ化すれば,直接接続されるのは 16 クラスタとなる.データ線1ビット分に注目すると,図7のような構成となる.この構成でも論理的にはクラスタ化を行わない場合と等価であり,結合の一様性は保たれている.**図7**の回路の信号遅延を実測したところ,①が 7 nsec,②が 11 nsec,③が 9 nsec であった.制御信号遅延は②+③と同程度で,ターンアラウンド遅延は 47 nsec となる.現在,より高速な TTLIC も入力可能であり,50 nsec 程度のバスサイクルも可能である.

これにより,共通バスの通信容量は 320 MB/sec となり,10 MFLOPS の処理を実現するには十分である.

競合制御には,**図8**に示すように4入力アービタを

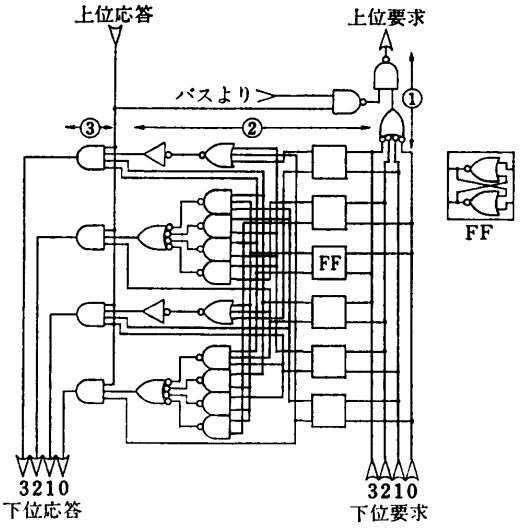

図 9 4入力アービタの回路図
Fig. 9 Circuit diagram of 4-input arbiter.

階層構成にした競合制御回路を用いる．4入力アービタの構成を**図 9** に示す．各要求入力間の発生順序を記憶する RS フリップフロップと，これらの出力から応答出力を作る組合せ論理回路とによって構成される．各部の信号遅延を実測したところ，①が 6 nsec，②が 9 nsec，③が 3 nsec であった．競合制御回路全体の遅延は①×3+②+③×3 で，36 nsec となるので，50 nsec のバスサイクルに対して十分高速である．

3.2 処理ノード構成

図 10 に処理ノードの構成を示す．以下では各部の機能と構成について述べる．

3.2.1 演算装置

演算装置は命令キューから命令を取り出して実行する．命令はアクタの演算内容を指定する命令コードと，そのオペランドのデータメモリ中の所在情報より成る．

命令の実際の内容は演算メモリに格納されており，命令コードで指定された処理を，データメモリに蓄えられたオペランドを参照しながら実行する．処理の実行中に得られた結果データは順次メッセージフォーマッタのバッファに書き込み，処理が終了するとメッセージフォーマッタに知らせる．

これらの機能を実現するために，演算装置には高性能のマイクロプロセッサを用いる．要求される処理能力については 5 章で検討する．オペランドデータはデータメモリから受け取り，結果データはメッセージフォーマッタに渡すため，これらは演算装置の主記憶のアドレス空間の一部として構成する．

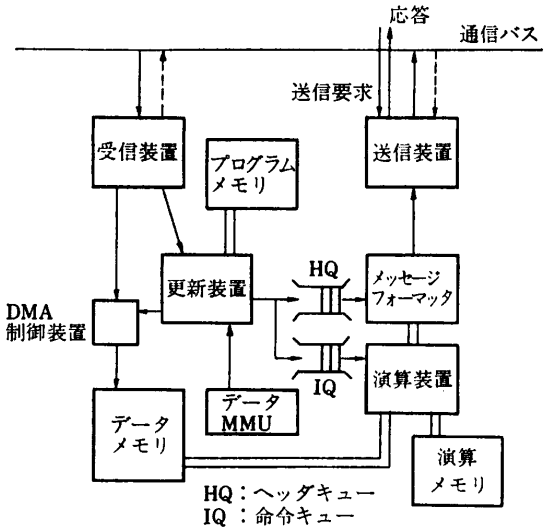

図 10 分散形データフロー計算機の処理ノード構成
Fig. 10 Processing node configuration of the distributed data flow computer.

3.2.2 メッセージフォーマッタ

メッセージフォーマッタは，ヘッダキューからメッセージヘッダを取り出し，バッファに格納された結果データに付加して送信装置へ転送するハードウェアである．

3.3.3 送信装置と受信装置

送信装置はバス送信手順に従い，バス制御装置に要求 (REQ) を出し，応答 (ACK) が返されたらバスにアドレスとデータを送出する．受信装置は，アドレスが自ノードの場合，データを受信し，受信応答 (RPLY) を返し，バスサイクルを終了する．受信可能でないときは受信応答が返されないので，送信装置は再び要求からやり直す．送信および受信装置には，高速のバスと比較的低速の処理装置との間の速度差を吸収するために高速の FIFO を置き，制御もハードウェアにより実現する．

3.3.4 更新装置とプログラムメモリ

更新装置は受信装置からメッセージのヘッダを受け取り，メッセージ長を解析し，データメモリ管理装置 (データ MMU) からデータメモリの空きブロック番号を受け取り，DMA 制御装置にこれらを与えて，受信装置とデータメモリ間の DMA 転送を起動する．次に，ヘッダから目的のアクタとリンク番号を解析し，プログラムメモリに蓄えられた命令セルを更新する．命令セルの構成を**図 11** に示す．タグビットはメッセージの到着を示すフラグであり，全部オンになる

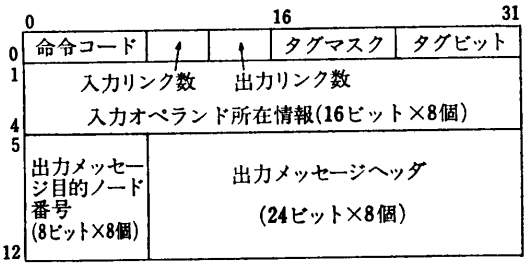

図 11 命令セルの構成
Fig. 11 Instruction cell format.

とこの命令が実行可能となる．タグマスクはタグビットの初期値である．入力オペランド所在情報は到着したメッセージデータを格納したデータメモリのブロック番号である．

更新により実行可能となった場合は，命令コードと入力オペランド所在情報を命令キューに入れ，また，出力メッセージ目的ノード番号と出力メッセージヘッダをヘッダキューに入れる．

これらの機能を実現するために，更新装置には高性能のマイクロプロセッサを用いる．要求される処理能力については5章で検討する．

3.3.5 データメモリ管理装置

データメモリ管理装置は，データメモリを数種類の大きさのブロックに分割管理し，更新装置からの要求により空きブロックを与える処理と，演算処理が終了して不要となったブロックの回収の処理を行う．マイクロプロセッサとハードウェア FIFO により実現する．

3.3.6 データメモリ

データメモリは DMA 制御装置からの書込みと演算装置からの読出しを処理するため，高速スタティック RAM と競合制御回路により，等価的に2ポートメモリとして実現する．

4. 縮小モデルデータフロー計算機 EDAC の構成

本章では，3章で述べた分散形データフロー計算機の構成に従い，この実現性の検証と詳細な検討のために試作した縮小モデルであるデータフロー計算機 EDAC の構成について述べる．

EDAC は，処理ノード2台とし，共通バスのサイクルは 280 nsec，アドレスバス幅は2ビット，データバス幅は 32 ビットの規模とした．したがって，バスの通信容量は提案システムの 0.045 倍となる．

ホスト計算機の OS は，米国ベル研究所開発の UNIX 第6版であり，プログラム開発，システムプログラムの初期ロード，データフロープログラムロード，実行時の入出力制御等に用いている．

表 1 EDAC の共通バスの信号線
Table 1 Signal lines of EDAC's common bus.

種類	信号名	内容
データ	DA 0～DA 31	メッセージヘッダ，データ
アドレス	ADR 0～ADR 1	メッセージ目的ノード番号
制御線	REQ 0～REQ 2	トランスミッタ→バスコントローラ，リクエスト
	ACK 0～ACK 2	バスコントローラ→トランスミッタ，ACK
	DOUT	トランスミッタ→バスコントローラ，データ出力あり
	DIN	バスコントローラ→レシーバ，データ入力
	CK 1～CK 4	制御用クロック
	MS	メッセージスタート
	ME	メッセージストップ
	RS, HRS	リセット

以下に共通バスと処理ノードの構成について述べる．

4.1 共通バス構成

バス線の構成を表1に示す．EDAC ではノード数がホストを含めて合計3台であるので，REQ 線，ACK 線とも3本であるが，ノード数を増やす場合はこれも増える．しかし，N台のノードを一次元的に配置しても，4入力アービタを最適に分散配置すると，REQ と ACK 線の総配線面積は $O(N \log_4 N)$ となり，256 ノードでも問題ない．

メッセージ構成は，図6において DA を9ビット，DL を3ビット，ML を4ビットとし，データ部を2～62バイトとした構成である．

バス制御装置は図9の4入力アービタを縮退させた3入力アービタと，同期のためのクロック発生器より成る．バス送受信手順を図 12 に示す．

送信装置から送信要求 (REQ) が出されると，バス制御装置は，アービタにより選択された一つの送信要求に対し，送信許可の応答 (ACK) を返す．メッセージ送出中は ACK は出し続ける．送信装置は，ACK が返ってくるとメッセージをバスに送出する．受信応答 (RPLY) が返されない場合は送出をやめ，一時 REQ を下げて再び送信手順を開始する．受信装置は，メッセージ開始 (MS) が出されたときにアドレス解析を行い，自ノード宛ての場合は，受信バッファである FIFO に1メッセージ分の空きがあれば RPLY を

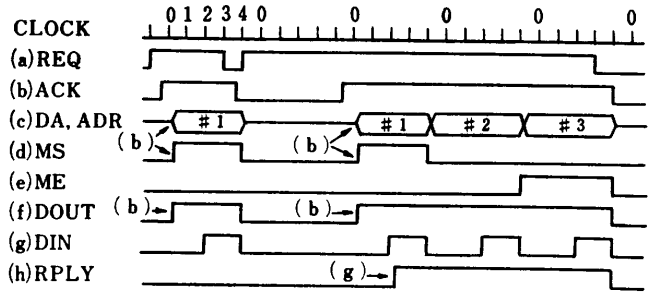

図 12 バス送受信手順
Fig. 12 Send and receive protocol of bus.

返す．メッセージ終了 (ME) が出されるまで RPLY は出し続ける．

この手順により，非同期に発生したメッセージを，任意の目的ノードへ相互の衝突なしに転送することが可能となる．

4.2 処理ノード構成

EDAC の処理ノードの構成は，図 10 に示した提案システムの処理ノード構成において，データ MMU を取り除き，その処理を更新装置に含めた構成であり，その他については，基本的構成は等しい．

更新装置には DEC 社の 16 ビットマイクロプロセッサ LSI 11/2，演算装置には同じく LSI 11/23 を用い，その他は LSI, MSI, SSI によるランダムロジックで実現している．

演算装置の処理の流れを**図 13** に示す．これは命令キューから命令コード，オペランド所在情報を取り出し，メッセージフォーマッタが入力可能であると，命令実行を開始する．終了すると，メッセージフォーマッタに終了信号を与え，次の命令実行に移る．試作システムのため，システムプログラムの保護機能等はない．

メッセージフォーマッタではハードウェア制御によるダブルバッファリングを行っており，演算装置からの書込みと送信装置への転送が並行して実行できる．

更新装置の処理の流れを**図 14** に示す．メッセージ受信処理は到着したメッセージの先頭の 1 ワード (16 ビット) を占めるヘッダを取り込み，メッセージ長を解析し，データメモリの空きブロックを確保して，DMA 制御装置を起動し，メッセージデータの格納を行い，次に，ヘッダから目的のアクタ番号とアクタのオペランドに対応するリンクの番号を解析し，プログラムメモリ内の命令セルを更新する．更新によりその命令が実行可能となると，実行可能命令処理を行う．この処理では，命令の種類を示す命令コードと，その実行に必要なオペランドデータの所在情報を命令キューに入れ，命令の演算結果をメッセージに組み立てるために必要なヘッダをヘッダキューに入力する．

命令セルは，図 11 において，入力オペランド所在情報を 8 ビット × 8，出力メッセージ目的ノード番号を 2 ビット × 8，出力メッセージヘッダを 8 ビット × 8 とした構成である．データメモリ管理処理で扱うブロックは固定長とした．

4.3 システムソフトウェア構成

これまで，EDAC の実行時の動作を中心にその構成を述べてきた．この構成において，データフロー計算機を実現するためには，更新装置と演算装置に制御のための制御プログラムが必要である．また，ホストノードにおいては，処理ノードにおいてハードウェア

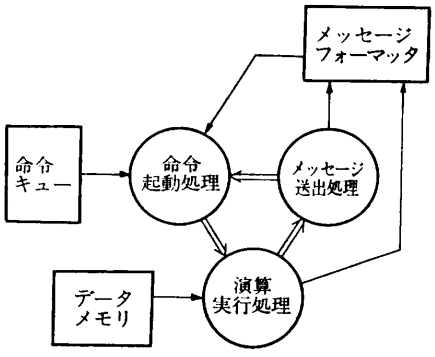

図 13 演算装置制御プログラムの構成
Fig. 13 Configuration of operation unit control program.

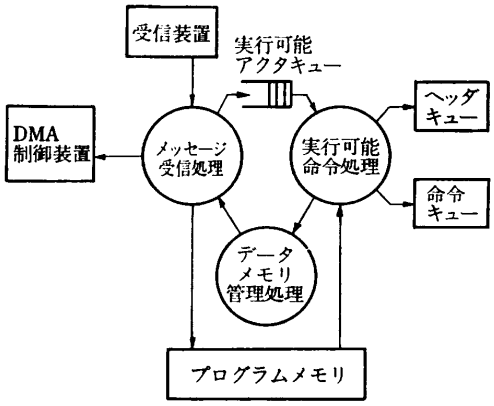

図 14 更新装置制御プログラムの構成
Fig. 14 Configuration of updater control program.

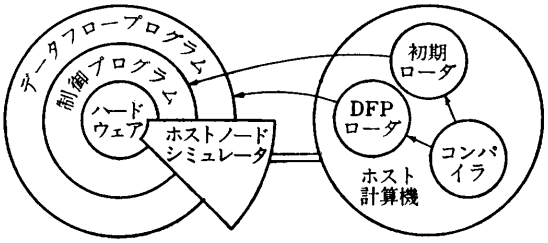

図 15 EDAC のソフトウェア構成
Fig. 15 Software configuration of EDAC.

で実現している処理を代行するソフトウェアのシミュレータが必要である．ホスト計算機としても，その他にいくつかのプログラムが必要である．

これらのソフトウェアの全体構成を**図 15** に示す．制御プログラムはホスト計算機の外部記憶装置に格納されており，システムの起動時に初期ローダにより通信バスを介して処理ノードへロードする．実用システムではこれらの制御プログラムは ROM 化して処理ノード中に常駐させるべきであるが，EDAC は実験システムであり，制御方式の変更に伴うプログラムの変更等が容易に行えるよう，このような方式を採用した．

EDAC 上で実際にデータフロープログラムを実行するには，ホスト計算機のコンパイラでコンパイルされた機械語表現のデータフロープログラムを処理ノードへロードする必要があり，これも DFP ローダにより通信バスを介して行っている．

5. EDAC の評価と提案システムの実現性の検討

4 章に述べた構成により EDAC を試作した．このシステムにおいて，演算装置と更新装置のシステムソフトウェアのオーバヘッドを測定した結果，演算装置の命令起動処理とメッセージ送出処理の合計の処理時間は 0.25 msec，また，更新装置の実行可能命令処理時間は 0.4 msec，1 メッセージ当りの受信処理時間は 0.85 msec であった．

数値計算等を考えると，1 命令当りの入力メッセージ数は 3 程度となることが多く，更新装置と演算装置の負荷のバランスをとるためには，演算装置の全処理の和が，更新装置の実行可能命令処理と受信処理の合計の処理時間，すなわち $0.4+0.85\times3\simeq3$ msec と等しくなることが望ましく，1 アクタ当りの演算時間は約 2.7 msec とすればよい．これは，演算装置の処理能力からすると，アクタの処理内容を約20浮動小数演算程度とするのが最も効率のよいアクタのレベルとなることを示す．

表 2 メッセージ受信処理ステップ数
Table 2 Number of steps in message receiving process.

処理内容	命令ステップ数
初期化，後処理	9
ヘッダ読込・解析	5
空ブロック確保	7
DMA セット	6
命令セル更新	4
実行可能アクタ処理	7
合　　計	38

実際の数値計算等を考えると，アクタのレベルを 5 ないし 10 浮動小数演算とするのがよい場合が多いと思われる．

EDAC で演算装置と更新装置に用いたマイクロプロセッサは，基本的な命令の実行時間が，それぞれ約 3 μsec と 8 μsec であり，上記の結果を考え合わせると，両プロセッサは同程度の処理能力をもつべきである．

提案システムの実現性を考えると，256 台の処理ノードで 10 MFLOPS を実現するためには，各演算装置の 1 浮動小数演算実行時間は 20 μsec 程度以内でなければならないから，1 アクタ当りの演算時間は約 200 μsec となる．これは EDAC の場合の 10 分の 1 以下であり，したがって，この実現には，処理能力を演算装置で約 3 倍，更新装置で約 13 倍にする必要がある．

演算装置については，図 10 において，命令キューを命令コードと各オペランド分をすべて並列に置くことにより，3 倍程度の処理能力の改善が期待できよう．したがって，演算装置の基本的処理能力は変えずに，浮動小数演算回路部だけを高速化することで対処できる．

更新装置については，演算装置と同じプロセッサを用いると，約 2.5 倍の性能となる．また，EDAC のシステムソフトウェアは C 言語で記述しているが，これを最適化することにより 40% 程度の改善が期待できる．これにより，実行可能命令処理時間が 100 μsec，受信処理時間が 200 μsec 程度となり，入力メッセージ数を 3 とした場合の 1 アクタ当りの更新装置の処理時間は 700 μsec で，さらに 3.5 倍の性能改善が必要である．

EDAC の受信処理の内容と処理ステップ数の内訳

を表2に示す．データメモリ管理機能のハードウェア化と，実行可能命令処理の機能分散化により，初期化，後処理，空ブロック確保が不要となり，さらにヘッダ解析と DMA 起動を組み合わせてハードウェア化することにより，メッセージ受信処理の処理量を3分の1程度にすることができるから，入力リンク数を3とすると，提案システムに要求される処理能力を得ることができる．

以上，縮小モデルの試作データフロー計算機による処理ノードの構成に関する評価を通じて提案システムの実現性に関する検討を行った．

通信バス，およびシステム全体の評価については，別に計算機シミュレーションによって行っており，目標性能の実現性の確認をしている．その詳細については別途，ご報告する予定である．

6. むすび

データフロー計算機の分類を行い，諸形態の得失を検討し，その結果にもとづき，10 MFLOPS の演算能力を実現するための分散形・中レベル・一様構成に基づくシステムを提案し，その全体構成を示した．本システムにおいては結合回路網に共通バスを採用し，処理ノードを多数置くことにより並列処理を行っている．また，処理ノード内の各種処理をパイプライン化し，処理能力の向上を図っている．

この構成に基づき，提案システムの実現性の検証のため，縮小モデルの試作データフロー計算機 EDAC を試作し，性能評価を行った．この結果，EDAC では小規模の試作モデルであるため更新装置の受信処理がボトルネックとなっていることが見いだされたが，機能分散とハードウェア化により提案システムの実現が可能であることは明らかになった．

謝辞 なお本研究は文部省科学研究費補助金一般研究A「分散形データフロー計算機の構成に関する研究」の補助を得て行われたものであり，ここに記して感謝の意を表す．

参考文献

1) Dennis, J. B. and Misunas, D. P.: A Preliminary Architecture for a Basic Data-flow Processor, *Proc. 2nd Annu. Symp. on Comput. Archit.*, pp. 126-132 (Jan. 1975).
2) 浅田，グェン，堀，斉藤，猪瀬：データフロー計算機における故障検出の一方式，情処学会分散処理システム研究会資料，1-4, pp. 1-10 (1979).
3) グェン，浅田，大山，斉藤，猪瀬：環状回路網を用いた分散型データフロー計算機の一方式，信学論，Vol. J65-D, No. 12, pp. 1528-1535 (1982).
4) Gostelow, K. P. and Thomas, R. E.: Performance of a Simulated Data-flow Computer, *IEEE*, Vol. C-29, No. 10, pp. 905-919 (1980).

(昭和58年3月11日受付)
(昭和58年7月19日採録)