計算機アーキテクチャ 105－8
ハイパフォーマンスコンピューティング 50－8
（1994．3．10）

ロボット制御のための並列処理システム開発

北垣高成* 末広尚士**

* 電子技術総合研究所
〒305 つくば市梅園 1-1-4

** 新情報処理開発機構
〒305 つくば市竹園 1-6-1 つくば三井ビル 17F

あらまし: 多数台トランスピュータを用いて構築された，実時間での状況認識を実現することの可能な拡張性の高いセンサベースド・ロボットマニピュレーションシステムのプロトタイプの概要を紹介する．

和文キーワード: ロボットシステム，マルチプロセッサ，並列処理システム，システムアーキテクチャ，センサデータ処理，トランスピュータ

Development of a Parallel Processing System for Robot Control

Kosei KITAGAKI* and Takashi SUEHIRO**

*Electrotechnical Laboratory
1-1-4 Umezono, Tsukuba 305, JAPAN
E-mail: kitagaki@etl.go.jp

**Real World Computing Partnership
Tsukuba Mitsui Building 17F, 1-6-1 Takezono, Tsukuba 305, JAPAN
E-mail: suehiro@trc.rwcp.or.jp

Abstract: In order to improve the ability of a robot by implement of new sensors and new processing techniques, the robot system which can be expanded without missing its real-time performance is required. This paper introduces an architecture for such robot system in which the recognition of the task conditions in real-time, namely real-time monitoring, can be available. Some problems on the developing process for a prototype system are briefly introduced. The system is a parallel processing system with transputers for a robot manipulator.

Keywords: Robot system, Multi-processor, Parallel processing system, System architecture, Sensor data processing, Transputer

1 はじめに

ロボットの機能を向上させ，より高度な作業を遂行させるためには，新たなセンサや処理手法のインプリメントは有効な手段のひとつである．センサ情報の統融合に関する研究は盛んであり[1]，今後次々と新たなセンサやセンシング手法，センシングデータ処理手法が提案されると考えられる．それらをロボットシステムに組み込むためには拡張性が必要となるが，加えて，ロボットにとってリアルタイム性は無視できない点であり[2]，この両者を兼ね備えたシステムが要求される．しかし，これらの要求を満たすようなシステム構築のための実際的な方法はあまり提案されていなかった．

筆者らはこれまで，実時間での状況認識，すなわちリアルタイムモニタリングを実現することの可能な拡張性の高いセンサベースド・ロボットマニピュレーションシステムのアーキテクチャを提案し，そのプロトタイプを構築してきた[3]．本発表ではその概要を紹介する．

第2章では一般的なロボットマニピュレーションシステムについて簡単に触れる．第3章では構築中のシステムの設計基本方針について述べる．そして，構築されたシステム概要を第4章で示す．

2 マニピュレーションシステムとは

マニピュレータとはロボットの腕であり，マニピュレータを制御するためのシステムをマニピュレーションシステムと言う．

2.1 ハードウェア

基本的なハードウェア構成図をFigure 1に示す．センサからのデータはセンサコントローラ，A/Dコンバータなどの汎用インターフェースを介してコンピュータに取り込まれる．また，マニピュレータはD/Aコンバータなどの汎用インターフェース，ドライバを介してアクチュエータに出力することにより駆動される．当然のことではあるが，このような制御システムでは処理ループがコンピュータ内だけで閉じていないため，インターフェースの変換速度やドライバの特性もシステムの性能に関係する．

2.2 制御の階層構造

上位のプランニングシステムまたは人間からの指令によりロボットを制御するためのシステムは，Figure 2に示されるようないくつかの階層により構成される[4]．

(1) サーボレベル

関節角度を読み込み，モータ出力を決定するといったフィードバックを行なう．数 ms 程度の周期の制御ループを構成している．通常，フィードバックデータ長は固定であることが多く，データ量も高々数百バイト程度であるが，それらのデータはサーボサイクルに同期してアクセスされる．

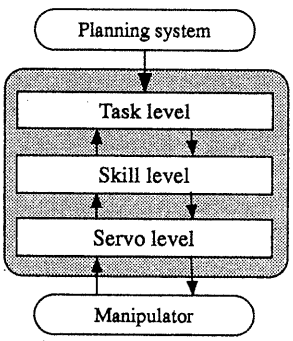

Figure 2 Robot control layer

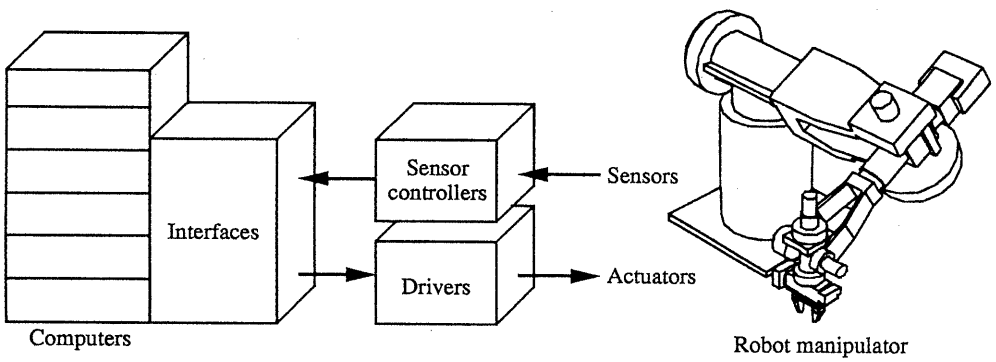

Figure 1 Hardware system for robot control

(2) スキルレベル

目的とする特定の状態を達成するための動作制御を行なう．センサの情報をもとに動作状態を検出し動作モードを決定する．数十 ms 程度の周期の制御ループを構成している．1つのスキルは数秒程度持続する．ただし，動作状態の検出はさらに短い周期で連続的に行なわれる必要がある．

(3) タスクレベル

作業環境，作業目標を認識しスキル動作を組み合わせることにより作業を実行する．

どのレベルにおいてもリアルタイム性が要求されるが，それぞれ処理時間の拘束は異なる．したがって，ロボットシステムを構築する際には，レベルに応じて高速化しなければならない処理とそうでないものがあることを考慮する必要がある．

3 システム構築の基本方針

ロボットの高機能化を実現するためには，新たなセンサやセンサ情報処理手法をインプリメントし，作業状況を確実に認識する必要がある．しかし，従来のロボットシステム，とくに市販のシステムはその構造が固定化されており，予め準備されている機能に加え，新たなる機能をインプリメントすることが難しい．

3.1 リアルタイム性と拡張性

ロボットシステムに要求される点はリアルタイム性と拡張性以外にも考えられるが，ここではリアルタイムモニタリングという視点からとくに重要と考えられる2点にしぼり考察を行なう．簡単にいえば，ここでの問題は「ターンアラウンドタイムを変化させずに拡張性を持たせるには，どのような点に考慮してシステムを構築すれば良いか」ということである．

拡張性を考えた場合，新たなセンサを容易に組み込むことができるかということに加えて新たなセンシングデータ処理法を組み込むことができるかという点も考慮する必要がある．さらに，それらのデータを用いて何らかの出力を生成するプロセスをも新たに組み込むことを考慮する必要がある．このように，新たなセンサあるいはセンシングデータ処理法を組み込むためには，システム全体を拡張することが必要となる．

このような拡張性とリアルタイム性を満足する手法としてシステムの分散並列化によるモジュール化が注目されており，いくつかの事例が発表されている．とくにセンシングデータはパイプライン処理を行なうことの可能なものが多く，並列処理を行なうことによりリアルタイム性を高めることができる．また，それまでのシステムの性能を損なうことなく新たなセンサ処理手法等をシステムに組み込むことができる点で分散並列化は有利と考えられる．

しかし，単にモジュール化すればよいというものではなく，モジュール間の結合度を十分考慮した上でシステムをモジュール化しないと拡張性，リアルタイム性の低下を招くことがある．リアルタイムモニタリングを実現しようとする場合，センシングデータ取得とその処理，動作の決定までのプロセスの結合は密であることに注意する必要がある．

また，内界センサ情報は位置・力制御などのフィードバック制御に用いられるだけではなく，状況認識などの上位レベルの処理にも用いられる可能性が高い点にも考慮する必要がある．従来のようにサーボシステムだけが内界センサデータを取得している場合には，外界センサのデータはサーボと独立に取得できるにもかかわらず，内界センサのデータのサンプリング時間はサーボに依存してしまい，それより短いサンプリング時間での処理を行なうことはできない．つまり，内界センサと外界センサの区別なくシステムを構築しないと拡張性の低下を招く可能性が生じると考えられる．

3.2 データ処理の流れのモデル

プロセス間の結合の度合いを確認し，システムの分散並列化を検討するための，センサ信号や目標値の入力からモータなどアクチュエータへの出力へのデータ処理の流れのモデルを示す．

システムへの入力情報はいくつかのプロセスを経て出力される．ここでは5つにプロセスを分類し，各プロセスとそれらの入出力の特徴について述べる．

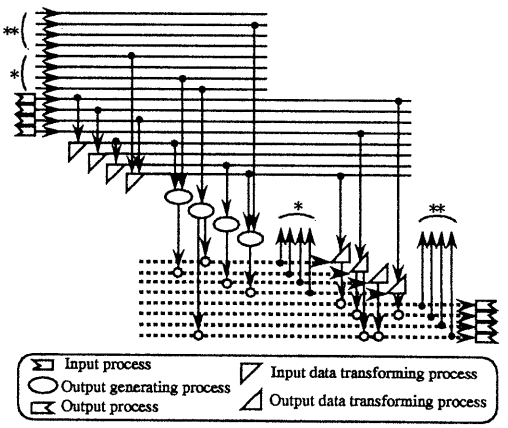

Figure 3 Flow model of data processing

Figure 3に提案するモデルを示す．図中，データは左から右へ，上から下へと流れる．ただし，ここではセンシングデータの流れに注目するため，各プロセスにおけるパラメタの変更やプロセス自体の制御を行なうためのデータの流れは省略した．それぞれのプロセスを以下に示す．

(1) 入力プロセス:I/O を介してエンコーダ，ポテンショメータや力覚センサ，視覚センサなどの内界，外界センサからのセンシングデータを得る．システムに対して与えられる目標値も含まれる．この段階ではフィルタリングなど，入力データ情報量が減少するような処理は行なわない．

(2) 入力情報変換プロセス:入力プロセスからのデータおよび (4), (5) のプロセスへのデータ (Figure1 中 *印，**印) を (3), (4) で利用可能な形式にするための処理を行なう．センシングデータのフィルタリングや座標変換などの下位レベルの処理から，接触状態遷移の検出など上位レベルの処理まで含まれる．上位レベルの処理は下位レベル処理結果を用いることができる．ロジカルセンサ[5]や仮想センサ[6]はこのレベルで実現することができる．

(3) 出力生成プロセス:(1),(2) からのデータおよび (4),(5) のプロセスへのデータ (Figure 3 中 *印，**印) をもとに上位レベルの動作決定を行ない，結果を出力情報変換プロセスや直接出力プロセスに出力する．スキル[7]はこのレベルのプロセスとして位置付けられる．

(4) 出力情報変換プロセス:(1)-(3) からのデータを前処理し (5) へ出力する．座標変換他，フィードバック制御もこの段階のプロセスに含まれる．

(5) 出力プロセス:(3),(4) からのデータに演算を施し I/O を介してモータなどにデータを出力する．

各プロセスは拡張される可能性があるため，インプリメント時にはそれぞれの入出力関係すべてが決定済みとは限らない．例えば，(1)のプロセスへの入力はそれぞれ決められた入力装置（センサ）からのデータ以外入力される可能性はない．一方，出力は(2)-(4)のプロセスからアクセスされることになっているが，(2)-(4)のプロセスは拡張の可能性があるためインプリメント後に新たに生成したプロセスからアクセスされる可能性がある．まとめると，(1)のプロセスのインプリメント時にその入力先は決定済すなわち固定であるが，出力先はすべてが決定されているとは限らないということになる．各プロセスの入出力先がインプリメント時に固定であるか不定どうかを Table 1 に示す．拡張性を持たせるために考慮すべき点は，未知の出力先や入力先に対してターンアラウンドタイムを変化させずに結合できるようにあらかじめ準備しておくという点にある．

Table 1 Feature of input and output for each processes at their implement

processes	input	output
input process	fix	non-fix
input data transforming process	fix	non-fix
output generating process	fix	fix
output data transforming process	non-fix	fix
output process	non-fix	fix

4 プロトタイプシステム

構築中のプロトタイプシステムについて述べる．

ハードウェアシステムは，手先にグリッパの装備された6自由度ダイレクトドライブマニピュレータ (ETA3s)，力覚センサ (ビー・エル・オートテック社:5/50)，ホストコンピュータ (SUN 社: SPARCstation2)，トランスピュータ (INMOS 社:T800,T801,T805) 複数台 (現在 50 台程度)，およびオペレータ用モニタ (NEC 社:PC9801DA) から構成される．制御基本部はすべてトランスピュータで構成されている．本システムでは，ビジョンセンサなど大量の情報を扱うセンサを拡張装備することを考慮し，それぞれのセンサ情報を流すためのラインを独立に確保するためにトランスピュータを採用した．1つのトランスピュータは4本のシリアルリンクを持ち，リンク同士をつなぐことで容易にプロセッサの数を増やすことができる．プロセッサ間の通信はシリアルリンク (10 または 20Mbit/s) を介して行なわれる[8]．システムは Occam2[9] と ANSI C[10]を混在させて記述されているが，とくに通信部分は Occam2，計算部分は ANSIC を中心に記述されている．アームサーボは作業座標サーボ[11]を採用しており，この部分は文献[12]を参考に構成されている．

コンピュータシステムの概略構成図を Figure 4 に示す．ただし，見易さのため機能ごとにいくつかのプロセスをまとめ，オペレータ用モニタを省略している．本システムでは 2 種類のデータ伝送方法を採用している．ひとつは汎用的なルーティングを確保するためルータソフトウェア[13]を利用する方法，もうひとつ

はハードウェアリンクを直接用いる方法である．Figure 4において，コマンドインタプリタ同士を結んでいる上部の線は前者，下部の太線は後者である．ルータソフトウェアを用いてもかなり高速に通信することが可能であるが，サーボやセンサ情報処理などとくに高速性を要求される処理に関しては専用に独立した伝送ラインが確保されている．こうすることにより，内界外界センサのデータを新たに組み込むプロセスからサーボと独立にアクセスすることが可能となる．

ホストコンピュータ上ではEuslisp[14]がインプリメントされており，トランスピュータネットワークとの通信関数が定義されている．ユーザはlisp環境でトランスピュータネットワークにアクセスすることができ，コマンドインタープリタに対してコマンド(オペコード)を転送することによりマニピュレータを制御することができる．オペコードはコマンドインタープリタで解析され，さらにアーム，グリッパ等それぞれのローカルのコマンドインタープリタへ転送される．具体的なトランスピュータネットワーク図をFigure 5に示す．

5 まとめ

高技能マニピュレーションのための拡張性の高いセンサベースト・ロボットシステムのプロトタイプシステムを紹介した．今後，定量的なシステムの評価基準について検討する予定である．また，上位レベルのセンシングプロセスやスキルを拡張することにより，システムの有効性を確認する予定である．

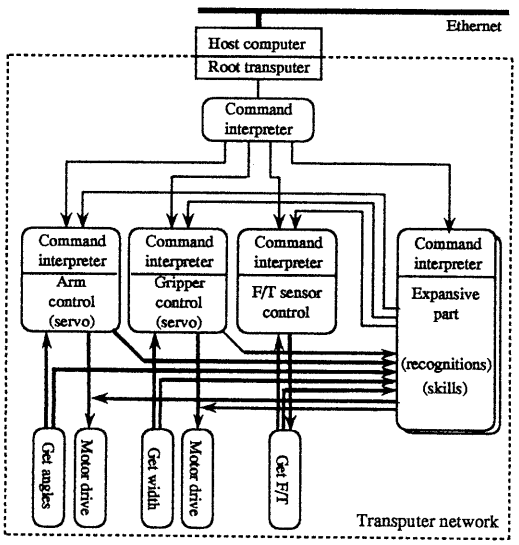

Figure 4 System configuration

最後に，本研究を行う機会を与えて頂いた高瀬知能システム部長，築根行動知能研究室長に感謝の意を表します．また，日頃よりご討論頂いている電子技術総合研究所ロボットグループの皆様に感謝致します．

参考文献

[1] 特集：センサ情報の統合，日本ロボット学会学会誌，Vol.8, No.6, 1990.

[2] 石川正俊，“センサフュージョンの課題”，日本ロボット学会学会誌，Vol.8, No.6, pp.735-742, 1990.

[3] 北垣高成，小笠原司，末広尚士，“拡張性の高いセンサ情報並列処理ロボットシステム”，第10回日本ロボット学会学術講演会予稿集，No.3, pp.1035-1036, 1992.

[4] 末広尚士，”高技能マニピュレーションシステムに関する研究”，電子技術総合研究所研究報告，No.912, 1990.

[5] T.Henderson, C.Hansen, and B.Bhanu,”The Specification of Distributed Sensing and Control”, J.of Robotic Systems, Vol.2, No.4, pp.387-396, 1985.

[6] 宮信二，長谷川健介，高橋宏治,”センサ融合システムの構築に関する基礎的研究”，SICE’89予稿集，Vol.1, pp.673-674, 1989.

[7] 末広尚士，高瀬國克,”スキルに基づくマニピュレーションシステム,” 日本ロボット学会学会誌，Vol.8, No.5, pp.551-562, 1990.

[8] Transputer Databook, INMOS Ltd., Redwood Burn Ltd., 1989.

[9] Occam2 Reference Manual, INMOS Ltd., Prentice Hall, 1988.

[10] ANSI C Toolset Language Reference, INMOS Ltd., 1990.

[11] 末広尚士，高瀬國克,”直接計算方式による作業座標サーボに基づくマニピュレーションシステム”，日本ロボット学会学会誌，Vol.3, No.2, pp.95-105, 1985.

[12] 下倉健一郎，武藤伸洋，末広尚士,”トランスピュータを用いたマニピュレータの作業座標サーボ”，機講論，No.910-31, Vol.A, pp.5-8, 1991.

[13] 末広尚士，北垣高成，”柔軟なロボットシステムのためのトランスピュータネットワーク用ルータソフトウェア”，第11回日本ロボット学会学術講演会予稿集，No.3, pp.901-902, 1993.

[14] T.Matsui and M.Inaba, “Euslisp: An Object-Based Implementation of Lisp”, J.of Information Processing, Vol.13, No.3, pp.327-338, 1990.

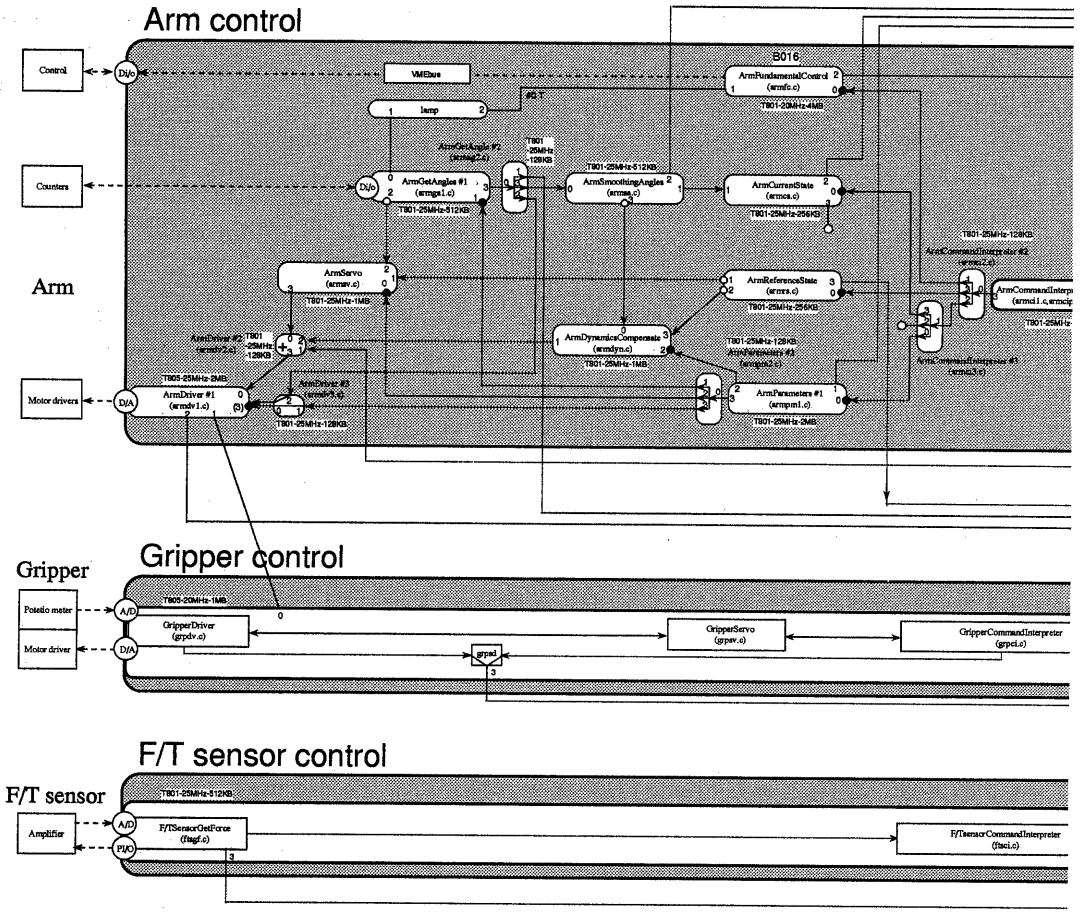

Figure 5-1 Transputer network

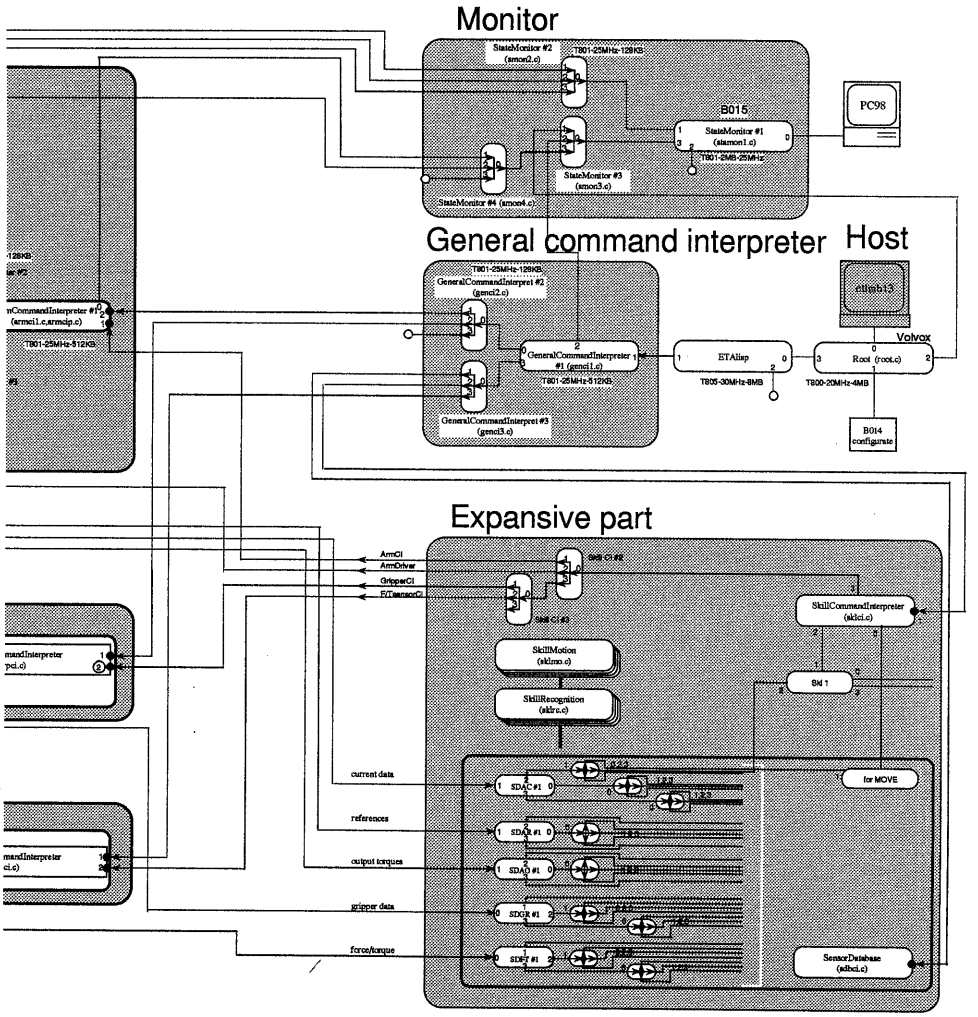

Figure 5-2 Transputer network