次期スーパーコンピューターシステム
SR8000/MPP の特長

（株）日立製作所

1. はじめに

1996 年 2 月に納入したスカラー超並列型スーパーコンピューターSR2201（1,024 プロセッサーモデル）の後継機として、2001 年 3 月に、スーパーコンピューターシステム SR8000/MPP（1,152 プロセッサー（144 ノード）モデル）を納入する予定です。

納入する SR8000/MPP は、単体の 64 ビット浮動小数点演算性能（理論ピーク性能）1.8GFLOPS の演算プロセッサーを 1,152 個備え、システム全体で 2073.6GFLOPS（2TFLOPS 強）の性能を実現します。

ここでは SR8000/MPP の特長についてご紹介します。

2. SR8000/MPP の特長

SR8000/MPP の主な特長は以下の通りです。

① 演算プロセッサーとして、最先端の CMOS テクノロジーを用いた自社開発の 64 ビットアーキテクチャの高性能 RISC マイクロプロセッサーを採用しました。
② プロセッサー間のデーターのやりとりを MPI 通信により高速に実行できます。
③ 実効性能向上のために、協調型マイクロプロセッサー機構と擬似ベクトル処理機構を採用しました。
④ 最先端の CMOS 技術、実装技術で、省電力、省スペースを実現しています。
⑤ きめ細かい超並列処理機構を採用し、柔軟な処理ができます。
⑥ 大規模で高速な計算をサポートする UNIX-OS、最適化コンパイラー等のソフトウエアを提供します。

表１に、東京大学情報基盤センターに納入された構成の SR2201 と今度納入する構成の SR8000/MPP の比較を示します。

表１ SR2201 と SR8000/MPP の比較

	SR2201（現行機種）	SR8000/MPP（新機種）
ノード構成	1 プロセッサーがメモリーに接続	9 プロセッサー（内 8 台が演算プロセッサー、1 台がシステム制御プロセッサー）がメモリー共有
演算プロセッサー性能	0.3GFLOPS	1.8GFLOPS
ノード性能	0.3GFLOPS	14.4GFLOPS
ノード当りメモリー容量	0.25GB（256MB）	16GB
演算プロセッサー数	1,024	1,152
ノード数	1,024	144
ノード間ネットワーク	3 次元クロスバー	3 次元クロスバー
ノード間転送性能	0.3GB/s（単方向）×2	1.6GB/s（単方向）×2
システム性能	307.2GFLOPS	2073.6GFLOPS

3. ハードウエアの特長

SR8000/MPP のハードウエア構成を図 1 に示します。

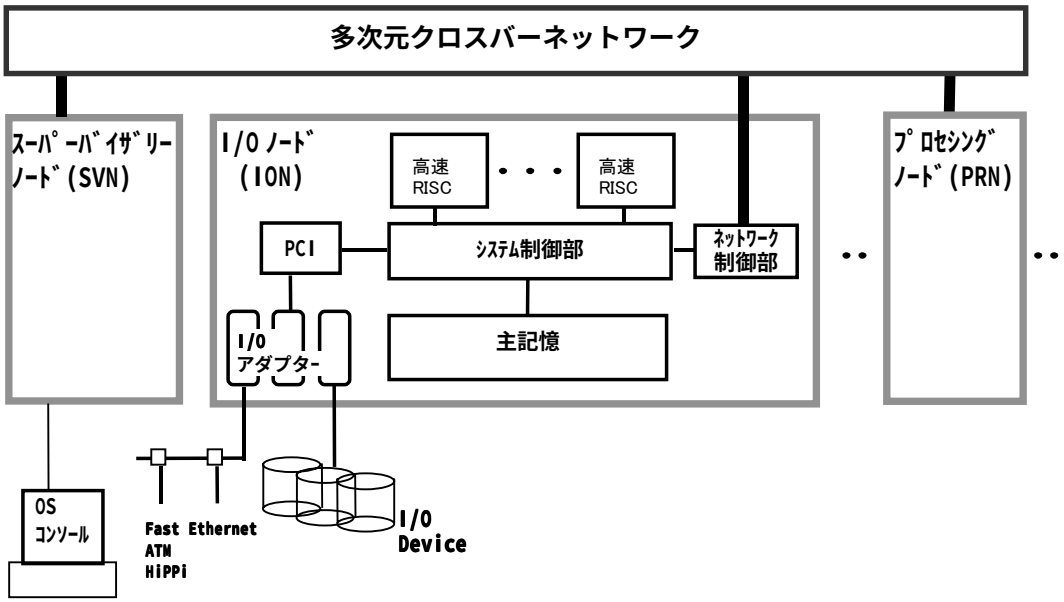


図 1　ハードウエア構成

ノードには以下の 3 種類があり、ノード間は多次元クロスバーネットワークで接続しています。

①スーパーバイザリーノード：演算処理も I/O 処理も行ない、かつ、ブートデバイス、OS コマンド入力用の OS コンソールを接続可能で、システム全体の制御を行なうノード

②I/O ノード：演算処理も I/O 処理も行なうノード

③プロセシングノード：演算処理を行なうノード

スーパーバイザリーノードはシステムに必ず 1 ノードだけ存在します。スーパーバイザリーノード以外のすべてのノードについては、 I/O ノード／プロセシングノードの選択ができます。スーパーバイザリーノードと I/O ノードには I/O アダプターが搭載可能で、ディスクアレイをはじめとする I/O 機器が接続できます。

ノードは、9 個の RISC マイクロプロセッサーがメモリーを共有する形態をとっており、内 8 個が演算プロセッサー、残り 1 個がシステム制御プロセッサーです。多次元クロスバーネットワークとメモリーとのデーターのやりとりは、ネットワーク制御部を通して行ないます。ノード内のプロセッサー間のデーターのやりとりは、MPI 通信により実行できます。ノード内における、8 個の演算プロセッサーが協調した SMP 処理も可能です。

プロセッサーの処理速度は最大理論演算性能 1.8GFLOPS、ノードの処理速度は最大理論演算性能 14.4GFLOPS、多次元クロスバーネットワークの転送速度は片方向で 1.6GB/s、双方向で 3.2GB/s です。

以下に SR8000/MPP の特長的機能を示します。

(1) 協調型マイクロプロセッサー機構

RISC マイクロプロセッサーベースのアーキテクチャのもとで、ベクトルプロセッサーと同等の高速性を実現するために、協調型マイクロプロセッサー機構をノードに採用しています。

図 2 に協調型マイクロプロセッサー機構の概要を示します。ノードを構成する各マイクロプロセッサー（図中の IP）は、演算処理をそれぞれ別のベクトル要素に対して実行します。1 つのマイクロプロセッサーが「スタート命令」の発行により、他の全マイクロプロセッサーに一斉に起動をかけます。これにより、全マイクロプロセッサーは演算を開始します。各マイクロプロセッサーは演算が終了すると、「エンド命令」の発行により、演算を終結します。上記起動（スタート）／終結（エンド）処理はハードウエアで実行されるので、非常に高速です。この高速な起動／終結処理は、高い実効性能に寄与しています（特に、小粒度の計算において）。また、コンパイラーが自動的にノード内の並列処理を行なうので、ユーザーは、ハードウエアを意識することなくコーディングできます。ノード間の並列処理については、ユーザーにてコーディングしていただきます。

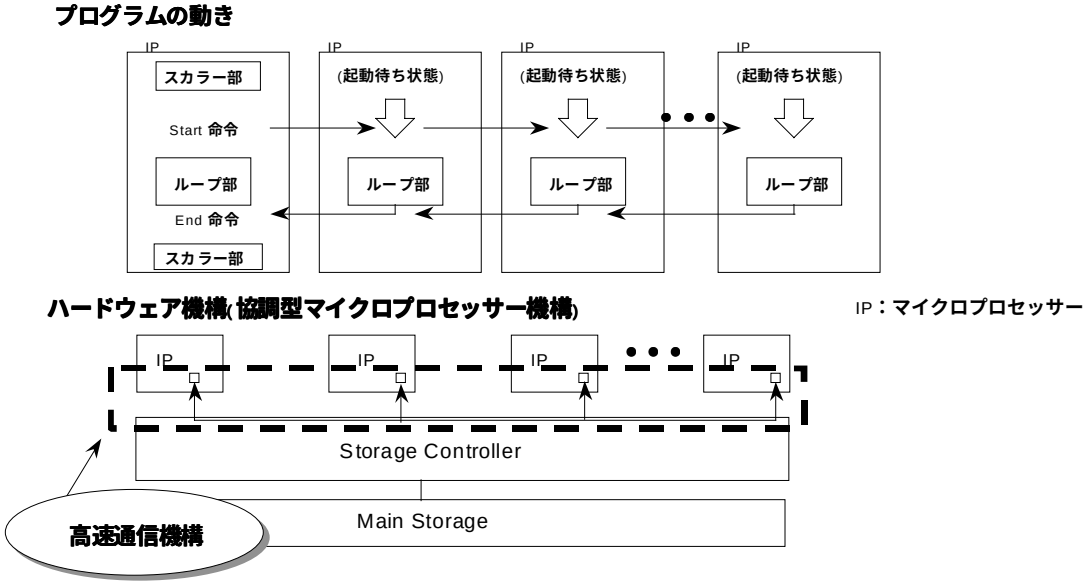


図 2　協調型マイクロプロセッサー機構の概要

(2) 擬似ベクトル処理機構

擬似ベクトル処理機構は、RISC マイクロプロセッサーでの数値計算を高速化するための機構です。図 3 に擬似ベクトル処理機構の概要を示します。

擬似ベクトル処理機構では、①多数の浮動小数点レジスターと大容量の L1 Cache、②パイプライン動作可能なメモリー構成、③後続命令を止めない制御により、演算器にデーターを連続的に供給し、ベクトル型スーパーコンピューターと同等のベクトル処理を実現しています。

演算器にデーターを連続的に供給するための手段として、(a)プリロード命令によるメモリ

ーから浮動小数点レジスターへの先読み、(b)プリフェッチ命令によるメモリーからキャッシュへの先読みの 2 つを備えています。

コンパイラーオプションにより擬似ベクトル処理機構を使用できます。

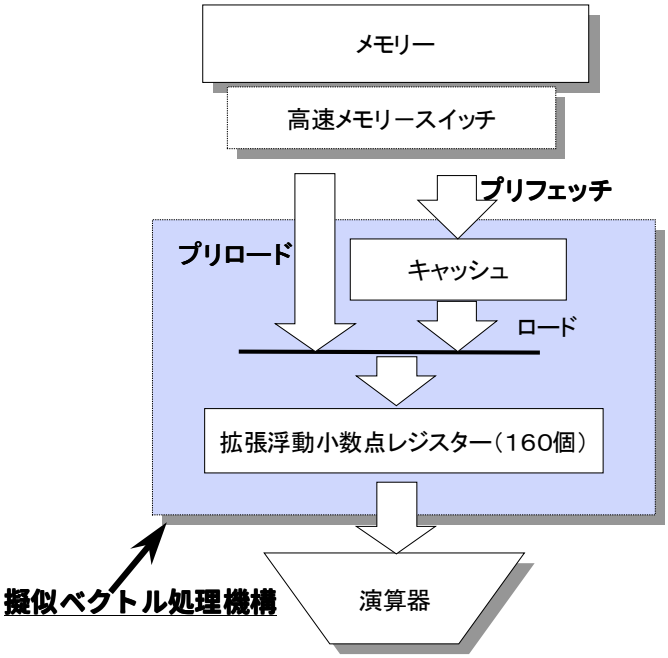


図 3　擬似ベクトル処理機構の概要

(3) 多次元クロスバーネットワーク

SR8000/MPP のネットワーク構成（納入する構成：144 ノード）を図 4 に示します。ノードは 3 次元に配列され、ノード間をクロスバースイッチが接続します。144 ノード構成は、8（X）× 8（Y）× 3（Z）構成から、X=0-7、Y=2-7、Z=2 のノードを除いた構成です。

多次元クロスバーネットワークは、論理的に他のトポロジーの結合網を大部分包含でき、しかも広範囲の転送パターンにおいて、結合網での制約（衝突）による性能の低下を起こすことなく、通信が可能となります。

(4) リモート DMA 転送

図 5 にリモート DMA 転送の概略を示します。

通常のノード間の通信は OS を介した通信であり、送信側ノードは、ユーザー空間内のデーターを OS の送信バッファーにコピーし､受信側ノードに転送します｡受信側ノードは、OS の受信バッファーにこれを受け、受信するユーザープロセスから要求を受けると、データーをユーザー空間にコピーします。

これに対しリモート DMA 転送は、 OS を介さずユーザー空間からユーザー空間へ直接データーを転送します。これによって、通信処理オーバーヘッドを低減でき、ネットワークの高速性を最大限に活用し、大量のデーターをノード間で高速に転送することができます。

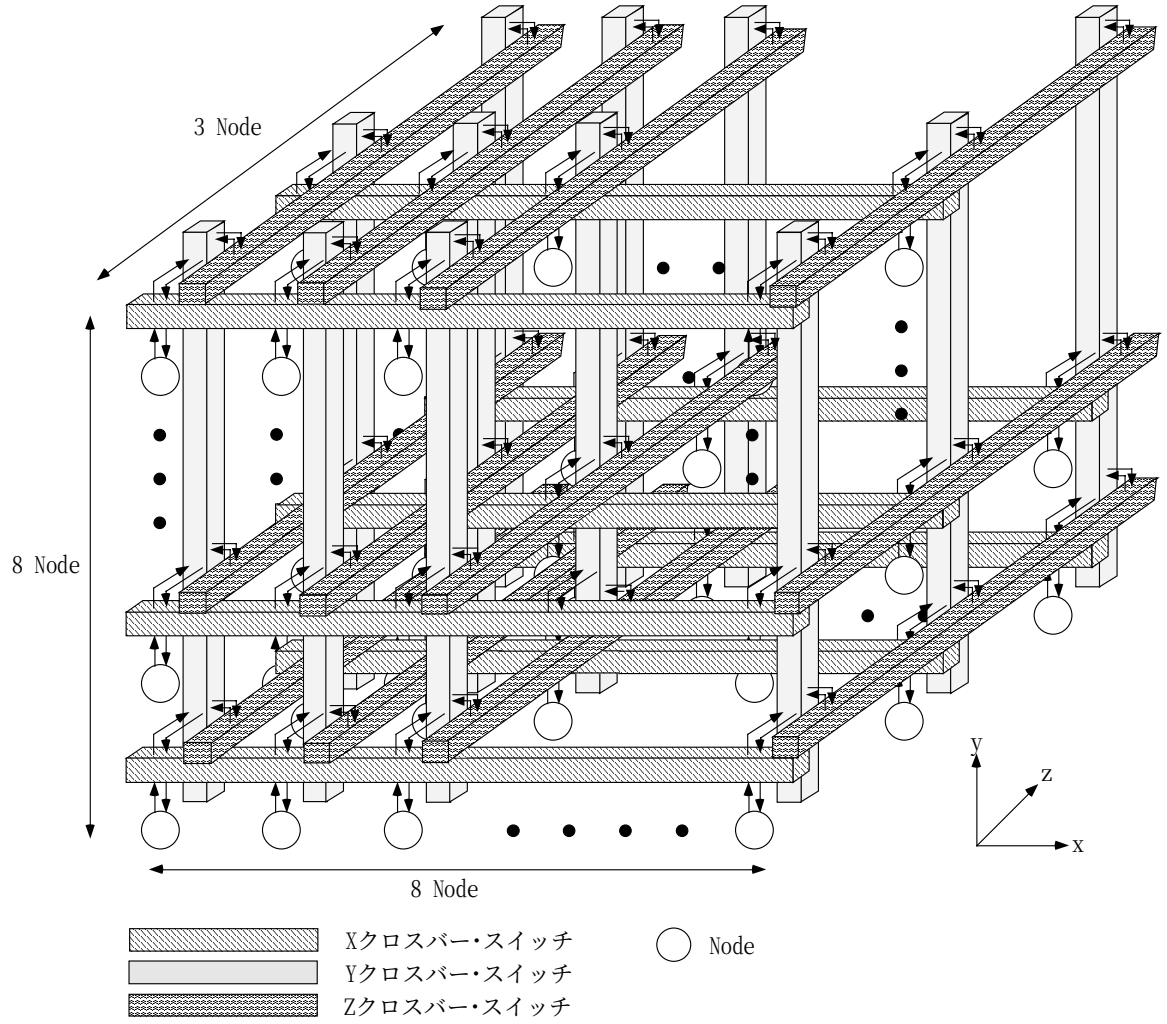


図 4　SR8000/MPP のネットワーク構成例（144 ノード構成）

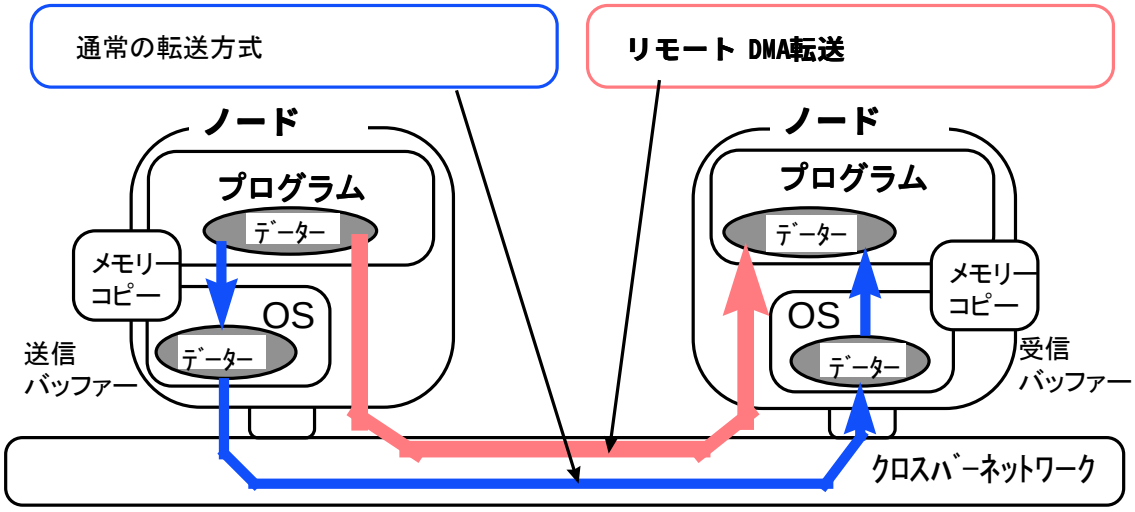


図 5　リモート DMA 転送の概略

(5)最先端の半導体と実装技術

図 6 にハードウエアの写真を示します。

①プロセッサーには、最先端のゲート長 0.14μm CMOS テクノロジーを駆使し、日立が開発した高性能 RISC マイクロプロセッサーを採用しました。

②RISC マイクロプロセッサー、メモリー制御 LSI、通信制御 LSI をコンパクトに実装する高密度ガラスセラミックモジュールを、その他制御 LSI やメモリーとともに高密度配線基板（パッケージ）に搭載しました。16GB のメモリーを持つ理論ピーク性能 14.4GFLOPS のノードを 534mm×413mm の面積に収めています。

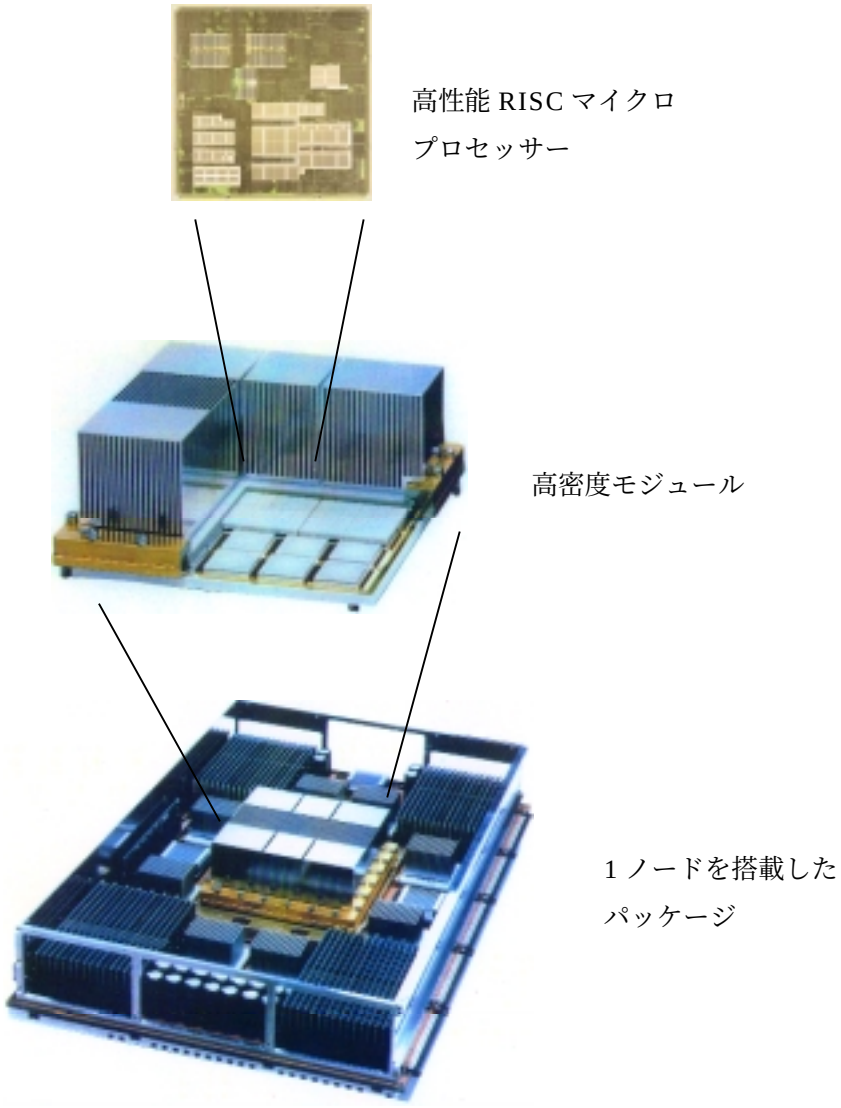
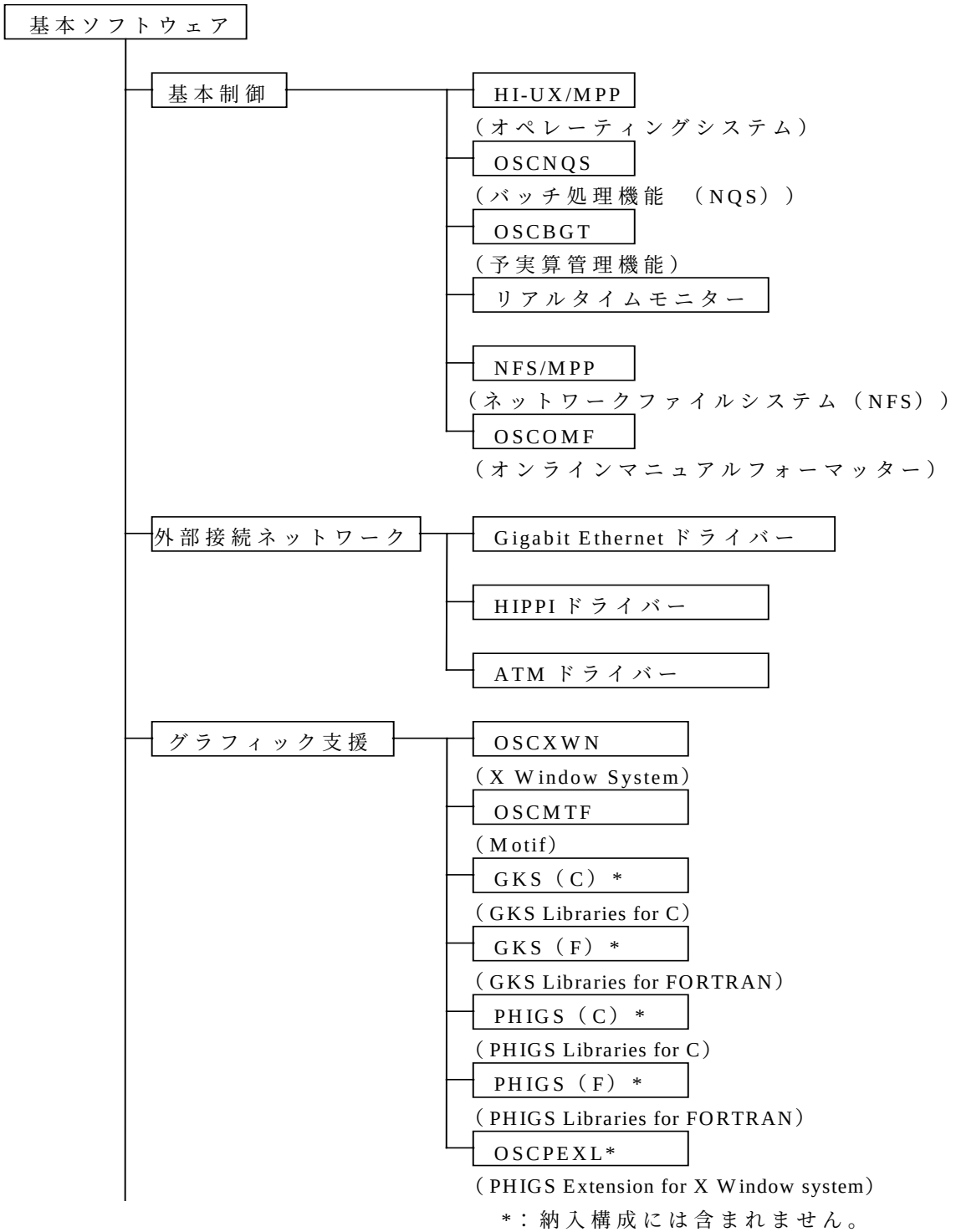


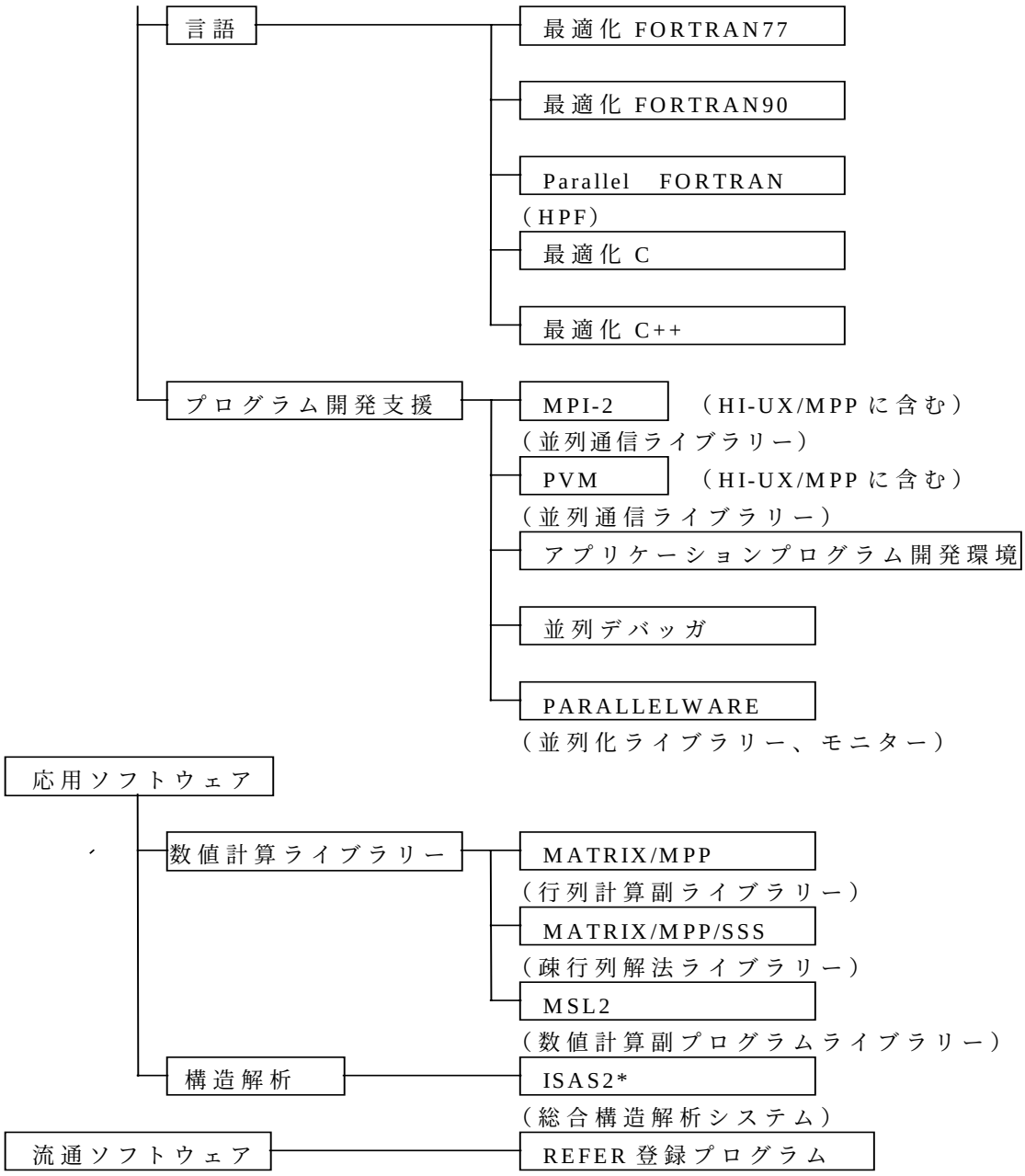
図 6 ハードウエアの写真

4. ソフトウェアの特長

SR8000/MPP 用オペレーティングシステムである HI-UX/MPP は、並列、分散用 OS として最適なマイクロカーネル技術を採用し、さらに協調型マイクロプロセッサー機構、擬似ベクトル処理機構をサポートしています。HI-UX/MPP はマイクロカーネル上に UNIX サーバーを搭載しており、SR8000/MPP 向けの並列処理機能とシングルシステムの運用機能を提供します。

4.1 SR8000/MPP のソフトウェア構成





4.2 ソフトウェアの特長

(1) シングル UNIX システム

SR8000/MPP の各ノードに搭載された OS は、ユーザーやプログラム、及びネットワークに接続された外部のコンピューターから 1 つの UNIX システム（ファイルシステム、プロセス管理、ネットワーク）に見えるようにしています。これにより、入出力装置や外部ネットワークが接続されていないノードからも入出力が実行でき、外部ネットワークにもアクセスすることができます。

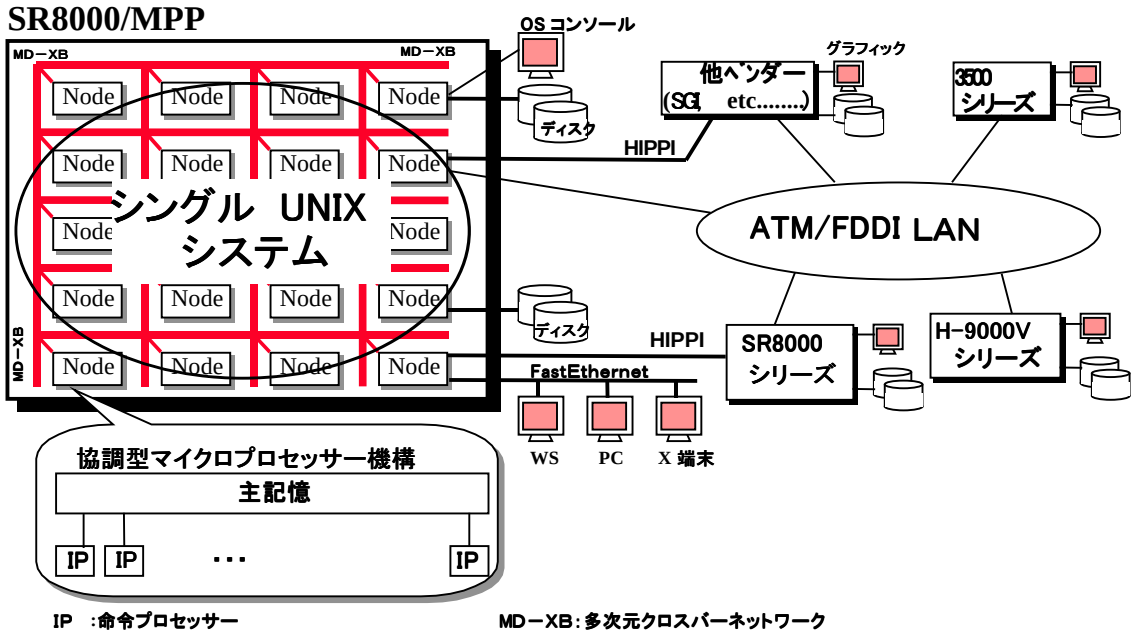


図 7 シングル UNIX システム概念図

(2) ハードウェア機構に対応した言語プロセッサー

SR8000/MPP では、言語プロセッサーとして、最適化 FORTRAN77、最適化 FORTRAN90、最適化 C、最適化 C++、および Parallel FORTRAN を提供します。

最適化 FORTRAN、C コンパイラーは、ホストや WS で実現した強力な最適化機能に加えて、SR8000/MPP の協調型マイクロプロセッサー機構を最大限に活かしたオブジェクトの生成機能を持ちます。最適化 FORTRAN、C コンパイラーではノード内の複数のプロセッサーを並列に使用する高性能オブジェクトを生成するためのノード内自動並列化機能を提供します。また、単体プロセッサー性能を引き出すための擬似ベクトル化機能も合わせて提供します。

最適化 FORTRAN コンパイラーは、オブジェクト中にソースを持ち、プログラム単位でソースの変更部分のみをコンパイルする機能を提供します。この機能によりコンパイルにかかる時間を削減することが出来ます。また、ソースが無くなってもオブジェクトから再コンパイルできる機能を提供致します。

最適化 C++はオブジェクト指向プログラミングの基幹言語として広く普及している C++言語のコンパイラーです。

Parallel FORTRAN は、HPF（High Performance FORTRAN）第 2.0 版をサポートし、データー分割を指示するだけで並列プログラムを作成することができます。

(3) 大規模・高速ファイルシステム

ファイルサーバーでは、2GB を超えるファイルサイズをサポートし、ファイルシステムの容量を最大 2TB に拡張します。大規模入出力と大量データーを必要とする大規模数値計算を可能とします。

HI-UX/MPP では、ストライピングファイル機構を適用し、SR8000/MPP に接続される全

てのディスクを一つのファイルシステムとすることにより、単体のディスク容量を超える大きさのファイルの入出力を可能とします。この大規模ファイルへの入出力性能を、演算性能に比例して確保するため、ブロックストライプ入出力機構、MPI-IO 機能を提供します。（本項目については、4.3 (1) で概要を説明しています。）

(4) 拡張記憶機能

中間ファイル入出力を高速化するため、主記憶の一部を擬似的に外部拡張記憶装置として使用できる機能を提供します。この拡張記憶機能では複数のノードの主記憶を統合して、一つの外部拡張記憶装置として使用することができ、ノード数に応じて容量を拡張することが可能です。（本項目については、4.3 (1) で概要を説明しています。）

(5) バイモーダル機能

64 ビットアドレッシングモードと、32 ビットアドレッシングモードの 2 つのアドレッシングモードを提供します。64 ビットアドレッシングモードは、大容量のデーターを処理する大規模演算プログラムに適しています。32 ビットアドレッシングモードは、既存の 32 ビットアーキテクチャのシステムで動作しているプログラムの移行のために提供します。

(6) 並列プログラミング向け通信ライブラリー

並列プログラミング向け通信ライブラリーMPI-2（Message Passing Interface - 2）と、PVM（Parallel Virtual Machine）をサポートします。MPI-2 は、従来の MPI の機能に、並列 I/O や、動的プロセス生成等の機能を追加したものです。これらのライブラリーでは、内部的にリモート DMA 転送を利用することにより、高速なデーター通信を可能にしています。

(7) 開発支援機能

並列化に対応した数値計算ライブラリー、アプリケーションプログラム開発環境、リアルタイムモニター、並列デバッガを提供することにより、アプリケーションプログラムの開発を容易にします。（アプリケーションプログラム開発環境及び並列デバッガについては、4.3 (2)(3)で概要を説明しています。）

(8) 柔軟な運用機能

以下に示す豊富な運用機能により効果的な利用が可能となります。

(a) シングルシステムとしての運用

SR8000/MPP はシステムを運用するオペレーターやシステム管理者からもシングルシステムとして扱うことができます。これにより、システムのインストールはスーパーバイザリーノード（SVN）のシステムディスクにインストールするだけで済みます。また、システムの開始や終了などシステム全体の操作が、1 台の OS コンソールやワークステーションから行えます。

(b) 多重ジョブ運転

ジョブを多重実行するために SR8000/MPP を複数のノード群に分割して運転する方法として、ノードパーティション分割を提供しています。

ノードパーティション分割は管理者が運用形態に応じて分割を設定でき、ノードやノードパーティションをユーザーやプログラムが占有／共有することができます。また、バッチ処理専用のパーティションも設定できます。

なお、ノードパーティション分割はシステム稼働中に動的に変更することができます。

(c) サブシステム分割運転

1 台の SR8000/MPP のプロセシングノードと I/O ノードを論理的な複数のサブシステムに分割し、各々が 1 つの独立したシステムとして運転できる機能を提供します。サブシステム分割運転により、大規模なシステムの負荷分散、サブシステムごとに独立した起動と停止、障害時のシステムへの影響の局所化ができます。さらに、セキュリティー面からもシステムの信頼性を向上することができ、システムの運用を柔軟にすることができます。

分割したサブシステム間は、多次元クロスバーネットワークを TCP/IP 通信路として高速に結合できます。

(d) バッチジョブと対話ジョブ

SR8000/MPP の対話環境、もしくは WS から NQS の qsub コマンドを投入することによってジョブをサブミットし、OSCNQS のジョブキューに対応したパーティションでジョブを実行するバッチジョブ運用と、SR8000/MPP にネットワークで接続された WS/PC からリモートログインしてジョブ起動コマンドを直接投入する対話ジョブ運用との 2 つの運用が可能です。

図 8、9 にバッチジョブ及び対話ジョブの運用イメージを示します。

(e) 自動運転

あらかじめ設定しておいた時刻にシステム電源の投入と OS の立ち上げを自動的に行い、システムを立ち上げることができます。また、指定された時刻にシステムの終了とシステム電源の切断を自動的に行うことができます。

なお、システムの終了では、実行中のジョブがあった場合には、強制的にジョブを打ち切ります。また、運転スケジュールは日毎に設定することもできます。

(f) 上限値管理機能

プロセス単位、及びプロセスグループ単位にシステム資源（CPU 使用量、メモリー使用量、ファイル使用量など）の使用状況をチェックすることによって、予め設定した上限値を超えたジョブの実行を打ち切ることができます。

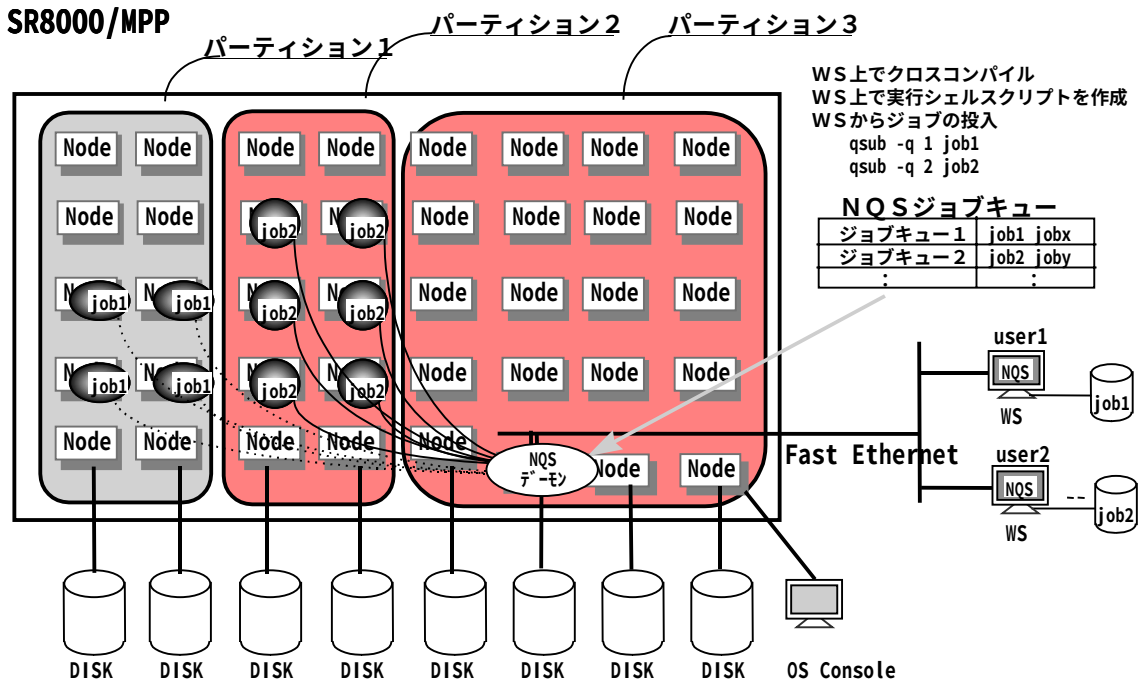


図 8 バッチジョブの運用イメージ

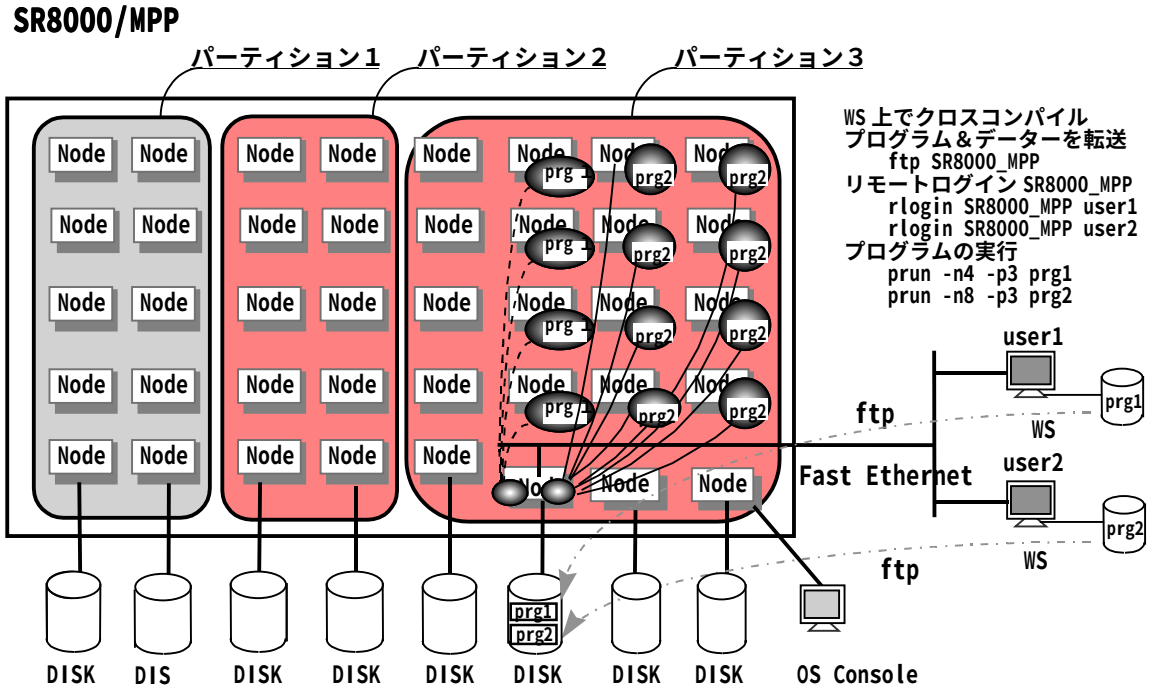


図 9 対話ジョブの運用イメージ

(g) 課金機能

プロセス単位の課金やセッション単位の課金の機能に加え、プロセスグループ単位やクラス別（バッチジョブや対話ジョブ）の課金情報の収集・操作ができます。さらに、ユーザーが使用したノード数に対応した課金機能も提供します。

(h) 機密保護機能

システム管理者の役割分離機能や、ACL（Access Control List）を使用したアクセス制御を実現し、機密保護機能を強化しています。

ACL は従来のファイルやディレクトリに設定された所有者、グループ、及び一般ユーザーに対する読み込み、書き込み、実行又はサーチ許可権限を拡張したアクセス制御で、ファイルやディレクトリーに設定された所有者、グループに加えて、特定のユーザーやグループを指定したアクセスの許可、不許可を設定可能にします。

(i) ジョブフリーズ機能

システムの定期的なメンテナンスや、システム変更にともなうシステム停止において、システム停止の前に、実行中のジョブをフリーズして保存しておき、システムの運用開始後に再開する機能を有します。SR8000/MPP の並列 MPI ジョブにも対応しています。

(j) ノード閉塞機能

SR8000/MPP の 1 ノードでハードウェア障害が発生した場合、システムの再起動を介して、自動的に障害発生ノードを切り離す機能を有します。障害発生ノードに接続されているディスク装置やネットワークアダプタについても、機能を交代するノードを用意しておくことで、1 ノードが障害により切り離された後も、ディスク I/O や外部ネットワークとの通信といった機能を引き継いで、システム運用が続行可能となります。

(k) 並列ジョブ

共有メモリーマルチプロセッサーで構成された 1 ノードでは、COMPAS 機能を利用した要素並列型の並列ジョブと、SR2201 同様の MPI 通信を用いたジョブを実行可能です。後者の場合、1 個のプロセッサーが SR2201 の 1 ノードに相当します。

1 ノード内で MPI ジョブを実行する場合、COMPAS 機能を利用して MPI プロセス間の同期をとったギャングスケジュールを可能としています。このギャングスケジュールを適用することにより、複数のノード内 MPI ジョブの多重実行が効率的に行えます。

複数のノードを使用する並列ジョブの形態としては、ノード内 MPI とノード間 MPI の組み合わせ及び要素並列とノード間 MPI（あるいはリモート DMA 通信）の組み合わせが可能です。

4.3 主な機能の概要

(1) 大規模・高速ファイルシステム

SR8000/MPP では、大規模ファイルへの入出力性能を高速化するため、ストライピングファイル機構、MPI-IO 機能、拡張記憶機能（ES）をサポートしています。以下に、ファイルシステムの概要を示します。

(a) シングルディレクトリーツリーのファイルシステム

各 I/O ノード上に分散して配置されているファイルシステムは、システム全体で 1 つの

ディレクトリーツリー構造のファイルシステムとして、ユーザーやプログラムから扱えます。したがって、アクセスするファイルがどの I/O ノード上に所在するかを意識せずに使用することができます。図 10 にファイルシステムの概念を示します。

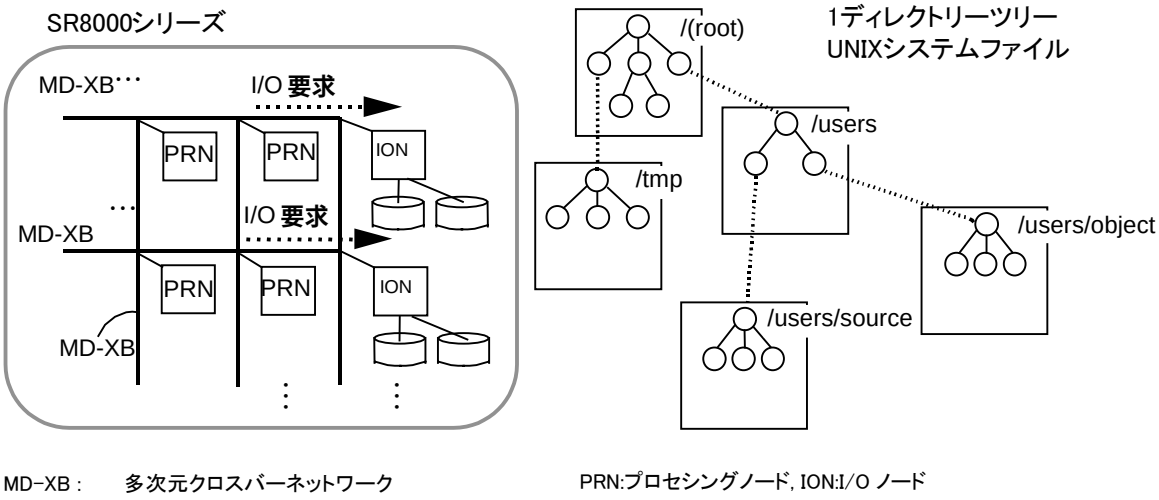


図 10 ファイルシステムの概念

(b) UNIX ファイルシステム

4.3BSD の UFS と互換性のあるファイルシステムを採用しています。また、NFS にも対応しており、ネットワークに接続された他のコンピューターと相互にファイルをアクセスすることができます。

(c) ストライピングファイル機構

複数の I/O ノード間にまたがった論理的なファイルを構築し、各々のプロセシングノードから直接各々の I/O ノードに入出力データーをストライピングして配置することによって SR8000/MPP の特性を活かしたファイルへの高速なアクセスができます。なお、NFS からも利用できます。ストライピングファイル機構では次に示す 2 つの機能を提供しています。

(i) ファイルストライプ

ファイルストライプは、複数の I/O ノード上に構成されるサブファイルシステムに、ファイルをファイル単位で自動分散します。自動分散はラウンドロビンあるいは空き容量に応じて行います。複数プロセスから複数ファイルを同時にシーケンシャルに入出力する場合に有効です。図 11 にファイルストライプの概念図を示します。

(ii) ブロックストライプ

ブロックストライプは、複数 I/O ノード上に構成されるサブファイルシステムに、ファイルをデーターブロック単位で自動分散し同時並行的に入出力を行います。単一の大容量ファイルの入出力を高速に行う場合に有効です。図 12 にブロックストライプの概念図を示します。

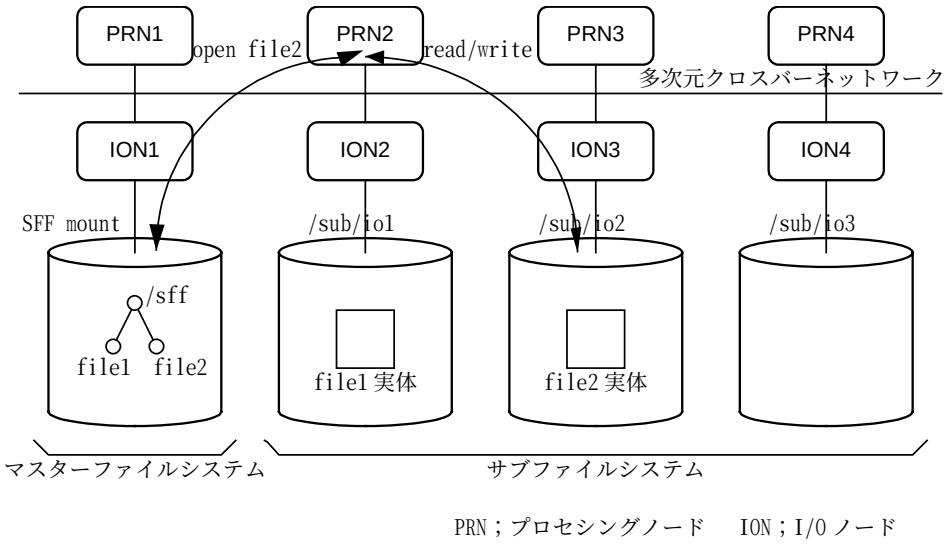


図 11　ファイルストライプ概念図

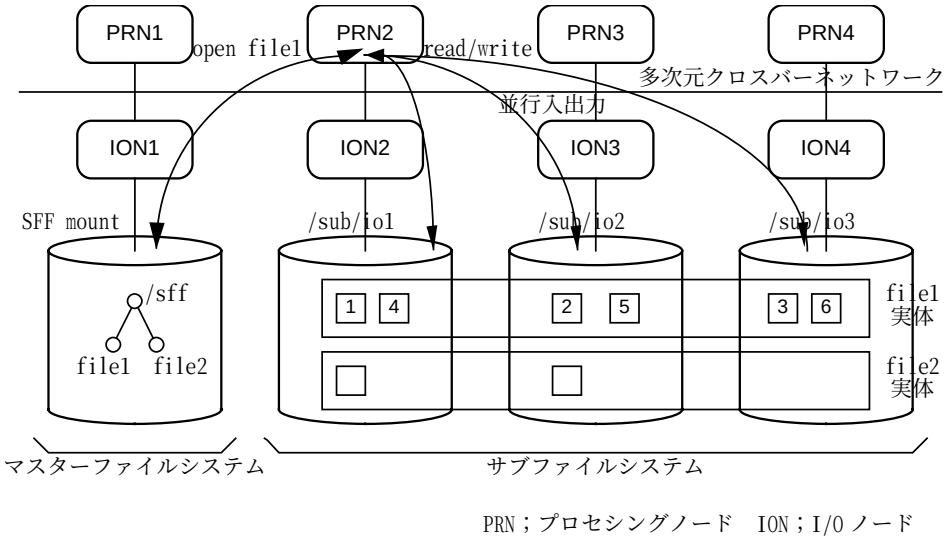


図 12　ブロックストライプ概念図

(d) MPI-IO 機能

業界標準の並列入出力インターフェースであるMPI-2に準拠したMPI-I/O機能をサポートします。MPI-I/Oでは、複数プロセスが同一ファイルにアクセスするコレクティブI/Oインターフェース、非同期I/O操作などの機能を実現しています。また、ファイルへのアクセスパターンをプロセス毎に定義する機能により、不連続ファイル領域へのI/O要求を連続領域アクセスイメージで発行できます。

図13に、コレクティブI/Oインターフェースを示します。

また、図14に示すように、コレクティブI/Oとストライピングファイル機構とを組み合わせることにより、複数プロセッサーからの入出力を、複数のディスク装置に対して並列に発行することができ、より高速なファイル入出力を可能にしました。

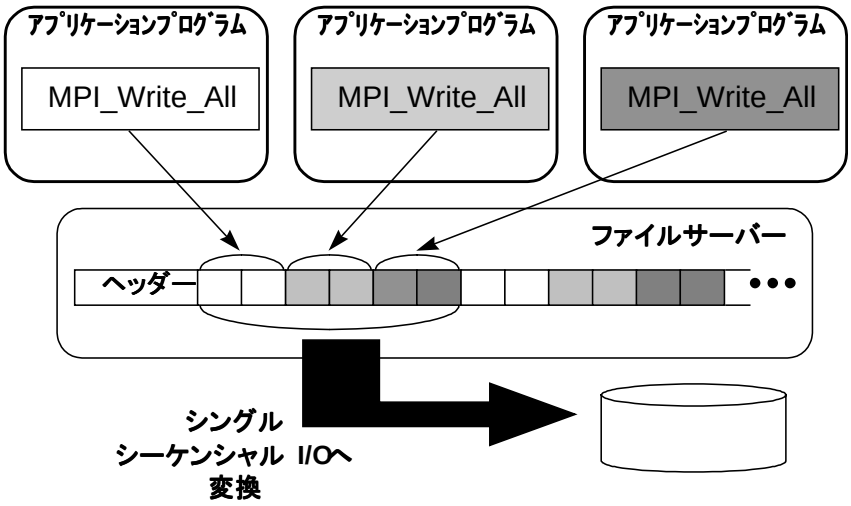


図 13 コレクティブ I/O インターフェース

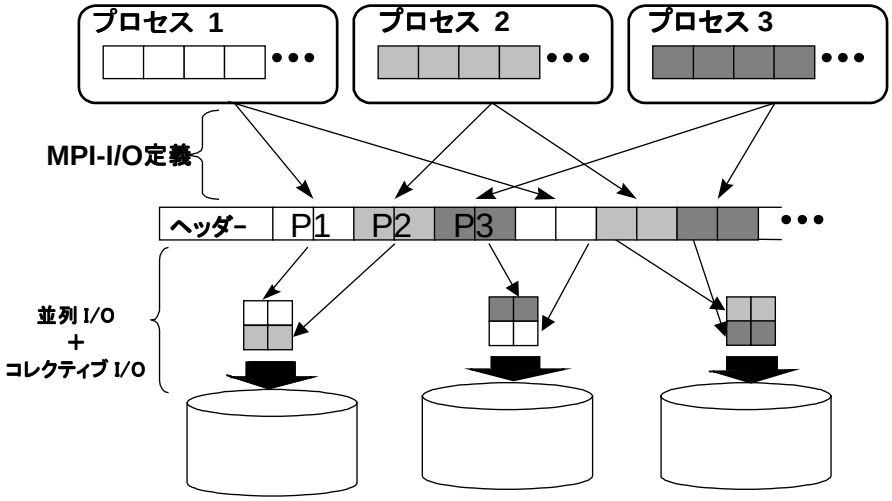


図 14　並列 I/O +コレクティブ I/O

(e) 拡張記憶機能（ES）

S-3000 シリーズの ES 機能と互換性を保ちつつ、主記憶の一部を擬似的に拡張記憶装置として使用できるようにし、高速な入出力を可能としています。また各ノードの ES を統合してひとつの大きな容量を持つ拡張記憶装置として構成できます。

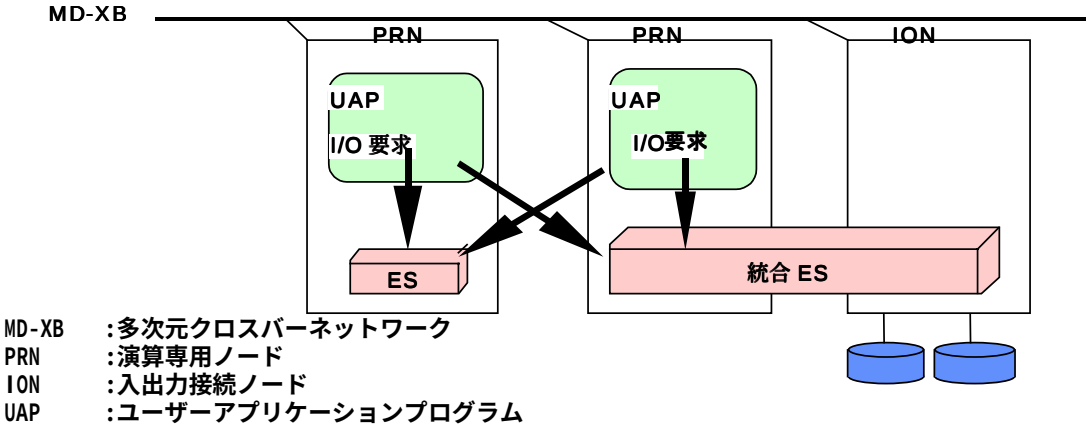


図 15 拡張記憶機能（ES）

(2) アプリケーションプログラム開発環境

アプリケーションプログラム開発環境には、性能情報をプログラムの実行により採取してコマンドで性能情報表示する性能モニタ機能と X Window System、Motif によるユーザーインターフェースを使用してプログラム開発のチューニングを支援する機能があります。

(2-1) 性能モニタ機能

SR8000/MPP は、プログラム実行に要した CPU 時間、総実行命令数、浮動小数点演算数などの実行時の各種性能情報を採取するためのハードウェア機構を持っています。性能モニタ機能は、このハードウェア機構を介して実行時の性能情報を読み取り、集計処理をした後、プログラム実行時の性能モニタ情報ファイルを出力します。

性能モニタ機能は、FORTRAN コンパイラー及び C コンパイラーの機能と連携しています。コンパイル及びリンク時に性能モニタ情報採取用オプションを指定することによって、コンパイラーは、性能モニタ情報を採取するライブラリー（性能モニタライブラリー）の呼び出しをオブジェクトコードに挿入します。性能モニタ関数は、プログラム実行時に採取した情報を集計してファイルを出力します。出力されたファイルの内容を、性能モニタ情報表示コマンド（pmpr）により参照します。

性能モニタ情報には次の特長があります。

・ソースコードを変更する必要がない。

・コンパイル及びリンケージオプションの追加だけで性能モニタ情報を利用することができる。

・ソースファイルごとに性能モニタ情報採取を指定できる。

性能モニタ情報には、プロセス情報、関数／手続き情報および要素並列情報があります。

(a) プロセス情報

プログラム全体(複数ノードで実行する場合はプロセス単位)の性能モニタ情報を表示します。プロセス情報は、CPU 時間、演算性能、総実行命令数、浮動小数点演算数および入出力回数等を表示します。図 16 に、プロセス情報に対する表示例を示します。

```
##############################################################
## プロセス                                                  ##
##############################################################
  日付                  : YYYY年MM月DD日(月) HH時MM分SS秒
  ノード番号            : 0
  プロセス番号          : xxxx
  ロードモジュール名    : a.out
  CPU時間               : xx.xxxxxx[秒]
  MFLOPS                : mmmm.mmm
  MIPS                  : nnnn.nnn
  入力回数              : xx
  出力回数              : xx
  入力量                : yyyyyyyy[バイト]
  出力量                : yyyyyyyy[バイト]
  ------------------------------------------
            FLOP     Inst    LD/ST  D-cache
  ------------------------------------------
  IP0     fffffff  iiiiiii  sssssss< xxxxxxx
  IP1     fffffff  iiiiiii  sssssss> xxxxxxx
  IP2     fffffff  iiiiiii  sssssss  xxxxxxx<
  IP3     fffffff> iiiiiii  sssssss  xxxxxxx
  IP4     fffffff  iiiiiii> sssssss  xxxxxxx>
  IP5     fffffff  iiiiiii< sssssss  xxxxxxx
  IP6     fffffff  iiiiiii  sssssss  xxxxxxx
  IP7     fffffff< iiiiiii  sssssss  xxxxxxx
  ------------------------------------------
  合計    fffffff  iiiiiii  sssssss  xxxxxxx
```

図 16 プロセス情報の表示例

(b) 関数／手続き情報

関数／手続き単位のランキング（CPU 時間の降順）と要素並列化された関数／手続きに対して詳細情報（IP 単位の情報）を表示します。

ランキング情報は、CPU 時間、実行回数、関数／手続き名等を表示し、CPU 時間をキーに降順にソートして表示します。詳細情報は、要素並列化された IP 毎の CPU 時間、実行回数、演算性能、総実行命令数、浮動小数点演算数等を表示します。図 17 に、関数／手続き単位の表示例を示します。

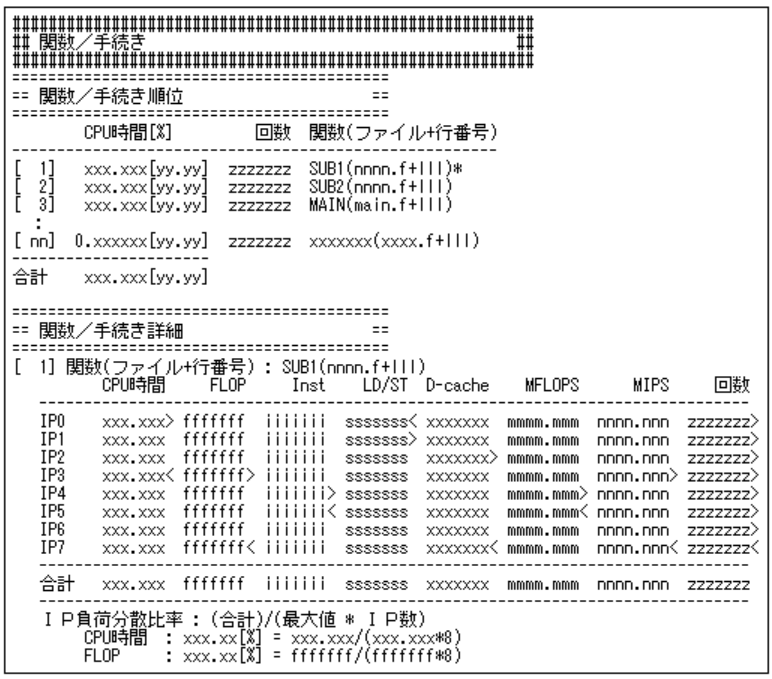


図 17　関数／手続き情報の表示例

(c) 要素並列情報

要素並列化単位のランキング（CPU 時間の降順）と詳細情報（IP 単位の情報）を表示します。

ランキング情報は、CPU 時間、実行回数、演算性能、関数／手続き名等を表示し、CPU 時間をキーに降順にソートして表示します。詳細情報は、要素並列化された IP 毎の CPU 時間、実行回数、演算性能、総実行命令数、浮動小数点演算数等を表示します。図 18 に、要素並列化された部分に対する表示例を示します。

(2-2) X ウィンドウ環境下でのチューニング

X Window System、Motif によるユーザーインターフェースを提供し、SR8000/MPP 上で稼動するアプリケーションプログラムのコンパイル、実行、性能チューニングを支援します。図 19 に、各機能のメニュー画面例を示します。（画面形式、表示内容等は、今後変わることがあります。）

```
############################################################
## 要素並列部分                                           ##
############################################################
========================================
== 要素並列部分順位                     ==
========================================
        CPU時間[%]        MFLOPS      MIPS     回数  関数[順位](ファイル+行番号)
------------------------------------------------------------------------------
[  1]   xxx.xxx[yy.yy]  mmmm.mmm  nnnn.nnn  zzzzzzz  SUB1[xx](nnnn.f+lll)
[  2]   xxx.xxx[yy.yy]  mmmm.mmm  nnnn.nnn  zzzzzzz  SUB2[xx](nnnn.f+lll)
[  3]   xxx.xxx[yy.yy]  mmmm.mmm  nnnn.nnn  zzzzzzz  MAIN[xx](main.f+lll)
  :
合計    xxx.xxx[yy.yy]

========================================
== 要素並列部分詳細                     ==
========================================
[  1] 関数[順位](ファイル+行番号) : SUB1[xx](nnnn.f+lll)
          CPU時間     FLOP     Inst     LD/ST  D-cache     MFLOPS      MIPS      回数
   ---------------------------------------------------------------------------------
   IP0    xxx.xxx> fffffff  iiiiiii  sssssss< xxxxxxx  mmmm.mmm  nnnn.nnn  zzzzzzz>
   IP1    xxx.xxx  fffffff  iiiiiii  sssssss> xxxxxxx  mmmm.mmm  nnnn.nnn  zzzzzzz>
   IP2    xxx.xxx  fffffff  iiiiiii  sssssss  xxxxxxx> mmmm.mmm  nnnn.nnn  zzzzzzz>
   IP3    xxx.xxx< fffffff> iiiiiii  sssssss  xxxxxxx  mmmm.mmm  nnnn.nnn> zzzzzzz>
   IP4    xxx.xxx  fffffff  iiiiiii> sssssss  xxxxxxx  mmmm.mmm> nnnn.nnn  zzzzzzz>
   IP5    xxx.xxx  fffffff  iiiiiii< sssssss  xxxxxxx  mmmm.mmm< nnnn.nnn  zzzzzzz>
   IP6    xxx.xxx  fffffff  iiiiiii  sssssss  xxxxxxx  mmmm.mmm  nnnn.nnn  zzzzzzz>
   IP7    xxx.xxx  fffffff< iiiiiii  sssssss  xxxxxxx< mmmm.mmm  nnnn.nnn< zzzzzzz<
   ---------------------------------------------------------------------------------
   合計   xxx.xxx  fffffff  iiiiiii  sssssss  xxxxxxx  mmmm.mmm  nnnn.nnn  zzzzzzz
   ---------------------------------------------------------------------------------
   ＩＰ負荷分散比率 : (合計)/(最大値 * ＩＰ数)
        CPU時間  : xxx.xx[%] = xxx.xxx/(xxx.xxx*8)
        FLOP     : xxx.xx[%] = fffffff/(fffffff*8)
```

図 18 要素並列情報の表示例

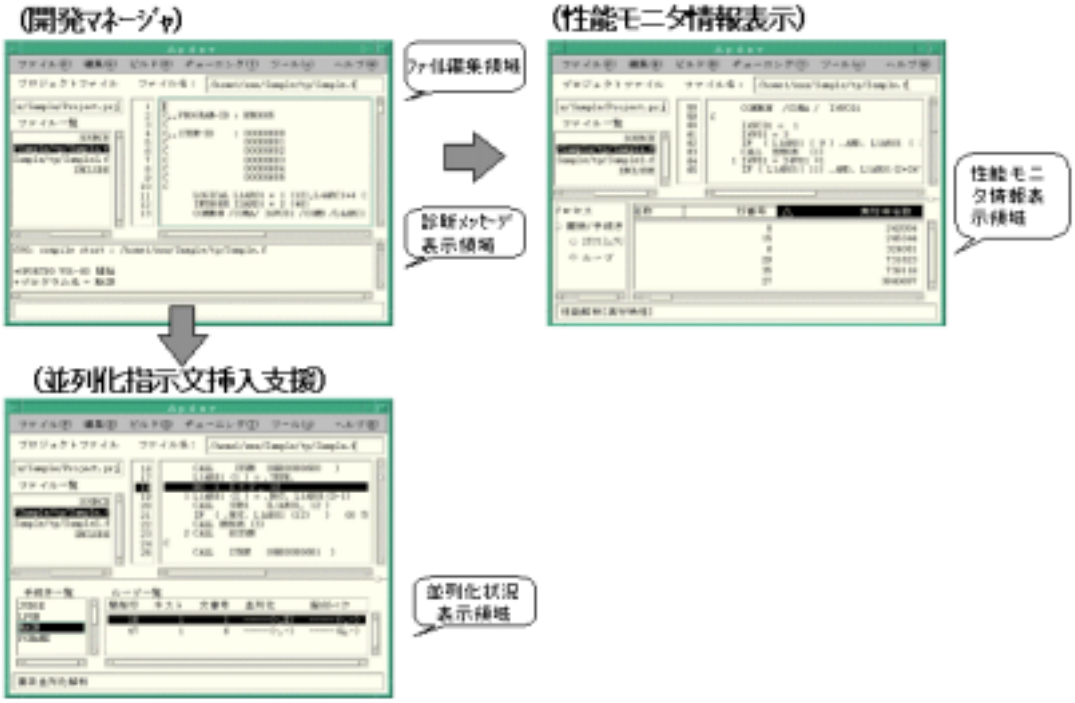


図 19 アプリケーション開発環境メニュー画面例

(a) 開発マネージャ

C、C++、FORTRAN77 および FORTRAN90 の各言語で記述されたプログラムの、コンパイル、リンクおよび実行ができます。また、これらの作業はプロジェクト（ファイルを一括して管理している単位）ごとに行えます。

(b) 並列化指示文挿入支援

C、FORTRAN77 および FORTRAN90 の各言語で記述されたチューニング対象プログラムの関数／手続き及びループをキーとして、コンパイラの診断メッセージがプログラムと対応して表示され、プログラムの要素並列化に問題がある場合は、質問文に応答することにより、挿入すべき並列化用の指示文を表示することができます。

(c) 性能モニタ情報表示

チューニング対象プログラムの性能モニタ情報を提供します。性能モニタ情報は、プログラム全体の総実行命令数、経過時間、演算性能、入出力回数等、および、プロセス、関数／手続き、要素並列化単位をキーとした実行回数、実行命令数、演算性能等が表示できます。

(3) 並列デバッガ

SR8000/MPP 上で稼動する逐次プログラム、MPI を使用したノード間並列プログラム、要素並列化プログラムに対して、次のデバッグ操作が可能なシンボリックデバッガとして、並列デバッガを提供します。

・対話デバッグ

デバッガの下でプログラムを実行させ、ブレークポイント設定、シングルステップ実行、変数／配列の値表示等の操作を組み合わせることにより、プログラムの実行不正となった部分を検出することができます。

・コア解析

プログラムが異常終了時に生成されるコアファイルの情報を解析することにより、異常終了の原因を検出することができます。

以上