55－R016

第５世代のコンピュータ

アーキテクチュア研究分科会報告書

昭和５６年３月

JIPDEC

財団法人　日本情報処理開発協会

この報告書は，日本自転車振興会から競輪収益の一部である機械工業振興資金の補助を受けて昭和５５年度に実施した「第５世代の電子計算機に関する調査研究」の成果をとりまとめたものであります。

序

わが国における社会経済は，資源，エネルギー問題を始めとして国際的な変動と，不確実性の流れのなかにある。同時に，的確な情報の加工利用が重要視される情報化社会の形成が指向されている。

コンピュータは，われわれの情報活用においてすでに不可欠なツールとなっているが，今後10年間には多くの諸問題を解決するため，更に高度な技術が要求され，新たな理論・技術にもとづくコンピュータ・システムの実現が望まれるであろう。

このため，当協会では「第５世代コンピュータ調査研究委員会」を設置し，1990年代に実用化されるべきコンピュータ・システム（第５世代コンピュータ）はどのようなものになるか，またその開発プロジェクトはどのように進めていくべきかについての調査研究を，昭和５４年度から２ヶ年の予定で開始した。

昭和５５年度は，本委員会のもとに設置した３分科会（システム化技術，基礎理論，アーキテクチャ）および多数のワーキング・グループによる調査研究活動，内外の大学等への研究委託，米国への技術調査等により，第５世代コンピュータのイメージ及び研究開発課題を明確化し，さらに，その研究開発計画・体制について検討した。

本報告書は，これらの調査研究結果のうち，アーキテクチャ研究分科会の活動成果をとりまとめたものである。

最後に調査研究にご協力いただいた第５世代コンピュータ調査研究委員会委員を始め，関係各位に厚く御礼申し上げる次第である。

昭和５６年３月

財団法人　日本情報処理開発協会
会長　上野幸七

アーキテクチャ 研究分科会（A分科会）

	氏　名	所　属
主　査	相磯秀夫	慶応義塾大学工学部電気工学科教授
委　員	飯川昭一	三菱電機㈱計算機製作所ソフトウェア製造部開発第5グループ部長代理
〃	飯塚肇	成蹊大学工学部経営工学科教授
〃	内田俊一	電子技術総合研究所ソフトウェア部情報システム研究室主任研究官
〃	小高俊彦	㈱日立製作所神奈川工場開発部主任技師
〃	坂間保雄	日本電信電話公社横須賀電気通信研究所データ通信部データ通信方式研究室研究専門調査員
〃	坂村健	東京大学理学部情報科学科助手
〃	杉本正勝	富士通㈱開発事業部開発技術部部長付
〃	武井欣二	東京芝浦電気㈱総合研究所情報システム研究所ソフトウェアグループ主任研究員
〃	田中英彦	東京大学工学部電気工学科助教授
〃	発田弘	日本電気㈱コンピュータ技術本部方式技術部技術課長
オブザーバ	安原弘	沖電気工業㈱研究所情報処理システム研究部研究主任

目　　次

1. 第5世代計算機アーキテクチャ …… 1

1.1 第5世代計算機アーキテクチャの目標 …… 1

1.1.1 技術的背景と研究目標 …… 1

1.1.2 現在の問題点と研究目標 …… 1

1.2 第5世代計算機アーキテクチャのとらえ方 …… 4

2. 新アーキテクチャ …… 7

2.1 計算機への要求と新アーキテクチャ …… 7

2.2 新アーキテクチャへのアプローチ …… 11

2.2.1 述語論理マシン …… 12

2.2.2 関数型マシン …… 27

2.2.3 抽象データ型マシン …… 37

2.2.4 関係代数マシン …… 46

2.2.5 データフローマシン …… 57

2.2.6 新（New）ノイマン型アーキテクチャ …… 67

3. 機能分散アーキテクチャ …… 77

3.1 機能分散処理方式 …… 78

3.1.1 機能分散の目的と効果 …… 78

3.1.2 機能分散のレベル …… 81

3.1.3 分散データベース …… 82

3.1.4 機能分散アーキテクチャの開発 …… 85

3.2 システム・アーキテクチャ …… 86

3.2.1 基本的な考え方 …… 86

3.2.2 システム構成方式 …… 87

3.2.3 ネットワーク・アーキテクチャ …… 92

3.3 システム制御方式 …… 101

3.3.1 基本的な考え方 …… 101

3.3.2 制御機能の分散 …… 102

3.3.3 オペレーティング・システムの分散形態 …… 104

3.3.4 オペレーティング・システム（ＯＳ）・ニュークリアスマシン …… 109
3.3.5 モニタ&コントロール …… 112
3.4 コンポーネント・マシン・アーキテクチャ …… 114
3.4.1 コンポーネント・マシンの考え方 …… 114
3.4.2 パーソナル・コンピュータ …… 115
3.4.3 マシン・サービス・マシン …… 120
3.4.4 ユーザ・サービス・マシン …… 129
3.4.4.1 高級言語マシン …… 129
3.4.4.2 データベース・マシン …… 133
3.4.4.3 科学計算マシン …… 144
3.4.5 コントロール・マシン …… 150
3.5 高度マンマシン・コミュニケーション・システム …… 157

4. ＶＬＳＩアーキテクチャ …… 165
4.1 概　　要 …… 165
4.2 ＶＬＳＩアーキテクチャ …… 166
4.3 ＶＬＳＩ－ＣＡＤ …… 174
4.4 新　素　子 …… 182
4.4.1 シリコンＶＬＳＩ素子 …… 182
4.4.2 化合物半導体素子 …… 182
4.4.3 ジョセフソン接合（ＪＪ）素子 …… 183
4.4.4 光関連技術 …… 183
4.4.5 磁性関連技術 …… 183

5. アーキテクチャにおけるシステム化技術 …… 185
5.1 システム化技法の重要性 …… 185
5.2 仮想化技術 …… 186
5.2.1 仮想化の必要性と体系化 …… 186
5.2.2 仮想化の階層とそのインタフェース …… 194
5.2.3 新アーキテクチャにおける仮想化 …… 197
5.3 諸アーキテクチャの融合技術 …… 202
5.3.1 融合技術の必要性 …… 202
5.3.2 機能分散アーキテクチャの各要素マシンの実現方式 …… 202

5.3.2.1 パーソナル・マシン …… 202
5.3.2.2 マシン・サービス・マシン …… 203
5.3.2.3 コントロール・マシン …… 205
5.3.2.4 ユーザ・サービス・マシン …… 209
5.3.3 具体的な使用形態における処理の流れとアーキテクチャの関連 …… 218
5.3.3.1 CAE／CADシステム …… 218
5.3.3.2 オフィス・オートメーション …… 221
5.3.3.3 意思決定支援システムの利用 …… 226
5.4 大規模高信頼化システム設計開発技術 …… 231
5.4.1 第5世代コンピュータシステム開発における重要性 …… 231
5.4.2 新アーキテクチャ開発法 …… 231
5.4.3 大規模ハードウェア設計開発技術 …… 234
5.4.4 大規模回路実装技術 …… 241
5.4.5 大規模ソフトウェア設計開発技術 …… 246
5.4.6 超高信頼化技術 …… 249

6. 研究開発の課題 …… 253
6.1 はじめに …… 253
6.2 研究開発課題 …… 254
6.2.1 述語論理マシン …… 255
6.2.2 関数型マシン …… 261
6.2.3 関係代数マシン …… 270
6.2.4 抽象データ型マシン …… 275
6.2.5 データフロー・マシン …… 283
6.2.6 新ノイマン・マシン …… 291
6.2.7 機能分散アーキテクチャ …… 299
6.2.8 ネットワーク・アーキテクチャ …… 309
6.2.9 高速演算マシン …… 317
6.2.10 データベース・マシン …… 323
6.2.11 高度マンマシン・コミュニケーション・システム …… 333
6.2.12 VLSIアーキテクチャ …… 333
6.2.13 知的VLSI－CAD …… 333
6.2.14 第5世代アーキテクチャのシステム化技術 …… 334

6.3　開発の目標と開発のステップ …… 339
6.3.1　開発の目標像 …… 339
6.3.2　開発ステップ決定の留意点 …… 340
6.3.3　開発ステップ …… 341
6.4　開発のすすめ方と研究体制 …… 345

1．第５世代計算機アーキテクチャ

1. 第5世代計算機アーキテクチャ

1.1 第5世代代計算機アーキテクチャの目標

1.1.1. 技術的背景と研究目標

第5世代計算機アーキテクチャ研究分科会は過去2年にわたり，1990年代に開発されるであろう新しい高性能計算機の機能および構成イメージを明確にし，その実現のために計算機アーキテクチャの観点から探求すべき研究開発課題の指摘に努力してきた。

本格的な電子計算機の誕生以来30余年を経た現在，関連する技術基盤は成熟し，更にハードウェアおよびソフトウェアの面で新技術への発展が期待されるようになってきた。また，著るしい応用分野の展開に伴って，"使いやすい計算機"を標題に，新しい問題解決手法の研究も人工知能やマンマシン・インタフェースの面から着実な成果が得られるようになってきた。

見方を変えれば，このような技術進歩は，現在の計算機の機能に対する不満あるいはユーザ・ニーズとのギャップを浮きぼりにする役割を果すようになったことを意味している。

このような背景の中から，第5世代計算機システムに少くとも期待されている機能を簡単に列挙すれば，

(i) 広範な応用分野に適応する構成を可能にする機能

(ii) ソフトウェア開発の生産性を改善する機能

(iii) 高度なマンマシン・インタフェースを備え，非専門ユーザでも使える機能

(iv) 意思決定支援など新しい問題解決手法を提供する機能

(v) 高度な分散処理・機密保護・高信頼性などを保証する機能

(vi) 価格性能比の優れたシステムを実現する機能

などが考えられる。

第5世代計算機アーキテクチャ研究の目標は，このような機能を実現する上で，計算機アーキテクチャ・レベルでいかに支援するかを追求することであるが，実際には上述のような新しい機能は計算機アーキテクチャの抜本的な変革なしに実現することは不可能に近いと考えられる。将来の計算機のあるべき姿を求める過程では，高度に飛躍した革新的（Revolutional）な構想が必要なことはいうまでもないが，それらは現在の技術を基盤として着実に進化（Evolutional）した構想と融合した形ではじめて実用的な意味をもつことを強調したい。

1.1.2. 現在の問題点と研究目標

過去の調査研究からも，現在の計算機は広範は応用分野において多くの問題点をかかえていることが明らかになっている。それらの問題点を列挙すれば，

(1) データ処理の増大と質の多様化

(2) マンマシン・インタフェースの改善

表１－１　現在の計算機の問題点と解決アプローチ

問題点	具体的な要因	解決するための技術またはシステムの例
(1)	大規模計算	超高速科学計算マシン，専用プロセッサ，データフロー・プロセッサ
	処理量の増大	マルチプログラム処理，マルチプロセッサ処理，分散処理システム
	ネットワーク化	コンピュータ・ネットワーク，ネットワーク・アーキテクチャ
	データベース化	データベース・マシン，分散データベース
	システム移行	エミュレーション，バーチャル・マシン
	新しい応用	画像処理，パターン認識，オフィス・オートメーション，推論機構，知識ベース，人工知能
(2)	音声・図形入出力	パターン認識
	使い易いプログラム言語・処理効率の改善	高レベル言語マシン，自然言語処理
	日本語データ処理	漢字処理
	使い易いファイルの実用化	単一レベル記憶，ファイルコンピュータ，データベース・マシン，問合わせシステム
	現場でのデータ処理	分散データ処理
	OS機能の充実と使い易さの改善	OS核・OS機能のファームウェア化
	無人化・省力化	OS機能の充実，診断機能，保守体制，自動運用システム
(3)	ハードウエアの信頼性向上	冗長技術，フォールト・トレラント・コンピュータ
	ソフトウエアの信頼性向上	構造化プログラミング，ドキュメント・システム
(4)	プログラム言語の改善 プログラム開発手法の改善	システム・プログラム言語，高レベル言語マシン 構造化プログラミング，開発管理技法
	プログラム・デバッグ・ツールの開発	デバッグ・サポート・ハードウェア
	アーキテクチャ・レベルでの機能支援	並列処理制御機構，連想処理，バックトラック制御機構
(5)	性能予測	モデル化，解析手法，シミュレータ，評価システム
	チューニング	モニタ機能，自動チューニング，適応化技術
(6)	コスト低減	ファームウェア化，LSI化，設計自動化
	性能向上	ファームウェア化，LSI化

(3)　信頼性の向上

(4)　ソフトウェア開発の生産性改善

(5)　システム評価の必要性

(6)　価格性能比の改善

などであろう。このような問題を提起している要因には種々のものが考えられるが，その主な

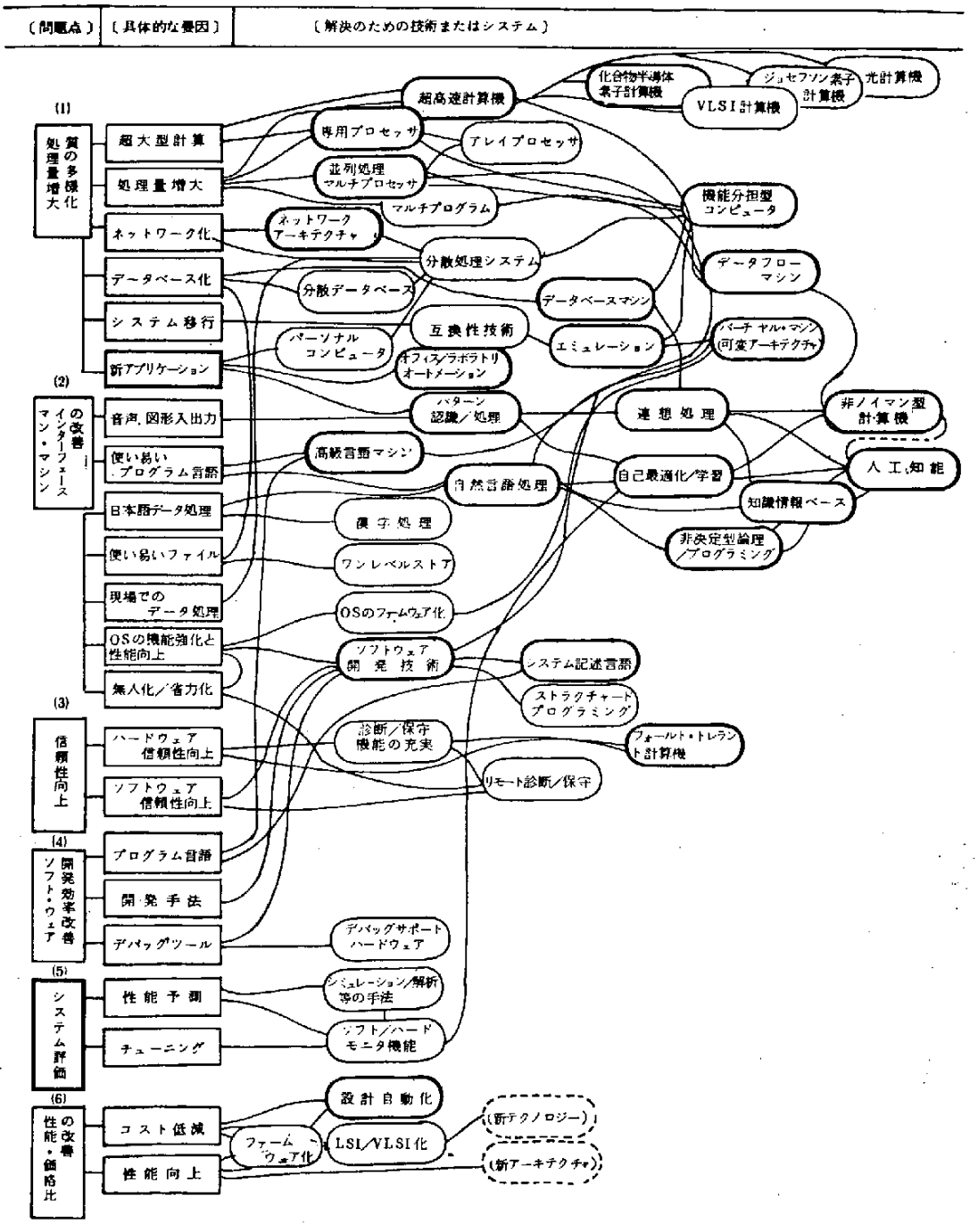


図１－１　現在の計算機の問題点と解決技術との関係

要因とそれを解決するために検討されている主要な技術との関係を表１－１に示す。

表１－１は現在の，いわゆる第3.5世代計算機あるいはその延長上にある第４世代計算機では実用的な観点から見て十分に応えることができない難題を多く含んでおり，第５世代計算機に対する期待を示しているといえる。

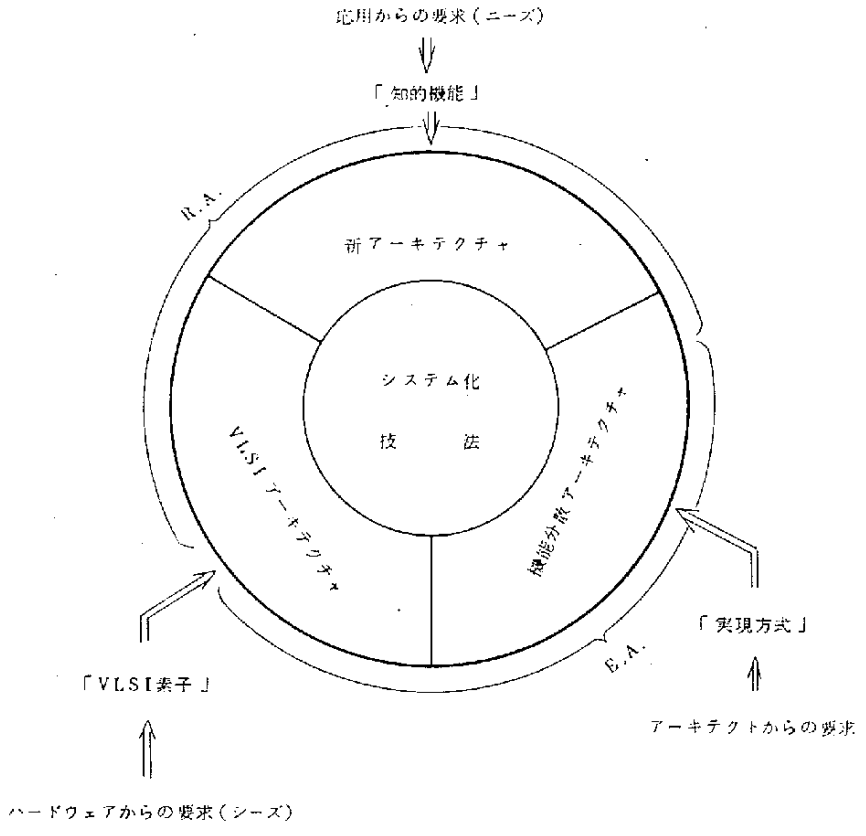
図1－１は現在の計算機の問題点と解決技術との関係を図示したものである。

第５世代計算機に課せられた今後の重要課題の一つは表１－１および図１－１に示した問題解決のために，計算機アーキテクチャ・レベルで支援すべき新しい技術を研究開発することである。ここで注意すべきことは，計算機アーキテクチャを単なるハードウェアの論理仕様というような狭い定義にこだわるべきでないことである。また，狭い考え方に基づく計算機構成法からも離脱する必要もある。従来の計算機開発に見られていたような構成素子（シーズ）を前提としたボトム・アップ的な考察も必要であろうが，応用（ニーズ）からの要求が計算機に及ぼす影響を無視することはできない。第５世代計算機においては，構成素子はＶＬＳＩ技術に依存することはいうまでもない。一方，応用からの要求はユーザにより，あるいは時代とともに変わってくるが，最近の要求は量から質へと変換しているように思われる。すなわち，従来の要求は大量のデータに対する高速な処理が基本的なものであったが，最近あるいは将来の要求はそのような量的な要求に加えて，"使い易さ"を追求した質的あるいは知的な要求を満足させる技術が中心課題となろう。換言すれば，第５世代計算機アーキテクチャの研究目標はボトム・アップからの要因である超ＬＳＩ素子の徹底利用とトップ・ダウンからの要因である知的機能の要求を総合的にシステムとして構造化することであるといえよう。

1.2　第５世代計算機アーキテクチャのとらえ方

第５世代計算機の課題は1.1節で示した技術的な問題点に対して，計算機アーキテクチャ・レベルから効率よく解決する方策を追求することである。このためには，前述のように，計算機アーキテクチャを単なるハードウェアの論理仕様としてとらえるだけでは十分でなく，より広い視野からとらえる必要がある。図１－２は第５世代計算機アーキテクチャのとらえ方を概念的に示したものである。

新しい計算機アーキテクチャを求める要因の一つはハードウェア，特に第５世代においてはＶＬＳＩの技術進歩がもたらすインパクト（シーズ）である。すなわち，ＶＬＳＩ技術を徹底的に活用することによって，ボトム・アップから計算機アーキテクチャが新しくなることが考えられる。一方，計算機の応用（ニーズ）からの要求も大きな要因の一つになっている。将来の応用分野からの要求は量的なものばかりでなく，質的な機能改善が中心になるものと考えられる。このことは将来の計算機にどの程度"知的な能力"を与えることができるかというトップ・ダウン的な要求課題に帰着する。要するに，前者はＶＬＳＩ技術の徹底利用の考え方に基



R.A.：革新的アーキテクチャ（Revolutionary Architectures）
E.A.：進化的アーキテクチャ（Evolutionary Architectures）

図１－２　第５世代計算機アーキテクチャのとらえ方

づく"ＶＬＳＩアーキテクチャ"であり，後者は知的機能の要求に応える"新アーキテクチャ"といえる。しかしながら，これらのアーキテクチャを計算機の中に組込み，実用化するためには，特徴ある機能をシステムとして効率的かつ容易に実現するための技術が必要になる。この技術を支えるのが"機能分散アーキテクチャ"である。見方を変えれば，機能分散アーキテクチャはアーキテクトの要望を満たす重要な技術ともいえ，計算機アーキテクチャをミドル・レベルからとらえたものと考えることができる。

これら大別して３つの重要なアーキテクチャには従来の，いわゆるノイマン型アーキテクチャから大きく脱皮した全く新しい概念に基づく，いわば"革新的アーキテクチャ（Revolutiouary Architectures）"と既に存在する考え方を発展させ，効率的な実用化を主眼とする，むしろ"進化的アーキテクチャ（ Evolutionary Architectures)"が共存することになる。もちろん，両アーキテクチャを明確に区別することは困難であり，"革新的"あるいは"進化的"という表現は概念的な呼称にすぎない。しかしながら，ここで注意すべきことは，革新的なアーキテクチャのみが研究開発に値し，進化的なアーキテクチャの開発は容易なことを意味するものでないということである。両者は相対立するものでもなく，また一方が他方の前段階的な役

割を果すものでもない。むしろ互に補いあう密接な相互関係をもっているというべきであろう。したがって，両者は進行して，積極的に，かつ新しい見方で研究開発しなければ互に有効な実を結ぶことにはならないと考えられる。

第５世代計算機には全く新しいアーキテクチャをはじめ素子，構造，実装など多くの未知，未経験の研究課題が含まれている。したがって，第５世代計算機の開発には複雑なシステムを構築するという前提に立って，新らしい“システム化技法”の体系化を急ぐ必要がある。換言すれば，上述の３つの主要なアーキテクチャをシステムとして実用化するためには新しい観点からとらえたシステム化技法を確立することが重要な課題になっているといえよう。

2．新アーキテクチャ

2. 新アーキテクチャ

2.1 計算機への要求と新アーキテクチャ

電子計算機が世に出てから，約30年（ＥＮＩＡＣの完成が１９４６年），ＩＢＭが，３６０度の全方位にわたって適用できるといううたい文句とともに開発し，一世を風靡した汎用計算機System/360 の設計開始から，２０年（正式発表は，１９６４年４月７日）が，経過した。(注) この間に，計算機をとりまく環境は，激変したといってよい。特に，その応用分野の拡大は，目をみはるものがあり，現在では，計算機は，初期の用途であった数値計算の分野が目立たなくなる程，ありとあらゆる領域で用いられている。これは，マイクロ・コンピュータなどに象徴的にみられる集積化技術に多くを負っている。応用の高度化，複雑化の面では，データベース処理や，ロボットなど，計算機をインテリジェントな人間の助手として機能させるようなものが現れてきた。また，造船や航空機，プラントの設計などにおいては，計算機によるシミュレーションによって，設計の時点で，それらの性能が検討できることなどから，必要不可欠の道具としての位置を占めるようになった。

これらに代表される応用分野では，ソフトウェアの巨大化，複雑化や，演算処理能力，記憶容量の不足などのハードウェアの限界が，大きな問題となっている。現在の計算機アーキテクチャは，フォン・ノイマンにより，その骨格が定められて以来，多くの改良が行なわれてきた。しかしながら，その改良も，限界に近くなっており，今後の１０年間といった将来を考える場合，ノイマン型から脱皮した新しいアーキテクチャを求める必要が生じる。このような主張は，ＩＢＭのJohn Backusらをはじめとするソフトウェアの研究者によってなされている。彼らは，巨大化，複雑化したソフトウェアをいかに作成し，またそれを信頼性の高いものにするか，保守なども容易なものとするかを目的として研究をすすめており，彼らの主張は，将来の計算機アーキテクチャを考える上で，傾聴すべきものである。Backusは，既存のプログラミング言語が，ノイマン型の計算機から離れられず，それが，ソフトウェアの問題を難しくしていると主張し，新しい計算モデルと，それに基づくアーキテクチャを要求している。

ここでは，これらの議論も含め，ノイマン型計算機とこの後で述べる新アーキテクチャとの対比を行い，新アーキテクチャの特徴を述べる。

(1) 計算機をとりまく環境とアーキテクチャ

現在の計算機の骨格をなすノイマン型アーキテクチャが考えられ，設計された当時の周囲の条件を考えると，次のような項目が挙げられる。

（注） Saul Rosen：Electronic Computers；A Historical Survey, Computing Surveys, Vol. 1, No 1, 1968

a) 周囲条件（ノイマン型の考えられた当時）

① 実装技術の未熟さ

- 素子（真空管やトランジスタも含めて）の信頼性の欠如，素子のサイズが大，高価
- アーキテクチャ技術の未成熟（アイデアはあったが，実装できないといった場合も多い）
- 入出力装置を含む周辺エレクトロニクス技術の未成熟

② 適用範囲が限定されていた

- 数値計算が主な用途（弾道計算，レンズの設計，微分方程式，統計処理，金銭管理など）
- 使用者（プログラマ）は，一部の優秀なスペシャリスト

このような条件のもとで，設計されたアーキテクチャは，何にもまして，実装技術の都合を最優先したものとしなければならない。この条件下では，ノイマン型アーキテクチャは，最良に近い解であった。その特徴は次のように言えよう。

- 単一プロセッサと単一メモリ
- 逐次実行とプログラム内蔵
- 集中した孤立型システム

当然のことながら，ハードウェアは，きわめて簡素にまとめられ，それでいて，十分な汎用性を備えていた。

その後，ハードウェアの進歩は，この骨格にいろいろな改良を加えた。プロセッサとメモリ間を結ぶ通信路が，高速化の妨げとなることから，キャッシュ・メモリを付加した。（この問題は，Backusが，フォン・ノイマン・ボトルネックと呼んだものである。）また，命令先取りや，命令実行のパイプライン化により，逐次実行の原則内で，できる限りの並列処理を実現しようとした。

このような努力は，ノイマン型の原則を守った上での改良であったが，実装技術の進歩にも支えられ，計算機の性能を飛躍的に向上させた。この結果，その適用範囲は拡大し，大きなシステムが誕生し，今や，人間にとってのインテリジェントな助手としての機能，それは，文字や音声，図形，より自然な言語を用いて会話できることが望まれるが，そのような機能までも，期待されるに至っている。

ハードウェアの面では，このような改良が行なわれたが，ソフトウェアの面からみると，依然として，逐次実行と単一プロセッサ，単一メモリとしての姿を変えてはおらずFORTRANなどの高級言語においてさえこの原則は守られている。そしてこれは，必ずしも，高度で複雑なソフトウェアを作成する基盤としては，望ましいとはいえないことが，近年，明らかになってきた。この問題は，ノイマン型のプログラミング・スタイルの弊害として議論されている。

単一プロセッサとしての，ハードウェアの性能は，回路内の信号の伝搬速度などにより限界に近づき，この解決策として，多くのプロセッサから成る複合体を目ざすようになり，プログラミング・スタイルの問題は，さらに深刻さを増した。すなわち，多数のプロセッサから成るシステムのプログラミングに，単一，逐次型を原則とする形式のプログラミングを適用したわけであり，無理があるのは当然である。これらは，ノイマン型のアーキテクチャの前提条件の枠組を越えたものであり，新しい前提に立った新アーキテクチャを求める動機ともなっている。

それでは，現在の計算機をとりまく周囲の条件は，いかなるものであろうか。これらは，次のようにまとめられよう。

b) 周囲条件（現在および近い将来）

① 実装技術の成熟

- ＶＬＳＩの実現さえも目前にせまった集積化技術の進歩
- 新素子の可能性（ジョセフソン接合素子，ガリウム砒素による素子，記憶素子としての光デバイス等）
- プロセッサの小型化に伴う機能分散型アーキテクチャの進歩
- 通信技術の進歩に伴うネットワーク化などの技術の普及
- 高度な並列処理に適したデータフローマシン理論など新アーキテクチャの種（タネ）の出現
- ソフトウェア工学などから出された，データ抽象などの新概念に基くプログラム構成手法の提案とアーキテクチャへの要求
- プログラムの検証，合成手法の研究
- 計算の基礎理論の進歩と新しい計算モデルの提案
- 人工知能，知識ベースなど高度な処理手法の基礎の充実

② 応用分野の飛躍的拡大

- 数値計算の範囲の拡大（ギガＦＬＯＰＳコンピュータの要求）
- 記号処理（事務処理，文書管理，データベース管理）の需要の拡大
- 分散データベースなど広域分散化したシステムの必要性，巨大システムの出現
- 人間のインテリジェントな助手としての役割（コンサルテーション・システム，ＣＡＥ／ＣＡＤ，ＯＡ(注)など）の要望，知識の利用
- 使用者の一般化と，素人の利用を前提

このような周囲の条件は，ノイマン型の考えられた時点のものとは，大幅に異なり，今後の社会の知識集約化や単純労働やルーチンワーク的な部門の労働力の不足を考慮すると，これらの条件は，ますます拡張されるものと考えられる。応用の高度化，複雑化は，ノイマン型のプ

(注) CAE：Computer Aided Engineering
CAD：〃 〃 Design
OA ：Office Automation

ログラミング・スタイルや，簡素なアーキテクチャを前提とする限り，ソフトウェアの負担を増し続け，現在抱えている問題が，ますます深刻になるものと考えられる。

新アーキテクチャを考える上での前提は，これらの問題を背景としたものでなければならない。その第一としては，応用面での都合を最優先したアーキテクチャであること，第二は，ＶＬＳＩや新素子などの革新的な実装手法を生かしきれるアーキテクチャであることであろう。

これをもとに，新アーキテクチャの特徴を挙げると次のようになる。

a) 数学的，論理学的基盤に立脚した計算モデルの採用（述語論理型や関数型のモデル）

これにより，次のことが可能となる。

① 巨大で複雑なシステムの整構造化が行いやすい。

② より高レベルのプログラミング言語が持てる。（素人向きの自然言語に近いプログラミング言語などとも対応をつけやすい）

③ プログラムの検証・合成が行い易い。

b) 並列処理を基本的に含む実行メカニズムを持つ（データフロー型のメカニズムなど）

① 高度な多重プロセッサ・システムを前提とした実行メカニズムである。

② 関数型プログラミング言語など，整った形式を持つ言語と直接の対応関係を持つ。（セマンティック・ギャップの解消）

③ ＶＬＳＩを前提とし，多くのプロセッサを含むことができ，処理能力の限界を押し上げられる。

c) ソフトウェアの作成の能率を改善し，信頼性を高められるメカニズムを有する。

① ソフトウェアのモジュール化，階層化を，抽象データ型アーキテクチャによりサポートする（抽象データ型マシンなど）

d) データベース管理を能率よく実行するアーキテクチャを含む（関係代数演算のためのメカニズム）

e) ネットワーク構成が可能な機能分散型のシステム・アーキテクチャを有する。

① パーソナル化したマシンのネットワーク構成とし，ＯＳや関連ソフトウェアの肥大化を防ぐ。

② ソフトウェア，ハードウェア資源の共有，交換を容易とする構成をもつ

③ ダイナミック・アーキテクチャやＵＨＭ（ユニバーサル・ホスト・マシン）などのシステム構成技術により，柔軟性，拡張性とともに，高いコスト・パフォーマンスを実現する。

④ 分散化により，一部の故障が，他に波及しない信頼性の高いシステムとなる。

f) マンマシン・コミュニケーションの充実化

パターン情報を扱う専用プロセッサと，新アーキテクチャによるマシン，さらに，推

論や知識ベースを扱うソフトウェアを組合せて，次のような機能を実現する。

① 自然言語に近い形の言語による対話

② 文字，図形の混在する文書の処理

③ 音声，図形を補助手段とするデータの入出力と対話。

④ マシン側よりのガイダンス（案内）を基本とした交信，対話。

以上のような特徴は，見方を変えれば，新アーキテクチャに対する要求事項でもある。そして，このような新しい機能は，現在のプログラミング言語よりは，数段高い，より応用目的に適した，人間向きのプログラミング言語の形で，ユーザに対して提供される。そこでは，現在のソフトウェア工学や人工知能など，知識情報処理の研究の成果が，より簡素に，また，能率よく実用的なものとして実装されるはずである。そのような，いわば，第5世代のソフトウェアとの組合せにより，新アーキテクチャは，真価を発揮し，上記のf)で述べたような，一般の人でも容易に使用でき，よき，人間の助手としての能力を与えられることが期待される。

2.2 新アーキテクチャへのアプローチ

１９９０年代の初期にプロトタイプができ，その後，実用化がすすめられ，その時代の計算機に対する高度な需要をまかなうのが，新アーキテクチャに基づく計算機システムの役割と仮定し，その新アーキテクチャへのアプローチを考える。これはもとより，極めて難しいテーマであるが，最近の計算機科学は，将来の計算機アーキテクチャについてのいくつかのヒントを与えてくれている。これらは，ここ２０数年にわたるノイマン型計算機を中心とした使用経験にもとずく問題点の整理と，反省であったり，知識情報処理のような新しい分野からの要請であったりする。このようなものの中から，新しいアーキテクチャの種となるものを見い出し，そこをアプローチの出発点とするのが妥当であろう。ここでは，次に述べる６つのものを種として選ぶこととした。これらは，現時点では，いずれも別個のテーマとして研究されているが，将来のアーキテクチャの満たすべき，重要な機能または重要な構成要素である。

① 述語論理マシン

② 関数型マシン

③ 抽象データ型マシン

④ 関係代数マシン

⑤ データフローマシン

⑥ 新ノイマンマシン

これらは，それぞれ独立のマシンとして扱われているが，将来においては，相互に融合するものもあるであろうし，また，独立に発展し，組合されて使用されるものもあると考えられる。

相互関係を簡単に説明すると，まず，①と②の述語論理型マシン，関数型マシンは，アーキ

テクチャを計算モデルおよび言語仕様の面からみたものといえる。述語論理型マシンは，推論の基本となる一階述語論理を基本とし，これをサポートするには，どのようなアーキテクチャが必要かというトップダウン的なアプローチを行う研究課題である。関数型マシンも同様であるが，述語論理型マシンよりも，対象とする計算モデルのレベルは低いものとできる。④の関係代数マシンも，数学的基盤に基づき関係代数をインタフェース言語とするマシンであるが，その処理は，集合演算を目的としており，将来のデータベース・システムの中核となると考えられるものである。将来の計算機システムでは，処理のほとんどが，適用分野の規則や作業手順などを主とする知識に関するものとなることが予想され，データベースの処理の比重が高まる。このことから，①と②のマシンと，④の関係代数マシンは，組合せて使われる等，きわめて密接に関連するものである。

これに対し，③の抽象データ型マシン，⑤のデータフローマシンは，将来のアーキテクチャを，演算処理の実行メカニズムやソフトウェアの構成手法などから見たアプローチといえる。抽象データ型マシンは，巨大で複雑なソフトウェアの実装をいかにしてアーキテクチャ側からサポートするかに重点をおき，将来の計算機のメモリの構造やプロセッサの機能を求めることが中心課題であろう。データフローマシンは，高度な並列処理の実行を目的とし，並列処理を基本的に含むデータフローモデルに基づくアーキテクチャの実現を目標とするものである。これら２つのマシンは，将来において，①②④のトップダウン的にアプローチされる３つのマシンの主要な実行メカニズムとして取り込まれる可能性がある。

⑥の新ノイマンマシンは，従来の改良された点を再整理し，その簡素なメカニズムを生かし，将来の計算機システムの重要な要素として，とり込まれると考えられるものである。従来，非ノイマン機能としてとり上げられてきたものの多くは，①～⑤のようなマシンと比べると，もともとのノイマン型マシンの改良として位置づけた方が適当なものも多く，これらは，この新ノイマンマシンの中で論じられる。新ノイマンマシンの重要な役割の一つは，既存システムから，新システムへの移行をスムーズに行えるようにすることであり，ＵＨＭを含むダイナミック・アーキテクチャは，移行性，互換性のための重要な技術でもある。

このほか，新アーキテクチャの特徴であげた，機能分散システムや，マンマシン・コミュニケーションについては，第３章で述べる。

以上の６つの種が，どのように成長し，どのような果実として結実するかは，多く今後の研究にまたねばならない。しかし，それは多くの期待をかけるに十分なものと思われる。

以下，順に，それぞれのマシンの目的と効果，動作原理，実現のための技術項目などを述べる。

2.2.1　述語論理マシン

(1)　述語論理マシンの目的と効果

論理は数学や哲学などをはじめとして，あらゆる分野で厳密な議論をする場合のことばとなるものである。その具体的な体系としては，命題論理，述語論理，様相論理などを代表にさまざまなものが知られている。

述語論理マシンは，述語論理をプログラミング言語として合理的に利用できるようにするためのアーキテクチャを追求し，具現化することを目的にした計算機である。したがって，その存在意義はつぎの２点に同時に依存していることになる。

— プログラミング言語としてみた述語論理の有効性。

— 述語論理プログラミング言語に適したアーキテクチャの可能性。

これらについてはすでに，参考文献1)に基礎的な議論がなされている。ここではあとの説明に必要な範囲での論点の整理をおこなうことにする。

① 述語論理プログラミングの有効性

プログラミング言語の有効性は複雑な利害得失を背景にして成立するものである。したがって，議論をあまり模型化するのは危険であるが，プログラミング言語としてみた述語論理の有効性の要点はつぎのとおりである。

述語論理は，形式言語としては自然言語に最も接近した記述力をもっている。もし，人間から計算機への指示を，まったく誤解の生ずる心配のないことを直接確かめうるかたちでやるとすると，そこでは形式言語の使用が不可欠の条件となる。自然言語には多くのあいまいさの現象がみられ，これらを完全に回避したり，解消したりすることは期待できない。述語論理には厳密な文法と意味の構造体系が与えられているので，あいまいさの余地がないばかりではなく，プログラムの正しさの保証に役立つ理論的な枠組も用意されており，実際にそれを活用するための研究も成果を上げつつある。

述語論理が厳密なかたちでの自然言語に近い表現力をもっていることは，将来のプログラミングのあり方を考えるとき，きわめて重要な意味をもつことになる。まず第一に，述語論理は完全に利用者志向のプログラミング言語であるということである。そこには本質的に機械との関連でしか説明のつかないような言語特性は皆無であり，逆に理論的な要請に偏した不自然さも避けられている。述語論理は，思考のプロセスを合理的に再構成して客観化するための規準的な枠組となることをその使命としているのである。つぎに，プログラミング言語としてみた述語論理は，記述型で，非決定的プログラミングが可能な高水準言語であることである。この特質は人工知能の分野での研究からも明らかなとおり，計算機を単なる計算の道具としてではなく，知識を高度に活用するための媒体として用いるときに，その威力が現われてくるものである。したがって，述語論理は知識社会の時代に最も適したプログラミング言語であるということができる。プログラムは問題を解くためのアルゴリズムとデータ構造を記述したものであり，アルゴリズムは，さらに，問題を解くのに用いられる知識

の内容を示す論理部と，この知識をどのように利用していけばよいかという解法上の方向づけを示す制御部とから構成される。論理部がアルゴリズムの意味を支配するのに対して，制御部はアルゴリズムの効率にだけに関係する。通常のプログラミング言語では２つの部分は非分離的に混在しているのに対して，自然言語の場合には分離度がある程度自由に変えられるようになっている。もし，これらを完全に分離したかたちで指定できるようになれば，プログラムの構造は見通しがよくなり，誤りが減るばかりではなく，効率の改善やプログラム変更の作業が大変容易になる。これを述語論理によるプログラミングの第三の特徴としてあげることができる。

このほか述語論理プログラミングの有効性を示す材料は数が多い。反対に，その欠点あるいは有効性の限界については，第三の特徴にあげた論理部と制御部の完全分離が，目標に向かって最短距離を進むようにすべきプログラミングの使命にとってマイナスとする指摘があり，最も大きな議論の焦点になっている。（このほか，データ構造の表現力などにも弱点があることが指摘されている。）

② 述語論理向アーキテクチャの可能性

述語論理プログラム，詳しくは述語論理で書かれたプログラムの実行は，定理の証明というかたちで実現される。その機構は，問題を解くのに用いるべき知識は公理の体系として表現し，問題そのものは関係式R（s，x）を満足するxを求めるという形式で与え，計算はこの関係式の否定形$\forall x \neg R(s,x)$と先に与えた公理体系が矛盾することを証明するというものである。しかも，この証明では，その矛盾がx＝aで発生することを突き止めることになるというわけである。

計算機による定理証明は非常に長い歴史をもっているが，その中心的な課題は証明の過程で発生する探索のための選択枝の数を，いかに少なく抑えうるかということであった。これに対する十分な保証がない限り，述語論理プログラミングは成立しえないことになるが，それはいわゆる導出原理[文献2)]の発見によって一応の見通しがえられている。導出原理が有効なのは節形式[文献1)]の一階述語論理の範囲に限られるが，実際上の問題では高階述語論理を必要とすることは稀とされ，また，一般の一階述語論理は節形式に変換可能であることが知られている。

これで述語論理プログラミング実現の１つの路が拓かれたが，具体化のためにはさらに他言語に見劣りしない実行効率と既成のプログラミング概念への親和性が要求される。そのために考えられたのが，一般的な節形式の取扱いを止め，Horn節とよばれる形式だけに制限することである。一階述語論理で表現しうる問題はすべてHorn節形式に表現し直すことができるうえ，Horn節を通常のプログラミング言語での手続き形式として直接読み換えうることが知られている。この制限の結果は，PROLOGという言語に定式化され，インタプリ

タやコンパイラの形式で普通の計算機上に実現されている。これらはすでに代数的記号処理，構造設計，薬品開発，コンパイラ作成などの大規模プログラムに利用され，実行効率においても既成言語に十分対抗しうるものであることが例証されている。さらに，自然言語処理，データベースなどへの応用についても効果的な活用方式が開発されつつある。

以上のような経過から，述語論理プログラミングの主な関心は現在ＰＲＯＬＯＧに集中しており，述語論理マシンとしても，まずＰＲＯＬＯＧマシンの実現が最初の課題になる。後に詳しく説明するように，ＰＲＯＬＯＧのプログラム構造，データ構造，制御構造は，すべて従来の言語とは大幅に異なっているので，それらに適合したアーキテクチャのあり方は現行マシンのそれとは異なっているはずである。この適合化に成功できればＰＲＯＬＯＧプログラム実行効率の格段の向上が期待される。これは述語論理プログラミングの応用領域，すなわち知識情報処理の飛躍的な発展の契機ともなりうるものである。

(2)　述語論理マシンの機能

述語論理マシンは，述語論理形式で記述されたプログラムを合理的に実行する機能をもつ計算機である。述語論理の形式にはさまざまなバリエーションがあるが，ここでは前項に述べた経緯から専らＰＲＯＬＯＧでの定式化に従うことにする。ＰＲＯＬＯＧはいわゆる高水準言語であるので，それを合理的に実行するための機能としては，コンパイル方式によるもの，インタプリタ方式によるもの，直接実行方式によるものが考えられる。

①　ＰＲＯＬＯＧ言語

ＰＲＯＬＯＧのデータ構造はつぎの通りである：

＜項＞：：＝＜定数＞｜＜変数＞｜＜複項＞

定数には整数や素項（文字列に相当）が含まれている。複項は構造データに相当するもので，関手の後にいくつかの引数を並べた表現をとる。たとえば，ｐｏｉｎｔ（Ｘ,Ｙ,Ｚ），＋(Ｘ，Ｙ)，＞(Ｘ，Ｙ)などである。引数は任意の項でよい。

ＰＲＯＬＯＧのプログラム構造はつぎの通りである：

＜ゴールまたは手続呼出し＞：：＝＜複項＞

＜頭部＞：：＝∅｜＜ゴール＞.

＜本体＞：：＝＜ゴール＞*.

＜節＞：：＝＜頭部＞：－＜本体＞.

＜プログラム＞：：＝＜節＞*

ゴールは複項としての形式をもち，それがどちらの意味で用いられているかは文脈だけで決められることになっている。ゴールとしての複項の関手を述語とよぶ。述語にはいわゆる組込関数に相当するものが用意されるのが普通で，それらは特に基本述語とよばれることになっている。基本述語には算術演算，文字列操作，制御，入出力，データベース操作，メタ

述語などがある。

ここでいう節はいわゆるHorn節である。"P：－Q,R,S."という節は"もし，Q,R,Sがすべて真であれば，Pも真である"あるいは"ゴールPが充足するには，ゴールQ,R,Sを充足せよ"という意味に解されることになっている。頭部が空の場合には，"？－P,Q."というように書き，"PおよびQを充足せよ"を意味し，プログラムの解くべき問題の提示に用いられる。本体が空の場合は"P."とのみ表記し，Pが恒に真であることを示す。各ゴールには変数が含まれているが，それらは節ごとの全称的な束縛変数として扱われることになっている。

通常のプログラミング言語での手続きに相当するのは，頭部ゴールに同じ述語をもった一連の節の集まりである。たとえば，"concatenate"という手続きに相当するのは，

concatenate（〔×|L1〕，L2；〔×|L3〕）：－concatenate（L1，L2，L3），

concatenate（〔〕，L，L）。

（PROLOGプログラムの具体例については付録参照。）

PROLOGの制御構造はつぎのように与えられている：1つのゴールを実行するというのは，プログラム内を探索し，そのゴールに一致または統合化しうる頭部ゴールをもった最初の節を検出することである。ここで統合化というのは，問題となる2つの項について，それぞれの変数に適当な項を代入し，両方が一致するような代入例のうちの最も簡単なもの（他のすべての代入例に，その一部として共通に含まれるようなもので，一意的に定まる）を求めることである。このようにして一致のとれるものがあった場合には，その節全体に同じ代入を適用してできる新しい節の活性化をおこなう。すなわち，その本体の各ゴールを左から右の順にその実行を開始する。あるゴールについて一致が見出せない場合には，後戻り，すなわち，一番最近の活性化を無効にし，それに先立っておこなわれた代入からもとのゴールを復元し，そのときの節に続く節から統合化の探索を再開する。こうした探索を続けるうち，どこかで本体のない節と一致がとれることがある。それが解の1つを与えることになる。可能なすべての探索を実行すれば，ありうる解がすべて得られることになる。

以上から明らかなように，PROLOGの制御構造はパターン・マッチングと自動的な後戻りを基礎にしている。後戻りの制御はプログラムの実行効率を大きく左右するものであるので，それを自動的にではなく，プログラマに指定させた方がよい場合がある。そのための機構が基本述語の中に用意されている（"／"）が，ここでは省略する。（プログラム実行の具体例については付録参照。）

② PROLOGプログラム実行機能

ｉ．実行機能実現のための課題

PROLOGには従来のプログラミング言語には見られなかった多くの特徴があり，そ

れをサポートすべき計算機に新しいアーキテクチャ上の課題を投げかけている。

a. プログラムとデータが同じ形式をもつ。

b. データ構造は一般性が高く，複雑なものも可能。

c. 構造データに対する操作はセレクタやコンストラクタ機能ではなく，パターン・マッチングによっている。

d. 手続きは複数の入力と複数の出力をもちうる。

e. 手続きの引数は，入力用／出力用の区別が呼出しごとに変わりうるようになっている。

f. 手続きは非決定的であり，後戻りによって高水準の繰返しが実現される。

g. 統合化は単なるパターン・マッチングとは異なり，変数の扱い方に特徴がある。

ii. 実行方式

高水準言語の実行方式としては，前述の通りコンパイル方式，インタプリタ方式，直接実行方式の３つが区別されているが，前の２つの方式は既成のアーキテクチャ上での実現に関心がある場合の区別であり，最後のものは逆に言語に合わせてアーキテクチャを最適化することに関心がある場合のものである。それゆえ第５世代に適用すべきものは直接実行方式ということになるが，この方式の場合でも実現手段としてのハードウェア技術の制約を定式化した一定のアーキテクチャ・モデルを基礎とせざるをえず，したがって，結局は前の２つの方式に帰着することになる。

一般に，同じプログラムが繰返し使用されることが多いという状況下ではコンパイル方式が，反対に実験的なプログラミングの比重が大きい場合にはインタプリタ方式が，それぞれ有利とされている。現在稼動中のものでの実測結果によれば，ＰＲＯＬＯＧの場合にも事情は同じであることが裏付けられている。文献4)

Absolute times Test	Marseille Interp. IBM	Edinburgh Compiler DEC	Edinburgh Interp. DEC	Waterloo Interp. IBM	IC-Prolog Interp. IBM
1. Basic speed.	5.02	0.08	0.74	0.20	16.67
2. Speed and printout.	45.11	0.40	1.02	0.33	17.77
3. Backtracking.	7.53	0.36	2.36	1.35	85.03
4. Indexing.	-	0.08	0.46	0.51	29.87
5. Read-in.	(462)	147.0	5.64	3.73	39.06
Relative Times					
1. Basic Speed	25.10	2.32	21.46	1.0	84.35
2. Speed and printout.	136.70	7.03	17.93	1.0	53.83
3. Backtracking.	5.58	1.55	10.14	1.0	62.99
4. Indexing.	-	0.91	5.23	1.0	58.57
5. Read-in	123.86	228.6	8.77	1.0	10.47

(3) 述語論理マシンの処理方式（動作原理）

ＰＲＯＬＯＧマシンの基本要素とその動作内容をモデル化するとおよそつぎのとおりである。

① プログラム／データの内部表現とメモリ機構

ＰＲＯＬＯＧプログラムは節の集合である。一般の節は１個のゴールからなる頭部と，任意個のゴールからなる本体とで構成されている。頭部ゴールの先頭にある述語は，活性化されたゴールによる検索のキーになるものであるから，節レベルでのメモリ構造はアレイやハッシュ・テーブルなど高速アクセスの可能なものが必要である。また，プログラムの部分構造として手続きがある。手続きは頭部ゴール述語の共通の節の集まりである。手続き内の節レベルについても，頭部ゴールの引数にもとづくアレイ構造やハッシュ・テーブルの利用が高速化に有効である。

ＰＲＯＬＯＧではプログラムとデータが共通の構造をもち，ゴールが通常言語でのレコードの記述として用いられるなど，両者に対するアプリオリな区別はない。したがって，マシン内部での取扱いも基本的には同等であるべきである。基本的なデータ構造は木構造であり，これを効率よく処理しうるような構造体メモリによるサポートが望ましい。

② 統合化機構

ＰＲＯＬＯＧプログラムの実行過程は統合化（パターン・マッチング）操作の繰返しである。統合化では同じ述語をもった複項が対照され，変数に対する代入や値の参照がおこなわれる。統合化は同じ複項に対して複数回，場合によっては再帰的に適用される。したがって，各節は原型を常に維持しておく必要があるとともに，変数値のためのメモリ領域はスタック構造化が必要である。詳しい議論は省略するが，基本的には２ないし３個のスタックが必要である。

③ 基本述語実行機構

基本述語は他の述語と異なって別のゴールを活性化することはなく，通常の計算機での命令動作に似た作用をおこなうものである。したがって，その実現には従来の考え方がそのまま適用できるはずである。

④ 後戻り機構

ある探索が失敗に終ったとき，その試行の出発点まで後戻りして，つぎの可能性に移る必要がある。このとき重要なことは，その試行のために用いたメモリ領域の解放を効率よくおこなうことである。

⑤ コンパイル機構

実行方式としてコンパイル方式をとる場合には，当然コンパイラの実現とコンパイリングに対するサポートのための機能についての配慮が必要である。ＰＲＯＬＯＧにはコンパイラを記述するのに十分な表現能力があることが実証されている。問題はその効率であるが，生

成されるオブジェクト・コードは通常のリスト処理言語に比して遜色がない。しかし，コンパイラのスペース利用効率に難があるため，大きなプログラムのコンパイルには限界があり，何らかの工夫が必要である。[3)]

⑥ 並列処理機能

ＰＲＯＬＯＧプログラムの実行を高速化するためには，何らかのかたちで並列処理を導入することが望ましい。並列処理の可能性としては，節レベルでの探索の並列化と，同じ節内での複数ゴールの同時活性化の２つが考えられる。この場合の主な問題点はメモリ・リミティッドになる可能性があること，共有変数の取扱いである。述語論理プログラムはデータフロー・グラフと密接な関係にあることが知られており，[8)]その事実を利用して述語論理マシンをデータフロー・アーキテクチャによって実現できる可能性がある。

(4) 実現上の問題点と研究課題

① 言語仕様

以上では述語論理プログラミングとして専らＰＲＯＬＯＧに焦点を合せてきたが，広い視野でみたときの言語仕様として現状のままのＰＲＯＬＯＧで十分かどうかは問題である。たとえば，PROLOGが効率を重視するあまり過度な単純化に陥っていて，第二のＦＯＲＴＲＡＮになる心配があるというようなものである。[5)]すなわち，表現能力不足の問題である。もう１つの批判は，ＰＲＯＬＯＧ固有の問題というより述語論理プログラミング一般に共通のものである。すなわち，述語論理プログラミングの基本命題である，

アルゴリズム ＝ 論理 ＋ 制御

の制御の部分に対する言語機能，特に解くべき問題に適合した強力な推論制御機構の不在という問題である。

表現能力拡張の方向としては，1)論理和と等号の導入，2)標準形式の導入，3) if & only if 形式の導入，4)様相論理への拡張，5)多層論理への拡張などの提案がある。これらの拡張に伴なって，導出原理そのものからの離脱が要請されるようになることもあり，問題の根は深い。

制御機能の強化についても，従来の人工知能言語研究からの成果をふまえて活発な提案がおこなわれている最中である。これを言語仕様として定着させるためには，実用での成り行きを待つ必要があるが，論理の部分とは分離したかたちで扱うことが可能であるので実験的な取扱いで処理できるはずである。

② 並 列 化

ＰＲＯＬＯＧプログラム並列処理の可能性については，すでに前項でみたとおりであるが，この可能性をアーキテクチャのレベルでの研究題目として整理するとつぎのようになる：

i. 並列計算モデル

例：データフロー・インタプリタ

ii. 並列制御方式

例：非決定性処理方式，資源管理手法

iii. 並列パターン照合機構

例：連想プロセッサ／連想メモリ，構造メモリ

③ 基本述語機能

実用の局面で頻出するような基本操作を，述語論理での原理的な定式化に忠実なかたちで実現するのでは，十分な高速性が期待できないことがある。たとえば，整数演算をペアノの公理のレベルにまで還元して計算しているような場合である。したがって，高速性が重要な応用分野に対しては，その分野での利用率の高い基本操作を基本述語として，直接実行できるかたちで実現できることが望ましい。これをどのような方式で可能にするかについては，言語仕様，効果などについて議論の余地があり，今後の研究に待つところが多い。[6),7)]

④ システム・モデルの設定と実現上の方策

述語論理マシンを具体化するためには，マシンの規模，利用目的，利用形態などが明確になっていなければならない。述語論理マシンの適用分野は非常に広汎であり，問題の規模についても特にアプリオリな限定を設けうるような事情は見当らない。したがって，開発で採りあげるべきシステム・モデルの設定は十分一般性を考慮しておこなうべきであると考えられる。しかし，限られた時間ならびに資源からみて，あまり手広く何種類ものモデルを期待することは許されない。このため，実現上の方策としては，まず小規模のもので，言語仕様の拡張など完成後の実験的変更にも耐えうる構造をもつものを早期に実現し，その成果を採りいれながら，並列処理機能を徹底的に盛り込んだ本格的な大規模マシンの開発をおこなうのが望ましい。

⑤ 述語論理プログラミングの可能性追求

"述語論理マシンの目的と効果"の項で説明したように，述語論理プログラミングにはプログラムの検証，プログラム自動作成など，従来のプログラミング言語では不可能に近かった課題に対する新しい可能性が含まれている。これらを追求して現実の機能とすることは，述語論理マシンそのものの開発に匹敵する意義をもつものということができるかも知れない。

(5) 実現の例と性能予想

現在までのところ述語論理マシンの実現例はない。ただ，マシンの性格からすると最近活発に開発のおこなわれているＬＩＳＰマシンでの実現例が，問題の所在と解決へのアプローチ，性能と価格などについての推定に参考となるところが多いと考えられる。（ＬＩＳＰマシンの実現例については，本報告書"関数型マシン"の項を参照。）[11)]

ＰＲＯＬＯＧ言語を通常の計算機で実現した場合の例については，"述語論理マシンの機能"

の項に若干の説明がある。これがＰＲＯＬＯＧマシンの実現によってどの程度に改善されると期待してよいか？ＬＩＳＰの場合にこれがどうなっているかを具体例でみると表（付録２）のようになる。[9]このデータについては多少疑問の余地があるが，一応数倍ないし１０数倍の改善率が期待できるとみてよく，専用マシンを開発する意味があることが保証される。

以上の議論を集約すると，開発すべき実現モデルのバリエーションはつぎのようになる。

言語		実行方式		規模		開発目的
ＰＲＯＬＯＧ 新言語	×	コンパイル インタプリート	×	共同利用（大規模） パーソナル（小規模）	×	パイロット 開発実験

これらの組合せのうち，必要な実現時期と要素技術を考慮に入れて，現実性のあるものを選別すると下記の３モデルに絞られることになる。

モデルⅠ：開発実験モデル

１９８５年完成を目標とするＰＲＯＬＯＧベースの小規模マシンで，汎用UHM（Universal Host Machine）上にファームウェアによって実現するものである。

性能目標は現行の通常マシン上にソフトウェアで実現されているものの約１０倍の水準である。コンパイル方式とインタプリタ方式の両方が利用可能である。

このモデルの開発目的は，ＰＲＯＬＯＧマシンひいては述語論理マシンの性能的な可能性を見極めるための実験台を提供することにある。

モデルⅡ：専用ファームウェア・ベース・マシン

１９９０年完成を目標とする述語論理マシンで，理論や言語研究をサポートするパーソナル計算機としての規模をもつものである。専用ファームウェア・ベース・マシン（記号処理向きＵＨＭ）としての形式に従うことを想定している。

性能目標は現行のものの約数１０倍から１００倍弱の水準であるが，インタプリタ方式としての性能に重点をおくものとする。

このモデルの開発目的は，商用パーソナル述語論理マシンのパイロット・モデルを提供することである。

モデルⅢ：並列述語論理マシン

１９９０年完成を目標とする高度並列化述語論理マシンで，大規模論理プログラムの高速実行処理のための共同利用向き大型機であり，データフロー方式の利用を想定している。

性能目標は現在の水準の数１００倍から数１０００倍の線で，主としてコンパイル方式での性能に比重をおくものとする。

このモデルの開発目的は，大規模述語論理マシンの実現可能性を実証すると同時に，その商用機のためのパイロット・モデルとなることである。

(6) 研究開発計画

前項に述べた３つのモデルの開発を中心にした研究開発計画の全容を以下に示す。

① 研究内容の枠組

- 言語開発と実装化*

（ＰＲＯＬＯＧ系） i. 拡張ＰＲＯＬＯＧ言語開発

ii. 実装化：コンパイル方式ならびにインタプリタ方式による実現

（新言語系） i. 新述語論理型言語開発

ii. 実装化：（同上）

*当初はPROLOG系と新言語系とを並行的に検討するが，最終的には一本化を目指すものとする。

- 並列化方式の追求

i. 基礎検討 { 並列制御方式（ＰＲＯＬＯＧ系並列化）
並列計算モデル（高度並列化）

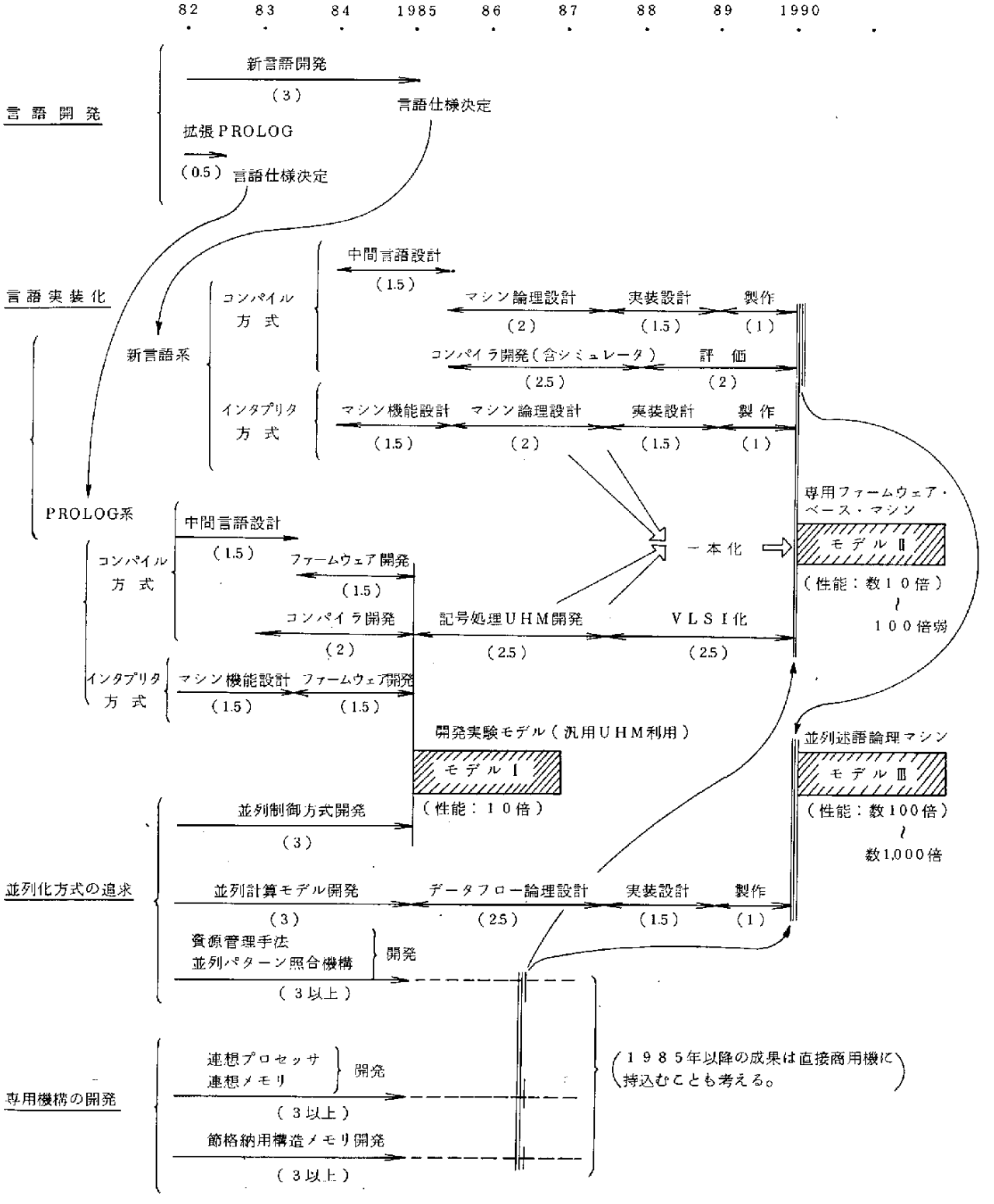
ii. 実装化検討 { 資源管理手法
並列パターン照合機構

- 専用的機構の開発

i. 連想機構 { 連想プロセッサ
連想メモリ

ii. 節格納用構造メモリ

② 研究開発スケジュール



(7) 参考文献

1) 日本情報処理開発協会：第５世代の電子計算機に関する調査研究報告書－マシンの理論－第４章論理プログラミングについて，

2) Robinson,J.A.: A Machine Oriented Logic Based on the Resolution Principle. J.ACM12(January 1965), pp.23-41.

3) Warren, D.H.D.: Logic Programming and Compiler Writing.Software-Practice and Experience, Vol.10,(1980),pp.97-125.

4) Moss, C.D.S.: The Comparison of Several PROLOG Systems, Proc. Logic Programming Workshop(July 1980),pp.198-200.

5) Mc Dermott, D.: The PROLOG Phenomenon,SIGARCH Vol.8,?(1980), pp.16-20.

6) 後藤滋樹：拡張ＰＲＯＬＯＧ言語について，情報処理学会第２１回全国大会(1980), pp.163-164.

7) Townley, J.A.: A Pragmatic Approach to Resolution-Based Theorem Proving, IJCIS, Vol.9,No.2(１９８０), pp.93-116.

8) Morris,P.: A Dataflow Interpreter for Logic Programs,Proc.Logic Programming Workshop(July1980),pp.148-159.

9) Boley, H.: A Preliminary Survey of Artificial Intelligence Machine, SIGARCH, Vol.8,No.?(１９８０), pp.21-28.

10) Kowalski,R. Logic for Problem Solving,1979,North Holland.

付録 1.　ＰＲＯＬＯＧプログラムの具体例とその実行

(1)　人口密度 ― データベース問合せ*

この例は，ＰＲＯＬＯＧにおいてはプログラムとデータが区別されていないことを示すためのものである。世界各国の人口（１００万単位）と面積（1,000平方マイル単位）からなるデータベースならびに人口密度（平方マイル当り）を計算する手続き density を与えて，いくつかの問合せに答えるようにするものである：

```
pop(china,825).          area(china,3380).
pop(india,586).          area(india,1139).
pop(ussr,252).           area(ussr,8708).
pop(usa,212).            area(usa,3609).
 ⋮                        ⋮
density(C,D) :- pop(C,P), area(C,A), Dis(P*1000)/A.
```

問合せの例として，人口密度をほぼ等しくする（５%以内の差）国々の一覧表を出す場合を考えると，そのためのデータベース問合せ手続き ansを用意したうえで，

$$ans(C_1,D_1,C_2,D_2) :- density(C_1,D_1),\ density(C_2,D_2),\ D_1 > D_2,\ 20*D_1 > 21*D_2.$$

問合せゴール $ans(C_1,D_1,C_2,D_2)$ を実行させればよい。

このプログラムの実行を等価なAlgol形式のプログラムにおきかえて示すとつぎのとおりである：

```
for C1 from 1 to N do
    int D1 := pop〔C1〕*1000/area〔C1〕;
    for C2 from 1 to N do
        int D2 := pop〔C2〕*1000/area〔C2〕;
        if D1 > D2 and 20*D1 < 21*D2
        then output(country〔C1〕,D1,country〔C2〕,D2);
    repeat
repeat
```

ここでの繰返しは，ＰＲＯＬＯＧの自動後戻りの機構によって実現されているわけである。

*Warren, D.H.D et al. : PROLOG - The Language and its Implementation Compared with LISP. in Proc. Symposium on AI and Programming Languages, SIGPLAN/SIGART, 1977

(2) 平方根計算プログラム*

```
quadrat(A,B,C,Realroots) : -
    discrim(A,B,C,D),quadrat1(A,B,C,Realroots).
discrim(A,B,C,D) : -mult(B,B,Bsquared),mult(A,C,P1),
    mult(4,P1,P2),add(Bsquared,D,P2).

quadrat1(A,B,D,[]) : -   -D<0.
quadrat1(A,B,D,[R]) : -D=0,add(B,MinusB,0),
    mult(2,A,TwoA),mult(R,TwoA,MinusB).
quadrat1(A,B,D,[R1,R2]) : -D>0,add(B,MinusB,0),
    sqrt(D,SqrtD),add(MinusB,SqrtB,Num1),
    add(Num2,SqrtD,MinusB),mult(2,A,TwoA),
    mult(TwoA,R1,Num1),mult(TwoA,R2,Num2).
```

（ここで大文字で始まるのは変数，add,mult,sqrtは基本関数である。）

*Bowen, K.A. : PROLOG, ?

付録 2. ＬＩＳＰマシンの性能

	Greenblatt LISP machine [CONS]	DECsystem-10 [KA10]	Improvement
LISP systems supported	LISP machine LISP [others adaptable]	MACLISP,INTERLISP,UCILISP..	see text
microprogram memory	16K 48-bit-words	none	absolute
instruction set	LISP-directed	not high-level lang. directed	higher level
system language	LISP/microcode	MACRO-10	easier
program swapping	microprogrammed paging	macroprogr. core shuffling	micro paging
word length	32 bits	36 bits	no
virtual address space	24 bits [16,777,216 addr.]	18 bits [262144 addr.]	6 bits [64 fold]
cache ['pdl buffer']	1K words 200 nanoseconds	none	absolute
main memory	core/semiconductor 64K - 1 Mega words 1 microsecond	core 64K-256K words 1 microsecond	semicond. 0K-768Kw. no
swapping memory	hard disk 16 Mega words 25 milliseconds	drum/disk arbitrary	no
run time [relative times for LUNAR system]	ellapsed real time	virtual run time with time sharing delays	3 fold [MACLISP] 6-12 fold [INTRLSP]
cost	80,000 $	300,000 $	220,000 $

2.2.2 関数型マシン

(1) 関数型マシンの目的と効果

① 目　　的

第５世代のコンピュータで重要になる知識ベースの実現とソフトウェアの生産性の向上には，関数型言語などの理論的基盤をもつプログラミング言語の利用が望ましい。一方，ＬＩＳＰとかＩＮＴＥＲＬＩＳＰ等で代表される関数型言語は次の理由から，従来のコンピュータでは処理速度が上がらない。

(a) 関数型言語処理で標準的に利用されるリスト構造データを順次に追う処理は従来のコンピュータは得意でない。

(b) 関数の呼び出し，特にリカーシブな関数呼び出しの処理は従来のコンピュータは得意でない。

また，要求される性能が現在のコンピュータの１桁～２桁上であるので，関数型言語処理に向いたマシンを新たに設計し，利用可能にする必要がある。

② 性能の実現目標

計算機のアーキテクチャの面から，知識情報処理をサポートするには，次の性能が望まれている。

(a) ネットワークの共通リソースとしての関数型マシン

現在の汎用コンピュータ（例えば，４ＭＩＰＳ）のリスト処理スピードの１０～１０００倍。

(b) パーソナル・マシンとしての関数型マシン

卓上型のマシンとして，現在の汎用コンピュータ（例えば，４ＭＩＰＳ）のリスト処理スピードの１～１０倍。

③ 効　　果

関数型マシンは関数型言語の利用者に役立つとともに，述語論理型言語等の言語の実行メカニズムとしての役を果たす。

これにより，知識ベースを利用可能なものとする。また，ソフトウェアの生産性の向上に貢献する。

関数型プログラミングとソフトウェアの生産性：

ＩＢＭ社のBackus氏は現在のプログラミング言語には次のような欠点があると指摘している。

現在の言語にはフォン・ノイマン・マシンの概念を引きついだ，フォン・ノイマン・プログラミング言語（言語のモデルがフォン・ノイマン・マシン）であり，

(a) プログラム作成者の思考が分断される。（Intellectual Bottleneck）

一語のデータを取り出しては，処理し，一語のデータを結果として格納することの繰り返し。

(b) ストラクチャード・プログラミングも，フォン・ノイマン・プログラミング言語に，ある種の秩序を入れたぐらいで，本質的な解答にはなっていない。

Backus氏はこの問題をＦＰ（Functional Programming：関数型プログラミング）で解決出来ると論じている。

(a) ＦＰでは，変数の値にhistory sensitivityがないのでプログラムの論理を考え易い。

— 関数合成が自然の形でやれる —

（call by name/value，変数のスコープ等に悩まされることがない。）

(b) 集合演算を考えることで，繰り返し文（for Ｉ＝1 to N)を排除できる。

従来のプログラミング言語の欠点を除くべく，Backus氏の指摘のように，ＦＰの技術の適用を検討すべきであろう。

但し，現在ＰＬ／ＩとかＣＯＢＯＬでプログラミングしている者が関数型言語（例えば，ＬＩＳＰ言語）に切り替えて，すぐに効果を上げられるとは考え難い。「教育」とか「慣れ」の問題の他に次の２点を研究する必要があると考える。

(i) 関数型言語自身の表現形式

Ｍ形式／Ｓ形式，関数グラフ形式等

(ii) 関数型言語の持つべき機能

(2) 関数型マシンの機能

① 現在のマシンとの相違点

(a) 関数型言語を高速に処理する。

(b) リスト構造のデータに関する次の処理が速い。

— トリー・ウォーク（Tree walk）

— 要素選択（Component selection）

— パターン・マッチング

低レベルの「単なる文字列の一致」から，高レベルでは「プロダクションやHornクローズの処理」に至る，パターン・マッチングの処理の高速化が期待されている。

(c) 関数の呼び出し，特にリカーシブな関数呼び出しを高速に処理する。

— スタッキング機構

(d) 動的な記憶域管理を高速に実行する。

— 並行ガーベジ・コレクション機構

② 他のマシンとの関係

関数型マシンは関数型言語の利用者をサポートするでなく，ＰＲＯＬＯＧ等を処理する述語論理型マシン，抽象データ・タイプ等を取り入れた言語のための抽象データ型マシンの実行メカニズムとしての役割をはたす。

また，集合演算機能を得意とする関係代数マシンも関数型マシンで実現することが十分考えられる。

関数型マシンの実現方式としてデータフローマシンがあり，興味ある研究がおこなわれている。

(3) 関数型マシンの処理方式

以下に関数型マシンの処理をインタプリタの入力となる言語レベルの観点から分類する。

(i) 言語を直接インタプリト

(ii) 言語→Ｓ式；Ｓ式をインタプリト

(iii) 言語→（専用ＭＬ）；（専用ＭＬ）をインタプリト

この方式のＬＩＳＰマシンの研究が多い。

(注) ＭＬは機械語（Machine Languageの略である）。

(iv) (ii)と(iii)の両方式をサポート

（電電公社ＥＬＩＳにその例が見られる）

これらのインタプリタの高速化を目指すマシンの研究が進んでいる。

次に関数型マシンの実現方式を分類する。

Ⅰ. 専用ＭＬの形式

(1) 逐次実行型専用ＭＬ

—DeutchのＬＩＳＰマシンのコード

—ＭＩＴのＣＯＮＳプロセッサのコード

—ユタ大のＭＢＡＬＭコード（図２－１参照）等

(2) 関数型グラフ言語

ユタ大のＦＧＬ（関数型グラフ言語）

Ⅱ. インタプリタの制御方式

(1) マイクロプログラミング方式

(a) 商用のマイクロプログラム方式コンピュータを利用する方式。

ユタ大学のＭＢＡＬＭマシン（Ｂ１７２６）

(b) 専用のマイクロプログラム方式コンピュータを設計する方式。

—ＣＭＵのＬＩＳＰマシン

—ＭＩＴ，ＣＯＮＳプロセッサ

—ＭＩＴ，Sussman の１チップ・プロセッサ（図２－２参照）

—電総研，ＬＩＳＰマシンⅡ

—京大，ＮＫ３

—慶大，ＬＩＳＰマシン

—神戸大，ＬＩＳＰマシン

—電電公社，ＥＬＩＳ

(2) ワイヤード・ロジック方式

Ⅲ. インタプリタの制御の多重度

(1) 単一制御

現在までに研究・試作されているＬＩＳＰマシンは逐時実行型である。

(2) 多重制御

関数評価の並列実行

(a) データフロー方式

—東大のＬＩＳＰマシン

(b) リダクション・マシン方式

—Mago のリダクション・マシン

—Berklingのリダクション・マシン

(c) デマント・ドリブン方式

—ユタ大学のKeller マシン（図２－３参照）

現在のＯＳのプロセス管理方式の「プロセス」を「関数呼び出し」に置き換えたようなもの。

（プロセッサの個数は，例えば１０００個を考えている）

(注) Mago のリダクション・マシンは言語の文字列を見渡し，リダクション（式の評価）のできる部分の集合に対して，並行してリダクションを行う。この動作の繰り返しで「言語の実行」をする。一次元配列のセルと，トリー型配列のセルとで構成される。ＶＬＳＩ向きの構成である。（図２－４，５，６参照）

Ⅵ. シンボルテーブル・サーチ方式

シンボルテーブルのサーチの高速化のためには，順次にテーブルを検索する処理を高速化する方式やハッシング方式がある。

(1) テーブル・サーチ方式

連想メモリの利用等，ハードウェアの論理で高速化することが考えられる。

(2) ハッシング方式

ファームウェアやハードウェアでハッシング検索機構を実現し高速にする。

理化学研のＦＬＡＴＳマシンはハードウェアでハッシングを行っている。

V. データ形式と構造メモリ

関数型マシンの実現には，以下のようなデータ形式が利用される。

- タグ付データ
- ディスクリプタ

また，インタプリタの中では，処理上次のようなデータ形式が利用される。

- スタック
- ランタイム・シンボル・テーブル
- ヒープ

これらのデータ形を高速に扱う機構が必要となる。特にＶＬＳＩを使用した構造メモリ（連想メモリ等）が期待される。

VI. ガーベジ・コレクション方式

記憶域の動的な管理を行うガーベジ・コレクションの方式は，各種研究が進んでいる。

(4) 実現上の問題点と研究課題

関数型マシンの開発のための，主要な研究・開発項目を次に述べる。

① 関数型言語の仕様の確立

ＬＩＳＰ，ＩＮＴＥＲＬＩＳＰ等の関数型言語を参考にして，第５世代コンピュータ用の関数型言語の持つべき機能を検討し，言語仕様を定める。この際，言語の理解容易な表現（グラフ表現等）にも重点を置く。

② 並列処理記述用の関数型言語の仕様の確立

①で開発した，関数型言語を拡張して，論理的に並列処理可能な処理を，最大限並列処理をすることが可能なように，並列処理記述可能な関教型言語の仕様を定める。

③ ファームウァア・ベースのパーソナル関数型マシンの関発

マイクロ・プログラム方式の関数型言語のインタプリタを持つパーソナル・マシンを開発する。これは，開発の第Ｉ期に重点的に開発し第５世代コンピュータの研究促進用に利用する。

④ ＶＬＳＩ記号処理用ＵＨＭの開発

関数型マシン，述語論理マシン，抽象データ型マシン等のマシンに共用して利用出来る記号処理用のＵＨＭ（汎用ホストマシン）を設計し，これをＶＬＳＩ化する。このＶＬＳＩプロセッサは，第Ⅱ期における並列プロセッサ方式による高速関数型マシンの要素プロセッサ

となる。

⑤　並列処理方式の確立

関数型言語の処理を高速化のための，並列プロセッサによる並列処理方式を確立しデータフローの処理技術と結びつける。

⑥　ＶＬＳＩ並列型構造メモリの開発

一般的なパターン・マッチングの高速化の技術（多数のHornクローズを高速に処理する技術等）を確立し，ＶＬＳＩ並列型構造メモリを開発する。

⑦　プロセッサ間結合方式の確立

関数型マシンと他の第５世代のコンピュータのマシンとの情報の受け渡し（処理要求と処理結果の受け渡し）を高速に行うための，プロセッサ間通信方式を確立する。

(5)　実現の例

関数型マシンの研究は，ＬＩＳＰマシンを中心に国の内外で盛んになってきている。マイクロ・プログラム方式のＬＩＳＰマシンは，ＭＩＴでは実用になっている。

ただし，第５世代で実現しようとしているマシンは現在の１桁～２桁上の性能のものであり，今後の研究に負うところが大きい。

研究開発期間を約１０年と仮定し，その期間を大きく３期に分けると，次に示す開発目標が実現可能性の大きなものとしてあげられる。

第Ⅰ期：ファームウェア方式のパーソナル関数型マシンを開発する。このマシンは，第Ⅱ期以降，複数台生産して，第５世代の研究促進用として，他のマシンの研究・開発や，ソフトウェアの研究・開発に利用する。

性能は，４ＭＩＰＳの汎用コンピュータでのリスト処理速度の２～１０倍を目標とする。

第Ⅱ期：並列プロセッサ方式で高速関数型マシンを開発する。

性能は，４ＭＩＰＳの汎用コンピュータでのリスト処理速度の１０～１００倍を目標とする。

第Ⅲ期：データフロー技術とＶＬＳＩ並列型構造メモリを持った超高速関数型マシンであり，性能は４ＭＩＰＳの汎用コンピュータでのリスト処理速度の１００～１０００倍を目標とする。

(6)　研究開発計画

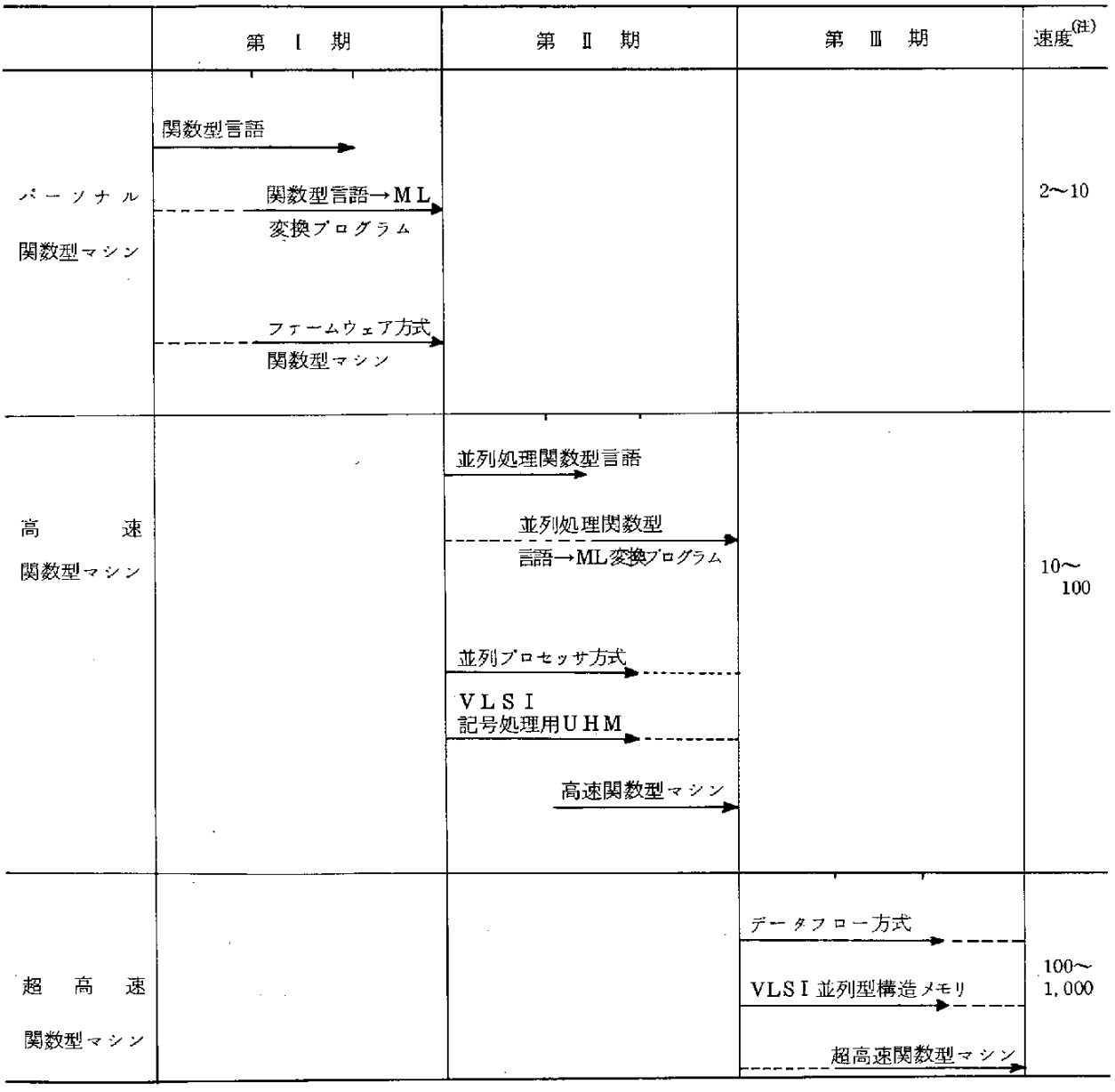
研究開発計画の一例を次に示す。

研究開発期間は約１０年として，その期間を大きく３期に分ける。第Ⅰ期では，研究促進用

のパーソナルな関数型マシンの研究・開発を主体とする。第Ⅱ期では，並列プロセッサ方式による高速関数型マシンを研究・開発する。第Ⅲ期では，データ・フロー方式とＶＬＳＩ並列型構造メモリを用いた超高速関数型マシンを研究・開発する。

但し，次に示す表に於て開発費はＶＬＳＩのＵＨＭの開発費を含んでいない。

また，ＶＬＳＩ－ＣＡＤシステムは，関数型マシンの開発とは別に開発されるものを使用することを前提としている。



(注)　４ＭＩＰＳの汎用コンピュータでのリスト処理速度に比較しての倍数

(7)　参考文献等

(1)　ＬＩＳＰマシンの開発の現状を探る

日経エレクトロニクス　1979.3.19

(2) 電子協　新技術動向専門委員会　報告書　１９８０.４

(3) "ＬＩＳＰ　Activites in Japan"
by kurokawa, Proc. IJCAI, 1979

(4) "A Note on Variable Bindings and Function Invocations"
by Kurokawa, 情報処理学会　記号処理研究会資料　１２－１３, １９８０.６.２０

(5) "A Network of Microprocessors to Execute Reduction Languages"
by G.A.Mago, International Journal of Computer and Information Science" Vol. 8, No. 5, 1979

(6) Proc.ＬＩＳＰ　Conference, Aug.１９８０

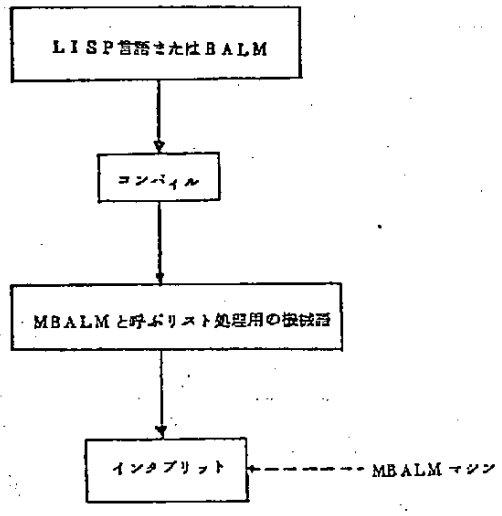


図２－１　ユタ大学のマイクロプログラム化ＬＩＳＰマシン

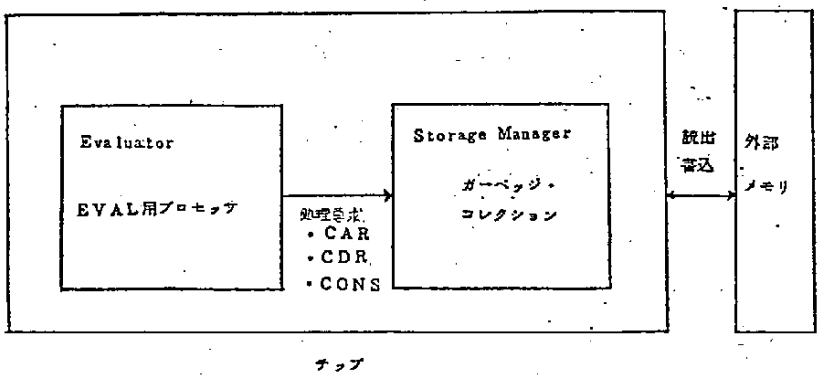


図２－２　ＭＩＴのＶＬＳＩ化ＬＩＳＰプロセッサ

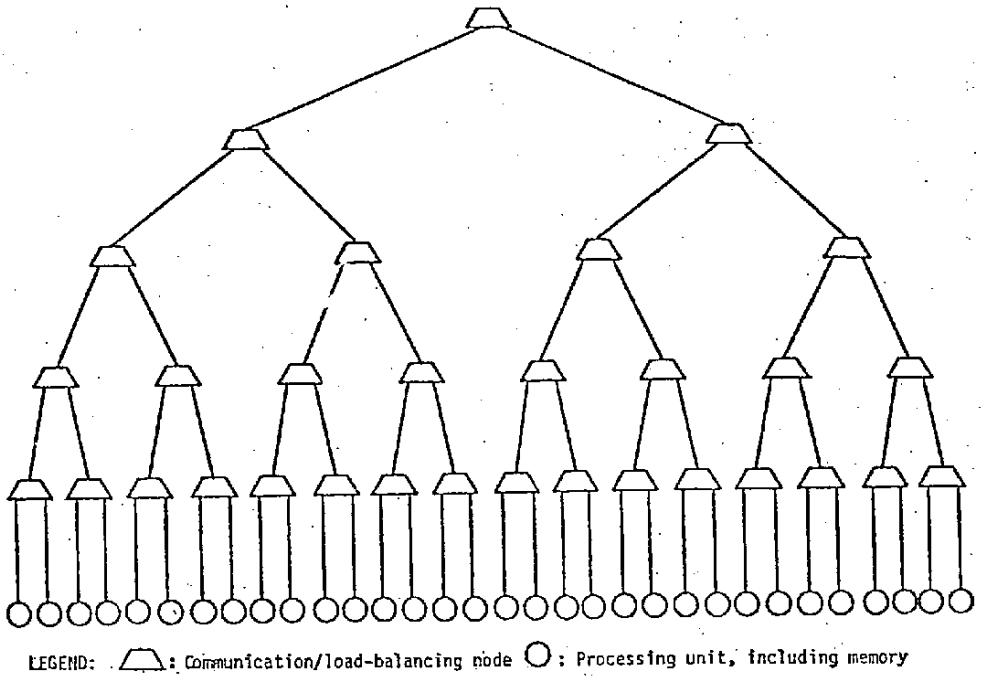


図２－３　ユタ大学のKellerマシン

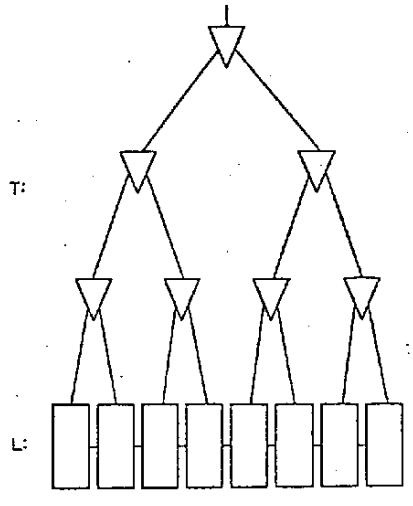


図２－４　Magoのリダクション・マシン

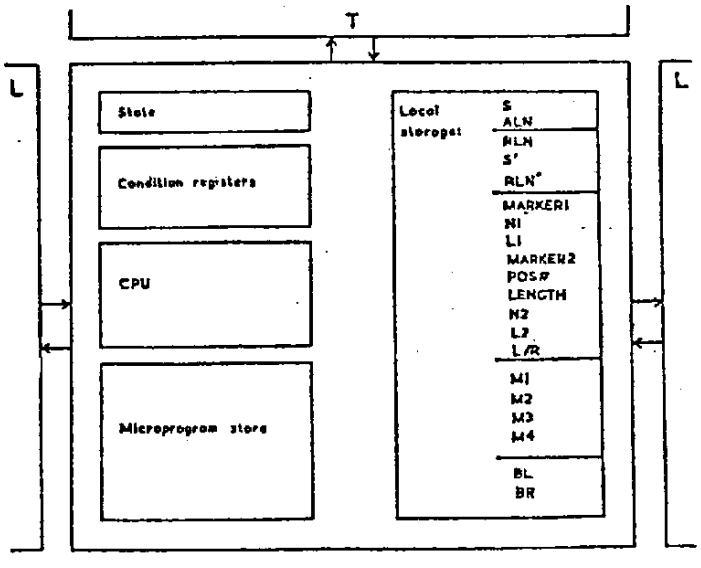


図２－５　Magoのリダクション・マシンのセルLの構造

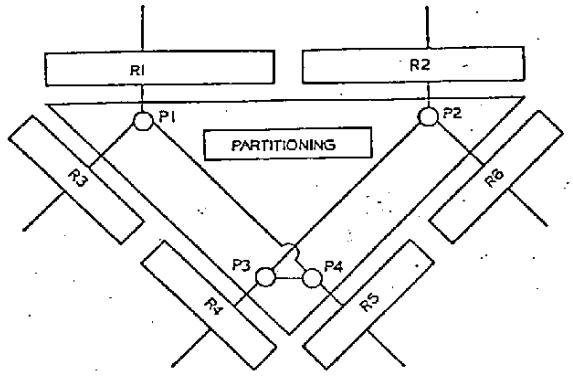


図２－６　Magoのリダクション・マシンのセルTの構造

2.2.3 抽象データ型マシン

(1) 抽象データ型マシン

① はじめに

抽象データ型マシンの発端は，ソフトウェアの言語サイド，もう少し広い見地からはソフトウェア工学上の要請からきており，抽象データ型に限れば，直接コンピュータ・アーキテクチャに反映するものではないが，その考え方の有効性が確認されるにつれ，アーキテクチャ内にもそういう概念を導入し易い機構を設けることが今後の情報処理技術の発展の重要なテーマになってきた。

抽象データ型は狭義に解釈すれば，通常のプログラム言語では固定化されているデータ型を，もっと自由に拡張して，原始的な要素から構成できるあらゆる型を定義可能なものにし，その型に対する処理内容（オペレーション）も同時に記述できるようにしたものであるが，広義に解釈すれば，抽象化の対象をデータ型だけでなくプログラムの制御構造やプログラムの意味構造にまで広げることが可能になる。本文では，この広い意味での抽象データ型を考える。

② 目　　的

抽象データ型マシンは，抽象データ型言語をアーキテクチャ面から支援し，効率よく実行する処理系をめざすものである。抽象データ型言語が目的とする所は，情報処理の基本要素を対象（Object）と呼び，対象は本来固定されたものでなく，処理分野毎に固有のものであり，応用分野が異なればおのずと対象の属性も異なるはずであるから，対象の範囲も最初から限定しないで，自由に定義できるものにしておくべきだとする考え方である。こうすることにより，応用分野毎に最適のシステムが仮想的に実現可能になる。

しかし，これを実際に計算機で実行すると処理能力やメモリ容量等の不足のため，期待した性能が得られない事が多い。抽象データ型マシンは，対象毎に最適な系をとりうる可変構造型アーキテクチャや並列処理の方式を研究することにより，現在の計算機の性能限界の壁に挑戦し，情報処理の理想に近づけることを目標とするものである。

③ 効　．　果

抽象データ型マシンは，ソフトウェア工学に起源をもっていることからも，ソフトウェア全体に与えるインパクトは非常に大きいものである。理論的には，プログラム正当性の検証で抽象データ型による形式的アプローチが有力な道具となっている。実際のシステムでもたとえばデータベースにおいて，データ型の変更は莫大な工数を伴うが，こういう傾向はシステム規模が大きくなる程，重要な問題であり，システム機能の拡張，変更を容易に実現する抽象データ型マシンはシステム開発に大きな恩恵となろう。プログラムの面からみると抽象データ型はモジュール化，カプセル化を指向したもので，高信頼なシステムであると同時に

そのカプセル内での最適化が可能になり，ソフトウェアのハードウェア化／ファームウェア化にもつながっている。またプログラムが階層化され，インタフェースが明確に定義されているため，プログラムの明解性が得られ，マシン従属部も限定されているので，マシンからの独立，プログラムのポータビリティにも寄与する。ユーザからみると，固有の対象が扱かえるため，問題向きプログラムに共通する使い易さ，単純性が得られる。

以上多くのソフトウェア上の効果は，ＯＳ作成者からエンドユーザに至る巾広い層にまたがっており，高性能な抽象データ型マシンの開発が望まれる。

(2) 抽象データ型マシンの機能

① データ型の抽象化

情報処理では，処理対象を計算機で実行できるようにモデル化する作業が重要であるが，この作業においてデータ構造を決定するさい，問題となる対象をよく把握して最適なものを選択しなければならない。例えば，スタック，テーブル，リスト，キュー等はよく知られたデータ構造であるが，より具体的には，部品，人間，組織等も定義することによって，処理が直観的に記述できることになる。抽象化されたデータは，階層化を許し，基本単位（例えばビット）で構成でき，ＳＩＭＵＬＡ６７のＣＬＡＳＳのように，実際のプログラム環境ではパラメータによってそれぞれ独立のデータとしてふるまえることが必要である。

② 制御構造の抽象化

今日のプログラム言語においてはＩＦ／ＴＨＥＮ／ＥＬＳＥによる条件判定，ＰＯＳＴ／ＷＡＩＴによる待合せ，ＦＯＲＫ／ＪＯＩＮによる並列処理の同期化等の制御構造の必要性はよく認識されているが，これ以外にも今後重要性をもつものとして例外／割込み処理の機能，記号処理におけるバック・トラッキング，コルーチン機能，並列処理におけるプロセス間通信や共有リソース管理に起因する多様な制御構造等を実現できるものが必要である。

③ 関数の抽象化

関数の抽象化とは，タスクとプロセスとか呼ばれる単位から，マシン命令に至るあらゆるレベルの処理機能を定義可能にし，かつそれらの関数を合成，分解する関数形式とよばれるものを実現することである。前者の例としては可変命令系がある。これはユーザが独自に命令体系を作成できるものであり，専用システムに有効である。後者は，サブルーチンコールが卑近な例であるが，知識情報処理で重要な役割をもつ関数の自動合成には有力な道具となろう。

④ 対象指向型

従来プログラムは，データとコードとを明確に分離する考えが主流であり，この思想はＯＳやアーキテクチャにまで影響している。対象指向型とはもう少し高い立場から両者を対等に扱うもので，普遍的な対象という概念（従って対象とはデータだけでなく関数も含んだも

のである）に基づくものである。対象概念が存在する対象空間は仮想空間を拡張したものとみなせ，リソース管理やメモリ管理において重要なcapabilityアドレッシング機能をアーキテクチャ上統一した形で実現するものである。[5],[12]

(3) 抽象データ型マシンの処理方式

抽象データ型マシンの処理方式の模式図を図２－７に示す。抽象データ型マシンは，言語によってマシンの構造等を定義して行くものであるから，ソフトウェア・マシンとみなすことができる。

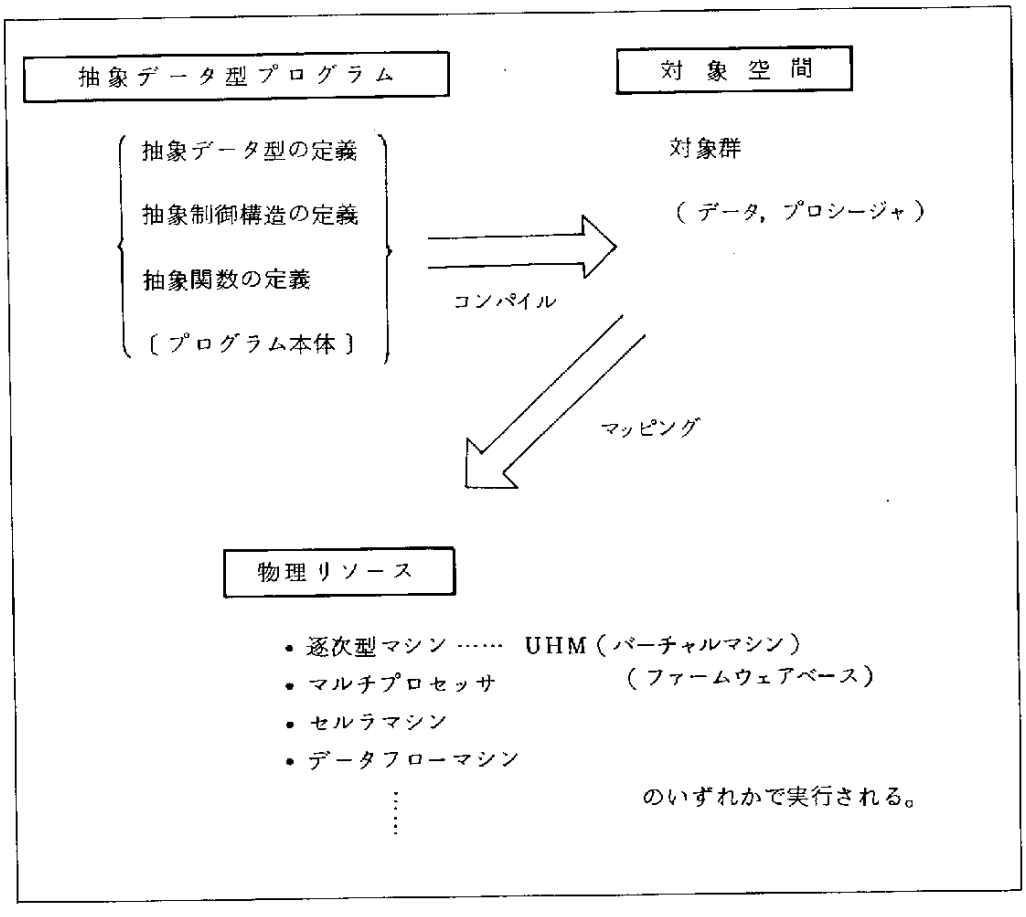


図２－７　抽家データ型マシンの処理フロー

まず抽象データ型をサポートするプログラム言語により，目的とするシステムに最適なようにデータ型，制御構造，関数を定義する。この作業は通常トップダウン・アプローチがとられ，最終レベルでは，マイクロプログラムまで記述可能なことが望ましい。もちろんこれらの作業の各段階では，全対象を最初から定義する必要はなく，データベースに登録されている定義済み対象を有効に利用する。定義が済めば，それらを用いてプログラムを作成する。

作成したプログラムをコンパイルすると，それらは論理的な対象空間に変換される。抽象データ型マシンの特徴は，実行時に初めて具体的な系が構成されるので（ダイナミック・バインディング），コンパイル時には，物理イメージを含んでいないのみならず，論理イメージも原型（あ

るいは虚像）にすぎない。起動がかゝって初めて対象空間にその姿（実像）が出現する。対象空間とは仮想空間を拡張したもので，各対象がユニークに場所を確保してアドレス可能な空間である。

プログラムは論理的には対象空間上で実行されるが，実際には有限のメモリやプロセッサを使用するから実行中は対象空間から物理リソースへのマッピングが必要となる。物理的な処理装置の形態としては，現在の逐次処理方式でも可能であるが，データフローマシンやセルラープロセッサによる並列処理によって実行することも可能であろう。

(4) 実現上の問題点と研究課題

抽象データ型を効率よく実行する上で言語，アーキテクチャから研究開発すべき課題を以下に述べる。

① 抽象データ型プログラミング言語

抽象データ型のプログラミング言語として現在ＳＩＭＵＬＡ６７（シミュレーション言語），ＭＥＳＡおよびＳＭＡＬＬ　ＴＡＬＫ（ゼロックス），ＣＬＵ（ＭＩＴ），ＡＬＰＨＡＲＤ（プログラム検証用言語），ＡＤＡ（米国防省の標準言語）等があるが，狭義の抽象データ型サポートが中心であり，広義の抽象データ型で対象指向の言語を開発する必要がある。そのための課題として，

a. ユーザ用高水準言語以外に，ファームウェア用の高水準言語や可変構造型マシンを制御するマシン構造記述用高水準言語が必要である。

b. 広義の対象を記述（定義）できるために普遍的なモデル（対象空間）を有すること。

c. ＯＳまで含めてシステム全体が明確なインタフェースをもっていること。

d. 並列処理の記述が可能で，各種の制御構造を実現できるもの。

等が重要である。

② 構造メモリ

対象指向のシステムでは，対象が多様であるのと，生成／消滅が発生するため，単純な１次元配列のメモリでは，処理能力上問題が多く，構造メモリが必要になる。その機能として，

a. 論理的には無限に等しい対象空間を含む大容量性と高速アクセスをもつこと，従ってメモリの階層化と多重アクセス機構が必要であろう。

b. 並列ガーベッジ・コレクションが可能である。

c. 対象を一元的に管理できるアドレッシング方式であり，その高速な変換／検索に連想メモリを用いる。

d. 対象を識別し，保護する機構として，タグ，ディスクリプタを有する。

e. スタック機構やヒープ機構をサポートするもの。

等がある。

③　プロセッサ

抽象データ型言語を処理するには，プロセッサも言語と同じ位柔軟性が要求され，新しい言語に適したものが必要である。研究課題として，

a．ダイナミック・アーキテクチャの手法

ダイナミック・マイクロ・プログラミングだけでなく，レジスタやバス幅の動的変更や系の再構成が可能なＵＨＭ（Universal Host Machine）の開発

b．コンピュータ・ネットワークに適した通信機能

c．高水準言語の高速処理

タスク切換え，プロテクション等のＯＳ機能を含む。

がある。

④　並列処理方式

抽象データ型言語を並列処理で実行することは，対象が本来逐次的に動作するものでないことから自然な方法であるが，それには，

a．並列環境下でのモデルを設定する。

並列処理では各対象が分散して処理されるため，対象間の通信が重要になるが，ACTOR理論[7]はデータや関数を　含したものであり，広義の対象に対応しており，対象が相互に通信することによって並列動作するモデルであるため，この理論を基礎にして抽象データ型言語で扱う対象空間を研究することが有望である。

b．並列処理装置の構成

ダイナミック・アーキテクチャとしてUHMの関発を述べたが，並列処理にあっても各処理装置はＵＨＭが必要となる。従ってUHMの多重構成が必要となるが，これ以外に構造メモリをどう実現するかが，重要な課題である。また多重処理装置に共通の課題として，通信ネットワークの問題がある。

c．並列処理装置への写像

ａでの並列モデルをｂの上で動作させる場合，各対象をどの装置に配置するのか，モデル上のアドレッシングが効率よく管理できるか，それは一元的かそれとも階層化されているか，等の問題が派生し，これを解決することが並列処理実現の鍵となる。

d．分散処理下での実現

抽象データ型を分散環境下で実現することも重要な課題である。これには並列モデルの拡張が必要であるが，対象指向のもつ柔軟性，拡張性により，十分実現できるものと思われる。

等を開発しなければならない。

⑤　従来言語との互換性

抽象データ型言語が在来型の言語と親和性があるかどうかは重要な問題であり，従来のシステムとうまく融合する必要がある。これには，対象のもつキャプセル化をマシンレベルにまで適用して，従来型システムを完全に閉じ込める技術を開発しなければならない。

⑥ そ の 他

抽象データ型マシンを，通常の応用に使用するには，入出力系，データベース等とのインタフェースの整合性が必要である。対象空間は本来入出力機能も包含しているから，実際のシステムでは，ＯＳの入出力制御部を介して行い，物理的には，メモリマップド入出力的に考えることで実現できよう。言語面から入出力をメモリへのアクセスと区別するかどうかは，一長一短で今後の研究課題となろう。

データベースに関しては，抽象データ型がどの程度メタ・モデルとしての記述能力をもつか，大規模データベースの対象空間への写像でローカリティの概念がマシンで実現できるか，分散データベースでも機能し得るか，等の研究課題がある。

(5) 実現の例

抽象データ型マシンの一例として，並列ＡＣＴＯＲに基づいたシミュレーション・システムを考えてみよう。並列ＡＣＴＯＲの概念図を図２－８に示す。対象空間は，データ，関数を包

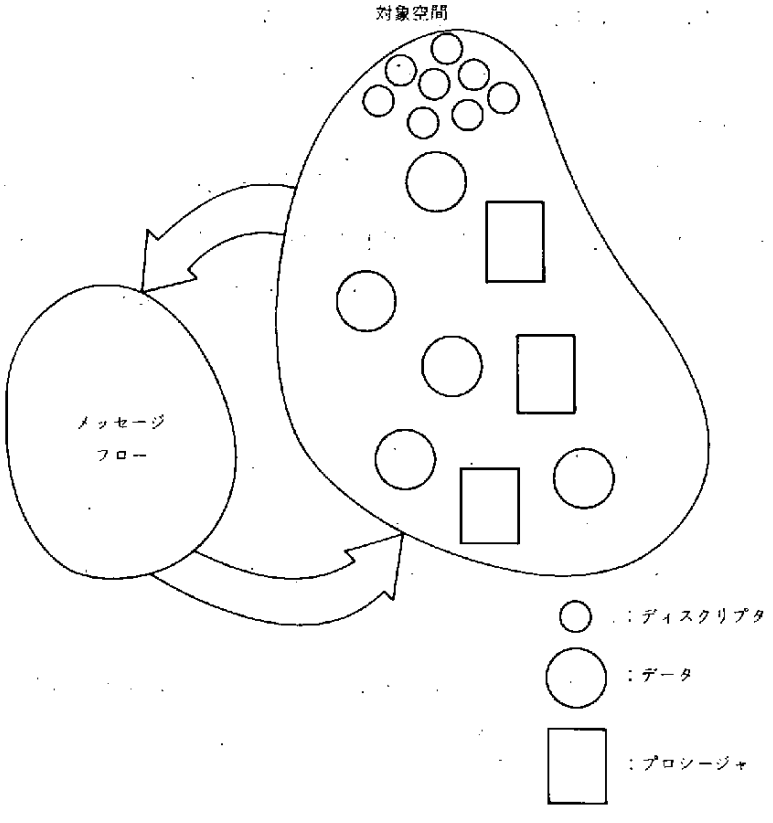
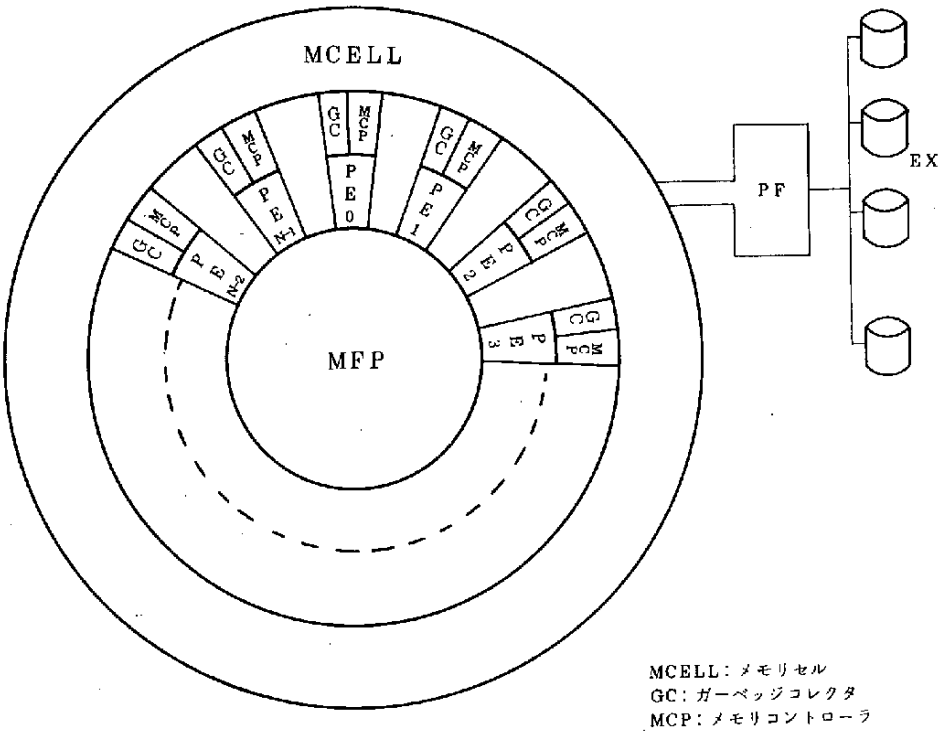


図２－８　並列ＡＣＴＯＲの概念図

含した対象からなり，それぞれ固有の識別名を与えられているものとする。識別名は，共通の祖先の名前と可変名からなっており，対象を表現するディスクリプタ内に置く。事象が発生することは，新たな対象を生成することであり，そのつど対象空間に領域を確保し，ディスクリプタで指示する。

シミュレーション・システムは，生成した対象が次々と関数に引渡されて，変化して行くものであり，対象空間では，対象が常時生成／消滅を繰返し，関数間にはメッセージが流れているものとみなせる。こういう一般的モデルの実現を考えてみる。

図２－９は並列抽象データ型マシンの概念図である。ＰＥｓは処理装置で，各関数は適当なＰＥで実行される。ＭＣＥＬＬは構造メモリであり，ＭＣＴＬによってアクセスされる。メモリへのアクセスはディスクリプタで行う。ＭＦＰはＰＥからＰＥへのメッセージ転送を制御する装置である。ＰＦはページング・ファイルで，ＭＣＥＬＬの２次記憶に相当する。ＥＦは外部ファイルで論理的に対象空間の外部に位置して，必要な時にＰＦにロードする。ＧＣはガーベッジ・コレクタで並列ガーベッジ・コレクションを行う。ＧＣのよしあしはシミュレーション・システムの性能を左右するので専用化する。以上の構成でＰＥの個数は１００個のオーダを考える。ＰＥはローカル・メモリをもったＵＨＭを使用する。ＵＨＭのダイナミック性を利用して大部分の抽象データ型の記述はＵＨＭで最適化される。データ幅の拡張等に対しては隣



MCELL：メモリセル
GC：ガーベッジコレクタ
MCP：メモリコントローラ
PEi：処理エレメント
MFP：メッセージフロープロセッサ
PF：ページングファイル
EX：外部ファイル

図２－９　並列抽象データ型マシンの実現例

接ＰＥと結合するものとする。シミュレーションにおいて，関数がうまくＰＥｓに配置でき，ＭＦＰがＰＥｓを１：１に結合できるとすると，いちど生成した対象はパイプライン的に複数ＰＥで処理されて行く。このような並列処理で最も困難なことは，論理的な対象をどこで実行するかであるが，対象自身の識別方法が確立していれば，可変構造の利点により，ＵＨＭを適当に対応づけることは可能であろう。

性能面からみるとＵＨＭ１台に比較して２０～３０倍の処理能力を達成できることから，かなり高性能なシミュレーション・システムといえる。価格的にみても，基本要素として，UHM，ＭＣＴＬ，ＧＣ，ＭＦＰ等があるがこれらはいずれもＶＬＳＩ技術で実現できれば，リピータビリティによって解決するものと思われる。

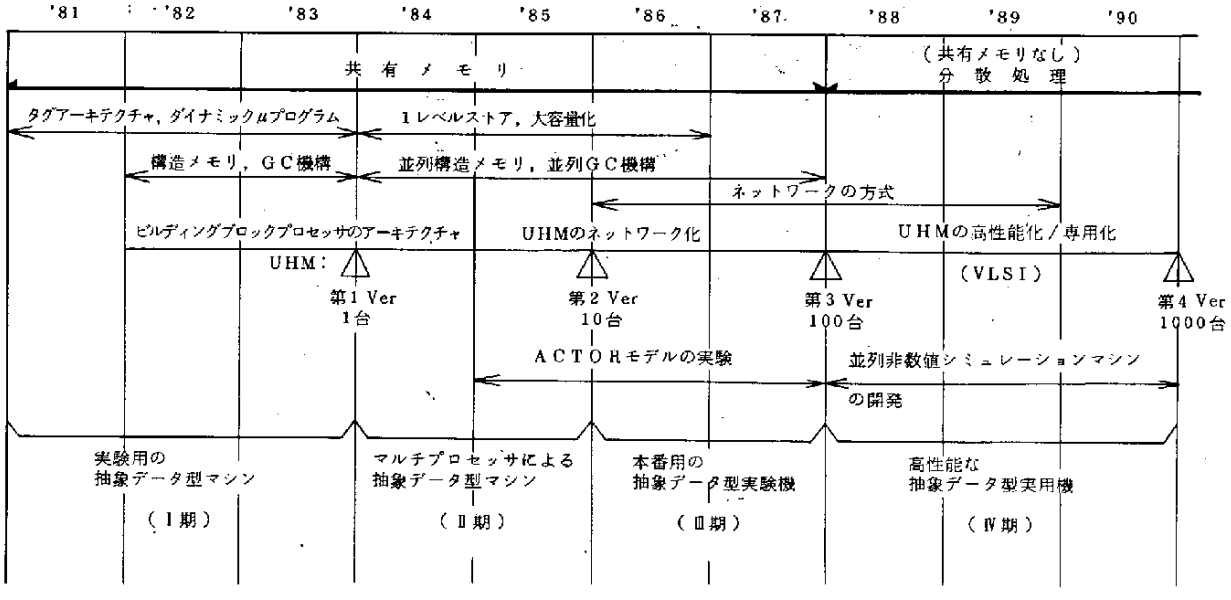
以上に示したように，処理装置と対象が全く自由に結合できるアーキテクチャであるので，自然言語，データベース，グラフィクス等にも抽象データ型を適用することにより，じゅうぶん実用的な処理装置の実現が可能であろう。

(6)　研究開発計画

表２－１に抽象データ型マシンの研究開発計画を示す。開発を４段階に分けて，各段階でＵＨＭを開発する。

第Ｉ期は逐次型による実験用抽象データ型マシンの開発で，タグ・アーキテクチャ，ダイナミック・マイクロプログラム機能をもったＵＨＭを作る。構造メモリ，ＧＣ機構も研究開発する。

表２－１　抽象データ型マシンの研究開発計画



第Ⅱ期は第Ⅰ期でのＵＨＭを改良して１０台程度のマルチ・プロセッサにする。メモリを大容量化すると共に，並列構造メモリ，並列ＧＣ機構の開発を始める。

第Ⅲ期は１００台程度の並列処理をめざすため，大規模並列用の方式を研究する。計算モデルとして，ＡＣＴＯＲ理論の実験を行う，ＵＨＭのアーキテクチャもそれに従って根本的に変革することもあり得る。処理装置を結合するネットワークの開発も始める。

第Ⅳ期は第Ⅲ期のＵＨＭをＶＬＳＩ技術によって高度化する。処理装置は１０００台以上として，実用的な並列処理抽象データ型マシンを実現する。１つの例として並列非数値シミュレーション・システムを開発する。

(7) 参考文献

(1) 第５世代電子計算機に関する調査研究報告書
マシンの理論 ；昭和５３年３月，財団法人日本情報処理開発協会

(2) Dahl,Nygaard：SIMULA－an ALGOL－Based Simulation Language, CACM, Vol 9,No. 9,Sept.,1966

(3) B.Liskov,S.Zilles：Programming with Abstract Data Types, SIGPLAN Notices, Vol. 9,No.4, April, 1974

(4) Alan Snyder：A Machine Architecture to Support an Object-Oriented Language,MIT/LCS/TR－209,March,1979

(5) G.Myers：Advances in Computer Architecture,John Wiley & SONS, Inc., 1978

(6) J.C.Browne：A PROPOSAL FOR PRELIMINARY DESIGN OF OBJECT ORIENTED COMPUTER SYSTEMS，第５世代調査資料，June,1980

(7) 米沢明憲：ＡＣＴＯＲ理論について，情報処理，Vol.20,No.7,July1979

(8) J.K.Iliffe：ARCHITECTUAL SUPPORT FOR ABSTRACTION

(9) B.Hansen：A KEYNOTE ADDRESS ON CONCURRENT PROGRAMMING, Proceedings Compsac 78, IEEE

(10) K.M.Kahn：Intermission — Actors in Prolog,Logic Programming Workshop,July,1980

(11) J.Ratter,G.Cox：Object-Based Computer Architecture,ACM Computer Architecture News,Vol.8,No.6,Oct,1980

2.2.4 関係代数マシン

(1) 関係代数マシンの目的と効果

知識情報処理の特徴は，いろいろな応用分野における基本的な規則や作業手順など，知識と呼ばれるものを蓄え，これを適宜利用して，計算機に知的な仕事をさせようとすることである。この場合の知識と呼ばれるようなデータやプログラムを蓄えるのが，関係台数マシンの役割である。

関係代数マシンの目的は，関係モデルに基く集合（Set）演算を高速に，能率よく実行することである。プログラマからみた場合は，関係代数，もしくは，関係論理で記述された集合演算を主とする操作を行うマシンとなり，関係データベース処理に用いられる。このため専用化されたハードウェアは，関係データベース・マシンとして，研究されている。

関係代数マシンは，そのインタフェースとなる基本操作が，明確な数学的基盤に基いて決められていることから，知識情報処理システムのような複雑なシステムを構築する上での見通しがよく，種々の推論などのより高レベル処理を行う述語論理型マシンなどの整合性も優れている。

関係代数マシンを，目的により高速化したり，大容量化することにより，自然言語処理，機械翻訳，コンサルテーション・システム，CADシステムなどの知識ベースの高能率化がはかれ，応用システムの性能が，量的にも，質的にも向上し，適用範囲を拡大させることができる。

(2) 関係代数マシンの機能，性能

マシンの機能は，そのインタフェースとなる言語の仕様により変わるが，ハードウェアとして実現する機能は，ほぼ似たものとなる。その性能は，マシンの応用される分野の性質，使用条件に対し，どの程度，専用化するかにより変わる。

関係代数マシンのハードウェアは，関係データベース・マシンとして，研究がすすめられており，これらも含めて，関係代数マシンの機能，性能について述べる。

a) マシンの言語インタフェース

言語インタフェースとしては，関係論理に基くものと，関係代数に基くものが研究され，いくつかの言語が提案されている。

① 関係論理（relational culculus）のレベル

一階述語論理に基く言語で，そのまま，高レベルのデータベース操作言語となる。このレベルは，マシンのハードウェアからみると，直接実現するには高すぎると思われ，よりハードウェアの実行形態に近いレベルに変換する必要がある。

この変換の中には，概念変換や，アクセスの最適化なども含めることが，実際上は必要となり，これらは，データベース管理に関する重要な研究課題となっている。

このレベルで提案されている言語としては，ALPHA，QBE，SEQUEL（SQLともいう）などがある。

② 関係代数（ relational algebra ）のレベル

関係台数は，関係論理に比べ，演算操作をより実行に促して指定することから，ハードウェアに近いものとなる。言語としては，集合演算式となり，次のものが基本となる。以下のもののほかにも，大小比較，条件判定など，主に制御構造を記述する演算処理がある。

i）和（ union ），積（ intersection ），差（ difference ）

ii）射影（ projection ）

iii）結合（ join ）

iv）商（ division ）

v）制約（ restriction ）又は，選択（ selection ）

これらのものが，マシン（ハードウェア）を構築する上での中心的機能であるが，これらのほかの細かな制御用の命令などの実現法との関連から，プログラミング言語として，いろいろな形式が考えられる。

b） マシンとしての機能と性能

関係代数マシンとしては，②の i）～ v）の演算を高速に実行する機構を備えることになるが，これらをアーキテクチャ的立場からみると，次の３つの基本的機能に集約できる。

i)′ 探索（ search ） — 与えられたKeyをもつレコードをさがしだす。

ii)′ 整列（ sort ） — レコードをKeyの順序に従って並べ変える。

iii)′ 結合（ join ） — ２つの集合中のレコードを共通のKey を持つものについて抜き出す。

これらの基本的機能を実現するにあたっては，マシンの速度や容量が，アーキテクチャに大きく影響する。一般的に，高速化の条件としては，処理が並列化できることが必要で，尚かつ，多くのハードウェアを使用することが必要である。

速度については，探索のアルゴリズムを基準として次のような段階づけが可能である。処理時間を，レコード数をｎとし，その関数として示す。[文献1)]（比例定数は省略）

ⓐ $T_1 = 1$（定数） → 連想処理

ⓑ $T_2 = \log_b n$ → 木構造探索

ⓒ $T_3 = \sqrt{n}$

ⓓ $T_4 = n$ → 逐次探索

ⓔ $T_5 = n \log_b n$

容量の段階づけは，困難であるが，一つの目安として，次のものが考えられる。１ＫＢ／項目程度と考える。

i） 小容量 — １Ｋ～１００Ｋ項目（１～１００Ｍバイト）

（高速） 連想メモリや連想プロセッサで実現できる範囲で，述語論理マシンの

メモリ・サポート等も用途に含む。

ii) 中容量 ― 10K～10M項目（10M～10Gバイト）
（高速～中速） 二次記憶を用いて仮想メモリが必要な範囲。一般データベースのサポート。

iii) 大容量 ― 1M～1000M項目（1G～1000Gバイト）
（中～低速） 大容量だが，アクセス速度は，それ程，速くない。何レベルかの記憶階層を持つ。最も低レベルの部分は，Mass Storage System(MSS)のようなデバイスか，又は，用途により，光ディスクのような新デバイスが使われる。

以上のような，探索時間と容量により，ハードウェア的には，いくつかの段階づけが行える。実際に，データベース・マシンとして使う場合には，レコードのディレクトリの管理とレコード本体の管理は，別の関係代数マシンを組合せて使用する等の方法*がとられると思われ，そのソフトウェアやデータの意味，概念変換など，より理論的な研究が必要である。

このような意味からも，関係代数マシンは，よりプリミティブなマシンとして位置づけられる。

*ディレクトリ管理には，高速・小容量のマシンを用いデータ本体の管理には，中速，大容量のものを用用いる等の，システム構成がとられると思われる。

(3) 関係代数マシンの処理方式（動作原理）

関係代数マシンは，関係データモデルに従い，数学の集合（set）と，関係（relation）の概念を用いて記述されたデータ構造を扱う。データは，共通の意味づけをもつものごとに集合を構成し，集合の要素間の対応した組をタプル（tuple）と呼ぶ。一般には，下のような表で示すとわかりやすい。(2)

学生

学籍番号	名　前	住　所
43001	ARISAWA	TOKYO
43002	ITO	YOKOHAMA
43003	UENO	TOKYO

キー：学籍番号

科目

科目名	必修選択
SUUGAKU	H
BUTSURI	S
EIGO	H

キー：科目名

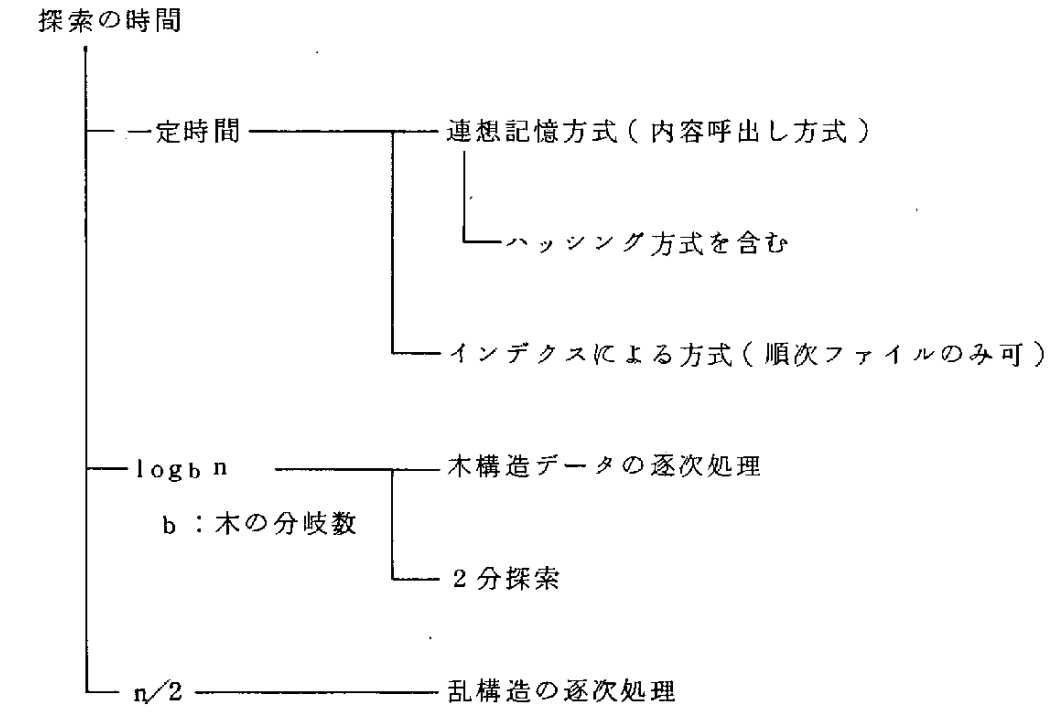
図２－１０　関係の表の例〔文献(2)による〕

このような表（関係ずけられた集合）のいくつかについて，②のi)～v)の関係代数演算を施すわけであるが，マシンとしては，探索，整列，結合の３つを基本的機能として実現する。

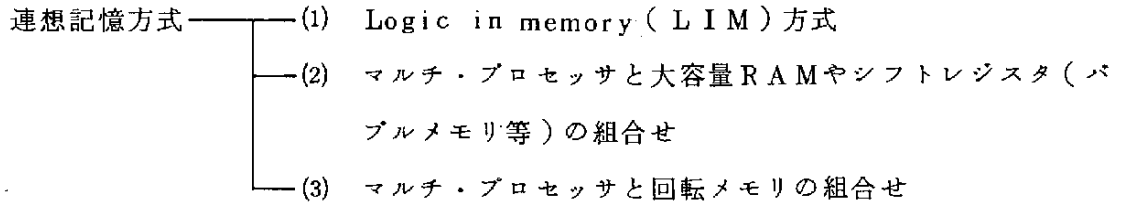
これらの実現方式については，最近，研究が開始され，その処理アルゴリズムやアーキテクチャが提案されている。以下，これらを整理して示す。

a) 探索の実現方式（アルゴリズム）

探索の高速化は，いろいろな並列処理の導入により達成される。これらを，そのアルゴリズムに基く，速度により分類すると以下のようになる。(1)



これらのうち，連想記憶によるものが，高速でありＶＬＳＩ化などとの整合性から，最も注目すべきものである。この方式については，次のものが研究されている。



これらのうち，(3)は，ディスク等の機械的な回転メモリを使用しており，これは，将来は，バブル・メモリ等の電子的なものに置きかわると考えられる。従って，小容量のものは，(1)，中容量のものは，(2)が有力である。しかし，いずれの場合も，実用規模のものを作るためには，ＶＬＳＩ化が不可欠である。

b) 整列と結合の実現方式

結合（join）は，整列（sort）を行ったデータの集合間で行う場合が多い。このため整列について述べる。

アルゴリズムの面から検討すると，一般に，逐次的に，整列を行うと，大小比較回数は，$\log_2 n! \approx n \log_2 n$ （但し，nはデータ数）回となる。これを並列処理により高速化することを考えると，次のようになる。

まず，n個のデータを，k個ずつのブロックに分けm個のブロックを作る。（$n = k \cdot m$）ブロック内は逐次処理を行うものとすれば，その理論的最小比較回数は，次式で与えられる。(1)

$$\frac{k\,m \log_2 (k\,m)}{m} = \underbrace{k \log_2 k}_{\text{ブロック内の逐次処理の回数}} + \underbrace{k \log_2 m}_{\text{ブロック単位の整列のための操作回数}}$$

しかし，一般には，この最小回数を実現するアルゴリズムは無く，実際は，これより多くの回数が必要で，その増加率をα，βとするとき，

$\alpha \cdot k \log_2 (\alpha k) + \beta k \log_2 m$ となる。

α，βの値は，各ブロックの処理を行う結合網の構成に依存し，高速化するほど，ハードウェア量は多く必要である。これらについては，表２－２のようにまとめられる。この表では，プロセッサの結合網において，あるプロセッサから，別のプロセッサへ，データを送ろうとする時，何個のプロセッサを経由するか（遅れ），結合線と接続装置の数（価格），ブロック数（≈プロセッサ数）が増えたとき，結合点がどのように増えるか（性能価格比）を示している。

たとえば，木構造に基く結合方式では，あるブロックから他のブロックへデータを転送するのに，平均$\log_2 n$のオーダの回数が必要である。価格は，ブロック数の増加に比例するnのオーダにとどまる。

遅れのオーダを$\log_2 n$のオーダにとどめれば，実際的には，高性能のマシンと考えられる。これらのどの処理方式を選ぶかは，データ数や処理速度，価格などの点を考慮して決定されることとなる。文献1)

表２－２　結合方式の分類

性能価格比	遅れ	価格	対応するアーキテクチャ（括弧内は図4.7の対応するアーキテクチャの番号）
$\overline{O}(1/\log_2 n)$	$\overline{O}(\log_2 n)$	$\overline{O}(n)$	木構造に基づく次数4（または3）の正規結合(a)，最適結合(b)
$\overline{O}(1/\log_2 n)$	$\overline{O}(\log_2 n)$	$\overline{O}(n \log_2 n)$	立方体結合(c)，変形立方体結合(d)
$\overline{O}(1/\sqrt{n})$	$\overline{O}(\sqrt{n})$	$\overline{O}(n)$	格子結合(e)
$\overline{O}(1/n)$	$\overline{O}(n)$	$\overline{O}(n)$	共通バス結合(f)，スイッチ付共通バス結合(g)，ループ結合(h)，スイッチ付ループ結合(i)，スター結合(j)，共通記憶結合(k)
	$\overline{O}(1)$	$\overline{O}(n^2)$	完全結合(l)，クロスバースイッチによるマルチプロセッサ結合(m)，多重バス結合(n)

〔文献１による。〕

プロセッサの個々のものの構成としては，次のようなものがある。これらは，プロセッサ内部についての処理であるから，小量のデータを高速に整列することが目的である。しかし，上に述べたプロセッサ間の結合方式と本質的には，同じである。

プロセッサ内部の処理方式（アルゴリズム）

- Batcher network …… $\log_2 n$ のオーダーで整列ができる。
  （立体結合，クロスバースイッチ，最適結合で実現可）
- 前川のアルゴリズム …… $\log_2 n$ のオーダー，結合（join）も扱う。
  （立方体結合）
- Thompson & Kungのアルゴリズム（ $\sqrt{n}$ のオーダーで整列）

そのほか，結合（join）に関しては，ハッシングを用いる方法もある。

以上，関係データベース・マシンとして研究されているものの中から，関係代数マシンの基本機能についての処理方式と動作原理をとり出して，述べた。実際のマシンの実現は，これらアルゴリズムを組合せて，応用目的に最適化したハードウェア構成をとらねばならない。

(4) 実現上の問題点と研究テーマ

a) 研究の現状と問題点

関係代数マシンの研究は，関係データベース・マシンとして研究され，ソフトウェアによって，既存汎用機上に試作されたり，ハードウェアとして実験機が試作されたりしている。また，マシンの構成についても，いくつかの提案がなされている。現状においては，関係代数マシンを，集合演算を得意とする関数型マシンととらえ，そのアーキテクチャを研究する方向と，データベース・マシンとして，そのプログラミング言語，データの意味論，アクセスに対する概念の変換，データのインテグリティの保障などの問題を対象とするソフトウェアサイドの研究の方向の２つに分けて考えられる。

前者の研究における問題は，(2)のマシンの機能の項で述べたような容量や処理速度に関し，十分な能力を有するハードウェアが妥当な価格性能比で，提供できるかということである。関係代数マシンは，集合演算に関し，強力な命令〔(2)の②のⅰ)～Ⅴ)〕を持ち，データベース・マシンとしての汎用性も有している。これらの有効な実現には，連想記憶方式等，従来のノイマン型計算機とは異る新しいアーキテクチャのサポートが不可欠である。

ソフトウェアにより，既存のノイマン型汎用機上に実装する場合，その処理速度は，きわめて遅いものとなり，実用の範囲は著しく制限される。このような理由から，実際には，航空機の座席予約システムのデータベースのような，限定された定形型のデータモデル専用のシステムが作られている。

知識情報処理で扱われるデータは，このような座席予約システムのような定形型のものと

異り，より一般的な，複雑なものとなる。これらを扱えるような汎用性を有するものとしては，現在のところ，関係モデルに基く関係代数マシンが有力な候補であり，特に，推論などをサポートする述語論理マシンなどとの機能的整合性を有する等の，複雑なシステムの構築に適した性質をもっている。

これまでに，試作されたシステムとしては，ソフトウェアによって実装された実験データベース・システムSystem Rがあり，商用化を目ざした研究がすすめられている。[文献(5)] 従来のノイマン型計算機から離れ，専用のアーキテクチャを求める研究としては，トロント大学のＲＡＰに代表される回転記憶の複数のヘッドに論理機能を付加し，内容呼出しを並列に行うもの，ウィスコシン大学のＤＩＲＥＣＴのように，クロスバー・スイッチにより，プロセッサやディスクを結合した多重マイクロ・プロセッサ・システム，電子技術総合研究所のＥＤＣのような磁気バブルとマイクロ・プロセッサを単一バスで結合した多重プロセッサ・システムなどがある。[文献(3)(1)]

いずれも，実験機であり，実用化に向けては，ＶＬＳＩ化などの実装手法，プロセッサ数を増やし大規模化する場合のシステム・アーキテクチャ，大容量化に伴う，メモリ階層の実現手法，制御用プログラムの構成手法などのハードウェア・サイドの研究や，プログラミング言語や管理用ソフトウェアの研究などが必要である。

ソフトウェア・サイドの研究としては，関係モデルとそれに基く関係代数演算をどのように，ハードウェアの上へマッピングするかという，一種の仮想化技術が重要であろう。このほか，データの構造化手法や，より理論的な，データの意味論，概念変換の手法，分散化データベースなどの問題も重要であり，これらは，報告の他の部分で議論されるものと思われる。

b) 研究テーマ

アーキテクチャ関連の研究テーマとしては，大別して，基本となる言語やアルゴリズムに関するもの，マシンのシステム・アーキテクチャ（構成方式）に関するもの，システムの構成要素となる部品に関するものがある。アルゴリズムについては，(3)項で述べた。また，それらは，システム・アーキクチャに大きく影響する。また，要素となる部品については，ＶＬＳＩ技術など実装関連技術が，現実のシステムの規模を決定する。これらを列挙すると次のようになる。

i) 言語＝関係論理，関係代数，マシンの基本演算の３者の間の変換の最適化。
- 制御構造等の記述に使われる個別データ処理の演算操作をどの言語レベルに入れるかなどの，言語インタフェースの設定。

ii) アルゴリズム＝探索，整列，結合のアーキテクチャを考慮したアルゴリズムの開発，改良。
- ＶＬＳＩ化に適したアルゴリズムの開発

iii) 処理方式＝機能分散を基本とする並列処理方式の開発。

- 並列型，逐次型の効果的な組合せ（パイプライン化など）
- 新方式の導入（データフロー方式など）

iv) プロセッサの構成方式＝要素プロセッサの処理方式。

- 逐次型
- 連想プロセッサ方式
- データフロー方式など

v) プロセッサ間結合方式＝アルゴリズムや理論的解析を含む。

- 接続装置とアーキテクチャ
- 基本素子とＶＬＳＩ化

vi) 記憶階層の構成方式＝データ転送の高速化手法とデバイスの大容量化，高速化（ex. 光デバイス）

vii) 基本ハードウェア要素＝マシンの基本要素とＶＬＳＩ化

- Logic in Memory 素子
- n×mの router
- その他の素子

(5) 実現の例

関係代数マシンの基本機能をサポートするマシンの提案は，いくつも発表されている。ここでは，バブルメモリを用いたＥＤＣの例と，整列と探索を基本とするストリーム型データベースマシンの例を示す。

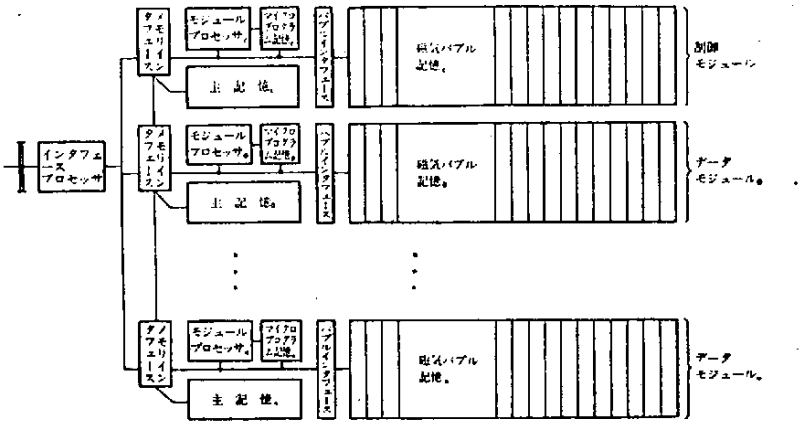


図２－１１　ＥＤＣの構成〔文献(1)による〕

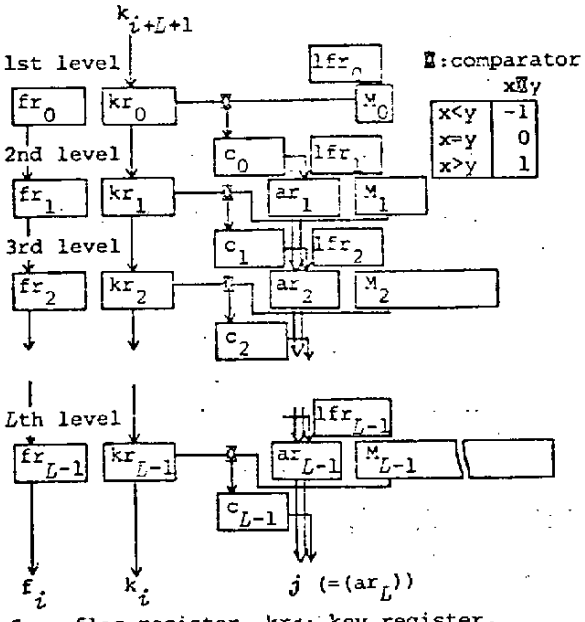
① ＥＤＣ（電子技術総合研究所）

ＥＤＣは実験機が作られ，２５６Ｋbits/chipのバブルメモリを用い，８ケのプロセッサ・モジュールより成る。これは，実験機であり，実用上は，より多くの容量とプロセッサが必要になる。[文献(1)] 図２－１１を参照。

② ストリーム型データベースマシン（北海道大学）

このマシンは，探索と整列を行うプロセッサ（Search Engine(ＳＥＥ)とSort Engine（ＳＯＥ））を基本として，多数のプロセッサを接続した構成となっている。

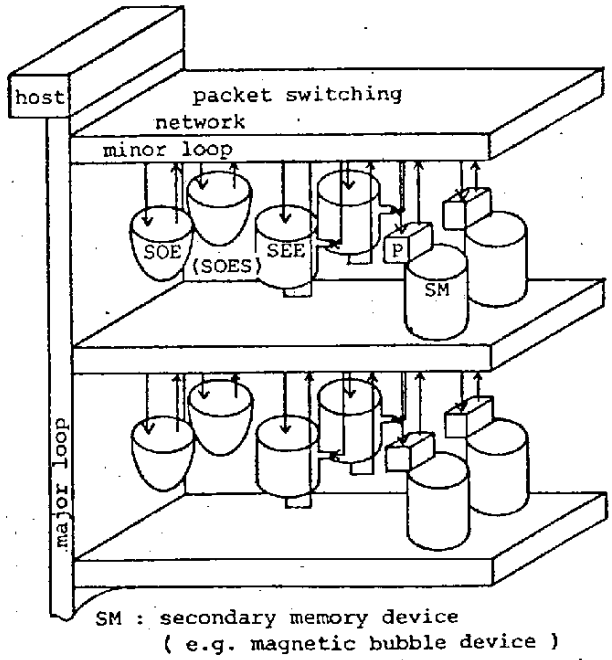
各プロセッサは，その内部に，単位となるセルが，木構造に接続されている。全体としては，下図のように，多数台のプロセッサが接続されている。このシステムは，非常に多くのデータ・セルを用いることを前提とし，処理もパイプライン処理と並列処理を組合せた興味深いものとなっている。[文献(6)]



fr_l: flag register, kr_l: key register,
lfr_l: loading factor register,
ar_l: address register, M_l: memory

Architecture of an SEE.

図２－１２ 一台のSearch Engineの構成

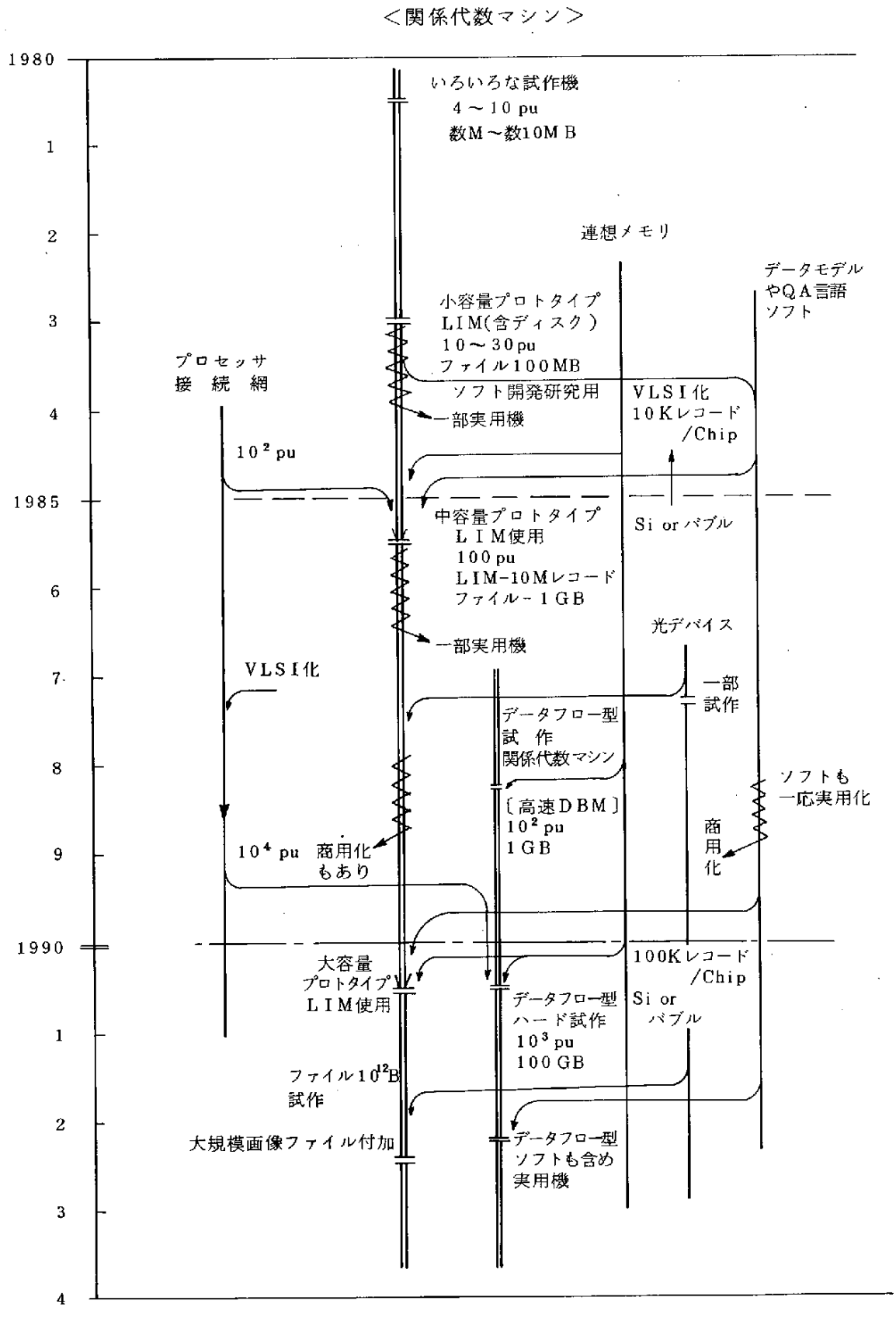


SM : secondary memory device
(e.g. magnetic bubble device)
P : microprocessor (file processor)

An outline of a data flow database computer architecture.

図２－１３ システムの全体構成

(6) 研究開発計画


＜関係代数マシン＞
1980
1
2
3
4
1985
6
7
8
9
1990
1
2
3
4
いろいろな試作機
4～10 pu
数M～数10M B
連想メモリ
データモデル
やQA言語
ソフト
小容量プロトタイプ
LIM(含ディスク)
10～30pu
ファイル100MB
プロセッサ
接 続 網
ソフト開発研究用
一部実用機
VLSI化
10Kレコード
/Chip
10² pu
Si or バブル
中容量プロトタイプ
LIM使用
100 pu
LIM-10Mレコード
ファイル-1GB
光デバイス
一部実用機
VLSI化
一部
試作
データフロー型
試 作
関係代数マシン
ソフトも
一応実用化
〔高速DBM〕
10² pu
1GB
商用化
もあり
10⁴ pu
商用
化
100Kレコード
/Chip
大容量
プロトタイプ
LIM使用
データフロー型
ハード試作
10³ pu
100 GB
Si or
バブル
ファイル10¹²B
試作
大規模画像ファイル付加
データフロー型
ソフトも含め
実用機

(7) 参考文献

1) 植村，前川：データベースマシン，情報処理学会１９８０年５月

2) 有沢：データベース理論，情報処理学会，１９８０年５月

3) Database Machines，ＩＥＥＥ　ＣＯＭＰＵＴＥＲ，vol，１２，３，ＰＰ.７－79，March 1979

4) E. F. Codd：A Relational Model of Data for Large Shared Data Banks，CACM，vol. 13，6，PP. 377-387，June 1970

5) M. M. Astraham：System R：A Relational Data Base Management System，IEEE COMPUTER，vol. 12，5，PP. 42～48，May 1979

6) Y. Tanaka，et，al：Pipeline Searching and Sorting Modules as Components of a Data Flow Database Computer，Proc. of IFIP '80，October，1980

2.2.5 データフローマシン

(1) データフローマシンの目的と効果

従来の計算機は，その基本原理を蓄積プログラム型の逐次処理に置いているが，幾つかの面に於いて問題が出て来ている。データフローマシンはその基本原理を問い直そうとするものであり，データフローマシンに期待される所は次の3点である。

① VLSI向きアーキテクチャ

多数のプロセッサを用いて1つのシステムを作り上げる場合に適した，分散型制御構造，非同期処理に向いているという性質を備えており，従来迄失敗に終っていた超多重プロセッサ構造に新しい指針を与える可能性があり，構造に同種素子を多く用いるという点でVLSIに向いている。

② ソフトウェア作成の容易化

データフローマシンは関数型言語が適しているが，関数型言語はプログラムの検証性が良く，副作用が無い為，ソフトウェアの作成，保守が容易となる可能があり，ソフトウェアの自動作成につながり易い。

③ 超高速マシンの作成技法

データフローマシン用の言語ではアルゴリズムに存在する並列性を自然な形で表現できるとともに，最大限の並列性を取り出し易い。従って，元々並列性が高いベクトル行列演算，偏微分方程式等に対する超高速処理が可能なマシンを，多数のプロセッサにより構成出来るとともに，従来困難であった一般のアルゴリズムの並列化による高速処理も行なえる可能性がある。

(2) データフローマシンの機能

従来の計算機構造との相異点は次のような点である。

① プログラム・カウンタが無い

プログラムの実行制御は各演算に必要なデータの準備状況によって自動的におこなわれる為，多くの演算が非同期に次々と併行して処理される。従って，1つの命令流を定めるというような従来の概念は無く，プログラム・カウンタは存在しない。但し，これはデータフローという点から見た場合の制御形態であって，各演算がどのようにハードウェア上で実行されるかについては述べていない。その演算自体が1つのサブ・プログラムであって，その実行は従来のようなプロセッサ上でプログラム・カウンタを用いて処理されることもあり得よう。

② メモリ・セルの概念が無い

データ・フローマシン上での処理は，関数型言語のように，データに作用を施して新しいデータを作り出すという形で進む。従って，演算結果をどこかに貯えておいて後程それを利用するというメモリの概念が無く，副作用や処理の経違依存性が無い。

③ 並 列 性

従来の計算機ではプロセッサが1台というのが基本であり，処理の並列性が無いのが普通である。しかし，データフローマシンでは処理がおこなわれる所（プロセッサ，演算器等）が多数あり，それらが並列して非同期に使われるという形が基本である。

データフローマシンに期待される応用分野としては，特に次のような所が考えられる。

① 超高速演算

大規模並列マシンにより，科学技術演算を超高速に処理する。偏微分方程式，行列演算等。

② データ・ベース処理

データベース処理は一般に多くのデータに対してフィールドの一致や抽出が行なわれ，原理的に多くの並列処理可能性を秘めている。未だ問題点も多いが，データベース・マシンを作る1つの技法としても期待されよう。

③ 知識情報処理マシンの基礎

従来の計算機システムに欠けている知識情報処理機能をシステムが備えれば，計算機システムの大きな変革が期待される。即ち，人間と同様な「常識」をシステムが備えることにより，人間とのインタフエースは格段に改善されよう。それが進んで計算機のソフトウェア自体が知識として蓄積出来る機能を持つようになれば，ソフトウェアの真の意味での蓄積が実現され，大規模ソフトウェアの漸進的な開発が可能となる。そのような枠組を与えるのが推論マシンであるが，その構造は，関係代数演算，多量データに対する探索や非決定的演算，非数値情報処理等を基本とし，データフローマシンに適した処理である。

(3) データフローマシンの処理方式

まず，データフローマシンを，データ駆動制御の単位であるアクティビティのレベルによって分類すれば次のようになろう。

① 基本演算レベルのマシン

加算，一致，判断といった従来の機械語レベルの演算をデータ・フローの考えで進めてゆくもので，MITのDennisによるマシン，ユタ大学のDDM1，ONERA—CERTで開発されたLAUマルチプロセッサ,がこれに相当する。狭義のデータフローマシンとはこれを指す。最高度の並列性が期待出来るが，従来の計算機構造とは大幅に変わった形での表現が必要になろう。

② 手続きレベルのマシン

プログラムの手続き相互間の処理制御を，その手続き処理に必要なデータの準備状況により独立に実行制御するもので，東大のTOPSTARがこれに相当する。取り出し得る並列性は，手続きのレベルに依る為必ずしもすべての並列性を取り出し得るとは限らないが，従来の計算機との整合性が良く，マイクロプロセッサの複合体としての実現が容易である。

処理方式については，計算機内の処理単位をアクティビティと呼び，それらの起動関係を示したものがプログラムになる。従って，

・アクティビティの表現：

オペレーション，使用パラメータ，結果の出先，状態フラグ等からなる情報のブロックである。

・プログラム：

アクティビティを相互にデータ依存関係を表わすポインタ等によって結合した網。

・トークン：

1つのアクティビティが動いてデータを生成すると，それはトークンという形で他のアクティビティに送られる。各アクティビティに於いて全所要トークンがそろうとその処理が開始される。

という形をとる。データフローマシンの基本構造は以上の通りであるが，処理をより柔軟なものとする為には，次のような拡張が必要である。

① 経緯依存性のある処理

一般に関数型プログラミングでは，過去の履歴に依存するような処理は，それだけでは困難である。特にデータベースの処理等では，このような過去の履歴に依存する処理が必要で，その為にデータフローの考え方を拡張する必要がある。これに対しては，stream とか path expression 等の導入によって扱い得ることが明らかになっている。

② 非決定的処理

OSの資源管理等の処理では一般に，多くの独立したプログラムからの要求を非決定的に扱う必要がある。即ち，それらの要求の発生は前以って定まっておらずランダムである。このような処理では，要求の発生を次々と受付けて処理をおこない，結果を対応する要求元へ正しく送付する必要がある。これに対しては，manager と呼ばれる機構の導入が提案されている。

③ 手続きレベルのアクティビティ

アクティビティのレベルとしては，前述のように機械語相当のレベルと手続き相当のレベルをどこに設定するかは問題であって，様々な観点からの検討が必要である。

(4) 実現上の問題点と研究テーマ

まず全般的な問題としては次のような事項があげられる。

① データフローマシンの可能性と限界

データフローマシンの適用分野は何か，あらゆる処理に対する汎用性を備えているか，そしてそれぞれの分野に於ける能力の限界は何によって抑えられるか，等で，これは，データフローマシン研究自体の目的であるとともに，実用化に当って，明らかにされねばならない特性である。

② 実用的規模のマシン作成とその評価

計算機システムの本当の評価は，実際にシステムを構成し，利用してみないと解らないことが多い。それも小さな"おもちゃ"的規模のものでは余り意味が無く，十分実用的な規模のものでなければならない。

③ データフローマシン用ＯＳの作成と評価

従来迄の研究は，データフロー概念の言語上からの検討と，ハードウェア構成の検討に止まっており，実際にシステムとして利用する場合の制御プログラム構造については殆んど未検討である。

④ 現在の計算機との融合構造の検討

狭義のデータフローマシンの構造は，現在の計算機構造と大幅に異なっている。世の中で使用される為には，従来のマシンとの整合性が必要となる。従来のマシン内の１つの機能モジュールとして独立した形でデータフローマシンを埋め込むとか，各アクティビティの処理をマイクロプロセッサでおこなう等，従来のマシンとの融合構造も検討されるべきである。

⑤ データフロー言語の開発

データフローに適した言語は従来より幾つか発表されているが，それらを様々な応用に照らして評価をおこなうとともに，将来の知識情報処理に向いた言語，例えば論理型プログラミング用データフロー言語の開発が望まれる。評価項目としては特に，あらゆる問題に対する記述可能性，記述の容易性，読み易さ，並列性の取出し易さが重要であろう。

次に，幾つかの点に着目した個別的な問題，及び研究テーマを列挙すれば次のようになる。

① 言語上の問題

- データフロー処理向き機械語セットの決定
- データフロー言語を完備なシステムとする為の機能拡張
- データフロー言語の特性評価

  専用言語か汎用言語か，又プログラミングの容易性，プログラムの読み易さ，既存言語との整合性又移行性
- プログラムの検証手法の組込み

データフロー言語は，正当性の証明が容易であると言われるが，実際に検討された例は無い。検討を進めるとともに検証手法を実用化し，データフローマシンのプログラミングに利用してゆく必要がある。

・経緯依存性のある問題の扱い

stream等の概念を一般化して，処理上望ましい又は必要な経緯依存性（例えばデータベースの扱い）を取扱い易くすること。streamを実装する場合の方法（トークン列とするか，グローバル・メモリを備えて“孔のある構造”として実装するかetc）も検討を要する。

・非決定的問題の取扱い

manager の概念を一般化したり，他の手法を考案すること。及びそれらの評価と実装法の検討が要る。従来の研究では，“こうすれば可能である”という点についての提案に止まり，実装を考慮した能率の良い汎用手法に関しては，全く検討が為されておらず，今後の課題である。

・所要メモリ量の減少化対策

言語の上で，同一のネームを持った所には1度だけしか代入することを許さない(データにユニークなネームを付ける）単一代入規則が用いられるが，そのままの形で各ネームにメモリ領域を割り合てていたのでは，それを実装する場合のメモリ量が著しく増加する恐れがある。従って，繰返し文形式の導入，参照回数の付いた構造メモリ，ブロック構造によるスコープの制限等の導入が必要であろう。

・データ・タイプの導入

データフロー言語にデータ・タイプを如何に導入し，そしてそれを如何に利用するか，は，今後の一般言語の動向とともに検討する必要がある。

② アーキテクチャ上の問題

・アクティビティ・レベルの検討

設定すべきアクティビティ・レベル（機械語レベル）を如何にすべきかの検討。そのレベルと達成しうる並列度との関係，マシン・オーバーヘッドの量，結合網構造との関係等がテーマとなろう。

・全システム構造の検討

プロセッサ，メモリ，スイッチ，入出力装置等の組合せ構造の検討で，アクティビティとの関係，VLSIを考慮した切り方，拡張性等が問題となる。

・プロセッサ内部構造

均一な汎用のマイクロプロセッサを数多く用いるのか，それぞれの機能に専用の演算器を集めた構造とするのか，マイクロ・プログラム制御構造とするべきか否か，

VLSIで実装する場合に高機能のプロセッサを1チップで作るか，低機能のプロセッサを数多く1チップに収めるか等が問題となる。

・アクティビティの制御法

データが揃って，処理可能となったアクティビティの検出と，それを処理部に送る為の機構をどうするか。ポーリング機構連想メモリ利用の検出用ハードウェア，データ設定時のテスト後転送機構など様々な構造が考えられる。

・結合網の構成

数多くの（何十台以上）プロセッサ，アクティビティ・メモリ等を結合する為の結合網のトポロジー，制御法，転送機構，スイッチ回路等の構成法で，如何にすれば性能を落さずに数多くのプロセッサを密に結合出来るかが問題となる。これは，対象とするシステムの総合性能と規模に依存する。10台程度以下の小さなプロセッサを結合する場合は，簡単なバス構造等が適当であろうし，100台を越す場合は本格的なスイッチ網が必要となるであろうが，現在迄の所そのようなスイッチの実用的な検討が欠けている。

・入出力装置の取扱い

キーボード，プリンタ，回線，及び二次記憶をプロセッサやアクティビティ・メモリとどのような形で接続し，処理上それを如何に扱うかについての研究は未だ殆んど為されていない。特に，二次記憶等，多量のデータをデータ・フローの環境に持込む為には，stream概念に止らずトークンとの関係，処理環境との関係を考慮したより実装を考慮した本格的な取扱いが必要である。

・構造メモリの実現

データフローマシンでは，単一代入規則により，異なった名前のデータが次々と作り出されるが，同一名前のデータを何回も用いる場合，その度にコピーを作っていたのでは不経済である。更に，構造を持ったデータのアクセスが便利に行なえる為には構造自体の制御をメモリ側でするような機構が望まれる。このような構造メモリには，多くのプロセッサがアクセスする為，そのスループットがシステム全体の性能を定めてしまう可能性がある。任意の構造を線型のメモリ上に実現する場合には，各構造に対する参照回数の制御，それに伴うガーベッジコレクション等が重要な機能となる。一般に高速並列演算が要求される応用では，行列のような構造が多く使われ，それに対するアクセスの効率は重要であり構造メモリ上でそれを如何に効率良く実現するか問題となる。構造メモリ内に実現する基本データ・タイプを何(整数, 文字, etc)にするか，構造メモリは1ケ所に集中するべきか，各プロセッサに分散配置すべきか等も問題となろう。

③　制御プログラム上の問題

・アクティビティ制御方式

アクティビティが処理可能となる為には，必要な全データがそろっていることの他に，次段のアクティビティ情報を蓄えておく為のバッファが空いている必要がある。そのようなバッファ制御は，へたをするとデッドロックに陥るし，又余りに強い制限を課すと並列性を十分利用出来ない。その制御を如何にすべきか，又，処理可能なアクティビティをどのプロセッサに割当てるべきかーは未だ十分に解決されていない項目である。処理の局所性を利用しつつ，転送オーバヘッドを出来る限り減らす手法の開発が望まれる。

・ジョブ制御手法

複数利用者，ジョブ優先度の取扱い，会話型処理の実現法等は，全く未検討の分野であり，並列に動く数多くのアクティビティをグループとして集中制御する機構の実現問題になる。データフローマシンは非同期であることを特長としている為，逆に集中制御は単純な問題でなく，その理論的な取扱いの検討の他，内部制御機構に工夫が必要になろう。

・割込機能の必要性と実現手法

従来の計算機システムに見られる割込機能は，データフローマシンで必要となるか否か。別の制御アルゴリズムとしてモデル化出来るかどうか。若し必要ならばその実現は如何にすべきか。関連するアクティビティを一度に止めたり，動かしたりする制御が必要な場合，その機構をどうするか，これも汎用機として実現する為には検討されねばならない項目の１つである。更に，プロセスという概念をどのようにとらえれば良いか，そのプロセスと他のプロセスとの交信を如何にモデル化するべきか，他の計算機との通信をどう考えるか等も問題である。これらは，従来のＯＳが持っていた概念，機能をデータフローマシン上で如何に位着付けるかという問題で，データフローの原理に立返った検討が必要である。

・資源管理手法

データフローマシン上で何を資源としてとらえるか，又その管理手法はどうするか，様々な手法の提案と相互評価が為されなければならない。これには，データフローグラフがコンパイル 時には定まらない非決定的な問題を扱う能力が必要である。

・障害対策・信頼化手法

各アクティビティが，エラーを起した場合の再試行をどのようにして実現するか，又，エラーが検出された時，どこ迄，遡って回復を試みるか。これらは，現在の計算機では重要な機能の１つである。数多くのアクティビティが併行して動いて

いる場合の統一的な手法が望まれる。一般には，各アクティビティは独立であるから，アクティビティ単位で再試行を試みればよく，データフローの概念を高信頼化技法の1つとして利用しようという提案もあるが，システム・ダウン時の回復法はチェック・ポイントも含めてより現実的・具体的な検討が必要であろう。

・ごみ集め手法

生成されるトークンの中には，条件分岐や予想していなかった特別のイベントの発生によって不要となりながらもメモリ内に残り続ける場合が考えられる。これらは，メモリの使用効率を悪くするので，これらを如何に検出して取除くかの対策が必要であり，プログラミング言語との関連や障害対策を考慮して検討する必要がある。何時迄経っても動き出さないアクティビティの中には，デッドロックが生じているものも考えられ，その区別をせねばならない。

・保護対策

セキュリティ，プライバシー問題に対して，その保護対策を如何に実装すべきかという問題である。一般的に言えば，共有メモリを持たない構造なのでより容易であると言えるが，従来の計算機とは構造が異なる為，"プロセス"のアクセス権が従来の延長で十分なのかどうか慎重に議論する必要がある。

(5) 実現の例

本格的なデータフローマシンを実現する為に，その途中段階に於いて試作するマシンとしては次のような3例が考えられる。

① 基本操作レベルのデータフローマシン実験機

手続きレベルのデータフローマシンは，既に試作されているので，その実験結果を利用して，基本操作レベルのマシンを製作する。メモリ容量16MB，演算器数16程度のシステムとし，これを用いてアクティビティ・レベルの評価を行ない，その制御法の開発，及びデータ・フローマシン一般の特性についての見通しを得る。

② データフローマシン・プロトタイプ

上記実験機の後，汎用の処理を目的としたデータフローマシンのプロトタイプを試作する。プロセッシング・ユニット100台，所要メモリ200MB程度のマシンで，性能としては，100MOPS(Millions Operations Per Second)程度を目標とする。VLSIを用いた構成とし，将来への見通しを立てる為のシステムで，言語，制御プログラム等の研究・試作に用いる。

③ 高性能データフローマシン・実験機

上記によって，データフローマシンの将来性への見通しが得られれば，高性能化への限

界を調べる為に，高性能実験機を試作する。

- プロセッシング・ユニット　：　10^3 ～ 10^4台
- 使用メモリ容量　：　1GB～10GB
- 性　能　：　1BOPS～10BOPS（Billions Operations Per Sec.：10^9 ）
- 二次記憶との転送速度　：　1GB／Sec

各素子には，VLSIを用い，偏微分方程式の処理，大規模データの高速処理等を対象としてその評価をおこなう。また，これと並行して，小規模なパーソナル・マシンをデータフローマシンによって実現する試作を行なう。プロセッシング・ユニット，32台，メモリ容量10MB，性能10MOPS程度のもので，知識情報処理のフロントエンド・マシンとして論理型プログラミングに適した構造を持たせる。

(6) 研究開発計画

年　代	計　　画
1983	基本操作レベルの実験機試作
	基本命令セット，言語の開発，制御法の開発
1982～86	構造メモリの実現
	初期版　4MB
	実験版　1GB
1982～87	プロセッサ相互結合網の実現
	VLSI向き
	10^3 ～10^4 端子用
1983～86	データフローマシン・プロトタイプ試作
	VLSI Version 1
1984～89	オペレーティング・システムの開発
	二次記憶・DBの考慮
1987～90	高性能データフローマシン試作
	VLSI Version 2
	データフロー型パーソナル・コンピュータ試作
1984～	データフローマシンの実用化研究
	低並列・小規模 大並列・大規模 } システム
	既存マシンとの融合方式

(7) 参考文献

1) Arvind, データフローアーキテクチャの研究開発, 電子協特別セミナ講演録, №55−C−391, 昭和55年4月。

2) 樋口哲野, 山口喜教, データフロー言語の研究動向, 信学誌vol.63, №1, PP.83−87, 昭和55年1月。

3) DDM1：データ駆動型マシン, 電子協技術動向委員会報告№54−C−370, PP.136−148, 昭和54年3月。

4) データフロー・ネットワークのためのOSの原理, 電子協技術動向委員会報告, №54−C−370, PP. 148−155, 昭和54年3月。

5) 栗原謙, 鈴木達郎, 元岡達, High Level Data Flow Machine (TOPSTAR) のシステムプログラム, 信学技法EC79−56, 昭和55年1月。

2.2.6　新（New）ノイマン型アーキテクチャ

〔1〕 目的と効果

NeWノイマン型アーキテクチャ構築の目的は過去30年以上に渡って多くの研究と実践を経て存在し続けてきたいわゆる「ノイマン型アーキテクチャ」を，長所にのみ注目してさらにVLSIの特質を考慮しつつ再構成することによって，完成の域に持っていこうとすることにある。ノイマン型の長所は，「汎用」を満足するアーキテクチャの提供にある。そしてNewノイマンをベースとしつつも，従来の研究より知れたノイマン型の欠点を，ベースを崩さないで改良しているということを表わしている。つまり，VLSIの特質を生かし再構成している，VLSI向きになっているという事の強調にある。

「汎用」とは何か？　汎用とは，まずプログラム可能でなければならない。プログラム可能という事がすべての応用に対しての適応の可能性をもたらす。汎用であることの利点は多い。汎用は大量生産の「きっかけ」をつくり，計算機の適用分野を広げる。プログラム（＝ノウハウ，知識）の流通を可能とし，知的コミュニケーションの輪を広げる。その反面，特定の応用に対しては十分その要求仕様を満足しない場合も出てくる。しかしその欠点を認めても汎用の魅力は大きい。

そして「汎用」性を提供するアーキテクチャとしては，ノイマンの提案したもの以外には現在のところないという訳なのである。

また，世の中の問題の多くはノイマン型で十分と思われるものも多い。汎用＝大量生産可能，よってコストダウンという論理にのる。コストダウンの要求が著しいアプリケーションにはいまだノイマン型は唯一のアーキテクチャ的解である。なにしろ大量生産がVLSIアーキテクチャの前提を支持する。

ところで，ノイマン型の誕生時にはVLSIは無かった。ノイマン型コンピュータの発展の歴史を振り返ると，このためどのようなことが起きたかが如実に知ることができる。即ち，ノイマン型アーキテクチャは，ノイマンにより提案された形とはずいぶん異なってインプリメントされた。アーキテクチャとは時の半導体の力にかなり影響されることは避けられない。半導体技術が未熟なため実現にあたってはかなり方式をデフォルメする必要が出てきてしまうことも多い。事実ノイマン型アーキテクチャはかなりデフォルメされ続けてきた。そして，この影響は実行効率の向上をもたらしたのであるが，その反面，プログラムの開発効率を落したり，その他別の面でのデメリットを多くもたらしたのである。

しかし，我々は現在これらの状況を今一度見直す時期に来ているように思える。即ち，VLSIが利用できるようになった。今や，素子数を減らすためにアーキテクチャを汚くする要因はどこにもない。また，思い切ったノイマン型の改良もできる。ノイマン型の汎用であるという長所だけを伸ばすことを考えて，現在のマシンとのコンパチビリティの確保が保障されると

いう点に注目して，ＶＬＳＩの特質を最大限に生かしたノイマンベース型コンピュータ，それがＮｅｗ ノイマンなのである。

〔2〕 Ｎｅｗノイマンマシンの機能

Ｎｅｗノイマン型の基本機能は，ノイマン型マシンの機能をすべて包括する。しかしここでは，Ｎｅｗノイマンマシンはノイマン型コンピュータのＶＬＳＩを前提とした構築という点に注目して以下議論を進める。よってＮｅｗノイマンマシンのすべての機能について述べる訳では無い。

議論の進め方として，まず，現在のノイマン型コンピュータが構成素子の未熟さのためにどのようにゆがめられているかを述べる。次に，Ｎｅｗとなる一つの大きな要因がＶＬＳＩであるという点を考え，ＶＬＳＩインパクトを述べる。その後，現在のノイマンを基としてＮｅｗとなる「きっかけ」をまとめる。

(1) ゆ が み

ここでは，G.J.マイヤーズに従い，ノイマン型コンピュータのゆがみを述べる。

① 遂次型メモリ

プログラムとデータは一つの順序づけられたメモリに貯えられる。順序づけられた記憶の概念は，高級言語において名前のついた変数として離散的な集合から成る記憶の概念とはまったく類似していない。

② 線形メモリ

メモリは一次元であるが，高級言語で扱うデータには線形とは限らない多次元のデータ（配列，構造体，リストなど）がある。

③ 命令とデータの区別がない。

命令をデータのように扱ったり，データのビット列を命令として実行できるのに対して，高級言語ではデータを実行したり，命令をデータのように参照する概念はない。

④ 意味はデータに備わったものではない

データの意味はプログラムの論理，命令によって与えられる。従って，例えば文字列やアドレスデータに対して浮動小数点の演算を行うことができる。高級言語においては，データの意味はデータそのものに結び付けられている。

⑤ ２進演算

人間とのやりとりのため１０進数と２進数の変換にシステムはかなりの時間を費やしている。また，ほとんどの１０進有理数は計算機内部で２進数の近似値として表現されるため，プログラミングのむずかしさ，エラー，言語の定義を混乱させる原因となっている。

⑥ 固定語長

固定語長であるために表わしうる数の最大値と最小値が決まっている。高級言語の持つ可変語長の概念を固定長の語に写像するのが難しい。

⑦　レジスタ

番地づけ可能なレジスタが存在することは，高級言語の概念と合わない。すべての算術演算に必ずレジスタを使うことや少ないレジスタの最適割り当てのためにコンパイラが生成するコードはソースプログラムの論理を表現するうえでは何の貢献もしていない。

(2)　ＶＬＳＩインパクト

ＶＬＳＩについて，より詳細な議論は４章に述べる。ここでは，その一部につき，特にNewノイマンと関係のあるところにつき述べる。また，ここでのＶＬＳＩとは百万トランジスタ／チップ以上の集積密度のあるものを言う。

①　高密度化により高機能化が図れる。かなり多くの論理素子が入る。よって問題を解く場合，直接必要ではないが，プログラムの開発をしやすくしたり，対故障回路などが，十分入るようになる。

②　ＶＬＳＩ化の中でも１チップ化により，従来外に出ていた結線などが内部に入る。

それにより，通信オーバヘッドが減り，従来速度が遅かったため特別の工夫が入った箇所などがスマートにインプリメントできるようになる。また，その結果もプログラムの生産性向上につながる。

③　ＶＬＳＩ化によりメモリが多く付くため特別の工夫が要求される。

巨大なメモリが付く可能性が出て来たため，論理アドレスも広く取れるようになる。従って，インプリメントに当っては，そのための回路，メモリ・マネージメント回路などが特別に必要となる。

(3)　要求される機能

要求される機能としては以下の通り

①　ソフトウェア開発コストの低減ための工夫

前述したように現在の多くのノイマン型マシンはインプリメントする場合の素子の貧弱さから，かなり悪い方向に（デフォルメ）された方式をとっている場合が多い。具体的には従来，マシン構築にあたりまず考えたことは実行効率を上げることであり，プログラム開発効率を上げることにはほとんど考慮が払われていない。従ってNewノイマンではこれらを改善することをまず考える。またソフトウェアを開発しやすくするための機能としては，プログラムを書くフェーズで書きやすくするための機能と，書いてしまったものを動くようにする（デバック）ための機能，また，でき上がったソフトウェアを管理しやすくするための機能の三つが考えられる。

②　高機能化

問題に適応させて命令セットを変更していくというような，主としてセマンティック・ギャップを減らす方向である。本来のノイマン・マシンにはこのアイデアは無かった。

③　高信頼化

高信頼化は，論理的レベルと物理的なレベルの二つに分けて考えられる。論理的なレベルでは，いわゆる不注意によるプログラム・ミスを防止するケーパビリティ・アドレシング・メカニズムの実現などである。他方，物理的なレベルでは自己故障診断機能，または，耐故障（設計）機能の実現ということになる。

④　コンパティビリティ

Newノイマンはもともとevolutionalな性質の方が大きいため，従来のマシンとのコンパティビリティがとりやすい。具体的にはエミュレーション機能が重要であるが，この場合単なる機械語だけでなく，オペレーティング・システムのエミュレーションをも含む広い意味でのエミュレーション能力が要求される。

〔3〕処理方式と効果

(1)　処理方式

Newノイマン・マシンを作り上げる処理方式として考えられるものとしては，以下のようなものが有る。

①　ダイナミック（アーキテクチャ）方式

ダイナミック・マイクロプログラミング制御方式をとっているマシンをユニバーサル・ホスト・プロセッサorマシン（UHP or UHM）という。ＵＨＰにより，問題に適応した命令セットの実現が可能となり，実行速度が上がるだけでなくソフトウェアの生産性向上ということにもつながる。また，ハードウェア・メンテナンスをやり易くするマイクロ・ダイアグノシス技術があり，耐故障機能＝FTC　の実現にもＵＨＰの基本機能であるダイナミック・マイクロプログラミングは必須である。

ダイナミック・マイクロプログラミング制御方式は少ない規則正しい論理素子で実現可能なためＶＬＳＩアーキテクチャの制御方式としても有用（ＶＬＳＩを生かしきれる方式）である。（詳細はＶＬＳＩの項参照）

②　スタック方式

ゼロ・アドレスマシン，いわゆるスタックがオペレーティング・システム，コンパイラなどの言語処理プログラムに有益で有ることはよく知られている。特に，ＦＩＦＯ動作だけでなく，自由度の大きいスタック（スタックの途中からデータの入出力ができたりするもの）や，ポインタ管理のハード化が重要である。

③　タグ方式

②と並び，これもオリジナル・ノイマン型マシンには見られないものである。データに性質を付ける。実行時には，これをみて動作を決める。従って，命令の数が減り（例えば，加算は一つあればよい。整数か実数かなどの区別はデータについており，これにより自動

的にタイプが分かる）これによるメリットはソフトウェアの作成の容易さと実行オブジェクトの信頼性向上である。

④　ディスクリプタ方式，モジュール・オブジェクト方式

タグの概念と似ているが，タグ（印）を付ける対象をもう少し拡大し，集団に付けるようにしたものである。

⑤　アドレス管理機構

これもオリジナルなノイマンのアイデアには無かったものである。ＶＬＳＩによりメモリ価格は下がり，巨大空間の確保が容易となった。つまり，そのための特別な工夫が必要となる。一つには巨大空間の管理機構である。また，スーパー・パーソナル・コンピュータではマルチ・プログラミングの必要性は必ずしもないが，マルチ・タスクなどの必要性はあるため，やはりメモリバンク切り替え機構などメモリ関係の管理メカニズムが重要である。

⑥　マルチNewノイマン機構

マルチ・マイクロ・プロセッサ・サポート機構，複数個のNewノイマンを接続するための機構。（ノイマン型の並列接続）

⑦　ノイマン型高速演算機構

マルチ・マイクロにしても一つの構成要素が速い方がシステム全体の速度が上がることは明らかである。従って，逐次処理機構も高速にする必要があり，パイプライン方式の拡張機構などが考えられる。

⑧　プロテクション・メカニズム

③，④などとも関係するが，もう少し広くシステムレベルでの（入出力を含む）プロテクション・メカニズムが考えられる。

(2)　効　果

以上の機能により，どのような効果（利点）が生まれるかの関係を次に示す。

機　　能	処理方式	効　　果
ソフトウェア開発 コスト低減	ダイナミック タグ ディスクリプタ	・ソフトウェアが作り易い。 （イニシャルに書き易い） ・デバックが容易。 ・ソフトウェア・メンテナンスがし易い。
高機能化	ダイナミック	・速度が上がる。

	スタック	
	アドレス管理機構	・使い易くなる。
	マルチNewノイマン機構	（ソフトウェアが作り易い）
	ノイマン型高速演算機構	
高信頼化	ダイナミック	・論理的プログラム・ミスが減る。
	タグ	（信頼性が上がる）
	ティスクリプタ	
	プロテクション・メカニズム	
	ケイパビリティ・アドレス	
		・信頼性の高いプログラムを作り易い。
		・ハードウェア・メンテナンスがやり易い。
		・システムとしての信頼度が上がる。
コンパティビリティ	ダイナミック	・コンパティビリティが保証され
	エミュレーション	ソフトウェア財産が生かせる。

〔4〕 実現上の問題点と研究テーマ

以上〔3〕で述べてきた多くの機構は，既にオリジナルなアイデアは提案より数年ないしは数十年経っている。しかしながら，〔1〕でも述べたようなアイデアとは別に実現の前提はＶＬＳＩにあるため，インプリメント上の問題は解決されていないものが多い。特にこれらの機構を組み合わせ一つのシステムとした場合の問題点は殆ど分かっていないと言える。（そのため，システムとしてまとめあげた結果，機構のオリジナル・アイデアを大幅変更する可能性も多いであろう。）

コンピュータ・システムとは実働して始めてその価値が出るのであり，実働という面から考えるなら理論的なアイデアの構築だけでは全く価値がない。よく実働させるためのアイデアは，本質ではないように考える人がいるがそれは間違っている。コンピュータとはエンジニアリング・プロダクトの一つと考えるなら，理論（メカニズムのアイデア）→実験→評価→メカニズムの変更というプロセスをふまなければ完成（実働）はできず，Newノイマンとしても例外ではない。

メカニズム考案に終らず，システム構築，評価というサイクルをとるには多大の研究費と時間がかかり地道な努力が必要とされる。特に我が国では新しいメカニズムなどに対して種々の

理由からこのサイクルが取られないでしまうことが多いが，ぜひ正常なサイクルで研究，開発を進めたいものである。実現上の問題点としてはこのようなアイデアだけでまだ本当は分かっていないものを完成させるということに対しての理解を深めさすという所にあるような気もする。アイデアが出て数年経つと何もしないでも解決したような錯覚に陥ることがよくあるが，今ここできちんと整理し直す時期に来ているように思えるのである。

また研究テーマとしては，〔3〕で述べた機構のより詳細な実現をふまえた検討の他，ノイマン型すべてに渡りＶＬＳＩを前提とした考え直し（見直し）という大きなテーマがある。これは多くの点で興味深いが，例えばメモリの速度が十分あるなら，レジスタの概念を止めてしまう（メモリ・アーキテクチャ）とかを行う。これによりコンパイラなどのレジスタ割り付けという厄介な話はすべてなくなる。また，今一度ＩＬＬＩＡＣ Ⅳ のようなノイマン型結合で並列処理を行うことを再検討してみる価値もあろう。さらに一つのプロセッサを徹底的に速くするメカニズムの追及，例えばＳＩＭＤアーキテクチャは，従来二桁程度の速度上昇しかもたらさなかったが，ＶＬＳＩ１チップでもっとできないのであろうか等も必要と思われる。

以下，項目的に整理する。

①Ｎｅｗノイマン機構のシステム
　アーキテクチャの観点からの研究
②Ｎｅｗノイマン機構のＶＬＳＩ化
　を前提とした研究

→ Ｎｅｗノイマンコンピュータを核としたパーソナルコンピュータの研究

③既存ソフトウェアのコンパティビリティ
　を保つための広域エミュレーションの研究
④Ｎｅｗノイマンマシンの効果測定法の研究

〔5〕 実現の例と計画

第５プロジェクトにおけるＮｅｗノイマンマシンは，ＶＬＳＩ１チップアーキテクチャの中身として考えるのが妥当である。Ｎｅｗノイマンをシステムとし，ＶＬＳＩ１チップでどう作るかだけに焦点を絞る。これはスーパー・パーソナルの中身として使う。従って，シングル・ユーザ用を意識すればよく，従来マルチ・ユーザを前提として作られていた部分はこの際取り入れる必要はない。ＵＨＰを主体とし，〔3〕で述べた要求される機能が１チップにおさまるようにし，分散システムの中でのプロテクションを考え，巨大論理アドレス・アクセスが可というようなものを作る。百万トランジスタ／チップ版と千万トランジスタ／チップ版は分けて考えた方が良く，百万トランジスタ／チップ版を１９８５年以前までに，千万トランジスタ／チップを１９９０年の目標として，プロジェクトをスタートさせる。

研究開発の進め方としては，まず〔2〕の(3)で述べた必要な機能のＶＬＳＩ化を前提とした研究が必要である。その後ＭＡＲＫ Ⅰへと進める。又，Ｎｅｗノイマン機能が第５世代コンピュータ

のコンポーネントとなることを研究するプロジェクトも必要である。

また，もう一つアーキテクチャ・データベース研究プロジェクトを発足させ，このようなNewノイマンＶＬＳＩチップをどのように応用問題に適応させるか（主としてＮｅｗノイマンのファームウェアとソフトウェア）の研究を行う。さらに，それぞれの計算速度はそれほどではないが，個々の仕事に合った計算機をそれぞれがどう持つかという，建築で言えばハウス５５に相当するようなきたるべき世の国民一人一人が皆コンピュータを持つような時代に備える計画が考えられる（これをマイク９０という）。研究の中心は，ＶＬＳＩ－ＣＡＤであるが，これにより目的にあったＶＬＳＩチップがユーザーにより作ることができるようになる。マイクロ９０計画のためには第5世代コンピュータは必要不可欠であり，第５世代の威力を示すための計画であるともいえよう。

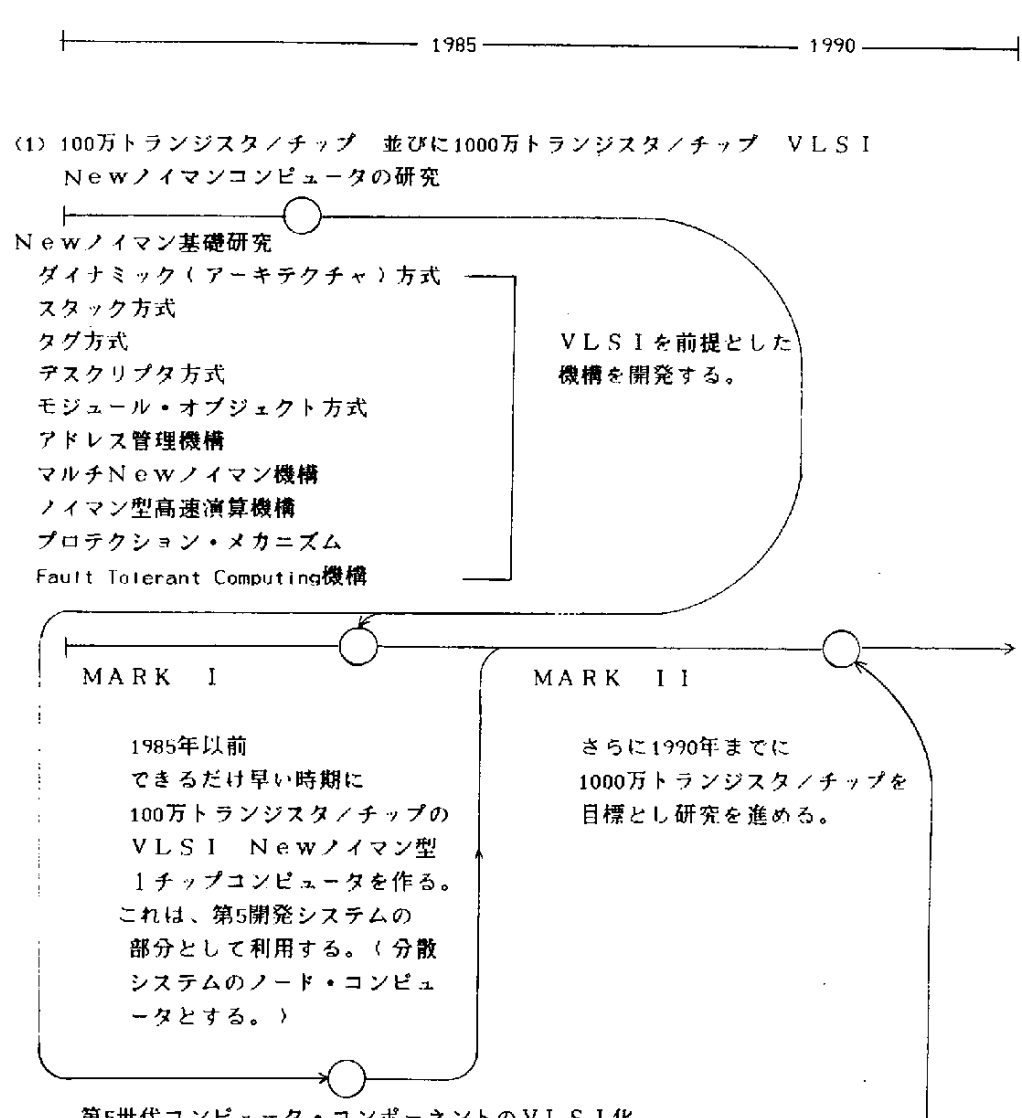
以上の計画を図示したものを次ページに示す。

参考文献

(1) Ｇ．Ｊ．ＭＹＥＲＳ：Ａdvances in Ｃomputer Ａrchitecture, Ｊohu Ｗiley & Sons, 1978

(2) 相磯，飯塚，坂村：ダイナミック・アーキテクチャ，共立出版，１９８０

(3) 第５世代の電子計算機に関する調査研究報告書－アーキテクチャ研究分科会－，（財）日本情報処理開発協会，1980


1985
1990
(1) 100万トランジスタ／チップ　並びに1000万トランジスタ／チップ　ＶＬＳＩ
Ｎｅｗノイマンコンピュータの研究
Ｎｅｗノイマン基礎研究
ダイナミック（アーキテクチャ）方式
スタック方式
タグ方式
デスクリプタ方式
モジュール・オブジェクト方式
アドレス管理機構
マルチＮｅｗノイマン機構
ノイマン型高速演算機構
プロテクション・メカニズム
Fault Tolerant Computing機構
ＶＬＳＩを前提とした
機構を開発する。
ＭＡＲＫ　Ｉ
1985年以前
できるだけ早い時期に
100万トランジスタ／チップの
ＶＬＳＩ　Ｎｅｗノイマン型
１チップコンピュータを作る。
これは、第5開発システムの
部分として利用する。（分散
システムのノード・コンピュ
ータとする。）
ＭＡＲＫ　ＩＩ
さらに1990年までに
1000万トランジスタ／チップを
目標とし研究を進める。
第5世代コンピュータ・コンポーネントのＶＬＳＩ化
(2) アーキテクチャ・データベースの研究
種々の応用にどのようなアーキテクチャが
最適であるかの研究
融合
融合
(3) マイクロ90
1990年目標で1000万トランジスタ／チップ
のＶＬＳＩコンピュータを作り上げるが、
この中身は目的指向アーキテクチャで、ユ
ーザーの応用に従い異なるというものである。
また、このためのＣＡＤは、第5システム上
に作られることが想定されている。

3．機能分散アーキテクチャ

3. 機能分散アーキテクチャ

本章では，第５世代アーキテクチャとしての機能分散型アーキテクチャについて述べる。基本的考え方は，前年度報告書にあげた"システムＡ"と同一であるが，更に詳細な展開を行ないあり得べき姿や，その実現にあたっての問題点が明らかになるように努力した。

以下では，まず3.1において機能分散処理方式を一般的に考察した後，3.2において全体の形を決めるシステムアーキテクチャについて，3.3において機能分散システム全体の制御方式（オペレーテイングシステム）について述べる。また，3.4においては分散された各機能を実行するコンポーネントマシン群について，3.5ではマンマシンインタフェースについて考察するが，本論に入る前に次の点を特に注意しておきたい。

それは機能分散アーキテクチャとはこれまで全くなかったアーキテクチャではなく，従来基本的に存在する考え方を発展させ，実用にたえ，効率的なものとすることにねらいがあることである。即ち，機能分散アーキテクチャはいわば，Evolutionary（進化的）アーキテクチャであり，２章に述べられたアーキテクチャのようにRevolutionary（革命的）なものではないことを意味している。

しかしながら，このことは機能分散型のすぐれたアーキテクチャの開発が容易なことを意味するものでもないし，２章のアーキテクチャより重要でないことを意味するものでもない。あえていうならば，Revolutionaryなアーキテクチャは，Revolutionaryであるが故に成功の暁には大きな効果が得られようが，情報処理システム全体から見れば，それだけが将来の情報処理システムの全てであると考えるのは誤りである。特に，Evolutionary アーキテクチャがRevolutionaryアーキテクチャの前段階で，ある時期をすぎると後者がとって代るというように考えるべきではない。

Revolutionary アーキテクチャの多くは外から見たマシン・アーキテクチャである。したがって，速度を問題にしなければ，大部分をソフトウェアで作成してもよいのであるが，それでは実用にならず，せっかくの利点を発揮できない。即ち，その目的とする効果をあげるためには，革新的アーキテクチャの工夫のみならず新しいハードウェア技術にもとづいたEvolutionaryアーキテクチャのサポートが不可欠である。逆に，データフローマシンの考え方のように，科学技術計算マシン等の機能分散型アーキテクチャに取り入れられて，効果を上げられる可能性が大きいものもあり，これらの考え方を機能分散アーキテクチャ実現の手段ととらえることもできよう。

要するに，少くとも，現在考えられている時期内では両者は相対立するものでも，一方が他方の前段階でもなく，互いに補いあって将来の情報処理システムを構築するために，並行して研究開発を進めることが必要であると考えられる。

3.1 機能分散処理方式

本節では，機能分散処理方式について，一般的に考察する。

3.1.1 機能分散の目的と効果

機能分散を行なう目的，あるいは，それによって得られる効果には多種のものがある。3.1.2.に述べるように，機能分散はいろいろなレベルで適用されるから，中には，それぞれのレベル固有の目的や効果もある。

まず，アーキテクチャ的見地から，ほぼどのレベルにおいてもあり得るものをあげると，次の5点になる。

(1) スループット又は実行速度の向上

複数の処理要素を同時に動作させる並列処理によってシステムのスループットを拡大したり，同一問題を処理する実行時間を減少させるという効果である。この特徴は，機能分散型と同一のスループットや実行時間を単一処理装置で達成しうる場合には，効果や目的として数えられるかどうかは，性能／コスト比の問題となる。しかし，素子の速度を一定として考えるならば，単一処理装置には限界があるのは確かであり，それを越えるスループットや処理速度を得るには，機能分散による並列処理効果を生かすことが不可欠である。

(2) システムの成長性／構成の自由度の確保

情報処理システムが高度化すると，これまでの全ての機能をリセットすることは事実上不可能になり，いきおい，システムには連続的な成長性が求められる。また，応用環境の多様化が進んでいるので，応用側の要請に合せた形にシステムを構築できる構成の自由度の確保も重要である。このような目的のためには，モジュール構成の機能分散アーキテクチャが，集中型単一システムに比べて秀れていることは明らかである。

しかし，データが分散されているような機能分散系において，データの分割形態の変更を行なうことは極めて困難であるから，設計段階で，機能やデータの分割形態及び，機能相互のインタフェースについて十分な配慮をした設計がなされないと目的は必ずしも達成されない。このシステム設計理念の確立は新しい機能分散アーキテクチャ開発における最大の課題の一つである。

(3) 高信頼性／高稼動率の達成

分散型のシステムでは一部構成装置の故障においても全システムをダウンさせることなくフェイルソフト系を容易に実現することが可能である。この点に関し，最近のシステムでは処理装置等のハードウェアの占める比率が減少しているから，処理装置等が多重になっても総合信頼度に対する寄与はあまり大きくなく，診断機能の豊富な単一大マシンを用いる方がよい場合もあるという指摘もある。確かに，現在の技術レベルでは，そのほうが現実的である場合も少く

ないが，１９８５年以降になれば，診断，チエック等の論理的機能はＬＳＩのコンポーネントプロセッフ内に現今の大型機以上の高度のものが容易に取り入れられると予想され，それを駆使し，構成の自由度を生かすことによって，高信頼性／高稼動率についてやはり多くを期待しうるものと思われる。また，必要に応じて，性能と信頼性のあいだのトレードオフを動的に変更することも機能分散アーキテクチャでは可能になる。

(4) 機能専用化による処理効率の向上

機能分割された各コンポーネントをその機能専用に設計することによって処理効率の向上が期待できる。専用化はハードウェアによることが効率上望ましいことは当然であるが，ハードウェアを個別に設計することは一般にコスト的に不利である（これを容易化するためにも高度の設計サポートシステムの開発が必要である）から，ユニバーサルホストプロセッサ上におけるファームウェアによる専用化も，大いに利用されるべきである。

この他，専用化には個々の機能に対する需要の変化，実現技術の進歩に対する迅速な対応，システム動作のモニタリングやチューニングが容易といった利点もある。

(5) システム構築の容易化

機能分散アーキテクチャにおける各コンポーネントの設計は，その機能とシステム内の他のモジュールに対するインタフェースが明確に定められ（論理的にも，物理的にも）ていれば，他とほぼ独立した設計を行なうことができ，複雑なシステムを一まとめに設計するよりはるかに容易である。また，インプリメンテーションの基礎技術となるＶＬＳＩは当然のことながら分散型システムに適しているし，前項にも述べたように機能単位に分割した方が，環境条件の変化に対応しやすい。

以上のほか，主にネットワーク・レベルの機能分散にしいていえるものとして，次のような効果も期待できる。

(6) 通信コストの低減による性能／コスト比の向上

パーソナル・コンピュータに代表されるローカルな処理機能を利用して，そのサイトで可能な処理をできるだけ行ない，通信回線による情報の転送量を減少させる。このためには，ローカルな情報はできる限り（場合によっては他と重復が相当程度あっても）ローカルに配量することが必要である。したがって，通信コストの低減はローカル記憶の増大とのトレードオフとなる。

(7) セキュリテイ向上

(6)と同様にローカルに情報を置くことに起因して情報のセキュリティの向上が期待できる。しかし，前と同様にセキュリティ確保に必要なハードウェア等他の要因とのトレードオフであるし，実際上の効果は定量的にはむろんのこと定性的にも評価されていない。いずれにしろ

機能分散を指向する主要因ではなく，副効果程度のものであろう。

(8) ニーズに合ったユーザ・インタフェースの提供

ローカルに分散した処理機能を利用することによって，個々のユーザの希望に即したインタフェースを提供できる，と同時に，応答性の向上も期待できる。

(9) 資源の共有

情報や，特殊な処理装置等のハードウェア及びソフトウェアを容易に共有あるいはアクセスすることができる。即ち，そうした資源をまれに使用するのであっても容易に利用できるようになるし，逆に，たまに利用するだけなら自分のサイトにおくための費用を低減できる。

次に，機能分散アーキテクチャを採用することによる逆効果を考えてみると次のようなものがあげられる。

A 機能分散によるオーバヘッド

ハードウェア的なものと，時間的なものが考えられる。前者は，例えば同一の情報を複数ケ所で記憶するための媒体や，一般に各コンポーネントの利用率が下るため等のハードウェア総量の増加であり，後者は機能間の通信を行なうためのプロトコルの作成や変換等に要する時間的オーバヘッドである。前者についてはハードウェア・コストの低減動向からすれば，将来的にはその程度は転減する方向にある。

B 機能マシンの種類増大

機能分散アーキテクチャでは多種のコンポーネント・マシンが必要になることのデメリットであるが，ユニバーサル・ホスト・プロセッサのような汎用処理ユニット利用や適応化技術，設計サポート技術の高度化によってカバーできるであろう。

C システムの複雑化

機能分散システムでは機能が各所に分散されるためにシステムが複雑化してわかりにくくなるという指摘もあるが，むしろ多くの話がローカルにすまされるようになるので，見た目程の複雑化は生じず，保守性等もむしろ向上する可能性がある。しかし，これを達成するには，特にオペレーティングシステムを初めとしたシステム制御機能について，新しい理念の開発が必要である。

以上機能分散アーキテクチャにおける肯定的な面と否定的な側面を整理した。もちろん，(1)～(9)にあげた全ての効果が常に得られるとは限らず，(1)等は分散の方法が悪ければ，欠点A等のために必ずしも達成されない場合もあろう。しかし，全体的な判断をするならば，欠点としてあげたものは，それほど大きなものではなく，たとえ(1)の利点が得られないような場合でもその他の利点から十分な利益を得られるし，このような目的が達成できなければ，将来の高度

な情報処理システムの構築は不可能であるといっても過言ではない。結論的にいって，これまでの議論から，第5世代アーキテクチャにおける機能分散アーキテクチャ採用の必然性は十分明らかであると考えられる。

3.1.2 機能分散のレベル

機能を分散するといっても，何を，どのように分散するかは大いに問題である。これまでも機能分散の考え方はいろいろな形で取り入れられてきたことは周知の事実であるが，惜しむらくは，全体を統一した一つの設計理念を持って設計されたものはない。この点は第5世代における機能分散アーキテクチャに最も求められることであるが，機能分散を適用するレベルということで考えると，次の3つのレベルでそれぞれ機能分散を行なうことができる。

(1) ユーザサイドと中央システム間の分散

ユーザの手元にあるパーソナル・コンピュータ，あるいは特定の工業プロセスを管理するプロセス・コンピュータ，各ディビジョンのデータ処理を行なうコンピュータ等と中央サイトの強力なコンピュータ・システムとの間の機能の分散である。

詳細は3.2.3のネットワーク・アーキテクチャの項で述べられるが，次のような点が主な特徴である。

- 第5世代アーキテクチャでは，ユーザサイドのパーソナル・コンピュータ等のローカルな処理機能が高度化し，通常の機能は全てまかなわれる。情報についても，使用頻度の高いものや，ローカル性の強いものは極力ローカル・サイトに配置し，通信コストの低減，応答速度の短縮がはかられる。
- 一方，中央の共同利用される高機能コンピュータ・システムは，ローカル・サイトでは機能不足な超高速処理や特殊処理の提供と共通情報の管理が中心となる。
- ローカル・サイトに配置される代表的装置のパーソナル・コンピュータについては，3.4.2に記述されるが，単に，処理機能とユーザ個有の情報管理等を行うだけでなく，ユーザ・インターフェース装置としてコマンドのサポート，中央サイトのコンピュータあるいはそれに接続される他の全ての処理系をアクセスするための制御機能もその重要な役割である。

(2) センタの高機能コンピュータにおける機能分散

ここでいう高機能コンピュータとは，システムAにおけるレベル2とレベル3を総合したものである。レベル2とレベル3は処理能力的には大きな違いがあるが，構成的には特別の差はなく，まとめて考えることができる。以下では，特にことわることなくこのような立場で議論を進める。

この高機能コンピュータ・システムのアーキテクチャについては，3.2.2において述べられるが，ここでの機能分散には次のような特徴がある。

- パーソナル・コンピュータでは不十分な超高速処理や特殊処理，特殊ハードウェア・デバイスの管理それに，共通データの管理，他のシステムやローカル・サイトとの通信管理が主な仕事となる。
- 3.1.1 に述べた機能分散アーキテクチャの特徴を有効に利用するためにおおむねタスク・レベルの作業を実行する専用マシンを機能分散の単位とする。
- マシンにローカルな情報はできるだけ各マシンに分散配置されるべきことは，ローカル・サイトと同様であるが，当然(1)の場合より機能マシン間の結合性は強くなるから統一性のある高速の機能マシン間通信系とシステム構成の自由度と高信頼性確保のための柔軟な構成方式が求められる。

(3) コンポーネント・マシンにおける機能分散

ユーザ・サービス・マシン及びマシン・サービス・マシンは，それぞれ性能を高度に追求した専用マシンとして構築され，各マシンそれぞれもその処理内容に応じた機能分散型構成を取ることが必要である。したがってここにおける機能分散の大きな特徴は，個々のマシンの処理内容に応じた構成が取られることである。それは 3.4 に個別に論じられるが，ここでの機能分散のねらいが主に，機能間の並列処理による処理速度の向上及び複雑な専用マシンの構築技術にあることはほぼ共通的にいえる特徴である。

3.1.3 分散データベース

計算機システムの主要な機能は，情報の処理と情報の蓄積である。機能分散の議論は前者に集中しがちであるが，後者の重要性も言を待たない。一般的にいって，処理機能は必要なものが直ちに得られれば，どのような形で存在してもユーザにとっての違いはあまりない。しかし，情報の利用の方は，蓄積された情報量が多く，共同で利用できる方がより有効であるから，本質的に集中化の傾向がある。したがって，データベースは，機能分散の考え方とそぐわないとも考えられるが，事実は，本来ローカルに存在するデータベースを結合して，システム内の全ユーザに蓄積された情報を有効に利用させるのであると逆に考えられるべきである。このように考えれば，システムAに代表される第5世代アーキテクチャにおける分散データベース構築方式は極めて重要な研究課題の一つであることは明らかであろう。

(1) 分散データベースの構造

- 各所（但，以下ではセンタコンピュータ内のデータベースを考え，パーソナルなものは考えない）に存在するデータベースを論理的に結合するための機構としては，次のような 4 層モデルが考えられている。

外部スキーマ　　：　応用ごと

　　｜（ＥＸＳ）

グローバル概念スキーマ　：　論理的一体化

　　｜（ＧＣＳ）

ローカル概念スキーマ　：　個別データベースを標準化したもの

　　｜（ＬＣＳ）

ローカル内部スキーマ　：　個別データベース

（ＬＩＳ）

- システム内のユーザは，これらのスキーマ間の変換を通して分散された全データベースを統一的に見て利用することができる。
- これらのスキーマ間の変換情報はデータ・ディクショナリ／ディレクトリと呼ばれているもので次の３種が考えられる。

｛ＥＩ｝　：　ＥＸＳ－ＧＣＳ

ＤＩ　　：　ＧＣＳ－ＬＣＳ

｛ＨＩ｝　：　ＬＣＳ－ＬＩＳ

(2) 機能分散アーキテクチャにおける実現形式

- ディクショナリ／デイレクトリの配置としては次のような形が考えられる。

｛ＥＩ｝　：　ユーザサイドのパーソナル・コンピュータ内とセンタ・コンピュータ・システム内

ＤＩ　　：　関連する全てのセンタ・コンピュータ・システム内

｛ＨＩ｝　：　データベースが存在するシステムのセンタ・コンピュータ内

- ＥＩはセンタ・コンピュータ内のサービス・マシンに置いてもよいが，ユーザに全データベースを仮想化して見せるためには，ＥＩをパーソナル・コンピュータ内に置く方がよい。
- ＤＩは関係代数のような形で，リレーション対応に記述され，量はあまり大きくならないので関連する全センタ・コンピュータ内に置ける。
- ＨＩは個々のデータベースが存在する場所に個別に配置される。

(3) アーキテクチャへの要請

- パーソナル・コンピュータは各ユーザのデータアクセス要求をその個人の応用目的に合わせたviewを環境として，標準的な問合せ／更新要求に変換する。
- その問合せはそのパーソナル・コンピュータの属するセンタ・コンピュータのデータベース・マシン（または，そのための専用マシンまたはマシン・サービス・マシン）において分解され，いくつかの個々のデータベースに対する問合せに変換される。

- これを実行すると，データベースを備えたセンタ・コンピュータ・システム（のマシン・サービス・マシン？）はそのシステム内のローカルな問合せに変換して実行し，例えば，リレーション単位のデータの授受があって，それらに関係代数演算を施してユーザに渡すデータが作成される。
- 上記のような操作においてシステムに求められる機能としては次のようなものがある。

① データベース・マシンの処理機能

パタンマッチ，関係代数演算，記号処理

1回の処理量は，ＰＬ／Ｉで２０００行程度といわれる。

② 情報転送速度

データベースを所有するセンタ・コンピュータ・システム間の情報転送に必要な伝送速度を評価すると次のようになる。

リレーションの大きさ　：　２Ｋバイト×１００

転送回数　：　数回（EX. 3回）

応答時間　：　1秒

同時アクテイブ数　：　２０

（パーソナル・コンピュータ数　：　３００（１システム））

$$\text{転送速度} \quad \frac{2K \times 100 \times 8 \times 3}{1/20} = 96\ \text{Mbps} \fallingdotseq 100\ \text{Mbps}$$

③ データー貫性の維持制御

＊ 信頼性の高い"放送"通信機構

１：ｎのセッション制御，高信頼通信路

＊ 計算処理の絶対時刻のマーク

通信クロックとコンピュータの周期

＊ データベース間の制御ユマンドを迅速に授受できること。

④ データ・セキュリティ維持

＊ 暗号化ハードウェアのサポートをマシン・サービス・マシン内に用意する。

＊ キー保管庫のサポート

⑤ 高速退避用二次記憶のサポート

大容量の主記憶（～１ＧＢ）のデータの退避10^3Ｂ幅，但，単純な転送のみでよい。

(4) 知識ベースとの関連

- データ・デイクショナリ／デイレクトリの発展形として知識ベースが組込まれていく。
- 複数データモデル間の対応付け等がデータの意味を用いてより精密，かつ知的に行なわれる。

- データベース自身が知識ベースの機能を持ち，分散データベースとしては，データ内容の一貫性維持，複数データベースの統合化等において知識ベースが重要な働きをする。
- 従来は，応用目的ごとに外部スキーマを作成していたが，述語論理型プログラミング環境と"常識"とから，かなりの部分が補われ，その操作が単純化され，ひいては概念スキーマ自身が簡単になる可能性がある。

3.1.4 機能分散アーキテクチャの開発

(1) 研究開発課題

機能分散アーキテクチャにおける研究開発課題は，各レベルまた，各マシンごとに，次節以下に詳しく議論されているが，ここでは共通的なものをいくつかまとめておく。

- 各レベルでの基本設計理念の確立

  ＊マシン間論理インタフェース，＊情報変換方式，＊情報配置（分散）方式

  ＊物理インタフェース
- 分散型データベースの諸問題

  3.1.3 参照
- 分散型オペレーティング・システム（システム制御方式）

  3.3 参照
- インプリメンテーション

  実装，高速接続方式，ＶＬＳＩ化
- データ収集，フィードバック方式

  既存データの収集，解析，データ収集機構の組込
- 複数プロセス，複数機能マシンの協調処理の諸問題

  同期，記述，デバック，etc
- 自動適応化機構

  命令セット，並列構造，情報配置，資源割当

  処理量と信頼度の自動調整
- 機能分散システムの記述と構成の生成

  記述言語，その処理系
- 高信頼化技法

  マシン内，及びマシン間における高信頼化，自動システム再構成方式

(2) 開発のポイント

本節では，機能分散アーキテクチャの諸問題を全体的に検討した。個々の構成方式，研究課題は次節以下に述べられる。

本節の議論からも明らかなように，機能分散方式は第５世代において採用すべきかどうかを検討するのではなく，もはや，"ＭＵＳＴ"になっていることはほぼ専門家の一致した見方といってよい。事実，新しいシステムはそのような方向に着実に流れている。では，放っておいても自然に実現されるかというとそう簡単なものではない。よいシステムをつくるためには，全体的に深く検討された統一的設計理念にもとづき，現実の要請とハードウェア等の技術の状況にあわせてシステムを設計せねばならない。機能分散がよいということさえわかっていれば，それが可能なのではなく，その実現には多大の知恵と時間と費用が不可欠である。

では，どのようなステップですぐれた機能分散アーキテクチャは構築すべきであろうか，前に述べたようにこのシステムはＥvolutionary アーキテクチャであるから，発展は段階的であり，いつまでに何をつくって，最終目標にいつ達成するという計画を明示することは困難に思われるが，大ざっぱにいえば，まず，従来型システムからの分散処理への連続的発展に並行し，設計理念確立のための検討，実験システムによる設計データの収集，基礎技術の開発を行ない，４～５年後に設計理念を確立して最終システムの構築をその後４～５年で行なうことになるであろう。

ともあれ，分散処理は基本アイデアは決して新しくはなく，ある種の新しい応用のようなはなやかさはないが，この際，ハードウェア，ソフトウェアのこれまでの蓄積された全技術を集約して，一定した設計理念のもとに，高効率，高信頼で，使い易く，構築しやすく，将来の技術改良に対応しやすく，高機能を盛りこむことのできる機能分散アーキテクチャの開発に国家的努力がなされることが必要であると考えられる。

3.2　システム・アーキテクチャ

ここでいうシステム・アーキテクチャとは，一般のアーキテクチャの分類の中でいえば，

- ＰＭＳレベルのアーキテクチャ
- ネットワーク・レベルのアーキテクチャ　　に対応する。

本節では，第５世代の機能分散アーキテクチャにおけるシステム・アーキテクチャの考え方を論じる。まず，3.2.1で全体の考え方を論じた後，3.2.2と3.2.3で上記２つのレベルを考察する。

3.2.1　基本的な考え方

(1)　システム・アーキテクチャへの要請

一般にシステム・アーキテクチャに求められるものとしては，次のような点があげられる。但，順番は重要度を表わすものではない。

① 各ユーザがシステム内にある資源及び情報を自由に（もちろん論理的に定められた範囲内で），かつ高速に利用できること。

② システム内の資源が効率的に使用されること。

③ 一部資源の故障が全体に影響を与えず，高信頼性と高可用性が保たれること。

④ セキュリティが確保され，故意又は誤りにより，情報が破壊されたり，資源が誤って使用されないこと。

⑤ システム構成に自由度があり，応用やユーザの要求にあった構成が簡単に取りうること。

⑥ 資源間通信のオーバヘッドが少ないこと。

⑦ インタフェースが整理され，新装置の接続等，将来の発展に容易に対応できること。

⑧ アルゴリズムに起因する応用側に存在する並列性を有効に利用して高速の処理を行なえること。

⑨ 管理ソフトウェアが簡単で，オーバヘッドが少ないこと。

⑩ 動作データの収集とそれによる管理方式の修正（チューニング）等，状況に対する適応機能があること。

(2) 機能分散型システム・アーキテクチャ

上記のようなシステム・アーキテクチャへの要請を満たすために，第5世代において基本的に考えるべきことには次のような点がある。

- 従来型の構成を部分的に修正していくような対応のし方ではなく，要請を満たすべく明確な思想を打出した新設計が必要である。
- システム・アーキテクチャは直接プログラムに影響する部分が少ないので互換性に対する制限は比較的少なく，この際新設計することによる損失も過大ではない。また，ある程度技術が成熟してきた現時点はその最良点であるともいえる。
- システム・アーキテクチャはある程度の統一（例えば，国家的な）があっても，技術の進歩を止めるようなことは比較的少なく，利用側の利点は大きい。
- 機能分散アーキテクチャの効果（3.1.1参照）と前記システム・アーキテクチャへの要請を比較検討するとき，機能分散型アーキテクチャが多くの要請を満たす可能性を持ったすぐれたものであることが理解できる。但，従来のようななんとはなしの分散処理では十分な効果を上げられないことも事実であって，目的と手段に明確ですっきりした設計思想をよく検討してかためることが絶対に必要である。

3.2.2 システムの構成方式

ここでは，センタにある高機能コンピュータにおける機能分散アーキテクチャについて検討する。

(1) 全般的考え方

設計にあたって必要な全体的考え方を以下に列挙する。

- 中央におかれる高機能コンピュータ・システムでは，3.1.1にあげた機能分散アーキテクチャの特徴(1)～(6)の全てを生かすことを目的として機能分散アーキテクチャが採用される。
- 機能分散のレベルは，特定のタスクを実行する　マシン　のレベルで行なわれる。したがって，特徴(1)の内，単一タスクの実行速度の向上はこのレベルの機能分散では達成されない。
- 機能マシンの単位は，3.4に詳述されるようなユーザ・サービス・マシン及びマシン・サービス・マシンであり，これにシステム管理や特殊入出力機器の制御を行なうコントロール・マシン群が加わる。この内コントロール・マシンに機能として存在することは明らかであるが，実体としては，かなりの部分が各サービス・マシン内に分散配置される可能性もある。（3.3参照）
- 機能分散効果の内，特に(2)，(3)，(5)を効果的に達成するために，コンポーネント・マシン間の論理的接続及び物理的接続の統一化，明確な設計思想の確立をはかる。
- 機能を分散することによる通信オーバヘッド（欠点A）をできるだけ軽減するために，各マシンにローカルな情報（プログラム，データ共）は，重復があっても，各マシン内に配置する。但，この点に関しては，分散を徹底するより共通の記憶制御プロセッサやファイル・プロセッサをつけて使用する方がよいとの考え方もある。設計時の検討によって定められるべきものであるが，これも設計思想の明確化が求められる例である。
- 各マシンは必要な情報のアクセスにおいてシステム内の情報位置によらず，統一的方法（例えば，主記憶空間として）で行なえること。
- マシン間の物理的接続は自由度が高く，統一されたインタフェースで高速の（１００ＭＨＺ以上）並列（～１００ビット）転送が可能であること。
- 利用側から見れば，マシンの接続，削除がオンラインで可能であり，システム装置の構成の認識や再構成は自動的に行なえること。いいかえれば，マシンは単にシステムに接続してスイッチを入れるだけでシステムに組込まれ動作する。
- 高信頼化のために，全てのマシン及び装置への接続パスは２重系以上の多重系とする。但，全ての接続が同一の高性能規格を満たす必要はない。
- システム管理マシン（3.4.5参照）にはシステム全装置への副インタフェース（最高性能でなく，例えば直列でもよいだろう）が用意される。
- システム動作状況のデータが常にオンラインで把握，蓄積され，システム構成や制御方式の自動適応化にオンラインで利用されること。

○　タスク・スケジュール，マシン・スケジュールは動作状態やシステム構成を把握した上で効率的に行なわれること。

○　機能マシン及びその相互関係が，明確に記述できること，また，そのための記述言語をシステム設計時から並行的に開発すること。

(2)　構成のイメージ

具体的なシステム構成は，基本となる設計理念によって大きく異なる。設計理念はそれ自身第5世代アーキテクチャの最大の研究課題の一つであるから，それを固定して議論を進めるのは困難である。大ざっぱな構成イメージは既に前年度報告書にシステムAとして述べられており，また，基本となる考え方は(1)に述べたので，ここではもう少し具体的な点を2.3コメントしておくにとどめる。また，個々のマシン内構成の一般的考え方については，3.4.1に述べる。

○　ユーザ・サービス・マシンとマシン・サービス・マシン及びコントロール・マシンの内のメディア制御マシンは統一された高レベル論理インタフェースにより接続される。

○　マシン間の物理的接続は，拡張の自由度，接続の容易さからバス構成となる。バスは単一では速度的に不十分であると考えられるから，マルチバス構成で，システム全体は，マシンの特性を考慮した階層構成となる。

○　情報アクセス方式は，主記憶アドレスをベースとするのが一方法である。

○　ファイル転送には別のバスを用意することも考えられる。

○　バスの形式はリングバスを提案するものもあるが，並列，階層接続が望ましい。

○　実装はＶＬＳＩ化を考慮し，コンパクトなものを用いる。（例えば，ユニバス等モジュール型ミニコンピュータの考え方を発展させる）

○　同種のユーザ・サービス・マシンを1まとめにして，一つのコントロール・マシンが他のマシンとインタフェースする形も考えられる。

以上から考えられるシステム構成の例として，分科会における検討の際，各委員の提出した構成図を図3－1～図3－5に示しておく。これらを比較するとき，基本的考え方の一致と具体化における多様性を読みとれるであろう。明確な設計理念の必要性を示す事実の一つである。

(3)　研究開発課題

(1)と(2)に述べたことからすぐれた機能分散型アーキテクチャを開発するのに必要な事項は自から明らかであるが，ここでは主なものをまとめておく，なお，各レベルに共通的なものについては，3.1.4を参照のこと。

①　マシン間インタフェースの開発

○論理インタフェース

○物理インタフェース

○情報の配置方式

②　システム構成方式

○追加，削除等の拡張または，個々のシステム構築のためのシステム再構成方式，例え

ば，常にオンラインでの接続，削除が可能であること等。

③　サービス・マシン制御方式（システム制御方式）

スケジュール・プロセッサの機能と実現，スケジュール方式オペレーティング・システムの機能分割，機能分散に適したオペレーティング・システムのコンセプトとその実現。

④　インプリメンテーション

マシン実装，高速度接続ハードウェア。

⑤　共通情報の管理方式

セキュリティ，ケーパビリティ制御

⑥　データ収集

設計データを現在のシステム又は実験システムにおいて収集すること。新アーキテクチャに組込む自動データ収集機構。

⑦　自動適応化機構

接続の自動適応化，並列構造の自動適応化，情報配置の自動適応化，資源割当等。

⑧　機能分散方式の記述

機能分散系の記述を行なう言語，言語記述からの実システム構築方式，言語は，機能モジュール自身の記述とモジュール間接続の記述ができ，それによる効果等のシミュレーション評価や最適設計ができるようなもの

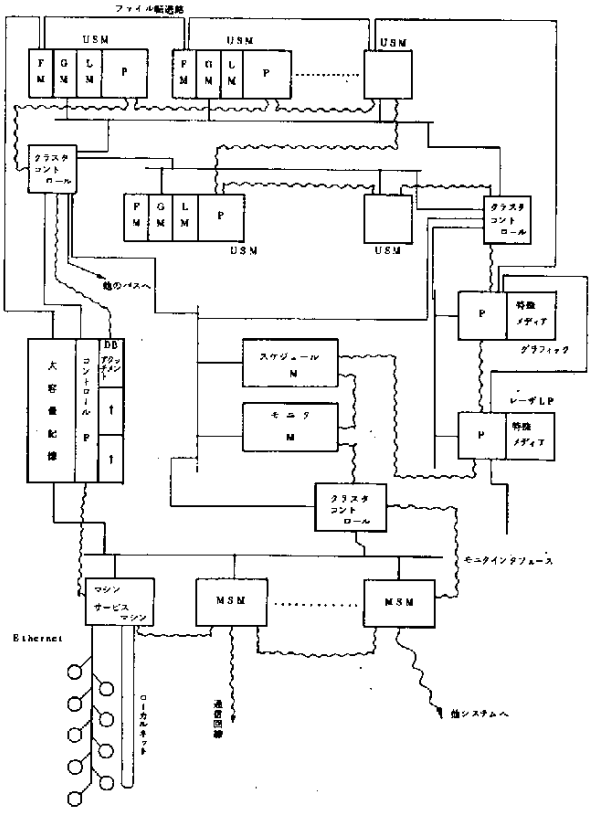


図３－１　システム構成例(1)

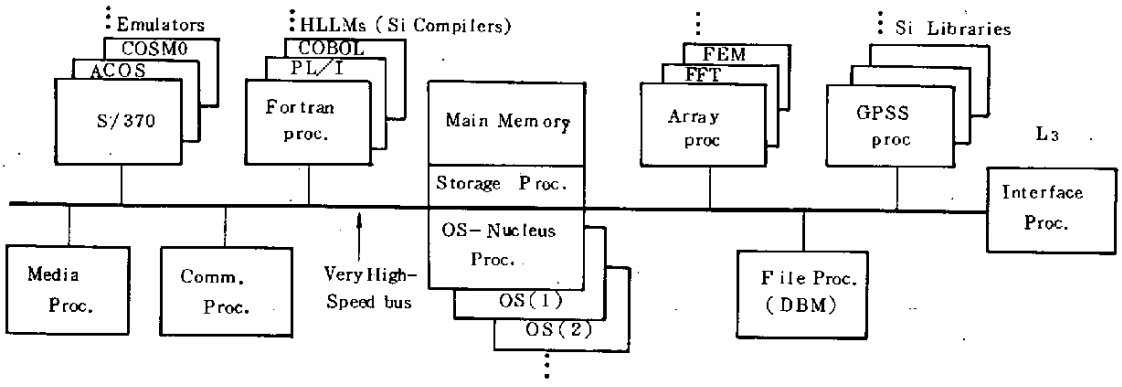


図３－２　システム構成例(2)

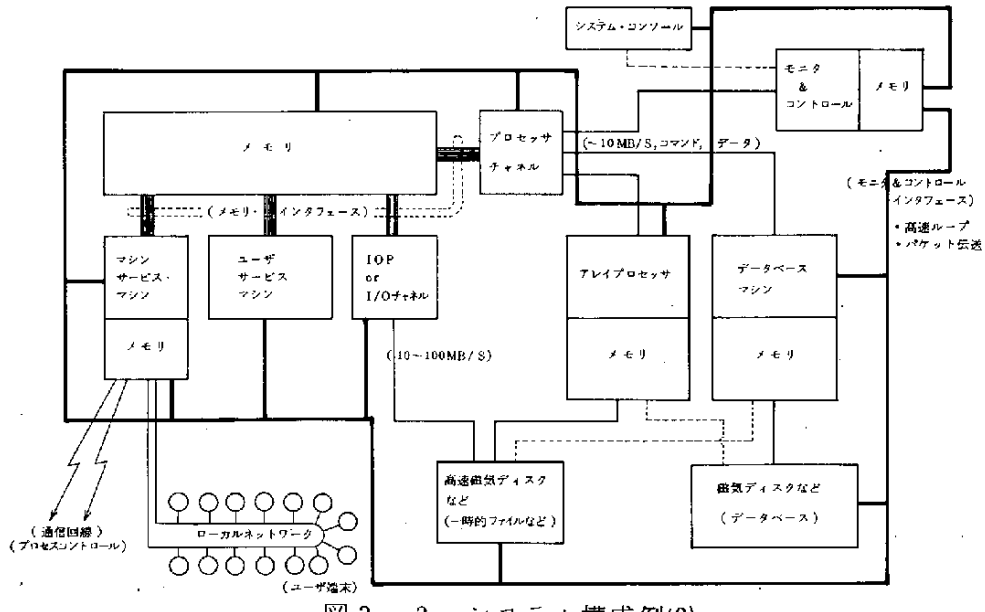


図３－３　システム構成例(3)

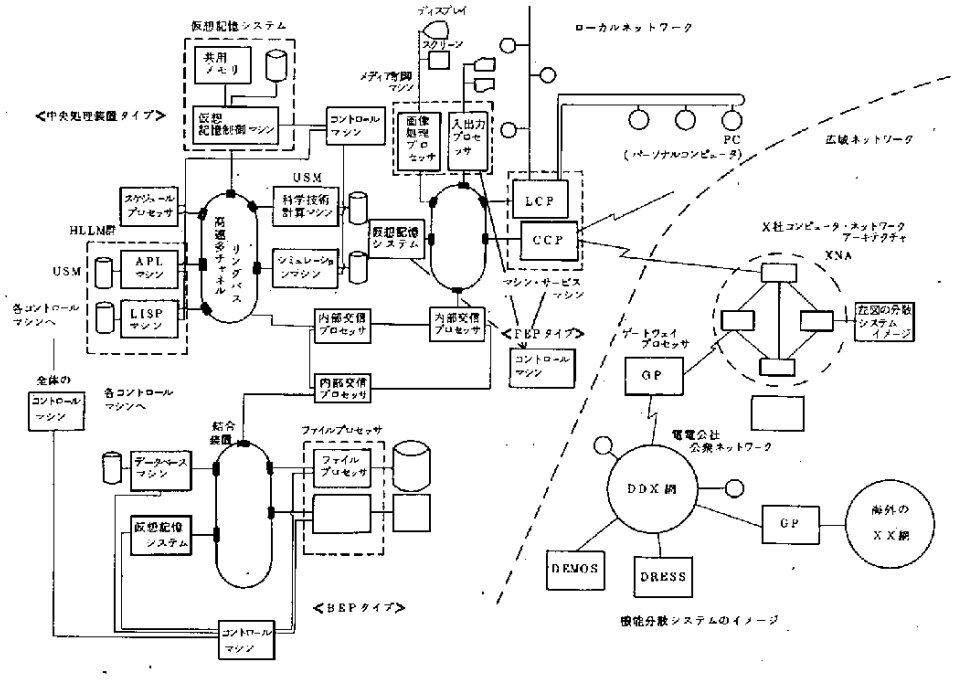


図３－４　システム構成例(4)

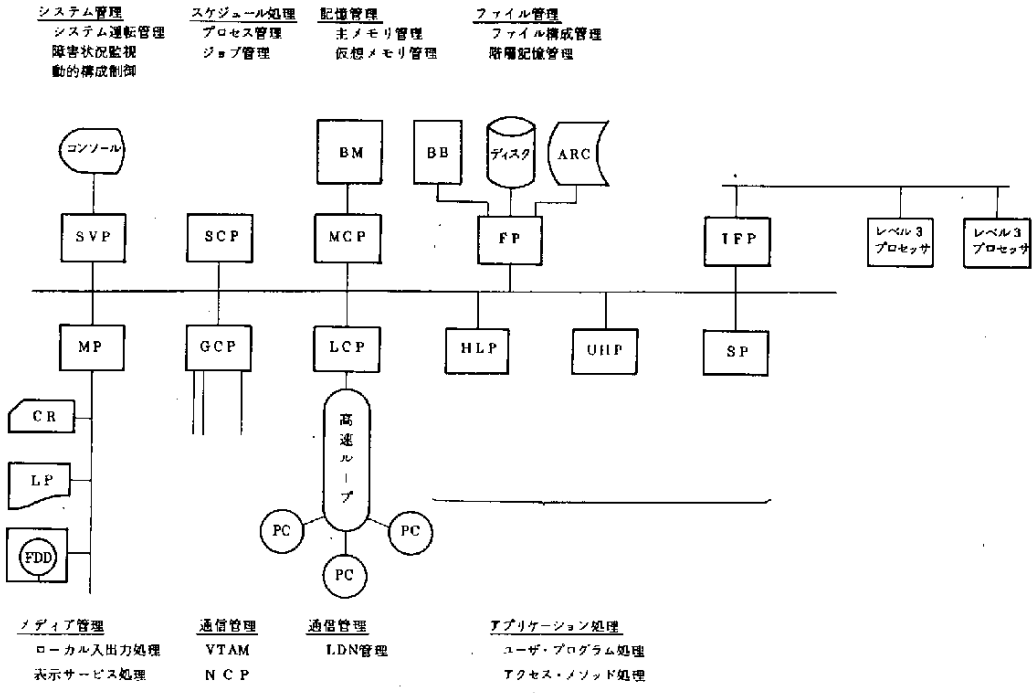


図３－５　システム構成例(5)

3.2.3　ネットワーク・アーキテクチャ

ネットワーク・アーキテクチャとは，地域的に離れて設置されているコンピュータ・システムを疎結合するために必要とされる論理構造と，通信のためのプロトコルを体系的に定めたものをいう。

ここでは，ネットワーク・アーキテクチャ作成のために必要なフィロソフィー，ネットワーク・アーキテクチャをベースに実現される１９９０年代のネットワークの機能と構成，およびネットワーク実現のための研究課題を述べる。

(1)　目的と必要とされるフィロソフィ

ネットワーク・アーキテクチャは，次のような目的のもとに開発する必要がある。

①　資源共用による経済化

- コンピュータ・システムを構成するリソース（各種マシンのハードウェア，ソフトウェア）を，そのシステムに属している複数の利用者だけでなく，他の異なるコンピュータ・システムのマシン，利用者からも共用利用できるようにする。

②　資源分散による信頼性の向上

- データベース等を複数のマシンに分散し，システム障害時の予備，災害時の危険分散を可能とする。

③　分散処理による価格性能比の向上

- 利用者に近い場所で一部の処理を行なうことにより，応答時間の短縮，通信コストの削減を図る。
- 各種ユーザ・サービス・マシンの活用により，全体として処理能力を向上させる。

④　高度な情報処理システム

- 複数のユーザ・サービス・マシンの共同利用，複数の異なるサービスを提供するコンピュータシステムとの結合により，高度な情報処理を可能とする。
- 地域的に離れて存在している各種のデータベースへ自由にアクセスすることを可能とする。

⑤ 開発・導入・運用・移行の容易化

- マシン種別，ネットワーク構成等に依存しないソフトウェア作成を可能とする。
- 新しいマシン，新しい端末，新しいサービスの導入を容易にする。

ボトム・アップに分散システムを設計するアプローチや，他のシステムを無視して，個別に特有のネットワークを構成するアプローチでは，上述の目的を達成することが非常に困難となる。このため，ネットワークを構成する各装置の役割りと，それらの間の通信に関するプロトコルを，ネットワーク・アーキテクチャとして体系的に定めることが重要となる。すなわち，物理的な構成要素からなるネットワークを，論理的なノードとリンクで表現し，すべてのノードに対して共通の論理的な階層を定め，各階層ごとにプロトコルを規定する必要がある。

ネットワーク・アーキテクチャの標準化は，ＩＳＯ，ＣＣＩＴＴをはじめとする機関や各コンピュータ・メーカにより進められているが，将来にわたって発展性，拡張性のあるネットワーク実現のためには，次の目標の達成が望まれる。

ⓐ 複数の異なるマシンに共通のプロトコルを提供する。

ⓑ 新データ網（パケット交換網等）と，マシン，端末等との間で，通信機能の適正な分担を行なうことができる。

ⓒ 公衆網と専用（私用）網の双方に適用できる。

ⓓ 統一された仮想端末プロトコルを提供する。

ⓔ 効率的に実現できるプロトコルである。

ⓕ 音声・画像等・各種マルチメディア用通信プロトコルを提供する。

ⓖ 本格的な分散処理プロトコルを提供する。

ⓗ 国際標準と整合する。

(2) ネットワーク・アーキテクチャを基にして実現されるネットワークの機能と構成のイメージ

ネットワークの機能と構成は，実現するサービス等により多種多様になる。このため，以下では，１９９０年代の重要または特徴的な事項を中心に述べる。

① ネットワークを実現する技術

ネットワーク・アーキテクチャの実現のためには，ⓐＶＬＳＩ技術ⓑ広域・高性能通信回線技術（光伝送，衛星通信等）ⓒプロトコルの標準化等が重要であるが，１９９０年代には，さらにⓓ各種サービスをサポートするネットワーク・オペレーティング・システム（ネット

ワークＯＳ）の実現技術が重要となる。

ネットワークＯＳは，ネットワーク利用の容易化，負荷の分散，リソースの共用等を達成するために必要なソフトウェアとプロトコルの集りである。このネットワークＯＳは，１つのマシンに集中的に配置されるのではなく，ネットワークを構成する各装置（ユーザ・サービス・マシン，マシン・サービス・マシン，パーソナル・コンピュータ等）に分散配置され，ネットワーク全体として実現される点が通常のＯＳとは異なる。

ネットワークＯＳの主な機能は，次の通りである。

(a) ネットワーク・ジョブ管理機能

(b) ネットワーク・データ管理機能

　① ネットワーク・ファイル管理機能

　② 分散データベース管理機能

(c) データ移送機能

(d) プロセス間通信機能

(e) ネットワーク制御機能

② 構成法の分類とイメージ

ネットワークの構成は，ネットワークの地域的な広がりの観点から，ローカル・ネットワークと広域ネットワークに分類できる。１９９０年代の構成イメージは，次のようになろう。

(a) パーソナル・コンピュータが発達し，これらを結合したローカル・ネットワークが各研究機関内，企業内，地域社会内，家庭内等で実現される。

(b) 高速の通信回線によるリソース共用網が広域ネットワークとして実現される。また，ローカル・ネットワークは広域ネットワークと結合される。

(c) 利用者は，通常，ローカル・ネットワーク内のリソースを使用するだけでよい。しかし，他のリソースを必要とする場合には，ローカル・ネットワークのゲートウェイを通して広域ネットワークのリソースにアクセスできる。

(d) 広域公衆ネットワークには，網接続装置を介して，電話をはじめデータ・テレフォン，ファクシミリ，テレメータ，テレコントローラ，ビデオ端末，ＯＣＲ端末等，各種装置を端末できる。

(e) 広域ネットワークは，他の国の広域ネットワークと接続される。

(f) ローカル・ネットワーク内のデータ伝送速度は１０.０～５００Ｍビット／秒，広域ネットワーク内のデータ伝送速度は加入者系で１～１０Ｍビット／秒程度になる。

③ ローカルネットワーク

ローカルネットワークは，通常，公衆網を利用せず，数ｍから２～３０Km以内に存在するパーソナル・コンピュータ，端末等を結合する。

(a) 機能イメージ

特徴的な機能は，次の通りである。

- ノード間の相互接続パターン（ネットワーク・トポロジー）に制約を付すことにより，効率のよい伝送方式を実現できる。
- 特定の利用目的に依存した独自の通信プロトコルを持つことができる。
- 1：n（同報通信），n：mで通信ができる。
- 機器の増設や撤去に容易に対処できると共に，業務の拡大，多様化に対しての融通性がある。
- 保守，維持費が少ない構成をとる。
- 障害の局所化や，障害時にネットワークの再構成を自動化できる。
- 各種のデータ通信に加えて，音声通信，ＦＡＸ通信，画像通信（静止画，動画）等を統合して扱うことができる。（各種メディアの統合通信）
- ローカル・ネットワークを使い易くするオペレーテイング・システムが準備されている。
- 広域ネットワークとの接続機能を持つ。
- プロセス制御用の場合は，データ収集だけでなく，優先割込み処理や機器の起動，停止，制御等のリアルタイム処理を行なうことができる。

(b) 構成イメージ

ネットワーク・トポロジーとしては ①スター ②ループ ③バス等が考えられる。なかでも，回線コスト面で有利なループやバスが多く採用されるが，これらの場合には，スターと異なり中央ノードを欠いているため，ネットワーク制御構造が必要になる。ネットワーク制御構造の例としては，次のものが考えられる。

- デイジー・チエイン
- コントロール・トークン（メッセージ多重方式）〔ＤＣＳ：Distributed Camputing System等〕
- メッセージ・スロット（チャネル／パケット多重方式）〔Cambridge Net等〕
- レジスタ挿入方式〔ＤＬＣＮ：Distributed Loop Computing Network等〕
- コンテンション・コントロール〔ＥＴＨＥＲＮＥＴ等〕

１９９０年代には，電磁誘導雑音に影響されない，高速である，制御が簡単である，拡張性や融通性がある，経済的に実現できる，等の理由から，光ループ網が普及すると考えられる。ループでは，１か所に障害が生じるとシステム・ダウンにつながる恐れがあるが，ループの２重化や，マイクロプロセッサにより各ノードに障害対策機能を持たせることにより，克服

できる。また，先に述べた各種メディアの統合通信では，チャネル多重方式が主流になると思われる。

一方，同軸ケーブルを利用したＥＴＨＥＲネットワークは，双方向に情報を送ることができ，経済性，拡張性，信頼性に優れているため，オフィス・オートメーションをはじめとする各種のシステムに適用される。また，電子構内交換機（ＥＰＡＢＸ）を利用したローカルネットワークも出現しよう。

④　広域ネットワーク

個々の網が統合されたデイジタル総合サービス網（ＩＳＤＮ：Integrated Services Digital Network）となり，この網を利用して多数のコンピュータや端末等が接続される。そして，通信処理と情報処理が融合したネットワーク・ユーティリティが形成される。

(a)　機能イメージ

特徴的な機能は，次の通りである。

- 電話，電信，テレックス，データ，ファクシミリ，画像通信等により，各種通信メディアを利用した多様なサービスを総合的，経済的に提供する。また，リソース共用網としての役割りを果す。
- ネットワークは，伝送機能の他，蓄積，変換，交換，処理機能を持つ。
- ネットワークには，標準化されたプロトコルが用意される。また，方式やプロトコルが異なる網との接続を可能とするゲートウェイ機能を持つ。これらにより，異機種マシンや異なるサービス間の通信が可能になる。
- 高度な機密保護機能が実現される。
- 国際ネットワークの運用や管理では，各国の権限が平等な分散制御方式となる。
- 料金即知や，利用者に対するオペレータ・ガイダンス機能を持つ。
- 高い信頼性と，障害の影響を局所化する機能を持つ。

(b)　構成イメージ

- ネットワークは，国内の電話網，電信網，専用線網，ＤＤＸ網，公衆データ網，公衆ファクシミリ網，私設線網や，国外の広域ネットワークを有機的に結合して構成され，さらに，ネットワークのデジタル化が進展する。　また各種メディア，各種サービスを同一ネットワークで実現する部分が増加する。
- 光ファイバ・ケーブルによる広帯域高速デジタル伝送が導入される。
- 基幹ノード間は，光ファイバ・ケーブルの他，既存の同軸ケーブル，マイクロウェーブ，あるいは衛生通信により接続され，高速デイジタル伝送を行なう。
- ネットワーク全体の運用状態の監視，障害の発見と診断，新通信用プログラムのリモート・ロード等を一元的に行なう網管理センタが必要に応じて設置される。
- 国内では，共通線信号方式が導入される。

(3) 現状と問題点

ネットワーク・ユーテイリテイ形成のために必要とされる基本技術の基礎は固まってきている。しかし，より高度な利用を行なう時には，(4)で述べるような研究開発課題が多く残されている。

理想のゴールから見れば，現在はスタート地点に立つために必要とされる道具だてと，スタート前のウォーミング・アップがすんだ段階にあるといって過言ではない。

たとえば，ネットワークアーキテクチャの標準化は，ＩＳＯやＣＣＩＴＴ等で進められており，国内でもＤＣＮＡを開発しているが，上位層に向けての国際標準の早期勧告と，その実施が望まれる。また，広域公衆デイジタル網も，ＤＤＸ網としてサービスを開始したばかりであり，サービス内容や伝送速度（最大４８Ｋビット／秒），接続できる装置等の制約が大きい。また，既存網の統合や異なる網の接続は，その計画といくつかの試みがスタートした段階である。伝送技術では，光ケーブルや衛星通信方式の実験や一部の商用化が進められているが，より高速で安価な回線を実現するためには，今後とも多大の努力を要する。ネットワーク・ユーティリティを支えるネットワークＯＳも，限定した利用範囲内で実用化が進められているのが実情である。通信処理と情報処理は，次第に区別がつかなくなり，現在は混沌とした状態にある。

なお，参考としてネットワーク・アーキテクチャの標準化活動を表３－１に示す。

(4) 研究開発課題

① ネットワーク・アーキテクチャの標準化

(1)で述べたように，ネットワークを効率よく体系的に実現するためには，ネットワーク機能の階層化とプロトコルの国際標準化が不可欠である。

② プロトコルの記述・作成・検証技術

ネットワークの高度化，広域化，多様化に伴ない，規定されるプロトコルも増加し，複雑なものとなる。このため，プロトコルを論理矛盾なく記述する方法や，インプリメントのために切り出されたプロトコルの検証を行なう技術がより重要となる。また，プロトコルを記述したドキュメントは，記述内容がわかり易いこと，記述内容が厳密であること，プロトコルのインプリメントに役立つこと，等の条件を満たしている必要がある。更に，ドキュメント管理システムの開発も重要である。たとえば，プロトコルの修正，追加に対して，従来プロトコルとの矛盾性を自動的にチェックしながら新しいドキュメントを自動生成する技術や，インプリメント条件を与えることにより，必要とされる検証されたプロトコルを自動生成する技術等の実現が望まれる。

③ ネットワーク・オペレーティング・システムの実現

ネットワーク利用の容易化，ファイルやデータベース等のリソースの共同利用等を達成す

るためには，(2)①で述べたようにネットワーク・オペレーティング・システムの実現技術が重要となる。また，このためには，高位レベルのネットワーク・アーキテクチャの標準化も必要になる。

④　網間接続技術

電話網とＤＤＸパケット網等，異なる網間を接続し，通信範囲の拡大，サービスや機能の拡充を図るためには，網間接続技術の開発が必要である。また，これを実現するためには，プロトコルの変換技術の確立が必要である。

⑤　マルチ・メデイア処理

今後は，音声，データ，ファクシミリ，ビデオ等多様化する通信サービスの総合化が望まれる。このためには，マルチメデイア用プロトコルの標準化と，伝送速度の異なる信号の多重，挿入，分離が容易なデイジタル網を実現する必要である。更にパケットで音声や画像を伝送する方式の実用化も急がれる。

⑥　機密保護機構のネットワークへの導入

履歴，犯歴，病歴，信用等の個人のプライバシーに関する情報や，企業内，国家の機密情報が伝送路上から盗まれ暴露される危険性や特殊装置をネットワークに接続することによる各種ファイルの破壊や故意の修正が行なわれる危険性が高まる。このため，暗号処理機能や個人識別機能等の機密保護機構をネットワークの中で実現する必要がある。

⑦　ＶＬＳＩ技術

ＶＬＳＩ技術は，電子交換機やネットワークの各ノードを構成するマシン等の論理素子や記憶素子の経済化，小型化，信頼性の向上に必須の技術である。また，伝送速度が向上すると，これに追従できるインタフェースが必要となる。このため，伝送インタフェースのＬＳＩ／ＶＬＳＩ化が不可欠である。

⑧　光ファイバ通信技術

光ファイバ通信は，

ⓐ　従来のケーブルと比較して低損失化をはかることができる。

ⓑ　広帯域伝送特性を有する。

ⓒ　電磁気的妨害に強い。

ⓓ　光ファイバ相互間の漏話は通常は問題にならない。

ⓔ　原料が豊富にある。

等の理由から，ＩＳＤＮを経済的かつ効率的に実現するために有望である。

このため，発光素子，光分波器等の光部品の開発，光ファイバケーブルの連続製造法，接続技術等の実用化が待たれる。また，

ⓐ　光ファイバの切断接続には高精度の技術が必要とされる。

ⓑ　光ファイバに外圧が加わると伝送特性が変化する恐れがある。

等の理由から，光伝送路の施設工法の開発も重要である。

⑨　衛星通信方式

衛星の持つマルチプル・アクセスやブロードキャースト転送が可能であるという特徴と，パケット交換機能を組み合わせることにより，同報通信，広域通信，柔軟な網構成，高速伝送等が可能になる。また，地上災害が発生した際の通信路を確保するためにも効果がある。地上系と衛星系の最適な機能分担によるネットワーク構成の研究，衛星に適したデータ伝送技術，衛星を利用したマルチメデイア通信技術等の実用化が急がれる。

以上述べた技術的な研究開発課題のほかに法律の改正や制定も重要である。すなわち，プライバシー保護法やデータ保護法，ソフトウェアの著作権に関する法律の制定，回線利用に関する法律の改正，国際ネットワークの運用をサポートする国際法の制定等も，ネットワーク・ユーティリティ推進のために必要とされる。

なお，すでに電電公社は，ＩＳＤＮの実現計画を発表している。電電公社は，各メーカの協力のもとに，利用者の声を十分に反映させながら，１９９０年代にふさわしい広域公衆ネットワークを実現していく必要がある。

表３－１　代表的なネットワーク・アーキテクチャ標準化活動

No.	分類	組織名	主な目的	組織と主な検討内容		備考
1	国	ISO	システムの使用者の立場から，リンクレベルの伝送制御手順，ユーザレベルの通信手順等の標準化を行なう。	TC97／SC 6	・ベーシック・モードの伝送制御手順 ・ハイレベル・データリンク制御手順（HDLC） ・システム・アーキテクチャとエンド・ツー・エンド・プロトコル*1主として，ネットワーク・レイヤ以下のサービスとプロトコル ・複数回線手順*2	*1 SC 16は高位レベルを担当（現在はSC16OSIが中心） *2 CCITTからの検討依頼
				TC97／SC16	・開放形システム・アーキテクチャモデル（OSI：OPen Systems） ・高位レベルのサービスとプロトコル*3	*3 各国からは，仮想端末，ファイル転送RJE等についての積極的な提案が多い
2	外	CCITT	通信網提供者の立場から，主として，公衆データ通信網に関する標準化を行なう。	SG Ⅶ	・回線交換網 －X20，X21等 ・パケット交換網*4 －パケット端末インタフェース X25，調歩端末インタフェース X3，X28，X29等 －網間プロトコルとマルチリンク手順，*5 X75	*4 1SOモデルとの調整が必要 *5 SC6へ検討依頼
				SG Ⅷ	・テレテックス，ファクミリに関するサービスとプロトコル	ISDNに関する事項が今後積極的にとりあげられよう
3	国	データ通信網アーキテクチャ（DCNA）共同研究	複数の計算機機種に共通のプロトコルを提供する。	電電公社，日本電気，日立，富士通，沖電気の4社から構成	・DCNA －（第1版） メッセージ転送に適用できる基本内容を規定した。 －（第2版） 第1版の内容に，ファイル転送・アクセス・プロトコル等の追加規定した。 －（第3版） 第2版の内容に，ジョブ転送プロトコル・データベース・アクセス・プロトコルの新規追加ネットワーク管理・ファイル管理機能の拡充等を行った。	
4	内	郵政省	国内標準プロトコルを設定する。	学識経験者，通信事業者，コンピュータ・メーカ等から広く意見を求めてとりまとめる。	・CCNP（Computer Communication Network Protocol） －リンクレベル：プロトコル，パケットレベル・プロトコル，プロセス間通信サブレベル・プロトコル等のとりまとめ*6	*6 一部について，「コンピュータ・コミュニケーション・ネットワークの推奨通信方式」としてS55年11月に告示

3.3　システム制御方式

3.3.1　基本的な考え方

機能分散方式が将来の計算機システムにとって重要な技術の一つになることは前節までに述べられたとおりである。前年度報告書で提案した＂System A＂は機能分散システムの考え方に基づいている。

機能分散システムにおいて常に問題となることは，①機能をどのレベルで分割し，分散処理するか，②全体の制御をどのように行うかである。3.4では機能分散の考え方に従った標準的な計算機システムの構成要素であるコンポネント・マシンについて述べている。個々の構成要素については，ハードウエアの開発はもとより，関連するソフトウエア開発の生産性が，従来の集中制御方式の計算機システムの場合よりもずっと高くなると予想され，望ましい方向に向いているといえる。しかし，複数台のコンポネント・マシンをユーザの要望に従って組合せ，応用にそくした一種のユーザ仮想マシンとして，価格性能比の観点からも効率よく稼動させることは簡単なことではない。この機能は従来の集中型計算機システムではオペレーティング・システム（ＯＳ）が果していたわけであるが，＂System A＂のような場合には，ＯＳの機能自身も分割され，コンポネント・マシンあるいはレベル間に分散されるので，今までに見られない新らしい考え方に基づいた全体制御が必要になる。ここでは，このような制御機能を総称して「システム制御」と呼んでいる。ここで注意すべきことは機能分散方式の考え方如何でシステム制御の実現のイメージが異なってくることである。したがって，まずコンポネント・マシンが備える機能と相互のインタフエース機能を整理することが重要な課題となる。

システム制御方式を考える上で重要な点は，次の点である。

(1)　モジュール化された簡潔な構成となること。

(2)　コンポネント・マシン間での通信のオーバ・ヘッドが過度とならないこと。

従来の大規模でかつ複雑なＯＳではなく，モジュール化された簡潔なシステム制御機構であることは第５世代の計算機システムでは，必須条件である。機能分散方式による仮想マシンのインタフエースに基づいてシステムを設計することにより，モジュール化された簡潔なシステム制御機構が実現する可能性が大いにある。

コンポネント間での通信オーバヘッドに関しては，第５世代コンピュータでは単にオーバヘッドの絶対量で評価すべきではなく，システム制御機構の簡潔さを第一として，相対的に評価すべきであろう。ソフトウエアの生産性，信頼性を重視し，かつハードウエアの性能を十分に引きだせるＶＬＳＩで実現される第５世代の計算機システムでは，このような立場が当然なものとなろう。

従来から研究されてきた階層化ＯＳの核（kernel）の考え方は，各コンポネント・マシンがローカルに用意すべきシステム制御機能の核の部分の設計の参考になると考えられる。

グローバルな通信ネットワークを含めた統一的なシステム（ネットワーク・オペレーティング・システム）の制御方式，論理的な機能分散構造とコンポネント・マシンの階層構造とのマッピング，コンポネント・マシン間の通信方式，セキュリティ制御，診断等のコンポネント・マシン群を管理する機能の分散または集中型制御の選択等は今後の研究課題である。

3.3.2 制御機能の分散

(1) ＯＳの機能への影響

現在の単一マシンで総ての機能を行なう形のＯＳは，機能別に各マシン対応に分解され，分解された後のＯＳの中で，さらにそれぞれの核になる部分，マシン間インタフェース，ユーザ・インタフエースなどの階層を作り，それぞれの階層の中で機能ごとのモジュール構造をとるような構成に変化して行くことが考えられる。

構造面では，このような機能分割化，階層化，モジュール化されたＯＳになると考えられるが，機能面では，次のような点が，現在のＯＳから変化する重要な点となるであろう。

(a) ＪＯＢの実行管理の方法

(b) マシン間通信の方法

(c) 共通主記憶へのアクセスの方法

(d) 障害処理の方法

(e) モジュール化の方法

ジョブの実行管理については，分散処理の各マシンで行なう部分と，共通に集中管理する部分とがあると考えられる。この共通に集中管理する部分はＯＳ核として，独立に存在するＯＳ核マシンで処理される形を考えることができる。

マシン間通信については，例えば，メッセージ型のデータを高速バスで転送し，大量のデータは主記憶経由で転送するなどの方法をとるなどの手段が必要となり，新しいマシン間の同期化機能や・メッセージ型データ転送用のプロトコルなどが必要となる。

共通主記憶は，一般的に仮想アドレスでアクセスされると考えられるが，アクセスの競合による性能低下を防止するための機能が必要となる。具体的には，各マシンに固有主記憶をもたせ，通常はこの固有主記憶を用いて処理し，共通のジョブ管理の仕事や，マシン間の同期と通信の仕事などは共通主記憶を使うなどの方法が必要となろう。

障害処理については，基本的には障害の発生したマシンを切離し，その状態で，例えばシステム管理マシンなどによる診断を行なうことになろう。障害を発生したマシンの誤動作の影響が，他のマシンへ伝播しないようにするハードウエア・ソフトウエアを含めた機能が重要な課題となろう。

モジュール化については，基本的にＯＳの単純化，階層化，拡張可能化などのために，あらかじめインタフエースを明確にしてモジュール化を図る必要がある。このようなインタフエースの

設定とモジュール化の方法は，今後の重要な研究課題となろう。

(2) 制御プログラム

① スーパバイザ

機能分散アーキテクチャでは，各マシンやその中のプロセスが相互に通信する機能を協力にサポートすることが必要である。この要請を満たすためには，従来のように同一マシン内はスーパバイザが管理し，他のシステムとの通信にはＯＳのデータ管理機能を利用すると言った2段式ではなく，プロセスがどこに配置されていても，スーパバイザが統一された形で処理する方法が好ましい。いずれにしろプロセス間，モジュール間の通信方式はＯＳとしてもしっかりした概念を用意して，対応するハードウエア方式に反映するようにしなければならない。

また，プロセス間通信に関連した障害回復機能も分散アーキテクチャのＯＳに対する重要な課題である。

② データ管理

システム間でファイルを授受し合うためには，分散ファイルのアクセス機構が必要であり，標準化されたファイル構造とアクセス形式が組込まれねばならない。そのためには，ファイル・カタログの充実が不可欠であり，システム間に渡ったカタログの参照，データ記述の標準化などが必要である。

③ ジョブ管理

分散された機能マシン内のプロセスを協調させて処理を行なうためには，それぞれのプロセスを定義し，いくつかのプロセスをまとめて1つのジョブを定義し，制御する機構が必要である。このことは，各システムのジョブ管理相互にまたがった管理が必要であることを意味し，ローカルなジョブ管理システムが相互に連絡をし合う仕組みがいる。

協調して動作するプロセスが分散し，複雑に組合さると，資源管理が難しくデッドロックを起しやすくなる。網の規模が大きくなると，網内資源をデッドロックなしに獲得したりまたは資源競合でデッドロックが発生していることを検出したりすることは，大変オーバヘッドのかかることであり，単純で実用的な新しい方法の開発が望まれる。そのためには，理論的あるいは実験的なものを含めて，多くの研究が必要である。

機密保護については，個々の機能マシン内，システム全体，他システム間などいろいろなレベルで考える必要があり，統一的な新しい概念が求められる。

④ 通信管理

通信管理のうち，実際のデータ転送にかかわる部分は，通信用マシン・サービス・マシンにより処理されることになる。コントロール・マシンとのインタフエースは，相当高いレベルのものとなるはずであるが，その設定には各方面からの十分な考慮が必要である。

3.3.3 オペレーティング・システムの分散形態

これまでのオペレーティング・システムは，一般に集中処理向きに設計され，一つの大きな計算機能や記憶機能を多数のユーザで効率よく共用することがその主眼となっていた。しかし，機能分散アーキテクチャではその基本コンセプトが異なるためにオペレーティング・システムは基本概念からして大きな変革を求められることになる。本節では機能分散アーキテクチャにおけるオペレーティング・システムの分散形態，即ち，全体構造について考察する。

(1) オペレーティング・システムの機能と機能分散

まず，オペレーティング・システムの機能をその機能の性質という立場で分類すると次の5種に分けることができる。

a. 仮想化

自分の接続されているマシン間，あるいはユーザとマシン間のインタフェースとして働き，背後にあるものを仮想化して，一つの定まった論理機能を与える。

b. システム資源の管理

システム内の資源が効率的に利用されるように，競合関係をモニタし，スケジューリングする機能である。通常，ジョブ管理，記憶管理，タスク管理等と呼ばれている機能である。

c'. 並列動作管理

複数のモジュール間の並列動作の同期管理を行なう機能である。対象となるモジュールの単位は低レベルのものから，タスク，ジョブ等の規模の大きいものまでいろいろ考えられる。

d. システム構成管理

システムを構成する（各種レベルの）モジュールの追加，削除，にもとづくシステム全体の構成を自動的に管理する。これらの操作において，当該部分以外の停止や，人手による複雑な手続きがあってはならない。

e. システム動作状況データの収集，自動チューニング

システムの動作状況，（例えば，低レベルでは命令の使用頻度等，高レベルでは特定のユーザ・サービス・マシンの使用率，特定の通信系の使用状況等）をモニタし，データを収集し，必要ならそのデータにもとづく調整を自動的に行なう。

次に，3.1に述べられている機能分散マーキテクチャの特質を生かして上記a～eの機能をうまく分散する上で基本的に考えなければならないことをあげると以下のようになる。

i) 機能モジュール，通信系等の内ローカルにすむものはできるだけその独立性を保つこと。

これによってオペレーティング・システムのモジュール化が可能となり，システムが巨大化してもオペレーティング・システムの複雑さをあまり増加させないですませることができる。いいかえれば，極めて多数の構成モジュールを含む機能分散システムでは，オペレーティング・システムの総量は従来同様の割合で増加するとしても，個々の機能単位に分散されたオペレー

ティング・システムは専用化され，論理の深さも浅くてすむのでむしろ小型化されるということである。即ち，3.1.1に述べた機能分散の利点はハードウエアのみならず，ソフトウエアにも適用される。

ii) ユーザからはシステム全体が一つの論理構造として仮想化されること，各マシンについても同様である。

このためにはハードウエア・モジュール間のみならず，オペレーティング・システムを構成するモジュール間においても，秀れた論理モデルにもとづき，インタフエースを明確に規定し，標準化することが重要である。これが実現されれば，新機能の追加，機能そのものや実現方式の改良等システムの成長性に大きく寄与することができる。しかしながら，場合によると，インタフエースの固定化のために追加が困難な新機能が生じることもありうるから，論理モデルの設定，インタフエースの設計には，将来を見こした万全の配慮が必要であることは，ハードウエア・モジュールの場合と全く同様である。なお，現在の技術レベルからすれば，今後数年の研究実験によって，第5世代においてそうした設計を実現することは可能であると考えられる。

iii) 資源利用率の向上のための機能は，各部分の必要度に応じて処理されること。

従来のオペレーティング・システムのようにあらゆる部分を総合してシステム資源の利用効率を最大にするような管理機能を持たせることは機能分散マーキテクチャーでは困難であるし，また，ハードウエア・コストの低減からその必要性もへっている。したがってこの種の機能の実現において重要なことは，その必要度に即して機能レベルを設定することである。即ち，ローカルなものや，低レベルのもの（3.1.2で述べた3つのレベルではコンポーネント・マシン内に相当）に対しては，従来同様利用率や並列処理度を最大限に上げる制御をすべきであるし，グローバルなもの（マシン間，システム間）では，i)，ii)の特徴をそこなわない範囲におさえることが正しい行き方であろう。

(2) オペレーティング・システムの基本構成

(1)での考察から機能分散マーキテクチャーにおけるオペレーティング・システムの構成は，

- 機能，構造，必要性に一致した多階層構成
- 秀れた論理モデルにもとづき，インタフエースを明確化したモジュール構成
- できるだけローカルな処理ができるようにした分散構成

という特徴が導かれる。

ここで，機能，構造，必要性に一致した多階層構成という意味をもう少し具体的に考えてみよう。今，このような基本構成にもとづき，先にあげたオペレーティング・システムの機能を考えるとき，aのような機能は，その性質上ローカル性が強いから，うまく分散できる可能性が大きいが，bのシステム資源の管理のようなものは，グローバルな処理を前提としているわけである

から，分散には必ずしもそぐわないし，無理な分散を行なえば，オーバヘッドが過大になる恐れが多分に存在する。このような場合の解決法は，それ自身研究課題であるが，次のようにシステムの構造をオペレーティング・システムの構造に反映させるのが現実的であると思われる。

即ち，資源管理がグローバルであるといっても，システム全体をあらゆる深さにおいてグローバルに管理しなければならないという理由はないから，関連する範囲ごとに資源管理を行ない，その上のレベルの管理には別の管理システムを用意するというわけである。もっと具体的にいうと，相互に互換性のあるユーザ・サービスマシン群では，これを一つにまとめて管理する専用のコントロール・マシンを用意し，他のユーザ・サービス・マシンやマシン・サービス・マシンからは全体が一つのユーザサービス・マシンに見えるようにするのである。また，あるユーザ・サービス・マシンがマルチ・プロセッサ構成ならその中の一つのプロセッサがそのマルチ・プロセッサのコントロール・プロセッサであるという構成がとられるであろう。なお，このようなコントロール・プロセッサの構成や性格については3.3.4で述べられる。

(3) オペレーティング・システムの階層構造

これまでの議論からわかるように，機能分散の各レベルに対応して，オペレーティング・システムにも次の3つのレベルが考られる。

① コンポーネント・コンピュータ間制御

あまりいい用語ではないが，要するに，ユーザ・サイドのパーソナル・コンピュータとセンタ・コンピュータ間，あるいはセンタ・コンピュータ間同志の制御を行なうものを指す，機能としてはaが主であり，d，eがこれにつぐ。それぞれのシステムは一応独立したコンピュータであるからb，cの並列動作に関する管理機能の重要度は小さい。ここの機能はネットワーク・マーキテクチャとの関連が強く，制御方式の決定においては，両者の融合が重要である。

② コンポーネント・マシン間制御

センタ・コンピュータにおけるユーザ・サービス・マシン及びマシン・サービス・マシン間の制御である。重要機能はaとbであるが，c，d，eにも十分配慮が必要である。なお，ここでいうユーザ・サービス・マシンやマシン・サービス・マシンは(2)で述べたように1プロセッサとは限らない。マルチ・マシン（プロセッサ）の場合には，そのコントロール・マシンによって代表される。いずれにしろ，マシン相互の通信機能を強力にサポートすることが必要である。

③ 単独ユーザ（又はマシン）サービス・マシン内の制御

各サービス・マシンが単一プロセッサで構成される場合，多くの機能が①，②へ移されるので，このレベルのオペレーティング・システムは簡略化される。しかし，第5世紀アーキテクチャにおけるサービス・マシンはその内部でも機能分散が行なわれ，マルチ・プロセッサ構成がとられると思われる，したがって，各コンポーネント・プロセッサを協調させて並列処理す

るためのオペレーティング・システムが重要である。このレベルでは個々の応用の特徴が強く出るため，①，②のように統一的に取扱うことは困難であるし，専用化の効果を生かすためにはそうすることは得策でもないと考えられるから，ケース・バイ・ケースの要素が強くなると思われる。共通的にいえることは，cの機能並列処理の重要性が他と比べられないほど高くなることで，これを効率的にサポートするための共通の処理方式の開発が重要である。

以上から，システム全体のオペレーティング・システムの構成を図にまとめると図３－６のようになる。

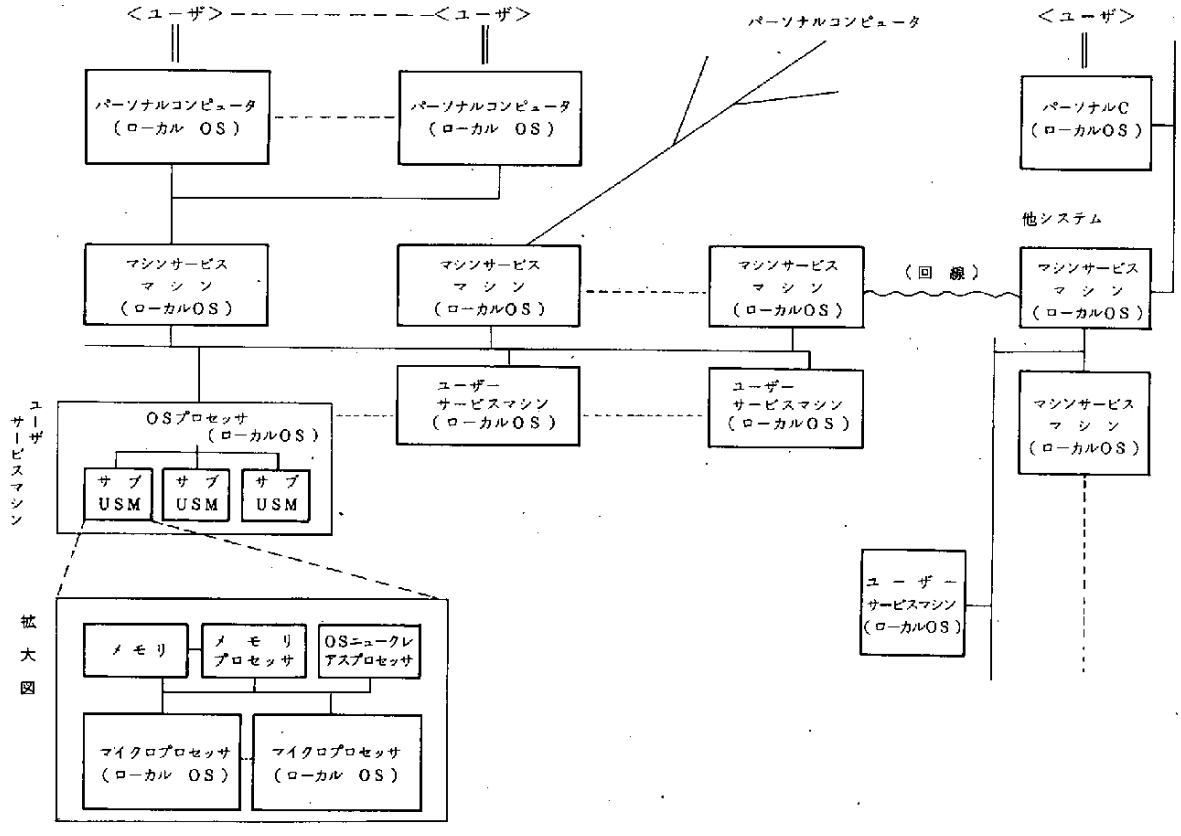


図３－６　オペレーティングシステムの分散形態

(4) オペレーティング・システムの分散における諸問題

ここでは機能分散アーキテクチャにおけるオペレーティング・システムの分散に関して検討すべき問題点をあげておく。但，順序は問題の重要性を示すものではない。

a. モジュール間の通信形態

機能の分散されたシステムでは，分散された各機能（モジュール）が相互に通信するための機構の協力なサポート（ハードウェア，ソフトウェアとも）が重要であることは既に述べた通りである。どこに配置されているどのようなモジュールに対しても統一された形式で通信の処理ができるような確たる論理モデルの設定を行ない，それを反映したハードウェア方式の開発が必要である。

b. 情報交換における障害対策

オペレーティング・システムを機能分散すると，当然モジュール間の情報交換において誤りの生じる確率が高くなる。したがって，このような障害対策にあまり注意をはらっていなかった集中型オペレーティング・システムと異なり，機能分散アーキテクチャでは情報交換に初めから障害を想定し，対応できるような配慮が必要である。この障害の検出と回復の問題には，同期，その他難しい問題が多く統一された一般的な方式の開発が必要である。

c. ケーパビリティ・アドレッシング，暗号化技術

a，bで述べた信頼性の高いモジュール間通信方式を具体化するための技術として，ケーパビリティ・アドレッシング方式や暗号化等が考えられる。これらの技術の更に深い追求，ならびにその実現に必要なハードウェア及びソフトウェアサポートの開発が望まれる。

d. ローカル・オペレーティング・システム・モジュールの構築

分散された各ローカル・オペレーティング・システム・モジュールはそれぞれのインタフェース規約にもとづいて設計されねばならない。個々のモジュールの規模は従来より小さくなると予想されるものの種類は増し，総量は多くなると考えられる。そこで，機能分散アーキテクチャではこれらローカル・オペレーティング・システム・モジュールの基本概念の確立，構築技術の開発が必要である。特に，パーソナル・コンピュータ内のローカル・オペレーティングシステムではネットワーク・アーキテクチャとの融合化を前提とした基本概念の設定が不可欠と思われる。また，構築技術としては，システム記述言語，ＯＳ用，ＬＳＩモジュール，各部分機能ＯＳのデータベース化等が考えられる。

e. コントロール・マシンの機能

センタ・コンピュータのサービス・マシン群では，これをいくつかまとめて制御するコントロール・マシン（ＯＳマシン）が必要になると思われるから，このマシンの処理方式，インプリメンテーション技術の開発が必要である。これについて詳しくは3.3.4に別に述べられる。

f. センタ・コンピュータ内のオペレーティング・システム機能の分散方式

各サービス・マシンに，オペレーティング・システム機能を分散する方式。サービス・マシン・ネットワークの構築，通信方式等の確立，eとの関連も大きい。

g. モジュール・インタフエースの記述

システム内の各ローカルオペレーティング・システムや，モジュール間通信インタフエースの記述，ならびに記述用言語，記述からの自動生成システム等の開発が必要である。

h. 分散系に対応するデータ管理機能の開発

分散されたシステム間でファイルを授受しあうために，分散ファイルのアクセス機構を分散されたオペレーティング・システムに組込ことが必要である。分散データベースではファイルカタログの充実が不可欠であるが，システム間に渡ったカタログの参照などそれに必要な機能を分散されたオペレーティング・システム内に実現する方式の開発が望まれる。

3.3.4 オペレーティング・システム（ＯＳ）・ニュークリアス・マシン

ＯＳの中枢機能の集合体をここではＯＳニュークリアス（核）と呼ぶが，機能分散システムにおいてはＯＳの中枢機能を分担するプロセッサ，すなわちＯＳニュークリアス・マシンの存在が重要になると考えられる。特に，機能分散型計算機として比較的規模の大きい一つのシステムを構築する上で有用な役割を果たすと思われるが，多種類の計算機（コンポネント・マシン）が複雑に階層化されて構築される一種の密結合計算機ネットワーク・システム全体にまたがるＯＳの考え方にＯＳニュークリアス・マシンの概念が有効に適用でをるか否かは今後の大きな研究課題である。

ＯＳニュークリアス・マシンの考え方に至る必然性を列挙すれば，

(i) ＯＳ開発の生産性を改善するためにＯＳの構成を単純化する。

(ii) 仮想マシンを容易に実現するためにＯＳの構成を階層化する。

(iii) ＯＳ機能の拡張性を保証する。

(iv) ＶＬＳＩの技術進歩で専用プロセッサの形で機能が分散する。

(v) 処理効率の向上を目指し，ＯＳ機能を考慮した高水準アーキテクチャを実現する。

(vi) 価格性能比の改善のために，ＯＳ機能のファームウエア化あるいはＶＬＳＩ化を図る。

(vii) ＯＳに徹底した保護機能や誤り処理機能を含める。

(viii) システムの基本動作をモニタする機能を組込む

などの要望が背景にあるものと考えられる。

(1) 構成イメージ

機能分散システムは分散化した機能を処理する専用プロセッサ群から構成されるが，ＯＳも専用プロセッサごとに機能分散されるものと考えられる。したがって，専用プロセッサに付随する機能単位のＯＳ（これを機能ＯＳあるいはＯＳ fi と呼ぶ）は互に殆んど独立的に存在すると考

えてよい。このため，機能分散システムで問題になるのはＯＳ fi とＯＳ fj 間の制御管理を行なうためのＯＳニュークリアス（ＯＳ n と略称）とユーザ・プログラムを処理するために必要な機能資源を確保し，仮想マシンを実現するためのＯＳ（仮想マシンＯＳあるいはＯＳ vi と呼ぶ）のあり方である。図３－７はＯＳの構成概念を示すものであるがＯＳ n を中核に仮想マシンに必要な専用プロセッサ群のＯＳ fi，j，…… から成ると考えられる。

ここで注意すべきことは，図３－７は一つの機能分散型計算機のＯＳに適用する構成の一形態を示したものであり，前述の密結合計算機ネットワーク・システムのような異機種計算機（それぞれがＯＳ n をもつ可能性がある）が階層的に組合わさったシステムにも通用するか否かは疑問なことである。この問題に関しては，3.3.3節を参照されたい。しかしながら，一般的に考えればこの場合にもＯＳニュークリアスの考え方は適用できると思われるが，システム全体を制御するＯＳ n を設けることは得策でなく，理想的にはＯＳも完全に分散し，可能な限り互に疎結合であることが望ましい。

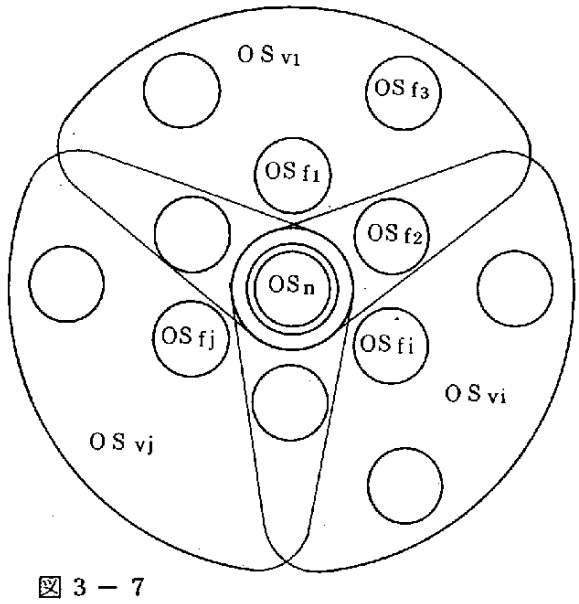


図３－７
仮想マシンＯＳの構成イメージ

(2) 機能イメージ

ＯＳ n の機能は専用プロセッサに与えられた機能（プロセスまたはタスク）ベースに多重処理制御を行なうと同時に並行プロセス実行環境を提供することである。したがって，ＯＳ n の機能の性格は，

(i) 普遍的な進行プロセス実行制御を行なう。

(ii) 受動的な動作を行なう。

(iii) 単一モジュールとして構成され，細分化されない。

(iv) ハードウエアとソフトウエアのインタフエースに位置づけされ，プロセス環境のシミュレーション，機能の抽象化，システムの仮想化を実現する。

(v) 実行頻度の高い基本操作命令を備える。

などであろう。一方，具備すべき基本操作機能としては，

(a) プロセスの多重化ならびにスケジューリング

(b) プロセスの実行環境の管理（プロセスならびにリソース管理）

(c) プロセス協調機構（Ｐ－Ｖ命令，同期，プロセス間通信方式）

(d) Capability 機能による保護機構

(e) 暗号化による機密保持

(f) 仮想マシンの管理

などに要するものであろう。

(3) 現状と問題点

ＯＳの単純化・階層化あるいはＯＳニュークリアスの考え方はかなり前からあるが，いずれも一つにまとまった単体のＯＳを対象としている。DijkstraによるOSの階層化，Hansen およびParnusによるＯＳ機能の分割・ＯＳ核の提唱，Harberman，土居らによる応用指向ＯＳファミリーの提言，Wulfらによる ＯＳ核をベースとしたＯＳ設計機構の実証，Garnesらによる Capability 機能の観点からのＯＳnの考察，Liskov らによるＯＳファームウェア化の実験など理論的な検討が見られるが，そのほかハワイ大学ＢＣＣ－５００，電電公社武蔵野通信研究所ＰＰＳなどの実験機，ＶＡＸ－11／780，ＰＲＩＭＥ 500，ＩＢＭ－Ｓ／38 などの商用機を通してＯＳ中枢機能の抽出，ファームウェア化による支援など実用機への摘要も試みられている。しかしながら，これらの考え方を高度な機能分散システムに適用するためには少くとも，

(i) 仮想マシンＯＳの構築ならびに管理方式

(ii) 多層レベルをもつシステム全体のシステム制御管理方式

などについて研究する必要がある。

(4) 研究課題

ＯＳニュークリアス・マシンに関する主な研究課題を列挙すれば次のとおりである。

(i) 分散処理システムにおける並列処理環境の提供

(a) ＯＳから見た機能の分割

(b) スケジューリング（多重処理）

(c) プロセスの協調

(d) プロセス（プロセッサ）間通信

(e) 保護機構（プログラム言語レベル）

(f) ハードウェア構成（プロセッサ結合方式，割込み，マッピング機構）

(g) インタフェース

(h) 評価（オーバヘッド）

(ii) 仮想マシンの構築・管理

(a) 仮想マシンＯＳの構築

(b) ＯＳマシンの可変構造機構

(iii) 多層レベルをもつシステム全体のシステム制御管理

3.3.5 モニター&コントロール

個別の機能をもつコンポネント・マシン群から構成される複雑な機能分散型計算機システムにおける重要な課題の一つはシステム全体の制御を効率的に行なうシステム制御方式を追求することである。しかしながら，効率的なシステム制御を実現するためには，システムの静的な状態を対象にするだけでは不十分であり，動的な状態を把握した上で，システム全体の最適化を図ることが望まれる。また，ユーザへの高度のサービスを提供することによって，より使い易いシステムにすることも大切である。モニタ&コントロール機構はシステム制御の中にあって，システムの最適化と高度な利用形態を提供することが目的で，第5世代計算機の大きな特徴になっている。

(1) 機能イメージ

モニタ&コントロール機構の基本的な機能を列挙すれば，

(i) モニタ機能：計算機ならびにプログラム内部の動的な状態情報を詳細に収集する。

(ii) 分析機能：収集した情報を目的別に解析する。

(iii) 最適化機能：収集した情報の分析結果を計算機あるいはプログラム内部にフィードバックし，処理効率やシステム構成の最適化を図る。

(iv) 学習機能：収集・分析した情報をデータベースに蓄積し，より効率のよいモニタ&コントロール機構の実現，あるいは将来の計算機アーキテクチャの在り方に活かす。

ことである。

モニタ&コントロール機構に期待される機能は次のとおりである。

(a) 性能の最適化（チューニング・サポート）

静的ならびに動的な最適化によるバランスのよい機器構成の設定，最適中間言語の設計，問題向き最適アルゴリズムの選択，誤り自動検出訂正機能の実現など。

(b) 高信頼性制御

各種故障の記録および分析・診断，自動修復，システム構成制御あるいは故障回避指示，誤差自動修正など。

(c) デバッグ・サポート

事象型高機能デバッガ，新しいソフトウエアへの移行を容易にするための高度は仮想マシンあるいはドキュメンティション・システムなどによるユーザ支援

(d) データ収集

ユーザならびにシステムのための各種データ収集・分析・表示・蓄積など。

(e) システム・コンソール機能

システムやプログラムの動作状態の表示システム構成指示機能など。

(f) システムの自動運転

計算機の無人運転あるいは自動管理・運用システムの実現など。

(g) ユーザ教育システム

ユーザの個人教育システム，高度なＨＥＬＰ機能などの完備。

(2) 構成イメージ

モニタ＆コントロール機構の実現に際しては次の３点に注意する必要がある。

(i) 上述の諸機能は必ずしも集中的に実現されるものではなく，必要に応じてそれぞれのマシンに付加される。

(ii) 種々の目的にあった多様なインタフエースを完備する。

(iii) モニタ機能は可能な限り高速なことが望まれる。特に，動的な最適化を行なう場合にはモニタ＆コントロール機構がシステム全体の処理速度を支配することがある。

(3) 現状と問題点

現在の計算機に見られるモニタ＆コントロール機能はＳＶＰ（Service Processor）あるいはＥＣＣ（Error Correcting Code）機構の中で故障対策のための機能として実現され，実用になっているにすぎない。将来，モニタ＆コントロール機構に期待する機能は多いが，その実現には個々の要求機能ごとに問題点を整理してみることが大切であろう。インタフエースの多様さも特徴の一つであるが，場合によってはＶＬＳＩチップ・レベルから考慮する必要も起りうる。モニタ＆コントロール機構に関連する動的チューニングについても研究が緒についたばかりで，具体的には未知であるが，今後の研究に待つところが多い。

(4) 研究開発課題

モニタ＆コントロール機構の実現にとって重要な技術を簡単に列挙すれば次のとおりである。

(i) ダイナミック・アーキテクチャ：マイクロプログラミング，可変構造ハードウェア，可変構造アルゴリズム，問題向き適応計算機アーキテクチャ

(ii) モニタ機構：ハードウェア／ファームウェア／ソフトウェアによるモニタ機構，バックトラック機構

(iii) 最適化機構：分析機構，フィードバック機構，再処理（計算）機構，自動チューニング機構，評価システム

(iv) 学習機構：学習用データベース，適応アルゴリズム，学習効果の評価システム

(v) 高信頼性制御：ＲＡＳ，ＦＴＣ（Fault-Tolerant Computing）

(vi) コンソール機能

(vii) デバッグ機能：事象型・高級言語・高機能デバッガ，高機能デバッグ・システム

(viii) ＣＡＩ：マニュアル・システム，高機能ＨＥＬＰ機構，個人教育システム

(ix) 自動運転システム

(x) 自動システム運用管理

3.4　コンポーネントマシン・アーキテクチャ

3.4.1　コンポーネント・マシンの考え方

機能分散アーキテクチャを構成するコンポーネントマシンは，パーソナル・コンピュータに代表されるユーザ・サイドのマシンと中央の高性能コンピュータを構成するマシン群に分かれる。後者は，更に通信回線を経由して他のマシンとの通信を制御するマシン・サービス・マシンと，高級言語マシン等ユーザに計算処理や情報管理等のサービスを提供するユーザ・サービスマシン，それに，これらマシン群の割付け等の管理や，特殊入出力機器等の制御を行なうコントロール・マシン群に分割される。なお，前年度の報告書にあるシステムAでは，ユーザ・サービス・マシンの中でも特に高度の性能を持つものをレベル3として位置づけたが，性能的な差はともかく，本質的差はあまりはっきりしないので，今回はまとめて考えている。

以下，3.4.2～3.4.5において，上記各マシンの機能と構成のイメージ及び，研究開発課題を詳しく検討するか，こうしたマシン自身も更に機能分割して機能分散アーキテクチャを採る可能性が大きい（3.1.2のレベル3）ので，各論に入る前にその点を少し考察しておく。

- このレベルの機能分散の目的は主に，並列処理による処理能力の向上とマシン構築の容易化にあると考えられる。
- 各マシン内部の機能分散であるから，その形を一般的に規定することは困難で，また，目的からして機能分割はそれぞれの応用に合せて考えられるべきものである。
- とはいっても，経済性の面から個別設計を少くするために一定範囲のマシン群に同一モジュールあるいは同一のコンセプトを適用できるように工夫することが望ましい。
- 並列動作が基本となるので，データ駆動型等の並列動作の効率的制御方式が求められる。
- 応用の持つ並列構造へのハードウエア構造の自動適応化等高速処理のための新機能の開発が必要である。
- モジュール間の結合は密結合であり，統一をとるために性能を犠牲にすることはできない。したがって個々のマシン依存になることもやむをえない。統一化よりも個別システムを容易に開発できるようにするため開発サポートを充実させることが必要である。

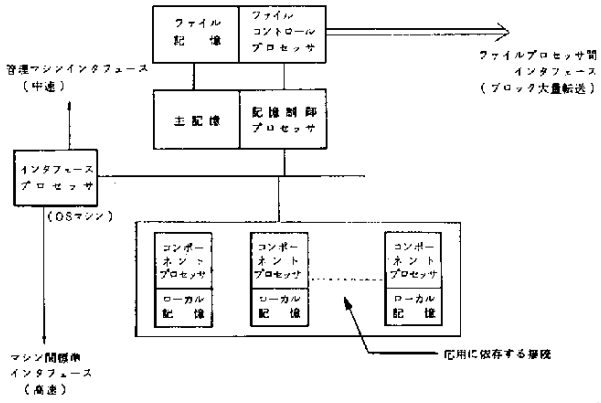


図3－8　コンポーネント・マシンの基本構成

- 標準的ユーザ・サービス・マシンの構成の一例を前頁に示した。（個々のマシンの構成は各論を参照されたい）。

＊　ここで，記憶制御プロセッサとファイルコントロールプロセッサはサービスマシンの種類によらずある程度共通化し，論理記憶（連想記憶）制御，データ構造管理，ケーパビリティー等アクセス制御等の機能を用意する。

＊　記憶制御プロセッサと主記憶は一体としてＶＬＳＩ化する。

＊　コンポーネント・プロセッサ群は標準のものを接続変更及びマイクロ・プログラムで応用適合させるものと，完全に専用マシン用として作られる場合があり得る。

＊　ローカル記憶も論理機能プロセッサが付属する。

＊　インタフエース・マシンとあるのはコントロール・マシンの一種で，主に他のマシンとの間の仮想化機能を与える。3.3でOSニュークリアス・マシンと呼んでいるものに対応する。

＊　図３－８にあるようなマシンがＯＳマシンの制御のもとに複数個まとまり，他のマシンからは一つのサービス・マシンとして機能するような構成も考えられる。

3.4.2　パーソナル・コンピュータ

１９９０年代のパーソナル・コンピュータとは，現在の大型機なみの性能を備え，機能的には，使い易さ，プログラム生産性向上のための配慮，図形，音声，画像などの新しいマンマシン・インタフエースのサポート等，大巾に改善された〝汎用高機能プロセッサ〟となると予想される。

ここでは，ハードウェア，ソフトウェアの各種オプションおよびパッケージの自由な流通を可能とする〝標準化〟の確立が最も重要な前提条件であろう。たとえば，ＶＬＳＩチップとして，各種ライブラリ，パッケージ等が標準規格の下に提供され，いずれのパーソナル・コンピュータにおいても，互換性を持って自由に着脱し，かつ動作させうることが求められよう。

また，パーソナル・コンピュータは，単一のイメージのものでなく，個々人のニーズに応じて，オプションを自由に組み合せて，パーソナライズされた一種の専用機にもなりうることが求められ，さらに，上位レベルの汎用機との有機的連携の下に動作するものとなろう。

従って，パーソナル・コンピュータ群および上位汎用機を含めたコンピュータ・ネットワークとして，人間生活の種々の面における計算機ニーズを満たしてくれる一種のユーティリティとして，現在では想像できないほど日常生活，経済活動，社会活動に密着した存在となろう。

(1)　機能イメージ

パーソナル・コンピュータは，それ自体が各種問題解決のための基本ツールとして使われる他，データ・エントリ，レポート生成，設計データの会話型グラフィックによる入力，修正等の上位汎用機の周辺処理を受け持ったり，上位汎用機のプログラムのデバッグ・マシンとして動作させることが求められる。従って，基本的には，レベル２のマシンとのプログラムのポータビリティ

のある仮想マシンを実現すること，レベル１，２の壁を意識することなく，自由にプログラム，データの交換，移動の行える分散データベース機能のサポートが必要とされよう。

以下では，１９９０年代のパーソナル・コンピュータの機能イメージを，ハードウェア，ソフトウェア，インタフェース等に分けて，その概要を描いてみる。

(a) ハードウェア

(i) 機能分散型ＶＬＳＩマルチ・プロセッサ：ＶＬＳＩプロセッサ・チップの組み合せによる機能分散型マルチ・プロセッサの構成をとるであろう。この機能としては，ＯＳ，高位言語処理，通信処理，ファイル処理，周辺装置処理などが挙げられよう。

(ii) 高位言語の直接実行型プロセッサ：ＣＯＢＯＬ，ＦＯＲＴＲＡＮ，ＰＬ／Ｉ，ＡＰＬ，ＰＡＳＣＡＬ　ＡＤＡ等の各種言語を，インタプリティブにあるいは，中間言語に翻訳して実行する各種高位言語プロセッサが実現されよう。

(iii) 組み込みシミュレータ機能：信号処理，アレー処理などを中心に，各種の高速演算用専用ＶＬＳＩチップが付加しうるハードウェア・アーキテクチャとなる。

(iv) 低価格高性能グラフィック処理：図形処理，画像処理専用のＶＬＳＩ高性能プロセッサと大容量メモリ，および高解像度表示装置などの実現により，使い易い低価格グラフィック処理が実現されよう。これもグラフィック・プロセッサとして機能分散の一部を形成することとなろう。

(v) 大容量パーソナル・ファイル：卓上型のコンパクト実装の下で従来の大型ディスク並の容量（１００～数１００ＭＢ）を持つ，パーソナル・ファイルが装備されよう。

(vi) 高信頼性の実現：特にＴＭＲ等のＦＴＣ技術に依らず，少い部品数で，かつ，高品質な部品により構成されることにより，家電品並みの信頼性が実現されよう。

(vii) 空調不要の耐環境性：ＶＬＳＩチップの発熱量は，かなりのものになると予想されるが，小型冷蔵庫的冷却技術の開発により，耐環境性の優れたものとなろう。
例えば，ＪＪ素子で作られ，小型魔法瓶的冷却器付等のイメージも考えられる。

(b) ソフトウェア

(i) シングル・ユーザ／コンパクトＯＳ：上位レベルとの連繋を密に行うことにより，単純で使い易い小型のＯＳが開発されよう。

(ii) Ｓｉチップ・ライブラリ：応用パッケージ・ライブラリ等が標準化され，市販のものを自由に装着して使うことが可能となろう。ＯＳの中核部もＳｉチップとして提供されよう。

(iii) プログラム・レスの実現：通常の適用業務処理，研究等の形態ではパッケージが充実し，パラメータ等の単純な指示で処理を進めることが可能となることにより，プログラム開発の必要性が大巾に削減されよう。

(iv) 学習機能の導入：推論，学習等の機能の導入により，使い易く，高度な使い方を許すものとなろう。

(v) 高度デバック機能の実現：プログラムを組む場合には，その効率促進の為の使い易い示唆に富むデバック機能が導入されよう。当然ＳＥＭＡＮＴＩＣ　ＧＡＰ等は解消されよう。

(c) インタフェース

(i) ネットワーク化：上位レベルとの機能分担，情報交換のための光ファイバなどによる高性能ＬＤＮ（ローカル・データ・ネットワーク）の確立，および各種他システムとの自由なデータ交換を許すネットワーク・サポートの標準化などが図られ，これらのための専用ＶＬＳＩプロセッサが市販されよう。

(ii) 人間との親和性の高いインタフェース：ユーザのニーズ，好みなどに合せ，音声，図形，画像，手書き文字などの各種入出力インタフェースを自由に選択し，組み合せることが可能となろう。

(iii) 日本語処理機能：手書き漢字を含む日本語の入力および出力機能の他，図形，画像などを含む自由な日本語文章の編集システムなどのインタフェース機能がサポートされよう。

(iv) オペレータ・ガイダンスの充実：初心者でも自由に使いこなせるためのＨＥＬＰ機能，オペレータ・ガイダンス機能が好みに応じて選択できることとなろう。

(d) その他一般的特徴

(i) 多種多様化への対応：レベル１のパーソナル・コンピュータは，多様化するユーザ・ニーズに対応する高機能端末として，現在の端末以上に発展するであろう。

(ii) 低価格性：現在の端末と同様，車一台分の価格と同じか，あるいは低価格で，汎用大型機並みの機能，性能が得られよう。

(iii) モジュラな拡張性とポータビリティ：小型，軽量でポータビリティに富むと同時に，オプションの自由な選択によるモジュラな拡張（*i.e.*, Lego）を持つものとなろう。

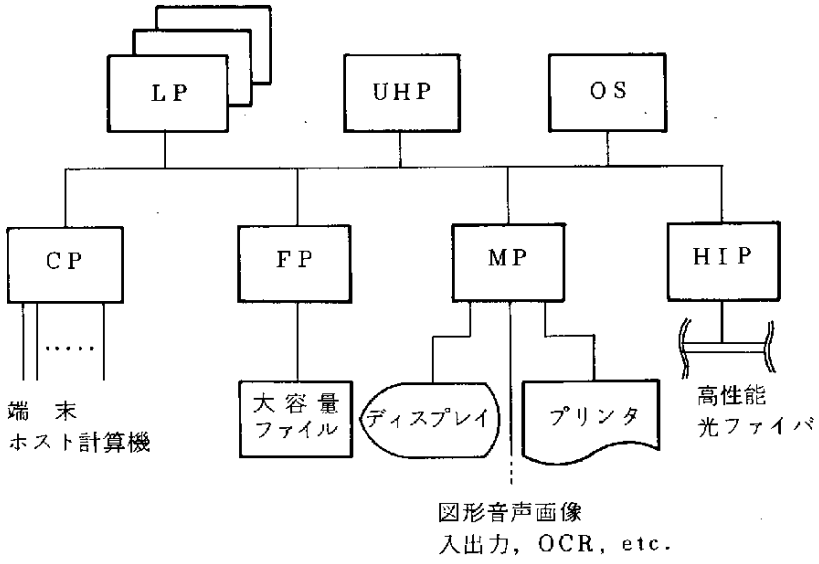
(2) 構成イメージ

１９９０年代のパーソナル・コンピュータの構成イメージの概要について以下に記す。

(a) 機能分散型ＶＬＳＩ多重プロセッサ：次頁の図に示される通り，各種のＶＬＳＩ専用プロセッサ・チップと，豊富な入出力の組み合せにより構成されよう。

(b) アーキテクチャの標準化：市販のライブラリ・チップ等の自由な着脱を可能とするため，ハードウェア・アーキテクチャの標準化が確立されよう。

(c) 本体部の概要：下記の如き構成を持つであろう。



ＯＳ：ＯＳ専用プロセッサ

ＵＨＰ：ユニバーサル・ホスト・プロセッサ

ＬＰ：高位言語プロセッサ

ＣＰ：通信制御プロセッサ

ＦＰ：ファイル・プロセッサ

ＭＰ：メディア・プロセッサ

ＨＩＰ：ホスト・インタフエース・プロセッサ

(i)　性能：処理能力は３～５Ｍｉｐｓ程度となろう。

(ii)　主メモリ：１～１６ＭＢ

(iii)　ローカル・ファイル：１００ＭＢ以上。シングル・レベル・ストア。

(iv)　特殊専用プロセッサ：信号処理，暗号処理，図形処理，画像処理，音声処理，等の導入。

(d)　入出力装置の概要

(i)　ディスプレイ：高解像度，カラー，低消費電力，低価格で人間工学的に見て優れたディスプレイの出現。

(ii)　プリンタ：上記ディスプレイに対応できる高速プリンタが出現しよう。

(iii)　図形入力：自由なイメージをディスプレイに手書き入力のできるもの，あるいは，ディジタイズ入力を自動的に行う高解像度，カラー，コンパクトで低価格の入力装置が出現しよう。

(iv)　音声入出力：dictation machine になりうる程度の音声認識（話者限定でよい）

能力を持つ入力装置の出現が望まれる。

(v) ＯＣＲ：手書き漢字等も含めて，もっと，自由な書体で人間の筆記能力に合う性能を持つ入力装置が望まれよう。場合によっては，個人の特性を学習する能力付でもよい。

(3) 研究開発課題

1９９０年代のパーソナル　コンピュータを実現するための研究課題のうち，主要と思われるものを，アーキテクチャ，ハードウェア，ソフトウェアの３つの面に分けて示す。

(a) アーキテクチャ

(i) ＶＬＳＩアーキテクチャ：チップレベルでの機能分散型計算機を構築するためのバス構造，チップ間コミュニケーション方法，メモリ構成法，チップ内蔵機能などのアーキテクチャ面での研究，および，それらの標準化の確立が求められよう。

(ii) 推論，学習機構：パーソナル・コンピュータのレベルで求められる推論，学習機能を探ると共に，それらをどのように実現するかのアーキテクチャ面での研究が必要である。

(iii) 高位言語プロセッサ：各種高位言語の効率的実行を目指すアーキテクチャの研究が求められる。

(iv) ＶＬＳＩアルゴリズム：各種ライブラリ機能を，ＶＬＳＩチップおよびＲＯＭチップ等の自由に着脱可能なものに組み込むためのアーキテクチャの研究と標準化の確立が求められよう。

(v) シングル・レベル・ストレージ：パーソナル・コンピュータの主メモリ，補助メモリ（バブル？）および着脱可能な記憶装置におけるデータ・ファイルの階層記憶制御を行うアーキテクチャの研究が必要である。

(b) ハードウェア

(i) ＶＬＳＩの開発：ＪＪ素子の可能性も含めて研究開発が必要

(ii) マン・マシン・インタフェース改善のための各種入出力装置：高解像度グラフイックディスプレイ，イメージ入出力装置，音声入出力装置，漢字ＯＣＲなどの研究開発が必要となる。

(iii) 大容量小型パーソナル・ファイル：大容量小型，高信頼性ファイルの開発が求められる。

(iv) 高速・低価格の標準化ローカル・ネットワーク・インタフェース：電気と同様に，プラグインで自由に接続できる標準ローカル・ネットワーク・インタフェースの研究，開発が必要である。

(c) ソフトウェア

(i) コンパクトでシンプルなＯＳ：ファームウェアも駆使して，機能は豊富であるが，自

由に選択して使え，コンパクトかつシンプルであり，標準化されたコンスプトに基づくＯＳの開発が望まれる。

(ii) 分散データベース：ホスト計算機（レベル２）または，他のレベル１のパーソナル・コンピュータ間で自由に送受信またはアクセス可能である標準化された分散データベース管理システムの確立が望まれる。

(iii) 新しい言語：単純で使い易く，かつ機能の豊富な，また，イメージ，図形などの処理も自由に組み込める言語の開発が望まれる。

(iv) プログラム・ポータビリティの実現：ネットワーク内で，自由にプログラムを流通させることのできる標準化された手法の確立が望まれる。

3.4.3 マシン・サービス・マシン

マシン・サービス・マシンは，地域的に分散され設置されている他のコンピュータシステム，プロセス制御を行なう特殊システム，パーソナル・コンピュータ，簡易入出力端末等の間で，通信制御を行ない情報を送受信するために必要なマシンであり，１９９０年代の巨大情報処理システム構築のために不可欠なマシンである。

１９９０年代のマシン・サービス・マシンは，ローカル・ネットワーク，広域国内，国際ネットワーク等の各ノードで使用される。その利用形態，実現形態は，ネットワークにおける位置付け，役割りにより，多様化したものとなろう。マシン・サービス・マシンの専用機能は，価格性能比に応じて，ハードウェア，ファームウェア，ソフトウェアで実現されるが，ネットワーク・アーキテクチャの標準化に伴ない，各種機能のＬＳＩ化が急速に進展する。

マシン・サービス・マシンは，ＬＳＩ技術，光通信技術，ネットワーク・アーキテクチャの標準化等の技術的インパクトを受けるだけでなく，１９９０年代における社会環境，人間の行動様式，価値観，法律等のインパクトをも受けることが予想される。これは，マシン・サービス・マシンが，１９９０年代の情報化社会に密着したものになるためである。このため，特に高度の信頼性が要求される。

(1) 機能イメージ

１９９０年代のマシン・サービス・マシンが実現する主な機能の概要は，以下のようになろう。

① 基本機能

マシン・サービス・マシンに必須の機能である。

(a) 回線制御機能

（例）・通信回線の接続，切断，監視制御，データ送受信のタイミングをとるための同期確立，基本的な誤り制御を行なう。

・データビットの組立てやデータビットへの分解を行なう。

・データのアセンブル，速度変換，トラフィック変動の吸収等のために，送受信データのバッファリングを行なう。

(b) 伝送制御機能

例 ・送信側，受信側双方で相手を確認し設定するデータリンクの確立／終結方法，データ送受信フォーマット，受信データに対する応答形式／応答方法，エラー回復処理方法等を取り決めた伝送制御手順に従って，データブロックの送受信を行なう。

・誤り制御を行なう。

② サービス機能

マシンサービス・マシンが属するネットワークが提供するサービスを実現するための機能である。

(a) 伝送・接続機能に付随するサービス

例 ・伝送速度，同期方式，伝送制御手順などの異なる通信回線を持つマシン間で通信を行なう。またマルチ・メディアを扱う。

(b) 蓄積機能

例 ・着信データ端末やマシンに代って，データを受信し保持する代行受信を行なう。

・同報通信を行なう。

・メール・ボックス・サービスを行なう。

(c) 変換機能

例 ・使用コード，書式制御方法，ファイル形式，ジョブの起動・実行方法，通信プロトコル等の異なるマシン間で通信を行なう。

・データの圧縮，復元を行なう。

・文字コードの画像情報への変換などのメディア変換を行なう。

(d) 高度のフロー制御機能

例 ・種々の誤りの検出，通知，あるいはエコーによる試験等のプロトコルを持つ。

・緊急情報の優先処理を行なう。

・過負荷時に，自動的に制御方法を変更する。

(e) ネットワークの保守・管理機能

例 ・マシン・サービス・マシンや通信回線の障害時に，ネットワークの自動再構成を行なう。

・ネットワークの状態管理，記録を行なう。

なお，本機能の一部は，モニタ＆コントロール・マシン等で実現されることもある。

(f) 暗号処理機能

例 ・送受信データの盗聴および故意の防害を防ぐための暗号処理，鍵の管理を行なう。

・暗号処理における障害を検出，通知する機能を持つ。

(g) 機密保護機能

（例）・特定の資源にアクセスした利用者や，指名手配された番号，利用者をチェックアウトし通知する。

(h) ネットワーク・ユーティリティを実現するための機能

（例）・分散データベースに関する処理を行なう。

・ネットワーク・ジョブに関する処理を行なう。

③ その他

(a) 拡張性・増設性が高く，またシステムの変更に容易にかつ柔軟に対処できるハードウェア構成を持つ。

(b) 特殊な設置環境を必要としない。

(c) インタフェースやプロトコルの自動検査証を行なう機能を持つ。

(d) 自己診断機能を持つ。

(e) マシン・サービス・マシンのソフトウエアの開発，ソフトウエアの修正やデバグを行なうために必要となる入出力装置を接続できる。

(f) 保守・診断・性能測定用のインタフェース機構（端子など）を持つ。

さらに，マシン・サービス・マシンは，次の条件を満たすものとなろう。

(i) 広域ネットワークに接続される場合，1～10Mビット／秒の伝送速度を持つデータの処理ができる。

(ii) ローカルネットワークに接続される場合は100～500Mビット／秒程度の伝送速度を持つデータの処理ができる。

(iii) 信頼性が，現在より1桁以上向上する。

以上述べたすべての機能が，すべてのマシン・サービス・マシンで実現されるのではなく，マシン・サービス・マシンの利用形態とその果すべき目的に応じて，必要な機能が選択され実現される。

マシン・サービス・マシンの代表的な利用形態とその主目的は次の通りである。

㋑ 前置処理装置（FEP：Front End Processor）

通常は，ホスト・マシンと同一の場所に設置される。ホストからマシン・サービスに関する処理を分離し，ホストの通信オーバヘッドを減少させ，システム全体の価格性能比の向上，他のマシン接続に対するホストの融通性の向上を目的とする。

㋺ 遠隔集線多重化装置（RC：Remote Concentrator）

RCは，遠隔地で多数のマシン，端末等からの低速回線を集線し，少数高速回線でホスト・マシンまたはFEPとの間を接続する。少数高速回線の有効利用により，通信回線コストを

削減することを目的とする。

㈠ メッセージ交換装置（ＭＳＷ：Message Switch）

ＭＳＷは，電話交換機やパケット交換機等に対応する。

ネットワークに接続されているマシンや端末間の通信を任意の組合せで行なうことを目的とする。

㈡ ゲートウェイ装置（ＧＰ：Gateway Processor）

ＧＰは，異なるネットワーク間を接続する。プロトコル変換，コード変換，メディア変換や，必要に応じて，異なるデータベースへアクセスすることを可能とするデータベース変換，データベース操作命令変換等を行なうことを目的とする。

この他，テレメータやテレコントロールにより，プロセス制御を目的とする利用形態，端末群を一括制御する利用形態がある。利用形態は今後も多様化する。

なお，分散データベースを１つの集中したデータベースとして共用利用するために必要な，先に述べたスキーマ間の変換処理等は，利用形態に応じて，ホストマシン（データベースマシン）およびマシン・サービス・マシンで分担されて実現されよう。

参考として１９９０年代におけるマシン・サービス・マシンのネットワークの中での位置付けの例を図３－９に示す。

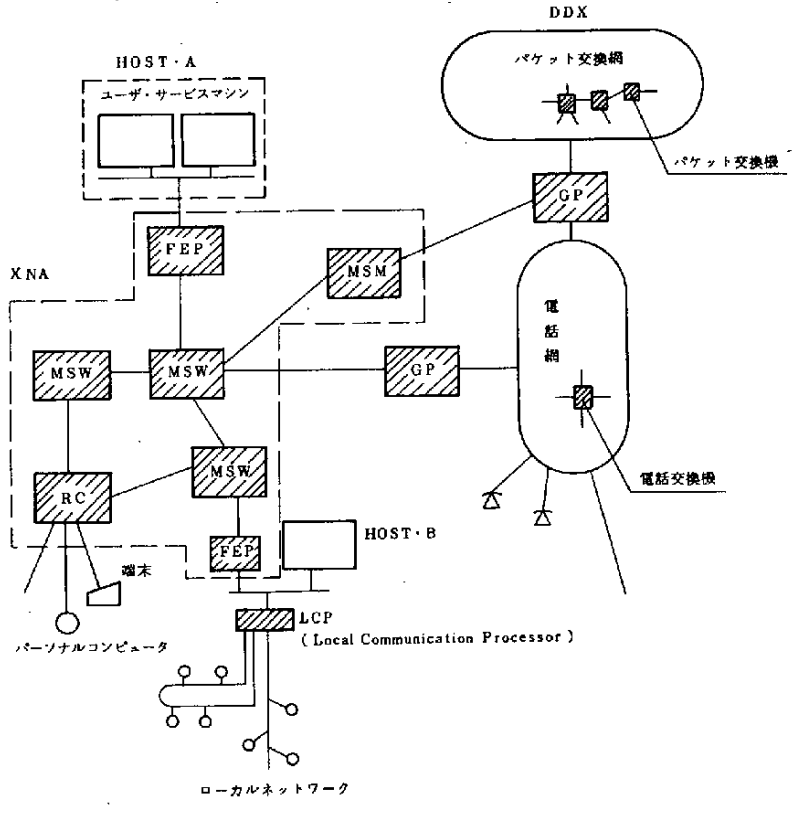


図３－９　マシン・サービス・マシンの位置付けと利用形態（例）

(2) 構成イメージ

① 構成法への主なインパクト

(a) マシン・サービス・マシンが実現する機能からのインパクト

- データの到着や信号の変化は一般に予期できない。また，オーバランによるデータの紛失を防いだり，一定時間内に伝送手順上の応答を返す必要がある。
- 基本機能やサービス機能が異なる多数のマシンからの回線に対して，処理を同時に並行して実現する必要がある。
- 収容回線種別，回線数，サポートするマシン，端末などは変動しやすい。
- サービス機能は，ネットワークの成長や利用者の希望に伴ない，多様化，高度化する。
- 高度の信頼性，機密性が要求される。

(b) ＶＬＳＩ技術からのインパクト

- ネットワーク・アーキテクチャの国際標準化に伴ない，基本機能のＬＳＩ／ＶＬＳＩ化やロジックのファームウェア化が進展する。
- データ暗号規格の制定に伴ない，暗号処理機能のＬＳＩ／ＶＬＳＩ化が進展する。
- 安価で高性能なマイクロ・プロセッサと通信用ＬＳＩ／ＶＬＳＩにより，機能を回線対応ごとに分散できる可能性が高まる。

② 構成法のイメージ

①から，以下のような構成が考えられる。

(a) 高速処理と信頼性向上のため，マルチプロセッサ，または，マルチ・マイクロ・プロセッサ構成が採用される。並列処理の可能性の例として，次のものが考えられる。

- マシン・サービス・マシンの機能は，ネットワーク・アーキテクチャのプロトコル階層に対応付けられるため，プロトコル・レベル毎に並列処理（パイプライン処理）を行なう。
- 回線対応ごとに並列処理（同時処理）を行なう。

(b) 多様化するサービス機能を実現するため，汎用命令セットと大容量主記憶，２次記憶装置を持つプロセッサが必要になる。

(c) ＧＰのような利用形態では，プロトコル変換等を高速化するため，連想メモリやハッシュ機構が一部採用される。

(d) 構成の一例を図３－１０に示す。

図中のプロセッサの特徴として，次のものが考えられる。

- レベル・マシン構造
- 割込み受付け機構
- ＤＭＡ方式

• バッファリング機構

• マルチ・ホスト機構（ただしＦＥＰの場合）

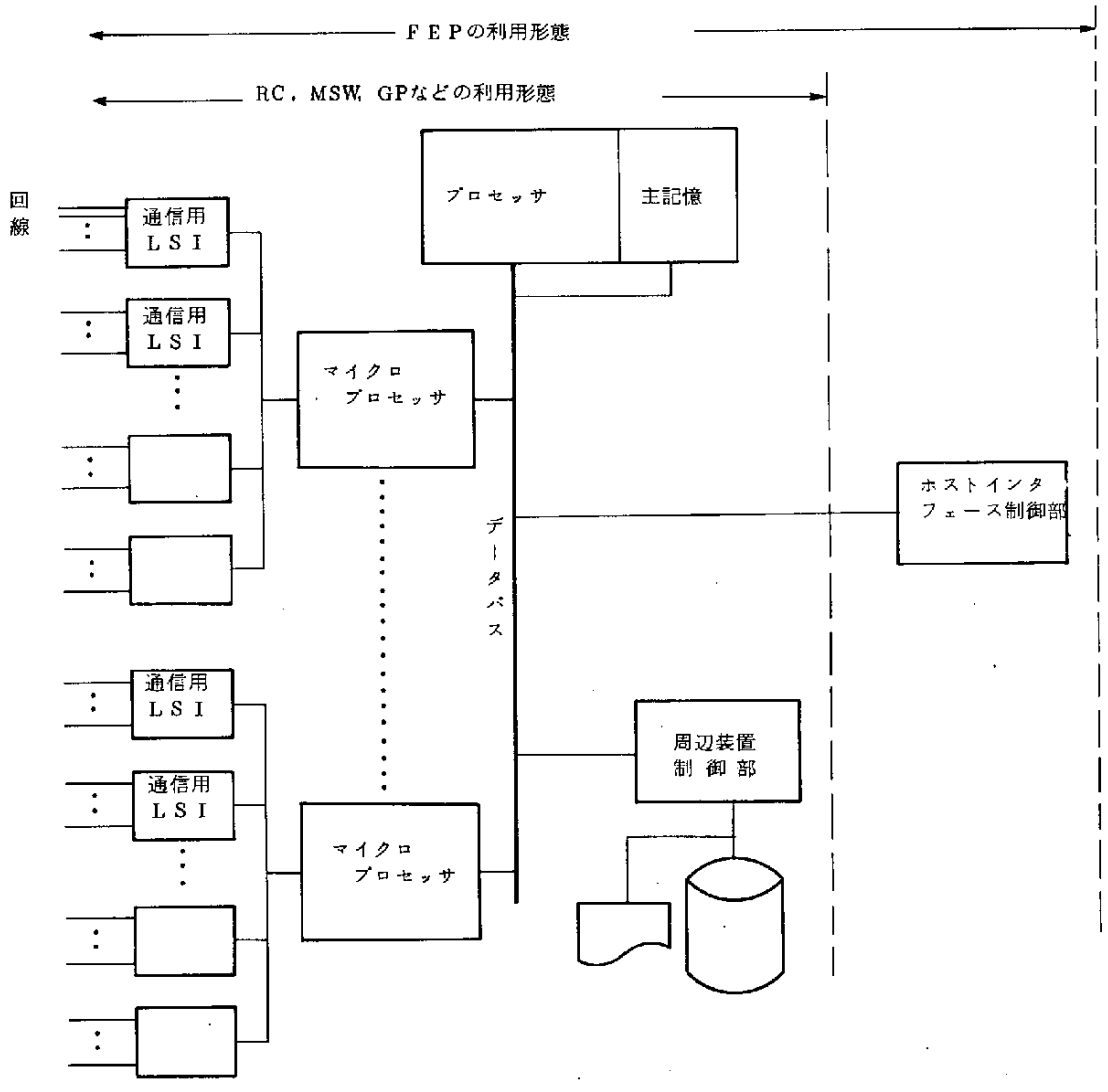


図３－１０　マシン・サービス・マシンの構成イメージ

③　構成法に関する補足

(a)　機能分散方法によっては，マシン・サービス・マシンの専用機能（主として基本機能）は，通信用ＬＳＩまたはその上位にあるマイクロプロセッサ内で実現されることになる。このため，図３－１０中のプロセッサの命令セットは，現在の通信制御プロセッサのものよりも汎用命令化が進む。また逆に，マイクロプロセッサは，現在のものより割込み処理や障害処理が強化されると共に，基本機能実現のための専用命令，機構が追加されると考えられる。

(b) 通信用ＬＳＩと上述の機能強化されたマイクロプロセッサが一体化され，通信処理用アダプタ（マシン・サービス用アダプタ）が構成される。少数低速回線を収容するマシン・サービス・マシンは，このアダプタを汎用マシン（ミニコン，小形コンピュータ等）に接続することにより実現できる。

(c) 暗号用ＬＳＩは，低速回線に対しては，集中設置（共用利用），高速回線に対しては，回線毎に設置されよう。

(d) 必要に応じて，マルチ・メディアに関する処理を実現するための専用プロセッサ（画像処理プロセッサ，アレイ・プロセッサＦＦＴプロセッサ等）が接続される。

(e) 必要に応じて，プログラム開発用，および各種データ収集用周辺装置が接続される。

(f) マシン・サービス・マシンの構成（特にＭＳＷとしての構成）は，交換機のディジタル化や構内交換機の電子化（ＥＰＡＢＸ）から発展したものと，汎用コンピュータの通信機能を強化したものとがあるが，両者が使用するプロセッサの差はほぼなくなりつつある。

(3) 現状と問題点

ネットワークアーキテクチャは，これまで汎用大型コンピュータを中心とする集中形のネットワークとして発展してきたが，今後は，高度のサービスを実現するために，本格的な分散処理方式のネットワークが必要とされている。このような動向を考えた時，現在の最大の問題点は，3.2.3で述べたようなネットワーク，ユーティリティを実現するために，マシン・サービス・マシンは，どの機能を，どのように分担して実現すればよいかが，十分に明らかにされていない点にあると言える。

一方，マシン・サービス・マシンのハードウェアは，今後も漸進的な技術進歩が続くと考えられる。

参考として，現状の通信用ＬＳＩ技術を表３－２に示す。

(4) 研究開発課題

主な研究開発課題は，次の通りである。なお，マシン・サービス・マシンは，ネットワークアーキテクチャの各ノードを構成する要素であるため，3.2.3で述べたネットワークアーキテクチャの研究開発課題と深い関連をもち，大部分はそれと重複する。

① ＶＬＳＩ技術

ＶＬＳＩ技術は，マシン・サービス・マシンの構成法に大きなインパクトを与える。ＶＬＳＩ化により，高性能のハードウェアが低価格で実現できると伴に，装置の小形・軽量化，機能のモジュール化も達成できる。プロトコルの標準化に伴ない，どの機能を，他の機能とのインタフエースをいかに高水準にしてＶＬＳＩ化すればよいかが重要となる。また，高速通信回線とのインタフエースをとるためにも，基本機能のハードウェア化が必要となる。

表3－2　通信用ＬＳＩ（高位レベルへのＬＳＩ化が進んでいる）

	レベル	種別	主な機能	LSI ECL1533	LSI WC2501/2511	備考
通信処理に必要な機能と実現方法の例	X.25 レベル2	回線制御	フラグ作成，0挿入除去，回線接続・切断制御，フレーム送出	○ 実現	○	通信用LSI
		状態制御	コマンドレスポンス作成と送信指示，状態管理，遷移制御	—	○	
		タイマ制御	タイマ歩進制御 タイムアウト通知制御	—	○	
	X.25 レベル3	接続制御	論理チャネル制御 呼設定・解放制御 ユーザファシリティ制御	—	—	等のソフトウェアマイクロプロセッサ/ミニコン
		データ転送制御	順序制御 誤り制御 フロー制御	—	—	
主な通信用ＬＳＩ	• ＥＣＬ－1533（DCNA共同開発，日電製作．～250Kb／S） • SD1933（Western Digital，～1.5Mb／S），WD2501／2511（W.D.，～1.6 Mb／S） • Intel－8273（～64Kb／S） • Fairchild－3846／6856（～1Mb／S） （ ）内は通信速度					
主な暗号用ＬＳＩ	• Intel－8294（80バイト／秒） • TI－9940（600バイト／秒） • WD－2001／2002（167Kバイト／秒） • Fairchild－9414（1.67Kバイト／秒） （ ）内は暗号／復号化速度					

② 通信プロトコルの標準化（ネットワークアーキテクチャの標準化）

コンピュータネットワークを効率よく実現し，各種のマシン，端末を矛盾なく接続するためには，各種通信プロトコルの標準化が不可欠である。

③ ゲートウエイ（変換）技術

通信プロトコルの標準化は不可欠であるが，現実のシステムは，標準化に先行して開発・実現されるものが多い。したがって，将来の標準化を意識しながら，異なるネットワーク・アーキテクチャとの通信を実現する技術の検討が重要である。また，オープン・ネットワークとしてのネットワーク・ユーティリティを実現するためにも重要である。

④ マルチメディア処理技術

情報化社会の進展に伴ない，コミュニケーション・メディアの多様化が進む。このため，音声，データ，画像情報などの異なるメディアを同時に扱う機能や，各メディア間の変換機能の実現法等が重要となる。

⑤ 高性能通信回線，通信方式との整合

今後は，光通信方式や，衛星通信方式が積極的に導入されるため，これらの通信方式に適し

た構成法を研究する必要がある。

⑥　暗号処理等の機密保護技術

通信回線を通して情報を送る場合には，特に機密保護が重要である。このためリアルタイム処理に適した暗号アルゴリズムの開発と，そのＬＳＩ／ＶＬＳＩ化，個人の識別方法等，通信における機密保護方式を確立し，マシン・サービス・マシンの機能として実現する必要がある。

⑦　ネットワーク・ＯＳの実現方法と，マシン・サービス・マシンでの機能分担資源共用等を目的としたネットワーク・ＯＳを構築するために必要となる諸機能（複数のホストでのネットワーク・ＪＯＢの実行，分散データベースへのアクセス，排他制御，ファイル転送等）を，ネットワークを構成するマシン・サービス・マシン間で最適に機能分担して実現する必要がある。

⑧　マシン・サービス・マシン用高水準言語

マシン・サービス・マシンの機能はソフトウエアによっても実現され，その量は，サービスの多様化と伴にますます増加する傾向にある。このため，ソフトウエアの生産性，保守性，信頼性の向上のために，マシン・サービス・マシンの機能の記述（例：プロトコルや非同期処理）に適した高水準言語の開発が必要となる。

⑨　プロトコルの記述法と検証技術

各種プロトコルは，プロトコル仕様に基き，ハードウエア，ファームウエア，ソフトウエアで実現される。したがって，プロトコル仕様にあいまいな点や矛盾点があると，マシン・サービス・マシンは正しく動作しない。このため，プロトコルを正確に記述する方法と，記述内容の論理的な正当性を保証する技術が不可欠である。また，プロトコルを自動検証する装置の開発も急がれる。

⑩　拡張性，柔軟性のある構成方法

ハードウェア，ソフトウェアのモジュール化構成を研究し，各種の変更に容易に対処できるようにする必要がある。

⑪　高信頼化設計技術

冗長構成技術，誤りの自動訂正，自動再構成技術等を開発する必要がある。

⑫　マルチ・プロセッサ構成技術

①，⑩，⑪等から，オーバヘッドが最小になり，しかもインタフェースレベルを高水準に設定できるマルチ・プロセッサ構成技術が重要となる。

(5)　参考文献

・橋本昭洋，山下正秀：〝通信制御プロセッサ〟，信学誌，Vol.No.11，P.1296 － P.1303（昭54－11）

3.4.4　ユーザ・サービス・マシン

ユーザ・サービス・マシンはパーソナル・コンピュータ・（レベル1マシン）で処理できないような仕事を処理するためのマシンで1990年代の巨大情報処理システムの中核をなす。

ユーザ・サービス・マシンはレベル2マシンおよびレベル3マシンから構成される。レベル2のユーザ・サービス・マシンは高性能の専用プロセッサ複数台から構成される機能分散型マルチ・プロセッサで，個々の専用プロセッサは現在の最高性能のコンピュータ以上の性能を持っている。それでも処理能力が不足する場合にはさらに高い性能をもった専用マシン，即ちレベル3マシンが接続できる。レベル2マシンとレベル3マシンの差は主として性能的なもので機能的には変わらないので以下においては特に区別しないで論ずるが，性能や量的なものについてはレベル3マシンを想定している。

ユーザ・サービス・マシンには種々の専用マシンが考えられるが，ここではその中でも特に重要と考えられる，高級言語マシン，データ・ベース・マシン，数値計算およびシミュレーション用のマシンをとりあげる。

ユーザ・サービス・マシンはＶＬＳＩ技術からのインパクトをうけ，またソフトウェア生産性向上や巨大情報システム構築，超高性能科学計算／シミュレーション・システム構築への強い要求からもインパクトをうける。ＶＬＳＩの利点を活かして，要求に応えられるユーザ・サービスマシンをいかに作るかは重要な研究課題である。

3.4.4.1　高級言語マシン

(1)　必　　要　　性

現在ではユーザ・プログラムはほとんどＣＯＢＯＬ，ＦＯＲＴＲＡＮ等の高級言語で記述されているのにコンピュータのハードウェア・アーキテクチャはフォン・ノイマン型のまつであり，ハードウェアが高価で，プログラムも機械語で書いていた時代のアーキテクチャと大差なく，そのセマンティック・ギャップは拡大しつつある。セマンティック・ギャップはソフトウェア開発上の種々の問題を引き起こす一因になっていると指摘されており，このギャップを小さくすることは1990年代のコンピュータの重要課題である。過去および現在においてもそのようなコンピュータは研究されており，いわゆる高級言語マシンとして知られているが方式的にみて確立したものはなく，商用化されているコンピュータも小型～超小型の比較的処理速度への要求が少ないものがほとんどである。

なお，以下にとりあげる高級言語マシンはフォンノイマン型コンピュータを想定して開発され，現在広く使用されているＣＯＢＯＬ，ＦＯＲＴＲＡＮ，ＰＬ／Ｉ，ＢＡＳＩＣなどの言語を対象としたもので，もっと新しい概念にもとづく言語や現在研究段階にあるような言語，たとえば述語論理プログラム言語，関数型言語，データフロー言語などは対象としていない。これらの言語を効率よく処理するには，いわゆるevolutionalなマシンでなく，

revolutionalなマシンが必要であると考えられ，2章で述べたような新しいアーキテクチャのマシンが実現されねばならない。その場合，もちろんそれらの新しいマシンも一種の高級言語マシンとして使用される。

(2) 機能イメージ

1990年代における高級言語マシンの機能は次のようなものになる，あるいはなるべきと考える。

① システム構成

(a) ユーザがシステムを使用する際にはレベル1のマシン（パーソナル・コンピュータ等）でプログラムのデバグを行い，完成したプログラムで実際の仕事を処理するときにレベル2，3のマシン（高級言語マシン）を使用する（デバグは会話型の処理が効果的であり，速度も遅くてよいのでレベル1で十分）。

したがって，高級言語マシンは実行が中心となり，実行効率の高いアーキテクチャにする必要があるが，レベル1でのデバグだけでは不十分であり，高級言語マシンでもレベル1以上に充実したプログラムデバング機能が必要である。

また，レベル1マシンでデバグする時にもデバグのためのデータや機能を適宜レベル2，3のマシンから補ってやることが必要で，ユーザはレベル2，3のマシンを意識することなくそのサービスを受けられる。

(b) 高級言語マシンはコンパイラ・マシンと実行マシンにわかれる。コンパイラ・マシンは各種言語に対し共通のマシンであり，実行マシンは言語毎に異なるマシンになるか，複数言語のために各々異なったハードウェア機能を持った一台のマシンになる。（直接実行型もあり得るが主としてレベル1マシンで実現する）。

② 基本機能

(a) 言語としてはCOBOL，FORTRANなど現在広く使用されているものを中心とする（前述のように新しい言語に対しては新しいアーキテクチャをもったマシンが高級言語マシンとして使用されるものと考え，ここでは除外した）。

(b) 実行マシンは実行速度重視のモードとデバグ機能重視モードを指定でき、デバグが不十分のプログラムは後者のモードで実行してデバグのための機能を十分に利用し，そのかわり実行速度を犠牲にする。十分にデバグの済んだプログラムではチェック機能をゆるめて実行速度をはやめた前者のモードで実行する。これによりデバグ機能を充実すると速度が低下してしまうという現在の問題を回避できよう。

(c) デバグがリース・ステートメント・レベルで出来る。これによって中間言語や機械語ベルでデバグする場合に比較してデバグ作業の効率がかなり改善されるであろう。機能的にはバックトラック機能などを備えている。

(d) 使用頻度の高いプログラムはＲＯＭ化されて豊富なライブラリを構成する。ＲＯＭ化することにより，使用する毎にメイン・メモリにロードする等の処理が不要になり，電源断時にも消えないなどの利点があり実行速度が改善される。さらにこれらのSilicon chipそのものがソフトウェア　ライブラリとして流通することが期待される。

(e) 言語仕様中のインプリメンター・デファイン部についてはパラメータ入力により相異を吸収する。将来の方向としては言語の標準化を進めてこのような必要がないようにするべきであるが完全に統一される迄にはかなりの年月を必要とするであろう。

③ サービス機能

(a) 複数の言語を用いて１つのプログラムを作ることができる。これによって，各種言語の特徴を活かした使い方，プログラム・ライブラリィの共通利用などが可能になる。

(b) ソース・プログラムに対する充実した編集機能を持ち，ドキュメント自動作成も可能。

(c) 自動的な適応化機能をもち実行速度を向上することができる。たとえば，プログラムの有する並列性を検出しユーザから指定されなくてもプログラムを並列処理する，効率の悪いコーディングを検出して効率のよいコーディングへの置換をユーザに教えるまたは自身で置換してしまう，ユーザのプログラミング上の特性（ステートメントの出現頻度，データ・タイプ）などを把握し，それに最適化する（ファームウェアの構造を変化させ出現頻度の高い処理を高速化するなど）。

(d) 言語仕様や使用法のガイダンスなどのＨＥＬＰ機能の充実により，十分な知識のない人でも容易に利用できる。

(e) モジュール・コンパイルとダイナミック・リンケージによりプログラムの作成，保守が容易になる。

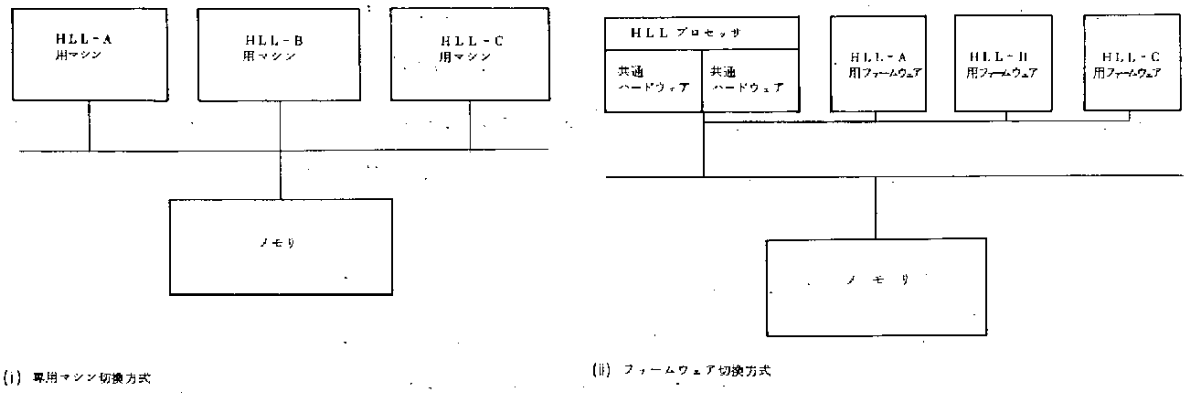
(f) プログラム実行時にハードウェアによって，range未定義データ参照，不正データ参照などの詳細なチェックが行なわれ，プログラムの誤りはその発生時点において検出される。

(3) 構成イメージ

以上のような機能を実現する高級言語マシンの構成は次のようになろう。

(a) 高級言語の処理を指向したハードウェア上でファームにウェアにより複雑な機能や言語毎に異なる機能を実現し，さらに高速化のために高級言語毎にその特徴に合わせた専用のハードウェア回路（language boardのようなもの）が追加される。

(b) 多種類の言語をサポートするためにシステムの構成を工夫する必要があり，言語毎に異なるマシンを置いて言語によりマシンを切換える，あるいは言語によってファームウェアを切換えるなどの方式がとられる。



(i) 専用マシン切換方式　　(ii) ファームウェア切換方式

図３－11　高級言語マシンの構成方式例

(c) 高級言語処理の速度向上のために新しい機能をもったハードウェア回路や新しい処理方式を導入する。たとえば連想メモリや連想処理ノスタック，パイプライン処理方式などが考えられる。連想メモリなどは従来技術では小容量のものしか実用化されていないが，もしＶＬＳＩ技術によって大容量のものが利用できるようになればアーキテクチャに大きなインパクトがあろう。

(d) 高級言語指向のアーキテクチャとしてTag，Descriptor，Capability Based Addressing などがあり，これらに上述のような新機能を追加したアーキテクチャが使用される。しかしながら，高級言語指向のアーキテクチャは現在まだ確立しておらずこれからの研究課題である。従来にない斬新なものが出現する可能性もある。

(e) プログラム・デバグを容易にするための機能が充実しており，専用のハードウェア，たとえばHistory Memoryなどが設けられる。

(f) 高級言語マシンは他のマシンとは機能分担型のマルチ・プロセッサを構成し，自身では直接入出力命令等は実行せず他の専用プロセッサに依頼する。

(g) プログラム・デバグ用に，使い易い優れたマン・マシンインタフェイスを提供する。そのためにカラー・グラフィック・デスプレイ，音声応答などが利用されよう。

(4) 現状と問題点

高級言語マシンは古くはバロースＢ５０００（１９６１）にはじまり最近ではＢ１７００，ＩＢＭ５１００などの商用機があり，研究レベルでは多数の例がある（５４年度報告書参照）。しかしながら現時点でまだ商用機のアーキテクチャの主流となっていないのは次のような問題があるからと考えられる。

・言語仕様にあいまいさがあり，ハードウェア化する上でマシン毎に解釈の異なる可能性がある。

・正常な動作に対して効率のよいアーキテクチャも異常，例外の処理まで考慮するとアーキテクチャが複雑になりすぎる。

・現在の単純な機械語命令しかないコンピュータでも完全な互換性を実現するのはむずかしいが高級言語マシンでは上述のような問題もあってよりむずかしくなる。

・フォンノイマン型アーキテクチャに打ち勝って普及するにはプログラム実行時のパフォーマンス／コストの優劣ではなく，ソフトウェア開発効率の観点からもコンピュータの評価がなされるような環境が必要である。

・現在普及しているＣＯＢＯＬ，ＦＯＲＴＲＡＮなどの言語はフォンノイマン型アーキテクチャのコンピュータでも比較的効率よく実行できるのでもっと言語が多様化しノイマン型アーキテクチャでは処理効率が悪い言語も普及する（あるいは要求が高まる）必要がある。

(5) 研　究　課　題

高級言語マシン実現のために今後研究しなければならない研究課題は以下のようである。

(a) 高級言語マシン用の中間言語の設定とできればそれを標準化してソフトウェアの互換性を高めること。

(b) 複数言語を処理するのに適したアーキテクチャ，ハードウェア構成方式など。

(c) 高級言語の仕様標準化あるいは仕様差を吸収するためのアーキテクチャ上の工夫。

(d) 並列処理の導入による高性能化。

そのために，並列処理アルゴリズム，並列性検出方式，ＶＬＳＩを利用した並列処理アーキテクチャなどの研究が必要。

(e) 適応機能，学習機能。

ユーザ・プログラムの特性を把握し，その処理効率を最適化するようにマシンの特性を適応化する機能を動的にやる方式および静的にやる方式。

(f) ＶＬＳＩの効果的な利用方法。

ＶＬＳＩにより高性能のＰＬＡ，マイクロ・プロセッサが実用化され，また大容量の連想記憶や論理メモリ（論理機能付き記憶装置）も実現可能になる。それらを効果的に利用した高級言語マシンのハードウェア・アーキテクチャ。

(h) ソース・ステートメント・レベルでのデバグ方式。

バック・トラック機能や最適なマン・マシン・インタフェース方式など。

3.4.4.2 データベース・マシン

(1) 必　要　性

データベースの利用は拡大しており今後も拡大していくと予想されていて，近い将来ビー

ク時10^6要求／秒の処理能力が要求されるかもしれないとの予測もあるが，現在のデータベースシステムは処理能力の高いものでも$10 \sim 10^2$要求／秒であり，画期的な改善がなければ1990年代の要求に応じることができないであろう。

フォンノイマン型アーキテクチャは数値計算むきで，データベース処理には適していないといわれているので，データベース処理にむいた新しいアーキテクチャをもち，大容量のデータベースに高速アクセスできる専用マシン（データベースマシン）が必要である。たとえば2章で述べた関係代数マシンなどは関係データベースを効率よく処理するのに適していると考えられ，その研究成果がデータベースマシンに応用されることになろう。

(2) 機能イメージ

1990年代のデータ・ベース・マシンのイメージは次のようなものになる。あるいはなるべきと考えられる。

(a) 関係データ・ベースを基本とするが現在使用されている各種のデータベースもエミュレーション機能等によってサポートする。

(b) 容量や処理能力は現在の大型データベース・システムに比較して100～1,000倍となる（レベル3マシン）。すなわち，データベースの容量は$10^{12} \sim 10^{14}$バイト処理能力はピーク時$10^4 \sim 10^5$要求／秒，処理による応答時間のバラツキは3秒～1分位である。

(c) 単純な自然語形式の問合せ言語が提供される。すなわち日本語により問合せができる（勿論他の言語も可能）。問合せには次のような各種のマン・マシン・インタフェースが提供される。

- 鍵盤入力，手書入力／文字出力（ディスプレイ）
- 音声入力／音声出力
- 図形入力／図形出力（ディスプレイ）
- その他……ディスプレイ画面上に手で触れることでシステムに入力できる。

ユーザはこれらの中から最も自分の用途，好みに適した方法を選択することができる。

(d) データベースは従来のコード化（coded）データだけでなく，図形，画像，音声などパターン情報なども含むことができ，これらが複合したデータベースも扱える。さらに知識ベースをサポートする。

(e) QBEをさらに発展させた機能をもち，詳細を知らなくてもデータベースに問合せて利用することができる。たとえば問合せ内容があいまいであってもそれなりに応答し，逆にユーザが求めているデータを類推してその検索方法を教える，などの機能をもつ。

(f) 高度のセキュリティ機能を有し，ユーザを判別して不当なアクセスを阻止する。ユーザの判別には単純なキーワードだけでなく声紋を利用することも考えられる。さら

にデータを暗号化することもできる。

(g) 応用分野ごとに異なる最適なデータ構造／データモデルを自動的に決定できる。たとえば，レコードのアイテム構成，レコード間の関係，使用するキー，論理式，定常利用形態などのパラメータを入力することにより最適なデータベース設計法が決定できる。

(h) データベースの変換，再構成が簡単に行える機能を提供し既存システムからの移行を容易にする。

(i) 異なる複数のデータ・ベース／データベースマシンへ自由にアクセスできる。

(j) 分散データベース機能が提供され，遠隔地にある各地のデータベースをあたかも1つのデータベースの如く利用できる。

(k) 高い信頼性を有し，障害が発生した場合にも短時間でデータ・ベースを復元できる。

(3) 構成イメージ

(a) 他の専用プロセッサと機能分担型マルチプロセッサを構成する。

また，業種（分野）別にそれぞれ異なるデータベースマシンを用いる構成も考えられる。

(b) データベースの処理を行うための集合演算系と記憶構成・制御系のハードウェア／マシンから構成される。

(c) ハードウェアにはパイプライン処理やデータフロー機構を取り入れて集合演算等の高速化をはかり処理能力を高める。

(d) 記憶装置は階層構造をとり，最下位の大容量記憶としてはＭＳＳや光メモリなどが利用される。上位には大容量連想メモリを利用して処理を高速化する。

(e) ハッシング機構のハードウェアをもちソフトウェアの処理を高速化する。

(f) データの暗号化／複号化のためのハードウェア機構が備わっており高度の機密を要するデータは暗号化させることができる。

(4) 現状と問題点

フォンノイマン型コンピュータを用いて実現された現在のデータベースシステムは次のような問題をかかえている （本分科会５４年度報告書参照）。

・複雑なデータ構造

問合せで与えられるシンボリック名をメモリ番地にマップするため複雑なデータ構造が必要で挿入，削除などの基本操作に対してこの複雑なデータ構造を正しく保持しなければならない。

・機能の多様性

データベースシステムは多くの機能的に異なる部分から成っており同一のハードウェア上に実現すると各所にボトルネックが生じバランスの良いシステムを構成することが困難となる。

・大規模ソフトウェア

データベース管理システムは今後大規模，複雑化することは明らかであり，また多様な応用の発生や使用パターンの変化に伴う再構成の問題を避けられない。このため大規模ソフトウェアの設計開発技術の改善と共に，データベース管理システムの基本機能をハードウエア化してソフトウエアの単純化をはかることが信頼性，処理能力等の向上のために必要である。

これらの問題を解決する手段としてデータベースマシンが注目されている。次にデータベースマシンの各種の方式，研究状況，商品化の動向などについて述べる。

(a) ＣＡＳＳＭ

フロリダ大学のＣＡＳＳＭ（Content Addressable Segment Sequential Memory）はセルラロジックをデータベースに適用した最初の試みとして知られており，次のような特徴をもっている。

・ネットワークモデル，階層モデル，関係モデルのいずれもサポート可能である。

・Segment Sequential Memoryはセルの集まりから成る。各セルは固定長セグメントを含み，ディスクのトラックとそれに付随するロジックから構成される。データはディスクのトラックに添って，ビット順次，語順次型に格納される。

・各ロジックは補助的メモリとしてＲＡＭビットアレイを持っており探索中間結果のマーク，出力用のマークのために用いられる。

・プログラム自身もデータメモリ中に格納する。ＣＡＳＳＭのアセンブラの命令はデータを探索してマークビットを立てるもの，本構造の階層データを検索するもの，実行シーケンスを制御するものなどかなり高いレベルに設定されている。

(b) ＲＡＰ

トロント大学では１９７５年にＲＡＰプロジェクトを開始し，１９７６年にＲＡＰ・１とよばれるプロントタイプが作られた。その後大巾な設計変更が行なわれて１９７７年にＲＡＰ・２が作られた。更に，アンカラ大学に移ったOzkarahanを中心にＲＡＰ・３を開発中である。

ＲＡＰ・１の特徴は，

・関係データベースをサポートする。

・セルはディスクのノトラックと１０２４ビットのＣＣＤメモリおよび付随するロジックから成る。データはビット順次，語順次型に格納される。タプルに対しては固定長表現が用いられ，一つのトラックには一つの関係だけが格納される。

などである。そのセルの構造を図３－１２に示す。

ＲＡＰ・２はコントローラにミニコンピュータ（ＰＤＰ 11/10）を使用，セルメモリと

して10^6 ビットのＣＣＤを用いた。命令セットを拡張したなどがＲＡＰ・１と異なり，ＲＡＰ・３はさらにメモリをＣＣＤからＲＡＭまたは磁気バブルに変えた。セルのプロセッサを複数にして市販のマイクロプロセッサ（インテル８０８６）にしたなどの改良を行っている。

ＣＡＳＳＭやＲＡＰのようにセル・ロジック・メモリを用いたデータベースマシンの研究としてはこの他にユタ大学のＲＡＲＥＳ，テクトロニクス社のＩＮＤＹ，ＩＢＭワトソン研究所のシステムなどがある。

(c)　ＤＢＣ

オハイオ州立大学のＤＢＣ（Database Computer）はＲＡＰ型の小容量高速のContent Addressable Memoryと可動ヘッドディスクを改良した疑似的なContent Addressable Memoryを組合せ，これにいくつかのデータベース機能プロセッサを付加した総合的なデータベースマシンで，その基本設計をもとにあるコンピュータ・メーカが試作中ともいわれる。

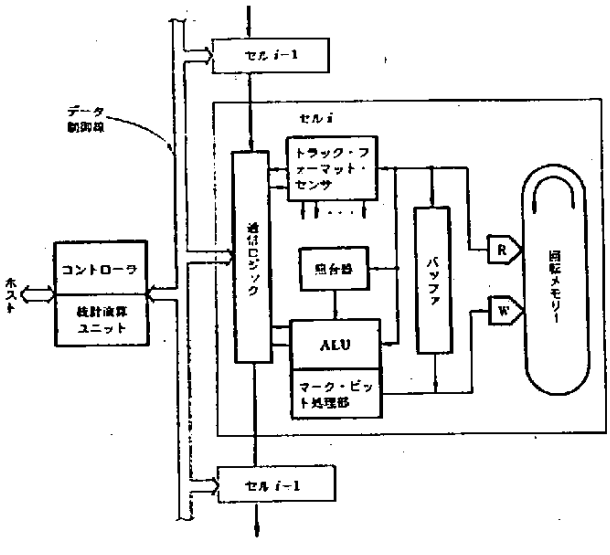


図３－12　ＲＡＰ・１のセル構造

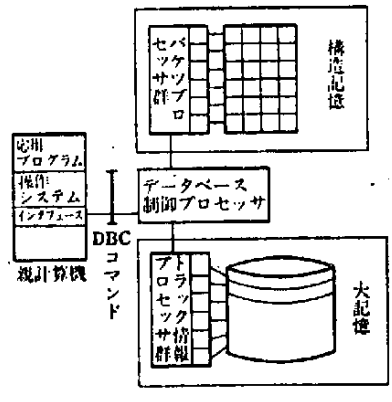


図３－13　ＤＢＣの構造

ＤＢＣにおけるデータベースは各ヘッドにロジックを持った可動ヘッドディスクに格納され１シリンダ分のデータの呼出しをディスクの１回転で終了できる。即ち現在の技術では0.5×10^6 バイトのデータを２０ms位で逐次探索できる。この方式により10^9バイト程度の大容量を容易に達成できる。しかしながらシリンダにまたがるデータ操作ではヘッドの移動が必要となり余分な時間がかかるのでStructure Memoryとよばれる補助記憶にデータ構造に関する情報を格納しておいてヘッドの移動回数を最小限にとどめる工夫をしている。この補助記憶はＲＡＰ型の固定ディスクを用いた装置でデータベース用可動ヘ

ッドディスクの1／100程度の容量をもっている。

ＤＢＣと同様の方式のデータベースマシンとしてはＭＩＴのINFOPLEX，フロリダ大学のＭＩＣＲＯＮＥＴ，ウィスコンシン大学のＤＩＲＥＣＴ，パーデュー大学のＤＩＡＬＯＧなどがある。

(d)　ＩＤＭ５００，他

Ｂｒｉｔｔｏｎ Ｌｅｅ 社のＩＤＭ５００は１９８０年２月に発表された，リレーショナルデータベース機能をもつ初の商用データベースマシンである。ＩＤＭ５００は図３－１４に示すようにＤＢプロセッサ，２５６Ｋバイトの主メモリ，ディスク制御装置，ホスト接続用のチャネル／通信インタフェース，から構成され，オプション追加でさらに大きいシステムを構成できる。リレーショナルデータベースは最大２ＫＢのタプルに最大２５０個のドメインを持ちリレーションには最大2×10^{9}個のタプルまで記憶できる。またデータベース当りの最大リレーションは３２,０００個，システム当り 最大５０種のデータベースを扱える。システム当りの磁気ディスク容量は最大３２ＧＢで，トランザクション処理能力は２～２５件／秒であるという。

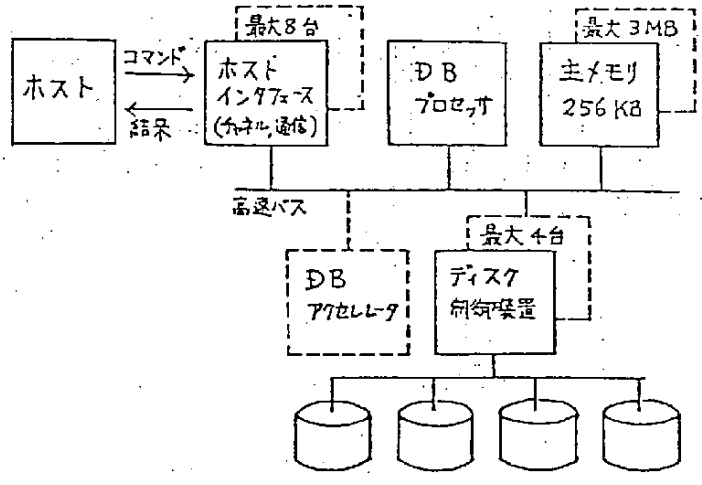


図３－１４　ＩＤＭ 500 の構成

ＩＤＭ５００と同様のアプローチとしては，ベル電話研究所のＸＤＭＳ，ＣＣＡ社のDatacomputer，カンサス州立大学のシステム，などがある。

また西独系のソフトウェアＡＧ社は１９８０年１０月に同様の型のデータベースマシンを発表している。これはＩＢＭ機と互換性のあるプロセッサ上にＡＤＡＢＡＳパッケージをのせ，ＩＢＭ系ホストコンピュータとチャネル結合するものである。また，MRI／Intel社はデータベース補助プロセッサを１９８０年３月に発表している。これはホスト上で同社のデータベースパッケージＳ2,000を使用し，データベース格納用の通常のディスクの他にIntel社の半導体高速ファイルメモリを接続した構成でＳ２０００用の転置ファイルの索引データを高速な上記半導体ファイルメモリ上に自動的に保持していくよう制御するものである。これは現在のデータベースシステムの性能を改善することが主目的と思われる。またＩＢＭはリレーショナルデータベースを内蔵したシステム３８を出荷しており，これを一種のデータベースマシンとする見方もある。

以上のように海外においてはデータベースマシンの研究が以前から行なわれており現在も盛んであるが，それらの研究成果をもとに実用化レベルのマシンが出てきている点が特に注目される。

(e) 国 内 の 現 状

北海道大学では専用のハードウェアによるサーチ，ソートを処理の単位とするデータフロー型のデータベースマシンを試作中である。サーチ，ソートを処理する専用ハードウェアは各々サーチエンジン（ＳＥＥ），ソートエンジン（ＳＯＥ）とよばれ，データベース格納用の大容量記憶とＳＥＥ，ＳＯＥ間のデータ転送とＳＯＥ，ＳＥＥでの処理をパイプライン化することによって検索処理の高速化をはかっている。ＳＥＥ，ＳＯＥは二分木構造を利用しており，たとえばＳＥＥではｎ個の検索されるキー（被探索キー）がその番号と共に大きさの順にソートされて記憶されており，ここにｍ個の探索キーが入力されると検索結果が被探索キーの番号によって出力される。ＳＥＥ中に等しいものがないとその旨出力される。

東京大学ではCellular Logic Typeのデータベースマシンを発展させた可変構造多重処理データベース・マシンによってリレーショナル・データベースをサポートする研究を行っている。このシステムは図３－１５に示すような構成をしており，光ファイバの高速ループに多数のプロセッシング・モジュール（ＰＭ），メモリモジュール（ＭＭ），コントロールモジュール（ＣＭ）を接続している。ＭＭはデータ格納用の磁気バブルメモリ及びマーキング用ＲＡＭから成り，データは必要なもののみ選択して出力される。ＰＭはそれらのデータのパタンマッチ処理をしてマーキング情報をＭＭに返すようになっている。

光ファイバループは固定数のチャネルに分割して使用しており，ＣＭがＭＭとＰＭにチャネルを割当てることによって任意のＰＭ－ＭＭ接続を実現することにより，従来のCellular Typeのようにプロセッサとメモリモジュールの結合が固定化しているシステムに比較して性能を改善しようとしている点に特徴がある。

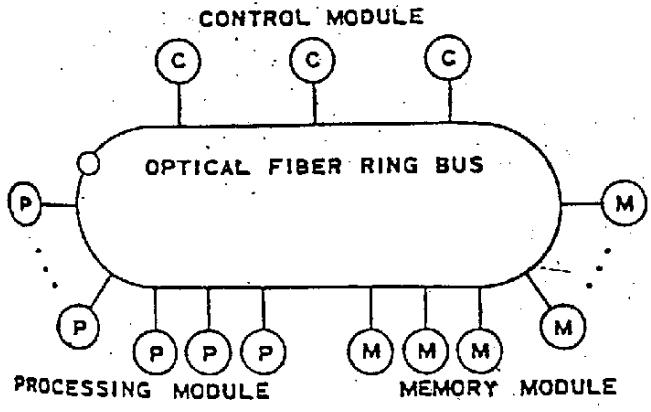


図３－15　可変構造多重処理データベース・マシンの構成（東京大学）

東大では，コンピュータ・システム全体を機能サブシステム，データサブシステム，入出力サブシステム，システム管理サブシステムから構成しようとする研究の一環としてファイルやデータ・ベースの管理を行うデータ・サブシステムの研究も進めている。データ・サブシステムの機能は，

(i) 通常のオペレーティング・システムにおけるデータ管理の機能を独立させて，サブシステム化する，即ちシステム全体で用いられる大容量ファイル記憶を管理し，他サブシステムに対しアクセスの手段を提供する。

(ii) さらに進んだ機能として関係データベースの管理やファイルに対するアクセスの多いユーティリティの機能をサポートする。

これによりサブシステム間の通信量を一層減少させることができる。

などでいる。上記機能のうち特に(i)の実現法を検討，評価するために実験システムが作られており，仮想記憶アクセス法（ＶＳＡＭ）の処理アルゴリズムを基本として拡張しファームウェア化したファイルアクセス法μ－ＶＳＡＭを用いている点に特徴がある。

東大ではこの他，同一の処理モジュールを多数ならべることによって大規模連想プロセッサを実現する方式（マルチ・モジュール構成）を研究中で，マルチ・モジュール構成の連想プロセッサＤＲＥＡＭ－Ⅱとホストコンピュータ（汎用ミニコンピュータ）からなるシステムを開発している。そのハードウェア構成を図３－１６に示す。

慶応大学においてはデータの格納および演算を属性単位で行い，演算の中間結果を動的

マークビットに保持する方式のリレーショナル・データベースマシンを機能分散型マルチマイクロ・プロセッサ構式で研究中である（ＳＰＩＲＩＴ）。この方式ではデータベース用の記憶部にマークビット用のスペースが不要になる，必要に応じて各演算毎にマークビットを動的に配置するので固定のマークビットの場合に起こる種々の制約を取除ける，マークビット数が無制限になり複雑な演算に対処でき複数の演算が一つのリレーションを共有できるようになるなどの特徴がある。

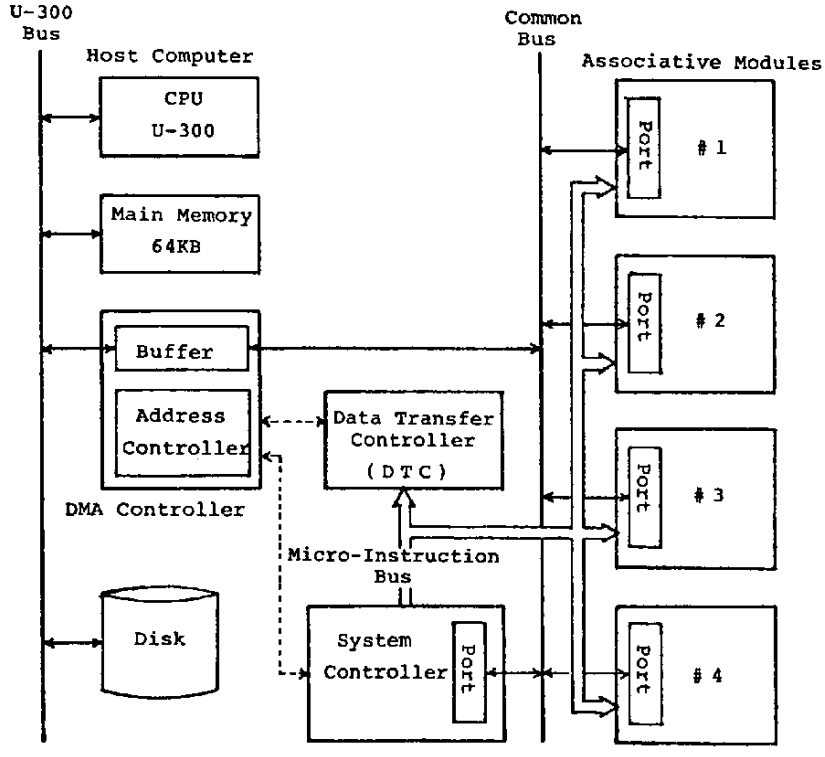


図３－16　ＤＲＥＡＭ－Ⅱの構成

横浜国立大では「情報源」（人物，名前など）とその間の関連を表わす「連想式」に基づいた独自のデータ・モデルをベースとし，汎用コンピュータとミニコンピュータの密結合で構成したデータベースマシン（ＡＩＳ）を試作している。

電総研では１９７６年から１９８０年にかけて磁気バブル記憶によるデータベースマシンＥＤＣを開発した。現在１メガバイトのメモリ・モジュール８個から成るＥＤＣ－Ⅱが完成して評価が行なわれている。ＥＤＣはマイクロ・プログラム制御のマイクロ・プロセッサ（ＰＵＬＣＥ），主記憶，磁気バブル記憶を組合せたデータ・モジュールを基本単位としこれを複数台共通バスに接続した並列処理型のデータベースマシンである（図３－17参照）。

日電・中研ではＣＯＤＡＳＹＬ仕様のデータベースマシンＧＤＳを研究中である。ＧＤＳはミニコンピュータ上に実現されており，汎用的に設計されていて任意のデータモデルが実現できるが実際の試作システムとしてはＣＯＤＡＳＬ仕様のデータベースの環境下で

動作している。システムの構成概念は図３－１８に示す通りでホストコンピュータとデータベースマシン間の通信オーバヘッドを最小にするためにホストコンピュータメモリを共有有している点が一つの特徴となっている。またミニコンピュータ上に実現されたデータベースマシンの諸機能は各種のデータモデルに適合できるように作られているだけでなく，頻繁に使用される機能をファーム化して高速化をはかっている。

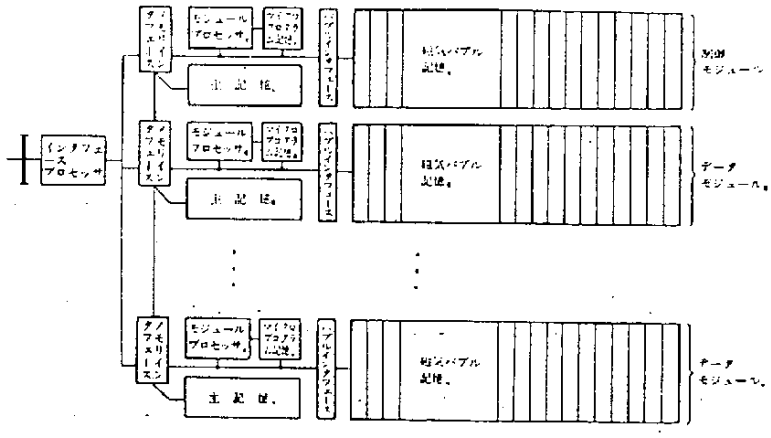


図３－１７　ＥＤＣの構成

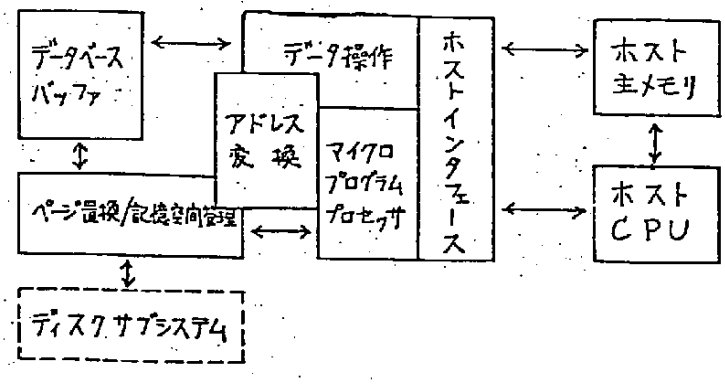


図３－１８　ＧＤＳの構成概念図

これら以外にも電電公社（横須賀通研および武蔵野通研），沖電気，東芝などで研究が進められている。

(5) 研究課題

大容量，高性能のデータベースマシンを実用化するための研究課題は以下の様である。

(a) ＶＬＳＩをいかに利用するか。論理メモリや機能アレイなど種々の利用法が考えられるが何が最も有効か。

(b) データベース（非数値）処理に適したアーキテクチャ処理方式。連想処理，探索・照合処理，パターン・マッチング機能など，およびこれらを統合した新しいアーキテクチャ。

(c) 冗長性やあいまい性をもった処理。

(d) 人工知能で用いられている手法をとり入れ高機能のデータベースマシンを実現する方式。たとえば知識ベースと推論機能の組合せなど。

(e) 使い易い問い合せ言語，マンマシンインタフェース特に日本語による問合せ言語，音声や図形による入力。

(f) 分散データベース。重複データの管理，最適分散方式，処理能力とネットワーク，ユーザが分散を意識しないで済ませる方式など基本的な問題がある。実用化までにはかなりの研究が必要である。

(g) モデルのマッピング技術

(h) 高信頼化技法

(i) 最適な記憶階層

(j) 高性能のデータベースマシンを実現するのに必要なハードウェア・テクノロジィ。たとえば大容量連想記憶，バブルメモリ／電子ディスク，ＪＪ，GaAs．

(k) ＶＬＳＩプロセッサを複数個用いた並列処理方式による実現。

(l) 既存データベースからの変換技術。

(m) データベースマシンと他の専用マシンとの機能分担。

(n) データベースアクセスパターン情報の収集／分析。

参 考 文 献

(1) 本分科会５４年度報告書

(2) 関野，「データベースマシンの研究開発と実用化」信学技報Vol. ８０，No. ２０５電子通信学会技術研究報告書ＡＬ８０－５４，１９８０年１２月

(3) 「ハードウェア後置型を中心に研究が活発なデータベースマシン」PP. １００～１３２，日経エレクトロニクス１９８１年１月５日号

(4) 植村，前川，「データベースマシン」情報処理学会，１９８０年７月

3.4.4.3 科学計算マシン

科学計算マシンは，科学技術の進歩とともに大規模化，複雑化する科学技術計算を高速処理するためのマシンである。物理学上の理論的研究，工学上の構造物の設計，その安全性，実験に代る数値シミュレーションなどに果す役割と重要性が，１９９０年代にはさらに増大し，専用マシンとして，科学計算マシンは必要不可欠なものとなる。

科学計算の専用マシンは，１９８０年代に普及し，１９９０年代には，この種の専用マシンがＬＳＩなどの進歩によりさらに高速化するとともに，情報処理システムの中に組込まれて利用形態が多様化すると予想される。

また，科学計算マシンのアーキテクチャは，ＶＬＳＩ，ジョセフソン接合素子などのハードウェア技術の影響を受けるとともに，目的とする科学計算の数値解法の影響を受ける。現在，すでに，目的とする科学計算には並列処理に向いているものや直列処理でなければ具合の悪いものなどが現われており，それぞれの目的別にアーキテクチャが多様化する傾向がある。

(1) 機能イメージ

大型の科学技術計算やシミュレーションを高速に処理できるマシンであり，高速演算機能をはじめとし，計算目的の多様化に沿う記憶容量の増大，記憶の階層化，ソフトウェア言語，中間結果の表示，計算精度，利用し易さなどの面で，高度の機能をもつ。

① 演 算 速 度

比較的に汎用性のある科学計算マシンとして10^3 ～10^4 ＭＦＬＯＰＳ程度のもの。

（極めて専用化されたシミュレーション用マシンなどでは10^5 ～10^6 ＭＦＬＯＰＳという意見もある）

② 記 憶 容 量

大容量の高速記憶をもつ。記憶容量は10^2 ～10^3 Ｍ語程度のもの。（１語は６４ビット以上）

一般の科学計算では，記憶容量と記憶スループットが演算速度を上げるために重要である。このため，さらに10^3 ～10^4 Ｍ語程度の高度二次記憶をもつなどの階層化された記憶構成をもつ。

③ ソフトウェア言語

ユーザから見た場合には，高級言語マシンに見えるような，コンパイラによる高度な自動最適化機能をもつことが望ましい。この最適化機能は，高級言語で記述されたユーザ・プログラムをベクトル・プロセッサ向けにベクトル化（Ｖｅｃｔｏｒｉｚａｔｉｏｎ）する機能や，並列処理プロセッサ向けに並列処理化する機能である。

使用言語は，現在の科学計算マシンではＦＯＲＴＲＡＮが最も多く使用されているので，

ＦＯＲＴＲＡＮベースのものが中心となるが，シミュレーションや図形処理などにより良く適応できる言語も必要になろう。

④　中間結果の表示

解の収束方向のチェック，パラメータの妥当性のチェック，プログラムのデバッグなどのため，中間結果を確認できる機能をもつ。

中間結果は，システムに組込まれているディスプレイ装置に表示される。

⑤　計算精度と数表現

計算精度は１語を６４ビットとし，その中に指数部と仮数部とを含む。（１語６４ビット以上という意見もある）

表現できる数の範囲を大きくするため，指数部を可変にするなどの新しい数表現の採用も考えられる。

⑥　可変アーキテクチャ

シミュレーションモデルに対応して，並列プロセッサ間の接続，システム構成などを変えられる機能をもち，ＦＦＴなどの特殊演算には，特殊機能を付加できるなどの最適化機能をもつ。

⑦　高　信　頼　性

並列プロセッサ構成の場合，予備プロセッサを用意するなど，ハードウェア，ソフトウェアを含めた自動再構成機能をもつ。

⑧　利　用　し　易　さ

遠隔のユーザから容易に利用できる機能をもつ。

計算結果の表現方法として，結果を３次元カラーディスプレイなどに理解し易い形で表示する機能をもつ。

(2)　構成イメージ

高速演算方式としては，パイプライン方式でパイプライン・ピッチを細かくして行く方法と，プロセッサを並列化して行く方法とがある。シングル・プロセッサでパイプライン・ピッチを細かくして10^3ＭＦＬＯＰＳを実現するには，ピッチのオーダーは１nsにならざるを得ないが，半導体技術ではこれを実現することは困難であろう。このためパイプライン・プロセッサによる並列方式あるいは並列プロセッサ方式を開発する必要がある。

①　構成法への主なインパクト

(a)　高速論理素子の開発と実用化

高速論素子としてガリウム砒素電界効果素子，ジョセフソン接合素子が開発され，高速演算論理として使用される可能性がある。

(b)　ＶＬＳＩの開発と実用化

ＶＬＳＩが実用化され，論理素子のＶＬＳＩ化が進む。

(c) 科学技術計算の大形化

科学技術計算が複雑化し，大形化する。このため，専用プロセッサあるいは専用シミュレータの開発が進むとともに，大形科学計算のために大容量主記憶が必要となる。

② 構成イメージ

構成法に対するインパクトから，次の様な３つの構成イメージが考えられる。

(a) パイプライン方式

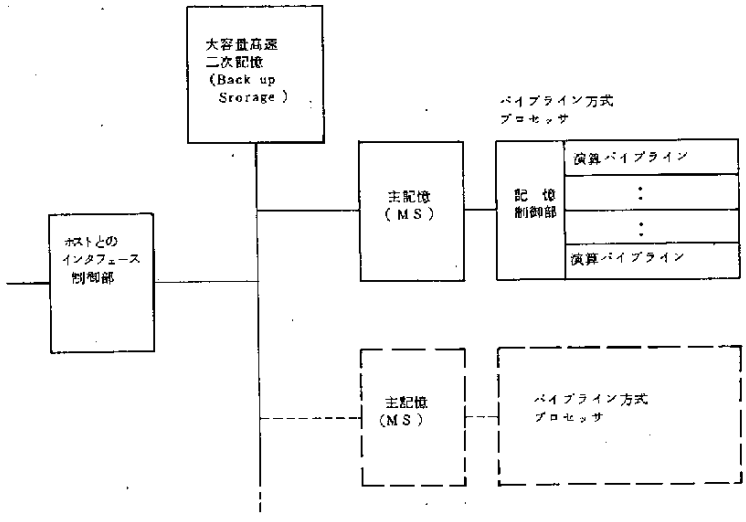


図３－19　パイプライン方式の構成

図３－１９にパイプライン方式の科学計算マシンの構成イメージを示す。

複数の演算パイプラインをもつパイプライン方式をもつプロセッサであり，二次記憶を共用する形でマルチ・プロセッサ構成を可能とする。

(b) 並列プロセッサ方式

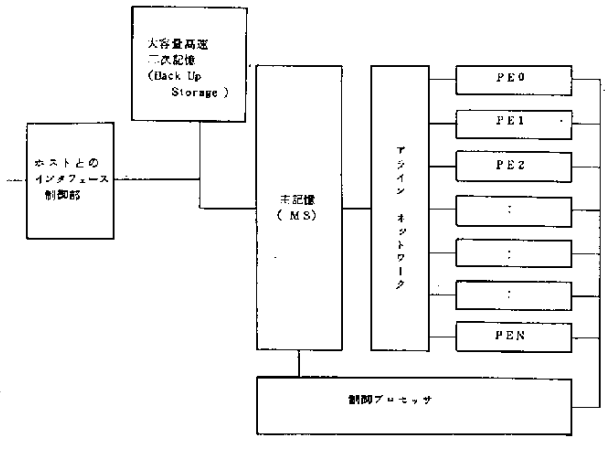


図３－20　並列プロセッサ方式の構成

図３－２０に並列プロセッサ方式の科学計算マシンの構成イメージを示す。

この方式は，多数のプロセッサ・エレメント（ＰＥ）を多数配列し，並列処理を行なわせる構成である。

(c)　プロセッサ・アレイ方式

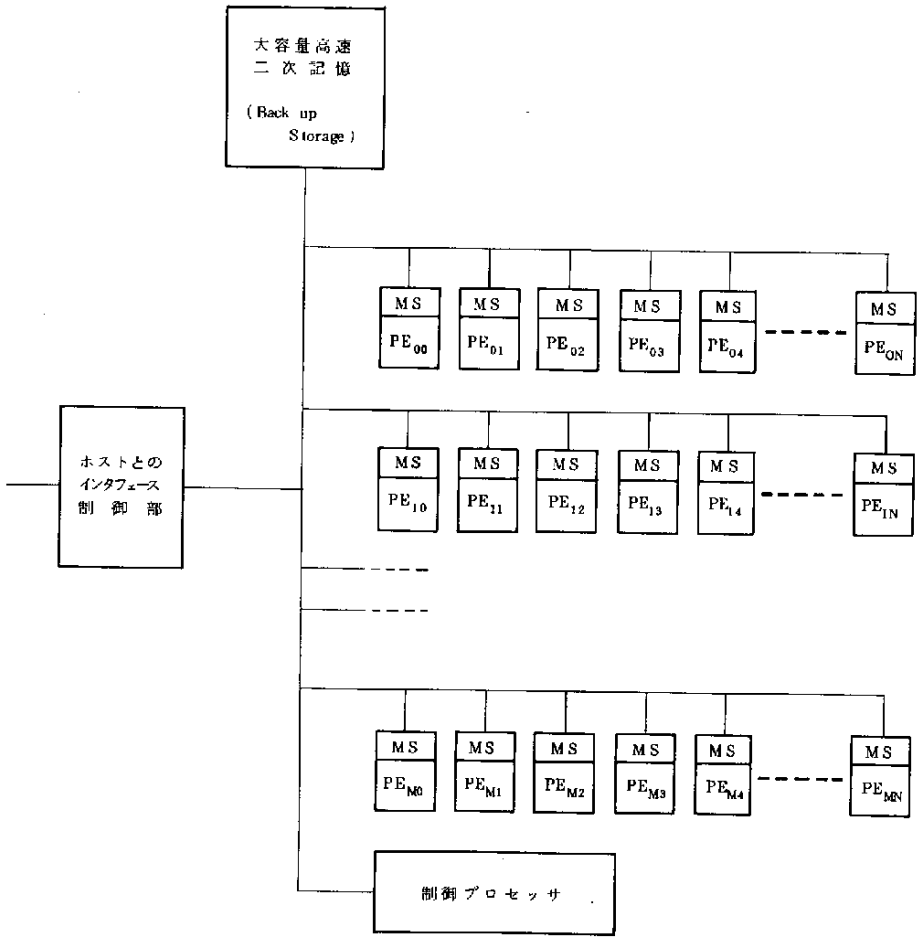


図３－21　プロセッサ・アレイ方式の構成

図３－２１にプロセッサ・アレイ方式の科学計算マシンの構成を示す。

この方式は，プロセッサ・エレメント（ＰＥ）をアレイ状に配列し，並列処理を行なわせる構成である。

(3)　現状と問題点

①　現状の科学計算マシン

最近に開発された科学計算マシンの処理方式と稼動時期を表３－３に示す。

表３－３　最近に開発された科学計算マシン

	名　　称	メ　ー　カ	処 理 方 式	稼動時間
1	CRAY　－　1	Cray Research	パイプライン	1976
2	B　S　P	Burroughs	並　列　処　理	1980
3	D　A　P	I　C　L	並　列　処　理 （プロセッサアレイ）	1979
4	CYBER 205	C　D　C	パイプライン	1980 発表
5	HITAC M－200H　IAP	日　　立	パイプライン	1979
6	FACOM 230／75　APU	富　士　通	パイプライン	1976

② 問 題 点

(a) 記憶容量

大形の科学計算では，データが主記憶容量を越えてしまい，演算時間よりも二次記憶と主記憶との間のデータ転送時間の方が大きくなる。

(b) コンパイラによる自動ベクトル化技術

科学計算専用機では，ベクトル処理化できたとき，並列処理が可能となり，高速処理できる。現実のプログラムには，条件文を含むステートメントやデータの参照関係が適当でないステートメントなど，コンパイラにより自動ベクトル化がむずかしいものや原理的にベクトル化できないものがある。

コンパイラによる自動ベクトル化の範囲を拡大する必要がある。

(c) ベクトル化できない部分の処理速度

ベクトル化された部分の処理が，高速化されるにつれ，ベクトル化できない部分の処理時間とのアンバランスが大きくなっている。

(4) 研 究 課 題

① 高速演算素子の開発と実用化

パイプライン・ピッチを小さくするため，高速の演算素子の開発が必要である。同時に配線遅れ時間を小さくするため，高密度実装を実現する必要がある。

これらを可能とする素子の例として，ガリウム砒素電界効果素子，ジョセフソン接合素子の開発および実用化の研究が重要である。

② 実装および冷却技術

高密度実装を実現するためのＶＬＳＩ技術，マルチ・チップのハイブリット技術の開発が必要である。同時に高密度ピンのコネクタ技術の開発が必要である。

また，素子の高速化，高密度化に伴ない，単位体積あたりの発熱量が増大するため，効率のよい冷却技術，例えば，液冷技術の開発が必要である。

③　並列処理技術

超高速演算を行なうためには，並列処理方式をある程度採用する必要があろう。しかし，例えば，連立偏微分方程式を連立一次方程式として，これを数値解法により解く場合を考えてみると，必ずしも並列処理に適したアルゴリズムで解かれていない。

このため並列処理化に向いたアルゴリズムの開発とそのアルゴリズムに適合し得る並列処理方式の開発が必要である。

④　科学技術計算用の超高級言語技術

物理学，および工学の数式で表現されたもとの問題を離散化し，数値解法の対象となる数式で表現し，これに数値解法のアルゴリズムを適用し，その手順をＦＯＲＴＲＡＮなどの高級言語で記述するなどのステップが踏まれている。

例えば，離散化した数式に近い表現のままで直接に記述できるようなイメージの超高級言語の開発なども重要な課題である。

⑤　自動並列処理化，最適化技術

コンパイラによる自動的なベクトル化技術，並列処理化技術，最適化技術の開発がパイプライン処理および並列処理プロセッサを有効に使用するために必要である。

⑥　ＶＬＳＩ化に向いたアーキテクチャ

ＶＬＳＩの開発により高速高密度素子が開発される可能性がある。このために，例えば，ＶＬＳＩにより実現できる１チップ演算素子を多数集めて超高速処理を実現するアーキテクチャなどの研究が重要となる。

⑦　ＶＬＳＩ化に向いたアルゴリズム

例えば，上記のような１チップ演算素子を多数集めて超高速処理を実現するアーキテクチャに対しては，それに適応する新たな数値解法のアルゴリズムが必要になる。

⑧　多数のマイクロ・コンピュータによる並列処理向きのアーキテクチャ

ハードウェア技術上は，超多数のマイクロ・コンピュータの複合体を構成することは，近い将来に，可能となるであろう。この方法で，超高速演算を可能とするアーキテクチャの開発が必要である。

特に，このような方式の場合，アイクロ・コンピュータの接続方式などにより，問題の構造の一部をハードウェアの内部に取入れることができるので，特定の目的に対しての高速シミュレーション・マシンを実現することもできよう。このようなシミュレータのアーキテクチャの開発も重要である。

⑨　高速大容量の二次記憶装置

現在の大型の科学計算では，いかに与えられた記憶容量に問題を収めるかが重要な課題になっている。この原因の一つは，１００ｎｓオーダーのアクセス・タイムの主記憶と，

１０msオーダーのアクセス・タイムの回転体の磁気記憶装置からなる二次記憶装置のアクセス・タイムのギャップにある。問題のデータが主記憶装置の容量を越えて，二次記憶装置に格納されると，計算時間が，二次記憶装置と主記憶間の転送時間で左右されるようになる。このためμsオーダーのアクセス・タイムの大容量の二次記憶装置の開発が，極めて重要である。

⑩　データフロー方式

高速計算向きのデータフローアーキテチャを開発することにより，高速並列計算を行なわせることができる可能性がある。

⑪　計算精度の明確な演算方式

計算誤差の伝播する範囲が明確にできるような，数値計算方法と，演算方式の研究が必要である。

参　考　文　献

(1)　小高，河辺「超高速演算の動向」
情報処理，PP. ９２７～９３７，Vol. ２１，No. ９，Sep. １９８０

(2)　" High Speed Computing and Algorithm Organization, "
Academic Press Inc. 1977

3.4.5　コントロール・マシン

コントロール・マシンは，機能分散処理システムが円滑にその機能を発揮するためのシステム運営，管理を目的としたマシンである。

コントロール・マシンは，ＯＳマシン，システム管理マシン，メデイア制御マシン，記憶制御マシンを含むマシン複合体である。

ＯＳマシンは，主としてタスクの分配管理を行なうと想定されるマシンであり，現在のＯＳのジョブ管理，タスク管理，メモリ管理などの一部を専用に行なうマシンである。具体的には，多重処理制御，スケジューリング，記憶保護領域の提供管理，仮想マシンの管理などが考えられる。このマシンについては，単独のマシンとしては存在せず，システムを構成する各マシンに分散された分散ＯＳの形で存在するという考え方もある。

システム管理マシンは，現在の汎用プロセッサに組込まれているサービス・プロセッサの発展した形のマシンであり，保守信頼性制御，システムの自動運転制御，システム・コンソール機能制御などを行なうマシンである。

メディア制御マシンは，現在の入出力プロセッサの発展した形のマシンであり，大規模なフ

ファイル管理機能はデータベース・マシンに譲るが，データベース管理以外のファイル制御，入出力装置の制御を行なうマシンである。

記憶制御マシンは，システムを構成する各マシンが固有にもつ固有主記憶と，共通にアクセスできる共通主記憶（二次記憶とも考えられる）へのアクセスを制御するマシンである。このマシンについては，独立なマシンとして存在するのではなく，現在の汎用プロセッサの中にある主記憶制御ユニットの発展した形として各マシンの中に分散して存在するという考え方もある。

(1) 機能イメージ

① ＯＳマシン

ＯＳマシンは，現在のＯＳのもつ機能のうち，ジョブ管理，タスク管理，システム資源管理機能などの一部の機能をもつマシンである。それらの機能イメージを挙げると，次のようである。

(a) ジョブ管理機能

マスタ・スケジューラ，ＴＳＳスケジューラ，バッチ処理スケジューラ，リアルタイム処理スケジューラ，ジョブ間通信管理などの機能など

(b) タスク管理

マルチ・タスク制御（プロセス切替え制御），タスク・スケジューラ，時刻と時間のサービス管理，プログラム・チェック処理（タスクの異常処理），タスク間情報交換サービス（マシン間情報交換サービス），一部のスーパバイザ・マクロのサービスなど

(c) システム資源管理機能

記憶保護領域の提供管理，システム資源を有効利用するためのジョブおよびタスクの適正配分の機能など

(d) その他

デバッグ・サポート，高度のＨＥＬＰ機能など

基本的には，ＯＳマシンは機能分散システム指向ＯＳの核を実行するマシンである。各マシンにはそれぞれの機能を遂行するのに必要なＯＳがあるというイメージなので，各マシンのもつＯＳに含ませることのできないＯＳ核を実行するマシンというイメージである。

② システム管理マシン

システム管理マシンは，スーパー・サービス・プロセッサの機能イメージであり，保守信頼性制御，システムの自動運転制御，モニタリングおよびシステムの最適化などの機能をもつマシンである。それらの機能イメージを挙げると次のようである。

(a) 保守信頼性制御

障害情報の収集および解析，保守診断および遠隔保守診断，障害時のシステム再構成，

システム内部状態の表示および変更機能など

(b) 自動運転制御

自動電源投入および切断，自動ＩＰＬ，ファイルの自動管理および自動セットアップ，定型スケジュールド・ジョブの自動走行，運転情報管理機能など

(c) システム監視および構成制御

システム・コンソール，システム制御パネル，システム運転状況の監視，構成制御および構成の表示機能など

(d) モニタリングおよび最適化

システム運用中にスループットやレスポンス・タイムをそれぞれ最大，最小にするような動的最適化，そのためのデータ収集，分析およびフィードバック機能など。

入出力装置台数，チャネル台数，記憶容量および各マシン台数などのシステム構成のバランスをとるような静的最適化，そのためのデータ収集，分析およびフィードバック機能など。

（これらを可能にするためには，各マシン側でも測定端子あるいは測定用内蔵機能を用意する必要がある）

(e) そ　の　他

デバッグサポート，例えば特定のアドレス領域のみの選択的な編集プリント，アドレス比較停止機能など。

③ メディア制御マシン

メディア制御マシンは，現在の入出力プロセッサを高インテリジェント化した機能イメージであり，ファイル制御，入出力装置の制御などの機能をもつマシンである。それらの機能イメージを挙げると，次のようである。

(a) ファイル制御

データベース管理を除くファイルへのアクセスに関するロジカル・インタフェースでのアクセス制御，ファイル・ハイアラーキの自動制御，ファイル装置管理機能など

(b) 入出力制御

プリンタ，カード装置，磁気テープ，その他各種入出力装置の制御，データ編集処理，仮想デバイス機能の提供（例えば，プリンタが使用できない場合に磁気テープで代替するなど），コード変換，メディア変換（図形情報，音声情報のコード化変換），入出力装置の管理機能など

④ 記憶制御マシン

記憶制御マシンは，仮想記憶管理および実記憶管理を中心に，これに伴なう記憶保護，アクセス制御などの機能をもつマシンである。それらの機能イメージを挙げると，次のようで

ある。

(a) 仮想記憶制御

各マシン固有の固有主記憶とシステム共通の共通主記憶（二次記憶とも考えられる）の記憶領域割当て，階層制御機能など。

さらに具体的には，ワンレベル・ストアあるいはそれに近い記憶制御機能をもち，仮想記憶領域割当て，実記憶領域割当て，ワーキング・セット制御，ページング制御，スワッピング制御機能などをもつ。

(b) そ の 他

ケーパビリティ制御，ガーベージ・コレクション，抽象データ構造サポート機能など。

(2) 構成イメージ

各マシンは，超高速バスを介して接続する構成イメージである。

ＯＳマシンは汎用プロセッサのイメージで１０ＭＩＰＳ程度，システム管理マシンはスーパー・サービス・プロセッサのイメージで数ＭＩＰＳ程度，メディア制御マシンはスーパー・ミニコン的なイメージで数ＭＩＰＳ程度のものである。

それぞれのマシンは固有の主記憶をもち，ＯＳマシンでは数１０ＭＢ，システム管理マシンでは数ＭＢ，メディア制御マシンでは数ＭＢ程度をもつ必要があると考えられる。

記憶制御マシンは特殊高速プロセッサであり，仮想記憶管理と実記憶管理のためのテーブルを格納する記憶部をもつ。

① 構成法への主なインパクト

(a) ＶＬＳＩの開発と実用化

ＶＬＳＩが実用化され，高速かつコンパクトな１チップ・プロセッサが開発され，実用化される。各マシンの構成法にインパクトがある。

(b) ファームウェア技術の発展

ＯＳの一部のファームウェア化技術が進歩し，シリコンＯＳなどが実用化される。ＯＳマシンの構成などにインパクトがある。

② 各マシンの構成イメージ

ＯＳマシンはＯＳのファームウェア化傾向により，大容量のファームウェアを内蔵する構成になることが考えられる。

また，異なる種類のＯＳで動作できる必要があるが，それぞれのＯＳに対して個別のマシンとなることが考えられる。

システム管理マシンは，サービス・プロセッサ機能のために，超高速バスを介しての接続以外に，各マシンとのサービス・インタフェースをもつ必要がある。専用のディスプレイ，ファイル，プリンタ，通信回線制御装置をもつが，マシンそれ自体はＶＬＳＩの１チッププ

ロセッサから成ることが考えられる。

メディア制御マシンは，高速チャネルをもつスーパー・ミニコン的なイメージであるが，ファイル制御とその他の入出力装置制御での機能とチャネル転送要求速度の差異などにより，それぞれを個別のプロセッサで構成することが考えられる。

記憶制御マシンは，共通の主記憶と超高速バスの間に位置し，特殊機能の高速プロセッサのイメージである。共通主記憶アクセスの競合による性能の低下を防止するために，マシン間通信方法の方式的な配慮が必要であるが，記憶制御マシンの構成についても高速の大容量テーブルメモリの構成方法の工夫が必要である。

図３－２２にコントロールマシンの構成イメージを示す。

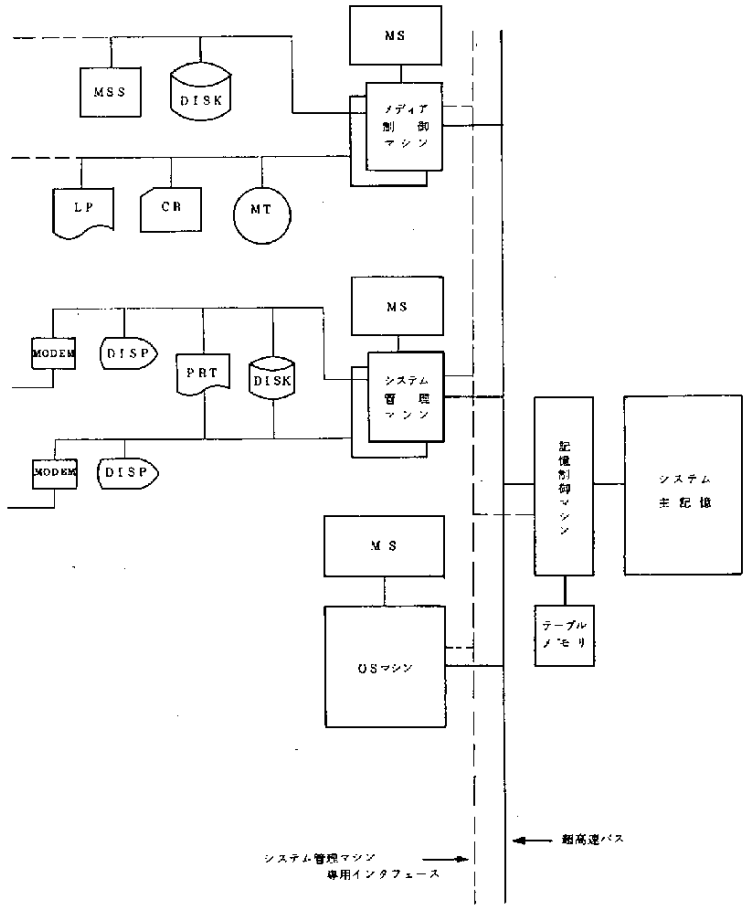


図３－22　コントロールマシンの構成イメージ

(3)　現状と問題点

①　現状のシステム制御

現状のシステム制御は大部分をＯＳが行なう形になっており，ＯＳが巨大化傾向にある。具体的には，メディア制御と記憶制御はほとんどをＯＳが行なっており，システム管理は一部をＯＳ，一部をサービス・プロセッサで行なっている。

現在のサービス・プロセッサのもつ機能は，オペレータコンソール機能と保守機能であるが，ミニコンやマイコンの高機能化，高速化に伴ない，それらで構成されるサービスプロセッサは，障害処理などに関してＯＳの一部を肩代りしつつある。

基本的にシステムのもつ機能としては，現在のＯＳのもつ機能を継承しなければならない面があるので，現在のＯＳのもつ機能を列挙し，ここで考えている分散処理システムではどのマシンに機能分散されるかのイメージを示すとおよそ次のようであろう。

• ジョブの実行管理	ＯＳマシン
• ジョブの入出力管理	メディア制御マシン
• ＴＳＳのサポート	パーソナル・コンピュータ
• データセットの作成	データベース・マシンおよびメディア制御マシン
• データセットへのアクセス機能	データベース・マシンおよびメディア制御マシン
• 通信アクセス法	通信制御マシン
• タスク制御	ＯＳマシン
• プログラム作成	各マシンに分散される
• サービス・プログラムの実行	各マシンに分散される
• センタの運営と管理	システム管理マシン
• 障害の対策	各マシンに分散されるまたはシステム管理マシン
• 記憶制御	記憶制御マシン

② 問　題　点

現状では，ＯＳに機能が集中しているために，ＯＳが複雑化し過ぎていることが問題と言えよう。これを機能分散して簡素化することが課題と考えられる。

さらに機能面での具体的な課題としては，次のようなものを挙げることができる。

(a) 論理アドレス機能の拡張

論理アドレス空間の拡大と，ワンレベルストアなどへの機能の拡張

(b) ケーパビリティ制御

融通性のあるケーパビリティ制御

(c) 入出力制御オーバヘッドの低減

(d) 記憶制御オーバヘッドの低減

(e) 信頼性向上のための機能強化

フェイルセイフのための機能の強化

(f) 自動運転機能

省力化傾向に沿う自動運転機能の強化

(g) 最適化運用機能

システムを最適化運用するための機能の強化

(4) 研 究 課 題

① ＯＳ分散方式

現ＯＳの機能を各マシンへ分散する方式，各マシン間の通信，同期，データ転送などの方式の開発

② 記憶制御方式

大容量階層記憶アクセス方式，ワンレベルストア方式の開発

これに伴なうケーパビリティ制御方式，ガーベージ・コレクション方式の開発

③ システム資源の最適化利用技術

システムの動的最適化利用技術の開発

④ ＯＳのファームウェア技術

ＯＳ核のファームウェア化によるコンパクト，高性能なＯＳマシンの開発

⑤ 自動運転技術

運用時の省力化，運用経費の低減のための夜間無人運転技術，異常検出技術，異常時自動停止技術などの開発

⑥ 信頼性向上方式

各マシンの診断方式，保守方式の開発

システム全体での信頼性向上方式の開発

Fault Torelant Computer技術の開発

⑦ メディア制御マシンアーキテクチャ

ロジカル・インタフェースでの入出力制御アーキテクチャの開発

プログラム作成効率およびシステム効率向上のための各種メディア制御アーキテクチャの開発

⑧ 図形処理，音声処理および文字処理技術

これらの効率のよい変換技術の開発

⑨ 適 応 化 技 術

運用状況のモニタリング技術（対象マシンとのインタフェース設定，同期化技術など），状況に適応するためのダイナミック・マイクロプログラミング技術，学習機構，チューニング技術などの開発

参 考 文 献

(1) 相磯，「機能分散型計算機システム」情報処理，Vol. 8，No. 4，１９７７

(2) 高平，「分散処理用オペレーティングシステムの課題」情報処理，Vol. ２０，No. 4，１９７９

3.5 高度マンマシン・コミュニケーション・システム

(1) 概　　要

人間と計算機との対話の円滑さは，今後の計算機システムを特徴づける大きな要素である。知識情報処理の一つの目的は，計算機に，意味を理解できる能力や簡単な"常識"を持たせることで，この対話の円滑さの向上を実現することである。対話のメディア（媒体）としては，従来は，英数字とカナ文字が中心であった。また，言語は，プログラミング言語やコマンド言語（ＪＣＬ）などであった。これを，人間の通常の対話のメディアである音声，図形，画像を含むものとし，言語は，自然言語を用いられるように，計算機の能力を質的に変換しようと試みる。

このような対話の実現には，音声や図形等の理解のための理論やソフトウェア面の研究が，第一に重要であるが，これと共に，この実現のためのハードウェアの研究も同時に進めねばならない。

このようなマンマシン・インタフェース用の入出力システム，特に，ハードウェア・システムの研究開発が，この課題の目的である。

第５世代コンピュータ・システムの研究開発においては，この入出力システムの研究開発の目的は，２種類に分けられる。その一つは，音声合成，出力システムや図形表示，印刷システムのように，理論やソフトウェア面の研究が，かなり進んでおり，ハードウェア面の研究開発が，早期に開始できるものである。この種のもののシステム開発は，主として，ＶＬＳＩ技術等を利用し，より高性能のものを作ることと，それを，研究開発支援ツールとして，できるだけ早く提供することが重要である。

第２のものは，音声入力や画像入力のように，その後にある認識や理解のための理論や方法に多くの研究を必要とする未知の分野を残しているものである。この種のものの研究開発においては，ハードウェアの開発は，音声理解や図形・画像理解の理論や方式面の研究と密接に関連づけて進めねばならない。

ここで開発するものとしては，音声や画像を信号レベルで処理する専用プロセッサなどが中心となり，理論やソフトウェアの研究のサポートの性格が強く，本格的なハードウェアの開発は，プロジェクトの後期に集中することになる。ここでは，カスタム化ＶＬＳＩを容易に設計・製作できるＶＬＳＩ－ＣＡＤシステムが不可欠となる。

また，表示装置や図形入力装置等は，プラズマ・ディスプレイや，高精度のブラウン管やＴＶカメラ等，部品・材料面での改良研究や新しい方式の開発等が重要である。しかし，これらは，物性面の研究等，広大な分野にわたるため，この課題の範囲を越えている。

このため，ここでは，これら部品，材料については，システム開発時点で使用可能なもののうち，最も優れたものを用いることとする。

(2) 必　要　性

入出力ハードウェア・システムの主なものをメディア別に述べる。

文字の入出力としては，漢字を含む和文の入出力システムが，研究開発ツールとして，早期に必要となる。これは，高精度の表示装置，印刷装置，また，カナ漢字変換機能を含む入力装置を含む。

音声に関しては，音声合成・出力装置が，研究開発ツールとして必要である。これは，機械翻訳や自然言語理解の研究のほか，種々のソフトウェア，ハードウェアの開発支援ツールとして，実験能率の向上に，大きく貢献することが期待できる。音声入力装置は，理論，ソフトウェアの研究開発をサポートすることを目的として，できる限り高性能のハードウェアを開発する必要がある。最終的なプロトタイプにおいては，ソフトウェア面の研究成果に基く仕様に従い，ＶＬＳＩ等を使用した専用ハードウェアの開発が必要である。

図形・画像の入出力に関しては，まず，白黒２値図形の入出力装置を開発し，研究開発支援ツールとして提供する必要がある。入力装置としては，高精度のタブレット型の座標入力装置を開発する。また，出力装置としては，高精度のグラフィックディスプレイ装置と，ハードコピー用の印刷装置が必要である。これらは，図面の入出力，グラフや表などの図版を含むドキュメンツの入出力に用いられるほか，文字入出力装置と組合せて，図面を含む書類などの作成に用いられる。

白黒濃淡，および色彩をもつ画像の入出力は，画像応用システムのソフトウェアや理論の研究のサポートを行うことが，第１の目的であるが，画像自身の意味や構造などの高度な処理を行わない応用においても，広く用いられることから，プログラミングやそのほかのデータ入出力用としての必要性も高い。

このため，色彩画像表示は，早期に開発支援を目的として，開発することが必要である。

以上述べたもののほかに，主に，開発支援用マシンに付加して用いることを目的とする，文字，音声，図形の各入出力機能を，組合せて，一つのマルチメディア用対話ステーションとした端末装置の研究開発が必要である。

ここでは，各入出力機能を，相互に関連づけ，コンパクトにまとめ，端末自身の中に，専用プロセッサを含むものが望まれる。

そして，この能力は，音声応用，画像応用の研究の進展に促して，改良されていくことが望ましい。

(3) 内外技術の現状

① 国　内　技　術

マンマシン・コミュニケーション・システムを構成する技術のうち，漢字かな混り文を対象とする文字入出力技術は，我国に固有のものであり，欧米では，あまり例が無い。この点を除く，英数字を用いる文字入出力，音声入出力，図形・画像入出力技術は，パターン情報処理プロジェクトの研究開発により，かなり進歩した。しかしながら，マンマシンの対話には，ソフトウェアの優劣が大きく影響することと，従来から我国では，ＴＳＳシステムの利用等が遅れたために，対話という使用形態が未発達であること等の理由から，実際のソフトウェアを含むマンマシン対話用のシステムの開発が十分ではない。

ハードウェア面での開発は，最近，進歩が見られ，これに伴って，東芝，富士通，シャープ等のメーカーから，日本語ワードプロセッサが発売され始めた。これらの中には，カナ漢字変換機能等を含むものも多いが，変換精度の向上の余地があるほか，それ自身コンピュータとしての機能を持たない等の問題点を有する。また，図面等の入出力との組合せ等，今後の改良，拡張が望まれる。

音声や図形・画像の入出力についての，基礎的な研究開発は，基礎応用システムの課題グループの中の，音声応用システム，および，図形・画像応用システムの研究開発課題中で述べられているので省略する。

ここでは，音声，図形・画像については，開発支援用マシンのマンマシン・インタフェースとして用いる場合に限って述べることとする。

コンピュータ用の端末機能としての音声入出力は，最近のパーソナル・コンピュータの普及に伴い，注目されてきた。音声の合成，出力については，一部，開発，市販されるまでになりつつある。しかし，実際には，電子翻訳機等の家電品への応用が先行している。

音声入力についても，コンピュータの端末用としては，まだ十分開発が行われていない。しかし，流通業や工場のラインの制御用としては，数十語～百語程度の音声入力装置も市販されている。（日本電気，日立など）

以上のように，技術的な基礎は，ほぼ確立されているので，入力可能語数の増加や認識，精度の向上などの研究をすすめることで，コンピュータの端末用音声入出力システムとしての実用化も可能と思われる。但し，入力した音声の意味の理解等も含めた高度な処理能力を持つものは，本プロジェクトの中心課題である知的な処理システムの開発を待たねばならない。

図形・画像の入出力については，本格的なものの開発は，十分行われていない。また，色彩画像の三次元の表示等の実用的レベルの研究開発も，アメリカ等に比べ，立ち遅れが目立っている。基礎的な技術のうちハードウェア面は，かなりの開発経験を蓄積しているが，使い易いユーザ・インタフェースを実現するソフトウェア面と，さらに，これをパーソナル・コンピュータなどと組合せる技術の面の研究開発が重要である。

以上の文字，音声，図形・画像の入出力機能を組合せて，コンピュータ用端末とするための関連技術開発が，本課題の重要テーマであろう。

② 海 外 技 術

海外での研究開発の中心はアメリカであるが，アメリカでは，ＴＳＳの早い普及，コンピュータグラフィックスの研究などに端を発するインタラクティブな対話処理の普及と，そのＣＡＤシステム等への応用の一般化，高精度カラー画像を用いた航空機や宇宙船のシミュレータの研究開発など，早くから，マンマシン・コミュニケーションに重点を置くハードウェア，ソフトウェアの研究が行なわれてきた。また，音声の入出力についても，マイクロコンピュータ用の簡単なシステムから，より高度なものまで，研究がすすめられている。最近では，高性能のパーソナル・コンピュータに，各種のマンマシン・コミュニケーション用のインタフェースを備え，これをネットワークで接続し，オフィス・オートメーションをはじめとする今後の応用に用いようとする試みもある。

上記の研究開発のうちでは，まず，カラー・グラフィックスについては，Evans and Sutherland社のものが有名であり，各種図形の３次元表示等の能力を持つ。濃淡色彩図形の表示とアニメーションでは，NASAの開発した宇宙船のシミュレータ用のものが有名である。

音声の入出力については，マイクロコンピュータ用のものがいくつかあるが，より本格的なものは，ＭＩＴで研究されているＭＹＴＡＬＫ（音声合成出力用）などがあり，専用ＶＬＳＩの研究と関連づけて開発が行われている。

マンマシン・コミュニケーションの機能の優れたパーソナル・コンピュータの開発は，いくつかあり，XeroxのＡＬＴＯマシン，その後のＤＲＡＤＯマシンが有名である。また，ＭＩＴで開発されたＣＡＤＲマシンも，同様の能力を有する。これらは，高精度のディスプレイと図形入力装置，キーボードを備え，きわめてよく工夫されたソフトウェアにより制御されている。ＤＲＡＤＯでは，音声の出力機能も付加される。

このような将来のコンピュータの一つのプロトタイプに属するものの開発の試みは，我国には見られず，本プロジェクトでは，まずこのようなコンピュータを研究開発ツールとして開発する必要がある。

(4) 研究開発内容

① 研究開発項目

a. 漢字を含む文字入出力装置

日本語文による対話を可能とする漢字を含む日本語文の入力，表示，印刷装置を開発し開発支援用コンピュータへ，まず，付加する。その後，文字サイズ，書体等も可変の装置を開発し，プロトタイプ・システムへ付加する。

b. 図形・画像入出力装置

まず，濃淡のないグラフや図面等の２値図形の入出力装置を開発する。解像度は，１０００×１０００点／画面程度とし，ラスタ走査方式のものを中心とする。これは，開発支援用コンピュータへ付加する。その後，より高解像度のものを目ざし，製図等の目的に使用できる表示装置や，タブレット等の座標入力装置を開発する。また，入出力の高速化，高機能化を目的とする専用ＶＬＳＩを含む専用プロセッサを開発し，付加する。

濃淡を含む一般画像の入出力に関しては，画像理解システムの課題のもとでの研究のサポートを兼ねて，与えられた仕様に基き入出力装置を開発する。これは，専用プロセッサの開発等も含まれる。

c. 音声入出力装置

音声の合成・出力に関しては，現在の技術の延長上にあり，これら技術に応じた専用ＶＬＳＩチップ等を開発し，開発支援用コンピュータ・システムへ付加する。これより高度の音声出力装置については，音声理解システムの課題のもとでの研究のサポートを主に行い，与えられた仕様に基き，ハードウェアの開発を行う。

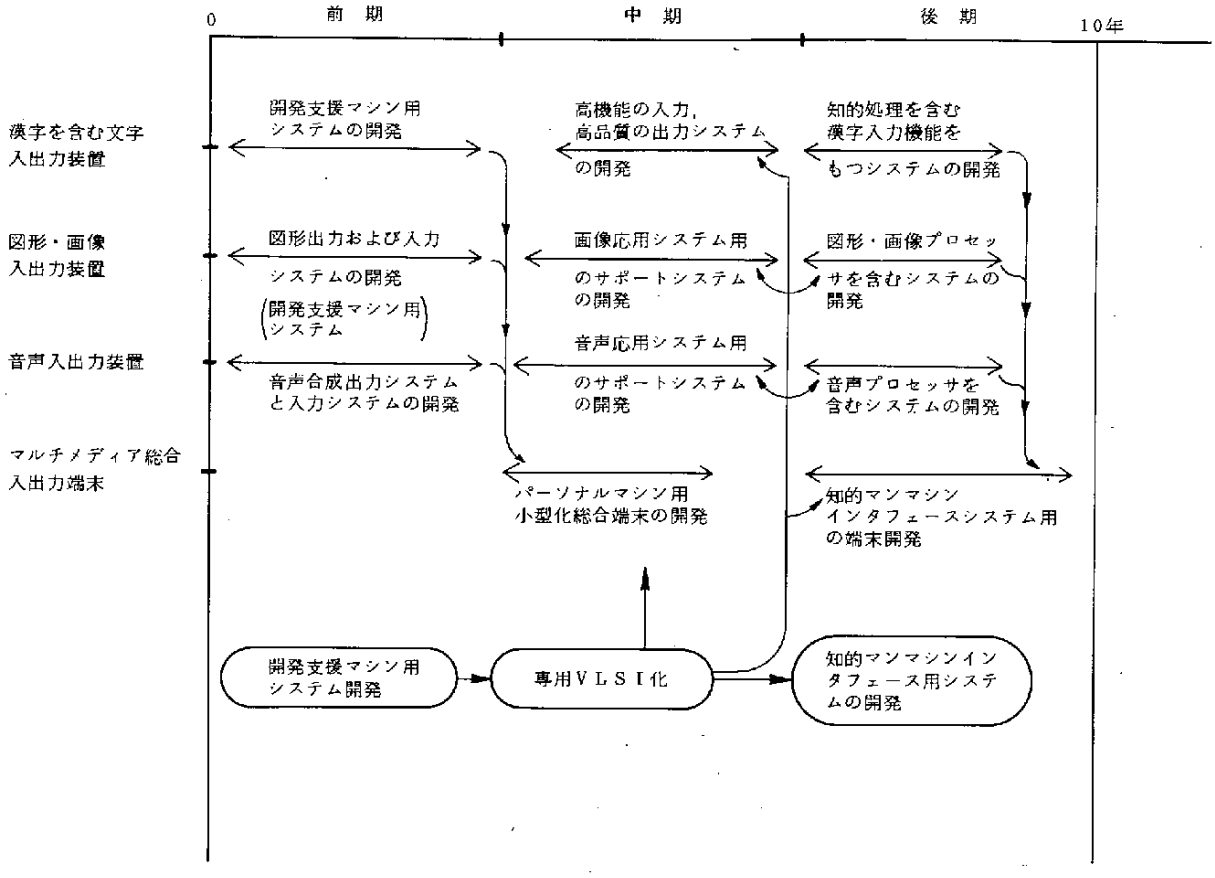
音声入力装置については，ディジタル信号処理用の専用プロセッサ等の開発を中心とするが，これは，音声理解システムの課題のもとでの研究のサポートを目的とする。

d. 文字，図形，音声の各入出力をインテグレイトしたシステムの開発

これは，開発支援を目的とし，それぞれの入出力メディアごとの研究開発状況に応じた技術に基く装置を集め，コンパクトにまとめた装置である。

研究の進展に伴って，いくつかの装置を開発し，研究開発支援コンピュータを付加する。

② 高度マンマシン・コミュニケーション・システムの研究開発スケジュール



③ 研究開発の目標・仕様

本課題の目標は，中期の初めまでに達成されるべき目標と，最後に第5世代コンピュータ・システムのプロトタイプに用いられるマンマシン・コミュニケーション・システムの最終目標の2つから成る。前者は，開発完了後，直ちに，開発支援用マシンの入出力システムとして実用に供されるものである。

後者は，基礎応用システムの課題グループに属する音声応用システム，図形・画像応用システムの課題に基く研究のハードウェア・サポートを行うとともに，最終的には，これら課題で開発される方式やソフトウェアと組合されて，知的インタフェース・システム用のマルチメディア入出力システムとしてまとめられるべきものである。

a. 漢字を含む入出力システム

- 漢字を含む文字3000～4000字種の入力機能，カナ漢字変換による入力とタッチメソッドによる入力の双方の機能を含む。
- グラフや線図形などとの混在を許す書類の編集能力を含むワード・プロセッシング機能。

- 音声出力による入力ミス警告機能つき。
- 4～5種類の異る字体を含み任意に選択できる文字表示装置と，十分な速度を有する印刷装置。高品質であること。
- 外字処理の機能を持つこと。但し，オプションでよい。

上記は，中間時点までの目標である。最終目標としては，音声入力との組合せによる漢字入力機能，意味理解を含むカナ漢字変換機能等を付加することを目ざすものとする。

b．図形・画像入出力システム

- 座標入力装置として，5000×5000～10000×10000点の精度のタブレットを備える。
- 画像入力としては，1000×1000点程度の精度を持つ濃淡色彩図形の入力機能を目ざす。
- 出力としては，1000×1000～4000×4000点の解像度をもつ表示装置を備える。
- 印刷装置としては，10000×10000点以上のものを備える。
- 表示装置は，3次元の表示等も可能な能力を付加できるようなものとする。

上記は，中間時点までの目標であり，最終目標は，図形・画像応用システムの研究課題よりの仕様に基いた，より知的な機能を有するものとする。

c．音声入出力システム

- 入力としては，500×1000語の前もって登録された単語の識別能力を有するものを目ざす。
- 出力としては，1000×3000語の単語の合成出力が可能なものとする。

これらは，中間時点までの，開発支援を目的とした目標であり，最終目標は，音声応用システムの課題で示す。意味理解を含み，自然言語も一部可能な，より知的な仕様を持つものを目ざす。

上記a～cの機能を組合せ，ＶＬＳＩ化を行い，パーソナル・コンピュータ用端末として，インテグレイトしたものである。

音声，図形の入出力等のメディア別の性能は，その時点での開発技術に応じたものとし，各メディア間の相互の効果的な組合せによる対話機能の高度化を目標とする。

この中には，プログラムの音声による入力機能，図形を含む文書の編集機能，コンピュータへのコマンドの音声や図形によれ入力，また，音声，図形による応答機能など，ソフトウェア面の機能を多く含み，マンマシン・コミュニケーションの高度化を目ざす。

4．VLSIアーキテクチャ

4. ＶＬＳＩアーキテクチャ

4.1 概　要

【１】 ＶＬＳＩの革新性

計算機誕生以来３０年経った現在，その歴史をマクロ的に見るとその進歩は驚異的であり，我々は従来人手だけでは決してできない問題の多くを計算機を使うことによって解決して来た。しかし，その様子をミクロに見るならば，進展の多くは質的変革ではなく，量的変革であると言えよう。即ち，ここ４～５年のスパンで見るなら，計算機のスペックに大きな変革があるとは思われない。構成素子が，真空管→トランジスタ→ＩＣと変わり，確かに計算機は小さくなり，速くなった。また安価にもなった。しかしながら，その機能的要素についてはほとんど変化していない。現在使われているアーキテクチャ，例えば機械語命令の多くは十年以上前に造られたものをベースとしており，またオペレーティング・システム，言語の多くについても同様である。なぜか？なぜ構成素子の進歩ほどにアーキテクチャ，ソフトウェアは進歩しないのか。ＩＣの技術が画期的であることは言うまでもないことである。その技術を集大成したマイクロ・プロセッサも一昔の半導体の技術を考えると驚くほど進歩している。しかし，よくその中身（アーキテクチャ）を見るならば，これまた一昔のアーキテクチャと全く変化していない事にも驚くのである。なぜマイクロ・プロセッサのアーキテクチャが，現在汎用機と呼ばれている計算機と同じ歴史をたどって進歩しなければならないのであろうか？多くの理由が考えられる。しかし，このような問いに対しての一番大きな理由は，結局計算機構成素子は進歩はしているのであるが，計算機アーキテクチャに変革をもたらす程は進歩していなかったということだったのである。また，このことは，計算機を今までどのようにつくってきたかを見ても明らかである。即ち，まず構成素子により，可能な大きさの論理部品を作り，それを組み合わせる事により，計算機を構成していく。そしてこの様子は，真空管→トランジスタ→ＩＣ（ＬＳＩ）と素子が進歩しても何ら変わってはいなかったのである。真空管・トランジスタ時代に，ボードに作られた物が，ＭＳＩ・ＬＳＩ時代ではそれが単にチップになったに過ぎない。マイクロ・プロセッサも同様 ── 高々数万トランジスタ／チップ程度の集積度に革新性を求めることはできない。

しかし，我々は今，集積度から言えば百万トランジスタ／チップというＶＬＳＩを手に入れるところまで来た。現在のＬＳＩの数千倍以上の集積度。半導体の理論から考えれば，確かに現在の延長線上にあろう。けれども，論理設計のレベルで考えれば，この大きな集積度の意味は大きい。特にチップの中に機能的なものが入るか，入らないかの違いは大きい。設計に対する考え方の大幅な変革が求められる。実現しようとするほとんどの論理回路がチップ内に入ってしまうことによって，従来の論理素子数を最少とするような設計法を使えず，チップ内面

積を最も占る結線部分を最少とする設計法への転換が必要となる。ＶＬＳＩにより従来実現できないとされていた機能が実現できるようになることも事実であるが，設計の方法の変革，構築方法の変換により，始めてＶＬＳＩを使い切った事となり，従来の機能を逸脱する飛躍的な計算機が登場するのである。革新的なものは，そのすべてに革新的でなければならない。そしてＶＬＳＩは十分その条件を満たしているのである。

【２】 何をする必要があるのか

前項で述べたように，ＶＬＳＩに対して革新性を求めるには，設計手法の変革がまず必要である。具体的には，アーキテクチャ構築に必要なＶＬＳＩの特質を整理し，構築したアーキテクチャが特質をどの程度利用しているのかを評価する評価法の開発などが要求されよう。また，これら設計法を設計者に教育する事も重要である。さらに，これら新しい構築法を意識したアーキテクチャ・アルゴリズムの開発が要求される。また，実際問題として，ＶＬＳＩを作り上げるには，アーキテクチャ設計のための設計法のみならず，製造レベルに沿っての新しい設計法も要求される事はもちろんである。そして，その規模からして，これら設計法に対する計算機支援は必須であり，総合的なＣＡＤが強く要求される。

4.2 ＶＬＳＩアーキテクチャ

【１】 概　要

ＶＬＳＩアーキテクチャとは，ＶＬＳＩ素子そのものの特質を十分に考慮し，それをアーキテクチャ，方式に反映させたものである。その特質とは以下の通りである。

(1) ＶＬＳＩでは，通信コスト（Communication Cost）の方が計算コスト（Computation Cost）をはるかに上回る。

(2) 少ないタイプの単純なセルによりチップを構成する方がチップ面積使用効率が良い。

(3) 例えば，バス構造のような単純・相対構成を取るより，ローカルな通信が多い構造の方がチップ面積使用効率も良く，速度も上がる。

(4) データ入出力時間をも最大限利用するには，パイプラインとマルチプロセッシング混合アプローチなどが優れている。

また，ＶＬＳＩアーキテクチャが従来のアーキテクチャと異なると思われる点を列挙すると以下のようになろう。

(1) ＶＬＳＩアーキテクチャでは，アーキテクチャの設計手法が従来に比べより重要視される。従って，何を作るかということに合わせて，どのように作るのかが同時に決定されなければならない。しかし，ＶＬＳＩアーキテクチャ構築法において有効な手段が提供された場合には，従来，製作する数に比べて設計時間がかかり過ぎるため，実現不可能とされていたようなアーキテクチャのＶＬＳＩ化が可能となる。（少量多品種のチッ

プの現実的可能性）

(2) ＶＬＳＩアルゴリズムとは，従来マルチチップにしていたために考慮しなければならなかった。アルゴリズムの本筋から離れた非本質的な部分（オーバーヘッド）をなくすことによって，革新的飛躍的に性能の向上を期待するものである。従って，同じ機能を実現するにしても全く異った手法を取る方が良いと思われる。

(3) 設計に於ける階層，ＰＭＳ構造→ＩＳＰ→ＲＴ→Logieレベルにおいて Logic レベルコンポーネントが従来は与えられていた（限定したものしか無かった）ことに比べ，ＶＬＳＩアーキテクチャ・アプローチではかなり自由度が高く，自由に階層のレベルが選べるという点と非常にスケールの大きいLogicレベルコンポーネントを使うことによって，設計における階層構造が吸収される。つまりアーキテクチャアルゴリズムが直接Logic レベルコンポネント上で，実現できるようになったことが従来の素子技術では実現可能性が全く無かったようなアーキテクチャルゴリズムを今一度検討し直す必要性を要求する。

ここでは，このような新しい可能性を持った革新的ＶＬＳＩアーキテクチャ構築の方法に対していくつかのアプローチを示す。

【２】 ＶＬＳＩアーキテクチャ・アプローチの方法

ここではＶＬＳＩ構築のアプローチを以下の三つに分ける。

(1) 応用目的指向型アーキテクチャ

前述したＶＬＳＩの特質を考慮しつつ，応用問題のアルゴリズムを構築する。例えば，ＶＬＳＩの特質の一つに，少ないタイプの単純なセルによりチップを構成する方がチップの面積使用効率が良いということがあるため，まずいくつかの上記条件を満たすセルを与え，応用問題を解法できるアルゴリズムを考案する。例えば(i)ＣＭＵのKungらはシストリック・アレイという非常に単純なセルを考え，この上で各種応用問題の構築を行っている（実例の項を見よ）が，これなどはその一例である。このアプローチではＶＬＳＩアーキテクチャ構築成功の鍵が，必ずしもトップダウンにアーキテクチャを決定することではないことを示している。実際問題，ＶＬＳＩアーキテクチャ構築において最もむずかしいのは，マスクパターンのレイアウトである。従って，マスクパターン・レイアウトをまず始めに決めてしまい，それからアーキテクチャの構築を始めると言うことが重要となろう。(ii)マスクパターン・レイアウトをよりやり易くすることを逆にデバイス側に要求することもできる。一例としてＰＬＡを挙げる。ＰＬＡはその規則的構造ゆえＶＬＳＩ向きの構造であるが，かつ設計の方法も規則的となり，デザインオートメーション化がし易い。ＰＬＡコンパイラのようなものを作成することにより，組み合わせ回路，順序回路の設計がプログラミング言語で行うことも可能となり，論理式より構造（マスクパターン）が生成されることとなる。(iii)別の例として，Bristle Block設計法を挙げる。この方法では，マスクパターン・レイアウトのし易さを考

慮し，まず始めにバス構造とブロックの種類を与え，そのブロック間を自由に接続し，任意のアーキテクチャを構築する。

(2) 汎用指向型アーキテクチャ

ＶＬＳＩにおいても大量生産により著しくコストダウンが図れるという点は従来と変わらない。従って，汎用型アーキテクチャを指向し，多くのアプリケーションに適用するようなチップを作るアプローチも依然として重要と思われる。しかし，集積密度の増大に伴い機能的には従来のものと比べ高いため，汎用のレベルを再検討する必要があり，これは大きな課題である。

(i) ダイナミック・アーキテクチャ

集積密度の増大に伴う機能向上により，ダイナミック・マイクロプログラムが可能となり，従来の汎用計算機の汎用とはＩＳＰアーキテクチャが固定で，それ以上のレベルで汎用ということであったのが，マイクロアーキテクチャ・レベルで汎用とし，それ以上で可変性を持たせることが可能となる。ここで重要なことは，ＶＬＳＩ１チップＵＨＰ（Universal Host Processor）構成法および,ＵＨＰをＩＳＰレベルで問題に適応させる適応化理論の二つである。ＵＨＰについては；従来理論的に可能であり，方式上の問題点が解明されているにもかかわらず，素子数や速度的に不可能であったものがＶＬＳＩにより可能となる。この中には例えば，バレルシフタ，ビットアドレシング機能，高度なレジデュアル制御などがあり，又，これらの機能の実現により，マイクロプログラムの最適化手法，マイクロプログラム開発の実現により，マイクロプログラムの最適化手法，マイクロプログラム開発高級言語などの再検討が要求されることも明らかである。適応化理論とは，従来のＩＳＰ機械語レベルより上のレベルに中間言語を設定し，問題解決を行う場合，どのようにして問題に適応したＩＳＰを決定していくかである。開発の現状としては，高級言語を限定し，その言語に適したＩＳＰを決めるという高級言語計算機のアプローチが現在のところ一般的である。タグやスタックの概念などが重要であることは判っているが，どのように１チップ化して行くかマイクロプログラムでどのように制御するかなどのテーマが重要である。また，今後は言語レベルよりもさらに応用問題に密着したレベルでの適応ＩＳＰの決定法，問題の性質を自動的に解析し，計算機自らが適応していくという自動チューニングの手法なども重要となろう。

(ii) 新（New）ノイマン型完全１チップ・アーキテクチャ

現在作られているメインフレームを１チップ化するという方向は現在各所で試みられているが（１チップ３７０など），この手法は過渡的な方法としては評価できる。即ち，過去に蓄積のあるソフトウェア財産の多くを利用することができるからである。しかしこの方法はあくまで過渡的なものであり，第５世代コンピュータが実現されるといわれている１９９０年代におけ

る重要性は高くはないと思われる。それよりも，現在広く行き渡っているノイマン型ベースではあるが，その欠点をなくしたときでのＶＬＳＩ　１チップ＝　Newノイマン型の研究を重視すべきであろう。（詳しくは　2.2.6 新（New）ノイマン型アーキテクチャ　参照）

(iii)　メモリ

メモリ・アーキテクチャ においては，ＶＬＳＩ化により大規模メモリの開発が可能となるため，ロジカルレベルでは，巨大論理空間のアクセス法（例えば，３２ビットをアドレス指標とした場合，仮想空間として４Ｇがアクセスできる。）の開発が重要となる。他方，ハードウェア的には，メモリ読み出し幅の増大マルチポートメモリなどの実現が可能となる。また，ＦＩＦＯ，キュー，連想処理などの機能をメモリに付加することが可能となる。いわゆる機能メモリの実現が可能となり，これらは，第5世代コンピュータ構築において重要な要素コンポーネントとなるであろう。

(3)　ＶＬＳＩマルチチップ

これは，ＶＬＳＩを組み合わせて，さらに大きな一つのプロセッサを構築する場合のアプローチ法である。即ちＶＬＳＩで1チップ構成が可能となってもさらにＶＬＳＩを組み合わせて超高速処理をねらう必要があるからである。問題としては，ゲート数に対して必要な接続ピン数をどのように用意するかと，その間をどう接続するかである。

【３】　内外技術の現状

ＶＬＳＩアーキテクチャはまだ始まったばかりである。よって，世界レベルでその研究は盛んに行われようとしている。しかるに，特に重要と思われる研究開発の多くは米国に於て行われている。我が国では，ＶＬＳＩ製造工程における研究で一部行われているものもあるが，最も重要と思われるアーキテクチャ・ＣＡＤにおいては十分な研究が行われているとは思われない。ＶＬＳＩは今後の計算機構築に当っては必要不可欠なものであると考えると，現在のままでは１９９０年を待たずとも，米国からの技術導入を行わなければならなくなる事は必須であり，研究開始は緊急を要すると考えられる。以下，いくつかの重要と思われる研究につきコメントする。

(1)　ＣＭＵのKungのSystolicアルゴリズム

KungのＶＬＳＩアーキテクチャ構築法では，まず図４－１に示すようなセルをつなぐ方法をいくつか用意する。次に，解こうとするアルゴリズムを実現するためにはこのセルではどのような計算をする必要があるかと，データの入出力法を決めていく（これがアルゴリズムとなる）。セルの接続法は，もともとＶＬＳＩの特質を最大に利用するよう決めてあるため，アルゴリズムが決まるとこれがＶＬＳＩにとっては最良のものとなる。Kungらはこの方法により，マトリクス計算，ＦＦＴを始め，データベース・マシンのコンポーネントに渡る広範囲なＶＬＳＩアルゴリズムの構築研究を行っている。

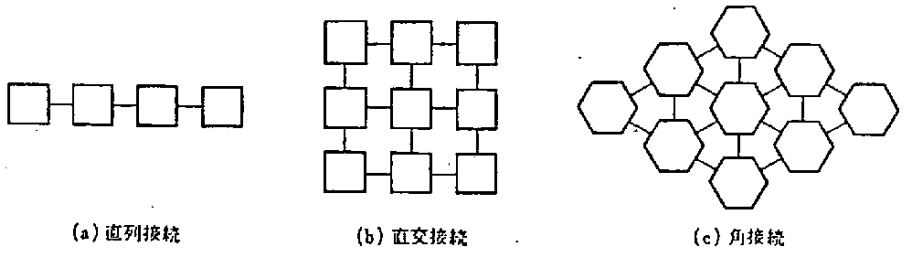


図４－１　"心臓"アレイ（可能な接続の例）

(2)　ＭＩＴの知的ＣＡＤ

ＭＩＴの人工知能研究所では，知的ＣＡＤの研究を行っている。これは，目的指向型のＶＬＳＩチップをつくりだすための基礎研究であるが，設計ルール・ノウハウを知らなくても設計ができる事をねらいとしている。既に，この手法の一部を使いＬＩＳＰ専用ＬＳＩチップを数か月でつくりだすことに成功している。

(3)　３２ビット・プロセッサ

１９８１年２月に開催されたＩＳＳＣＣ（International Solid-State Circuits Conference），ＣＯＭＰＣＯＮなどで発表されたＶＬＳＩプロセッサを表４－１にまとめる。集積密度としては，ＨＰ社の４５万トランジスタ／チップＶＬＳＩが注目される。また，ベル研究所のＣＡＤは非常に強力で，ＣＡＤデータ・ベースのデータ利用により，設計パラメータの変更に対してもある程度はのりきる事ができる。表４－１に示すＶＬＳＩはすべてＣＡＤで作られていることも注目すべきで，ほとんど数か月のオーダーでチップが完成している。

表４－１　ＶＬＳＩ３２ビットマイクロ・プロセッサ開発の現状（１９８１年２月）

	HP	BELL MAC 32	INTEL APX 432	NS 16000
チップ構成	CPU	CPU	GDP 43201 43202 IP 43203	CPU 16032 MEMORY MANAGE 16082
構成素子数	45万	10万	11万（43201） 4.9万（43202） 6万（43203）	
プロセス	2.5μmNMOS ↓ 1.5μmNMOS	3.5μmCMOS ↓ 1.5μmCMOS	HMOS	
マイクロサイクル	55ns	31ns	125ns	100ns
マイクロROM	9K×38bit		4K×16bit（GDP） 2K×16bit（IP）	1.3K×18bit
アーキテクチャ	スタックマシン	ノイマン型 高級言語指向	ADAマシン （スタック）	ノイマン型 高級言語指向
高信頼性化	セルフテスト		チェッカーモード	
その他	DEBUGGING AID	CAD DATABASE	CAD	

※　以下の項目は全社のものが満たしている。

(1)　パイプライン制御　(2)　高級言語サポート　(3)　$\overline{\text{O}}$Sサポート　(4)　バーチャル・メモリ

【４】　ＶＬＳＩアーキテクチャ研究開発内容

(1)　概　要

①　新アーキテクチャ構築技法（基礎研究）

前述したようにＶＬＳＩアーキテクチャにおいて，アーキテクチャの設計の方法を変えるということが重要であり，その方法論を構築する研究を早急に始める必要がある。具体的には，

(i)　ＶＬＳＩデザインにおいても最も困難とされるマスクパターンを容易にするため，パターンにある程度枠組を加え，論理設計を進めるというような，いわゆるトップダウンでもボトムアップでもない設計手法の開発

(ii)　ＶＬＳＩアルゴリズム

ＶＬＳＩの特質をアルゴリズム解法のレベルより考慮するような手法の開発

(iii) 計算機支援を前提とした開発手法

② ＶＬＳＩアーキテクチャ

①にもとづく具体的なアーキテクチャ。これは，大別して１チップＶＬＳＩアーキテクチャと計算機構成パーツとしてのＶＬＳＩアーキテクチャがある。

③ ＶＬＳＩチップシステム構成法

個々に作られたスペシャルチップ（構成パーツも含む）のチップ間を接続して，システムを構成する技法。これは機能分散アーキテクチャの基礎的研究にも相当するが，どのように機能分散させるのか（チップ化していくか）を広いレベルで決定するアルゴリズムと具体的なチップ構成法を明らかにすることが重要である。

(2) 開発項目と必要な機能

開発概要にもとづき，項目を列挙すれば，以下のとおり

① ＶＬＳＩ基礎研究

(i) ＶＬＳＩデザイン・ルール・ブック

新アーキテクチャ開発設計法をルール・ブックとしてまとめたもの。セルフ・スタンディング法により作られている必要があり，関係学部を卒業した学卒の人間が６か月以内にマスターできる（実習を含む）必要がある。

(ii) 設計Ｑ・Ａシステム

設計に就いてのＱ・Ａを計算機を介して行うシステム。解は，始め専門家が入れるが，二回目からは，システムに用意されるＱ・Ａデータベースにより自動回答される。またこれは，将来の知的ＣＡＤのためにも使われる。

(iii) アーキテクチャ・データベース

応用別のコンピュータをＶＬＳＩを使って作るためのデータベース，アーキテクチャの項目分類がなされており，応用指定により最適アーキテクチャの指定が行われる機能が必要である。（なお 2.2.6 新（New）ノイマン・アーキテクチャの項も参照せよ）

(iv) ＶＬＳＩアーキテクチャのためのＣＡＤ

アーキテクチャ記述言語，ＶＬＳＩマスク・デザイン言語を備えている事が必要。さらにアーキテクチャ・データベース，設計Ｑ・Ａシステム，製造用ＣＡＤと連結する機能が必要である。

② ＶＬＳＩアーキテクチャ

百万トランジスタ／チップ・アーキテクチャ，千万トランジスタ／チップ・アーキテクチャは，それぞれ別に考える必要がある。開発項目としては，

(v) 完全１チップ・アーキテクチャ

(vi)　機能パーツ・アーキテクチャ

の二つがある。両者とも必要な機能をどのように入れるのかの他，自己診断機能や，自己故障修復機能を備えることは必要である。

③　ＶＬＳＩシステム

ＶＬＳＩチップをつなぎ合わせて作られたシステムで，これには色々なものが考えられるが，知的マシン・アーキテクチャが第５世代の目的からしても最も適当である。

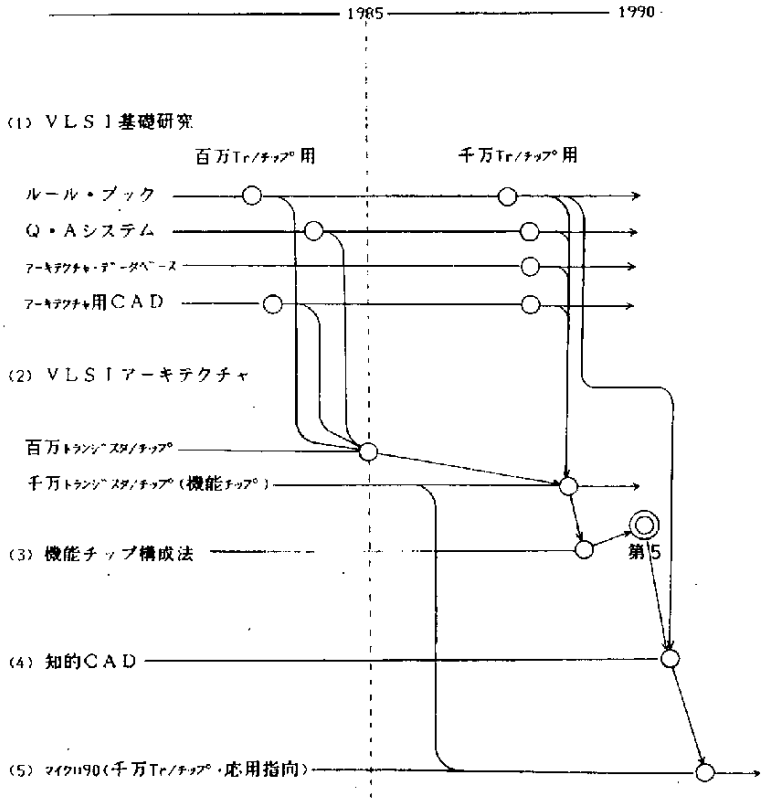
(vii)　機能分割，接続法

そのための基礎研究で(vi)と密接な関係がある。

(3)　第５世代での位置付けとストーリー

第５世代プロジェクトでの開発ストーリーは，以下のようになろう。

まず，基礎研究は早急に行う必要があり，特にルール・ブックはプロジェクト開始後一年以内に完成すべきである。その他の基礎研究は，まず百万トランジスタ／チップに焦点を絞った後は,千万へと第５世代プロジェクト全般に渡り継続させるべきである。次に，アーキテクチャではまず百万トランジスタ／チップに焦点を絞り，これの中身は New ノイマン・アーキテクチャとし，開発システムとする。さらに千万版を構築するが，これはまず第１に第５本体（スーパー・ロジック・マシン）のためとすべきである。そして，そのためのチップ接続法が重要となる。また，第２にアーキテクチャ・データベースを使い，千万版の応用指向チップ開発（マイクロ９０）を行う。



4.3 ＶＬＳＩ－ＣＡＤ

【１】 概　要

今後１０年以内にＶＬＳＩの規模は論理系で数１０万ゲートに達することが予想される。この状況はちょうど数１０メガバイト付きコンピュータを与えられたプログラマに類似している。プログラマ達はそれに対して，アセンブラ言語，高水準言語，ソフトウェア工学に基づくプログラミング技術等の開発を行って進歩してきた。これと同様にＶＬＳＩの集積度が今日のように加速度的に進歩すると，新たにハードウェア危機あるいはＶＬＳＩ危機というものが出現する可能性がある。ソフトウェア危機以上に困るのは，対象がハードである点で，ＶＬＳＩの設計ミスをソフトウェアのバグと同様に考えることができない所に問題の重大性が潜んでいる。この困難を解決するのがＶＬＳＩ－ＣＡＤ技術であり，その理想はＶＬＳＩの機能，性能，構成を実際にチップを作成しないで，全く相似形でソフトウェア・シミュレーションによって実現することといえる。これが可能になれば，ＶＬＳＩ危機はソフトウェア危機に転化され，コンピュータ・アーキテクトや論理回路設計者は,もはや製造工数を考える必要がなく失敗を恐れず新しいＶＬＳＩの設計が可能になる。特に新しいアーキテクチャの実験評価としてＶＬＳＩ－ＣＡＤは必須の道具であり，ゲート数，タイミング，性能などの情報がＶＬＳＩを製造しなくてもプログラムによって可能になり，コンピュータ・アーキテクチャの進歩に非常に寄与するであろう。従来，大学や研究所では提案するだけで終っていたシステムが現実味をおびてくることも予想される。

ＶＬＳＩ－ＣＡＤは基本的にはこれ迄の装置設計とＬＳＩ設計技術の総合化であり，従来の設計手法と全く異るわけではない。しかし大規模論理回路が対象であるから階層化，モジュール化した設計手法が必要である。また論理記述も高レベル記述言語が使われる傾向がある。これは大規模ソフトウェアの設計と非常に類似している。データベースも重要な要素である。ＶＬＳＩ－ＣＡＤのデータベースは，ＣＡＤ工程の一貫管理に不可欠であり，ＣＡＤのオンライン化，ＴＳＳ化とも密接に関係する。データベースの情報には，論理データ等のデジタル情報だけでなく，回路図やパターン情報等を扱える図形データベースも必要である。また知的ＣＡＤという面からみると，アーキテクチャ・データベースと呼ばれるものに各種の設計情報をライブラリ化することが重要である。以上のような総合的一貫システムを効率よく運用するためには，設計環境の整備が必要になる。設計し易い環境としては，オンラインＴＳＳで端末として高性能グラフィック・ターミナルがパーソナルに使えるのが望ましい。ＣＡＤの普及については，ＣＡＤのインタフェース仕様の標準化，特に記述言語の標準化が必要となる。さらにネットワークによるサービスも必要である。ＶＬＳＩ－ＣＡＤでの別の問題として，わずかな外部インタフェースを介して巨大なブラック・ボックスを閉じ込めることによる故障診断技術がある。このためにテスト容易化設計法を，アーキテクチャとも関係して研究する必要が

ある。

ＶＬＳＩ－ＣＡＤは単にＶＬＳＩ設計支援にとどまらず，コンピュータ・アーキテクチャと一体であるという考え方もある。実際，もしコンピュータ記述言語の研究が進み，コンピュータ・アーキテクチャのデータベースが蓄積されれば，コンピュータは内包的に自己のアーキテクチャおよびＶＬＳＩ－ＣＡＤを保有することが可能になり，アーキテクチャのダイナミック性や自己増殖といった概念と関係して来る。ＶＬＳＩ－ＣＡＤを能動的に把えることは興味深い問題であろう。

ＶＬＳＩ－ＣＡＤの研究は多くの問題をかかえており，それらは簡単に普遍的な解がみつかるものではなく，デバイスの進歩，設計の規模にあわせて少しずつ変化しており，あくまでも第１の目的は，最小のマンパワーで，短期的に品質のよいＶＬＳＩ設計を可能にし，生産性を向上することである。

【２】 必要性

ＶＬＳＩ製造における工数は，数１０万ゲートで１００人年以上を必要としており，今後とも増大傾向をもっている。図４－２は現在のＬＳＩ設計のフローを示しているが各工程は一貫しておらず，運用を複雑化している。また論理シミュレーションや配置・配線の入力データ作成は回路規模に指数的に比例して厖大な数になっており，ケアレスミスも増大して行く傾向にある。従って工程を一貫管理し，入力データの自動化，高水準化を確立する必要がある。

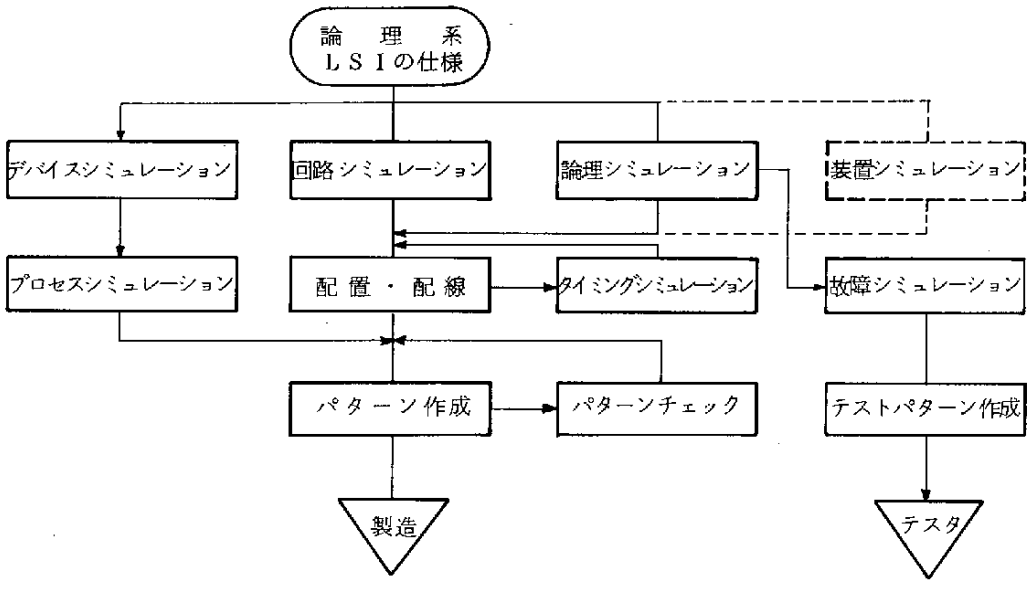


図４－２　論理系ＬＳＩの設計フロー

多品種少量，量産品ともＶＬＳＩ化の傾向が増大するが，多様化するニーズに対しては生産性向上策を考慮しなければ，ＶＬＳＩ危機の可能性があり，設計技術者の生産性を高めるＶＬＳＩ－ＣＡＤが必要である。このためのＣＡＤとしては，使い易さが大切な条件である。

現在ＣＡＤは各メーカ内部で開発されており，ユーザの仕様に基づいてＶＬＳＩを製造することは，１部のマスタスライスを除いて商業化されていない。しかし，ＶＬＳＩ化の要求は，コンピュータ・アーキテクチャに関係する者にとっては強く，特に大学や研究所で顕著である。このような外部の者が共通に利用できるＶＬＳＩ－ＣＡＤは今後商業的にも出現しようが，そのためには，利用言語の設定やインタフェースを統一して，ＣＡＤの標準化作業が必要である。

ＣＡＤの歴史をみるとそれはその時代の素子の技術，実装技術に依存しているのがわかる。集積度が拡大したり，素子がかわれば，従来のＣＡＤを利用することは不可能になっている。これはＣＡＤの進歩に較べて，デバイスの進歩が急激だったからで，なるべく素子に依存しないで，ＶＬＳＩ技術の進歩に耐えられるＣＡＤを開発する必要がある。

ＶＬＳＩ－ＣＡＤは単にチップ製造の道具ではなく，アーキテクチャ評価のために非常に重要な技術である。第５世代のアーキテクチャとして多くの新アーキテクチャが提案されているが（「2.新アーキテクチャ」参照），それらのシミュレーションをＶＬＳＩ－ＣＡＤで実現可能ならばアーキテクチャの進歩に貢献しよう。このためには，ハードウエアのプロトタイプ・シミュレーションと同等機能をシミュレートできるＣＡＤを開発する必要がある。

【３】 内外技術の現状

(1) 国内技術

(i) ＶＬＳＩ論理設計

従来，ゲートレベル，パッケージレベル，装置レベルと別々に作成されていたが，最近のＬＳＩ規模の拡大により，それらを統合し，混合して使用できるシステムが進んでいる。既に大型コンピュータの設計は回路規模では数１０万ゲートになっており，論理記述言語の開発は各メーカともＲＴ（レジスタ・トランスファ)レベルで行っている。ゲートレベルは，トランスレータで行うもの，同一言語で記述できるもの，別の言語で記述するもの等さまざまである。ただし，ここで使用されるＬＳＩは数百～数千ゲートが中心で，このレベルではハードウェア・シミュレーションを必要としない程進歩しているが，ＶＬＳＩに対しては，使用する計算機の記憶容量，処理時間から問題が多い。上記混合型はあくまでもデジタル・システムの設計支援であり，ＶＬＳＩ－ＣＡＤでの階層的アプローチはまだ研究段階である。

(ii) ＶＬＳＩ実装設計

この部分のＣＡＤは配置配線，パターン・チェックがある。これらは使用する用途により各種の設計手法がある（図４－３）。ＬＳＩレベルでは比較的ＣＡＤが自動化されているマスタスライスやビルディング・ブロックが多用されているが，ＶＬＳＩになるにはチップ面積の関係からカスタム化に向わざるを得なく，この方向のＣＡＤは余り進んでいない。パターン図と論理回路図のチェックはかなり進んでおり，パターン図上での設計ルー

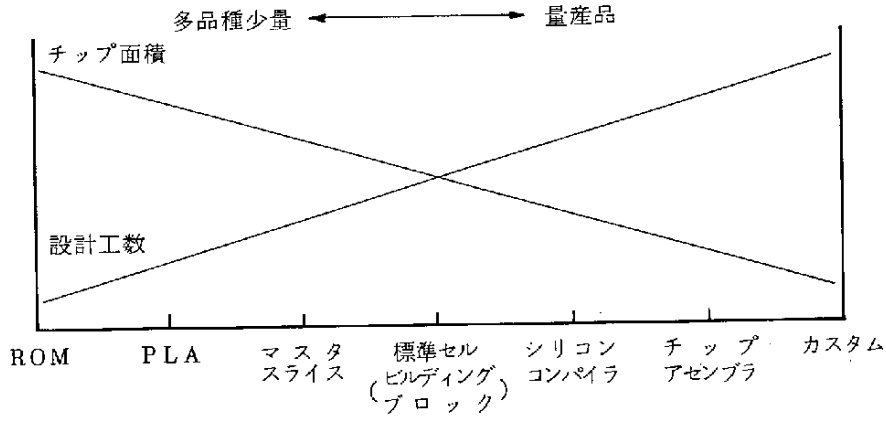


図４－３　ＬＳＩ設計方法と特徴

ルの自動チェックやパターン図から回路図を出力し，接続チェックを行うものも開発されている。

(iii)　テスト容易化設計法

このアプローチには，テストパターン生成プログラムを強化することやＬＳＳＤ（Level Sensitive Scan Design）が現在採用されている。ＬＳＳＤは数％のオーバヘッドでスキャンパスを組込んでいる。しかしＭＯＳ系ＬＳＩでは，効率が悪い。ＶＬＳＩでは両方向以外のアプローチが必要であろうという認識もある。

(iv)　ＣＡＤ使用環境

ＣＡＤ運用の指令はＴＳＳベースにより行われている。ホストは大型機が多い。レイアウト・パターンの修正は高精度のグラフィック・ディスプレイでインタラクティブに行っている。データベースを介したＣＡＤ工程の一貫化はＬＳＩレベルではかなり進んでおり，回路設計，実装設計以降はほとんど統合されている。ＶＬＳＩに対しては，現在開発中のものが多い。

(2)　海外技術

米国においては先駆的研究が多い。米国のＣＡＤ推進者は，コンピュータ・メーカだけではなく，大学，研究所，ＣＡＤメーカでも活発に進められている。特に大学では盛んで，ＩＣ製造設備を有したり，ネットワークでメーカと直結している。図４－４にＣＡＤ開発状況を示す。その中で我国では余り試みられていないものとして，チップアセンブラやシリコンコンパイラによるＶＬＳＩチップ構築法，ＭＩＴで行われている回路情報を知識ベース化したものがある。これらの技術は，ＶＬＳＩ－ＣＡＤにおいて必須のものと予想され日本での遅れがめだつ。

その他にはテスト容易化設計法としてＬＳＳＤがＩＢＭのＳ／３８のＬＳＩに使われて，９９％の故障検出率を得ている。ＬＳＩ工程の統合化はベル研究所で報告されており，３５

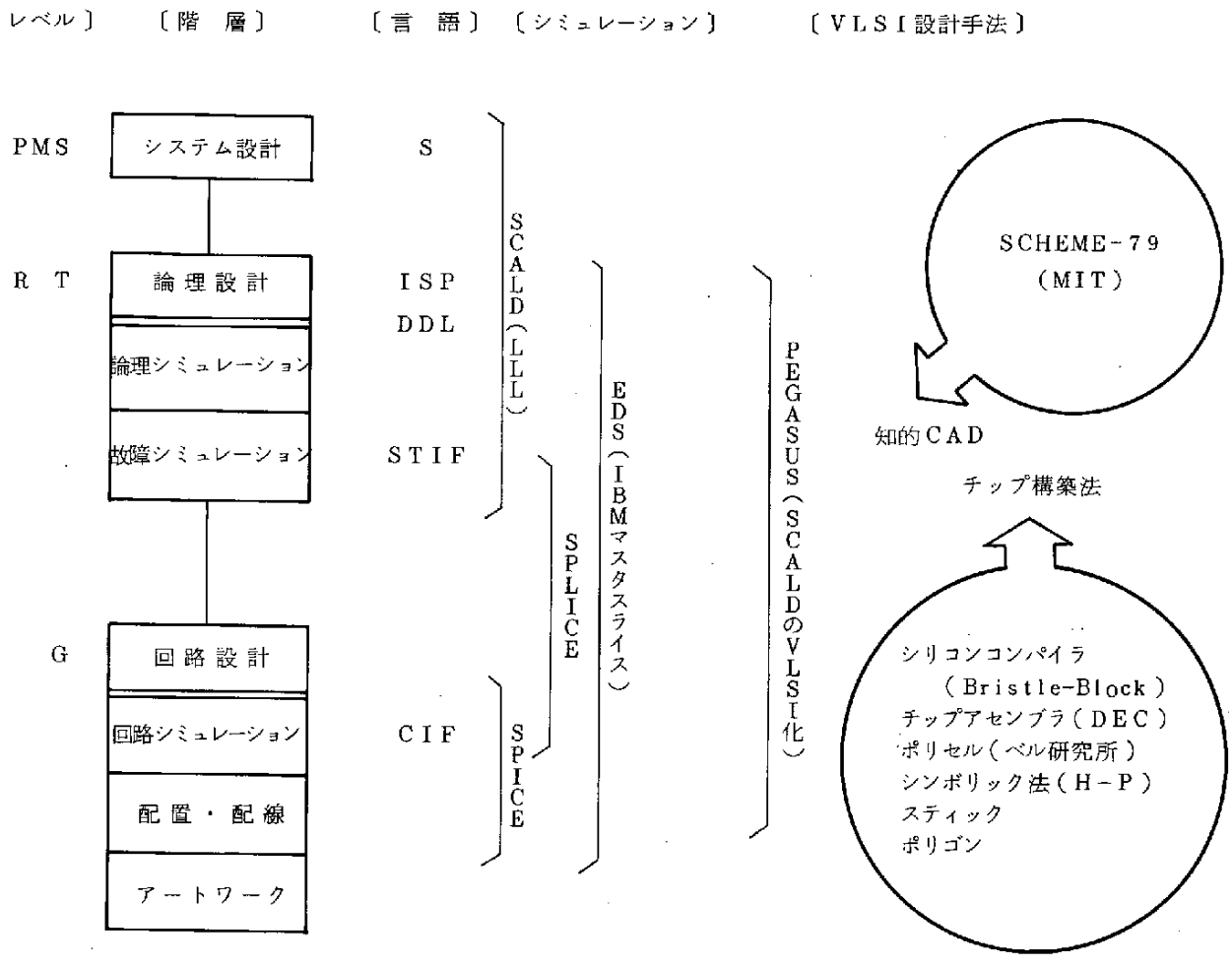


図４－４　米国におけるＶＬＳＩ－ＣＡＤ開発状況

Ｋトランジスタ回路を１人年に完成している。知識情報処理を設計の初期の段階から製造部分まで導入しようとする知的ＣＡＤシステムがＭＩＴの人工知能研究所で試みられている。

【４】 研究開発内容

(1) 研究開発項目

①. ＣＡＤ標準言語

ＣＡＤ資産の共同利用を可能にするためＣＡＤ言語の標準化を行う。

ＣＡＤ言語はシステム・レベル→レジスタ・トランスファ・レベル→ゲート・レベル→パターン・レベルと階層化され，それぞれのレベル毎に標準言語を設定し，高位レベルは低位レベルの言語にコンパイルできるものとする。従って，ユーザが任意のレベルで記述

したものが，標準インタフェースで混合できるが，ソフトウエアの言語と同様に，高位レベルの言語サポートを強化し，デバイス独立なＣＡＤをめざす。各種のデバイス特性に適応するためには，デバイス従続部は低位レベルで記述し，高位レベルとリンクすることによって汎用性をもたせる。

②　ＣＡＤ用統合データベース（アーキテクチャ・データベース）

ＶＬＳＩ開発を統括し，各工程を一貫管理するため統合データベースを開発する。ＶＬＳＩ－ＣＡＤに要求されるデータベースは，多数の端末から共有データベースに会話型でアクセスされること，記憶情報がＣＡＤの各レベルに対応して階層化されていること，データベース変更の頻度が多いこと，数値情報以外に図形情報をもっていること等の特徴がある。

③　テスト容易化設計法

ピン数制限下でのＶＬＳＩテストは大きな問題であり，解決手段としてＶＬＳＩに適したＬＳＳＤ方式，回路を２重化，３重化したテスト方式，チップ内に故障検出診断回路を組込むＢＩＴ（Built in test），ＢＩＤ（Built in diagnositics）方式等を開発する。

④　ＣＡＤ専用ハードウェア

ＶＬＳＩ－ＣＡＤを高速に実行するハードウエアを開発する。このアーキテクチャは①項でのＣＡＤ言語を高速に処理するものであり，新アーキテクチャで開発される技術を応用する。技術課題としては，高性能シミュレーション・マシン，高性能図形プロセッサがある。②項のＣＡＤ用データベースを高速化するためにデータベース・マシンを開発する。図形情報用に光ディスク等の大容量ファイルを開発する。設計情報の入出力装置としては，高性能カラー・グラフィック・ディスプレイ，高速グラフィック・プリンタ，図面読取り装置等を開発する。

ＣＡＤ有効利用のためにネットワークによる会話型処理を開発する。開発要素としては光ファイバーを用いた高速データ・ハイウェイ，ローカル・ネットワーク，ローカル・データベースと高性能グラフィック・ディスプレイを有するパーソナル・ステーション等がある。

⑤　知的ＣＡＤシステム

ＶＬＳＩ－ＣＡＤの生産性を向上させるため以下の各項目を開発する。

(i)　高度なアルゴリズム

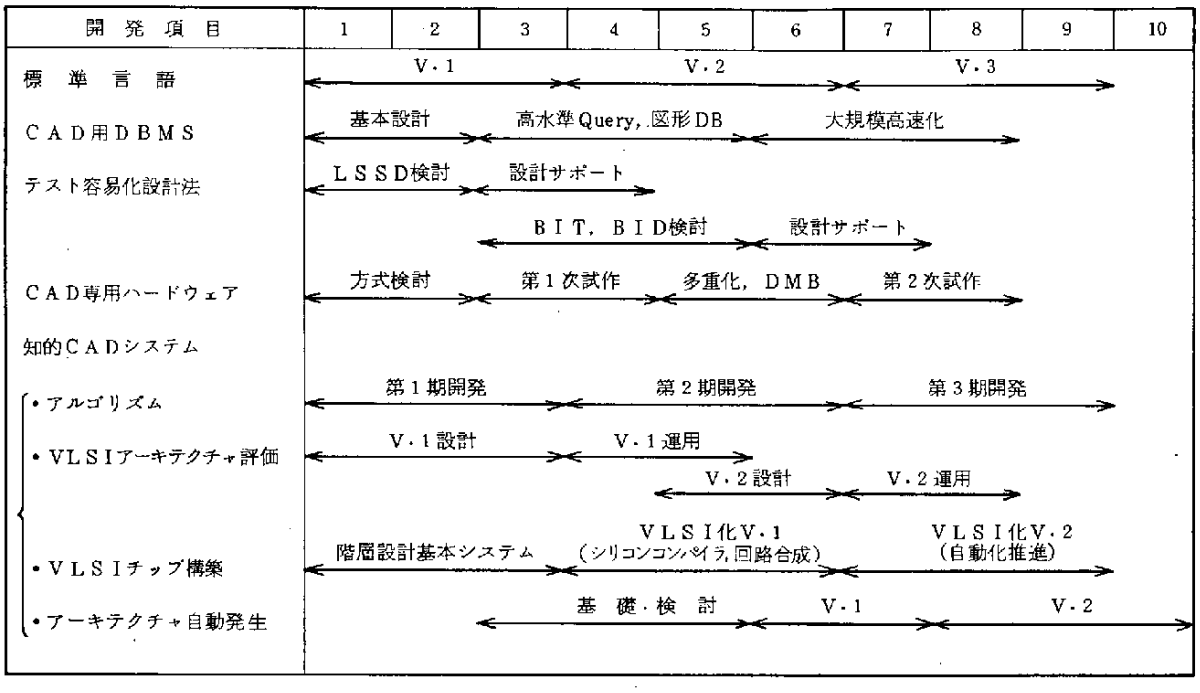
ＣＡＤの自動化を進めるため，テスト・パターン自動発生のアルゴリズム，各種アルゴリズムを組合せ，最短の配線長，最小の面積解を与える発見的プログラミング技法，パターンを設計ルールに従って検査し，パターン図面から回路図に逆変換し，接続チェ

ックを行う図形理解検査システム等を開発する。これらはライブラリ化して②のデータベースに格納する。

(ii) ＶＬＳＩアーキテクチャ評価用ＣＡＤ

①で開発した標準言語で記述されたＶＬＳＩアーキテクチャを評価するため，各種設計手法によるゲート数，タイミン　，チップ面積，ピン数，処理能力を概算し，ＶＬＳＩ化の適否を判断できるようなシステムを開発する。

(2) 研究開発スケジュール



(3) 研究開発の目標・仕様

① 対象とするＶＬＳＩは論理系で数１０万ゲート，記憶系でその１０倍位を想定して開発する。

② ＣＡＤマシンの性能は，処理能力：数１００～数１０００ＭＩＰＳ，記憶容量：数100MB/ユーザ，転送能力：数100MB／secを目標とする。

③ カラー・グラフィックの性能は，画素数：４０９６×４０９６，ディスプレイ速度：10万ベクトル／sec，転送速度：数10MB／secを目標とする。

④ 設計工数として，数人年／１０万ゲートを目標とする。

⑤ 量産ＶＬＳＩ，カスタムＶＬＳＩのいずれにも適用できることとする。

⑥ ＶＬＳＩだけでなく，大型コンピュータ等の大規模システムの設計にも使えることとする。

⑦ 使用するハードウェア（計算機，Ｉ／Ｏ）に従属しないものを開発し，広く流用で

きるものとする。

⑧ 新しい設計手法，新しいデバイス技術に迅速に対応できるオープン・エンデッドなシステムとする。

⑨ 設計技術者がＣＡＤをブラックボックスと考え，必要な情報を人間に分りやすい形式で入力できるようなシステムとする。

⑩ 言語，データ形式，入出力インタフェース，ネットワーク・プロトコール等を標準化し，アーキテクチャの情報交換を促進し，ＶＬＳＩアーキテクチャの同時分散的開発を可能にする。

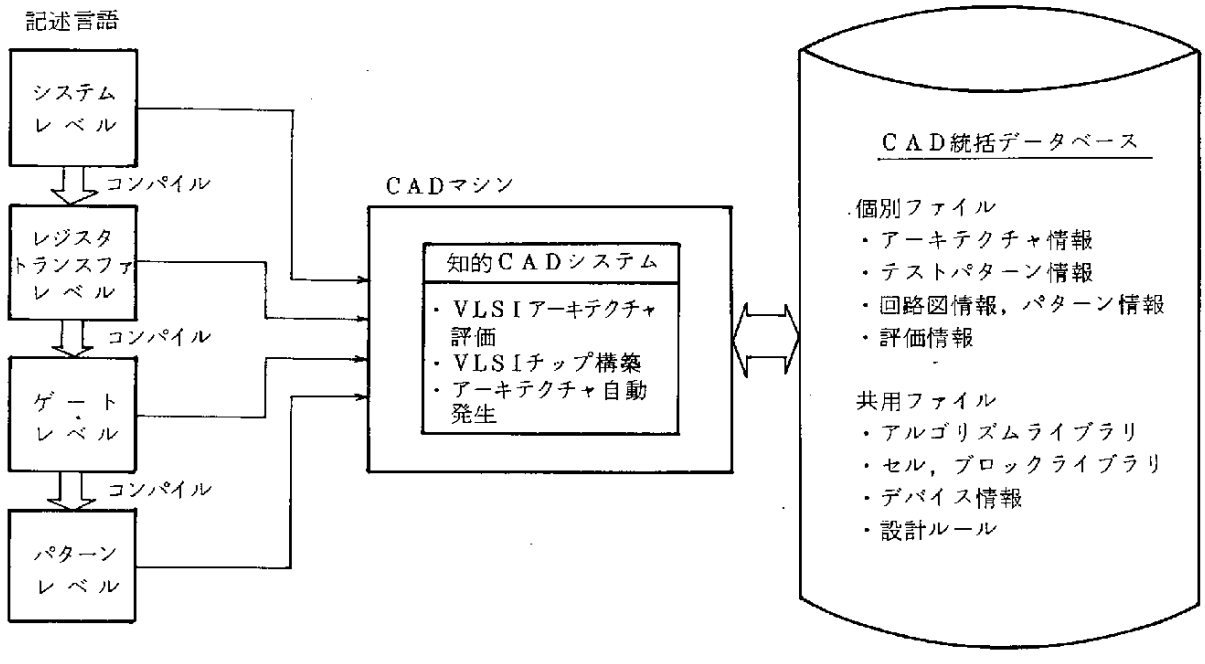


図４－５　知的ＣＡＤの概念図

4.4　新素子

将来の計算機アーキテクチャの基盤を支えるものとして，素子技術は極めて重要な位置を占める。将来有望視されている素子技術としては，

(1)　シリコンを素材とするＶＬＳＩ素子

(2)　化合物半導体素子

(3)　ジョセフソン接合素子

(4)　光関連技術

(5)　磁性関連技術

などである。これらの概要について以下に述べるが，いずれの場合もＶＬＳＩ技術が適用できなければ，実用性の観点からはほとんど意味をもたないことに注目すべきである。

4.4.1　シリコンＶＬＳＩ素子

シリコン（Si）半導体技術は着実な進歩を示しており，将来の計算機構成素子としても最も信頼性の高い中心技術である。論理回路のＶＬＳＩ化研究も著しい成果を示しており，１９８５年頃には１００万トランジスタ，１９９０年頃には1,000万トランジスタが１チップ上に実装されるものと予測されている。

一般にＶＬＳＩ技術を発展させるためには，

(i)　電子ビームあるいはＸ線による微細加工技術の開発：１～0.２５μmのリソグラフィ

(ii)　欠陥数の少ない精密な大型半導体製造プロセス：欠陥数４個／cm^{2}以下のプロセス，５～10インチの大型ウエファなど

(iii)　単純構造論理回路と多重製造チップの開発：ＮＭＯＳのスケーリング，１０～３０層の多層配線，100～200個のピン数，放熱と冷却技術など

(iv)　注文設計を容易にするＣＡＤシステムの開発：回路設計，論理設計，マスクパターン設計，検証，製造，検査に関連する総合的なＣＡＤシステム

(v)　ＶＬＳＩ向きアーキテクチャの考案：ＶＬＳＩアルゴリズム，論理記憶など

が重要な課題である。シリコンＶＬＳＩ素子は今後１０年間は主流を占めるものと考えられるが，将来にわたって，多様化する機能要求に対し，少量多品種化を効果的に実現する設計・製造・検査技術が最も重要になると考えられる。

4.4.2　化合物半導体素子

ＧａＡｓ（ガリウム砒素）ＩｎＳｂ（インジウムアンチモン）などの化合物半導体は将来のＳｉ（シリコン）に比らべ電子の易動度が高く，エネルギー・ギャップが大きい。したがって，高速のスイッチング速度と温度に対する安定性の大幅な改善が期待できる。実際に，スイチング速度はＧａＡｓでＳｉの７倍のものが得られ，特殊な高速論理回路には既に利用されている。特に，ＧａＡｓ素子は動作の安定もよく，実用が進んでおり，間違いなく１９８０年代の新素子として脚光をあびるものと思われる。し

かしながら，化合物半導体は高速であるが，電力消費が大きいのが欠点で，Ｓｉ ほど大規模なＬＳＩ化は期待できそうもない。また，現在の技術的問題点は良質で均質な結晶，安定の絶縁膜，効率的な放熱方式の開発が実用化への課題となっている。

4.4.3 ジョセフソン接合（ＪＪ）素子

ＪＪ素子は超電導現象を利用した極めて優れたスイッチング特性を有している。スイッチング速度は１０ｐｓ，消費電力は２μｗ以下が期待でき，遅延電力積で現在のシリコンの２桁以上改善されている。また，超電導状態で電力消費が極めて小さく，しかもＶＬＳＩ技術がそのまま適用できそうな点が特異な性質で，新しい計算機アーキテクチャをもたらす要因にもなっている。実用性の意味で，シリコン素子に次ぐデバイスとして注目をあびており，活発な研究が進められている。ＪＪ素子に課せられた技術的な問題点を挙げれば，

(i) 物性・材料面での改良，加工技術の開発

(ii) 安定な素子，回路の製造技術の開発

(iii) 高集積化技術の適用

(iv) 冷却，実装，外部インタフェース技術の開発

(v) ＪＪ素子向き論理構造，アルゴリズム，アーキテクチャの考案

(vi) 特殊環境下の検査・保守方法の確立

などであろう。いずれにしても，安定したデバイスの開発が先択問題といえそうである。

4.4.4 光関連技術

光技術を用いたデバイスで将来の計算機に大きな影響を与えそうなものは光通信とファイル・デバイスとしての光メモリと思われる。光通信は光ファイバーを用いたデータ伝送システムに活用されており，特に高速ローカル計算機ネットワークの主流を占めるものと予想される。光ファイバーの特徴は高速，広帯域，低損失，低電力消費の他，相互干渉（雑音誘導）など，安全で実装が容易，しかも低価格で省資源化に貢献するといった点にある。一方，現在の問題点は高速大電力の伝送が困難で直接分岐が不可能に近く，使用法に限界がある。また，効率のよい電気・光変換装置や丈夫で取扱い易いコネクタの開発などに若干の問題点が残されているようである。

光メモリは記憶密度の高さやホログラフィのような光学的演算機能などの面で注目されている。実際に，磁気記録に比らべ，記憶密度は１～２桁高くなる可能性があり，しかも光学的に高速に情報の読み出しができるが，光学的に書き込みを高速に，かつ頻ぱんに行うことが困難な点が実用化の範囲に限界を与えている。今後の展開としては，その大容量性を活かし，図形や画像などのファイルとして有用な役割を果すものと期待されている。

4.4.5 磁性関連技術

長い間の使用実績をもつ磁性デバイスも着実な技術進歩が見られる。特に，ＧＢ（ギガ・バ

イト）クラスの大容量ファイルとしての磁気ディスクあるいは磁気テープには，これに代わるものは当面考えられない。しかしながら，比較的容量の小さい１００ＭＢ程度のファイルには半導体デバイスであるＣＣＤや磁気バブル素子を用いた高速で信頼性の高い電子ファイルが多用されるものと考えられる。大容量ファイルには通常の磁気ディスクあるいは磁気テープに依存する以外にないと思われるが，従来の技術に頼っている限り現在の１０倍程度の改善しか期待できないといわれている。このような意味では垂直記録方式の実用化や光技術の適用などに期待がかけられている。

表４－２は上述のデバイス技術の開発動向，計算機への適用における問題点などを整理して示したものである。

表４－２　デバイス技術の開発動向と計算機への適用可能性

デバイス	A.開発動向	B.計算機への適用可能性	C.技術的問題点	D.80年代の課題
シリコンVLSI	集積度： メモリ素子 2倍／年 ロジック素子 2～3倍／2年 価格低下率： メモリ素子 40%／年 ロジック素子 25%／年	MOS： 0.1ns／gate 1μw／gate 10^6gates／chip バイポーラ（ECL）： 0.05ns／gate 1～10mw／gate 10^4gates／chip ⋮	微細加工技術 精密プロセス 高速単純構造回路 CAD	→1～0.25μm →欠陥数 数個／cm² →SOS NMOS，I^2L … →Siコンパイラ … 大型ウェファ 10^7 多重配線 10～30層 ピン数 155～300
化合物半導体（GaAs）	Mobility： Siの7倍以上 （InSbは更に10倍） Energy Gap： Siの1.4倍	FET論理回路： 30～400ps／gate 0.1～40mw／gate 遅延電力積はSiのCMOSと同等（0.1pJ以下） ガン効果機能回路： 10～100ps／gate 30～100mw／gate	材料特性の研究 均一な基板製造、 （結晶・絶縁物） 冷却技術（熱電導率はSiの1／3） 信号の振幅（0.6～0.8V）	オンチップCPU（FET）： 10^4～10^5gates／cm² 20W，1GHz バッファ・メモリ（FET）： 100ns，1KB 機能モジュール（ガン）： カウンタ・乗算器など 1,000～2,000gates／chip
ジョセフソン接合素子	スイッチング特性： 10～100ps 消費電力： 2～5μw 遅延電力積： Siの2桁以上改善	論理回路： 100ps／gate 4×4bits乗算器 45gates，27ns メモリ回路： 64bits 2.3ns 16Kbits 7ns 計算機システム： IBM 370／168 （16MB）	物性面からの研究 安定な素子製造（材料，絶縁物，熱サイクル） 集積化技術 冷却技術 実装技術（外部インタフェース） 液体ヘリウム資源	基礎研究 超高速機能モジュールの試作・実験
光素子	光ファイバ： 伝送帯域10MHz・km～10GHz・km 伝送損失0.5dB／km（1.3μm）～3dB／km（0.85μm） 発光・受光素子： 変調速度数10Mb／s（LED）～数Gb／s（LD） 寿命100万時間（LED）～1万時間（LD） 光ケーブル： 12芯×12層ケーブル	光伝送路： 10Mb／s～300Mb／s 光機能素子： FETなど 光メモリ： ホログラフィなど	高集積化光ケーブル 発光・受光素子の高信頼化・長寿命化・高利得化 光・電気高速変換技術 光分岐技術 光スイッチング素子 光機能素子	光ローカル・コンピュータ・ネットワーク 入出力機器接続（入出力チャネル） 機能モジュール接続（光バス） VLSI素子接続
電子ディスク	CCD素子： 実装密度RAMの4倍 128kb／chip，平均アクセス時間400μs，データ転送レート5Mb／s 磁気バブル素子： 1Mb／chip，平均アクセス時間40ms，データ転送レート100Kb／s	CCDメモリ： 超高速科学計算機用ファイルメモリ 256M×64bit語 1ms 磁気バブル・メモリ： パーソナル・ファイルメモリ100MB，10ms	高密度素子開発 低価格化	CCD素子： 1～4Mb／chip 磁気バブル素子： 10Mb／chip
大容量ファイル・メモリ	磁気ディスク： 記録密度3倍／4年 記録面密度10Mb／in²（限界100Mb／in²） 磁気テープ： VTR式磁気テープ 10^{13}bits 1～30Mb／cm	大容量磁気ディスク： 1GB／ドライブ 集積ヘッド 大容量磁気テープ VTRホーム・コンピュータ用ファイル・メモリ	高密度記録方式： 垂直磁気記録方式 光磁気再生方式 光または電子ビーム書きかえ方式 ホログラフィ連想メモリ 多重記録メモリ	磁気ディスク： 現在のディスクの10倍 光メモリ： 現在のディスクの100倍 磁気テープ： VTR式10^{14}bits

5．アーキテクチャにおけるシステム化技術

5. アーキテクチャにおけるシステム化技術

5.1 システム化技法の重要性

ここでは，第5世代コンピュータ・システムを開発するのに必要なシステム構築のための技術や開発を支援するツールについて述べる。

第5世代コンピュータ・システムは，知識情報処理の実現を目ざす，きわめて野心的なシステムを目標としている。一般に，このような未知の分野を多く含む目標へ到達するためには，いろいろな要素技術をいかに選択し，整理し，まとまった体系に組み上げていくかというシステム化の技法がきわめて重要である。これは，実際に物を作るとき必要なノウハウに類するものから，開発の哲学，方針といった抽象的なものまで，いくつかのレベルがある。既存の少々古めかしいが安定した要素技術で，確実なものを作ろうとか，逆に，積極的に先端的技術を採用し，少々の危険を冒しても革新的なものを作ろうといったものが，開発方針，哲学のレベルの例であろう。第5世代コンピュータ・システムの開発は，後者の方針に沿ったものである。

しかし，ここでは，このようなレベルよりは，より実際的なアーキテクチャやハードウェアを開発することを目的としたシステム化技法に範囲を絞ることとする。また，これと同時に，システム化の道具についても，触れることとする。

コンピュータ・システム構築におけるシステム化技法は，いろいろな要素となるサブシステム，すなわち要素となるソフトウェア群，コンピュータのようなマシン群，これを作るためのＶＬＳＩのような部品群，さらには，コネクタやケーブルのようなより基礎的な材料，部品群のいろいろなレベルにおける開発方針，開発のために採るべき手法のことと考える。そして，これは，部品の仕様の標準化とか，組上げるときの規則，目的のものができたか否かの検証・評価の手順といった形で，具体化される。さらに，このような規則や手順に従ったものが，実現されるために必要な開発支援ツールとして，実際にマシン（工作機械と考えてよい）が作られる。

コンピュータの開発においては，その開発支援ツールは，やはりコンピュータであるのが普通である。開発目標に至る過程では，その中間的性能のコンピュータを何段階か作り，少しずつ性能を向上させていかねばならない。これは，自己増殖のプロセスと類似している。

ここでは，以上述べたような考え方に基いて，第5世代コンピュータ・システムの構築に必要なシステム化の技法を，以下のように分けて述べる。

まず第1は，仮想化の技術である。知識情報処理システムでは，そのソフトウェアが，きわめて重要で，かつ規模の大きなものとなり，また，それを支えるハードウェアも複雑で，高機能のものとなる。このため，ソフトウェアで構築する論理マシン（仮想マシン）とハードウェア構築する物理マシン（実マシン）の間を結ぶ仮想化の技術が重要となる。ここでは，仮想化

を行う上でのモデルや基本的な考え方，仮想化をいくつかの階層に分けて行う場合のインタフェース，新アーキテクチャを仮想化の考え方に従って整理した場合の関係などについて述べる。

第2は，融合化技術である。アーキテクチャの研究は，いくつかの柱を中心に，別々の目的を持つグループに分けられる。本報告書では，高レベルの計算モデルや言語に基く新アーキテクチャや，機能分散アーキテクチャ，ＶＬＳＩアーキテクチャを3つの柱としてとらえている。これらのアーキテクチャをいかにして融合化し，オフィス・オートメーション・システムや，ＣＡＥ／ＣＡＤシステム等を構成するか，また，知識情報処理実現の機能との対応はどのようにするかといった点についての方針は重要であり，ここでは，このいくつかの場合について，総合的指針を与えている。

第3は，大規模，高信頼化システム設計・開発技術である。大規模なハードウェア，ソフトウェアの構築には，人間の能力の限界を越える数の部品やモジュールを用いねばならない。このため，システィマティックな構成手法と，その機械化（コンピュータ化）が不可欠となる。また，新しい手法の開発には，試行錯誤がつきものである。このような視点から，アーキテクチャ，ハードウェア，回路，ソフトウェアについてのシステム化技術について述べる。さらに，高信頼化のための技術は，特別な技術分野であり，特色あるシステム化技術を含んでいる。ここでは，このような技術について述べる。

尚，本章の内容は，第3章の機能分散アーテクチャと，きわめて密接な関連を持つ。中でも仮想化，融合化は，3.3節のシステム制御方式が，具体的実現技術となる。このため，これら双方を参照されることが望ましい。

5.2　仮想化技術

5.2.1　仮想化の必要性と体系化

(1) 仮想化の意味と効用

仮想化は，ある認識対象をその所与の存在形式から分離し，問題としている論理的な諸性質の範囲内で，他の可能的な存在形式に等価変換する操作を意味している。こうした操作を行なう目的は，ある対象の論理的なイメージの転換を計ることによって，それに対する論理操作を考え易くすることにある。たとえば，計算機システムにおける仮想化の代表例である仮想記憶についてみると，これは補助記憶（プログラマからみた存在形式という意味で，すでに一種の論理イメージとなっている）という存在を，主記憶という論理形式に等価変換したものということができる。これによって，プログラマはプログラムのオーバレイ操作や，補助記憶に対する入出力操作から解放されることになる。

このように，仮想化は対象に対する認識形式間での変換操作であるが，変換前の形式でみた対象を＂ホスト＂，変換後のそれを＂イメージ＂と呼ぶことが多く，以下でもこれにならうこ

とにする。

仮想化が有効であるのは，ホストの形式よりもイメージでの形式の方が，何らかの点で有利になる場合であることはいうまでもない。そのようなケースとして可能性が高いのは，

① 既成の形式への還元となる場合，

例： ＩＢＭでいう仮想計算機

② 抽象化のサポート手段となる場合，

例： 高級言語実行システム

③ 複数の異なる形式のものを１つの形式のものとして統合化する場合，

例： 仮想端末

などである。仮想記憶は見方によって，①であったり，③であったりする。

仮想化と密接な関係にある概念として，抽象化があり，混用されていることが多いが，これらは区別した方がよい。抽象化も認識形式の転換を意味している点では，仮想化と共通しているが，抽象化というからには転換の前後において，認識形式のもつ抽象度に有意な変化がなければならない。これは仮想化には要求されていない条件である。仮想化の有効性①に挙げた例の場合，抽象度にどれだけの変化があったかは，言うのが困難である。

複雑なシステムの設計においては，階層構造化の手法がよく用いられる。これは明らかに抽象化の操作である。しかし，このとき，ある上位の階層に立って，そこをそれより下位の階層に対する仮想化と呼ぶこと，逆に下位の階層に立って，それより上位の階層をその仮想化とみることは，いずれも正しい。この場合の有効性は，上記②のケースであるとすることができる。この場合，仮想化の役割は従属的なものであるに過ぎない。仮想化が本領を発揮するのは，む

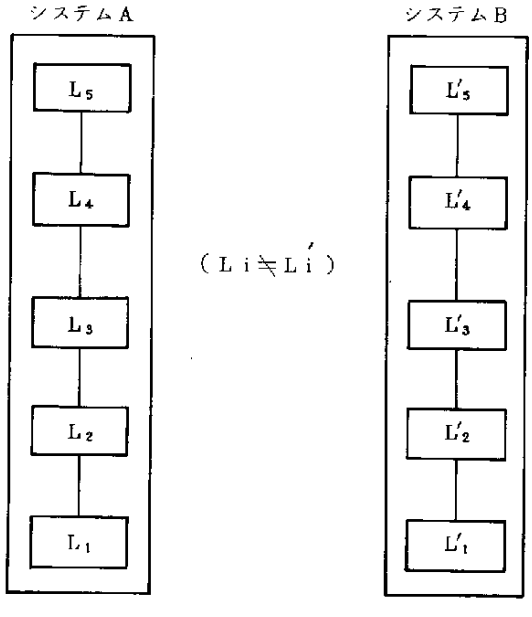


図５－１　階層間コミュニケーション

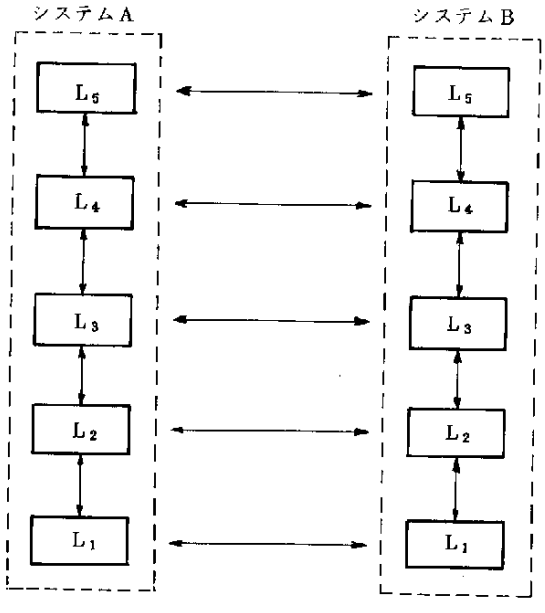


図５－２　階層別コミュニケーション

しろ，つぎのような場合である。すなわち，同じような階層構造をもつ複数個のシステムがあって，それらの互に対応する階層間で何らかのコミュニケーションをもとうとする場合である。

このとき，各システムの対応する階層で用いられている概念体系が，互に一致しているという保証がないとすれば，コミュニケーションの成立は望みえなくなる。そこで主役としての出番が与えられるのが，仮想化の有効性③のケースである。ネットワーク・アーキテクチャの標準化において仮想化概念が重要視されているのは，まさにこの点においてである。

仮想化の基本的な意味と効用は以上の通りであるが，仮想化の利点を利用するには，それ相応の対価を支払う必要がある。最初の定義に述べてあるように，仮想化は等価変換の操作である。イメージの動作は，それに等価的なホストの動作に変換されて実行されなければならないが，この変換が相当なオーバヘッドをもたらす。それが仮想化の対価である。

(2) 機能分散システムにおける必要性

機能分散システムは，それぞれ何らかの専用性ないし応用指向性をもった計算機の集合体として構成されるシステムである。個々の計算機は，それ自身十分な複雑さをもった準システムであり，一定のルールに従って，相互にある関係を維持しながら，求められた機能を果していくようになっている。

このシステムの利用者は，その目的に応じて必要な計算機（群）や端末（群）などを選択し，それらの間を自由に接続して，問題の処理を実行する。といっても，これは即物的な説明としての話であって，通常のユーザにこのように見えるようになるという意味ではない。

以上から明らかなように，機能分散システムでは，異質なシステム要素が離合集散してさまざまなサブシステムを繰返し構成していくことになる。このようなサブシステムの編成を見通しよくするためには，利用目的に合わせた基準パターンとして，超高水準の応用向仮想計算機の設定を考える必要がある。この設定に対しては，その実現性の保証として，ハードウェアに始まる実現手段の階層的な積上げのための枠組が，用意されていなければならない。この枠組のあり方については，次節で議論されるので，ここでは触れない。

機能分散システムを全体として把えるにせよ，上記のようなサブシステムにおいて把えるにせよ，それらは何よりもまず，異質な要素の集合である。この集合にシステムとしての全体性と自己制御性をもたせるためには，必要な相互関係を維持する手段としてのコミュニケーションの成立が，絶対条件となる。各要素はそれ自身がすでに複雑な構造をもつ計算機システムであることが多い。しかも，それぞれの計算機システムは，その応用指向性から，まったく異なった情報体系をもつことがある。このような情報体系の間に，共通の形式を設定することが，コミュニケーション成立のために不可欠である。ここに機能分散システムにおける仮想化の，もう1つの必要性がある。次節での仮想化が，機能分散システムにたて糸を通すものであるとすれば，コミュニケーション成立のための仮想化は，よこ糸を通すものであるということがで

きる。以下では，この後者の仮想化のあり方について，何が研究課題になるかを検討する。

(3) 仮想化のあり方－統合化の観点から

ここでは，機能分散システムを構成する計算機システム相互間に，完全なコミュニケーションが成立するための条件を具体的に解明し，それらの条件を満足するためには，仮想化がどうあらねばならないかを問題にする。

CONTEXT

MESSAGE

ADDRESSER ― ― ― ― ― ― ― ― ― ADDRESSEE

CONTACT

CODE

図５－３　言語コミュニケーションの構成要素

一般に，ことばによるコミュニケーションは，図5-3のように6つの要素によって構成されている（Ｒ．ヤコブソン）。すなわち，コミュニケーションは話し手（ADDRESSER）と聞き手（ADDRESEE）の間に成立する関係である。この両者は音声，文書などの接触手段によって結ばれていなければならない。そして，この結合を通じて，話し手から聞き手にメッセージが送られる。メッセージは接触手段に従属する適当な形式（ＣＯＤＥ）によって表現される。メッセージの内容が話し手の意図通りに聞き手に伝わるためには，両者の間に共通のコンテクストが成立している必要がある。

図において，破線から下の部分はコミュニケーションにおけるシンタクティックな側面を示し，上の部分はそのセマンティックな側面を示している。従来のネットワーク・アーキテクチャにおけるプロトコル標準化の対象とされていたのは，このシンタクティックな側面だけである。計算機システム間接続を完全に作動させるためには，セマンティックな側面に対する標準化も考える必要がある。

i) シンタクテックな側面の検討

機能分散システムにおける計算機システム間接続標準化をどうすべきかは，今後の重要な研究課題である。この問題を考える場合に忘れることができないのは，各種の公共的な標準規約の存在である。特に，計算機システム間接続に関するものとしては，現在ＩＳＯ（国際標準化機構）のＴＣ９７（電子計算機と情報処理）による。開放型システム間接続（Open Systems Interconnection，以下ではＯＳＩとする）の標準化活動がある。これは上述の意味でのシンタクティックな側面を対象とするものであるが，一般性という点では最も包括的であり，検討課題の枠組を識るためにだけでも十分尊重するに足るものである。

ＯＳＩの概念

ＯＳＩの考え方の特徴はつぎの２点に集約される。

"(a)　利用者が，その目的に応じて計算機や端末等を選択し，それらの間を自由に接続して業務処理を実現するために必要な各種の標準を全体的に体系化することがＯＳＩの最終的な目標である。このためには利用者向きプロトコルを含め，すべてのプロトコルを標準化の対象として位置付ける大掛りな論理モデルを設定する必要がある。

(b)　一方，すべての装置間で(a)に述べた標準化プロトコルを適用することに必ずしも効率的でない場合がある。例えば同一室内に設置した同一メーカ製品の間では非標準のローカルなプロトコルを使用することを認めてもよい。"

この引用で注意すべきことは，そこに仮想化のあり方が端的に述べられている点である。すなわち，(a)での論理モデル設定はまさに仮想化の役割が何かを明示しており，(b)は仮想化適用に限界があることを指摘したものとなっているのである。以下ではこうしたコメントをいちいちつけないが，引用の狙いが仮想化のあり方を探ることにあることはいうまでもない。

"１台ないし複数台の装置（計算機，端末等）とそのソフトウェア，オペレータなどを組合せた１式のシステムを単位と考え，このシステムが，他のシステムと通信する場合にＯＳＩ標準に従うならばこのシステムは，open system（以下では簡単のために単に，systemと書く）であるとみなす。system内の装置間の通信は必ずしもＯＳＩ標準に従わなくてもよい。ＯＳＩとは，このようなsystemを通信回線網などの物理媒体に接続しsystem上のapplication−process（利用者プロセス，管理プロセス，オペレータ等）の間で通信を行い，分散処理を実行するために守るべき諸標準と，これらの標準の総合的相互関連を記述した文書（参照モデル）との体系であると定義できる。"

"ＯＳＩ参照モデルは，ＯＳＩのための各種標準の開発作業を相互調整するための共通基盤を提供し，更に既存の標準を全体の参照モデル内に位置付けるための便宜を与えることをねらいとしている。しかし，このモデルは，実際のハードウェア，ソフトウェアの実現方法についての仕様を定めるものではなく，標準を定めるための概念的，機能的な枠組を設定するものである。"

"本モデルで対象とする（通信関連）機能の全体を複数のlayerに分割する。分割のための原則のうち，主なものをあげると，

(a)　隣接layer間の境界で定義すべきserviceの記述がなるべく簡単になるように境界の場所を選ぶ。

(b)　特に他と質の異なるfunctionを集めて一つのlayerとする。

(c)　特定のlayer内のfunctionやprotocolを完全に入れ換えても他のlayerに影

響がないように分割する。

全体を layer に分割することにより，各 system も layer 対応の subsystem に分割される。各 subsystem には１個ないし複数個の entity （機能モジュール）が存在する。”

“上の原則に基づいて設定されたＮ２２７参照モデルでは，７つの layerを定義している。”

表５－１　各Layerの役割

N	layerの名称	役　　割
7	Application Layer	Application－entity の存在する最上位 layer であり，管理用および利用者向きの application－protocol を実行する。
6	presentation Layer	構造を持つデータの入力・授受・表示・制御を行う。仮想端末プロトコル，および仮想ファイルプロトコルとジョブ転送プロトコルの基本部分を実行する。
5	Session Layer	session と呼ぶ connection を設定し，データの送信・受信の制御，同期の制御などを行う。
4	Transport Layer	session－entity の間に汎用的な transport－connection を設定し，トランスペアレントなデータ転送を行う。
3	Network Layer	一つ又は複数の通信網を通して routing を行い，通信相手の application－entity の存在する system に至る network－connection を提供する。
2	Date Link Layer	二つの system の間でデータを送受信し，物理的通信回線の伝送誤りに対処する data－link 制御手順を実行する。
1	Physical Layer	physical－connection（即ちデータ回線）を設定・維持・解除するための機械的・電気的・機能的および手順上の規約を含む。

ii） セマンティックな側面

計算機システム間で交換されるメッセージには，さまざまな形式のものがあり，また，それらが解釈されるべきコンテクストもまちまちである。例えば，メッセージにはコマンド，プログラム，構造データなどの形式があり，コンテクストにはプロセス制御，データベース処理，言語処理などの処理的環境が考えられる。

メッセージの内容はコンテクストに従って解釈されるので，コミュニケーションの枠組を設定するという目的からは，コンテクストのあり方を先にとりあげるべきである。一般に，コンテクストには「言語的コンテクスト」と呼ばれるものと，「場面のコンテクスト」と呼ばれるものがあるが，ここではまだメッセージの言語的構造には立入っていないので，当面は後者の意味でのコンテクストを問題にする。

計算機システムにおけるメッセージ理解のための場面設定がどうあるべきかという問題は，人工知能における知識を表現するための枠組のあり方と同種の問題であるとみることができ

る。そこでの議論にみられるように，この問題はシステム効率化への基本戦略を巡って，かなり深刻な意見の対立をはらんでいるものである。ということは，これは機能分散システムにおいても，意思決定を迫られることになる大きな研究課題であることを意味している。

計算機システム間でのコミュニケーション成立のために行なう場面設定の大枠として，常識的に思い付くモデルは，オペレーティング・システムに用いられている機能的な区分化である。このような区分化の例として，現在のところ最も完備的で将来性のありそうなものは，ＩＢＭシステム／３８にみられるものである。その区分はつぎの通りである。

- オブジェクト制御機能

  オブジェクト管理／アドレッシング／コンテクスト管理／資格認知管理
- プログラム機能

  プログラム管理／計算と分岐／プログラム実行
- スーパバイザ／コントロール機能

  プロセス管理／イベント管理／例外管理／リソース管理
- データ機能

  データベース管理／インデックス管理／キュー管理／スペース管理
- ソース／シンク機能

  Ｓ．Ｓオブジェクト／構成とステート／回線網関係／Ｓ．Ｓ命令／Ｓ．Ｓ例外とイベント／ＳＮＡコンセプト／ロード・ダンプ関係／Ｓ．Ｓオブジェクト回復
- マシン・サポート機能

  システム３８サポート機能／マシン観察機能／回復機能

この枠組は１つの目安として引用したものであり，これがそのまま機能分散システムに適用できるものではないことは明らかである。

まず第一に，基本的な問題として，機能分散システムに属するすべての計算機システムに対して，一元的に同じ枠組を適用することが必要かどうか，また，本当に可能かどうかが問われなければならない。さらに，全部の機能についてそうなのかどうかということも問題になる。適用範囲の局所化は，一般性という点からも，最初から想定しておいた方がよいかも知れない。こうした統一化に対しては，効率の観点からも疑義が出される可能性があるからである。

しかし，計算機システム間にコミュニケーションが存在する以上，コンテクストの共通化・標準化が何らかの範囲で必要なことは確かである。システム／３８にみられるような機能分類の枠組のあり方が，具体的な検討の対象とされるのは，この段階に入ってからのことである。ここでこの枠組に期待される役割は，まえに述べたＯＳＩと同じような論理モデルとしてのそれである。

こうした論理モデルに要請される条件としては，まず，本質的ではない細部の特異性を消去するために十分抽象的であることが必要であり，また，落ちこぼれを生じないようにするために，十分に包括的でなければならない。システム／３８がこれらの条件を満足しているかどうかは，既成の計算機システムに適用してみればある程度の見当がつくはずである。しかし，問題は将来の計算機システムに対して，その発展の可能性を阻害することがないかどうかである。例えば Actor モデルとの折合いはどうなるのか。複数の論理モデルの設定が必要となるかも知れない。

さらに細目についてみると，システム／３８の枠組にはかなりの補強が要りそうである。例えば，分散プログラミングや分散データベースなどのための機能である。これらはいずれも今後の重要な研究課題の１つとなるものである。分散データベースは，複数のデータベースの間に共通のコンテクストが設定されたシステムとみることができる。この意味で，いま考えている問題に最適のモデル・ケースとなるものである。そこにどんな課題があるかを知るために若干の引用を行なう。

"分散型データベース・システム設計において，解決されるべき問題点としては，

1）その構成要素たるデータベース・システムの異種性の解決と，

2）地理的に分散し，かつ論理的に異なった意味を持つデータベース・システムを１つの論理的（仮想的）データベース・システムに統合すること，

の２点である。"これらの問題には，データ・モデルとアクセス言語という構文的な側面と，意味構造という意味論的な側面がある。データ・モデルについては，マキーマの層別構造を活用して，適当なレベルに共通のデータ・モデルを設定することによって解決できる。言語についても共通言語の設定を行なう。

構文的には同種化が成立した個別データベース・システムから，"分散型データベース・システム全体にある１つの意味構造を持った論理的データベース・システムを設定することを統合化と呼ぶ。統合化とは，ネットワーク共同体の要求を満たすようなデータベースの意味の記述を各データベース・システムの局所スキーマを基にしてつくることである。"

"分散した局所スキーマが１つの全体スキーマへ統合されるためには，これらの局所スキーマは互にある意味論的関係性をもたねばならない。"この関係性を表わす１つの方法として，セマンティック・リンクの概念の導入がある。"セマンティック・リンクは，２つの局所スキーマ・リレーション間に存在し，２つのリレーション内の各々の部分集合間の関係性を表わ"すものである。

以上でコミュニケーション成立のためのコンテクスト面での条件について，問題点の概要をみた。つぎに，メッセージ面であるが，問題の本質はコンテクストでの議論で尽きている。要はメッセージ個有のシンタックスとセマンティクスについて，共通の了解を成立させる問

題であり，改めて論議するまでもなさそうである。

(4) 仮想化にともなう問題

仮想化のデメリットとしてオーバヘッドの問題があることは，仮想記憶での経験などからすでによく認識されているところである。

このオーバヘッドの存在が致命的なものとならない限り，仮想化の適用はかなり自由に行ないうるものである。しかし，それだけに目的別にその導入が進められて，結果として多重的な適用になる場合には，それら相互間の整合性が問題になってくる可能性がある。例えば，ＤＣＮＡではネットワーク全体の仮想化として論理ネットワークが導入され，さらにそれの論理的区分化として仮想ネットワークが設定できる。これと分散型データベースのスキーマ構成をどう対処させるかなどである。

5.2.2 仮想化の階層とそのインタフェース

第５世代の計算機システムとして，各種の知識情報処理システムを構築して行くのに必要とされるであろう各種技術の相互関係を以下にまとめてみる。

ここでは，図５－４に示される通り，知識情報処理システムをＶＬＳＩを用いて実現するに至るステップを６段階に分け，その各々の階層および階層間のマッピング（仮想化）について，簡単に解説を試みてみる。

(1) 知識情報処理システムと仮想マシン

ＣＡＥ／ＣＡＤシステム，意思決定支援システム，知能ロボットシステム等の各種知識情報処理システムを構築する場合には，そのシステム設計者に対し，設計のための記述言語を与える必要がある。このためには，知識情報処理システム構築の基礎技術とされている知識ベース管理システム，知的プログラミング・システム，新プログラミング言語(注)などの＂記述言語インタフェース＂を明確に設定し，その言語で記述されるプログラムの実行を司どる＂仮想マシン＂のイメージを明確にする必要がある。この言語インタフェースの基礎となる技術としては，自然言語処理，述語論理，関数型論理などに基づく新しいプログラミング言語の確立などがあげられよう。

（注） これらについては，基礎理論研究分科会の報告書に記述されている。

(2) 仮想マシンの構造

仮想マシンの定義に従い，これを実現するために，この仮想マシンの機能を大きく＂仮想処理マシン＂，＂仮想データベース・マシン＂，＂仮想ネットワーク・マシン＂の３種に分けてみる。この諸コンポーネント仮想マシンに対する＂アクセス・インタフェースとして，＂仮想命令セット＂，＂リレーショナル・データベース・アクセス命令セット＂，＂仮想ネットワーク・アクセス命令セット＂，などが設定されよう。

これらのアクセス命令セットについては，各々の分野で既に数多くの研究が進められており，これら諸技術の融合化が重要なテーマとなるであろう。たとえば，仮想処理マシンのアクセス

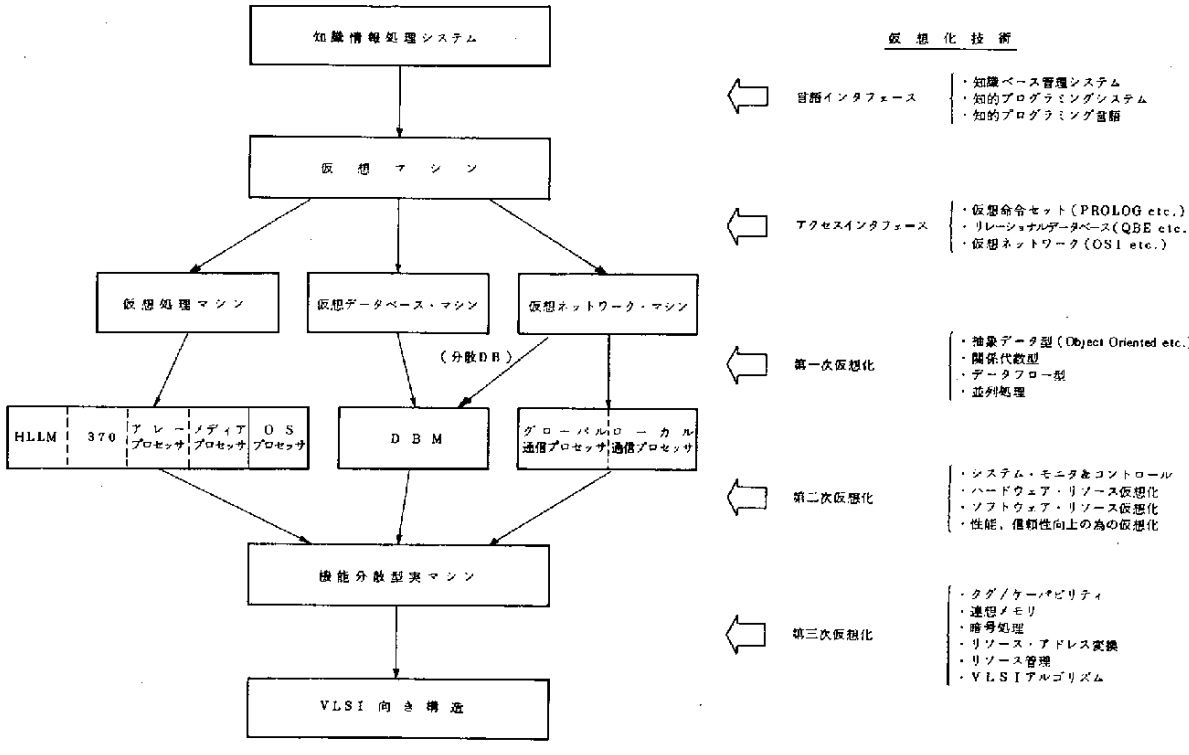


図５－４　知識情報処理システムの実現ステップ

・インタフェースの例としては，ＰＲＯＬＯＧマシンなどがあげられるし，仮想データベース・マシンとしては，リレーショナル・データベース・モデルに基づく，アクセス言語および非手続き型コマンド言語などがその例として挙げられる。また，仮想ネットワーク・マシンのアクセス・インタフェースとしては，ＩＳＯで討議されているＯＳＩまたはＳＮＡにおけるＶＴＡＭの如きものがあげられよう。

また，仮想マシンの言語インタフェースで書かれたプログラムを，アクセス・インタフェースの命令記述列に変換するコンパイラも必要となる。

(3)　コンポーネント仮想マシンの構造

各種のアクセス命令セットを持つコンポーネント仮想マシンを実現するべく，アーキテクチャ・レベルでの方式設計を行う。これを第一次仮想化と名付ける。ここでは，"抽象データ型"，

"関係代数型"，"データフロー型"等に基づくアーキテクチャ・レベルでの機能方式の選択および具体的設定がなされる。その結果として，仮想処理マシンに対応しては，高位言語プロセッサ，アレー処理プロセッサ，ＯＳプロセッサ，メディア・プロセッサなどの各種実マシンのイメージが確立される。仮想データベース・マシンに対応しては，実データベース・マシンが，また，仮想ネットワーク・マシンに対応しては，その特性に応じてグローバル通信プロセッサ，ローカル通信プロセッサなどの実マシンのイメージが明確化される。

(4) 実マシンの統合化

これは，昨年来，本分科会で討議されてきたステップであり，上述の諸実マシンを，レベル１～３に分け，かつそれらの配置のローカル／グローバルを問わず一つの統合化された実マシン（"機能分散型実マシン"）として構成する"第二次仮想化"のステップである。

ここでは，システム・モニタ＆コントロールの機能，および，各種ハードウェア・リソース，プログラム／データ等のソフトウェア・リソースを，その量的制限の枠にとらわれずに使用できること。および，多重輻奏使用に伴う論理的一貫性の確保などを目的とするリソース仮想化技術，例えば，ＴＳＳ，ＶＭ，リソース・インターロック等などが技術課題としてとりあげられよう。

また，単に機能面における仮想化のみならず，性能，信頼性の面においても，一種の仮想化により，その向上の図られることもある。たとえば，キャシュ技術による処理性能の向上であるとか，冗長性の導入による信頼性の向上などが，その例として挙げられる。

(5) 実マシンのミクロ・レベルでのアーキテクチャ設計

上述の"機能分散型実マシン"の個別プロセッサの処理機能を，ＶＬＳＩ技術を有効に駆使して，コスト・パフォーマンス良く実現するためのステップを"第３次仮想化"と名付ける。ここでは，チップ・レベルおよびファームウェアでの基本機能の実現が期待される。この種の技術課題としては，タグ方式，ケーパビリティ管理，連想メモリ，暗号処理，リソース・アドレス変換，リソース管理ＶＬＳＩアルゴリズム等のミクロ・レベルでのアーキテクチャの設定を伴うものがある。

(6) ＶＬＳＩマシンの実現

機能分散型ＶＬＳＩマシンを実現して行くには，大規模論理設計技術，インテリジェントＣＡＤシステム，高性能実装，冷却技術，などの基礎技術の確立が必要とされる。

以上の如き仮想化のステップで第５世代の計算機システムが構築されるとしたら，エンド・ユーザと実ハードウェアの間に横たわる複雑なる構造を持つ"ＳＥＭＡＮＴＩＣ　ＧＡＰ"の解決と，これらをのり越えて着実にシステムを実現するための"ＬＡＹＥＲＥＤ　ＳＴＲＵＣＴＵＲＥ"に基づく明確なる層間インタフェースの設定の２点が重要課題となるであろう。

5.2.3 新アーキテクチャにおける仮想化

(1) 仮想化の要因．

仮想化を行う要因とその効果について，簡単に分類すると次の３つに大別できる。

i) マシンの機能や負荷分散に伴う仮想化

ii) 資源の（地域的）分散に伴う仮想化

iii) 資源の共有に伴う仮想化

これらを，新アーキテクチャの６つの候補となっているマシンについて適用すると，これにより，これら６つのマシン間の関係が，より明らかとなると考えられる。

① マシンは機能や負荷分散に伴う仮想化

マシンの規模を拡大し，能力を高めようとする時，マシンアーキテクチャの設計に際して，機能分散化と負荷分散化をはかる。通常，これらは，中間言語の設定やアーキテクチャの階層化などの仮想化を伴う。

機能分散に伴う仮想化とは，マシンのある機能を専用モジュールにまかせるとき，その機能のレベルで中間言語もしくは，アーキテクチャにおける一つの階層を形成することを指している。その中間言語より上のレベルで使用する場合には，そのレベルの機能，もしくは命令の一部が，異なるハードウェア，もしくはソフトウェア・モジュールにより実行されるようには見えず，同一のハードウェア等が，存在するように見えるわけで，ここに仮想化が行われる。この例は，汎用機に付加されるアレイ・プロセッサや，入出力チャネル等に見られる。

負荷分散に伴う仮想化とは，ホモジニアスなマルチ・プロセッサによく見られるように，マシンの能力向上のため，多数の処理プロセッサを備えて，一つのジョブを並列実行する場合の仮想化である。このとき，この仮想化のレベルより上で使用する人には，一台のプロセッサがあるようにしか見えないことになり，仮想化の一階層が設定されたことになる。

このようなマシン構成の都合から設定される仮想化は，そのアーキテクチャを考える上で有効な指針を与える。

② 資源の分散に伴う仮想化

これは，ハードウェア，ソフトウェアの資源が，主に地域的に分散することにより生じる。この例としては，ネットワーク中に分布する分散型データベースにおいて，一人のユーザが，いくつかの分散したノード中に存在する データベースを融合したようなビューを設定したような場合に生じる。ユーザは，データベースが地域的に分散しているとは思わず，あたかも一ケ所に集中したデータベースにアクセスしているかのように考えるわけで，ここに仮想化が行われるわけである。

この種の仮想化は，ローカル・ネットワークをはじめとする種々のネットワークに伴うも

のが主となる。ネットワーク化は，一種のマシンの規模の空間的拡大とも考えるわけであるが，①よりは，より大きなスケールのものと考えてよいと思われる。

③　資源の共有に伴う仮想化

これは，ハードウェアやソフトウェア資源を，複数のユーザにより共有する場合に生じる仮想化で，ＴＳＳシステムが，１つの例である。そのほか，入出力デバイスの共有やプログラムコードの共有などもあり，それぞれのレベルで仮想空間が形成され，その空間の間の分離のための手法と，空間の間の安全な交信の手法が重要な問題となる。

この種の仮想化は，当然，分散データベースやネットワーク中にも見られる。そして，この仮想化は，マシンの応用形態が，高度化していくと生じるものと考えられる。

(2)　新アーキテクチャの構成と仮想化

新アーキテクチャの候補としてあげた６つのマシン（第２章参照）について，その構成を仮想化の立場から検討してみる。

新アーキテクチャは，その多くのものが今後の研究により明らかになるものであるため，細かな議論は難しい。このため，ここでは，主に，前節(1)の①に述べた機能分散又は負荷分散の考え方に基づく仮想化を中心に検討する。

①　仮想化の枠組

新しいアーキテクチャに基づく第５世代マシンは，述語論理型や関数型のプログラミング言語を持つ高級言語マシンであることが予測できる。小規模な実験機の場合には，直接ファームウェアにより高級言語をインタプリートするような仮想化のレベルの少い実装手法がとれるが，高速，大容量化，さらには，ネットワークの構成要素となる場合など，大規模化することで，いくつもの仮想化のレベルが必要となる。仮想化のレベルに対応して，中間言語が存在するものと仮定すると，次のようになると考えられる。（図５－５参照）

模式的に考えれば，負荷分散又は機能分散を行うごとにアーキテクチャとしては，より低レベルの機能を実現するレベルへと下っていき，ついにはゲート・レベルの記述（これも一種の中間言語と考える）に至る。

図中の機能モジュールは，同一のプロセッサを多数用いる負荷分散を行うものであったり，関数の機能ごとに異る機能モジュールのある機能分散の形態であったりする。

機能モジュールは，上位レベルでは，データベース・マシンであったり，アレイ・プロセッサであったりする。また下位レベルでは，ＶＬＳＩ化された音声プロセッサであったり，または，パターン・マッチング用チップであったりする。

以上のような枠組を仮定して，新しいアーキテクチャの構成を検討する。

②　新アーキテクチャと仮想化

新アーキテクチャの候補としてあげた６つのマシンは，それぞれ第５世代マシンを目ざす

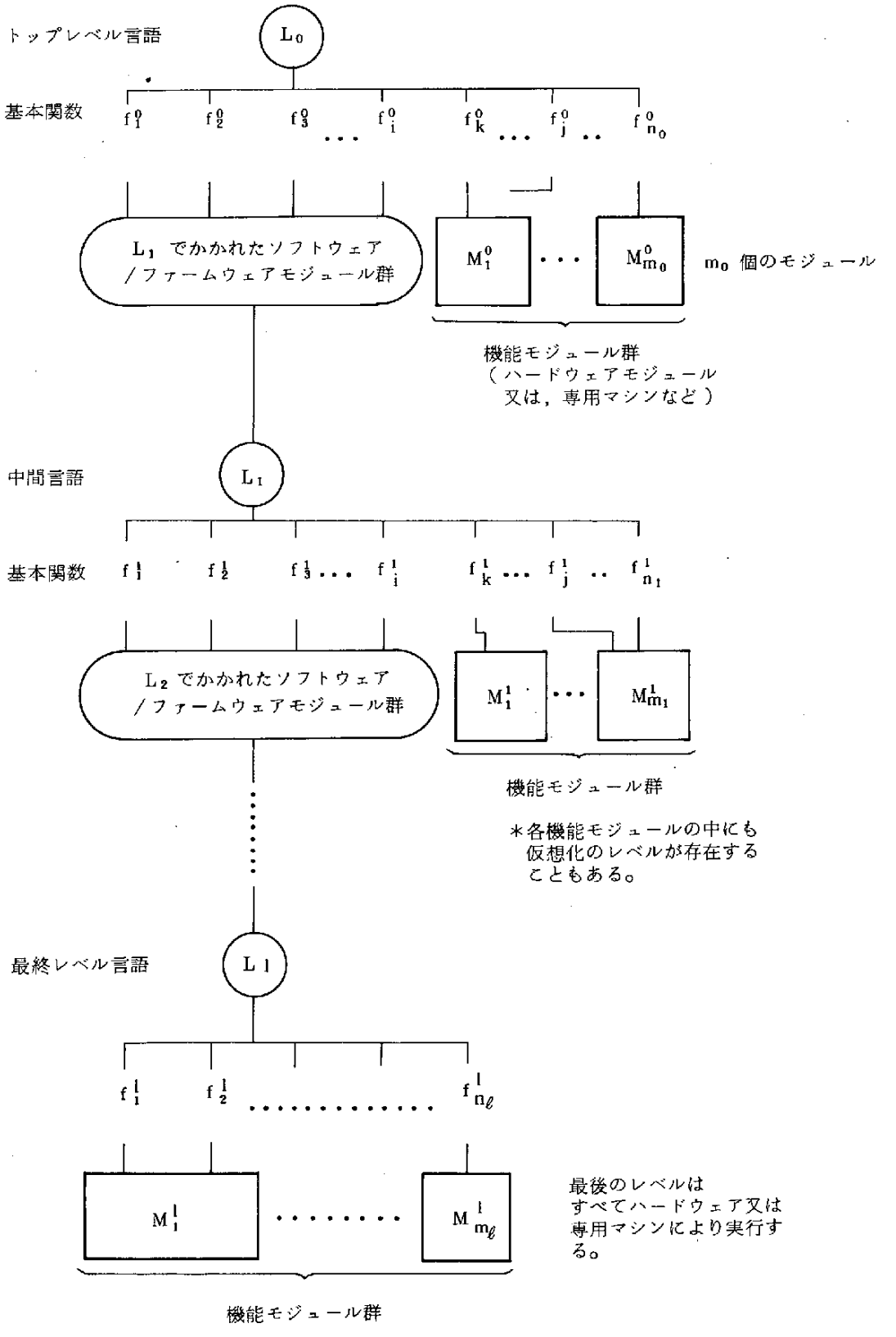


図５－５　中間言語のレベルと機能・負荷分散の対応

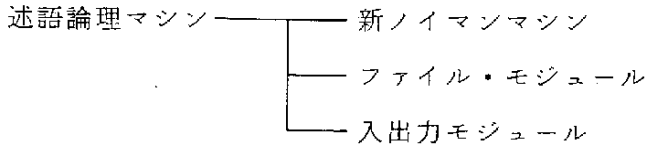
研究の出発点と考えられる。第5世代マシンを山の頂上とすれば，これらは6つの登山口と考えてもよい。しかし，その出発点の位置関係はかなり異り，トップダウン的なものやボトムアップ的なものが混在する。

ここでは，述語論理マシンと関数型マシンを，マシンのトップレベルの言語と考え，そこからマシンを構築する上で存在し得ると思われる仮想化を考える。

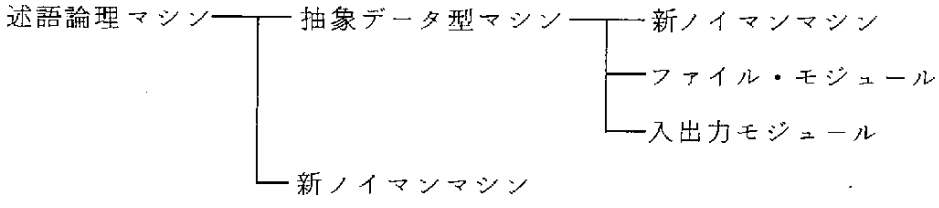
a）述語論理マシンを中心とする場合

述語論理マシンの最も簡単な構成は，ファームウェアによりそのインタプリタを実現するものであろう。

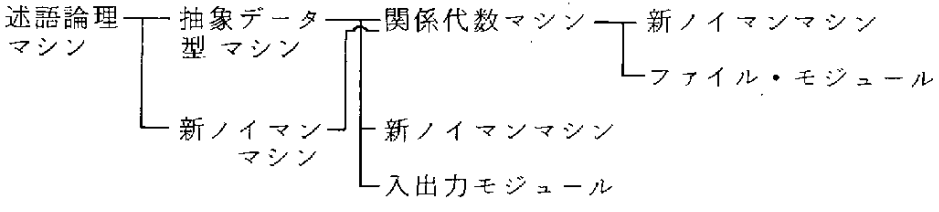
これは，次のように書ける。



述語論理型のプログラミング言語に，抽象データ型の機能をとり入れると仮想化のレベルが一段増えると考えられる。

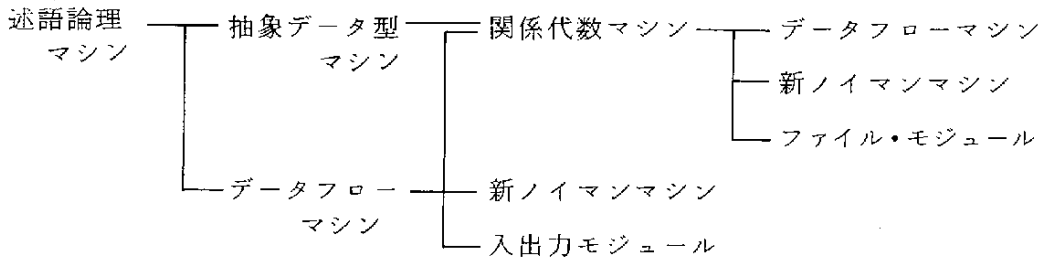


新ノイマンマシンを中心として，そのままプログラムや規則（クローズ）の大容量化をはかった場合，関係代数マシンのようなデータベース用機能モジュール（専用マシン）を用いることが考えられる。新ノイマンマシンは，パターン・マッチャーなどＶＬＳＩ化した高機能モジュールも含むと仮定している。



高速化をはかる場合はデータフローマシンのような高度な並列処理方式の導入が必要であろう。データフローマシンは，関係代数マシンの実現にも使われる可能性があろう。

述語論理マシンとデータフローマシンの間の関連性の明確化は，今後の重要な研究テーマである。

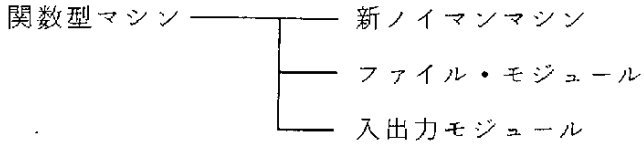


以上，いくつかの場合について，可能を思われる組合せを検討した。この図の意味するところは，それぞれのマシンが独立に開発されて，これらのように接続されるのではなく，研究成果が利用されて，一つのマシンを開発するとき融合化されることを示している。この点は，注意が必要である。

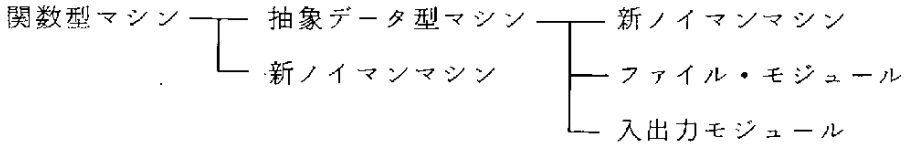
b）関数型マシンを中心とする場合

上記a）と同様に，次のようないろいろな構成例が考えられる。

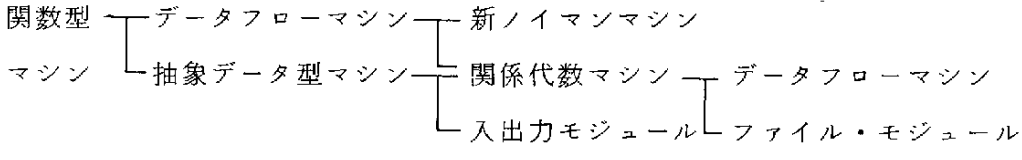
ファームウェアを用いたＬＩＳＰマシンなどの場合が，最も簡単な例である。



データ抽象などをとり入れた新言語のサポートの場合には，次のようになろう。



高度な並列処理の利用には，関数型言語の場合，直接データフローマシンと関連ずければよい。



以上，a），b）において，２つの言語よりはじめて，機能を展開したが，トップレベルの言語を，抽象データ型をとり入れたノイマン型言語にすれば，直ちに抽象データ型マシンが対応し，データベース用に関係論理型言語を設定すれば，関係代数マシンが対応する。

このような議論の示すところは，新アーキテクチャにあげられている６つのマシンは，研究テーマとして，将来のマシンの重要な要素技術をカバーしていることである。

第５世代マシンの仮想化は，その要素技術となるそれぞれのマシンの本質が明確にされるにつれ，より正確な構造が描けるようになるはずである。

5.3　諸アーキテクチャの融合技術

5.3.1　融合技術の必要性

知識情報処理システムを実現する為のコンピュータを設計する場合の過程を考えると次のようになろう。まず，知識情報処理に必要な様々な応用を考え，それを処理する為の仮想的な処理機構を幾つか設定する。これは適当な仮想マシンを設定することに相当するが，何段階もの階層構成として考えることが出来よう。この仮想マシンの機能は，処理，データベース，ネットワークの３つに大別出来るが，それを実現する現実のマシンが機能分散型アーキテクチャに基づくコンピュータであって，その構造を定める為の考慮要素が新アーキテクチャであり，ＶＬＳＩアーキテクチャである。従って，言い変えれば，知識情報処理からの要求（それは仮想マシンとして表現されているが）を実現するのに，新アーキテクチャの手法を用い，VLSI構造を意識して，機能分散型システムを作り上げると言うことになろう。

これら３アーキテクチャを組み合わせ融合させた形に持ってゆくのが現実のシステム設計になろうが，これらのバランスを取ってゆくことは現実問題としてかなり高級な作業となる。今後のシステム設計に於いては，「ダイナミック・プログラミング」にも似たこの設計作業を進める態度が大切であると考える。

以下の節では，この融合の試みの一例を示す意味からまず5.3.2節で，機能分散アーキテクチャを構成する各要素マシンを新アーキテクチャとＶＬＳＩの目から捉えた見方を示し，次いで5.3.3節では知識情報処理の主な応用である３分野として，ＣＡＥ，オフィス・オートメーション，意思決定支援を取りあげ，それぞれの分野に於ける具体的な仕事が如何にこの機能分散アーキテクチャ上で実現されるかの例を示す。

5.3.2　機能分散アーキテクチャの各要素マシンの実現方式

5.3.2.1　パーソナル・マシン

パーソナル・マシン一般に要求される事項としては次のような項目があげられる。

① 利用者からの個別処理のかなりの部分を通常は処理する。
② 高度マン・マシン・インタフェースのフロント・エンド機能を果たす。
③ 多種多様な処理及びインタフェースの要求を満たすこと。
④ 利用者に対するサービスを各利用者の要求に合わせてカスタマイズ出来ること。
⑤ 上位レベルのシステムへの高レベルインタフェースを提供すること。

②は，知識情報を備えて使い易いインタフェースとすることを意味し，⑤は成長する上位レベルシステムの複雑さ，サービスの多さ，ネットワーク結合形態，諸資源のありか等を利用者から隠して，単なる高級サービスとしてしか見せない為の機能を言う。

次に，パーソナル・マシンに必要な個別機能をあげれば次のようなものがある。

1. 高級言語プロセッサ

2. 組み込み型のシミュレータ

3. 個人用データ・ベース

4. 多種入出力機器

5. 知識組み込み型高級インタフェース

6. 自動カスタマイズ／学習機能

7. 非数値（記号）処理

以上のような要求を，新アーキテクチャとＶＬＳＩを用いて実現する構造は次のように考えられる。

① 各種ＶＬＳＩチップの組み合わせとしてパーソナル・マシンを実現する。

オプションを豊富に用意し，シリコン・チップ・ライブラリを形成する。高信頼度が要求され，チップの標準化が必要になる。

② 推論機能チップの実現

述語論理マシンをＶＬＳＩによって実現し，個人用の知識ベースを置いて，高度且つカスタマイズされたインタフェースを実現する。

③ 分散データ・ベースのフロント・エンド機能

個人用データベースを置き，上位レベルシステムに置かれた分散データベース仮想化インタフェースを利用する。データベース処理には関係代数型マシンチップを利用し，ローカル網には高速回線が必要になる。

④ 非数値処理向きプロセッサ

2.の推論チップとも関連するが，パーソナル・マシン内では数値処理よりも記号処理のような非数値処理がより重要になると思われ，Lisp マシン等の関数型マシン・チップが有効であろう。

⑤ 高速演算用のオプション・チップ

②の豊富なオプションに含まれるが，オプションとしては例えば，ある程度の高速演算を手軽に実行する為にそれ用のオプション・チップを，Systolic Array 等のVLSIによって提供することが考えられる。

5.3.2.2 マシン・サービス・マシン

マシン・サービス・マシンは，ネットワークのノードを構成する要素である。通信回線によって接続された装置（コンピュータ，端末等）の間で情報を正しく送受信することが要求される。

マシン・サービス・マシンの機能は，回線制御機能と伝送制御機能からなる基本機能と，高度なネットワーク制御や情報の変換機能，暗号処理等行なうサービス機能に大別される。一般に，マシン・サービス・マシンの機能階層は，ネットワーク・アーキテクチャの階層構成に対

応する。

① 基本機能

a. 回線制御機能

・ 通信回線の接続・切断・監視・同期確立，データビットの組立て，データビットへの分解等を行なう。

b. 伝送制御機能

・ データリンクの確立／終結，データ送受信フォーマット，受信データに対する応答形式／応答方法，エラー回復方法等を規定した伝送制御手順に従って，データブロックの送受信を行なう。

② サービス機能

a. 蓄積機能

b. 変換機能

・ データの圧縮，復元，通信メディア変換，プロトコル変換等を行なう。

c. ネットワークの保守・管理機能

d. 暗号処理，機密保護機能

e. ネットワーク・サービスの仮想化機能（ネットワークＯＳ機能）

・ ネットワーク・ジョブ管理機能

・ 分散データベース（ＤＢ）管理機能

－ＤＢのスキーマ変換（ローカル／グローバルスキーマ）

－Commitment／Concurrency制御サポート

－分散ディレクトリ管理等

以上のような機能を，新アーキテクチャとＶＬＳＩを用いて実現する構造は次のように考えられる。

① 回線制御機能／伝送制御機能のＶＬＳＩチップによる実現（通信用ＶＬＳＩ）

ネットワーク・アーキテクチャの国際標準化に伴ない，基本機能のＶＬＳＩ化やロジックのファームウェア化が進む。

② 変換処理および暗号処理のＶＬＳＩチップによる実現

ＶＬＳＩアルゴリズムを利用し，特殊用途チップとして実現する。

③ メディア変換では，必要に応じて知識データベースと述語論理マシン，関数型マシン，新ノイマンマシン等を利用する。

④ ネットワーク・サービスの意味記述と探索機能の実現のため，述語論理マシン，関数型マシン，関係代数マシン等を利用する。スキーマ変換にもほぼ同じ新技術が利用される。

⑤ 高信頼性，高性能ネットワークの実現光通信方式，衛星通信方式を導入するとともに，

マシン・サービス・マシンをマルチプロセッサ構成とする。また，データフローの概念を利用する。

5.3.3.2　コントロール・マシン

コントロール・マシンは機能分散型システムを円滑に運営してその機能，性能を発揮させるためのマシンで，

- システムの効率の向上
- システムの使い易さの向上
- システムの信頼度の向上

などを実現するための要となる。現在のシステムにおいてはこれらの機能はＯＳに集中されており，そのオーバヘッド（時間的およびシステム資源）は無視できないほどに増大して，その改善は重要課題になっている。機能分散型システムにおいては，この機能をはたす専用マシン（コントロール・マシン）が必要となるが，このマシンの性能はシステム全体のスループットを左右するので，システムコントロールに適したアーキテクチャを持った高性能のマシンが要求される。

コントロール・マシンに要求される機能は次のようである（詳細は3.4.5参照）。

a. ジョブ管理機能
b. タスク管理機能
c. システム資源管理機能
d. デバッグ・サポート機能
e. 高度のＨＥＬＰ機能
f. 保守信頼性制御機能
g. 自動運転制御機能
h. システム監視機能
i. 構成制御機能
j. モニタリング及び最適化
k. 入出力制御
l. ファイル制御
m. 仮想記憶制御
n. その他
   ケーパビリティ制御
   ガーベージ・コレクション
   抽象データ構造サポート

以上の要求を満たすコントロール・マシンは次のような構成になろう。

(1) 基本構成

コントロール・マシンはその主要機能別に専用化された複数のマシンで構成され，それ自身で機能分散型構成をとる。これはシステム制御・管理機能はかなり異質の機能が含まれていて全体を1台でカバーするより，複数の専用マシンで分担した方が高性能化できることによる。

基本構成としては次の四種のマシンを考えるがこれらはシステムの規模，性能に応じて1台で複数のマシンの機能を分担する場合もある。

- ＯＳマシン：前述の機能のうち，主にａ～ｅを分担するマシン
- システム管理マシン：同上ｆ～ｊを分担
- メディア制御マシン：同上ｋ～ｌを分担
- 記憶制御マシン：同上ｍ～ｎを分担

これらの中で特にＯＳマシンについてはその機能が各機能分散プロセッサに分散され，独立したマシンとしては存在しないとの考え方もあるがここでは独立したマシンを想定する。

(2) ＯＳマシン

ＯＳマシンは数値計算的な処理はほとんどなく，制御の切換え，キューやテーブルの管理，状態情報の退避／回復などが中心である。これらを効率よく行うには，現アーキテクチャ（フォンノイマン・アーキテクチャ）の改良の範囲で考えるならば（これを以下新ノイマン機能とよぶ），次のような方法が考えられる。

- ファームウェア化：現在ソフトウェア（機械語レベルプログラム）で処理している機能の中で特に頻繁に使用される部分をファームウェア化して高速化する。これによって（ハードウェア構造にもよるが）ファームウェア化した部分は数倍に高速化される可能性がある。
- スタックの利用：状態情報の退避／回復などにプッシュ・ダウン・スタックが適しているが現在ではソフトウェア制御で普通のメモリを用いて実現しているマシン（商用マシン）が多い。これをハードウェア化して活用すれば性能を改善できよう（研究的マシンや一部商用マシンでは導入されている）。
- 大容量アドレス空間の管理・制御：メモリチップの大容量化，コンピュータの高性能化，利用の高度化にともなってコンピュータ・システムの記憶容量は今後著しく増大すると考えられる。したがって，ＯＳマシンは大きな実メモリ及び仮想メモリを効率よく管理できねばならない。これにはたとえば大きなアドレス空間を指定するためのアドレス生成／命令語の形式などに新しい工夫を必要とする。

  また，今後予想される新機能（新アーキテクチャ）やハードウェア・テクノロジイの利用については次のように考えられる。

- ＶＬＳＩ技術：ＶＬＳＩ技術によって論理機能付きのメモリチップ，たとえば大容量

の連想メモリなどが実現されればテーブルを検査するなど多くの利点が考えられる。この場合従来のアーキテクチャでは性能が低下してしまうのでハードウェア・アーキテクチャ上のサポートが必要である。

(3) システム管理マシン

このマシンは現在のサービス・プロセッサが高性能・高機能になったものでハードウェアに密接した各種の制御が中心となる。しかし，その機能は多様なものが要求され高度の融通性・拡張性が必要である。

現在の新ノイマン機能としては次のようなものが考えられる。

- 診断機能：ハードウェアの障害情報の収集，診断のためのデータの送出と結果のとり込みおよび分析などを効率よく行うもので，現在のアーキテクチャでも比較的うまく処理できるであろう。対象マシンとのインタフェースの構成などに新しい工夫が必要である。
- 自動運転機能：コンピュータ・システム運用の省力化，無人化の要求にこたえるために必要になる機能であるが処理プロセッサだけでなく対象装置側での対応，制御インタフェースなども重要な問題である。
- モニタ機能：動作状況のデータ収集と最適化のためのフィードバックを行う。どのようなデータをとり出し，どうフィードバックするのかなどが研究課題であるが，処理プロセッサとしては現アーキテクチャのマシンでも十分に利用できよう。

システム管理マシンを将来の技術で構成するとすれば次のような事が考えられる。

- 動的最適化機能：システムの動作環境に応じて動的にシステムの特性を変化させスループットやレスポンス・タイムを改善する。システム管理マシンはそのためのデータ収集や分析をやってフィードバックする中心となるが，動的最適化の機能自身はシステム管理マシンによって制御される各機能分散プロセッサ側に必要とされる。ファームウェアやソフトウェアの構成を運用状況に応じて動的に変化させるadaptable machineの概念がそこに取り入れられるであろう。
- ＶＬＳＩ技術：ＶＬＳＩによって実現される高性能のマイクロプロセッサはシステム管理マシンを構成するハードウェアの中心となる。また，大容量の連想記憶が実現されれば性能向上のために各種の利用が可能であろう。

(4) メディア制御マシン

メディア制御マシンはいわゆる入出力プロセッサを高機能化したマシンでソースデータの入力，結果の出力，ファイルへのアクセスなどを効率よく実行することを狙いとしていて，新ノイマン機能としては，ファームウェアの利用，高速チャネルなどが考えられる。

- ファームウェア化：入出力処理に付随する割込み処理，データのコード変換など定型

的でしかも高速を要求される部分についてはファームウェア化によって高速化することがよい解決法となろう。

- 高速チャネル：今後の入出力／ファイル装置の高速化に対応してチャネルの高性能化，インタフェースの高速化が必要であり，データ転送方式の新しい工夫，チャネルや入出力装置，ファイル装置のインテリジェント化などが行なわれよう。
- その他：割込情報の格納などにスタックが用いられよう。

さらに将来のマシンでは次のような構成が考えられる。

- ＶＬＳＩ技術の活用：メディア制御マシンにおいても連想記憶，高性能マイクロ・プロセッサなどはハードウェア構成の中心となる。
- データフローマシン：データ駆動，要求駆動でＩ／Ｏタスクを管理していくためにデータフローマシンの概念が利用できよう。

(5) 記憶制御マシン

記憶制御マシンは仮想記憶，実記憶の管理および記憶保護，アクセス制御などを行うマシンで次のような構成をとることが考えられる。

- ワンレベル・ストア：メディア制御マシンとの協力でワンレベル・ストアを実現してファイルとメモリ空間の一元的管理をする。このために，仮想記憶機能，記憶保護，アクセス制御などの新ノイマン機能をもったマシンとなる。
- タグ・アーキテクチャ：タグ・アーキテクチャは記憶容量が（タグの分だけ）余計に必要になる点を除くと多くの利点を持っており，機能分散型システムにおいて利用される可能性がある。その場合タグに関係した制御の一部は記憶制御マシンによってなされることになろう。同様にケーパビリティ制御についてもその機能の一部は本マシンが分担する。

将来の新機能としては記憶制御マシンによる抽象データ構造のサポートなどが考えられよう。

なお，記憶制御マシンは高い性能が要求されるので処理はプログラムではなく布線論理と一部ファームウェアの組合せで実現されることになろう。

(6) サブシステム間結合，その他

システム・コントロール・マシンのサブシステム間結合には高速バスの技術が必要で光ファイバを利用することも考えられる。

また，システムの管理のために各構成要素との間でインタフェースが必要であり，個別にケーブルを張るとケーブルが多くなりすぎるのでループ・インタフェースなどが導入されよう。

また，システム・コントロール・マシンには高い信頼性が要求されるのでFault Tolerant Computer技術の導入が不可欠である。

5.3.2.4 ユーザ・サービス・マシン

(1) 高級言語マシン

主として現在広く使用されている言語をもつと効率よくサポートするマシン構成がまず問題となる。それにはまったく新しいアーキテクチャを導入するアプローチもあるがここでは，フォンノイマン型アーキテクチャの強化，改良で対処する場合について述べ，まったく新しいアーキテクチャは新言語のサポートとの関連で導入されるとして述べる。

- タグ・アーキテクチャ：データの属性とプログラムの独立性を保つためおよび記憶保護，アクセス制御などにタグ・アーキテクチャは有利であり高級言語マシンに適している。従来はタグの分だけ余分にメモリが必要になることなどが難点とされていたがメモリ価格の低下によってこの問題は軽減されつつあるといえよう。
- デスクリプタ：タグと同様にデータ属性の独立した定義が可能，アクセス制御などに有効であるほか，データのアロケーションにも使用できる。高級言語マシンに適したアーキテクチャとして導入される可能性が大きい機能である。
- スタック：スタックは高級言語マシンにかぎらず有効な利用ができるが，特に高級言語指向のマシンでは利用価値が高く，B5000をはじめとして多くのコンピュータで採用されてきている。
- 仮想記憶／ワンレベル・ストア：仮想記憶はプログラムの作りやすさ等の観点から必要不可欠な機能であるがさらにファイルまでも仮想記憶の空間にとり入れてメモリをワンレベル化することで使い易さが改善されると思われる。現在は商用機レベルでは例が少ないしＴＳＳ（MULTICS）や小型機（ＩＢＭシステム３８）に限られているが今後は大型機領域まで適用が拡大するであろう。
- ＵＨＭ（Universal Host Machine）：ユーザの使用する言語は複数あり，各言語毎に専用の高級言語マシンを設けることはかならずしも許されない（コスト面から）ので，１台のハードウェアで複数言語を処理することが要求される。従ってＵＨＭの概念が必要であり，ファームウェアの切換えだけでなく特定言語専用のハードウェア回路の追加で性能を高める工夫も必要であろう。

新プログラミング言語をサポートする高級言語マシンにおいては上述のような新ノイマン機能をもったマシンの他にデータフローマシンや抽象データ型マシンが必要となろう。新プログラミング言語とこれらのマシンとの対応を図５－６に示す。実際の問題を解決するためのプログラムはたとえばPROLOGのような言語でかかれ，それらは関数型マシンおよびデータフローマシン上で実行する形にコンパイルされる。関数型マシンは実際には，新ノイマン・マシンやデータフローマシンの上に仮想マシンとして作られており，ハードウェア上にファームウェアで作られるのは新ノイマン・マシン，データフローマシンなどであろう。

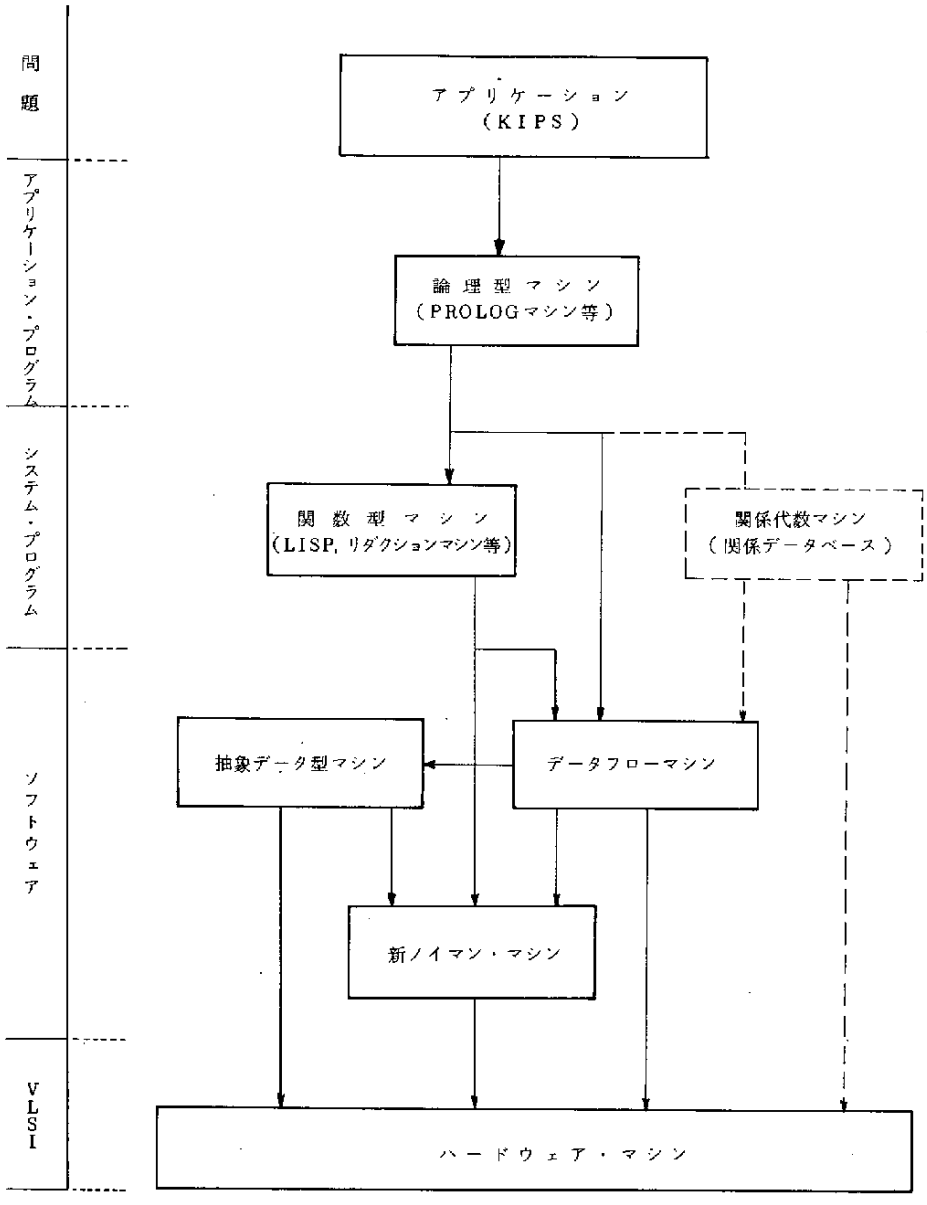


図５－６　新言語と各種マシンの関係

したがって高級言語マシンの構成としては前述の新ノイマン機能とさらにデータフローマシン，抽象データ型マシンが組合わされたもので，恐らく新ノイマン・マシンは不可欠の要素となりそれを補う形でデータフローマシン（データフロー機構）や抽象データ型マシン（抽象データ・サポート機構）が導入されよう。

- 新ノイマン機能：前述の他ＶＬＳＩ技術を利用した大容量の連想記憶などが必要となろう。
- 抽象データ型サポート機構：タグ・アーキテクチャ，ワンレベル・ストアなどを含む構造メモリ。
- データフロー機構：ＶＬＳＩ技術による高性能演算回路を多数用いた並列処理など。

(2) データベース・マシン

現在技術の延長で考えられるデータベース・マシンの構成は次のようである。

- 高速メモリ：ＶＬＳＩ技術によって得られる大容量メモリ（ＲＡＭ，ＣＣＤ，バブルなど）はデータベースの処理を高速化するためのバッファ・メモリ（ディスク・キャッシュとしての使用など）や固定ヘッド・ディスクにかわる非回転型電子ディスクとして使用される。
- マイクロ・プロセッサ：高性能で安価なマイクロプロセッサの出現によりこれを多数用いて並列処理による高速化を狙う。たとえば磁気ディスクの各ヘッド毎にマイクロ・プロセッサを付随させ，磁気ディスクの複数トラックの内容を同時に検索することによって，処理を高速化することが考えられる。これは論理セル方式とよばれ，磁気ディスクを時間遅れのある連想メモリ化するものである。（図５－７参照）。

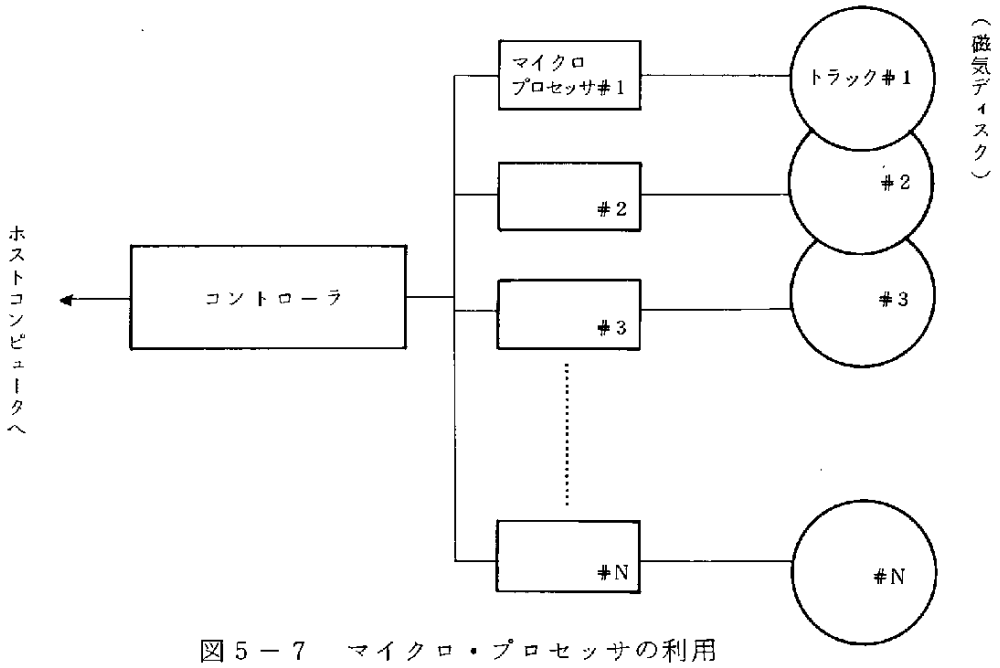


図５－７　マイクロ・プロセッサの利用

- ファームウェア：データベース・マシンの処理プロセッサは処理内容が固定化されているのでファームウェアによって特定の機能を高速化した専用マシンとすることが容易である。特に頻繁に利用される機能を中心にソフトウェアをファームウェア化するか，方式によってはデータベース・マシン全体がファームウェアで制御されることも考えられる。
- セキュリティ機能：機密保護のために暗号が使用されると予想され，ＬＳＩ化された暗号化／復号化回路がデータベース・マシンに組込まれよう。
- ハッシング機構：ハッシングを用いて関係データベースにおけるＪＯＩＮなどの演算を高速化することができ，このためにハッシュ関数を実行するハードウェアが導入さ

れる。

さらに将来の新技術（新アーキテクチャ，新デバイス）を用いたデータベース・マシンでは次のような構成がとられる（ただしデータベース・マシンの方式には種々のものが提案されており，以下にのべるものはそれらを区別していないので1台のマシンにすべてが導入されるという意味ではない）。

- 関係代数マシン：データベース・マシンの本命は関係データベースをサポートするものといわれている。これは関係モデルが集合論に基づいて数学的にきれいに整理されていることと，複雑な演算が多くてハードウェアでサポートした時の効果が大きいことによる。関係データベースをサポートするには集合演算を高速化した関係代数型マシンが適すると考えられ，新しいアーキテクチャをもったプロセッサが中心となる。
- データフロー機構：データをパイプライン中に流し，サーチ，ソートなどの処理を高速化することが考えられよう。
- ＶＬＳＩアーキテクチャ：単純な比較機能をもったセルを二次元的にならべて，２つのリレーションのタプルをアレイの両側から入れると徐々に演算結果がアレイから出力される（KungのSystolic Arrays）という方式で比較，積，ＪＯＩＮなどの集合演算を高速に処理することができる。このようなチップはＶＬＳＩ技術で実現するのに適しており，データベース・マシンに導入されよう。
- ＭＩＭＤ：前述のマイクロ・プロセッサを並列にならべた方式では全プロセッサは同時に同一命令を実行するいわゆるＳＩＭＤ型になっている。これでは複数ユーザから同時に問合せがあったときなどは逐時処理になってしまうので各プロセッサが独立した命令シーケンスを実行できるようにして性能を向上することが考えられる。
- 連想プロセッサと連想メモリ：連想処理によって大容量データの検索が高速に出来る。したがってＪ－Ｊ素子などで大容量，高速の連想メモリが出来ると高性能のデータベース・マシンが実現される。コスト的には高価になるがオンライン・データベース・システムのように高速性が重要な場合には導入されよう。
- 高機能化：推論機能をもちあいまいな問合せに対しても答えることができるようなデータベース・マシンを実現するために人工知能で研究される新しい考え方やアーキテクチャが導入される。
- 電子ディスク：大容量の磁気バブル・メモリによる電子ディスクが磁気ディスクに対抗できるビット・コストで実現されるとデータベース全体をこれに格納することにより高性能，高信頼化できる。また，磁気ディスクを使用する場合に比しデータベース・マシンの構成もずっとシンプルなものに出来る可能性があるし，マイクロ・プロセッサとの組合せで並列処理（並列検索）する方式においてもマイクロ・プロセッサ台

数を非常に多くできる可能性をもっている，など利点が多い。

(3) 超並列計算（ベクトル，行列計算等）マシン

超並列計算マシンは，科学技術計算の進歩とともに，大規模化・複雑化する科学技術計算を，並列処理技術を駆使して高速処理するためのマシンである。主として，限られた時間内での処理が要求される場合に，回路素子で達成できる以上の高速処理を実現するために必要とされ，次の分野で使用される。

① 資源探査（石油探査，地震波解析，等）

② 気象予報

③ 航空工学（航空管制も含む）

④ 原子物理学（原子炉設計，等）

⑤ 画像処理

⑥ 構造計算（船体構造，ダム構造設計，等）

⑦ 宇宙航空工学（人工衛星やミサイルの軌道計算，等）

今後は，特にベクトル，行列計算問題に帰着できるシミュレーション・システムの中で多用されると考えられる。

超並列計算マシンに必要とされる主な機能には，次のようなものがある。

① ユーザは，マシンでどのような並列処理が行なわれるかを意識することなく，高級言語でプログラムを記述できる。なお，特定の問題では，パラメータを入力するだけでよい。（プログラミング・レス）。

このため，ユーザプログラムの中から並列性を自動検出し，マシンの並列構造にプログラムを写像する機能を持つ。

② あるいは上記とは逆に，計算規模やプログラムの持つ並列構造にマシンを tune させるため，マシンを構成する要素（演算装置，等）と，要素間接続方式を可変にできる機能を持つ。また必要に応じて，ある特定の問題処理用に特化された専用ハードウェアをマシン内に組込むことができる機能を持つ。

③ 計算式の正当性，解の収束方向と収束速度，入力パラメータ（初期値，等）の妥当性，等をチェックするために，中間結果を必要な時点で確認できる機能を持つ。

④ 高い信頼性を確保するため，自動再構成（ハードウェア，ソフトウェア）機能を持つ。

⑤ レベル1マシン（パーソナル・マシン，等）から容易に利用できる。また，シミュレーションの場合には，結果を3次元カラー・ディスプレイ装置などに人間が理解しやすい形（アニメーション，等）で表示する機能を持つ。

⑥ 演算速度は $10^3 \sim 10^4$ MFLOPS，メモリ容量は $10^2 \sim 10^3$ M語程度である。

超並列計算マシンの実現のために，新アーキテクチャやＶＬＳＩアーキテクチャが例えば次のように関係して来よう。

① データフローマシンの利用により，並列度を最大限にとり出す。

② 時間的並行処理（パイプライン）方式では，パイプライン・ピッチを小さくするため，高速の演算素子を利用する。素子として，例えばGaAs素子，ジョセフソン接合素子を用いる。

③ 空間的並列処理方式では，ＶＬＳＩ向きアルゴリズムの開発と，そのアルゴリズムを実現した演算チップを多数配置して利用する。あるいは，多数のマイクロ・プロセッサを複合体とする。この方式では，計算すべき問題の持つ並列性を直接アーキテクチャに写像できるようにするため，演算チップや，マイクロ・プロセッサの結合パスを可変とする。

④ さらに，時間的並列処理方式と空間的並列処理方式を組み合わせた方式とする。

⑤ 高速大容量ＶＬＳＩメモリを利用する。

⑥ 機能モジュール間接続に光パスを用いる。

⑦ ジョブ転送プロトコル等，ネットワーク・アーキテクチャの標準化が必要となる。

⑧ シミュレーション結果をアニメーション表示するため，大規模画像メモリや，画像処理用チップを利用する。

(4) 超高速直列計算マシン

超高速直列計算マシンは，主として，並列処理が適用できない問題を高速処理するために必要とされる。

超高速直列計算マシンは，素子技術に負うところが大きく，マシンのアーキテクチャは，他のユーザ・サービス・マシン（超並列計算マシン，データベース・マシン，等）より単純である。すなわち，従来のノイマン型マシンを改良したものである。

超高速直列計算マシンの導入により，先に述べたように，シーケンシャル・アルゴリズムの高速実行が可能になるという以外にも，次のような効果が期待できる。

① ソフトウェアの作成・デバッグ・保守等が容易になる。これは，構造化プログラミング等のソフトウェア生産性向上技術によりソフトウェア・オーバヘッドが増加しても，マシン自体が高速のため，特に問題とならないからである。

② アーキテクチャが比較的単純であるため，ハードウェアの保守が容易になる。障害回復（ソコト，ハード）も容易である。

③ 従来のノイマン型マシン上のソフトウェアとの高いコンパティビリティを保証できる。

超高速直列計算マシンの実現のために，新アーキテクチャやＶＬＳＩアーキテクチャが例えば次のように関係して来よう。

① 高性能シリコンＶＬＳＩを利用する。

② シリコンに比べ，常温でも数倍程度の大きなモビリティを持ち，しかも半絶縁性基板が利用できるGaAs素子を利用する。

③ GaAs，GaAlAs，InP，InSb等を基本材料とした光素子を利用する。

④ 機能モジュール間接続に光パスを用いる。

⑤ 超伝導現象を利用することにより，スイッチングの高速化と低消費電力化を達成できるジョセフソン接合素子（ＪＪ素子）を利用する。

⑥ 冷却形シリコンＶＬＳＩ素子およびGaAs素子を利用する。

⑦ 配線遅れ時間を最小にするため，高密度実装技術を採用する。

⑧ 大容量ＶＬＳＩメモリを利用するとともに，電子ディスク（例：ＣＣＤメモリ，磁気バブル・メモリ），大容量高速ファイル・メモリを利用し，新たなメモリ階層構成を採用する。

⑨ ＯＳ機能のハード化を進める。例えば，割込み情報格納のためのスタックやポインタ管理のハード化や，プロセス・スイッチ用命令を実現する。また，論実アドレス変換処理のために，連想メモリを利用する。

⑩ ダイナミック・マイクロ・プログラミングにより，ＩＳＰレベルを可変構造とし，セマンティック・ギャップを狭める（ＵＨＭの利用）。

(5) 大規模シミュレーション・マシン

大規模シミュレーション・マシンは，主として次のように利用される。

① 大規模実験設備に替るコンピュータ・シミュレーション（金額的，時間的負担が大きいため，大規模実験設備の作成によるシミュレーションができない場合。例：風洞シミュレーション）

② 設計支援システム（ＣＡＤ）用シミュレーション（正しい設計を行えるようにする支援システム）

③ 社会・経済・環境等のシミュレーション（人間生活に密着したテーマを扱い，具体的な決定に伴う問題や危険性を事前に回避したい場合。例：高層ビル火災時の避難方法シミュレーション）

④ 経営問題のシミュレーション（企業における各種意志決定のためのシミュレーション）

大規模シミュレーション・マシンに必要とされる機能には，次のようなものがある。

① シミュレーション・モデルの自動生成，およびシミュレーション・モデルの自動検証機能

② シミュレーション・モデルの実行を高速に行ない，適切な応答時間を保つことができる機能（現在の最高速マシンの１０倍程度の処理能力では不十分な場合がすでに存在している。例：原子炉シミュレーション，風洞シミュレーション）

③　シミュレーション経過または結果を，人間が理解しやすい形態で表示できる機能（カラー・グラフィック表示機能，等）

④　観測データから規則性を発見し，新たなモデルの確立や，既存モデルの修正を行う機能

⑤　遠隔地のパーソナル・コンピュータや入出力装置との接続機能

大規模シミュレーション・マシンの実現のために，新アーキテクチャやＶＬＳＩアーキテクチャが例えば次のように関係して来よう。

①　並列計算（ベクトル，行列計算，等）に帰着できる場合は，超並列計算マシンと同じ技術が利用できる。

②　並列計算に帰着できない場合は，超高速直列計算マシンと同じ技術が利用できる。

③　離散形シミュレーションでは，マルチ・マイクロ・プロセッサ構成とし，データフローの概念（トランザクション・ドリブン）を利用して並列処理を行なう。

④　シミュレーションに必要とされる非数値計算（記号）処理の高速化を図るため，関数型マシンを利用する。また，集合演算の高速化のため，関係代数マシンや，ソーティング用ＶＬＳＩチップを利用する。

⑤　推論等を含む高度なシミュレーションを実現するため，知識ベースと，述語論理マシン，関数型マシンを利用する。

⑥　ＵＨＭを利用する。

⑦　マルチメディア入出力によるシミュレーションを可能とするため，レベル１マシン内やネットワーク内に，音声理解／応答装置やメディア変換装置が組み込まれる。これらの装置では，ＦＦＴ処理，画像の前処理／後処理，ディジタル・フィルタリング処理等のためのＶＬＳＩチップや連想メモリが使用される。

⑧　正しいシミュレーション・モデルを自動生成したり，モデルの検証や自動修正を行なうために，知識ベースや推論が必要とされ，述語論理マシン，関数型マシン，関係代数マシン等が利用される。

⑨　ネットワーク・アーキテクチャの体系化・標準化が必要とされる。また，高速回線も必要である。

(6)　大規模知識ベース・システム

データベースと知識ベースはこれらを一体化して考えることも出来ようが，その各特徴を考えれば区別した方が自然であろう。すなわち，データベースは，何らかの数値のセットのように単なるデータの集合であって，その使用形態はスキーマ記述によってほぼ限定されている。またその構成に当っては記憶効率やアクセスの速さが非常に重要であり，データ自体の意味については余り問題にせずシステム内にその情報は余り置かれていない。一方，知識ベースは，

知識というより一般的普遍的なデータを対象とし，その使用形態は様々である。

従って，これらは今後ともかなり独立して存在すると考えられるが，スキーマ情報にデータのセマンティクスを記述してゆこうという動きがあり，データベースの使用に関しては知識ベースがより密接に関連して来るものと思われる。従って将来的に見れば，データと知識とは外から見ると隔合して見えるようになるかも知れないが，内部的には区別された形で存在する期間が長く続くものと思われる。また，分散データベースの実現には知識ベースが必要であり重要な役割を果たす可能性がある。知識ベースの内，人間とのインタフェース改善に関わりあいがあり，又個人的色彩の濃いものは，パーソナル・マシン内に置くことが考えられるが，より一般的で量の多い知識は上位レベルシステム内に置かれる。これをここでは大規模知識ベースと呼んでいる。

大規模知識ベースは，上位レベル・マシン群によって実現されるが，知識の要素としては例えば次のようなものが考えられる。

① シソーラス

② 辞典（百科，英和，国語etc）

③ 公式集（数学，物理，化学etc）

④ 文法（日本語，古文，英語etc）

また，これらの使用形態としては例えば次のような形があろう。

① 電子物識り博士

② データベースのスキーマ記述を補う意味ベース

③ 知的思考の補助

こうした大規模知識ベースを用いて単語の意味や関係を与えることになる。実際上の構造としては各地に幾つかの知識ベースが置かれ，分散知識ベースの形態を取ることになるものと思われるが，各所知識相互の関連付けは，日本語等の共通概念をベースにして再帰的に（関連を与える為の知識間の関連）おこなわれる。

大規模知識ベースの実現には，新アーキテクチャやＶＬＳＩアーキテクチャが例えば次のように関係して来よう。

① 述語論理マシンの利用

並列処理を推論に導入し，高速化を図ることが必要である。

② 非数値処理マシンの利用

知識の処理に，関数型マシンを利用する。

③ データ・フロー概念の利用

関係した知識すべてが起動され，必要な処理を並列的に行なう。

④ 大規模連想メモリの利用

ＶＬＳＩを用いて構成し，十分実用に耐えるだけの容量と速度を持たせると，知識処理や推論には強力な武器となろう。

⑤　関係代数マシンの利用

関連データにセットとしての演算を施し迅速に必要なデータを作り出す。

⑥　ネットワーク・アーキテクチャ

網仮想知識ベースを作り上げる為には，ネットワーク・アーキテクチャ上の様々な標準化が必要である。

5.3.3　具体的な使用形態における処理の流れとアーキテクチャの関連

5.3.3.1　ＣＡＥ／ＣＡＤシステム

１９９０年代のＣＡＥシステムのイメージは次のようなものになろう（但し，種々の場合を含めて述べており特定のシステムではこの中から選択された装置，機能のみがサポートされる）（システム化技術研究分科会の報告書参照）。

・　システムの中心にはエンジニアリング・データベースがあって，各技術者に必要なデータや共通的に使用されるデータが蓄積されていて自由に利用できる。

このデータベースにはコード化されたデータのみならず，図形や画像も格納されている。

・　開発技術者は専用の端末装置（ワークステーション：ＷＳ）を介してこのデータベースを利用しながら次のような作業ができる。

基本計画：マクロなシミュレーションなどができる。

基本設計：部品の配置，装置の構造などの設計ができる。

形状設計：部品の形状，スタイリングなどの設計ができる。

・　ＷＳには次のような各種のマン・マシン・インターフェース機器およびプライベート・ファイルが付属している。

－音声入力装置
－手書図面入力装置
－三次元カラー・ディスプレイ
－高級ランダム・スキャン型ディスプレイ
－大型平板ディスプレイ
－ワードプロセッサ（日本語，英語）
－インテリジェント・コピア
－プライベート・ファイル／データベース

・　実験代替用シミュレータによる高度の数値シミュレーションで実験を代用できるあるいは実験不可能な事象のシミュレーションによる研究ができる。たとえば超音速風洞のシミュレーション（計算機風洞）や核融合のプラズマのシミュレーションなどができる。

このＷＳには大型の３次元ディスプレイが用いられてシミュレーションの途中の状態がカラー・アニメーションで表示される。またカラー・ハードコピー装置でそれらを即時にハードコピーとして出力できる。

・　プロセスのシミュレータにより新しいプロセスの創造をサポートする。たとえば代替エネルギー開発に必要とされるような巨大な複合型プロセスの創造に利用できる。このため実験プラントのラボラトリ・オートメーション・システムと接続してダイナミックなプロセス・シミュレーションもできる。ＷＳとしては実験代替用シミュレータと同様カラー・アニメーション表示可能なディスプレイが用いられる。

・　仕様書，取扱い説明書，プロポーザルなどの作成を助けるドキュメント作成システムがあり，外国語のドキュメントも容易に作成できる。そのための機能として，

－外国語の翻訳

－外国文字の入出力

－図形，画像，写真などの入ったドキュメントが作れる。

－音声で指示できる

－カラー印刷も可能

などがある。このためＷＳとしては，

－図形，写真などの入力装置

－音声入力装置

－手書図形の入力装置

－カラー・ディスプレイ

－カラー・プリンタ

－音声応答装置

を備え，さらに，コード化データ，ビデオデータ，マイクロフィルム，および音などのアナグロ情報を保管し，自由にアクセスできる多形態データベースを持っている。

これを実現するシステムの構成例を図５－８に示す。

このシステムは高速のバスに各種の専用のＷＳが接続され，それらをサポートする高性能の機能分散プロセッサと結合している。

・　ユーザ・サービス・マシン：各ユーザはＷＳを介してこのマシンからサービスをうけるがユーザ・サービス・マシンは必要に応じて専用プロセッサを使用する。

・　データベース・マシン：エンジニアリング・データベースを格納しており，ユーザ・サービス・マシンからの問合せに対して答えを与える。簡単な問合せはＷＳから直接データベース・マシンへ出すこともできる。

・　超高速技術計算／シミュレーション・マシン：ユーザ・サービス・マシンの処理能力をこえるような大規模な計算やシミュレーションを実行する。ここには特定モデルの専用シミュ

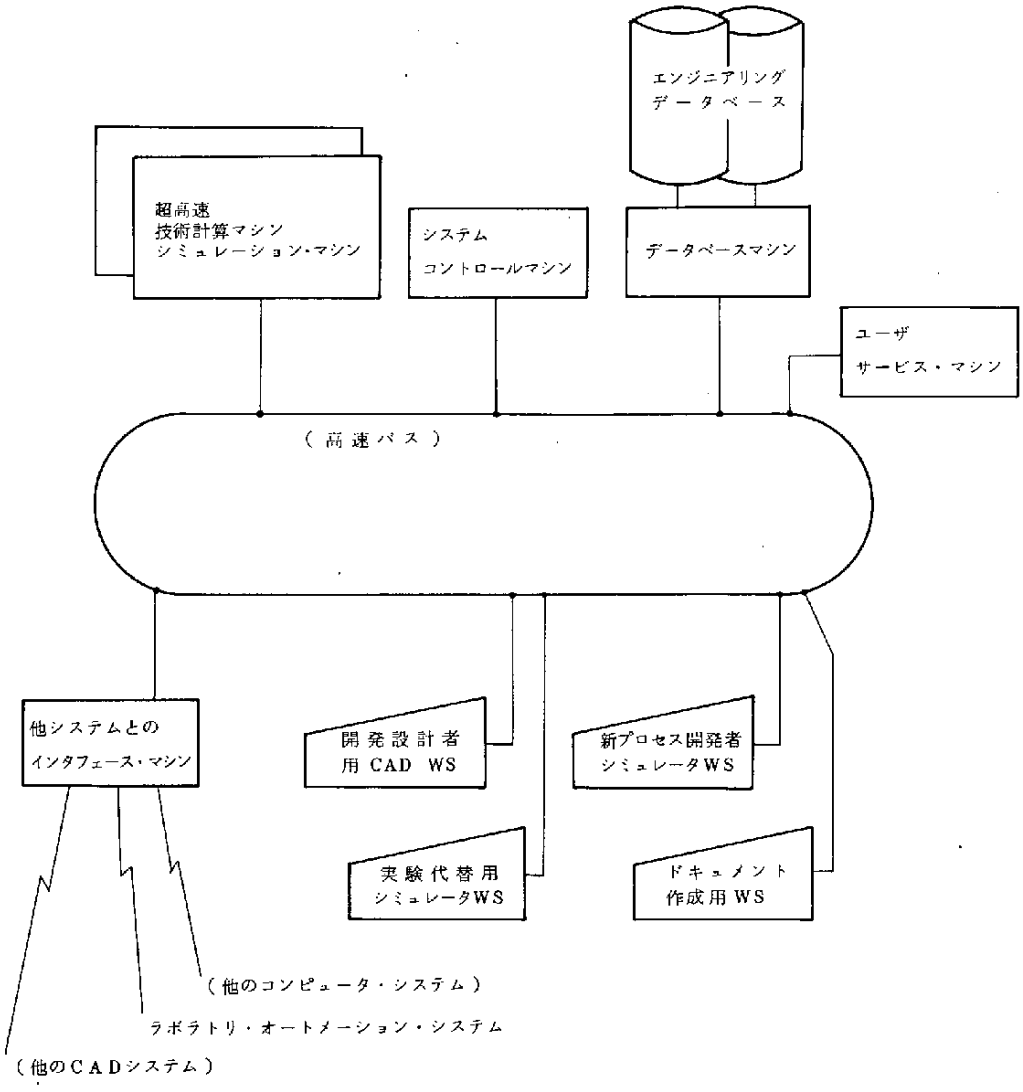


図５－８　ＣＡＥ／ＣＡＤシステムの構成イメージ

レータが存在することもある。

・　システム・コントロール・マシン：システム全体の制御や監視，ＲＡＳ機能などを分担する。

・　各種ＷＳ：各作業毎の専用ＷＳ

・　他システムとのインタフェース：ここを介して他のコンピュータ・システムとの通信，ラボラトリ・オートメーション・システム（およびその先の実験プラント）や他のＣＡＤシステム（市販のターンキー・システムなど）との通信が行なわれる。

・　高速バス：各サブシステムをつないでデータや指令を転送する高速伝送路。

次に，以上述べたようなシステムを構成する上で第５世代コンピュータの新技術がどのようにマッピングされるかについて考察する。

・　ユーザ・サービス・マシン：機能分散型システムのユーザ・サービス・マシンがそのままマッピングできる。現在の汎用コンピュータに相当する機能を果たすものでそれを実現する上では

－新ノイマン機能

－ＶＬＳＩデバイス（特にメモリ）

などが重要となろう。

・　データベース・マシン：関係データベースを処理するための集合演算の高速化などのほか画像，図形などを扱うための機能が必要となる。これらは現在のデータベース・マシンにはないもので第５世代コンピュータの新機能となろう。このシステムは全体として知識ベース・システムであることが要求され，データベース・マシンもそれをサポートする機能をもつことになる。

・　超高速技術計算マシン／シミュレーションマシン：超並列プロセッサ（時間並列，空間並列），データフロー・プロセッサなどが考えられ，それを実現する上でＶＬＳＩアルゴリズムが利用されよう。また素子の面では採来はＪ－ＪやGaAsを用いて超高速化がはかられる。

・　システム・コントロール・マシン：一般的な機能分散システムのシステム・コントロール・マシンが適用でき，それを実現するために高性能のマイクロ・プロセッサなどが利用されよう。

・　各種ＷＳ：これらの中心になるのは小型高性能のプロセッサで，マイクロ・プロセッサを機能分散型で接続して構成したマルチ・マイクロ・プロセッサ・システムなどが適用できるであろう。

またここに接続されている各種の入出力機器・マンマシン・インタフェースはいずれも現在実用化されていないものや現在の装置を飛躍的に高性能化したものであり，第５世代コンピュータの新しい入出力機器の技術が大いに利用されることになろう。

これらのＷＳの中でも特にドキュメント作成用ＷＳにおいては新しい入出力機器のみならず翻訳プロセッサや知識ベース／データベースが要求され，新アーキテクチャを備えたプロセッサが用いられる。

・　高速バス：光ファイバーによる高速ループ・インタフェースなどが用いられよう。

・　他システムとのインタフェース・マシン：機能分散システムのマシン・サービス・マシンが相当する。

以上のべたような本システムは本分科会で相定している機能分散システムをうまく適用することができる。

5.3.3.2　オフィス・オートメーション

図５－９に，１９９０年代のオフィス・オートメーション・システムのイメージを示す。（シ

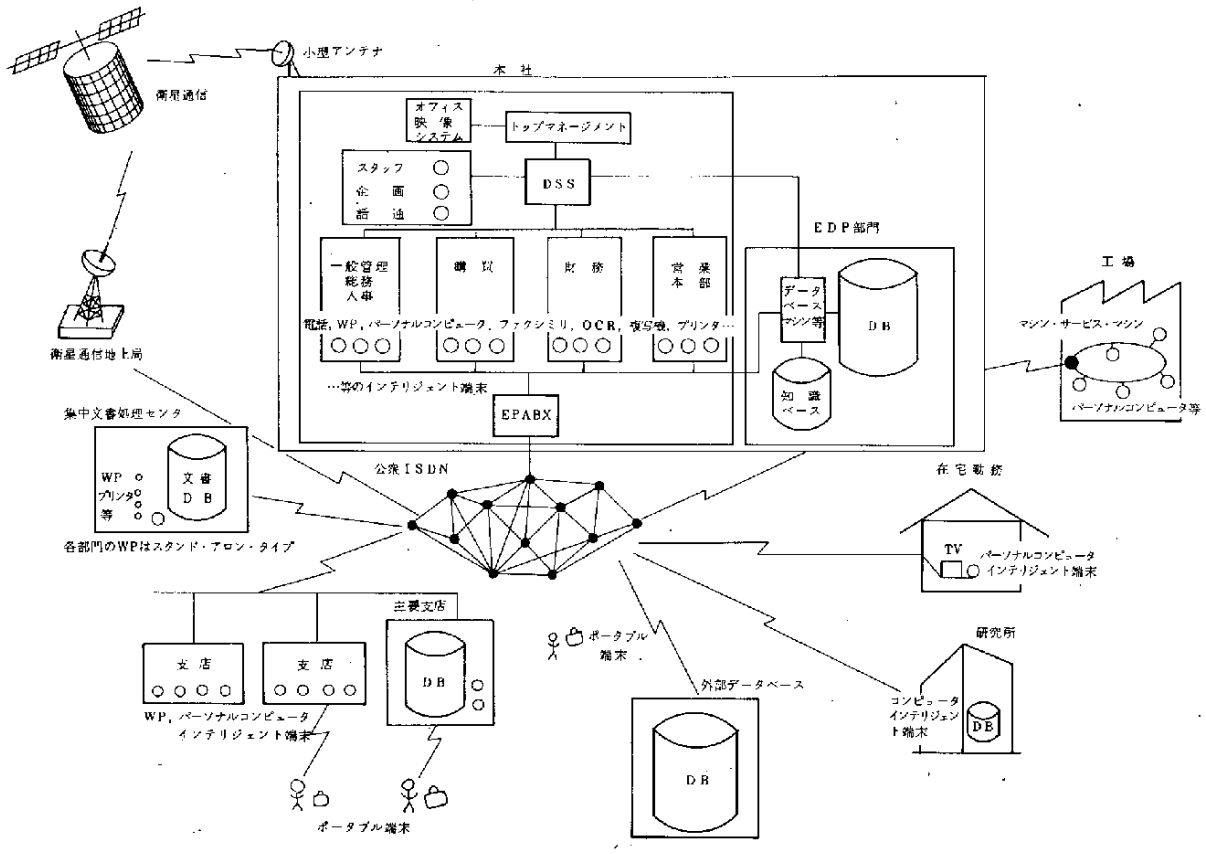


図５－９　ＯＡのイメージ

ステム化技術研究分科会報告書による。）

①　ＥＤＰ部門と，他のオフィス・フロアにおける機能が，有機的に結合される。

②　ディシジョン・サポート・システム（ＤＳＳ：Decision Support System）*により，各段階の管理者は，パーソナル・コンピュータや，簡単に操作できるインテリジェント端末を利用して，企業戦略等に必要な情報を，多重多様な形態で得ることができる。

③　マルチメディア・データベース（数値情報だけでなく，日本語文章，図面，写真，音声等の非数値情報を含むデータベース）が定着し，蓄積可能なあらゆる情報が，蓄積・検索される。

④　情報発生点で，原始情報をそのままの形で入力装置からデータベースへ投入できる。また，検索用のインデックスやシソーラスが自動生成される。あいまいな質問による検索も可能である。

これにより，フォーマルな情報やインフォーマルな情報（個人のメモ，スケジュール表，自

＊　ＤＳＳについては**5.3.3.3** で述べる。

己管理表，個人用住所録等）を迅速・的確に入手できる。

⑤　データベース機能とスケジュール／カレンダー機能とを組合せて，あらかじめ設定した期日に，自動的に必要情報を取り出すことができる。

⑥　自社の他（支店，研究所，工場等）データベースや異企業のデータベースとの間で，オンラインで情報を流通できる。

⑦　レス・ペーパーとなり，部門間，個人間の情報交換が電子化される。情報交換制御のため，ＥＰＡＢＸやマシン・サービス・マシンが利用される。また，タイプライタ，日本語ワードプロセッサ，複写機，印刷機，製図機等の事務機器には，通信機能が付加され，これらがネットワークに接続される。

⑧　文書作成，文書伝達，会議では，インテリジェント・ワークステーション，ワードプロセシング・システム，オフィス映像システム等が使用される。ここでは，音声，印刷文字，手書き文字，濃淡図形（写真を含む），色彩図形等の混在入出力と，これらの混在加工・処理が可能である。（マルチメディア処理機能）

⑨　電子メイル機能により，書類の郵送が迅速化・自動化される。また，メッセージをいったんメイル・ボックスに受け，手空きの時間に取り出す（非同期通信）ことができる。これにより，仕事の中断を減らすことができると共に，連絡を受ける場合に必ずオフィスにいなければならないという必要性を減少できる。

⑩　営業部門等の現場部門では，オンライン・ポータブル端末を利用して，いつ，どこからでも，情報を入出力できる。さらに，オフィス機能が分散しているため，在宅勤務や家に最も近いオフィスでの勤務が可能になる。

⑪　コンピュータ会議，テレビ会議が利用でき，出張のための時間や費用を削減できる。

⑫　高度のセキュリティ，プライバシー保護機能が実現されている。

オフィス・オートメーションの技術は，情報（文書）の作成，転送，出力，蓄積，検索，加工（処理）等の基本機能を組み合わせた技術である。以下では，これらの基本機能が，機能分散アーキテクチャ，ＶＬＳＩ技術，新アーキテクチャ等の新技術をどのように利用して実現されるかを述べる。

①　情報の作成と入出力機能

ⓐ　日本語ワードプロセッサ（文書作成）

・　日本語をパターン形式で直接入力するためにはⓑで述べる新技術を利用する。

・　単純な日本語文書作成機能の実現の場合には，汎用ＶＬＳＩプロセッサ（マイクロプロセッサやパーソナル・コンピュータ）の利用で対処できる。しかし，㋑日本語マニュアル入力支援機能（例：カナ漢字変換），㋺日本語文書のチェック機能（例：漢字や用語のチェック，構文や意味のチェック）等，を実現するためには，日本語辞書や知識ベース

を利用する。更に，高速処理のため，述語論理マシン，関数型マシン，新ノイマンマシン等を利用する。

・　作成中または作成済み文書の機密保護のため，文書暗号化用や個人識別用ＶＬＳＩチップを利用する。

ⓑ　マルチメディア入出力処理＊

・　各装置に，汎用マイクロ・プロセッサを組込むか，各装置をパーソナル・コンピュータに接続する。

・　ディジタル・フィルタリング処理，Ａ／Ｄ変換処理，ＦＦＴ処理，画像の前処理／後処理，パターン・マッチング処理，等のためのＶＬＳＩチップを，前述のプロセッサに付加する。

・　音声，印刷／手書き文字，画像，等の認識率向上のために，辞書や知識ベースを利用する。また，学習機能により知識ベースを修正・追加する。認識処理の高速化のため，新ノイマンマシン，述語論理マシン，関数型マシン，等を利用する。メディア変換処理（例：音声文字変換）もほぼ同じである。

・　各装置には，他装置，他コンピュータとの通信を可能とするコミュニケーション

・　アダプタ（マイクロ・プロセッサと通信用ＶＬＳＩチップから構成される）が付加される。

②　情報の加工，処理

ⓐ　日本語理解処理

・　日本語理解処理は，日本語ワードプロセッサにおける文書チェックや文書の自動合成（例：定型文を組み合わせて，新たな意味を持つ文章を合成する。－契約書等），後述する機械翻訳処理，日本語情報検索システム，等の中核になる。各種の知識ベース，新アーキテクチャ，専用ＶＬＳＩチップ等，新技術を最大限に利用して実現される。

ⓑ　機械翻訳処理

・　契約書や技術マニュアル等を，各国語に翻訳する。上述ⓐとほぼ同じ技術で実現される。翻訳マシン（新アーキテクチャ）と，専門用語辞書，各国語文法書，いいまわし法則や各種の知識ベース等を利用する。また，翻訳結果を人間が編集するのを容易にするため，高精度，高機能グラフィック・ディスプレイ等を使用する。

ⓒ　画像理解処理

・　図面作成，印鑑照合，画像データ検索（例：入力と類似のデザイン画を検索する）処理，ＦＡＸ情報処理，等を実現するために，画像処理向き高性能アレイプロセッサ，データフローマシン，画像メモリと一体化されたＶＬＳＩ完全並列型画像プロセッサ，微分

＊　主としてパターン情報について述べる。

・ＦＦＴ・情報圧縮用ＶＬＳＩチップ，大容量画像ファイル，等を利用する。

ⓓ　数値計算処理，シミュレーション処理

・　通常はパーソナル・コンピュータの利用で対処できる。超高速処理が必要な場合は，次に述べる新技術を導入したレベル２またはレベル３マシンを使用する。

・　並列処理が可能な場合には，アレイ・プロセッサやデータフローマシン等を利用する。また，並列処理が不可能な場合には，ＧａＡｓ，ＪＪ等の新素子を積極的に導入した新ノイマンマシンを利用する。

③　情報の蓄積・検索（データベース処理）

ⓐ　高度な情報検索システム

・　マルチメディアによる日本語情報検索システムが構築され，日本語でシステムと会話しながら情報検索ができる。このシステムの実現には，各種辞書や知識ベースを使用しながら，日本語構文解析，意味理解，推論等を行なう必要があり，日本語理解処理とほぼ同じ新技術を利用する。

・　マルチメディアを使用した検索を可能とするため，端末やパーソナル・コンピュータ，ネットワーク内に，音声理解／応答装置やマルチメディア変換装置（例：コード化情報を画像情報に変換する）等が組み込まれる。これらの装置では，処理速度向上のため，専用ＶＬＳＩチップ（例：音声合成チップ，メディア変換チップ），新ノイマンマシン，述語論理マシン，関数型マシン，等が利用される。

ⓑ　データベース自動構築

・　従来，データベースの構築にとって入力費用が大きく，この自動化には，費用削減効果が大きい。このため，情報発先源で，原始情報を直接データベースへ蓄積する高速マルチメディア入力端末を使用する。ここでの新技術は①ⓑとほぼ同じである。

・　利用者の要求に適合した情報がもれなく検索できるようにするため，原始情報の直接入力と共に，キーワードの自動抽出，シソーラス，抄録の自動作成を行なう。このためには，日本語理解処理とほぼ同じ新技術を利用する。

ⓒ　分散データベース

・　本社，支店，工場，研究所等に分散配置されているデータベースを論理的に統合するため，スキーマ変換処理，Ｃｏｍｍｉｔｍｅｎｔ　Ｃｏｎｃｕｒｒｅｎｃｙ制御，データベース内容の変換処理等を，ネットワークのノードとなるマシン・サービス・マシンや，データベースマシン，パーソナル・コンピュータ等で，機能分担して実現する。

ⓓ　大規模データベース，知識ベース

・　大規模データベース・システムにおける処理を高速化するため，処理装置の専用化が進む。数十～数千個のＶＬＳＩプロセッサや，連想メモリから構成されるマシンや，関

係代数マシン，等のデータベース専用マシンが利用される。（レベル２または３）

・　日本語理解処理等，知識ベースを使用する処理の高速化のため，更に，述語論理マシン，関数型マシン，等が利用される。

・　大量のデータ蓄積のため，大容量ＶＬＳＩメモリ，電子ディスク等が利用される。

④　情報の転送

・　オフィス内では，マイクロプロセッサ，ＶＬＳＩメモリ，通信用ＶＬＳＩチップ等が組み込まれディジタル化された，電子構内交換機（ＥＰＡＢＸ）や，マシン・サービス・マシンが，情報転送，交換の中核になる。これらには電話機だけでなく，複写機，ＣＲＴディスプレイ，ＴＶ受像機，ＦＡＸ，パーソナル・コンピュータ等が接続される。光ファイバを使用した高速ローカル・ネットワークも形成される。

・　広域にわたって，マルチメディアで情報を転送するため，ディジタル総合サービス網（ＩＳＤＮ）が使用される。ＩＳＤＮには，光通信方式や衛生通信方式が導入され，ＴＶ会議や電子メール等をサポートする。

・　高度の情報ネットワークの構築のため，ネットワークＯＳが実現される。

・　ＩＳＤＮに有線または無線で簡単に接続できる各種ポータブル端末が利用される。

⑤　その他

・　図５－９で示したＯＡシステムは，以上述べた新技術を利用した機能分散システムになる。

・　参考として，音声による情報検索システムにおける機能分散例を，図５－１０に示す。

5.3.3.3　意思決定支援システムの利用

意思決定支援とは，様々な高度の情報を人間に提供し，その人の思考過程を支援して最善の意志決定を可能とすることであるが，そのような支援システムの利用形態は幾つかのパタンに分けられる。その各パタンに付き処理の流れとコンピュータ機能との関連を分析すると次のようになろう。

(1)　情報の調査

簡単なコマンドの遣り取りで欲しい情報を捜し調べることで，これには次の様な機能を利用することになる。

①　コマンドの高級化／専用化

知識ベースを利用して常識を備えたシステムにみせる。不完全な情報より推論する為の推論マシン機構を利用する。システムは利用者個々の使用状況に自動的に適応する。

②　欲しいデータの同定援助

一般に欲しいデータとは何か自体が不明であるので，十分なデータ・ディクショナリ/ディレクトリ情報を備え，知識ベースを利用して利用者の欲しいものを捜し出す。

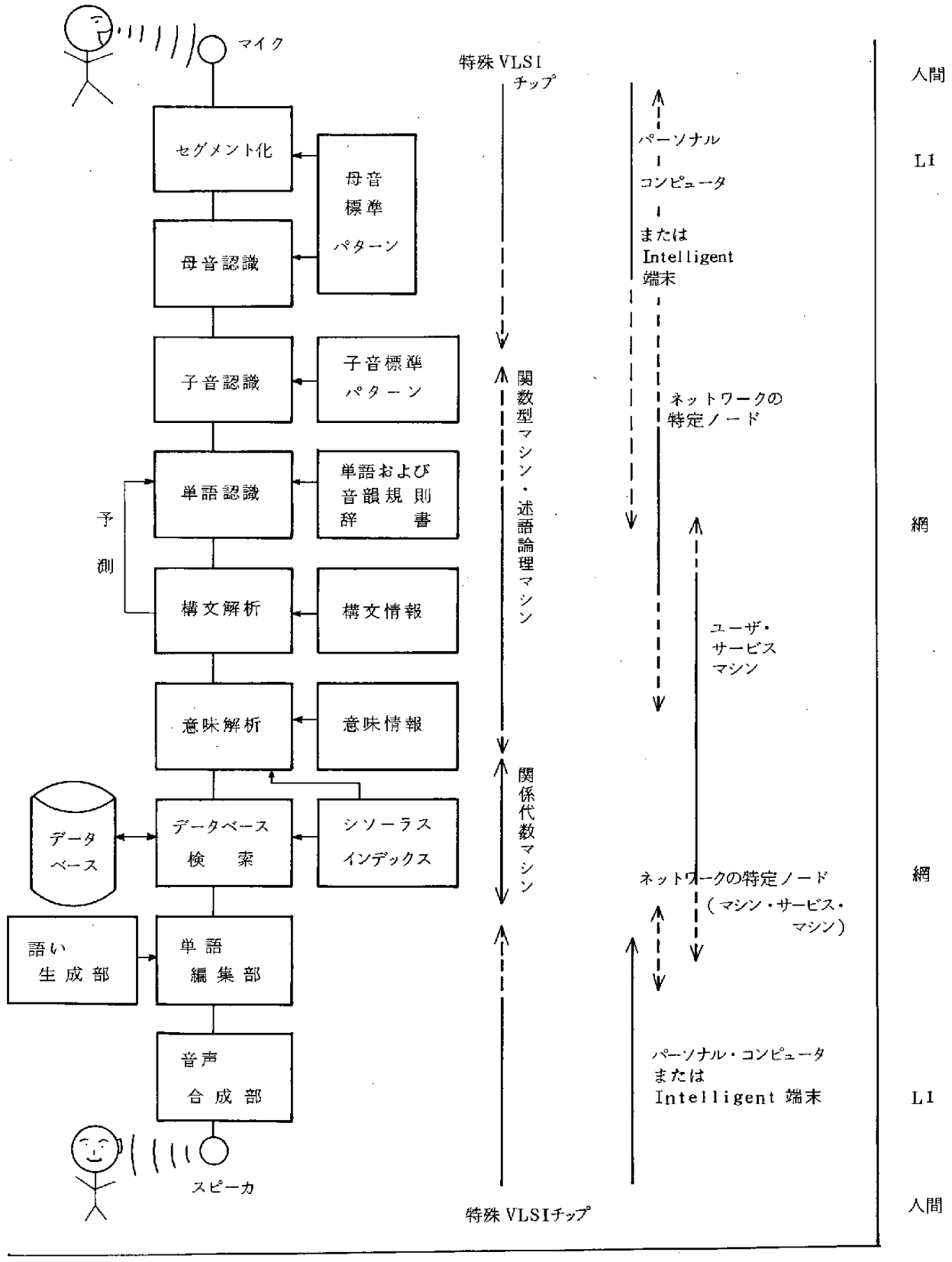


図５－１０　機能分散の例（音声による情報検索システム）

③　分散データベースの利用

意思決定に用いられるデータは一般に非常に広い範囲／分野にまたがる。従って各所に置かれた様々な，そして膨大なデータを調べることが必要となり，分散データベース機能は不可欠である。又，その利用状態は非定型であり予測し難い。

従って，機能分散アーキテクチャ上のパーソナル・マシンとしては，個人専用の知識ベースを備え，高級なディスプレイ機能の付いたものが必要であろう。上位レベル・マシンとしては，その分散データベース，知識ベース機能が使用されるとともに，述語論理マシンは諸推論に使われ，関係代数マシンがデータベースに対する諸操作に使われよう。

(2)　モデル作成

対象となる組織の構造を把握しその特性を理解する為にモデルを作成する。それには次のような作業を組み合わせて行なう。

①　システム分析

対話的に実行し，その結果を知識ベースに貯える。

②　モデル作成

モデルのデータベースを用い，対話的に編集してモデルを作成する。

③　モデルの検証／確認

一般常識を用いてモデルに不都合な所の無いことをチェックし，シミュレーションによりその妥当性を確認する。

従って，上位レベルのシステムでは，モデルデータベースや知識ベースが必要で，複雑なモデルを迅速にシミュレートする為のマシンを利用する。これにはＶＬＳＩに向いた並列アルゴリズムが有効である。更に，常識の利用等に述語論理マシンが使われ，データベース処理には関係代数マシンが使われよう。

(3)　シミュレーション実行

作成したモデルを用い，未知の環境下でのその動作を調べるとデータを取る為にシミュレーションを実行する。これには，

①　新環境を与えるデータの準備

各所のデータベースを利用し必要なデータを作成するのに分散データベース機能を利用する。

②　シミュレーションの実行

各所にモデルの一部であるサブモデル用のプログラムを置き，互いに協調させてシミュレーションを実行する。これには分散型ジョブ管理機能を利用する。

という手続きを踏むが，上位レベルマシン群内では次のような機能を使うことになる。

①　データベース・マシン

関係代数マシンがその内部構造である。

② フォーマット変換

ＶＬＳＩ，連想メモリを用いて実現

③ 高速シミュレーション

データフローの考え方で並列実行を制御し，各並列演算にはＶＬＳＩ用のアルゴリズムを利用する。

④ ネットワーク・ＯＳ

コントロール・マシン，マシン・サービス・マシンに分散ジョブ管理の機能をもたせる。パーソナル・マシンはその詳細を知らず，全体的な指示を行なうのみである。

(4) 結果の提示

シミュレーションの結果を，見易い形で利用者に提示することで，これには次のような機能を用いる。

① 結果の解釈と要約

数値結果を文章や図に変換したり，重要事項を抽出する。

② 諸表示デバイスの活用

高解像度ディスプレイ，カラー表示，プロッタ，音声等の利用。

パーソナル・マシン上では豊富な表示装置の機能が必要であり，個人用の知識ベースを利用してその人向けの解釈，表示を行なう。上位レベルのマシンでは，データの加工に諸演算機構を用いたり，文合成，要約作成，等の知識情報処理に関数型マシンを利用する。

(5) 打合わせ用通信

人と人との間に高級な通信手段を提供することで，例えば次のような機能が必要であろう。

① 電子会議

ＴＶをふるに活用したテレビ会議装置の他に，会議用の共用データベースを設定し，そこに会議データ，結果等を収納する。また会議進行の為の様々な支援を用いる。

② 電子郵便

アドレス・リストの完全サポート，カレンダ機能等が必要である。

パーソナル・マシンでは，大画面のディスプレイが必要な他，音声，画像，データを統合した端末，通信方式が必要である。従って，あらゆる情報をパケットの形で扱うような高速パケット通信網が便利であろう。上印レベルマシンでは，音声・データ・画像の総合編集・蓄積の機能を利用し，様々な種類のデータをデータベースとして扱えるようなデータベース管理システムを用いる。また，通訳／翻訳等の機能が必要になることもあろうが，それには関数型マシンが適していよう。更に，これらの通信では一般にかなり程度の高い機密保護機能が必要で，暗号化ＶＬＳＩチップを使うことになろう。

(6) 規則性の抽出

シミュレーション結果，観測データ等多くのデータより規則性を抽出する作業で，マシン内の帰納処理機能や学習機能を使うことになる。これには様々な知識処理が必要であると同時に，知識ベースを活用し，新追加知識によって知識全体の一貫性が保たれるかどうかの検証も行なわねばならない。

上位レベルマシンでは，述語論理マシンが使われ，大規模データの処理には関係代数マシンを内蔵したデータベースマシンが使われる。

5.4　大規模高信頼化システム設計開発技術

5.4.1　第5世代コンピュータ・システム開発における重要性

第5世代コンピュータ・システムの開発においては，その設計・開発技術，開発支援ツール，部品・材料の面においても，従来とは異なるものが必要である。

システムの設計・開発技術の点からいえば，まず，設計・開発対象となるソフトウェア，アーキテクチャ，ＶＬＳＩを含むハードウェア，ＶＬＳＩを使用する回路のそれぞれのレベルについて次のような特徴がある。

① 開発対象が未知の分野に属し，研究面の比重が大きい。

② 新しい技術，材料を積極的に利用することが望まれている。

③ ソフトウェア，ハードウェアともに，規模が大きい。従って，設計・開発支援用コンピュータ・システムを効果的に利用したシスティマティックな設計，評価の手法を必要とする。

④ 高い信頼性が必要である。

ここでは，これらに対応するシステム化のための技術についての研究テーマについて述べる。まず，新しいアーキテクチャを開発するときの考え方や方針について，5.4.2で述べる。

次に，5.4.3で，アーキテクチャやハードウェアを，その上で実行される演算処理のアルゴリズムやプログラムと，どう絡めて開発するか，また，それの設計・実装技術として何が必要かを述べる。5.4.4では，ＬＳＩや，ＶＬＳＩを用いて，回路を設計・実装する場合の研究すべき点について，5.4.5では，大規模なソフトウェアを作成する場合に重要な技術について述べる。最後に，5.4.6で，高い信頼性を持つハードウェアの実現のための技術と研究すべき点を述べる。

以上の各項で，述べる研究すべきテーマは，多くは，第5世代コンピュータ・システムの開発に向けて，必要となる不可欠の技術であり，特に，その中には，多くの基礎研究を含むものも多い。

これらの，目立たない基礎的な技術があってはじめて，優れたコンピュータが実現され得ることを，再確認することが必要である。

5.4.2　新アーキテクチャ開発法

(1)　問題の所在

ここで"新アーキテクチャ開発法"というのは，新しいアーキテクチャ概念，いいかえれば新しい計算モデルにもとづく計算機の，方式設計ならびに論理設計のための方法論のことである。

従来の通常の計算機においては，ノイマン・モデルが唯一の存在であり，それを暗黙の前提として方式設計や論理設計の枠組が形成されていた。したがって，このノイマン・モデルからの離脱を試みようとする新アーキテクチャの場合には，これまで設計者の判断のよりどころと

なってきた通念や規則・規準，すなわちパラダイムがそのまま通用するという保証がなくなってしまっていることになる。

このことは，既成の設計開発手法をそのまま新アーキテクチャに適用することは危険であり，開発を順調に展開していくためには，適用の可否を事前に十分吟味してかかる必要のあることを示唆している。

しかも，適用が不可となった場合には，短期間のうちに代替の定型的な手法を確立することは期待できず，ある程度の試行錯誤を見込んでおく必要がある。このときには，試行のための探索戦略を工夫して，錯誤を効率的に切抜けることが大切であるのはいうまでもない。

計算機の設計開発技術は，現在すでにＶＬＳＩ技術による革命的なインパクトの渦中にある。新アーキテクチャはこれにさらに輪をかけることになるかも知れない。しかし，新アーキテクチャはそれ自身がＶＬＳＩインパクトへの合理的な適応という意味をもつことは，データフロー計算機の主張の中に明白にされているとおりである。それからいえば，新アーキテクチャ開発の場こそ将来の計算機技術のパラダイム形成の"るつぼ"となるべきであり，そのためには多少の廻り道を恐れず，明確な指導原理の下である程度の実験的な試みを行なう余地を用意することも必要である。（ここでいう指導原理は，開発方向のありうべき選択肢において，判断の基準を与えるような命題のことである。）

(2) 開発の具体的展開

第１段階：新アーキテクチャ要求仕様の明確化

当該の計算機を開発する目的と開発上の条件をもとに，新アーキテクチャに対する要求仕様を明確化する。ここでは，いわゆるシステム要求仕様定義技法が利用できる。要求仕様に対する認定責任の所在と確定の時期を明確にしておくことが大切である。

第２段階：概念設計と影響範囲の検討

（第１段階）
- ① アーキテクチャのラフ・スケッチ作成
- ② 既成の方式的枠組との対比
- ③ 相違点の整理
- ④ 問題点の検討／予備的方式設計
- ⑤ 方式・論理設計パラダイムへの影響範囲の検討

第３段階：方式・論理設計手法の検討・再構成

① 分析・予測技法の見直し

② 評価技法の見直し

③ ＣＡＤ技法の見直し

④ 試行錯誤戦略策定

第４段階：詳細設計の実行

（新アーキテクチャによる特異点はなさそう。内容略。）

第5段階：評価

不満足な場合には，その原因を追及し，その所在に応じて適当な段階に戻ってやり直す。

(3) 主な問題点とその対策

1) 未知部分が多い。

対策① できる限り理論による展開をはかる。

このためには新アーキテクチャの理論的基盤がしっかりしていなければならないが，提案されているものは，この条件を満たしている。

② シミュレーション／実験

ＵＨＭを利用すれば仮想的な新アーキテクチャの実現は容易である。ＵＨＭの早期開発が条件になる。

2) 開発環境にハンディ・キャップがある。

その１ 開発要員の知識・経験不足

対策① 理論教育

設計内容の形式的体系化

② 既成の理論的な枠組を使用

代数・述語論理などの枠組を借用し，場合によっては馴染みやすい形に変換したものを使う。

その２ 既成リソース（実験／検査用プログラムとデータの削減化）

対策① 新プログラミング言語の利点（プログラム作成容易）を最大限に活用

言語仕様の早期確立

② 海外リソースの活用

述語論理マシンの場合などでは，すでにヨーロッパなどに大きなプログラムが存在している。活用のためには国際的な協力体制のあり方が問題

3) 開発課題の過重

その１ 設計開発ツールの再構築

対策① ＶＬＳＩ化のための再構築に便乗

ツールへの要求仕様を早期にまとめてよく調整をとることが重要

② 本質的に必要な変更とそうでないものの峻別新アーキテクチャの本質をよく理解させる。

その２ 互換性機能

対策① ＣＰＵは共存方式でいく。

基本方式をできるだけ早く決定することが肝要

② その他の部分はケース・バイ・ケース方式で，互換性への要求仕様を早期に確立する必要あり。

その３　一般要員ならびにユーザのための教育体制

対策①　新言語の特質を活用

ＰＲＯＬＯＧの場合には，従来の言語より習得がはるかに容易であることが知られている。

5.4.3　大規模ハードウェア設計開発技術

(1)　コンピュータ設計の流れ

コンピュータの設計手順は一般に図５－１１に示す通りで，まず製品の仕様が作られるとそれを満足するコンピュータの方式設計（アーキテクチャの設計）が行なわれる。次にこれにもとづいて具体的な論理回路を設計する論理設計，マイクロ・プログラムの設計，使用する回路素子や実装構造の設計，専用ＬＳＩを設計する場合にはＬＳＩの設計などが行なわれる。その後，実装設計を経て製造手配用の各種データ，製造・検査用データなどが作成される。

この過程において大量のデータの処理やくり返し作業的なところには現在コンピュータが導

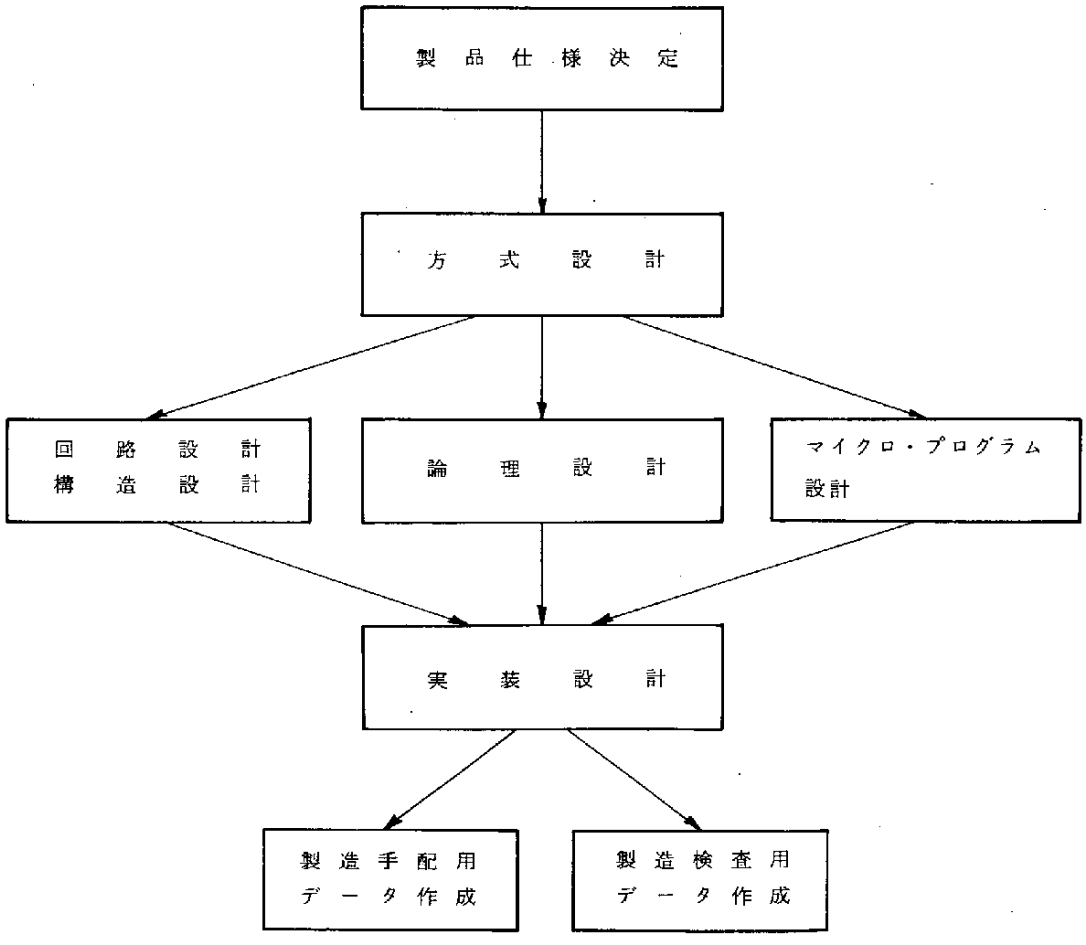


図５－11 コンピュータ設計・開発の流れ

入されており設計者の専門的知識や創意工夫を必要とする作業は人手で行なっている。即ち製品仕様を決めたり，方式設計などは人手で行ない，論理設計などは一部コンピュータによる自動化をとり入れており，それ以後の作業についてはかなり自動化が進んでいる。

(2) アーキテクチャ設計開発

現在，アーキテクチャの設計開発のうちオリジナルなアイディアを考える部分については完全に研究者，技術者の創作活動で，機械化や自動化の入る余地はなく将来も少くとも１０年後位までは同じであろう。したがって設計・開発法として重要なのはアイディアを実現するために必要な各種の評価作業をいかに効率的にやるか，どんなツールを用意したらよいかという点にあろう。

このような観点から，問題点，展望，研究課題などを列挙すると以下のようである。

(a) アーキテクチャの評価は時間がかかる。

新しいアーキテクチャを開発した場合，その効果を総合的に評価するにはハードウェア，ＯＳ，アプリケーション・プログラムまで完成してシステムとして運用してみる必要がある。この時点で判明した欠点があってもそれを修正すると，ハードウェア，ＯＳ，アプリケーション・プログラムのすべてに影響を及ぼすため，商用機ではフィードバックがむずかしいことが多い。

従ってある「アーキテクチャ」を定義したら実際のハードウェア，ＯＳ，アプリケーション・プログラムを作らなくても各種の評価ができるシミュレータ（各種アーキテクチャに対応できる万能型でかつ高速のもの）が望まれる。

(b) アーキテクチャ設計のアプローチ

従来のコンピュータではまず汎用のアーキテクチャを考え，対象の問題を解くためにソフトウェアを種々工夫した。

しかし今後コンピュータの専用化が進むとまず問題があってそれをうまく解くためのアーキテクチャを考えねばならず当然プログラム言語などもいっしょに考えることになる（図５－１２参照……これは並列処理アーキテクチャへのアプローチの相違を示している。今後はこの中のＦＩＧ．３のアプローチをとることになろう）。

このため今後はアーキテクチャ設計技術者に対し，対象の問題，ハードウェア，ソフトウェアなどについても深い知識が要求されてくるのでそういう人材を育成する環境が必要である。

(c) 最適トレードオフ

アーキテクチャの設計は多くの場合目的の性能を出すために最適（通常は最小のコスト）のハードウェア／ファームウェア／ソフトウェアの機能分担を決めることである。

しかし，あるアーキテクチャがほんとうに最適なのかを判断することは難しく，しかもそ

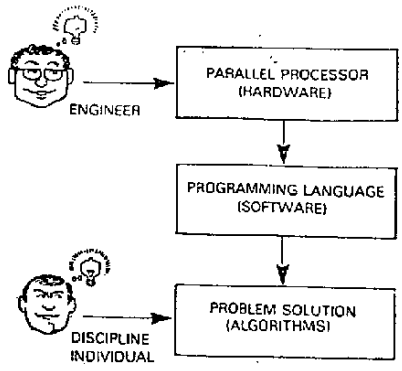


FIG. 1. THE TRADITIONAL APPROACH TO PARALLEL PROCESSING

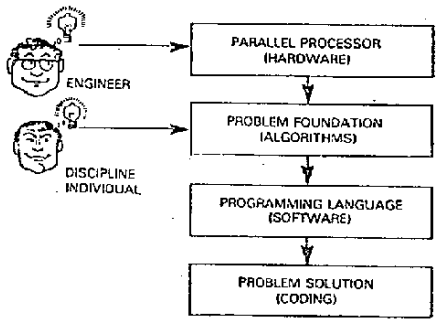


FIG. 2. MODIFIED TRADITIONAL APPROACH TO PARALLEL PROCESSING.

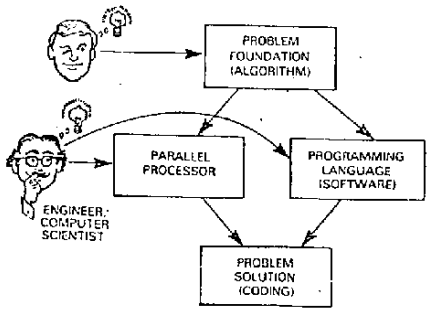


FIG. 3. A NEW APPROACH TO PARALLEL PROCESSING

図５－１２　並列処理アーキテクチャへのアプローチ

の基準は時代の推移と共に変わってくる（たとえば昔はハードウェアが安く作れるアーキテクチャが良かったが将来はソフトウェアが安く作れるアーキテクチャが良いアーキテクチャとなろう）。このため容易に使用できる評価ツールが必要である。

(d) アーキテクチャ記述言語

前述のような評価ツールを利用可能にするためにもアーキテクチャ記述言語の確立が望まれる。過去に，ＡＰＬを使用した例があり，また研究レベルでは種々の試みがあるが，現在，商用機の設計においては自然言語の文章（仕様書）によるのが主で，あいまいさなどの問題がある。

(e) 性能評価

あるアーキテクチャで完成したハードウェアの性能を測定し，評価のためのデータを収集するにはリフトウェア・モニタとハードウェア・モニタが使用される。ソフトウェア・モニタはソフトウェア／システム的な意味を持ったマクロなデータがとれるがデータを採るためにシステムを乱すこと（ソフトウェア・モニタのオーバヘッドが加わる），高速に変化する事象のデータは採れないなどのデメリットがある。

ハードウェア・モニタは詳細なデータを多量にとることができ，システムの動作に影響を与えずに測定することが可能で，かつ高速に変化する事象の測定も可能であるが，一方ソフトウェア／システム的意味づけは別途やらねばならず，ソフトウェア・モニタとは長短相補う関係にある。したがって今後は両者が一体となったモニタが必要で，そのためのハードウェア機能があらかじめ組込まれていなければならない。

個別部門でコンピュータが作られていた時代にはハードウェア・モニタ用信号は後からでも取出せたが，ＬＳＩ／ＶＬＳＩ化されると最初の設計の段階から必要信号を決めて少くともチップの外部へ出しておかないと後では取出せなくなる。したがってＬＳＩ／ＶＬＳＩのチップ設計時にこの配慮が必要である。

(f) 適応型アーキテクチャ

ある程度自分自身で使用環境に対し適応化できるアーキテクチャのコンピュータはアーキテクチャ評価の不完全さを補うものとしても有効と思われる。但しソフトウェアから見えるアーキテクチャが変わると影響が大きいので見えない範囲でやることになろう。

(3) 論理設計

コンピュータの設計の流れを論理設計中心に少しくわしく示したのが図５－１３である。これらの作業は自動化されている部分がかなりあるが，論理設計に関してはまだ人手にたよっているのが大半でその効率向上，論理不良の早期摘出などが重要な課題となっている。ＬＳＩ／ＶＬＳＩの導入によりその重要性は一層高くなる。たとえばＬＳＩ／ＶＬＳＩを用いた場合には論理設計の誤りはＬＳＩ／ＶＬＳＩの作り直しにつながり，開発期間に著しい影響を与える

からのトップダウンの設計が混在していたのをシステム・レベル→機能レベル→ゲート・レベルと段々下へ落していく方式でスタンフォード大のＳＣＡＬＤなどの例がある。ＩＢＭでもＡＬＥＸと称する階層設計のシステムを発表しておりＪ．Ｊの回路設計に適用しているという。

(b)　マン・マシン・インタフェース

ＣＡＤを使用するマン・マシン・インタフェースはグラフィック・ディスプレイを用いた会話型のものが指向されており，前述のＡＬＥＸやＳＣＡＬＤでも実施されている。

グラフィック・ディスプレイやタブレットの高性能（高精度）のものが必要である。

(c)　論理設計の入力

論理設計の入力もネックになっており，カードによるバッチ入力とキーボードからの入力の組合せ，回路図形式（図形）での入力（ＡＬＥＸ），などの工夫が行なわれている。

機能レベルの記述言語で入力して論理合成とVerificationの自動化を行う研究もなされている。

マン・マシン・インタフェースの問題と合わせて，効率的な方式を研究，開発する必要がある。

(d)　論理シミュレーション

論理シミュレーションはゲート・レベルでやるものと機能レベルでやるものがあり，両方を組合せて行っている。たとえば，あるユニットのゲート・レベルのシミュレーションを完全にやるためにそのユニットに隣接する部分は機能レベルでシミュレーションする，あるいは機能レベルとゲート・レベルの両方でシミュレーションして結果を比較するなどの方法がある。

論理シミュレーション・システムとしては次のようなことが要求される。

- 論理設計の入力が容易なこと。
- テスト・データが容易に準備できること。
- 短時間にシミュレーションできること。
- シミュレーション結果の確認が容易なこと。
- 設計の段階に応じたシミュレーションができること。
- 実装設計システムとデータベースが共用できること。

(e)　将来の方向

今後の方向としてはあらかじめ登録されているモジュールを使用した組合せと論理合成の自動化により機能レベル中心のＣＡＤになっていくのではないかと思われる。標準的なＬＳＩ／ＶＬＳＩはその中ではブラック・ボックス的に使用されることになろう。

(f)　そ　の　他

であろう。また，今後の高性能コンピュータでは論理設計の量（ゲート数）は膨大なものになり（現在の超大型コンピュータでは数十万ゲートであるが将来は１００万ゲートに達するであろう）論理設計の作業効率が改善されないと開発が困難になることも予想される。

以下，いくつかのテーマについて問題点，展望，研究課題などを述べる。

(a) 論理設計の自動化

設計作業の効率化のために自動化が必要であるが現在まだ実用化されていない。

過去にＩＢＭがやった例（ＡＬＥＲＴ）があるがゲート数が人手による設計の３倍にもなり，改良しても1.8倍にしかならないと予想されてその後中断していた。しかし最近では再び要求が高まっておりＩＢＭでも研究を再開して７００ゲートのＬＳＩ設計に使用して，人手と同程度のものを設計できたといわれている。

したがって設計自動化の研究は今後の重要なテーマである。

現時点では自動化が困難なので効率向上のために階層設計，トップダウン設計などの方法が試みられている。これは，従来，ゲート・レベルからのボトムアップとシステム・レベル

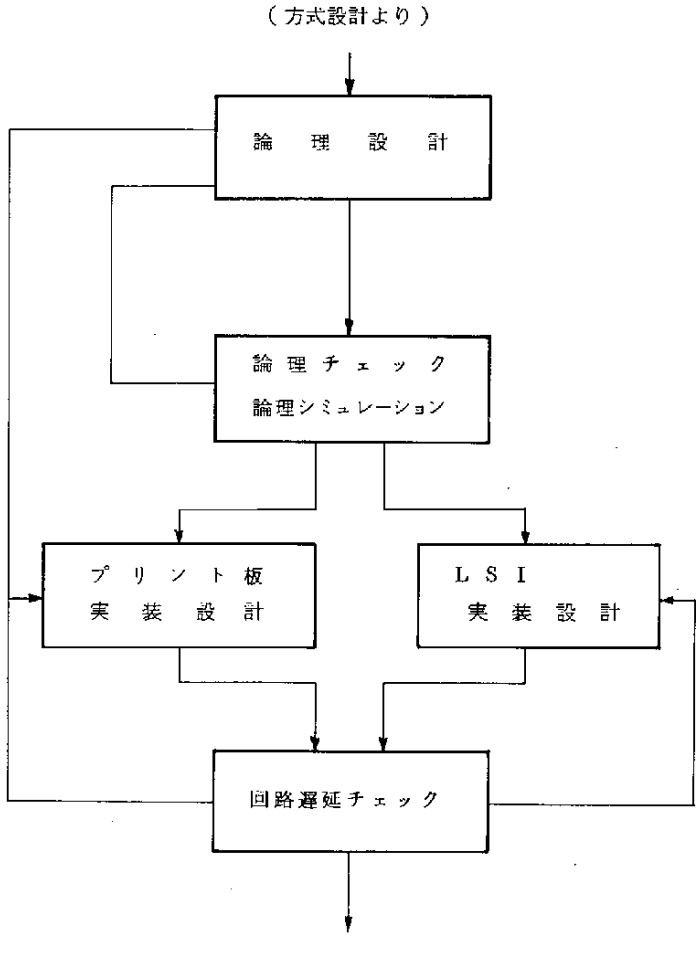


図５－１３　論理設計に関係した作業

以上の他にＣＡＤに関係して次のようなことがある。

・診　断

現在はゲート・レベルでやっているが今後ゲート数が増えると診断にかかる時間は膨大になる。したがってチップ自身，あるいはＦＲＵ（Field Replaceable Unit）単位に診断機能をもつ必要があろう。ＩＢＭではチップ内の５～２０％の回路を診断のため（スキャン・パス）に用いているがＶＨＳＩではそれ以上を診断機能のために用いるという。

・ファームウェア

ファームウェアの記述はコンパイラ・レベルのものが試みられているもののほとんどがアセンブラ・レベルと思われる。

コンパイラは最適化が難しいのが難点だが将来多量のファームウェアが作られるようになるとすればコンパイラ・レベルで書ける必要がありまたソフトウェア開発手法の多くが流用できねばならないだろう。しかしながら性能やメモリ容量の制約はソフトウェアの場合より数段きびしいのでかなりの困難が予想される。

マイクロ・プログラムの作り易さ，コンパイラ化の可能性などの点からは水平型より垂直型のマイクロ・プログラムの方が適しているのが性能的には劣るので一台のマシンで水平型と垂直型を併用することがよいかもしれない。水平型だけで大きなプログラムを作るのは難しいと思われる。

・ＬＳＩ／ＶＬＳＩ化

～超高性能を狙う場合，現在予想される集積度では多数の専用ＬＳＩ／ＶＬＳＩを用いることになろう。論理設計からＬＳＩ／ＶＬＳＩの最適切出しを自動的に行うことも研究課題である。この場合，ピン数の観点で切出すかrepeatabilityの観点で切出すかなどの使い分けが可能であることも要求されよう。

集積度がさらに大きくなってまとまった機能単位が入るようになればこの問題は軽減される。

〔参考文献〕

(1) "Issues in Parallel computing：A non-euclidean examination" P.B. Schneck, Proc. of 1979 international conference on Parallel Processing.

(2) "ＡＬＥＸ：A conversational, hienarchical logic design system" K.A. Duke, Proc. of 1980 Design Automation Conference

(3) "Verifging deep logic hierarchies with ＡＬＥＸ" G.M. Koppelman, Proc. of 1980 Design Automation Confercnce

(4) "電子装置のＣＡＤ(1)"電子装置設計技術研究委員会，情報処理，Vol.20 ， No.11 （Nov.1979）

(5) "超大形電子計算機ＨＩＴＡＣＭ－２００Ｈの論理シミュレーション" 鬼塚宣彦他，情報処理学会論文誌， Vol•21 ， No.5 （Sep.1980）

(6) "ＭＩＸＳの概念"山田昭彦他，昭和５５年度情報処理学会，第２１回全国大会

(7) "大形コンピュータの自動設計"加納弘他，日立評論， Vol.62 ， No.7（１９８０－７）

(8) "ハードウェアの機能設計段階における検証" 丸山文宏，情報処理，Vol.21 ， No.5 （May 1980）

5.4.4 大規模回路実装技術

(1) 集積技術

半導体の集積技術は，今まで通り進歩することが予想され，１９９０年代には，１μテクノロージが可能となり，

- ＭＯＳメモリ　　４Ｍビット／チップ
- バイポーラメモリ　　１Ｍビット／チップ
- ＭＯＳゲート　　10^6 ゲート／チップ
- バイポーラゲート　　10^5 ゲート／チップ

程度の集積度の素子が実現できると予想される。

現在，ＭＯＳメモリなどは，年率２倍の集積度増加率を示しているが，それらはシリコン・チップの大形化，微細寸法化およびデバイスと回路の改良により達成されている。最後の要因のデバイスと回路の改良は８０年代で限界となり，集積度増加率は２年で２倍程度に低下すると予想されるが，2,000年頃までは増加が続くと予想されている。

従って，実装技術は高密度，高集積化の方向に進むが，一方，これを阻害する要因としては，半導体の高集積化に関連するものとし，１μテクノロジー以下でのエレクトロ・マイグレーション，ホット・エレクトロン，直流抵抗値の増大，半導体内部接続の複雑化，発熱などが予想される。さらに，重要な阻害要因として，半導体チップに必要な入出力ピン数の増大とその接続の問題がある。

エレクトロ・マイグレーションやホット・エレクトロンの問題は，半導体技術の改良により克服する必要があり，微細加工に伴なう直流抵抗値の増大，半導体内部接続の複雑化の問題はチップ形状の大型化，ＣＡＤ技術の改良などにより解決して行く必要がある。

残りの発熱の問題と入出力ピンの問題およびそれらピン間接続の問題は，高集積度の半導体を生かす器としての実装技術上の課題である。

(2) 実装技術

大規模集積化する半導体に対し，これを生かすための実装技術の課題は，冷却，半導体チップの入出力ピンの集積度向上，その接続の配線密度向上およびコネクタ・ピンの集積度向上などである。

発熱については，１９８０年には１W／cm²程度であったものが，半導体のPJの改善を考慮しても１９９０年代には１０W／cm²程度のものを冷却する要求が生ずるであろう。

半導体チップの入出力ピンについては，図５－１４に示すように10^4～10^5ゲートの論理を生かすためには接続ピンとしては1,000ピン程度必要となると予想されている。[文献(1)]

仮りに1,０００ピン程度用意できたとしても実装密度を向上させるためには，ピン間を接続する配線密度向上が必要である。

ピン密度と配線密度の向上については，図５－１５に示すような年次推移の実績例がある。[文献2)]

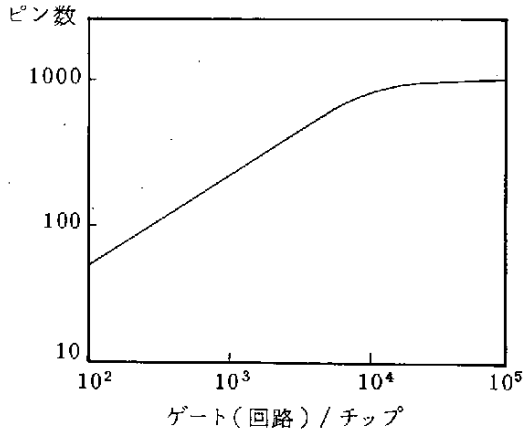


図５－１４　集積回路数と必要なピン数

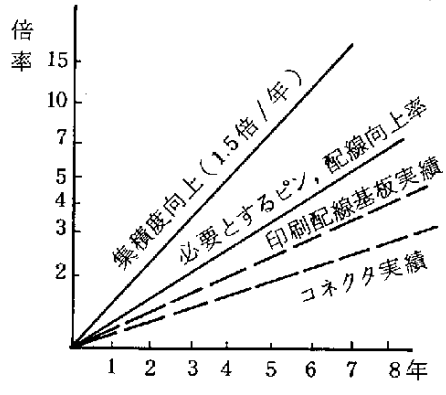


図５－１５　実装密度向上実績

ここで必要とするピンと配線の向上率は，ゲート数をGとしたとき，ピン数$P = K \cdot G^{0.6}$としている。

この例は，半導体の集積度の向上速度に比し，印刷配線基板およびコネクタ・ピンの実装密度向上速度はかなり遅く，今後の技術的解決を必要としている重要な問題であることが解る。

向上速度の最も遅いものは，ピン密度と予想され，これが実装面積および体積を決定して行くと考えられる。ゲート数に対して必要な接続ピン数の関係をさらに堀り下げると，一般に次のことが言われている。

① １つのプロセッサの論理を小さく分割すると，そのピン数とゲート数の間には

$P = K \cdot G^n$　　P：ピン数　　G：ゲート数　　K：定数　　$n < 1$

という関係がある。

② 1つのプロセッサの論理を比較的大きな単位（1／10以上）に分割するとピン数は飽和し，ゲート数／ピン比率は増大する。

③ プロセッサ論理を1つにまとめてしまったとき，最もゲート／ピン比率は増大する。

①の領域を第1領域，②の領域を第2領域ということがある。そこで③の場合を第3領域と呼んでおく。

ピン密度を考えた場合には，このような特性を良く考慮しておく必要がある。

一方，配線密度については，印刷配線基板よりもさらに配線密度を向上させ得るものとして，セラミック基板がある。このため，セラミック基板の上に半導体を実装する方法も，実装密度を向上させるために有力な方法と考えられている。

従って，半導体の実装方法としては，大別して，

① パッケージに収めて，それを印刷配線基板上に実装する方法（従来から用いられてきた方法）

② 半導体チップを，直接セラミック基板上に実装する方法（比較的新しい方法）

との2通りが考えられる。

半導体チップをセラミック基板上に直接実装する場合にも，問題となるのは半導体と入出力ピンとの接続方法とピンの数である。

ピンとの接続方法としては，現在，

(a) CCB（Controlled Collapse Bonding）方式

(b) TAB（Tape Automated Bonding）方式

(c) WB（Wire Bonding）方式

(d) Beam Lead方式

などの方法がある。

セラミック基板上実装したのち，そのセラミック基板にどれだけの数のピンが用意できるかが，重要な問題となる。

一方，半導体チップをパッケージに収める場合には，WB方式でピンと接続される。これを印刷配線基板上に実装したのち，基板上の配線密度と，その基板にどれだけのピンが用意できるかが，重要な問題となる。

大型のプロセッサの場合，チップを直接実装した段階では，ゲート／ピン比率は第1領域となることが予想され，10^4 ～ 10^5 ゲート程度実装したところで，1,000ピン程度必要となろうが，その次の階層へ接続する段階では第2領域までつめてしまう方法が考えられる。

チップを直接，基板に実装した段階で，第2領域へもって行く方法も考えられるが，この場合にはゲート／ピン比率は改善されるが，ピンの絶対数は増大することと，単位面積あたりの発熱量が増大することが問題となろう。

規模の小さいプロセッサでは，チップを直接に基板に実装した段階で，第３領域にもって行くと，ゲート／ピン比率はさらに改善される。このとき，ピンの絶対数を第２領域の場合よりも低減できる可能性がある。

当面，10^4～10^5ゲート程度実装したところで，1,000ピン程度を用意する方法が課題となるが，このことは，１００ミル・ピッチの全面格子配列のピン形状でも１０cm×１０cmの大きさが必要であるので，技術的な問題が多い。

次に問題となる冷却方法としては，現在は，ファンやブロアによる強制空冷が行われており，一部には，水やフレオンを通した熱交換器が用いられている。発熱密度は，通常１w／cm^2以下であるが，今後の実装密度の向上を考慮し，かつ，半導体の回路技術の進歩によるｐＪの減少分を遅延時間の改善に向けたとすれば，発熱密度は１０w／cm^2を越えることが予想され，フレオン浸漬などの冷却技術が必要となろう。

冷却能力についての見通しを図５－１６に示す。[文献1]

新しい冷却方法の限界は，フレオン浸漬沸騰強制対流などの方法により２０w／cm^2程度と予想されているが，このレベルを実現するための技術的課題も多い。

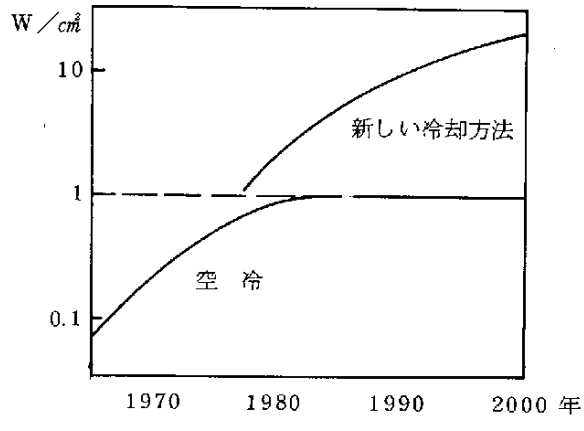


図５－１６　冷却能力の見通し

(3)　実装技術を支援する論理設計技術

実装技術の問題で当面の課題となるゲート／ピン比率については，論理設計技術面からの支援が必要となろう。具体的には，ゲート／ピン比率の大きいＶＬＳＩを組合せてプロセッサを構成するための論理分割技術が必要となる。

このためには，１つのプロセッサ論理を分割してＶＬＳＩに収めるという方法ではなく，ＶＬＳＩ分割を前提としてプロセッサ論理を組立てる必要があり，このような手法は，ＬＳＩの集積度が向上するに従い，新たに開発される必要がある。

例えば，演算機構について，加算器，乗算器，除算器，１０進演算器などをそれぞれ１つのＶＬＳＩで構成し，それを組合せて演算ユニットを構成するなどの方法が考えられる。この場合には，ゲート／ピン比率の向上は期待できるが，ＶＬＳＩの繰返し性はない。

ＶＬＳＩの繰返し性をもたせながら，ＶＬＳＩの組合せによるプロセッサの構成方法は，ＶＬＳＩの利用面での重要な研究課題である。このような問題を解決しようとすると，新しいアーキテクチャの開発が必要となることが予想される。

(4) 研究課題

① 低誘電率で電気信号を高速伝播できる印刷基板の開発

② 熱膨張率がセラミックまたは半導体チップに合った印刷配線基板の開発

これは半導体チップキャリアを直接に印刷配線基板に実装する，あるいは半導体チップを直接にプリント基板に実装するなどの方法を可能とするために必要である。

③ 印刷配線基板の配線密度を向上させるために，配線の線抵抗を増大させずに微細配線する開発。

④ 多ピン・コネクタの開発

ピン配列を多列にし，ピン密度を向上させるとともに，挿入，抜去の際の力が少なくてすむコネクタの開発が必要である。

⑤ 次のような性質をもつセラミック基板材料の開発

- 安全な絶縁体である。
- 半導体の熱膨張率と同等の熱膨張率をもつ。
- 熱伝導率が大きい。
- 機械的に強く，ピンが取付け易い。

⑥ セラミック基板の配線技術の開発

薄膜配線，厚膜配線などを行う最適なプロセス技術の開発

⑦ 半導体チップから最大数の信号ピンを取出す方式の開発

⑧ その他の技術の開発

チップ交換技術，生産自動化技術，不良率低減のための技術の開発が必要である。

⑨ ＶＬＳＩを組合せて構成することを前提とした大形プロセッサの論理構成方法の開発

ゲート／ピン比率の大きい論理ブロックに分割して大形プロセッサを構成するような方法の開発が重要である。

〔参考文献〕

(1) Landman, et al, "On A Pin Versus Block Relationship for Partitions of Logic Graphs", IEEE Trans. on Computers, vol. C-20, no. 12, pp. 1469-1479, Dec. 1971

(2) 矢田豊，「保守単位からみた実装技術の限界」昭和５０年，電気４学会連合大会講演論文

(3) 千葉常世，古谷勝美，「論理速度から見た実装技術の限界」昭和５０年，電気４学会連合大会講演論文

(4) Keyes, "The Evolution of Digital Electronics Towards VLSI," IEEE Jounal of Solid-State Circuits, vol. SC-14, No. 2, April 1972

5.4.5 大規模ソフトウェア設計開発技術

(1) 大規模ソフトウェアの開発の重要性

今後はオフィス・オートメーションの時代と言われたり，マイマシン・インタフェース向上のための人工知能の時代であるといわれたりしている。確かに，ハードウェアの進歩により，その可能性は十分にあっても，ソフトウェアの生産にボトルネックがあれば実現はおぼつかない。是非とも，ソフトウェアの生産性を飛躍的に向上させる必要がある。そして従来以上に，大規模ソフトウェア，特に大規模アプリケーション・ソフトウェアの効率の良い開発が重要である。

(2) 大規模ソフトウェアの開発の問題点

大規模ソフトウェア開発の主要な問題点を示すと次のようになる。

① ソフトウェアの開発はマンパワー集約型であり，人件費は上昇の一途にある。

② マンパワー集約型であるために，共同作業をまとめる管理体制が強調され過ぎる傾向がある。

③ システムとしての一貫性を保つのが大変に困難であり，製品の信頼性に問題が出ることがある。

④ 十分な製品テストを保証するテスト法が確立していない。

⑤ ソフトウェアの構造とか，動作原理を説明するドキュメントの量が膨大になってしまう。

⑥ 「必要とする機能」と「ソフトウェアで実現される機能」とが異なってしまうことがある。

⑦ 機能の追加・修正（メンテナンス）に大変な作業が必要になる。

(3) 大規模ソフトウェアの問題解決の方向

従来のソフトウェアの生産法は，何と言っても人間集約型であり，コンピュータのハードウェアの製造技術の進歩で，コンピュータが低価格になってもソフトウェアの低価格にはつながらない。反対に人件費は年々，上がる一方であるので，ソフトウェアの価格も高くなってしまう。

そこで，コンピュータのハードウェア（プロセッサ，メモリ等）の能力を大量に使い，ソフトウェアの生産過程で人間は細かな作業をしなくてもよいように，省力化するべきである。例えば，従来の高級言語に比べ，より高水準の言語を用いて，ソフトウェアの生産性を上げて行く必要がある。

ソフトウェア工学では，次の3項目が重要とされている。

① 機能の高レベル化

アセンブラ言語よりは高級プログラミング言語を使用した方が能率よくプログラミング出来るように，マイマシン・インタフェースを高レベル化し，人間は創造的な作業のみをする

ようにすること。

② 仮想化

仮想記憶システムでは，プログラムがメモリの実装容量を意識しなくてもよいように，プログラムはハードウェアの性質にとらわれることなくつ論理面にのみに集中できるようにする。ワンレベル・ストーレッジの技術はディスク装置等の二次記憶装置を仮想化するものである。

③ 標準化

高レベル化したインタフェースと仮想化した機能を標準化することで，ソフトウェアの生産上の向上と互換性が達成できる。

(4) ソフトウェア工学の動向

現在のソフトウェア工学では，ソフトウェアのライフサイクル全体に対して研究が行われている。

次に主な研究課題を示すと次のようになる。

① 要求仕様技術（ＩＳＤＯＳ，ＳＡＤＴ等）
② 設計仕様技術（構造化設計，モジュラー設計，設計言語等）
③ 段階的詳細化のサポート・ツール（一貫性チェック等）
④ インプリメンテーション用言語（プログラミング言語）
⑤ デバック／テストのためのツール
⑥ インストレーションのためのツール
⑦ メンテナンスのためのツール
⑧ コンバージョンのためのツール
⑨ システムのモデリング技法
⑩ ダイヤグラム（図形）による仕様記述の技術
⑪ ソフトウェアの部品化のアプローチ
⑫ ソフトウェアの開発の管理技術
⑬ ソフトウェアの開発者のためのワークステーション
（パーソナルな作業端末の仕様）

ツールの研究・開発と利用を通じて，マンパワー集約の問題が順次解決されて行くと考えられる。

このような方向の中で，特に①の要求仕様技術，②の設計仕様技術，④のインプリメンテーション用言語（プログラミング言語）の記述の高レベル化，標準化をすることで，ドキュメント量の問題，製品の品質の問題，メンテナンスの問題が解決して行くと考えられる。

(5) ソフトウェア工学と知識情報処理

第5世代コンピュータで実現される知識情報処理システムが，ソフトウェア開発に大いに貢献すると考えられる項目について説明する。

① ソフトウェア設計知識ベース（自然言語処理と知識ベースの応用）

ソフトウェアの設計・作成・保守用の日本語のドキュメントをコンピュータに登録し，特定のソフトウェアに関する情報をコンピュータに向って問い合せると，所要のデータが端末に表示されるようなシステムがあれば，ソフトウェアの生産性向上に大いに貢献することは確実であり，また膨大なドキュメント量を減少させる有力な手段である。

このような機能は，知識情報処理の研究の目標でもあり，自然言語処理の研究，知識ベースの研究等が今後必要である。なお，当面はコンピュータ処理を考えた制約条件の強い日本語であっても，かなり役に立つと考える。

② 要求仕様技術

作成するソフトウェアが何をすべきかを，無矛盾であいまいさなく，完全に定義するための技術である。ソフトウェアの要求定義でよく使われる基本概念を表現する用語をキーワードとして設定した要求定義用語の研究が盛んである。また，図形表現の要求定義技術も研究されている。

日本では，日本語を用いかつ形式的な要求定義言語と人間のパターン認識能力を活用する図形表現を両用する必要があろう。

仕様における矛盾を自動的に検出する機能は，従来はそのアルゴリズムと処理時間の両面から不可能とされてきたが，知識情報処理により第5世代には可能となろう。

③ 高級プログラミング言語

ＣＯＢＯＬ，ＰＬ／Ｉ，ＦＯＲＴＲＡＮ，ＰＡＳＣＡＬ等の高級プログラミング言語は，長年使用されてきており，これらに対抗できるような言語は他にないように見えるが，しかし１９９０年代のプログラミング言語としては，次の機能等が不足している。

- 抽象データタイプのサポート
- 集合やリストなどのデータ構造のサポート

そこで述語論理型プログラミング，関数型プログラミング，抽象データタイプ等をサポートする新言語の確立が期待されている。

④ 人間の視覚を活用する図形表現の利用

ソフトウェアの設計に図形表現を用いる研究が盛んである（ＩＢＭ社のＴＥＬＬシステム，ＮＳチャート，グリーン・プリント，日立のＰＡＤ，日電のＳＰチャート，富士電のＳＤＤ等）。論理を与える時の視覚の重要さは医学的にも指摘されており，今後，グラフィック・ディスプレイ端末の普及とともに実用化されるであろう。

⑤　機械翻訳機能

ソフトウェアの製品を海外に輸出する時には，日本語のドキュメントは機械翻訳の機能で各国語に翻訳する。これも知識情報処理技術の応用技術である。

5.4.6　超高信頼化技術

１９９０年代の計算機を開発する際に求められる技術の一つに，高信頼性の確保が挙げられる。この面では，次の２種のアプローチが併行して進められるであろう。

- 故障の少い高信頼性を持つＶＬＳＩチップの開発
- システムまたはアーキテクチャ・レベルで，故障発生時にその影響を除去する対応策を内蔵させる手法の開発（耐故障性の高い計算機〔Fault Tolerant Computer〕の開発）

当然のことながら，この２者のアプローチは相互に独立なものではなく，お互いに作用し合うことにより，所定の超高信頼化が実現されるものと考えてよいであろう。従って，以下では，その中心的要素となると思われるFault Tolerancy（耐故障性）に重点を絞って考えてみる。

(1)　フォールト・トレラント・コンピュータの重要性

計算機の社会活動，経済活動での適用の場が広まるにつれて，そのシステムの故障に伴う影響の大きさが問題となってきている。たとえば，航空機制御，病院制御，プラント制御などのリアルタイム制御システムでは，計算機の誤動作，故障は，人命に係る重大事故を招きかねないし，人工衛星，ロケットなどに積載されている計算機の誤作動，故障は，そのミッションの達成を不可能とし，莫大な資金を無駄にしてしまう。また，電子交換システム，各種の予約システムなど公共のシステムでのトラブルは，社会問題にまで発展する要素を含んでいる。

このように考えてみると，計算機の適用分野の拡大とその普及に伴い，誤動作，故障のないフォールト・トレラント・コンピュータの適用分野が増大してきていると云える。さらに，通常の計算機システムでも，保守コストの増大につれて，故障の少いものが求められてきている。

(2)　フォールト・トレラント・コンピュータの課題

誤動作，故障のない，あるいは少い計算機を作る上での主要な課題には，次のものがあげられる。

- 誤動作，故障の確実なる検出
- 一時的誤動作に対する自動再試行等の回復手法の確立
- 障害発生ユニットの自動切離しと，システムの自動再構成技術の確立
- ソフトウェアの動作も含め，システム全体での自動回復技術の確立

以上の課題は，計算機の歴史の当初から現在に至るまで，常に求められてきたテーマであり，特定の用途に限られたシステム対応で，種々の解決手法が提案，または採用されてきているが，

一般的に普及しているものは少い。しかし，今後は，ＶＬＳＩの進歩に伴い，一般的に普及する技術として確立されるであろうことは，ほぼ確実と考えられる。

(3) 問題解決の手法

既に提案され，あるいは，実用化されてきたフォールト・トレランシを実現する技術の主なものには，以下のものがある。

(a) ロジック・レベルでの解決

現在では，かなり小規模のミニ・コンピュータ，マイクロプロセッサでも採用されだしているメモリの誤りの自動訂正（ＥＣＣ方式）であるとか，電子交換機等の特殊システム，汎用大型機などで採用されはじめている主要論理回路の２重化を図り，並列処理を行って出力または，途中演算結果の比較を行うことにより誤りを検出する方式など，種々の実用化が進んできている。最近ではマイクロ・プロセッサ・レベルで，この２重処理方式を採用する動きも見えはじめている。

(b) モジュール・レベルでの解決

既に１０年以上前から航空宇宙関係のオンボード・コンピュータを中心に試みられている手法にＴＭＲ（Tripple Modular Redundancy）がある。これは，計算機を構成する各種モジュール・レベルで，最低３個の出力を，多数決論理回路を通して比較することにより，障害の検出と，障害モジュールの切り分けまで行う手法である。さらに，予備モジュールを準備しておき，障害モジュールの切離後，予備モジュールを系に投入し，長期間に渡り同一レベルのフォールト・トレランシを実現するセルフ・リペア方式にまで発展している。最近では，ＶＬＳＩカスタム・チップおよび汎用マイクロプロセッサ・チップを利用して単一モジュール内で，障害の自動検出を行い，予備モジュールへの切換えを行うＳＣＣＭ（Self Checking Computer Module）の研究が進められている。

(c) サブシステム・レベルでの解決

モジュールよりは一段上位ランクのサブシステム，たとえば，ＣＰＵ，チャネル，メモリなどのレベルで冗長性を持たせる手法である。ここでは，冗長性のある密結合マルチプロセッサ・システムを採用し，障害サブシステムの切離しと予備サブシステムの投入，切離されたサブシステムの障害部診断および修復を，オンライン処理を中断することなく行う方式が求められる。古くは，３６０／６５と５０をサブシステム・レベルで組み合せて構成した航空管制用システムであるＩＢＭ９０２０システムがあり，最近では，Ｎｏｎ－Ｓｔｏｐ運転を旗印に，ユニークな立場を占めつつあるＴＡＮＤＥＭ－１６の例がある。

(d) システム・レベルでの解決

計算機の歴史のかなり初期から各種制御システム，リアルタイム・システムを中心に採用されてきている手法である。これは，複数の独立した計算機システムで，入出力装置と計算機本体とを切換え装置を介して接続し，障害発生時，予備系に切換える待機予備運転方式として発展してきたものである。この例には枚挙にいとまがない程である。

いずれにしろ，上述の各種手法は，ロジック・レベルでの解決手法は除くと，特定のシステムにだけ採り入れられ，普及している手法ではない。

(4) 今後の研究課題

フォールト・トレラント技術が，一般の計算機に採り入れられて普及して行くための主要な研究課題には，次のものが挙げられよう。

- ロジック・レベルで，冗長性，故障検知能力を備えたＶＬＳＩチップの開発
- マイクロ・プロセッサ等のモジュール・レベルで，ＴＭＲ手法を採用するためのVoter Chip等の開発
- マイクロ・プロセッサ・レベルでの数１０個におよぶ密結合マルチプロセッサ・システムの実現
- ソフトウェアのモジュール単位でのFunctional Recoveryを行う一般的手法の確立と，ファームウェア化によるオーバヘッド縮減手法の確立

〔参考文献〕

(1) 相磯：ＦＴＣ-Fault Tolerant Computer, bit, VOL.12, No.10, pp. 192－195

(2) 当麻，南谷：フォールト・トレラント・システム，電子通信学会誌，VOL.66, No.10, pp. 1031－1041

(3) Nicolas Mokhoff：Four Target of VLSI, IEEE Spectrum, June 1980, pp. 34－36

6．研究開発の課題

6. 研究開発の課題

6.1 はじめに

第5世代のコンピュータ・アーキテクチャの大きな目標は，知識情報処理システムや高度な分散処理システムの実現をハードウェア技術の面からサポートすることである。この中でも，巨大化，複雑化するソフトウェアの負担を，いかにして，ハードウェアで分担し，全体として，より高度なシステムを構築できるようにするかという問題が，きわめて重要である。

このためには，従来の高級言語よりもさらに一段レベルの高く，数学的基盤に基く形式化の行い易い述語論理型や関数型の言語をサポートし，高度なソフトウェアの作成を可能とする高級言語マシン，これらマシンの高性能化の中心的な役割を担うと期待されるデータフローマシン，このほか，関係データベースのサポートを目的とする関係代数マシンなどの構築を目ざす新アーキテクチャの研究が重要である。

また，これらの新しいマシンを設計・開発する場合や，これらのマシンを有機的に結合し，各種の応用目的に適合した柔軟で，高い信頼性，高いパフォーマンスを有するシステムを構築するためには，機能分散アーキテクチャの研究が不可欠である。第5世代では，ごく普通のコンピュータ・システムの実現形態となると考えられる，ローカル・ネットワークをベースとし，パーソナル・コンピュータや，種々の機能別マシンを組合せた分散システムは，機能分散アーキテクチャの大きな目標でもある。

このような高度なシステムは，必然的に，現在のコンピュータ・システムより規模の大きなものとなる。特に，ハードウェア・システムは，ソフトウェアの負担軽減をめざし高機能化が求められることから，そこに含まれる素子数は，大幅に増加すると考えられる。ＶＬＳＩ技術は，このような問題を解決する鍵であり，特に，ＣＡＤシステムの基盤ともなるＶＬＳＩアーキテクチャの研究は，第5世代コンピュータ・アーキテクチャの実現上の中心的課題である。

以上の，新アーキテクチャ，機能分散アーキテクチャ，ＶＬＳＩアーキテクチャは，第5世代コンピュータ・アーキテクチャを形成する3つの柱である。

これら，3つのアーキテクチャを融合させ，知識情報処理システムを指向したコンピュータを作り上げるためには，新しいシステム化の技法が必要である。ここでは，ソフトウェア，ハードウェアの双方を含んだ仮想技術，融合化技術，また，大規模なシステム構築のための技術が，重要な研究テーマとなる。

この第5世代アーキテクチャのシステム化技術を含めた4つの柱が，アーキテクチャ研究分科会の重要な研究開発課題のグループを構成する。

以下，これらの課題グループに含まれる個々の研究開発課題の詳細，開発のすすめ方などについて述べる。

6.2 研究開発課題

第5世代のコンピュータ・システムの開発におけるアーキテクチャ関連の研究開発課題は，4つの課題グループから成る。これをまとめると以下のようになる。

このそれぞれの課題について，必要性，内外技術の現状，研究開発内容を述べる。

アーキテクチャ関連研究開発課題

1）新アーキテクチャ

1－1）　述語論理マシン

1－2）　関数型マシン

1－3）　関係代数マシン

1－4）　抽象データ型マシン

1－5）　データフローマシン

1－6）　新ノイマンマシン

2）機能分散アーキテクチャ

2－1）　機能分散アーキテクチャ

2－2）　ネットワーク・アーキテクチャ

2－3）　高速演算マシン

2－4）　データベース・マシン

2－5）　高度マンマシン・コミュニケーション・システム

3）ＶＬＳＩアーキテクチャ

3－1）　ＶＬＳＩアーキテクチャ

3－2）　知的ＶＬＳＩ－ＣＡＤ

4）システム化技法

4－1）　第5世代アーキテクチャのシステム化技術

6.2.1 述語論理マシン

〔1〕 概　要

述語論理マシンは，述語論理型言語によって記述されたプログラムを効率よく実行することを目的にした計算機である。

述語論理型言語というのは，文字通り，述語論理の枠組に従ったプログラミング言語一般を指すものである。

論理は論証を厳密に記述するための機構ないしことばであるが，特に述語論理は，数学的な証明の表現を行なうのに必要な表現力を十分に備えているばかりでなく，厳密な表現手段としては，人間による推論形式に最も近く，したがって人間にとって自然な形式であるということができる。

プログラミング言語の目的は，解くべき問題とその一般的な解法，すなわち，アルゴリズムを記述して，人間から機械に伝達するための表現手段を提供することである。アルゴリズムの中味を化学方程式に書くと，

アルゴリズム＝ロジック＋コントロール

となる。ここで，ロジックというのは，問題を解くのに使われる情報ないし知識の体系のことであり，コントロールというのは，それらの情報ないし知識をどのように使っていくかの制御のことである。

述語論理型言語によるプログラムでは，アルゴリズムのロジック部の記述に述語論理を用い，コントロール部はそれ専用の簡単な言語を用いたり，実行機構に委ねたりすることになっている。したがって，ロジック部は公理，定理，それに推論規則などの集合であり，コントロール部は演繹の進行手順から成り，両部は完全に分離して存在することになる。

これに対して，従来のプログラミング言語では，命令あるいはステートメントなどのような，プログラム記述の最小単位のレベルで，すでにロジック部とコントロール部の化合が行なわれている。この際立った対照は述語論理型言語の革新性を端的に示すものである。この化合は，もともとノイマン型計算機の基本的な性格に由来するもので，その遺伝は高級言語のレベルにも濃厚に認められる。

これを裏返していえば，述語論理型言語をノイマン型計算機に適合させるのには，相当な困難があるかも知れないということである。しかも，この問題以前に，述語論理をプログラミング言語化するために克服すべき致命的な課題があった。述語論理型言語は計算を定理の証明という形で実現するものであるが，これは正に定理の証明を機械で自動化することを意味している。

定理証明の機械化は，原理的には，公理あるいは定理の組合せに推論規則を適用して新しい定理を導くという操作を繰返して，目的とする定理が生み出されるまで続けるという形をとることになる。しかし，これを組合せをしらみつぶしに作って調べるというようなやり方をした

のでは，組合せの数が多くて実用にはならなくなる。そこで，実際的なアプローチとしては，述語論理の形式に制限をつけて（したがって，それだけ自然さが失なわれる），組合せのケースを圧縮するか，あるいは，人間の直感力に相当する能力をコントロール部に組込むというようなことになる。後者のアプローチは，むしろ，今後の研究課題であり，現状で確実に使いものになることが実証されているのは，前者のアプローチによるものである。それでは，述語論理をHorn節という形式に制限することになっている。

そして，このHorn節形式で表現された公理あるいは定理を，従来プログラミング言語での手続き形式としてよみ換えることによって，ノイマン型計算機に適応させるのに成功した述語論理型言語に，ＰＲＯＬＯＧがある。

ＰＲＯＬＯＧは英国において開発され，以来，同国ならびにフランス，カナダの大学において，実装化，改良，いくつかの大規模応用における有効性実証など広汎な努力が続けられている。これらの実績を通じて，人工知能分野を中心に，データベース関係，ある種の技術計算類などにおけるＰＲＯＬＯＧの定評は，ほぼ確立されたものとみることができる。この間にあって，米国はＰＲＯＬＯＧに対してほとんど沈黙の状態にあるのが注目される。これは米国特有のＮＩＨ（Not Invented Here)的気風のしからしめるところであるらしい。

ＰＲＯＬＯＧによって述語論理型言語は完全に実用の段階に入ったということができる。今後の課題は，より大規模な応用のために，より高いパフォーマンスの水準に達することである。このためには，単にノイマン型計算機の枠内での改良を考えるだけでは不足で，述語論理本来の性格に適合した計算機アーキテクチャの追求が必要になっている。

述語論理の基本演算は，演繹（ deduction ）である。演繹の基本操作は，変数を含んだ２つの記号列に対して，変数に許された範囲での代入置換を行ないながら，列全体のパターンの一致を調べることである。長さの異なる多数の記号列を記憶し，記号パターンによって適当な候補の列を探し出す機能，上述のようなパターン照合（ユニフィケーション）を行なう機能，一致のとれた記号列に対してある演繹操作を行なって，新しい記号列を作り出す（リゾリューション）機能，こうした機能を繰返しても求める結果に到達できなかったとき，まえに見捨てられていた候補の列に戻って再出発する（バックトラック）機能，これらの機能を効率よく実行できる計算機というのが，述語論理マシンのイメージである。

開発すべき述語論理マシンに期待される性能は，現在のノイマン型マシンで数秒から数１０秒を要する問題を，ミリセカンドで処理し終える程度のものである。これだけの性能向上を実現するためには，極めて高度な並列処理機能の導入が必要である。

〔2〕 必要性

述語論理マシンは述語論理型言語によるプログラム（以下では単に述語論理プログラムと略す）を効率よく実行するために存在する計算機であるから，その必要性の根源的な部分は，結

局，述語論理型言語そのものの有効性・必要性に依存することになる。そして，必要性の残りの部分は，述語論理プログラムを述語論理マシンで実行することの合理性にかかっている。

述語論理型言語の必要性・有効性について考える場合に大切なことは，この言語が従来の情報処理の枠を超えた新しい応用分野，すなわち，知識情報処理を活動の場としていることである。したがって，その評価はこの新しい場に即して行なわれなければならないのである。「脱工業化社会の到来」の著者であるD.ベルによれば,知識とは"事実あるいは思想を述べた論述の組織的なまとまりで合理的な判断や実験結果を提示するもの"で，"通信媒体により，体系的な形で他人に伝達される"ものである。このような知識を対象にした情報処理の内容は，必然的に人間の知的活動と呼ばれているもののうちの機械化可能な部分ということになる。何がどのようにして機械化できるかの見通しについては，当然議論のあるところであるが，少なくとも現在までに人工知能研究の成果として得られているものは，その確定分であるといえる。

知識の機械的な表現は，記号化された形式的表現，すなわち，記号列である。このような形式的表現の中心は，いうまでもなく，述語論理による命題形式と数学における関数形式である。序にいえば，従来の情報処理に用いられているデータ構造形式も，この仲間に加えてよいものの1つである。

知識に対する知的な操作のうち，機械化が原理的なレベルにせよ保証されているものとして有力なのは，論理的な推論や関数式に対する変形操作などであり，さらに現在ＬＩＳＰで行なわれているような，データ構造に対する記号的変形操作もこれに含めることができる。

以上が，知識情報処理において述語論理，したがって述語論理型言語に期待される役割の説明である。つぎは，述語論理マシンの存在理由を明らかにする番であるが，それはただよりよい効率の追求ということにすべてがかかっている。前述したように，述語論理型言語とノイマン型計算機の間には大きな不適合があるので，最初から述語論理プログラムの目的にして，それに適したアーキテクチャを開発すれば，相当な性能の改善を期待することができる。まだ述語論理マシンの具体化がなされていないので，これを直接示すことはできないが，ＬＩＳＰマシンでの実績などからの類推によって若干の見通しをうることができる。

〔3〕 内外技術の現状

(1) 国内技術

述語論理における定理証明の機械化については，わが国においてもすでに１９５０年代からの研究史があるが，実用的な形での述語論理プログラミングの開始には，やはりPROLOG言語の導入をまたなければならなかった。

ＰＲＯＬＯＧの処理系は，電総研をはじめ，電電公社武蔵野通研，東京大学などに実装され，データベース処理，自然言語処理，問題解決などおける実験的使用が試みられている。その結果はＰＲＯＬＯＧの有効性を裏付けるものであった。現在は，さらに，ＰＲＯＬＯＧ

に対する効率化のための改良，並列処理化，図形言語化などに研究者の関心が集まっている。

また，ここ１，２年来，述語論理マシンの実現，特にデータフロー計算機としての実現方式についての研究が各所で開始されている。

(3) 海外技術

ヨーロッパ各地におけるＰＲＯＬＯＧへの関心は非常に高く，これを巡って多面にわたる研究が精力的に進められているようである。実用化の面では，性能の良いコンパイラが実現されたため，本格的な応用レベルでの研究が広汎な分野において展開されつつある。また，ＰＲＯＬＯＧに対する改良提案も活発に行なわれており，ソフトウェア工学指向の新機能をどのように取入れるか，コントロール部のあり方は，といった所にも興味も呼んでいる。

ＰＲＯＬＯＧの並列実行方式，データフロー形式のＰＲＯＬＯＧマシンを対象とする研究が，その口火を米国で切っているのが注目される。

米国においては，ＰＲＯＬＯＧ以外の述語論理型言語の研究が多くなる傾向にあるようである。

〔4〕 研究開発内容

(1) 研究開発項目

研究開発内容を大分類化するとつぎのようになる。

a) 述語論理型言語の開発と実装化

b) 基本技術開発

c) 述語論理マシンの実現

以下，これらの項目をさらにブレーク・ダウンして説明する。

a) 述語論理型言語の開発と実装化

基本的には，ＰＲＯＬＯＧ言語に即したものと，自然演えきの考え方に従うものとの２系統を，並行的に追求する。

実装化については，プログラム開発の局面での効率化を目的にした高級言語計算機としての実現と，大規模なプロダクション・ジョブの実行効率化を目的にした，コンパイル方式での実現とを，並行して進めるが，機種上での実装に当っては，その用途に合わせていずれかに特化することもある。

以上をまとめると，つぎの通りである。

- ＰＲＯＬＯＧ系
  - i. 拡張ＰＲＯＬＯＧ言語開発
  - ii. 実装化：コンパイル方式および高級言語計算機方式
- 新言語系
  - i. 新述語論理型言語開発
  - ii. 実装化：（同上）

b) 基本技術開発

述語論理プログラムの実行速度を飛躍的に向上させるための基本技術の開発を行なうものである。性能向上のための基本的なアプローチは，プログラム並列実行方式の導入と，述語論理型言語の特性に適合する専用的なプログラム実行機構の導入の２点である。これらの基本技術は，その性格上言語のやゝ細部にわたる構造に依存する部分が多く，上述のように２系列の述語論理型言語を追求することに対応して，基本技術開発も一部で２系列化することが必要である。

ここでの開発内容の概要はつぎの通りである。

○並列化方式の追求

ⅰ．基礎検討
- 並列制御方式
  （ＰＲＯＬＯＧ系並列化）
- 並列計算モデル
  （高度並列化）

ⅱ．実装化検討
- 資源管理手法
- 並列パターン照合機構

○専用的機構の開発

ⅰ．連想処理機構
- 連想プロセッサ
- 連想メモリ

ⅱ．節格納用構造メモリ

c) 述語論理マシンの実現

２系列の論理型言語のそれぞれについて，プロダクション・ジョブの処理を目的にする大規模マシンと，プログラム開発用を目的にする小規模（パーソナル）マシンのバリエーションが考えられる。また，開発のステップとして，開発実験的モデルと本番開発としてのパイロット・モデルの２段階が必要である。しかし，これらのすべての組合せを実現することは，明らかに現実的ではない。必要な実現時期と，前提となる要素技術の完成見込時期とを考慮に入れて，最小限必要と考えられるものを選別すると，つぎの３モデルに絞られる。

1) モデルⅠ：開発実験モデル

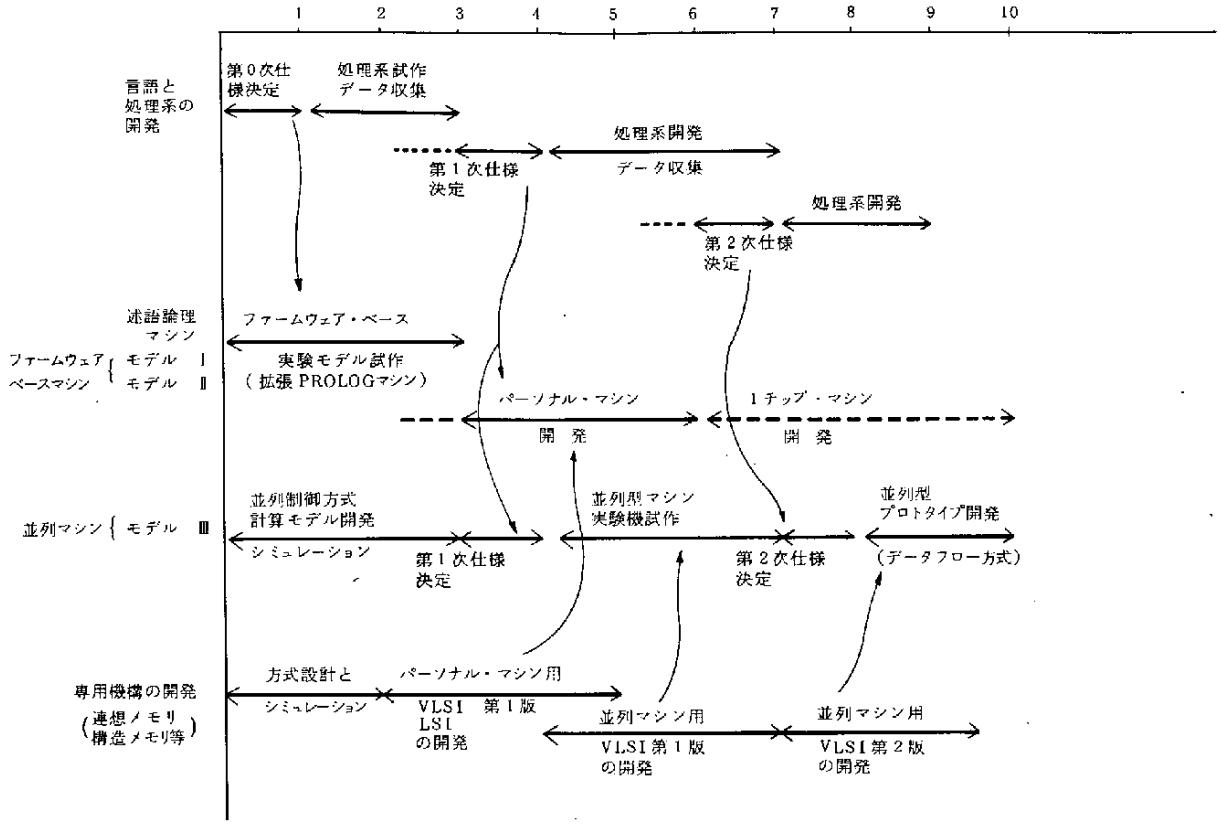
2) モデルⅡ：専用ファームウェア・ベース・マシン

3) モデルⅢ：並列述語論理マシン

(2) 研究開発スケジュール

研究開発スケジュールは表６－１の線表の通り，

表６－１　述語論理マシンの研究開発スケジュール



(3)　研究開発の目標・仕様

①　拡張ＰＲＯＬＯＧ言語

主目標は知識情報処理システムの研究開発でのツールとなることであり，現在までの研究成果を集大成して，仕様に反映するという程度のものを予定。初期に第ゼロ次仕様を決定し，実験モデルを試作する。中期に第１次仕様を決定し以後実用に供せられることが必要。

②　新述語論理型言語

長期にわたる安定した使用に耐えられるだけの完備した仕様をもつものでなければならない。中期末までに最終仕様を決定し，プロトタイプマシン上での実用化を目ざす。

③　述語論理マシン

モデルⅠ：開発実験モデル

前期に完成を目標とするＰＲＯＬＯＧベースの小規模マシンで，ファームウェアによって実現するもの。

性能目標は現行の通常マシン上にソフトウェアで実現されているものの約１０倍の水準である。コンパイル方式と高級言語計算機方式の両方が利用可能である。

このモデルの開発目的は，ＰＲＯＬＯＧマシンひいては述語論理マシンの性能的な可

能性を見極めるための実験台を提供することである。

モデルⅡ：専用ファームウェア・ベース・マシン

中期に完成を目標とする述語論理マシンで，理論研究や言語研究をサポートするパーソナル計算機としての規模をもつものである。専用ファームウェア・ベース・マシン（記号処理向ＵＨＭ）としての形式に従うことを想定している。

性能目標は現行のものの数１０倍から１００倍弱の水準であるが，高級言語計算機方式としての性能に重点をおくものとする。

このモデルの開発目的は，商用パーソナル述語論理マシンのパイロット・モデルを提供することである。

モデルⅢ：並列述語論理マシン

１９９０年完成を目標とする高度並列化述語論理マシンで，大規模な述語論理プログラムを高速処理するための共同利用向き大型機であり，データフロー方式の利用を想定している。

性能目標は，現在の水準に対して数１００倍から数1,000倍の線で，主としてコンパイル方式での性能に重点をおくものとする。このモデルの開発のために，中期に実験機を試作する。最終のプロトタイプマシンの目的は大規模述語論理マシンの実現可能性を実証するとともに，このクラスの商用機のためのパイロット・モデルとなることである。

6.2.2　関数型マシン

〔1〕 概要

第５世代のコンピュータにおいて重要になる知識ベースの実現とソフトウェアの生産性の向上には，関数型言語などの理論的基盤をもつプログラミング言語の利用が望ましい。一方，ＬＩＳＰとかＩＮＴＥＲＬＩＳＰで代表される関数型言語は以下の理由から，従来のコンピュータでは処理速度が上がらない。

① 関数型言語処理で標準的に利用されるリスト構造データを順次に追う処理は従来のコンピュータは得意でない。

また，要求される性能が現在のコンピュータの１桁～２桁上であるので，関数型言語処理に向いたマシンを新たに設計し，利用可能にする必要がある。

計算機のアーキテクチャの面から，知識情報処理をサポートするには，次の性能が望まれている。

① ネットワークの共通リソースとしての関数型マシン

現在の汎用コンピュータ（例えば，４ＭＩＰＳ）のリスト処理スピードの10～1,000倍。

② パーソナル・マシンとしての関数型マシン

卓上型のマシンとして，現在の汎用コンピュータ（例えば，４ＭＩＰＳ）のリスト処理スピードの１～１０倍。

関数型マシンは関数型言語の利用者に役立つとともに，述語論理型言語等の言語の実行メカニズムとしての役を果たす。これにより，知識ベースを利用可能なものと出来るし，ソフトウェアの生産性の向上に貢献する。

関数型プログラミングとソフトウェアの生産性：

ＩＢＭ社のBackus氏は現在のプログラミング言語には次のような欠点があると指摘している。

現在の言語にはフォン・ノイマン・マシンの概念を引きついだ，フォン・ノイマン・プログラミング言語（言語のモデルがフォン・ノイマン・マシン）であり，

① プログラム作成者の思考が分断される。（Intellectual Bottleneck）

一語のデータを取り出しては処理し，一語のデータを結果として格納することの繰り返し。

② ストラクチャード・プログラミングも，フォン・ノイマン・プログラミング言語に，ある種の秩序を入れたぐらいで，本質的な解答にはなっていない。

Backus氏はこの問題をF.P（Functional Programming:関数型プログラミング）で解決出来ると論じている。

① ＦＰでは，変数の値にhistory sensitivityがないのでプログラムの論理を考え易い。

—関数合成が自然の形でやれる—

（call by name/value，変数のスコープ等に悩まされることがない。）

② 集合演算を考えることで，繰り返し文（for I=1 to N）を排除出来る。

従来のプログラミング言語の欠点を除くべく，Backus氏の指摘のように，ＦＰの技術の適用を検討すべきであろう。

但し，現在ＰＬ／ＩとかＣＯＢＯＬでプログラミングしている者が関数型言語（例えば，ＬＩＳＰ言語）に切り替えて，すぐに効果を上げられるとは考え難い。

「教育」とか「慣れ」の問題の他に次の２点を研究する必要があろう。

(i) 関数型言語自身の表現形式

M形式／S形式，関数グラフ形式等

(ii) 関数型言語の持つべき機能

〔2〕 必要性

以下に現在のマシンとの相違点及び，第５世代コンピュータの他のマシンとの関係を示すことで，関数型マシンの必要性を示す。

(1) 現在のマシンとの相違点

① 関数型言語を高速に処理する。

② リスト構造のデータに関する次の処理が速い。

—トリー・ウォーク（Tree walk）

—要素選択（Component selection）

—パターン・マッチング

低レベルの「単なる文字列の一致」から，高レベルでは「プロダクションやHornクローズの処理」に至る，パターン・マッチングの処理の高速化が期待されている。

③ 関数の呼び出し，特にリカーシブな関数呼び出しを高速に処理する。

—スタッキング機構

④ 動的な記憶域管理を高速に実行する。

—並行ガーベジ・コレクション機構

(2) 第5世代コンピュータの他のマシンとの関係

関数型マシンは関数型言語の利用者をサポートするだけでなく，ＰＲＯＬＯＧ等を処理する述語論理型マシン，抽象データ・タイプ等を取り入れた言語のための抽象データ型マシンの実行メカニズムとしての役割をはたす。

また，集合演算機能を得意とする関係代数マシンも関数型マシンで実現することが十分考えられる。

関数型マシンの実現方式としてデータフローマシンがあり，興味ある研究がおこなわれている。

(3) 内外技術の現状

以下に関数型マシンの処理をインタプリタの入力となる言語レベルの観点から分類し，内外技術の現状を示す。

(i) 言語を直接インタプリト

(ii) 言語→Ｓ式；Ｓ式をインタプリト

(iii) 言語→（専用ＭＬ）；（専用ＭＬ）をインタプリト

この方式のＬＩＳＰマシンの研究が多い。

(注) ＭＬは機械語（Machine Languageの略である）。

(iv) (ii)と(iii)の両方式をサポート

（電電公社 ＥＬＩＳにその例が見られる）

これらのインタプリタの高速化を目差すマシンの研究が国内外で進んでいる。

次に現在研究段階にある関数型マシンの実現方式を分類する。

Ⅰ．専用ＭＬの形式

(1) 逐次実行型専用ＭＬ

－Deutch のＬＩＳＰマシンのコード

－ＭＩＴのＣＯＮＳプロセッサのコード

－ユタ大のＭＢＡＬＭコード（図６－１参照）等

(2) 関数型グラフ言語

ユタ大のＦＧＬ（関数型グラフ言語）

Ⅱ．インタプリタの制御方式

(1) マイクロプログラミング方式

(a) 商用のマイクロプログラム方式コンピュータを利用する方式。

ユタ大学のＭＢＡＬＭマシン（Ｂ１７２６）

(b) 専用のマイクロプログラム方式コンピュータを設計する方式。

－ＣＭＵのＬＩＳＰマシン

－ＭＩＴ，ＣＯＮＳプロセッサ

－ＭＩＴ，Sussman の１チップ・プロセッサ（図６－２参照）

－電総研，ＬＩＳＰマシンⅡ

－京大，ＮＫ３

－慶大，ＬＩＳＰマシン

－神戸大，ＬＩＳＰマシン

－電電公社，ＥＬＩＳ

(2) ワイヤード・ロジック方式

Ⅲ．インタプリタの制御の多重度

(1) 単一制御

現在までに研究･試作されているＬＩＳＰマシンは逐時実行型である。

(2) 多重制御

関数評価の並列実行

(a) データフロー方式

－東大のＬＩＳＰマシン

(b) リダクション・マシン方式

－Magoのリダクション・マシン

－Berklingのリダクション・マシン

(c) デマント・ドリブン方式

—ユタ大学のKellerマシン（図６６－３参照）

現在のＯＳのプロセス管理方式の「プロセス」を「関数呼び出し」に置き換えたようなもの。

（プロセッサの個数は，例えば１０００個を考えている）

(注) Magoのリダクション・マシンは言語の文字列を見渡し，リダクション（式の評価）のできる部分の集合に対して,並行してリダクションを行う。この動作の繰り返しで「言語の実行」をする。一次元配列のセルと，トリー型配列のセルとで構成される。ＶＬＳＩ向きの構成である。（図６－４，６－５，６－６参照）

Ⅳ．シンボルテーブル・サーチ方式

シンボルテーブルのサーチの高速化のためには，順次にテーブルを検索する処理を高速化する方式やハッシング方式がある。

(1) テーブル・サーチ方式

連想メモリの利用等，ハードウェアの論理で高速化することが考えられる。

(2) ハッシング方式

ファームウェアやハードウェアでハッシング検索機構を実現し高速にする。

理化学研のＦＬＡＴＳマシンはハードウェアでハッシングを行っている。

Ⅴ．データ形式と構造メモリ

関数型マシンの実現には，以下のようなデータ形式が利用される。

- タグ付データ
- ディスクリプタ

また，インタプリタの中では，処理上次のようなデータ形式が利用される。

- スタック
- ランタイム・シンボル・テーブル
- ヒープ

これらのデータ形を高速に扱う機構が必要となる。特にＶＬＳＩを使用した構造メモリ（連想メモリ等）が期待される。

Ⅳ．ガーベジ・コレクション方式

記憶域の動的な管理を行うガーベジ・コレクションの方式は，各種研究が進んでいる。

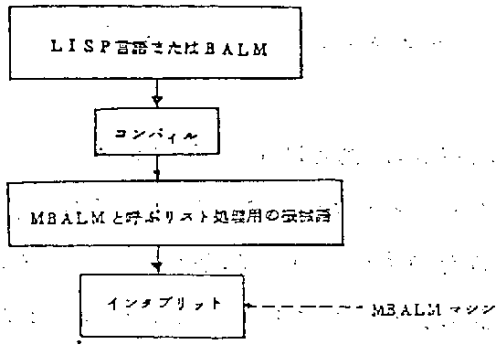
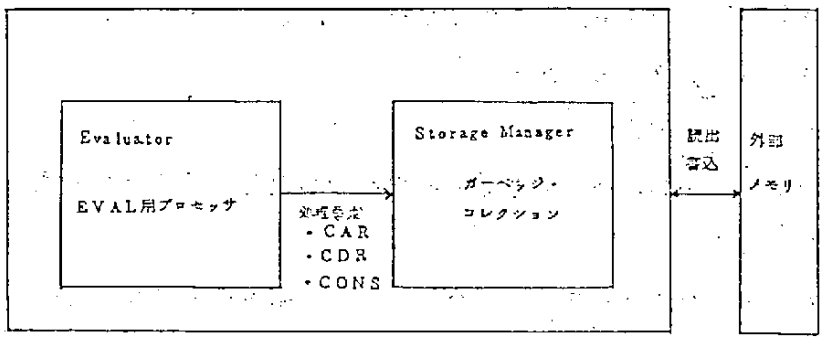


図６－１　ユタ大学のマイクロプログラム化ＬＩＳＰマシン



とのＭＩＴのＬＩＳＰプロセッサも基本的には、ユタ大学の方式と同様

ＬＩＳＰプログラムを一度オブジェクト・コードに変換した後に実行する方式

（コンパイル方式）

図６－２　ＭＩＴのＶＬＳＩ化ＬＩＳＰプロセッサ

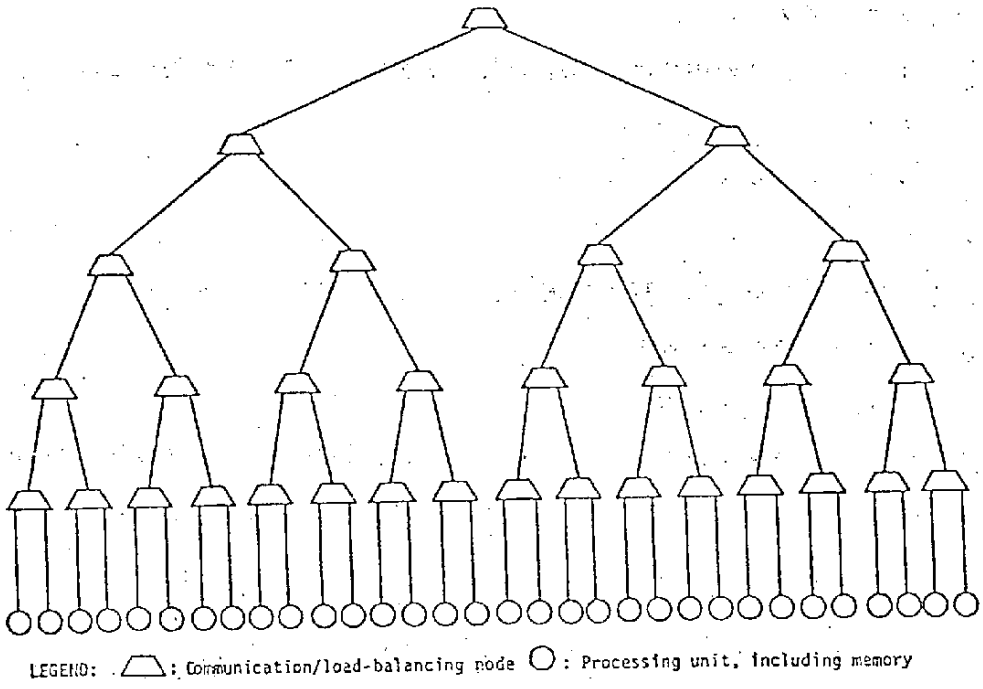


図６－３　ユタ大学のKellerマシン

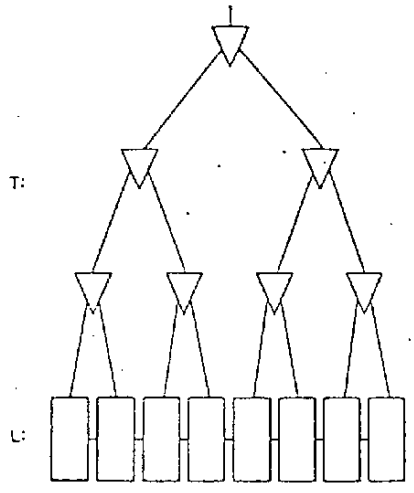


図６－４　Magoのリダクション・マシン

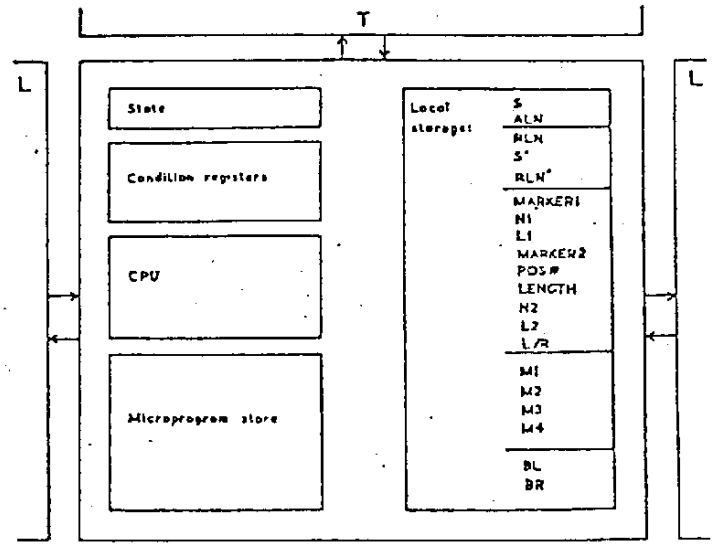


図６－５　Magoのリダクション・マシンのセルＬの構造

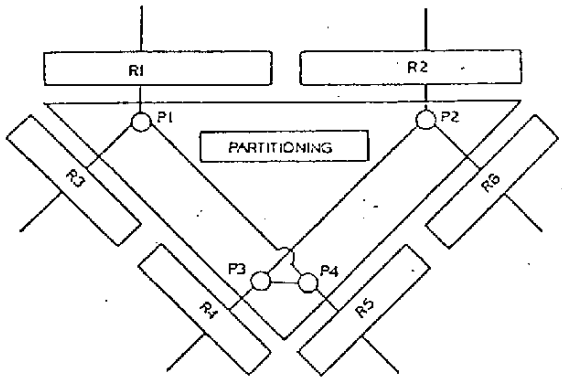


図６－６　Magoのリダクション・マシンのセルＴの構造

〔4〕 研究開発内容

(1) 研究開発項目

関数型マシンの開発のための，主要な研究・開発項目を次に述べる。

① 関数型言語の仕様の確立

ＬＩＳＰ，ＩＮＴＥＲＬＩＳＰ等の関数型言語を参考にして，第５世代コンピュータ用の関数型言語の持つべき機能を検討し，言語仕様を定める。この際　言語の理解容易な表現（グラフ表現等）にも重点を置く。

② 並列処理記述用の関数型言語の仕様の確立

①で開発した，関数型言語を拡張して，論理的に並列処理可能な処理を，最大限並列処理することが可能なように，並列処理記述可能な関数型言語の仕様を定める。

③ ファームウェア・ベースのパーソナル関数型マシンの開発

マイクロ・プログラム方式の関数型言語のインタプリンタを持つパーソナル・マシンを開発する。これは，開発の前期に主点的に開発し第５世代コンピュータの研究促進用に利用する。

④ ＶＬＳＩ記号処理用ＵＨＭの開発

関数型マシン，述語論理マシン，抽象データ型マシン等のマシンに共用して利用出来る記号処理用のＵＨＭ（汎用ホストマシン）を設計し，これをＶＬＳＩ化する。このVLSIプロセッサは，中期における並列プロセッサ方式による高速関数型マシンの要素プロセッサとなる。

⑤ 並列処理方式の確立

関数型言語の処理を高速化のための，並列プロセッサによる並列処理方式を確立しデータフローの処理技術と結びつける。

⑥ ＶＬＳＩ並列型構造メモリの開発

一般的なパターン・マッチングの高速化の技術（多数のHornクローズを高速に処理する技術等）を確立し，ＶＬＳＩ並列型構造メモリを開発する。

⑦ プロセッサ間結合方式の確立

関数型マシンと他の第５世代のコンピュータのマシンとの情報の受け渡し（処理要求と処理結果の受け渡し）を高速に行うための，プロセッサ間通信方式を確立する。

(2) 研究開発スケジュール

研究開発期間は約１０年として，その期間を大きく３期に分ける。前期では，研究促進用のパーソナルな関数型マシンの研究・開発を主体とする。中期では，並列プロセッサ方式による高速関数型マシンを研究・開発する。後期では，データ・フロー方式とＶＬＳＩ並列型構造メモリを用いた超高速関数型マシンを研究・開発する。

表６－２　関数型マシンの研究開発スケジュール

	前　　期	中　　期	後　　期	速　度(注)
パーソナル関数型マシン	関数型言語 関数型言語→ＭＬ変換プログラム ファームウェア方式関数型マシン			２～１０
高速関数型マシン		並列処理関数型言語 並列処理関数型言語→ML変換プログラム 並列プロセッサ方式 VLSI記号処理用UHM 高速関数型マシン		１０～１００
超高速関数型マシン			データフロー方式 VLSI型構造メモリ 超高速関数型マシン	１００～１,０００

(注) 4MIPSの汎用コンピュータでのリスト処理速度に比較しての倍数。

(3)　研究開発の目標・仕様

前期　ファームウェア方式のパーソナル関数型マシンを開発する。このマシンは，中期以降，複数台生産して，第５世代の研究促進用として，他のマシンの研究・開発や，ソフトウェアの研究・開発に利用する。

性能は，４ＭＩＰＳの汎用コンピュータでのリスト処理速度の２～１０倍を目標とする。

中期　並列プロセッサ方式で高速関数型マシンを開発する。

性能は，４ＭＩＰＳの汎用コンピュータでのリスト処理速度の１０～１００倍を目標とする。

後期　データフロー技術とＶＬＳＩ並列型構造メモリを持った超高関数型マシンであり，

性能は4MIPSの汎用コンピュータでのリスト処理速度の100～1000倍を目標とする。

6.2.3 関係代数マシン

〔1〕 概要

知識情報処理においては，いろいろな応用分野における専門的知識を，知識ベースに蓄え，これを適宜参照することで，計算機に知的な，すなわち，気の効いた処理を行わせる。

知識の内容は，各種考えられるが，単語や文法の辞書，設計・製造上の規則，作業手順のようなもので，計算機内で扱われる場合は最終的にはデータやプログラムの集合となる。

その構造は一般的に，いくつかの階層から成るが，ハードウェアに近いレベルでは，表（テーブル）のような簡単な構造のものが望しい。もちろん，表のみでは実現できず，内容呼出し記憶や，表の間を結び連想検索などを必要とするが，その基本構造は，簡単で，ＶＬＳＩ化などハードウェア化に適したものであることが望しい。それとともに，その構成の規則が，数学的理論に裏打ちされたものであり，検索，更新操作が関数型言語などにより記述され得るものである必要がある。

ここで述べる関係代数マシンは，関係型データモデルに基くデータベース処理を，大規模のデータについて能率よく実行することを目的とするマシンである。関係型データモデルは，表形式の構造を基本とし，演算としては，検索，整列，結合を基礎とする関係代数演算を中心としている。さらに，この上に，データベースに対する質問応答や管理システムを実現するための関係論理に基く言語と演算が備えられる。ユーザからこのマシンを見る場合は，関係論理に基くデータベース向き高級言語マシンとなる。

このように整った形式的仕様を持つことから，将来のデータベースの大規模化，複雑化に対処できる新しいデータベース・マシンとして注目されているとともに，知識ベースに重要な推論のメカニズム等とも，言語面や理論面での整合性が良い。実現に際しては，関係代数マシンが，大規模システムの構築をめざし，ソフトウェアや理論面の要請が主となっていることから，ソフトウェア面での負担が，ハードウェアへと移されている。このための汎用計算機などのいわゆるノイマン型アーキテクチャでは，高性能のマシンを作り得ない。このため，新しい並列処理手法に基く，強力なハードウェア・プロセッサの開発を必要とする。このためには，数百台以上の要素プロセッサから成り，数十ギガバイト以上の記憶容量を持つマシンをめざす開発研究が必要で，当然，ＶＬＳＩ技術が必須である。これと同時に，ソフトウェア面での関係論理，関係代数レベルの言語と処理系の研究開発，データベース管理用ソフトウェアの研究開発が必要である。

関係代数マシンの用途としては，その開発の中間段階で関係データベース・マシンとして実用化の可能性がある。最終的には，知識ベースをサポートする高機能マシン（モジュール）と

して，第５世代マシンの主要部分をなすものと考えられる。

以上のような理由から，第５世代マシン開発の重要な研究開発課題である。

〔2〕 必要性

関係代数マシンは，そのインタフェースとなる基本操作が，明確な数学的基盤に基づいて決められていることから，知識情報処理システムのような複雑なシステムの構築に適するとともに，推論などのより高レベル処理を行う術語論理マシンのデータベース・サポート機能などを受持つことができる。関係代数マシンのソフトウェア，ハードウェアを目的に応じ適合化させることで，自然言語処理，機械翻訳，コンサルテーション・システム，ＣＡＤシステムなどの知識ベースの高速化，高能率化，大容量化などがはかれ，応用システムの性能が，量的にも質的にも向上する。

第５世代マシンにおいては，データベースのサポートは，きわめて重要であり，これまでの座席予約や在庫管理，文献検索などの定型的なシステムとは異るものが要求される。

特に，知識ベースの高度な利用分野である機械翻訳や，医学や法律などの専門的知識を扱うコンサルテーション・システム，経営情報，事務手続き規則なども扱うオフィス・オートメーション・システムなどでは，単にデータをとり出したり貯えたりするのみならず，不完全な質問に対する応答，いくつもの事実を組合せ推論して新たな解答を与える能力，なぜそのような解答を与えたかが説明できる能力などが必要となる。

このような能力を実現する手法は知識ベース管理システムの研究テーマであるが，これをサポートするマシン・アーキテクチャやハードウェアとしては，きわめて高い性能が要求される。現時点で考えられるマシンの方式としては，論理的推論や演算処理の基本機能が最もよく整理されている関係型データベースとそれをサポートする関係代数マシンが有力である。また，蓄える知識の量，すなわち，マシンの記憶容量も大きくなることが予想され，汎用マシンによる実現では，要求性能の実現は困難と思われ，関係代数マシンのような機能分散の考え方をとり入れた専用ハードウェアの開発が是非とも必要である。

当然これに伴い，マシンを構成する要素プロセッサや接続メカニズムのＶＬＳＩ化，大容量，高速の記憶システムを実現するための連想メモリ・チップ，高速ディスク，光ディスクのような大容量デバイスの研究開発も必要である。

〔3〕 内外技術の現状

関係代数マシンの研究は，関係データベース・マシンとして研究され，ソフトウェアによって，既存汎用機上に試作されたり，ハードウェアによる実験機が試作されたりしている。また，マシンの構成についてもいくつか提案がなされている。現状においては，関係代数マシンを集合演算を得意とする関数型マシンととらえ，そのアーキテクチャを研究する方向と，データベース・マシンとして，そのプログラミング言語，データの構成手法，データの意味論，概念変

換，データのインテグリティの保障を対象とするソフトウェア面の研究の方向の２つに大別できる。

アーキテクチャ面の研究における問題は，容量や処理速度に関し，十分な能力を有するハードウェアが妥当なパフォーマンスで，実現できるかという点にあり，このためには，アルゴリズムの改良や，ＶＬＳＩ化などの研究と深く関係する。このため，ソフトウェアから部品面にわたる総合的な研究開発を必要とする。現時点では，ソフトウェアによるシミュレーションと，小規模な実験機の製作を併用した，アーキテクチャの模索が続けられている。

ソフトウェア面の研究は，質問応答のための言語，マシンの階層化に対応する中間レベルの言語の研究，データベース管理システムの構成の研究，マシン化する場合のアルゴリズムの研究が行われている。言語としては，ＡＬＰＨＡ，ＱＢＥ，ＳＥＱＵＥＬ（ＳＱＬともいう）などが提案され処理系も試作されている。しかし，データの意味論や，知識表現に適したデータモデル論の研究は，まだはじめられたばかりで，知識情報処理の重要な課題として，今後の発展が望まれる。

以下，主に，ハードウェアやアーキテクチャ面の研究を中心に内外の状況を述べる。

(1) 国内技術

国内では，関係代数に基く基本演算を行うアーキテクチャの研究が近年盛んである。アーキテクチャのレベルでは，関係モデル以外のもののサポートを同時に狙ったものもある。

電総研では，バブルメモリを用いたマルチプロセッサ方式のデータベース・マシンEDCを試作した。また，ソフトウェア面では，知的データベース・アクセスの研究も行っている。通研では，最近，関係代数の基本演算を行うマシン・アーキテクチャの研究を開始している。北大では，ストリーム処理を導入した新しいマシンの構成法について研究している。このほか，関連する研究が，横浜国大，東大，慶大などで行われている。

また，東芝をはじめとするいくつかのメーカの研究所でも，基礎的な研究が行われている。

(2) 海外技術

海外では，まず，関係モデルを提案したＩＢＭのCoddの先駆的研究がある。IBMでは，その後，既存マシン上へ関係データベース・システムＳＹＳＴＥＭ－Ｒを実装し，商用化をめざした研究が続けられている。

ハードウェアの研究としては，トロント大学のＲＡＰシステムが有名である。これは，固定ヘッドのディスクを用い，このヘッドのおのおのにマイクロ・プロセッサを付加し，連想処理を実現しようとするもので，回転記憶を用いたアーキテクチャの代表的なものである。このほか，ウィスコシン大学では，クロスバー・スイッチによりプロセッサやディスクを結合した多重マイクロ・プロセッサ・システムＤＩＲＥＣＴを開発している。このほか，いく

つかのマシンの提案や実験機の試作の例があるが，いずれも基礎的なものである。

しかし，ＳＹＳＴＥＭ－Ｒにみられるように，商用化システムがソフトウェア的に整ってくるとともに，ハードウェア面も徐々に実用化をめざしたものが研究されるものと思われる。

〔4〕 研究開発内容

(1) 研究開発項目

① インタフェース言語と処理系の開発

マシン言語として，関係論理，関係代数，マシン基本演算の３段階のものを開発する。関係論理のレベルは，最も高いレベルとし，ソフトウェアで処理系を作る。これによりデータベースをアクセスする。関係代数は中間言語的役割を持たせ，この実行は，探索，整列，結合，制御用スカラー演算，条件判定演算などを含むマシン基本演算により行う。これら，３者間の言語変換・最適化のための処理系を開発する。

② データベース管理システムの開発

データベース・システムとして，データを構造化し，参照するのに必要な管理システム（ＯＳ）を開発する。これは，データベースのインテグリティの管理や機密保護の機能も持つ

③ マシン基本演算用アルゴリズムの研究

マシン基本演算の主なもの（探索，整列，結合，その他）について，これを効果的に実行し得るアルゴリズムについて研究する。これは，直接，マシン・アーキテクチャの決定に影響するため重要である。特に，並列処理を多用することから，並列処理向きアルゴリズムを開発する。また，ＶＬＳＩによる実装を考慮し，ＶＬＳＩ向きのアルゴリズムも研究する。

④ マシン・アーキテクチャの開発

並列処理を基本とし，高速化，大容量化に対応できる柔軟なアーキテクチャを開発する。また，実装に際し，ＶＬＳＩを開発する。アーキテクチャの基本方式として，マルチ・プロセッサ方式，パイプライン方式などを用い適宜組合せることとする。また，データフロー方式等の新しいアーキテクチャをとり入れる

⑤ 要素プロセッサおよび接続用ハードウェアの開発

マシン構成要素となるプロセッサ，メモリ，接続用ネットワークの構成部品などについて，個別に研究し，部品としての汎用性を高めるほか，ファームウェア技術などを活用し適応性を備えるよう工夫する。これらは，ＶＬＳＩ化することを目標とする。

⑥ 記憶階層の構成方式と記憶デバイスの開発

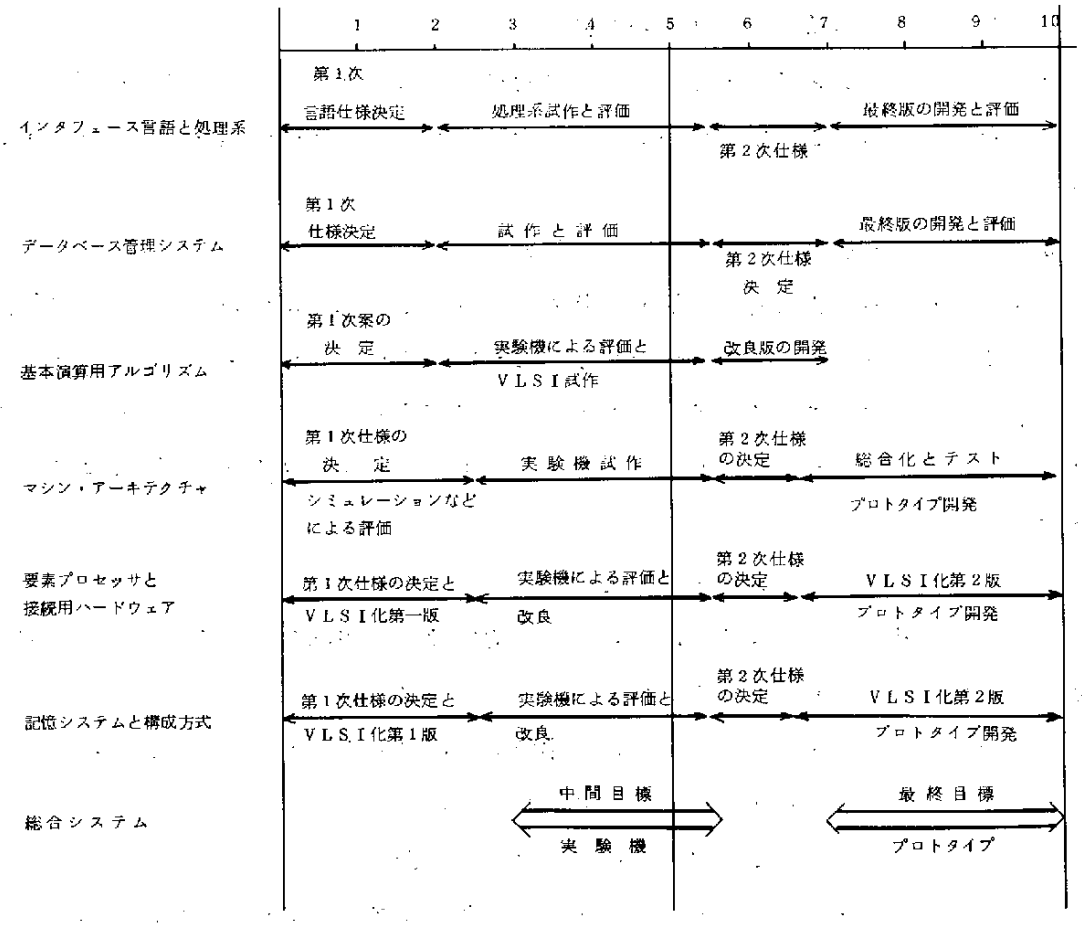
マシンの大容量化を容易にし，尚かつ，高速化するために，内容呼出し記憶のＶＬＳＩによる一部実装，階層間のデータ転送の高速・高機能化，仮想記憶化の技術開発，実装を行

う。また，光デバイスなどの新デバイスの使用も考慮する。また，目的に応じ，小容量・高速型から，大容量・低速型まで，柔軟に適応できる構成方式を開発する。

(2) 研究開発スケジュール

中間段階で，実験機を試作する。この実験機上で，ハードウェアの評価と最終仕様決定のための作業を行う。また，言語や管理システムなどのソフトウェアの開発を一部行っておく。最終のプロトタイプの試作は，実験機の製作に使用したＶＬＳＩ化した部品などの改良した第２段を用いて行う。プロトタイプ試作のためには，ソフトウェアも含めた評価結果もとりこめるような体制をとる。

表６－３　関係代数マシンの研究開発スケジュール



(3) 研究開発の目標・仕様

① 関係論理，関係代数に基づくインタフェース言語と処理系を開発する。

② 関係型データベース・システムを管理するソフトウェアを開発するとともに，知識ベース管理システムとソフトウェア的に結合する。

③ 記憶容量については，小容量・高速型の構成で，１０～１００Ｍバイト，中容量・高

～中速型の構成で，１００～１０Ｇバイト，大容量・中～低速型の構成で，１０～1000Ｇバイトを目標とする。大容量型は，画像データベースも扱えるものとする。アクセス時間は，述語論理型マシンの大規模なもののデータ格納用として十分な速度を目標とする。

④　並列処理プロセッサにより実現する場合は，要素プロセッサ数，数百台以上を目ざす。（中間目標，１００台以下）

⑤　使用する要素プロセッサ，接続用スイッチ，内容呼出し記憶などは，できる限りＶＬＳＩ化することとする。

6.2.4　抽象データ型マシン

〔1〕 概要

情報処理が対象とする情報に注目すると，それぞれ独特のデータ構造と属性をもっていることがわかる。例えば，数値処理では，整数，実数，複素数等が基本のデータ型で，データ構造としては，ベクトル（１次元配列）や行列（２次元配列）等をとる。リスト処理では，アトムとよばれる基本要素が木構造に配置されている。このように情報に結びついた固有性は，その情報を処理する場合に非常に有効で，処理をわかり易くしている。

情報処理の発展と共に情報の特性に対する理解が深まって来た。具体的な対象に対する情報処理知識が増大する一方で，それら具象を包含して統一的観点から把えてみようとする動きもでてきた。抽象データ型の概念はその１つとして発生してきたもので，最近はソフトウエア工学における階層型設計手法やプログラム正当性の検証等に使われて，概念の有効性を示している。

狭義の抽象データ型は，データに対する抽象化で，具体的にはスタック，キュー，リスト，行列等の構造のわくを定義しておき，実体はその定義された抽象データにパラメータを渡して生成し，それを対象（Object）と呼び，同一の抽象データに対し，多数の具象データを対応させる。さらに抽象データ型は手続きの定義を許し，データへの作用も記述できる。上例ではＰＵＳＨ，ＥＮＱ，ＣＡＲ，ＭＵＬＴ等の手続きが作用する。

広義の抽象データ型は，上述で定義する対象をデータに限定しないで，関数（あるいはプロセス）や制御構造にまで拡張して広汎なシステム記述能力を実現しようとするものである。この定義によるシステムの記述方法は，知識の集積に似ており，表現手段が豊富になることにより，知識の拡大につながるものである。

抽象データ型マシンは，抽象データ型言語を効率よく実行するものである。抽象データ型の有効性として，モジュール化プログラム，対象を識別する能力（Capability）からくる高信頼化プログラム等があるが，これらは，仮想的なモデルとしてみた場合であり，実際に従来の計算機で実行すると，オーバーヘッドが多くなり，実用的なシステムにはなり難い点がある。

計算機アーキテクチャの進歩およびＶＬＳＩ技術の進歩は，仮想マシンとしての抽象データ型マシンを実マシン上にマッピングすることを現実的なものとしつつあり，今後の情報処理技術の飛躍的発展の１つの方向を示すものである。

〔2〕 必要性

情報処理の対象は，初期の数値処理，事務処理から，図形・画像処理，大規模データベース・システム，信号処理，記号処理，自然言語処理等と広がっている。現在のシステムはこれら多様な分野を複雑に組合せたものとなり，この傾向はシステムの大型化と共に増大している。

情報処理システムでは，上記分野に対応してそれぞれプログラム言語を設定し，設計の容易化を図っている。分野が異なればおのずとデータ型が異なるはずで，情報の範囲が広がるにつれ，それをあつかうデータ型もふまえている。図形・画像といった情報は以前はコンピュータの処理外であったが今日では身近なものとなりつつある。さらに今後知識情報がコンピュータ処理に加わろうとしているが，その情報量は厖大である。

情報処理分野の拡大，言語の多様化，データの質的，量的拡大は，すでにソフトウェア危機を招いており，ソフトウエア技術者は，その１つの解決策として抽象データ型言語の概念を考案したがそれを現在のコンピュータでインプリメントすることは，抽象データ言語のもつ柔軟性，拡張性，ダイナミック性等のよさを制限することであり，新しいアーキテクチャによる抽象データ型言語の高速処理が必要である。

〔3〕 内外技術の現状

(1) 国内技術

抽象データ型をプログラム理論の代数的アプローチや，ソフトウェア工学における階層設計方式等に適用する研究が，大学，研究所で行われている。(表6－4参照) また一部メーカやソフトウエアハウスでも，実験的にシステム設計に適用した報告がある。

これらの試みは限定した範囲にとどまっており，未だプログラム言語としては普及していない。最近米国で発表されたＡＤＡ言語にこの概念が一部とり入れられており，この言語の紹介の中で抽象データ型に対する議論が活発に展開されるようになり，情報処理技術者の間にこの考え方が普及するものと思われる。

抽象データ型言語を実行する計算機アーキテクチャに関しては，ほとんど研究されておらず，言語の確立が先決の状態である。ただこの研究の重要性は認識されつつあり，今後多くの研究発表でとりあげられるであろうし，メーカも抽象データ型アーキテクチャを有する計算機への対応策をせまられて，研究を開始するものと思われる。

(2) 海外技術

抽象データ型言語はすでに６０年代に出現している。その後，表６－５に示すように米国においていくつかの言語が設計され，実際のシステムに適用されているものがある。長近話

表６－４　国内における抽象データ型言語の試み

名　　称	開発元	用　　途
ＨＩＳＰ	電総研	階層的プログラミング
Ｍｅｔａ－８８	電総研	問題解決向き抽象アルゴリズム
ＣＬＵ処理系	東工大	ＭＩＴのＣＬＵインプリメント／ＦＡＣＯＭ２３０－４５Ｓ
イオタ言語	京　大	プログラム検証システム
ＲＯＭＵＬＵＳ	慶　大	抽象データ型をライブラリ化（ＲＥＭＵＳ）
ＬＥＸ	九　大	問題向き言語の作成

題になっているＡＤＡは，抽象データ型の概念をとり入れており，商業的にも使われるのでその動向が注目される。

抽象データ型をコンピュータ・アーキテクチャに導入しようとする動きが，米国にみられる。（表６－６参照）　すでにＳ／３８は，対象を指向したアドレッシングをもつ商用機であり，インテルも，ＡＤＡをサポートするマイクロコンピュータを発表している。またMITやテキサス大学（図６－７参照）　では，対象指向を実現するアーキテクチャを提案している。

表６－５　海外における抽象データ型言語

名　　称	開　発　元	用　　途
ＳＩＭＵＬＡ６７	ノルウェー計算センタ	シミュレーション用
ＳＭＡＬＬＴＡＬＫ	ゼ　ロ　ッ　ク　ス	教育用
ＭＥＳＡ	ゼ　ロ　ッ　ク　ス	システム記述用
ＡＬＰＨＡＲＤ	カーネギーメロン大	プログラム検証用
ＣＬＵ	ＭＩＴ	抽象データ型一般言語
ＡＤＡ	米　国　防　省	エンベッディッド・コンピュータ用

表６－６　対象指向型計算機

名　　称	開発元	特　　徴
S / 3 8	I B M	1レベルストア，ケイパビリティ・アドレッシング
I A P X 4 3 2	インテル	ADAをサポートする予定
C L Uマシン	M I T	C L Uを実行するアーキテクチャ
Object Oriented Computer	テキサス大	オブジェクト・プロセッサ，DBを用いるアーキテクチャ

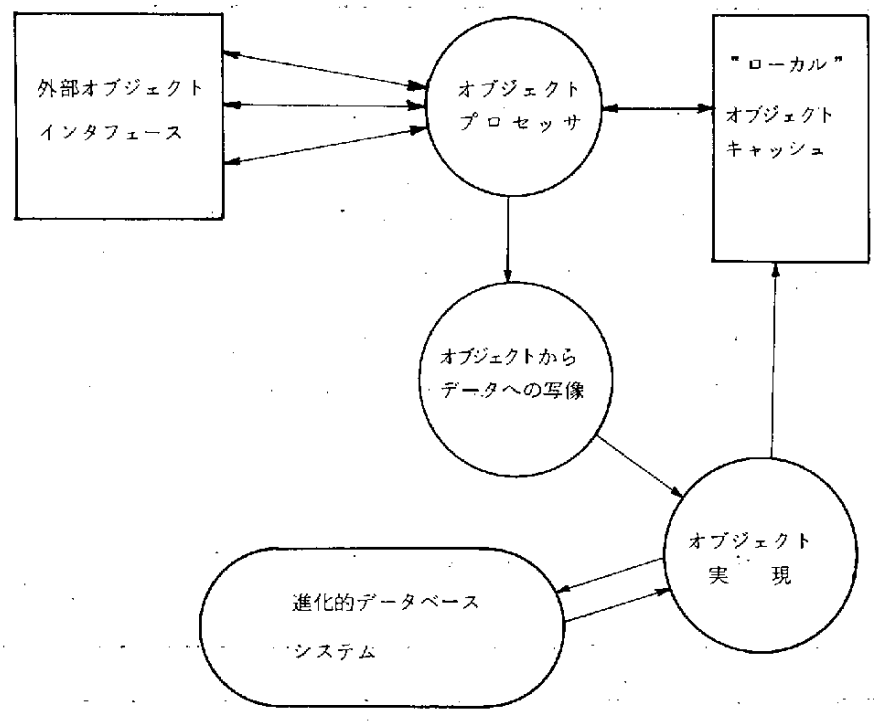


図６－７　テキサス大学の対象指向アーキテクチャの概念図

〔4〕 研究開発内容

(1) 研究開発項目

① 抽象データ型モデルの整理体系化と言語の開発

抽象データ型を幅広い応用分野に適用するには，それぞれの分野の対象（Object）の型や属性が正確に表現できること，個々の対象がユニークに識別できるcapabilityアドレッシングをもつことが必要で，これらをストアする記憶空間（対象空間）は，仮想空間や1レベルストアの拡張であり，情報の多様性からみてより我々人間の空間概念に近づくものであろう。このモデルは，形状概念だけでなく，時間概念や意味概念を含んでおり知

識情報処理のデータベース表現法とも関係する。

抽象データ型の対象をデータだけでなく，手続きや関数に拡張することは，システムの柔軟性に必要である。それにはマイクロ・プログラムによる新命令の記述能力，ハードウェア記述言語による系の再構成能力，関数型言語で使われる関数を対象とする関数形式の能力が抽象データ型言語に要請され，それを対象空間に導入する必要がある。

上記データや関数を包含した対象空間としてＡＣＴＯＲモデルがある。ＡＣＴＯＲとは"手続きや関数などのように計算や操作を行うもの，記憶場所，データ構造などのように情報を格納するためのもの，数や文字列などのようにデータそれ自身に対応するものの４つの異なる範疇に属するものをメッセージを受けることによって能動化される手続的な対象として統一したものである。"これは対象だけでなく対象間の通信を重視するものであり情報ネットワーク，分散処理の研究につながる。

対象空間の研究は，抽象データ型の計算機処理方式の基礎となるものだが，この研究は計算機自身もモデル化することになり，計算機アーキテクチャや人工知能の研究とも深くかかわって来る。

抽象データ型言語は上記対象空間を記述する言語であり，対象の範囲が広がっても使えるためには，問題向き言語のレベルよりも高位のものでメタ言語的性格を持つ。マシン・レベルの記述能力としては，高水準なマイクロプログラムやハードウェア記述言語まで可能なものが望ましい。

② 物理リソースへの写像方式の研究

抽象データ型で定義された対象は，通常の言語のようにコンパイルしてすぐ物理イメージができるものではなく，単に対象の原型が構成されるにすぎない。実際に起動がかかったときに，適当な物理リソースと結合するのが特徴である。従って，物理リソースと対象空間を関係づける写像テーブル（仮想空間の場合はページ・テーブル）が必要である。この構造は，Capabilityチェックを行う部分と，物理リソースをアサインする部分からなるが，この部分のオーバヘッドはシステム性能を大きく左右するので，高速な連想処理技術の研究が必要となる。

③ ガーベッジ・コレクタの開発

抽象データ型の１つの特徴は，対象が生成／消滅を繰返すことであり，これを放置しておくと，すぐにメモリ・リソースが不足し，システムが動作しなくなる。ガーベッジ・コレクションは不用なメモリセルを回収する機能を有するもので，システム性能を上げるに

は，マシンの実行と並列して行う並列ガーベッジ・コレクションを，専用ハードウェア化する研究が必要である。

④　構造メモリの開発

抽象データ型のデータ構造は，単純変数や配列だけでなく，複雑な構造をもっており，1次元アドレッシングでは，処理能力が低下する。構造体メモリは，そのようなデータを自然に写像できる記憶装置で，必要な機能として，

a. 高速大容量化のために多重アクセス可能で階層構成のメモリ装置

b. 並列ガーベッジ・コレクションのサポート機構

c. 対象を識別し，保護するためのタグ付きデータ

d. メモリを効率よく利用するための，スタックやヒープ機構

等がある。

⑤　抽象データ型プロセッサ・アーキテクチャの研究

抽象データ型で使用するプロセッサは，データの多様性のため1種類ですますことは不可能である。例えば数値処理，信号処理，言語処理，画像処理を複合したシステムの場合それぞれ特有のプロセッサを配置する必要がある。しかし，それらを同一のエレメント・プロセッサで構成する方法は可能であり，抽象データ型がもつ柔軟性，拡張性にも合致している。またリピータブルな点で，ＶＬＳＩ向きともいえる。仮にこのエレメントをＵＨＭ(Universal Host Mechine)とすると次のような研究があげられる。

a. ダイナミック・マイクロプログラミングが可能なもの

b. プログラマブルにレジスタやバス幅を拡大／縮小できるもの

c. 系の多重化，再構成が容易なもの

d. 抽象データ型言語を指向したもの

e. タスク切換え，プロテクション，例外処理等のＯＳ機能を有するもの

f. 分散処理に適したもの

それ以外に信頼性，高速性の追求が必要である。

⑥　並列処理方式の研究

抽象データ型言語を並列処理環境下で動作させることは，ＡＣＴＯＲモデルの考え方の実現につながる大きな課題であり，性能の飛躍的向上が期待できる。研究としては，

a. メッセージ授受のプロトコール

b. メッセージ・スイッチの方式

c. 対象空間から並列処理装置への写像

d. データの完全性，機密性

等がある。

⑦ その他の項目

a. 入出力系の方式

抽象データ型における入出力は，ワンレベル・ストアと同様に，特定対象空間（入出力ポート）へのRead/Writeとみなせば，特別視する必要はない。物理的には，メモリ・マップド入出力技術によって行う方法が考えられる。

b. 従来言語との関係

抽象データ型言語は，対象空間での相互干渉を禁止する作用を有するので，従来言語で記述されたソフトウェア資産を閉じた領域に隔離する（Encapsulation）方法で，それらを有効利用することを検討する必要がある。

c. ＯＳの研究

抽象データ型言語はシステム記述言語の能力も有しており，ＯＳのモジュール化に適している。抽象データ型のシステムで重要なＯＳ機能として，物理リソースの管理および障害回復技術等がある。またモジュール化による，多重ＯＳを研究する必要があるが，これらはすべて対象空間のモデルに整合していなければならず，カーネルＯＳは軽くしてUHMにとり入れることを考え，大半のＯＳ機能は，応用システム毎に最適なモジュールを組合せて構成するのがのぞましい。（図６－８参照）

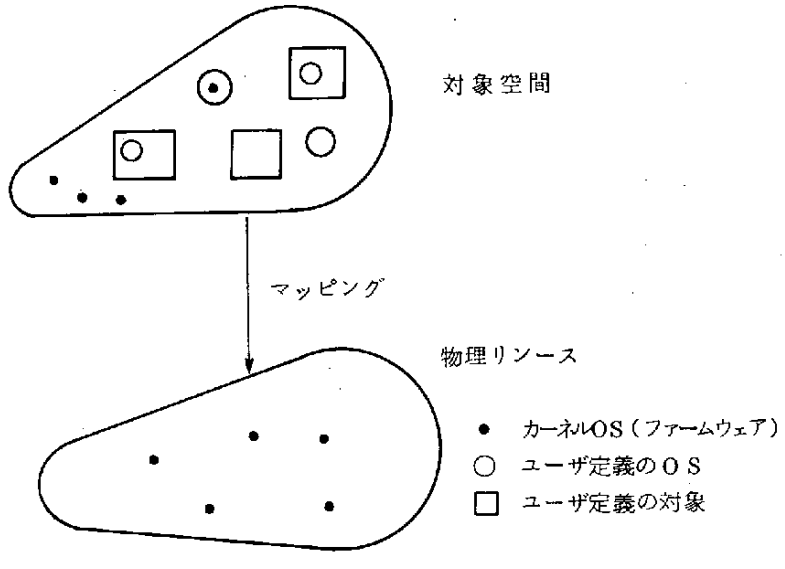


図６－８　抽象データ型におけるＯＳ

d．データベースの研究

抽象データ型でデータベースをとり扱えれば，データベースの管理を非常に容易にしてくれるが，それが対象空間にうまく埋込むことができるか，ローカリティを利用して物理リソースに効率よくマッピングできるか等を解決することが課題である。

大規模システムに抽象データ型マシンを適用するには，抽象データ型の厖大な定義部分をデータベース化し，高速にアクセスする必要があり，データベース・マシン技術の開発が必要となる。

e．分散処理の研究

対象空間を使って分散処理をモデル化し，抽象データ型マシンのコンピュータ・ネットワーク技術を開発する。またデータベース技術とも組合せた分散データベース技術も開発する。

(2) 研究開発スケジュール

抽象データ型マシンの実現形態としては，ファームウェア・ベースの単一計算機で実行するものから，専用化したマシンによる並列処理に至るまで順々に開発して行くのがモデルの難易度からも妥当である。表６－７に開発スケジュールを示す。各段階ともアーキテクチャに先行して対象空間のモデルを検討する。抽象データ型言語とコンパイラの設計もこれに歩調を合せる必要がある。

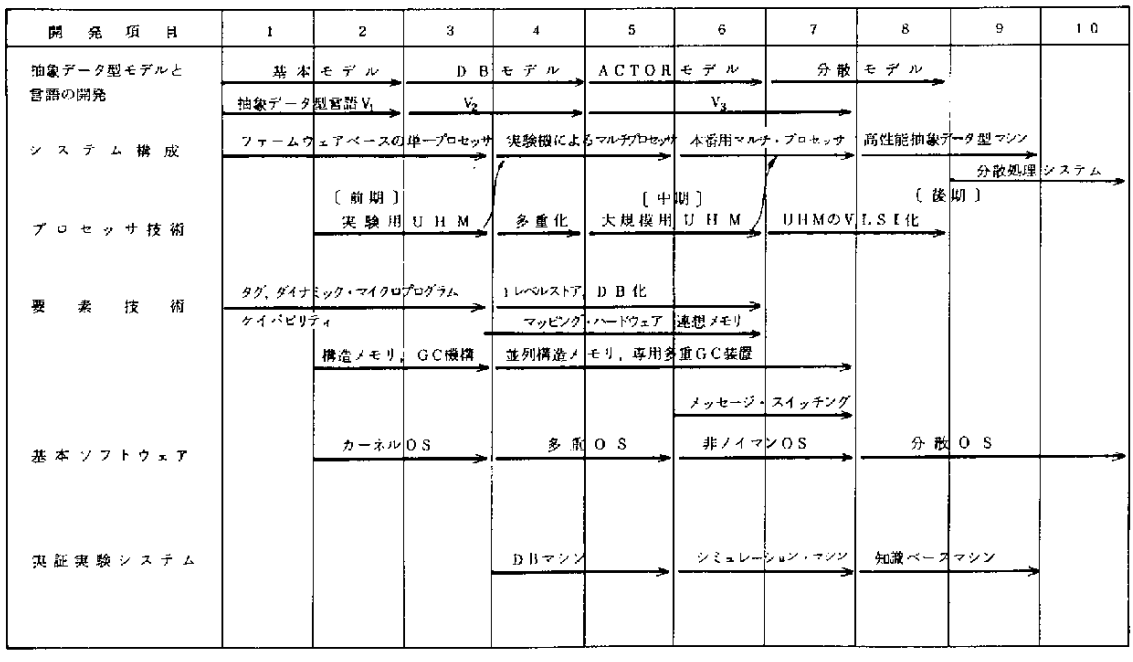
前期はファームウエアによる抽象データ型マシンで，基本的なモデルを設定する。実験用ＵＨＭの設計が中心で，構造メモリ，ガーベッジ・コレクション，capabililyアドレッシングのハードウェア／ファームウェア・サポートを行う。

中期は，プロセッサを複数台にし，構造メモリの多重化を図る。モデルとしては，データベースを組込んだものを検討する。

また中期の後半では，本番用の抽象データ型マシンを複数台使ったシステムの開発で，ＵＨＭの再検討を行う。このモデルは複数のデータ，プロセスが，非同期に通信するため，ＡＣＴＯＲモデルを開発する。物理上は，ネットワークの方式を開発しなければならない。

後期は，実用的な抽象データ型マシンとして，シミュレーション・マシンとかデータベース・マシンを実験するのと並行して，分散処理をモデル化した分散処理用抽象データ型マシンを開発する。

表６－７　抽象データ型マシンの研究開発スケジュール



(3)　研究開発の目標・仕様

①　抽象データ型実験機の製作

②　１００台程度の実験機による抽象データ型マシンの製作

③　１０００台程度の本番用抽象データ型マシンの製作

④　分散処理用抽象データ型マシンの実験

⑤　従来言語によるソフトウェアの継承

⑥　多くの応用分野に適用して，その有効性を確認する（主要な分野として，データベース，シミュレーション，知識ベースへの適用を試みる）。

⑦　ＡＣＴＯＲ理論や関数型言語との融合を図る

⑧　ＶＬＳＩ化した高性能なエレメント・プロセッサを多数結合して，性能価格比の向上を図る

6.2.5　データフロー・マシン

〔1〕　概要

従来の計算機はその基本原理を蓄積プログラムの逐次処理に置いているが，計算機利用技術の発達，応用分野の拡大にともない，幾つかの面に於いて問題点が顕著になって来た。データフロー・マシンはその基本原理を問い直そうとするものである。データフロー・マシンではプログラムの実行制御は各演算に必要なデータの準備状況によってそれぞれ個々独立に行なわれる

為，多くの演算が非同期に次々と併行して処理される。従って並列処理が自然と実行される点に特徴がある。

データフロー・マシンの研究は未だ歴史が浅く実現するに当って解決してゆかねばならないことは数多いが，アルゴリズムに内在する並列性を取出し易い，データフロー・マシン用言語はプログラムの正しさを検証するのに容易である等，非常に優れた性質を備えており，将来の計算機構造に影響を与える有力な方式である。

〔2〕 必要性

従来の計算機は次々と命令を直列に実行してゆく方式を基本としている為，元々並列に実行し得る処理でも直列に実行される。従って処理能力は1つの命令を実行する時間によって定められ，素子の速度が抑えられるとそれで能力に限界が来てしまう。従来迄にパイプライン方式等によって，処理単位を出来る限り小さくすることに注がれある程度の効果をあげて来た。しかし，世の中に於いては計算機の利用が多くなるにつれて増々高い計算機能力が求められており，その要求には際限が無い。現在の計算機構造は既にその素子限界に近ずきつつある。この限界を突破するには新しい原理に基づく素子の開発と並列処理が有力な方策でありそれぞれに努力が傾けられて来た。

プロセッサを何台も並列に並べる形の処理の研究は古く，多くの提案が為されて来たが十分実用性のあるものは出て来ていない。従来の計算機の基本構造は元々並列処理に向いていない為，解くべき問題に内在する並列性の抽出手続きが言語やプログラマに押し付けられて来た。データフロー・マシンはその基本構造が並列処理マシンであり，プログラムに内在する並列性を容易に取出すことが出来る。しかもそれがプログラマに対し殆んど負担を与えない。従ってデータフロー・マシンは並列処理の基礎を与えるものとして大変重要な技術である。

また，今後の計算機システムを構成する素子としてはＶＬＳＩが不可欠となろうが，データフロー・マシンのようにその構造に繰返しパタンが多いものは製作が容易であり適していると言えよう。

一方，現在の情報処理システムに於ける大きな問題はソフトウエアの生産性・蓄積性の向上である。これは今後情報処理が更に普及し発展してゆくに従い増々大きな問題となろう。これを解決してゆく為には，作成したプログラムの正しさを自動的に検証するシステムやプログラムの自動合成システム等の援助が必要である。所が，従来よく用いられ来た言語はこの検証に向いておらず非常な手間のかかることが知られている。一方，関数型言語と呼ばれるものは一般に副作用が少なく，検証が容易であるが，データフロー・マシンにはこの言語が非常によく適合する。従ってデータフロー言語を用いることはプログラムの生産性向上に役立つであろう。

プログラムの蓄積性に関しては論理式を次々と独立に並べたような形の論理型プログラミングが有望であるが，その処理は式の数が増すと一般に大変時間のかかるものとなる。その解決

には並列処理が有望であり，関数型言語と並列処理に向いたデータフロー・マシンは有力な武器となるであろう。

〔3〕 内外技術の現状

(1) 国内技術

国内に於けるデータフロー・マシンの研究はまだ日が浅い。1977年に東大より発表されたＳＡＭＤ計算機（単一アルゴリズム・マルチ・データ・ストリーム）はデータフローと同種の考え方に基づいていたが，データフロー・マシンとして研究が活発になり始めたのはここ１～２年である。東大のＴＯＰＳＴＡＲ，D³C，沖電気のD³P，武蔵野通研のデータ駆動型汎用マシン及び科学技術計算用マシン，電総研の数値計算用マシン，東北大の関数型データフロー・システム等であるが，この内実際に製作され動いているのは東大の２マシンだけである。他のマシンはいずれもこの１年位の間に発表され，現在設計や試作がおこなわれている。

ＴＯＰＳＴＡＲは１号機が1978年に，２号機が1980年に完成しているが，いずれも汎用マイクロ・プロセッサを用いたもので，手続き相互の制御をデータ駆動又は要求駆動により行なっている。２号機は２４台のプロセッサを用い，Ｃ言語とデータフロー記述言語を組合せて使用しており，Ｌｉｓｐものせられ使われている。D³Cは，同じく汎用のマイクロプロセッサを使用したものであるが，データフロー・マシンのシステム冗長性に着目し，適当な機能を付加することにより自律的故障診断機能を持たせた所が特徴であり，1979年に完成している。

D³Pは階層構成のマルチ・プロセッサという形を取り，タスクと機械語の２レベルの並列処理を行うのが特徴で，1981年には試作が完了することになっている。武蔵野通研の汎用システムはプロセシング・モジュール多数を結合網で接続した構成であり，各モジュールに１つの関数の実行を割付けている。科学技術計算用の方は偏微分方程式等を解く為のもので，多くのプロセッサを２次元に配置した形を取る。現在１６台のマイクロプロセッサを用いて試作中である。

電総研のマシンは，プログラム・メモリを内蔵した処理モジュールを複数バス接続した構成で，データフロー用言語としてはＩｄをベースにしたＭＩＤを考えている。同じく電総研では記号処理用データフロー・マシンの為の言語としてＬｉｓｐの拡張も検討されている。

東北大の関数型システムは，言語の関数としての性質を徹底させることを考えており，試作がおこなわれている。

その後，Combinatorのリダクション・システムをデータフロー的に実現する機構の提案（東工大）や，トークンのプロセッサ割当てをハードウェアにより高速に実行するシステムの試作（阪大）等が行なわれている。

(2) 海外技術

データ駆動の理論的基礎は古く1966年頃に出された並列計算のモデルにその芽を見ることが，実際のマシン・アーキテクチャへの拡張を試みたのはＭＩＴのJ.B.Dennis である（1974）。それ以後，米，英，仏に於いて幾つかの研究が見られる。データフロー・マシンの研究は言語の研究が先行しており，ハードウェアの研究自体は遅れている。機械語に相当するものとしては次のようなものがある。

① ＤＤＦ：ＭＩＴ，図式表現，最初のデータフロー語（1974）

② ＤＤＮ：Utah大学，7種類のセルによる図式的表現（1975）

また，コンパイルすることによって上記データフロー図式に相当する機械語に変換される幾つかの高級言語も提案されている。

① ＴＤＦＬ：ＭＩＴ，Algol60 にその基礎を置く

② ＬＡＵ：ＯＮＥＲＡ－ＣＥＲＴ（仏），単一代入言語

③ Id：ＵＣＩ，経緯依存性の取込み，非決定性表現の実現

④ VAL：MIT，厳密なデータ型チェック

⑤ ＣＡＪＯＬＥ：London大学，名前の局所化機械

⑥ ＦＧＬ：Utah大，フローグラフLisp

マシンが実際に製作されたものとしては，仏政府研究機関ＯＮＥＲＡ－ＣＥＲＴのＬＡＵ，Utah大学のＤＤＭ1がある。ＬＡＵマルチ・プロセッサは，基本命令レベルの漸新的な構造を持つシステムでプロトタイプ№0が1980年に完成し，言語ＬＡＵが動いている。ＤＤＭ1はバロース社に於いて1976年初期のものが作られ，現在Utah大学に於いて改良された形のものが動いているが，ＤＤＮを直接実行するマシンで再帰構造を持つトリー型のシステムである。

この他に，英国Manchester大学に於いても単一リング構造のデータフロー・マシンが試作されており，1981年中に完成の予定である。ＭＩＴに於いても，基本命令レベルのマシンを試作中であり，命令を貯えるセル・ブロックとプロセッサ群を結合網で接続してループを形成したシステムで，小規模なものを現在作っている。またカリフォルニア大学Irvine分校でも双方向リングバスを用いたシステムを試作中である。Utah大学では，DDM1 の他に，ＦＧＬ言語を用いるＡＭＰＳというシステムが提案されている。トリー構造の粗結合マルチ・プロセッサであるが，参照の局所性を利用して通信のオーバヘッドを減らしＶＬＳＩに向いた構成としているのが特徴である。

〔4〕 研究開発内容

(1) 研究開発項目

データフロー・マシンの研究開発項目としては，次のような項目があげられる。

① 機械語セットの設計

データ駆動によって制御するアクティビティの単位をどのようなレベルの処理とするべきか，又その時用意すべき基本操作には何が必要かの検討で，そのレベルと達成し得る並列度との関係，マシン・オーバーヘッドの程度等を考慮して定める必要がある。

② データフロー向き高級言語の設計

現在迄に幾つかの言語が提案されているが実際のデータフロー・マシン上にのせられたものはＬＡＵ以外にない。その他のものはシミュレータの上で動かしたもの，又は単なる理論上の提案である。関数型言語はデータフロー・マシンに適しているとは言え，現実のあらゆる問題の記述に適している訳ではない。過去の履歴に依存する問題の記述は，データベースの扱いに必要であるし，資源管理プログラムを記述する為には，どこからその要求が来るか前以って定められない状況を記述（非決定性）出来る必要がある。また，大規模なプログラムを記述し分割コンパイルを利用する為には用いた変数のネームのスコープを明確に定める必要がある。これらの問題に対して幾つかの解が考えられているが，未だその評価は全く為されていない現状である。プログラム検証手法との関連を考慮しつつ十分な検討が必要であろう。

③ マシンの全体構造の決定

データフロー・マシンの全体構造としては，アクティビティ・メモリと処理素子及び結合網でもってループ構成を取るもの，プロセッサとメモリを合わせたものを単位としてそれらをトリー状や網状，又はバスによって結合したもの等多くの形が提案されているが，未だ試作機の数が少ないこともあって，その望ましい形が明確になっていない。発火可能アクティビティの検出ハードウェア・サポート，入出力装置の位置付け，中でも特に２次記憶装置の結合法等は殆んど研究が為されていない状況である。

④ 結合網の構成

データフロー・マシンは多くの処理素子，メモリ素子が並列に動く所に特徴があるが，それをハードウェア上で可能にする主要な要素は結合網の構成である。多くのトークン（データを含む）を同時に適切なアクティビティに渡す部分がそれで，システムのスループットを定める重要部分である。多対多の間で同時に多くの通信が可能な結合網が望まれる。

⑤ 構造メモリの実現

複雑なデータ構造を能率良くプログラムに提供する為には，データ構造の実現をハードウェアによってサポートし，プログラムとデータ構造の物理構造とを分離することが望ましい。データフロー・マシンではメモリ域を意識しないので変数名が多くなる傾向がある為，構造メモリの実現に当ってはメモリ領域の有効利用を図ることが特に大切であり，各

データに対しては参照回数というタグを付けたり，不要データを消して空き領域を確保する為のごみ集めをハードウェアで支援する等の工夫が必要である。

⑥ アクティビティ制御方式の確立

データフロー・マシンでは一般に，必要なデータがそろえば直ぐに実行を始めることが出来るはずであるが，実際上は他に条件がある。即ち，処理結果を渡す相手先のバッファが空いていない限り処理を完結出来ない。このバッファは一般に複数ケ置けるが，その空き状況管理をアクティビティの制御と対応させないとデッドロックを生じてしまう。処理の並列度を抑えずに出来る限り有効な制御方式が必要である。更に，処理の高速化を図る為に，発見的な手法では多くの代案を並列に評価するという手法が取られることがあるが，このような場合，不要となった代案を如何に能率良く止めるかということが問題となる。従って一般に，例外が生じた時にその旨を他の関連するアクティビティへ如何に迅速に伝えるか，不要となったトークンを如何に吸収するかが問題であり，能率良い手法の開発が望まれる。

⑦ ＯＳ機能の実現方式

従来の計算機システムでは多くの利用者に能率良く使わせる為にジョブ管理，資源管理等の管理機能がＯＳとして実現されている。データフロー・マシンに於いてこれらの機能を如何に実現するかの検討が必要である。ジョブの優先度，スケジュール等を如何に実現するか，集中した資源管理を如何に行なうか，これらのことは分散処理システムでは一般にやっかいな問題であり，データフロー・マシンに於いてはハードウ　ア構造にも関係する可能性がある。

⑧ 障害対策・保護対策

計算機のハードウェア又はソフトウェアによる障害対策はシステムを使用する上で重要な問題である。一般にデータフロー・マシンに置いては各アクティビティ毎の独立性が保証されている為，対策は容易になると思われるが，関連するアクティビティをいっせいに止める等難しい問題も存在する。安全性対策も同様で，これらの事柄については全く手が付けられていない現状である。

⑨ 従来計算機との融合構造

データフロー計算機はその構造が従来のマシンと大幅に異なっている為，その実現に当ってはソフトウェアの継続性から従来マシンとの融合構造も検討するべきであろう。従来の計算機のプロセッサ内の１つの特殊演算装置として付加する方法，個々の処理自体は従来の構造をしたマシン上で行ない，それら相互の制御のみをデータ駆動で行なう等幾つかの案が考えられる。データフロー・マシンの今後の研究状況によっては，むしろ，データフローの概念をそのような形で取り込む方がより容易であり実用的であるかも知れな

い。

⑩ データフロー・マシン・プロトタイプの開発

データフロー・マシンの能力を正しく評価する為には，実用的規模のマシンを実際に設計試作することが不可欠である。そうしてこそ真の能力，発展性，ボトルネックが明確になる。勿論これにはプログラミング言語やＯＳの開発も含めて行なう必要がある。実際上はプロトタイプの開発を３段階に分けて行ない，始めには基本操作レベルの実験機を試作して機械語相当のセットを定め，基本的な制御方式を確立する。次に実用的規模のマシンを開発してデータフロー・マシンの特性評価をあらゆる面から実施する。最後に，大規模マシンを試作しその能力限界と可能性を調べるというステップが考えられる。

⑪ パーソナル・データ・フロー・マシンの開発

上記第２，３ステップと並行して，パーソナル・コンピュータをデータフロー・マシンによって実現することの研究も行なわれるべきであろう。将来のマン・マシン・インタフェースは知識情報を利用した高度なものとなることが考えられるが，その鍵を握るのは論理型プログラミング等を能率良く高速に実行するパーソナル・コンピュータである。単一ユーザ用のＯＳとして単純な制御プログラムの制御の下，その特定個人用の知識を活用して，複雑な上位のシステムを単純で理解し易い形に仮想化する機能が必要で，論理型プログラミングに適合したデータフロー・マシンの活躍分野となる可能性がある。

⑫ データベース管理機能との融合

今後の計算機はデータベース管理機能が不可欠であるが，データフロー・マシンに於いてそれとどのような形でインタフェースを取るか，又逆にその機能自体の実現にデータフローの考え方がどのように役立つのかという問題は重要なテーマである。一部提案も為されているが，本格的な検討は今後の課題である。

(2) 研究開発スケジュール

研究開発を３段階に分けてほぼ次のような形で実施する。

前期：

データフロー用高級言語の諸検討からデータフロー・マシンの機械語レベルの基本操作セットを決定する。アクティビティ制御方式を確立し，プロセッサ内部構造を定め，プロセッサ間相互結合網や構造メモリの検討を進めてシステムの全体構造を明らかにして，基本操作レベルのデータフロー実験機を試作する。

中期：

プロセッサ間結合網と構造メモリの開発をおこない，実用規模のデータフロー・マシン・プロトタイプを試作する。これにはＶＬＳＩを用いる。幾つかのデータフロー用高級言語を開発してプロトタイプ・マシン上に実装し，様々な応用に適用して汎用化の評価を行

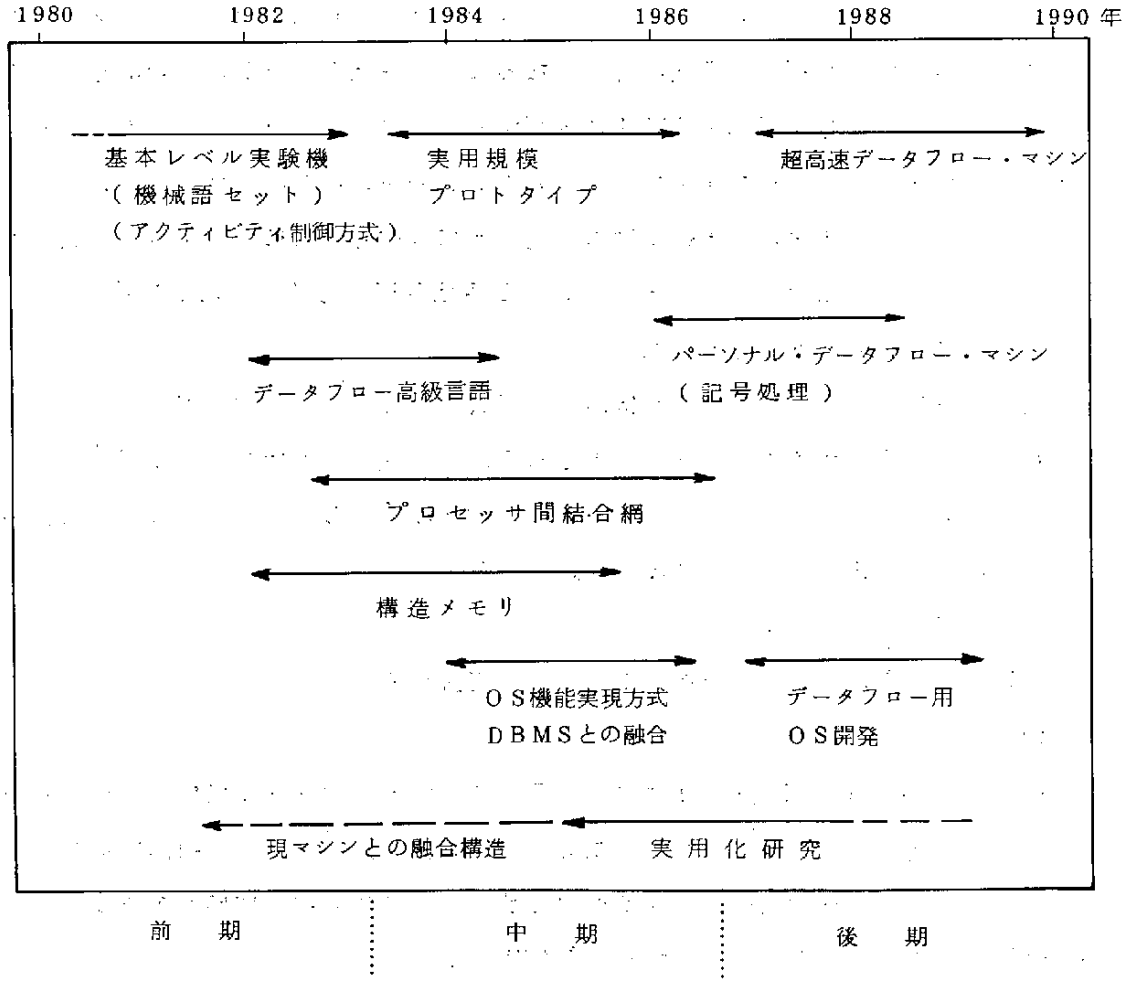
なう。更に，ＯＳ諸機能の方式を確立し，データベース管理機能との融合を実現する。

後期：

データフロー・マシン用ＯＳを開発してプロトタイプ・マシン上に実装する。大規模な処理（知識情報処理）を主目的とした超高速データフロー・マシンを開発する。これには超高速素子を用い，ＶＬＳＩによって実装する。更に，記号処理等の非数値処理を主目的とするパーソナル・データフロー・マシンを開発し知識情報処理のフロント・エンドにする。同時に，多くの実用化の為の研究を進めるが，従来のマシンとの融合化等の試みは前期の頃から連続して進める。

以上のようなスケジュールを年代との関係で示せば，ほぼ次の表６－８のようになろう。

表６－８　データフロー・マシンの研究開発スケジュール



(3)　研究開発の目標・仕様

基本操作レベル実験機：

プロセッサ　　１６台

メモリ容量　　１６ＭＢ

実用規模プロトタイプ：

プロセッサ　　１００台

メモリ容量　　２００ＭＢ

性　能　　１００ MOPS（Millions Operations Per Second ）

ＶＬＳＩ構造（初期）

超高速データフロー・マシン：

プロセッサ　　10^3 ～ 10^4 台

メモリ容量　　1ＧＢ～１０ＢＧ

性　能　　１ＢＯＰＳ～１０ＢＯＰＳ（Billions OPS）

ＶＬＳＩ構造，超高速素子

パーソナル・データフロー・マシン

プロセッサ　　３２台

メモリ容量　　１０ＭＢ

性　能　　１０ MOPS

ＶＬＳＩ構造

プロセッサ間結合網：

ＬＳＩ向き構造（ＴＴＬ）

10^3 ～ 10^4　台収容

構造メモリ：

容　量　　１ＭＢ～１０ＧＢ

6.2.6　新ノイマン・マシン

〔1〕 概要

(1) 第５世代プロジェクトでの位置付け

第５世代コンピュータ・プロジェクトにおける新（New）ノイマン・マシン開発計画は，大別して二つに分けられる。一つは１００万トランジスタ／チップ（１チップで１００万トランジスタを含むデバイス）のＶＬＳＩベース・コンピュータの開発であり，もう一つは１０００万トランジスタ／チップのコンピュータの開発である。第５世代コンピュータ・プロジェクトにおける位置付けを示すならば，１００万トランジスタ／チップ版（以下，ＭＡＲＫ Ⅰという。）は第５世代コンピュータ開発サポート・システムの中心的コンピュータ並びに第５世代コンピュータへのつなぎの役割を果すブリッジ・コンピュータとしてとらえることができ，また１０００万トランジスタ／チップ版（以下，MARK Ⅱという。）は第５世代コン

ピュータの上に作られた知的ＣＡＤシステムを使い作られる応用目的指向適応型コンピュータと考えられる。第５世代コンピュータ完成の主要な応用の一つに知的ＣＡＤがあるが，これを使い１０００万トランジスタ版をブートストラップし，第５世代コンピュータそのものも１０００万トランジスタ／チップＶＬＳＩを使ったものにしようとするわけである。（図６－９参照）

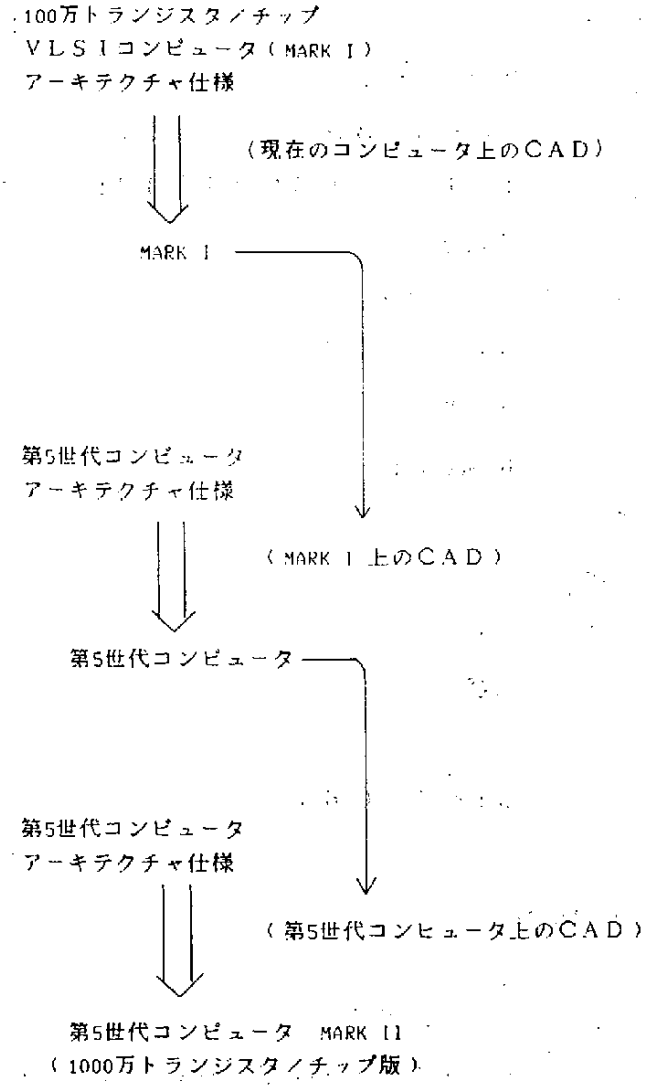


図６－９　第５世代プロジェクトにおけるNewノイマンマシンの開発過程

(2)　現在のマシン（ノイマン型アーキテクチャに基くもの）の欠点

第５世代開発サポートシステムは研究者・開発者にそれぞれ専用のパーソナル・コンピュータを持たせ，それをローカル・ネットワーク技術で接続したような，いわゆる分散型コンピュータ・システムのイメージである。システムの中核となるパーソナル・コンピュータに要求される機能は，

① 高度マンマシン・インタフェース・サポート

（1000×1000ドット程度の高密度画面をサポートする高級エディタ，アブストラクト・データタイプ・サポート言語の装備等）

② 十分な速度

（現在の大型コンピュータを一人で使用する程度の速さ）

③ ローカル・コンピュータ・ネットワーク・サポート

などであり，これが低価格で提供されなければならない。アーキテクチャ的にも現在の汎用機はパーソナル・ユースということを意識して設計されているわけではないので問題は多いが，それよりも低価格で提供ということが現在と同じ製法・素材で作っていたのでは到底この期待に沿うことはできない。よって，MARK Ⅰ,MARK Ⅱともコンピュータ構成素材に革新性がないかぎりこの目標を達成することはできないわけである。第5世代プロジェクトではこの革新的な素材にＶＬＳＩを使うことによってのりきる。現在のコンピュータは言うまでもなくＶＬＳＩを使って作られているわけではないため，すべてのアーキテクチャをＶＬＳＩベースで今一度考えなおさなければならないと思われる。「Newノイマンの Newはノイマンをベースとしつつも従来の研究より知れたノイマン型の欠点をベースをくずさないで改良しているということを表わしている」が，このような半導体技術が未熟なため現在のノイマン型アーキ型アーキテクチャはかなりデフォルメされている。（表6－9参照）

そして，この影響は実行効率の向上をもたらしたがプログラムの開発効率を著しく落している。ＶＬＳＩ1チップ化によるNewノイマンマシンMARK Ⅰ,MARK Ⅱ共にこれらの欠点をすべてなくすということが開発における最大の目的となる。

(3) 望まれる機能

望まれる機能とは以下の通りである。

① ソフトウェア開発コストの低減のための工夫

前述したように現在の多くのノイマン型マシンはインプリメントする場合の素子の貧弱さから，かなり悪い方向に（デフォルメ）された方式をとっている場合が多い。具体的には従来，マシン構築にあたりまず考えたことは実行効率を上げることであり，プログラム開発効率を上げることにはほとんど考慮が払われていない。従ってNewノイマンではこれらを改善することをまず考える。またソフトウェアを開発しやすくするための機能としては，プログラムを書くフェーズで書きやすくするための機能と，書いてしまったものを動くようにする（デバック）ための機能，また，でき上がったソフトウェアを管理しやすくするための機能の三つが考えられる。

表６－９　G.J.マイヤーズによるノイマン型コンピュータのゆがみ

① 逐次型メモリ

プログラムとデータは一つの順序づけられたメモリに貯えられる。順序づけられた記憶の概念は，高級言語において名前のついた変数として離散的な集合から成る記憶の概念とはまったく類似していない。

② 線形メモリ

メモリは一次元であるが，高級言語で扱うデータには線形とは限らない多次元のデータ（配列，構造体，リストなど）がある。

③ 命令とデータの区別がない。

命令をデータのように扱ったり，データのビット列を命令として実行できるのに対して，高級言語ではデータを実行したり，命令をデータのように参照する概念はない。

④ 意味はデータに備わったものではない。

データの意味はプログラムの論理，命令によって与えられる。従って，例えば文字列やアドレスデータに対して浮動小数点の演算を行うことができる。高級言語においては，データの意味はデータそのものに結び付けられている。

⑤ ２進演算

人間とのやりとりのため１０進数と２進数の変換にシステムはかなりの時間を費やしている。また，ほとんどの１０進有理数は計算機内部で２進数の近似値として表現されるため，プログラミングのむずかしさ，エラー，言語の定義を混乱させる原因となっている。

⑥ 固定語長固定語長であるために表わしうる数の最大値と最小値が決まっている。高級言語の持つ可変語長の概念を固定長の語に写像するのが難しい。

⑦ レジスタ

番地づけ可能なレジスタが存在することは，高級言語の概念と合わない。すべての算術演算に必ずレジスタを使うことや少ないレジスタの最適割り当てのためにコンパイラが生成するコードはソース・プログラムの論理を表現するうえでは何の貢献もしていない。

② 高機能化

問題に適応させて命令セットを変更していくというような，主としてセマンティック・ギャップを減らす方向である。本来のノイマン・マシンにはこのアイデアは無かった。

③ 高信頼化

高信頼化は，論理的なレベルと物理的なレベルの二つに分けて考えられる。論理的なレベルでは，いわゆる不注意によるプログラム・ミスを防止するケーパビリティ・アドレシング・メカニズムの実現などである。他方，物理的なレベルでは自己故障診断機能，または，耐故障（設計）機能の実現ということになる。

④ コンパティビリティ

Newノイマンはもともとevolutionalな性質の方が大きいため，従来のマシンとのコンパティビリティがとりやすい。具体的にはエミュレーション機能が重要であるが，この場合単なる機械語だけでなく，オペレーティング・システムのエミュレーションをも含む広い意味でのエミュレーション能力が要求される。

又，本機能の実現のための具体的な処理方式は以下の通りである。

① ダイナミック（アーキテクチャ）方式

② スタック方式

③ タグ方式

④ ディスクリプタ方式，モジュール・オブジェクト方式

⑤ アドレス管理機構

⑥ マルチNewノイマン機構

⑦ ノイマン型高速演算機構

⑧ プロテクション・メカニズム

⑨ Fault Tolerant Computing機構

〔2〕 必要性

ここでは，Newノイマン・アーキテクチャの必要性をMARK Ⅰ，MARK Ⅱの二つに分けて考察する。

(1) 共通項目

MARK Ⅰ，MARK Ⅱともに共通して必要とされる項目は，

① ソフトウェアの生産性向上

② 構成素子の進歩による高機能化

③ 構成素子の進歩による高信頼化

④ コンパティビリティの確保

である。

ここで，特に④に関してコメントをつけるならば，第5世代コンピュータ本体はそれ自身は現在のコンピュータと比べるとかなり革新的であり，コンパティビリティはとりにくいのに比べ，Newノイマン型は同じく第5世代コンピュータではあるが，ノイマン型をベースとしているため，現在のマシンとコンパティビリティをとることが可能である。そして，Newノイマンは第5世代コンピュータでは分散ノードの一つとして位置付けられる。

(2) MARK Ⅰ

MARK Ⅰは,第5世代機開発サポート・システムのノード・コンピュータであるスーパー・パーソナル・コンピュータの核として必要である。

(3) MARK Ⅱ

MARK ⅡはMARK Ⅰの改良，延長型という，より高機能な第５世代コンピュータ実現の１９９０年に手に入る素子を最大に生かしたコンピュータということの他，第５世代コンピュータ・システムの上に作られる知的ＣＡＤの実用性を確かめるデモンストレーションとして必要である。

〔3〕 内外技術の現状

内外技術の現状を考える場合，Newノイマン型アーキテクチャの基本方式の研究は従来も行われてきた。しかし，システムとしてまとまったものはまだ存在しない。その理由は構成素子のＶＬＳＩ １００万トランジスタ/チップ作成能力が現在ないからである。しかし，いくつかの芽にあたる研究開発が公表されている。１９８１年２月現在，実現されているものは，４５万トランジスタ/チップのものであり，これは米国，Bell研究所，HP社によるものである。（詳細はＶＬＳＩアーキテクチャの項）

また，ＶＬＳＩベースではないがNewノイマン機構を積極的に取り入れているものとして，ＩＢＭ社のシステム３８，インテル社のＩＡＰＸ４３２プロセッサＡＤＡマシンなどがある。我国では公表されているものはない。

さらに，ＭＩＴのＡＩラボにおける知的ＣＡＤの研究はMARKⅡ開発研究の芽と考えることができる。なお，我国ではこの種の研究も行われていない。

〔4〕 開発内容

(1) 概要

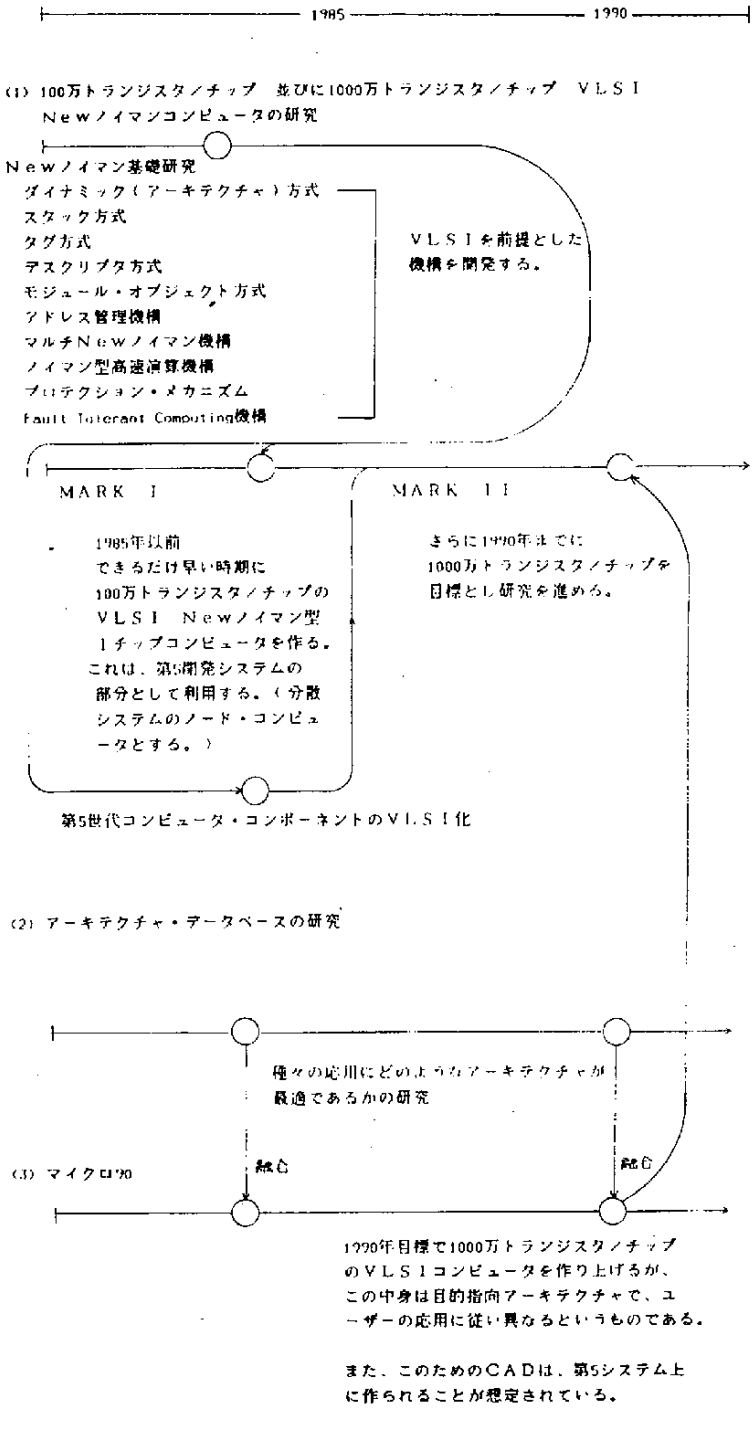
MARK Ⅰは第５世代開発サポート・システムのノード・コンピュータ・システムであるスーパー・パーソナル・コンピュータの中身として考える。Newノイマンをシステムとし，ＶＬＳＩ１チップでどう作るかだけに焦点を絞る。従って，シングル・ユーザ用を意識すればよく，従来マルチ・ユーザを前提として作られていた部分はこの際取り入れる必要はない。ＵＨＰを主体とし，〔1〕で述べた要求される機能が１チップにおさまるようにし，分散システムの中でのプロテクションを考え，巨大論理アドレス・アクセスが可というようなものを作る。MARK ⅠとMARK Ⅱは分けて考えた方が良く，MARK Ⅰを１９８５年以前までに，MARK Ⅱを１９９０年の目標として，プロジェクトをスタートさせる。

研究開発の進め方としては，まず〔1〕(3)で述べた必要な機能のＶＬＳＩ化を前提とした研究が必要である。その後，MARK Ⅰへと進める。又，Newノイマン機構が第５世代コンピュータのコンポーネントなることを研究するプロジェクトも必要である。

また，もう一つアーキテクチャ・データベース研究グループにより，このようなNewノイマンＶＬＳＩチップをどのように応用問題に適応させるか（主としてNewノイマンのファームウェアとソフトウェア）の研究を行う。さらに，それぞれの計算速度はそれほどではない

が，個々の仕事に合った計算機をそれぞれがどう持つかという，建築で言えばハウス５５に相当するようなきたるべき世の国民一人一人が皆コンピュータを持つような時代に備える計画も考えられる（これをマイクロ９０という）。研究の中心は，ＶＬＳＩ－ＣＡＤであるが，これにより目的にあったＶＬＳＩチップをユーザーが作ることができるようになる。マイクロ９０のためには第５世代コンピュータは必要不可欠であり，第５世代の威力を示すための計画ともいえよう。

(2) 研究開発項目とスケジュール



(3) 研究開発の目標・仕様

① MARK Ⅰ

- Newノイマンベース１００万トランジスタ/チップ プロセッサ
- マイクロプログラム・サポート・ツール
- シングル・ユーザ用オペレーティング・システム
- アブストラクト・データタイプ・サポート・システム記述言語
- ローカル・ネットワーク・サポート・オペレーティング・システム

② アーキテクチャ・データベース

- Newノイマシン・アーキテクチャ要素の分類並びにデータベースの作成
- 応用分類並びにデータベースの作成
- 応用別評価機構

③ マイクロ９０

主要目標はＶＬＳＩ ＣＡＤの項参照。

- 知的ＣＡＤとアーキテクチャ・データベースの結合
- 応用別目的指向アーキテクチャ選択システム

④ MARK Ⅱ

- Newノイマンベース１０００万トランジスタ/チップ プロセッサ
- 適応型オペレーティング・システム
- 応用指向言語

6.2.7　機能分散アーキテクチャ

〔1〕 概要

機能分散アーキテクチャは計算機システム全体の構造を定めるアーキテクチャであって非常に広い概念である。機能分散アーキテクチャにもとづくシステムを構築するためには，各コンポーネント・マシンや結合アーキテクチャを開発しなければならないが，ここでは，主にシステム構築のためのアーキテクチャという側面に重点を置いて一般論的に議論することにする。

(1) 機能分散アーキテクチャの特徴

機能分散アーキテクチャの意味は，一言でいえば，“計算機システムのあらゆる部分，あらゆるレベルにおいて，機能の分散を可能な限り行なってシステムを構築するアーキテクチャ”ということである。このような機能分散形式を採用する理由は，現在，または今後10年程度に予想されるハードウェア技術により，同じく予想される応用側の要請を満たす計算機システムを最も効果的に実現できると考えられるアーキテクチャだからである。

より具体的にいえば，このアーキテクチャの持つ次のような特徴が，望ましいものだからである。

① スループットまたは実行速度の向上

複数の処理要素を同時に動作させる並列処理によってシステムのスループットを拡大したり，同一問題を処理する実行時間を減少させるという効果である。

② システムの成長性，即ち構成の自由度の確保

情報処理システムが高度化するにしたがい，システムの連続的な成長性が強く求められるようになる。また，応用環境の多様化が進んでいるので，応用側の要請に合せた形にシステムを構築するための自由度の確保も重要である。このような目的のためには，モジュール構成の機能分散アーキテクチャが集中型単一システムに比し明らかに優れている。

③ 高信頼性高稼動率の達成

機能分散アーキテクチャでは，一部の構成装置の故障においても，全システムを停止させることなく，フェイルソフト系を容易に実現することが可能である。

④ 機能の専用化による処理効率の向上

機能単位に分割された各単位装置をその機能専用に設計することによって処理効率の向上が期待できる。

⑤ システム構築の容易化

機能分散アーキテクチャにおける各構成単位の設計は，その機能とシステム内の他のモジュールに対するインタフェースが明確に定められ（論理的にも，物理的にも）ていれば，他とほぼ独立した設計を行なうことができ，複雑なシステムをひとまとめに設計するよりはるかに容易である。

この特長や②の特長は直接ユーザから見た計算機の性能に反映されるものではないが，計算機構築の立場からは非常に価値あるものである。

機能分散アーキテクチャにはこの他にも優れた特長をあげることができるが，それらはある特定の部分の機能分散に適用できるものであったり，上記の特長から派生的に生じるもので，基本的には上記の４つの特性に集約することができる。もちろん，こうした特性を実現するには，バックボーンとなるよい理論モデルの考案，各種の実験による設計データの収集等多くの努力が必要である。安易な設計では，機能分散の最大の弱点とされる(時間的，また，物理的等）オーバヘッドによって，使いものにならないシステムができ上ってしまう可能性も少なくないことを銘記すべきである。

(2) 機能分散のレベル

機能分散の考え方は計算機システムのいろいろな部分に適用することができるが，基本的には次の３つレベルに分けることができ，それぞれ主たるねらいや考え方が若干異なっている。

① ユーザ・サイドと中央システム間，あるいは複数の中央システム間の機能分散

ユーザの手元にあるパーソナル・コンピュータあるいは工業プロセスを管理するプロセス・コンピュータや各部門のデータ処理を行なうコンピュータ等と，中央にある強力なコンピュータ・システムとの間の機能分散である。また，複数の情報システム間の結合もこの中に含めて考えることができる。この部分の機能分散についてはネットワーク・アーキテクチャとして別の独立課題として取り上げられているか，特徴は次の通りである。

- 第５世代アーキテクチャではユーザ・サイドのパーソナル・コンピュータ等のローカルな処理機能が高度化し，通常の処理は全てまかなわれる。情報についてもローカル性の強いものや使用頻度の高いものは極力ローカル・サイトに配置される。
- ローカル・サイトの代表的装置はパーソナル・コンピュータで，これは単に処理機能とユーザ個有の情報管理を行なうだけでなく，ユーザとシステムのインタフェース装置として，システム全体を仮想化してユーザに使い易く提供する役割も持つ。このパーソナル・コンピュータは用途がパーソナルであっても現在同じ名称で呼ばれているものとは比較できないほど高度の機能を持つ機能分散アーキテクチャの重要な構成要素であり，新しい思想のもとに開発を行なわねばならないものである。
- 一方，中央サイトの共同利用される高機能コンピュータ・システムは，ローカルサイトでは機能不足な超高速処理や特殊処理機能の提供と共通情報の管理を主な役目とする。

② センタ高機能コンピュータにおける機能分散

センタの高機能コンピュータは，個々の機能を実行する専用マシンの集合体として構成

される。即ち，機能はマシンを単位に分散され，全体として汎用コンピュータとして動作する。マシンは　ユーザ・サービス・マシン（ＵＳＭ），マシン・サービス・マシン（ＭＳＭ），コントロール・マシン（ＣＴＭ）の３種に分けられる。この内ＵＳＭはパーソナル・コンピュータでは不十分な超高速処理や特殊処理・共通データの管理等の作業を行なうマシン群であって，データーベース・マシン，超高速科学技術計等マシン，各種高級言語マシン等が含まれる。ＭＳＭは遠隔地にあるシステムやパーソナル・コンピュータの制御を行なうマシン群であり，ネットワーク・アーキテクチャの課題中に詳しく論じられている。またＣＴＭは特殊装置やＵＳＭ，ＭＳＭ群を制御し，協調して動作させるためのものである。

このレベルの機能分散アーキテクチャにおいて重要な点をまとめると，個々の超高性能マシンの開発と，それらを統合して動作させるための制御方式の開発，そのバックボーンとなるすぐれた論理モデルの設定等である。

③　コンポーネント・マシン内の機能分散

ＵＳＭ，ＭＳＭ等の各マシンはそれ自身も，更に機能分散されたマルチ・プロセッサ構造をとって，先に述べた機能分散の特性が生かされるようにすべきである。この場合，前の２つと異なる点は個々のマシンの特殊性に応じた機能分散構成をとって並列処理効果を最大限に高めねばならないことである。したがって，高度並列処理技術，マルチ・プロセッサ・マシンの構成技術等の研究開発が重要である。

(3)　システムの構成

機能分散アーキテクチャを採用することにしたとしても，その具体的なシステム構成は，基本となる設計理念によって大きく異なる。この設計理念の確立はそれ自身一つの大きな研究課題である。そこでここでは非常にマクロに表わしたシステム構成の例を図６－１０～１４にあげておくことにとどめる。

これらを比較するとき多階層の機能分散が行なわれ，それぞれのレベルで，しっかりした論理モデルにもとづいた接続，通信方式が定められているという基本的考え方の一致と具体化における多様性を読みとれる。

〔2〕 必要性

第５世代のコンピュータ・システムは，単に，特定の応用を処理できるだけでなく，トータル・システムとして効率的で使いやすいものでなければならない。そのためには，個々の応用の問題解決技法を改善するだけでなく，システムが，絶えず進歩し続けるハードウェア(素子)技術を最大限に利用して，これまた進歩し続ける要求される問題の解決を最小コストで実現できるように構築されていなければならない。

このような要請を達成するために最も有効と考えられているシステム・アーキテクチャが機

能分散アーキテクチャである。その理由は前節に述べた特徴から明らかであろう。新しいシステムはそのような方向に着実に流れている。

このようなシステムを構築するためには，全体的に深く検討された統一的設計理念にもとづき，現実の要請とハードウェア，ソフトウェア及び応用技術の状況にあわせてシステムを設計しなければならない。機能分散アーキテクチャの実現には多大の知恵と時間と費用が不可欠である。

機能分散の基本アイデアは決して新しいものではなく，ある種の新しい応用のような華やかさはないが，この際ハードウェア，ソフトウェアのこれまで蓄積された全技術を集約して，一貫した設計理念のもとに高効率，高信頼で，使い易く，構築しやすく将来の技術改良に対応しやすく，高機能を盛込むことのできる機能分散アーキテクチャの開発に努力がなされることが必要である。

〔3〕 内外技術の現状

機能分散アーキテクチャは従来からあった考え方を発展させ，実用に耐え，効率的なものとしようとするところに開発のねらいがある。

機能分散の考え方は，古くからあり，部分的な研究は多数行なわれている，統一された論理モデルにもとづき，確立した設計理念によって構築されたシステムの開発は，未だ全くといってよいほど行なわれていないということができる。

(1) 国内技術

海外，特に米国に比べるとかなり遅れているのが現状である。

ネットワーク・レベル，センタ・コンピュータ内，コンポーネント・マシンのそれぞれの分野における機能分散アーキテクチャについて，個別に，電子技術総合研究所，電気通信研究所，及び，東京大学，慶応大学等いくつかの大学において小規模の実験レベルの研究が行なわれ，ある程度の成果と基礎技術が得られている。しかし，トータル・システムとしての機能分散アーキテクチャの研究は殆んど行なわれておらず，基本的設計理念の確立を初めとして，研究開発しなければならない項目は多い。

(2) 海外技術

ネットワーク・レベルではＡＲＰＡ網を初めとし，多く実例がある。センタ・コンピュータにおける機能分散アーキテクチャについても古くからいろいろな研究があるが，本格的に機能分散アーキテクチャを追求したものとしてはＢＣＣ－５００（１９７０～７３）が有名である。また，最近でもカーネギーメロン大学等を中心に各種の研究が進められている。論理モデルについてもいくつかの提案がなされており，（ＯＳＩ標準化モデル，Actorモデル）また，システム制御方式として分散型オペレーティングシステムについても同様に大学等で研究が進められているし，分散データベース，各コンポーネントマシンのアーキテ

クチャについての研究も少くない。

しかし，システム全体を機能分散アーキテクチャの基本思想のもとに統一的に整理して開発を試みているものはまだない。

[4] 研究開発内容

(1) 研究開発項目

① 機能分散の基本設計理念の確立

機能分散アーキテクチャにおける機能分散の方法，機能モジュール間の通信形態を統一的に表現する論理モデルの研究開発，およびそれにもとづく，効率的機能分割法，マシン間論理インタフェース，物理インタフェース，情報交換方式，情報の分散配置方式等の開発，機能分散システムの記述方法とその処理系，及びシミュレーション等の評価システムの開発。

② システム制御方式（オペレーティング・システム）

機能分散アーキテクチャによるシステムの制御方式は未知の分野である。統一的な思想にもとづくよい方法を開発する必要がある。特に，ＯＳは従来のものが集中処理向きに設計され，一つの大きな計算機能や記憶機能を多数のユーザで効率よく共用することが主眼となっていたのに対し，機能分散アーキテクチャではその基本コンセプトが異なるためにその基本概念からして大きな変革を求められる。したがって多くの研究開発努力を注入する必要があるが，具体的には次のような事項についての研究が必要と思われる。

ＯＳの分散方式，ＯＳモジュール間通信方式（インタフェース），ＯＳモジュール間の情報交換における障害対策，モジュール間同期方式，単純で実用的なデッドロック回避方式，機密保護方式（暗号化，ケーパビリティアドレシング），ジョブ管理，タスク管理等の従来のＯＳ機能の機能分散ＯＳ上への展開，ファームウェア化，ＶＬＳＩ利用のハードウェア化，デバッグサポート，ユーザ教育，ネットワーク・アーキテクチャと統合して設計されたローカル・ＯＳ（特にパーソナル・コンピュータ上のもの），ＯＳ用コントロール・マシン（ＯＳニュークリアス・マシン）の構成，モニタとその結果による自動チューニング方式，ＯＳの開発サポート技術等。

③ 分散型データベース

計算機システムの主要な機能は，情報の処理と情報の蓄積利用である。一般的にいって処理機能は必要なものが直ちに得られれば，どのような形で存在してもユーザにとっての違いはあまりない。しかし，情報の利用のほうは，蓄積された情報量が多く，共同利用できる方がより有効であるから，本質的に集中化の傾向がある。したがって，データベースは機能分散の考え方にそぐわないとも見られるが，事実は，本来ローカルに存在するデータベースは結合してシステム内の全ユーザに蓄積された情報を有効に利用させるのである，と

逆に考えられるべきで，分散データベース構築方式は極めて重要な研究課題の一つである。

具体的には，異なるスキーマの変換とその処理方式，非数値処理機能，高速データ転送，信頼性の高い放送通信機構，データベース間の制御コマンドの迅速な授受方式，データ・セキュリティ（暗号化方式），高速大容量補助記憶（10^3 B幅～1GB），知的アクセス等の技術の開発が必要である。

④ コンポーネント・マシンのアーキテクチャ

各機能の処理を担当するコンポーネント・マシンも現存のものに比しはるかに高性能のものを開発する必要がある。6個の新アーキテクチャの課題はその構築技術としてぜひ研究しなければならないものである。個々のマシンでは計算という作業のために最高性能を出すことを目標とする高速演算マシンと情報記憶処理の中心となるデータベース・マシンは別の独立した課題としてあげられているし，マシンサービスマシンはネットワークアーキテクチャの課題に含まれている。

この他ではユーザと直接インタフェースする高性能パーソナル・コンピュータの開発とセンタ・コンピュータ内の高性能高級言語マシンのアーキテクチャが非常に重要である。具体的にいうと前者に関しては基本となるVLSI化ユニバーサル・ホスト・プロセッサとその上でのファームウェア高級言語マシンのアーキラクチャを初めとして，OS，学習機構，図形，音声，文字等の処理技術とそのための入出力装置，大容量小型パーソナル・ファイル，単純で使い易く，図形も処理できる新しい高級言語等の研究開発を行なう必要がある。後者については，中間言語の設定方式，複数言語処理アーキテクチャ，並列処理アーキテクチャ，高度デバッグ機能，適応機能，言語仕様の差の吸収方式等の研究開発が重要である。

⑤ 自動適応化方式（ダイナミック・アーキテクチャ）

機能分散アーキテクチャでは，システムの柔軟性を生かし，各レベルで与えられた要求や環境にシステム側が自動的に適応できる機能を組込むことが望ましい。そのためには，自動化方式の理論的背景の確立，命令セット，並列構造，情報配置，資源割当，マシン結合構成，信頼性と処理能力等の自動適応化方式，学習機構，動作データの収集，ダイナミック・マイクロ・プログラミング技術等の新規開発，あるいは現存技術の改良の研究が必要である。

⑥ インプリメンテーション

実際に機能分散アーキテクチャによるシステムを構築するには，実装．高速通信の可能な接続方式，VLSI化等のインプリメンテーション技術も機能分散アーキテクチャの特徴を生かせるように改良する必要がある。

⑦ その他

以上の他，一般的な基礎技術として，次のような技術の高度化のための努力が必要である。

機密保護方式，記憶制御方式，信頼性（可用性）向上方式等。

(2) 研究開発スケジュール

本課題の研究開発スケジュールを表６－１０に示す。

(3) 研究開発の目標・仕様

前項で示したスケジュールにあるように，機能分散アーキテクチャ・システムの開発は次の３つのフェーズに分けて行なうのが適当である。

① 機能分散基礎システム

機能分散アーキテクチャは，システム・アーキテクチャであるから，各種の基礎技術をうまく統合することが必要である。よって，このフェーズでは，いくつかの部分的な小規模実験システムの構築，理論的検討，シミュレーション等によって次のフェーズ以後に利用するための基礎技術を確立する。具体的には次のようなサブテーマについて研究する。

a. 機能分散のバックボーンとなる論理モデルの確立

b. いくつかのアプローチについて小規模実験システムを構築し，コンポーネント間通信方式，コンポーネントマシンの方式，並列処理方式等の基礎アーキテクチャを確立する。

c. 分散ＯＳを中心としたシステム制御方式の基礎技術，機能分散システム記述言語，その処理系の開発。

d. 機能分散の各レベルにおける並列処理構造，命令セット，信頼度等の環境に対する動的適応（ダイナミック・アーキテクチャ）方式の開発。

e. 物理インタフェース，実装方式等機能分散システムのインプリメンテーション技術の開発。

f. ユーザ・マシンとして接続される各種専用マシン・アーキテクチャと，その構築技術。

② 機能分散実験システム

①及び他の課題の中間成果を取り入れながら，機能分散アーキテクチャによる実験システムを構築する。具体的にはパーソナル・コンピュータ２０台程度のローカル・ネットワークとし，中心のコンピュータは，１０台程度の各種高級言語マシンと他課題で完成した実験マシン（科学技術計算マシン，データベース・マシン等）を組込む。これに関し本課題中で開発を行なう分は次の通りである。

a. パーソナル・コンピュータ

基本となるＶＬＳＩユニバーサル・ホスト・プロセッサ及び，パーソナル・コンピュ

ータの基本アーキテクチャを開発し，現在の標準大型機程度の計算能力を持ち，マンマシン・インタフェースを備えたシステムを開発する。

b. 高級言語マシン群

１０種程度の専用又は汎用高級言語マシンを開発し，システムに接続する。

c. ローカル・ネットワーク

a,b及び他課題による成果を総合したローカル・ネットワーク中心の機能分散システムの構築。

③ 機能分散ネットワーク・システム

①，②の成果及び，他課題の最終成果を統合し，複数のセンタ・コンピュータをグローバル・ネットワークで結合したシステムの構築。

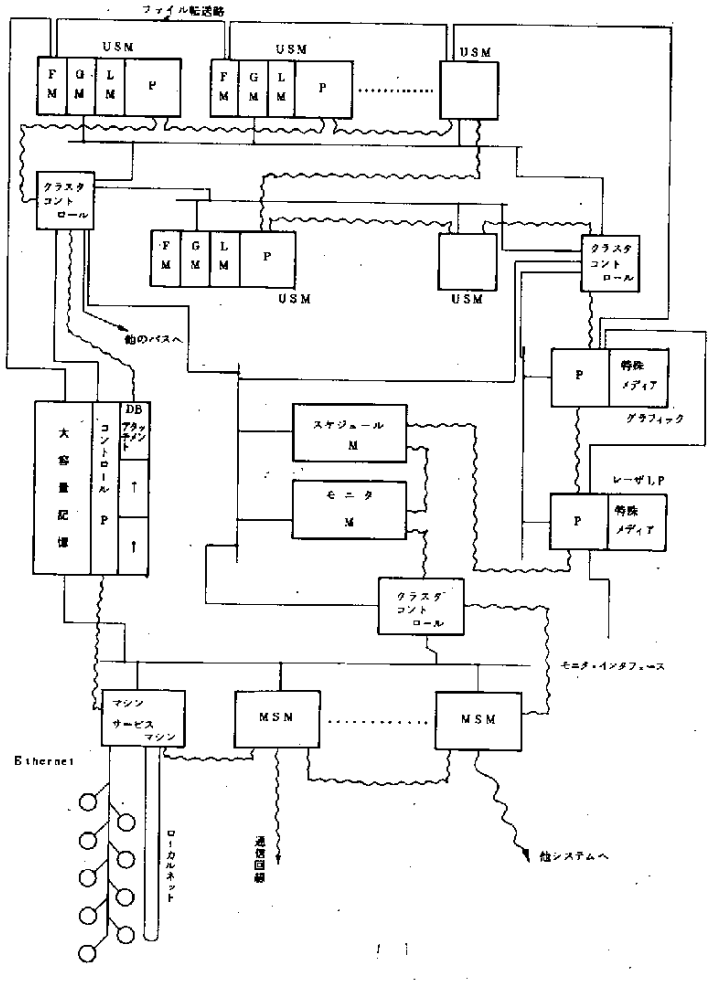


図６－１０　システム構成例　(1)

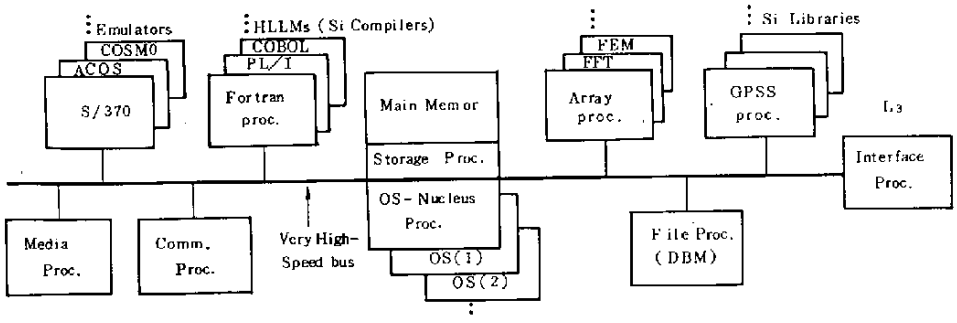


図６－１１　システム構成例　(2)

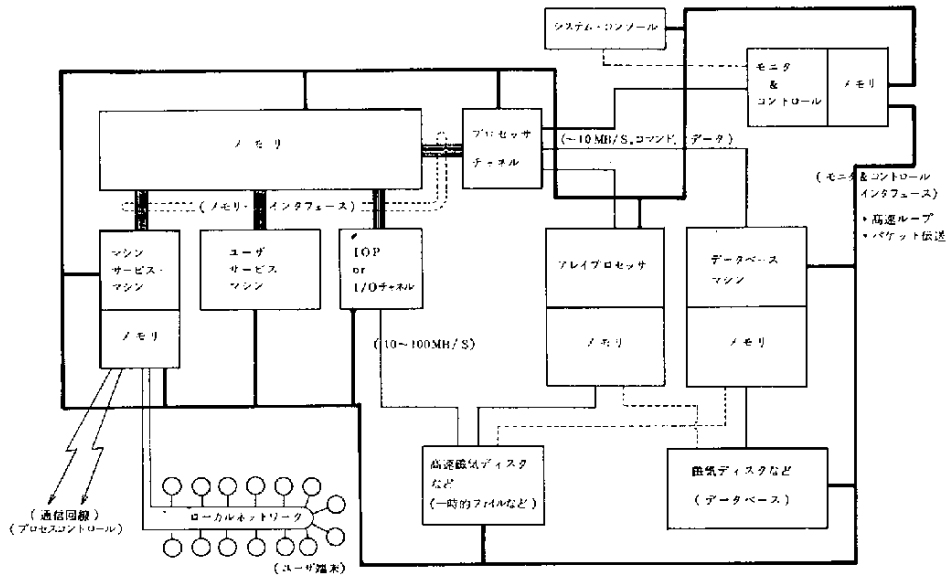


図６－１２　システム構成例　(3)

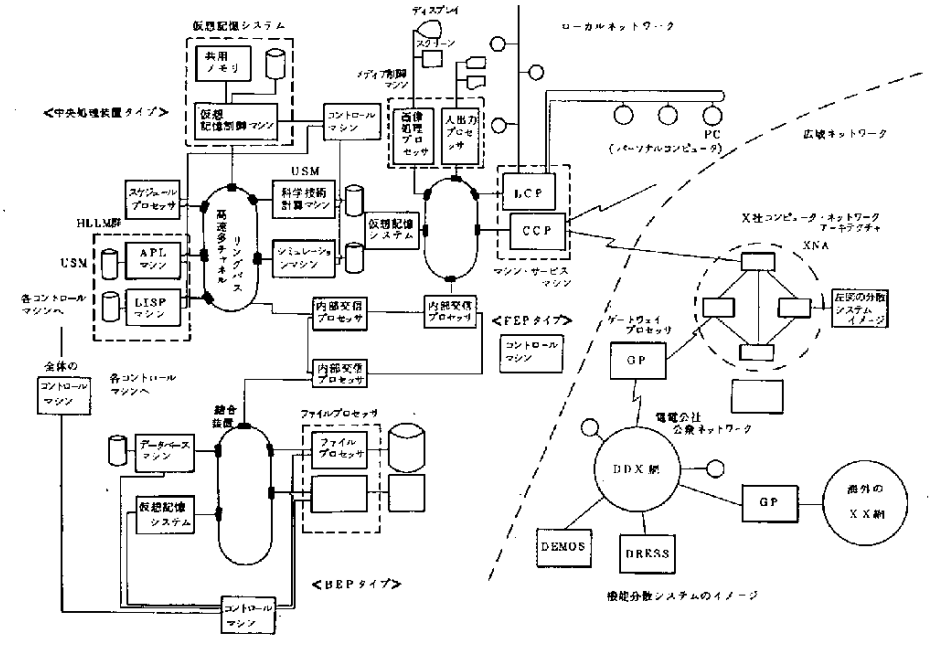


図６－１３　システム構成例　(4)

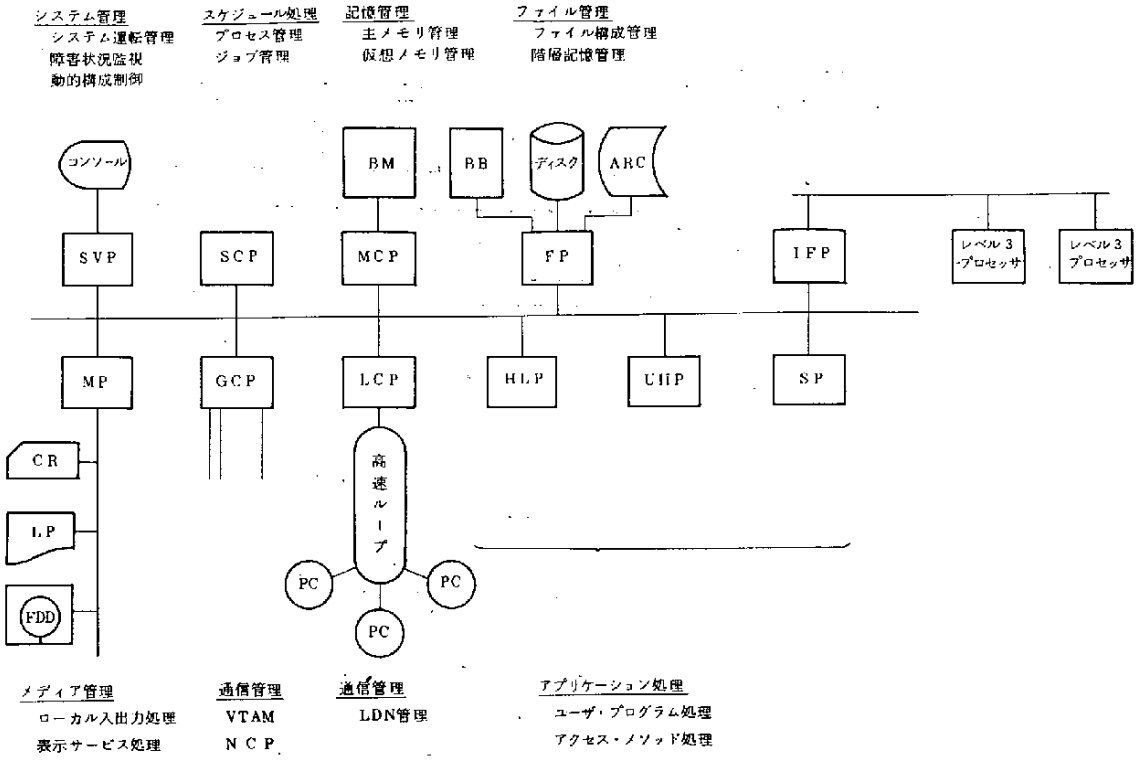
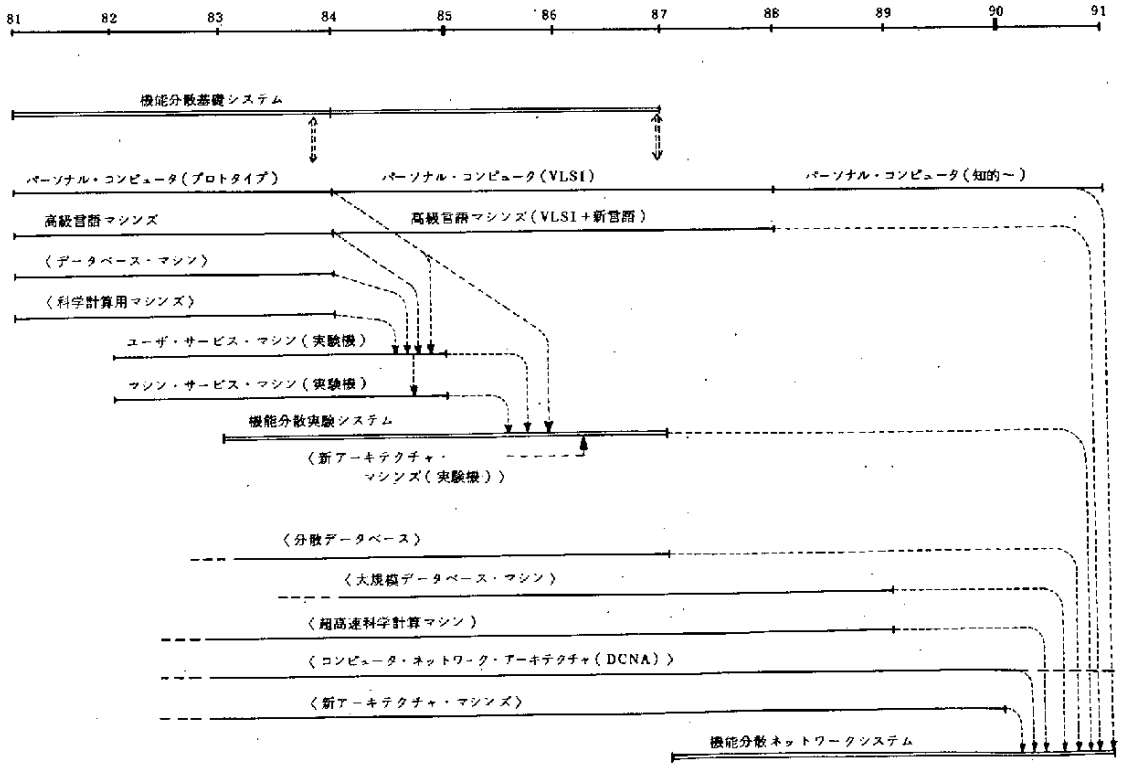


図６－１４　システム構成例　(5)

表６－１０　機能分散アーキテクチャの研究開発スケジュール



6.2.8 ネットワーク・アーキテクチャ

〔1〕 概要

コンピュータ技術と電気通信技術の進歩，個人の社会活動や企業活動の多様化・高度化・広域化に伴ない，情報化社会の構築が進展している。情報化社会では，情報の生産・流通が価値を生む社会であり，地域的に分散配置されている複数のコンピュータや入出力端末を通信回線で有機的に結合したコンピュータ・ネットワークが実現される。情報化社会の中核としての役割りを果すコンピュータ・ネットワークは，以下の条件を満たす必要がある。

① コンピュータ・システムを構成する資源（各種マシンのハードウェア，ソフトウェア）を，そのシステムに属している利用者だけでなく，他の異なるコンピュータ・システムに属しているマシンや利用者からも，経済的に共用利用できるようにする。———— 資源共用による経済化

② データベース等を複数のマシンに分散し，災害時・システム障害時の危険分散を可能とする。———— 資源分散による信頼性の向上

③ 利用者に近い場所で一部の処理を行なうことにより，応答時間や通信コストの削減をはかる。———— 分散処理による価格性能比の向上

④ コンピュータや端末の種別，ネットワークの構成に依存しないソフトウェア作成を可能とする。また，新しいコンピュータ，端末，新しいサービスの導入を容易にする。

———— 仮想化，および開発・導入・運用・移行の効率化

ボトム・アップにコンピュータ・ネットワークを実現するアプローチや，他のコンピュータ・システムを無視して，個別に独自のネットワークを構成するアプローチでは，上述の条件を達成することが非常に困難となる。このため，コンピュータ・ネットワークを構成する各装置の役割りと，それらの間の通信に関する規約（プロトコル）を確立する必要がある。

このような考え方から生まれたものが，ネットワーク・アーキテクチャという概念である。すなわち，ネットワーク・アーキテクチャとは，物理的な構成要素から成るネットワークを，論理的なノード（例：ユーザ・サービス・マシン，マシン・サービス・マシン，パーソナル・コンピュータ，端末等をモデル化したもの）とリンク（例：通信回線，チャネル等情報を運ぶ媒体をモデル化したもの）で表現し，すべてのノードに対して共通の論理的な階層を定め，各階層ごとにプロトコルを規定したものをいう。

機能分散アーキテクチャと同じく，コンピュータ・ネットワークを実現するためには，このネットワーク・アーキテクチャの研究が不可欠である。

一方，ネットワーク・アーキテクチャに基いて実現される第5世代のコンピュータ・ネットワークは，ネットワークの地域的な広がりの観点から，ローカル・ネットワークと広域ネットワークに分類され，その主な構成・機能は，以下のようになると考えられる。

① パーソナル・コンピュータが発達し，これらを結合したローカル・ネットワークが，

研究機関，大学，企業内，地域社会内等で実現される。

② 高速通信回線による資源共用ネットワークが広域ネットワークとして実現される。また，ローカル・ネットワークは，広域ネットワークと結合される。

③ 利用者は，通常，ローカル・ネットワーク内の資源にアクセスするだけでよい。しかし，他の資源を必要とする場合は，特別の配慮を必要としないで，ローカル・ネットワークのゲートウェイを通して広域ネットワークの資源にアクセスできる。

④ 広域ネットワークは，企業等が独自に実現する専用ネットワークと，ＮＴＴが提供する公衆ネットワークから成り，両者は必要に応じて結合される。また，これらは，海外の広域ネットワークとも結合される。

⑤ ネットワークには，電話をはじめ，データ・テレフォン，ファクシミリ，家庭用ＴＶ，パーソナル・コンピュータ，センサ，ＯＣＲ，複写機，等を接続でき，音声，データ，図形，画像等の多様な通信メディアを利用した通信が可能となる。

⑥ 高度の機密保護と高い信頼性を実現している。

上記のようなコンピュータ・ネットワークを実現するためには，ネットワーク・アーキテクチャの研究の他，ⓐ高性能通信回線技術（光通信等），ⓑ広域通信方式（衛星通信方式等），ⓒネットワークの中核となるマシン・サービス・マシン，ⓓネットワーク利用を容易にするネットワーク用ソフトウェア（ネットワークＯＳ）の実現，ⓔマルチメディア処理技術，等の技術の研究が必要である。

本研究開発課題は，①ネットワーク・アーキテクチャの確立と，②コンピュータ・ネットワーク実現技術，に関するものであり，特に，機能分散アーキテクチャの研究開発課題と密接な関連がある。

〔2〕 必要性

コンピュータ・ネットワークは，情報の伝達・蓄積・交換・高度の情報処理を行なうシステムであり，種々のコンピュータ，入出力端末等を通信回線で疎結合したものである。このため，機能分散アーキテクチャとしてとらえることも可能である。

コンピュータ・ネットワークを効率的に実現するためには，ネットワーク・アーキテクチャという基本概念の確立が必要なことは，先に述べたことからすでに明らかである。

ネットワーク・アーキテクチャの体系化・標準化による効果を，コンピュータ・ネットワークを実現する立場から，ここで改めて要約すれば以下のようになる。

① コンピュータ・ネットワークを構成する各種マシンの機能分担を明確にし，製品の体系化をすることができる。

② 情報処理と通信処理の機能分担を明確にし，機能の重複に伴なう二重投資等を事前に回避できる。

③ 独立に，目的別に実現された他のコンピュータ・ネットワークとの接続が容易になる。

機能分散アーキテクチャと同じく，ネットワーク・アーキテクチャの確立は，放っておいても自然に達成できるものではない。第5世代の世の中のニーズと，それを実現するために必要とされる機能を十分に把握した上で，トップ・ダウン的なアプローチを重視して研究を進める必要がある。

これまでの経験から，コンピュータ・ネットワーク実現のために必要な技術の基礎は固まってきている。しかし，より高度なネットワーク実現のためには，多くの問題点が残されている。

例えば，ネットワーク・アーキテクチャの標準化は，ＩＳＯ，ＣＣＩＴＴ等で進められているが，上位層に向けての国際標準の早期勧告が望まれている。また，既存の異なるコンピュータ・ネットワークの統合は，その計画といくつかの試みがスタートした段階にある。ネットワークＯＳも限定された利用範囲内で実用化が進められているのが実情である。通信処理と情報処理は，次第にその区別がつかなくなり，現在は混沌とした状態にある。

理想のゴールから見れば，現在はスタート地点に立つために必要とされる道具だてと，スタート前の基礎実験が終った段階にあるといっても過言ではない。

〔3〕 内外技術の現状

(1) 国内技術

ネットワーク・アーキテクチャの研究は，１９７４年にＩＢＭがＳＮＡを発表して以来，各コンピュータ・メーカを中心に進められた。現在，各メーカは，自社のネットワーク・アーキテクチャを開発している。これらには，ＡＮＳＡ（東芝），ＤＩＮＡ（日電），ＤＯＮＡ（沖），ＦＮＡ（富士通），ＨＮＡ（日立），ＭＮＡ（三菱電機）等がある。ＮＴＴは，メーカ4社と協力してＤＣＮＡを開発している。しかし，これらの間に非両立性は，ネットワークの高度利用が進むにつれて特に問題となる。また，郵政省も国内標準プロトコルの設定を目標としたＣＣＮＰをまとめており，その一部は１９８０年１１月に告示された。

ネットワークＯＳの研究は，ＪＩＰＮＥＴ，Ｎ－１（大学間コンピュータ・ネットワーク）等で進められている。高位プロトコル・サポート，ネットワークＪＣＬ，等が実現されている。

ローカル・ネットワークの研究も進められている。例えば，Ethernetに関しては，慶大でAcknowledging Ethernetを，東大でPriority Ethernetを，早大でReservation Ethernetを試作している。また，光ループネットワークも注目されてきている。

広域公衆ネットワークとしてＤＤＸが１９８０年に主要都市に建設された。また，ＫＤＤは，国際ネットワークの実現を進めており，その第１期として，１９８０年にＩＣＡＳ（International Computer Access Service）を開始した。専用コンピュータ・ネットワーク・システムは多数実現されている。今後は，異なるシステム間の結合が進展しよ

う。

公衆通信への適用を狙った光通信方式は，一部現場試験を終了している。中容量ディジタル光伝送方式（６～１００Ｍビット／秒）は，比較的早期に商用化できよう。

衛星通信方式は，ＣＳ*，ＢＳ**を使用して基礎実験を進めている段階である。

(2) 海外技術

ネットワーク・アーキテクチャの代表例は，ＩＢＭのＳＮＡである。またネットワーク・アーキテクチャ標準化組織として，ＩＳＯとＣＣＩＴＴがある。主として，前者はシステム利用者の立場から，後者は通信網提供者の立場から，それぞれ標準化を進めている。

ネットワークＯＳの研究は，アメリカでは主にＡＲＰＡＮＥＴを使用して行なわれている。ＸＮＯＳ，ＲＳＥＸＥＣ，分散データベース用のＳＤＤ－１／２等のプロジェクトがある。例えば，ＲＳＥＸＥＣでは，ネットワーク・ファイル・アクセスの統一，重要ファイルの多重複写，コマンド／プログラムレベルのファイル操作，既存のコンピュータ・システムで作られたプログラムがそのままネットワーク環境下で使えるカプセル化技法，等を実現している。

ローカル・ネットワークの代表例としては，カリフォルニア大アーバインのＤＣＳ，ベル研のＳpider，Xeroxで開発されたＥthernet，ＣＡＴＶ利用のＭＩＴＲＩＸ，等がある。ＤＥＣ，Xerox，インテルの３社は，Xeroxが特許を持つＥthernet の拡張と製品化についての共同開発を進めている。

広域ネットワークの建設も進められており，Telenet，ＴＹＭＮＥＴ（以上アメリカ），Datapac（カナダ），ＴＲＡＮＳＰＡＣ（フランス），ＥＵＲＯＮＥＴ（ＥＣ９か国）等がある。

光通信方式の研究も活発である。イギリスでは，８，１４０Ｍビット／秒の現場試験を既に終え，イタリアも１４０Ｍビット／秒の現場試験を終了した。***

衛星通信方式は，海外では一部商用化されている。現在注目されている衛星通信を利用したコンピュータ・ネットワーク・システムには，アメリカのＳＢＳとＸＴＥＮがある。これらは，画像会議サービス，電子郵便サービス，等を提供する予定である。

マシン・サービス・マシンは，国内外で多数実用化されている。現在，ハードウェアは漸進的進歩が続いている。ネットワークＯＳ等の機能分担とその実現が大きな問題になっている。また，ＬＳＩ技術の進歩とネットワーク・アーキテクチャの標準化により，通信用LSI

---

*）ＣＳ：実験用中容量静止通信衛星，愛称さくら。

**）ＢＳ：実験用中型放送衛星，愛称ゆり。

***）公衆通信用

や暗号用ＬＳＩが実現されてきている。前者には，ＥＣＬ－１５３３（ＤＣＮＡ共同開発），ＷＤ－２５０１/２５１１（W.D.社），Intel－８２７３，等があり，後者には，Intel－８２９４，ＴＩ－９９４０，ＷＤ－２００１/２００２，Fairchild－９４１４，等がある。マシン・サービス・マシンの回線スループットは，最大４８Ｋビット/秒クラスのものが多い。また，最大収容回線数は，２５６回線クラスが中心であるが，８００～1,000 回線まで収容可能なものも出現している。なお，最近は，構内交換機にデータの処理機能を付加したＥＰＡＢＸも，マシン・サービス・マシンとして利用されてきている。

〔4〕 研究開発内容

コンピュータ・ネットワークの実現に関する研究開発内容は，広い分野にまたがっている。このため，第５世代の研究では，コンピュータ・アーキテクチャに直接関連する部分を中心にとり上げ，光伝送通信方式や衛星通信方式等，他の研究に委ねることとし，ここでは特にとり上げない。

(1) 研究開発項目

① ネットワーク・アーキテクチャの確立

ネットワークを効率よく実現し，各種のコンピュータ，入出力端末等を容易に接続するために，ネットワーク・アーキテクチャの体系化・標準化を行う（基本理念の確立）。

将来にわたって発展性，拡張性のあるコンピュータ・ネットワーク実現のためには，次の目標の達成が重要である。

ⓐ 複数の異なるマシンに共通な論理構造とプロトコルを提供する。

ⓑ 新データ網（パケット交換網）と，コンピュータ，端末等との間で，通信機能の適正な分担を行なうことができる。

ⓒ 公衆ネットワークと専用ネットワークの双方に適用できる。

ⓓ 統一された仮想端末プロトコルを提供する。

ⓔ 効率的に実現できるプロトコルである。

ⓕ 音声・データ・図形・画像等のマルチメディア用通信プロトコルを提供する。

ⓖ 本格的な分散処理プロトコルを提供する。

ⓗ 国際標準と整合する。

また，ネットワークの高度利用を可能とするため，次に述べるネットワークＯＳの研究とあわせて，高位レベルの概念を確立する研究を重視する。

② ネットワークＯＳの確立とその実現

ネットワークＯＳは，ネットワーク利用の容易化，負荷の分散，資源の共用等を達成するために必要なソフトウェアとプロトコルの集まりである。このネットワークＯＳは，１つのマシンに集中的に配置されるのではなく，ネットワークを構成する各マシンに分散配

置され実現される。

第５世代で必要とされる主な機能には，次のようなものがある。

ⓐ　ネットワーク・ジョブ管理機能

ⓑ　ネットワーク・データ管理機能

・ネットワーク・ファイル管理機能

・分散データベース管理機能

ⓒ　データ移送機能

ⓓ　プロセス間通信機能

ⓔ　ネットワーク制御機能

なお，分散データベースの実現技術に，特に重点を置く必要があろう。

③　プロトコルの記述．作成，検証技術

コンピュータ・ネットワークの高度化に伴ない，規定されるプロトコルも増加し，複雑なものとなる。このため，プロトコルを論理矛盾なく簡潔に記述する方法や，インプリメントのために切り出されたプロトコルを実験装置等を使用して検証する技術が，これまで以上に重要となる。

第５世代では，この技術をさらに進めて，インプリメント条件（物理的な構成条件とサービス仕様）を与えることにより，検証されたプロトコルと各マシン用のソフトウェアを自動生成する技術や，プロトコルの修正・追加等に対して，すでに規定されたプロトコルとの矛盾を自動的にチェックしながら，新しいドキュメントを編集・出力する技術の研究，等を行なう。

④　マルチメディア処理技術

第５世代では，高度のマン・マシン・コミュニケーションシステムを構築するため，コミュニケーション・メディアを多様化させる必要がある。したがって，音声，データ，図形，画像信号等，異なるメディアを統合して扱う技術や，各種メディア間の変換技術を研究する。

例えば，各種のメディア信号を画一化でき，さらに伝送速度の異なる信号の多重，挿入，分割が容易なディジタル・ネットワークや，パケット形式で音声や画像を伝送する方式を，マシン・サービス・マシン・アーキテクチャ開発の一環として行なう。

⑤　マシン・サービス・マシンの実現

第５世代におけるマシン・サービス・マシンの代表的な利用形態は，以下のようになると考えられる。

㋑　前置処理装置（ＦＥＰ：Ｆront Ｅnd Ｐrocessor）

㋺　遠隔集線多重化装置（ＲＣ：Ｒemote Ｃoncentrator）

㈡ メッセージ交換装置（ＭＳＷ：Message Switch ）

㈢ ゲートウェイ装置（ＧＰ：Gateway Processor ）

この他，テレメータやテレコントロール等を目的とする形態，端末群を一括管理する形態等がある。

このような多様な利用形態に耐えうると共に，先に述べたネットワークＯＳ等の高度な機能を実施するマシン・サービス・マシンを開発する。すなわち，ⓐより一層の高速回線のサポートや，マルチメディア機能，ネットワークＯＳ機能をリアルタイムに処理できる高い処理能力を実現するだけでなく，ⓑ２４時間連続して運転ができる。ⓒ特殊な設置環境を必要としない，ⓓ高い信頼性を持ち，無保守である，ⓔ小形・軽量・低消費電力・安価である，ⓕ拡張性，増設性が高く，ネットワークの変更に柔軟に対処できる，等の条件を満たすマシン・サービス・マシンを開発する。また，超高速伝送インタフェース用ＶＬＳＩ，ディジタル信号処理や伝送制御処理等を行なう通信用ＶＬＳＩ；次に述べる暗号処理用ＶＬＳＩ，等についても開発を進める。

⑥ 機密保護機構の実現

個人のプライバシーに関する情報や，企業内，国家の機密情報が伝送路上から盗まれる危険性や，特殊装置を回線に接続することにより，各種ファイルの破壊や修正が行なわれる危険性等を減少させるため，暗号処理機能や個人識別機能等をネットワークの中で実現する。

(2) 研究開発スケジュール

研究開発スケジュールを表６－１１に示す。

研究開発は，以下のように行なう。

① 前期（ステップⅠ）

ネットワーク・アーキテクチャの基本理念の確立を行なう。

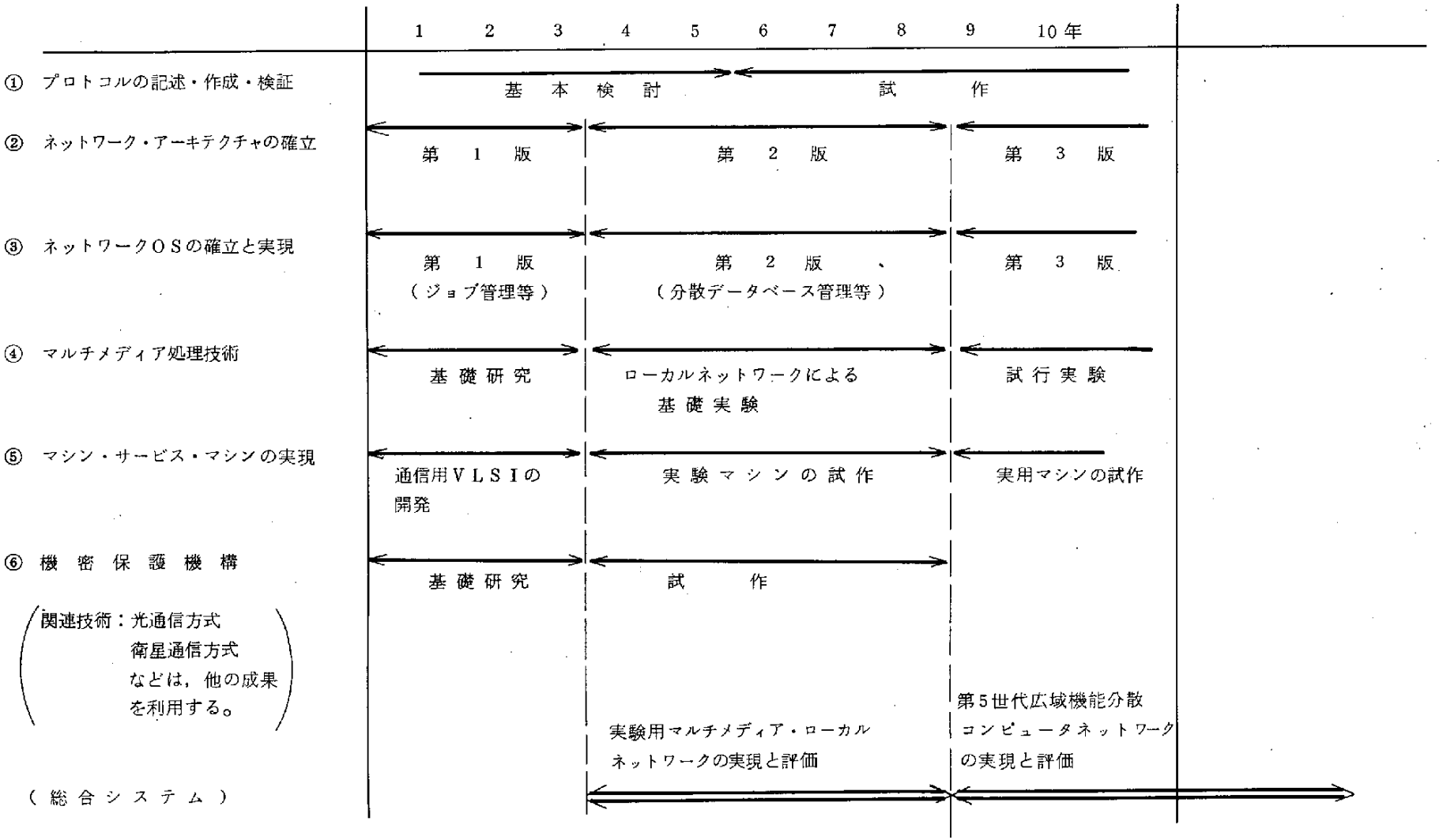
② 中期（ステップⅡ）

パーソナル・コンピュータおよび各種入出力端末を結合した，実験用マルチメディア・ローカル・ネットワークと，そのために必要とされるマシン・サービス・マシンを開発する。この上で，ネットワークＯＳの実現等の基礎実験を行ない，コンピュータ・ネットワーク実現技術を確立する。

③ 後期（ステップⅢ）

実験用マルチメディア・ローカル・ネットワークと，高性能ユーザ・サービス・マシン群を結合し，本格的な広域機能分散システムを実現する。この上で分散データベースシステム，オフィス・オートメーション・システム等を構築し，第５世代の知識情報処理用コンピュータ・ネットワークとする。

表６－１１　ネットワーク・アーキテクチャの研究開発スケジュール



なお，ＮＴＴは，すでに広域公衆ネットワークとしてＩＳＥＮ（Integrated Services Digital Network）の開発を発表している。したがって，必要に応じて強調して研究を進める。

(3) 研究開発の目標・仕様

① ネットワークＯＳ機能を実現するとともに，高位レベル・プロトコルを確立する。

② マルチメディアを利用した各種サービスを総合して提供できるコンピュータ・ネットワークとする。

③ ローカル・ネットワーク内のデータ伝送速度は，１００～５００Ｍビット／秒以上とする。また，ローカルネットワークを広域公衆ネットワーク（DDX等）に接続できるようにする。

④ ホストとなるコンピュータ・システムの数は，３システム以上とする。

⑤ マシン・サービス・マシンの信頼性を現在よりも１桁以上向上させる。また，ＶＬＳＩ化することを目標とし，マイクロプロセッサや通信用ＶＬＳＩチップを積極的に導入する。

6.2.9 高速演算マシン

〔1〕 概要

科学技術の進歩とともに大規模化，複雑化する科学技術計算を高速処理するための高速演算マシンが強く望まれている。

物理学上の理論研究，工学上の構造物の設計，その安全性，さらには高価な費用を必要とする実験に代る数値モデルによるシミュレーションなどのために高速演算マシンの果す役割と重要性は，１９９０年代にはますます増大すると予想される。

現在のマシンで処理できない問題を，現実的に意味ある時間内に処理できるようにするために，実効的に現在の最高速マシンの１００倍程度の性能をもつものが望まれる。

具体的には，10^3～10^4 ＭＦＬＯＰＳの演算処理能力と10^2～10^3 Ｍ語を主記憶をもち，強力な並列処理向け言語とそのコンパイラをもつマシンの開発が目標である。

この目標に到達するためには，高速論理素の開発または高密度実装の開発が必要であるとともに，並列処理方式の新しいアーキテクチャの開発が必要である。

高速論理素子としては，ガリウム砒素電界効果素子，ジョセフソン接合素子などの新しい素子の研究も必要である。また，大規模集積化技術，高密度実装技術，冷却技術の開発が必要である。アーキテクチャ面では，対象となる問題の数値解法の研究とそれに適合できる並列処理方式の開発が必要である。

〔2〕 必要性

高速演算マシンを必要とする科学技術計算分野としては，気象計算，原子力計算，核融合研究，分子科学研究，航空工学，土木建築構造計算，半導体製造，資源探査，画像処理などが挙げられる。

気象計算では，現在の予報よりも時間間隔の長い中期ないし長期の予報，台風の進路予報などの大規模計算が容易になれば，その社会的，経済的効果が大きい。

原子力関係では，原子炉設計計算をはじめ安全性の解析，シュミレーションなど原子力発電所の安全性の問題に寄与することができる。

核融合関係では，電磁流体の安定性，そのシミュレーションなどの計算により，電磁流体発電などの未来のエネルギー開発に寄与することができる。

分子科学関係では，分子の電子軌道の計算などにより，化合物生成の経済的な方法などを探ることができる。

航空工学では，高速飛翔体の設計計算に寄与するところが大きい。特に，費用のかかる風洞実験の代りに数値計算シミュレーションがある程度まで可能になるので，その経済的効果が大きい。

構造計算関係では，大型化する建造物の安全性，耐震性などの計算に寄与する。

半導体製造では，大規模集積化する電子回路の計算が，半導体の設計に不可欠であり，その計算も集積度の向上に伴ないますます大規模化しつつある。

資源探査では，石油の探査などの探査技術が進むにつれて探査信号の処理量の要求は，ますます増大しつつある。

画像処理では，例えば，国土資源調査のための画像処理など新しい要求が生じつつあり，画像処理が必要とする計算量は，今後ますます増大する。

〔3〕 内外技術の現状

(1) 国内技術の現状

大型の技術計算の分野では，現在，主として汎用大型機が使用されている。これらの汎用大型機の性能は，近年の半導体技術の進歩によりＬＳＩ化が進んだことおよびキャッシュ・メモリ，先行制御，パイプライン制御などの方式設計技術の進歩などにより向上し，十数ＭＩＰＳ達している。この性能は海外の汎用大型機のそれと肩を並べている。

また，技術計算の高速処理のニーズに対処して，技術計算専用マシンの開発も行なわれつつある。

技術計算専用マシンとして開発されたものに，ＦＡＣＯＭ２３０／７５ＡＰＵ（Array Processor Unit）およびＨＩＴＡＣ　Ｍ－２００Ｈ　ＩＡＰ（Integrated Array Processor）とがある。前者は汎用マシンＦＡＣＯＭ２３０／７５と非対称マルチ・プロ

セッサを構成し，後者は汎用マシンM－２００Hの内部に付加機構として内蔵される。

最近，後者のタイプのものとして，ＡＣＯＳ1000統合アレイプロセッサ，ＣＯＳＭＯ７００ＩＡＰなどが発表されている。

ＦＡＣＯＭ２３０／７５ＡＰＵは，アレイ・プロセッサ専用のＦＯＲＴＲＡＮを用意し，ユーザはアレイ・プロセッサを意識してプログラムを作成することができる。

一方，内蔵型のタイプは，汎用マシンの付加機構として用意され，標準FORTRANで使用できるなど，性能価格比の面や使い易さの面で特長をもち，実効的な処理能力を高めている。

国内で開発されたものは，いずれもパイプライン方式に分類されるもので，並列処理型やプロセッサ・アレイ型のものは，まだ開発されていない。

しかしこれらの技術計算専用マシンは，米国を中心に開発されているスーパー・コンピュタと呼ばれる技術計算専用マシンと比較すると，未だ十分な性能を持っているとは言えない。

一方，論理素子としては，ＥＣＬまたはＣＭＬ素子を使用しており，集積度は１チップあたり１００のレベルに達している。ガリウム砒素電界効果素子の研究は１チップあたり 100 ゲート以下の小，中規模ＩＣを作るレベルに来ている。

ジョセフソン接合素子は，現在は論理回路としては，単位ゲートの実験レベルであり，記憶回路では数１０～２Ｋビットでの実験レベルである。

(2)　海外技術の現状

汎用大型機の分野では，ＩＢＭ社を中心に，高性能の機種が開発されつつある。汎用大型機の性能は，国産機と同等程度であるが，今後，ＩＢＭ社による大容量の主記憶をもつさらに高性能な機種の発表が予想されている。

高速演算を必要とする分野に対しても，主として汎用大型機が使用されてきたが，既に１９６０年代の終りから技術計算専用マシンの開発が行なわれてきた。近年，ＣＲＡＹ－１の開発により，技術計算専用マシンの効果が確められ，高速演算へのニーズが高まりつつある。

技術計算マシンの開発とその状況は次のようである。

１９７０年代始めに，ＩＬＬＩＡＣ－Ⅳ，ＳＴＡＲ－１００，ＡＳＣ（Advanced Scientific Computer）などが開発された。これらはスーパーコンピュータとも呼ばれ，ＩＬＬＩＡＣ－Ⅳは並列処理，ＳＴＡＲ－１００とＡＳＣはパイプライン処理方式のマシンである。

１９７６年にＣＲＡＹ－１が開発され，その後，ＢＳＰ（Burroughs Scientific Processor），ＣＹＢＥＲ－２０３，ＣＹＢＥＲ－２０５，ＤＡＰ（Distributed Array Processor）などが開発された。これらには最新のハードウェア技術が使用され，１９７０年代始めのものに比し，よりコンパクトでより高性能なものになっている。

処理方式の面では，ＣＲＡＹ－１とＣＹＢＥＲがパイプライン方式，ＢＳＰが並列処理

方式（ＢＳＰは８０年１２月に製品化を中止された），ＤＡＰがプロセッサ・アレイ型と多様である。

ＣＲＡＹ－１は，パイプラインのクロックが１２.５ｎｓと小さく，また最新の高速半導体メモリを多量に使用して高速処理を実現しており，現在世界中に最も数多く設置されている。また，ＣＲＡＹ－１は標準ＦＯＲＴＲＡＮ仕様のままで使用でき，ある程度の汎用性をもつ技術計算マシンとして開発されている。

一方，高速処理の要求はさらに高まりつつあり，ＮＡＳＡではＮＡＳＦプロジェクトとして１Ｂ　ＦＬＯＰＳ以上の処理能力をもつNavier Stokes Equation Solver を計画している。

技術計算プログラム用の言語としては，一般には，ベクトル計算用の拡張仕様を設けたＦＯＲＴＲＡＮが使用されているが，そうした拡張仕様を統一化し，アレイ・プロレッサ向けの標準ＦＯＲＴＲＡＮ仕様を作成しようとする動きもある。

一方，論理素子としては，国内と同じくＥＣＬまたはＣＭＬ素子が主流であり，国内とほぼ同等の集積度のものを使用している。

ガリウム砒素電界効果素子は，今後の高速演算素子として世界中で注目されている。中でも米国では，国防省の援助による研究が進められている。

ジョセフソン接合素子は，ＩＢＭ社，ベル研究所を中心に１０年以上の研究が続けられており，最近は３０ｐｓ／ゲートのＯＲ回路，２ＫビットのＲＡＭが実現されている。

〔4〕 研究開発内容

(1) 研究開発項目

開発の最終目標は高速演算マシンであるが，総合システムとして稼動させるためには，ハードウェア技術からソフトウェア，入出力装置に至るまでの技術開発が必要である。ここでは，システムの運用に必要なものなどは除いて考えることにし，高速演算マシンの開発そのものに必要な開発項目を挙げる。

開発内容は，高速演算マシンとして必要な高速演算機構と記憶装置およびシステム制御機構である。

開発すべき技術を大分類化すると次のようになる。

(a) ハードウェア技術の開発

(b) アーキテクチャと論理方式の開発

(c) ソフトウェアの開発

(d) 大容量高速二次記憶の開発

(e) マンマシン・インタフェースの開発

これらについてさらに分解して以下に説明する。

① 高速論理素子の開発と実用化

高速演算を行なうため，高速の論理素子の開発が必要である。同時に配線遅れ時間を小さくするため，高密度実装を可能にする必要がある。

これらを可能にする新しい素子として，ガリウム砒素電界効果素子やジョセフソン接合素子などのもつ可能性を研究する必要がある。

② 高密度実装および冷却技術

高密度実装を可能にするためのＶＬＳＩ技術，マルチ・チップのハイブリッド実装技術などの開発が必要である。同時に高密度ピンのコネクタ技術の開発が必要である。

また，素子の高速化，高密度化に伴ない，単位体積あたりの発熱量が増大するため，効率のよい冷却技術，例えばフレオン冷却などの技術の開発が必要である。

③ アーキテクチャ

高速演算マシンの論理仕様として，命令語の形式，種類，詳細仕様を開発する必要がある。また，アドレス方式，数表現など技術計算と密接に関連する仕様についての研究が必要である。

④ 並列処理方式

10^3～10^4ＭＦＬＯＰＳの演算を行なうためには，並列処理方式を採用して行く必要があると予想される。しかし，例えば，連立偏微分方程式を連立一次方程式として，これを数値解法により解く場合を考えてみると，現在は必ずしも並列処理に適したアルゴリズムとは言えない面もある。

並列処理化に向いたアルゴリズムの開発とアルゴリズムに適合する並列処理方式との両面の研究開発が必要である。

⑤ 技術計算用の高級言語仕様

現在は，物理学，工学などの数式で表現されたもとの問題を離散化し，数値解法の対象となる数式で表現し，これに数値解法のアルゴリズムを適用し，その手順をＦＯＲＴＲＡＮなどの高級言語で記述している。

並列処理方式の高速演算マシンに適合したＦＯＲＴＲＡＮ拡張言語仕様の開発などが必要である。さらに，離散化した数値に近い表現のままで直接に記述できるようなイメージの超高級言語の研究も必要である。

⑥ 自動並列処理化，最適化コンパイラ

コンパイラによる自動ベクトル化技術，並列処理化技術，オブジェクト・コードの最適化技術の開発が必要である。

⑦ 高速大容量二次記憶

現在は，二次記憶装置として１０ｍｓオーダーのアクセス・タイムの回転体の磁気記憶

装置が使用されている。計算したい問題のデータが主記憶装置の容量を越えて二次記憶装置に格納されると，計算時間が二次記憶装置と主記憶間の転送時間で決まるようになる。演算速度を高速にしても，二次記憶装置の高速化を図らなければ，全体の計算時間を短縮することができない。

主記憶は１００nsオーダーのアクセス・タイムなので，μsオーダーのアクセス・タイムの大容量二次記憶装置の開発が必要である。

⑧　高機能マンマシン・インタフェースの開発

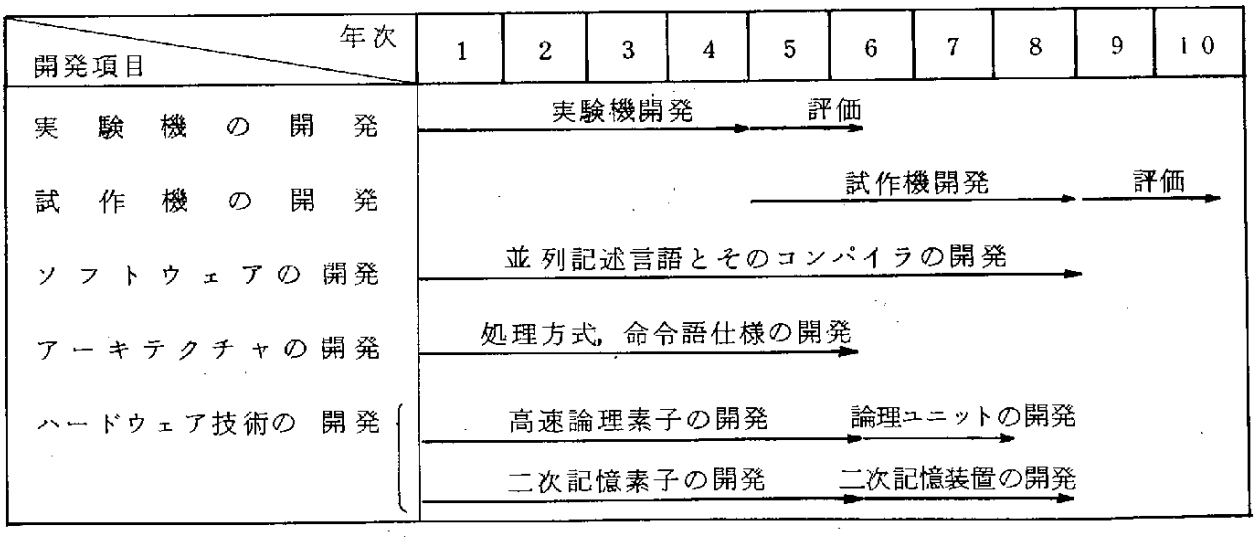
計算の中間結果を容易に表示できるシステムの開発および２次元，３次元表示の容易なグラフィック・ディスプレイ装置の開発が，計算結果の確認および理解を早めるために重要である。

以上の技術開発を行なう目的で，初めに実験機を開発し，アーキテクチャ，論理方式，基本となるソフトウェア，基礎的なハードウェア技術などの確認を行なう。

次に試作機を開発する。これは高速高密度実装を用いて開発し，ソフトウェア，大容量二次記憶，マンマシン・インタフェースなどについても最終仕様のものを開発する。

(2)　研究開発スケジュール

表６－１２　高速演算マシンの研究開発スケジュール



(3)　研究開発の目標・仕様

開発は，実験機と試作機とに分けて行なう。

①　試作機

(a)　性能　$10^3 \sim 10^4$ MFLOPS

(b)　アーキテクチャ

並列演算方式

(c) ソフトウェア

並列記述言語とそのコンパイラ

制御用プログラム

(d) 高速主記憶　容量２５６ＭＢ

(e) 大容量高速二次記憶　容量１～１０ＧＢ

(f) 大容量高速ファイル

容量１０～１００ＧＢの高性能ファイル

(g) 高機能グラフィック・ディスプレイ

② 実験機

(a) 性能　10^2～10^3ＭＦＬＯＰＳ

(b) アーキテクチャ

並列演算方式

(c) ソフトウェア

ＦＯＲＴＲＡＮ拡張言語とそのコンパイラ制御用プログラム

(d) 高速主記憶　容量　６４ＭＢ

(e) 大容量二次記憶　容量２５６ＭＢ

6.2.10 データベース・マシン

〔1〕 概要

コンピュータは元来，数値計算のために発明されたものであるが事故処理（いわゆるデータ処理）への応用によって飛躍的にその利用が拡大し，コンピュータの著しい発展をもたらした。今日においてコンピュータによる高速，大量のデータ処理機能は不可欠のものになり，しかも応用の高度化に伴って，多数のオンライン・ユーザが共有化したデータへ並列的にアクセスするシステムが必要となり，データベース・システムが導入されてきた。

データベース・システムは今後ますます大容量化，高速化が要求されると考えられている。しかしながら現在のコンピュータのアーキテクチャは数値計算指向で作られたフォンノイマン型が主流であり，データベース・システムで要求される大量のデータの管理，ファイル装置とのデータの入出力などには適していないと云われており，既存のデータベース・システムですら十分に性能要求を満していない場合が多いのが現状でこのまゝでは将来において要求と実現性能の差は一層拡大すると思われる。そのためデータベース・システムに適したアーキテクチャを持った専用コンピュータ即ち，データベース・マシンの研究が盛んになり，既に商用データベース・マシンも出現している。

データベース・マシンの一般的な方式は；

① 階層的に作られたホスト上のＯＳのデータベース管理機能の内下層の諸機能をデータベース・マシンにオフロードする。

② データ操作／管理のための効果的な実行機構をもつ。

③ ホスト・コンピュータとデータ記憶僧置間のデータ転送ネックを解消するための工夫をする。

などである。

〔2〕 必要性

データベースの利用は拡大しており今後も拡大をつづけると予想されていて，近い将来にはピーク時10^6要求/秒以上の処理能力が要求されるようになるかもしれないとの予測すらある。しかしながら現在のデータベース・システムは性能の高いものでも$10 \sim 10^2$要求/秒程度の処理能力しかない。またデータの量も現在のそれ（現在でも大きいシステムでは10^{12}バイトを越す）よりはるかに大きくなると考えられ一説では10^{18}バイト程度も実現性があるといわれている。このような大規模，高速のデータベース・システムを現在のコンピュータで実現することは性能の面で困難であるのみならず必要になるであろうハードウェアやソフトウェアの規模，複雑さから考えて信頼性の面でも大きな問題を生ずると思われる。現在のコンピュータのアーキテクチャは数値計算向きでありデータベース・システムで要求されるデータ操作／管理やファイル記憶装置との高速のデータ入出力などには最適とはいえず，データベース・システムに専用化したアーキテクチャを持ったコンピュータに依らなければ高度化する要求に適合したデータベース・システムは構築できないと考えられる。こうした背景からデータベース・システムの基本機能を専用ハードウェアで支援し，性能およびコストを改善し，ソフトウェアも単純化するデータベース・マシンが注目されている。

データベース・マシンの必要性について現在のデータベース・システムの問題点との関連でまとめると以下のようである。

① 現在のデータベース・システムのソフトウェアの複雑さの最大の原因は問合せで与えられるシンボリック名をメモリ番地へマップするために複雑なデータ構造が必要であり，挿入，削除等の基本操作に対し，この複雑なデータ構造を正しく保持しなければならない点にある。これに対し現在のフォンノイマン型コンピュータは数値計算や単純なデータ処理を主目的として最適化されており，この複雑なデータ構造の処理，ファイル記憶装置（磁気ディスク等）とのデータの入出力制御等の目的には必ずしも適したものと言えない。

② 現在のデータベース・システムは多くの機能的に異なる部分（例えば問合せのパージング，ディレクトリのアクセス，ディレクトリの処理，データ検索，追加・削除，セキュリティ管理）から成っている。これらを現在のコンピュータで同一のハードウェア上に実現するとバランスのよいシステムを構成するのが困難で多くの部分にボトルネックを生じやすい。これに対し，ホスト・コンピュータとの処理機能をうまく分担したデータベース・マシンに

よって解決をはかることが可能と考えられるし，更にデータベース・マシンに各種の専用のハードウェアを付加することにより多様な機能に対する処理能力のバランスをはかることもできるであろう。

③　データベース・システムは今後ますます大規模，複雑化することは明らかであり，また多様な応用の発生あるいは使用パターンの変化に伴う再構成の問題を避けて通ることはできない。このため，データベース・システム用のソフトウェアは大規模化しそれを実現するには大規模ソフトウェアの設計開発技術の確立と共に，データベース・システムの基本機能のハードウェア化によるソフトウェアの単純化が必要である。

これらの問題を解決するにはフフォンノイマン型コンピュータと本質的に異なったアーキテクチャが必要と考えられ，種々のデータベース・マシンが研究されている。その基本構成は図６－１５に示すようなものであり，

基本概念は次のようである；

①　バック・エンド型

フォン・ノイマン型のホスト・コンピュータをフロント・エンドに置きデータベース・マシンをバックエンドに置く，データベース・マシンにデータベース・システムのどの範囲の機能を持たせるかは各システムにより異なる。

②　コンテント・アドレッシングと並列処理

シンボリック名から直接二次記憶装置へアクセス可能なコンテント・アドレッシングの機構がデータベースマシンのアーキテクチャの基本となる。この結果当然前述の複雑なデータ構造が必要でなくなるあるいは簡素化される。コンテント・アドレッシングでは探索単位に

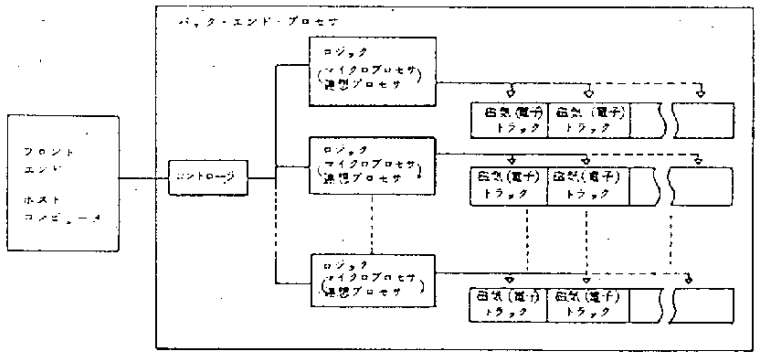


図６－１５　データベース・マシンの基本構成

従って二次記憶装置を適当に分割することにより並列的探索が可能になるので性能が大巾に改善できる。

③ データベース・システムの高次機能をデータベース・マシンのプリミティブとすることにより性能向上を得られる。例えば関係データベースにおけるセレクション，ジョイン等をプリミティブとしてハードウェア機能に持たす試みは多い。

〔3〕 内外技術の現状

(1) 国内技術

データベース・マシンあるいはデータベース・マシンの基礎となる技術の研究は国内においても最近盛んになり，多くの大学，研究所，メーカ等で進められているが商品化されたものはまだない。大学では北海道大学，東大，慶応大，横浜国立大などで研究が行なわれている。

北海道大学では専用のハードウェアによるサーチ，ソートを処理の単位とするデータフロー型のデータベース・マシンを試作中である。東京大学ではCellular Logic Typeのデータベース・マシンを発展させた可変構造多重処理データベース・マシンによってリレーショナル・データベースをサポートする研究を行っている。このシステムでは高速の光ループに多重のプロセッサ・モジュールを接続し，プロセッサ・モジュールとメモリ・モジュールの結合を可変にすることで高性能化を狙っている。東大ではこの他データ管理機能を独立したサブシステム化したデータ・サブシステムの研究や同一の処理モジュールを多数ならべた大規模連想プロセッサの研究も行っている。慶応大学では，データの格納および演算を属性単位で行い，演算の中間結果を動的マーク・ビットに保持する方式のリレーショナル・データベース・マシンを機能分散型マルチ・マイクロ・プロセッサで実現する研究を進めている。横浜国立大は独自のデータ・モデルをベースとした汎用コンピュータとミニコンピュータの密結合型データベース・マシンを試作している。

電総研は1976年から1980年にかけて磁気バブルを用いたデータベース・マシンを開発した。このシステムはマイクロ・プロセッサ，主記憶，磁気バブルメモリを組合せたデータ・モジュールを基本単位とし，これを複数台共通バスに接続した並列処理方式をとっている。

日電中研は汎用コンピュータをホストとしこれとメモリを共有したミニコンピュータを用いたデータベース・マシンを研究している。このシステムは汎用的に設計されていて任意のデータモデルを実現できるが，現在研究中のものはCODASYL仕様のデータベース環境下で動作している。

以上の外にも電電公社，沖電気，東芝などで研究が進められている。

このように国内においても研究は盛んになっているが実用レベルの規模のマシンは少ない

など海外に比較して進んでいるとはいえない。

(2) 海外技術

ここでは米国の状況を中心に述べるがヨーロッパにおいても研究は盛んである。データベース・マシンの研究は１９７０年代はじめの，

- 高速連想プロセッサＳＴＡＲＡＮのデータ管理への応用の研究
- ＡＲＰＡＮＥＴ用のデータ・ユティリティとしてのＣＣＡ社によるDatacomputerの研究
- ベル電話研究所におけるバックエンド型のデータベース・マシンＸＤＭＳの研究

などに端を発し，米国の大学を中心に盛んになった。１９７２年にフロリダ大学のＣＡＳＳＭプロジェクト，１９７４年にカナダのトロント大学のＲＡＰプロジェクトなどがはじまっている。そして１９８０年には初の商用データベース・マシンがＢｒｉｔｔｏｎ－Ｌｅｅ社から発表され（ＩＤＭ５００）実用化時代が始まった。これはまだ小規模なデータベース・システムを指向したものであるが，今後の発展を考えると重要な意義を持っていると思われる。

フロリダ大学のＣＡＳＳＭはセルラ・ロジックをデータベース・マシンに適用した最初の試みとして知られており，ディスクとその各トラックに付随したロジックから成るセルから構成し，ネットワーク・モデル，階層モデル，関係モデルのいずれもサポート可能である。トロント大学では１９７５年にＲＡＰプロジェクトを開始し１９７６年にはＲＡＰ．１として完成，その後大幅な設計変更をしてＲＡＰ．２を１９７７年に作っている。現在はアンカラ大学に移ったOzkarahanを中心にＲＡＰ．３が開発されている。ＲＡＰもＣＡＳＳＭと同様にセルラ・ロジック方式をとっており，同様の方式の研究はユタ大学，テクトロニクス社，ＩＢＭワトソン研究所などでも行なわれている。またオハイオ州立大学ではＲＡＰと同様に，小容量高速のContent Addressable Memoryと可動ヘッド・ディスクを改良した疑似的なContent Addressable Memoryを組合せてこれにいくつかのデータベース機能プロセッサを付加したデータベース・マシンを研究中でその基本設計を基に，あるコンピュータ・メーカが試作中ともいわれている。同様のデータベース・マシンにはＭＩＴのＩＮＦＯＰＬＥＸ，フロリダ大学のＭＩＣＲＯＮＥＴ，ウィスコンシン大学のＤＩＲＥＣＴ，パーデュー大学のＤＩＡＬＯＧなどがある。

Ｂｒｉｔｔｏｎ Ｌｅｅ社はリレーショナル・データベース機能をもつ初の商用データベース・マシン，（ＩＤＭ５００）を１９８０年２月に発表した。これはＤＢプロセッサ，２５６Ｋバイトの主メモリ，ディスク制御装置，ホスト接続用のチャネルなどから構成され，システム当りの最大磁気ディスク容量は３２ＧＢである。西独系のソフトウェアＡＧ社も１９８０年10月にデータベース・マシンを発表している。これは，ＩＢＭと互換性のあるプロセッサ上にＡＤＡＢＡＳパッケージをのせＩＢＭ系ホスト・コンピュータとチャネル結合するものであ

る。この他，ＭＲＩ／ＩＮＴＥＬ社のデータベース補助プロセッサがあるし，ＩＢＭシステム／３８をリレーショナル・データベースを内蔵した一種のデータベース・マシンとする見方もある。

このように海外においては以前から行なわれていたデータベース・マシンの研究の成果をもとに実用レベルのマシンが出てきている点は特に注目されるべきである。将来におけるデータベース・マシンの重要性を考えると日本においてもその研究や実用化を促進する強力な施策が望まれる。

〔4〕 研究開発内容

(1) 研究開発項目

大容量，高性能のデータベース・マシンを実用化するための研究開発項目には次のようなものがある。

(a) データベース・マシン・アーキテクチャの研究

- データの探索，照合，などを効率よく扱える新しいマシン・アーキテクチャの研究が必要である。たとえば関係代数マシンはリレーショナル・データベースの処理に向いていると云われているがその具体的研究が必要である。
- 連想処理もデータベース・マシンの高性能化に有効と考えられ，連想処理をベースとした新しいアーキテクチャのプロセッサ（連想プロセッサ）の研究も必要である。
- 大容量のデータベースを格納するメモリは低速，即ちアクセス・タイムが長くならざるを得ず，したがって問合せに対して短い応答時間を達成するには小容量高速のメモリを組合せてうまく使用することが必須となる。この記憶階層の構成法，や制御方法の研究が必要である。
- データベース・マシンの処理速度を高めるには並列処理の概念を導入せざるを得ないであろう（たとえば並列にサーチするなど）。したがってデータベース・マシンの中に多くの並列処理的要素を取り入れるための研究およびデータベース・マシンを並列に用いる研究を進めねばならない。
- データベース・マシンと他の機能分散プロセッサとの間でどのように機能分担するのが最も効率がよいかの研究が必要である。通信のオーバヘッドが少なく，論理的，物理的にシンプルな切り口（インタフェース）を設定しなければならない。
- この他研究項目としては，複数ホストからの共有方式，複数タスクに対する排他制御やデッドロック制御の方式，機密保護方式（パスワード，アクセス制御リスト，ケイパビリティ制御，暗号化など），などがある。
- 大容量のデータベース・マシンに対しては高い信頼性が要求されるので，高信頼

化の研究は非常に重要となる。ハードウェアの冗長構成，障害回復方式（バックアップ方式を含む）の研究が必要である。

(b) 各種データベースのサポートについての研究

- リレーショナル・データベースがデータベース・マシンにも適していると考えられているが現存する他の各種データベースに適用できるデータベース・マシンが当面は重要である。

  それにはそれらの各種データベースに対して前述のような研究を進めなければならない。

- またハードウェアとしてはリレーショナル・データベースを基本としたマシン構造とし，その上に各種データベース・モデルを構築する方法（シミュレーションなどを含む）の研究も必要であろう。

- さらに，既存データベースからリレーショナル・データベースに変換する方法についての研究が必要である。大容量のデータベースの変換はコンピュータで自動的に処理できないかぎり実現性がないと思われる。

(c) 高機能データベース・マシンの研究

- 厳密な手順に従った問合せだけでなく，あいまいな問合せに対しても推論機能などを活かして応答できるデータベース・マシンが望まれ，そのために人工知能などの研究成果をとり入れた高機能化の研究が必要である。

(d) マンマシン・インタフェースの研究

- 使い易いマンマシン・インタフェースを提供するために，自然言語（日本語）によって問合せが出来るとか，音声や図形で入力や出力が行なえることも望まれ，このための研究が必要である。

- また，それ以前の問題として現在の問合せ言語をもっと使い易いものに改良することが必要で，ＱＢＥを改良したようなものとか，まったく別のアプローチによる問合せ言語を研究しなければならない。

(e) 分散データベースの研究

- 分散データベースを実現するには多くの解決すべき問題がある。データベース・マシンを分散データベース・システムに用いるときにもそれらの問題は伴うので，その解決のための効果的サポート（ハードウェア機能等）を研究する必要がある。ユーザが分散配置されたデータベースを分散を意識せずに使用できるようにする方式，重複データの管理，最適分散方式，ネットワークを介してアクセスするときのプロトコルの標準化，異なるデータベースへ自由にアクセスできるようにする方式，障害管理などのテーマがあり，データベース・マシンのハードウェアとソフトウェア

へも影響があろう。それに対応した研究が必要である。

(f) ＶＬＳＩ技術の利用に関する研究

- ＶＬＳＩ技術の研究により現在は入手できない，あるいはコストが高くて利用できないような素子が利用できる可能性が出てくる。それらをうまく利用することによってデータベース・マシンの性能やコストを著しく改善できると予想されそのための基礎研究が必要である。また，逆にデータベース・マシンで利用価値の高い素子（その機能，構成，性能など）を研究して，提案しＶＬＳＩ技術で実現してもらうよう方向づける必要もある。

  以下に現在考えられる例について触れる。

- ＶＬＳＩによって機能アレイや論理メモリなど繰り返し構造をもった素子をつくり，これを各種のアプリケーションに利用する研究が行なわれている（例：Kungのシストリック・アレイなど）。このアプローチはＶＬＳＩの特質をうまく活かしたものでデータベース・マシンへの応用も重要な研究課題である。

- 大容量の連想メモリがＶＬＳＩで実現できるようになるとデータベース・マシンでも利用価値が高い。連想プロセッサのデータベース・マシンへの適用の研究の一環としてとりあげるべき課題である。

- 磁気バブルメモリなどを利用した電子ディスクはＶＬＳＩ技術によって大容量，低コスト化できると予想されるのでその利用方法，データベース・マシンに適した磁気バブルメモリの構成などの研究が必要である。

- ＪＪやＧａＡｓなどが実用化されるとデータベース・マシンのアーキテクチャも現在考えられているものと根本的に異なるものになるかもしれない。たとえばＪＪによって，超高速，大容量の連想メモリが実現できたとすればそれを中心としてシンプルな構造の高性能なデータベース・マシンが実現できるであろう。

  このように新しい素子の実用化は既存のアーキテクチャへのインパクトが大きく，十分な研究が必要である。

(g) データベース・マシン設計の基礎データの収集や分析

- よいデータベース・マシンを設計するには基本となる各種のデータを収集し分析しておかねばならない。たとえばデータベースへのアクセスパターンが応用分野によってどう異なるのかなどのデータを収集分析してデータベース・マシンのアーキテクチャ，構成，記憶階層などお設計に反映しなければならない。

(h) システムとしての綜合評価

- データベース・マシンのように複雑で新しいシステムは完成後実際に使用して各種の性能評価データを収集し，ボトルネックとなるところを改良する研究がなけれ

ば実用的なマシンにはならないと予想される。このため実験レベルのマシンでハードウェア，ソフトウェアを含めて十分な評価研究を行い，それを実用レベルのマシンの設計に反映することが重要である。

(2) 研究開発スケジュール

１９８５年頃を目標に現在のデータベース・システムの１０～１００倍の性能，容量をもったデータベース・マシンを開発し，これを実験機として十分に評価，研究を行う。その結果を反映して，さらに１０倍以上の性能をもった実用レベルのマシンを １９９０年頃に開発する。スケジュールの詳細についてを表６－１３に示す。

(3) 研究開発の目標・仕様

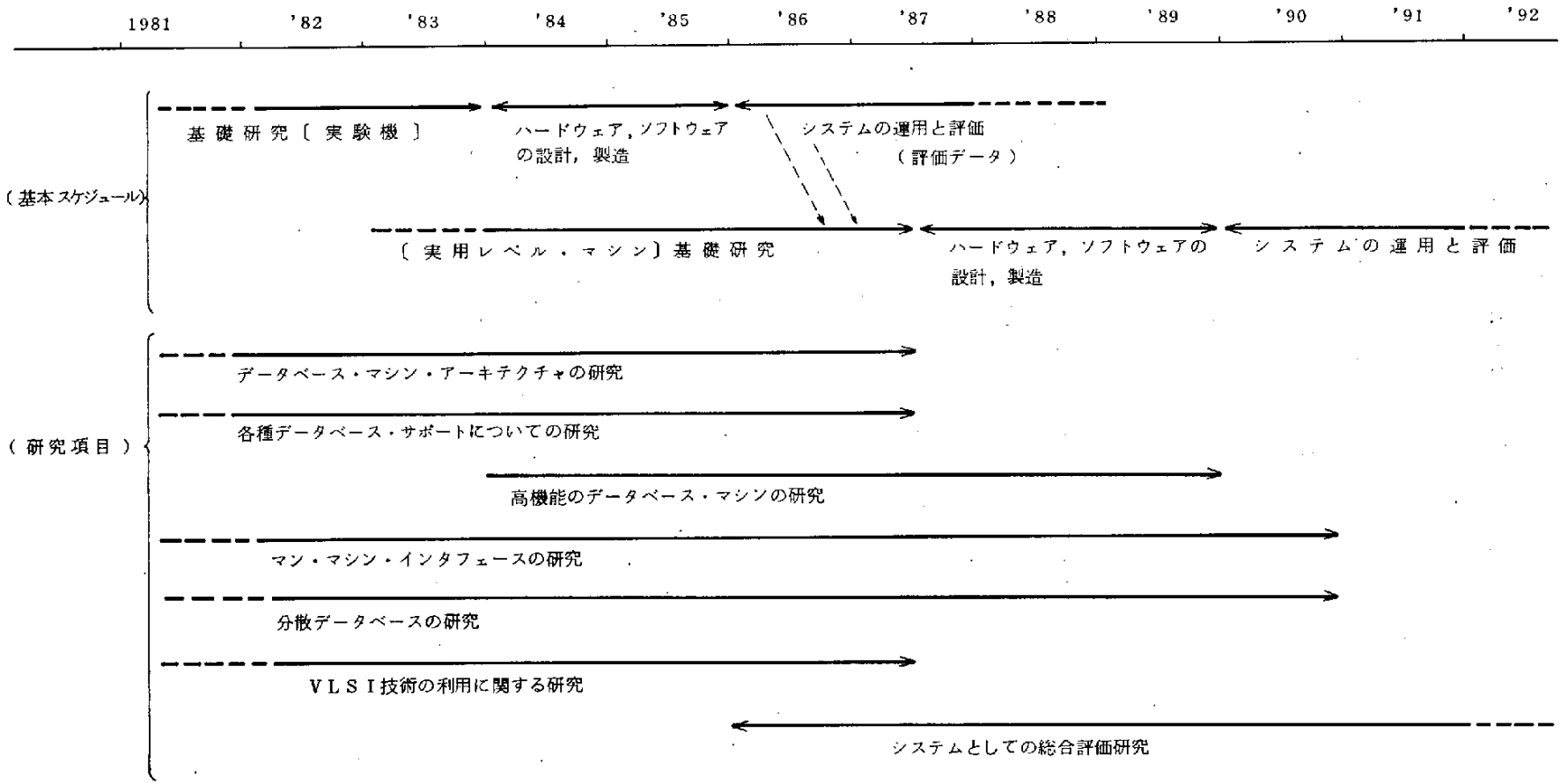
① 実験機（１９８５年頃）

- 容　　量　：最大１００ＧＢ，但し，オンラインで高速にアクセスされる部分で，バックアップのための大容量記憶装置（ＭＳＳなど）は含まない。
- 処理能力　：10^3 件／秒
- データモデル：リレーショナルを基本にしてハードウェアを構成するが，システムとしてはリレーショナル以外のデータベースもサポートできるようにする。このため必要なハードウェア機能があれば付加する。

② 本番機（１９９０年頃）

- 容　　量　：最大１０００ＧＢ，但し前述の如くバックアップ・メモリの容量はこれに含まない。
- 処理能力　：10^4 件／秒
- データモデル：リレーショナルを基本とするが他のモデルもサポートする。他のデータベースからの変換ツールもサポートする。
- そ の 他　：自然言語による問合せなど使い易いマンマシン・インタフェースを提供する。

表６－１３　データベース・マシンの研究開発スケジュール



6.2.11 高度マンマシン・コミュニケーション・システム

「3.5高度マンマシン・コミュニケーション・システム」を参照のこと。

6.2.12 ＶＬＳＩアーキテクチャ

「4.2ＶＬＳＩアーキテクチャ」を参照のこと。

6.2.13 知的ＶＬＳＩ－ＣＡＤ

「4.3ＶＬＳＩ－ＣＡＤ」を参照のこと。

6.2.1 4　第５世代アーキテクチャのシステム化技術

〔1〕 概　　要

第５世代コンピュータ・システムは，知識情報処理の実現を目指す，きわめて高度なシステムの構築を目標としている。一般に，このような未知の分野を多く含む目標へ到達するためには，いろいろな要素技術をいかに選択し，整理し，まとまった体系に組上げていくかというシステム化の技術がきわめて重要である。

これは，実際に物を作る時のノウハウに類するものから，開発の哲学，方針といったものまで，いろいろなレベルのものがある。ここでは，この中においても，より実際的なアーキテクチャやハードウェアを開発することを目的としたシステム化技術に範囲を絞って述べる。また，同時に，システム化のための道具についても触れることとする。

コンピュータ・システム構築のためのシステム化技術は，要素となるいろいろなサブシステム群，すなわち，ソフトウェア・モジュール群，マシンを構成するハードウェア・モジュール群，これを作るためのＶＬＳＩのような部品群，さらには，コネクタやケーブルのようなより基礎的な部品・材料群などの，設計・開発，および，それらを組立てるに必要とする技術と考える。

これは，部品の選択のための評価手法，各モジュールや部品の仕様の標準化，組立てる場合の規則の統一化，目的のものができたか否かの検証，評価の手順の明確化といった形で，具体化される。さらに，これらの規則や手順に従ったものが，スムーズに作られ，実現されるように，各種の工具や開発支援ツールが開発される。

コンピュータの開発においては，その開発支援ツールは，やはりコンピュータであり，開発目標に至る過程では，その中間的性能を有するコンピュータが，何段階かにわたって作られる。これにより，少しずつ性能の高いものを作っていくことになる。これは，まさに，自己増殖のプロセスであり，この過程を，効率よく管理していくのもシステム化技術である。

以下，このような考え方に基き，第５世代コンピュータ・システムの開発に必要なシステム化技術について述べる。

〔2〕 必要性

システム化技術は，広い概念であり，いくつか，要素に分けて考える必要がある。ここでは，次の４つに分け，その必要性を述べる。

まず，第１は，仮想化の技術である。知識情報処理を目ざす第５世代コンピュータ・システムでは，そのソフトウェアが，きわめて重要な位置を占め，その規模も大きなものとなる。また，それを支えるハードウェアも複雑で，高機能のものとなる。このため，ソフトウェアで構成する論理マシン（仮想マシン）とハードウェアで構成する物理マシン（実マシン）の間を結ぶ仮想化の技術が必要となる。

第2は，いくつかのアーキテクチャ関連技術の融合化技術である。この中には，ソフトウェアも含まれる。アーキテクチャは，機能分散を軸としたアーキテクチャ，ＶＬＳＩを軸としたアーキテクチャ等，いくつかの異る軸をもつアーキテクチャが存在し，これとソフトウェアを組合せた上での，融合化技術が必要である。この中には，応用目的とする仕事をどのように分割し，どのアーキテクチャでサポートするかといった，いろいろなレベルの負荷分散，評価の手法も含まれる。

第3は，大規模，高信頼化システムの設計開発技術である。大規模なハードウェア，ソフトウェアの構築には，人間の管理能力を越える数の部品やモジュールを用いねばならない。また，その信頼性を高めるためには，その組立てに際し，いろいろな技術を必要とする。このような条件のもとでの開発では，まず，システィマテックな構成手法の開発と，その機械化（コンピュータ化）が不可欠となる。また，新しい手法の開発には，試行錯誤的プロセスが必然的に含まれる。このような技術は，未知の領域を多く含む第5世代コンピュータ・システムの開発では，きわめて重要である。

第4は，システム化技術を効果的に適用するための開発ツールとしてのコンピュータ・ネットワークである。ネットワークは，研究開発グループ間の情報交換，研究開発成果の交換に有効なだけでなく，互に，ソフトウェアやハードウェアのモジュール・インタフェースの統一に有効である。ネットワークの各インタフェース仕様に従わねば，用いることができないから，いろいろなレベルでの仕様の標準化がすすめられる。また，ネットワークに付加される開発支援用コンピュータについても，言語の統一化等をはかることが必要である。

以上述べたような技術を重点項目として，システム化技術の開発を行うことが必要である。

〔3〕内外技術の現状

(1) 国内技術

国内においては，研究開発に重点を置くシステム化技術の蓄積は，十分とは言いがたい。コンピュータの本体を中心とするシステム化技術は，近年，かなりの進展を見せ，アメリカを中心とする欧米の技術水準に伍するまでとなってきた。しかしながら，ローカルおよびグローバル・ネットワーク技術については，日本では，電電公社主体で行われているもの以外は，制御用コンピュータシステム等で用いられているものが主であり，立ち遅れが目立っている。特に，ネットワーク用のソフトウェアは，近年，開発がすすめられたばかりであり，そのノウハウ等の蓄積も十分とは言いがたい。

また，大規模ハードウェアの構築用のＣＡＤシステムなどは，現在は，多くがアメリカよりの輸入に頼っており，進歩的なＣＡＤになる程，この傾向が強い。

ソフトウェアの構築に関するシステム化の技術は，多くが今後の研究テーマであるが，この面でも，立ち遅れが目立ち，今後の精力的な研究が必要である。

(2) 海外技術

海外技術の中心は，やはり，アメリカである。仮想化や融合化の技術，大規模システムの構築の技術とも，公的な研究機関（ＡＲＰＡやＤＯＤ，大学），メーカー（ＩＢＭ，ＤＥＣ，Ｘｅｒｏｘ 等）の双方において，よく研究されている。

ソフトウェアの仮想化は，ネットワーク技術と関連して研究されており，ＡＲＰＡネットワーク等で実証されている。また，小規模なローカル・ネットワークにおけるシステム化技術も，Ｘｅｒｏｘの Ｅｔｈｅｒ のネットワーク等で実証されている。

大規模なソフトウェアのシステム化技術は，ＩＢＭを中心とするメーカーで実際的な研究が行われ，大学等では，ソフトウェア工学等の基礎的な研究が行われており，多くの成果が報告されている。ＶＬＳＩを目的とするＣＡＤシステムも，ＣＡＬＭＡ社を始めとするメーカーで，実用的で優れたシステムが開発されており，より高度なアーキテクチャの設計から，マスク・パターンまでの一環した設計手法とそのシステムについては，ＭＩＴ，ＣＭＵ等で，よく研究されている。

我国でも，ネットワークから，ＣＡＤシステムまでの一連のシステム化技術の開発を強力にすすめる必要があろう。

〔4〕 研究開発内容

(1) 研究開発項目

① 仮想化とシステム構成技術

プログラミング言語，オペレーティング・システム，ネットワーク・インタフェース，アーキテクチャ・レベル，機能モジュール・レベル，ＶＬＳＩを含むコンポーネントのレベルのそれぞれの階層において，基準となるべき階層間のインタフェース仕様を決定する手法を開発する。この際によりどころとなる要素間接続の論理モデル（たとえば，ＯＳＩ標準モデルのようなものや，ＡＣＴＯＲモデルのようなもの）の研究を行う。これら技術によって，ソフトウェアの研究開発成果を応用目的に応じて，柔軟に選択構成するための手法の開発及び，その標準化を行う。

② 融合化と負荷配分の最適化技術

知識情報処理を目的とした応用システムの実現に際し，機能分散システム上への負荷配分と，これを行うために，ソフトウェア，新アーキテクチャに基づくマシン，その他の要素マシン，マシン中の機能モジュールのそれぞれの間での，役割と機能の分担を決定する手法について研究を行う。また，その負荷の配分の最適化や，システム構成についての評価のための手法を開発する。

③ 大規模システム設計・開発技術

新アーキテクチャに基づくマシンの設計開発のための方針や，アーキテクチャ設計のた

めの自動化ツール（ＣＡＤシステム），評価ツール，回路実装のために必要な部品，材料についての研究，開発を行う。また，大規模なソフトウェアの開発に必要な要求仕様技術，開発・評価ツール，高品質化のための技術の研究開発を行う。ソフトウェア，ハードウェア，ＶＬＳＩまでの一環した設計開発支援システムをネットワーク上に開発する。

④　超高信頼化技術

ソフトウェア，ハードウェアの高信頼化のための技術を開発する。このために，ロジック・レベル，モジュール・レベル，サブシステム・レベル，システム・レベルのそれぞれにおける耐障害化のための設計・実装技術を研究開発する。

⑤　アーキテクチャ開発ツールとしてのローカル・ネットワークの開発

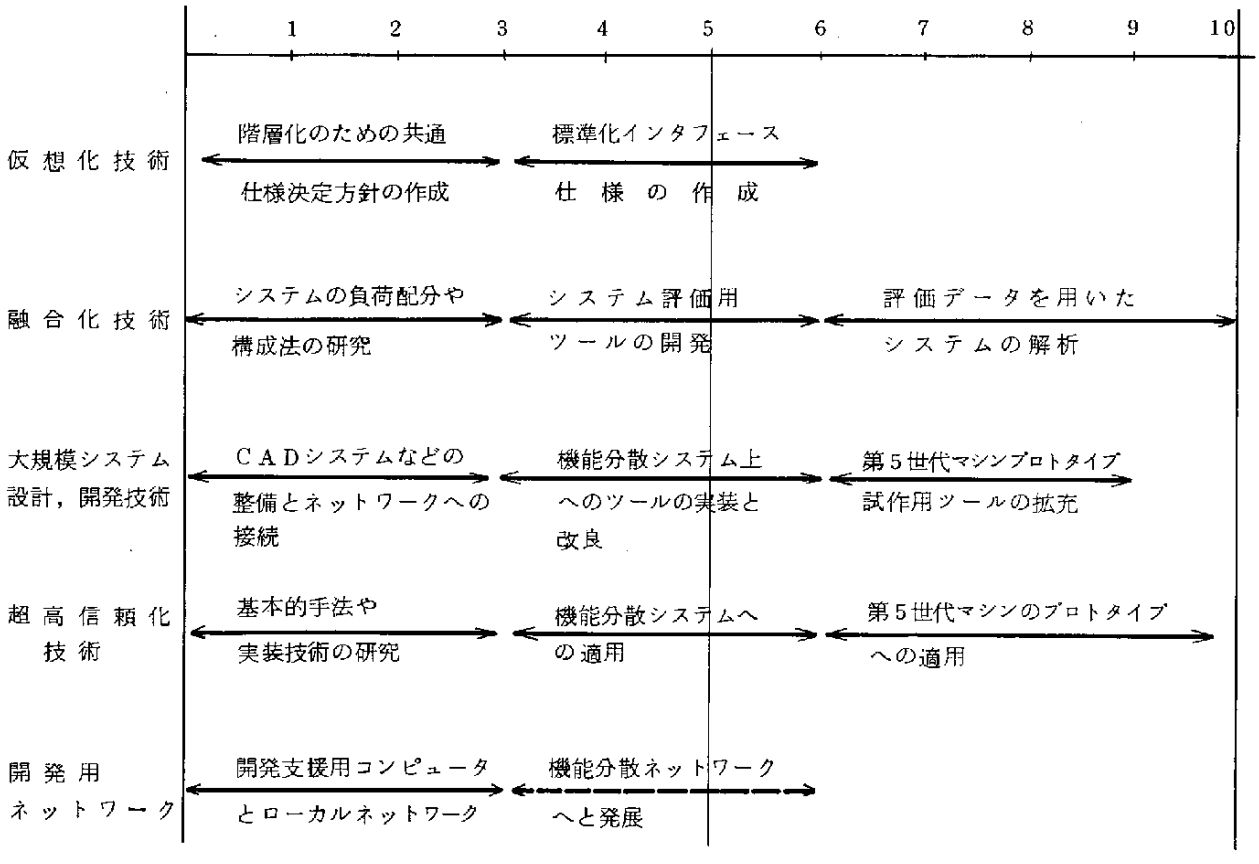
ハードウェアのシミュレーション，実験機の試作，プログラミング言語や処理系の開発，ＶＬＳＩ等のＣＡＤシステムの開発などの支援のためにローカル・ネットワークを開発する。各種の実験・試作・評価は，このネットワークを用いて行う。特に，ハードウェアとソフトウェアを組合せた総合評価は，ネットワークが有効なツールである。

ローカル・ネットワーク間は，適宜，グローバル・ネットワークで接続する。

(2)　研究開発スケジュール

開発初期の段階で，シミュレーションやハードウェア試作のワーキング・ベンチの目的で

表６－１４　第５世代アーキテクチャのシステム化技術の研究開発スケジュール



使用するコンピュータを準備するとともに，コンピュータやＣＡＤシステムなどの間を結ぶローカル・ネットワークを付設する。

仮想化や融合化の手法については，初期にその基本方針を示し，これに従ったソフトウェア，ハードウェアの開発を促す，開発の進展に伴い，仕様の標準化などを行い，研究成果の幅広い利用と，構成の柔軟性を保障する。大規模システムの設計，開発については，自動化をすすめ，短いターン・アラウンド時間で設計や変更の行えるインタラクティブ・システムを開発していく。ＶＬＳＩのＣＡＤシステムを含め，これらのツールは，ネットワークを介し，多くの研究者に利用できるようなものとする。高信頼化技術は，ＶＬＳＩを含むハードウェアの能力を十分に生かす技術を開発し，設計開発に役立てる。

(3) 研究開発の目標・仕様

① 仮想化技術

開発支援マシンおよびローカル・ネットワーク上へ作られる実験用システム，中間目標システムのモジュール間インタフェースの標準化，統一化のための作業を行い，インタフェース仕様の標準化を行う。

また，この標準化仕様を普及させるためのマニュアル，手引書などを作成する。

このほか，システムの階層化の規準を作る。

② 融合化技術

シミュレーション等により，負荷配分，システム構成方法などの解析を行う。開発支援マシンおよびローカル・ネットワーク上に，マシンの負荷の測定ツールや，ネットワーク上のトラヒックの測定ツールを付加し，システム評価用のシステムを開発する。

③ 大規模システム設計開発技術

ソフトウェアの開発を容易にする各種ユーティリティ・プログラムを開発する。また，開発されたソフトウェアの蓄積，交流をはかるため，ソフトウェア・モジュールとそのドキュメント類も含めて貯えられる集中データベース・システムを開発する。これには，ハードウェア開発に用いるシミュレータ等のソフトウェアやハードウェアの説明書等も蓄えるものとする。

ハードウェアの開発に関しては，種々のシミュレータや評価ツールのうち，共通に用いられるようなものを開発する。回路等の記述言語や処理系，シミュレータなどが，この対象となる。また，ＣＡＤシステム等も一部開発する。

④ 超高信頼化技術

高信頼化のためのソフトウェア，ハードウェアの実装手法を開発し，各種システムにその基本要素となるモジュール，部品等を供給する。

⑤　開発用ネットワークと開発支援マシン

研究開発の初期にまず既存技術によるネットワークを設備する。また，開発支援マシンを開発もしくは，購入して，研究開発ツールとして提供する。ネットワークとしては，ローカル・ネットワークおよびグローバル・ネットワークを設備する。

初期の開発用ネットワークのローカル・ネットワークとしては，データ転送速度１～５Ｍbits／ｓのものを，グローバル・ネットワークとしては，公衆回線を利用し，データ転送速度５～１０Ｋbits／ｓのものを設ける。①～④で開発したシステム化のためのツールは，すべて，このネットワークへ接続して使用できるものとする。

6.3　開発の目標と開発のステップ

6.3.1　開発の目標像

第５世代アーキテクチャ関連研究の目標は，知識情報処理のための数々のソフトウェア群のサポートを行い，実用に供し得る能力を有するハードウェア・システムのプロトタイプを開発することである。

開発されるハードウェア・システムは，新アーキテクチャに基づくいくつかの機能別マシン群，開発支援を目的として開発されたパーソナル・コンピュータをはじめ，データベース・マシン等の開発支援マシン群，これらを結合する機能分散アーキテクチャに基づくローカル・ネットワークシステムから構成される。

このネットワークは，ソフトウェアと組合されて，分散データベース等も含む知識情報処理ネットワークを形作る。

構成要素となるマシン群は，新しい核言語を共通の言語インタフェースとして持つことを原則とするが，ユーザ・レベルの言語としては，その応用目的に適合したいくつかの言語が用いられる。

知識情報処理の中核となる推論マシンや知識ベース・マシンは，新アーキテクチャの研究開発課題となっているマシンの研究成果を融合した結果得られるものから成っている。各種マシンのうちで，高性能のものは，データフロー型実行機構等の並列処理技術を多用したものと考えられる。また，パーソナル・マシン等では，小型化を優先したファームウェア・ベースのアーキテクチャが用いられると考えられる。

このような高い機能を持つマシンを構成要素とする機能分散システムの構成技術は，これら機能別マシン群の開発と並行して研究され，新しい機能分散ネットワークとして実装されることになる。また，各マシン内部の構成にも，新しいモジュール化やダイナミック・アーキテクチャの利用等，柔軟性や適応性の付加，高パーフォマンス化を目ざす技術が駆使される。

ＶＬＳＩ技術は，まずＶＬＳＩアーキテクチャの研究により，ＶＬＳＩ向きアルゴリズムやアーキテクチャ，回路構成法等が追求され，いくつかの記述言語により，アルゴリズムから，

マスク・パターンに至る各レベルでの設計対象の記述が蓄積され，アーキテクチャ・データベースが作り上げられる。これは，ＣＡＤシステムの基盤となり，最終的には，知的ＣＡＤシステムとして整理体系化され，実用に供される。この開発の途上においても，新アーキテクチャ実験機の開発等に用いるＶＬＳＩチップの製造をサポートする。

以上のような目標のより細かな仕様は，新しい言語や処理系の研究や分散ＯＳの研究，さらには，その上に展開されるより高度な推論システムを用いる音声，自然言語理解システムの研究など，ソフトウェアや基礎理論の研究の結果と照し合せて決定されるべきものである。

本プロジェクトは，多くの未知の研究分野を含むことから，いくつかチェックポイントを設け，その時点での研究成果の評価と目標の見なおしを行いつつ，目標の精密化を行うのが適当と思われる。

6.3.2　開発ステップ決定の留意点

１９９０年初頭をめざした場合の，アーキテクチャの研究開発のすすめ方は，次のようないくつかの点を考慮して決定されるべきである。

①　この計画は，１０年間という長い将来を目ざしたものであり，また，その目標も，知識情報処理とＶＬＳＩを中心としている。これらは，未知の要素を多く含むため，当然のことながら，研究の進展状況に応じた見直しを必要とする。

②　計算機システム全体としては，ソフトウェア工学や人工知能研究などの成果，ネットワーク・アーキテクチャや通信などの研究成果も，アーキテクチャやＶＬＳＩなどの研究と共に重要な要素となる。これらの研究開発成果が，相互に有機的に結合されるような開発方法が必要である。

③　アーキテクチャの研究開発においても，まず，アーキテクチャ研究の道具となるようなものを先行して進め，その上で新アーキテクチャ等の実験試作ができるようなステップをとるべきである。

④　知識情報処理に重要な推論や知識ベースの構築等第５世代の中核となるソフトウェアや新アルゴリズム等の研究を，早期に開始させるため，これに必要な研究開発用のツールも，先行して開発する必要がある。

⑤　従来からのソフトウェア資源の蓄積を生かし，さらに，研究開発が開始された後においても，そこで開発されたソフトウェアが，順次蓄積され，有効利用され，拡大再生産されるような組織ならびに，道具を整備すべきである。これは，既存アーキテクチャから，新アーキテクチャへのスムーズな移行性の問題でもあり，機能分散に基づくローカル・ネットワークが，最も有効な解決策である。

⑥　ＶＬＳＩの製造技術は，計画全体の基礎となるものであるから，新アーキテクチャや機能分散アーキテクチャとの関連を明確に設定し，順次，実用的な部品を短いターン・アラウンド・タイムで試作できるよう整備する必要がある。

⑦ アーキテクチャの研究開発は，常に実現可能性（Ｆｅａｓｉｂｉｌｉｔｙ）を前提とし，応用面からの要求と，実現のための技術（素子技術，実装技術）やコスト（開発費，人件費，保守費，その他）のバランスの上に成立する。研究開発の進み方に従って，これらトレード・オフの問題も絡めての研究管理が必要である。

⑧ 新アーキテクチャの開発には，多くのシミュレーションや試作，および評価の過程，および，そのソフトウェアの研究・試作の過程が必要であり，それ自身，多くの計算パワー，多くの人間資源，研究開発投資，研究の管理，融合化のための組織を必要とする。これらを含む体制の早期確立も重要なテーマである。

以上，あげたもののほかにも，いろいろな問題点があるが，開発対象が，多くの未知の要素を含むことから，最初は，なるべく多くの課題にとり組み，その研究の進展により，その先が予測できるようになるに従って，統合していくことが中心的ポリシーであろう。

実現のステップとしては，上記①～⑧に述べた問題点の中にもあるように，課題間の順序関係を考慮しつつ，当面の中間目標をたて，その時点までに，できる限り多くの未知の要素を解明するよう努力することが妥当であろう。

6.3.3 開発ステップ

アーキテクチャ関連研究開発課題の開発のステップを，プロジェクトの全期間を３期に分けて述べる。これらを，前期，中期，後期と呼ぶこととする。

各期において行うべき主要な研究，開発の項目と，重要な方針についての概要は，表６－15に，まとめて示した。この表では，アーキテクチャの課題と密接に関連する基礎ソフトウェアのグループに含まれる課題も示してある。

以下，アーキテクチャ関連の課題を，新アーキテクチャ，機能分散アーキテクチャ，ＶＬＳＩアーキテクチャ，システム化技法のそれぞれの課題グループごとに示す。

１）前　　期

Ａ１）新アーキテクチャ

① この期間は，新アーキテクチャの候補となっている各マシンについて，シミュレーションや，ハードウェアの実験機（ハードウェアシミュレータ）の試作により，設計用基礎データの収集を，主に行う。このためには，既存の大型機等を主に用いる。

② 知識情報処理用ソフトウェアの基礎研究の開発支援を目的とする開発支援マシンを開発する。これは，ＰＲＯＬＯＧやＬＩＳＰをサポートし，ファームウェア・ベースのマシンで，新ノイマン技術により製作する。

Ａ２）機能分散アーキテクチャ

① 機能分散システムの設計手法確立のため，実験モデルの作成，シミュレーションに

表6－15　開発のステップの概要

	1　2　3	4　5　6	7　8　9　10
T1 基礎ソフトウェア	①ＰＲＯＬＯＧを核とする核言語ゼロ次等の決定→（Ａ1－③）〈初期〉 ②基礎ソフト各テーマの基礎研究と第1次仕様決定→（A′1－①②） ③一部システムの試作，データ収集	①核言語の改良，拡張を行う。また，マシンのソフトウェア側からの最終仕様決定（第2次仕様）→（Ａ〃1） ②関係型データベース・マシン上に知識ベース管理システムのプロトタイプを試作する。 ③知的インタフェースは，開発支援マシン上にプロトタイプを開発し提供→（Ｔ2～Ｔ3）専用ハード仕様→（A〃2－②）	①知識ベース問合せ機能などの追加による核言語の拡張と処理系の開発 ②最終システムの開発と第5Ｇマシン上への移行と一体化
A1 新アーキテクチャ	①各候補マシンのシミュレーション等による評価，基礎データ収集 ②一部，実験機試作 ③ファームウェア・ベースのＰＲＯＬＯＧ，ＬＩＳＰマシンを開発支援マシンとして開発→（Ａ4－③）	①前期評価とＴ1－②による候補の融合 ②実験機の中規模試作（ＶＬＳＩ第1版使用）ソフトウェア試作のサポート ③ファームウェア・ベース開発支援マシンの拡張とＶＬＳＩによるパーソナル化→（A′4－③）	①候補マシンの最終の絞り込みとその仕様を決定する。 ②プロトタイプを開発し，ソフトウェアと結合する。第5Ｇプロトタイプ・マシン（ＶＬＳＩ第2版使用）
A2 機能分散アーキテクチャ	①機能分散の設計手法の確立 ②分散型ＯＳやローカル・ネットワーク技術の基礎研究 ③開発支援ローカル・ネットワークの拡充→（Ａ4－①）	①開発支援マシンを含む機能分散ネットワーク（スーパ・パーソナル・コンピュータ・ネットワーク）の開発→（A′4－①） ②分散型ＯＳや関連ソフトウェア，データベース・マシンなどハードウェアの開発 ③ＵＨＭなどの試作とパーソナル化サポート	①第5Ｇプロトタイパ・マシンを中核とする機能分散ネットワークに拡張する。（知識情報処理ネットワーク） ②知的インタフェース用ハードウェアを開発する。
A3 ＶＬＳＩ技術	①カスタム化ＬＳＩによる新アーキテクチャ研究サポート→（Ａ1－②③） ②ＶＬＳＩ－ＣＡＤのためのソフトウェアの研究と試作とＣＡＤ整備 ③ＶＬＳＩ向きアーキテクチャの基礎研究	①カスタム化ＶＬＳＩによる新アーキテクチャ研究サポート→（A′1－②③）（A′2－③） ②ＶＬＳＩ－ＣＡＤシステムの開発とその提供 ③ＶＬＳＩ向きアーキテクチャのアーキテクチャ・データベース開発	①カスタム化ＶＬＳＩによるプロトタイプ開発サポート→（A〃1－②）（A〃2－③） ②知的ＶＬＳＩ－ＣＡＤへの拡張 ③ＶＬＳＩアーキテクチャの知識ベース化
A4 アーキテクチャにおけるシステム化技術	①開発支援ローカル・ネットワークの設置と開発支援マシンの接続 ②公衆回線による簡易グローバル・ネットワークの開設 ③ＰＲＯＬＯＧ，ＬＩＳＰマシンの開発支援ツールとしての拡充 ④ＬＳＩ－ＣＡＤの整備	①機能分散アーキテクチャによるローカル・ネットワークの設置（ローカルネットの拡張と更新） ②公衆回線によるグローバルネットの拡充，データベース・ステーションの設置 ③核言語（第1版）を持つスーパ・パーソナル・コンピュータによる開発支援 ④ソフトウェアやハードウェアのモジュール化やインタフェース決定の標準化や規格の決定を行い，研究成果の交流を促進する。	①機能分散システムとローカル・ネットワークの拡張によるシステム化のサポート ②グローバルネットの拡充 ③システム評価手法や評価ツールの開発と提供

よる評価等を行う。

② 開発支援ローカル・ネットワーク上において，分散ＯＳや通信制御システム等のプロトタイプ・システムを作成し，機能分散型ネットワーク開発の基礎研究を行う。

③ 高解像度ディスプレイ，図形入力などの機能を有するパーソナル・コンピュータ・システムを開発する。また，その他の機能分散型ネットワーク構成要素マシンの仕様を決定する。

Ａ3） ＶＬＳＩアーキテクチャとＣＡＤ

① カスタム化ＬＳＩを製作する環境を整え，新アーキテクチャ実験機や，開発支援用マシンの開発をサポートする。

② これと同時に，カスタム化チップ作成用のソフトウェア，ハードウェアも開発・整備し，ＶＬＳＩ－ＣＡＤ開発のためのデータ収集や，ソフトウェアの蓄積を行う。

③ ＶＬＳＩ向きのアーキテクチャの基礎研究を行う。

Ａ4） システム化技法

① 開発支援を目的とする既存技術によるローカル・ネットワークを設備する。ここに，開発支援用ＰＲＯＬＯＧ／ＬＩＳＰマシン，既存マシンを接続して使用する。

② 公衆回線利用のグローバル・ネットワークを設備し，いくつかの研究サイト間を接続する。

③ ＬＳＩ－ＣＡＤは，開発支援ローカル・ネットワークへ接続し，研究者に提供する。

＊この時点で，アメリカのＸｅｒｏｘやＭＩＴの持つ，Ｅｔｈｅｒネット，ＣＨＡＯＳネットに類するローカル・ネットワークを設備する。

＊ＬＳＩの作成は，アメリカで行われているように，標準化パタン記述言語ＣＩＦのようなものを規定して行う。

2） 中　　期

A′1）

① 基礎ソフトウェアの研究により決定される言語仕様に基づいて，新アーキテクチャによるマシンの実験機（中間目標機）の開発を行う。これに際しては，カスタム化VLSIの第１版を使用する。

② このとき，新アーキテクチャによるマシンの研究成果を融合し，マシンの候補を６種から３種程度に絞り込む。

③ 前期に開発した開発支援用ＰＲＯＬＯＧ／ＬＩＳＰマシンのＶＬＳＩ化を行い，小型のパーソナル・コンピュータとし，性能も向上させる。

④ 新アーキテクチャ・マシンの一部は，ソフトウェアの研究用に提供する。

A′2) 機能分散アーキテクチャ

① 前期の研究成果に基づき，機能分散ネットワークを開発する。これにより，それまでの開発支援ローカル・ネットワークを拡充，更新する。また，分散ＯＳなどのソフトウェアの基本部分を実装する。

② 開発支援用のパーソナル・コンピュータやデータベース・マシン等も接続し，スーパ・パーソナル・コンピュータ・ネットワークとする。また，グローバル・ネットワークも拡充する。（パケット交換網などの利用）

③ ひき続き，機能分散ネットワークのソフトウェアの研究や，ユニバーサル・ホストマシン等のマシン・コンポーネントの研究等を行う。

A′3) ＶＬＳＩアーキテクチャとＣＡＤ

① カスタム化ＶＬＳＩ用のＣＡＤシステムを開発し，新アーキテクチャや機能分散アーキテクチャの研究開発をサポートする。

② ＶＬＳＩアーキテクチャの研究に基づき，機能モジュールや回路，マスク・パターン等の記述言語や処理系の開発をすすめる。また，これらにより記述したものを，蓄積し，体系化を行って，知的ＶＬＳＩ－ＣＡＤのためのアーキテクチャ・データベースの開発をめざす。

A′4) システム化技法

① ソフトウェアやハードウェアの研究開発成果の交流を促進するため，モジュール化手法やそのインタフェースの規格化，標準化などを行う。

また，これら成果を蓄積するデータベース・システムと管理体制を設備する。

② 機能分散ネットワークやグローバル・ネットを開発支援ツールとして用いる。また，パーソナル・コンピュータ（ＰＲＯＬＯＧ／ＬＩＳＰマシン）の周辺を整備する。

3) 後　　期

A″1) 新アーキテクチャ

① 基礎ソフトウェアの研究成果として開発される新しい言語仕様（最終案）に従って，新アーキテクチャによるマシンのプロトタイプを開発する。

この際には，中期の成果に基づき，候補マシンを１～２種に絞る。また，このマシンを規模別に段階ずけし，応用ソフトウェアに応じたものを，いくつか開発する。

② このプロトタイプのマシン群の開発には，ＶＬＳＩの第２版を使用する。

③ 最終仕様に基づくソフトウェアと組合せて，応用システムとしての評価と改良を行う。

A″2) 機能分散アーキテクチャ

① 中期で開発した機能分散ネットワークを拡張し，新アーキテクチャに基づくマシンを中核とする知識情報処理ネットワークを構築する。

② 音声，図形，自然言語などの処理を行う専用ＶＬＳＩチップや専用プロセッサを開発する。これは，基礎応用システムの研究に基づき決定された仕様に従って行う。

③ 知識情報処理のためのソフトウェア群の機能分散ネットワーク上への実装を行う。また，これらを含めた評価と改良を行う。

A″3） ＶＬＳＩアーキテクチャとＣＡＤ

① 新アーキテクチャのプロトタイプ開発に必要なＶＬＳＩ第２版をＶＬＳＩ－ＣＡＤシステムにより製造し，供給する。

② 音声，図形等の処理用の専用チップの開発も，同様にサポートする。

③ ＶＬＳＩアーキテクチャ・データベースの拡充をすすめ，ＣＡＤシステムと結合し，さらに，知識ベースの研究成果をとり入れて知的ＶＬＳＩ－ＣＡＤシステムのプロトタイプを開発する。

また，ここにおいて，アルゴリズムの設計，プログラムの設計，アーキテクチャの設計，回路設計，パターン設計の一連のコンピュータ設計の一体化を行うことをめざす。

A″4） システム化技法

① 開発されるプロトタイプのマシン間でのモジュールや入出力チャネル等のいろいろなレベルでのインタフェース仕様の統一化など，構成の柔軟性を増すような規格化をすすめる。

② 開発成果の整理・体系化を行い，データベース化を行う。

③ 開発されたシステムの評価手法や評価ツールを開発し，提供する。

6.4 開発のすすめ方と研究体制

第５世代コンピュータ・システムの研究開発の特徴は，知識情報処理システムというきわめて汎用的なコンピュータ・システムを目指す先進的，かつ野心的なプロジェクトであるという所にある。

この中では，機械翻訳や音声理解のような応用システムから，ＶＬＳＩのような部品のレベルに至るまで，多くの先端的研究テーマが，網羅されている。

これらのテーマの多くは，未開拓の分野を多く残しており，その中では，従来の日本では，研究実績の乏しい基礎理論やソフトウェアの研究が大くの部分を占めている。ハードウェア・システムも，ソフトウェア指向型の性格が強い。

このため，アーキテクチャ関連の研究開発課題に注目した場合でも，従来と異なる研究のすすめ方と，ツールが必要となる。

このうちのいくつかを以下に述べる。

1） ネットワークを前提とした開発

第５世代においては，現在の分散処理を基本とする処理形態が，普遍化すると考えられ，こ

のため，各コンピュータ・システムのソフトウェア，ハードウェアは，ネットワークのノードとして使用されることを前提として，開発される必要がある。

このためには，開発の初期から，ネットワークを設備し，これを成長させる形で，最終目標にアプローチすることが望ましい。初期においては，ネットワークは，まず，ローカルネットワークを基本とし，これに種々の開発支援用コンピュータを付加して使用する。この結果，この上で蓄積されるソフトウェアは，ネットワークを前提としたものとなる。

最初，単独システムとして開発し，後から接続することは，複雑なソフトウェア，ハードウェアを持つ第5世代コンピュータでは，不可能に近い。

2) ソフトウェアとハードウェアを組合せた開発，評価体制の整備

従来のコンピュータ・システムと異なり，第5世代コンピュータでは，ソフトウェアの負担をどこまでハードウェアが分担できるかという点が重要な問題となる。これは，ハードウェアをどこまで高レベル化できるかを，試行錯誤的に試みるという手続きをくり返すことで研究がすすめられる。

このためには，ソフトウェアの開発グループとハードウェアの開発グループ間での情報の交換を迅速に能率よく行う環境を準備し，さらに，これを活発化させる管理体制が必要である。道具としては，まず，ネットワークにより，双方のグループの道具として使用するコンピュータを接続しておけばよい。しかし，これと同時に，マニュアル等のドキュメンツの整理や，相互に利用できるような使い易いシステム作りを心がけねばならない。

このようにすることで，ハードウェアの研究者は，ソフトウェアの研究者の開発した言語や処理系を介して，自分の開発したハードウェアを用いることができ，より能率よく，ハードウェアの評価が行えるばかりでなく，ソフトウェア，ハードウェアの双方を組合せた上での評価も行えることとなる。これは，ソフトウェアの研究者にとっても，同様の利益をもたらし，より優れたシステムを作るアイデアも生まれやすい。

このほか，ネットワークを介して，別の評価専門のグループを設けることで，より使いやすいソフトウェアへと改善していくことも可能であろう。特に，ソフトウェアは，多くの人に使いこまれることで，改善されていく度合が大きく，このような体制は，有効であろう。

3) 研究成果の管理と交流

第5世代コンピュータ・システムの開発では，多くのソフトウェアが生産される。ハードウェアの研究においても，シミュレータや，ＶＬＳＩのための記述言語や処理系が，各種開発されよう。同じようなものの開発の重複を避けるとともに，これらのソフトウェアの有効利用をはかるためには，この管理と交流のための体制と道具が必要である。道具としては，ネットワークとこれに接続されたデータベース・システムを設備することで十分であろう。管理体制としては，生産したソフトウェアと，その使用法を説明するドキュメンツをデータベースに登録

させることを義務化することが重要である。また，登録されたソフトウェアの宣伝も必要である。日本語ワードプロセッサの利用の促進等も重要であろう。実質的作業の機械化は容易であるから，研究者への習慣づけが課題である。

これに類する例は，アメリカでは，ＡＲＰＡネットワークを介して広く行われており，各研究者が，これにより得られるメリットはきわめて多い。

以上，研究をサポートする道具と管理の方法を簡単に述べた。このほかにも，研究者の交流等の点で，考慮すべき事は多い。

多くの研究者の知恵をいかにして集めるか，また，アイデアの生れ易い環境をいかにして作るかは，研究体制，管理・運営の方法に多くを負っており，今後の重要な課題として議論すべきものと思われる。

— 禁 無 断 転 載 —

昭和56年3月発行

発行所　財団法人　日本情報処理開発協会
東京都港区芝公園3－5－8
機械振興会館内
TEL　(434) 8211 (大代表)

印刷所　株式会社　正　文　社
東京都文京区本郷3－38－14
TEL　(815) 7271

55－R016