5. 逐次型推論マシン(SIM)：その研究開発経過

ICOT研究所第3研究室長代理　内田俊一
ICOT研究所第3研究室長　横井俊夫

アブストラクト　第五世代コンピュータ・システム（FGCS）プロジェクトでは，核言語に論理型言語を採用しており，これが目標とするコンピュータ・システムのソフトウェア，ハードウェア両方の概念的枠組を定めている．従って，プロジェクト当初からロジック・プログラミングのための効率の良いプログラミング環境を，研究者に提供する必要があり，新しいツール，すなわち，新しいコンピュータ・システムの開発が必要となる．

この新システムは論理型言語に基づく核言語第0版（KL 0）と呼ばれる高級マシン語を，逐次的に実行するもので，逐次型推論マシン（SIM）と呼ばれている．

このシステムの開発の為にサブプロジェクトが計画され，これは SIM プロジェクトと名付けられている．このプロジェクトには，パーソナル逐次型推論マシン：PSI のほか，ローカル・エリア・ネットワーク・システム：ICOT-Net やプログラミング/オペレーティング・システム：SIMPOS 等，ハードウェア，ソフトウェア双方の開発が含まれている．

本稿では，アーキテクチャ，及び，ハードウェア・システムの開発を中心に，SIM プロジェクトの研究開発経過について述べるほか，ソフトウェア・システムから要求される KL 0 の機能とハードウェア・システムへのフィードバックについても，簡単に触れる．これは，SIMPOS，及び，そのシステム記述言語である ESP を含むソフトウェアシステムが，アーキテクチャと深く関わっており，その為，これから要求されるフィードバックは，アーキテクチャとハードウェア・システムに対してだけでなく，KL 0 の機能に対しても拡張と変更を要求するからである．

又，本稿では，中期の研究開発計画についても述べる．

1. 概　　要

1.1 動機

FGCS プロジェクトは，基本的な言語として論理型言語を採用した．その理由は，論理型言語が，第5世代におけるソフトウェア・システムとハードウェア・システムとのインタフェイスとして最も適していると考えられるからである．すなわち又，論理型言語は，高度の並列アーキテクチャと VLSI 技術を使った高性能のハードウェア・システム，及び，知識情報処理システム向きの高度なソフトウェア・システムを実現する基礎として，最適と思われる．このような条件のもとで研究開発を行うにあたっては，いくつかのツールがきわめて重要である．特に，効率の良いプログラミング環境は，必須であり，その為，論理プログラミングをサポートするツールとなる新しいコンピュータ・システム開発のサブ・プロジェクトが計画された．

このコンピュータ・システムは，逐次型推論マシン―SIM―と名付けられ，そのサブプロジェクトは，SIM プロジェクトと呼ばれている．SIM プロジェクトは，FGCS

の10年間のプロジェクトの前期における最も重要なサブプロジェクトであり，そのゴールは，研究者に効率の良いプログラミング環境を提供し，数多くのソフトウェア実験を可能にするような，ソフトウェアとハードウェアのシステムを開発することである．

1.2 必要条件

SIMシステムでは，ソフトウェア・システムとして効果的で使い易いプログラミング環境を実現するとともに，ハードウェア・システムが，充分な処理能力と記憶容量を有することが必要である．

処理能力と記憶容量として，ハードウェア・システムは，少なくともDEC 2060に搭載されている世界最高速のソフトウェア・プロセッサである，DEC-10 Prologと同等の処理能力をもつ必要があり，ソフトウェア研究者の中からは，大規模なソフトウェア実験の為，もっと速いハードウェア・プロセッサを望む声も出ている．

この様な基本的な要求を満たすためには，スーパー・パーソナル・コンピュータに高度のマンマシンインタフェースを備えたシステムが最良と考えられる．そのようなシステムのソフトウェアは，プログラム開発の生産性および信頼性を高めるために，新しい機能を数多く持つことになるのであろう．そうして，これらの機能を実現するソフトウェア・サブシステムとして，マウス付きのマルチ・ウィンドウ・システムや効率の良いエディタ，使い易いデバッガ，ソフトウェア・モジュールの管理を行うライブラリアン・コーディネイタ（あるいは，セッション・マネージャ）などがある．

パーソナル・コンピュータは，又，メッセージやプログラム等を交換するためのローカル・エリア・ネットワーク・システムを持っていなければならない．従って，ソフトウェア・システムも，ユーザが，他のコンピュータの処理や，離れた所にあるファイルにアクセスできるようなネートワーク・サブシステムを備えたものとなる．

この様な高度で複雑なソフトウェア・システム，即ち，プログラミング/オペレーティング・システムを実現するには，高機能システム記述言語が必要である．システム記述言語で書かれたプログラムは，機械語にコンパイルされ実行される．

FGCSプロジェクトが実質的に活動を開始したのは，1982年6月であるが，それ以前に，上述した様なソフトウェア，ハードウェア双方のシステムに関する必要条件を調査し，研究開発計画のアウトラインが作られた．

1.3 研究開発のスケジュール

SIMシステムは，プロジェクトの標準ツールとして使用されるべきものであり，プロジェクトの中期には使用可能でなければならない．従って，ソフトウェア，ハードウェア両システムの研究開発は，プロジェクトの最初の三年間，つまり，前期の間に完成しなければならない．さらにソフトウェア開発の最終段階において，実際のハードウェア・システムの上で開発を進められるようにするためには，ハードウェア・システムを，できるだけ早く，一年くらいの間で，作らなければならない．

FGCSプロジェクトが始まり，ICOTが研究開発活動を開始した後で，SIMプロジェクトの詳細が決められたが，その中では，「基本ハードウェア・システム」と「拡張ハードウェア・システム」の二つのタイプのハードウェア・システムを作ることにした．基本ハードウェア・システムは，ソフトウェアの開発に使用するので，一年以内にハードウェアを開発しなければならない．拡張ハードウェア・システムは，もう少し時間的余裕があり，二年程度で完成する計画なので，基本ハードウェア・システムより高速なシステムを目指すこととした．

1.4 SIMシステムの構成

プロジェクト管理の観点から見ると，SIMシステムは，次の様な各部分に分けられる．

（1） SIMハードウェア・システム

・基本ハードウェア・システム

このシステムは，パーソナル逐次型推論マシン：PSI—と，ローカル・エリア・ネットワーク：ICOT—Netで構成される．又，PSIのファームウェアの開発と，ファームウェア開発サポートツールもこの中に含まれる．

・拡張ハードウェア・システム

このシステムの主要部分は，高速プロセッサ・システムで，PSIのバックエンド・ハードウェア・プロセッサとなるべく設計されている．図形画像入出力装置等の専用入出力装置もこの中に含まれている．

（2） SIMソフトウェア・システム

SIMのソフトウェア・システムの全体は，SIMPOS (SIM Programming and Operating System) と呼ばれている．

・オペレーティング・システム

オペレーティング・システムは，次の様な機能を提供する．

・プロセッサ，メモリおよび入出力装置を管理する機能
・抽象的な資源，即ち，プロセス，プール，ストリーム等を形成する機能
・マルチ・ウィンドー・システム，ファイル・システム，ネットワーク・システム等の高度な入出力機能
・プログラミング・システム

プログラミング・システムは，システム記述言語 ESP の処理系や，エディタ，デバッガ，コーディネータ等のプログラミングの作成に必要な機能を持つ．

2. 研究開発の経過

2.1 機械語の設計

SIM システムの設計は，PSI の機械語である核言語第0版 (KL 0) の設計から始められた．KL 0 は，論理プログラミングに基づく言語であるというのがその基本的な設計方針でありこの種の言語としては，エジンバラ大学の DEC-10 Prolog (Bowen, 他 1983) や，マルセイユ大学の Prolog-II (Van Caneghem 1982) 等があった．しかし，既存の論理プログラミング言語は，機械語として，又は，システム記述言語として設計された言語ではない．

従って，オペレーティング・システムや広汎なアプリケーション・プログラムを書くためには，制御構造，データ型および低レベルの組込み述語について拡張や再構成せねばならなかった．

制御機構の設計にあたっては，DEC-10 Prolog の簡単なカット・オペレーションは，インタプリタ，デバッガ，エラー・ハンドラ等を実現するのに不十分であるとは考えられたことから，KL 0 では，マルチ・レベル・カットの操作を追加することとした．更に，Prolog-II にあるようなバインド・フック操作および，例外処理機能を追加することとした．

データ型の拡張にあたっては，ストリング，ベクタといったデータ型について，その必要性を検討した後，追加することとし，このための組込み述語を追加した．これらのデータ型の中でも，特にベクタは，オペレーティング・システム中の種々の制御用テーブルを実現するのにしばしば使用されるものである．

ハードウェア制御用などの低レベルの組込み述語に関しては，特殊な組込み述語を数多く追加する必要があった．これは，CPU，メモリおよび入出力装置を制御したり各種のハードウェア・レジスタのようなハードウェア資源にアクセスし，制御するためである．このような特殊な組込み述語は，主として，オペレーティング・システムの実現，特に，メモリ管理およびデバイス管理に使用される．上述の述語以外にも，整数および浮動数点演算の様な通常の算術論理演算用の述語も数多く準備した．

KL 0 の設計にあたってもう一つ検討を加えなければならなかった重要な点は，KL 0 の言語レベルをどの程度に設定するかという点であった．この問題に関しては，通常の Prolog と同程度の高レベルにするかあるいは，DEC-10 Prolog の中間コードと同程度の低レベルに設計するかがポイントであった．DEC-10 Prolog では，ソース・プログラムがこの中間コードにコンパイルされ，その中間コードを PLM と呼ばれる仮想ソフトウェア・プロセッサがインタプリトする．(Warren 1977)．しかし，KL 0 の処理速度を高める為にはファームウェアおよびハードウェアを工夫する余地を大きくとることが得策と考え，KL 0 のレベルは，DEC-10 Prolog のソース・プログラムと同程度の高レベルとすることとした．

KL 0 の設計は 1982 月 7 月に開始され多くの人々の参加を得て進められたが最終的な KL 0 の仕様 (第一版) は，三人の研究者によって，同年 11 月頃にまとめられた．

2.2 主要構成要素の設計

各々の主要構成要素の機能および性能のゴールは，種々の要因，例えば，開発完了までの期間，ゴールを達成するにあたっての技術的困難度，動員可能な人的資源および研究者と技術者の予想される技術レベル等を考慮して決定された．

SIM のオペレーティング・システムとプログラミング・システムを開発するソフトウェア技術者に対して早期にハードウェア・システムを提供する必要があることから，SIM ハードウェアシステムの一部を基本ハードウェア・システムとしてシステム全体から分け，できるだけ短い期間で開発することとした．つまり，SIM ハードウェア・システムは基本ハードウェア・システムと拡張ハードウェア・システムに分けられた．

2.2.1 基本ハードウェア・システム

基本ハードウェア・システムは，1983 年の終まりまで

に，試作を完了することを目指して計画された．PSIとICOT-Netが，本システムに含まれる．

PSI設計の為6人のチームが，1982年夏に結成され，PSIのハードウェアとファームウェア双方のシステム設計にとりかかった．PSI設計チームは，まずPSIを，ビット・マップ・ディスプレイの様な新しい入出力装置やローカル・エリア・ネットワークを備えたスーパー・パーソナル・コンピュータにすることを決定した．又，PSI設計チームは，PSIを，性能面では，DEC-2060上のDEC-10 Prologと同等とし，記憶容量はそれより10倍以上の大容量のシステムとすることとした．

PSI設計チームが最初に試みたのは，PSIのアーキテクチャとハードウェア構成，ファームウェアで実現されるKL 0インタプリタ機構および開発支援システム（ワークベンチとして使用されるミニ・コンピュータを含む）の大きなスケッチを描くことであった．アーキテクチャの設計は，KL 0の内部表現，即ち，メモリ内のデータ形式とインタプリタ機構の設計から始まった．

検討の結果，ガーベッジ・コレクションに関係するメモリ管理と，入出力制御に関わるマルチ・プロセス・サポート用の機構の研究は，興味深いが，きわめて難しい問題であるということがわかった．そこで，これらに対しては，ハードウェア・サポートを行い，プロセス切り換えを高速に行うこととした．これにより，ソフトウェア研究者は，KL 0で，デバイス・ハンドラの様な低レベル制御プログラムも書けることとなり，KL 0の記述能力を試す機会が持てることとなった．

記憶装置に関しては，タイミングよく256 Kビット・メモリチップの使用が可能となり，この結果40ビット・16メガ・ワードの主記憶装置を，キャビネットの3分の1以下の大きさで実装できるようになった．これにより，メモリ拡張のために仮想記憶システムを付加する必要性が薄れ，また，ほとんどのアプリケーション・プログラムにとっては，16メガ・ワードというのは，十分な容量だと思われたことから仮想記憶システムは，付加しないこととした．

PSIの設計と並行して他のチームによって，ICOT-Netの設計が行なわれた．ICOT-Netは，Xerox Ethernet等と同様のCSMA/CD方式のネットワークである．その特徴の一つは，LIA（LAN Interface Adapter）と呼ばれるアダプタを，それぞれのノード・コンピュータに付加し，低レベルのプロトコルを処理することにより，各コンピュータ（この場合，個々のPSI）の負荷を軽減していることである．

この段階での設計の作業というのは，ジグソウ・パズルに例えることができる．設計チームの各メンバは，PSI全体のイメージを頭に描きながら，各自いくつかのパートの設計を行い，各パートの境界部分の整合をとっていった．PSIの機能仕様（第一版）は，1982年終りに完成した．

次の段階は，PSIの詳細設計である．マイクロインタプリタ，即ち，KL 0をインタプリトするマイクロプログラムの設計およびアイクロ命令の設計がこの段階から始まった．それと並行して，CPUのデータ・パス，シーケンサ，メモリ制御部等，主要なハードウェア・モジュールの詳細設計も行われた．マイクロ命令の仕様(第一版)は，1983年春に完成した．この段階の途中から，メーカの研究者に対して，SIMプロジェクトへの参加が要請され設計の最終段階においては，ICOTのメンバと各メーカとの混成チームにより設計が行われた．

PSIのファームウェア，即ちKL 0インタプリタの設計は，ハードウェアの設計と並行して始められた．設計の最初のステップでは，KL 0の内部表現，即ちKL 0の機械語命令の形式およびそのインタプリタ機構の設計が行われた．この機構は，KL 0で書かれたプログラムを可能な限り高速に実行し，かつ前述したバインド・フックやマルチ・レベル・カット操作の様な制御述語を有効的に実現することができねばならない．この設計データを得るとともにユニフィケーション，実行制御，スタック・マニュプレーション等の基本オペレーションのアルゴリズムを検証するために，たくさんのサンプル・プログラムが作られた．この結果，構造体共有方式を採用し，4個のスタックを使用することに決定した．又，データ型を拡張したことから，ヒープ・データ領域の取扱いには，DEC-10 Prologと多少異なる方法を用いることにした．

その他，設計しなければならないファームウェア・モジュールとして，多重プロセス制御をサポートするメカニズム，多重仮想スタックを実現する為のメモリ管理，割込み処理を含む入出力制御があった．こういった低レベルの機能の実現機構とKL 0の解釈機構との調和をとるには，多くの工夫が必要であった．

上記の様な機構の設計は，ハードウェアの設計とも深

く関係しており，これらは並行して進められた．PSIのファームウェア・システムの詳細設計は1983年の1月頃に完成し，その後，ファームウェアの作成が始められた．

2.2.2 拡張ハードウェア・システム

拡張ハードウェア・システムの開発目的は，より高度な処理能力を要する大規模なソフトウェア実験に対処すべく，SIMシステムの能力を高めることにある．もう一つの目標として，図形画像の入出力機能を付加することがある．

SIMシステムの概念設計の段階で，処理能力と記憶容量の将来のニーズを調査した結果，アプリケーション・システムの中には，DEC-10 Prolog以上の処理能力が要求されるものもあるという結論に達した．そのため，それよりも高速のハードウェア・プロセッサを開発する計画が立てられた．

そこで，目標となったのは，実現可能な最大速度を持つプロセッサの開発である．その様なプロセッサの開発は，PSIの開発に比べると，より研究指向が強く，当然，より大きなリスクを伴うものである．しかし，そのプロセッサにより，PSIでは対処できない様な大規模なソフトウェア実験が可能になると思われた．

概念設計の段階で，2つの異なる構造が考えられた．一つは，PSIの上位互換機で，もう一つは，PSIのバックエンド高速プロセッサである．この選択についてメーカの研究者を含めた討論を重ねた結果，PSIのバックエンド高速プロセッサとすることとした．こうすることによりこのプロセッサの設計においては高速なプロセス切換や入出力制御をそれ程重視しなくてすむこととなり，プロセッサのスピードアップに専念できることになった．

この高速プロセッサには，PSIと異なるアプローチをとる試みがなされた．つまりPSIと違って構造体コピー方式に基づく解釈方式，機械語命令セット，アーキテクチャおよびデバイス技術を取入れた．相違点として重要なものは，高速プロセッサは解釈機構が構造体コピー方式に基づくものである点，および機械語のレベルがPSIより低い点である．機械語を低レベルとすることにより，コンパイラは，プログラム実行速度を上げる為，より強力な最適化が行える可能性をもつこととなる．

しかし，言語の開発に関しては，上述の様な相違点を越えて，KL 0のサブセットをサポートできるようなものとした．その中には，重要な組込み述語も含まれている．しかしハードウェアに依存する組込み述語は，当然ながら，PSIと高速プロセッサとでは異った物になる．この結果，ESPで書かれたユーザ・プログラムの大部分は，高速プロセッサ用のコンパイラを使用することにより，ほとんどそのままこの高速プロセッサにかけることができることとなる．

このほか高速化の為，ハードウェア実装には CML (Current Mode Logic) を使用することとし，CPU部分には，一部に局所的並列処理方式の導入を試みた．このような新しい試みは，機能仕様の実現可能性をより厳密に検証するためハードウェア・モジュールと解釈機構の重要な部分に関して一部詳細設計を行うことを必要とした．このため，機能設計に数ヶ月要することとなった．しかしながら，ハードウェア・システムの機能仕様（第一版）は，1983年の秋頃にはまとめられその後，詳細設計は，1984年春に一応の完成をみた．

2.2.3 ソフトウェア・システム

SIMのソフトウェア・システムは，SIMプログラミング/オペレーティング・システム，即ち，SIMPOS概念の設計から始まった．SIMPOSは，まず最新のパーソナル・コンピュータ・システムにさらに高度な機能を付加したパーソナル・オペレーティング・システムの完成を目指した．多くのPSIをLANシステムで結ぶ分散システムの構築・円滑なマン・マシン・インタラクションの実現などが，付加されるべき機能である．更に，これらの機能のほか，エディタ，言語プロセッサ，デバッガー等の機能や，多くのユーティリティ・ソフトウェア・モジュールを使用して，効率の良いプログラミング・システムを作る必要がある．

SIMPOSの機能は，LISPマシンの様なスーパー・パーソナル・コンピュータと同様の高度な機能を持つ革新的なオペレーティング・システムを目指している．しかし，SIMPOSの場合には論理型の核言語第0版という全く新しい言語でSIMPOSを書くというリスクの高い条件があった．又，SIMPOS開発チームの多くの者は，当然，このような言語については素人であった．

このような条件のもとで1982年の終りまでに，ウィンドウ・システムや，ファイル・システム，又，エディタ等の重要なソフトウェア・モジュールおよびサブシステムの機能仕様の設計がまず行われた．オペレーティング・システムのモジュール構造の設計は，オブジェクト

指向の概念に基づくものとした．又，その他に，ハードウェアシステムの制御や，オペレーティング・システムの実現の為の基本的な機能が，KL 0 の組込み述語として加えられた．

SIMPOS の機能仕様は，1983 年度末にまとめられた．この段階での設計は ICOT の研究者から成る 10 名程度のチームで行われた．

上述の作業とほぼ並行して，システム記述言語 ESP の設計が始められた．特に，注意を払った点は，KL 0 の仕様および SIMPOS の設計方針から生ずる必要条件に関する点である．SIMPOS のプログラミングとデバッグを円滑に行う為の ESP のモジュール化の方法が，論点であった．討論を重ねた結果，ESP のモジュール化には，多重継承機構を持つクラス・システムを採用することとした．これらを含む ESP の仕様は，1983 年 7 月頃までまとめられた．

ソフトウェア，ハードウェア双方の開発活動が円滑に行われるよう，1983 年 4 月以降，SIM プロジェクトに関係する ICOT の研究陣は，ICOT の研究所内の一研究室に全員集められた．

1983 年 6 月頃からは SIMPOS 開発チームも 30 名程度に増員され，詳細設計が始まった．メーカから新規に参加する研究者にとっては，新しく勉強しなければならないことがいくつもあった．それは，論理プログラミング，オブジェクト指向プログラミング，KL 0，ESP 等であった．彼らは，当初は当惑気味であったが，数ヶ月後には，それらの事柄について一応理解できるまでになっていた．

SIMPOS の各サブシステムの詳細設計を開始するにあたって，開発チームは，いくつかの小人数のグループに分かれて詳細設計にあたった．8 月頃には DEC 2060 上で ESP のコンパイラとシミュレータの第一版がクロス・システムとして作成され，これにより，実際に，ESP でサンプル・プログラムを書くことができるようになった．詳細設計の作業としては，クラス・モジュールを実際に設計することと，各サブシステムの実現に必要なインタフェースの仕様を設計することであった．SIMPOS の詳細仕様（第一版）は，1983 年 11 月末までにまとめられた．

2.3 SIM の製作

2.3.1 PSI と LAN の製作

PSI のハードウェア・システムの製造一部は，1983 年 5 月に始められ，メモリ・システムや入出力バス・インタフェイス等の詳細設計と並行して行われた．この間に保守性を高めるために RAS システムが付加され，これにより，保守も容易となった．1983 年 6 月には，ハードウェア・システムの詳細設計が完了し，システムの製造が開始された．その後，細かい修正と改良を何回か行った後，入出力装置を含め，最初の PSI ハードウェア・システムが ICOT に搬入されたのは，1983 年の 12 月であった．

PSI の開発サポート・ツールの一つとして PDP 11/23（ミニ・コンピュータ）が導入され，DEC net を通じて DEC 2060 に接続された．このミニ・コンピュータは，SVP（PSI の Supervisor processor）と名付けられ，ハードウェアおよびファームウェアのデバッグに使用されることになっていたが，後で SIMPOS の核部分，例えば，IPL（initial program loading）プログラムのデバッグにも使われるようになった．

2.3.2 PSI のファームウェアの開発

ファームウェア開発の過程で，最初の重要な仕事は，マイクロプログラムの作成とデバッグのための高性能マイクロ・シミュレータおよび使い易いマイクロ・アセンブラを開発することであった．マイクロ・アセンブラには，コーディング・エラーを可能な限り正確に探知できることと，高レベルのプログラミング言語タイプの表現が使用できることの二つが要求された．

汎用マイクロ・アセンブラは，Prolog で書くこととした．汎用マイクロ・アセンブラのプロセッサには，通常の場合，マイクロ・アーキテクチャ特有の細かい制約を規定する機能が不十分なため，正確なエラーチェックがむずかしい．しかし，今回開発したアセンブラは，Prolog インタプリタをアセンブラ・プロセッサとみなし，アーキテクチャの規定を，Prolog のプログラムのようにして書くことができるものとしたので，きわめて正確に，制約を規定できることになった．その結果，アセンブラ言語は，細かなエラー・チェック能力を持つ判読性の高いアセンブラ言語となった．

マイクロ・シミュレータに要求される主な能力としては，デバッグ能力が優れていることおよび，使い易いヒューマン・インタフェース機能を持つことの二つがあった．設計の基本方針として，シミュレータのデバッグ環

境は，SVP の実際のマシンより生産効率が良く使い易いものをめざすこととした．こうして 1983 年 6 月には，PASCAL で書かれた約 6000 行のシミュレータの第一版が，DEC 2060 上で使用できるようになった．

PSI のファームウェア，即ち，KL 0 のマイクロ・インタプリタの製作は，詳細設計と並行して進められていった．

KL 0 には 100 以上の組込み述語があり，その中には，設計と，プログラミングにあまり時間のかからないものもあるが，基本制御モジュールや，ユニフィケーション・モジュールのようにマイクロ・インタプリタの核部分に当たるものは，設計に長い時間を要するものもある．PSI ファームウェアの仕様（第一版）は，1983 年度末完成した．

マイクロ・アセンブラとマイクロ・シミュレータは，数回の改良の後，きわめて使い易く，かつ信頼性の高いツールとなった．マイクロ・プログラムのコーディングとデバッギングの大部分は，それらを使用して DEC 2060 上で行われた．マイクロ・プログラムのデバッグとしては，1983 年 11 月に最初のサブセットがクロス・システム，即ち DEC 2060 上でデバッグされ，メーカのハードウェア開発者に，PSI のハードウェア・システムのテスト用に提供された．マイクロプログラムのうち，プロセス切換えの制御，メモリ制御，ガーベッジ・コレクションに関する部分以外の大部分は，1984 年 2 月以前に，クロス・システム上でデバッグされていた．

SVP 上でファームウェアのデバッグをサポートするモニタ・プログラムも，ファームウェアとともに開発が進められ，1984 年 1 月頃に完成した．こうして，実際のマシンを使用したファームウェアのデバッグが同年 2 月に始まり，SIMPOS のマイクロ・プログラムのうち，最初のサブセットのデバッグは，同年 2 月中に完了した．これにより，オペレーティング・システムの開発チームが，IPL プログラムのデバッグに取りかかることができるようになった．

SIMPOS の開発に必要なマイクロ・プログラムの大部分が完成したのは，1984 年度末である．マイクロ・プログラム全体のステップ数は，約 12 K である．ファームウェア開発チームは，ピーク時には，12 名の研究員を擁し，うち，ICOT のメンバーは，3 名であった．

2.3.3 拡張ハードウェア・システムの製作

高速プロセッサの製作は，一部，1984 年 5 月に始められ，現在も進行中である．高速プロセッサのファームウェアの開発は，1984 年春に開始された．ハードウェアの製作は，1984 年中に完了する予定である．

高速プロセッサのハードウェア，ファームウェアおよびソフトウェアの開発には，VAX 11/780 と，VAX 11/750 が使用され，多くの開発サポート・ツールが開発された．高速プロセッサは，現在，スーパバイザ・プロセッサとして，VAX 11/750 に接続されている．ハードウェアとファームウェアの開発サポート・ツールは，主として，C 言語で書かれ，VAX 11/780 上に作られた．ソフトウェア開発のサポート・システムも，同様に，VAX 11/780 のシステム上に作られた．

高速プロセッサ開発の完了予定は，1984 年度末である．プロセッサのファームウェアとソフトウェアのそれぞれに新たに採用した機能が評価されることにより論理型言語の実行速度を上げるための新しい多くの知見が得られることが期待される．

2.3.4 ソフトウェア・システムの開発

SIMPOS は，1983 年 11 月に重要なモジュールのサンプル・コーディングをもって開始され，詳細設計の完了を待って本格的な製作にかかった．ESP 用の DEC 2060 上のクロス・ソフトウェア開発システムの仕様は，大型のクラス・モジュールの処理と，操作性向上のため，数回にわたって改良された．

1983 年の終り，ハードウェア，ファームウェア，および SIMPOS のオペレーティング・システムの仕様の最終決定の後，IPL プログラムの設計とコーディングが始められた．IPL プログラムには，ESP の実行時サポート・ルーチンと，オペレーティング・システムのモジュールの多くが含まれている．その大部分は，PSI の実際のハードウェアおよびファームウェアを使ったデバッグを行う必要があった．システムはそれ程大きいものではなかったが，複雑であったため，デバッグが完了したのは，1984 年 4 月であった．

その後，6 台の PSI と，DEC 2060 上のクロス・システムを使用して，SIMPOS の製作が進められ，同年 7 月には，ウィンドウ・サブシステムの一部動作が可能となった．ウィンドウ・システムは，オペレーティング・システムの中でも最も複雑なサブシステムであるが，これにより，ゴール達成までのめどがたった．

ウィンドウ・システムは，システム全体の動作状況をみるテスト・プログラムとして使うこともできる．7月初め，ウィンドウ・システムの応答はかなり時間がかかった．ファームウェア中に，まだ実現されていない組込み述語がいくつか残っており，これがソフトウェアによってサポートされていたためである．又，その時点では，オペレーティング・システム中のモジュールの多くは，正しく組上ることに重点を置いた設計となっており，高速性を優先して設計されたものではなかった．

つまり，経験の少ない開発者の手によって短期間で開発しなければならなかったことから SIMPOS の設計にあたっては，短期間で正しく組上ることに最大の注意を払った．その後，改良個所をいくつか選び出し，実際に，ファームウェアおよびオペレーティング・システムの改良を行った．これらモジュールの改良は，プログラミング・システムのソフトウェア・モジュールの製作と並行して行われた．

ウィンドウ・システムがツールとして使用可能といえるようになるには，応答時間を短縮する必要があった．その方法としては，例えば，クローズ・インデックス・サポートをつける等，オペレーティングシステムの改良とファームウェア・サポートの追加が考えられた．研究の結果，KL 0 は拡張され，ESP の実行時サポート・ルーチンの一部，即ち，メソッド・コールとスロット・アクセスをサポートするルーチンの組込み述語を含むことになった．これにより当初に比べ 2 ケタ近い速度改善が為された．現在，この拡張の作業が行われている最中であり，今後，更に改良が加えられる．

SIMPOS は，現在も，開発中であり，その開発チームは，ICOT のメンバ 6 名を含む 40 名以上の研究者で構成されている．SIMPOS の大きさを正確に言うのは難しいが，その行数は，全体で ESP で約 90 K 行となっている．

3. SIM の主要構成部分

3.1 基本ハードウェア・システム

PSI のファームウェアおよびハードウェアのシステムの詳細に関しては，他の論文を参照されたい．(Taki 他，1984)，(Yokota 他，1984) ここでは，アーキテクチャおよびハードウェアの特性の主なものについてのみ，要約を記述する．

3.1.1 PSI のアーキテクチャ

(1) タグ・アーキテクチャ：全てのメモリ・セルは，40 ビットで，その構成は，8 ビットのタグと，32 ビットのデータからなっている．

(2) マイクロ・プログラム制御：コンパイルされた KL 0 の内部コードは，マイクロ・プログラム化されたインタプリタによって実行される．

(3) ハードウェア・スタック機構：メイン・メモリは，論理的に独立している 256 の領域に分割されている．その領域のそれぞれを，ヒープ・エリア，又はスタック・エリアとして使用することができる．又，デマンドに基づいたページ割つけ機構が提供されている．

(4) 多重プロセス・サポート：ハードウェアがサポートするプロセスが，最大 63 個まで使用できる．

(5) 入出力バス：標準 IEEE バスを使用している．

(6) LAN： Ether-net と同型の CSMA/CD 方式を採用．転送速度は，10 M ビット/秒．

3.1.2 PSI のハードウェア

(1) 実行速度：約 30 KLIPS(Logical Inference Per Secend)

(2) マイクロ・プログラム記憶装置：64 ビット×16 K ワード

(3) マシン・サイクル・タイム：200 ナノセカンド

(4) 記憶容量：256 K チップを使用．40 ビット×16 M ワード．

(5) キャッシャ・メモリ：40 ビット×4 K ワードのメモリを 2 組使用．

(6) デバイス・テクノロジー：CPU には主に TTL を採用し，メイン・メモリには NMOS を採用した．

3.2 拡張ハードウェア・システム

拡張ハードウェア・システムは，現在開発中である．ここでは，主な特性を記述する．

3.2.1 高速プロセッサ・モジュール

(1) 機械語：スタック指向型の命令セットを用いており，言語レベルは，KL 0 より低く設定してある．(Tick 他，1984) 170 以上の組込み述語を持ち，KL 0 と ESP をサポートできるように設計されている．

(2) 解釈部：構造体コピー方式に基づいており，3 個のスタックを使用している．

(3) タグ・アーキテクチャ：全てのメモリ・セルは

36ビットで，その構成は，4ビットのタグと32ビットのデータ，又は7ビットのタグと29ビットのデータのいずれかである．

（4） マイクロ・プログラム制御：機械語はマイクロ・プログラム化されたインタプリタによって解釈される．

（5） メモリ管理：メイン・メモリは，論理的に独立している8つの領域に分割されている．その領域のそれぞれを，ヒープ・エリア，又はスタック・エリアとして使用することができる．又，デマンドに基づいたページ割つけ機構が提供されている．

（6） 入出力部：スーパーバイザ・プロセッサを通して，PSIへ接続できるように設計されている．

（7） 実行速度：200 KLIPS 以上

（8） マイクロ・プログラム記憶装置：80ビット×11 Kワード

（9） マシン・サイクル・タイム：100ナノセカンド

(10) 記憶容量：36ビット×64 Mワード

(11) キャッシャ・メモリ：36ビット×8 Kワード

(12) デバイス・テクノロジー：CPUには主にCMLを採用し，メイン・メモリにはCMOSを採用した．

このプロセッサの設計に関しては，まだ研究すべき課題がいくつも残されており，機能のいくつかは，今後の研究に基いてこれから改良することが前提である．このプロセッサは，研究的な性格の強いものであるが，それでも，実行速度としては200 KLIPS以上のスピードを持つことが期待されている．

3.2.2 専用入出力装置

図形・画像処理の研究ツールとして，図形，画像入出力用の専用ワーク・ステーションの開発も進められており，図形画像出力のための高解像度ディスプレイとカラー・プリンタ，および図形画像入力のためのTVカメラ，ファクシミリなどが，ワーク・ステーションに含まれている．ワーク・ステーションは，マイクロ・プロセッサで制御され，専用の図形発生チップを使用して，グラフィック機能をサポートすることとなっている．また，ワーク・ステーションは，PSIの入出力バスに接続され，図形画像を使った実験に使用される．

3.3 ソフトウェア・システム

SIMPOSとそのシステム記述言語であるESPについては，他の論文に詳しく書いてあるので(Chikayama, 1984)(Yokoi他，1984)，ここでは，SIMPOSのプロフィルについて以下に述べる．

（1） 多重プロセス・サポートを備えたパーソナル・オペレーティング・システムである．

（2） そのモジュール構造は，オブジェクト指向の考え方に基づくものである．

（3） 日本語の文字に入出力を含むウィンドウ・サブシステムを通して，マン・マシン・コミニケーションが行なわれる．従って，SIMPOSでは，全ての文字は，1文字につき16ビットで表わされる．

（4） ユーザは，ネットワーク・サブシステムにより，リモート・プロセスやリモート・ファイル等の離れた所にある資源にアクセスすることができる．

（5） プログラミング・システムに含まれているのは，ライブラリ・モジュール(ESPコンパイラを含む)，インタプリタ，デバッガ，エディタ等である．

SIMPOSのソフトウェア・モジュールは，全てESPで書かれており，システム・プログラマだけでなく，SIMPOSのユーザにとっても，ESPが標準言語となる．これは，DEC-10 Prologのスーパセットとなっている．

4. 将来の計画

4.1 SIMの改良

SIMプロジェクトの最も重要なゴールは，FGCSプロジェクトの研究者に，論理プログラミング環境と使い易いツールを提供することにある．

SIMプロジェクトは，一部，現在も進行中である．このプロジェクトの中で，最も早く目標を達成すると思われるのは，基本ハードウェア・システム，即ち，PSIとICOT-Net，およびソフトウェア・システム，即ち，SIMPOSである．これらのシステムは，FGCSプロジェクトの中期には，研究者に提供される．しかし，今後も，ユーティリティ・ソフトウェア・パッケージの拡張や，入出力装置の追加等，改良と拡張が必要である．

拡張ハードウェア・システムに関しても，また，多くの改良と拡張を行い，基本ハードウェア・システム同様，安定した信頼性の高いシステムにする必要がある．この様なわけで，FGCSプロジェクトの中期においても，SIM改良のための努力は，ひき続き行われることとなる．

4.2 並列ソフトウェア・開発サポート

中期における最も重要な研究課題は，並列処理システ

ムの開発である．現在，KL1と呼ばれる並列論理型言語の設計が，行われている．KL1のための実験的な言語プロセッサとシミュレータを実現するには，使い易く安定した並列ハードウェア・システムが必要である．

従って，PSIを使用して，小規模な多重プロセッサを開発することが計画されている．このシステムは，何台かの接続されたPSIで構成され，各PSIのファームウェアを拡張してKL1をサポートできるようにしようというものである．PSIが逐次型マシンであるため，KL1で書かれたプログラムの一部は，逐次的に実行される．しかし，その他の部分に関しては，並列的に実行される．そのシステムにより，プログラムをいくつかの並列処理モジュールに分割する方法の研究や，並列処理に要するコミュニケーション・コストの評価を行うことができるであろう．

5. 結　　論

SIMプロジェクトは，現在進行中のプロジェクトであるが，初期の目標達成は，もう目前であると言える．基本ハードウェア・システム，即ち，PSIとICOT-Net，およびソフトウェア・システム，即ち，SIMPOSには，まだ改良すべき点が残っているが，ツールとして十分使用に耐えうるものになると考えられる．

このプロジェクトは，すでにいくつかの成果を生み出しており，その1つは，KL0に関するものである．KL0のベースは，初め，DEC-10 Prologであったが，それがより強力なESPに変わり，今では，ESPが，FGCSプロジェクトの標準言語となっている．従って，KL0の仕様は，論理プログラミングとオブジェクト指向概念の双方をサポートするべく拡張された．

また，このプロジェクトは，「Prologでオペレーティング・システムを書くことは可能か？」という問いに対する答えを出したと言えよう．論理プログラミングの概念にオブジェクト指向の概念をつけ加えはしたが，ESPが，オペレーティング・システムを書くのにきわめて有効であるということは，十分に証明された．

更に，このプロジェクトによって，ESP型の言語がアーキテクチャ研究者に対し，実行速度向上のための新しい機構を導入する機会を提供するものであるということが証明された．この研究は，直接，ソフトウェアの生産性と信頼性の向上に貢献するものとなる．

謝　辞

SIMプロジェクトは，ICOTと，メーカ5社との強い協力の元で行われたものであり，ここに，プロジェクトに参加した研究者および技術者全員に，感謝の意を表したい．また，Prolog処理系実装の技術的基盤を築き，多くのアドバイスを与えて下さったD. H. D Warren博士と，その他，内外のたくさんの研究者の方々にもこの場を借りて，お礼を申し上げたいと思う．