講座：手作りスーパーコンピュータのススメIII

2. 電子回路を知らない人のための入門編(2)

～ステップアップ：実際の製作からスーパーコンピュータを越えるまで～

伊 藤 智 義
(群馬大学工学部)
(1994年2月3日受理)

Guide for Beginners (2)

ITO Tomoyoshi
(Received 3 February 1994)

Abstract

The previous and this chapters are guides for beginners of hardware. Today's advanced technology can get us to make a hand-made computer for our own purpose. Making a computer is not so difficult as many people might think. Hardware is not hard. In this section, I describe the process to develop a hand-made computer by using the equipments which I introduced in the previous chapter. In addition, I discuss the merits and potential of a special-purpose computer.

Keywords:

hardware, computer, electronic engineering, numerical simulation, special-purpose computer, supercomputer,

1．実際の製作 ―設計から完成まで―

前回，ハードウェア開発のための設備や使用する電子部品の紹介をした．今回はそれらを用いて実際に専用計算機を開発する過程をお話ししたい．

(1) 概念設計

何をどんな風に作るかをはじめに決めなければならない．ここでの方針が今後の開発を左右し，ひいては完成した計算機の優劣を決定する．当然のことではあるが，もっとも重要な第一歩である．

ここ数年，いくつかの分野で専用計算機が作られている．それらをみると，開発に際して二つの考え方が存在するように思える．一つは「自分の扱っている数値計算はスーパーコンピュータには乗り難いので，専用計算機を作る」というものと，もう一つは「スーパーコンピュータには適した数値計算ではあるが，現在のスーパーコンピュータでもまだまだ能力不足なので，専用計算機を作る」というものである．前者は，例えば第1章を担当した川合さんらが開発した並列型計算機PAX [1] であり，後者は前回触れた重力多体問題専用計算機GRAPE [2-4] である．

もう少し別の言い方をすると，前者が物理モデルに立って考えているのに対し，後者は計算コー

Department of Electronic Engineering, Gunma University, Kiryu 376.

ドに立って考えているともいえる．Applegateらは1985年に太陽系の数値計算のための専用計算機 Digital Orrery [5] を開発したが，それは9つのプロセッサをリング状に接続した構成になっており，一つ一つのプロセッサが太陽系のそれぞれの惑星を担当する．前者の典型である．それに対してGRAPEは，計算コードの中でもっとも計算量の大きいサブルーチンに注目して，それをハードウェア化したものである．

どちらの方法が有効かは問題に依存し，もちろん，どちらの考え方も重要である．ただし両者は，開発の難易度からみると天と地ほどの開きがある．

前者の，物理モデルに立脚した開発は，計算機システムを，ネットワークの問題やソフトウェアも含め，一から構築しなければならず，大変な仕事である．残念ながら，簡単に「手作り」というわけにはいかない．

一方，後者の，計算コードに立脚した専用計算機作りは，はるかに見通しが明るい．自分の計算コードを高速化するために，ソフトウェアで計算していた部分を専用ハードウェアに置き換えていくという立場であるので，「専用計算機」は「ハードウェア・サブルーチン」のことであり，開発方針は非常に明確になる．こちらは十分「手作り」が可能である．そこで，ここでは，こちらの立場で話を進める．

その場合のシステムの基本構造は図1のようになる．もっとも計算量の大きい部分だけを専用計算機で計算し，その他の部分はホスト計算機で処理する．ホスト計算機には市販のパソコンやワークステーションをあてる．

実はこの構成は基本的にスーパーコンピュータと同じである．スーパーコンピュータは内部にスカラ計算機とベクトル計算機を持っている．スカラ計算機は図1のホスト計算機に当たり，ベクトル計算機は専用計算機の部分に相当する．スカラ計算機とは一つのCPUですべてを処理していく計算機であるのに対し，ベクトル計算機とはいくつかの演算器をつないで流れ作業的にデータを処理していくものである．ベクトル計算機の計算方法はパイプライン方式と呼ばれ，同時に動作する演算器の数が多いほど演算速度は向上する．並列計算の一形式である．

スーパーコンピュータは数値計算用に開発された「専用計算機」なので，構成が同じなのは当然のこととともいえる．ただし，スーパーコンピュータは数値計算に関しては汎用性を持たされているので，その部分をさらに自分用に専用化すれば，スーパーコンピュータをしのぐ性能が期待される．そういう意味では，スーパーコンピュータに乗りやすい数値シミュレーションは，専用ハードウェアを作ることで，同じコストならば，必ずスーパーコンピュータをしのぐ性能が実現できるともいえる．

図1の構成は，開発レベルにおいて，さらに二つの大きな利点を持っている．一つは，手作りの部分が小さいということであり，もう一つは今まで使っていた計算コードがそのまま動くということである．これは，ハードウェア化する部分が計算コードの一部分であることによる．簡単だが計算量の大きい部分は専用計算機に任せ，複雑だが計算量の小さな部分はホスト計算機が受け持つようにすればよいのである．

具体的に例を上げて説明しよう．現在，群馬大学の私たちの研究室ではホログラフィー専用計算機を開発している [6, 7]．ホログラフィーとはいわゆる三次元写真のことで，普通の写真が光の強度のみを記録するのに対して，ホログラフィーでは，物体を単色光で撮影し，その際，基準となる参照光を加えることによって，強度の他に位相の情報も記録する．そのため，三次元情報を二次元平面に記録することが可能になる．

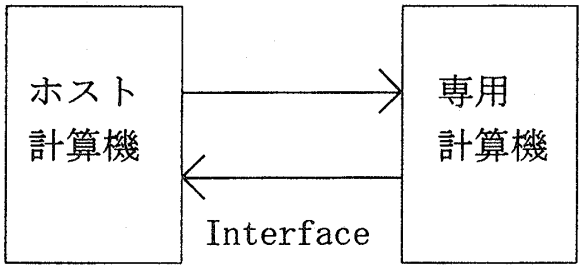

図1　専用計算機の基本システム

ホログラフィーでは，写真でいうフィルムに相当する部分をホログラムと呼ぶが，最近ではコンピュータ内の画像イメージから計算によってホログラムを作ること（Computer Generated Hologram）も盛んに行われるようになってきている．ここではホログラフィーの解説をするのが本意ではないので詳しい説明は省くが，その際，計算すべきものは次式のような単色光の重ね合わせである．

$$\phi_{\alpha j} = \frac{A_j}{R_{\alpha j}} \exp(ikR_{\alpha j}) \tag{1}$$

$$\Phi_\alpha = \sum_j \phi_{\alpha j} \tag{2}$$

ここで，A_j は物体各点の明るさ，$R_{\alpha j}$ は物体各点とホログラム上の1点との間の距離，kは単色光の波数である．

式自体は単純なものであるが，その計算量が膨大である．例えば，物体を100 × 100 × 100の画素で表し，ホログラムを1000 × 1000の格子点で表現すると，一枚のホログラムを作るのに(1)式の計算を$100^3 \times 1000^2 = 10^{12}$回行わなければならない．

ところが，これに対して，その他の計算量は，例えば各点のグラフィック表示や位置の設定などは各点に対してそれぞれ1回ずつ行えばよいので，$100^3 + 1000^2 = 2 \times 10^6$にしか比例しない．したがって，計算量のほとんどすべては(1)式の計算で占められる．逆にいえば，(1)式さえ速くできれば，計算全体を加速できる．専用計算機に非常に向いた問題の一つである．

そこで私たちはその専用計算機をHORN（HOlographic ReconstructioN）と名付け，開発を行った．図2に試作2号機HORN-2 [8]（昨年11月に完成している）のパイプライン構成を示す．xy 平面ごとに計算するなど少々作為的なところもあるが，基本的には(1)式の計算に合わせて演算器を並べただけである．精度は32bit（単精度）とし，三角関数の計算には前回紹介した線形補間回路を使った．

ホスト計算機にはパソコンを使い，インターフェースにはGPIBを用いた．材料費は70万円くらいで，10MHzで動作しており，演算速度は0.5Gflopsくらいである．実効でワークステーションより100倍程度速い．

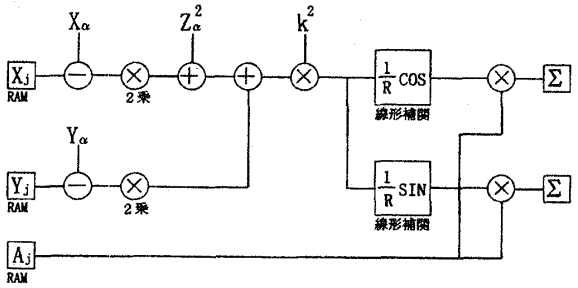

図2　HORN-2のパイプライン構成
左のメモリに全画素（格子）数分のデータを入れておき，1クロックごとに右の演算器に送り，流れ作業式にデータを加工していき，最終段で積算する．

演算速度の見積もりは，専用計算機が行う演算が汎用計算機ではどれくらいに相当するかを考えて求めればよい．まず，演算数を計算する．加算と乗算を1に数える．他の関数は多項式展開するなりして，加算と乗算に換算する．三角関数をいくつに数えるかにもよるが，(1)式は大ざっぱに50演算くらいに相当する．パイプライン方式では，この演算が1クロックごとに同時になされるので，演算速度は（演算数×動作周波数）で表現される．HORN-2の場合，50 × 10MHz = 0.5Gflopsである．これはもちろんピークスピードのことであり，その他に通信時間などがかかるので，実効速度はそれらを考慮して求める．HORN-2では，通信時間は計算時間の10%程度なので，実効速度はピーク速度の90%くらいになる．

ここで，概念設計の段階で行う作業をまとめておこう．

1．全体のシステム構成を考え，専用計算機による効果が開発に見合うかどうかを見積もる．
2．専用パイプラインの構成を考える．
3．計算精度を決定する．
　計算精度は開発難度を決定的に左右する．誤差解析を行って，必要十分な精度を見積もる．前回触れたように，精度が32bit（単精度）を越えると開発は途端に難しくなる．

4．ホスト計算機とインターフェースを決める．
通信時間が計算時間の数十％程度に収まるくらいなら何の問題もないが，通信時間があまり大きくなると，計算機システムとしては格好が悪くなる．ただし，通信時間がたとえ計算時間と同程度になったとしても，インターフェースが無限に速くて通信時間が0のときに比べても，システム全体のスピードは高々2倍しか違わない，という考え方もできる．手作りなのだからと開き直り，見てくれよりも実を取るのも良いかもしれない．

5．クロック・スピードと主要なICの選定を行い，開発費の見積もりを出す．
クロック・スピードは，ユニバーサル基板での試作段階では，10MHzくらいがいいところである．クロック・アップするためには，プリント基板を作ることが必要になるので，次の段階で考える問題である．

特に初めて回路を組もうという場合は，可能な限り簡単なところから始めることをすすめる．基本構想さえしっかりしていれば，性能の向上は後からいくらでもできるからである．例えば，HORNの試作1号機HORN-1は，演算精度を16bitに設定してROMを演算器に使い，インターフェースにはRS232Cを使って，約7万円で開発した（写真1）．いろいろな制約はあるが，スピードはワークステーションより100倍速い．どんなに簡単なものでも，形になっているものがあると，後が楽である．逆に，最初で挫折してしまうと，そこで終わってしまうことになりかねない．

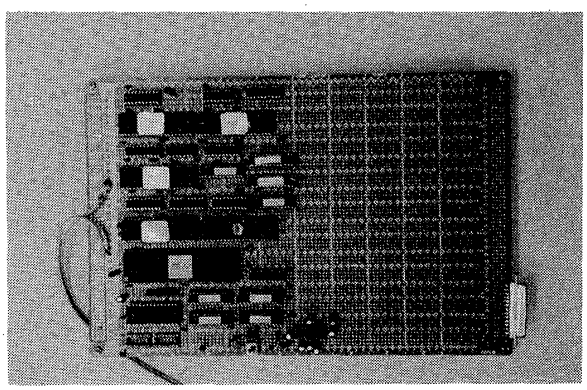

写真1　HORN-1

（2）詳細設計

概念設計をもとに，使用するICのデータシートを取りそろえて，詳細設計を行う．タイミングチャートを作ったり，PLDの仕様を決めながら，最終的な設計図に仕上げていく．

設計の基本は，一相クロックの同期式である．つまり，一つのシステム・クロックで回路全体を同期して動作させるという意味である．非同期式ではタイミングが場当たり的になり，ノイズにも弱くなる．完全同期式で作っておけば，将来回路をチップ化するときにもスムーズに移行できる．

詳細設計を進める上でもっとも重要なのはカウンタである．カウンタを制するものは回路設計を制するともいえる．回路を制御する部分は順序回路（シーケンサ）で作られるが，順序回路はカウンタで作られるからである．

代表的なカウンタにはバイナリ・カウンタ（図3）やジョンソン・カウンタ（図4）がある．

バイナリ・カウンタはもっとも効率のよいカウンタである．例えば同じ3bitカウンタでも，図を比較すればわかる通り，バイナリ・カウンタでは（0，0，0）から（1，1，1）まで8状態が作れる

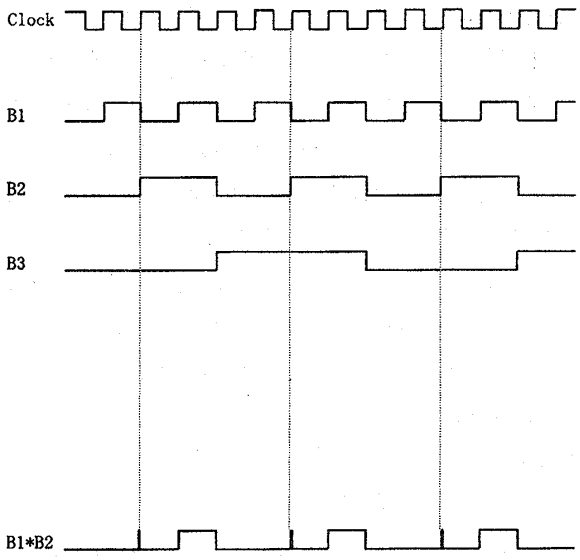

図3　3bitバイナリ・カウンタ（上：B1，B2，B3）
ハザードの例（下：B1*B2）
非同期でANDをとった場合，B2のスイッチングがB1より速いと図の位置にノイズが出る．

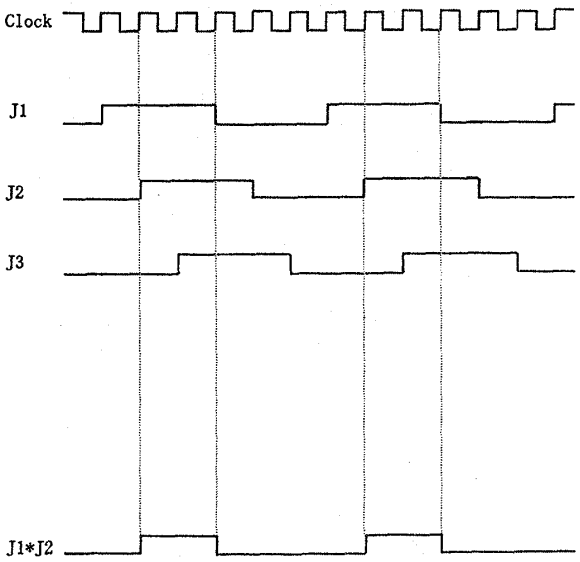

図4　3bit ジョンソン・カウンタ(上：J1, J2, J3)
ジョンソン・カウンタではハザードは出ない.(下：J1*J2)

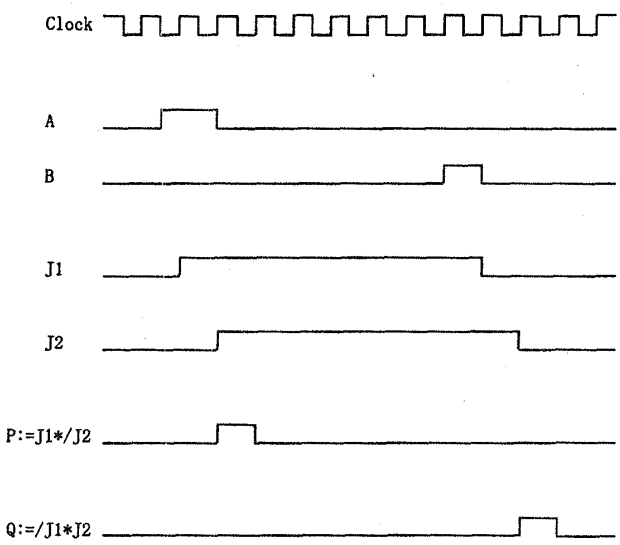

図5　ジョンソン・カウンタによる順序回路の例
ただし，:＝はクロック同期を，/は補を表すものとする．PとQはクロックに同期しているため，状態は1クロック遅れて変化する．

のに対し，ジョンソン・カウンタでは (0, 0, 0), (1, 0, 0), (1, 1, 0), (1, 1, 1), (0, 1, 1), (0, 0, 1) の6状態しか作れない．

その反面，複数の信号の論理積 (AND) をとった場合，バイナリ・カウンタではスイッチングのタイミングにずれがあるとスタティック・バザード (いわゆるヒゲというもの) が出ることがあるが，ジョンソン・カウンタにはそれがない．

状況に応じてカウンタを使い分ければよいのだが，順序回路の基本はジョンソン・カウンタだと覚えておいても間違いではない．

簡単な例として，次のような，ホスト計算機と専用パイプラインを仲介するような順序回路をジョンソン・カウンタで作ってみる．

1．ホスト計算機が演算開始の信号 (A) を順序回路に送る．
2．順序回路は (A) を受けるとパイプラインを動かす命令 (P) を出す．
3．パイプラインは (P) を受けると演算を開始し，演算を終了すると1クロック分だけ信号 (B) を順序回路に送る．
4．順序回路は (B) を受け取ると，演算が終了したことをホスト計算機に知らせ (Q)，初期状態に戻り，次の命令 (A) を待つ．

このタイミング・チャートは図5のようになる．言葉でいうと複雑だが，タイミング・チャートで見ると単純なものである．

さて，実際に問題になるのは，図5のようなジョンソン・カウンタをどう作るかであるが，PLD (Programmable Logic Device) を使えば簡単である．

PLD は専用の設計ツール (ソフトとライタ) で作る．ツールはいくつか市販されているが，基本的なことは同じである．入出力ピンをそれぞれ適当なピンに割り当て，それらの信号を使って出力ピンの論理式を書く．あとはツールがデバイスに書き込み可能なデータに変換してくれるので，それをライタで書き込めばよい．

図5のジョンソン・カウンタの場合，論理式は次のようになる．

$$J1 := \overline{RESET} * (A + J1) * /B$$

$$J2 := \overline{RESET} * J1$$

ここで，:＝はクロックに同期して状態が変化することを示し，＊は論理積 (AND)，＋は論理和 (OR) であり，$\overline{\ }$ は補の状態を表すものとす

る．*RESET* はシステム起動時にカウンタを既知の初期状態（J1＝L,J2＝L）にするためのものとして書き加えておいた．タイミング・チャートと一緒に見ると理解しやすい．

デジタル回路も意外と簡単である．作業時間は，慣れるまで大変だが，1～2ヶ月くらいをみておけばよいだろう．作業期間は一人で行った場合の目安を示した．以下でも同じである．

（3）材料集め

詳細設計と平行して，部品集めを行う．発注してから手元に届くまで結構かかる場合がある（時には数カ月も）ので，仕様が決まってきたら早めに注文を出しておく方がよい．とりあえず，基板とソケット類は詳細設計が終わったらすぐに必要になるので，早めに取り揃えておく．IC 類は配線が終わるまで必要がないので，後からでもよい．

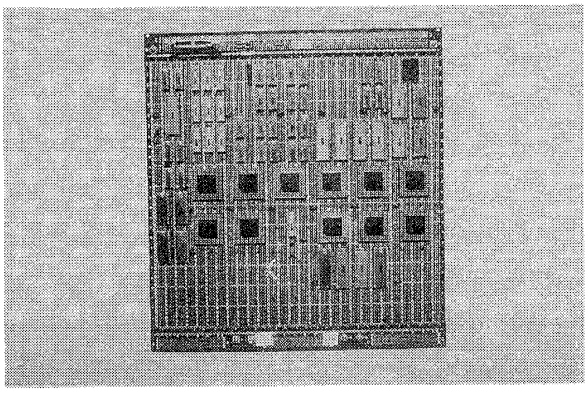

写真2　レイアウト
実際に基板にソケットを差し込んで考えるのが手取り早い．

（4）レイアウト

詳細設計が終わったら，基板上に各 IC をどう配置するかを考える．基本は配線の長さが最短になるようにする．10MHz 程度の動作クロックでは，それほど厳密に考えることはないが，長い配線はノイズの元になるので注意する．クロックは特に大切なので，中央付近に置くとよい．

レイアウトが決まったら，上から見た図と下から見た図を作っておく．上から見た図はデバッグの時に，下から見た図は配線の時に必要になる．だいたい1日くらいの作業である．

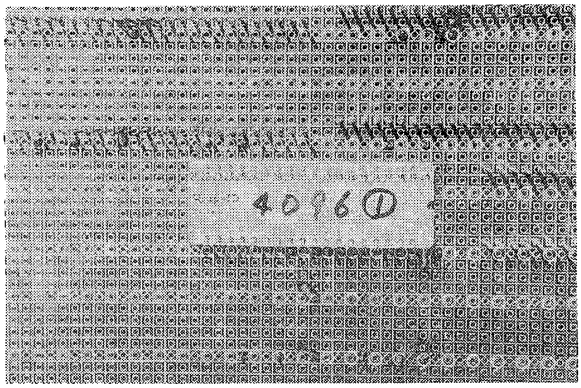

写真3　ハンダ付け
電源・グランドピンをハンダ付けし，チップ名を書いた ID カードをかぶせておく．

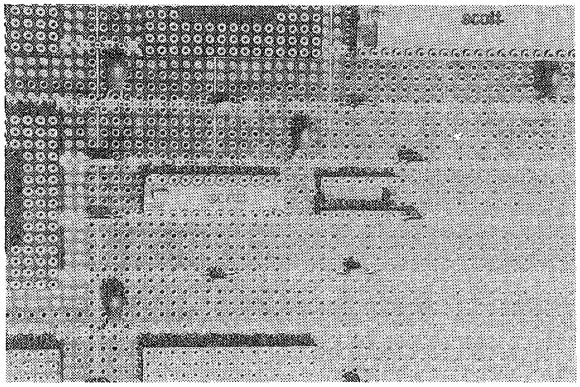

写真4　パスコン
小さいものは0.1μF，大きいものは10μF のコンデンサ．

（5）ハンダ付け

レイアウトができたら，ソケットの電源・グラウンドピンを基板にハンダ付けして，ソケットに ID カードを差し込んでおく．その際，ID カードにはチップ名を記入しておく．

その後，パスコンを基板全体に付けていく．だいたい IC 1個につき0.1μF 程度のものを1個，IC 10個につき10μF 程度のものを1個の割合である．

作業はテスタでチェックしながら進める．短絡しているのに気づかずに作業を続けていくと，短絡箇所がわからなくなり，後で取り返しのつかないことになる．作業日数は1～2日程度．

（6）配線

設計図に沿って，前回紹介したラッピング・ツールで行う．はじめは楽しいが，単調な仕事なのですぐに飽きてくる．根気よく続けよう．作業は

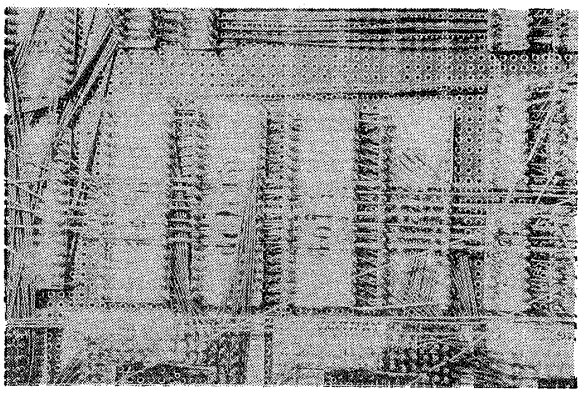

写真５　配線(1)
長い線から配線すると，たるんだ線がなくなって，見た目がきれいになるが，短い線から配線した方がデバッグのとき修正が楽である．

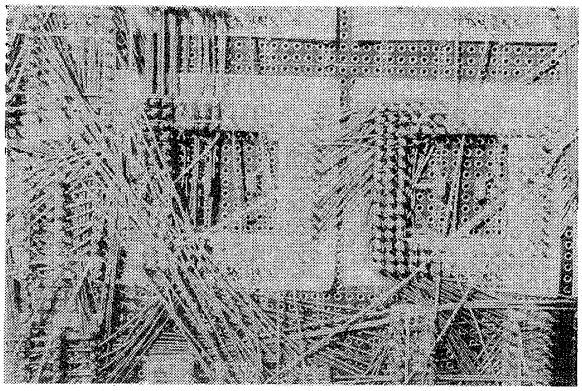

写真６　配線(2)
PGA の配線は少し大変である．

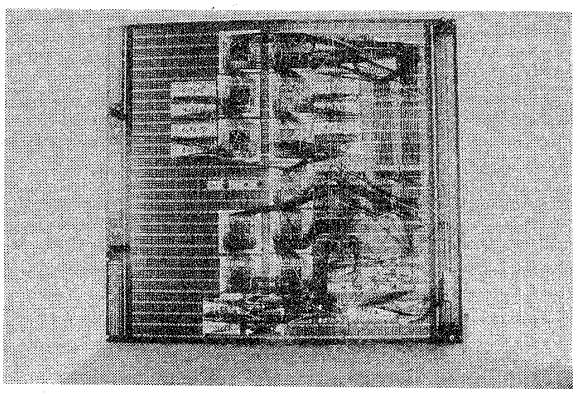

写真７　配線(3)
このくらいの回路では，配線数は全体で2,000本くらいである．いろいろな色のワイヤーを使っておくと，デバッグのとき見やすくなる．

1時間当たり数十本くらい．回路規模にもよるが，全体で１ヶ月くらいの仕事である．

（7）デバッグ

配線が終わったら，デバッグに入る．まずはじめに，IC をのせる前に電源を入れてみる．電解コンデンサの電極を間違えてハンダ付けしていたりすると，コゲてくるので，そんな場合は電源を切って修正する．

次に IC を乗せて電源を入れる．各 IC をさわってみて，異常に熱くなっているものがあったら，それには致命的な配線ミスがあるので，電源を切って調べる．電源・グランド線を間違えていないか，出力信号が重なっていないかなどを調べてみる．このようなチェックには，前回紹介したペンシル型のロジック・トーンが便利である．

さて，以上のような基礎的なチェックが終わったら，いよいよ回路を動かしてみる段階である．ドキドキしながら初めてプログラムを動かすときの，あの緊張感と期待感は，計算機を開発した者にしか味わえない．同時に，自分の開発した計算

写真８　デバッグ
IC に洗たくバサミのような IC テストクリップをつけると，ロジアナのプローブがつけやすい．

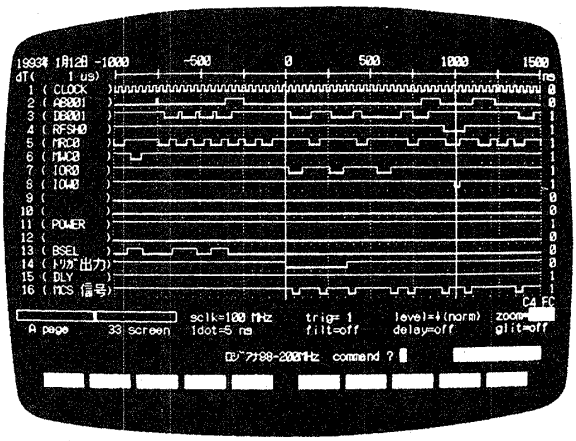

図６　ロジック・アナライザの画面表示例
（アスコム社のカタログより）

機が，うんともすんとも言わないときの，あの落胆も，開発した者にしか味わえない．普通は両方を同時に味わうことになる．

一見完成したように思えても，ここまででようやく前半戦が終わったところと考えるべきである．これから後半戦（デバッグ）が始まる．デバッグは，私の経験では，短いときで1週間，長いときには3ヶ月かかっている．

デバッグには大きく2種類ある．論理ミス（設計ミスや配線ミス）の修正と，ノイズ対策である．

まず，論理ミスの修正から行う．ノイズの影響を下げるためにクロックを1MHzくらいに落して行うとよい．デバッグの方法は，ロジック・アナライザを使って，データの流れにそって1段ずつチェックしていくのが，確実である．最初は何の反応もしなかったとしても，最後には必ず動くようになる．

論理ミスがすべてなくなったら，クロックスピードを上げてみる．設計通りのスピード（例えば10MHz）でも正常に動作したなら，そこで完成である．

もし，1MHzでは正常なのに，10MHzにしたらノイズが出るようになったとしたら，引続きノイズ対策を行う．ただし，論理ミスが必ずデバッグできるのに対し，電気的なノイズは必ずなくなるという保証はない．

対策としては，パスコンを増やしたり，クロックやファンアウトの多い信号線にバッファを入れて補強してみたりする．発熱が問題になるときは扇風機などで空冷してみる．この程度でも，かなりの効果がある．現実問題として，システムが組み上がったあとでは対策にも限界がある．

もし，それでもノイズが取れない場合は，仕方がないので，クロックアップを諦める．10MHzでは動作しなくとも，5MHzで動作したのなら，それで満足すべきである．一番大切なことはシステムを組み上げることであり，スピードアップは次に考えればよい．

デバッグ期間は一概には言えないが，1ヶ月くらいが平均的なところだろうか．

以上，大まかに作業の流れを見てきたが，トータルの開発期間は，回路規模にもよるが，半年程度である．

2．スーパーコンピュータへの道

専用計算機開発の最終目標は，スーパーコンピュータを越える，その分野においては世界最速の計算機システムを開発することである．そもそも専用計算機を開発しようとした動機は，現在の計算機パワーでは扱えない未知のシミュレーションに挑むためであったからである．この節では，専用計算機がスーパーコンピュータを越えるまでの道筋を簡単にご紹介する．

（1）専用LSIを作る

専用計算機を高速にするには，今まで基板レベルで行ってきたものをチップレベルで行うことが必要になってくる．LSI化することにより，実装密度が上がり，動作周波数も大幅に上げることが可能になる．

東京大学教養学部の宇宙地球科学教室では2年前に，牧野淳一郎さんを中心に重力多体問題専用のLSI，GRAPEチップを開発した．GRAPEチップを48個使って作った専用計算機GRAPE-3は，10Gflopsに達し，実効速度としては当時最速のスーパーコンピュータさえも上回った．

現在は，次回の筆者である泰地真弘人を中心に，別のタイプの重力多体問題専用LSI，HARPチップを開発中で，このチップをもとにした今秋完成予定の専用計算機は，現在のスーパーコンピュータより10倍速い（実効ではもっと），1Tflopsに達する見込みだということである．

GRAPEチップは富士ゼロックスとの共同開発で，大学側が負担した予算は2千万円くらいであり，HARPチップには文部省科学研究費の特別推進研究による予算2億円があてられているという．

専用LSIを作るといっても，何も特別変わったテクノロジが入ってくるわけではない．専用LSIはゲートアレイやスタンダードセルと呼ばれるデバイスで作られるのだが，これらは前回お話したPLDやFPGAと基本的に同じものだと思っ

表1　ASIC用デバイスとゲート数

デバイス	ゲート数
PLD	10～1,000
FPGA	1,000～10,000
ゲートアレイ スタンダードセル	1,000～500,000

てもらっても差し支えない．違いは何かといえば，書き込める回路規模である．

ユーザがカスタマイズしてつくるICは，ASIC (Application Specific IC) －特定用途向けIC－と呼ばれる．広義には表1に挙げるすべてのデバイスが含まれるが，ふつうはゲートアレイやスタンダードセルを指すことが多い．

ユーザからみれば，その違いはゲート数だけなので，回路規模に応じて使い分けることになる．単精度 (32bit) の加算器や乗算器は1個当たり5000ゲートくらいになるので，数値計算用の専用パイプラインを1チップに収めるためには，どうしてもゲートアレイやスタンダードセルが必要になってくる．

作業自体も規模が違うだけで基本的には同じである．ワークステーション上で，専用のCADを使って，論理シミュレーションを繰り返しながら，論理設計を行っていく．最近ではCADの仕様を標準化しようという動きもあり，そうなると，まず論理設計を行い，その後，回路規模をみてFPGAにするかゲートアレイにするか決めることが可能になる．

ただし，PLDやFPGAが自分で書き込みまでできるのに対し，ゲートアレイやスタンダードセルはデバイスの会社に発注することになる．したがって，チップができ上がった後の設計変更は容易ではない．

その上，開発コストが桁違いに高い．自分で論理設計を行ったとしても，初期開発に1000～2000万円くらいかかる．そのチップを1000個作るとすると，1個数万円として，さらに数千万円かかる．チップ単価はゲート数や量産する数によって変わる．論理設計までも外注すると，さらにまた1000万円程度かかる．

したがって，専用LSIを開発するポイントは，いかに予算を取ってくるかにかかっている．スーパーコンピュータを買うことに比べたら桁違いに安い額ではあるが，1研究室がまかなうには途方もない額である．現実には，手作りの計算機で有効性をアピールして，額の大きい予算を申請するか，所属する研究機関のメインプロジェクトに押し上げるか，企業を巻き込んで共同研究に持ち込むか，などの方策を講じなければならないだろう．かくいう私も，いまだに専用LSIを作ったことはない．

開発期間は，東京大学教養学部の状況を見てみると，半年から1年くらいのようである．

(2) プリント基板を作る

専用チップを高速に動作させるためには，基板も専用のプリント基板を設計する必要がある．外注して，費用は200～500万円くらいである．

(3) システムをグレードアップする

専用計算機側が高速になるので，それに合わせてシステム全体もグレードアップする必要がある．ホスト計算機やインターフェースを高速なものにするわけだが，最近では1000万円も出せば，かなりのものがそろう．

プリント基板の開発もシステムのグレードアップも単独にみれば高額だが，専用LSIを開発する総費用からみると1～2割程度である．

3．おわりに：専用計算機の可能性

誰でもが計算機を作れる時代がやって来ている－そのことは，数値シミュレーションが理論系から実験系に研究スタイルを変えていくことを示唆している．

今までのスーパーコンピューティングの図式は，まず汎用のスーパーコンピュータがあり，そのまわりにユーザである数値シミュレーションの研究者がいるというものであった．スーパーコンピュータは自動車でいうとF1カーに相当するだろう．ただし，実際のF1レースは年間16戦し，各サーキットに応じてマシンがチューンナップされるのに対し，現在のスーパーコンピュータは，どのサーキット (分野) でも，チューンされるこ

とはない．スピードを決定するのは，唯一，ドライバ（研究者）の腕だけである．

専用計算機というのは，そこにメカニックを導入したものに相当する．スポンサさえつけば，どちらが勝利するかは明白であろう．勝負の行方は，もはやチーム力で決定されることになる．

例えば，重力多体問題というサーキットでは，先に紹介したGRAPEは突出した能力を有する．それはまさに，ソフトウェアとハードウェアを融合したチーム力から産まれたものといえる．

同時にもう一つ大切な点は，その恩恵をもっとも受けるのが，当然のことではあるが，それを開発した者たちだということである．それは，実験系の自然科学が，すぐれた実験装置を生み出した人たちによって進歩してきた状況に似ている．

ライバルグループが有利な立場を奪い取るには，さらにすぐれた計算機を開発しなければならない．それは高エネルギー物理などが歩んできた巨大科学の道に通ずる．事実，素粒子物理学の分野では，そのような競走がすでに垣間見れる．筑波大学のQCDPAXをはじめ，ローマ大のAPE100，IBMワトソン研究所のGF11など，世界のいくつかの研究機関で素粒子物理専用計算機の開発が競われているのである[9]．この分野は，加速器の建設が予算的にみて限界に近づいているため，早い時期にも，専用計算機開発競走に主体が移っていくかもしれない．

しかしそれは，汎用計算機がなくなっていくという意味では，もちろんない．むしろ，今日の状況からみると市販の汎用計算機はますます身近なものになっていくだろう．ただそれは，科学者が使う実験装置としては，あまりにも工夫がなさすぎるのである．そこでこれからは，高度な事務処理機器としての汎用計算機から，最先端の実験装置としての専用計算機が分化していくのではないか，と思えるのである．専用計算機にはそのくらいの可能性があると私は思っているのだが，皆さんはどうだろうか．

個人的な見解をもとに，2回に渡って解説させて頂いた．お聞き苦しい点や説明が足りない箇所も多々あったと思いますが，ご容赦頂きたい．もし専用計算機という分野が成り立つとすれば，今後10年くらいがもっとも面白い時代となるはずである．それは，まだ手つかずの分野が多数残っているように思われるからである．この講座記事が専用計算機について何らかの興味を呼び起こし，少しでもお役に立てれば，幸いに思います．

参考文献

[1] Y. Iwasaki, T. Hoshino, T. Shirakawa, Y. Oyanagi and T. Kawai, Comput. Phys. Commun. **49**, 449 (1988).

[2] Publ. Astron. Soc. Japan **45** (1993) GRAPE特集号.

[3] 杉本大一郎：「手作りスーパーコンピュータへの挑戦」講談社ブルーバックス (1993).

[4] 杉本大一郎編：「専用計算機によるシミュレーション」朝倉書店 (1994).

[5] J. H. Applegate, M. R. Douglas, Y. Gusel, P. Hunter, C. L. Seitz and G. J. Sussman, IEEE Trans. Comput. **C 34**, 822 (1985).

[6] T. Yabe, T. Ito and M. Okazaki, Jpn. J. Appl. Phys. **32**, L1359 (1993).

[7] T. Ito, T. Yabe, M. Okazaki and M. Yanagi, submitted to Comput. Phys. Commum.

[8] T. Ito, T. Yabe, S. Takahashi and K. Yoshida, in preperation.

[9] 立花隆「電脳進化論」朝日新聞社刊 (1993).