3T-7

ＱＣＤＰＡＸのプログラミング環境

吉田善幸、　白川友紀、　星野　力
（　筑波大学　）

１．はじめに

量子色力学の計算を高速に行うことを目的とし、高並列計算機ＱＣＤＰＡＸの作製が現在進められている。ＱＣＤＰＡＸのホスト計算機(SUN3/260)およびＰＵ(Processing Unit)を効率よく制御し、その能力を最大限に発揮させるためには、種々のシステムソフトウェアとそのソフトウェアを有効に使いこなし並列計算機特有のコーティングやチューニング技術を持つ有能なプログラマーが必要となる。ここでは、前者に重点を置き、その有効性について述べる。

２．システムソフトウェアの概要

ＰＡＸ計算機における並列処理のモデルは、主従関係にあるホスト計算機（ＨＰＩ[HOST-PU Interface]）と多数のＰＵよりなる。ＨＰＩには、全ＰＵからの同期・放送・終了要求があれば全ＰＵを要求どうり制御する機構がある。ＰＵ中のユーザプログラムは、内部の処理と隣接ＰＵとの通信を記述するだけでよい。そのためのソフトウェアとしては、各ＰＵの内部処理と通信を記述する高級言語、同期・放送・終了のためのプリミティブがある。その他、割り込みによって起動されるＰＵ中のエラー処理とホストへの報告用のソフトウェアがある。

ユーザが記述するプログラムは、ホスト計算機用とＰＵ用に二分される。両者ともホスト計算機上で開発され、最後にＰＵ用オブジェクトがＰＵにロードされて、ホスト計算機が計算をスタートさせる。開発の流れを図１に示す。

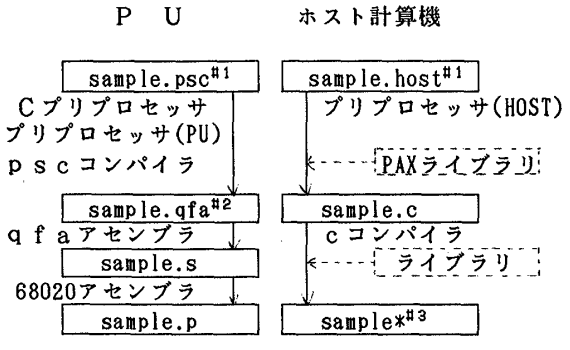

図１　アプリケーションソフト開発の流れ

（１）ホスト計算機用ソフト

主にＰＵとの入出力インターフェイスとして用いる。ここでは、ＰＵとのバスのオープン・クローズやＰＵへ実行ファイルのロードを行う。また、ＨＰＩに接続されているグラフィックディスプレイとの入出力も行う。

処理の流れは、ホスト用プログラム(*.host、拡張子としてhostを用いる、以下同様)をＱＣＤＰＡＸ専用のライブラリとリンクし、ＰＵと共通のメモリ領域を確保し、複素数の処理を行うHOST用プリプロセッサを通し、それをＣコンパイラにかけて実行ファイルを生成する。

（２）ＰＵ用ソフト

複素数を展開するPU用プリプロセッサ、Ｃ言語のサブセットをコンパイルするｐｓｃコンパイラ、新しく仕様決定された浮動小数点演算用アセンブリ言語ｑｆａをアセンブルするｑｆａアセンブラがある。

ＰＵ用プログラム(*.psc)をPU用プリプロセッサに通してから、ｐｓｃコンパイラでコンパイルし、オブジェクト(*.qfa)を出力する。ユーザは、このレベル(*.qfa)で最適化のためのチューニングを行う。そして、このオブジェクトをｑｆａアセンブラでアセンブルする。ｑｆａアセンブラは、浮動小数点演算部分だけをオブジェクト変換して出力する(*.s)。これを、68020のアセンブラにかけて、ＰＵ用のオブジェクト(*.p)が得られる。

３．浮動小数点演算機構用アセンブリ言語ｑｆａ

(Quick Floating Assembly language)

大別すると、計算を逐次的に実行するスカラ演算とスカラ演算と同等の命令群をＷＣＳに格納しておいてそれをDNA転送して実行するベクトル演算とから成る。ｑｆａは、浮動小数点演算機構の演算命令や制御命令とほぼ一対一に対応するように仕様が決められているが、一部マクロ命令を用いている。また、予めマイクロプログラムとしてメモリに格納されている初等関数を呼び出すこともできる。

（１）各演算の並列性

FPUとして用いているLSIの性能により、ALU計算とMPY計算が同時に実行できる。ベクトル演算では、初期設定のあと制御をFPUCに移すと、高速メモリにアクセスしない限りCPUとFPUの並列性が保たれる。

（２）ニーモニック表現

浮動小数点演算機構を実現するハードウェアで、実行可能な事はすべて記述できる様に仕様を決められている。

Programming Environment of QCDPAX
Yoshiyuki YOSHIDA, Tomonori SHIRAKAWA, Tsutomu HOSHINO
Univ.Tsukuba

①スカラ演算

転送命令と演算命令を一命令として記述できる。しかし、転送命令と演算命令は、逐次的に実行するのでそのことを明示するために区切り記号としてセミコロン(;)を用いる。並列に実行できる演算のALU命令とMPY命令は、区切り記号としてアンパーサンド(&)を用いる。また転送命令には、リダイレクション記号(>)を用いている。この命令群の例を図２に示す。この図の棒線はコメントを表す。

```
B > b ;                 | ﾃﾞｰﾀBをbﾚｼﾞｽﾀにｾｯﾄ
C > c ; m=b*c & a=max(b,c) ;
      ; a=a+m ;
a > A ;
```

図２　A=B*C+max(B,C) の記述例

②ベクトル演算

実際には、FPUの命令をFPU-FPUCでDMA転送することによって実行する。まず転送レジスタ(tr)・増分値レジスタ(ir)・終了値レジスタ(er)・実行開始位置(fpc)・ＷＣＳ(Writable Control Storage)を設定する。設定終了後、FPUCにベクトル演算のスタートをかける。この流れの最も単純な例を図３に示す。また、漸化式の例を図４に示す。

```
tr00: _B > b ;              | trﾚｼﾞｽﾀのｾｯﾄ
tr01: _C > c ;
tr02: _B+1 > b ;
tr03: a > _A, e ;
ir04: 1 ;                   | irﾚｼﾞｽﾀのｾｯﾄ
@lab-1:                  | 以下WCSへの設定
        tr00,ir00 ;
@lab-2:
        tr01,ir00 ;
        tr02,ir00 & a=b+c ;
        tr03,ir00 ; ja @lab-2 ;
er: _A+99 & fpc: @lab-1 & start ;
          | erﾚｼﾞｽﾀとfpcのｾｯﾄ
```

図３　A[i]=B[i]+C[i](ﾍﾞｸﾄﾙ長=100)の記述例

```
tr00: A > b ;
tr01: A+2 > c ;
tr02: a > A+1, e ;
ir01: 1 ;
@lab-1:
        tr00,ir01 ;
        tr01,ir01 & a=b ;
@lab-2:
        tr02,ir01 & a=a+c ;
        tr01,ir01 ; ja @lab-2 ;
er: A+99 & fpc: @lab-1 & start ;
```

図４　漸化式の記述例
(A[i]=A[i-1]+A[i+1]; ベクトル長=100)

４．ｑｆａの特徴

（１）MC68020用アセンブリ言語との混在

ｑｆａはMC68020用アセンブリ言語とは、表現方法を根本的に変えてあるため、この両者間で、どの様な混在をも許す。

（２）ｑｆａ命令の展開

CPUのレジスタの値をFPUのレジスタにロードできなく、また実数値も直接ロードできない。このため、その値を一時的にメモリに格納してから、それをFPUのレジスタにロードする。これらはｑｆａレベルで記述する命令が冗長になるため、ｑｆａアセンブラによって展開を行う。

（３）バイトとロングワードの混在

CPUの演算単位はバイトであり、FPUの演算単位はロングワードであるため、ｑｆａの記述では、この２種類が混在する。出力としては、入力命令により演算単位を使い分ける。例えば、trレジスタ設定の出力はロングワード単位であり、erレジスタ設定の出力はバイト単位である。

（４）ＦＰＵＣ命令モード

FPUC命令は、転送命令(12bit)・分岐命令(15bit)・演算命令(13bit)の40bitから成るが、これを32bitの命令に圧縮するために、３つのモードに分割されている。このため、ｑｆａアセンブラが最適なモードを選択して、そのモード用の出力を行う。

（５）ＷＣＳへのＦＰＵＣ命令の書き込み

WCS用のロケーションカウンタを操作することにより、WCSの任意の場所にFPUC命令を書き込むことができる。これに対応して、WCSの任意の位置より実行可能となる。ロケーションカウンタの指定はラベル行で以下のように行う。

```
@*= @lab-8 :
@*= 512 :
```

（６）ＷＣＳ内のFPUC命令の再利用

既にWCSにロードされている命令を、tr設定やir設定だけ変更して実行する事によって、ベクトル演算のたち上げのためのオーバーヘッドを減少させることができる。

（７）エラー処理

ユーザがｑｆａレベルでチューニングすることを前提としているので、文法エラーやハードウェア的に実行不可能な命令のチェックは厳密に行い、エラーメッセージを出力する。

５．おわりに

大規模なプログラム開発を行うためには、数々のシステムソフトウェアが準備され、かつそれらが高性能であることが望まれる。しかし現在のところ、高速計算を第一の目的としているため使い勝手の悪いところが残っている。将来、この使い勝手の悪さをできるだけ解消していきたい。

本研究は、科研費特別推進研究(62060001)による。

参考文献

1）吉田、白川、星野：ＱＣＤＰＡＸのアセンブラ
情報処理学会第36回全国大会7C-5