

Title	SX-3Rシリーズの言語プロセッサと開発支援ツール
Author(s)	片山, 博; 桜井, 弘之; 板垣, 治敏
Citation	大阪大学大型計算機センターニュース. 1992, 87, p. 44-52
Version Type	VoR
URL	<a href="https://hdl.handle.net/11094/65994">https://hdl.handle.net/11094/65994</a>
rights	
Note	

*Osaka University Knowledge Archive : OUKA*

<https://ir.library.osaka-u.ac.jp/>

Osaka University

# SX-3Rシリーズの言語プロセッサと 開発支援ツール

片山 博\*1 桜井 弘之\*2 板垣 治敏\*3

## 1. まえがき

今日、科学技術の分野における計算では、無限大に近い演算時間が要求されてきています。これらの要求を満足するために、さまざまなハードウェア技術により、より高速なスーパーコンピュータが開発されてきています。しかし、これらの技術の中には演算バイライン方式やマルチプロセッサ方式など、一般的なソフトウェア技術ではハードウェアの性能を十分に引き出すことは不可能でした。このような状況を踏まえ、SX-3Rシリーズでは、オペレーティングシステムSUPER-UXのもとで動作する言語プロセッサに高速化技術を導入し、ハードウェアの性能を用意に引き出せるようにしています。また、開発支援プログラムや性能向上支援ツール（以降これらを総称して開発支援ツールと呼びます）により、極限までの性能向上を容易に行えるようにしています。

（図1参照）

本稿では、これらの言語プロセッサや開発支援ツールのもつ機能や特徴のうち、SX-3Rシリーズを有効に利用できるよう拡張した部分を中心に紹介します。

表1 FORTRAN77/SXの特徴

言語仕様	JIS FORTRAN 上位水準を包含 8バイト整数などの種々の拡張
最速化機能	手続きのインライン展開 ループアンローリング インストラクションリオーダリング
自動ベクトル化機能	基本ループ、IF文を含むループ、 マクロ演算、ループ分割、外側ループ ループの一重化、入れ換え
並列処理機能	マクロタスク マイクロタスク（自動並列化）
互換性 数値範囲の拡大	CRAY、IBM、ACOS、EWSとの互換 拡張浮動小数点数（～10の4900乗）

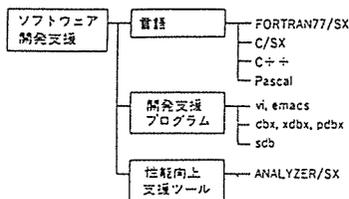


図1 ソフトウェア開発支援関連の体系図

## 2. FORTRAN77/SX

### 2.1 概要

スーパーコンピュータの世界においては、大規模科学技術計算プログラムのほとんどはFORTRANで書かれています。FORTRAN77/SXは、SX-3シリーズの高い性能を誰でもが容易に利用できるようにということを目的として開発されたFORTRAN処理系であり、特に、その優れた自動ベクトル化能力、最適化能力は、SX-2の時代から定評があります。

SX-3Rシリーズでは、SX-2で培ったこれらの技術を踏襲し、SX-3Rシリーズ向けに発展させ、性能向上を実現しています。

SX-3RシリーズのFORTRAN 77/SXはそのほかにも多彩な機能を持ち、スーパーコンピュータの性能を容易に利用できるようにしています。

表1はこれらの代表的な特徴をまとめたものです。

## 2.2 最適化

FORTRAN 77/SXは、最適化技術としてSX-2ですでに適用していた共通式削除、不変式のループ外への移動などの一般的最適化技術や命令の並べ換えに加え、次のような新しい技術を取り入れています。

### (1) 外部手続きのインライン展開

外部関数や外部サブルーチンをソースレベルで、呼び出された場所にインライン展開することにより、手続きの呼び出しにおけるオーバーヘッドを減らします。さらに得られる効果として、従来、引用側のDOループ中に外部手続き呼び出しがある場合、ベクトル化不可あるいは部分ベクトル化されていたケースも、インライン展開後のソースイメージで最適化および自動ベクトル化されるため、より高度な最適化やベクトル化が施されたオブジェクトコードを得ることができ、これによりハードウェアの性能を十分、引き出すことができます。(図2参照)

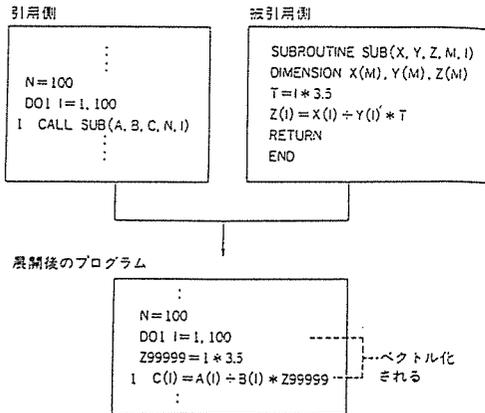


図2 サブルーチン副プログラムのインライン展開例

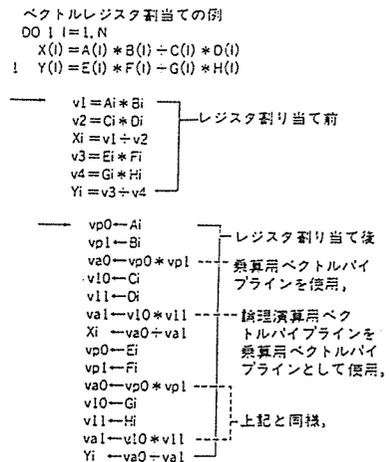


図3 ベクトル演算パイプラインの二重化を利用した例

### (2) 新アーキテクチャ対応

SX-3Rシリーズにおいては、SX-2に比べて以下に示すようなアーキテクチャの強化がなされています。

#### 1) ベクトル演算パイプラインの二重化

加減算用のベクトル演算パイプラインとシフト演算用のベクトル演算パイプラインは、互いにどちらのパイプラインとしても使用できます。同様に、乗算用のベクトル演算パイプラインと論理演

算用のベクトル演算パイプラインも、互いにどちらのパイプラインとしても使用できます。

## 2) ベクトルレジスタの容量拡大

最大144Kバイトに拡大されています(SX-2では、最大80Kバイト)。

3) 128個全てのスカラレジスタをベース用のレジスタとして使用可能(SX-2では64個)。

## 4) 新命令の追加

8バイト整数型関連命令(四則演算、型変換、シフト演算などのスカラ/ベクトル命令)、ベクトルロード命令に対する機能強化命令(データをメモリからロードするとき、ベクトルデータレジスタへも同時にロードする命令)、通信レジスタ関連命令(並列処理制御用レジスタに関する命令)などが追加されています。

FORTRAN77/SXは、上記を考慮して、SX-2より行っていたハードウェア動作のシミュレートによる中間コードレベルと機会コードレベルの並べ換えを発展させています。これによりレジスタ更新待ちや演算パイプライン待ち、演算パイプラインの空きができるだけ発生しないようにしています。また、プログラム中のループを基本としたレジスタ割り当てにより、レジスタ間の競合やレジスタと記憶域との間の移送を減少させています。これらの技術により、ハードウェアの性能を引き出すために最適なオブジェクトコードを生成し、プログラムの実行時間の短縮を図っています。(図3参照)

## 2.3 自動ベクトル化機能

FORTRAN77/SXは、自動ベクトル化機能についても種々の強化が行われています。ここでは、新たに実装された機能の主なものを中心に説明します。

### (1) 行列演算の高速処理

これはプログラム中に存在する行列と行列または行列とベクトルの積を計算するループを、コンパイラが自動的に検出し、その部分を内部的に用意した高速ライブラリ呼び出しに置換してしまうものです。この高速ライブラリルーチンは、ハードウェアの性能を最大限引き出せるように作成されたものであり、普通にベクトル化する場合と比較して格段の性能向上が見られます。図4はコンパイラが行う置き換えを示したものです。配列の参照パターンとしては、例えばA(O, O, ···, I+K, ···, 2\*J, ···, O)のようなものも行列A<sub>ij</sub>として認識します。またループの構造については、配列の初期化を含むもの、値が一定の配列要素をループ外に移動したものを認識することができます。(図5参照)。

### (2) 自動アンローリング機能

アンローリングとはループ長を1/nにしてループ内の演算の数をn倍にするチューニング技法のことです。このようにすることにより、演算の数に対するメモリへのアクセス数が減少し、実行時間の短縮が可能になります。図6に示した例の場合、下線を施した配列要素の引用をレジスタ上

```

DIMENSION A(LDA, *), B(LDB, *), C(*)

DO 10 I=1, N
  DO 10 J=1, N
    DO 10 K=1, N
      10  C(I, J) = C(I, J) + A(I, K) * B(K, J)
          :
          !
          :
    DO 10 J=1, N
  10  CALL lib1(A, LDA, B, LDB, C, I, N, N)

```

図4 行列積のライブラリ呼び出しへの置き換えの例

(1) 初期化を含む形のループ

```

DO 10 I=1, N
  DO 20 J=1, N
    20  C(I, J) = 0.0
  DO 30 K=1, N
    DO 30 J=1, N
      30  C(I, J) = C(I, J) + A(I, K) * B(K, J)
  10  CONTINUE

```

(2) 不変な要素がループ外へ移動された形のループ

```

DO 10 I=1, N
  DO 10 K=1, N
    T = A(I, K)
    DO 10 J=1, N
      10  C(I, J) = C(I, J) + T * B(K, J)

```

(3) 加算を変数上で行う形のループ

```

DO 10 I=1, N
  DO 10 J=1, N
    T = C(I, J)
    DO 20 K=1, N
      20  T = T + A(I, K) * B(K, J)
  10  C(I, J) = T

```

図5 ライブラリ呼び出しに置き換えられるパターン

```

DO 10 K=1, 1000
  DO 10 I=1, 1000
    10  X(I) = X(I) + A(I, K)
        !
  DO 10 K=1, 1000, 2
    DO 10 I=1, N
      X(I) = X(I) + A(I, K)
    10  X(I) = X(I) + A(I, K+1)

```

図6 アンローリングの例

```

DO 10 I=1, K
10  A(I) = A(I+L) + B(I)
    !
    IF (LGE.0) GOTO 1
    IF (LGT.-K) GOTO 2
1  DO 10 I=1, K
10  A(I) = A(I+L) + B(I) } ベクトルで実行
    GOTO 3
2  DO 20 I=1, K
20  A(I) = A(I+L) + B(I) } スカラで実行
3  CONTINUE

```

図7 条件ベクトル化の例

```

DO 10 I=1, N
  IF (D(I).LT.0.0) THEN
    IF (A(I).GT.B(I)) THEN
      T = A(I)
    ELSE IF (B(I).GE.0.0) THEN
      T = B(I)
    ELSE
      GOTO 10
  ENDIF
  C(I) = T
  ENDIF
  T = D(I)
10  CONTINUE

```

図8 IF条件下における変数の定義/引用の解析機能強化によるベクトル化の例

```

DO 10 I=1, N
  X = A(I)
  B(I) = X * 2
  A(I+1) = B(I) + 1.0
10  CONTINUE

```

図9 複数の文にまたがるマクロ演算の認識によるベクトル化の例

で行うことができ、メモリへのアクセス数が減少します。また、演算数を増やすことにより4本のベクトル演算パイプラインを効率的に動作させることもでき、ハードウェアの高速性能を十分に引き出すことができます。

本機能は最適化オプションにより制御することができ、以下の処理を行います。

1) 対象となっているループをアンローリングしても、データの依存関係が正しく保たれるか否かをチェックする。

2) データの参照パターンが、アンローリングによる効果が期待できるか否かをチェックする。

3) 1)、2)でアンローリングすべきと判断されたループをアンローリングする。ループ長がアンローリングする段数の倍数でない場合、端数を処理するループも自動的に生成する。

また、利用者が個別のループを指定してアンローリング機能を働かせることもできます。この場合、コンパイラはデータの依存関係が正しく保たれることのみをチェックして、アンローリング処理を行います。

### (3) 条件ベクトル化機能

コンパイラはD Oループ中に存在する配列の定義・参照関係やD Oループの繰り返し回数を解析して、ベクトル化して良いか否かを判断します。しかし、配列の添え字の値や繰り返し数が実行してみないと確認できない場合、コンパイラはベクトル化して良いか否かが判断できないことがあります。たとえば、下記D Oループでは、

$$D O \quad I O \quad I = 1, K$$
$$I O \quad A(I) = A(I+L) + B(I)$$

$L \geq 0$  または  $L \leq -K$  であるときにはベクトル化して実行することができるのですが、そのほかの場合ベクトル化での実行ができないため、以前はベクトル化不可として処理していました。本機能は、これらのベクトル化の条件を実行時に判断できるようにして、その条件によりベクトル化された命令列を実行したり、通常のスカラの命令列を実行したりできるように、オブジェクトコードを生成するようにしたものです。上の例の場合、図7のように変形されたオブジェクトコードを生成します。したがって、 $K$ や $L$ の値を実行時に判断することにより、可能な限りベクトル化された命令列が実行できるようになります。

このほかにも、I F文による条件下で定義されている変数に関する解析機能の強化(図8参照)や複数の文にまたがるマクロ演算の認識(図9参照)など、S X-2から行っている機能の大幅な強化を行っています。また、従来からベクトル化されているループでも、S X-3 Rシリーズのハードウェアの特徴を生かした命令列が生成されるように改善を加えています。

## 3. C/S XとC++

### 3.1 概要

C/S XはC言語プログラムを翻訳し、S X-3 Rシリーズ向けに最適化されたオブジェクトコ

ードを生成するCコンパイラであり、以下の機能を備えています。

- ・最適化機能
- ・自動ベクトル化機能

C言語は、ベル研究所でB. W. KernighanとD. M. RitchieによりUNIXシステムの記述言語として開発された言語であり、その言語仕様は、両開発者の頭文字をとってK&R Cと呼ばれています。一方、近年C言語が普及し、さまざまな処理系にCコンパイラが実現されるようになると、処理系によって言語仕様が異なり、ある処理系で記述されたCプログラムを他の処理系に容易に移植できない状況になってきました。これは、K&R Cの仕様が厳密に定義されていないということおよびライブラリ関数に関する記述がないということに起因しています。これらを解消し国際的な言語仕様を作成することを目的としてANSI (American National Standard Institute: 米国規格協会)のもとに委員会が発足され、6年間の検討の結果、標準規格としてANSI Cが規格化されました。また、ISO (International Organization For Standardization 国際標準化機構)において、本規格を国際規格にする作業が進められています。

C/SXでは言語仕様としてANSI CおよびK&R Cをサポートしています。また、SX-3 Rシリーズの性能を引き出すための拡張が行われています。

C++はC言語を基にした汎用プログラミング言語です。Cから発生した言語ということで、C記述を基本的に受け付け、さらにクラス、インライン関数、演算子や関数の多重定義、参照、関数の引き数チェック、型変換などの機能が加えられています。これらにより、「データ抽象化」や、「オブジェクト指向プログラミング」といった手法でプログラミングを行うことができます。

C++は現在でも成長を続けており、1991年(平成3年)9月下旬にUSL (Unix - System Laboratories, Inc.)からC++R3.0がリリースされています。また、Cと同様にANSI、ISOで標準化の作業も行われています。

SUPER-UXでは、現在AT&T C++ Language System Release 2.0に、SUPER-UX用に拡張を加えたC++R2.0を提供しています。

C++は、C++ソースをCソースに変換し、変換されたCソースを上記のCコンパイラを使ってコンパイルするトランスレータです。したがってC++で書かれたプログラムはCプログラムにトランスレートされるので、Cコンパイラが提供している最適化機能や自動ベクトル機能の恩恵も受けることができます。

### 3. 2 最適化機能

C/SXは最適化機能として次のような機能を持っています。

#### (1) 関数のインライン展開

C/SXコンパイラは、ファイル中に宣言された関数に対する呼び出しがあると、可能な限りそ

の関数呼び出しをインライン展開することにより、関数呼び出しのオーバーヘッドを削除します。

#### (2) コードに関する最適化

C/SXコンパイラは、制御フロー解析、データフロー解析技術によって、プログラムを解析し、不要なデータの移し換えを極力排除することで、等価でより高速な処理に変換します。

#### (3) 広域レジスタ割り付け

C/SXコンパイラは、豊富なレジスタを利用し、頻繁に参照されるデータをレジスタに割り付けることで、データ参照の高速化を図ります。

本コンパイラにおけるレジスタの割り付けは、フロー解析によって得られるデータの生存範囲や参照状況を考慮した高度な技術で実現されています。これにより、割り付け効果の高いデータを、最適な範囲でレジスタに割り付けることが可能です。

#### (4) 命令の並べ換え

SX-3Rシリーズのハードウェアは、ベクトル演算に関しては、多重並列パイプライン方式、スカラ演算に関しては、演算パイプライン方式によって高速な処理を実現しています。しかし、命令の開始および実行順序によっては、レジスタ更新待ち（演算結果を後の命令で使用しようとしたときに生じる）による遅れが発生し、SX-3Rシリーズの性能を十分に引き出せない場合が生じます。

C/SXコンパイラは、ハードウェアの動作をある程度シミュレートし、実行結果の政党性が保たれるようにオブジェクトコードを並べ換えることによって、レジスタ更新待ちや演算器待ちなどができるだけ生じないようなオブジェクトコード列を生成します。

### 3.3 自動ベクトル化機能

C/SXコンパイラも、自動ベクトル化機能を有しています。C/SXにおけるベクトル化は、forループを対象とし、int, long, long long, float, doubleなどの型を持つ数値演算に対して適用されます。この機能により、行列演算などを非常に高速に行うことができます。

## 4. 開発支援ツール

### 4.1 概要

SX-3Rシリーズでは開発支援ツールとして、UNIX上で一般に提供されているツールのほかに、FORTRAN 77/SXによりSX-3Rシリーズのハードウェア性能を引き出すためのプログラム開発を容易にするために、種々の性能向上支援ツールやデバッグツールを備えています（図1参照）。

これらは、ベクトル処理や並列処理といったSX-3Rシリーズ特有の機能に対して支援を可能としており、利用者の高性能なプログラム開発のために大いに役立てることができます。特に性能向上支援ツールは、チューニング作業の全般にわたって作業を円滑に行うことを支援しています。

## 4.2 ANALYZER/SX

ANALYZER/SXは、FORTRAN 77/SXで作成されたプログラムのベクトル化および並列化に関するチューニングのための、静的および動的特性を解析する性能向上支援ツールです。解析手段としては、ソースプログラムを解析してプログラム全体のモジュール構造などの静的特性を解析する静的解析と、実際にプログラムを実行して実行時間や実行回数など、プログラムの動的特性を解析する動的解析があります。この解析で得られた情報は、リスト形式で出力することができます。

## 4.3 デバッガ

SX-3Rシリーズには標準的なデバッガとして、シンボリックデバッガd b xおよびs d b gがあります。

特にd b xはBSDシステムの標準的なシンボリックデバッガでわかりやすいコマンドインタフェースを備えており、他のUNIXシステムでも多くのユーザに使用されています。

スーパーコンピュータでは大規模なプログラムを開発、実行することが主目的であり、多くの場合FORTRAN言語が使用されています。SX-3Rシリーズにおいても、プログラム開発の中心はFORTRAN 77/SXによる記述であり、FORTRAN 77/SXにより作成したプログラムのデバッグを容易にすることが重要です。そこで、d b xでは次のような機能の強化を行い、開発支援を強固なものにしています。

- (1) d b xの諸元を拡大し大規模なプログラムを扱えるようにしています。
- (2) FORTRAN言語向けに次の機能を追加しています。
  - ① 複素数型、論理型、文字型、4倍精度実数型データのサポート。
  - ② 7次元配列と添え字に上限/下限の指定をサポート。
  - ③ 擬字法仮配列、擬文字長配列の取り扱いを可能とした。
  - ④ 拡張浮動小数点形式のサポート。
- (3) SX-3Rシリーズのアセンブラレベルでのプログラムデバッグを容易にするため次の機能を追加しています。
  - ① 逆アセンブル機能
  - ② ベクトルレジスタの参照更新機能

このような基本機能を備えるd b xに加えて、SX-3RシリーズではX W i n d o wシステムの動作するワークステーション上にd b xのグラフィカルユーザインタフェースをサポートするビジュアルデバッガx d b xもサポートしています。x d b xはSX-3Rシリーズ上で動作するクライアントプログラムで、SUNやEWS 4800などのX W i n d o wが動作するワークステーション上にx d b xのウィンドウを表示し、ソースプログラムを表示しながらマウスによる操作でブレークポイントを設定したり、データの内容を調査するなどのデバッグ作業を行うことができま

す。

- [執筆者紹介] \*1 **NEC** 第一基本ソフトウェア開発本部第一言語開発部  
\*2 **NEC** 第一基本ソフトウェア開発本部第二言語開発部  
\*3 **NEC** スーパーコンピュータ販売推進本部