

並列推論マシン PIM/i の開発支援環境 — 翻訳系 —

佐藤正俊、久野英治

masatosi@okilab.oki.co.jp

沖電気工業株式会社

1 はじめに

我々は、第五世代コンピュータプロジェクト [1] の一貫として並列推論マシン PIM/i の研究開発を行っている [2]。本稿では、PIM/i 上の KL1 処理系開発の支援環境である翻訳系について述べる。KL1 を PIM/i で実行するには、KL1 を抽象機械語の KL1b にコンパイルし [3]、さらに KL1b を PIM/i の機械語に変換し直す方法を取る。ここで、我々は、KL1 から KL1b へのコンパイラは、(財)新世代コンピュータ技術開発機構 (ICOT) で開発されたものを使用し、KL1b から機械語への変換を支援環境として整備した。

ここでは、KL1 から PIM/i の機械語に翻訳する方法を翻訳例に従って述べる。

2 翻訳系の構成

KL1 から機械語にコンパイルする処理は、以下のように入別できる。

1. KL1 コンパイラ
2. KL1b ポストコンパイラ
3. アセンブラ / リンカ
4. テンプレートジェネレータ

ここで、KL1 コンパイラは、KL1 のソースプログラムを KL1b 命令列に変換するもので、ICOT で開発されたものを使用する。以下、2 から 4 について説明する。

3 KL1b ポストコンパイラ

KL1b ポストコンパイラは、KL1b 命令列、テンプレートデータベース、外部参照ラベル情報をもとに PIM/i アセンブラ (以下単にアセンブラと呼ぶ) コード列に変換する。テンプレートデータベースは、KL1b がアセンブラにどのように変換されるかを規定しているデータベースである。外部参照ラベル情報は、テンプレート中で参照するラベル名とアドレスである。テンプレート

データベース、外部参照ラベル情報は、テンプレートジェネレータで生成される。

KL1 プログラム

```
go(a) :- true | true.
```

を例にとると、KL1b 命令列は、以下のようになる。

```
procedure((go),(1)).
label((go)/(1)).
try_me_else((go)/(1)/(1)).
wait_constant((a),(1),(go)/(1)/(1)).
proceed.
label((go)/(1)/(1)).
suspend((go)/(1)).
```

KL1b ポストコンパイラは、以下のものを生成する。

```
:: listing [predicate_id,go,1].
:: label(go_1).
:: listing [(listing),(wait_atom)].
wait_atom(a00,a,'Instr!Label!0',go_1_1)
:: label('Instr!Label!0').
:: listing [(listing),(proceed)].
proceed('Instr!Label!1',none)
:: label('Instr!Label!1').
:: label(go_1_1).
:: listing [(listing),(suspend)].
suspend(go_1_1,'Instr!Label!2',none)
:: label('Instr!Label!2').
:: listing [end_procedure].
```

try_me_else は、wait_atom の飛び先ラベルで吸収している。wait_constant は、コンスタントの種類により、更に命令を最適化し、wait_atom に変換している。

wait_atom や proceed は、テンプレートジェネレータで生成されたマクロである。テンプレートデータベースが定義ファイルであり、アセンブル時に wait_atom や proceed は、マクロ機構によりアセンブラに展開される。

また、:: で始まる行は、アセンブラに対するコマンドであり、:: label は、ラベルを定義するもので、KL1b 命令間等で必要なラベルをポストコンパイラで生成している。:: listing は、アセンブル時のリスティング情報に使用している。

外部参照ラベル情報は、テンプレートデータベースで指定された情報を基に外部参照ラベルをまとめる。

*A Developing Support Environment for Parallel Inference Machine PIM/i — compiler —
Masatoshi SATO, Eiji KUNO,
Oki Electric Industry Co., Ltd.

4 アセンブラ / リンカ

アセンブラは、アセンブラコードをオブジェクトコード、シンボル情報、ラベル情報に変換する。シンボル情報は、ストリング列とアトム番号の対応表であり、ラベル情報は、外部参照ラベルとオブジェクトコードでのアドレスの対応表である。

本アセンブラは、プログラム開発が効率良くできるように2種類のアセンブラレベルを設定し、マクロ機能を使用可能としている。

アセンブラのレベルは、プログラム開発者向けの高レベルアセンブラと機械語に1対1に対応した低レベルアセンブラ(主に、テストプログラムの自動生成用)の2種類である。アセンブラでは、高レベルアセンブラで記述されたソースプログラムを低レベルアセンブラの命令列に変換し、低レベルアセンブラのニーモニックから機械語への変換が行われる。

コード内ラベルの解決は、このアセンブル時点で行われる。外部参照ラベルの解決、アトム番号及び述語識別子の決定はリンク時まで延期される。アセンブラで生成されるファイルは、リロケータブルファイルの他にデバッグサポート用としてマクロ展開後のソースと機械語の対であるリスティングファイルがある。

マクロ機能は、C言語のプリプロセッサを使用し、C言語がサポートしている全てのマクロ機能が使用できるようにした。

5 テンプレートジェネレータ

KL1b ポストコンパイラがアセンブラに変換する過程で必要になる情報(テンプレートデータベース、外部参照ラベル情報)をテンプレート定義から生成する。テンプレート定義は、各KL1b毎にKL1bがアセンブラにどのように変換されるかを規定したものである。

wait_atom の場合を例に説明する。以下がテンプレート定義である。ここで、%i の行は、wait_atom から参照する外部参照ラベルを宣言している。% は、テンプレート定義中で出現するレジスタで、ポストコンパイラがKL1b から引き継ぐレジスタ名を宣言する。同様に、%c は、整数やアトムなどコンスタントとして使われる引数を宣言する。また、a00 や a は、デバック時の単体テスト用初期データを示しており、デバックモードでテンプレートを生成する時に使用される。この他に、suspend の第一引数のラベルは、%l と宣言する。これらの宣言は、アセンブル時の処理の違いにより必要となる。テンプレート定義は、%(と%)の間に記述する。

```
/**** wait_atom Ai atom ***/  
%i Sub4_psv_deref_if_unbound_then_push  
% Ai a00  
%c C_ATOM a  
%{ wait_atom Ai, Atom  
  gr00 := Ai, if ( notREF( Ai )) goto $start.
```

```
  jmpsub Sub4_psv_deref_if_unbound_then_push.  
$start:  
  Ai := gr00, if ( notATOM(gr00) ) goto $not_atom.  
  SetAtom(gr00, C_ATOM).  
  gr00 := -gr00 + Ai, if (zero) goto Next_inst.  
$not_atom:  
  goto Alter.  
%} end of wait_atom
```

このテンプレート定義は、テンプレートジェネレータにより、以下のように変換される。ここで、Next_inst は、次のKL1b 命令を、Alter は、次候補のKL1b 命令を示すラベルである。:: listing は、アトム番号をリンク時まで延期するための情報である。

```
#define wait_atom(Ai,C_ATOM,Next_inst,Alter) \br/>  r00 := Ai, if ( tag! Ai == 1 ) goto $start. \br/>  jmpsub Sub4_psv_deref_if_unbound_then_push. \br/>$start: \br/>  Ai := r00, if ( tag!r00 == 5 ) goto $not_atm. \br/>  r00 := 1000, merge_tag(tag(5,0) !r00 , 7 ). \br/>:: listing [symbol, C_ATOM ,1 ,0,15]. \br/>  (31,16)!r00 \= 1000. \br/>:: listing [symbol, C_ATOM ,2 ,0,15] . \br/>  r00 := -r00 + Ai, if (zero) goto Next_inst. \br/>$not_atm: \br/>  goto Alter .
```

6 おわりに

PIM/i での開発支援環境である翻訳系のKL1b ポストコンパイラ、アセンブラ / リンカ、テンプレートジェネレータについて述べた。これらの開発支援環境は、クロスシステム上で、現在、稼働中であり、これを用い、KL1 処理系のサブセットを作成 / 評価を進めている。今後は、ここで得られた知見を基に、フルセットのKL1 処理系を、KL1 で記述したセルフシステムとして開発する予定である。

謝辞

日頃、御助言を頂く ICOT 第一研究室、および、沖電気
の PIM 担当者諸氏に感謝する。

参考文献

- [1] A. Goto et al. Overview of the Parallel Inference Machine Architecture (PIM). In FGCS 1988, pp 208 - 229, Nov. 1988.
- [2] 大原他: 並列推論マシン PIM/i の概要、情報処理学会第 40 回全国大会、pp 1185-1186(1990).
- [3] Y. Kimura et al. An Abstract KL1 Machine and its Instruction Set. In SLP'87 (1987)