

汎用アセンブラのPSI上への移植

立野 裕和

高木 茂行

(三菱電機(株))

(財) ICOT

1. はじめに

ICOTにおいて、PSIの開発用に作成された汎用アセンブラ(1)をPSI上へ移植した。

移植した汎用アセンブラは、DEC 20をTSSで利用した場合、その利用者が非常に少ない状態での実質的なアセンブル速度(応答速度)注1)と同等の性能を示した。

移植に要した人工は3人月であった。

本報告では、移植で得られたPROLOGからESPへの書き換えに関する知識を中心に述べる。

注1) 応答速度とは、アセンブラを起動後アセンブルするマイクロプログラム名を入力した時刻からアセンブルが終了した時刻までの経過時間。

2. 汎用アセンブラ概説

汎用アセンブラの構成を図1に示す。機械定義をハードウェアに合わせ決められた形式で記述する。この機械定義を前処理プログラムに入力することで機械定義部が得られる。機械定義部とアセンブラ核部を合わせて専用アセンブラが得られる。

汎用アセンブラのモジュールは表1に示した4種のモジュールから構成されている。

3. 移植

PSIのシステム記述言語はESP(Extended self contained prolog) (2)と呼ばれ、PROLOGにオブジェクト指向が付加されたものである。PSI上の全てのプログラムはこのESPで記述せねばならない。今回の移植作業では、ESPの特長であるオブジェクト指向性を利用することはあまり考えていない。(デーモンの機能を少し利用した程度)

即ち、表1に示した各モジュールを1クラスとするようなコーディングとなっている。移植作業では以下の2点の扱い方が問題となった。

- (1) 副作用の問題 (assert/restruct系の相込述語)
- (2) ESPでサポートされているマクロ展開機能

表1 汎用アセンブラモジュールリスト

プログラム名	モジュール名
前処理プログラム	GEN, SUB
アセンブラ核部	ASM, SUB
リンカ	LK, SUB

(1) 副作用の問題

汎用アセンブラではassert/restructが大域情報の記憶場所として活用されている。大域情報としては、アセンブルしようとするマイクロプログラムのアドレスや、ラベル名などのアトミックなデータ、及びタームの類である。これらのデータはESPスロットやSIMPOSの提供するプールサブシステム(3)の機能を利用して実現することができた。ここでは以下にターム(Stack object)の扱いについて述べる。

汎用アセンブラは、アセンブルしようとするマイクロプログラムの表記方法をequationと呼ばれる方法でマイクロプログラムが自由に変更する機能を提供している。

```
:: ( r=0 ) equ zero_flag
   if r=0 goto label.
```

図2 equ命令の利用例

今、仮りにマイクロプログラムのシンタックスとしてzero_flagなるものがあつたとする。図2に示したように、r=0とzero_flagをequateすることで、それ以後r=0とzero_flagは同じ意味としてアセンブラは解釈する。即ち、アセンブルしようとするマイクロプログラム中の図3に示したような宣言から、aをkeyとして検索すればbが得られるようなテーブルをアセンブラは作成する必要がある。

```
:: a equ b.
```

図3 equ命令による定義

一般に、a,bはstack vectorなのでファイルから読み込んだa,bともにheap objectに変換した後に、SIMPOSのプールサブシステムを利用してテーブルを作成した。

(2) マクロ展開

前処理プログラムは機械定義からESPのプログラムを自動生成するプログラムである。機械定義中には、任意のオペレータ宣言がなされる。このため、これらのオペレータがESPで規定しているマクロ展開の対象となり、生成されたプログラムが意図と異なるものになる事がある。

これらを防ぐため、前処理プログラム中では、マクロ展開を抑止するために、termを必要に応じてクォートする必要があった。

The Conversion of the General Purpose Assembler to PSI

Hirokazu Tateno ¹⁾Shigeyuki Takagi ²⁾

1) MITSUBISHI ELECTRIC

2) ICOT

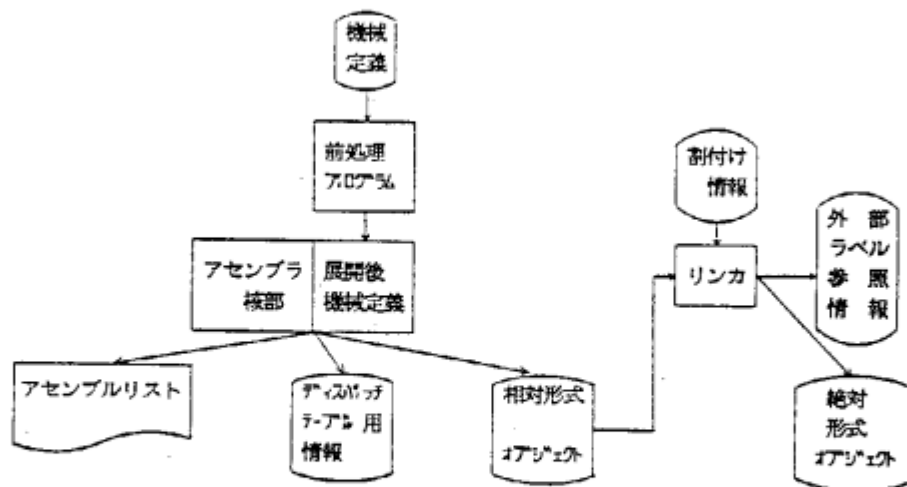


図1 汎用アセンブラの構成

4. 評価と高速化

アセンブラの実行速度を測定した。測定には、PSIの組込述語 add のマイクロプログラムソース(96steps)及びガーページコレクタの1モジュール gc_mark(1253steps)を利用した。

表2にその結果を示す。DEC 20上のアセンブラと比べ約4倍の実行時間がかかり、応答速度においても約2倍の差がある。(PSIではCPU時間、応答速度ともほとんど同じである)。そこで移植の頁で述べた図2のテーブルの扱い方を変更することを考えた。

表2 応答速度

	PSI	DEC 20 (cpu time)
add	122秒	65秒 (32秒)
gc_mark	1597秒	720秒 (620秒)

注) ()内はcpu時間

アセンブラはマイクロプログラムを一行読み込むごとに数回〜10数回テーブルを検索する。このテーブルの検索には、図4に示すように毎回termからstringへの変換処理が必要である。

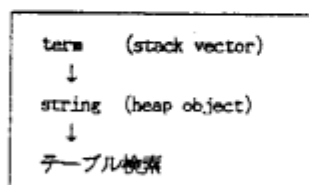


図4 テーブル検索の前処理

そこで、図3のような一行を読むたびにそれを SIMPOS の提供する標準入出力の機能を利用し buffering しておく、

全ての equation 部分が終了した時点で buffer から equation の情報を読み出し、動的にライブラリへ登録するように変更した。

この変更により、アセンブル時間は表3に示すように高速化された。

表3 高速化の効果 (応答速度)

	P S I		DEC 20
	高速化前	高速化後	
add	122秒	70秒	65秒
gc_mark	1597秒	870秒	720秒

5. おわりに

今回の報告では、汎用アセンブラの移植において、得られた知識を紹介した。又、移植した汎用アセンブラは、DEC 20上のそれとほぼ同等の実行性能を示し一応の目標を達成した。

現在、移植した汎用アセンブラを利用し、PSI-II用の専用アセンブラもPSI上に準備されている。

移植作業中、SIMPOSの開発メンバーから多くの有益な助言を頂いたことを感謝しております。

参考文献

(1) 汎用型マイクロプログラム・アセンブラ
高木茂行 TR-021

(2) ESP Reference Manual
近山 隆 TR-044

(3) SIMPOS使用説明書

(4) KLF 組込述語説明書