2G－5

1つの実行形式で複数の浮動小数点表現を扱えるコードを生成する言語CコンパイラCAT／N

中原雅彦，早川栄一，岡野裕之，並木美太郎，高橋延匡
（東京農工大学　工学部　情報工学講座）

1．はじめに

我々の研究室では，複数の浮動小数点表現法の評価を行うことを目的として，OS／omicron用言語Cコンパイラ CATに複数の浮動小数点表現法（現在はIEEE[1]とURR[2]）を実現している[3]．また，演算の高速化とURRの実用研究を目的に，URRの演算を行うハードウェア（以下URRプロセッサと呼ぶ）の試作を行い[4]，このURRプロセッサや市販のIEEE用プロセッサMC68881に対応したコード生成を行うCAT／Nの開発も行ってきた[5]．このうち，1つの実行形式で複数の浮動小数点表現法を扱うものにエミュレータ・トラップによる方式が実現されていたが[3]，実行速度が遅いという問題があった．今回、この問題を改善する方式を考案した．

2．CAT／Nの浮動小数点に対するコード生成[5]

CAT／Nでは，浮動小数点に関してユーザの使用目的に応じて自由に選択できる次の4種類のコード出力形式を有している．

(1) 関数コール型（URR，IEEE）
(2) URRプロセッサ対応型
(3) MC68881対応型
(4) エミュレータ・トラップ型（実行時選択型）

このうち(1)はハードウェアを持たない場合に使用する．(2)，(3)は各々の浮動小数点方式の高速演算用である．(4)が今回改良の対象となったコード生成方式で，主として浮動小数点表現法の評価を目的として使用するものである．

3．実行時選択型の設計方針とその問題点

我々は，複数の浮動小数点表現法を評価する環境として，以前から次のような設計方針のもとに開発を行っている[3]．

(1) 1回のコンパイル・リンクの作業で（すなわち1つのロードモジュールで）複数の浮動小数点表現法が適用できること
(2) 比較対象となる浮動小数点表現法の数を限定しないこと

このような環境を実現する場合に次のような問題がある．

(1) ロード時に定数変換を行うための情報をどのように持たせるか．

上記のような設計方針を実現するためには，プログラム中に現われる定数をコンパイル時に内部表現に変換することはできない．このため，以前に開発されたエミュレータ・トラップ方式では，定数はロードモジュール中に文字列のまま残し，実行時にインタプリートし，対応する表現形式に変換していた[3]．これがエミュレータ・トラップ方式における実行速度低下の大きな原因の一つであった．この問題を解決する方法として，コンパイル後リンケージエディットし，その後でロードを行うが，そのプログラムロード時に変換を行う方式がある．この場合，何等かの形でロード時に定数変換を行うための情報をロードモジュールに持たせる必要がある．

(2) ロード時に，選択された浮動小数点表現に対応した演算ルーチンをリンクするにはどうしたらよいか．

選択する浮動小数点表現によって演算方法は異なる．特に，演算にハードウェアを利用しようとすると，ハードウェア間のインタフェースの差を吸収する必要が生じる．そのためロードモジュール中では関数呼出しの形にしておき，関数をコールした先でハードウェアによる演算を行う方式となる．しかし，これらの演算ルーチンをリンク時にリンクしてロードモジュールに含めてしまうと，上記(2)の目的，すなわち，浮動小数点表現の数を限定しないということと矛盾が生ずる．このため，プログラムロード時に選択された浮動小数点に対応する演算ルーチンを決定し，ロードする方式にしなければならない．

これらの問題点に対して我々は，ロードモジュール中へ型情報を導入することで解決した．以下の章でこの機能について述べる．

4．型情報

ロード時に浮動小数点定数の変換や演算実行ルーチンとの結合を行うためには，それを行うための情報が必要となる．このためのロードモジュールの情報を型情報と呼び，この型情報として次のものを設定する．

(1) 型の名前
(2) 初期化したい情報．型の名前と実体の組で表す
(3) 初期化される対象のアドレスと初期化情報

ローダは，(1)の型名からロード時にどのような初期化を行うかを知ると同時に，対応するルーチンを決定し，

The C Compiler CAT／N generating Object Codes for Multiple Schemes of Floating Point Representation
Masahiko NAKAHARA, Eiichi HAYAKAWA, Hiroyuki OKANO, Mitarou NAMIKI and Nobumasa TAKAHASHI
Department of Computer Science, Faculty of Technology, Tokyo University of Agriculture and Technology

ロードする．そして，(2) の実体情報と型名からロードモジュールに実際に埋め込むデータを生成し，(3) の初期化対象アドレスの情報によってロードされたプログラムのアドレスを修正する．これらの型に関する情報をロードモジュール中に定義するために，アセンブラレベルで表1に示す4つのアセンブラ制御指令を定義した．言語Cとこれらの制御指令との対応例を図1に示す．

そして，表1に示したアセンブラ制御指令によってロードモジュール中に型情報に関するレコードが生成される(図2)．

図1の例を見てもわかるように，上記の4つのアセンブラ制御指令を定めるだけで，第3章に示した問題を解決し，ローダに新しい型を追加するだけで機能拡張ができる設計となっている．

5．浮動小数点プロセッサを使用する場合の問題

演算に浮動小数点プロセッサ（URRプロセッサ，MC68881）を使用する場合，マルチタスクの環境下では第3章の問題とは別に，浮動小数点プロセッサのコンテキストスイッチの問題を考慮しなければならない．この点については拡張コンテキスト[6]によって問題を解決した．詳しくは参考文献[6]を参照してもらいたい．

6．おわりに

最近“ソフトウェアIC”と称して，従来ソフトウェアで行っていたものをペリフェラルプロセッサ化することが盛んになりつつある．今回我々は複数の浮動小数点表現法をサポートするという観点から論じてきたが，この型情報と拡張コンテキストを応用することで，このようなペリフェラルプロセッサを効率よくシステムに接続するための一手法となると考えている．

参考文献

[1] Stevenson，D．et al．：A Proposed Standard for Binary Floating−Point Arithmetic，Draft 8．0 of IEEE，Computre，Vol.14，No.3(1981)，pp.51-62.

[2] 浜田穂積：二重指数分割に基づくデータ長独立実数値表現法II，情報処理学会論文誌，Vol.24，No.2(1983)，pp.149-156.

[3] 森岳志，他：各種浮動小数点表現法の評価方式の実現，情報処理学会論文誌，Vol 29，No 8(1988)，pp 807-814.

[4] 中原雅彦，他：URR浮動小数点演算コプロセッサのアーキテクチャの検討と言語C処理系 cat による利用方式，情報処理学会第36回全国大会(1988,3)，pp.219-220.

[5] 中原雅彦，他：複数の浮動小数点方式を処理する言語C処理系CAT／Nのコード生成方式，情報処理学会第38回全国大会(1989,3)，pp.917-918.

[6] 早川栄一，他：OS／omicronにおける新しいデバイスの追加に対するOSの機能拡張の一方式，情報処理学会第40回全国大会(1990,3)

表1．　型情報定義用アセンブラ制御指令

制御指令名	機　能
def_type	型名を定義する
type	実体を定義する
def_op_type	typeで定義された型に対して行う操作名を定義する
op_type	typeで定義された実体に操作を加えて新しい実体を定義する

言語Cソース

```
float  a = 2.0 + 3.0 ;

f( ) {
    float  a , b ;
    a = b + 3.0 ;
}
```

アセンブラソース

```
        def_type     単精度定数
        def_type     単精度演算
        def_op_type  単精度加算
単精度乗算   type    単精度演算
        data
定数1   type      単精度定数,"2.0"
定数2   type      単精度定数,"3.0"
定数3   op_type   単精度定数,単精度加算,定数1,定数2
a       dc.l      定数3
        proc
f
        move.l    b,-(SP)
定数4   type      単精度定数,"3.0"
        move.l    #定数4,-(SP)
        move.l    リターンアドレス,(An)
        jmp       単精度乗算
リターンアドレス
        move.l    (SP)+,a
```

図1．　言語Cとアセンブラの対応例

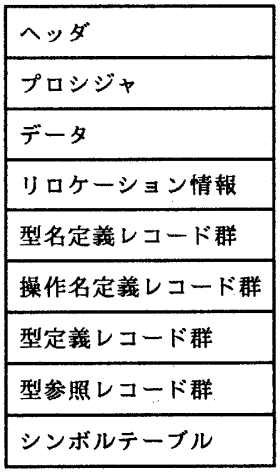

図2．　ロードモジュールの構成