2F−7

32ビットマイクロプロセッサTXシリーズ用Cコンパイラの最適化手法

林田 聖司　田丸 喜一郎

(株)東芝　半導体技術研究所

1．はじめに

本報告では、TRON仕様*1に基づく32ビットマイクロプロセッサTXシリーズ用[1]に開発中のCコンパイラが用いている最適化手法を述べる。このCコンパイラでは、一般的なグローバル・オプティマイズ[2]の他に、TXシリーズ特有の機能を活用するための最適化を行っている。ここでは、この最適化手法を中心に説明する。

このCコンパイラは、東芝が独自に開発を進めているもので、言語仕様は、ISO/ANSI規格[3]に従っている。プログラムは、プリプロセッサ、中間コードジェネレータ、コードジェネレータ、およびこれらのツールを起動するCCドライバの4つから構成されている。

2．TXシリーズ特有の機能

TXシリーズは、CISCタイプのプロセッサで、複雑なアドレッシングモードや高機能命令をサポートしている。この機能をコンパイラが有効に活用することにより、プロセッサの性能を引き出すことができる。ここでは、最適化手法で用いているTXシリーズ特有のアドレッシングと命令について説明する。

・多段間接モード

多段間接モードでは、加算、スケーリング、間接参照のアドレッシングを多段（TXシリーズでは最大4段まで）に組み合わせて、ひとつのオペランドとして扱うことができる。

このアドレッシングモードによって、複数の命令を必要とする処理が一命令で行えるようになる。

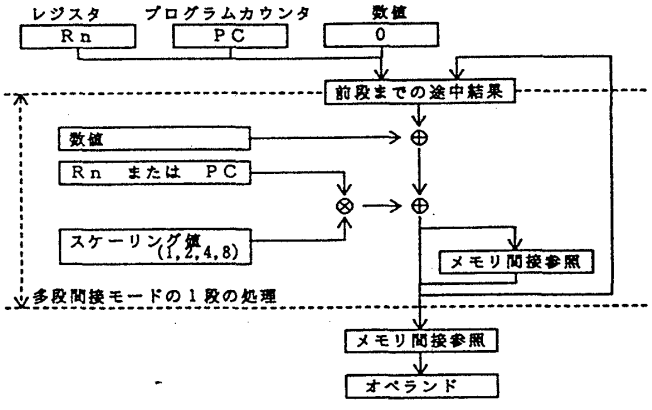

図1．多段間接モード

・ACB/SCB命令

ACB命令は、図2のように加算、比較、条件分岐を一つにした命令である。SCB命令は、加算の代わりに減算を行う。ただし、比較の条件が、ACB命令ではlt(<)、SCB命令ではge(>=)に限られ、カウンタとして用いるvregにはレジスタしか指定することができない。また、実行時間とオブジェクトサイズを考慮すると、ACB/SCB命令を使用した効果があるのは、stepは1、limitは1～64の整数かレジスタを指定した場合に限られ、これ以外の場合では逆に性能が悪化する場合がある。

```
ニモニック
    ACB step,vreg,limit,pcdisp
・機能
    vreg += step;
    if(vreg < limit)
      goto pcdisp;
```

図2．ACB命令の機能

・SSTR命令

SSTR命令は、先頭アドレスと長さで指定されるメモリ領域を一定値で埋める命令であり、書き込むデータサイズとして、1,2,4バイトのいずれかが指定できる。この1命令によって、連続したメモリ領域の初期化を高速に行うことができる。

3．最適化手法

TXシリーズの特有の機能を活用するための最適化手法について説明する。これらの最適化は、コードジェネレータの最適化処理に含まれており、処理の順番は、図3のようになっている。「中間コードの最適化」で、一般的なグローバル・オプティマイズも行われている。

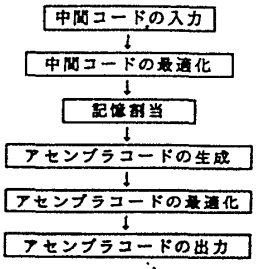

図3　コードジェネレータの処理の順番

・多段間接モード

このアドレッシングモードを活用することにより、図4のような最適化が可能となる。

このアドレッシングモードは、「アセンブラコードの生成」で作成している。作成には、一時変数の

Optimizing Tecchnique for TX-Series C Compiler

Seiji Hayashida,Kiichiro Tamaru
Toshiba Corp. Semiconductor Device Engineering Lab.

使用-定義連鎖の情報を利用している。

このアドレッシングモードを１つ作成するのに、中間コードの複数文が必要となる。「アセンブラコードの生成」の前に「記憶割当」を行っているので、多段間接モードとなる中間コードに割り当てられたレジスタは無駄になる。この無駄を省くために、出現頻度の高い１段の多段間接モードに相当するオペランドを中間コードにも設け、中間コードの段階においても、多段間接モードを作成することにした。

```
Cのプログラム
  int i,j,k; extern char arry[100];
  retrun arry[10+i+j*2+k*4];
コンパイル結果
  mov @(r4,r5*2,_arry+10,r6*4).b,r0
```

図４．多段間接アドレッシングを用いた最適化

・ACB/SCB命令

ACB/SCB命令は、「アセンブラコードの最適化」で覗き穴最適化によって生成されている。しかし、ACB/SCB命令が生成できる条件は限られているので、「中間コードの最適化」の誘導変数の判定式の置き換えで、元の判定式がACB/SCB命令になる条件であり、置き換え後の判定式がそうでない場合は、置き換えを行わないという処理を追加している。

また、ACB命令が生成されるのには、加算、比較、条件分岐の命令が連続している必要があるので、「中間コードの最適化」の覗き穴最適化でこの順番になっていない中間コード列がある場合、命令列の並べ変えを行っている。

```
Cのプログラム
  int i;
  for(i=0;i<10;++i)
     func();
コンパイル結果
  mov #0,r4
L1:
  bsr _func
  acb #1,r4,#10,L1
```

図５．ACB命令を用いた最適化

・SSTR命令

この命令が、たとえばfor文で配列を初期化するような場合に、コンパイラが生成できるようにしている。また、図６のようにネストしたループによって多次元配列を初期化する場合にも対応している。

この最適化処理は、「中間コードの最適化」の誘導変数の最適化の中で、ループ内の中間コード列がある条件を満たす場合は、SSTR命令に相当する中間コードに置き換える処理によって実現されている。

```
Cのプログラム
int i,j; extern int a[10][10];
   for(i=0;i<10;++i)
     for(j=0;j<10;++j)
       a[i][j] = 1;
コンパイル結果
  mov #_a,r1
  mov #1,r3
  mov #100,r2
  sstr.w
```

図６．sstr命令を使った最適化

４．性能評価

性能評価として、スタンフォードのベンチマーク(4)の中からPuzzleというサブルーチンの実行時間とオブジェクトサイズの比較結果を表１に示す。実行時間は、TX1 20MHz/0 waitで測定した。

PCCは、TXシリーズ用のポータブルＣコンパイラで、グローバル・オプティマイズが行われていない。従来手法は、開発中のＣコンパイラで、説明したTXシリーズ特有の機能を活用する最適化を行わない場合の結果である。本手法が、TXシリーズ特有の機能を活用する最適化を行った場合の結果である。

グローバル・オプティマイズによって、実行時間が約1/2になり、オブジェクトサイズも約24%減少することができた。さらに本手法を追加することで、従来手法に対して実行時間を約6%、オブジェクトサイズを約30%減少することができた。

表１．実行時間とオブジェクトサイズの比較

	実行時間（m秒）	オブジェクトサイズ（バイト）
PCC	2,513	1,870
従来手法	1,259	1,514
本手法	1,189	1,170

５．まとめ

一般的なグローバル・オプティマイズに、プロセッサ特有の機能をサポートする最適化を加えることにより、さらに最適化を進めることが可能となった。

今後も、より高性能な最適化ができるコンパイラを開発していく。

参考文献

(1) 32ビットマイクロプロセッサ TX1 ファミリー データブック,東芝 1991
(2) A.V.Aho, R. Sethi, and J.D. Ullman, "Compilers Principles, Techniques, and Tools", Addison-Wesley, 1986
(3) ISO/IEC DIS 9899:1990
(4) B.A. Naused at al., "A 32-Bit 200-MHz GaAs RISC", IEEE MICRO, December, 1987

注: TRONは、"The Real Time Operating System Nucleus"の略称です。