数値プロセッサ使用による数学ライブラリの改良

秦野　和郎・宮　　小元・竹松　英夫

Improvement of Mathematical Library by making use of the Numerical Co-Processor

Kazuo HATANO, Gong Xiao YUAN and Hideo TAKEMATSU

The 16 bits Personal Computers using i8086/i8087 are widely used. The FORTARAN compiler employed on these Personal Couputers only support up to double Precision variables. The registers of i8087 are 80 bits length. We can make better many mathematical libraries if we use i8087 effectively. We tried to make better some mathematical libraries using i8087 on our MULTI 16 personal computer.

1，まえがき

最近 Intel 系の CPU を使用する16ビットパソコンの基本ソフトウェアが非常に豊富に供給されるようになってきた。数値計算用の FORTRAN だけに限っても 6 種類以上を数えることができる（MS-FORTRAN, DR-FORTRAN, Pro FORTRAN, ai-FORTRAN86, R/M FORTRAN, PC-FORTRAN など)。それらのコンパイラの殆んどは ANSI FORTRAN-77規格に基づいており一部はフルセット仕様になっている。i 8086の最大の壁であった一変数64KB の制限も新しいバージョンでは取り払われてきている。

過去数十年，数値計算用の言語としては専ら FORTRAN が使われてきた。この傾向は今後も変わらないと思われる。最近では大規模な科学技術計算にスーパーコンピュータが使われているが，言語としてはやはり FORTRAN が使われている。それのみか機種によってはユーザーに解放されているのは FORTRAN だけである。スーパーコンピュータを使用するには FORTRAN でプログラムを書かざるを得ない状況である。

従来，科学技術計算をするには共同利用の大型計算機を使う必要があった。現在でも数 MB 以上もの主記憶を必要とし大量の計算を要するときは共同利用の大型計算機に頼らざるを得ない。しかし大型計算機を数秒程度しか使わないような小規模な計算は現在ではパソコンで十分に処理できる。又パソコンは共同利用の大型計算機に比較して使い勝手が非常によい。

共同利用の計算機利用者が，使用する計算機をパソコンに切り換えようとするときいくつかの困難に遭遇する。その一つはパソコンではライブラリが非常に乏しい事である。最近では処理系によってはグラフィックパッケージ，数学ライブラリが市販されるようになってきたがまだ十分とは言えない（Pro　FORTRAN における GRAP-PC, PLOT-PC, CORE-PC, TOOL-PC, MATH-PC など)。しかし大型計算機上で開発された数学ライブラリは FORTRAN で書かれていればパソコン上で殆んど無修正で動く。従ってこの面での困難は大型計算機で使われているプログラムをソースの形で入手できれば問題は少ない。

ところで i 8086はそれ単独では非常に魅力の乏しい計算機であるが数値プロセッサ i 8087と一緒に使うとその組合せは非常に興味ある計算機に一変する。その機能を適切に使えばプログラムライブラリの多くは若干の修正により，より高精度にする事が可能である。

ここでは i 8087の特長を有効に生かすべく数学ライブラリの一部を高精度化する事を試みた。このためにはアセンブラの使用を必要とするので互換性という点では後退であるが今後 Intel 系の CPU が高性能化し広く普及する可能性が高いのでこのような試みは意義があると思われる。

2，16ビットパソコンによる数値計算の環境

一部の例外を除き現在市販されている16ビットパソコンは CPU に i 8086又は i 8088を使いオプションとして数値プロセッサ i 8087を使えるようになっている。

モニタとしては CP/M-86, MS-DOS などを使う事ができ，その下で数種類の FORTRAN コンパイラやアセンブラが動く。

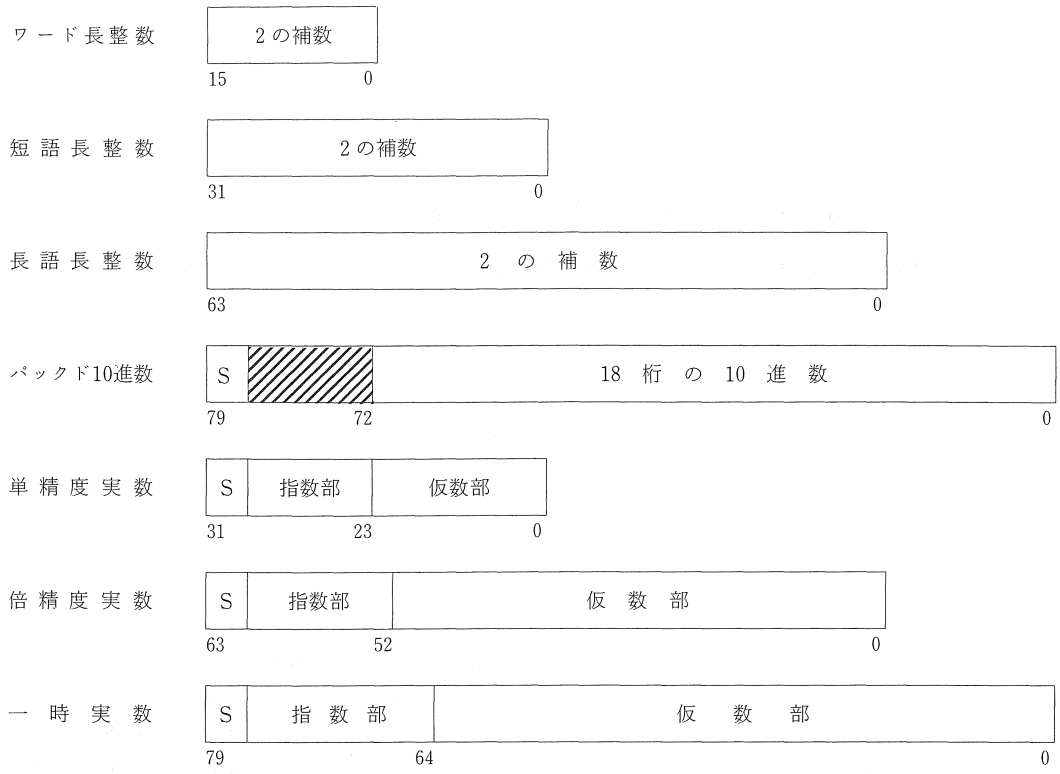

図1　i 8087のデータ形式

ここで使用したのは三菱電機製の MULTI 16 である。既に旧式の部類に属するが CPU としては i 8088を使い数値プロセッサ i 8087を使えるようになっている。CP/M-86 Ver2.2が動きその下でマイクロソフト社の MS-FORTRAN Ver3.13, マクロアセンブラ MACRO-86 Ver3.10が動く。科学技術計算用のプログラム開発の環境としては一応の水準に達していると思う。

i 8086/ i 8088には演算レジスとして16ビット/語の汎用レジスタが4個ある(AX, BX, CX, DX レジスタ)。この4個のレジスタですべての計算を行う。これでは機械命令で直接実行できるのは16ビット以下の整数の加減乗除算までである。

大型計算機は普通, 32ビット以下の整数の加減乗除算, 32ビット及び64ビットの浮動小数点数の加減乗除算を行う機械命令を持っている。科学技術計算ではその程度の機能が最低限必要とされる。i 8086/ i 8088のみを使うパソコンではこれらの計算をすべてソフトウェアで行う。FORTRAN ライブラリにそのようなプログラムが含まれている。

このために浮動小数点数の計算に大変に時間がかかり浮動小数点数の計算の比重が高い科学技術計算では非常に具合が悪い。しかしこの困難は数値プロセッサ i 8087の使用により解決される。

i 8087は整数(16, 32, 64ビットの3種類), 10進数(18桁), 単精度 (32ビット) 及び倍精度 (64ビット) の実数を扱う事ができるが従来の大型計算機と大きく異なる点は実際の計算をすべて80ビット長のレジスタで行う事である(図1)。この形式は一時実数と呼ばれ指数部15ビット, 仮数部64ビットの精度を持っている。i 8087にはこのレジスタが8個ある。たとえ16ビットの整数でもメモリから i 8087にロードされるときには80ビットに変換してロードされる。逆に80ビットのレジスタで行なった計算結果を16ビット長の整数としてメモリに格納するときには80ビットの数を16ビットに丸めて格納される。

i 8087のこのような動作を有効に使うと数学ライブラリの一部を高精度化する事ができる。

3. 数値プロセッサ使用による数学ライブラリの改良

数値計算の手順の中に積和の計算がしばしばあらわれる。すなわち

$$s = \sum_{i=1}^{n} a_i b_i$$

の形の計算である。これは連立一次方程式, 固有値問題, 数値積分, 関数値の計算など非常に多くの数学ライブラ

リに出現する。この計算をFORTRANで普通にコーディングすれば次のようになる。すなわち

```
      IMPLICIT   REAL * 4 (A-H, O-Z)
      DIMENSION   A(100), B(100)
        ....
      S = 0.0
      DO   10   I = 1, N
      S = S+A(I)* B(I)
   10 CONTINUE
        ....
```

図2

である。このときS=S+A(I)* B(I)なる式において積A(I)* B(I)の結果は単精度に丸められてSに加えられる。従ってDOループを出た後Sには一回の丸めで生ずる誤差のN倍が累積する事になる。

これを次のようにコーディングしたとする。

```
      IMPLICIT   REAL * 4 (A-H, O-Z)
      REAL * 8   S1
      DIMENSION A ( 100 ), B ( 100)
          ....
      S1 = 0.0D0
      DO   10   I = 1, N
      S1 = S1+DBLE(A(I))* DBLE(B(I))
   10 CONTINUE
      S = S1
        ....
```

図3

ここでDBLEは単精度の数を倍精度の数に変換する組込関数である。

このようにすると積和の計算をすべて倍精度で行なうので丸め誤差の累積は起らない。最後のS=S1で一度単精度に比めるだけである。このために図3のプログラムの方が一般に誤差が小さい。特にNが大きいときその効果は顕著である。又，大低の計算機では図2のようにコーディングしても図3のようにコーディングしても計算時間は殆んど変らない。従って単精度のライブラリの良心的なプログラムは図3のようにコーディングしてあるのが普通である。

一方倍精度のプログラムでは通常このような手段をとれない。積和の計算は

```
      IMPLICIT   REAL * 8 (A-H, O-Z)
      DIMENSION   A(100), B(100)
          ....
      S = 0.0D0
      DO   10   I = 1, 10
      S = S+A(I)* B(I)
   10 CONTINUE
          ....
```

図4

となるがこれを図3のような形で改良する事はできない。但し一部の大型計算機では4倍精度計算が可能である。そのような計算機では図4のようなプログラムを次のように書き変えて誤差を少なくする事ができる。

```
      IMPLICIT   REAL * 8 (A-H, O-Z)
      REAL * 16   S1, T1
      DIMENSION   A(100), B(100)
          ....
      S1 = 0.0Q0
      DO   10   I = 1, N
      T1 = A(I)
      S1 = S1+T1 * B(I)
   10 CONTINUE
      S = S1
        ....
```

図5

このような事が可能なのは一部の計算機に限られ一般性は少ない。

i8086又はi8088を使う16ビットパソコンにはいくつかのFORTRAN処理系が市販されているがどのFORTRANも倍精度数までしか扱かわない。筆者らが使っているMS-FORTRANでも80ビットの数を直接扱かう事はできない。

積和の部分だけをサブルーチン化し，それをアセンブラで書けば中間結果を80ビットに保持する事ができて積和計算で生ずる丸め誤差の累積を軽減する事ができる。この方針で種々なプログラムを改良する事ができるがここではLU分解法により連立一次方程式を解くプログラムにこの方針を適用した例について述べる。

図6はLU分解法で連立一次方程式を解くプログラムである。このプログラム中に積和計算をする部分が3ケ所ある。29行目～31行目，53行目～55行目及び64行目～66行目である。この部分をサブルーチン化してMACRO-86で書き中間結果を80ビットのままで保持して積和を計

算するプログラムが図8である。この中には二つのサブルーチンNPROA1及びNPROA2が含まれている。図7のプログラムはそれらを呼び出してLU分解法により連立一次方程式を解く。

同一の問題を図6のDLUF1Aを使って解いた解と，図7のNLUF1Aを使って解いた解を比較すると80元程度の連立一次方程式の場合で1～2桁(10進)，精度を改善しうる事がわかった。

4．むすび

i 8086やi 8088を使うパソコンの機械語は非常に複雑でアセンブラの使用は容易でない。又サブルーチンの呼出し手順はFORTRAN処理系ごとに異なる。アセンブラの使い方もアセンブラ処理系ごとに異なるのである。たとえばMS-FORTRAN(マイクロソフト社製)とDR-FORTRAN(デイジタルリサーチ社製)とではサブルーチンの呼出し手順が異なる。又それぞれと対になっているリロケータブルアセンブラはMACRO-86(マイクロソフト社製)及びRASM86(デイジタルリサーチ社製)であるがその仕様はかなり異なっている。どちらのアセンブラもまだかなりのバグがあり仕様書通りには動かない。RASM86ではi 8087を使う命令の一部はまだ使うことさえできない。

そのような状況であるからi 8087を使うためのアセンブラプログラミングには相当な困難を伴うが一旦動くプログラムができればそれは有用なライブラリになる。

i 8086はi 80286へと発展しi 80286とi 80287を使うパソコンが最近各社から発売されるようになった。そのような高性能パソコンの能力を有効に引き出すためには適切なオペレーティングシステムが必要であるがそれは1986年1月現在まだ入手困難である。しかしまもなくXENIXなるオペレーティングシステムが普及すると思われる。i 8087のプログラミングとi 80287のプログラミングとはほとんど同じと言われている。

更にi 80286/i 80287は32ビットプロセッサi 80386/i 80387へと発展し数年後にはそれを使用した非常に高性能の32ビットパソコンが出現すると思われる。i 8087で数値プロセッサの扱いに慣れておけばその技術は相当長い期間活用しうる筈である。

本稿では連立一次方程式を解く一つの例を示したにすぎないが今後種々なプログラムの改良を試みて行く予定である。

参考文献

1）秦野和郎，宮小元，竹松英夫：数値プロセッサ使用による数学ライブラリの改良，昭和60年度電気関係学会東海支部連合大会，講演番号484

2）内藤祥雄訳，Stephen P. Morse原著：8086入門，システムソフト，福岡，1984

3）御牧義訳，John F. Palmer，Stephen P. Morse原著：8087入門，啓学出版，東京，1985

4）John R. Rice: Matrix Computations & Mathe-mathecal Software, McGraw-Hill, New York, 1983

（受理　昭和61年1月25日）

```
A:PIP CON:=B:DLUF1AA1.FOR[N]
    1: C*
    2: C********** DLUF1A ** DLUF1AA1 **********************************
    3: C*                                                              *
    4: C*       SOLUTION OF SIMULTANEOUS LINEAR EQUATION BY            *
    5: C*              LU-DECOMPOSITION  (FULL MATRIX)                 *
    6: C*                                                              *
    7: C*     KIND=0...SIMULTANEOUS LINEAR EQUATION                    *
    8: C*     KIND=1...LU-DECOMPOSITION                                *
    9: C*     KIND=2...FOWARD AND BACKWARD SUBSTITUTION                *
   10: C*                                                              *
   11: C**********          **            **********************************
   12:       SUBROUTINE DLUF1A(A,B,N,KIND,ICON,ND)
   13:       IMPLICIT INTEGER*4 (I-N)
   14:       IMPLICIT REAL   *8 (A-H,O-Z)
   15:       DIMENSION A(ND,20),B(ND)
   16:       DATA EPS / 1.0D-30 /
   17:       IF(KIND.EQ.2) GO TO 1230
   18: C     LU-DECOMPOSITION
   19:       ICON=0
   20:       DO 1220 IP=1,N
```

```
21:           DO 1140 IR=1,N
22:             SUM1=A(IR,IP)
23:             IF(IR.LT.IP) THEN
24:               ICU=IR-1
25:             ELSE
26:               ICU=IP-1
27:             ENDIF
28:             IF(ICU.GE.1) THEN
29:               DO 1120 IC=1,ICU
30:                 SUM1=SUM1-A(IR,IC)*A(IC,IP)
31:  1120         CONTINUE
32:             ENDIF
33:             IF(IR.EQ.IP) THEN
34:               IF(DABS(SUM1).LT.EPS) THEN
35:                 ICON=-IP
36:                 SUM1=EPS
37:               ENDIF
38:               PIVI=1.0D0/SUM1
39:             ENDIF
40:             IF(IR.GT.IP) THEN
41:               SUM1=SUM1*PIVI
42:             ENDIF
43:             A(IR,IP)=SUM1
44:  1140     CONTINUE
45:  1220   CONTINUE
46: C        FOWARD AND BACKWARD SUBSTITUTION
47:  1230   IF(KIND.EQ.1) GO TO 1310
48:         NP1=N+1
49: C        FOWARD SUBSTITUTION
50:         DO 1270 IP=2,N
51:           SUM1=B(IP)
52:           ICU=IP-1
53:           DO 1260 IC=1,ICU
54:             SUM1=SUM1-A(IP,IC)*B(IC)
55:  1260     CONTINUE
56:           B(IP)=SUM1
57:  1270   CONTINUE
58: C        BACKWARD SUBSTITUTION
59:         DO 1300 IPD=1,N
60:           IP=NP1-IPD
61:           SUM1=B(IP)
62:           IF(IP.NE.N) THEN
63:             IPP1=IP+1
64:             DO 1280 IC=IPP1,N
65:               SUM1=SUM1-A(IP,IC)*B(IC)
66:  1280       CONTINUE
67:           ENDIF
68:           B(IP)=SUM1/A(IP,IP)
69:  1300   CONTINUE
70:  1310   RETURN
71:         END
```

図6

```
A:PIP CON:=B:NLUF1AA1.FOR[N]
      1: C*
      2: C********** NLUF1A ** NLUF1AA1 ********************************
      3: C*                                                            *
      4: C*       SOLUTION OF SIMULTANEOUS LINEAR EQUATION BY          *
      5: C*             LU-DECOMPOSITION  (FULL MATRIX)                *
      6: C*            (PARTIALLY TEMPOLLARY PRECISION)                *
      7: C*                                                            *
      8: C*     KIND=0...SIMULTANEOUS LINEAR EQUATION                  *
      9: C*     KIND=1...LU-DECOMPOSITION                              *
     10: C*     KIND=2...FOWARD AND BACKWARD SUBSTITUTION              *
     11: C*                                                            *
     12: C********** **  ********************************************
     13:         SUBROUTINE NLUF1A(A,B,N,KIND,ICON,ND)
     14:         IMPLICIT INTEGER*4 (I-N)
     15:         IMPLICIT REAL   *8 (A-H,O-Z)
     16:         DIMENSION A(ND,20),B(ND)
     17:         DATA EPS / 1.0D-30 /
     18:         IF(KIND.EQ.2) GO TO 1230
     19: C       LU-DECOMPOSITION
     20:         ICON=0
     21:         DO 1220 IP=1,N
     22:           DO 1140 IR=1,N
     23:             SUM1=A(IR,IP)
     24:             IF(IR.LT.IP) THEN
     25:               ICU=IR-1
     26:             ELSE
     27:               ICU=IP-1
     28:             ENDIF
     29:             IF(ICU.GE.1) THEN
     30:               CALL NPROA1(A,SUM1,ND,IP,IR,ICU)
     31:             ENDIF
     32:             IF(IR.EQ.IP) THEN
     33:               IF(DABS(SUM1).LT.EPS) THEN
     34:                 ICON=-IP
     35:                 SUM1=EPS
     36:               ENDIF
     37:               PIVI=1.0D0/SUM1
     38:             ENDIF
     39:             IF(IR.GT.IP) THEN
     40:               SUM1=SUM1*PIVI
     41:             ENDIF
     42:             A(IR,IP)=SUM1
     43:  1140     CONTINUE
     44:  1220   CONTINUE
     45: C       FOWARD AND BACKWARD SUBSTITUTION
     46:  1230   IF(KIND.EQ.1) GO TO 1310
     47:         NP1=N+1
     48: C       FOWARD SUBSTITUTION
     49:         DO 1270 IP=2,N
     50:           SUM1=B(IP)
     51:           ICU=IP-1
     52:           ICS=1
     53:           CALL NPROA2(A,B,SUM1,ND,IP,ICS,ICU)
     54:           B(IP)=SUM1
     55:  1270   CONTINUE
     56: C       BACKWARD SUBSTITUTION
     57:         DO 1300 IPD=1,N
     58:           IP=NP1-IPD
     59:           SUM1=B(IP)
```

```
60:           IF(IP.NE.N) THEN
61:             IPP1=IP+1
62:             CALL NPROA2(A,B,SUM1,ND,IP,IPP1,N)
63:           ENDIF
64:           B(IP)=SUM1/A(IP,IP)
65:  1300 CONTINUE
66:  1310 RETURN
67:       END
```

図7

```
A:PIP CON:=B:NPROA1M1.ASM[N]
 1: ;*
 2: ;********** NPROA1 ** NPROA1M1 **********************************
 3: ;*                                                               *
 4: ;*        INNER PRODUCT                                          *
 5: ;*                                                               *
 6: ;**********          **              **********************************
 7: DGROUP     GROUP DATA,STACK
 8:            ASSUME CS:PROA1,DS:WORK1,SS:DGROUP
 9: PROA1      SEGMENT 'CODE'
10: ;       SUM1=SUM1-A(IR,IC)*A(IC,IP)
11: ;                          IC=1,2,...,ICU
12: ;       SUBROUTINE NPROA1(A,SUM1,ND,IP,IR,ICU)
13:            PUBLIC NPROA1
14: NPROA1     PROC FAR
15:            PUSH BP
16:            MOV BP,SP
17:            LES BX,DWORD PTR 18[BP]        ;ND
18:            MOV AX,WORD PTR ES:[BX]
19:            MOV BX,08D
20:            MUL BX
21:            MOV DI,OFFSET DS:ND8
22:            MOV WORD PTR DS:[DI],AX
23:            LES BX,DWORD PTR 10[BP]        ;IR
24:            MOV AX,WORD PTR ES:[BX]
25:            DEC AX
26:            MOV BX,08D
27:            MUL BX
28:            MOV DI,OFFSET DS:IRC
29:            MOV WORD PTR DS:[DI],AX
30:            LES BX,DWORD PTR 14[BP]        ;IP
31:            MOV AX,WORD PTR ES:[BX]
32:            DEC AX
33:            MOV DI,OFFSET DS:ND8
34:            MUL WORD PTR DS:[DI]
35:            MOV DI,OFFSET DS:ICP
36:            MOV WORD PTR DS:[DI],AX
37:            MOV CX,00D
38:            LES BX,DWORD PTR 6[BP]         ;ICU
39:            MOV DX,WORD PTR ES:[BX]
40:            LES BX,DWORD PTR 22[BP]        ;SUM1
41:            FLD QWORD PTR ES:[BX]
42: LOOP1:     MOV DI,OFFSET DS:IRC
43:            MOV AX,WORD PTR DS:[DI]
44:            LES BX,DWORD PTR 26[BP]        ;A
45:            ADD BX,AX
46:            FLD QWORD PTR ES:[BX]          ;A(IR,IC)
47:            MOV DI,OFFSET DS:ICP
```

```
48:             MOV AX,WORD PTR DS:[DI]
49:             LES BX,DWORD PTR 26[BP]         ;A
50:             ADD BX,AX
51:             FLD QWORD PTR ES:[BX]           ;A(IC,IP)
52:             FMULP ST(1),ST
53:             FSUBP ST(1),ST
54:             INC CX
55:             CMP DX,CX
56:             JE EXIT1
57:             MOV DI,OFFSET DS:IRC
58:             MOV AX,WORD PTR DS:[DI]
59:             MOV SI,OFFSET DS:ND8
60:             ADD AX,WORD PTR DS:[SI]
61:             MOV WORD PTR DS:[DI],AX
62:             MOV DI,OFFSET DS:ICP
63:             MOV AX,WORD PTR DS:[DI]
64:             MOV BX,08D
65:             ADD AX,BX
66:             MOV WORD PTR DS:[DI],AX
67:             JMP LOOP1
68: EXIT1:      LES BX,DWORD PTR 22[BP]         ;SUM1
69:             FSTP QWORD PTR ES:[BX]
70:             MOV SP,BP
71:             POP BP
72:             RET 24
73: NPROA1      ENDP
74: ;       SUM1=SUM1-A(IP,IC)*B(IC)
75: ;                           IC=ICS,ICS+1,...,ICE
76: ;       SUBROUTINE NPROA2(A,B,SUM1,ND,IP,ICS,ICE)
77:             PUBLIC NPROA2
78: NPROA2      PROC FAR
79:             PUSH BP
80:             MOV BP,SP
81:             LES BX,DWORD PTR 18[BP]         ;ND
82:             MOV AX,WORD PTR ES:[BX]
83:             MOV BX,08D
84:             MUL BX
85:             MOV DI,OFFSET DS:ND8
86:             MOV WORD PTR DS:[DI],AX
87:             LES BX,DWORD PTR 14[BP]         ;IP
88:             MOV AX,WORD PTR ES:[BX]
89:             DEC AX
90:             MOV BX,08D
91:             MUL BX
92:             MOV CX,AX
93:             LES BX,DWORD PTR 10[BP]         ;ICS
94:             MOV AX,WORD PTR ES:[BX]
95:             DEC AX
96:             MOV DI,OFFSET DS:ND8
97:             MUL WORD PTR DS:[DI]
98:             ADD AX,CX
99:             MOV DI,OFFSET DS:IPC
100:            MOV WORD PTR DS:[DI],AX
101:            LES BX,DWORD PTR 10[BP]         ;ICS
102:            MOV AX,WORD PTR ES:[BX]
103:            DEC AX
104:            MOV BX,08D
105:            MUL BX
106:            MOV DI,OFFSET DS:ICC
107:            MOV WORD PTR DS:[DI],AX
```

```
108:            LES BX,DWORD PTR 10[BP]        ;ICS
109:            MOV CX,WORD PTR ES:[BX]
110:            DEC CX
111:            LES BX,DWORD PTR 6[BP]         ;ICE
112:            MOV DX,WORD PTR ES:[BX]
113:            LES BX,DWORD PTR 22[BP]        ;SUM1
114:            FLD QWORD PTR ES:[BX]
115: LOOP2:     MOV DI,OFFSET DS:IPC
116:            MOV AX,WORD PTR DS:[DI]
117:            LES BX,DWORD PTR 30[BP]        ;A
118:            ADD BX,AX
119:            FLD QWORD PTR ES:[BX]          ;A(IP,IC)
120:            MOV DI,OFFSET DS:ICC
121:            MOV AX,WORD PTR DS:[DI]
122:            LES BX,DWORD PTR 26[BP]        ;B
123:            ADD BX,AX
124:            FLD QWORD PTR ES:[BX]          ;B(IC)
125:            FMULP ST(1),ST
126:            FSUBP ST(1),ST
127:            INC CX
128:            CMP DX,CX
129:            JE EXIT2
130:            MOV DI,OFFSET DS:IPC
131:            MOV AX,WORD PTR DS:[DI]
132:            MOV SI,OFFSET DS:ND8
133:            ADD AX,WORD PTR DS:[SI]
134:            MOV WORD PTR DS:[DI],AX
135:            MOV DI,OFFSET DS:ICC
136:            MOV AX,WORD PTR DS:[DI]
137:            MOV BX,08D
138:            ADD AX,BX
139:            MOV WORD PTR DS:[DI],AX
140:            JMP LOOP2
141: EXIT2:     LES BX,DWORD PTR 22[BP]        ;SUM1
142:            FSTP QWORD PTR ES:[BX]
143:            MOV SP,BP
144:            POP BP
145:            RET 28
146: NPROA2     ENDP
147: PROA1      ENDS
148: ;
149: WORK1      SEGMENT 'DATA'
150: IRC        DW ?
151: ICP        DW ?
152: IPC        DW ?
153: ICC        DW ?
154: ND8        DW ?
155: WORK1      ENDS
156:            END
```

図8