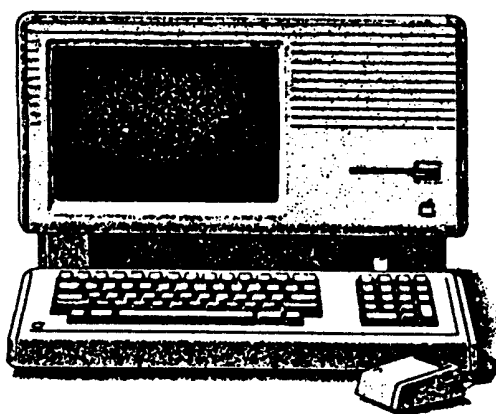


Doc# 77

## Apple Lisa Information



FILE NAME

Lisa Floating Point Savage Benchmark<sup>2</sup>

DISK #

COMMENTS

DTC

David T. Craig  
736 Edgewater, Wichita, Kansas 67230  
(316) 733-0914

The **Lisa**  
Professional

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

# **APPLE LISA FLOATING POINT SAVAGE BENCHMARK 2**

October 2, 1987

Written by

David T. Craig

736 Edgewater  
Wichita, KS 67230

## **Table of Contents**

Introduction .....	2
Benchmark Results .....	3
Pascal Source Code Listing .....	6
Assembly Source Code Listing .....	9
WorkShop Compilation Listings.....	18

*Page 1 of 19*

*by David Craig*

"77-01.PICT" 94 KB 2000-12-24 dpi: 300h x 300v pix: 1928h x 3109v

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

## Introduction

This document describes the testing of the Savage benchmark upon the Apple Lisa computer. The Savage benchmark tests the numeric performance of computers. Concentration is paid to the calculation of the following mathematical functions:

Tangent  
ArcTangent  
Exponent ( $e^x$  [ $e = 2.71...$ ])  
Logarithm (base  $e$ )  
Square Root

The benchmark was run on a Lisa 2 system which consisted of 1 MByte of RAM, an internal 10 MByte hard disk drive, and an internal micro disk drive. The test program was written in Pascal and tested in the Lisa WorkShop 3.9 development environment.

The Lisa computer uses a floating point engine called SANE (Standard Apple Numeric Environment) which was designed with numeric accuracy as the most important criteria. For a comprehensive discussion of SANE (i.e., what it is and why it is better) refer to the Apple Computer publication titled "Standard Apple Numeric Environment" (the Lisa WorkShop 3.0 manual set [product # 620-6149-B] also provides a very detailed discussion of SANE, but has little information concerning why SANE is better than other numeric engines).

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

## Benchmark Results

The following listing was directly generated by the Savage Residue benchmark program. The listing shows the number of iterations performed, the computation time (in seconds), and the computation residual value (the magnitude of the difference between the true value and the calculated value).

### Apple Lisa Floating Point SAVAGE RESIDUE Benchmark

Savage Benchmark is executing ...

Iterations	Duration (sec)	Residue
200	19.020	0.00000000000002
400	38.239	0.00000000000001
600	57.636	0.00000000000017
800	76.863	0.00000000000035
1000	96.474	0.00000000000031
1200	115.780	0.00000000000030
1400	134.929	0.00000000000110
1600	154.393	0.00000000000164
1800	174.032	0.00000000000136
2000	193.782	0.00000000000202
2200	213.294	0.00000000000093
2400	232.565	0.00000000000114
2600	251.774	0.00000000000302
2800	270.953	0.00000000000808
3000	290.407	0.00000000000721
3200	310.041	0.00000000000477
3400	329.747	0.00000000000744
3600	349.493	0.00000000001306
3800	369.325	0.00000000000515
4000	389.197	0.00000000000394
4200	408.981	0.00000000001003
4400	428.393	0.00000000000908
4600	447.763	0.00000000000710
4800	467.092	0.00000000000337
5000	486.364	0.00000000002172
5200	505.694	0.00000000000614
5400	524.936	0.00000000000057
5600	544.273	0.00000000001869
5800	563.537	0.00000000000868
6000	583.291	0.00000000001282
6200	603.065	0.00000000003960
6400	622.800	0.00000000004946
6600	642.631	0.00000000006289

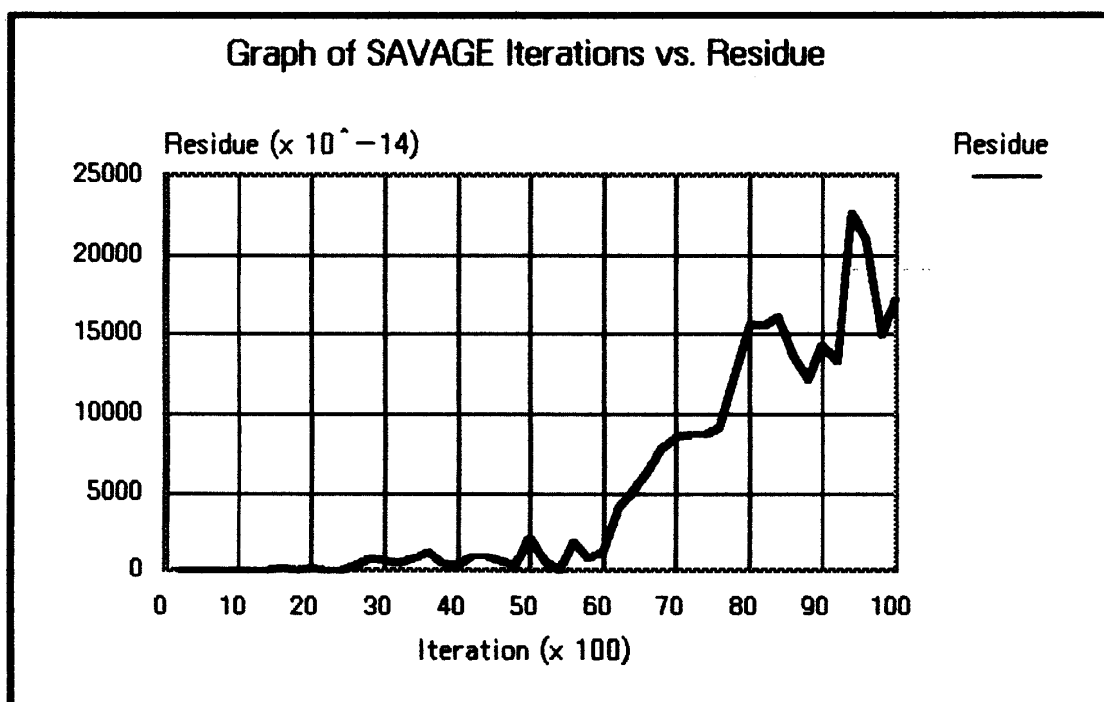
*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

6800	662.393	0.00000000007725
7000	682.252	0.00000000008491
7200	702.079	0.00000000008717
7400	721.968	0.00000000008664
7600	741.870	0.00000000009070
7800	761.843	0.00000000012604
8000	781.807	0.00000000015592
8200	801.987	0.00000000015477
8400	821.613	0.00000000016094
8600	841.129	0.00000000013611
8800	860.606	0.00000000012138
9000	880.060	0.00000000014387
9200	899.493	0.00000000013334
9400	918.896	0.00000000022605
9600	938.312	0.00000000020860
9800	957.717	0.00000000014886
10000	977.110	0.00000000017072

That's all, Folks...

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

In order to better understand the pattern behind the residues, the iteration count vs. the residue values was plotted. From the below plot one can see that the residues generally increase as the iteration count increases. But the increase is erratic. For example, iteration count 9400 had a higher residue than 9200, and iteration count 9600 was smaller than the residue of iteration count 9400.



### Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)

## Pascal Source Code Listing

```
[ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]  
[ ]  
[ ]      FLOATING POINT SAVAGE RESIDUE BENCHMARK  
[ ]      FOR THE APPLE LISA AND SANE FP ENGINE  
[ ]  
[ ]      October 2, 1987  
[ ]  
[ ]      Compiled using Lisa Pascal 3.76 in the Lisa WorkShop  
[ ]  
[ ]      by David T. Craig  
[ ]      [736 Edgewater, Wichita, KS 67230]  
[ ]  
[ ]      Part of source code listing from Byte, Oct. 1987, p. 277  
[ ]
```

**PROGRAM Savage Residue FP Benchmark:**

```
{SR- } { Turn OFF range checking }
{$ASM+} { Turn ON 68000 assembly code listing }
```

```

USES { $L- }
    { $U LISA/HARDWARE.OBJ } HARDWARE, { Lisa Hardware interface }
    { $U LISA/SANELIB.OBJ } SANE;      { Lisa FP engine interface }
    { $L+ }

```

```
TYPE
    t_FP_Number = Extended; { 80-bit precision FP value (Apple SANE type) }
                          { accurate to 19 decimal places }
```

```
VAR
  g_iteration : INTEGER;      { Iteration counter      }
  g_time       : MilliSeconds; { Execution time (in ms) }
  q_residue    : t_FP_Number; { Residue value         }
```

{ ..... }

[illegible]

```
BEGIN { ----- Show_Execution_Results ----- }
  IF show_title_line THEN
    BEGIN
      WRITELN('Iterations      Duration (sec)      Residue');
      WRITELN('-----');
    END
  ELSE
```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

        WRITELN(iteration_count:10,' ',
                (exec_duration / 1000.0):14:3,' ',
                ABS(residue_value):18:14);
    END; { ----- Show_Execution_Results ----- }

{ ===== }

PROCEDURE Perform_SAVAGE_Algorithm(    iter_loop : INTEGER;
                                     VAR exec_time : MilliSeconds;
                                     VAR residue   : t_FP_Number);

VAR
    i      : INTEGER;      { Loop counter          }
    a      : t_FP_Number;  { Computation variable }
    start  : MilliSeconds; { Starting time (in ms) }
    finish : MilliSeconds; { Ending   time (in ms) }

BEGIN { ----- Perform_SAVAGE_Algorithm ----- }
    start := Timer; { Start the timing }

    a := 1.0; { Initialize the benchmark variable }

    FOR i := 1 TO iter_loop DO { Perform the actual benchmark }
        a := Tan(ArcTan(Exp(Ln(Sqrt(a*a)))) + 1.0;
        { <----- = a -----> }

    finish := Timer; { End the timing }

    exec_time := finish - start; { Determine real execution time }

    residue := ((iter_loop * 1.0) + 1.0) - a; { Return the residue value }
    END; { ----- Perform_SAVAGE_Algorithm ----- }

{ ===== }

BEGIN { ----- Savage_Residue_FP_Benchmark ----- }
    WRITELN('Apple Lisa Floating Point SAVAGE RESIDUE Benchmark');
    WRITELN;
    WRITELN('Savage Benchmark is executing ...');
    WRITELN;

    { Make certain the SANE FP engine code is loaded }

    g_residue := SIN(10.0) * 1.2345; { Don't Care values }

    { Perform the actual SAVAGE benchmarks }

    Show_Execution_Results(TRUE, 0, 0, 0.0);

    FOR g_iteration := 1 TO 50 DO
        BEGIN
            Perform_SAVAGE_Algorithm    (g_iteration*200, g_time, g_residue);
            Show_Execution_Results      (FALSE, g_iteration*200, g_time, g_residue);
        END
    END

```



### Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)

[illegible]

### Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)

## Assembly Source Code Listing

**Lisa Pascal Compiler V3.76 (05-Apr-85)**

Lisa Pascal NC60000 Code Generator V3.65 (20-Mar-85)

08:22:23 02-Oct-87

00:23:03 02-Oct-07

```

1      1 --
2      2 --
3      3 --
4      4 --
5      5 --
6      6 --
7      7 --
8      8 --
9      9 --
10     10 --
11     11 --
12     12 --
13     13 --
14     14 --
15     15 --
16     16 --
17     17 --
18     18 --
19     19 --
20     20 --
21     21 --
22     22 --
23     23 --
24     24 --
25     25 --
357    26 --
358    27 --
359    28 --
360    29 --
361    30 --
362    31 --
363    32 --
364    33 --
365    34 --
366    35 --
367    36 --
368    37 --
369    38 --
370    39 -- A
371    40 --
372    41 --
373    42 --
374    43 --
375    44 O- A

( [ ] )

        FLOATING POINT SAVAGE RESIDUE BENCHMARK
        FOR THE APPLE LISA AND SANE FP ENGINE

                October 2, 1987

        Compiled using Lisa Pascal 3.76 in the Lisa WorkShop

                by David T. Craig
                [736 Edgewater, Wichita, KS 67230]

        Part of source code listing from Byte, Oct. 1987, p. 277

[ ] )

PROGRAM Savage_Residue_FP_Benchmark;

(SR-) { Turn OFF range checking }
($ASH+) { Turn ON 68000 assembly code listing }

USES ($L-)
      ($SU LISA/HARDWARE.OBJ) HARDWARE, { Lisa Hardware interface }
      ($SU LISA/SANELIB.OBJ) SANE; { Lisa FP engine interface }
($L+)

TYPE
  t_FP_Number = Extended; { 80-bit precision FP value (Apple SANE type) }
                  { accurate to 19 decimal places }

VAR
  g_iteration : INTEGER; { Iteration counter }
  g_time       : Milliseconds; { Execution time (in ms) }
  g_residue    : t_FP_Number; { Residue value }

{ ===== }

PROCEDURE Show_Execution_Results(show_title_line : BOOLEAN;
                                iteration_count : INTEGER;
                                exec_duration   : Milliseconds;
                                residue_value    : t_FP_Number);

BEGIN { ----- Show_Execution_Results ----- }

000000 4A6F EFF6          SHOW_EXE TST.W $EFF6(A7)
000004 4E56 FFEC          LINK.AG,$FFEC
000008 206E 000B          MOVE.L $000B(A6),A0
00000C 43EE FFF6          LEA.$FFF6(A6),A1
000010 4A10              TST.B.(A0)
000012 22D8              MOVE.L.(A0)+,(A1)+
000014 22D8              MOVE.L.(A0)+,(A1)+
000016 3290              MOVE.W.(A0),(A1)

376    45 --            IF show_title_line THEN
000018 102E 0012          MOVE.B $0012(A6),D0
00001C 6730              BEQ.S L0001 : 0000004E
```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

377      46 1-      BEGIN
378      47 --      Writeln('Iterations   Duration (sec)   Residue');

      00001E 2F2D 000C      MOVE.L $000C(A5), -(A7)
      000022 487A 0116      PEA      Cst0004      ; 0000013A
      000026 4267      CLR.W      -(A7)
      000028 4EBA 0000      JSR      %W_STR
      00002C 2F2D 000C      MOVE.L $000C(A5), -(A7)
      000030 4EBA 0000      JSR      %W_LN

379      48 --      Writeln('-----');

      000034 2F2D 000C      MOVE.L $000C(A5), -(A7)
      000038 487A 000C      PEA      Cst0003      ; 00000106
      00003C 4267      CLR.W      -(A7)
      00003E 4EBA 0000      JSR      %W_STR
      000042 2F2D 000C      MOVE.L $000C(A5), -(A7)
      000046 4EBA 0000      JSR      %W_LN
      00004A 6000 0096      BRA      L0002      ; 000000E2

380      49 -1      END
381      50 --      ELSE
382      51 --      Writeln(iteration_count: 10, ' ',
383      52 --      (exec_duration / 1000.0): 14:3, ' ',
384      53 --      ABS(residue_value): 18:14);

      00004E 2F2D 000C      L0001 MOVE.L $000C(A5), -(A7)
      000052 302E 0010      MOVE.W $0010(A6), D0
      000056 48C0      EXT.L      D0
      000058 2F00      MOVE.L      D0, -(A7)
      00005A 3F3C 000A      MOVE.W      #$000A, -(A7)
      00005E 4EBA 0000      JSR      %W_I
      000062 2F2D 000C      MOVE.L $000C(A5), -(A7)
      000066 487A 008E      PEA      Cst0001      ; 000000F6
      00006A 4267      CLR.W      -(A7)
      00006C 4EBA 0000      JSR      %W_STR
      000070 2F2D 000C      MOVE.L $000C(A5), -(A7)
      000074 486E 000C      PEA      $000C(A6)
      000078 486E FFEC      PEA      $FFEC(A6)
      00007C 3F3C 280E      MOVE.W      #$280E, -(A7)
      000080 4EBA 0000      JSR      FP68K
      000084 487A 0076      PEA      Cst0002      ; 000000FC
      000088 486E FFEC      PEA      $FFEC(A6)
      00008C 3F3C 0006      MOVE.W      #$0006, -(A7)
      000090 4EBA 0000      JSR      FP68K
      000094 486E FFEC      PEA      $FFEC(A6)
      000098 3F3C 000E      MOVE.W      #$000E, -(A7)
      00009C 3F3C 0003      MOVE.W      #$0003, -(A7)
      0000A0 4EBA 0000      JSR      %W_F_EXT
      0000A4 2F2D 000C      MOVE.L $000C(A5), -(A7)
      0000A8 487A 004C      PEA      Cst0001      ; 000000F6
      0000AC 4267      CLR.W      -(A7)
      0000AE 4EBA 0000      JSR      %W_STR
      0000B2 2F2D 000C      MOVE.L $000C(A5), -(A7)
      0000B6 41EE FFEC      LEA      $FFEC(A6), A0
      0000BA 43EE FFF6      LEA      $FFF6(A6), A1
      0000BE 20D9      MOVE.L      (A1)+, (A0)+

```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

0000C0 20D9          MOVE.L (A1)+, (A0)+
0000C2 3091          MOVE.W (A1), (A0)
0000C4 022E 007F FFEC    ANDI.B #$007F, $FFEC(A6)
0000CA 486E FFEC        PEA    $FFEC(A6)
0000CE 3F3C 0012        MOVE.W #$0012, -(A7)
0000D2 3F3C 000E        MOVE.W #$000E, -(A7)
0000D6 4EBA 0000        JSR    $M_F_EXT
0000DA 2F2D 000C        MOVE.L $000C(A5), -(A7)
0000DE 4EBA 0000        JSR    $M_LN

0000E2 4E5E          L0002  UNLK    A6
0000E4 205F          MOVE.L (A7)+, A0
0000E6 DEFC 000C        ADDA.W #$000C, A7
0000EA 4ED0          JMP    (A0)

0000EC D34B 4F57 5F45    .WORD  $D34B, $4F57, $5F45 ; ".HOW_E"
0000F2 5B45          .WORD  $5B45 ; "XE"

0000F4 006C          CstSize .WORD  Last-CstSize-2
0000F6          Cst0001
0000F6 04          .BYTE  4
0000F7 2020 2020        .ASCII  '  '
0000FB 00          .BYTE  $00
0000FC          Cst0002
0000FC 400B FA00 0000    .WORD  $400B, $FA00, $0000
000102 0000 0000        .WORD  $0000, $0000
000106          Cst0003
000106 32          .BYTE  50
000107 2020 2020 2020    .ASCII  '-----'
00010D 2020 2020 2020    .ASCII  '---- '
000113 2020 2020 2020    .ASCII  ' ----'
000119 2020 2020 2020    .ASCII  '-----'
00011F 2020 2020 2020    .ASCII  '---- '
000125 2020 2020 2020    .ASCII  ' ----'
00012B 2020 2020 2020    .ASCII  '-----'
000131 2020 2020 2020    .ASCII  '-----'
000137 2020          .ASCII  '---'
000139 00          .BYTE  $00
00013A          Cst0004
00013A 27          .BYTE  39
00013B 4974 6572 6174    .ASCII  'Iterat'
000141 696F 6E73 2020    .ASCII  'ions '
000147 2020 4475 7261    .ASCII  ' Dura'
00014D 7469 6F6E 2028    .ASCII  'tion ('
000153 7365 6329 2020    .ASCII  'sec) '
000159 2020 5265 7369    .ASCII  ' Resi'
00015F 6475 65          .ASCII  'due'
000162          Last

385 54 -D A          END: { ----- Show_Execution_Results ----- }
386 55 --
387 56 --          { ----- }
388 57 --
389 58 -D A          PROCEDURE Perform_SAVAGE_Algorithm( iter_loop : INTEGER;
390 59 --          VAR exec_time : Milliseconds;

```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

391      60 --                                VAR residue : t_FP_Number);
392      61 --
393      62 --                                VAR
394      63 --                                i      : INTEGER;      { Loop counter }
395      64 --                                a      : t_FP_Number;  { Computation variable }
396      65 --                                start  : Milliseconds; { Starting time (in ms) }
397      66 --                                finish : Milliseconds; { Ending time (in ms) }
398      67 --
399      68 0- A      BEGIN { ----- Perform SAVAGE_Algorithm ----- }

                                000000 4A6F EFEC      PERFORM_ TST. W  $EFEC(A7)
                                000004 4E56 FFB8      LINK      A6, #FFB8
                                000008 48E7 0700      MOVEM. L  D5-D7, -(A7)

400      69 --                                start := Timer; { Start the timing }

                                00000C 42A7                                CLR. L      -(A7)
                                00000E 4EBA 0000      JSR      TIMER
                                000012 2C1F      MOVE. L  (A7)+, D6

401      70 --
402      71 --                                a := 1.0; { Initialize the benchmark variable }

                                000014 41EE FFF4      LEA      $FFF4(A6), A0
                                000018 43FA 0114      LEA      Cst0001, A1      ; 0000012E
                                00001C 20D9      MOVE. L  (A1)+, (A0)+
                                00001E 20D9      MOVE. L  (A1)+, (A0)+
                                000020 3091      MOVE. W  (A1), (A0)

403      72 --
404      73 --                                FOR i := 1 TO iter_loop DO { Perform the actual benchmark }

                                000022 306E 0010 FFEA      MOVE. W  $0010(A6), $FFEA(A6)
                                000028 7E01      MOVEQ   #01, D7
                                00002A 6000 0084      BRA      L0001      ; 00000080

405      74 --                                a := Tan(ArcTan(Exp(Ln(Sqrt(a*a)))))) + 1.0;

                                00002E 486E FFE0      L0003      PEA      $FFE0(A6)
                                000032 486E FFD6      PEA      $FFD6(A6)
                                000036 486E FFCC      PEA      $FFCC(A6)
                                00003A 486E FFC2      PEA      $FFC2(A6)
                                00003E 486E FFF4      PEA      $FFF4(A6)
                                000042 41EE FFB8      LEA      $FFB8(A6), A0
                                000046 43EE FFF4      LEA      $FFF4(A6), A1
                                00004A 20D9      MOVE. L  (A1)+, (A0)+
                                00004C 20D9      MOVE. L  (A1)+, (A0)+
                                00004E 3091      MOVE. W  (A1), (A0)
                                000050 486E FFB8      PEA      $FFB8(A6)
                                000054 3F3C 0004      MOVE. W  #0004, -(A7)
                                000058 4EBA 0000      JSR      FP68K
                                00005C 486E FFB8      PEA      $FFB8(A6)
                                000060 3F3C 0012      MOVE. W  #0012, -(A7)
                                000064 4EBA 0000      JSR      FP68K
                                000068 486E FFB8      PEA      $FFB8(A6)
                                00006C 4EBA 0000      JSR      $LN
                                000070 588F      ADDQ. L  #04, A7
                                000072 486E FFC2      PEA      $FFC2(A6)
                                000076 4EBA 0000      JSR      $EXP
                                00007A 588F      ADDQ. L  #04, A7
                                00007C 486E FFCC      PEA      $FFCC(A6)

```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

000080 4EBA 0000      JSR      $_ARCTAN
000084 588F      ADDQ.L  #$4, A7
000086 486E FFD6      PEA      $FFD6(A6)
00008A 4EBA 0000      JSR      TAN
00008E 588F      ADDQ.L  #$4, A7
000090 487A 009C      PEA      Cst0001      ; 0000012E
000094 486E FFE0      PEA      $FFE0(A6)
000098 4267      CLR.W   -(A7)
00009A 4EBA 0000      JSR      FP68K
00009E 41EE FFF4      LEA      $FFF4(A6), A0
0000A2 43EE FFE0      LEA      $FFE0(A6), A1
0000A6 2009      MOVE.L  (A1)+, (A0)+
0000A8 2009      MOVE.L  (A1)+, (A0)+
0000AA 3091      MOVE.W  (A1), (A0)
0000AC 5247      ADDQ.W  #$1, D7
0000AE 690B      BVS.S   L0002      ; 000000B8
0000B0 BE6E FFEA      CMP.W   $FFE0(A6), D7
0000B4 6F00 FF7B      BLE     L0003      ; 0000002E

406 75 --      { <----- = a -----> }
407 76 --
408 77 --      finish := Timer; { End the timing }

0000B8 42A7      L0002  CLR.L  -(A7)
0000BA 4EBA 0000      JSR      TIMER
0000BE 2A1F      MOVE.L  (A7)+, D5

409 78 --
410 79 --      exec_time := finish - start; { Determine real execution time }

0000C0 2005      MOVE.L  D5, D0
0000C2 9086      SUB.L   D6, D0
0000C4 206E 000C      MOVE.L  $000C(A6), A0
0000C8 2080      MOVE.L  D0, (A0)

411 80 --
412 81 --      residue := ((iter_loop * 1.0) + 1.0) - a; { Return the residue value }

0000CA 486E 0010      PEA      $0010(A6)
0000CE 486E FFE0      PEA      $FFE0(A6)
0000D2 3F3C 200E      MOVE.W  #$200E, -(A7)
0000D6 4EBA 0000      JSR      FP68K
0000DA 487A 0052      PEA      Cst0001      ; 0000012E
0000DE 486E FFE0      PEA      $FFE0(A6)
0000E2 3F3C 0004      MOVE.W  #$0004, -(A7)
0000E6 4EBA 0000      JSR      FP68K
0000EA 487A 0042      PEA      Cst0001      ; 0000012E
0000EE 486E FFE0      PEA      $FFE0(A6)
0000F2 4267      CLR.W   -(A7)
0000F4 4EBA 0000      JSR      FP68K
0000F8 486E FFF4      PEA      $FFF4(A6)
0000FC 486E FFE0      PEA      $FFE0(A6)
000100 3F3C 0002      MOVE.W  #$0002, -(A7)
000104 4EBA 0000      JSR      FP68K
000108 206E 000B      MOVE.L  $000B(A6), A0
00010C 43EE FFE0      LEA      $FFE0(A6), A1
000110 2009      MOVE.L  (A1)+, (A0)+
000112 2009      MOVE.L  (A1)+, (A0)+

```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

000114 3091          MOVE.W (A1), (A0)
000116 4C0F 00E0      MOVEH.L (A7)+, D5-D7
00011A 4E5E          UNLK     A6
00011C 205F          MOVE.L (A7)+, A0
00011E DEFC 000A      ADDA.W  #$000A, A7
000122 4ED0          JMP      (A0)

000124 0045 5246 4F52  .WORD   $0045, $5246, $4F52 ; ".ERFOR"
00012A 405F          .WORD   $405F ; "H_"

00012C 000A          CstSize .WORD   Last-CstSize-2
00012E          Cst0001
00012E 3FFF 8000 0000 .WORD   $3FFF, $8000, $0000
000134 0000 0000      .WORD   $0000, $0000
000138          Last

413 82 -0 A          END; { ----- Perform_SAVAGE_Algorithm ----- }
414 83 --
415 84 --          { ===== }
416 85 --
417 86 0-          BEGIN { ----- Savage_Residue_FP_Benchmark ----- }

000000 4EBA 0000      SAVAGE_R JSR      $_BEGIN
000004 4E56 0000      LINK      A6, #$0000
000008 2C5F          MOVE.L (A7)+, A6
00000A 4E55 FFE6      LINK      A5, $FFE6
00000E 9FED 0010      SUBA.L  $0010(A5), A7
000012 4EBA 0000      JSR      $_INIT

418 87 --          Writeln('Apple Lisa Floating Point SAVAGE RESIDUE Benchmark');

000016 2F2D 000C      MOVE.L  $000C(A5), -(A7)
00001A 487A 0142      PEA      Cst0006 ; 0000015E
00001E 4267          CLR.W  -(A7)
000020 4EBA 0000      JSR      $_STR
000024 2F2D 000C      MOVE.L  $000C(A5), -(A7)
000028 4EBA 0000      JSR      $_LN

419 88 --          Writeln;

00002C 2F2D 000C      MOVE.L  $000C(A5), -(A7)
000030 4EBA 0000      JSR      $_LN

420 89 --          Writeln('Savage Benchmark is executing ...');

000034 2F2D 000C      MOVE.L  $000C(A5), -(A7)
000038 487A 0102      PEA      Cst0005 ; 0000013C
00003C 4267          CLR.W  -(A7)
00003E 4EBA 0000      JSR      $_STR
000042 2F2D 000C      MOVE.L  $000C(A5), -(A7)
000046 4EBA 0000      JSR      $_LN

421 90 --          Writeln;

00004A 2F2D 000C      MOVE.L  $000C(A5), -(A7)
00004E 4EBA 0000      JSR      $_LN

422 91 --
423 92 --          { Make certain the SAME FP engine code is loaded }
424 93 --

```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

425  94 --      q_residue := SIN(10.0) * 1.2345; { Don't Care values }

      000052 486D FFE6      PEA      $FFE6(A5)
      000056 487A 0004      PEA      Cst0004      ; 00000132
      00005A 4EBA 0000      JSR      %_SIN
      00005E 588F      ADDQ.L  #84, A7
      000060 487A 0006      PEA      Cst0003      ; 00000128
      000064 486D FFE6      PEA      $FFE6(A5)
      000068 3F3C 0004      MOVE.W  #0004, -(A7)
      00006C 4EBA 0000      JSR      FP68K
      000070 41ED FFF0      LEA      $FFF0(A5), A0
      000074 43ED FFE6      LEA      $FFE6(A5), A1
      000078 20D9      MOVE.L  (A1)+, (A0)+
      00007A 20D9      MOVE.L  (A1)+, (A0)+
      00007C 3091      MOVE.W  (A1), (A0)

426  95 --
427  96 --      { Perform the actual SAVAGE benchmarks }
428  97 --
429  98 --      Show_Execution_Results(TRUE,0,0,0.0);

      00007E 1F3C 0001      MOVE.B  #0001, -(A7)
      000082 4267      CLR.W  -(A7)
      000084 42A7      CLR.L  -(A7)
      000086 487A 0006      PEA      Cst0002      ; 0000011E
      00008A 4EBA 0000      JSR      SHOW_EXE

430  99 --
431 100 --      FOR q_iteration := 1 TO 50 DO

      00008E 387C 0001 FFFE      MOVE.W  #0001, $FFE(A5)
      000094 6032      BRA.S  L0001      ; 000000C8

432 101 1-      BEGIN
433 102 --      Perform_SAVAGE_Algorithm      (q_iteration*200,q_time,q_residue);

      000096 303C 00C8      L0002      MOVE.W  #00C8, D0
      00009A C1ED FFFE      MULLS  $FFE(A5), D0
      00009E 3F00      MOVE.W  D0, -(A7)
      0000A0 486D FFFA      PEA      $FFFA(A5)
      0000A4 486D FFF0      PEA      $FFF0(A5)
      0000A8 4EBA 0000      JSR      PERFORM_

434 103 --      Show_Execution_Results      (FALSE,q_iteration*200,q_time,q_residue);

      0000AC 4267      CLR.W  -(A7)
      0000AE 303C 00C8      MOVE.W  #00C8, D0
      0000B2 C1ED FFFE      MULLS  $FFE(A5), D0
      0000B6 3F00      MOVE.W  D0, -(A7)
      0000B8 2F2D FFFA      MOVE.L  $FFFA(A5), -(A7)
      0000BC 486D FFF0      PEA      $FFF0(A5)
      0000C0 4EBA 0000      JSR      SHOW_EXE
      0000C4 526D FFFE      ADDQ.W  #1, $FFE(A5)
      0000C8 0C6D 0032 FFFE L0001      CMPL.W  #0032, $FFE(A5)
      0000CE 6FC6      BLE.S  L0002      ; 00000096

435 104 -1      END;
436 105 --
437 106 --      { Aufwiedersehen ... }
438 107 --
439 108 --      WRITELN;

```



*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

```

000000 2F2D 000C      MOVE.L  $000C(A5), -(A7)
000004 4EBA 0000      JSR    %N_LN

440      109 --      WRITELN('That's all, Folks...');

000008 2F2D 000C      MOVE.L  $000C(A5), -(A7)
00000C 487A 002A      PEA    Cst0001 ; 00000108
000010 4267      CLR.W  -(A7)
000014 4EBA 0000      JSR    %N_STR
000018 2F2D 000C      MOVE.L  $000C(A5), -(A7)
00001C 4EBA 0000      JSR    %N_LN

000020 4EBA 0000      JSR    %_TERM
000024 4E5D      UNLK   A5
000028 4EBA 0000      JSR    %_END
00002C 4E75      RTS
000030 4E5E      UNLK   A6
000034 4E75      RTS

000038 D341 5641 4745 .WORD  $D341, $5641, $4745 ; ". AVAGE"
00003C 5F52      .WORD  $5F52 ; "_R"

000040 008A      CstSize .WORD  Last-CstSize-2
000044 0008      Cst0001
000048 14      .BYTE  20
00004C 5468 6174 2773 .ASCII  "That's"
000050 2061 6C6C 2C20 .ASCII  ' all '
000054 466F 6C6B 732E .ASCII  'Folks. '
000058 2E2E      .ASCII  '...'
00005C 00      .BYTE  $00
000060 0011E      Cst0002
000064 0000 0000 0000 .WORD  $0000, $0000, $0000
000068 0000 0000 0000 .WORD  $0000, $0000
00006C 00128      Cst0003
000070 3FFF 9E04 1893 .WORD  $3FFF, $9E04, $1893
000074 74BC 6A7F      .WORD  $74BC, $6A7F
000078 00132      Cst0004
00007C 4002 A000 0000 .WORD  $4002, $A000, $0000
000080 0000 0000 0000 .WORD  $0000, $0000
000084 0013C      Cst0005
000088 0013C 21      .BYTE  33
00008C 5361 7661 6765 .ASCII  'Savage'
000090 2042 656E 6368 .ASCII  ' Bench'
000094 6D61 726B 2069 .ASCII  'mark i'
000098 7320 6578 6563 .ASCII  's exec'
00009C 7574 696E 6720 .ASCII  'uting '
0000A0 2E2E 2E      .ASCII  '...'
0000A4 0015E      Cst0006
0000A8 0015E 32      .BYTE  50
0000AC 4170 706C 6520 .ASCII  'Apple '
0000B0 4C69 7361 2046 .ASCII  'Lisa F'
0000B4 6C6F 6174 696E .ASCII  'loatin'
0000B8 6720 506F 696E .ASCII  'g Poin'

```

### Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)

```
000177 7420 5341 5641 .ASCII 't SAVA'
000170 4745 2052 4553 .ASCII 'GE RES'
000183 4944 5545 2042 .ASCII 'IDUE B'
000189 656E 6368 6D61 .ASCII 'enchma'
00018F 7268 .ASCII 'rk'
000191 00 .BYTE $00
000192 Last
```

[illegible]

Elapsed compilation time: 15.917 seconds.  
Compilation complete - no errors found. 440 lines.  
Elapsed code generator time: 11.230 seconds.  
Total code size = 1068

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

## WorkShop Compilation Listings

### Pascal Compiler:

```
Lisa Pascal Compiler V3.76 (05-Apr-85)                08:28:38 02-Oct-87
(c)1981 SVS, Inc. (c)1983, 1984 Apple Computer, Inc.

Input file - [.TEXT] LISA/FP_SAVAGE_RESIDUE
List file - [.TEXT]
Output file - [LISA/FP_SAVAGE_RESIDUE] [.OBJ]

[231545 words] SHOW_EXE
[231394 words] PERFORM_
[231472 words] SAVAGE_R

Elapsed time: 13.507 seconds.

Compilation complete - no errors found.  448 lines.
```

### Code Generator:

```
Lisa Pascal MC68000 Code Generator V3.65 (20-Mar-85)    08:29:02 02-Oct-87
(c)1981 SVS, Inc. (c)1983, 1984 Apple Computer, Inc.

Input file - $I+
Input file - [.I] LISA/FP_SAVAGE_RESIDUE
Output file - [LISA/FP_SAVAGE_RESIDUE] [.OBJ] LISA/FP_SAVAGE_RESIDUE

SHOW_EXE - SHOW_EXE    Code size =   354
PERFORM_ - PERFORM_    Code size =   312
SAVAGE_R - SAVAGE_R    Code size =   402

Elapsed time: 3.413 seconds.

Total code size = 1068
```

### Linker:

```
Linker - M68000 Object Code  v0.9.3.1 08-Apr-85 15:26:15
Copyright Apple Computer, Inc. 1985

Beginning memory:      269656
After initial allocation: 233146
Input file [.OBJ] ? LISA/FP_SAVAGE_RESIDUE
Input file [.OBJ] ? IOSPasLib
Input file [.OBJ] ? LISA/SANELIBASH
Input file [.OBJ] ? SYS1LIB
Input file [.OBJ] ?
Listing file [-CONSOLE] / [.TEXT]
Output file ? [.OBJ] LISA/FP_SAVAGE_RESIDUE
Reading file: LISA/FP_SAVAGE_RESIDUE.OBJ
Reading file: LISA/SANELIBASH.OBJ
Reading file: IOSPasLib.OBJ
Reading file: SYS1LIB.OBJ
Input summary:
  4 Files      , max =   100
 32 Segments   , max =  4096
180 Modules    , max = 32768
138 Entries    , max = 65536
 78 Ref. Lists, max = 65536
240 References, max = 65536
Linking Main Program.
Reading Library Directory: -#12-INTRINSIC.LIB
```

*Apple Lisa Floating Point SAVAGE RESIDUE Benchmark (2-Oct-87)*

Active: 13 of 180 read.  
Visible: 9 of 138 read.  
Global data: \$00001A  
Common data: \$000000  
Number of segments in file = 4, number of Jump Table entries = 9  
Linking segment: file (JT) seg: 1 size: 1068  
Linking segment: SANE file (JT) seg: 2 size: 1556  
Linking segment: FP68K file (JT) seg: 3 size: 4528  
Linking segment: Elens68K file (JT) seg: 4 size: 4410  
0 ERRORS detected.

LISA/FP\_SAVAGE\_RESIDUE.00J is an executable program file.

Elapsed time: 31.507 seconds.  
That's all Folks!

--- FINIS ---