Integrated MicroProcessor-16

IMP-16F/400

FLOATING POINT Firmware

TECHNICAL DESCRIPTION

January 1975

National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
INTRODUCTION

This subroutine set provides an IMP-16 microprocessor with double-precision and floating-point capability. Double-precision and floating-point computation is very useful when additional precision is needed or the range of numbers is expected to be large. In the following pages, individual subroutines are described along with necessary user instructions. Subroutines included in the set are listed in the table below.

ARITHMETIC SUBROUTINE SET

<table>
<thead>
<tr>
<th>SUBROUTINE</th>
<th>MNEMONIC</th>
<th>ENTRY POINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Precision Multiply</td>
<td>MULT</td>
<td>FC01</td>
</tr>
<tr>
<td>Single Precision Divide</td>
<td>DIV</td>
<td>FC1A</td>
</tr>
<tr>
<td>Double Precision Multiply</td>
<td>DPMUL</td>
<td>FCCF</td>
</tr>
<tr>
<td>Positive Double Precision Divide</td>
<td>DPDIV</td>
<td>FDE3</td>
</tr>
<tr>
<td>Double Precision Square</td>
<td>DPSQUARE</td>
<td>FCCD</td>
</tr>
<tr>
<td>Double Precision Complement</td>
<td>DPCOMP</td>
<td>FDBA</td>
</tr>
<tr>
<td>Double Precision Shift</td>
<td>DPSH</td>
<td>FDBB</td>
</tr>
<tr>
<td>Double Precision Shift Right</td>
<td>DPSHR</td>
<td>FDCE</td>
</tr>
<tr>
<td>Double Precision Shift Left</td>
<td>DPSHL</td>
<td>FDBB</td>
</tr>
<tr>
<td>Quadrant Tests</td>
<td>QUAD</td>
<td>FDA0</td>
</tr>
<tr>
<td>Sine</td>
<td>SIN</td>
<td>FCFA</td>
</tr>
<tr>
<td>Cosine</td>
<td>COS</td>
<td>FCFC</td>
</tr>
<tr>
<td>Arctangent</td>
<td>ARCTAN</td>
<td>FD37</td>
</tr>
<tr>
<td>Floating Point Add</td>
<td>FPADD</td>
<td>FC30</td>
</tr>
<tr>
<td>Floating Point Multiply</td>
<td>FPMUL</td>
<td>FC5D</td>
</tr>
<tr>
<td>Floating Point Divide</td>
<td>FPDIV</td>
<td>FC6C</td>
</tr>
<tr>
<td>Floating Point Complement</td>
<td>FPCOMP</td>
<td>FCB9</td>
</tr>
<tr>
<td>Check Zero Exponent</td>
<td>CZERO</td>
<td>FC63</td>
</tr>
<tr>
<td>Extract Exponent to Stack</td>
<td>EXTEXP</td>
<td>FC7C</td>
</tr>
<tr>
<td>Add Exponent from Stack</td>
<td>ADDEXP</td>
<td>FCBD</td>
</tr>
<tr>
<td>Left Normalize</td>
<td>LFNORM</td>
<td>FC97</td>
</tr>
<tr>
<td>Double Left Normalize</td>
<td>DLNORM</td>
<td>FCSA</td>
</tr>
<tr>
<td>Fraction to Floating Point</td>
<td>FLOAT</td>
<td>FC85</td>
</tr>
<tr>
<td>Floating Point to Fraction</td>
<td>SFO</td>
<td>FCC7</td>
</tr>
</tbody>
</table>

GENERAL CHARACTERISTICS

The arithmetic subroutine set uses 512 memory words located at addresses FC00 through FDFF (64512 through 65023, decimal). All subroutines are contained wholly within the set; they require no external program code. The IMP-16 extended instruction set is required. Operands and results, in most cases, use the general registers. The only memory locations written into by the set are in the range 00E0 through 00EF. All constants are contained within the subroutine set.

EXECUTION TIMES

Execution times shown on each of the subroutine data sheets are expressed in terms of three variables, where

\[
R = \text{Number of main memory read cycles} \\
W = \text{Number of main memory write cycles} \\
N = \text{Number of microprogram cycles}
\]
These execution time expressions can thus be used to calculate subroutine execution time for any IMP-16 implementation. A few subroutine execution times are also dependent on user-specified variables (such as shift counts in shift instructions). These are defined on the individual data sheets. All execution times are based on worst-case conditions.

NUMBER REPRESENTATION

Numbers in a computer program often represent real world parameters. In this subroutine set, four different modes of numerical representation are used: integer, fractional, double-precision, and floating-point. All four modes have important applications in IMP-16 systems.

Integer notation is the most common and obvious mode of numerical representation. It is simply an integer count of something. In the IMP-16, numbers up to $2^{16}$ may be represented because of the 16-bit word length. To represent both positive and negative integers, a twos-complement system is used. This gives an integer range of

$$-2^{15} \leq \text{integer} < 2^{15}$$

With twos-complement notation, the most significant bit, bit 15, can be thought of as a sign bit, and the binary point can be thought of as being to the right of bit 0.

FRACTIONAL NOTATION

Many real-world parameters are continuous rather than discrete. Such values can never be indicated with perfect precision because of the limited word length of any computer. Take the airspeed of an aircraft as an example. It may have a potential range from zero to 1000 miles per hour. We would rarely expect its speed to be an exact round number. At some instant the speed may be 556.234.... If we want to express this speed as precisely as possible given a 16-bit word length, an integer notation of 556 miles per hour would not be the best choice. A more convenient technique is to use fractional notation. The parameter, in this case speed, can be scaled so that 1000 miles per hour is equivalent to 1.0000.... This would allow the full 16-bit capacity to be utilized. After scaling, the range of numerical expression of a fractional number is

$$-1.0 \leq N < 1.0$$

With fractional notation, the binary point can be thought of as being to the right of bit 15. Thus, the largest positive binary number that can be represented is 0.111111... or almost one. Precision is about $3.1 \times 10^{-5}$.

DOUBLE PRECISION

When precision greater than 16-bits is required, double-precision numbers can be used. Double-precision numbers give a precision of one part in $2^{31}$ or about $4.6 \times 10^{-10}$. Double-precision numbers are an extended fractional notation. Note that two IMP-16 words are required.

FLOATING POINT

Floating-point representation is a technique to express numbers in a form similar to scientific notation, with a fractional mantissa and an exponent. The value of any floating-point number so represented is equivalent to mantissa * (2**exponent). The mantissa consists of 24 bits and gives a precision of one part in $2^{23}$. The exponent has 8 bits and allows exponents in the range

$$-2^7 \leq \text{exponent} < 2^7$$
The great advantage of floating-point numbers is that the scaling factor of a fractional number becomes a visible attribute which can be manipulated. Thus, the scaling factor can be altered to maintain precision or facilitate computations.

**NUMBER REPRESENTATION EXAMPLES**

- **INTEGER**
  - RANGE: $-32768 \leq N \leq 32768$
  - MAXIMUM PRECISION: 1
  - SMALLEST VALUE: 1

  EXAMPLES:
  - $5 = 0005_{16}$
  - $-5 = FFFB_{16}$

  SPEED = $556.234 - 556_{10} = 022C_{16}$

- **FRACTIONAL**
  - RANGE: $-1 \leq N < 1$
  - MAXIMUM PRECISION: $1/2^{15} = 3.1 \times 10^{-5}$
  - SMALLEST VALUE: $2^{-15}$

  EXAMPLES:
  - $1/2 = 4000_{16}$
  - $-1/2 = C000_{16}$
  - $-1 = 8000_{16}$

  RANGE: $-1 \leq N < 1$
  SPEED = $1000.000 = 0.556234_{10} = 4732_{16}$

- **DOUBLE PRECISION (FRACTIONAL)**
  - RANGE: $-1 \leq N < 1$
  - MAXIMUM PRECISION: $1/2^{31} = 4.6 \times 10^{-10}$
  - SMALLEST VALUE: $2^{-31}$

  EXAMPLES:
  - $1/2 = 40000000_{16}$
  - $-1/2 = C0000000_{16}$
  - $-1 = 80000000_{16}$

  SPEED = $4732\ ACFB_{16}$

- **FLOATING POINT**
  - RANGE: $-2^{127} \leq N \leq 2^{127}$
  - PRECISION: $2^{-23}$ ($2^{E}$)
  - SMALLEST MAGNITUDE: $2^{-127}$
  - LARGEST MAGNITUDE: $(1-2^{-23}) 2^{127}$

  MANTISSA (M)  EXponent (E)
  \[ N = M \times 2^E \]

  EXAMPLES:
  - $1/2 = 4000\ 0016$ (0016)
  - $1/2 = 2000\ 0016$ (0116)
  - $1/2 = 1000\ 0016$ (0216)
  - $1/4 = 2000\ 0016$ (0016)
  - $1/4 = 4000\ 0016$ (FF16)

  SPEED = $556.234 = 4587\ 7A_{16} = 0A_{16}$
ANGULAR REPRESENTATION

Angles are expressed as a double-precision fraction of $\pi$ radians.

NOTATION

In the descriptions which follow, R0, R1, R2, and R3 represent the contents of the four accumulators = AC0, AC1, AC2, and AC3.

Normalization of a floating point number means that the mantissa is shifted left until the most significant bit is a 1, and the exponent adjusted accordingly.
### SUBROUTINE: SINGLE PRECISION MULTIPLY

**LABEL: MULT**

**DESCRIPTION:** Multiplies two single-precision, signed, fractional 16-bit numbers to give a sign plus a 30-bit result

**ENTRY POINT:** FC01
**ENTRY CONDITIONS:** $R_0 = X$  

**EXIT CONDITIONS:** $R_0, R_1 = X \times Y$  

**EXECUTION TIME:** $30R + 243N + W$

**TEMPORARY LOCATIONS USED:** 00E0
**SUBROUTINES CALLED:** DSHL, DPCOMP
**STACK WORDS NEEDED:** 2

**COMMENTS:**
1. All addresses are expressed in hexadecimal notation.
2. Fractional differs from an integer multiply in that the result of a fractional multiply is shifted left 1 bit.

### SUBROUTINE: SINGLE PRECISION DIVIDE

**LABEL: DIV**

**DESCRIPTION:** Divides a signed, fractional 16-bit number into a signed, fractional 32-bit number, providing a quotient and an integer remainder

**ENTRY POINT:** FC1A
**ENTRY CONDITIONS:** $R_0, R_1 = \text{DIVIDEND}$  

**EXIT CONDITIONS:** $R_0 = \text{QUOTIENT}$  

**EXECUTION TIME:** $49R + W + 343N$

**TEMPORARY LOCATIONS USED:** 00E0
**SUBROUTINES CALLED:** DSHR, DPCOMP
**STACK WORDS NEEDED:** 2

**COMMENTS:**
1. All addresses are expressed in hexadecimal notation.
2. Overflow flag is set if MSB of dividend $\geq$ divisor.
SUBROUTINE: DOUBLE PRECISION MULTIPLY  
DESCRIPTION: MULTIPLIES DOUBLE-PRECISION OPERANDS TO GIVE A DOUBLE-PRECISION PRODUCT  
ENTRY POINT: FCCF  
ENTRY CONDITIONS: \( R_0, R_1 = X \quad R_2, R_3 = Y \)  
EXIT CONDITIONS: \( R_0, R_1 = X \times Y \quad R_2, R_3 \text{ ALTERED} \)  
EXECUTION TIME: \( 200R + 8W + 1125N \)  
TEMPORARY LOCATIONS USED: 00E0, 00E1, 00EE, 00EF  
SUBROUTINES CALLED: QUAD, MULT, DPCOMP, DSHL  
STACK WORDS NEEDED: 4  
(MAXIMUM DEPTH)  
COMMENTS: ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.  

SUBROUTINE: POSITIVE DOUBLE PRECISION DIVIDE  
DESCRIPTION: UNSIGNED FRACTIONAL DIVIDE OF \( R_2, R_3 \) INTO \( R_0, R_1 \), BOTH DOUBLE-PRECISION POSITIVE FRACTIONS  
ENTRY POINT: FDE3  
ENTRY CONDITIONS: \( R_0, R_1 = \text{POSITIVE DIVIDEND} \quad R_2, R_3 = \text{POSITIVE DIVISOR} \)  
EXIT CONDITIONS: \( R_0, R_1 = \text{QUOTIENT} \quad R_2, R_3 = \text{REMAINDER} \quad \text{SELFFF} = 0 \)  
EXECUTION TIME: \( 510R + 97W = 2578N \)  
TEMPORARY LOCATIONS USED: 00E0, 00EC-00EF  
SUBROUTINES CALLED: NONE  
STACK WORDS NEEDED: 0  
(MAXIMUM DEPTH)  
COMMENTS: ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.
<table>
<thead>
<tr>
<th>SUBROUTINE: DOUBLE PRECISION SQUARE</th>
<th>LABEL: DPSQUARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION: SQUares THE DOUBLE-PRECISION NUMBER IN R₀, R₁</td>
<td></td>
</tr>
<tr>
<td>ENTRY POINT: FCCD</td>
<td></td>
</tr>
<tr>
<td>ENTRY CONDITIONS: R₀, R₁ = X</td>
<td></td>
</tr>
<tr>
<td>EXIT CONDITIONS: R₀, R₁ = X² R₂, R₃ ALTERED</td>
<td></td>
</tr>
<tr>
<td>EXECUTION TIME: 202R + 8W + 1137N</td>
<td></td>
</tr>
<tr>
<td>TEMPORARY LOCATIONS USED: 00E0, 00E1, 00EE, 00EF</td>
<td></td>
</tr>
<tr>
<td>SUBROUTINES CALLED: DPMULT, QUAD, MULT, DPCOMP, DSHL</td>
<td></td>
</tr>
<tr>
<td>STACK WORDS NEEDED: 4</td>
<td></td>
</tr>
<tr>
<td>(MAXIMUM DEPTH)</td>
<td></td>
</tr>
<tr>
<td>COMMENTS: ALL ADDRESSED ARE EXPRESSED IN HEXADECIMAL NOTATION.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SUBROUTINE: DOUBLE PRECISION COMPLEMENT</th>
<th>LABEL: DPCOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION: COMPUTES TWOS COMPLEMENT OF A DOUBLE-PRECISION NUMBER</td>
<td></td>
</tr>
<tr>
<td>ENTRY POINT: FDBA</td>
<td></td>
</tr>
<tr>
<td>ENTRY CONDITIONS: R₀, R₁ = X</td>
<td></td>
</tr>
<tr>
<td>EXIT CONDITIONS: R₀, R₁ = TWOS COMPLEMENT OF X R₂, R₃ PRESERVED SELFF = 0</td>
<td></td>
</tr>
<tr>
<td>EXECUTION TIME: 16R + 53N</td>
<td></td>
</tr>
<tr>
<td>TEMPORARY LOCATIONS USED: NONE</td>
<td></td>
</tr>
<tr>
<td>SUBROUTINES CALLED: NONE</td>
<td></td>
</tr>
<tr>
<td>STACK WORDS NEEDED: 0</td>
<td></td>
</tr>
<tr>
<td>(MAXIMUM DEPTH)</td>
<td></td>
</tr>
<tr>
<td>COMMENTS: ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.</td>
<td></td>
</tr>
</tbody>
</table>
## DOUBLE PRECISION ARITHMETIC SHIFT

**DESCRIPTION:** Shifts a double-precision number to right or left as specified by \( R_2 \). Absolute value of \( R_2 \) indicates number of bits to shift. Positive \( R_2 \) indicates left shift; negative \( R_2 \) means right shift.

**ENTRY POINT:** FDDB
**ENTRY CONDITIONS:** \( R_0, R_1 = X \quad R_2 = M \) (SHIFT COUNT)
**EXIT CONDITIONS:** \( R_0, R_1 = R_2 \text{ SHIFTED } X \quad R_2 = \text{ALTED} \quad R_3 \text{ PRESERVED}
**EXECUTION TIME:** LEFT \( 5R + 20N + \left[8R + 32N\right] M \)
RIGHT \( 7R + 26N + \left[4R + 21N\right] M \)

**TEMPORARY LOCATIONS USED:** NONE
**SUBROUTINES CALLED:** NONE
**STACK WORDS NEEDED:** 0
(MAXIMUM DEPTH)

**COMMENTS:** All addresses are expressed in hexadecimal notation. Right shift is arithmetic; that is, sign bit is preserved.

## DOUBLE PRECISION ARITHMETIC SHIFT RIGHT

**DESCRIPTION:** Shifts double-precision number right \( M \) bits, where \( R_2 = M \)

**ENTRY POINT:** FDCE
**ENTRY CONDITIONS:** \( R_0, R_1 = X \quad R_2 = M \) (BITS TO SHIFT)
**EXIT CONDITIONS:** \( R_0, R_1 = \text{RIGHT SHIFTED } X \quad R_2 \text{ ALTERED} \quad R_3 \text{ PRESERVED}
**EXECUTION TIME:** \( 5R + 20N + (4R + 21N) M \)

**TEMPORARY LOCATIONS USED:** NONE
**SUBROUTINES CALLED:** NONE
**STACK WORDS NEEDED:** 0
(MAXIMUM DEPTH)

**COMMENTS:** All addresses are expressed in hexadecimal notation. Sign bit is preserved.
### SUBROUTINE: DOUBLE PRECISION SHIFT LEFT

**LABEL:** DPHSL

**DESCRIPTION:** Shifts double-precision number left \( M \) bits, where \( R_2 = M \)

<table>
<thead>
<tr>
<th>ENTRY POINT:</th>
<th>FDDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRY CONDITIONS:</td>
<td>( R_0, R_1 = X ), ( R_2 = M ) (Number of bits to shift)</td>
</tr>
<tr>
<td>EXIT CONDITIONS:</td>
<td>( R_0, R_1 = ) left shifted ( X ), ( R_2 ) altered, ( R_3 ) preserved</td>
</tr>
<tr>
<td>EXECUTION TIME:</td>
<td>( 5R + 20N + (8R + 32N) M )</td>
</tr>
</tbody>
</table>

**TEMPORARY LOCATIONS USED:** NONE
**SUBROUTINES CALLED:** NONE
**STACK WORDS NEEDED:** 0 (Maximum Depth)

**COMMENTS:** All addresses are expressed in hexadecimal notation.

---

### SUBROUTINE: QUADRANT TESTS

**LABEL:** QUAD

**DESCRIPTION:** Given two signed double-precision numbers \( C_1 \) and \( C_2 \), the initial sign of each is indicated by a flag in memory and the absolute value of the numbers is given.

<table>
<thead>
<tr>
<th>ENTRY POINT:</th>
<th>FDAO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRY CONDITIONS:</td>
<td>( R_0, R_1 = C_1 ), ( R_2, R_3 = C_2 )</td>
</tr>
<tr>
<td>EXIT CONDITIONS:</td>
<td>( R_0, R_1 = ) abs ( (C_1) ), ( R_2, R_3 = ) abs ( (C_2) )</td>
</tr>
<tr>
<td>EXECUTION TIME:</td>
<td>( 47R + W + 175N )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( C_1 )</th>
<th>( C_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>+</td>
</tr>
</tbody>
</table>

**WHERE ZERO IS:** +

**TEMPORARY LOCATIONS USED:** 00E1
**SUBROUTINES CALLED:** DPCOMP
**STACK WORDS NEEDED:** 1 (Maximum Depth)

**COMMENTS:** All addresses are expressed in hexadecimal notation.
### SUBROUTINE: SINE

**DESCRIPTION:** Calculates sine of double-precision angle in \( R_0, R_1 \)

- Calculation is made: \( \sin(x) = \cos(x - \pi/2) \)
- Accuracy is 0.000000 0002(10)

**ENTRY POINT:** FCFA
**ENTRY CONDITIONS:** \( R_0, R_1 = X \)
**EXIT CONDITIONS:** \( R_0, R_1 = \sin(x) \quad R_2, R_3 \) altered
**EXECUTION TIME:** 1697R + 82W + 9300 N

**TEMPORARY LOCATIONS USED:** 00E0–00EB, 00EE, 00EF
**SUBROUTINES CALLED:** DPCOMP, DPMULT, PEXPN
**STACK WORDS NEEDED:** 6

**COMMENTS:** All addresses are expressed in hexadecimal notation.

### SUBROUTINE: COSINE

**DESCRIPTION:** Calculates cosine of double-precision angle in \( R_0, R_1 \)

- Polynomial approximation used is:
  \[
  \cos(x) = 1 + x^2 \left( A_2 + x^2 \left( A_4 + x^2 \left( A_6 + x^2 (A_8 + x^2 (A_{10})\right)\right)\right)
  \]
- Where: \( A_2 = -0.4999999993 \quad A_4 = 0.04166666418 \quad A_6 = -0.0013883397 \quad A_8 = 0.0000247609 \quad A_{10} = -0.00000002605 \)
- For angles in the first quadrant
- Accuracy is 0.000000 0002(10)

**ENTRY POINT:** FCFC
**ENTRY CONDITIONS:** \( R_0, R_1 = X \)
**EXIT CONDITIONS:** \( R_0, R_1 = \cos(x) \quad R_2, R_3 \) altered
**EXECUTION TIME:** 1677R + 82W + 9285N

**TEMPORARY LOCATIONS USED:** 00E0–00EB, 00EE, 00EF
**SUBROUTINES CALLED:** DPCOMP, DPMULT, PEXPN
**STACK WORDS NEEDED:** 6

**COMMENTS:** All addresses are expressed in hexadecimal notation.
SUBROUTINE: ARCTANGENT

DESCRIPTION: Computes arctangent (C1/C2), where C1 and C2 are double-precision numbers in the range:

\[-1 \leq C1, C2 \leq 1\]

The polynomial approximation used was:

\[\text{ARCTAN}(X) = X (1 + X^2 (A_2 + X^2 (A_4 + X^2 (A_6 + X^2 (A_8 + X^2 (A_{10} + X^2 (A_{12} + X^2 (A_{14} + X^2 (A_{16})))))))))\]

WHERE

\[
\begin{align*}
A_2 &= -0.33333 & A_{10} &= -0.07528 & 96400 \\
A_4 &= 0.19933 & A_{12} &= 0.04290 & 96138 \\
A_6 &= -0.14208 & A_{14} &= -0.01616 & 57367 \\
A_8 &= 0.10656 & A_{16} &= 0.00286 & 62257 \\
\end{align*}
\]

Accuracy is 0.00000 002(10)

ENTRY POINT: FD37
ENTRY CONDITIONS: R0, R1 = C1  R2, R3 = C2
EXIT CONDITIONS: R0, R1 = ARCTAN(C1/C2)  R2, R3 ALTERED
EXECUTION TIME: 2985R + 231W + 15892N

TEMPORARY LOCATIONS USED: 00E0-00EF
SUBROUTINES CALLED: QUAD, DPCOMP, DPDIV, DPMUL, PEXP, DSHR
STACK WORDS NEEDED: 7
(MAXIMUM DEPTH)

COMMENTS: 1. All addresses are expressed in hexadecimal notation.
2. Singularity case where C1 = C2 = 0 yields \(\text{ARCTAN}(C1/C2) = 0\).

SUBROUTINE: FLOATING POINT ADD

DESCRIPTION: Adds two floating-point numbers.

ENTRY POINT: FC30
ENTRY CONDITIONS: R0, R1 = X  R2, R3 = Y
EXIT CONDITIONS: R0, R1 = X + Y  R2, R3 ALTERED
EXECUTION TIME: 147R + 7W + 613N

TEMPORARY LOCATIONS USED: 00E0, 00E1, 00E2
SUBROUTINES CALLED: EXTEXP, DSHR, CZERO, ADDEXP
STACK WORDS NEEDED: 1
(MAXIMUM DEPTH)

COMMENTS: All addresses are expressed in hexadecimal notation.
### SUBROUTINE: FLOATING POINT MULTIPLY

**LABEL:** FPMUL

**DESCRIPTION:** MULTIPLIES TWO FLOATING-POINT NUMBERS

| ENTRY POINT: | FC5D |
| ENTRY CONDITIONS: | $R_0, R_1 = X$  $R_2, R_3 = Y$ |
| EXIT CONDITIONS: | $R_0, R_1 = X \times Y$  $R_2, R_3$ ALTERED |
| EXECUTION TIME: | $1215R + 63W + 5077N$ |

**TEMPORARY LOCATIONS USED:** 00E0, 00E1, 00E5, 00E6, 00EE, 00EF
**SUBROUTINES CALLED:** DLNORM, DPMUL, ADDEXP, CZERO
**STACK WORDS NEEDED:** 4  (MAXIMUM DEPTH)

**COMMENTS:** ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.

---

### SUBROUTINE: POSITIVE FLOATING POINT DIVIDE

**LABEL:** FPDIV

**DESCRIPTION:** DIVIDES TWO POSITIVE FLOATING-POINT NUMBERS

| ENTRY POINT: | FC6C |
| ENTRY CONDITIONS: | $R_0, R_1 = X$  $R_2, R_3 = Y$ |
| EXIT CONDITIONS: | $R_0, R_1 = X/Y$ |
| EXECUTION TIME: | $1540R + 152W + 6584N$ |

**TEMPORARY LOCATIONS USED:** 00E0, 00E1, 00E2, 00E5, 00E6, 00EC, 00ED, 00EE, 00EF
**SUBROUTINES CALLED:** DLNORM, DSHR, DPDIV, ADDEXP, CZERO
**STACK WORDS NEEDED:** 4  (MAXIMUM DEPTH)

**COMMENTS:** ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.
### SUBROUTINE: FLOATING POINT COMPLEMENT

**LABEL:** FPCOMP

**DESCRIPTION:** Complements a floating-point number in \( R_0', R_1 \)

| ENTRY POINT: | FC89 |
| ENTRY CONDITIONS: | \( R_0 = X_1 \), \( R_1 = X_2, \text{ EX} \) |
| EXIT CONDITIONS: | \( R_0 = X_1 \), \( R_1 = X_2, \text{ EX} \), \( R_2, R_3 \) PRESERVED |
| EXECUTION TIME: | \( 56R + 2W + 13N \) |

**TEMPORARY LOCATIONS USED:** 00E0

**SUBROUTINES CALLED:** EXTEXP, DPCOMP, ADDEXP

**STACK WORDS NEEDED:** 1

**COMMENTS:** All addresses are expressed in hexadecimal notation.

### SUBROUTINE: CHECK ZERO EXPONENT

**LABEL:** CZERO

**DESCRIPTION:** If mantissa of a floating-point number is zero, exponent is forced to zero.

| ENTRY POINT: | FC63 |
| ENTRY CONDITIONS: | \( R_0, R_1 = X_1, X_2 \) (EX) |
| EXIT CONDITIONS: | IF \( X_1, X_2 = 0 \) \( R_0 = 0, R_1 = 0,0(0) \) \( R_2, R_3 \) PRESERVED |
| IF \( X_1, X_2 \neq 0 \) \( R_0, R_2 = X_1, X_2 \) (EX) |
| EXECUTION TIME: | \( 36R + 2W + 13N \) |

**TEMPORARY LOCATIONS USED:** 00E0

**SUBROUTINES CALLED:** EXTEXP, ADDEXP

**STACK WORDS NEEDED:** 1

**COMMENTS:** All addresses are expressed in hexadecimal notation.
# SUBROUTINE: EXTRACT EXPONENT TO STACK

**DESCRIPTION:** EXTRACTS EXPONENT FROM A FLOATING-POINT NUMBER, STORES EXPONENT ON STACK, AND CLEARS EXPONENT FIELD OF NUMBER.

<table>
<thead>
<tr>
<th>ENTRY POINT</th>
<th>FC7C</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRY CONDITIONS:</td>
<td>R₀, R₁ = X₁, X₂ (EX)</td>
</tr>
<tr>
<td>EXIT CONDITIONS:</td>
<td>R₀, R₁ = X₁, X₂ (0) STACK = EX R₂, R₃ PRESERVED</td>
</tr>
<tr>
<td>EXECUTION TIME:</td>
<td>13R + W + 51N</td>
</tr>
</tbody>
</table>

**TEMPORARY LOCATIONS USED:** 00E0
**SUBROUTINES CALLED:** NONE
**STACK WORDS NEEDED:** 0 (MAXIMUM DEPTH)

**COMMENTS:** ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.

---

# SUBROUTINE: ADD EXPONENT FROM STACK

**DESCRIPTION:** AN EXPONENT IN THE STACK IS PLACED IN A FLOATING-POINT WORD IN R₀, R₁.

```
R₀

X₁
BEFORE

R₁

X₂  X₃

STACK

EX  NEXT

X₁  X₂  EX
AFTER

NEXT₁  NEXT₂
```

<table>
<thead>
<tr>
<th>ENTRY POINT</th>
<th>FCB9D</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENTRY CONDITIONS:</td>
<td>R₀, R₁ = X₁, X₂ (X₃) STACK = EX</td>
</tr>
<tr>
<td>EXIT CONDITIONS:</td>
<td>R₀, R₁ = X₁, X₂ (EX) R₂, R₃ PRESERVED</td>
</tr>
<tr>
<td>EXECUTION TIME:</td>
<td>13R + W + 51N</td>
</tr>
</tbody>
</table>

**TEMPORARY LOCATIONS USED:** N 00E0
**SUBROUTINES CALLED:** NONE
**STACK WORDS NEEDED:** 1 FOR INPUT EXPONENT ONLY (MAXIMUM DEPTH)

**COMMENTS:**
1. ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.
2. **WARNING:** ACCESS THIS SUBROUTINE ONLY BY A JSR INSTRUCTION!
### SUBROUTINE: LEFT NORMALIZE  
#### LABEL: LF NOR

**DESCRIPTION:** LEFT NORMALIZES A FLOATING-POINT NUMBER IN $R_0$, $R_1$

| ENTRY POINT: | FC97 |
| ENTRY CONDITIONS: | $R_0$, $R_1 = \times$ |
| EXIT CONDITIONS: | $R_0$, $R_1 = \times$ NORMA LIZED ;$R_2$, $R_3$ ARE PRESERVED |
| EXECUTION TIME: | $449R + 24W + 1745N$ |

**TEMPORARY LOCATIONS USED:** 00E0, 00E1  
**SUBROUTINES CALLED:** CZERO, FPCOMP, EXTEXP, DSHL, ADDEXP  
**STACK WORDS NEEDED:** 2  
(MAXIMUM DEPTH)

**COMMENTS:** ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.

### SUBROUTINE: DOUBLE LEFT NORMALIZE  
#### LABEL: DLNORM

**DESCRIPTION:** NORMALIZES TWO FLOATING-POINT NUMBERS $R_0$, $R_1$ AND $R_2$, $R_3$, WITH THE RESULTANTS SWITCHED (IN REGISTERS $R_2$, $R_3$ AND $R_0$, $R_1$, RESPECTIVELY). NEW EXONENTS ARE IN MEMORY.

| ENTRY POINT: | FC8A |
| ENTRY CONDITIONS: | $R_0$, $R_1 = X_1$, $X_2$ (EX)  
$R_2$, $R_3 = Y_1$, $Y_2$ (EY) |
| EXIT CONDITIONS: | $R_0$, $R_1 = Y$ NORMA LIZED (EX = 0)  
$R_2$, $R_3 = X$ NORMA LIZED (EX = 0)  
00E6 = EY  
00E5 = EX |
| EXECUTION TIME: | $947R + 52W + 3691N$ |

**TEMPORARY LOCATIONS USED:** 00E0-00E2, 00E5, 00E6  
**SUBROUTINES CALLED:** LF NOR, EXTEXP  
**STACK WORDS NEEDED:** 3  
(MAXIMUM DEPTH)

**COMMENTS:** ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.
**SUBROUTINE: FRACTIONAL TO FLOATING POINT**

**DESCRIPTION:** CONVERTS DOUBLE-PRECISION FRACTIONAL NUMBER TO FLOATING-POINT FORMAT BY TRUNCATING LEAST SIGNIFICANT 8 BITS.

**ENTRY POINT:** FC85
**ENTRY CONDITIONS:** $R_0, R_1 = X_1, X_2$
**EXIT CONDITIONS:** $R_0, R_1 = X_1, X_2' (0)$  $R_2, R_3$ PRESERVED
**EXECUTION TIME:** $3R + 9N$

**TEMPORARY LOCATIONS USED:** NONE
**SUBROUTINES CALLED:** NONE
**STACK WORDS NEEDED:** NONE (MAXIMUM DEPTH)

**COMMENTS:** ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.

---

**SUBROUTINE: FLOATING POINT TO FRACTIONAL**

**DESCRIPTION:** CONVERTS A FLOATING-POINT NUMBER TO A FRACTIONAL DOUBLE-PRECISION NUMBER

![Diagram: Floating Point to Fractional]

**ENTRY POINT:** FCC7
**ENTRY CONDITIONS:** $R_0, R_1 = X_1, X_2 (EX)$
**EXIT CONDITIONS:** $R_0, R_1 = X$ DOUBLE PREC.  $R_2, R_3$ PRESERVED
**EXECUTION TIME:** $207R + W + 822N$

**TEMPORARY LOCATIONS USED:** 00E0
**SUBROUTINES CALLED:** EXTEXP, DSH
**STACK WORDS NEEDED:** 2 (MAXIMUM DEPTH)

**COMMENTS:** ALL ADDRESSES ARE EXPRESSED IN HEXADECIMAL NOTATION.
CAUTION: VALUE OF INPUT MUST BE $< 1.0$ IN MAGNITUDE.
National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
(408) 732-5000
TWX: 910-359-9240

National Semiconductor (UK) Ltd.
Larkfield Industrial Estate
Greenock, Scotland
Telephone: GOOROCK 33251
Telex: 778 652

NS Electronics SDN BHD
Batu Berendam
Free Trade Zone
Malacca, Malaysia
Telephone: 5171
Telex: NSELECT 519 MALACCA (c/o Kuala Lumpur)

NS Electronics (PTE) Ltd.
No. 110 Lower Delta Rd.
Singapore 3
Telephone: 630011
Telex: NATSEMI RS 21402

REGIONAL AND DISTRICT SALES OFFICES

ALABAMA
DIXIE REGIONAL OFFICE
3222 Memorial Parkway, S.W. #57
Huntsville, Alabama 35802
(205) 881-0622
TWX: 810-720-2207

ARIZONA
*ROCKY MOUNTAIN REGIONAL OFFICE
7349 Sixth Avenue
Scottsdale, Arizona 85251
(602) 945-5473
TWX: 910-830-1195

CALIFORNIA
*NORTH-WEST REGIONAL OFFICE
2680 Bayshore Frontage Road, Suite 112
Mountain View, California 94043
(415) 851-1874
TWX: 910-379-6432
NATIONAL SEMICONDUCTOR
*LOS ANGELES REGIONAL OFFICE
Valley Freeway Center Building
15300 Ventura Boulevard, Suite 205
Sherman Oaks, California 91403
(213) 763-8727
TWX: 910-495-1773
NATIONAL SEMICONDUCTOR
*SOUTHERN CALIFORNIA REGIONAL OFFICE
17422 Irvine Boulevard, Suite M
Tustin, California 92680
(714) 852-9113
TWX: 910-995-1523

CONNECTICUT
DISTRICT OFFICE
Commerce Park
Danbury, Connecticut 06810
(203) 744-2390

*MID-ATLANTIC REGIONAL SALES OFFICE
25 Sylvan Road South
Westport, Connecticut 06880
(203) 229-6033

INTERNATIONAL SALES OFFICES

AUSTRALIA
NS ELECTRONICS PTY, LTD.
Cnr. Stud Road & Mountain Highway
Bayswater, Victoria 3153
Australia
Telephone: 03-729-6333
Telex: 32096

BELGIUM
NATIONAL SEMICONDUCTOR BELGIUM
789 Ave. Houta de Strooper
1620 Brussels
Telephone: 02-476-3400
Telex: 61 007 NatSem B

CANADA
NATIONAL SEMICONDUCTOR
DISTRICT OFFICE
1111 Finch Avenue West
Suite 154
Downsview, Ontario
M3J 2E5
(416) 430-5751
TWX: 610-492-1337

DENMARK
NATIONAL SEMICONDUCTOR DENMARK
Nyhan 69
1581 Copenhagen
Telephone: (1) 153110
Telex: 160 39

*Microprocessor System Specialist Available

FLORIDA
*CARIBBEAN REGIONAL OFFICE
2721 South Bayshore Drive, Suite 121
Miami, Florida 33133
(305) 446-3309
TWX: 810-848-9725

ILLINOIS
NATIONAL SEMICONDUCTOR
WEST-CENTRAL REGIONAL OFFICE
600 E. Northwest Highway, Suite 203
Mt. Prospect, Illinois 60056
(708) 594-0840
TWX: 910-689-3346

INDIANA
NATIONAL SEMICONDUCTOR
NORTH-CENTRAL REGIONAL OFFICE
P.O. Box 40073
Indianapolis, Indiana 46240
(317) 255-5822

MARYLAND
CAPITAL REGIONAL SALES OFFICE
300 Hospital Drive, No. 232
Glen Burnie, Maryland 21061
(301) 760-5220
TWX: 710-961-0519

MASSACHUSETTS
*NORTH-EAST REGIONAL OFFICE
17 Wallis St.
Lexington, Massachusetts 02173
Telephone: (617) 861-6090
TWX: 710-332-0166

MICHIGAN
*REGIONAL OFFICE
32629 Liberty Street
Farmington, Michigan 48024
(313) 477-0400
TWX: 810-242-2902

ENGLAND
NATIONAL SEMICONDUCTOR (UK) LTD.
The Precinct
Broxbourne, Hertfordshire
EN 107 HY
England
Telephone: Hoddesdon 69571
Telex: 267-204

FRANCE
NATIONAL SEMICONDUCTOR FRANCE
EXPANSION 10000
28 rue de la Redoute
92-250 Fontenay Aux Roses
Telephone: 660.01.40
Telex: NSF 29565F-

GERMANY
NATIONAL SEMICONDUCTOR GmbH
8000 Munich 81
Cosmosstr. 4/1
Telephone: 089/916027
Telex: 05-22772

HONG KONG
NS ELECTRONICS (HONG KONG) Ltd.
11th Floor
4 Hing Yip Street
Kwun Tong
Kowloon, Hong Kong
Telephone: 3-411241-8
Telex: 73866 NSE '4K' HK
Cable: NATSEMII

ITALY
NATIONAL SEMICONDUCTOR ITALY
Via Valassina 24
20159 Milano
Telephone: (02) 688 4617
Telex: 36-540

JAPAN
*NATIONAL SEMICONDUCTOR JAPAN
Nakawas Building
1-19 Yodoyu, Shintoku-Ku 160
Tokyo, Japan
Telephone: 03-359-4671
Telex: J 28592

SWEDEN
NATIONAL SEMICONDUCTOR SWEDEN
Sivtogen 17
13500 Tyreso-Stockholm
Telephone: 08/87 1294 80
Telex: 11263

TAIWAN
NS ELECTRONICS (HK) LTD.
TAINAN LIASON OFFICE
602 Teh Hwe Street
P.O. Box 68-332
Taipei, Taiwan ROC
Telephone: 633326
Cable: NSTW TAIPEI

MINNESOTA
DISTRICT SALES OFFICE
8053 Bloomington Freeway, Suite 101
Minneapolis, Minnesota 55420
(612) 888-3046
Telex: 290-766

NEW JERSEY
DISTRICT SALES OFFICE
14 Sylvan Avenue
Englewood Cliffs, New Jersey 07632
(201) 461-6599
TWX: 710-991-9734

NEW YORK
CAN-AM REGIONAL SALES OFFICE
104 Pickard Drive
Syracuse, New York 13211
(315) 455-5825

OHIO
DISTRICT SALES OFFICE
Financial South Building
5035 Far Hills, Suite 214
Dayton, Ohio 45429
(513) 434-0097
TWX: 810-459-1615

TEXAS
*SOUTH-CENTRAL REGIONAL OFFICE
5025 Forest Lane, Suite 260
Dallas, Texas 75230
(214) 233-8801
TWX: 910-980-2501

WASHINGTON
DISTRICT OFFICE
300 12th Avenue N.E.
Building 2, Suite 205
Bellevue, Washington 98005
(206) 454-4600

NATIONAL SEMICONDUCTOR CORP. PRINTED IN U.S.A.