http://www.6502.org/source/floats/wozfp1.txt 31 October 2004

TABLE OF CONTENTS

Floating Point Routines for the 6502 by Roy Rankin and Steve Wozniak

Originally published in the August 1976 issue of Dr. Dobb's Journal, these floating point routines allow 6502 users to perform most of the more popular and desired floating point and transcendental functions, namely: Natural Log, Common Log, Addition, Subtraction, Multiplication, Division, and conversions between floating and fixed point numbers.

Errata for Rankin's 6502 Floating Point Routines by Roy Rankin

In the November/December issue of Dr. Dobb's Journal Roy Rankin published three error corrections to the Floating Point Routines presented above.

Floating Point Implementation in the Apple II by Steve Wozniak

An almost identical set of the above routines appeared in the original manual for the Apple II (the Red Book, January 1978). Documentation for these routines appeared in another book, the Wozpak II, in November 1979.

\_\_\_\_\_

Floating Point Routines for the 6502 by Roy Rankin and Steve Wozniak

Originally published in the August 1976 issue of Dr. Dobb's Journal, these floating point routines allow 6502 users to perform most of the more popular and desired floating point and transcendental functions, namely: Natural Log, Common Log, Addition, Subtraction, Multiplication, Division, and conversions between floating and fixed point numbers.

Dr. Dobb's Journal, August 1976, pages 17-19.

Floating Point Routines for the 6502

by Roy Rankin, Department of Mechanical Engineering, Stanford University, Stanford, CA 94305 (415) 497-1822

and

Steve Wozniak, Apple Computer Company 770 Welch Road, Suite 154 Palo Alto, CA 94304 (415) 326-4248

Editor's Note: Although these routines are for the 6502, it would appear that one could generate equivalent routines for most of the "traditional" microprocessors, relatively easily, by following the flow of the algorithms given in the excellent comments included in the program listing. This is particularly true of the transcendental functions, which were directly modeled after well-known and proven algorithms, and for which, the comments are relatively machine independent. These floating point routines allow 6502 users to perform most of the more popular and desired floating point and transcendental functions, namely:

Natural Log - LOG Common Log - LOG10 Exponential - EXP Floating Add - FADD Floating Subtract - FSUB Floating Multiply - FMUL Floating Divide - FDIV Convert Floating to Fixed - FIX Convert Fixed to Floating - FLOAT

They presume a four-byte floating point operand consisting of a one-byte exponent ranging from -128 to +127 and a 24-bit two's complement mantissa between 1.0 and 2.0.

The floating point routines were done by Steve Wozniak, one of the principals in Apple Computer Company. The transcendental functions were patterned after those offered by Hewlett-Packard for their HP2100 minicomputer (with some modifications), and were done by Roy Rankin, a Ph. D. student at Stanford University.

There are three error traps; two for overflow, and one for prohibited logarithm argument. ERROR (1D06) is the error exit used in the event of a non-positive log argument. OVFLW (1E3B) is the error exit for overflow occuring during calculation of e to some power. OVFL (1FE4) is the error exit for overflow in all of the floating point routines. There is no trap for underflow; in such cases, the result is set to 0.0.

All routines are called and exited in a uniform manner: The arguments(s) are placed in the specified floating point storage locations (for specifics, see the documentation preceeding each routine in the listing), then a JSR is used to enter the desired routine. Upon normal completion, the called routine is exited via a subroutine return instruction (RTS).

Note: The preceeding documentation was written by the Editor, based on phone conversations with Roy and studying the listing. There is a high probability that it is correct. However, since it was not written nor reviewed by the authors of these routines, the preceeding documentation may contain errors in concept or in detail.

-- JCW, Jr.

In the Exponent: 00 Represents - 128 ... 7F Represents - 1 80 Represents 0 81 Represents +1 ... FF Represents +127

SPECEPTER SM AMAMAM MAMAM MAMAMAM MAMAMAMAM n n+1 n+2 n+3 JULY 5. 1976 $BASIC FIGATING POINT ROUTINES FOR 6502 MICROPROCESSOR BY R. RANKIN AND S. WOZNIAK CONSISTING OF: NATURAL LOG COMMON LOG EXPONENTIAL (E*X) FLOAT FIX FADD FSUB FOUL FOL FAUL FOL FA$	Exp	onent	t T	'wo's Co	omplement M	Mantissa
n n+1 n+2 n+3 JULY 5. 1976 BASIC FLOATING POINT ROUTINES FOR 6502 MIC ROPPOCESSOR BY R. RANKIN AND S. WOZNIAK CONSISTING OF: NATURAL LOC COMMON LOG EXPONENT LI. ( $E^{+}X$ ) FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT LI. ( $E^{+}X$ ) FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTISSA FYTES 2-4 MANTISSA TWO'S COMPLIMENT EXPERSENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT REPRESENTATION THE RANCE 1. TO 2. EXCEPT WHEN THE NUMBER IS LESS THAN 2**(-128). EXPONENT: THE ST HE EXPONENT REPRESENT SPONESS OF TWO. THE REPRESENTATION IS 2 S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENT REPORTS POWERS OF TWO. THE REPRESENTATION S POWER IS INCE THEY ARE STORED IN INCREASING MUMBER IS LESS THAN 2**(-128). EXPONENT: THE EXPONENT FOR SIZE SINCE THEY ARE STORED IN INCREASING MUMBER CAL SEQUENCE EANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS BESS INCE THEY ARE STORED IN INCREASING MUMBERICAL SEQUENCE MANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS AND GO (-128) TO SFF (+127) (S MEANS NUMBER IS AND GO (-128) TO SFF (+127) (S MEANS NUMBER IS AND FOR SIZE POINT RUPRESENTATION ALLOWS DECIMAL NUMBERS: THE PRESENT FLOATING POINT RUPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT RUPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT RUPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT RUPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT RUPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT RUPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE REPRESENTATION OF DECI	SEE	EEEEI	E SM.	MMMMMM	MMMMMMM	MMMMMM
<ul> <li>JULY 5. 1976         BASIC FLOATING POINT ROUTINES         FOR 6502 MICROPROCESSOR         BY R. RANKIN AND S. WOZNIAK         CONSISTING OF:             NATURAL LOG         COMMON LOG         EXPONENTIAL (E**X)         FLOATING POINT REPRESENTATION (4-BYTES)         EXPONENT BYTE 1         MANTISSA BYTES 2-4         MANTISSA STREESENTATION WUTH SIGN IN         MSB OF HIGH-ORDER BYTE 1         MANTISSA STREESENTATION WUTH SIGN IN         MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN         ASSUMED DECIMAL POINT BETTE. MANTISSA IS NORMALIZED WITH AN         ASSUMED DECIMAL POINT BETTE. MANTISSA IS NORMALIZED WITH AN         ASSUMED DECIMAL POINT BETTE. MANTISSA IS NORMALIZED WITH AN         ASSUMED DECIMAL POINT BETTE. MANTISSA IS NORMALIZED WITH AN         ASSUMED DECIMAL POINT BETTE. MANTISSA IS NORMALIZED WITH AN         ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF TIME INCEPT         WHEN THE NUMBER IS LESS THAN 2**(-128).         EXPONENT IN THE XEVENT HAIT HE SIGN         BYTE. THUS THE MANTI SSA IS IN THE RANCE I. TO 2. EXCEPT         WHEN THE NUMBER IS LESS THAN 2**(-128).         EXPONENT THE XEVENT THAT THE SIGN         BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DI RECT COMPARISON         OF EXPONENT SFOR SIZE SINCE THEY ARE STORED IN INCREASING         NUMER IS LESS THAN 2**(-128).         EXPONENT FOR SIZE SINCE THEY ARE STORED IN INCREASING         NUMER IS LESS THAN 2**(-128).         EXPONENT FOR SIZE SINCE THEY ARE STORED IN INCREASING         NUMER CAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127)         (S MEAN NUMBER IS LESS THAN 2**(-128).         EXPONENT FOR SIZE SINCE THEY ARE STORED IN INCREASING         NUMER IS LESS THAN 2**(-128).         EXPONENT FOR SIZE SINCE THEY ARE STORED IN INCREASING         NUMER IS ALLOWS DECIMAL NUMBERS. THE PRESENT FLOATING         POINT REPRESENTATION OF DECIMAL NUMBERS. IN THE APPROXIMATE         RANGE OF ID**(-38) THROUGH 10**(-138) WITH 6 TO 7 SIGNIFICANT         DIGITS.         FORESENTATION OF DECIMAL NUMBERS IN THE APPROXIMATE</li></ul>		n		n+1	n+2	n+3
<ul> <li>JULY 5. 1976         BASI C FLOATING POINT ROUTINES         POR 6502 MI CROPROCESSOR         BY R. RANKIN AND S. WOZNIAK         CONSISTING OF:         NATURAL LOG         COMMON LOG         EXYONENTIAL (E**X)         FLOATI FLX         FADD FSUB         FMUL FDIV         FLOATING POINT REPRESENTATION (4- BYTES)         EXYONENTIAL (E**X)         FLOATING POINT REPRESENTATION (4- BYTES)         EXYONENT BYTE 1         MANTISSA: TWO'S COMPLIMENT EXPRESENTATION WITH SIGN IN         MSB OF HIGH-ORDER BYTE 1         MANTISSA BYTES 2-4         MANTISSA: TWO'S COMPLIMENT EXPRESENTATION WITH SIGN IN         MSB OF HIGH-ORDER BYTE 1         MANTISSA IS IN THE RANCE 1. TO 2. EXCEPT         WHEN THE NUMBER IS LESS THAN 2**(-128).         EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE         REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE APPROXIMATE         BUT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN         OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING         OF EXPONENT EXCEPT THAT THE SIGN         FOONT REPRESENTATION ALLOWERS: THE PRESENT FLOATING         POINT REPRESENTATION OF DECL MAL NUMBERS IN THE APPROXIMATE         RANCE OF 10^+(-38) THROUGH 10^+(-38) WITH 6 TO 7 SIGNIFICANT         DIG TS.         COR 3 SET BASE PAGE ADRESSES         ONO 00 00 00 00 MI ESS 3 MANTISSA 1</li></ul>						
BASIC FLÖATING FOI NT ROUTH NES FOR 6502 MI CROPROCESSOR FOR 8502 MI CROPROCESSOR FOR 8.502 MI CROPROCESSOR CONSISTING OF: NATURAL LOG COMMON LOG (E*X) FLOATING POINT REPRESENTATION (4-BYTES) EXPONENTIAL (E*X) FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTISSA BYTES 2-4 MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MASE OF HIGH-ORDER PYTE MANTISSA IS NORMALLZED WITH AN ASSUMED DECLMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS NORMALLZED WITH AN ASSUMED DECLMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS NORMALLZED WITH AN ASSUMED DECLMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS NORMALLZED WITH AN ASSUMED DECLMAL POINT REPRESENTATION THE SALOS DI RECT COMPARISON WHEN THE NUMBER IS LESS THAN 2**(-128). WHEN THE NUMBER IS LESS THAN 2**(-128). WHEN THE NUMBER IS LESS THAN 2**(-128). WHEN STRICTION IS 2'S COMPLIMENTE EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTE. THE SALONG DI RECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMBERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL). * REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL VIMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGTS. * OOOS ON 00 00 MI BSS 3 MANTISSA 2 MANTISSA 2 MOP EXPONENT 2 MOR SIGN MANTISSA 1 MOR SIGN MANT SSA 1 MOR SIGN MANTISSA 1 MOR SIGN MANTISSA 1 MOR SIGN MANTISSA 1 MOR SIGN MANTISSA 1 MOR SEX 4 MATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * * * * * * * * * *				*	ШЛ	Y 5 1976
FOR 6502 MICROPEOCESSOR         BY R. RANKIN AND S. WOZNIAK         BY R. RANKIN AND S. WOZNIAK         CONSISTING OF:         NATURAL LOG         COMMON LOG         EXPONENTIAL (F**X)         FLOATI FIX         FADD PSUB         FMUL       FDIV         FLOATING POINT REPRESENTATION (4-BYTES)         EXPONENTIAL         EXPONENTIAL         FLOATING POINT REPRESENTATION (4-BYTES)         EXPONENTIAL         FLOATING POINT REPRESENTATION (4-BYTES)         EXPONENTIAL         FLOATING POINT REPRESENTATION (4-BYTES)         EXPONENTIAL         WANTISSA:         TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN         MANTISSA:         TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN         MANTISSA:         TWENT RETURNED         EXPONENT:         THE EXPONENT REPRESENTATION WITH SIGN IN         MEENTIC THE NUMBER IS LESS THAN 2**(-128).         ************************************				*	BASIC FLOA	ATING POINT ROUTINES
BY R. RANKIN AND S. WOZNIAK         CONSISTING OF:         MATURAL LOG         COMMON LOC         COMMON LOC         EXPONENTIAL (E**X)         FLOAT FIX         FADD FSUB         FMUL FDIV         FUNIT FIX         FADD FSUB         FMUL FDIV         FUNIT FIX         FADD FSUB         FMUL FOLV         FUNIT FIX         FADD FSUB         FMUL FOLV         FUNIT FIX         FLOATING POINT REPRESENTATION (4-BYTES)         EXPONENT BYTE 1         MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN         MSB OF INCHORMALIZED WITH AN         ASSUMED DECIMAL POINT REPRESENTATION WITH SIGN IN         MSS OF THUS THE MANTISSA IS IN THE RANCE I. TO 2. EXCEPT         WHEN THE NUMBER IS LESS THAN 2**(-128).         EXPONENT: THE EXPONENT REPRESENTATION IN CERTOR OPONER         BIT (BIT 7) IS COMPLIMENT REPOND OF INTE SIGN IN INCREASING         NUMERICAL SEQUENCE RANCING FROM 500 (-128) TO SFF (+127)         ('S MEANS NUMBER IS HANDECIMAL)         WHEN NERSENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING         POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE         RANCE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT				*	FOR 6502	2 MI CROPROCESSOR
CONSISTING OF: NATURAL LOG COMMON LOG EPPONENTIAL (E**X) FLOAT FIX FADD FSUB FMUL FDIV FLOATH FIX FADD FSUB FMUL FDIV FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTISSA BYTES 2-4 MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT BETPRESENT SIN MORE OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS IN THE RANCE 1. TO 2. EXCEPT WIEN THE VUMBER IS LESS THAN 2**(-128). EXPONENTS THE SIZE SINCE THEY ARE STORED IN INCERSION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCERSATION 0 F EXPONENTS FOR SIZE SINCE THEY ARE STORED IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT 0 IG TTS. 0 F EXPONENT 1 0 IG TO 0 MIN BSS 3 0 MANTISSA 1 0 IG 0 INT BSS 1 1 IDO 0 INT BSS 1 1 IDO 0 INT BSS 1 1 IDO 0 INT REST AND 0 IG 0 INT BSS 1 1 IDO 0 INT BSS 1 1 IDO 1				*	BY R. RA	ANKIN AND S. WOZNIAK
<ul> <li>CONSISTING OF: NATURAL LOG COMMON LOG EXPONENTIAL (E**X) FLOATI FIX FADD FSUB FMUL FDIV FAUL FDIV FUST FAUL FDIV FUST FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTISSA EYTES 2-4 MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MANTISSA: THO'S COMPLIMENT REPRESENTATION WITH SIGN IN MANTISSA: SINCE TREPRESENTATION WITH SIGN IN MANTISSA: SINCE TREPRESENTATION WITH SIGN IN MANTISSA: SINCE TREPRESENTATION WITH AN MASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS IN THE RANGE 1. TO 2. EXCEPT WHEN THE NUMBER: SI LESS THAN 2**(-128). EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL). REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANCE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. RANCE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT BOIG SIGNO 00 00 MMZ BSS 3 MANTISSA 1 DIGITS. RANCE OF 10*(-38) MANTISSA 1 DOO 00 00 MMZ BSS 4 SEXP MESS 4 SIGN NOP CORG SIDOO STARTING LOCATION FOR LOG REPRESENTATION DECEMAL PUT HITH RESULT IN MANT/EXP1 REPRESENTERDENCHINENT ALLOG</li></ul>				*		
<ul> <li>NATURAL LOG</li> <li>COMMON LOG</li> <li>EXPONENTIAL (E**X)</li> <li>FLOAT FIX</li> <li>FADD FSUB</li> <li>FMUL FDLV</li> <li>FLOATI NG POINT REPRESENTATION (4-BYTES)</li> <li>EXPONENT BYTE 1</li> <li>MANTISSA BYTES 2-4</li> <li>MANTISSA TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN</li> <li>MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN</li> <li>ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER</li> <li>BYTE. THUS THE MANTISSA IS NORMALIZED WITH AN</li> <li>ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER</li> <li>BYTE. THUS THE MANTISSA IS IN THE RANGE 1. TO 2. EXCEPT</li> <li>WHEN THE NUMBER IS LESS THAN 2**(-128).</li> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE</li> <li>REPRESENTATION IS 2'S COMPLIMENTE EXCEPT THAT THE SIGN</li> <li>OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING</li> <li>NUMERI CAL SEQUENCE RANGING FROM \$00 (-128) TO SFF (+127)</li> <li>(S MEANS NUMBER IS HEXADECIMAL).</li> <li>REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING</li> <li>POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE</li> <li>RANCE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT</li> <li>DIGITS.</li> <li>ORG 3 SET BASE PAGE ADRESSES</li> <li>MOP</li> <li>EXPONENT 1</li> <li>ONOP</li> <li>EXEST AS A SCRATCH</li> <li>ONOP</li> <li>C BSS 4</li> <li>SCRATCH</li> <li>ONOP</li> <li>C C BSS 4</li> <li>SCRATCH</li> <li>ONOP</li> <li>C RG SIDOO STARTING LOCATION FOR LOG</li> <li>NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1</li> <li>IDOO A5 09</li> <li>LOG</li> <li>LOA MI</li> </ul>				*	CONSI STI NO	G OF:
* COMMON LOG EXPONENTIAL (F**X) FLOAT FIX FADD FSUB FAUL FDIV * FADD FSUB FAUL FDIV * FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 * MANTISSA BYTES 2-4 * MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN * MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS IN OWALLIZED. IT 0 2. EXCEPT * WHEN THE NUMBER IS LESS THAN 2**(-128). * EXPONENT: THE EXPONENT REPRESENTATION, THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN * OF EXPONENTS FOR SIZE SINCE THEYA RE STORED IN INCREASING * UNWERL CAL SEQUENCE FANGING FROM SOO (-128) TO SFF (+127) * (S MEANS NUMBER IS HEXADECIMAL). * REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING * DINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE 0F 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. * 0003 ORG 3 SET BASE PAGE ADRESSES 0003 EA SIGN NOP EXPONENT IS ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE * RANGE 0F 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. * 0004 EA X2 NOP EXPONENT 1 0005 OU 00 00 MZ ESS 3 MANTISSA 1 0006 EA X1 NOP EXPONENT 1 0007 EB SS 4 SCRATCH 0008 ON 00 00 MI BSS 3 MANTISSA 1 1000 ON ON MI BSS 1 * 0008 SEA X1 NOP EXPONENT 1 1000 ORG \$1000 STARTING LOCATION FOR LOG * * 1000 A5 09 LOG LDA MI				*	NATURAL	L LOG
<ul> <li>EXPONENTIAL (E**X)</li> <li>FLOAT FIX</li> <li>FADD FSUB</li> <li>FADL FOIV</li> <li>FANL FOIV</li> <li>FANL FOIV</li> <li>FAUL FOIV</li> <li>FOOT IN CPOINT REPRESENTATION (4-BYTES)</li> <li>EXPONENT BYTE 1</li> <li>MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN</li> <li>MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN</li> <li>ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER</li> <li>BYTE. THUS THE MANTISSA IS IN THE RANGE 1. TO 2. EXCEPT</li> <li>WHEN THE NUMBER IS LESS THAN 2**(-128).</li> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE</li> <li>REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN</li> <li>BIT (BIT 7) IS COMPLIMENTEXCEPT THAT THE SIGN</li> <li>OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING</li> <li>NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127)</li> <li>(S MEANS NUMBER IS HEXADECIMAL).</li> <li>REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING</li> <li>POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE</li> <li>RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT</li> <li>DION AC AS C3 SET BASE PAGE ADRESSES</li> <li>MOME EA X2 NOP</li> <li>EXPONENT 1</li> <li>ORG 3 SET BASE PAGE ADRESSES</li> <li>MON BES 3 MANTISSA 1</li> <li>DOOS CO 00 00 M2 BSS 3 MANTISSA 1</li> <li>DOO AD 00 M2 BSS 3 MANTISSA 1</li> <li>ORG \$1DOO STARTING LOCATION FOR LOG</li> <li>*</li> <li>*</li> <li>NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1</li> <li>*</li> <li>NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1</li> <li>*</li> <li>*</li></ul>				*	COMMON	LOG
FLOAT FIX FADD FSUB FMUL FDIV FMUL FDIV FULL FULL FULL FULL FULL FULL FULL FUL				*	EXPONEN	NTI AL (E**X)
FADD FSUB FMUL FDIV FMUL FDIV FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTISSA STIES 2-4 MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS IN THE RANGE 1. TO 2. EXCEPT WHEN THE NUMBER IS LESS THAN 2*(-128). EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS OF TWO. THE REPRESENTATION SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL). REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIG ITS. COOS ORG 3 SET BASE PAGE ADRESSES NOOS EA SIGN NOP EXPONENT 1 CONS ON 00 00 MZ BSS 3 MANTISSA 2 DOOS OO 00 00 MZ BSS 3 MANTISSA 1 DOOM EA X2 NOP EXPONENT 1 DOOM AD SEY A OUT OF DEST ADATION CLOCATION FOR LOG * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * DOOM AD DO DE CARDING DECIMAL DIMENTION ALDOW FOR LOG * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * DODM AD 000 C BED ERROR				*	FLOAT	FIX
FMUL       FDLV         **       FLOATI NG POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTI SSA BYTES 2-4         **       MANTI SSA:       TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTI SSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE.         **       EXPONENT:       THE EXPONENT REPRESENTS POWERS OF TWO.         **       EXPONENT:       THE EXPONENT REPRESENTS POWERS OF TWO.         **       REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTED.       THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM S00 (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL).         **       REPRESENTATION OF DECIMAL NUMBERS:       THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUCH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS.         **       O003       ORG 3       SET BASE PAGE ADRESSES         0004       EA       X1       NOP         **       O004       EX       NANTI SSA 1         0005       OO 00 00       MI       BSS 3         0006       FA       X1       NOP         **       OOC       E       BSS 4         0001       T       BSS 1       **         0002       ORG \$1DO       STARTI NG LOCATI ON FOR LO				*	FADD	FSUB
FLOATI NG POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTI SSA BYTES 2-4 MANTI SSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTI SSA IS NORMALIZED WITH AN ASSILMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTI SSA IS IN THE RANGE 1. TO 2. EXCEPT WHEN THE NUMBER IS LESS THAN 2**(-128). ** EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DI RECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMBERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL). ** REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. ** 0003 00G 3 SET BASE PAGE ADRESSES 0003 EA SIGN NOP 004 EA X2 NOP EXPONENT 1 005 00 00 00 MI BSS 3 MANTISSA 1 0065 00 00 00 MI BSS 3 MANTISSA 1 0006 Z BSS 4 0016 Z BSS 4 0016 Z BSS 4 0017 Z BSS 4 0018 SEXP BSS 1 1000 ** 0RG SIDOO STARTING LOCATION FOR LOG ** NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 ** 1000 A5 09 LOG LDA MI				*	FMUL	FDI V
FLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS IN THE RANCE 1. TO 2. EXCEPT WHEN THE NUMBER IS LESS THAN 2**(-128). EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) ('S MEANS NUMBER IS HEXADECIMAL). REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANCE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. ORG 3 SET BASE PAGE ADRESSES OW M2 BSS 3 MANTISSA 2 0005 EA SIGN NOP EXPONENT 1 0006 EA XI NOP EXPONENT 1 0007 Z BSS 4 0014 T BSS 3 0018 SEXP BSS 4 0014 T BSS 1 1000 AS 09 LOG LDA MI 1000 AS 09 LOG LDA MI				*		
<ul> <li>PLOATING POINT REPRESENTATION (4-BYTES) EXPONENT BYTE 1 MANTISSA BYTES 2-4</li> <li>MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTISSA IS IN THE RANGE 1. TO 2. EXCEPT</li> <li>WHEN THE NUMBER IS LESS THAN 2**(-128).</li> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENTE EXCEPT THAT THE SIGN</li> <li>BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127)</li> <li>(S MEANS NUMBER IS HEXADECIMAL).</li> <li>REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION OF DECIMAL NUMBERS: IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS.</li> <li>REPRESENT AT ON ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS.</li> <li>THOO NO MI BSS 3 MANTISSA 2 0003 CRG 3 SET BASE PAGE ADRESSES</li> <li>MOD 2 BSS 4 0014 T BSS 1 0014 T BSS 3 MANTISSA 1 0005 00 00 MI SEX 4 SCRATCH 0014 T BSS 1 0016 ON INT BSS 1</li> <li>ORG \$IDOO STARTING LOCATION FOR LOG</li> <li>NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1</li> <li>NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1</li> </ul>				*		
CAPONENT DITE 1     MANTI SSA BYTES 2-4     MANTI SSA BYTES 2-4     MANTI SSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN     MSB OF HIGH-ORDER BYTE. MANTI SSA IS NORMALIZED WITH AN     ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER     BYTE. THUS THE MANTI SSA IS IN THE RANGE 1. TO 2. EXCEPT     WHEN THE NUMBER IS LESS THAN 2**(-128).     EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE     REPRESENTATION IS 2'S COMPLIMENTE EXCEPT THAT THE SIGN     BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIECT COMPARISON     OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING     NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127)     (S MEANS NUMBER IS HEXADECIMAL).     REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING     POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE     RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT     DIGITS.     T     OOO3     ORG 3 SET BASE PAGE ADRESSES     OOO3 EA SIGN NOP     EXPONENT 1     DOO 40 MU BSS 3 MANTI SSA 1     ONP EXPONENT 1     OOO3     ORG SIDOO STARTING LOCATION FOR LOG     *     NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1     *     NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1     *     NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1     *				*	FLUATING F	PUINI KEPKESENIAIIUN (4-BIIES)
MANTISSA DILES 2-4  * MANTISSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN  * MGB OF HIGH-ORDER BYTE. MANTISSA IS NORMALIZED WITH AN  ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER  BYTE. THUS THE MANTISSA IS IN THE RANGE 1. TO 2. EXCEPT  * WHEN THE NUMBER IS LESS THAN 2**(-128). *  EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING * NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) * (S MEANS NUMBER IS HEXADECIMAL). *  REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING * POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANCE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. **  O003 ORG 3 SET BASE PAGE ADRESSES O003 EA SIGN NOP CXPONENT 2 O005 00 00 0M 2 BSS 3 MANTISSA 2 ** ** O003 ORG 3 SET BASE PAGE ADRESSES O003 EA X1 NOP EXPONENT 1 0000 Z BSS 4 O010 Z BSS 4 O010 Z BSS 4 O010 Z BSS 4 O010 OX MI BSS 1 ** ** ** ** ** ** ** ** ** ** ** ** **				*		EAPUNENT DITE T MANTICSA DVTES 2 4
<ul> <li>MANTI SSA: TWO'S COMPLIMENT REPRESENTATION WITH SIGN IN MSB OF HIGH-ORDER BYTE. MANTI SSA IS NORMALIZED WITH AN ASSUMED DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTI SSA IS IN THE RANGE 1. TO 2. EXCEPT WHEN THE NUMBER IS LESS THAN 2**(-128).</li> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMBER CLAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL).</li> <li>REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS.</li> <li>ORG 3 SET BASE PAGE ADRESSES</li> <li>ONO3 ORG 3 SET BASE PAGE ADRESSES</li> <li>O004 EA X2 NOP EXPONENT 2 0005 00 00 MZ BSS 3 MANTISSA 2 0006 EA X1 NOP EXPONENT 1 0000 C E BSS 4 SCRATCH 0010 Z BSS 4 0010 Z BSS 3 MANTISSA 1 0000 C E BSS 4 SCRATCH 0011 T BSS 1 *</li> <li>ORG \$1D00 STARTING LOCATION FOR LOG **</li> <li>**</li> <li>ORG \$1D00 STARTING LOCATION FOR LOG</li> <li>*</li> <li>**</li> <li>**<td></td><td></td><td></td><td>*</td><td></td><td>MANIISSA DIIES 2-4</td></li></ul>				*		MANIISSA DIIES 2-4
<ul> <li>MINITOSA THE ORDER BYTE. MANTI SSA IS NORMALIZED WITH AN ASSUMED DECI MAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER BYTE. THUS THE MANTI SSA IS IN THE RANGE 1. TO 2. EXCEPT</li> <li>WHEN THE NUMBER IS LESS THAN 2**(-128).</li> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN</li> <li>BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRCEASING</li> <li>OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING</li> <li>NUMERI CAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127)</li> <li>(S MEANS NUMBER IS HEXADECI MAL).</li> <li>REPRESENTATION OF DECI MAL NUMBERS: THE PRESENT FLOATING</li> <li>POINT REPRESENTATION ALLOWS DECI MAL NUMBERS IN THE APPROXIMATE</li> <li>REPRESENTATION OF DECI MAL NUMBERS: IN THE APPROXIMATE</li> <li>REPRESENTATION OF DECI MAL NUMBERS IN THE APPROXIMATE</li> <li>REPRESENT EXPONENT 2</li> <li>OOO3</li> <li>ORG 3</li> <li>ORG SI MANTISSA 1</li> <li>OOO4 M2</li> <li>BSS 3</li> <li>MANTISSA 1</li> <li>OOO5 OO 00 M1</li> <li>BSS 3</li> <li>MANTISSA 1</li> <li>OOO6 M2</li> <li>DES 4</li> <li>SCAP BSS 4</li> <li>OOO1 Z BSS 4</li> <li>OOO3 OO 00 MI BSS 3</li> <li>MANTISSA 1</li> <li>OOO AND BSS 4</li> <li>OOO AND BSS 4</li> <li>OOO AND BSS 4</li> <li>OOO AS CA SIDOO STARTING LOCATION FOR LOG</li> <li>*</li> <li>*&lt;</li></ul>				*	MANTI SSA-	TWO'S COMPLIMENT REPRESENTATION WITH SICN IN
ASSUMED DECI MAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER * BYTE. THUS THE MANTI SSA IS IN THE RANGE 1. TO 2. EXCEPT * WHEN THE NUMBER IS LESS THAN 2**(-128). * EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) * (\$ MEANS NUMBER IS HEXADECIMAL). * REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. * ORG 3 SET BASE PAGE ADRESSES 0003 EA SIGN NOP 0004 EA X2 NOP EXPONENT 2 0005 00 00 00 M2 BSS 3 MANTISSA 2 0005 00 00 00 M1 BSS 3 MANTISSA 1 0000 EA X1 NOP EXPONENT 1 0000 00 00 M1 BSS 3 MANTISSA 1 0001 Z BSS 4 0014 T BSS 4 0014 SEXP ASS 4 0014 SEXP BSS 4 0014 T BSS 4 0016 O INT BSS 1 * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * DIGITS BS 0 0004 EA X2 NOP EXPONENT 1 1000 A5 09 LOG LDA MI				*	MSB OF F	HIGH-ORDER BYTE MANTISSA IS NORMALIZED WITH AN
<ul> <li>BYTE. THUS THE MANTI SSA IS IN THE RANGE 1. TO 2. EXCEPT         <ul> <li>WHEN THE NUMBER IS LESS THAN 2**(-128).</li> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE</li></ul></li></ul>				*	ASSUMED	DECIMAL POINT BETWEEN BITS 5 AND 6 OF THE HIGH-ORDER
<ul> <li>WHEN THE NUMBER IS LESS THAN 2**(-128).</li> <li>**</li> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLMENTED. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL).</li> <li>**</li> <li>REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS.</li> <li>*</li> <li>OCO3</li> <li>ORG 3</li> <li>SET BASE PAGE ADRESSES</li> <li>OO04 EA</li> <li>X2</li> <li>NOP</li> <li>EXPONENT 1</li> <li>OO05 OO 00 M2</li> <li>BSS 3</li> <li>MANTISSA 2</li> <li>OO06</li> <li>E BSS 4</li> <li>SCRATCH</li> <li>Z BSS 4</li> <li>OO14</li> <li>T BSS 4</li> <li>OO15</li> <li>OO INT</li> <li>BSS 1</li> <li>**</li> <li>ORG \$1D00</li> <li>STARTING LOCATION FOR LOG</li> <li>*</li> <li>*</li> <li>*</li> <li>ORG \$1D0 STARTING LOCATION FOR LOG</li> <li>*</li> <li>*</li> <li>*</li> <li>ORG \$1D0 STARTING LOCATION FOR LOG</li> <li>*</li> <li>*</li> <li>*</li> <li>OO2 60 02</li> </ul>				*	BYTE 1	THUS THE MANTISSA IS IN THE RANGE 1. TO 2. EXCEPT
<ul> <li>EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECIMAL).</li> <li>REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS.</li> <li>ORG 3 SET BASE PAGE ADRESSES</li> <li>O003 ORG 3 SET BASE PAGE ADRESSES</li> <li>O004 EA X2 NOP EXPONENT 2 0005 00 00 ML BSS 3 MANTISSA 2 0008 EA X1 NOP EXPONENT 1 0009 00 00 00 MI BSS 3 MANTISSA 1 0000 E BSS 4 SCRATCH 0010 Z BSS 4 0014 T BSS 4 0016 OI INT BSS 1</li> <li>ORG \$1D00 STARTING LOCATION FOR LOG</li> <li>**</li> <li>**<td></td><td></td><td></td><td>*</td><td>WHEN THE</td><td>E NUMBER IS LESS THAN 2**(-128).</td></li></ul>				*	WHEN THE	E NUMBER IS LESS THAN 2**(-128).
<ul> <li>* EXPONENT: THE EXPONENT REPRESENTS POWERS OF TWO. THE REPRESENTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN BIT (BIT 7) IS COMPLIMENTE D. THIS ALLOWS DIRECT COMPARISON OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127) (S MEANS NUMBER IS HEXADECI MAL). * REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. * 0003</li> <li>ORG 3 SET BASE PAGE ADRESSES 0003 EA SIGN NOP EXPONENT 2 0005 00 00 00 M2 BSS 3 MANTISSA 2 0008 EA X1 NOP EXPONENT 1 0000 Z BSS 4 0010 Z BSS 4 0010 Z BSS 4 0011 T BSS 1 1000</li> <li>ORG \$1DOO STARTING LOCATION FOR LOG * * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * 1000 A5 09 LOG LDA MI 1002 FO 02</li> </ul>				*		
<ul> <li>* REPRESENTATI ON 1S 2'S COMPLIMENT EXCEPT THAT THE SIGN</li> <li>* BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON</li> <li>* OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING</li> <li>* NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127)</li> <li>* (\$ MEANS NUMBER IS HEXADECIMAL).</li> <li>* REPRESENTATI ON OF DECIMAL NUMBERS: THE PRESENT FLOATING</li> <li>* POINT REPRESENTATI ON ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE</li> <li>* RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT</li> <li>* DIGITS.</li> <li>*</li> <li>*</li> <li>*</li> <li>ORG 3 SET BASE PAGE ADRESSES</li> <li>OO03 ORG 3 SET BASE PAGE ADRESSES</li> <li>O004 EA X2 NOP EXPONENT 2</li> <li>O005 00 00 M2 BSS 3 MANTISSA 2</li> <li>O006 EA X1 NOP EXPONENT 1</li> <li>O008 EA X1 NOP EXPONENT 1</li> <li>O000 C E BSS 4 SCRATCH</li> <li>O010 Z BSS 4</li> <li>O014 T BSS 4</li> <li>O014 T BSS 4</li> <li>O014 T BSS 1</li> <li>*</li> <li>1000 A5 09 LOG LDA M1</li> <li>ID00 A5 09 LOG LDA M1</li> </ul>				*	EXPONENT:	THE EXPONENT REPRESENTS POWERS OF TWO. THE
<ul> <li>BIT (BIT 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON</li></ul>				*	REPRESEN	NTATION IS 2'S COMPLIMENT EXCEPT THAT THE SIGN
<ul> <li>* OF EXPONENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING</li> <li>* NUMERICAL SEQUENCE RANGING FROM SOO (-128) TO SFF (+127)</li> <li>* (S MEANS NUMBER IS HEXADECIMAL).</li> <li>* REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING</li> <li>* POINT REPRESENTATION OF DECIMAL NUMBERS: IN THE APPROXIMATE</li> <li>* RANCE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT</li> <li>* DIGITS.</li> <li>*</li> <li>*</li> <li>ORG 3 SET BASE PAGE ADRESSES</li> <li>OO03 OR G 3 SET BASE PAGE ADRESSES</li> <li>O004 EA X2 NOP EXPONENT 2</li> <li>O005 O0 00 M2 BSS 3 MANTISSA 2</li> <li>O005 O0 00 M1 BSS 3 MANTISSA 1</li> <li>O000 E BSS 4 SCRATCH</li> <li>O010 Z BSS 4</li> <li>O014 T BSS 4</li> <li>O014 T BSS 1</li> <li>*</li> <li>*</li> <li>ORG \$1DOO STARTING LOCATION FOR LOG</li> <li>*</li> <li>*</li> <li>*</li> <li>ORG \$1DOO STARTING LOCATION FOR LOG</li> <li>*</li> <li>*</li> <li>*</li> <li>ORG \$1DOO STARTING LOCATION FOR LOG</li> <li>*</li> <li>*</li> <li>*</li> <li>DO0 A5 09 LOG LDA M1</li> <li>D02 F0 02</li> </ul>				*	BIT (BIT	Γ 7) IS COMPLIMENTED. THIS ALLOWS DIRECT COMPARISON
<ul> <li>NUMERI CAL SEQUENCE RANGI NG FROM SOO (-128) TO SFF (+127)</li> <li>(\$ MEANS NUMBER I S HEXADECI MAL).</li> <li>REPRESENTATI ON OF DECI MAL NUMBERS: THE PRESENT FLOATI NG</li> <li>POI NT REPRESENTATI ON ALLOWS DECI MAL NUMBERS IN THE APPROXIMATE</li> <li>RANGE OF 10**(-38) THROUGH 10**(38) WI TH 6 TO 7 SI GNI FI CANT</li> <li>DI GI TS.</li> <li>ORG 3 SET BASE PAGE ADRESSES</li> <li>OO03 ORG 3 SET BASE PAGE ADRESSES</li> <li>OO04 EA X2 NOP EXPONENT 2</li> <li>O005 00 00 M2 BSS 3 MANTI SSA 2</li> <li>O006 EA X1 NOP EXPONENT 1</li> <li>O0000 EA X1 NOP EXPONENT 1</li> <li>O0000 E E BSS 4 SCRATCH</li> <li>O011 Z BSS 4</li> <li>O012 OO INT BSS 1</li> <li>T BSS 4</li> <li>O012 OO INT BSS 1</li> <li>NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1</li> <li>*</li> <li>*</li> <li>*</li> <li>ID00 A5 09 LOG LDA M1</li> <li>ID02 F0 02</li> </ul>				*	OF EXPON	NENTS FOR SIZE SINCE THEY ARE STORED IN INCREASING
* (\$ MEANS NUMBER IS HEXADECIMAL). * REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING * POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT * DIGITS. * * 0003 ORG 3 SET BASE PAGE ADRESSES 0003 EA SIGN NOP 0004 EA X2 NOP EXPONENT 2 0005 00 00 0M2 BSS 3 MANTISSA 2 0008 EA X1 NOP EXPONENT 1 0009 00 00 00 MI BSS 3 MANTISSA 1 0000 E BSS 4 0010 Z BSS 4 0014 T BSS 4 0014 T BSS 4 0014 ORG \$1D00 STARTING LOCATION FOR LOG * * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * 1D00 A5 09 LOG LDA M1 1D02 F0 02 BEO ERROR				*	NUMERI CA	AL SEQUENCE RANGING FROM \$00 (-128) TO \$FF (+127)
<pre>* REPRESENTATI ON OF DECI MAL NUMBERS: THE PRESENT FLOATI NG POINT REPRESENTATI ON ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT DIGITS. * 0003 CR 3 SET BASE PAGE ADRESSES 0003 EA SIGN NOP 0004 EA X2 NOP EXPONENT 2 0005 00 00 00 M2 BSS 3 MANTI SSA 2 0008 EA X1 NOP EXPONENT 1 0009 00 00 00 M1 BSS 3 MANTI SSA 1 0000C E BSS 4 SCRATCH 0010 Z BSS 4 0014 T BSS 4 0014 T BSS 4 0016 00 INT BSS 1 * 1D00 ORG \$1D00 STARTING LOCATI ON FOR LOG * * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * 1D00 A5 09 LOG LDA M1 1D02 F0 02 BEQ ERROR</pre>				*	(\$ MEANS	S NUMBER IS HEXADECIMAL).
REPRESENTATION OF DECIMAL NUMBERS: THE PRESENT FLOATING POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE RANCE OF 10**(-38) THROUGH 10**(38) WI TH 6 TO 7 SI GNI FI CANT DI GI TS. * 0003 ORG 3 SET BASE PAGE ADRESSES 0003 EA SI GN NOP 0004 EA X2 NOP EXPONENT 2 0005 00 00 00 M2 BSS 3 MANTI SSA 2 0008 EA X1 NOP EXPONENT 1 0009 00 00 00 M1 BSS 3 MANTI SSA 1 0000C E BSS 4 SCRATCH 0010 Z BSS 4 0014 T BSS 4 0014 T BSS 4 0014 SEXP BSS 4 0016 00 INT BSS 1 * 1D00 ORG \$1D00 STARTI NG LOCATI ON FOR LOG * * NATURAL LOG OF MANT/EXP1 WI TH RESULT I N MANT/EXP1 * 1D00 A5 09 LOG LDA MI 1D02 F0 02 BEQ ERROR				*		
<ul> <li>POINT REPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE</li> <li>RANGE OF 10**(-38) THROUGH 10**(38) WITH 6 TO 7 SIGNIFICANT</li> <li>DIGITS.</li> <li>0003</li> <li>ORG 3</li> <li>SET BASE PAGE ADRESSES</li> <li>0004 EA</li> <li>X2</li> <li>NOP</li> <li>EXPONENT 2</li> <li>0005 00 00 00</li> <li>M2</li> <li>BSS 3</li> <li>MANTI SSA 2</li> <li>0008 EA</li> <li>X1</li> <li>NOP</li> <li>EXPONENT 1</li> <li>0009</li> <li>00 00 00</li> <li>M1</li> <li>BSS 3</li> <li>MANTI SSA 1</li> <li>0000</li> <li>C</li> <li>E</li> <li>BSS 4</li> <li>0014</li> <li>T</li> <li>BSS 4</li> <li>0014</li> <li>T</li> <li>BSS 4</li> <li>0014</li> <li>ORG \$1D00</li> <li>STARTI NG LOCATI ON FOR LOG</li> <li>*</li> <li>*</li> <li>NATURAL LOG OF MANT/EXP1</li> <li>WITH RESULT IN MANT/EXP1</li> <li>*</li> <li>1D00</li> <li>A5 09</li> <li>LOG</li> <li>LDA M1</li> <li>LOG</li> <li>LDA M1</li> <li>LOG</li> <li>LDA M1</li> <li>LOG</li> <li>LDA M1</li> </ul>				*	REPRESENTA	ATION OF DECIMAL NUMBERS: THE PRESENT FLOATING
*       DI GI TS.         *       DI GI TS.         *       *         0003       ORG 3       SET BASE PAGE ADRESSES         0004       EA       SI GN NOP         0004       EA       X2       NOP         0005       00 00 00       M2       BSS 3         0008       EA       X1       NOP         0009       00 00 00       M1       BSS 3         00000       E       BSS 4         00010       Z       BSS 4         0014       T       BSS 4         0010       Z       BSS 4         0011       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       *         1D00       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       *         1D00       A5 09       LOG         LDA M1       HO2       FO 02         BEO ERROR       EROR				*	POINT RE	EPRESENTATION ALLOWS DECIMAL NUMBERS IN THE APPROXIMATE
*       *         0003       ORG 3       SET BASE PAGE ADRESSES         0003       EA       SI GN       NOP         0004       EA       X2       NOP       EXPONENT 2         0005       00       00       M2       BSS 3       MANTI SSA 2         0008       EA       X1       NOP       EXPONENT 1         0009       00       00       MANTI SSA 1         00000       E       BSS 4       SCRATCH         0010       Z       BSS 4         0014       T       BSS 4         0010       Z       BSS 4         0011       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *         1D00       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *         1D00       A5 09       LOG         LOG       LDA M1         1D02       F0 02				*	RANGE OF	$F = 10^{++}(-38)$ IHROUGH $10^{++}(38)$ WITH 6 IU 7 SIGNIFICANT
*       ORG 3       SET BASE PAGE ADRESSES         0003       EA       SIGN       NOP         0004       EA       X2       NOP       EXPONENT 2         0005       O0 00 00       M2       BSS 3       MANTI SSA 2         0008       EA       X1       NOP       EXPONENT 1         0009       O0 00 00       M1       BSS 3       MANTI SSA 1         0000C       E       BSS 4       SCRATCH         0010       Z       BSS 4       SCRATCH         0014       T       BSS 4         0016       O       INT       BSS 1         *       NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1				*	DI GI 15.	
0003       EA       SI GN       NOP         0004       EA       X2       NOP       EXPONENT 2         0005       00 00       M2       BSS 3       MANTI SSA 2         0008       EA       X1       NOP       EXPONENT 1         0009       00 00       M1       BSS 3       MANTI SSA 1         0000       C       E       BSS 4       SCRATCH         0010       Z       BSS 4       SCRATCH         0014       T       BSS 4       SCRATCH         0016       O       INT       BSS 4         0016       O       INT       BSS 1         *       *       NATURAL LOG       OF MANT/EXP1 WITH RESULT IN MANT/EXP1         *       NATURAL LOG       OF MANT/EXP1 WITH RESULT IN MANT/EXP1         *       NATURAL LOG       OF MANT/EXP1 WITH RESULT IN MANT/EXP1         *       NATURAL LOG       OF MANT/EXP1 WITH RESULT IN MANT/EXP1				*		
00003       EA       SI CN       NOP       EXPONENT 2         0004       EA       X2       NOP       EXPONENT 2         0005       00 00 00       M2       BSS 3       MANTI SSA 2         0008       EA       X1       NOP       EXPONENT 1         0009       00 00 00       M1       BSS 3       MANTI SSA 1         0000C       E       BSS 4       SCRATCH         0010       Z       BSS 4         0014       T       BSS 4         0018       SEXP       BSS 1         *        ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       NATURAL LOG OF MANT/EXP1       WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1       WI TH RESULT IN MANT/EXP1         *       *       NATURAL LOG OF MANT/EXP1       WI TH RESULT IN MANT/EXP1         *       *       BEQ ERROR       BEQ ERROR	0003				ORG 3	SET BASE PACE ADRESSES
0004       EA       X2       NOP       EXPONENT 2         0005       00 00 00       M2       BSS 3       MANTI SSA 2         0008       EA       X1       NOP       EXPONENT 1         0009       00 00 00       M1       BSS 3       MANTI SSA 1         0000       E       BSS 4       SCRATCH         0010       Z       BSS 4         0014       T       BSS 4         0018       SEXP       BSS 4         0012       00       INT       BSS 1         *       *       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1	0003	EA		SI GN	NOP	
0005       00       00       M2       BSS 3       MANTI SSA 2         0008       EA       X1       NOP       EXPONENT 1         0009       00       00       M1       BSS 3       MANTI SSA 1         0000       E       BSS 4       SCRATCH         0010       Z       BSS 4       SCRATCH         0010       Z       BSS 4       SCRATCH         0014       T       BSS 4         0015       OO       INT       BSS 1         *       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1	0004	EA		X2	NOP	EXPONENT 2
0008       EA       X1       NOP       EXPONENT 1         0009       00       00       M1       BSS 3       MANTI SSA 1         0000       E       BSS 4       SCRATCH         0010       Z       BSS 4       SCRATCH         0014       T       BSS 4         0015       SEXP       BSS 4         0016       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         1D00       A5 09       LOG       LDA M1         1D02       F0 02       BEQ ERROR	0005	00 (	00 00	M2	BSS 3	MANTI SSA 2
0009       00       00       M1       BSS 3       MANTI SSA 1         0000       E       BSS 4       SCRATCH         0010       Z       BSS 4         0014       T       BSS 4         0018       SEXP       BSS 1         1000       INT       BSS 1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         11000       A5 09       LOG         LOG       LDA M1         11002       F0 02	8000	EA		X1	NOP	EXPONENT 1
000C       E       BSS 4       SCRATCH         0010       Z       BSS 4         0014       T       BSS 4         0018       SEXP       BSS 1         001C       00       INT       BSS 1         *       0RG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       NATURAL LOG OF MANT/EXP1         1D00       A5 09       LOG       LDA M1         1D02       F0 02       BEQ ERROR	0009	00 (	00 00	M1	BSS 3	MANTI SSA 1
0010       Z       BSS 4         0014       T       BSS 4         0018       SEXP       BSS 4         001C       00       INT       BSS 1         *       0RG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       NATURAL LOG OF MANT/EXP1         1D00       A5 09       LOG       LDA M1         1D02       F0 02       BEQ ERROR	000C			E	BSS 4	SCRATCH
0014       T       BSS 4         0018       SEXP       BSS 4         001C       00       INT       BSS 1         *       0RG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       *         1D00       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       NATURAL LOG OF MANT/EXP1 WI TH RESULT IN MANT/EXP1         *       BEQ ERROR	0010			Z	BSS 4	
0018       SEXP       BSS 4         001C       00       I NT       BSS 1         1D00       ORG \$1D00       STARTI NG LOCATI ON FOR LOG         *       *       *         *       NATURAL LOG OF MANT/EXP1       WI TH RESULT IN MANT/EXP1         *       *       NATURAL LOG OF MANT/EXP1         1D00       A5 09       LOG       LDA M1         1D02       F0 02       BEQ ERROR	0014			Т	BSS 4	
001C 00 INT BSS 1 * ORG \$1D00 STARTING LOCATION FOR LOG * * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 1D00 A5 09 LOG LDA M1 1D02 F0 02 BEQ ERROR	0018			SEXP	BSS 4	
1D00 ORG \$1D00 STARTING LOCATION FOR LOG * * * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * 1D00 A5 09 LOG LDA M1 1D02 F0 02 BEQ ERROR	001C	00		INT	BSS 1	
1DOO ORG \$1DOO STARTING LOCATION FOR LOG * * * NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * 1DOO A5 09 LOG LDA M1 1DO2 F0 02 BEQ ERROR	1000			*		
* NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * 1D00 A5 09 LOG LDA M1 1D02 F0 02 BEQ ERROR	1D00			ale	ORG \$1DOC	D STARTING LOCATION FOR LOG
* NATURAL LOG OF MANT/EXP1 WITH RESULT IN MANT/EXP1 * 1D00 A5 09 LOG LDA M1 1D02 F0 02 BEQ ERROR				*		
1D00 A5 09 LOG LDA M1 1D02 F0 02 BEQ ERROR				*		
1D00 A5 09 LOG LDA M1 1D02 F0 02 BEQ ERROR				*	NATURAL LU	JG UF WANT/EAFT WITH RESULT IN MANT/EAFT
1D02 F0 02 BEQ ERROR	1 D O O	A5 (	19	LOG	LDA M1	
	1D02	F0 (	02	200	BEQ ERROF	R

1D04 1D06	10 01 00	ERROR	BPL CONT BRK	I F ARG>0 OK ERROR ARG<=0
1D07 1D0A	20 1C 1F A5 04	CONT	JSR SWAP LDA X2	MOVE ARG TO EXP/MANT2 HOLD EXPONENT
1DOC 1DOE	AU 80 84 04		LDY = 380 STY X2	SET EXPONENT 2 TO 0 (\$80)
1D10	49 80		EOR =\$80	COMPLIMENT SIGN BIT OF ORIGINAL EXPONENT
1D12	85 OA		STA M1+1	SET EXPONENT INTO MANTISSA 1 FOR FLOAT
1D14 1D16	A9 00 85 09		LDA =0 STA M1	CLEAR MSB OF MANTESSA 1
1D18	20 2C 1F		JSR FLOAT	CONVERT TO FLOATING POINT
1D1B	A2 03		LDX =3	4 BYTE TRANSFERS
1D1D	B5 04	SEXP1	LDA X2, X STA Z V	CODV MANTI SSA TO 7
1D11 1D21	B5 08		LDA X1. X	COFT MANTISSA TO Z
1D23	95 18		STA SEXP, X	SAVE EXPONENT IN SEXP
1D25	BD D1 1D		LDA R22, X	LOAD EXP/MANT1 WITH SQRT(2)
1D28 1D24	95 08 CA		SIA XI, X DEX	
1D2B	10 F0		BPL SEXP1	
1D2D	20 4A 1F		JSR FSUB	Z-SQRT(2)
1D30	A2 03	CAVET	LDX = 3	4 BYTE TRANSFER
1D32 1D34	95 14	SAVEI	STATX	SAVE EXP/MANII AS I
1D36	B5 10		LDA Z, X	LOAD EXP/MANT1 WITH Z
1D38	95 08		STA X1, X	
1D3A 1D3D	BD DI ID 95 04		LDA R22, X STA X2 X	LUAD EXP/MANIZ WITH SQRI(2)
1D3F	CA CA		DEX	
1D40	10 F0		BPL SAVET	
1D42	20 50 1F		JSR FADD	Z+SQRT(2)
1D45 1D47	AZ U3 B5 14	TM2	LDX = 3 LDA T X	4 BYTE TRANSFER
1D49	95 04	11112	STA X2, X	LOAD T INTO EXP/MANT2
1D4B	CA		DEX	
1D4C	10 F9 20 0D 1E		BPL TM2	$T_{-}(7, SOPT(2)) / (7, SOPT(2))$
1D4E 1D51	A2 03		LDX = 3	4  BYTE TRANSFER
1D53	B5 08	MI T	LDA X1, X	
1D55	95 14		STA T, X	COPY EXP/MANT1 TO T AND
1D57 1D59	95 04 CA		SIA XZ, X DFX	LUAD EXP/MANIZ WITH I
1D5A	10 F7		BPL MIT	
1D5C	20 77 1F		JSR FMUL	T*T
1D5F 1D62	20 1C 1F		JSR SWAP	MOVE T*T TO EXP/MANT2 4 RVTE TRANSFER
1D62 1D64	BD E1 1D	MI C	LDA C, X	4 DITE TRANSPER
1D67	95 08		STA X1, X	LOAD EXP/MANT1 WITH C
1D69	CA		DEX	
1D6A 1D6C	10 F8 20 4A 1F		ISR FSUB	Т*Т-С
1D6F	A2 03		LDX = 3	4 BYTE TRANSFER
1D71	BD DD 1D	M2MB	LDA MB, X	
1D74 1D76	95 04 CA		STA X2, X	LOAD EXP/MANT2 WITH MB
1D70 1D77	10 F8		BPL M2MB	
1D79	20 9D 1F		JSR FDIV	MB/(T*T-C)
1D7C	A2 03	MOA 1	LDX = 3	
1D7E 1D81	во оч ID 95 04	M∠A1	LDA AI, X STA X2 X	LOAD EXP/MANT2 WITH A1
1D83	CA		DEX	Long Lan, Mentrix III III
1D84	10 F8		BPL M2A1	
1D86	20 50 1F		JSR FADD	$MB/(T^*T-C) + A1$

1D89	A2 03			LDX	=3	4 BYTE TRANSFER
1D8B	B5 14		M2T	LDA	Т, Х	
1D8D	95 04			STA	X2, X	LOAD EXP/MANT2 WITH T
1D8F	CA			DEX	MOT	
1D90	10 F9	1 17		BPL	M21	
1D92	20 77	ΙF		JSR	FMUL	(MB/(1*1-C) + A1)*1
1D95	A2 03	4.5	1 (0) (0) 1	LDX	=3	4 BYTE TRANSFER
1097	BD E5	ID	MZMHL	LDA	MHLF, X	
1D9A	95 04			SIA	X2, X	LUAD EXP/MANIZ WITH MHLF (.5)
1090				DEX	MOMIT	
1000	10 50	1 17		DPL	MAMHL	. ศ
1D9F 1DA9	20 00 A2 02	ΙГ		JOK	ГАДД _2	+. J A DVTE TDANCEED
1 DAL	AZ 03		I DEXD		SEAD A	4 DITE TRANSFER
1DA6	95 04		LDLAI	STA	$X^2 X$	LOAD FXP/MANT2 WITH ORIGINAL FXPONENT
1DA8	CA			DEX	<i>n</i> ≈, <i>n</i>	
1DA9	10 F9			BPL	LDEXP	
1DAB	20 50	1F		JSR	FADD	+EXPN
1DAE	A2 03			LDX	=3	4 BYTE TRANSFER
1DBO	BD D5	1D	MLE2	LDA	LE2, X	
1DB3	95 04			STA	X2, X	LOAD EXP/MANT2 WITH LN(2)
1DB5	CA			DEX		
1DB6	10 F8			BPL	MLE2	
1DB8	20 77	1F		JSR	FMUL	*LN(2)
1DBB	60			RTS		RETURN RESULT IN MANT/EXP1
			*			
			*	COMM	ON LOG O	F MANT/EXP1 RESULT IN MANT/EXP1
1000		4.5	*	IGD	100	
IDBC	20 00	ID	LOGIO	JSR	LOG	COMPUTE NATURAL LOG
1 DC1	AZ U3	1 D	T 10		=3	
		ID	LIU		LNIU, X	$I \cap AD = EVD (MANTO WITH 1 (IN(10))$
1DC4 1DC6	95 04 CA			DEV	<b>λ</b> <i>λ</i> , <b>λ</b>	LUAD EXP/MANIZ WITH I/LN(IU)
1DC0				DEA	T 10	
	20 77	1 F		DF L ISD	ENU	$I \cap (1 \cap (Y) - I \cap (Y) / I \cap (1 \cap (Y))$
1DC3	20 77 60	11		RTS	TIMOL	L0010(X) - LN(X) / LN(10)
1000	00		*	1010		
1DCD	7E 6F		LN10	DCM	0. 434294	45
	2D ED					
1DD1	80 5A		R22	DCM	1.414213	36 SQRT(2)
	02 7A					
1DD5	7F 58		LE2	DCM	0. 69314'	718 LOG BASE E OF 2
	B9 OC			5 61 6		
TDD9	80 52		AI	DCM	1. 29200	/4
1000	80 40 91 AD		MD	DCM	2 6200	577
עעעד	81 AD		MD	DCM	- 2. 0398	
1DF1	80 49 80 64		C	DCM	1 656765	26
IDEI	00 0A 08 66		C	DCM	1. 05070/	20
1DE5	7F 40		MHLF	DCM	0.5	
1020	00 00			2011	010	
			*			
1E00				ORG	\$1E00	STARTING LOCATION FOR EXP
			*			
			*	EXP (	OF MANT/I	EXP1 RESULT IN MANT/EXP1
			*			
1E00	A2 03	. –	EXP	LDX	=3	4 BYTE TRANSFER
1E02	RD D8	ΙE			LZE, X	
1EU5	95 U4			SIA	λλ, λ	LUAD EXP/MANIZ WITH LUG BASE Z UF E
1EU/ 1E00	UA 10 E9			עדע זיסק	EVD 9	
	10 Fð 20 77	1 F		DLT DL	llar+2 FMIII	$I \cap (2) $ *Y
1 EOA	20 // Δ2 Ω2	1 I.		1 DX	-3	A BYTE TRANSFER
1EOF	B5 08		FSA	L.DA	X1. X	
			1 011			

1E11	95 10		STA Z, X	STORE EXP/MANT1 IN Z
1E13	CA		DEX	
1E14	10 F9		BPL FSA	SAVE $Z=LN(2) *X$
1E10	20 E8 IF		JSK FIX	CUNVERI CUNIENIS OF EXP/MANII IO AN INTEGER
1E19 1E1D	A5 UA 85 1C		LDA MI+I STA INT	CAVE DECHIT AS INT
	38		SEC	SAVE RESULT AS INT SET CARRY FOR SURTRACTION
1F1F	F9 7C		SBC = 124	INT-124
1E1D	A5 09		LDA M1	
1E22	E9 00		SBC = 0	
1E24	10 15		BPL OVFLW	OVERFLOW INT>=124
1E26	18		CLC	CLEAR CARRY FOR ADD
1E27	A5 OA		LDA M1+1	
1E29	69 78		ADC =120	ADD 120 TO INT
1E2B	A5 09		LDA M1	
1E2D	69 00		ADC =0	
1E2F 1E21	10 0B		BPL CONTIN	IF KESULI PUSIIIVE CUNIINUE
1631	A9 00		LDA = 0 LDY = 3	A RATE MOVE
1E35 1E35	AZ 03 95 08	ZEBO	STA X1 X	SFT FXP/MANT1 TO ZERO
1E37	CA		DEX	
1E38	10 FB		BPL ZERO	
1E3A	60		RTS	RETURN
		*		
1E3B	00	OVFLW *	BRK	OVERFLOW
1E3C	20 2C 1F	CONTI N	ISR FLOAT	FLOAT INT
1E3F	A2 03	00001110	LDX = 3	
1E41	B5 10	ENTD	LDA Z, X	
1E43	95 04		STA X2, X	LOAD EXP/MANT2 WITH Z
1E45	CA		DEX	
1E46	10 F9		BPL ENTD	
1E48	20 4A 1F		JSR FSUB	Z*Z-FLOAT(INT)
1E4B	A2 03	ROLL	LDX =3	4 BYTE MOVE
IE4D	B5 08	ZSAV	LDA XI, X	
1E4F 1E51	95 10		SIA Z, X	SAVE EXP/MANTI IN Z CODV EVD/MANT1 TO EVD/MANT2
1651	95 04 CA		DEY	CUPI EAF/MANII IU EAF/MANIZ
1E55 1F54	10 F7		BPI ZSAV	
1E56	20 77 1F		JSR FMUL	7.*7
1E59	A2 03		LDX = 3	4 BYTE MOVE
1E5B	BD DC 1E	LA2	LDA A2, X	
1E5E	95 04		STA X2, X	LOAD EXP/MANT2 WITH A2
1E60	B5 08		LDA X1, X	
1E62	95 18		STA SEXP, X	SAVE EXP/MANT1 AS SEXP
1E64	CA		DEX	
1E65	10 F4		BPL LAZ	7*7.40
1E07 1E64	20 30 IF		JSK FADD	$L^{+}L+A\lambda$
1E0A 1E6C	$A^{2}_{RD} = 0.05$	I R2	LDA = 3 IDA B2 X	4 DILE MOVE
1E6C	95 04	LDL	STA X2 X	LOAD FXP/MANT2 WITH B2
1E71	CA		DEX	
1E72	10 F8		BPL LB2	
1E74	20 9D 1F		JSR FDIV	T=B/(Z*Z+A2)
1E77	A2 03		LDX =3	4 BYTE MOVE
1E79	B5 08	DLOAD	LDA X1, X	
1E7B	95 14		STA T, X	SAVE EXP/MANT1 AS T
1E7D	BD E4 1E		LDA C2, X	
1E80	95 U8		SIA XI, X	LUAD EXP/MANII WITH CZ
1E02 1E91	DJ 18 95 04		LDA SEAP, Ά STA X2 V	I OAD FYP/MANT? WITH SFYP
1E84 1F86	CA 55 04		DEX	LUAD LAI/IMANIA WI III SEAF
1E87	10 F0		BPL DLOAD	
1E89	20 77 1F		JSR FMUL	Z*Z*C2

1E8C	20 1C 1F	7	JSR SWAP	MOVE EXP/MANT1 TO EXP/MANT2
1E8F	A2 03		LDX =3	4 BYTE TRANSFER
1E91 1E02	B5 14	LIMP	LDA I, X	
1E95 1E95	95 U8		DEY	LUAD EXP/MANII WITH I
1E95 1F96	10 F9		BPI ITMP	
1E98	20 4A 1F	7	JSR FSUB	C2*7*7-B2/(7*7+A2)
1E9B	A2 03		LDX = 3	4 BYTE TRANSFER
1E9D	BD E8 1E	E LDD	LDA D, X	
1EAO	95 04		STA X2, X	LOAD EXP/MANT2 WITH D
1EA2	CA		DEX	
1EA3	10 F8	_	BPL LDD	
1EA5	20 50 1H	-' -	JSR FADD	D+C2*Z*Z-B2/(Z*Z+A2)
I EAB	20 10 16	1	JSK SWAP	MOVE EXP/MANII IO EXP/MANIZ
	AZ US B5 10	IΕΛ	LDX = 3 LDX 7 Y	4 DITE TRANSFER
1 EAD $1 F \Delta F$	95 08	LIA	STA X1 X	LOAD FXP/MANT1 WITH 7
1EB1	CA		DEX	
1EB2	10 F9		BPL LFA	
1EB4	20 4A 1F	7	JSR FSUB	- Z+D+C2*Z*Z-B2/(Z*Z+A2)
1EB7	A2 03		LDX =3	4 BYTE TRANSFER
1EB9	B5 10	LF3	LDA Z, X	
1EBB	95 04		STA X2, X	LOAD EXP/MANT2 WITH Z
1EBD	CA		DEX	
1 EBE	10 F9	7	BPL LF3	7/(****)
1EC0	20 9D IF	1	JSK FDIV	$L/(\uparrow\uparrow\uparrow\uparrow\uparrow)$
1ECS	AZ US BD E5 11	) ID12	LDA = 5 IDA MHIE X	4 DITE TRANSFER
1EC3	95 04	LD12	STA X2 X	LOAD FXP/MANT2 WITH 5
1ECA	CA		DEX	
1 ECB	10 F8		BPL LD12	
1 ECD	20 50 1F	7	JSR FADD	+Z/(***)+.5
1EDO	38		SEC	ADD INT TO EXPONENT WITH CARRY SET
1ED1	A5 1C		LDA INT	TO MULTIPLY BY
1ED3	65 08		ADC X1	$2^{**}(INT+1)$
1ED7	85 08		SIA XI	KEIUKN KESULI IU EXPUNENI DETUDN ANG $( f, 7/(7, 0, 0))$ (0, 7, 7, 0) (7, 7, 0) (7, 7, 0)
	60 80 5C	1.95	KIS DCM 1 44260	$\begin{array}{llllllllllllllllllllllllllllllllllll$
IEDO	60 SC 55 1F	LLE	DCM 1. 442093	JU409 LUG DASE 2 OF E
1 EDC	86 57	A2	DCM 87.41749	97202
1200	6A E1			
1 EEO	89 4D	B2	DCM 617.972	2695
	3F 1D			
1EE4	7B 46	C2	DCM . 0346573	35903
4.000	FA 70	5		
1EE8	83 4F	D	DCM 9.95459	57821
	A3 03	*		
		*		
		* ]	BASIC FLOATI	NG POLNT ROUTINES
		*		
1F00			ORG \$1F00	START OF BASIC FLOATING POINT ROUTINES
1F00	18	ADD	CLC	CLEAR CARRY
1F01	A2 02		LDX =\$02	INDEX FOR 3-BYTE ADD
1F03	B5 09	ADD1	LDA M1, X	
1F05	75 05		ADC M2, X	ADD A BYTE OF MANT2 TO MANT1
1F07 1F00	90 U9 CA		SIA MI, Å DEV	ADVANCE INDEX TO NEXT MORE SIGNIE DVTE
1F04	10 F7			IOOP INTII DONF
1FOC	60		RTS	RETURN
1FOD	06 03	MD 1	ASL SIGN	CLEAR LSB OF SIGN
1F0F	20 12 1F	7	JSR ABSWAP	ABS VAL OF MANT1, THEN SWAP MANT2
1F12	24 09	ABSWAP	BIT M1	MANT1 NEG?
1F14	10 05		BPL ABSWP1	NO, SWAP WITH MANT2 AND RETURN

1F16 1F19 1F1B	20 8F 1F E6 03 38	JSR FCOMPL INC SIGN ABSWP1 SEC	YES, COMPLIMENT IT. INCR SIGN, COMPLEMENTING LSB SET CARRY FOR RETURN TO MUL/DIV
		* SWAP EXP/MAN	IT1 WI TH EXP/MANT2
1F1C 1F1E 1F20 1F22 1F24 1F26 1F28 1F29 1F2B	A2 04 94 0B B5 07 B4 03 94 07 95 03 CA D0 F3 60	* SWAP LDX =\$04 SWAP1 STY E-1, X LDA X1-1, X LDY X2-1, X STY X1-1, X STA X2-1, X DEX BNE SWAP1 RTS *	INDEX FOR 4-BYTE SWAP. SWAP A BYTE OF EXP/MANT1 WITH EXP/MANT2 AND LEAVEA COPY OF MANT1 IN E(3BYTES). E+3 USED. ADVANCE INDEX TO NEXT BYTE LOOP UNTIL DONE.
		* CONVERT 16 E * RESULT IN EX *	BIT INTEGER IN M1(HIGH) AND M1+1(LOW) TO F.P. XP/MANT1. EXP/MANT2 UNEFFECTED
1F2C 1F2E 1F30 1F32	A9 8E 85 08 A9 00 85 0B	FLOAT LDA =\$8E STA X1 LDA =0 STA M1+2	SET EXPN TO 14 DEC CLEAR LOW ORDER BYTE
1F34 1F36 1F38 1F3A	F0 08 C6 08 O6 0B 26 0A	NORM1 DEC X1 ASL M1+2 ROL M1+1	NORMALIZE RESULT DECREMENT EXP1 SHIFT MANT1 (3 BYTES) LEFT
1F3C 1F3E 1F40 1F41	26 09 A5 09 OA 45 09	ROL M1 NORM LDA M1 ASL EOR M1	HI GH ORDER MANT1 BYTE UPPER TWO BITS UNEQUAL?
1F43 1F45 1F47 1F49	30 04 A5 08 D0 ED 60	BMI RTS1 LDA X1 BNE NORM1 RTS1 RTS	YES, RETURN WI TH MANT1 NORMALI ZED EXP1 ZERO? NO, CONTI NUE NORMALI ZI NG RETURN
		* * EXP/MANT2-EX *	XP/MANT1 RESULT IN EXP/MANT1
1F4A 1F4D	20 8F 1F 20 5D 1F	FSUB JSR FCOMPL SWPALG JSR ALGNSW *	CMPL MANT1 CLEARS CARRY UNLESS ZERO RIGHT SHIFT MANT1 OR SWAP WITH MANT2 ON CARRY
		* ADD EXP/MANT *	1 AND EXP/MANT2 RESULT IN EXP/MANT1
1F50 1F52 1F54 1F56	A5 04 C5 08 D0 F7 20 00 1F	FADD LDA X2 CMP X1 BNE SWPALG JSR ADD	COMPARE EXP1 WITH EXP2 IF UNEQUAL, SWAP ADDENDS OR ALIGN MANTISSAS ADD ALIGNED MANTISSAS
1F59 1F5B 1F5D 1F5F	50 E3 70 05 90 BD A5 09	ADDEND BVC NORM BVS RTLOG ALGNSW BCC SWAP RTAR LDA M1	NO OVERFLOW, NORMALIZE RESULTS OV: SHIFT MANT1 RIGHT. NOTE CARRY IS CORRECT SIGN SWAP IF CARRY CLEAR, ELSE SHIFT RIGHT ARITH. SIGN OF MANT1 INTO CARRY FOR
1F61 1F62 1F64 1F66	0A E6 08 F0 7E A2 FA	ASL RTLOG I NC X1 BEQ OVFL RTLOG1 LDX =\$FA	RI GHT ARI TH SHI FT I NCR EXP1 TO COMPENSATE FOR RT SHI FT EXP1 OUT OF RANGE. I NDEX FOR 6 BYTE RI GHT SHI FT
1F68 1F6A 1F6C 1F6D	A9 80 B0 01 0A 56 0F	$\begin{array}{ccc} \text{ROR1} & \text{LDA} = \$80 \\ \text{BCS} & \text{ROR2} \\ \text{ASL} \\ \text{ROR2} & \text{LSR} & \texttt{F+3} & \texttt{X} \end{array}$	SIMILATE ROR F+3 X
1F6F	15 OF	ORA E+3, X	SIMOLITE IVIV L+0, A

1F71 1F73 1F74 1F76	95 OF E8 D0 F2 60		¥	STA E+3, X I NX BNE ROR1 RTS	NEXT BYTE OF SHIFT LOOP UNTIL DONE RETURN
			*		
			* I *	EXP/MANT1 X	EXP/MANT2 RESULT IN EXP/MANT1
1F77	20 OD 1	1 F	FMUL	JSR MD1	ABS. VAL OF MANT1, MANT2
1F7A	65 08	117		ADC X1	ADD EXP1 TO EXP2 FOR PRODUCT EXPONENT
1F7C 1F7F	20 CD 1 18	IF		CLC	CLEAR CARRY
1F80	20 66 1	1F .	MUL1	JSR RTLOG1	MANT1 AND E RIGHT. (PRODUCT AND MPLIER)
1F83	90 03			BCC MUL2	IF CARRY CLEAR, SKIP PARTIAL PRODUCT
1F85	20 00 1	1F		JSR ADD	ADD MULTI PLI CAN TO PRODUCT
1F88 1E90	88 10 E5		MUL2	DEY PDI MIII 1	NEXT MUL ITERATION
1F8B	10 F5 46 03		MDEND	LSR SIGN	TEST SIGN (EVEN/ODD)
1F8D	90 AF		NORMX	BCC NORM	I F EXEN, NORMALI ZE PRODUCT, ELSE COMPLEMENT
1F8F	38		FCOMPL	SEC	SET CARRY FOR SUBTRACT
1F90	A2 03			LDX =\$03	INDEX FOR 3 BYTE SUBTRACTION
1F92	A9 00		COMPL1	LDA = \$00	CLEAR A SUDTRACT DVTE OF EVD1
1F94 1F96	F5 08 95 08			SDU AI, A STA X1 X	RESTORE IT
1F98	CA			DEX	NEXT MORE SI GNI FI CANT BYTE
1F99	DO F7			BNE COMPL1	LOOP UNTIL DONE
1F9B	FO BC			BEQ ADDEND	NORMALIZE (OR SHIFT RIGHT IF OVERFLOW)
			*		
			* I	EXP/MANT2 / 1	EXP/MANT1 RESULT IN EXP/MANT1
1 F9D	20 OD 1	1F	FDI V	ISR MD1	TAKE ABS VAL OF MANT1 MANT2
1FAO	£5 08			SBC X1	SUBTRACT EXP1 FROM EXP2
1FA2	20 CD 1	1F		JSR MD2	SAVE AS QUOTIENT EXP
1FA5	38		DI V1	SEC	SET CARRY FOR SUBTRACT
1FA6	A2 02		סע זם	LDX = \$02	INDEX FOR 3-BYTE INSTRUCTION
1ΓΑΟ 1ΓΔΔ	БЗ 05 F5 0C		DIVL	SBC F X	SUBTRACT A BYTE OF F FROM MANT2
1FAC	48			PHA PHA	SAVE ON STACK
1 FAD	CA			DEX	NEXT MORE SIGNIF BYTE
1FAE	10 F8			BPL DIV2	LOOP UNTIL DONE
1FB0	A2 FD		סע זם	LDX =SFD	INDEX FOR 3-BYTE CONDITIONAL MOVE
1FBZ 1FB3	68 90 02		DI V3	PLA BCC DI VA	LE MANT2-E THEN DONT RESTORE MANT2
1FB5	95 02			STA M2+3. X	II' MANIZ IIIEN DONI RESIORE MANIZ
1FB7	E8		DI V4	I NX	NEXT LESS SI GNI F BYTE
1FB8	DO F8			BNE DI V3	LOOP UNTIL DONE
1FBA	26 OB			ROL M1+2	DOLL OHOTHENT LEFT CADDY INTO LOD
1FBC 1FBF	26 0A 26 09			ROL MI+I ROI MI	ROLL QUUITENT LEFT, CARRY INTO LSB
1FCO				ASL M2+2	
1FC2	26 06			ROL M2+1	SHI FT DI VI DEND LEFT
1FC4	26 05			ROL M2	
1FC6	BO 1C			BCS OVFL	OVERFLOW IS DUE TO UNNORMALIZED DIVISOR
1FC8 1FC9	δο ΠΟ ΠΛ			DEI BNF DIVI	NEAL DIVIDE LIERATION LOOP UNTLE DONE 23 LTERATIONS
1FCB	FO BE			BEQ MDEND	NORMALIZE QUOTIENT AND CORRECT SIGN
1 FCD	86 OB		MD2	STX M1+2	
1FCF	86 OA			STX M1+1	CLR MANT1 (3 BYTES) FOR MUL/DIV
1FD1	86 09 BO OD			STX M1	
1FD3 1FD5	30 04 RO OD			BUE OVCHK	IF EAP CALE SEI CARRY, CHECK FOR OVEL LE NEC NO UNDERFLOW
1FD7	50 04 68				POP ONE
1FD8	68			PLA	RETURN LEVEL

1FD9 1FDB	90 B2 49 80	MD3	BCC NORMX EOR =\$80	CLEAR X1 AND RETURN COMPLIMENT SIGN BIT OF EXP
1 FDD	85 08		STA X1	STORE IT
1FDF	AO 17		LDY =\$17	COUNT FOR 24 MUL OR 23 DIV ITERATIONS
1FE1	60		RTS	RETURN
1FE2	10 F7	OVCHK	BPL MD3	IF POS EXP THEN NO OVERFLOW
1FE4	00	OVFL	BRK	
		*		
		*		
		*	CONVERT EXP/	'MANT1 TO INTEGER IN M1 (HIGH) AND M1+1(LOW)
		*	EXP/MANT2 U	INEFFECTED
		*		
1FE5	20 5F 1	F	JSR RTAR	SHIFT MANT1 RT AND INCREMENT EXPNT
1FE8	A5 08	FI X	LDA X1	CHECK EXPONENT
1FEA	C9 8E		CMP =\$8E	IS EXPONENT 14?
1 FEC	DO F7		BNE FIX-3	NO, SHI FT
1 FEE	60	RTRN	RTS	RETURN
			DND	

\_\_\_\_\_

Errata for Rankin's 6502 Floating Point Routines by Roy Rankin

In the November/December issue of Dr. Dobb's Journal Roy Rankin published three error corrections to the Floating Point Routines presented above.

Dr. Dobb's Journal, November/December 1976, page 57.

ERRATA FOR RANKI N' S 6502 FLOATI NG POI NT ROUTI NES

Sept. 22, 1976

Dear Jim,

Subsequent to the publication of "Floating Point Routines for the 6502" (Vol.1, No.7) an error which I made in the LOG routine came to light which causes improper results if the argument is less than 1. The following changes will correct the error.

1.	After: Add:	A2	00	CONT	JSR LDX	SWAP =0	(1D07) LOAD X FOR HIGH BYTE OF EXPONENT
2.	After: Delete:				STA LDA STA	M1+1 =0 M1	(1D12)
	Add:	10 CA 86	01 09		BPL DEX STX	*+3 M1	I S EXPONENT NEGATI VE YES, SET X TO \$FF SET UPPER BYTE OF EXPONENT

3. Changes 1 and 2 shift the code by 3 bytes so add 3 to the addresses of the constants LN10 through MHLF whenever they are referenced. For example the address of LN10 changes from 1DCD to 1DD0. Note also that the entry point for LOG10 becomes 1DBF. The routines stays within the page and hence the following routines (EXP etc.) are not affected.

Yours truly,

Roy Rankin Dep. of Mech. Eng. Stanford University

\_\_\_\_\_

Floating Point Implementation in the Apple II by Steve Wozniak

An almost identical set of the above routines appeared in the original manual for the Apple II (the Red Book, January 1978). Documentation for these routines appeared in another book, the Wozpak II, in November 1979.

Woz 6502 Floating Point Routines

Apple II Reference Manual (Red Book), January 1978, pages 94-95.

			******	*****	******	
			*		*	
			* <b>APPIF</b> -1	I FI	ATING *	
			* POINT		NFS *	
			*	10011	*	
			* CODVDI	ידד 10	77 PV *	
			* ADDIE CO			
			* AFFLE U	JMPUI	ER INC. *	
				עדר ח	·	
			* ALL KIG	HIS KI	LSERVED *	
			* С 1		۰. ۲۷ +	
			* S. I	WOZNI	۰۰ AF	
			~ * * * * * * * * * * *	*****	*	
				• • • • • • • • • • • •		
			IIILE FI	LUAIII	NG PUINI	ROUTINES
			SIGN	EPZ	SF3	
			XZ	EPZ	SF4	
			M2	EPZ	SF5	
			X1	EPZ	SF8	
			M1	EPZ	SF9	
			E	EPZ	SFC	
			OVLOC	EQU	\$3F5	
				ORG	ŞF425	
F425:	18		ADD	CLC		CLEAR CARRY
F426:	A2	02		LDX	#\$2	INDEX FOR 3-BYTE ADD.
F428:	B5	F9	ADD1	LDA	M1, X	
F42A:	75	F5		ADC	M2, X	ADD A BYTE OF MANT2 TO MANT1
F42C:	95	F9		STA	M1, X	
F42E:	CA			DEX		INDEX TO NEXT MORE SIGNIF. BYTE.
F42F:	10	F7		BPL	ADD1	LOOP UNTIL DONE.
F431:	60			RTS		RETURN
F432:	06	F3	MD1	ASL	SI GN	CLEAR LSB OF SIGN.
F434:	20	37 F4		JSR	ABSWAP	ABS VAL OF M1, THEN SWAP WITH M2
F437:	24	F9	ABSWAP	BI T	M1	MANT1 NEGATI VE?
F439:	10	05		BPL	ABSWAP1	NO, SWAP WITH MANT2 AND RETURN.
F43B:	20	A4 F4		JSR	FCOMPL	YES, COMPLEMENT IT.
F43E:	E6	F3		I NC	SI GN	INCR SIGN, COMPLEMENTING LSB.
F440:	38		ABSWAP1	SEC		SET CARRY FOR RETURN TO MUL/DIV.
F441:	A2	04	SWAP	LDX	#\$4	INDEX FOR 4 BYTE SWAP.
F443:	94	FB	SWAP1	STY	E-1, X	
F445:	B5	F7		LDA	X1-1, X	SWAP A BYTE OF EXP/MANT1 WITH
F447:	B4	F3		LDY	X2-1, X	EXP/MANT2 AND LEAVE A COPY OF
F449:	94	F7		STY	X1-1, X	MANT1 IN E (3 BYTES). E+3 USED
F44B:	95	F3		STA	X2-1, X	· ·
F44D:	CA			DEX		ADVANCE INDEX TO NEXT BYTE

F44E:	DO 1	F3		BNE	SWAP1	LOOP UNTIL DONE.
F450:	60			RTS		RETURN
F451:	A9 8	8E	FLOAT	LDA	#\$8E	INIT EXP1 TO 14,
F453:	85	F8		STA	X1	THEN NORMALIZE TO FLOAT.
F455:	A5 ]	F9	NORM1	LDA	M1	HI GH-ORDER MANT1 BYTE.
F457:	C9 (	CO		CMP	#\$C0	UPPER TWO BITS UNEQUAL?
F459:	30 (	OC		BMI	RTS1	YES, RETURN WITH MANT1 NORMALIZED
F45B:	C6 ]	F8		DEC	X1	DECREMENT EXP1.
F45D:	06 ]	FB		ASL	M1+2	
F45F:	26	FA		ROL	M1 + 1	SHIFT MANT1 (3 BYTES) LEFT.
F461:	26	F9		ROL	M1	
F463:	A5 ]	F8	NORM	LDA	X1	EXP1 ZERO?
F465:	DO	EE		BNE	NORM1	NO, CONTI NUE NORMALI ZI NG.
F467:	60		RTS1	RTS		RETURN.
F468:	20	A4 F4	FSUB	JSR	FCOMPL	CMPL MANT1, CLEARS CARRY UNLESS O
F46B:	20	7B F4	SWPALGN	JSR	ALGNSWP	RIGHT SHIFT MANT1 OR SWAP WITH
F46E:	A5 ]	F4	FADD	LDA	X2	
F470:	C5 ]	F8		CMP	X1	COMPARE EXP1 WITH EXP2.
F472:	DO	F7		BNE	SWPALGN	IF #, SWAP ADDENDS OR ALIGN MANTS.
F474:	20	25 F4		JSR	ADD	ADD ALI GNED MANTI SSAS.
F477:	50	EA	ADDEND	BVC	NORM	NO OVERFLOW, NORMALIZE RESULT.
F479:	70 (	05		BVS	RTLOG	OV: SHIFT M1 RIGHT, CARRY INTO SIGN
F47B:	90 (	C4	ALGNSWP	BCC	SWAP	SWAP IF CARRY CLEAR,
			*	ELSE SI	HIFT RIGH	T ARI TH.
F47D:	A5 ]	F9	RTAR	LDA	M1	SIGN OF MANT1 INTO CARRY FOR
F47F:	0A			ASL		RI GHT ARI TH SHI FT.
F480:	E6 ]	F8	RTLOG	I NC	X1	INCR X1 TO ADJUST FOR RIGHT SHIFT
F482:	F0	75		BEQ	OVFL	EXP1 OUT OF RANGE.
F484:	A2 ]	FA	RTLOG1	LDX	#\$FA	INDEX FOR 6: BYTE RIGHT SHIFT.
F486:	76	FF	ROR1	ROR	E+3, X	
F488:	E8			I NX		NEXT BYTE OF SHIFT.
F489:	DO 1	FB		BNE	ROR1	LOOP UNTIL DONE.
F48B:	60			RTS		RETURN.
F48C:	20 3	32 F4	FMUL	JSR	MD1	ABS VAL OF MANT1, MANT2
F48F:	65	F8		ADC	X1	ADD EXP1 TO EXP2 FOR PRODUCT EXP
F491:	20	E2 F4		JSR	MD2	CHECK PROD. EXP AND PREP. FOR MUL
F494:	18			CLC		CLEAR CARRY FOR FIRST BIT.
F495:	20	84 F4	MUL1	JSR	RTLOG1	M1 AND E RIGHT (PROD AND MPLIER)
F498:	90 (	03		BCC	MUL2	IF CARRY CLEAR, SKIP PARTIAL PROD
F49A:	20	25 F4		JSR	ADD	ADD MULTI PLI CAND TO PRODUCT.
F49D:	88		MUL2	DEY		NEXT MUL I TERATI ON.
F49E:	10	F5		BPL	MUL1	LOOP UNTIL DONE.
F4A0:	46	F3	MDEND	LSR	SI GN	TEST SIGN LSB.
F4A2:	90 ]	BF	NORMX	BCC	NORM	IF EVEN, NORMALIZE PROD, ELSE COMP
F4A4:	38		FCOMPL	SEC		SET CARRY FOR SUBTRACT.
F4A5:	A2 (	03		LDX	#\$3	INDEX FOR 3 BYTE SUBTRACT.
F4A7:	A9 (	00	COMPL1	LDA	#\$0	CLEAR A.
F4A9:	F5 1	F8		SBC	X1, X	SUBTRACT BYTE OF EXP1.
F4AB:	95	F8		STA	X1, X	RESTORE IT.
F4AD:	CA			DEX		NEXT MORE SI GNI FI CANT BYTE.
F4AE:	DO	F7		BNE	COMPL1	LOOP UNTIL DONE.
F4B0:	F0 (	С5		BEQ	ADDEND	NORMALIZE (OR SHIFT RT IF OVFL).
F4B2:	20 3	32 F4	FDI V	JSR	MD1	TAKE ABS VAL OF MANT1, MANT2.
F4B5:	E5 1	F8		SBC	X1	SUBTRACT EXP1 FROM EXP2.
F4B7:	20 ]	E2 F4		JSR	MD2	SAVE AS QUOTIENT EXP.
F4BA:	38		DI V1	SEC		SET CARRY FOR SUBTRACT.
F4BB:	A2 (	02		LDX	#\$2	INDEX FOR 3-BYTE SUBTRACTION.
F4BD:	B5 ]	F5	DI V2	LDA	M2, X	
F4BF:	F5 1	FC		SBC	Е, Х	SUBTRACT A BYTE OF E FROM MANT2.
F4C1:	48			PHA		SAVE ON STACK.
F4C2:	CA			DEX		NEXT MORE SI GNI FI CANT BYTE.
F4C3:	10	F8		BPL	DI V2	LOOP UNTIL DONE.
F4C5:	A2 ]	FD		LDX	#\$FD	INDEX FOR 3-BYTE CONDITIONAL MOVE
F4C7:	68		DI V3	PLA		PULL BYTE OF DIFFERENCE OFF STACK

F4C8:	90 02		BCC	DI V4	IF M2 <e don't="" m2.<="" restore="" th="" then=""></e>
F4CA:	95 F8		STA	M2+3, X	
F4CC:	E8	DI V4	INX	DIVO	NEXT LESS SIGNIFICANT BYTE.
F4CD:	DO F8		BNE	DI V3	LOOP UNTIL DONE.
F4CF:	26 FB		ROL	MI+2	
F4D1:	26 FA		RUL	MI + I	RULL QUUITENT LEFT, CARRY INTO LSB
F4D5:	20 F9 06 E7		KUL ACI	M1 M2 2	
Г4DЭ. Е4D7-	00 F7 26 F6		POL	$M\mathcal{L} + \mathcal{L}$ M2 + 1	CULET DIVIDEND LEET
F4D7.	26 F5		ROL	M2 + 1 M2	SHITI DIVIDEND LEFT
F4DB	BO 1C		BCS	OVEI	OVEL IS DUE TO UNNORMED DIVISOR
F4DD:	88		DEY	OVIL	NEXT DI VI DE LITERATI ON
F4DE:	DO DA		BNE	DI V1	LOOP UNTIL DONE 23 I TERATIONS.
F4E0:	FO BE		BEQ	MDEND	NORM. QUOTI ENT AND CORRECT SI GN.
F4E2:	86 FB	MD2	STŇ	M1+2	v
F4E4:	86 FA		STX	M1 + 1	CLEAR MANT1 (3 BYTES) FOR MUL/DIV.
F4E6:	86 F9		STX	M1	
F4E8:	BO OD		BCS	OVCHK	IF CALC. SET CARRY, CHECK FOR OVFL
F4EA:	30 04		BMI	MD3	IF NEG THEN NO UNDERFLOW.
F4EC:	68		PLA		POP ONE RETURN LEVEL.
F4ED:	68		PLA		
F4EE:	90 B2	1.5.4	BCC	NORMX	CLEAR X1 AND RETURN.
F4F0:	49 80	MD3	EOR	#\$80	COMPLEMENT SIGN BIT OF EXPONENT.
F4F2:	85 F8		STA	XI	STURE IT.
F4F4:	AU 17		LDY	#\$17	COUNT 24 MUL/23 DIV TIERATIONS.
F4F0: E4E7:	00 10 E7	OVCUK		MD 2	KEIUKN. LE DOSITIVE EVD THEN NO OVEL
Г4Г7. F4F0.	10 F7 4C E5 03	OVEI	DFL IMD		IF PUSITIVE EAF THEN NU UVFL.
141.5.	40 15 05	OVIL	ORC	SF63D	
F63D	20 7D F4	FI X1	ISR	RTAR	
F640:	A5 F8	FLX	LDA	X1	
F642:	10 13	1 1 / 1	BPL	UNDFL	
F644:	C9 8E		CMP	#\$8E	
F646:	DO F5		BNE	FI X1	
F648:	24 F9		BI T	M1	
F64A:	10 OA		BPL	FI XRTS	
F64C:	A5 FB		LDA	M1+2	
F64E:	F0 06		BEQ	FI XRTS	
F650:	E6 FA		I NC	M1 + 1	
F652:	DO 02		BNE	FI XRTS	
F654:	E6 F9		1 NC	M1	
F656:	60	FIXRTS	RTS	11 Å Å	
F65/:	A9 UU	UNDFL	LDA	#\$U	
F059:	85 F9		SIA	MI.	
гоэв: герр	00 FA		DTC	M1 + 1	
F05D:	00		R13		
****	*******	< * * * * * * * * * * *	****	******	*****

Wozpak ][, November 1979, pages 109-115.

### FLOATING POINT PACKAGE

The mantissa-exponent, or 'floating point' numerical representation is widely used by computers to express values with a wide dynamic range. With floating point representation, the number 7.5 x 10^22 requires no more memory to store than the number 75 does. We have allowed for binary floating point arithmetic on the APPLE ][ computer by providing a useful subroutine package in ROM, which performs the common arithmetic functions. Maximum precision is retained by these routines and overflow conditions such as 'divide by zero' are trapped for the user. The 4-byte floating point number representation is compatible with future APPLE products such as floating point BASIC. A small amount of memory in Page Zero is dedicated to the floating point workspace, including the two floating-point accumulators, FP1 and FP2. After placing operands in these accumulators, the user calls subroutines in the ROM which perform the desired arithmetic operations, leaving results in FP1. Should an overflow condition occur, a jump to location \$3F5 is executed, allowing a user routine to take appropriate action.

### FLOATING POINT REPRESENTATION



Exponent Signed Mantissa

1. Mantissa

The floating point mantissa is stored in two's complement representation with the sign at the most significant bit (MSB) position of the high-order mantissa byte. The mantissa provides 24 bits of precision, including sign, and can represent 24-bit integers precisely. Extending precision is simply a matter of adding bytes at the low order end of the mantissa.

Except for magnitudes less than  $2^{-128}$  (which lose precision) mantissa are normalized by the floating point routines to retain maximum precision. That is, the numbers are adjusted so that the upper two high-order mantissa bits are unequal.

HI GH-ORDER MANTI SSA BYTE 01. XXXXXX Positive mantissa. 10. XXXXXX Negative mantissa. 00. XXXXXX Unnormalized mantissa. 11. XXXXXX Exponent = -128.

2. Exponent.

The exponent is a binary scaling factor (power of two) which is applied to the mantissa. Ranging from -128 to +127, the exponent is stored in standard two's complement representation except for the sign bit which is complemented. This representation allows direct comparison of exponents, since they are stored in increasing numerical sequence. The most negative exponent, corresponding to the smallest magnItude, -128, is stored as \$00 (\$ means hexidecimal) and the most positive, +127, is stored as \$FF (all ones).

EXPONENT	STORED AS			
+127	11111111	(\$FF)		
+3 +2	10000011 10000010	(\$83) (\$82)		

+1	1000001	(\$81)
0	1000000	(\$80)
- 1	01111111	(\$7F)
- 2	01111110	(\$7E)
- 3	01111101	(\$7D)
- 128	00000000	(\$00)

The smallest magnitude which can be represented is  $2^{-150}$ .



The largest positive magnitude which can be represented is +2^128-1.

\$7F	\$7F	\$FF	   \$FF 
EXP		MANTI SSA	

# FLOATI NG POI NT REPRESENTATI ON EXAMPLES

DECI MAL NUMBER	HEX EXPONENT	HEX MANTI SSA
+ 3	81	60 00 00
+ 4	82	40 00 00
+ 5	82	50 00 00
+ 7	82	70 00 00
+12	83	60 00 00
+15	83	78 00 00
+17	84	44 00 00
+20	84	50 00 00
+60	85	78 00 00
- 3	81	A0 00 00
- 4	81	80 00 00
- 5	82	BO 00 00
- 7	82	90 00 00
- 12	83	AO OO OO
- 15	83	88 00 00
- 17	84	BC 00 00
- 20	84	BO 00 00
- 60	85	88 00 00

# FLOATI NG POI NT SUBROUTI NE DESCRI PTI ONS

FCOMPL subroutine (address \$F4A4)

Purpose: FCOMPL is used to negate floating point numbers.

Entry: A normalized or unnormalized value is in FP1 (floating point accumulator 1).

Uses: NORM, RTLOG.

Exit: The value in FP1 is negated and then normalized to retain precision. The 3-byte FP1 extension, E, may also be altered but FP2 and SIGN are not disturbed. The 6502 A-REG is altered and the X-REG is cleared. The Y-REG is not disturbed.

Caution: Attempting to negate  $-2^{128}$  will result in an overflow since  $+2^{128}$  is not representable, and a jump to location \$3F5 will be executed, with the following contents in FP1.



Example: Prior to calling FCOMPL, FP1 contains +15.



After calling FCOMPL as a subroutine, FP1 contains -15.



FADD subroutine (address \$F46E)

Purpose: To add two numbers in floating point form.

Entry: The two addends are in FP1 and FP2 respectively. For maximum

precision, both should be normalized.

Uses: SWPALGN, ADD, NORM, RTLOG.

Exit: The normalized sum is left in FP1. FP2 contains the addend of greatest magnitude. E is altered but sign is not. The A-REG is altered and the X-REG is cleared. The sum mantissa is truncated to 24 bits.

Caution: Overflow may result if the sum is less that  $-2^{128}$  or greater than  $+2^{128-1}$ . If so, a jump to location \$3F5 is executed leaving 0 in X1, and twice the proper sum in the mantissa M1. The sign bit is left in the carry, 0 for positive, 1 for negative.



(For carry=0, true sum=+X.YYY x 2^128)

Example: Prior to calling FADD, FP1 contains +12 and FP2 contains -5.



After calling FADD, FP1 contains +7 (FP2 contains +12).



FSUB subroutine (address \$F468)

Purpose: To subtract two floating point numbers.

Entry: The minuend is in FP1 and the subtrahend is in FP2. Both should be normalized to retain maximum precision prior to calling FSUB.

Uses: FCOMPL, ALGNSWP, FADD, ADD, NORM, RTLOG.

Exit: The normalized difference is in FP1 with the mantissa truncated to 24 bits. FP2 holds either the minued or the negated subtrahend, whichever is of greater magnitude. E is altered but SIGN and SCR are not. the A-REG is altered and the X-REG is cleared. The Y-REG is not disturbed.

Cautions: An exit to location S3F5 is taken if the result is less than  $-2^{128}$  or greater than  $+2^{128}-1$ . or if the subtrahend is  $-2^{128}$ .

Example: Prior to calling FSUB, FP1 contains +7 (minuend) and FP2 contains -5 (subtrahend).



After calling FSUB, FP1 contains +12 and FP2 contains +7.



FMUL subroutine (address \$F48C)

Purpose: To multiply floating point numbers.

Entry: The multiplicand and multiplier must reside in FP1 and FP2 respectively. Both should be normalized prior to calling FMUL to retain maximum precision.

Uses: MD1, MD2, RTLOG1, ADD, MDEND.

Exit: The signed normalized floating point product is left in FP1. M1 is truncated to contain the 24 most significant mantissa bits (including sign). The absolute value of the multiplier mantissa (M2) is left in FP2. E, SIGN, and SCR are altered. The A- and X-REGs are altered and the Y-REG contains \$FF upon exit.

Cautions: An exit to location 3F5 is taken if the product is less than  $-2^{128}$  or greater than  $+2^{128}-1$ .

Notes: FMUL will run faster if the absolute value of the multiplier

mantissa contains fewer  $^{\prime}1^{\prime}s$  than the absolute value of the multiplicand mantissa.

Example: Prior to calling FMUL, FP1 contains +12 and FP2 contains -5.



After calling FMUL, FP1 contains -60 and FP2 contains +5.



FDIV subroutine (addr \$F4B2)

Purpose: To perform division of floating point numbers.

Entry: The normalized dividend is in FP2 and the normalized divisor is in FP1.

Exit: The signed normalized floating point quotient is left in FP1. The mantissa (M1) is truncated to 24 bits. The 3-bit M1 extension (E) contains the absolute value of the divisor mantissa. MD2, SIGN, and SCR are altered. The A- and X-REGs are altered and the Y-REG is cleared.

Uses: MD1, MD2, MDEND.

Cautions: An exit to location 3F5 is taken if the quotient is less than -2^128 or greater than +2^128-1

Notes: MD2 contains the remainder mantissa (equivalent to the MOD function). The remainder exponent is the same as the quotient exponent, or 1 less if the dividend mantissa magnitude is less than the divisor mantissa

magnitude.

Example: Prior to calling FDIV, FP1 contains -60 (dividend), and FP2 contains +12 (divisor).



After calling FMUL, FP1 contains -5 and M2 contains 0.



FLOAT Subroutine (address \$F451)

Purpose: To convert integers to floating point representation.

Entry: A signed (two's complement) 2-byte integer is stored in M1 (high-order byte) and M1+1 (low-order byte). M1+2 must be cleared by user prior to entry.

Uses: NORM1.

Exit: The normalized floating point equivalent is left in FP1. E, FP2, SIGN, and SCR are not disturbed. The A-REG contains a copy of the high-order mantissa byte upon exit but the X- and Y-REGs are not disturbed. The carry is cleared.

Notes: To float a 1-byte integer, place it in M1+1 and clear M1 as well as M1+2 prior to calling FLOAT.

FLOAT takes approximately 3 msec. longer to convert zero to floating point form than other arguments. The user may check for zero prior to calling FLOAT and increase throughput.

\* LOW-ORDER INT. BYTE IN A-REG \* HI GH-ORDER BYTE IN Y-REG \* 85 FA XFLOAT STA M1+1

84	F9			STY	M1	INIT MANT1
AO	00			LDY	#\$0	
84	FB			STY	M1+2	
05	D9			ORA	M1	CHK BOTH
						BYTES FOR
DO	03			BNE	TOFLO/	AT ZERO
85	F8			STA	X1	IF SO CLR X1
60				RTS		AND RETURN
4C	51	F4	TOFLOAT	JMP	FLOAT	ELSE FLOAT
						I NTEGER

Example: Float +274 (\$0112 hex)

#### CALLING SEQUENCE

A0	01		LDY	#\$01	HI GH- ORDER
					INTEGER BYTE
A9	12		LDA	#\$12	LOW- ORDER
					INTEGER BYTE
84	F9		STY	M1	
85	FA		STA	M1 + 1	
A9	00		LDA	#\$00	
85	F8		STA	M1+2	
20	51	F4	JSR	FLOAT	

Upon returning from FLOAT, FP1 contains the floating point representation of  $\pm 274.$ 



FIX subroutine (address \$F640)

Purpose: To extract the integer portion of a floating point number with truncation (ENTIER function).

Entry: A floating point value is in FP1. It need not be normalized.

Uses: RTAR.

Exit: The two-byte signed two's complement representation of the integer portion is left in M1 (high-order byte) and M1+1 (low-order byte). The floating point values +24.63 and -61.2 are converted to the integers +24 and -61 respectively. FP1 and E are altered but FP2, E, SIGN, and SCR are not. The A- and X-REGs are altered but the Y-REG is not.

Example: The floating point value +274 is in FP1 prior to calling FIX.

FP1:	\$88	\$44	\$80	0	(+274)



After calling FIX, M1 (high-order byte) and M1+1 (low-order byte) contain the integer representation of +274 (\$0112).



Note: FP1 contains an unnormalized representation of +274 upon exit.

# NORM Subroutine (address \$F463)

Purpose: To normalize the value in FP1, thus insuring maximum precision.

Entry: A normalized or unnormalized value is in FP1.

Exit: The value in FP1 is normalized. A zero mantissa will exit with X1=0 (2 exponent). If the exponent on exit is -128 (X1=0) then the mantissa (M1) is not necessarily normalized (with the two high-order mantissa bits unequal). E, FP2, SIGN, AND SCR are not distubed. The A-REG is disturbed but the X- and Y-REGs are not. The carry is set.

Example: FP1 contains +12 in unnormalized form (as .0011 x 2).



Upon exit from NORM, FP1 contains +12 in normalized form (as 1.1 x 2).



NORM1 subroutine (address \$F455)

Purpose: To normalize a floating point value in FP1 when it is known the exponent is not -128 (X1=0) upon entry.

Entry: An unnormalized number is in FP1. The exponent byte should not be 0 for normal use.

Exit: The normalized value is in FP1. E, FP2, SIGN, and SCR are not not disturbed. The A-REG is altered but the X- and Y-REGs are not.

ADD Subroutine (address \$F425)

Purpose: To add the two mantissas (M1 and M2) as 3-byte integers.

Entry: Two mantissas are in M1 (through M1+2) and M2 (through M2+2). They should be aligned, that is with identical exponents, for use in the FADD and FSUB subroutines.

Exit: the 24-bit integer sum is in M1 (high-order byte in M1, low-order byte in M1+2). FP2, X1, E, SIGN and SCR are not disturbed. The A-REG contains the high-order byte of the sum, the X-REG contains FF and the Y-REG is not altered. The carry is the '25th' sum bit.

Example: FP1 contains +5 and FP2 contains +7 prior to calling ADD.



Upon exit, M1 contains the overflow value for +12. Note that the sign bit is incorrect. This is taken care of with a call to the right shift routine.



ABSWAP Subroutine (address \$F437)

Purpose: To take the absolute value of FP1 and then swap FP1 with FP2. Note that two sequential calls to ABSWAP will take the absolute values of both FP1 and FP2 in preparation for a multiply or divide.

Entry: FP1 and FP2 contain floating point values.

Exit: The absolute value of the original FP1 contents are in FP2 and the original FP2 contents are in FP1. The least significant bit of SIGN is complemented if a negation takes place (if the original FP1 contents are negative) by means of an increment. SCR and E are used. The A-REG

contains a copy of X2, the X-REG is cleared, and the Y-REG is not altered.

RTAR Subroutine (address \$F47D)

Purpose: To shift M1 right one bit position while incrementing X1 to compensate for scale. This is roughly the opposite of the NORM subroutine.

Entry: A normalized or unnormalized floating point value is in FP1.

Exit: The 6-byte field MANT1 and E is shifted right one bit arithmetically and X1 is incremented by 1 to retain proper scale. The sign bit of MANT1 (MSB of M1) is unchanged. FP2, SIGN, and SCR are not disturbed. The A-REG contains the least significant byte of E (E+2), the X-REG is cleared, and the Y-REG is not disturbed.

Caution: If X1 increments of O (overflow) then an exit to location \$3F5 is taken, the A-REG contains the high-order MANT1 byte, M1 and X1 is cleared. FP2, SIGN, SCR, and the X- and Y-REGs are not disturbed.

Uses: RTLOG

Example: Prior to calling RTAR, FP1 contains the normalized value -7.



After calling RTAR, FP1 contains the unnormalized value -7 (note that precision is lost off the low-order end of M1).



Note: M1 sign bit is unchanged.

#### RTLOG subroutine (address \$F480)

Purpose: To shift the 6-byte field MANT1 and E one bit to the right (toward the least significant bit). The 6502 carry bit is shifted into the high-order M1 bit. This is useful in correcting binary sum overflows.

Entry: A normalized or unnormalized floating point value is in FP1. The carry must be cleared or set by the user since it is shifted Into the sign bit of M1.

Exit: Same as RTAR except that the sign of M1 is not preserved (it is set

to the value of the carry bit on entry)

Caution: Same as RTAR.

Example: Prior to calling RTLOG, FP1 contains the normalized value -12 and the carry is clear.



After calling RTLOG, M1 is shifted one bit to the right and the sign bit is clear. X1 is incremented by 1.



Note: The bit shifted off the end of MANT1 is rotated into the high-order bit of the 3-byte extension E. The 3-byte E field is also shifted one bit to the right.

RTLOG1 subroutine (address \$F484)

Purpose: To shift MANT1 and E right one bit without adjusting X1. This is used by the multiply loop. The carry is shifted into the sign bit of MANT1.

Entry: M1 and E contain a 6-byte unsigned field. E is the 3-byte low-order extension of MANT1.

Exit: Same as RTLOG except that X1 is not altered and an overflow exit cannot occur.

MD2 subroutine (address \$F4E2)

Purpose: To clear the 3-byte MANT1 field for FMUL and FDIV, check for inital result exponent overflow (and underflow), and initialize the X-REG to \$17 for loop counting.

Entry: the X-REG is cleared by the user since it is placed in the 3 bytes of MANT1. The A-REG contains the result of an exponent addition (FMUL) or subtraction (FDIV). The carry and sign status bits should be set according to this addition or subtraction for overflow and underflow determination.

Exit: The 3 bytes of M1 are cleared (or all set to the contents of the X-REG on Entry) and the Y-REG is loaded with \$17. The sign bit of the

A-REG is complemented and a copy of the A-REG is stored in X1. FP2, SIGN, SCR, and the X-REG are not disturbed.

Uses: NORM.

Caution: Exponent overflow results in an exit to location \$3F5. Exponent underflow results in an early return from the calling subroutine (FDIV or FMUL) with a floating point zero in FP1. Because MD2 pops a return address off the stack, it may only be called by another subroutine.

###