

GenRef
v1.00

MDOS Reference guide.

Math Library

(C) Copyright 2004
Beery W. Miller
ALL RIGHTS RESERVED

Math - CONTENTS

| | |
|--|-----------|
| MATH OVERVIEW | 3 |
| MATH DETAILS – RADIX 100 | 4 |
| CALLING MATH FUNCTIONS..... | 5 |
| FLOATING POINT COMPARE | 6 |
| FLOATING POINT SUBTRACT | 7 |
| FLOATING POINT ADD | 8 |
| FLOATING POINT MULTIPLY | 9 |
| FLOATING POINT DIVIDE..... | 10 |
| FLOATING POINT POWER | 11 |
| FLOATING POINT E^x | 12 |
| FLOATING POINT LN(x) | 13 |
| FLOATING POINT SQR(x) | 14 |
| FLOATING POINT COS(x) | 15 |
| FLOATING POINT SIN(x)..... | 16 |
| FLOATING POINT TAN(x)..... | 17 |
| FLOATING POINT ATN(x)..... | 18 |
| FLOATING POINT GREATEST INGETER..... | 19 |
| CONVERT FLOATING POINT TO INTEGER | 20 |
| CONVERT INTEGER TO FLOATING POINT | 21 |
| CONVERT STRING TO INTEGER..... | 22 |
| CONVERT STRING TO FLOATING POINT | 23 |
| CONVERT FLOATING POINT TO STRING | 24 |

MATH - OVERVIEW

All math management routines in MDOS are provided to aid a programmer in writing applications requiring math operations beyond the immediate instruction set of the TMS 9995 microprocessor. The following math operations are supported within the operating system:

- | | |
|-------------------------------------|-------|
| • Floating Point Compare | FCOMP |
| • Floating Point Subtract | FSUB |
| • Floating Point Add | FADD |
| • Floating Point Multiply | FMULT |
| • Floating Point Divide | FDIV |
| • Floating Point Power | PWR |
| • Floating Point e^x | EXP |
| • Floating Point $\ln(x)$ | LOG |
| • Floating Point \sqrt{x} | SQR |
| • Floating Point $\cos(x)$ | COS |
| • Floating Point $\sin(x)$ | SIN |
| • Floating Point $\tan(x)$ | TAN |
| • Floating Point $\text{atan}(x)$ | ATN |
| • Floating Point Greatest Integer | GRI |
| • Convert Floating Point to Integer | CFI |
| • Convert Integer to Floating Point | CIF |
| • Convert String to Integer | CSINT |
| • Convert String to Floating Point | CSN |
| • Convert Floating Point to String | CNS |

MATH Details – Radix 100

All floating point arguments must be on an even byte boundary and the calling registers must be in PAD from >F000 to >F060. Floating Point representation is in Radix 100 format:

```

0 =    00 00 xx  xx  xx  xx  xx  xx
+n =    e0 m0 m1  m2  m3  m4  m5  m6

```

e0 is $\text{int}(\log_{100}(n)) + >40$

m0 – m6 are numbers from >00 to >63 (0 to 99)

m0 is the most significant digit of mantissa,

m6 is the least significant digit of mantisa.

In normalized numbers, decimal is between m0 and m1

-n is same as “n” except first word is negated ... -(e0 m0)

Examples:

| <u>Decimal</u> | <u>Floating Point</u> | | | | | | | |
|----------------|-----------------------|-----|-----|-----|-----|-----|-----|-----|
| 7 | >40 | >07 | >00 | >00 | >00 | >00 | >00 | >00 |
| 70 | >40 | >46 | >00 | >00 | >00 | >00 | >00 | >00 |
| 2,345,600 | >43 | >02 | >22 | >38 | >00 | >00 | >00 | >00 |
| 23,456,000 | >43 | >17 | >2D | >3C | >00 | >00 | >00 | >00 |
| 0 | >00 | >00 | >xx | >xx | >xx | >xx | >xx | >xx |
| -7 | >BF | >F9 | >00 | >00 | >00 | >00 | >00 | >00 |
| -70 | >BF | >BA | >00 | >00 | >00 | >00 | >00 | >00 |
| -2,345,600 | >BC | >FE | >22 | >38 | >00 | >00 | >00 | >00 |

CALLING MATH FUNCTIONS

The MDOS Math Library must be called from within a machine code program running as a task under MDOS. You pass arguments to the Math Library via the calling registers.

The MDOS Video Library is invoked from a machine code program when software trap number zero (XOP 0) is called with a library number of 10. The calling program's R0 must contain the 16-bit subprogram at the time of the XOP. The following code fragment will convert a string to a floating point number.

| | | |
|------|----------|----------------------------------|
| LI | R0,>0011 | Convert String to Floating Point |
| LI | R1,RES | Result Pointer |
| LI | R2,STR1 | String Pointer |
| LI | R3,3 | 3 characters to convert |
| XOP | @MATH,0 | Access subprogram |
| | | |
| MATH | DATA | 10 |
| STR1 | TEXT | "123" |
| | BYTE | 0 |
| | | Null terminated |
| RES | DATA | 0,0,0,0 |
| | EVEN | Floating Point Result |

In the preceding example, two hidden assumptions were made. First it is assumed that STR1 is located on a page which is currently mapped into a memory page which has the same 16-bit address page number as its Virtual address page number (read the section on Memory Management.) The second assumption is that MATH is actually at the virtual address MATH, not in some overlay segment with a different virtual address.

Math Library

Floating Point Compare

| | |
|-------------------|---|
| Function | Floating Point Compare |
| Parameters | R0 = >0000 R2 = Pointer to Float 1 R3 = Pointer to Float 2 |
| Results | Status Register = AG set if (Float2 > Float1) EQ set if (Float2 = Float1) |

Parameter Description

| | |
|--------|--|
| Float1 | 8 byte representation in Radix 100 format. |
| Float2 | 8 byte representation in Radix 100 format. |

| |
|--------------------------------|
| Floating Point Subtract |
|--------------------------------|

Function Floating Point Subtract

Parameters R0 = >0001
 R1 = Pointer to Result Destination
 R2 = Pointer to Float 1
 R3 = Pointer to Float 2

Results R0 = Error Code
 R1 Pointer = Float2 – Float1

Parameter Description

| | |
|--------|--|
| Float1 | 8 byte representation in Radix 100 format. |
| Float2 | 8 byte representation in Radix 100 format. |
| Result | 8 byte representation in Radix 100 format. |

| |
|---------------------------|
| Floating Point Add |
|---------------------------|

Function Floating Point Add

Parameters
 R0 = >0002
 R1 = Pointer to Result Destination
 R2 = Pointer to Float 1
 R3 = Pointer to Float 2

Results
 R0 = Error Code
 R1 Pointer = Float2 + Float1

Parameter Description

| | |
|--------|--|
| Float1 | 8 byte representation in Radix 100 format. |
| Float2 | 8 byte representation in Radix 100 format. |
| Result | 8 byte representation in Radix 100 format. |

| |
|--------------------------------|
| Floating Point Multiply |
|--------------------------------|

Function Floating Point Multiply

Parameters R0 = >0003
 R1 = Pointer to Result Destination
 R2 = Pointer to Float 1
 R3 = Pointer to Float 2

Results R0 = Error Code
 R1 Pointer = Float2 * Float1

Parameter Description

| | |
|--------|--|
| Float1 | 8 byte representation in Radix 100 format. |
| Float2 | 8 byte representation in Radix 100 format. |
| Result | 8 byte representation in Radix 100 format. |

| |
|------------------------------|
| Floating Point Divide |
|------------------------------|

Function Floating Point Divide

Parameters R0 = >0004
 R1 = Pointer to Result Destination
 R2 = Pointer to Float 1
 R3 = Pointer to Float 2

Results R0 = Error Code
 R1 Pointer = Float2 / Float1

Parameter Description

| | |
|--------|--|
| Float1 | 8 byte representation in Radix 100 format. |
| Float2 | 8 byte representation in Radix 100 format. |
| Result | 8 byte representation in Radix 100 format. |

| |
|-----------------------------|
| Floating Point Power |
|-----------------------------|

Function Floating Point Power

Parameters R0 = >0005
 R1 = Pointer to Result Destination
 R2 = Pointer to Float 1
 R3 = Pointer to Float 2

Results R0 = Error Code
 R1 Pointer = Float2^{Float1}

Parameter Description

| | |
|--------|--|
| Float1 | 8 byte representation in Radix 100 format. |
| Float2 | 8 byte representation in Radix 100 format. |
| Result | 8 byte representation in Radix 100 format. |

| |
|-------------------------------------|
| Floating Point e^x |
|-------------------------------------|

Function Floating Point e^x

Parameters R0 = >0006
 R1 = Pointer to Result Destination
 R2 = Pointer to Float 1

Results R0 = Error Code
 R1 Pointer = e^{Float1}

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

Floating Point ln(x)

Function Floating Point ln(x)

Parameters R0 = >0007
R1 = Pointer to Result Destination
R2 = Pointer to Float 1

Results R0 = Error Code
R1 Pointer = ln(Float1)

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

| |
|------------------------------|
| Floating Point SQR(x) |
|------------------------------|

Function Floating Point sqr(x)

Parameters R0 = >0008
 R1 = Pointer to Result Destination
 R2 = Pointer to Float 1

Results R0 = Error Code
 R1 Pointer = SQR(Float1)

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

| |
|------------------------------|
| Floating Point COS(x) |
|------------------------------|

Function Floating Point cos (x)

Parameters R0 = >0009
R1 = Pointer to Result Destination
R2 = Pointer to Float 1

Results R0 = Error Code
R1 Pointer = COS(Float1)

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

Floating Point SIN(x)

Function Floating Point sin(x)

Parameters R0 = >000A
R1 = Pointer to Result Destination
R2 = Pointer to Float 1

Results R0 = Error Code
R1 Pointer = SIN(Float1)

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

| |
|------------------------------|
| Floating Point TAN(x) |
|------------------------------|

Function Floating Point tan(x)

Parameters R0 = >000B
R1 = Pointer to Result Destination
R2 = Pointer to Float 1

Results R0 = Error Code
R1 Pointer = TAN(Float1)

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

| |
|------------------------------|
| Floating Point ATN(x) |
|------------------------------|

Function Floating Point atn(x)

Parameters R0 = >000C
R1 = Pointer to Result Destination
R2 = Pointer to Float 1

Results R0 = Error Code
R1 Pointer = ATN(Float1)

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

| |
|--|
| Floating Point Greatest Integer |
|--|

Function Floating Point Greatest Integer

Parameters R0 = >000D
R1 = Pointer to Result Destination
R2 = Pointer to Float 1

Results R0 = Error Code
R1 Pointer = || Float1 ||

Parameter Description

Float1 8 byte representation in Radix 100 format.

Result 8 byte representation in Radix 100 format.

| |
|--|
| Convert Floating Point to Integer |
|--|

Function Convert Floating Point to Integer

Parameters R0 = >000E
 R2 = Pointer to Float 1

Results R0 = Error Code
 R1 = Integer

Parameter Description

Float1 8 byte representation in Radix 100 format.

Integer 2 byte (16 bit) integer

| |
|--|
| Convert Integer to Floating Point |
|--|

Function Convert integer to Floating Point

Parameters R0 = >000F
 R1 = Pointer to Result Destination
 R2 = Integer

Results R0 = Error Code
 R1 Pointer = Float(integer)

Parameter Description

Float1 8 byte representation in Radix 100 format.

Integer 2 byte (16 bit) integer

| |
|----------------------------------|
| Convert String to Integer |
|----------------------------------|

Function Convert String to Integer

Parameters R0 = >0010
 R2 = Pointer to String

Results R0 = Error Code
 R1 = Integer

Parameter Description

Float1 8 byte representation in Radix 100 format.

String ASCII text string nul terminated.

| |
|---|
| Convert String to Floating Point |
|---|

Function Convert String to Floating Point

Parameters R0 = >0011
 R1 = Pointer to Result Destination
 R2 = Pointer to String
 R3 = String Length

Results R0 = Error Code
 R1 Pointer = Float(String)

Parameter Description

Float 8 byte representation in Radix 100 format.

String ASCII character text string.

| |
|--------------------------------|
| Convert Float to String |
|--------------------------------|

Function Convert Float to String

Parameters

R0 = >0012
R1 = Pointer to String
R2 = Pointer to Float1
R3 = opt1
 Bit 0: 0 = free form (ignore opt2, opt3)
 1 = fixed (opt2, opt3 are field sizes)
 Bit 1: 1 for explicit sign
 Bit 2: 1 to show sign of positive number as a '+' instead of space
 Bit 3: 1 for extended E-notation
 Bit 4: 1 for extended E-notation (bit 3 must also be 1)
R4 = if fixed format, number of places to left of decimal point,
 including explicit sign.
R5 = if fixed format, number of places to the right of decimal point,
 And including decimal point

If fixed format, with exponent, R4,R5 exclude the 3 places for an exponent.

Results

R0 = Error Code
R1 Pointer = String

Parameter Description

Float1 8 byte representation in Radix 100 format.

String ASCII character text string.