Chapter 5 Floating-Point Operations

Although the MPC-2000 series can handle floating-point operations, in order to distinguish them from general control operations, floating-point operations are executed only by the independent FLOAT command and the following macro commands.

5-1 Floating-Point Arithmetic Macro Commands

Macro commands are commands used for position corrections in image processing. Because the AFFIN command can rotate two-dimensional vectors, it can efficiently perform coordinate corrections. For the details, see Command Reference.

AFFIN	Rotational conversion of point data
ATAN	Obtaining an angle with ATAN
ATAN2	Obtaining an angle with ATAN (General use)
COS	COS arithmetic
SIN	SIN arithmetic
TAN	TAN arithmetic
GETDG	Angle formed by a line connecting two points and the X axis

5-2 Floating-Point Arithmetic Operations

Double-precision array variable (FP(n))

FP(n) is a special array variable. Eight of them, FP(0)~FP(7), are prepared, and can be used as floating-point arithmetic compatible variables. Examples are shown below. When a formula which substitutes FP(n) is described, the formula becomes a floating-point arithmetic formula, and data are stored in FP(n) in a floating-point format unlike ordinary variables.

- Display a numerical value directly in the E format.

```
#FP(1)=10/3
#pr FP(1)
 3.333333E+00
#
```

- Multiply by 10000, convert into an integer, and display.

```
#FP(1)=10/3
#pr FP(10000,1)
 33333
#
```

- Calculation of Napier's constant

```
10 FLOAT   FP(2)=1
15 a=1
20 FOR   i=1 TO 100
30  a=a*i
40  FLOAT   FP(2)=FP(2)+1/a
50  NEXT
70  FORMAT   "0.0000"
80  PRINT   STR$(FP(10000,2))
#run

 2.7182
#
```

- Solution of a quadratic equation x2 + 4x + 3 = 0

```
10     a=1 :  b=4 :  c=3
20 FLOAT   FP(0)=(SQR(b*b-(4*a*c))-b)/2/a
30 FLOAT   FP(1)=(SQR(b*b-(4*a*c))*-1-b)/2/a
40 PRINT   FP(10000,0) FP(10000,1)
#run

 -10000 -30000
#
```

FLOAT command

MPC-2000 series arithmetic operations ordinarily deal with integers. In order to distinguish these operations, a FLOAT command is prepared. When FP(n) is specified as a substituted variable, the FLOAT command is automatically added to the arithmetic formula. Arithmetic operations in the FLOAT command prioritize multiplication (*) and division (/) over addition and subtraction in the same manner as in ordinary arithmetic formulas.
Examples:

[Example of a FLOAT command being added]

```
#10 fp(2)=1/3
list
10      FLOAT   FP(2)=1/3
#
```

[Example of prioritizing multiplication]

```
10      FLOAT   a=SQR(3*3+4*4)
20      PRINT   a
#run

 5
#
```

When a substituted variable is an ordinary integer variable, if a FLOAT command is added, the result is that although substitution occurs as an integer, the internal arithmetic becomes a floating-point operation. Along with this, the square root function SQR() also becomes a floating-point arithmetic operation.

```
10      FLOAT   a=SQR(3)*10000000
20      PRINT   a
#run

 17320508
#
```

Floating-point compatible functions

Arithmetic functions which can be used in a FLOAT command are as follows. These functions are regarded as double-precision floating-point functions in a FLOAT command. Their behavior differs from those in an ordinary integer arithmetic formulae.

SQR	Square root calculation	
SQ	Square calculation	
SIN	Trigonometric function SIN	Input is in radians.
COS	Trigonometric function COS	Input is in radians.
TAN	Trigonometric function TAN	Input is in radians.
ATAN	Trigonometric function ATAN	Output is in radians.
ACOS	Trigonometric function ACOS	Output is in radians.
RAD	Conversion from degree to radian	Output is in radians.
DEG	Conversion from radian to degree	Output is in degrees.
VAL	Obtain a character string as a floating-point value.	

Illustrated below is an example showing that the square sum of SIN and COS of an arbitrary angle integer i is 1.
Because the arguments of SIN and COS are in radians, an integer value of 100 degrees is converted to radians with an RAD() function.

```
10      FLOAT   a=SQR(SQ(SIN(RAD(i)))+SQ(COS(RAD(i))))*1000000
20      PRINT   a
#i=100
#run

 1000000
#
```

Further, π is calculated using ATAN.
Because TAN(45 degrees) = 1, ATAN(1) becomes π/4 in radians.

```
10      FLOAT   FP(0)=ATAN(1)*4
20      PRINT   FP(10000,0)
#run

 31415
#
```

In order to obtain the value of p itself, the following is a short-cut.

```
10      FLOAT   FP(5)=RAD(180)
20      PRINT   FP(5)
#run

 3.141593E+00
#
```

In actual applications, there are cases where floating-point numbers in EXP expression from external equipment are incorporated. For this, the VAL function is used.
The VAL function becomes a floating-point function in a FLOAT command and can read character strings of a type, ±X.XXXXXXE(e)YYY

```
10     a$="C41$=Mx+9.7042e+002 C42$=My-6.3210e+002 "
20      FLOAT   a=VAL(a$) FP(0)=VAL(0) b=VAL(0) FP(1)=VAL(0)
30      PRINT   a FP(0) b FP(1)
#run

 41 9.704200E+02 42 -6.321000E+02
#
```

For conversion of FP(n) into a character string, FP$(n) is used.
Illustrated below is an expression of FP$(n) and the fixed-point format of its integer-converted value.

```
10      FLOAT   FP(5)=RAD(180)
20      PRINT   FP$(5)
25      FLOAT   A=FP(5)*1000000
30      FORMAT   "0.000000"
40      PRINT   STR$(A)
#run

 3.141593E+00
 3.141592
#
```

Speed of floating-point arithmetic operations

The speed of a floating-point arithmetic operation is evaluated as follows:

```
LIST
95      SYSCLK=0
100       FOR   i=1 TO 1000
110        FLOAT   FP(0)=DEG(ATAN(5/100))
120       NEXT
130       PRINT   SYSCLK
140       PRINT   FP(0)
#run 95
95-
 707
```

```
 2.862405E+00
#
```

In this example, one ATAN calculation and the conversion into an angle took 707 m seconds.
Next, an example of complex arithmetic is executed.

```
LIST
10     SYSCLK=0
20      FOR   i_=1 TO 180
30       FLOAT   a_=SQR(SQ(SIN(RAD(i_)))+SQ(COS(RAD(i_))))
40       IF   a_!=1 THEN  : PRINT   "FL_NG " : END   : END_IF
50      NEXT
60      PRINT   SYSCLK
#run

 145
#
```

The complex arithmetic took 145/180 = 0.806 → 806 μ seconds.
 * 66/180 = 0.366 → 366 μ seconds in MPC-2200 case.
Because complicated floating-point operations take a long time,they should beused in complex arithmetic applies an application after evaluating the time it takes.