API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
20	D	ADD	P	S_1 S_2 D	Addition				

Type / OP	Bit Devices				Word devices										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F
S_1					*	*	*	*	*	*	*	*	*	*	*
S_2					*	*	*	*	*	*	*	*	*	*	*
D								*	*	*	*	*	*	*	*

Program Steps

ADD, ADDP: 7 steps

DADD, DADDP: 13 steps

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

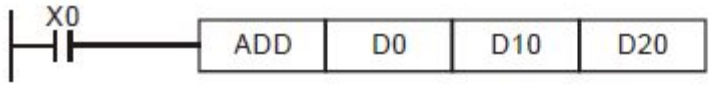
Operands:

S_1: Summand **S_2**: Addend **D**: Sum

Program Example 1:

In 16-bit BIN addition:

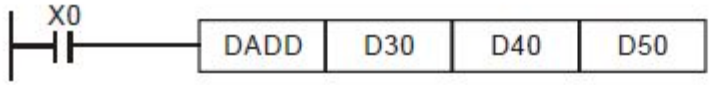
When X0 = ON, the content in D0 will plus the content in D10 and the sum will be stored in D20.



Program Example 2:

In 32-bit BIN addition:

When X0 = ON, the content in (D31, D30) will plus the content in (D41, D40) and the sum will be stored in (D51, D50). D30, D40 and D50 are low word; D31, D41 and D51 are high word



(D31, D30) + (D41, D40) = (D51, D50)

Operation of flags:

16-bit instruction:

1. If the operation result is "0", the zero flag M1020 will be ON.
2. If the operation result exceeds -32,768, the borrow flag M1021 will be ON.
3. If the operation result exceeds 32,767, the carry flag M1022 will be ON.

32-bit instruction:

1. If the operation result is "0", the zero flag, M1020 will be ON.
2. If the operation result exceeds -2,147,483,648, the borrow flag M1021 will be ON.
3. If the operation result exceeds 2,147,483,647, the carry flag M1022 will be ON

16-bit instruction:

API	Mnemonic			Operands	Function	Controllers			
21	D	SUB	P	(S1) (S2) (D)	Subtraction	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	SUB, SUBP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DSUB, DSUBP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	
D								*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

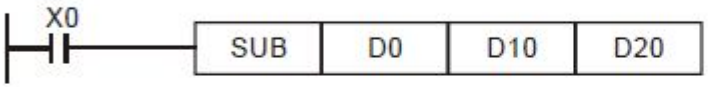
Operands:

S_1: Minuend **S_2**: Subtrahend **D**: Remainder

Program Example 1:

In 16-bit BIN subtraction:

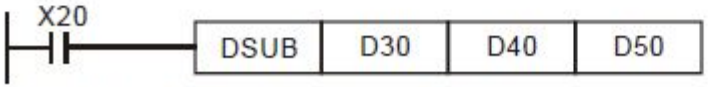
When X0 = ON, the content in D0 will minus the content in D10 and the results will be stored in D20



Program Example 2:

In 32-bit BIN subtraction:

When X10 = ON, the content in (D31, D30) will minus the content in (D41, D40) and the results will be stored in (D51, D50). D30, D40 and D50 are low word; D31, D41 and D51 are high word



(D31, D30) – (D41, D40) = (D51, D50)

API	Mnemonic			Operands	Function	Controllers			
22	D	MUL	P	S_1 S_2 D	Multiplication	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	MUL, DMULP: 7 steps
S_1					*	*	*	*	*	*	*	*	*	*		DMUL, DMULP: 13 steps
S_2					*	*	*	*	*	*	*	*	*	*		
D								*	*	*	*	*	*	*		

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

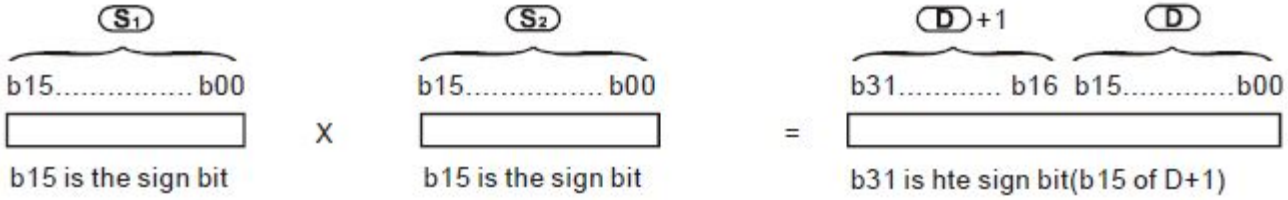
S_1: Multiplicand **S_2**: Multiplicator **D**: Product

Program Example:

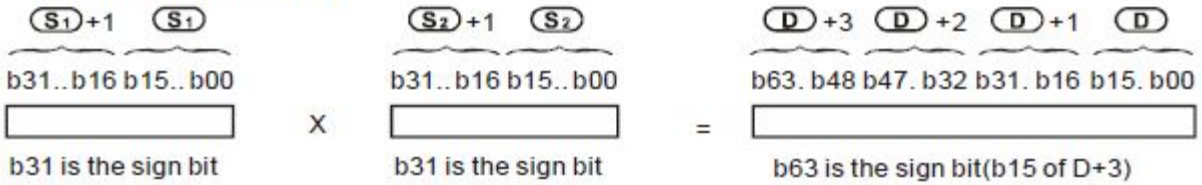
The 16-bit D0 is multiplied by the 16-bit D10 and brings forth a 32-bit product. The higher 16 bits are stored in D21 and the lower 16-bit are stored in D20. ON/OFF of MSB indicates the positive/negative status of the operation result.

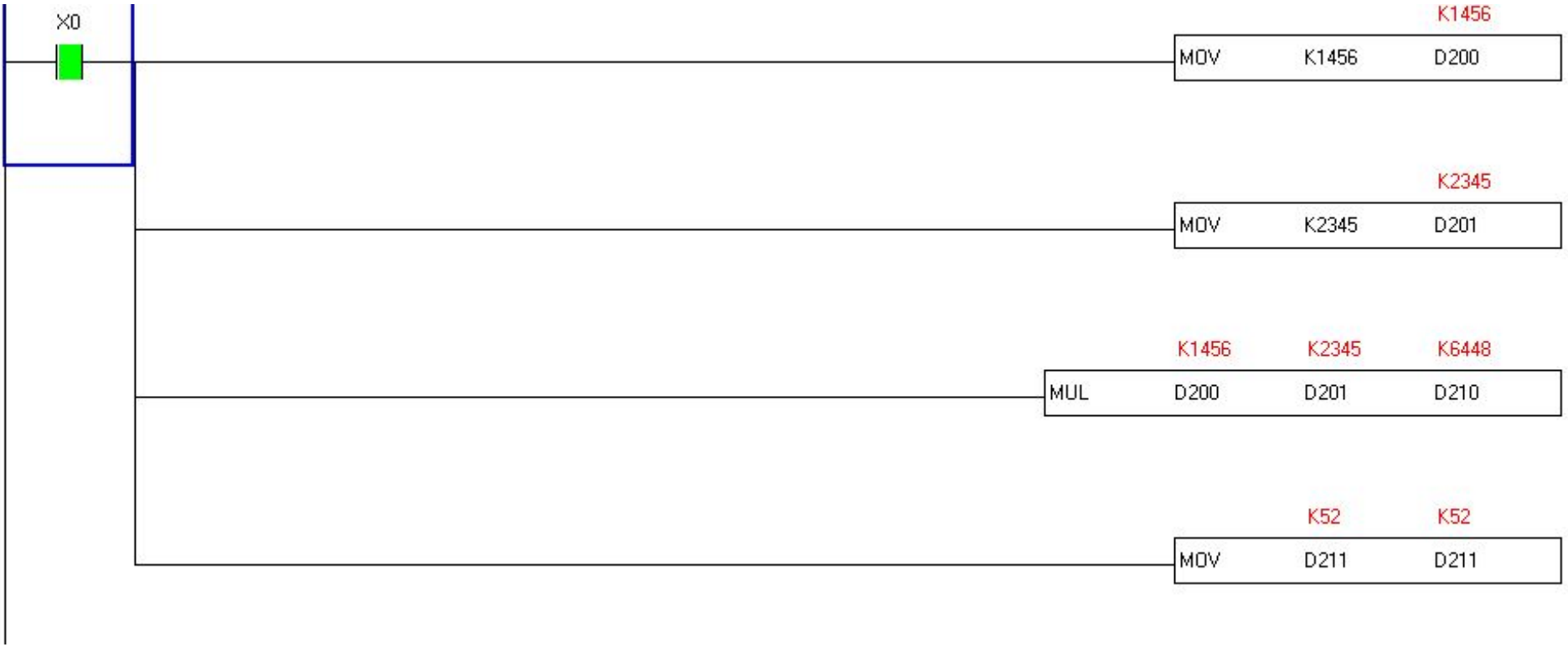
(D0) × (D10) = (D21, D20)

16-bit × 16-bit = 32-bit



32-bit BIN multiplication





1456 * 2345 = 341 4320

Result : (D211)*(2^16) + D210

: (52) *65536 + 6448

3407872 + 6448 =3414320

API	Mnemonic			Operands	Function	Controllers			
23	D	DIV	P	S_1 S_2 D	Division	ES2/EX2	SS2	SA2 SE	SX2

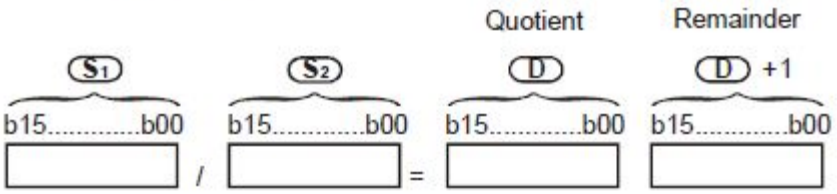
Type / OP	Bit Devices				Word devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*	*	*	*	*	*	*	*	*		DIV, DIVP: 7 steps
S_2					*	*	*	*	*	*	*	*	*	*		DDIV, DDIVP: 13 steps
D								*	*	*	*	*	*	*		

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

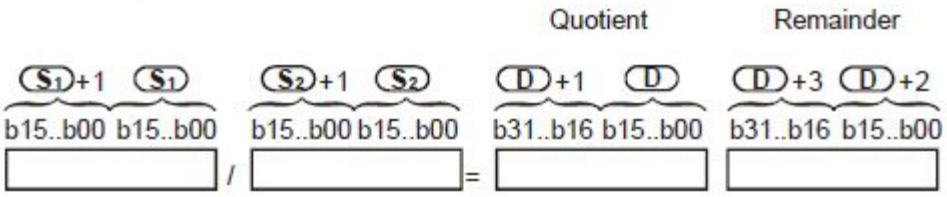
Operands:

S_1: Dividend **S_2**: Divisor **D**: Quotient and remainder

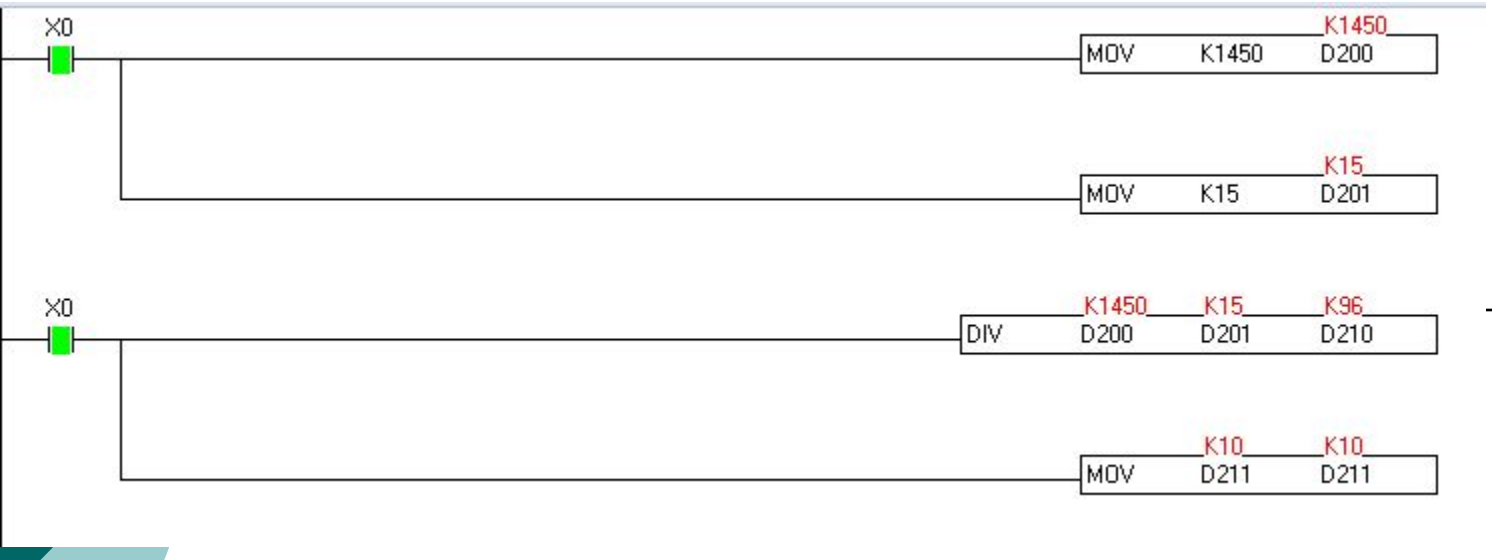
16-bit BIN division:



32-bit BIN division:



If D is specified with a bit device, it can designate K1 ~ K8 to store a 32-bit result. Users can use consecutive 2 32-bit registers to store the quotient and remainder.



96.6666 = 1450/15
Result : D210 + D211/15
96.666 = 10/15+ 96 :

Floating Point Operation

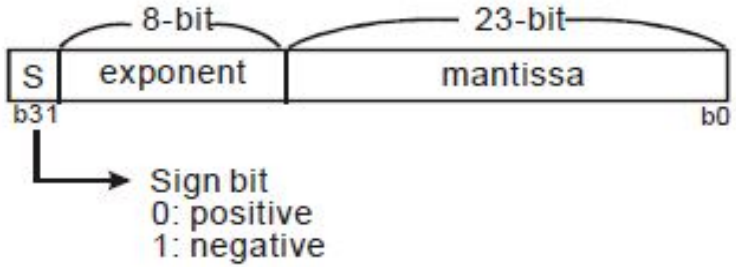
The operations in DVP-PLC are conducted in BIN integers. When the integer performs division, e.g. 40 ÷ 3 = 13, the remainder will be 1. When the integer performs square root operations, the decimal point will be left out. To obtain the operation result with decimal point, please use floating point instructions.

Application instructions revelant to floating point:

FLT	DECMP	DEZCP	DMOVR	DRAD
DDEG	DEBCD	DEBIN	DEADD	DESUB
DEMUL	DEDIV	DEXP	DLN	DLOG
DESQR	DPOW	INT	DSIN	DCOS
DTAN	DASIN	DACOS	DATAN	DADDR
DSUBR	DMULR	DDIVR		

Binary Floating Point

DVP-PLC represents floating point value in 32 bits, following the IEEE754 standard:



Equation $(-1)^S \times 2^{E-B} \times 1.M; B = 127$

Therefore, the range of 32-bit floating point value is from $\pm 2^{-126}$ to $\pm 2^{+128}$, i.e. from $\pm 1.1755 \times 10^{-38}$ to $\pm 3.4028 \times 10^{+38}$.

Example 1: Represent "23" in 32-bit floating point value

Step 1: Convert "23" into a binary value: 23.0 = 10111

Step 2: Normalize the binary value: $10111 = 1.0111 \times 2^4$, in which 0111 is mantissa and 4 is exponent.

Step 3: Obtain the exponent: ∵ E – B = 4 → E – 127 = 4 ∴ $E = 131 = 10000011_2$

Step 4: Combine the sign bit, exponent and mantissa into a floating point

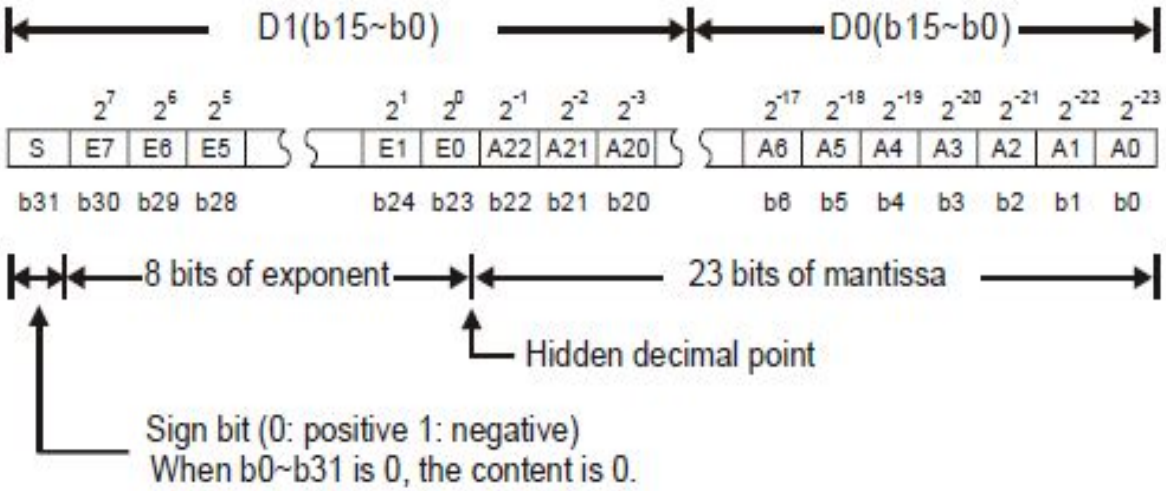
$$0\ 100\mathbf{0001}1\ 011\mathbf{1000}0000\mathbf{0000}0000\mathbf{0000}_2 = 41B80000_{16}$$

Example 2: Represent "-23.0" in 32-bit floating point value

The steps required are the same as those in Example 1 and only differs in modifying the sign bit into "1".

$$1\ 100\mathbf{0001}1\ 011\mathbf{1000}0000\mathbf{0000}0000\mathbf{0000}_2=C1B80000_{16}$$

DVP-PLC uses registers of 2 continuous No. to store a 32-bit floating point value. For example, we use registers (D1, D0) for storing a binary floating point value as below:



Decimal Floating Point

- Since the binary floating point value is not very user-friendly, we can convert it into a decimal floating point value for use. However, please note that the floating point operation in DVP-PLC is still operated in binary floating point format.

- The decimal floating point is represented by 2 continuous registers. The register of smaller number is for the constant while the register of bigger number is for the exponent.

Example: Store a decimal floating point in registers (D1, D0)

Decimal floating point = [constant D0] × $10^{[\text{exponent D1}]}$

Constant D0 = ±1,000 ~ ±9,999

Exponent D1 = -41 ~ +35

- The decimal floating point can be used in the following instructions:

D EBCD: Convert binary floating point to decimal floating point

D EBIN: Convert decimal floating point to binary floating point

- Zero flag (M1020), borrow flag (M1021), carry flag (M1022) and the floating point operation instruction

Zero flag: M1020 = On if the operational result is "0".

Borrow flag: M1021 = On if the operational result exceeds the minimum unit.

Carry flag: M1022 = On if the absolute value of the operational result exceeds the range of use.

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SA2 SE	SX2
49	D FLT P	(S) (D)	Floating Point				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			FLT, FLTP: 5 steps
D													*			DFLT, DFLTP: 9 steps

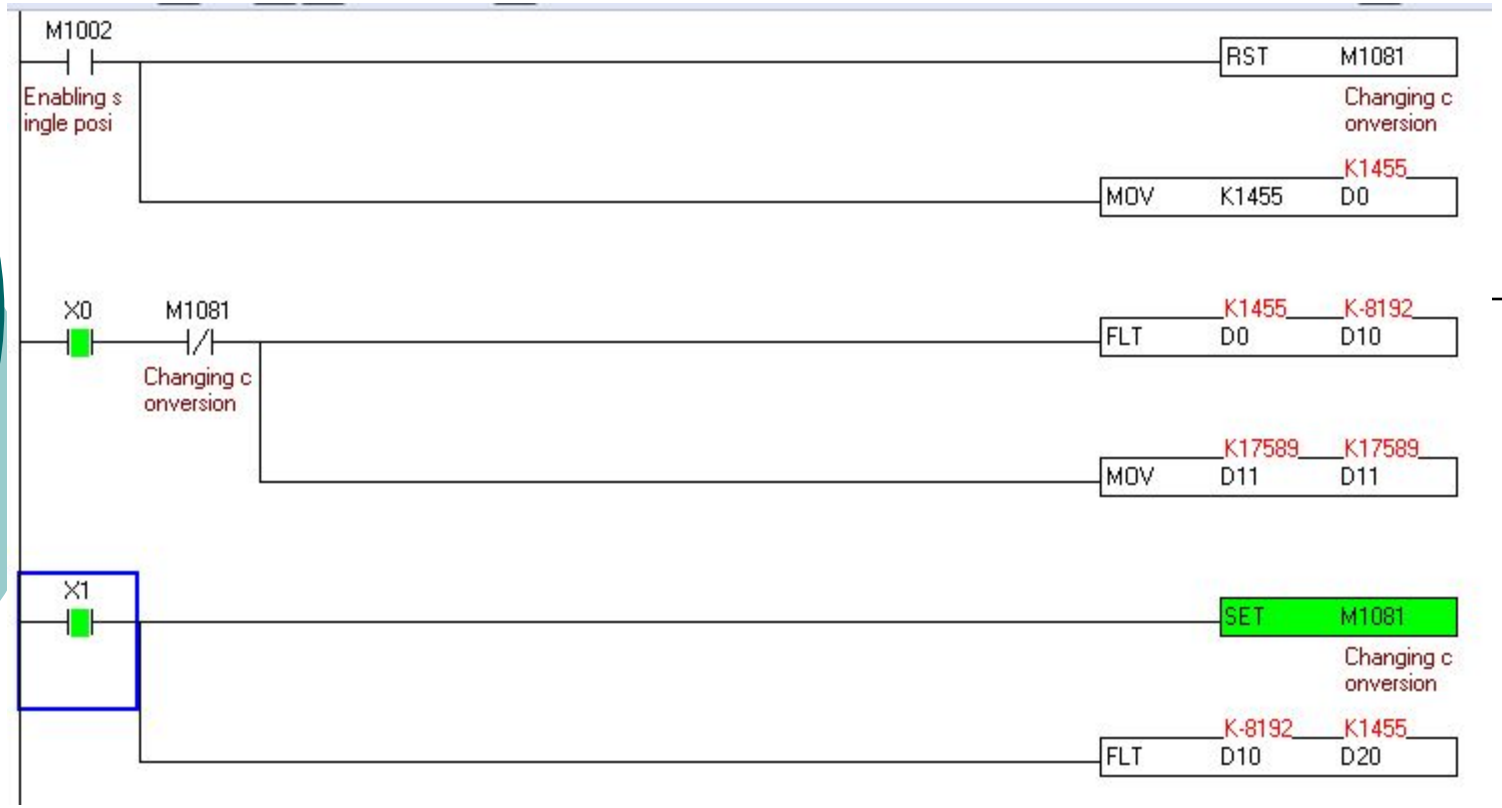
PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Device for storing the conversion result

Explanations:

1. When M1081 = OFF, the source **S** is converted from BIN integer to binary floating point value. At this time, 16-bit instruction FLT occupies 1 register for **S** and 2 registers for **D**.
   a) If the absolute value of the conversion result > max. floating value, carry flag M1022 = ON.
   b) If the absolute value of the conversion result < min. floating value, carry flag M1021 = ON.
   c) If conversion result is 0, zero flag M1020 = ON.
2. When M1081 is ON, the source **S** is converted from binary floating point value to BIN integer. (Decimal ignored). At this time, 16-bit instruction FLT occupies 2 registers for **S** and 1 register for **D**. The operation is same as instruction INT.
   a) If the conversion result exceeds the available range of BIN integer in D (for 16-bit: -32,768 ~ 32,767; for 32-bit: -2,147,483,648 ~ 2,147,483,647), **D** will obtain the maximum or minimum value and carry flag M1022 = ON.
   b) If the decimal is ignored, borrow flag M1021=ON.
   c) If the conversion result = 0, zero flag M1020=ON.
   d) After the conversion, D stores the result in 16 bits.



D11 = 17589 = 0100 0100 1011 0101
D10 = - 8192 = 1110 0000 0000 0000
D11,D10 =0100 0100 1011 0101 1110 0000 0000 0000
+,S = 0
E = 1000 1001 =137 -127 = 10
M=1.01101011110000000000
M *2^10 = 1011 0101 111 .00000 =1455 = 1455

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
118	D	EBCD	P	(S) (D)	Float to scientific conversion				

OP \ Type	Bit Devices				Word devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			DEBCD, DEBCDP: 9 steps
D													*			

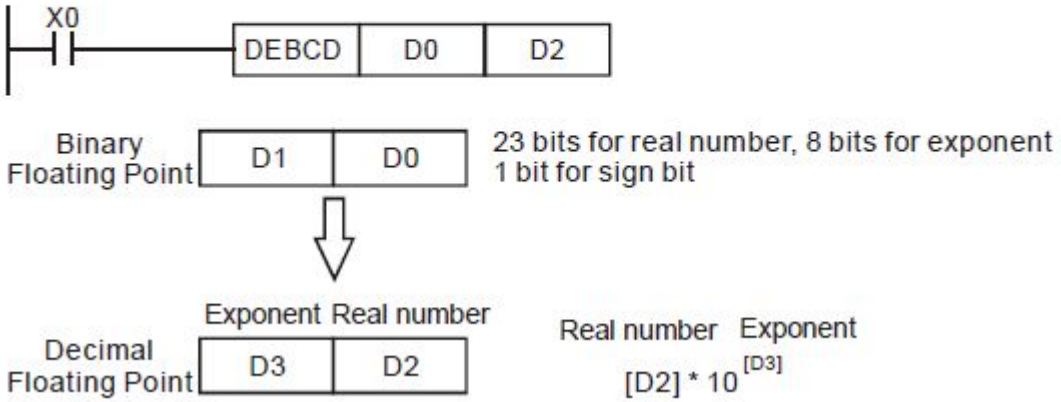
PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Conversion result

Program Example:

When X0 = ON, the binary floating point value in D1, D0 will be converted to decimal floating point and the conversion result is stored in D3, D2.



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
119	D	EBIN	P	(S) (D)	Scientific to float conversion				

OP \ Type	Bit Devices				Word devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S													*			DEBIN, DEBINP: 9 steps
D													*			

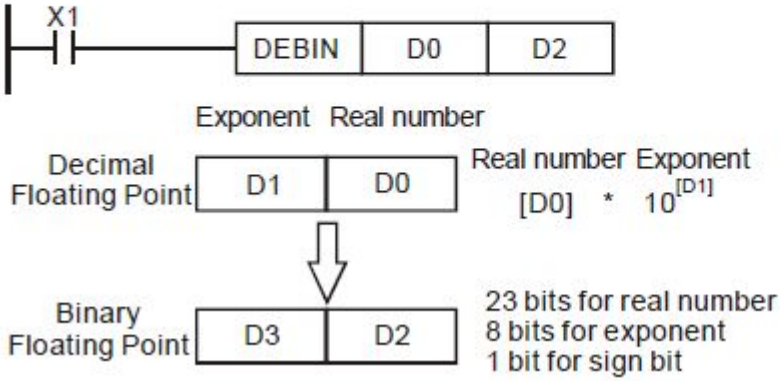
PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S: Source device **D**: Conversion result

Program Example 1:

When X1 = ON, the decimal floating point value in (D1, D0) will be converted to binary floating point and the conversion result is stored in (D3, D2).



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
120	D	EADD	P	S_1 S_2 D	Floating point addition				

Type / OP	Bit Devices				Word devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DEADD, DEADDP: 13 steps
S_2					*	*							*			
D													*			

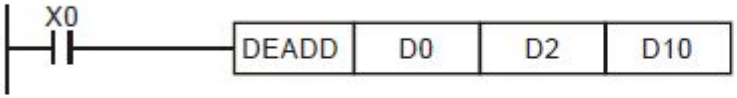
PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Augend **S_2**: Addend **D**: Addition result

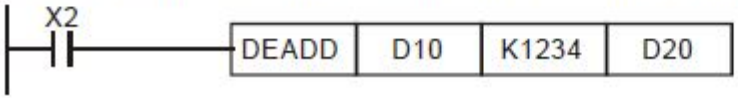
Program Example 1:

When X0 = ON, add the binary floating point value (D1, D0) with binary floating point value (D3, D2) and store the result in (D11, D10).



Program Example 2:

When X2 = ON, add the binary floating point value of (D11, D10) with K1234 (automatically converted to binary floating point value) and store the result in (D21, D20).



API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
121	D	ESUB	P	S_1 S_2 D	Floating point subtraction				

Type / OP	Bit Devices				Word devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DESUB, DESUBP: 13 steps
S_2					*	*							*			
D													*			

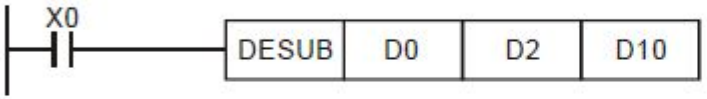
PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Minuend **S_2**: Subtrahend **D**: Subtraction result

Program Example 1:

When X0 = ON, binary floating point value (D1, D0) minuses binary floating point value (D3, D2) and the result is stored in (D11, D10).



Program Example 2:

When X2 = ON, K1234 (automatically converted into binary floating point value) minuses binary floating point (D1, D0) and the result is stored in (D11, D10).

API	Mnemonic			Operands			Function	Controllers			
122	D	EMUL	P	S_1	S_2	D	Floating point multiplication	ES2/EX2	SS2	SA2 SE	SX2

OP \ Type	Bit Devices				Word devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DEMUL, DEMULP: 13 steps
S_2					*	*							*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Multiplicand **S_2**: Multiplicator **D**: Multiplication result

Program Example 1:

When X1 = ON, binary floating point (D1, D0) multiplies binary floating point (D11, D10) and the result is stored in (D21, D20).

Program Example 2:

When X2 = ON, K1234 (automatically converted into binary floating point value) multiplies binary floating point (D1, D0) and the result is stored in (D11, D10).

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
123	D	EDIV	P	S_1 S_2 D	Floating point division				

Type / OP	Bit Devices				Word devices										Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1					*	*							*			DEADD, DEADDP: 13 steps
S_2					*	*							*			
D													*			

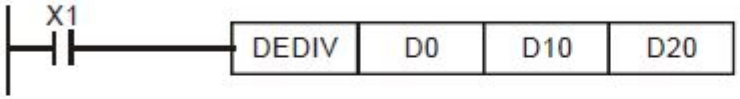
PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Dividend **S_2**: Divisor **D**: Quotient and Remainder

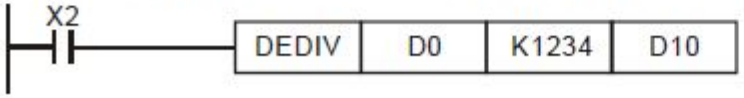
Program Example 1:

When X1 = ON, binary floating point value of (D1, D0) is divided by binary floating point (D11, D10) and the quotient and remainder is stored in (D21, D20).

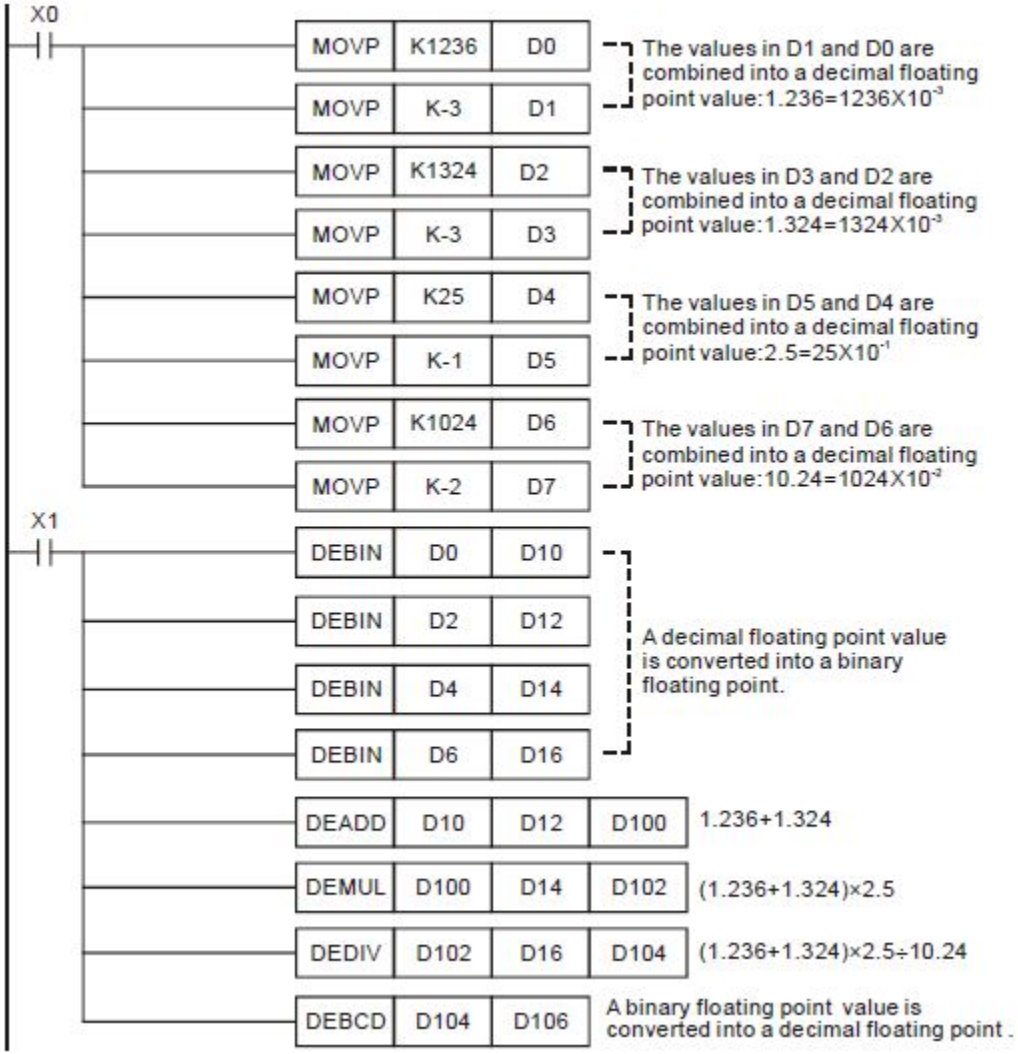


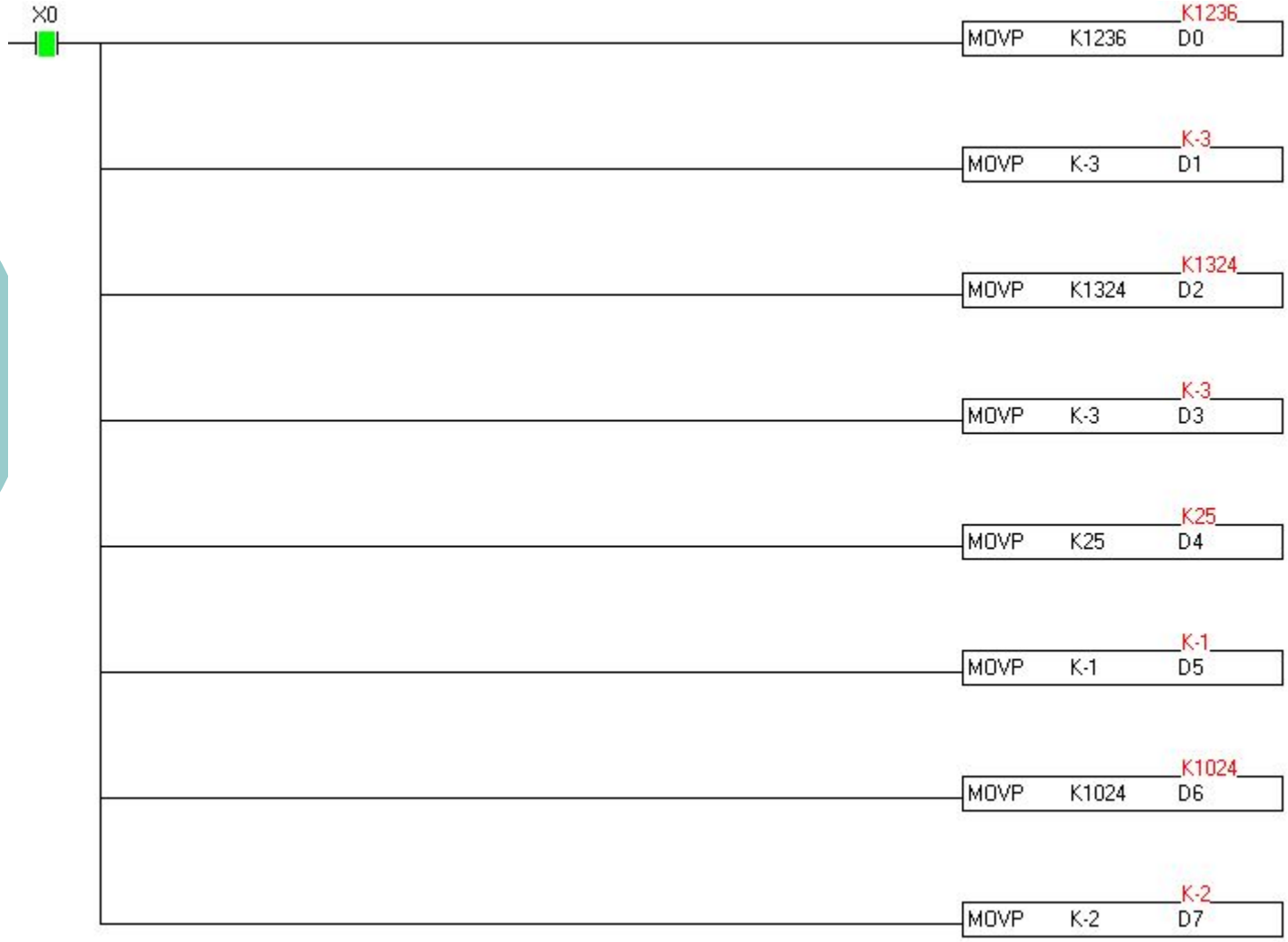
Program Example 2:

When X2 = ON, binary floating point value of (D1, D0) is divided by K1234 (automatically converted to binary floating point value) and the result is stored in (D11, D10).



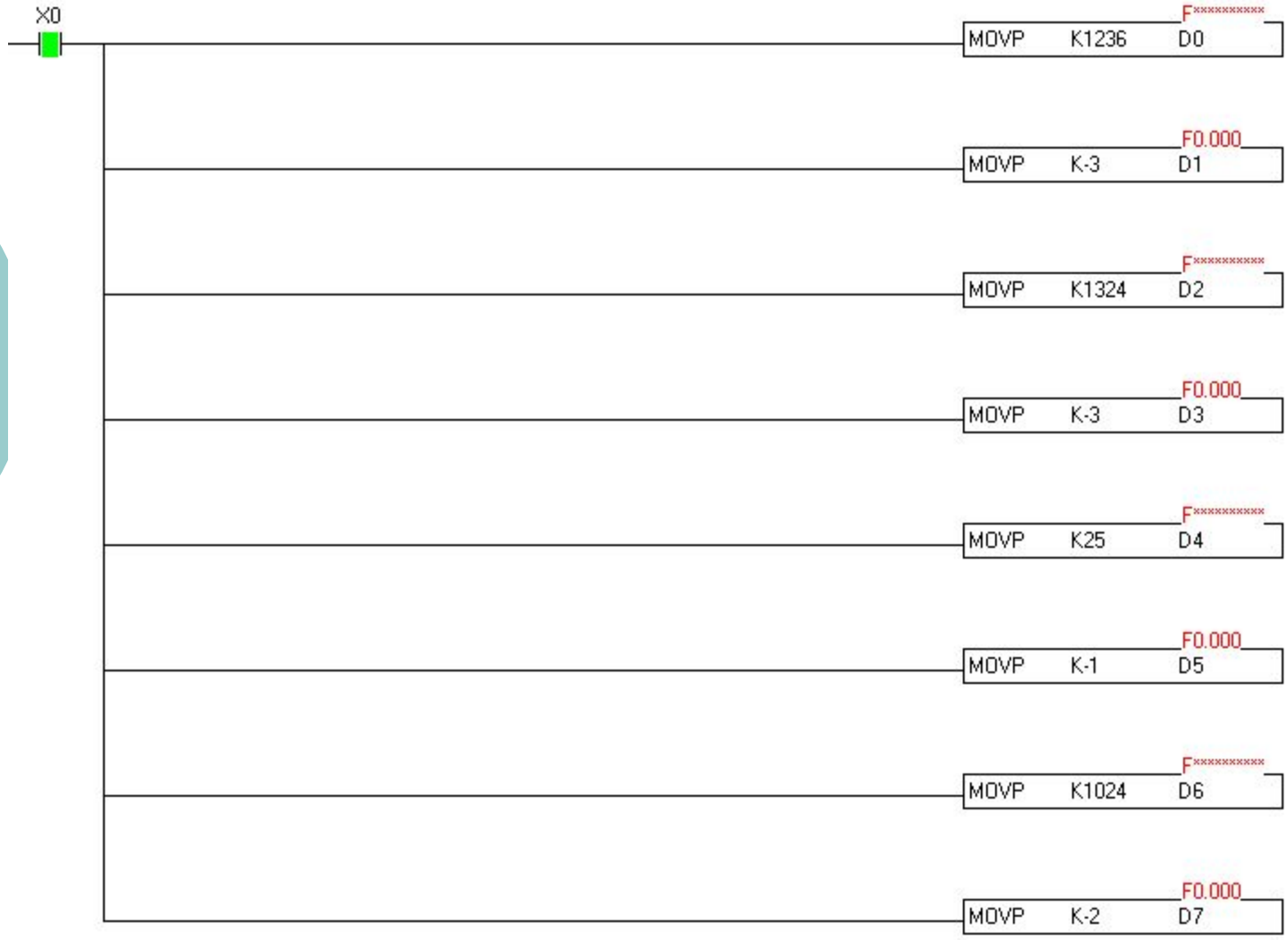
- Perform the operation (1.236+1.324)×2.5÷10.24 by Delta's binary floating point operation instruction.



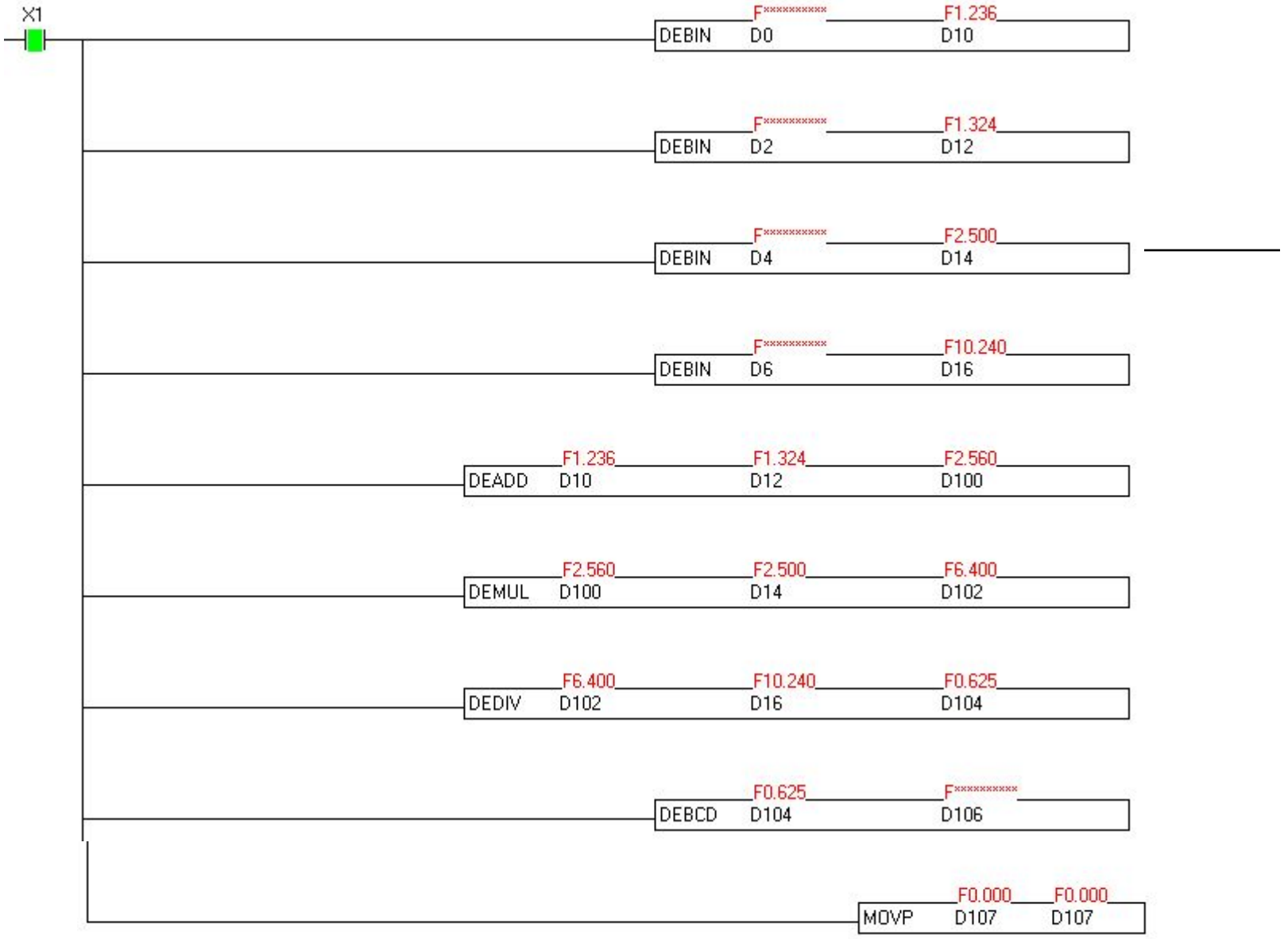


Monitor Format Signed Decimal

Monitor Format Signed Decimal



Monitor Format Float



Monitor Format Floagt

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
175	D	DIVR	P	S_1 S_2 D	Floating point division				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DDIVR: 13 steps
S_1													*			
S_2													*			
D													*			

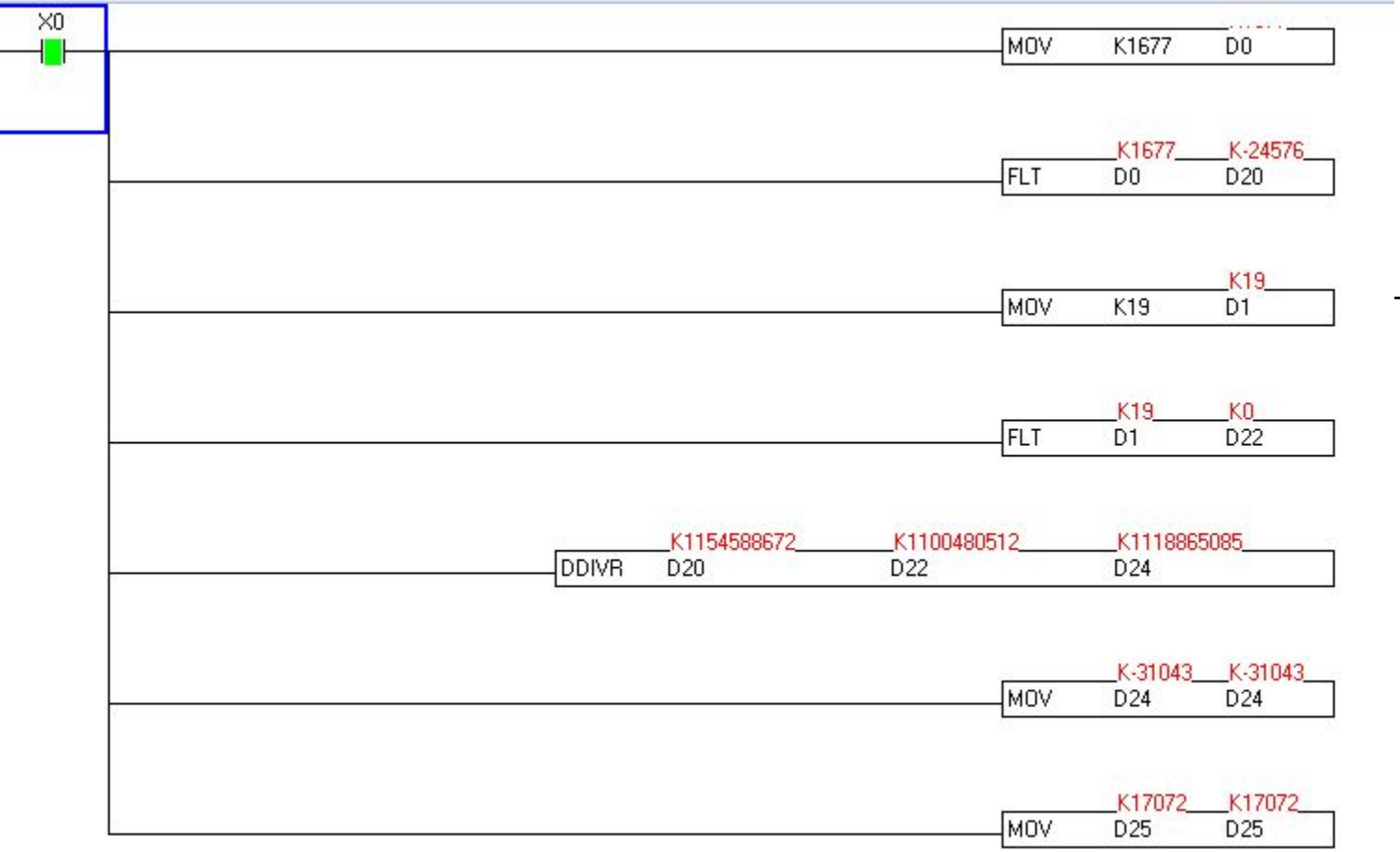
PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Floating point n dividend **S_2:** Floating point divisor **D:** Quotient

Explanations:

1. DIVR instruction divides **S_1** by **S_2** and stores the operation result in D
2. In DIVR instruction, floating point values can be directly entered into **S_1** and **S_2**.
3. In DDIVR instruction, floating point values (e.g. F1.2) can be either entered directly into **S_1** and **S_2** or stored in data registers for operation.
4. When **S_1** and **S_2** is specified as data registers, the function of DDIVR instruction is the same as API 123 EDIV instruction.
5. If **S_2** = 0, operation error occurs and M1067 = ON, M1068 = ON. D1067 stores the error code 0E19 (HEX).
6. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON.



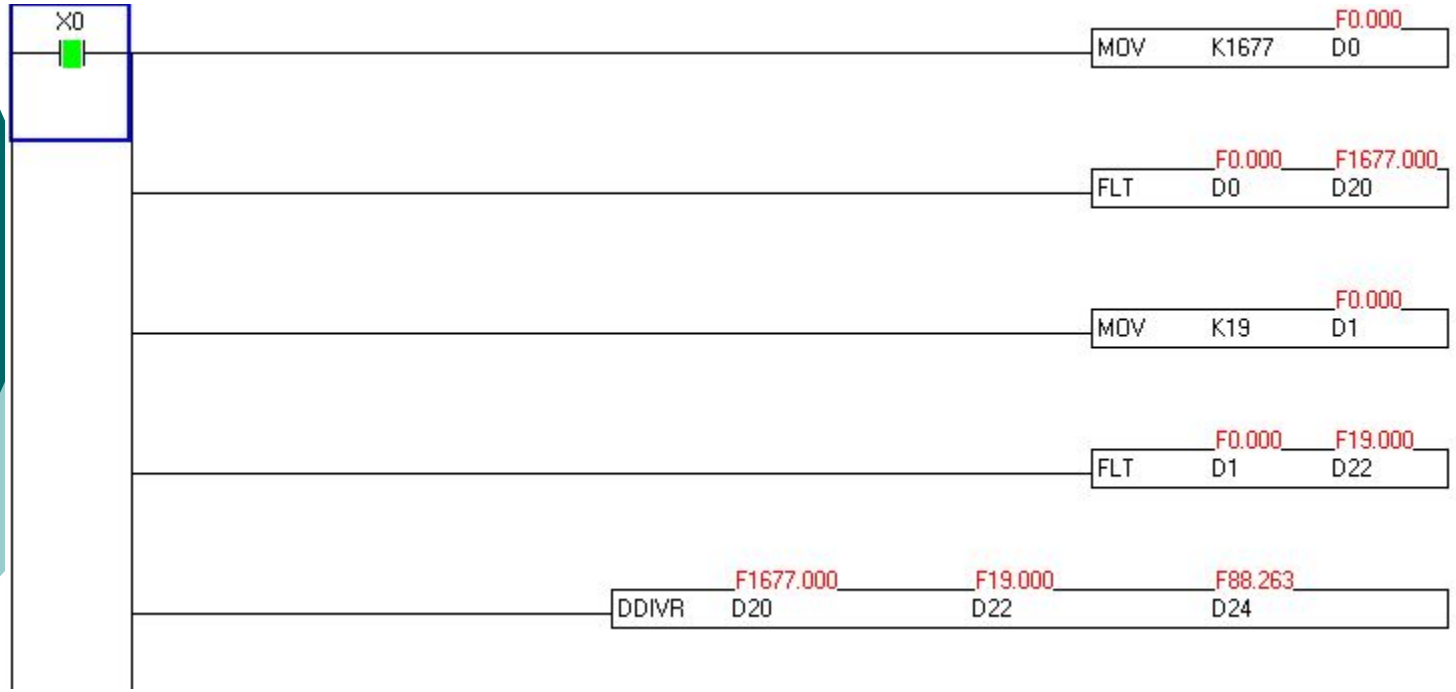
88.2631 = 1677/19

D25,D24 =0100 0010 1011 0000 , 1000 0110 1011 1101

E=10000101 = 133 -127 = 6

M = 1.01100010000110

N= M *2^6 = 1011000.10000110



By changing the monitoring data format to float In view

API	Mnemonic			Operands	Function	Controllers			
						ES2/EX2	SS2	SA2 SE	SX2
172	D	ADDR	P	(S1) (S2) (D)	Floating point addition				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	DADDR, DADDRP: 13 steps
S_1													*			
S_2													*			
D													*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

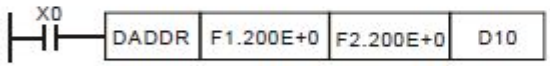
S_1: Floating point summand **S_2:** Floating point addend **D:** Sum

Explanations:

1. ADDR instruction adds the floating point summand **S_1** with floating point addend **S_2** and stores the operation result in D.
2. In ADDR instruction, floating point values can be directly entered into **S_1** and **S_2**.
3. In DADDR instruction, floating point values (e.g. F1.2) can be either entered directly into **S_1** and **S_2** or stored in data registers for operation.
4. When **S_1** and **S_2** is specified as data registers, the function of DADDR instruction is the same as API 120 EADD instruction.
5. **S_1** and **S_2** can designate the same register. In this case, if the instruction is specified as "continuous execution instruction" (generally DADDRP instruction) and the drive contact is ON, the register will be added once in every scan.
6. Flags: M1020 (Zero flag), M1021 (Borrow flag) and M1022 (Carry flag)
   If absolute value of the result exceeds max floating point value, carry flag M1022 = ON.
   If absolute value of the result is less than min. floating point value, borrow flag M1021 = ON.
   If the conversion result is 0, zero flag M1020 = ON

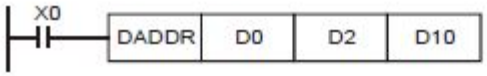
Program Example 1:

When X0 = ON, add floating point number F1.200E+0 (Input F1.2, and scientific notation F1.200E+0 will be displayed on ladder diagram. Users can set monitoring data format as float on the function View) with F2.200E+0 and store the obtained result F3.400E+0 in register D10 and D11.



Program example 2:

When X0 = ON, add floating point value (D1, D0) with (D3, D2) and store the result in (D11, D10).



Contact Type Compare Instruction:

224	LD=	Comparison contact is ON when S1 = S2 is true
225	LD>	Comparison contact is ON when S1 > S2 is true
226	LD<	Comparison contact is ON when S1 < S2 is true
228	LD<>	Comparison contact is ON when S1 ≠ S2 is true
229	LD<=	Comparison contact is ON when S1 ≦ S2 is true
230	LD>=	Comparison contact is ON when S1 ≧ S2 is true
232	AND=	Comparison contact is ON when S1 = S2 is true
233	AND>	Comparison contact is ON when S1 > S2 is true
234	AND<	Comparison contact is ON when S1 < S2 is true
236	AND<>	Comparison contact is ON when S1 ≠ S2 is true
237	AND<=	Comparison contact is ON when S1 ≦ S2 is true
238	AND>=	Comparison contact is ON when S1 ≧ S2 is true
240	OR=	Comparison contact is ON when S1 = S2 is true
241	OR>	Comparison contact is ON when S1 > S2 is true
242	OR<	Comparison contact is ON when S1 < S2 is true
244	OR<>	Comparison contact is ON when S1 ≠ S2 is true
245	OR<=	Comparison contact is ON when S1 ≦ S2 is true
246	OR>=	Comparison contact is ON when S1 ≧ S2 is true

API	Mnemonic		Operands	Function	Controllers			
232~238	D	AND※	(S1) (S2)	Serial Type Comparison	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	AND※: 5 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DAND※: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

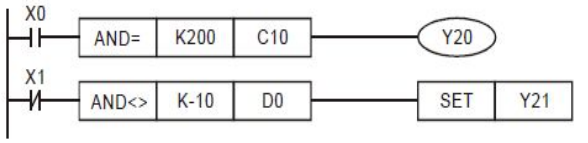
Explanations:

1. This instruction compares the content in **S_1** and **S_2**. Take API232 (AND =) for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
2. AND※ (※: =, >, <, <>, ≤, ≥) instruction is used for serial connection with contacts.

API No.	16 -bit instruction	32 -bit instruction	Continuity condition	Discontinuity condition
232	AND=	DAND=	$S_1 = S_2$	$S_1 \neq S_2$
233	AND>	DAND>	$S_1 > S_2$	$S_1 \leqq S_2$
234	AND<	DAND<	$S_1 < S_2$	$S_1 \geqq S_2$
236	AND<>	DAND<>	$S_1 \neq S_2$	$S_1 = S_2$
237	AND<=	DAND<=	$S_1 \leqq S_2$	$S_1 > S_2$
238	AND>=	DAND>=	$S_1 \geqq S_2$	$S_1 < S_2$

Program Example:

1. When X0 = ON, and the content in C10 = K200, Y20 = ON
2. When X1 = OFF and the content in D0 ≠ K-10, Y21= ON and latched.



3. When the MSB (16-bit instruction: b15, 32-bit instruction: b31) of **S_1** and **S_2** is 1, the comparison value will be viewed as a negative value in comparison.
4. When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DAND※). If 16-bit instruction (AND※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash.

API	Mnemonic		Operands	Function	Controllers			
240~246	D	OR※	S_1 S_2	Parallel Type Comparison	ES2/EX2	SS2	SA2 SE	SX2

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	OR※: 5 steps
S_1					*	*	*	*	*	*	*	*	*	*	*	DOR※: 9 steps
S_2					*	*	*	*	*	*	*	*	*	*	*	

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

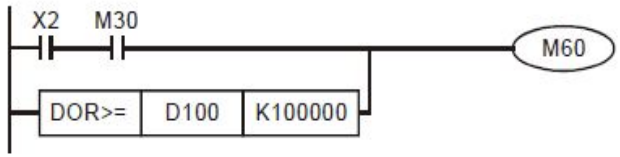
Operands:

S_1: Source device 1 **S_2**: Source device 2

Explanations:

1. This instruction compares the content in **S_1** and **S_2**. Take API240 (OR =) for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled
2. OR※ (※: =, >, <, <>, ≤, ≥) instruction is used for parallel connection with contacts.

API No.	16-bit instruction	32-bit instruction	Continuity condition	Discontinuity condition
240	OR=	DOR=	$S_1 = S_2$	$S_1 \neq S_2$
241	OR>	DOR>	$S_1 > S_2$	$S_1 \leq S_2$
242	OR<	DOR<	$S_1 < S_2$	$S_1 \geq S_2$
244	OR<>	DOR<>	$S_1 \neq S_2$	$S_1 = S_2$
245	OR<=	DOR<=	$S_1 \leq S_2$	$S_1 > S_2$
246	OR>=	DOR>=	$S_1 \geq S_2$	$S_1 < S_2$



3. When the MSB (16-bit instruction: b15, 32-bit instruction: b31) of **S_1** and **S_2** is 1, the comparison value will be viewed as a negative value in comparison..
4. When 32-bit counters (C200 ~ C254) are used in this instruction, make sure to adopt 32-bit instruction (DOR※). If 16-bit instruction (OR※) is adopted, a "program error" will occur and the ERROR indicator on the MPU panel will flash

Floating Point Contact Type Comparison:

275	FLD=	Floating Point Contact Type Comparison
276	FLD>	Floating Point Contact Type Comparison
277	FLD<	Floating Point Contact Type Comparison
278	FLD<>	Floating Point Contact Type Comparison
279	FLD<=	Floating Point Contact Type Comparison
280	FLD>=	Floating Point Contact Type Comparison
281	FAND=	Floating Point Serial Type Comparison
282	FAND>	Floating Point Serial Type Comparison
283	FAND<	Floating Point Serial Type Comparison
284	FAND<>	Floating Point Serial Type Comparison
285	FAND<=	Floating Point Serial Type Comparison
286	FAND>=	Floating Point Serial Type Comparison
287	FOR=	Floating Point Parallel Type Comparison
288	FOR>	Floating Point Parallel Type Comparison
289	FOR<	Floating Point Parallel Type Comparison
290	FOR<>	Floating Point Parallel Type Comparison
291	FOR<=	Floating Point Parallel Type Comparison
292	FOR>=	Floating Point Parallel Type Comparison

API	Mnemonic	Operands	Function	Controllers			
				ES2/EX2	SS2	SA2 SE	SX2
275~280	FLD※	(S1) (S2)	Floating Point Contact Type Comparison LD※				

Type / OP	Bit Devices				Word devices											Program Steps
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	E	F	
S_1											*	*	*			FLD※: 9 steps
S_2											*	*	*			

PULSE				16-bit				32-bit			
ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2	ES2/EX2	SS2	SA2 SE	SX2

Operands:

S_1: Source device 1 **S_2**: Source device 2

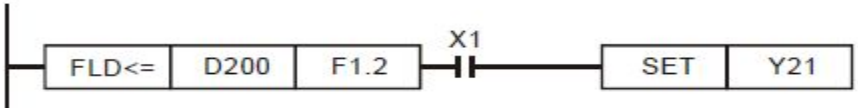
Explanations:

1. This instruction compares the content in **S_1** and **S_2**. Take "FLD=" for example, if the result is "=", the continuity of the instruction is enabled. If the result is "≠", the continuity of the instruction is disabled.
2. The user can specify the floating point value directly into operands **S_1** and **S_2** (e.g. F1.2) or store the floating point value in D registers for further operation.
3. FLD※ instruction is used for direct connection with left hand bus bar.

API No.	32 -bit instruction	Continuity condition	Discontinuity condition
275	FLD=	$S_1 = S_2$	$S_1 \neq S_2$
276	FLD>	$S_1 > S_2$	$S_1 \leq S_2$
277	FLD<	$S_1 < S_2$	$S_1 \geq S_2$
278	FLD<>	$S_1 \neq S_2$	$S_1 = S_2$
279	FLD<=	$S_1 \leq S_2$	$S_1 > S_2$
280	FLD>=	$S_1 \geq S_2$	$S_1 < S_2$

Program Example:

When the content in D200(D201) ≤ F1.2 and X1 is ON, Y21 = ON and latched.



H.W

A temp sensor with 0—1000 °C is connected to PLC with 12 Bits ADC
Writ PLC. program to set Alarm (Y0) when the temp is between 500.5 and 600.7
And run a ventilator when the temp above 700.45 (Y1)