

MITSUBISHI **PROGRAMMABLE CONTROLLER** **MELSEC-K**

Instruction Manual

Application Instructions and the practical use of MELSEC-KOJ

Al Baien personal copy

- CONTENTS -

1.	INTRODUCTION	1
2.	APPLICATION INSTRUCTIONS LIST	1
3.	DATA REGISTER ASSIGNMENT AND APPLICATION	3
4.	FUNCTIONS AND PRACTICAL USE OF APPLICATION INSTRUCTIONS	5
4.1	8-bit data association	5
4.1.1	Functions	5
4.1.2	Circuit applications	6
4.2	16-bit data dissociation	6
4.2.1	Functions	6
4.2.2	Circuit applications	7
4.3	16-bit data AND operation	7
4.3.1	Functions	7
4.3.2	Circuit applications	8
4.4	16-bit data OR operation	9
4.4.1	Functions	9
4.4.2	Circuit applications	9
4.5	Batch shift of temporary memory M	9
4.5.1	Functions	10
4.5.2	Circuit applications	13
4.6	Batch shift of data register D	17
4.6.1	Functions	18
4.6.2	Circuit applications	21

4.7	Batch reset of data register D	24
4.7.1	Functions	24
4.7.2	Circuit applications	24
4.8	Indirect reading of T,C,D (Timer, Counter, Register D)	25
4.8.1	Functions	25
4.8.2	Circuit applications	26
4.9	Indirect writing of T,C,D	29
4.9.1	Functions	29
4.9.2	Circuit applications	30
4.10	Data transference from Y to D	32
4.10.1	Functions	32
4.10.2	Circuit applications	35
4.11	4↔16 Decode/encode	37
4.11.1	Functions	37
4.11.2	Circuit applications	40
4.12	16-bit check	42
4.12.1	Functions	42
4.12.2	Circuit applicatins	44
4.13	Data inversion	46
4.13.1	Functions	46
4.13.2	Circuit applications	46
4.14	Application instruction executing time	47
5.	FUNCTIONS AND PRACTICAL USE OF HIGH-SPEED PROCESSING INSTRUCTIONS		
5.1	High-speed processing instructions and application data registers	48

5.2	Circuit applicatins	48
5.3	Program for 10mS high-precision timer	51
5.4	Programming error display	54

1. INTRODUCTION

MELSEC-K0J is provided with, besides the standard instruction set consisting of 18 control instructions and 8 data instructions, all of which are available in the standard K0, K1 or K2 CPU, additional 15 application instructions.

The application instruction set includes 13 arithmetic operation instructions and 2 high-speed program processing instructions.

This manual describes the functions and practical use of the application instructions.

(For the standard instruction set, refer to "MELSEC-K Programming Manual".)

2. APPLICATION INSTRUCTIONS LIST

No.	Category of instruction	Name	Instruction	Data register used
1	Arithmetic operation instruction	8-bit data association (register pairing)	OUT F110	2 wds., D110, D111
2		16-bit data dissociation (register pairing off)	OUT F111	2 wds., D110, D111
3		16-bit data AND operation	OUT F112	2 wds., D110, D111
4		16-bit data OR operation	OUT F113	2 wds., D110, D111
5		Batch shift of M	OUT F114	3 wds., D110 ~ D112
6		Batch shift of D	OUT F115	3 wds., D110 ~ D112

No.	Category of instruction	Name	Instruction	Data register used
7		Batch reset of D	OUT F116	2 wds., D110, D111
8		Indirect reading of T, C, D	OUT F117	2 wds., D110, D111
9		Indirect writing of T, C, D	OUT F118	2 wds., D110, D111
10		Y→D Data transfer	OUT F119	2 wds., D110, D111
11		4↔16 Decode/encode	OUT F108	3 wds., D110 ~ D112
12		16-bit check	OUT F109	2 wds., D110, D111
13		Data inversion	OUT F100	1 wrd., D110
14	High-speed processing instruction	High-speed program call instruction	SET F126	2 wds., D126, D123
15		High-speed program return instruction	RST F126	

* All available peripheral equipment is applicable and program input/output are possible with PU or GPP.

3. DATA REGISTER ASSIGNMENT AND APPLICATION

No.	Name of function	Function No.	Data register					
			D110	*	D111	*	D112	*
1	8-bit data association	F110	Lower 8-bit D No.	I	Upper 8-bit D No.	I		
			Associated D No.					
2	16-bit data dissociation	F111	D No. before dissociation	I	Upper 8-bit D No.	I		
			Lower 8-bit D No.					
3	16-bit data AND	F112	D No.	I	D No.	I		
			Operated result D No.					
4	16-bit data OR	F113	D No.	I	D No.	I		
			Operated result D No.					
5	Batch shift of M	F114	Head No. of M	I	Number of bits	I	Shift direction	I
6	Batch shift of D	F115	Head No. of D	I	Number of registers	I	Shift direction	I
7	Batch reset of D	F116	Head No. of D	I	Number of registers	I		
8	Indirect reading of T,C,D	F117	T,C,D No.	I	Contents read	0		
9	Indirect writing of T,C,D	F118	T,C,D No.	I	Data written	I		
10	Y→D Data transfer	F119	Y No. and number of digits	I	D No. transferred	I		
11	4↔16 Decode/encode	F108	Data	I	Decode or encode	I	Encoded or decoded result	0

No.	Name of function	Function No.	Data register					
			D110	*	D111	*	D112	*
12	16-bit check	F109	Check data	I	Cumulated bits	0		
13	Data inversion	F100	D No.	I				
14	High-speed processing instruction	F126	D126		D126			
			High-speed program head step No.	I	High-speed program head step No.	I		

* "I" shows the data to be prepared before execution of application instruction.

"0" shows the resultant data after execution of application instruction.

NOTE: D No. shown in the above list is data register No. (0 - 95) for operand. Arithmetic operation is actually performed with the specified data register.

4. FUNCTIONS AND PRACTICAL USE OF APPLICATION INSTRUCTIONS

4.1 8-bit data association

When AND or OR operation is performed with 16 bit data in the standard K series CPU, only plus integer ranging from 0 to 9999 can be specified.

When the additional function is used, however, 16-bit data can be formed as follows:

4.1.1 Functions

Function No.: F110

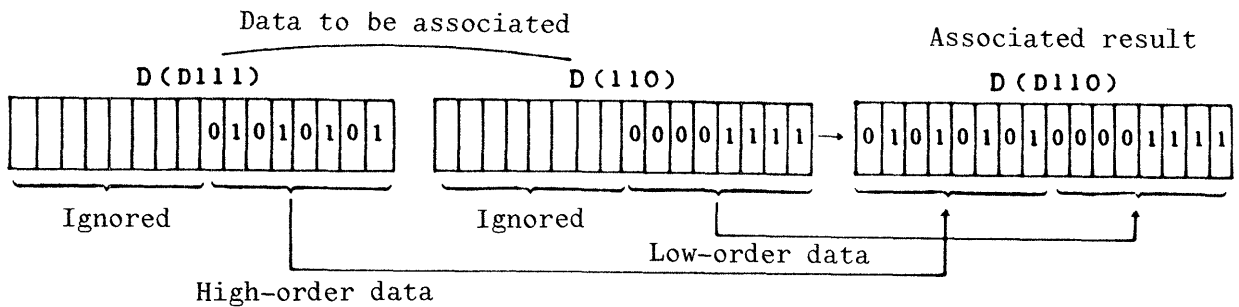


Fig. 4.1 Data association

NOTE: D(D110) indicates the contents of the data register specified by D110.

- (1) When D No. storing lower (low-order) 8-bit data to be added with counterpart and other D No. storing higher 8-bit data are set in D110 and D111 respectively, and F111 is executed, the resultant 16-bit data is placed in D (D110).
- (2) The contents in D (D111) do not change.
- (3) The data to which another data is associated, may be either BCD 2-digit data, or binary 8-bit data.

(4) When the newly formed decimal 16-bit data is larger than 9999 and BCD instruction is executed, "RUN" display flickers. No substantial problem is caused by data exceeding 9999 as far as BCD instruction is not executed.

However, monitor data by PU or GPP cannot be normally displayed.

4.1.2 Circuit applications.

Refer to Fig. 4.5.

4.2 16-bit data dissociation

The result from AND or OR with 16-bit data is divided into one pair of 8-bit data when this function is used.

4.2.1 Functions

Function No.: F111

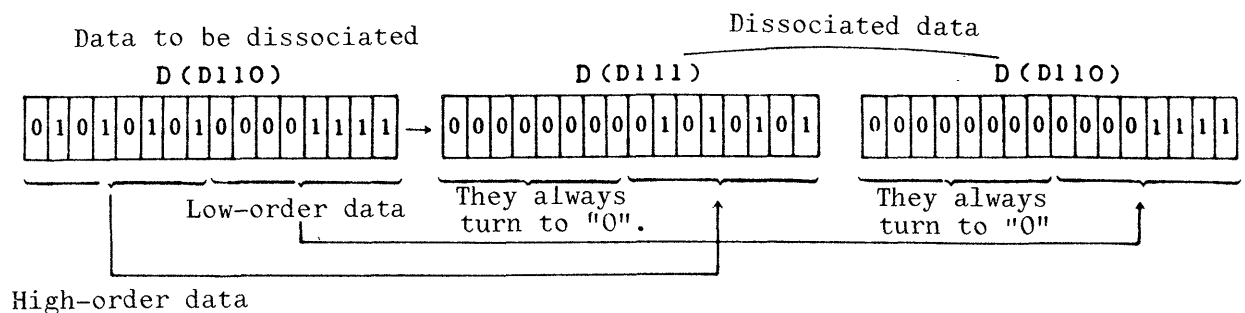


Fig. 4.2 Data dissociation

(1) When D No. storing 16-bit data to be dissociated is entered in D110 and D No. in which upper 8-bit data is placed after the dissociation is entered in D111, and F111 is executed, the dissociated two data are placed in D(D111) and D(D110).

- (2) The data to be dissociated may be binary 16-bit data or BCD 4-digit data.

4.2.2 Circuit applications

Ex.: Contents (BCD 4-digit data) of D5 are divided into two-digit BCD data, and placed in D5 and D20 separately.

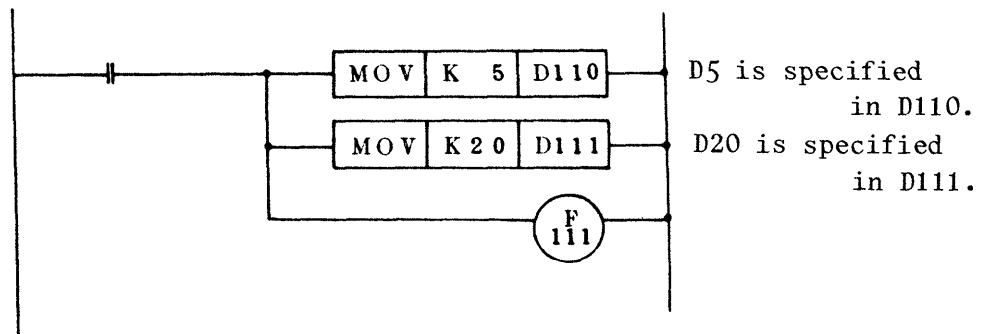


Fig. 4.3 Data dissociation

4.3 16-bit data AND operation

Each bit-to-bit AND operation is performed between two data registers.

4.3.1 Functions

Function No.: F112

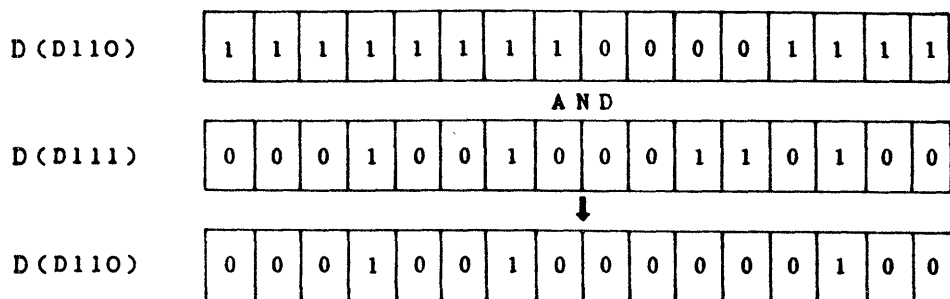


Fig. 4.4 16-bit data AND operation

4.3.2 Circuit applications

Ex.: The third digit of BCD 4-digit data of D10 should be masked with "0".

When D10 is "1 2 3 4", for example,

D10 = 1 2 3 4 → 1 2 0 4

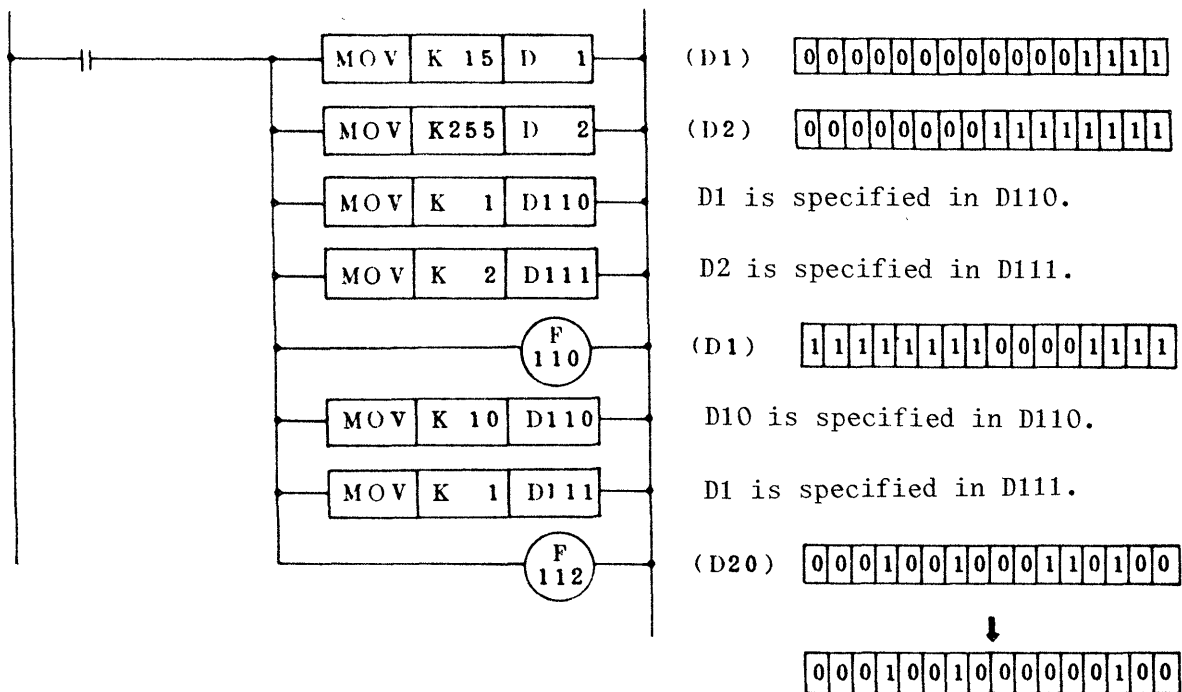


Fig. 4.5 AND operation circuit composition

4.4 16-bit data OR operation

Each bit-to-bit OR operation is performed between two data registers as follows:

4.4.1 Functions

Function No.: F113

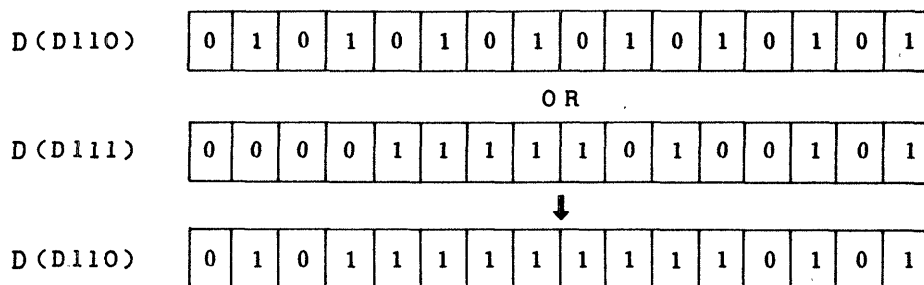


Fig. 4.6 16-bit data OR operation

4.4.2 Circuit applications

Ex.: OR operation is performed between D10 and D20 and the result is placed in D10.

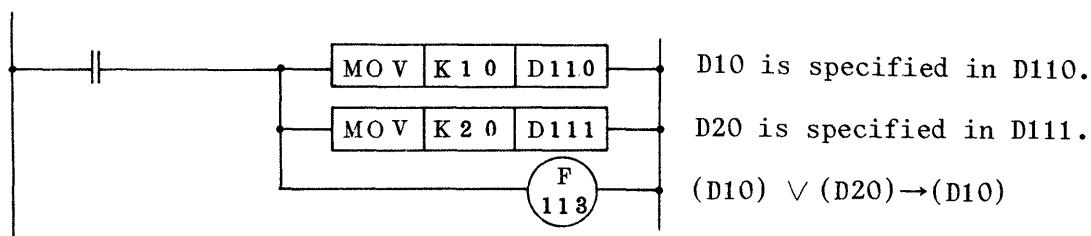


Fig. 4.7 16-bit data OR operation circuit

4.5 Batch shift of temporary memory M

Temporary memory M shift (SFT) instruction available in the standard CPU is of one bit shift instruction, and several steps are required for shift instruction with plural bits.

When the function F114 is used, contents of desired number of bits may be batch shifted leftward or rightward from the specified head No. of M.

4.5.1 Functions

Function No.: F114

Head No. of M:	D110	<table border="1"><tr><td>0</td><td>0</td><td>6</td><td>0</td></tr></table>	0	0	6	0	Binary numerals
0	0	6	0				
Number of bits to be shifted:	D111	<table border="1"><tr><td>0</td><td>0</td><td>3</td><td>0</td></tr></table>	0	0	3	0	Binary numerals
0	0	3	0				

NOTE

M may be specified between the range from M0 to M249.

If number of bits specified in D111 exceeds this range, shift is not executed.

Shift direction	<table border="1"><tr><td>D112</td></tr></table>	D112	<table border="1"><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	<table border="1"><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>	0	0	0	1
D112												
0	0	0	0									
0	0	0	1									
(leftward/rightward)		Leftward shift	Rightward shift									

- (1) The head No. of shift register to be formed is placed in D110. The No. should be junior one no matter whether shift is leftward or rightward, and written with binary numerals* in D110.
- (2) The length of shift register, that is, number of bits to be shifted, is written in D111 with binary numerals*.

*Writing with binary numerals

When decimal numeral "n" is written on PU or GPP in the form of MOV Kn D110 D111, it is automatically converted into binary numerals. However, binary numeral should be used as converting from decimal numeral when head No. and number of bits are specified in BCD code.

(3) Direction of shift should be specified in D112. When the contents in D112 are "0", the shift is leftward from junior No. to senior No. The shift, however, is rightward from senior No. to junior No. when the contents are "1".

(4) Circuit composition

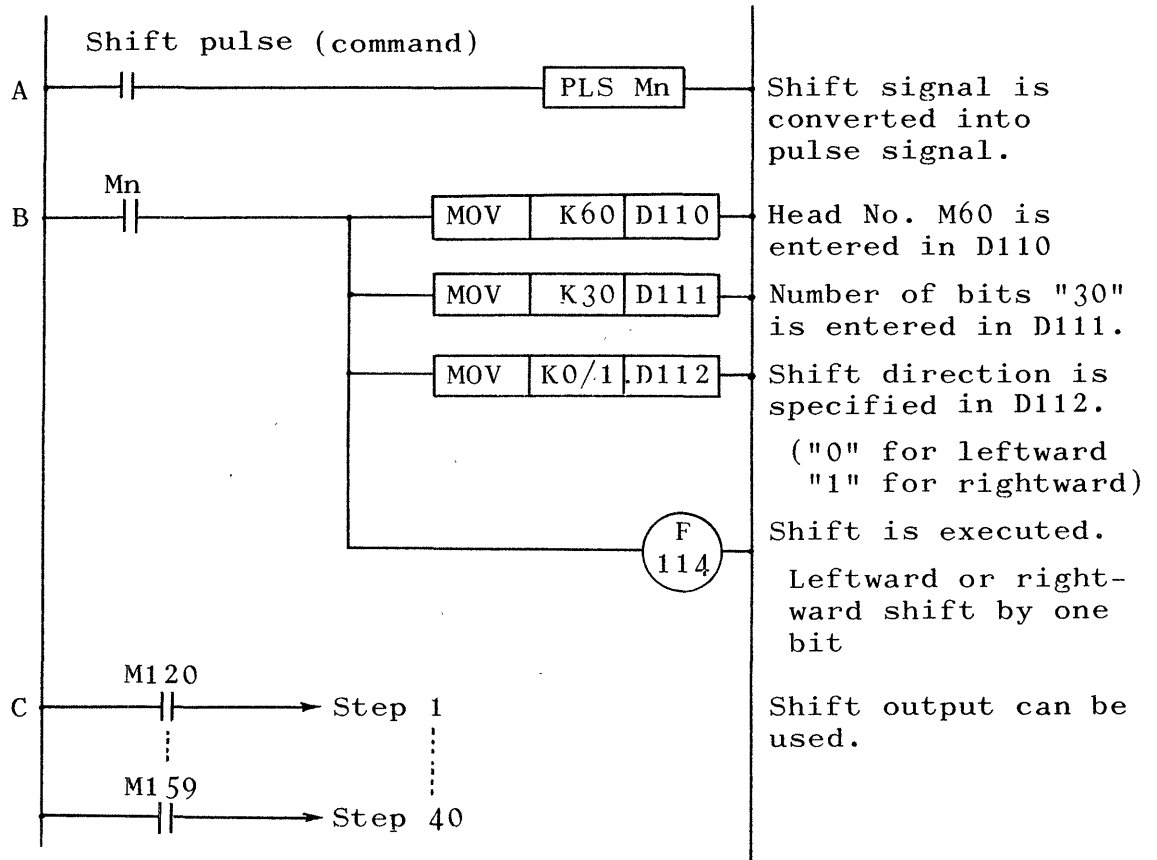


Fig. 4.8 Shift register circuit composition

- ① Shift command should be converted into pulses, otherwise "racing" (shift goes on without interruption although only one pulse is given) might occur.
- ② When only one shift register is available in a program, it is recommended to enter the data (head No., number of bits, etc.) at start of execution, thereby "B" block may be simplified with only Mn and F114.
- ③ Actual shift register requires, in addition to the cir-

cuit shown in Fig. 4.8, reset circuit and data set circuit. For details, refer to para. 4.5.2.

- ④ One-bit shift occurs each time when Mn turns on.
- (5) The status of the shift register exemplified in Fig. 4.8 is as follows:

① Leftward shift

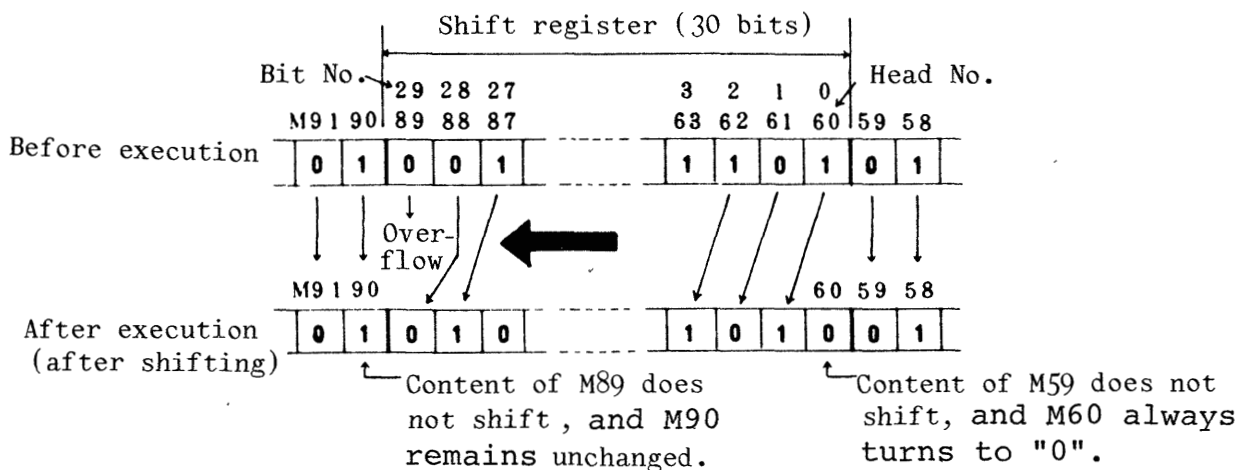


Fig. 4.9 Leftward shift

② Rightward shift

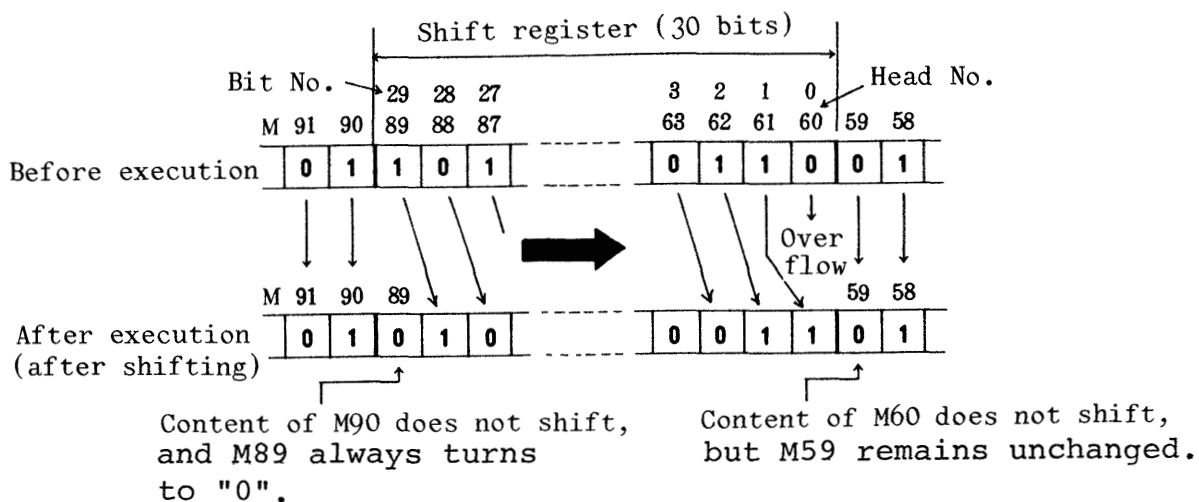


Fig. 4.10 Rightward shift

(a) In leftward as well as rightward shift, the first

bit (M60 for leftward shift, and M89 for rightward shift) is not affected by shifting, and replaced with "0" at all times. If any shift data must be entered, it should be placed after the shifting.

(b) The contents at the final bit of the shift register (M89 for leftward shift, and M60 for rightward shift) is erased due to overflow.

(6) Any bit "Mn" in the shift register may be set or reset with set (SET) instruction or reset (RST) instruction.

4.5.2 Circuit applications

(1) Leftward shift register

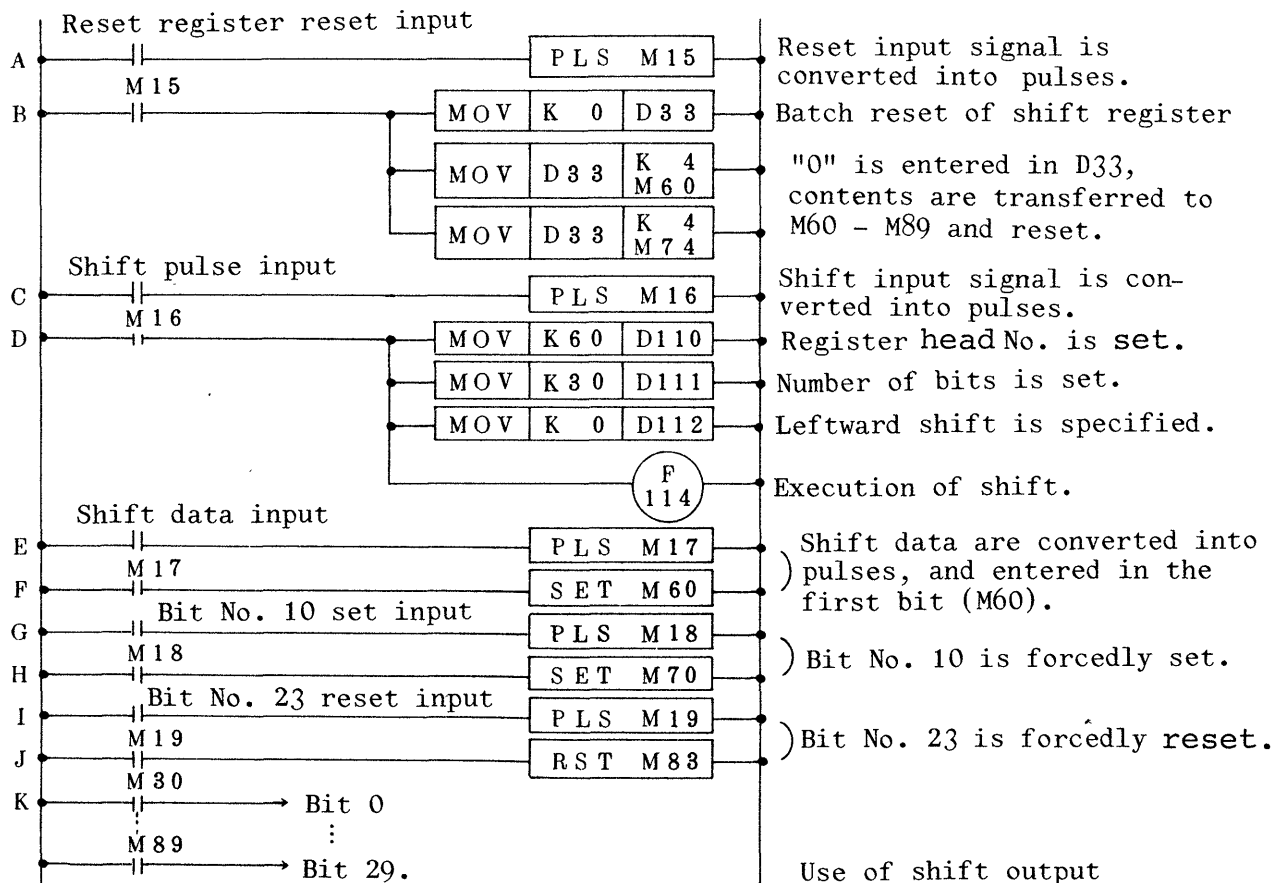


Fig. 4.11 Leftward shift register circuit

- ① Contents shift from junior M No. to senior M No..
- ② "B" circuit block is for multi-bit batch reset that is accomplished as follows:

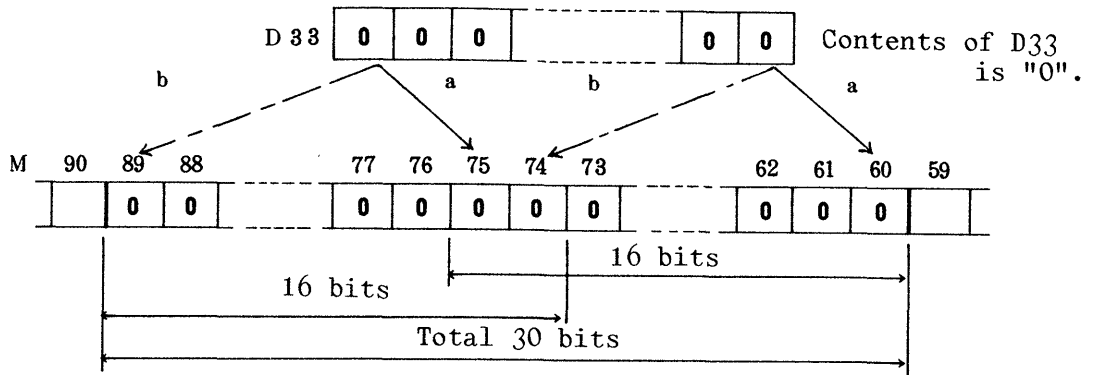


Fig. 4.12 Batch reset of M

In "a", "0" is transferred to M60 ~ M75 with `MOV D33 K4M60`; "0" is transferred to M74 - M89 with `MOV K 4 M74` in the case of "b".

Total 30 bits may be reset at the same time.

Although no function or instruction is available for batch reset, batch reset can be programmed as shown in Fig. 4.12.

- ③ The "E" and "F" circuit blocks are for setting of shift data to the head bit. The setting is accomplished without synchronization with shift pulses, and immediately realized when data are given.

If it needs synchronization with shift pulses, a circuit block "D" shown in Fig. 4.14 should be employed.

- ④ Any bit in the shift register may be forcibly set or reset with input signal, as the case may be with "G", "H"

"I" and "J" circuit blocks.

- ⑤ It is recommended for prevention of overlap of timing to convert each input signal into pulses, like in "A", "C", "E", "G" or "I" circuit block.
- ⑥ Timing chart of circuit operation.

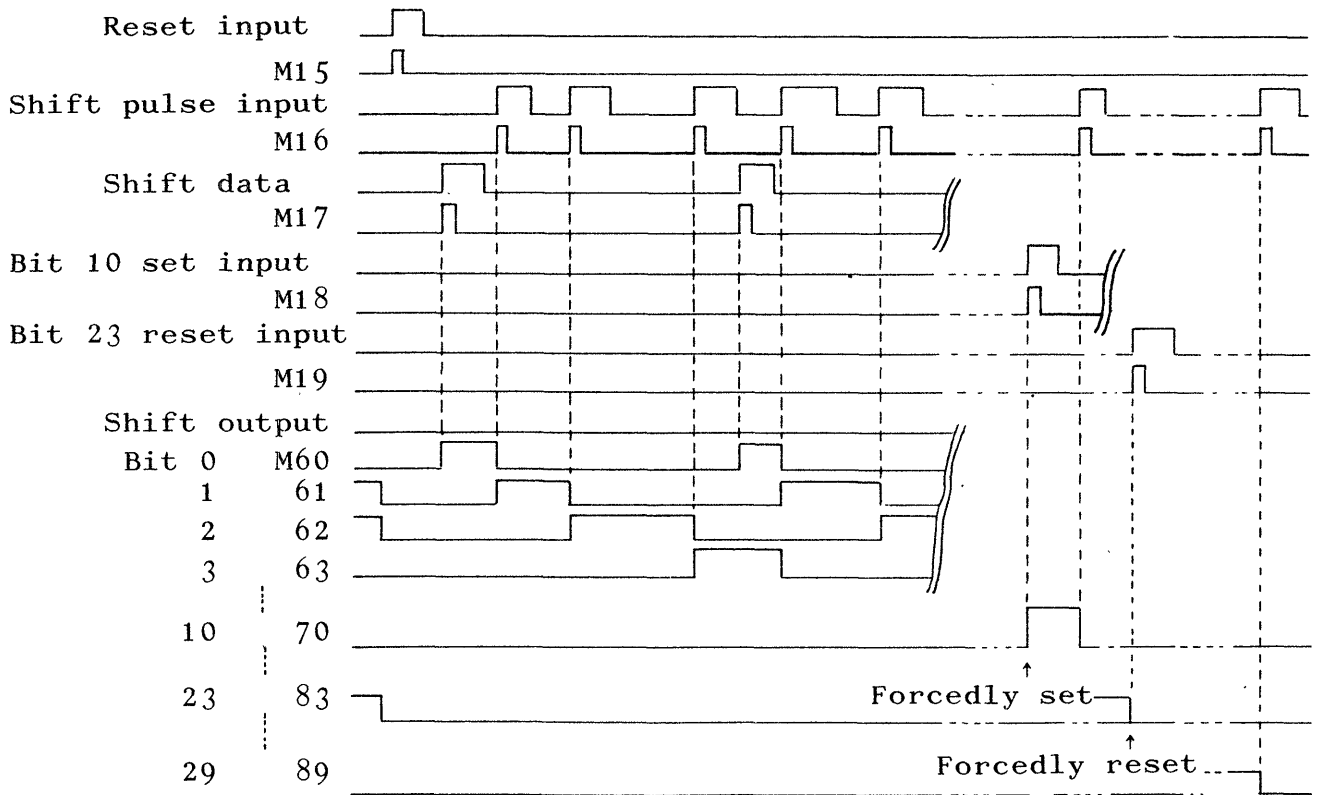


Fig. 4. Leftward shift timing chart

(2) Rightward shift register

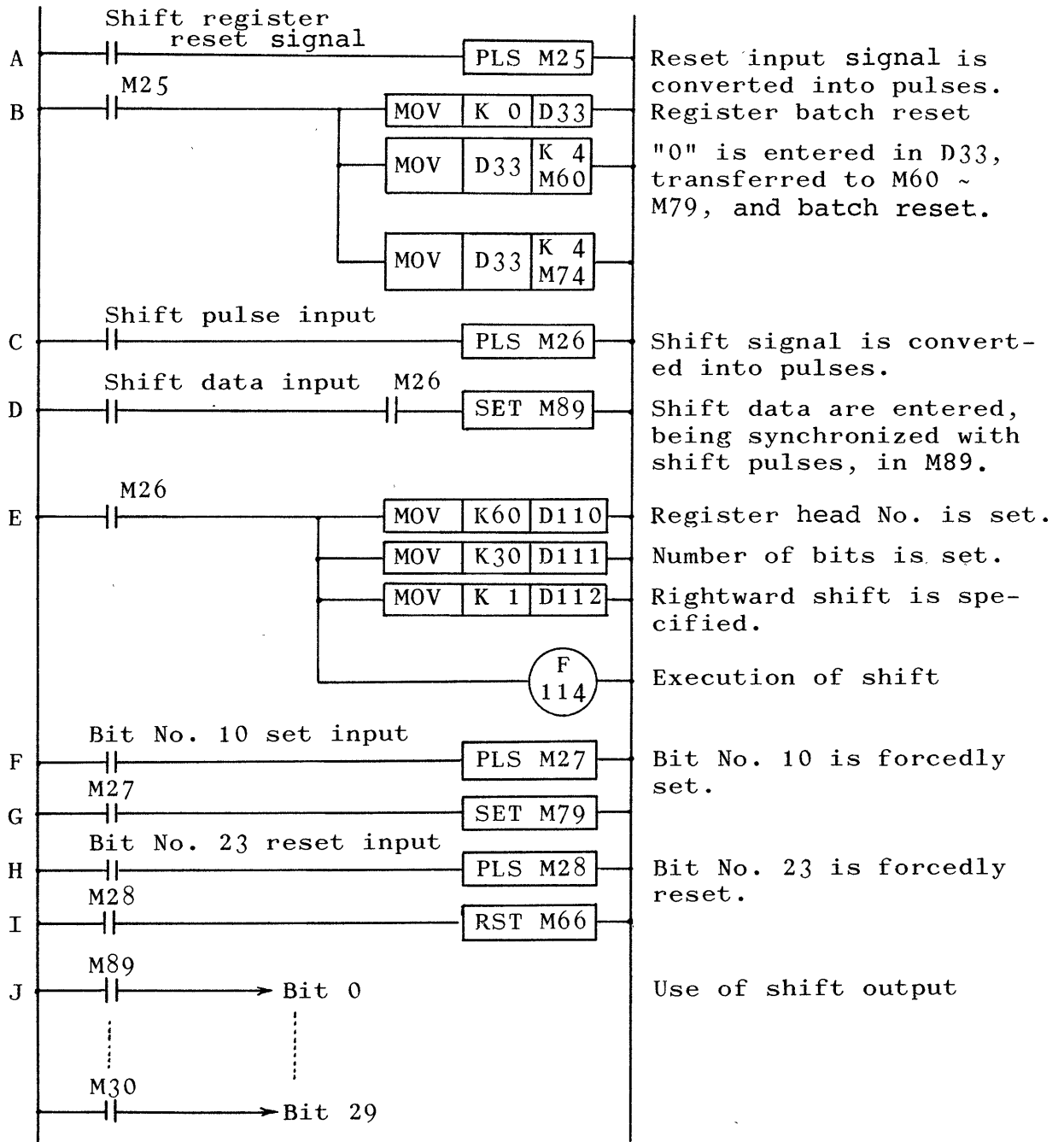


Fig. 4.14 Rightward shift register circuit

- ① "D" circuit block reads shift data, being synchronized with shift pulses, and sets the head No. of the rightward shift register "M89".
- ② Other functions and operations are same as those of the

leftward shift register.

③ Timing Chart of circuit operation

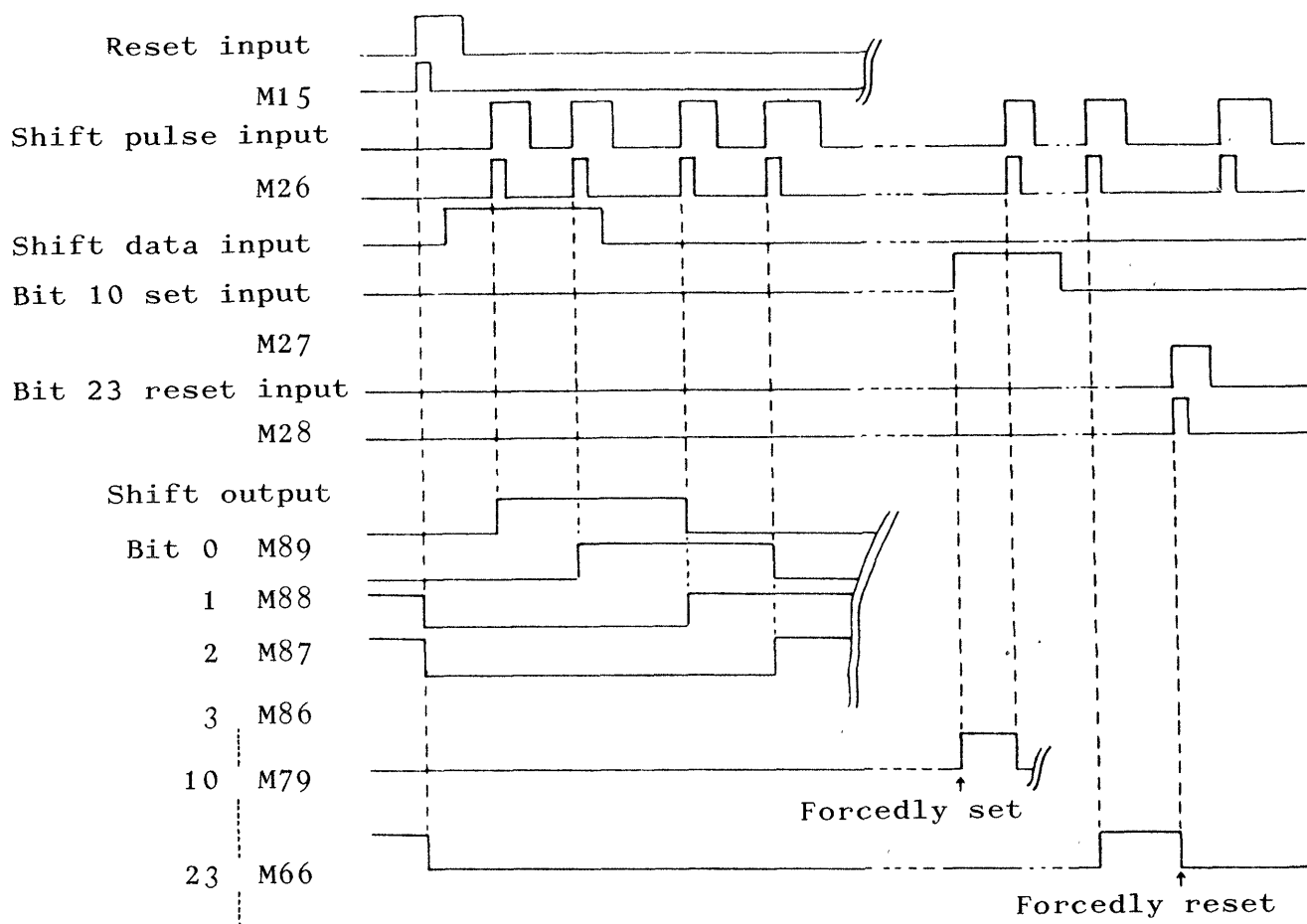


Fig. 4.15 Rightward shift register timing chart

4.6 Batch shift of data register D

In order to shift contents in data register of the standard CPU, MOV instruction must be repeated the same times as number of data to be shifted and therefore considerable number of steps must be programmed for large number of data.

When the additional function is used, the data shift may be accomplished only by specifying head No. of data register, length of data (number of bits), and direction of shifting,

thus facilitating programming with short number of steps.

4.6.1 Functions

Function No.: F115

Head No. of register D: D110 0 0 5 7 Binary numerals

Number of data registers to be shifted: D111 0 0 1 6 Binary numerals

Direction of shift: (leftward/rightward) D112 0 0 0 0 0 0 0 1
Leftward Rightward

NOTE

Applicable number of register range is from D0 to D95.

If number of registers specified by D111 exceeds D95, batch shift becomes impossible.

(1) Shift status

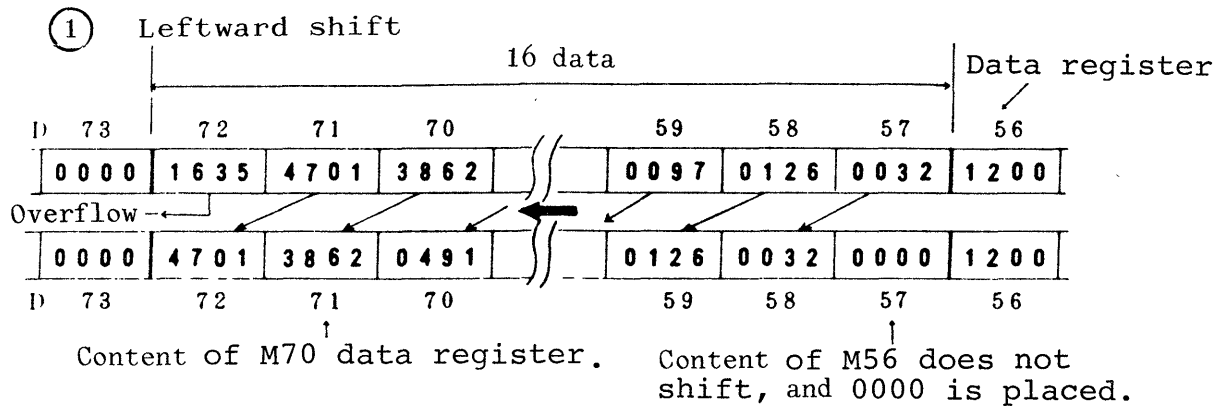


Fig. 4.16 Data shift (leftward shift)

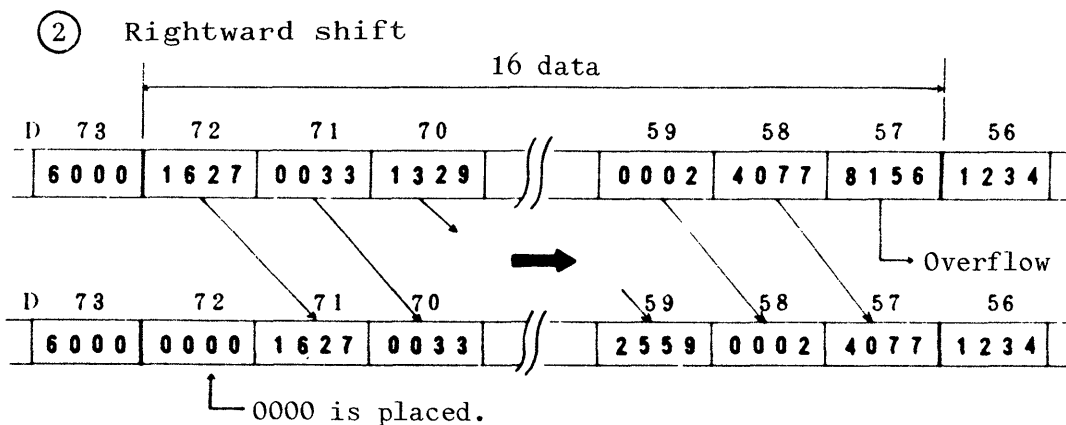


Fig. 4.17 Data shift (rightward shift)

- (2) The head No. of data register (junior No.) to be shifted should be specified in D110. The head No. should be one with junior No., no matter whether data shift is leftward or rightward, and be "D57" in Fig. 4.16 and Fig. 4.18.
- (3) Length of data (words) to be shifted should be specified in D111. When 16 data are desired to be shifted, for example, "16" is entered in D111. Care should be taken not to enter data over the range from D95 (leftward shift) to D0 (rightward shift).
- (4) Direction of shift is specified in D112.

Leftward	<table border="1"><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	Junior No. → Senior No.
0	0	0	0			
Rightward	<table border="1"><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	Senior No. → Junior No.
0	0	0	0			

- (5) With a shift signal, leftward or rightward batch shift (a group of data is shifted at the same time) occurs once.
- (6) Circuit composition

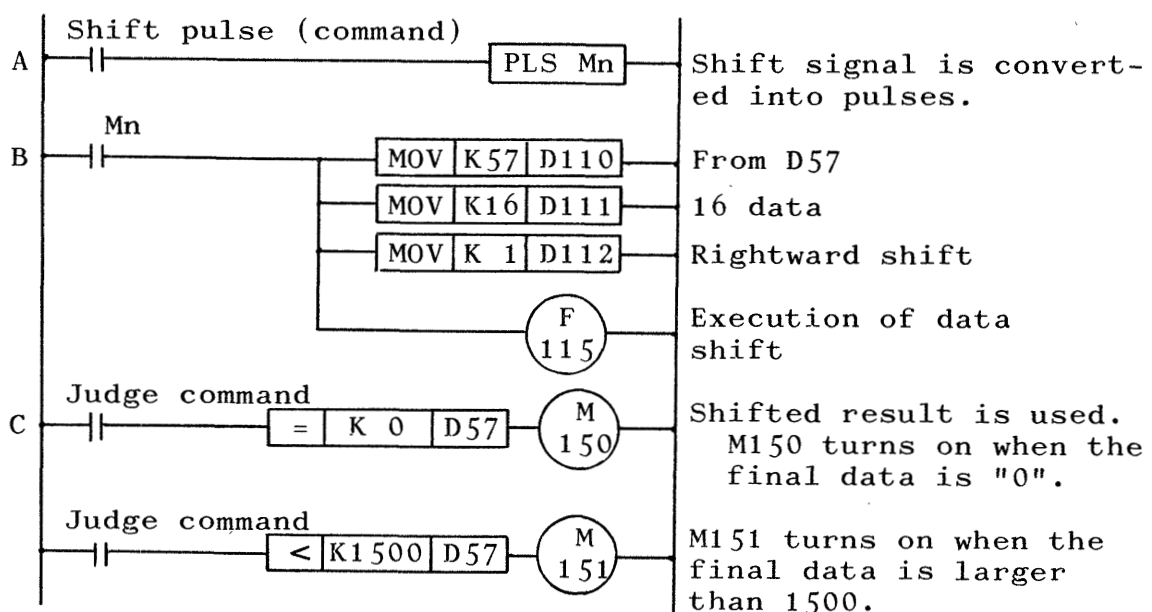


Fig. 4.18 Data shift circuit composition

- ① In order to prevent racing, be sure to convert shift signal into pulses.
 - ② "B" circuit block may be simplified to only Mn and F114 when one program has only one coil (one data shift) in F115 and three MOV instructions in that block are given at initial time.
 - ③ A separate circuit is required to enter data in the register with Shift top (D72 in this example).
 - ④ Each time shift signal Mn turns on, batch data shift occurs once.
 - ⑤ As the result of data shift, the contents of the final data register (D57 for rightward shift in Fig 4.17, and D72 for leftward shift in Fig. 4.16 are judged as in "C" circuit block and the judged result (M150, 151) may be used in a sequential control system.
- (7) The shift head register contents become "0", as shown in Fig. 4.16 and 4.17, when F115 is executed. Therefore, the data set mentioned in ③ should be performed at a step coming after the execution of F115.
- (8) The last register contents are erased after the execution of F115, due to overflow.
- (9) Contents in the data registers out of shifting range are not affected by data shifting.

4.6.2 Circuit applications.

The example is that data shift is performed for "tracking" (tracing of data) in the working line shown below.

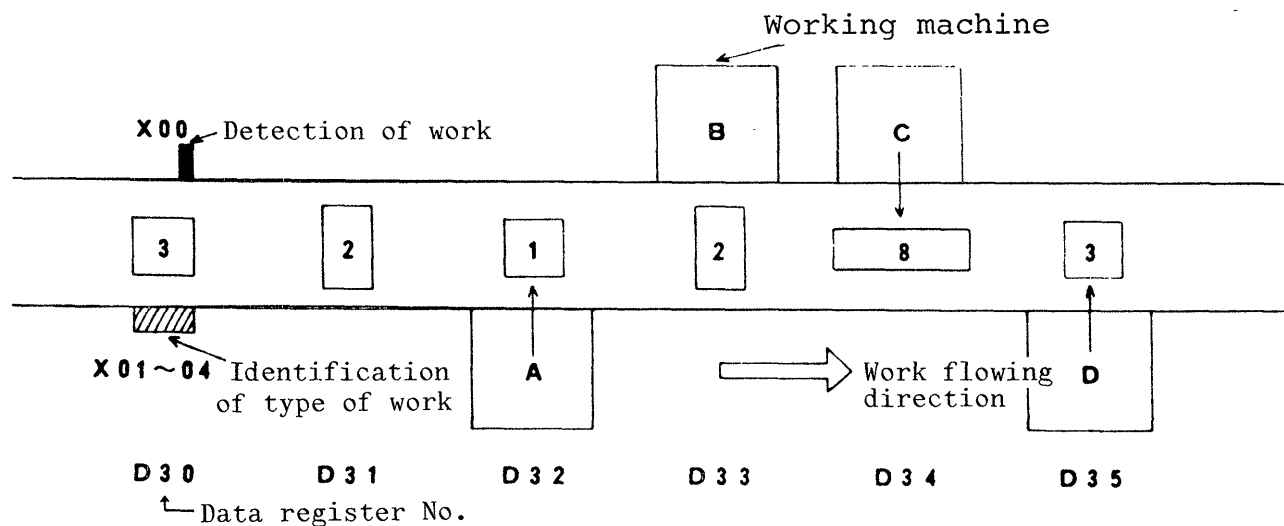


Fig. 4.19 Data shift model

The control sequence in the example shown in Fig. 4.19 is as follows:

Data registers D30 ~ D35 are assigned for each stop station (6 stations) on the working line.

At the entrance of the line, whether work exists or not is checked. When work exists ($X00 \rightarrow ON$), type of the work is identified (identification No. is used to prepare work data). Signal "X00" is used as shift pulse (command) signal.

The working machines are arranged so that the working operation well corresponds to the work data, as shown in Fig. 4.20 (Circuit block D - G).

When working has been completed at a machine, the work is fed to the next station, and stops with "X00" turned on.

When "X00" turns on, the work data are shifted and the work

No. is read. The read work No. is entered in the head data register D30.

After the shifting, work No. is discriminated at each machine and the working starts again when the conditions are satisfied.

The circuit exemplified here shows only the basic operation. In actual control system, various interlocks, conditions and peripheral circuits must be incorporated.

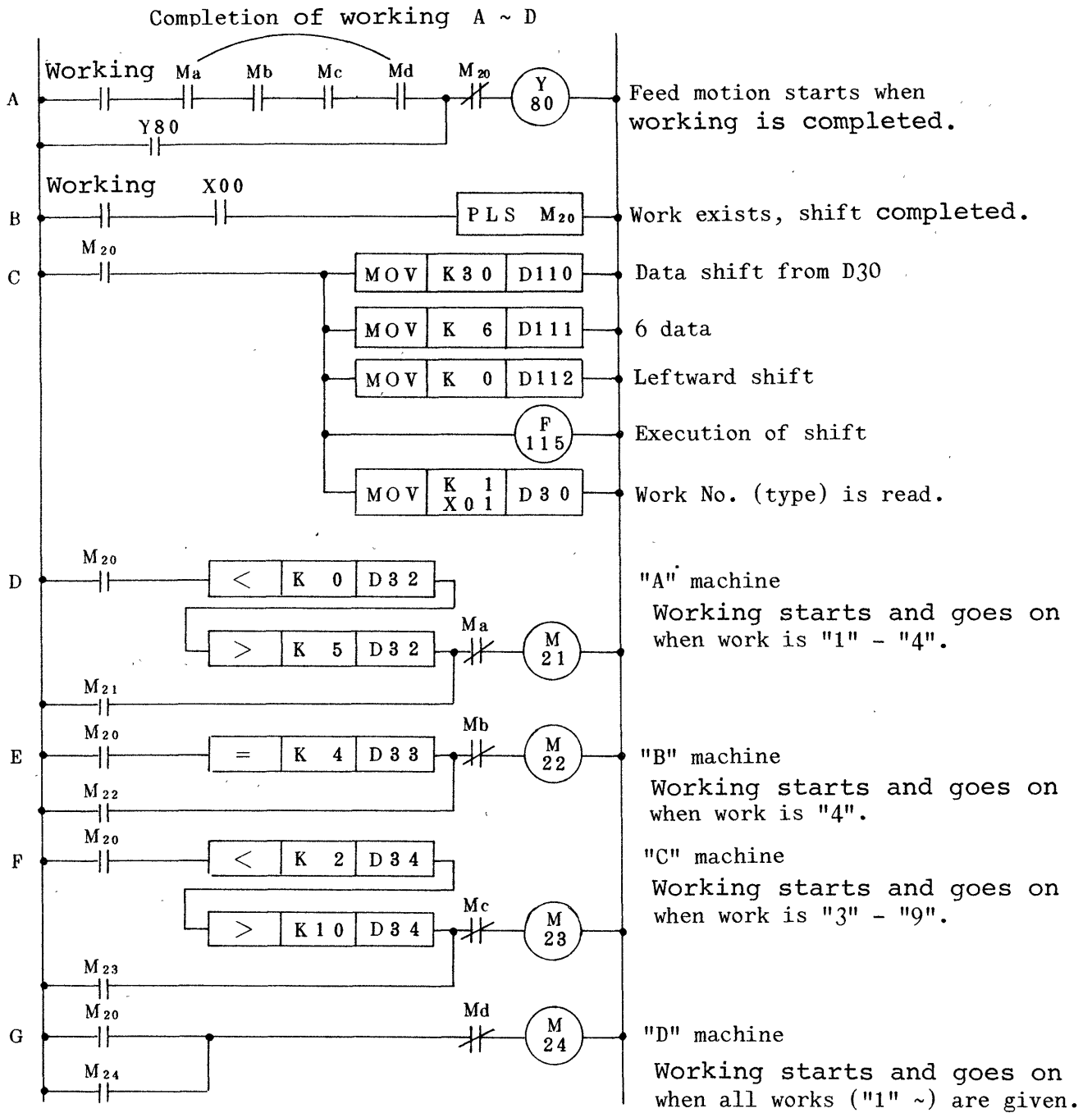


Fig. 4.20 Data shift and its application circuit

4.7 Batch reset of data register D

All consecutive data register contents are cleared to "0".

4.7.1 Functions

Function No.: F116

Head No. of register D: D110 0 0 4 5 Binary numerals

Number of data registers to be reset: D111 0 0 2 0 Binary numerals

NOTE

Number of data registers ranges from D0 to D95. Batch reset becomes impossible when the number of data registers is out of this range.

- (1) The head No. of data registers to be reset is specified in D110.
- (2) Number (length) of data to be reset is specified in D111.
- (3) When F116 is executed under the above-mentioned conditions (1) and (2), the all contents in the data registers up to 20 wds. from D45 (D45 ~ D64) are cleared to "0".

4.7.2 Circuit applications.

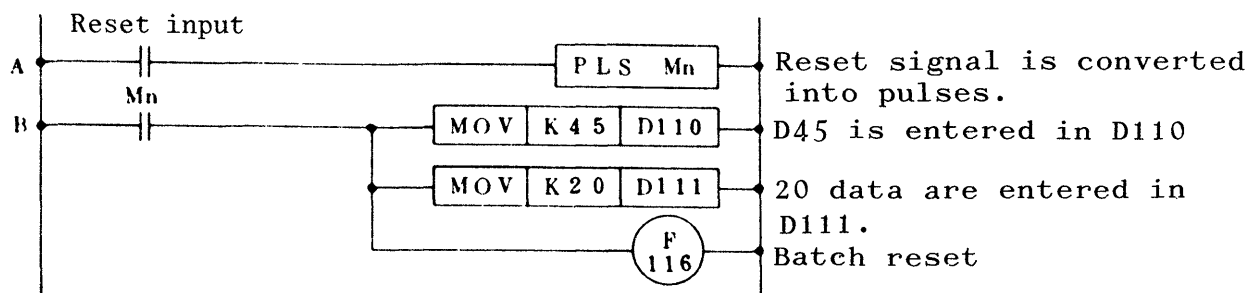


Fig. 4.21 Data register batch reset circuit

- ① Reset input signal should be converted into pulse signal at all times.
- ② The register D not backed up (latched) for power failure is automatically reset when the programmable controller is turned on or CPU "RESET" switch is operated.
- ③ The functions may be used to reset the register D backed up for power failure.

4.8 Indirect reading of T,C,D (Timer, Counter, Register D)

When current values of timer (T) or counter (C), or contents of register (D) are read in the standard CPU, each one circuit must be programmed for each No. to be read.

On the contrary, desired current values or contents can be readily read with this additional function, only by specifying No. of T, C or D and executing this function(F117).

Therefore, the use of F117 is very advantageous when externally display or determination is desired for reading of current values or contents of T, C or D because the circuit may be simplified.

4.8.1 Functions

Function No.: F117

No. of T, C or D: D110

0/1	0	6	4
-----	---	---	---

 Binary numerals

↑
T,C,D No.
— Specifies T, C or D

0: T, C
1: D

Contents read: D111

9	8	7	6
---	---	---	---

NOTE

T or C No. should be within a range from 0 to 127, and D within a range from 0 to 95. F117 may not be realized when No. out of this range is specified.

- (1) Desired No. of T, C or D is binary specified in D110.

When timer or counter current value is read, the 4th digit is filled with "0" (decimal numeral).

The 4th digit is filled with "1" when register content is read.

Ex.: 0 0 6 4 T64 or C64
 1 0 6 4 D64

- (2) When F117 is executed, the current value or content of specified NO. is read out in D111.

It should be noted that timer/counter current value is in binary notation, while register content is of binary, or BCD, or bit pattern.

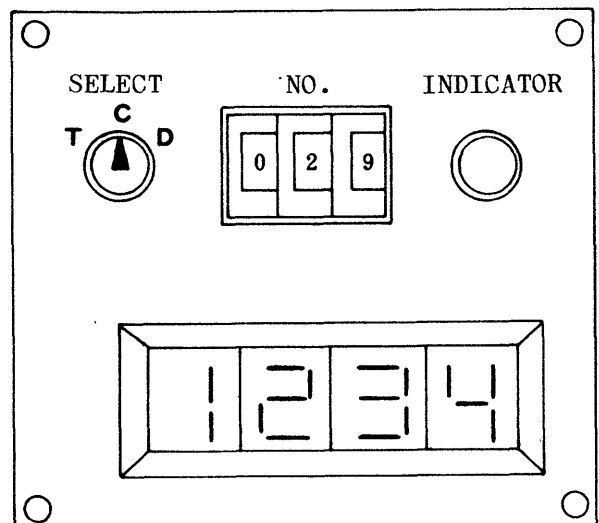
4.8.2 Circuit applications

Fig. 4.22 shows an example of display panel capable of indicating timer/counter current value or register content (binary).

Name of each part of the panel and the connection with programmable controller are shown in Fig. 4.22.

T: X0 No.: X4 ~ F
C: X1 Indicator: X3
D: X2 Readout:
 Y10 ~ 1F

Fig. 4.22 T,C,D Display panel



In the circuit shown in Fig. 4.23, when X2 turns on (register content is read), register No. is set by adding "1000" to read No. As for timer/counter current value, no additional operation is required.

Since data is read in binary notation after the execution of F117, it should be output to the readout through BCD conversion.

In practical use, a provision will be necessary to check that setting is not out of the specified range (T128, for example).

The circuit blocks "B" and "C" may be modified as shown in Fig. 4.25.

In this circuit, the above-mentioned checking circuit is also incorporated.

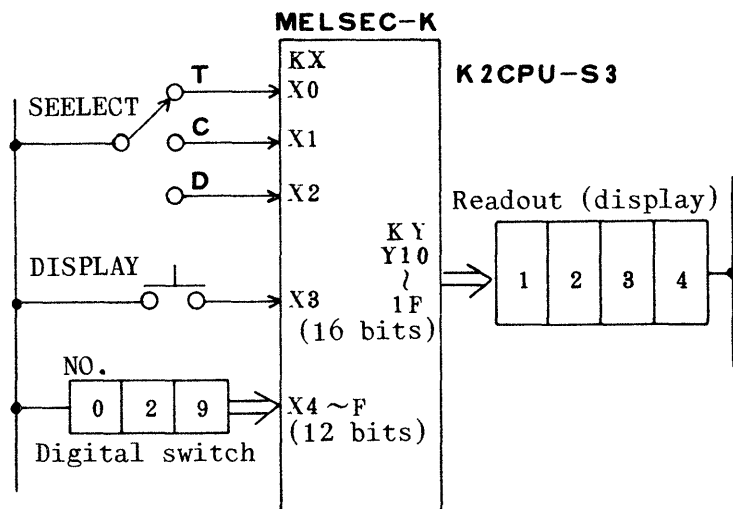


Fig. 4.23 External connection

4.9 Indirect writing of T,C,D

With this function, current values of timer (T), counter (C) or register (D) contents are written with the No. of timer or register specified.

The indirect writing is mainly used to change contents of register D, and utilizing this, it may be usable to change set values of T, C.

Although current values of timer (T) or counter (C) may be changed by using this function, it is recommended to change the current values by usual program edition.

It should be noted that T, C set value with Kn has been written at programming steps and therefore may not be changed with this function.

To change T, C set value, Dn should be specified for set value and the content of Dn should be changed with this function.

4.9.1 Functions

Function No.: F118

T, C, D No.: D110

0/1	1	0	3
-----	---	---	---

Binary numerals

T,C,D No.

Specifies T, C or D.

0: T or C

1: D

Data to be written: D111

5	4	3	2
---	---	---	---

Binary numerals

NOTE

T, C No. should be specified within a range from 0 to 127 and D No. should be specified within a range from 0 to 95.

If specified No. is out of the range, F118 cannot be executed.

- (1) T, C or D No. is entered in D110 with binary numerals.
Numerals ranging from 0 to 127 may be used for T, C No. and numerals added with "1000", ranging from 1000 to 1095, may be used for D No.
- (2) Data is written in D111 with binary numerals.
T, C or D content is reset (cleared) when data is "0".
- (3) The data entered in D111 is written to the T, C or D No. specified in D110 when F118 is executed.

4.9.2 Circuit applications

An example of application of indirect writing, where T or C current values, or D contents are totally changed, is described here.

Fig. 4.26 is an example of practical application to "writer panel" and Fig. 4.27 shows the external connection diagram.

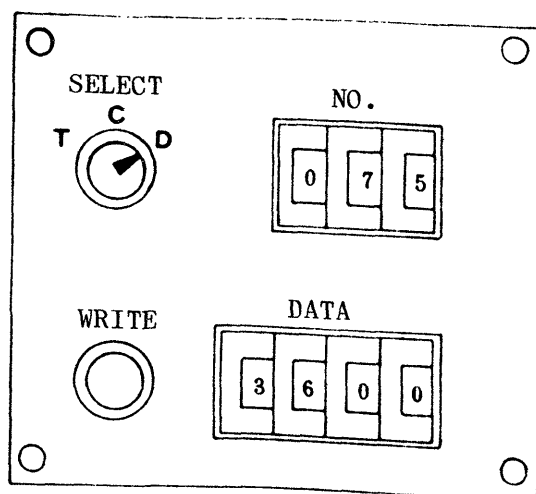
Actual circuit becomes as shown in figure 4.28.

The circuit block composition from "A" to "E" is same as that

shown in Fig. 4.25, where No. is entered and checked.

In the circuit block "F", data is entered, and written to the No. specified in D110.

If No. setting is not acceptable, data reading and writing are impossible and T, C, D current value/content cannot be changed.



T: X0	Write	X3
C: X1	No.	X4 ~ F
D: X2	Data	X10 ~ 1F

Fig. 4.26 T,C,D writer panel

When setting error occurs, M21 turns off. Alarm lamp or buzzer may be used when circuit block "G", for example, is added.

When the panel combination consisting of those shown in Fig. 4.22 and Fig. 4.26 is built up and the exemplified programs are prepared, they may be used in debugging.

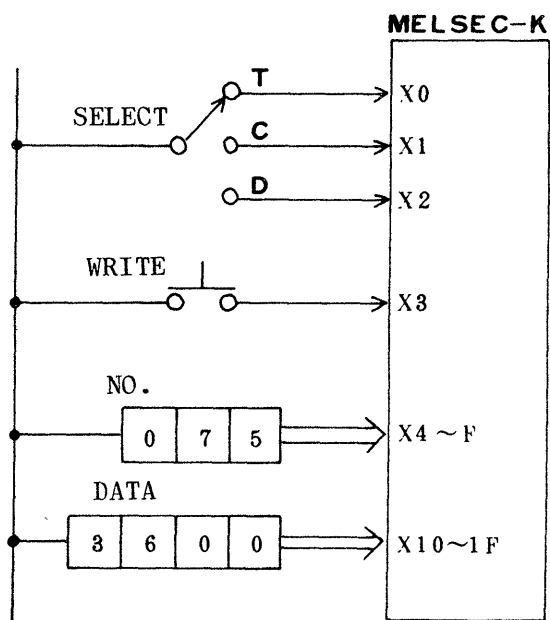


Fig. 4.27 External connection

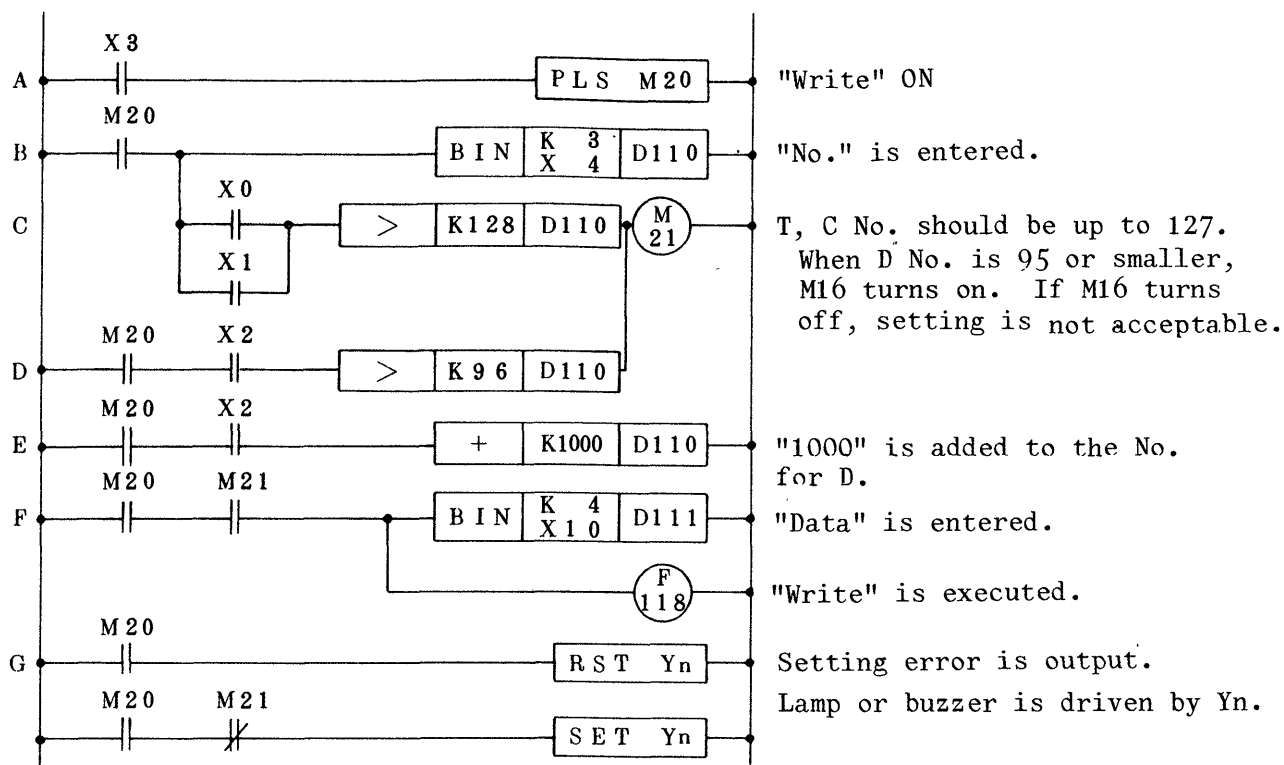


Fig. 4.28 T, C, D indirect writing circuit composition

4.10 Data transference from Y to D

Although the standard CPU permits data transference from D to Y, reverse data transference from Y to D is impossible. 16-bit ON-OFF (L-H) status of Y, grouped for each four bits, can be transferred to the specified D when this function is executed.

The function may be used to check bit pattern (ON-OFF status) output from Y against the standard bit pattern, or to latch bit pattern in D in case of power failure.

4.10.1 Functions

Function No.: F119

Y No. and digits: D110

n	0	Y.No
---	---	------

 Binary numerals

↑ Head No. of Y: 00 ~ 31
 ↑ To be "0" at all times
 ↑ Number of digits: 1-4

D No. to be transferred:

D111

1	D.No
---	------

 Binary numerals

↑ D No.: 000 ~ 095
 ↑ D is specified here.
 To be "1" at all times.

(1) D110 can be set up as follows:

① n: Number of 4-bit groups 1-4

Number of bit groups	Total number of bits
1	4
2	8
3	12
4	16

② 0: To be "0" at all times.

③ Y. No.: Head No. of Y..... 00 ~ 31

NOTE:

D should be within a range from 0 to 95.

If D is out of this range, F119 cannot be executed.

(a) Y No. should be the head No. of 16-bit group, and expressed in decimal notation, as shown in Table 4.1.

Table 4.1 Y No. and applicable code

Y No.	Code
000~00F	00
010~01F	01
020~02F	02
030~03F	03
040~04F	04
050~05F	05
060~06F	06
070~07F	07
080~08F	08
090~09F	09
0A0~0AF	10
0B0~0BF	11
0C0~0CF	12
0D0~0DF	13
0E0~0EF	14
0F0~0FF	15

Y No.	Code
100~10F	16
110~11F	17
120~12F	18
130~13F	19
140~14F	20
150~15F	21
160~16F	22
170~17F	23
180~18F	24
190~19F	25
1A0~1AF	26
1B0~1BF	27
1C0~1CF	28
1D0~1DF	29
1E0~1EF	30
1F0~1FF	31

(b) The head No. of Y is the head No. of 16-bit group. Therefore, transference of partial bits, such as 4 bits from "018", is impossible.

(c) Example of setting of D110

Contents of D110	Area of which data can be transferred	Bits
3000	Y000 ~ 00B	12
1011	Y0B0 ~ 0B3	4
2022	Y160 ~ 167	8
4030	Y1E0 ~ 1EF	16

(2) D111 is used to specify No. of D to which contents of Y (4 bits ~ 16 bits) are transferred, and filled with D No. plus "1000".

Ex.: D 0: 1000
D39: 1039
D74: 1074

- (3) Even when transferred data contents are of 4 - 12 bits, one register D should be used.

4.10.2 Circuit applications

As an example of the function F119, a circuit that permits checking of bit pattern with 12 bits starting from Y70 against the standard bit pattern is described here.

The description also includes checking of 8-bit pattern from YB0, and how to form the standard bit pattern.

Number of bits and destination of transference are also exemplified in this paragraph.

- ① When 12-bit contents starting from Y70 are transferred to D8

Y 7B	7A	79	78	77	76	75	74	73	72	71	70
0	1	0	1	1	0	0	0	1	0	1	1
2048	1024	512	256	128	64	32	16	8	4	2	1
	↓		↓	↓				↓		↓	↓
	1024	+	256	+ 128		+		8	+	2	+ 1 = 1419

- ② When 8-bit contents from YB0 are transferred to D24

Y B7	B6	B5	B4	B3	B2	B1	B0
1	0	1	1	1	0	1	0
128	64	32	16	8	4	2	1
↓		↓	↓	↓		↓	
128	+	32	+ 16	+ 8	+	2	= 186

When K1419 for Y70, or K186 for YB0, is entered to the specified D by using MOV instruction, the bit pattern in the D becomes as shown above and the standard bit pattern can be obtained.

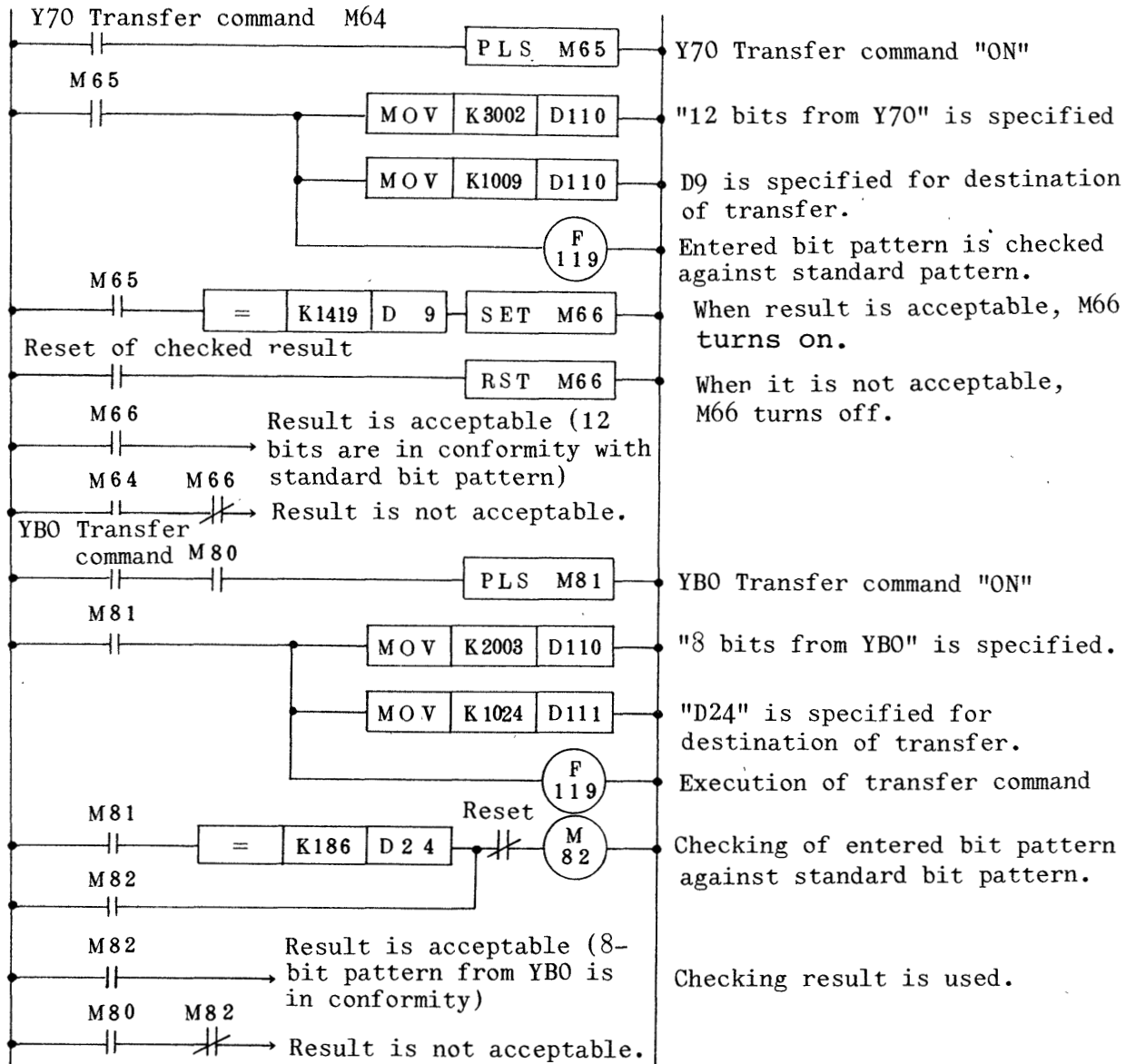


Fig. 4.29 Y→D Data transfer circuit composition

4.11 4↔16 Decode/encode

Decode means that 4-bit binary numerals are converted into 16-bit pattern. On the contrary, encode means that 16-bit pattern is converted into 4-bit binary numerals.

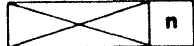
The decode function in the MELSEC-K0J includes transference of converted 16-bit pattern to M, thereby M is used as control means. With this function, 1-bit data shift register or counter may be readily formed.

In the encode function, one "1" bit in 16 bits is identified, and converted into binary numerals, thereby significant bit position is found.

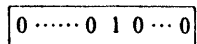
The encode function may be used to identify step No. of counter, or shift position in 1-bit shift register.

4.11.1 Functions

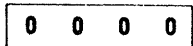
Function No.: F108

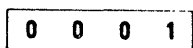
Data to be decoded or encoded: D110  Decode

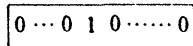
Lower 4 bits are effective and upper 12 bits are ignored in decoding.

 Encode

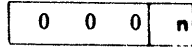
Only one "1" (H) bit exists in 16 bits in the case of encoding.

Decode/encode designation: D111  Decode

 Encode

Decode/encode result: D112  Decode

Only one "1" bit exists in 16 bits in the case of decoding.

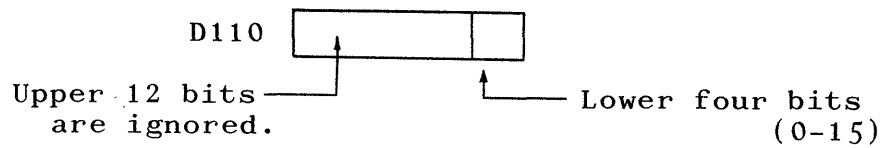


Encode

In encoding, bit position is specified in lower four bits. Upper digits are filled all with "0".

(1) The data to be decoded or encoded is stored in D110.

(a) In decoding, only lower four bits (0 ~ 15 in decimal notation) are effective, and converted into bit pattern to be stored in D112.



(b) In encoding, data having only one "1" (H) bit in 16 bits is handled. This single "1" bit is converted into 4-bit binary numerals, and stored in D112. The data handled in encoding should have only one "1" bit in 16 bits.

(2) Discrimination between decode and encode is specified in D111.

D111

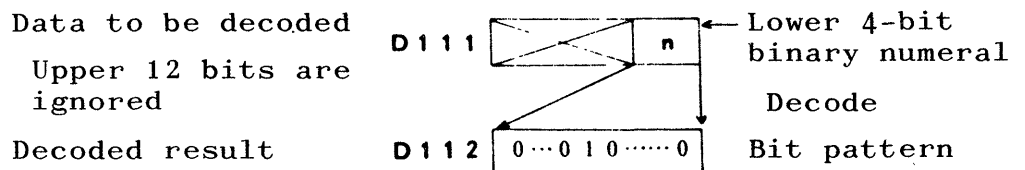
(a) Decode 0 0 0 0

(b) Encode 0 0 0 1

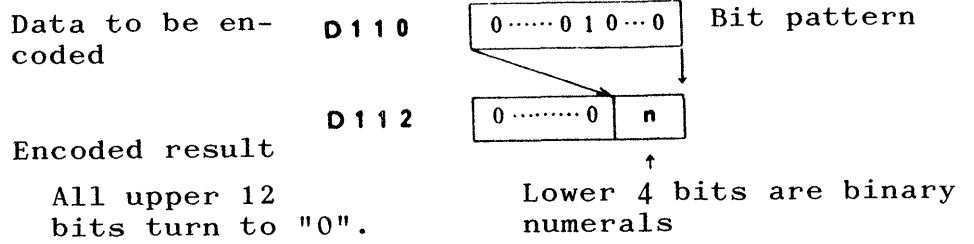
(3) Decoded or encoded result is stored in D112.

(Refer to step (4).)

(a) Decode



(b) Encode



(4) Decoded/encoded result

Table 4.2 Decoded/encoded result

ENCODE	D112					D110																
DECODE	D110															D112						
	15~4	3	2	1	bit 0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	bit 0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
2	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
3	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
5	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
6	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
7	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
9	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
10	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
12	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(a) In decoding, upper 12 bits of the data (D110) to be decoded are ignored.

(b) In encoding, all upper 12 bits of the encoded result (D112) turn to "0".

(5) Circuit composition

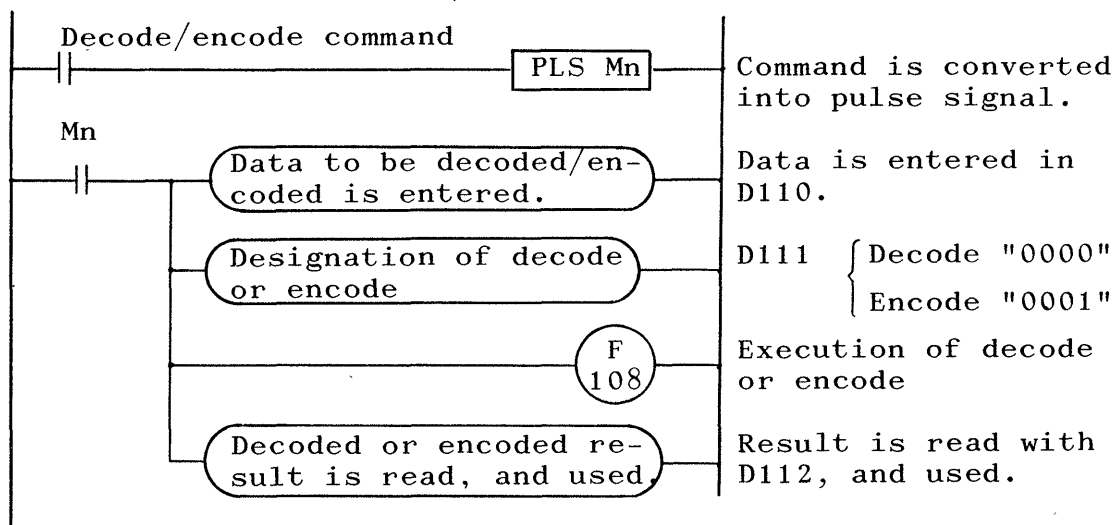


Fig. 4.30 Decode/encode circuit composition

4.11.2 Circuit applications

(1) Decode circuit

In this example, numerical data are taken from computer or N/C unit and converted into bit pattern for sequential control.

Input data and timing are assumed as follows:

Data: X0 ~ 3 4 bits, binary numerals

Data transmission: X4

Input timing: X0 ~ 3

X4



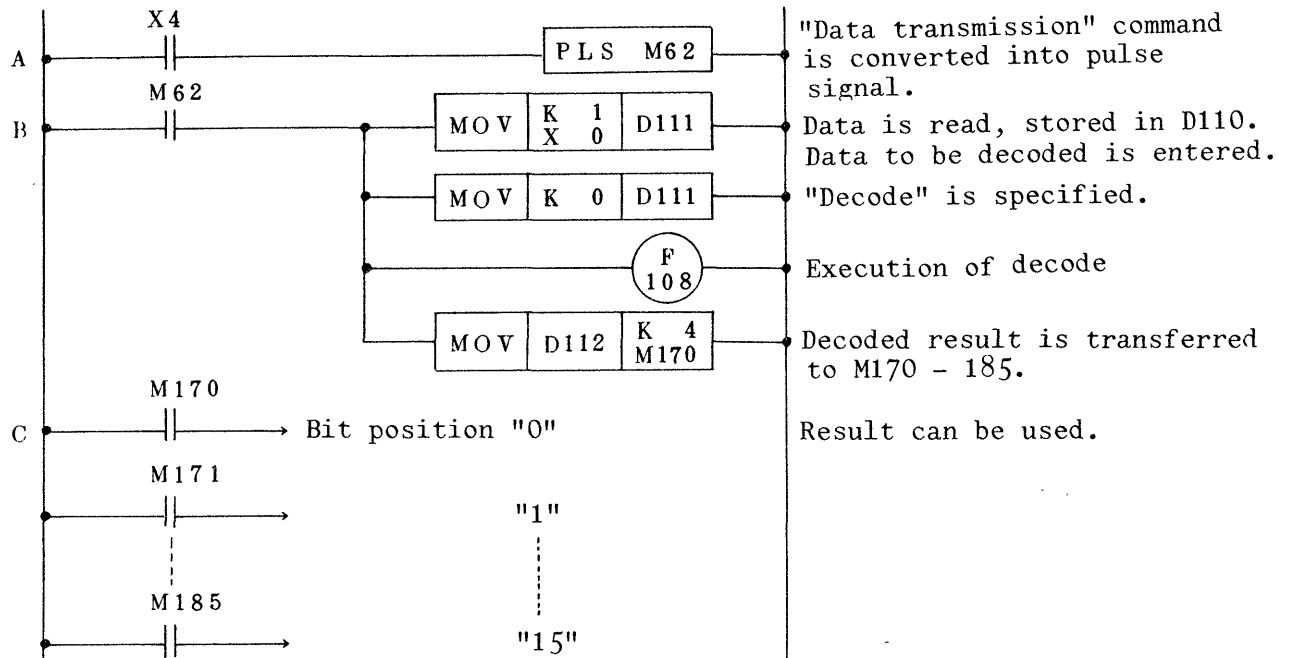


Fig. 4.31 Decode circuit composition

(2) Encode circuit

The example is that control command is changed by significant bit position where counter (1-bit data shift register) is ON.

Counter: M130 ~ 145 16 bits

Bit position: 3 ~ 6 "A" control
 7 ~ 11 "B" control
 12 ~ 15 "C" control

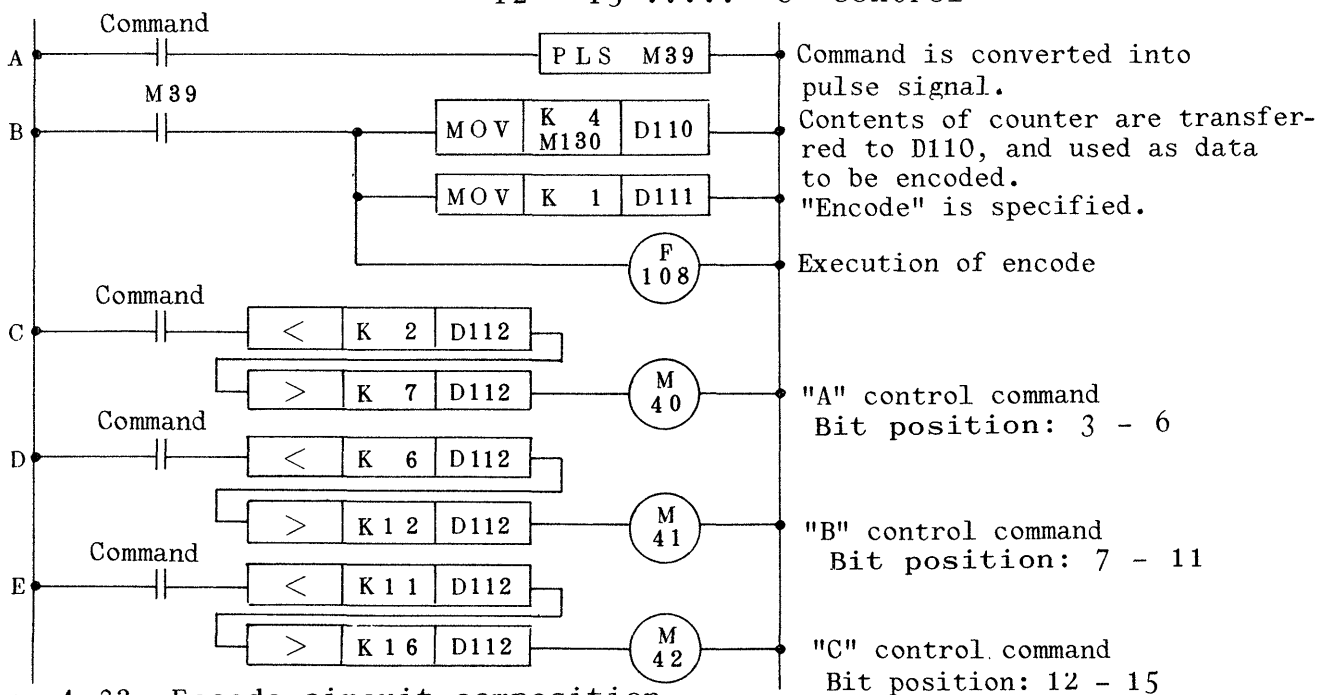


Fig. 4.32 Encode circuit composition

In this circuit, D112 is directly used for comparison. When D112 is used in any other circuit, M40 - M42 should be self-held, or D112 contents should be transferred to Dm for comparison.

4.12 16-bit check

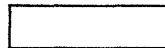
This function is used to check and examine how many "1" (ON) bits exist in 16 bits of one register.

Practical application example of the function is that number of stored goods can be identified when flow of goods is traced in conveyor or any transfer, or material handling control system by making use of shift register functions, or the case where how many outputs are in ON within a certain range of output Y is examined.

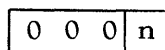
4.12.1 Functions

Function No.: F109

Check data: D110



Cumulative number of bits: D111



↑
Cumulative number of bits
... Binary numeral
(0 - 16)

- (1) Check data (how many bits of which are in ON is to be identified) are entered in D110.
- (2) When F109 is executed, total number of bits in ON is stored in the lower four bits of D111 in the form of binary code and all upper 12 bits turn to "0".

(3) Circuit composition

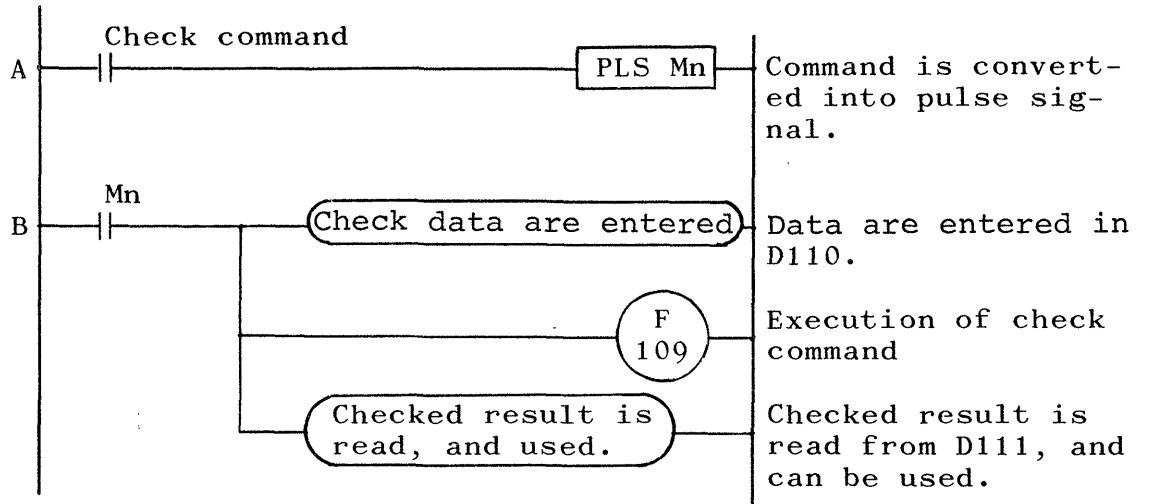


Fig. 4.33 16-bit check circuit composition

4.12.2 Circuit applications

(1) Checking of number of bits "1" (ON) in shift register

This application example is used when number of goods in a conveyor line must be numerically displayed, for example.

Ex.: Shift register M50 - M81 ----- 32 bits
 Display unit Y80 - Y87 ----- Decimal 2 digits

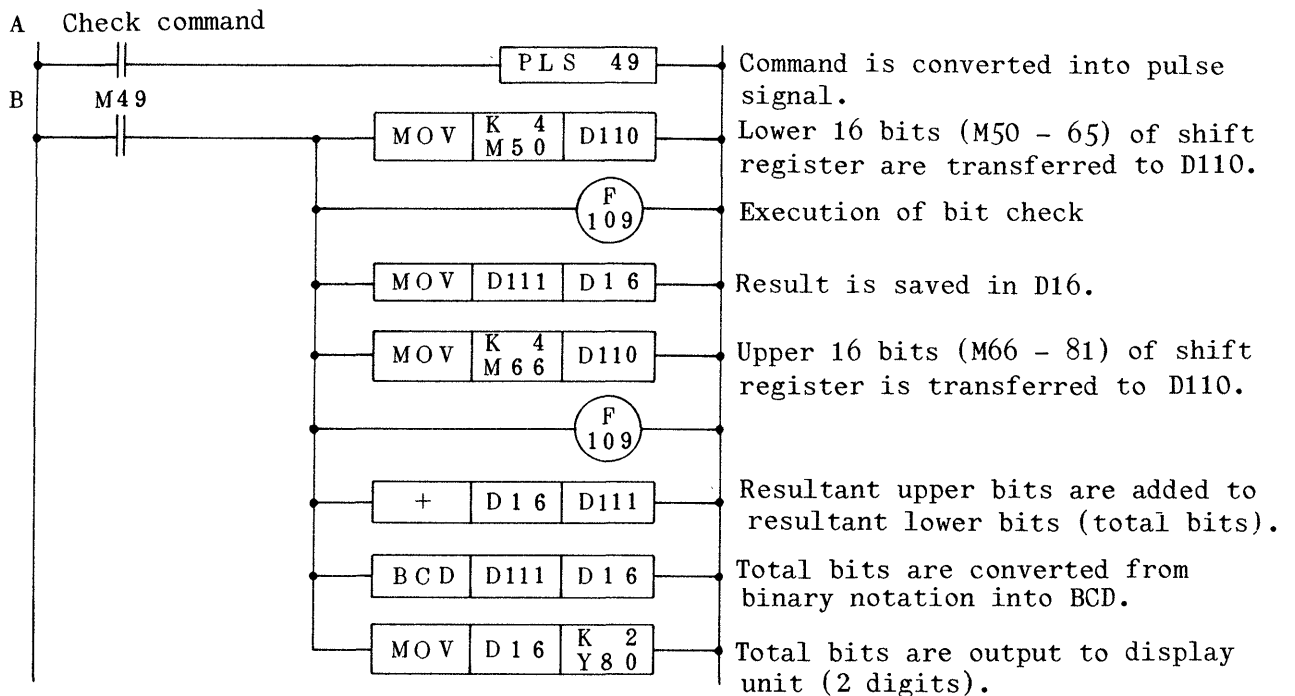


Fig. 4.34 Display of number of bits "1" in shift register

In the example shown in Fig. 4.34, number of stocked goods is numerically displayed. However, this circuit may be used for other control, such as quantitative comparison or definition of quantitative range of stored goods, by using instructions (>, <, =).

(2) Checking of number of bits "1" (ON) of output Y

In this example, number of bits being in "1" in output Y (4 bits) is checked.

This application example may be actually employed for error detection.

Ex.: Output Y120 - 12B 12 bits

Error exists when number of bits "1" is larger than 9 bits.

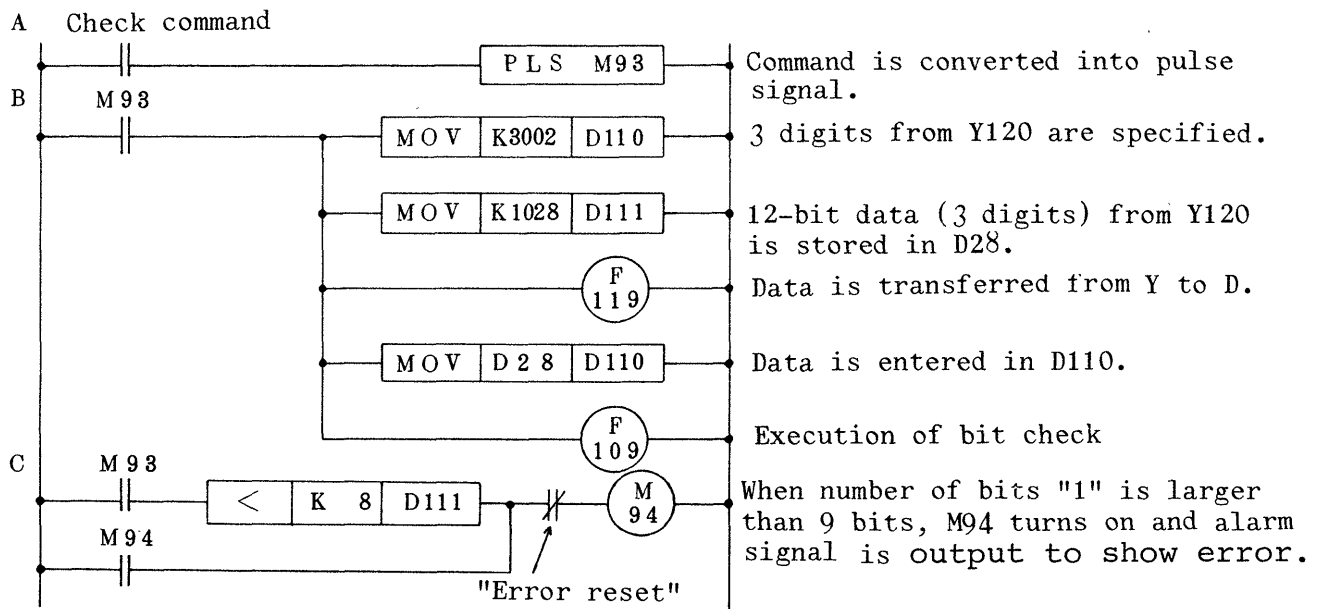


Fig. 4.35 Checking of number of bits "1" of output Y

4.13 Data inversion

With this function, the contents in the specified data register are inverted, that is, "1"'s complement is obtained.

4.13.1 Functions

Function No.: F100

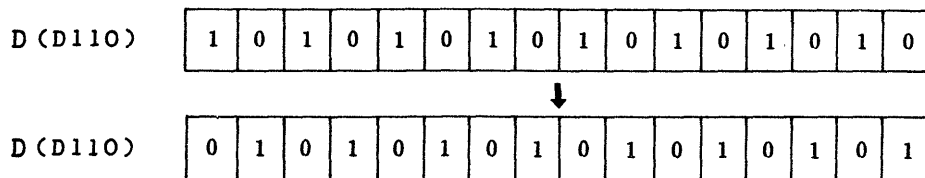


Fig. 4.36 Data inversion

4.13.2 Circuit applications

Ex.: Since the subtracted result is obtained as "2"'s complement when the result is minus, it must be changed to absolute value as follows:

When (D10) - (D20) is equal to (D10), for example,

$$5 - 7 = -2 \longrightarrow 2$$

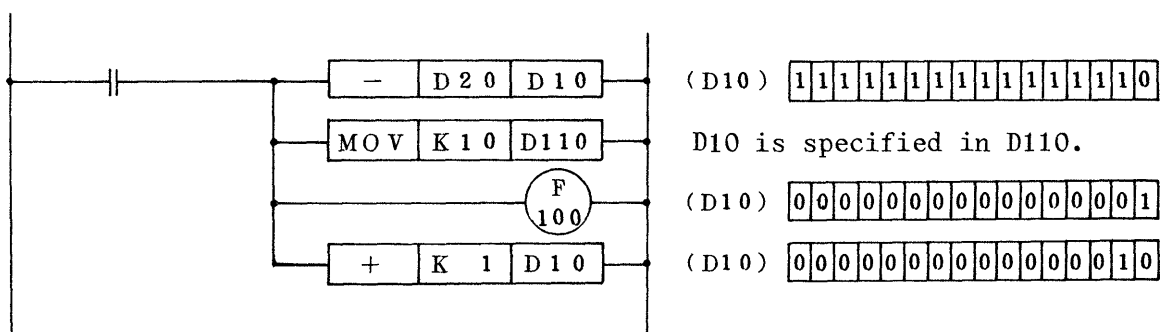


Fig. 4.47 Data inversion circuit

4.14 Application instruction executing times

The time taken for execution of each application instruction is as follows:

No.	Category of instruction	Name of function	Instruction	Executing time	Remarks
1	Application instruction	8-bit data association	OUT F110	140 μ S	
2		16-bit data disassociation	OUT F111	140 μ S	
3		16-bit data AND operation	OUT F112	170 μ S	
4		16-bit data OR operation	OUT F113	170 μ S	
5		Batch shift of M	OUT F114	10bits-250 μ S 50bits-630 μ S 100bits 1130 μ S	Liftward shift is identical to rightward shift
6		Batch shift of D	OUT F115	10data 340 μ S 30data 650 μ S 50data 970 μ S	
7		Batch reset of D	OUT F116	10data 290 μ S 30data 520 μ S 50data 750 μ S	
8		Indirect reading of T,C,D	OUT F117	T,C 220 μ S D 210 μ S	
9		Indirect writing of T,C,D	OUT F118	T,C 220 μ S D 210 μ S	
10		Y \rightarrow D Data transfer	OUT F119	4wds. 550 μ S 8wds. 670 μ S 12wds. 790 μ S 16wds. 870 μ S	
11		4 \leftrightarrow 16 Decode/encode	OUT F108	Decode 180 μ S Encode 310 μ S	
12		16-bit check	OUT F109	290 μ S	
13		Data inversion	OUT F100	100 μ S	

5. FUNCTIONS AND PRACTICAL USE OF HIGH-SPEED PROCESSING INSTRUCTIONS

The high-speed processing instructions are that a program component required to high-speed processing is picked up, and high frequently executed several times as subroutine during execution of the main program at low speed, thus the executing time can be apparently shortened.

5.1 High-speed processing instructions and application data registers

- (1) SET F126: High-speed program call instruction
- (2) RST F126: High-speed program return instruction
- (3) D126: Register for storing of high-speed processing program head step No. (when 10mS timer is used)
- (4) D123: Register for storing of high-speed processing program head step No. (when call instruction is used)

5.2 Circuit applications

The circuit shown below is an example of practical application of the high-speed processing instructions, where it is intended to minimize timing error when Y10 and Y11 turn off.

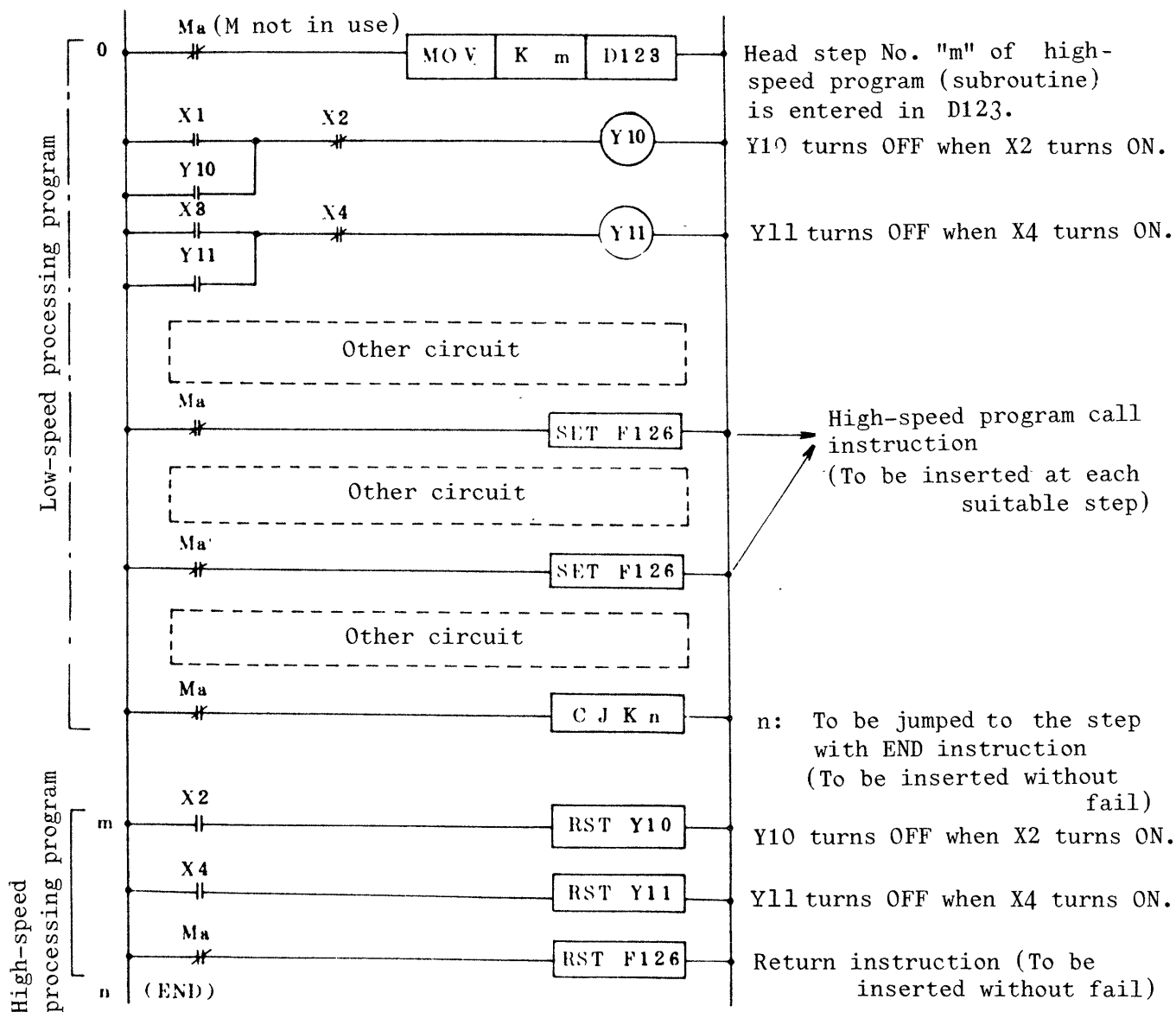


Fig. 5.1 High-speed processing circuit

- (1) In order to minimize timing error at the time $Y10$, $Y11$ turn off, only $Y10/Y11$ reset circuit is programmed for high speed processing.

Both $OUTY10/RSTY10$ and $OUTY11/RSTY11$ are allowed to exist

at the same time.

- (2) At the head of the low-speed processing program, the head step No. "m" of high-speed processing program component should be placed in D123, as shown in Fig. 5.1.
- (3) High-speed program call instruction F126 should be inserted with suitable step interval.

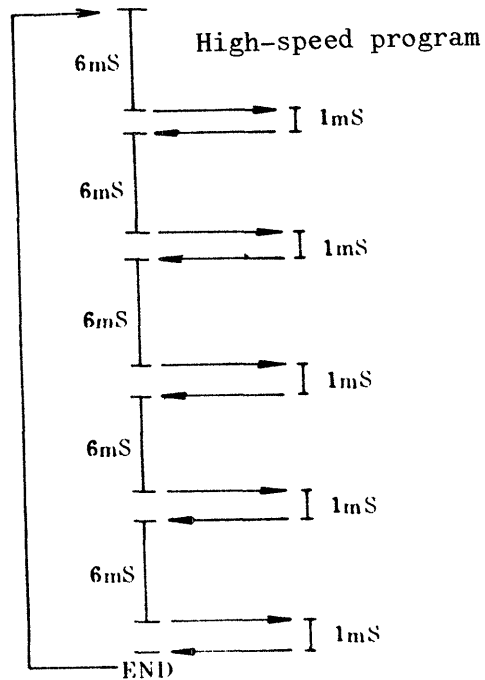
When high-speed processing must be performed in every 6mS, for example, the high-speed program should be inserted at every about 200 steps.

$$\frac{6\text{mS}}{30\mu\text{S}} = 200 \text{ steps}$$

- (4) At the end of the low-speed processing program, CJ instruction should be inserted without fail and program sequence should be jumped to the step with END instruction.
- (5) At the end of the high-speed processing instruction, return instruction F126 should be inserted without fail.
- (6) END instruction should come next to the return instruction. This means that the high-speed processing program should be inserted before the END instruction.
- (7) Program execution flow chart and execution time

Fig. 5.2 shows an example of execution flow chart and time, where the low-speed processing program has about 1000 steps and the high-speed processing program has 33 steps with call instruction inserted at every 200 steps as exemplified in above mentioned (3).

Low-speed program



Note 1: High-speed execution time:

$$33 \times 30\mu\text{S} = 1\text{mS}$$

Note 2: Scanning time: 35mS

Note 3: When scanning time exceeds 100mS, WDT error occurs.

Fig. 5.2 Program execution flow chart and execution time (When used with call instruction)

5.3 Program for 10mS high-precision timer

96 timers (100mS), T0 - T95, and 32 timers (10mS), T96 - T127, are incorporated in MELSEC-KOJ.

Fig. 5.3 shows an example of circuit used in programming for 10mS precision timers.

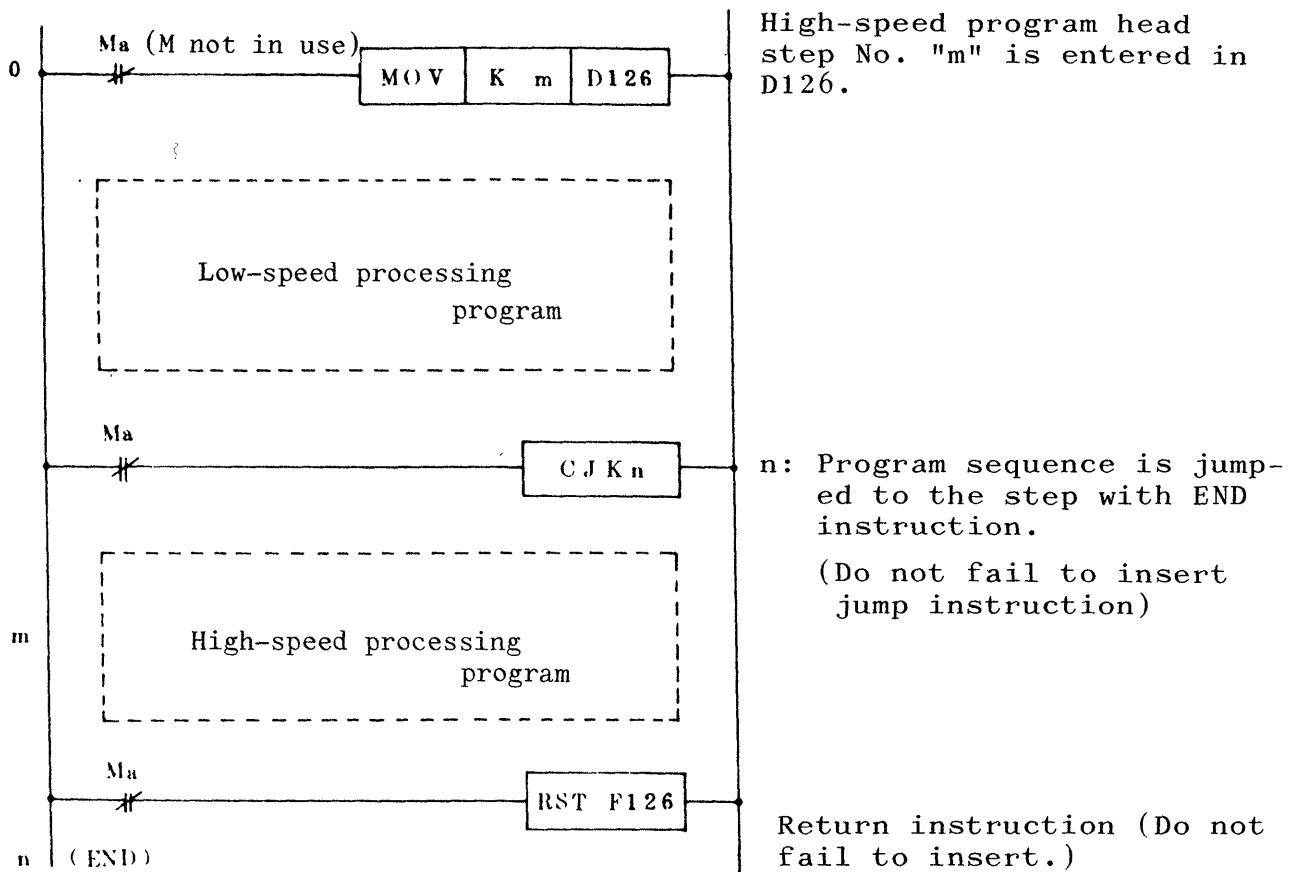


Fig. 5.3 Circuit for programming with 10mS timers

- (1) When 10mS timer is used, the high-speed program head step No. "m" should be entered in D126 without fail. (High-speed program is called at every 10mS when D126 is not in "0".) Notice that the number is different from D123 of paragraph 5.2.
- (2) CJ instruction should be inserted at the end of low-speed program to let the program sequence jump to the step with END instruction.
- (3) Return instruction RST F126 should be inserted at the end of high-speed program.
- (4) Coil of 100mS timer (T0 - T95) should be inserted in the low-speed program, and coil of 10mS timer (T96 - T127) in

the high-speed program.

(5) Program execution flow chart and execution time

Fig. 5.4 shows an example of execution flow chart and execution time, where the low-speed processing program has about 1000 steps and the high-speed processing program has 33 steps (same as the example shown in above mentioned (7) of 5.2).

In this example, execution times are 30mS and 1mS for low-speed program and high-speed program respectively.

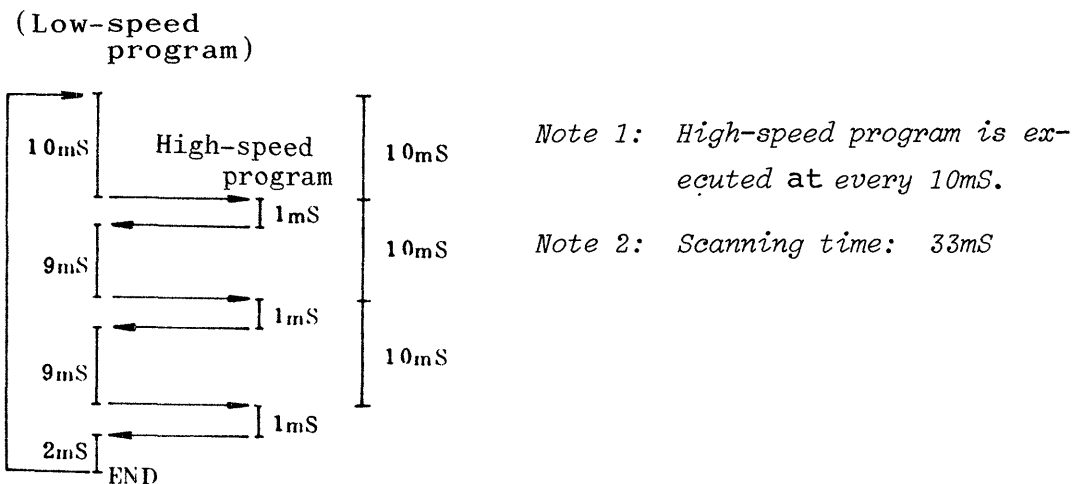


Fig. 5.4 Program execution flow chart and execution time (When used with 10mS timer)

(6) High-speed timer function may be associated with call instruction SET function F126.

When the both functions are used at the same time, high-speed program head step No. "m" should be entered in D126 as well as D123.

In the case where the example in Fig. 5.2 is combined with the example in Fig. 5.4, the high-speed program is called 3 times and the scanning time is 33mS.

5.4 Programming error display

The following two types of error check are available in the programming related to high-speed processing program.

When error is found, "RUN" display flickers.

- (1) High-speed processing program time over (Error No.: 5030)

If execution time exceeds 10mS in high-speed processing (timer 10mS is used), the time over error occurs.

◎ The high-speed processing should be executed within 10mS.

- (2) Programming error (Error No.: 5031)

This error occurs when CJ Kn, RST F126 shown in Fig. 5.1 and 5.3 are not entered.

Note: Error No. may be verified in test mode of PU and GPP.

