YASKAWA

FS100 OPTIONS
INSTRUCTIONS

INFORM EXTENTION FUNCTION
STRUCTURED PROGRAM LANGUAGE

Upon receipt of the product and prior to initial operation, read these instructions thoroughly, and retain for future reference.

MOTOMAN INSTRUCTIONS

MOTOMAN-□□□ INSTRUCTIONS
FS100 INSTRUCTIONS
FS100 OPERATOR'S MANUAL
FS100 MAINTENANCE MANUAL

Part Number: 159643-1CD
Revision: 0

YASKAWA ELECTRIC CORPORATION

YASKAWA

MANUAL NO.
HW1480732

MANDATORY

- This manual explains the structured program language for INFORM extension function of the FS100 system. Read this manual carefully and be sure to understand its contents before handling the FS100.
- General items related to safety are listed in Chapter 1: Safety of the FS100 Instructions. To ensure correct and safe operation, carefully read the FS100 Instructions before reading this manual.

CAUTION

- Some drawings in this manual are shown with the protective covers or shields removed for clarity. Be sure all covers and shields are replaced before operating this product.
- The drawings and photos in this manual are representative examples and differences may exist between them and the delivered product.
- YASKAWA may modify this model without notice when necessary due to product improvements, modifications, or changes in specifications.
  If such modification is made, the manual number will also be revised.
- If your copy of the manual is damaged or lost, contact a YASKAWA representative to order a new copy. The representatives are listed on the back cover. Be sure to tell the representative the manual number listed on the front cover.
- YASKAWA is not responsible for incidents arising from unauthorized modification of its products. Unauthorized modification voids your product's warranty.

Notes for Safe Operation

Read this manual carefully before installation, operation, maintenance, or inspection of the FS100.

In this manual, the Notes for Safe Operation are classified as “WARNING”, “CAUTION”, “MANDATORY”, or “PROHIBITED”.

 WARNING	Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury to personnel.
 CAUTION	Indicates a potentially hazardous situation which, if not avoided, could result in minor or moderate injury to personnel and damage to equipment. It may also be used to alert against unsafe practices.
 MANDATORY	Always be sure to follow explicitly the items listed under this heading.
 PROHIBITED	Must never be performed.

Even items described as “CAUTION” may result in a serious accident in some situations.

At any rate, be sure to follow these important items.

To ensure safe and efficient operation at all times, be sure to follow all instructions, even if not designated as “CAUTION” and “WARNING”.

WARNING

- Confirm that no person is present in the manipulator's operating range and that you are in a safe location before:
  - Turning ON the FS100 power.
  - Moving the manipulator with the programming pendant.
  - Running the system in the check mode.
  - Performing automatic operations.

Injury may result if anyone enters the manipulator's operating range during operation. Always press the emergency stop button immediately if there is a problem. The emergency stop button is located on the right of the programming pendant.

- Observe the following precautions when performing teaching operations within the manipulator's operating range:
  - View the manipulator from the front whenever possible.
  - Always follow the predetermined operating procedure.
  - Keep in mind the emergency response measures against the manipulator's unexpected motion toward you.
  - Ensure that you have a safe place to retreat in case of emergency.

Improper or unintended manipulator operation may result in injury.

- Before operating the manipulator, check that servo power is turned OFF when the emergency stop button on the programming pendant is pressed.
  When the servo power is turned OFF, the SERVO ON LED on the programming pendant is turned OFF.

Injury or damage to machinery may result if the emergency stop circuit cannot stop the manipulator during an emergency. The manipulator should not be used if the emergency stop button does not function.

Fig. : Emergency Stop Button

- In the case of not using the programming pendant, be sure to supply the emergency stop button on the equipment. Then before operating the manipulator, check to be sure that the servo power is turned OFF by pressing the emergency stop button.
  Connect the external emergency stop button to the 5-6 pin and 16-17 pin of the robot system signal connector (CN2).
- Upon shipment of the FS100, this signal is connected by a jumper cable in the dummy connector. To use the signal, make sure to supply a new connector, and then input it.

If the signal is input with the jumper cable connected, it does not function, which may result in personal injury or equipment damage.

WARNING

- Once the emergency stop button is released, clear the cell of all items which could interfere with the operation of the manipulator. Then turn the servo power ON.

Injury may result from unintentional or unexpected manipulator motion.

Fig. : Release of Emergency Stop

CAUTION

- Perform the following inspection procedures prior to conducting manipulator teaching. If problems are found, repair them immediately, and be sure that all other necessary processing has been performed.
  - Check for problems in manipulator movement.
  - Check for damage to insulation and sheathing of external wires.
- Always return the programming pendant to the hook on the cabinet of the FS100 after use.

The programming pendant can be damaged if it is left in the manipulator's work area, on the floor, or near fixtures.

- Read and understand the Explanation of Warning Labels in the FS100 Instructions before operating the manipulator:

Definition of Terms Used Often in This Manual

The MOTOMAN is the YASKAWA industrial robot product.

The MOTOMAN usually consists of the manipulator, the FS100 controller, manipulator cables, the FS100 programming pendant (optional), and the FS100 programming pendant dummy connector (optional).

In this manual, the equipment is designated as follows:

Equipment	Manual Designation
FS100 controller	FS100
FS100 programming pendant	Programming pendant
Cable between the manipulator and the controller	Manipulator Cable
FS100 programming pendant dummy connector	Programming pendant dummy connector

Descriptions of the programming pendant keys, buttons, displays and keyboard of the PC are shown as follows:

Equipment		Manual Designation
Programming Pendant	Character Keys	The keys which have characters printed on them are denoted with [ ]. e.g. [ENTER]
	Symbol Keys	The keys which have a symbol printed on them are not denoted with [ ] but depicted with a small picture. e.g. PAGE key  The cursor key is an exception, and a picture is not shown.
	Axis Keys Numeric Keys	“Axis keys” and “Numeric keys” are generic names for the keys for axis operation and number input.
	Keys Pressed Simultaneously	When two keys are to be pressed simultaneously, the keys are shown with a “+” sign between them. e.g. SHIFT key  + COORD key 
	Mode Key	Three kinds of modes that can be selected by the mode key are denoted as follows: REMOTE, PLAY, or TEACH
	Button	Three buttons on the upper side of the programming pendant are denoted as follows: HOLD button START button EMERGENCY STOP button
	Displays	The menu displayed in the programming pendant is denoted with { }. e.g. {JOB}
PC Keyboard		The name of the key is denoted. e.g. Ctrl key on the keyboard

Description of the Operation Procedure

In the explanation of the operation procedure, the expression “Select • • • “ means that the cursor is moved to the object item and the SELECT key is pressed, or that the item is directly selected by touching the screen.

Registered Trademark

In this manual, names of companies, corporations, or products are trademarks, registered trademarks, or brand names for each company or corporation. The indications of (R) and TM are omitted.

1 Function of Structured Language

[Structured language] is for making easier to editing job by supporting Branch, Selection, repeat instruction.

1. Branch

   IF ~ THEN ~ ELSEIF ~ ELSE ~ ENDIF

2. Selection

   SWITCH ~ CASE ~ DEFAULT ~ ENDSWITCH

3. Repeat

   FOR ~ NEXT

   WHILE ~ ENDWHILE

2 Instructions Detail

2.1 Branch

- **IF ~ THEN ~ ELSEIF ~ ELSE ~ ENDIF**

Function

It is flow control statement to control branch execution based on condition of equation result.

Syntax

IF *condition* **THEN**

[*statements*]

[ELSEIF *condition-n* **THEN**]

[*elseifstatements*]] . . .

[ELSE]

[*elsestatements*]]

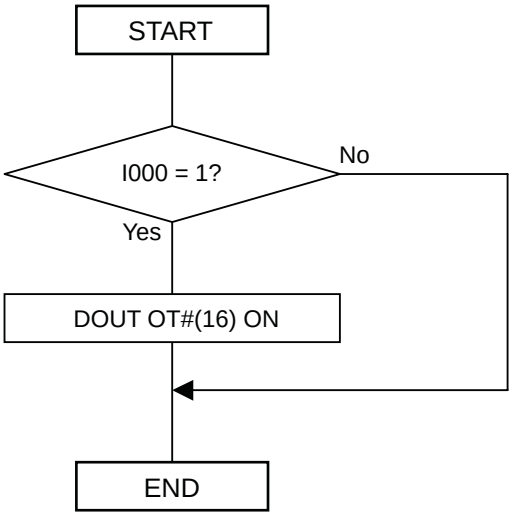
ENDIF

Designated Item	**Contents**
condition	Condition equation for checking (True) or (False) Up to 3 individual conditions can be used by connecting “AND” or “OR” condition.
statements	If *condition* is (True), then this statement is executed.
condition-n	Same as above described *condition*.
elseifstatements	If *condition-n* is (True), then this statement is executed.
elsestatements	If All defined conditions before Else are false, then this is executed.

Example 1 • Basic example

(1) If I000 = '1', then OT#(16) is ON.

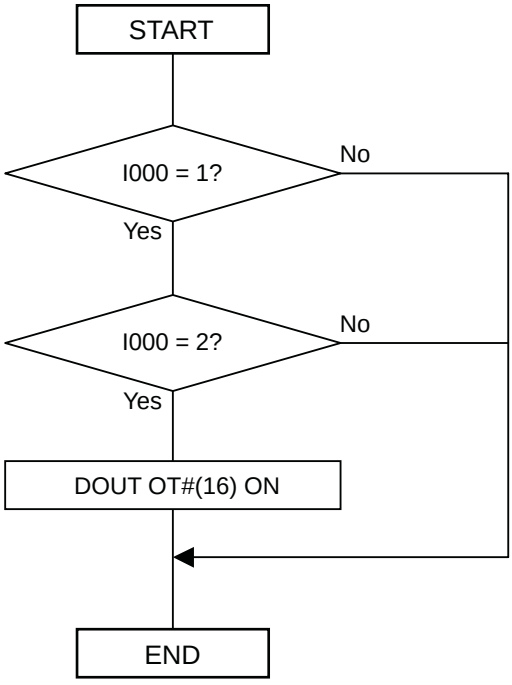
```
IF  I000 = 1 THEN
   DOUT  OT#(16)  ON
ENDIF
```



Example 2 • AND

(1) IF I000= '1' and I001 = '2', then OT#(16) is ON.

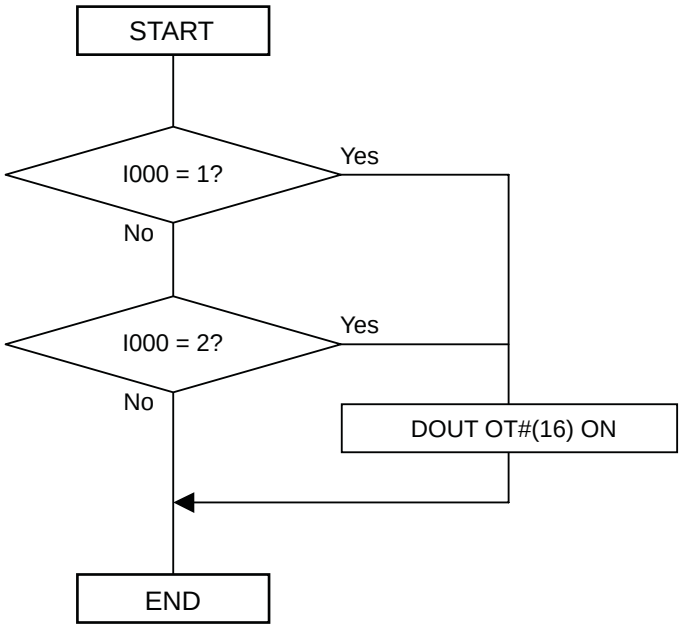
```
IF  I000 = 1  AND  I001 = 2  THEN
   DOUT  OT#(16)  ON
ENDIF
```



Example 3 • OR

(1) If I000 = '1' or I001 = '2', then OT#(16) is ON.

```
IF  I000 = 1  OR  I001 = 2  THEN
   DOUT  OT#(16)  ON
ENDIF
```

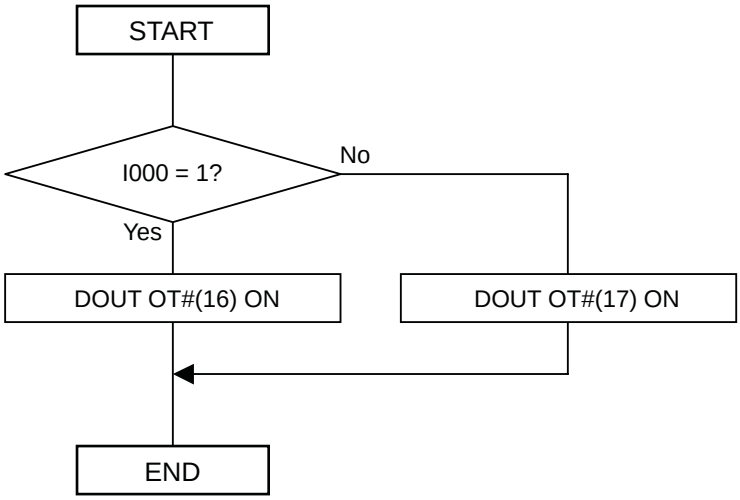


Example 4 • ELSE

(1) IF I000 = '1' then OT#(16) is ON.

(2) Otherwise OT#(17) is ON.

```
IF  I000 = 1  THEN
   DOUT  OT#(16)  ON
ELSE
   DOUT  OT#(17)  ON
ENDIF
```

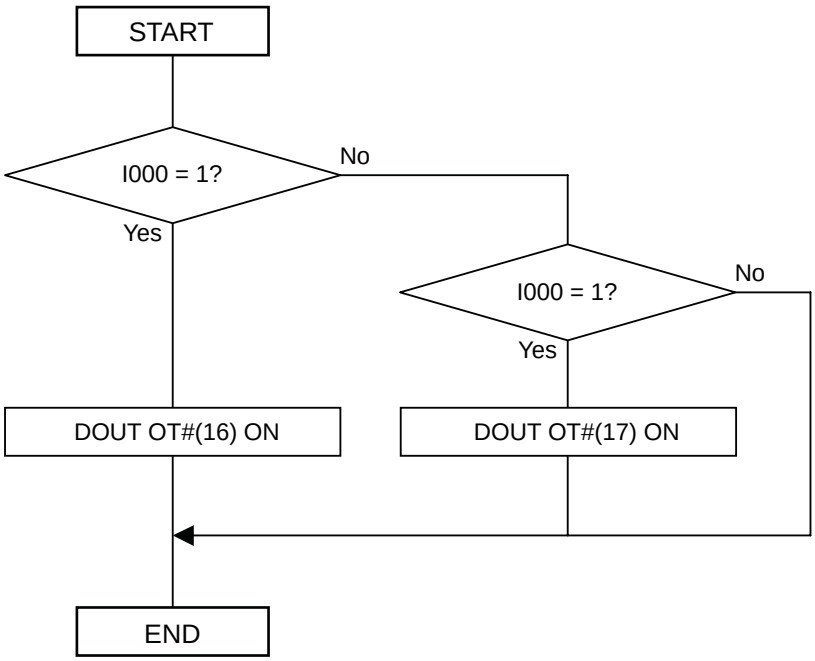


Example 5 • ELSEIF

(1) If I000 = '1' then OT#(16) is ON.

(2) Else if I001 = '1', then OT#(17) is ON.

```
IF  I000 = 1  THEN
   DOUT  OT#(16)  ON
ELSEIF  I000 = 1  THEN
   DOUT  OT#(17)  ON
ENDIF
```



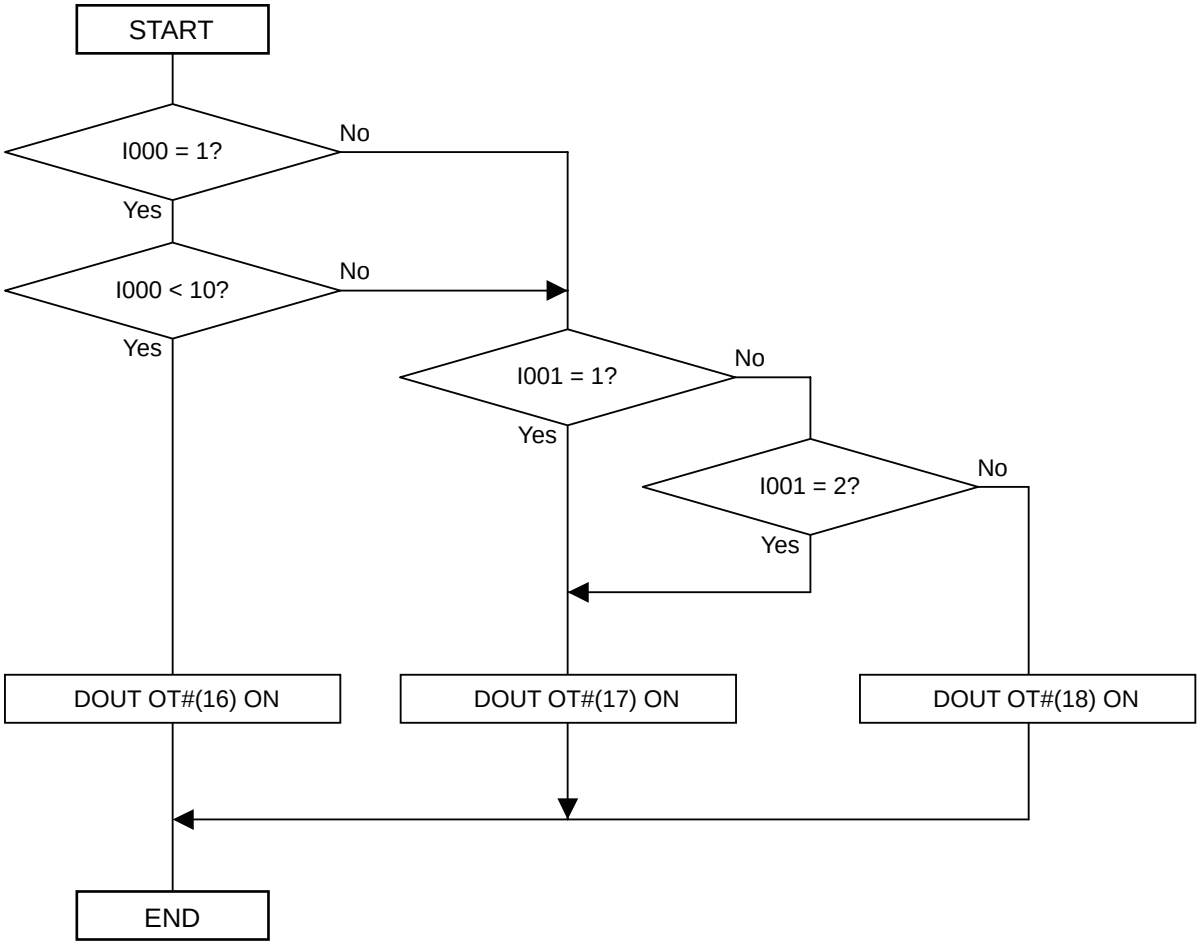
Example 6 • Multi condition

(1) If I000 >= '1' and I000 < 10, then OT#(16) is ON.

(2) Else if I001 = '1' or '2', then OT#(17) is ON.

(3) Else OT#(18) is ON.

```
IF  I000 >= 1  AND  I000 < 10  THEN
   DOUT  OT#(16)  ON
ELSEIF  I001 = 1  OR  I001 = 2  THEN
   DOUT  OT#(17)  ON
ELSE
   DOUT  OT#(18)  ON
ENDIF
```



2.2 Selection

- **SWITCH ~ CASE ~ DEFAULT ~ ENDSWITCH**

Function

Selection one of the statement based on condition result.

Syntax

SWITCH (*testexpression*) **CASE** *expressionlist*

[*statements*]

[CASE *expressionlist-n*]

[*statements-n*]] . . .

[DEFAULT]

[*elsestatements-n*]]

ENDSWITCH

Designated Item	Contents
condition	Specify an arbitrary expression.
expressionlist	Must be set.
statements	If *testexpression* corresponds *expressionlist,* then this is executed.
expressionlist-n	If needed CASE must be set.
statements-n	If *testexpression* corresponds one of the *expressionlist-n*, then corresponded statements-n is executed.
elsestatements	If *testexpression* doesn't correspond any CASE, then this *elsestatements* is executed.

Only "I variable" can be used for testexpression

SWITCH (I000) **CASE** 0

Explanation

If Argument testexpression corresponds any of the **CASE** expressionlist, then corresponded statement described before next **CASE or ENDSWITCH is executed**. After that flow is going to next line of **ENDSWITCH** line. **DEFAULT** statement is executed if no condition is corresponded.

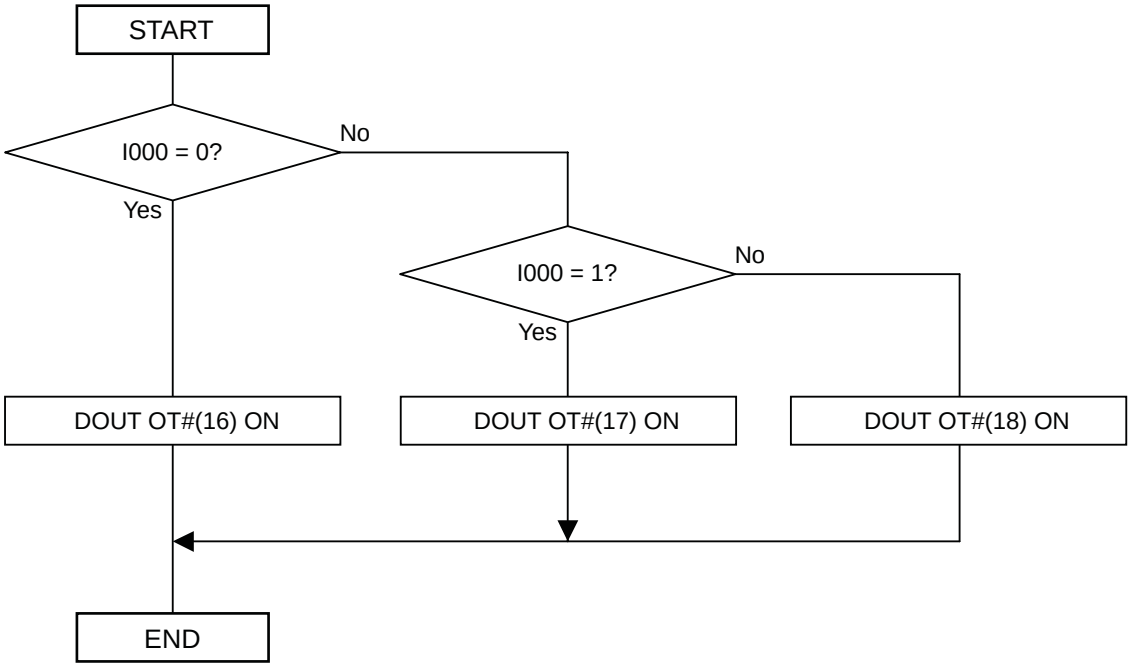
Example

(1) IF I000 = '0', OT#(16) is ON.

(2) IF I000 = '1', OT#(17) is ON.

(3) IF I000 is other than '0' or '1', OT#(18) is ON.

```
SWITCH (I000) CASE 0
  DOUT OT#(16) ON
CASE 1
  DOUT OT#(17) ON
DEFAULT
  DOUT OT#(18) ON
ENDSWITCH
```



2.3 Repeat

- **FOR ~ NEXT**

Function

Repeating one set of the statement by designated number of loop.

Syntax

FOR *counter* = *start* **TO** *end* [**STEP** *stepcount*]

[*statements*]

NEXT *counter*

Designated Item	Contents
counter	Loop counter variable
start	Start value of the counter
end	End value of the Counter
statements	One set of the statement of executed repeatedly
stepcount	Count up vale by each loop

Only "I variable" can be used for counter

FOR (I000) = 1 **TO** 10

Explanation

The one set of the statement is executed repeatedly until the counter value is over the end value.

The counter is set start value as initial value in the beginning.

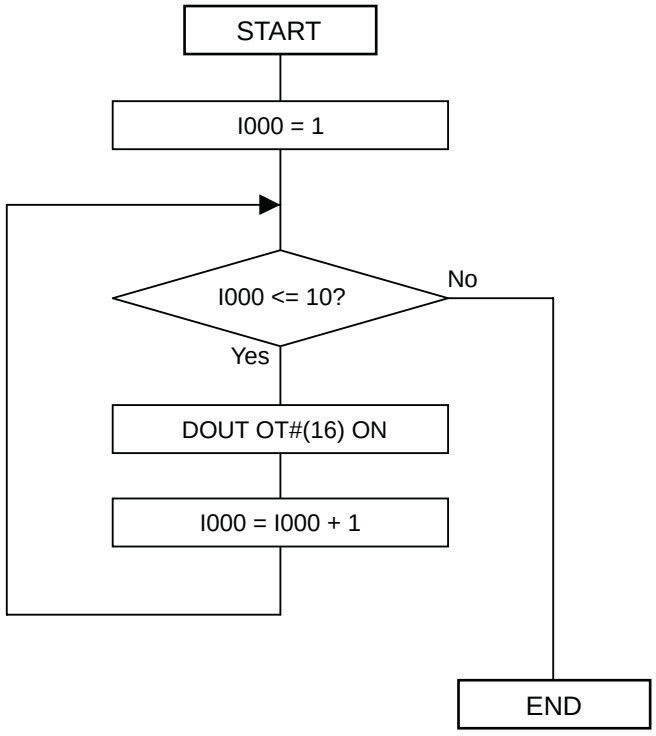
If without [STEP stepcount] parameter, step value = 1.

If [STEP stepcount] exists, the counter is added the stepcount at every loop times.

Example

(1) I000 is used for loop counter, and statements of OT#(16) setting ON are executed 10 times. (As a result, OT#(16) is turned ON 10 times.)

```
FOR I000 = 1 TO 10
   DOUT OT#(16) ON
NEXT I000
```



• **WHILE ~ ENDWHILE**

Function

While designated condition is (True), one set of the statements are executed repeatedly.

Syntax

WHILE *condition*

[*statements*]

ENDWHILE

Designated Item	Contents
condition	Condition for checking (True) or (False). Using AND or OR enables you to specify three conditions in a conditional expression.
statements	Executed statements while the *condition* is (True).

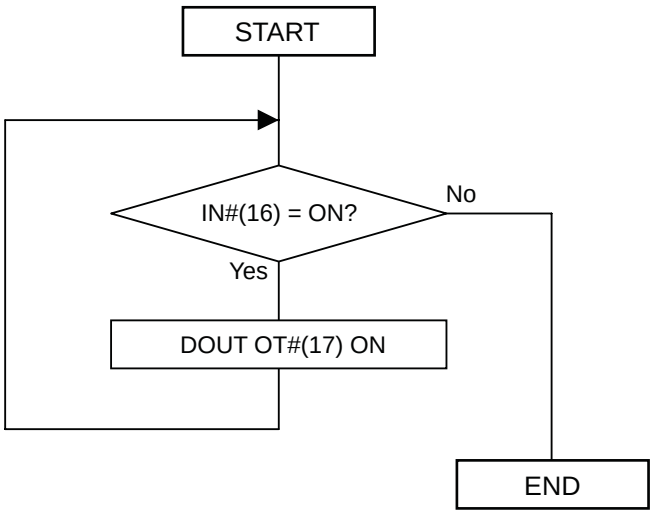
Explanation

While condition is (True), all of the statement before ENDWHILE are executed. When execution reaches ENDWHILE, control returns to WHILE again, and the argument named “condition” is checked. This processing is repeated while this argument is true. If the argument is not true, control moves to the line after ENDWHILE.

Example

(1) While IN#(16) is ON, OT#(17) is ON repeatedly. As a result, OT#(17) is repeatedly turned ON until IN#(16) is turned OFF.

```
WHILE  IN#(16) = ON
   DOUT  OT#(17)  ON
ENDWHILE
```



3 Editing (Insertion into JOB and Modification)

3.1 Insert into JOB

Push the [Inform list] in job editing screen, and push one of the [IF], [SWITCH], [FOR] or [WHILE].

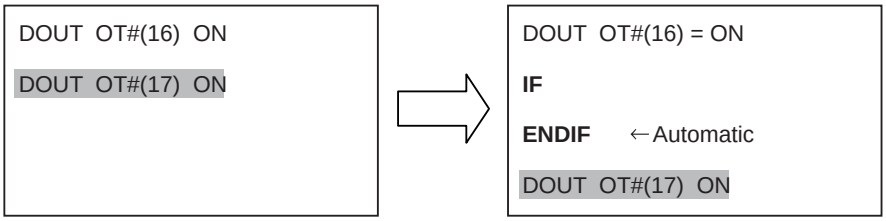
NOTE

Before using a structured language, set {Expanded} to {Language level} on the teaching condition window displayed by selecting {Install controller} and then {Specify teaching conditions} in the {Teach} mode.

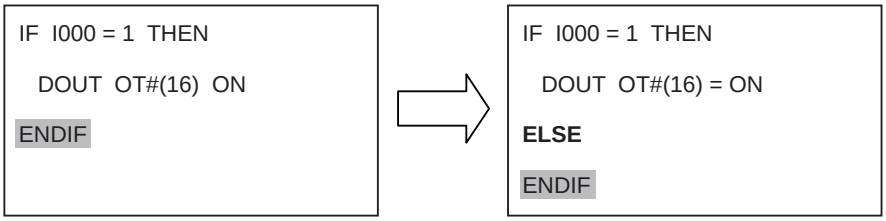
∗ **IF ~ ENDIF: Insert IF instruction**

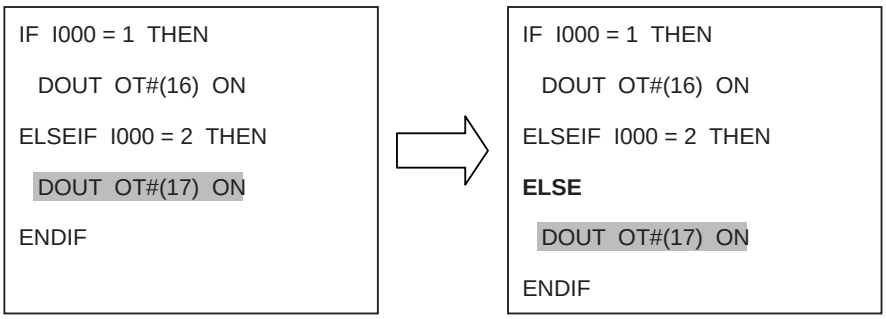
Selecting [IF]	: Also [ENDIF] is inserted automatically.
Selecting [ELSE]	: To Insert this, registered [IF ~ THEN] or [ELSEIF] is needed.
Selection [ELSEIF]	: To insert this, registered [IF ~ THEN] or [ELSEIF] is needed But if just before line of insertion is [ENDIF] or [ELSE], insertion cannot be done.
[ENDIF]	: It is inserted automatically when [IF] is inserted.

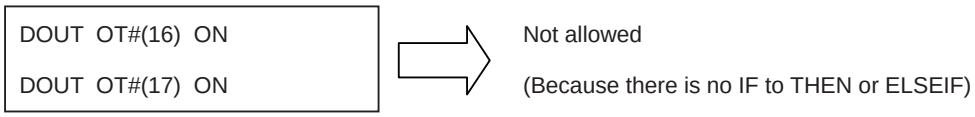
Example) Inserting [IF]



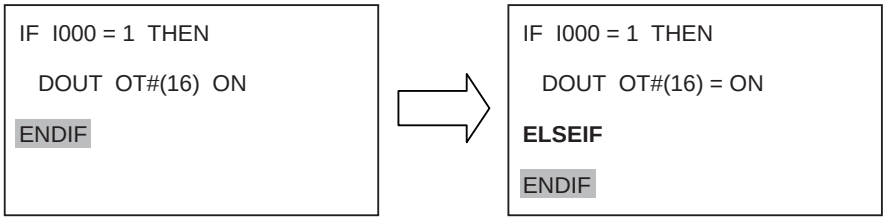
Example) Inserting [ELSE]

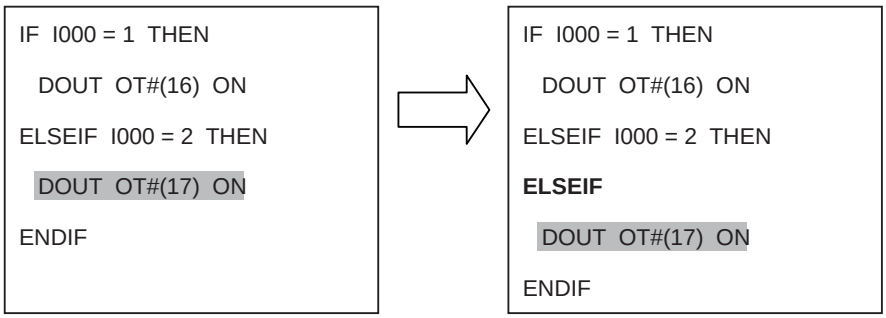


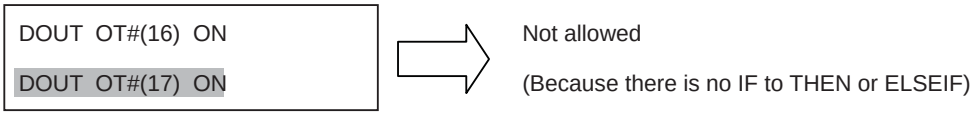


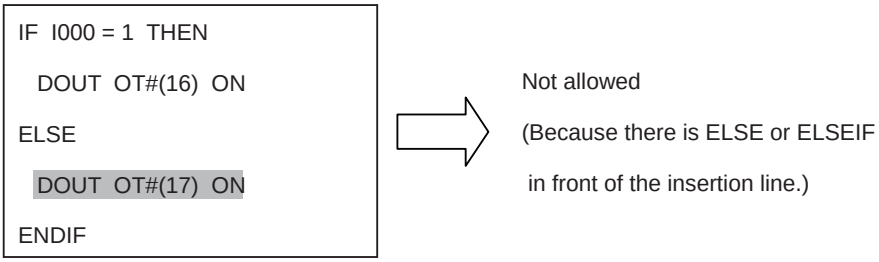


Example) Inserting [ELSEIF]





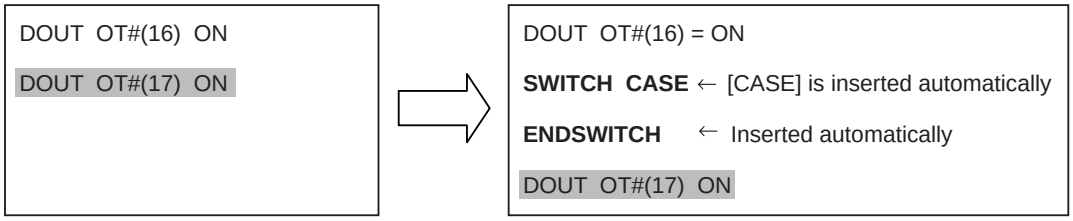




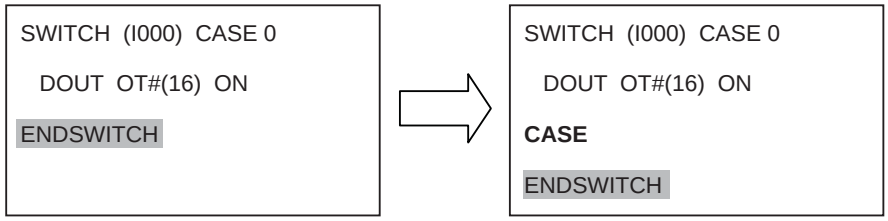
∗ **Insert SWITCH ~ ENDSWITCH**

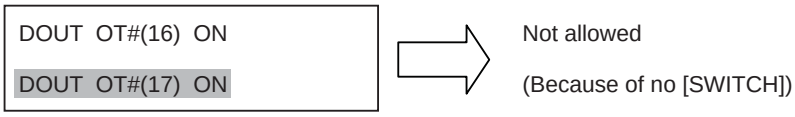
Selecting [SWITCH]	: Also [CASE] and [ENDSWITCH] are inserted automatically.
Selecting [CASE]	: To insert it, [SWITCH] should be inserted before.
Selecting [DEFAULT]	: To insert it, [SWITCH] should be inserted before. But it is not allowed if [CASE] exists below of insertion line.
[ENDSWITCH]	: It is inserted automatically when [SWITCH] is inserted.

Example) Inserting [SWITCH]

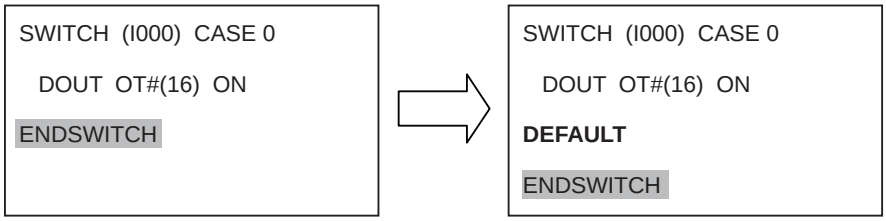


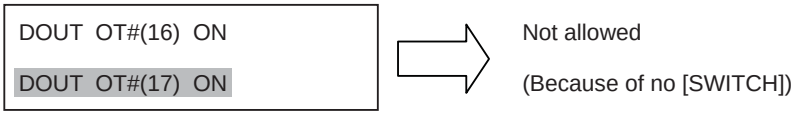
Example) Inserting [CASE]

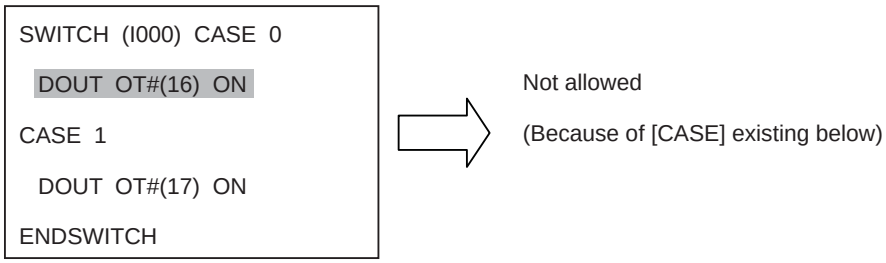




Example) Inserting [DEFAULT]



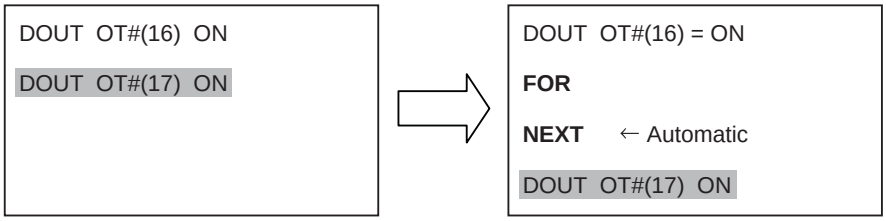




∗ **Insert FOR ~ NEXT**

Selecting [FOR] : Also [NEXT] is inserted automatically.
[NEXT] : Inserted it automatically at the time of [FOR] insertion.

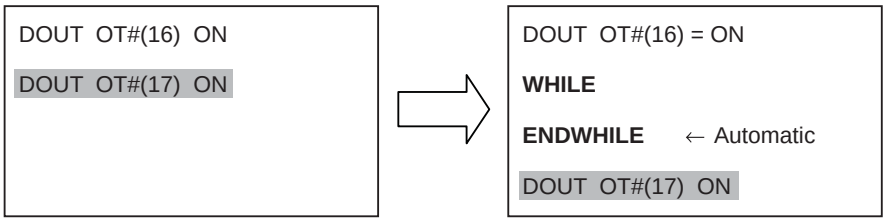
Example) Inserting [SWITCH]



∗ **Insert WHILE ~ ENDWHILE**

Selecting [WHILE] : Also [ENDWHILE] is inserted automatically.
[ENDWHILE] : Inserted it automatically at the time of [WHILE] insertion.

Example) Inserting [WHILE]



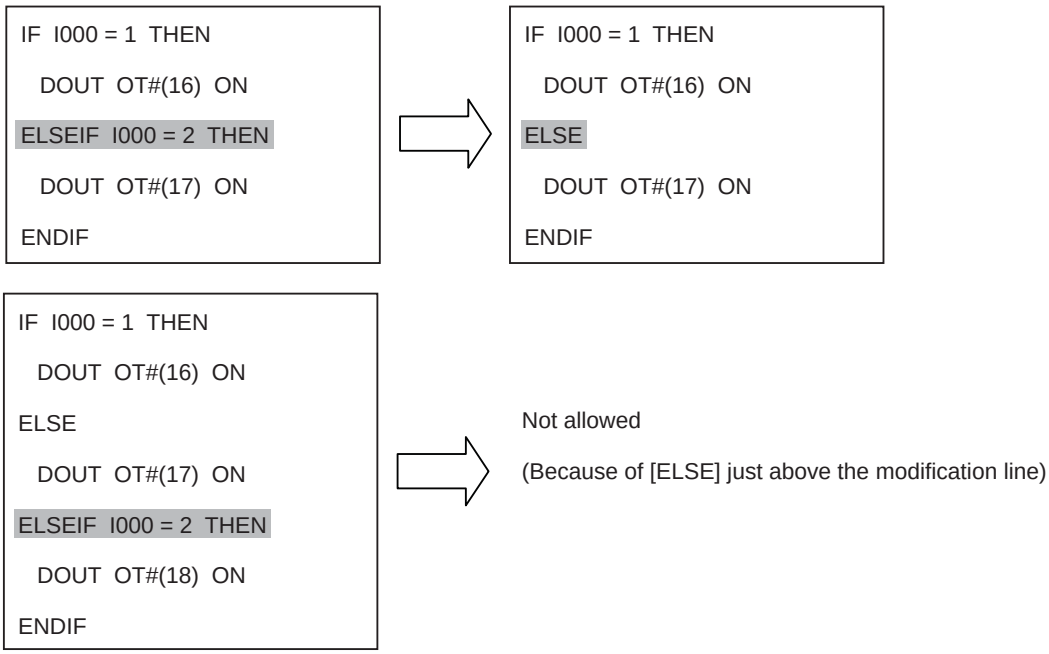
3.2 Modify Structured Instruction in JOB

Select [edit] in the JOB edit screen, and select one of the structured instruction.

* **Modify IF ~ THEN**

[IF]	: Only allowed modification is like as line editing.
[ELSE]	: Not allowed to modify
[ELSEIF]	: Only allowed modification is like as line editing.
[ENDIF]	: Not allowed to modify

Example) [IF] → [ELSE]



∗ **Modify SWITCH ~ ENDSWITCH**

[SWITCH] : Only allowed modification is like as line editing.

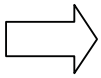
[CASE] : Possible to modify, but in case of other [CASE] existing after modifying [CASE], [CASE]→[DEFAULT] can not be changed.

[DEFAULT] : Not allowed to modify

[ENDSWITCH] : Not allowed to modify

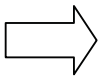
Example) [CASE] → [DEFAULT]

```
SWITCH (I000) CASE 0
  DOUT OT#(16) ON
CASE 1
  DOUT OT#(17) ON
CASE 2
  DOUT OT#(18) ON
ENDSWITCH
```



```
SWITCH (I000) CASE 0
  DOUT OT#(16) ON
CASE 1
  DOUT OT#(17) ON
DEFAULT
  DOUT OT#(18) ON
ENDSWITCH
```

```
SWITCH (I000) CASE 0
  DOUT OT#(16) ON
CASE 1
  DOUT OT#(17) ON
CASE 2
  DOUT OT#(18) ON
ENDSWITCH
```



Not allowed

(Because of [CASE] existing below of the modifying line)

∗ **Modify FOR ~ NEXT**

[FOR] : Only allowed modification is like as line editing.

[NEXT] : Not allowed to modify

∗ **Modify WHILE ~ ENDWHILE**

[WHILE] : Only allowed modification is like as line editing.

[ENDWHILE] : Not allowed to modify

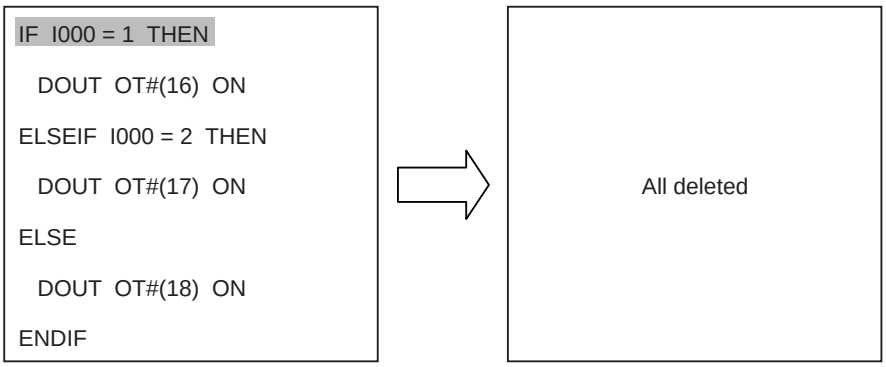
3.3 Delete Structured Instruction from JOB

Select [delete] in the JOB editing screen and select structured instruction wanted to delete.

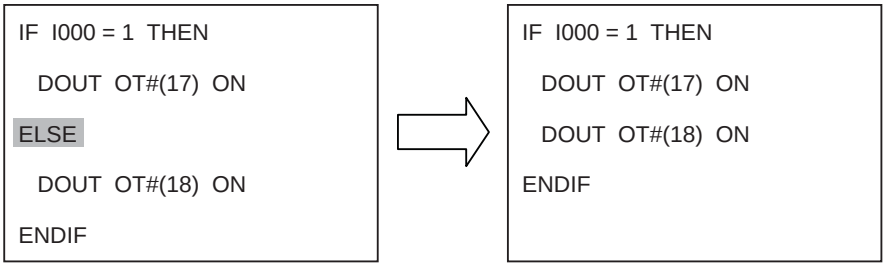
* **Delete IF ~ ENDIF**

[IF]	: Delete all line from [IF] to [ENDIF] corresponded the [IF]
[ELSE]	: Allowed to delete
[ELSEIF]	: Allowed to delete
[ENDIF]	: Not allowed to delete just [ENDIF]

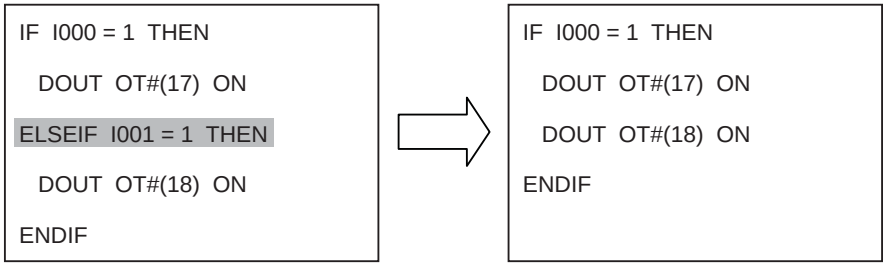
Example) Deleting [IF]



Example) Deleting [ELSE]



Example) Deleting [ELSEIF]



∗ **Delete SWITCH ~ ENDSWITCH**

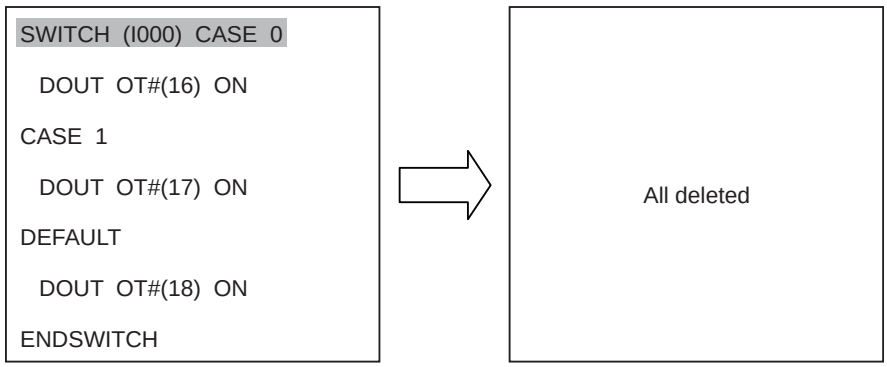
[SWITCH CASE] : Delete all line from [SWITCH] to [ENDSWITCH] corresponded the selected [SWITCH].

[CASE] : Allowed to delete

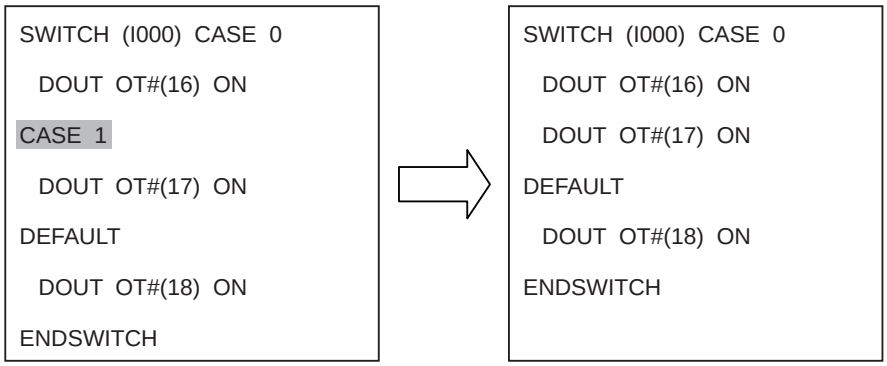
[DEFAULT] : Allowed to delete

[ENDSWITCH] : Not allowed to delete just [ENDSWITCH]

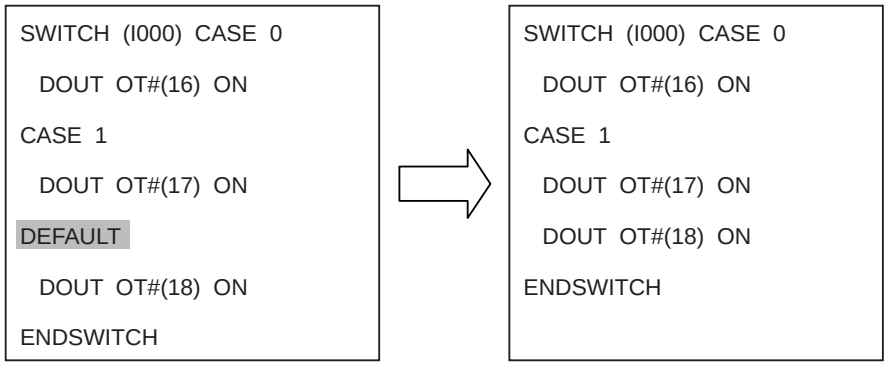
Example) Deleting [SWITCH]



Example) Deleting [CASE]



Example) Deleting [DEFAULT]



* **Delete FOR ~ NEXT**

[FOR]	: Delete all line from [FOR] to [NEXT] corresponded the selected [FOR].
[NEXT]	: Not allowed to delete

* **Delete WHILE ~ ENDWHILE**

[WHILE]	: Delete all line from [WHILE] to [ENDWHILE] corresponded the selected [WHILE].
[ENDWHILE]	: Not allowed to delete

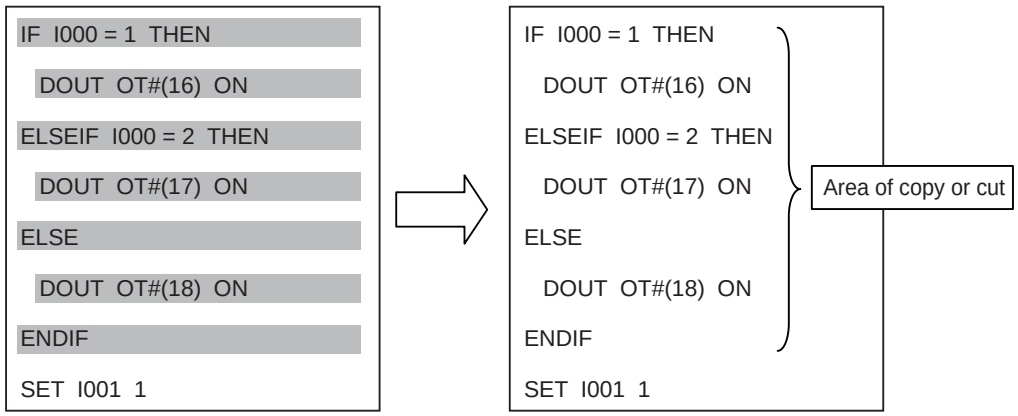
3.4 Copy or Cut and Paste Structured Instruction

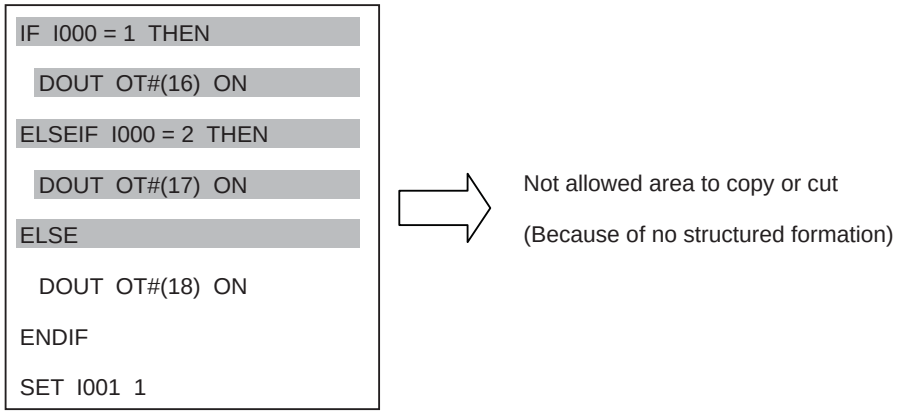
Designate structured instruction in the JOB editing screen and select [COPY] or [CUT].

* Reverse paste cannot be done if structured instruction included in the selected area.

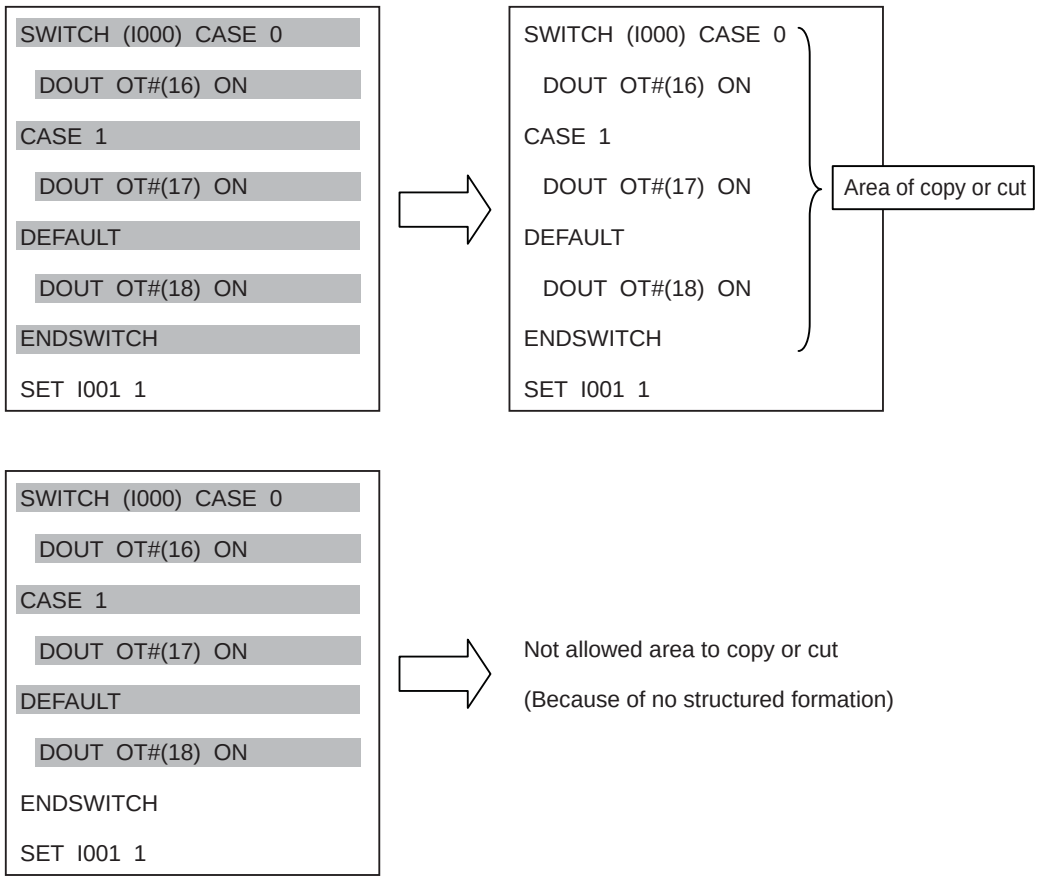
* The area not to be formation of the structure cannot be copied or cut.

∗ **Copy or cut IF ~ ENDIF**

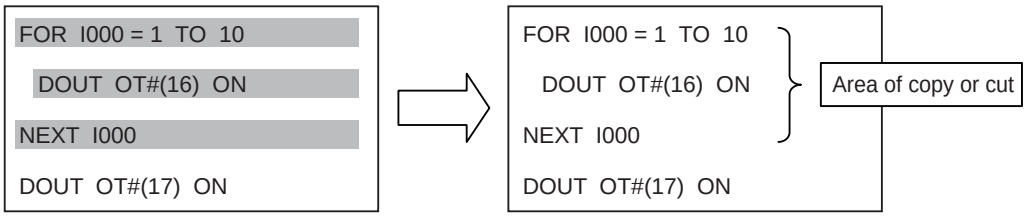


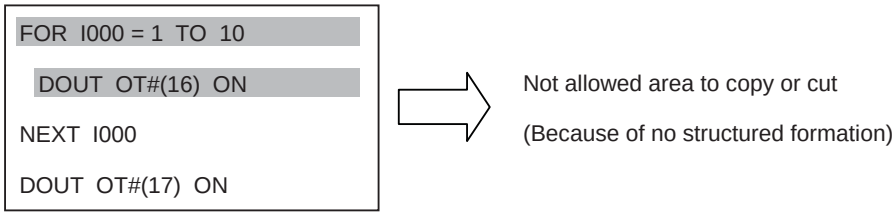


* **Copy or cut SWITCH ~ ENDSWITCH**

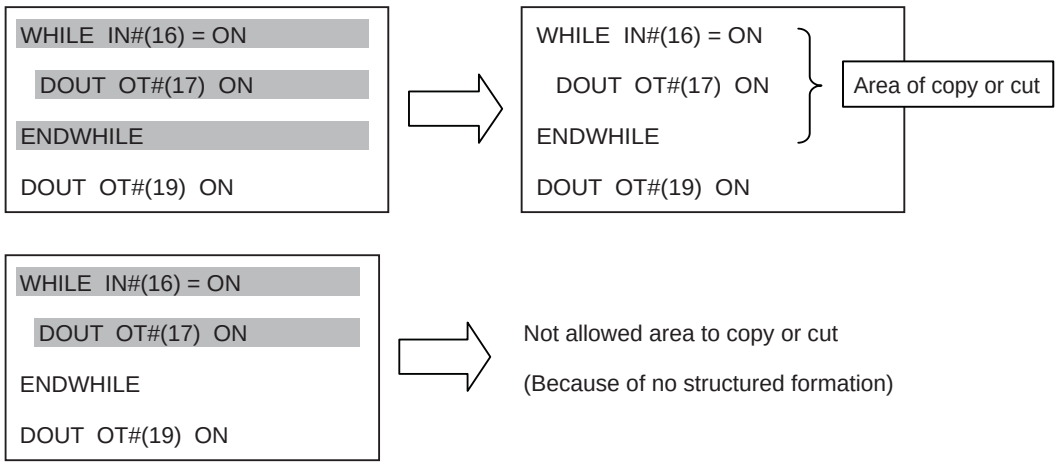


* **Copy or cut FOR ~ NEXT**





∗ **Copy or cut WHILE ~ ENDWHILE**



4 Format Conversion at External Output

When a job created in a structured language is output to an external memory device such as a CompactFlash card, the formats of some commands are converted to avoid name conflict between the existing instructions and the commands in the structured language.

1. When a job is output to an external storage, if an existing instruction has the same name as one of the structured language's commands used in the job, the command is converted into a format that does not cause conflict before output to the external storage.
   When the output job containing an structured language command with the converted format from an external storage to the controller, the converted command is reverted to the original format and loaded.

e.g.) In the case of a job that includes the sequential processing, “IF”

Because the structured language command “IF” conflicts with the existing instruction, “IF” is replaced with “IFTHEN” before output to the external storage. When the output job is loaded into the controller, “IFTHEN” is replaced with “IF”.

2. When a job is output to an external storage, if any existing instruction does have the same name as one of the structured language's commands used in the job, the displayed formats are output to the external storage as is.

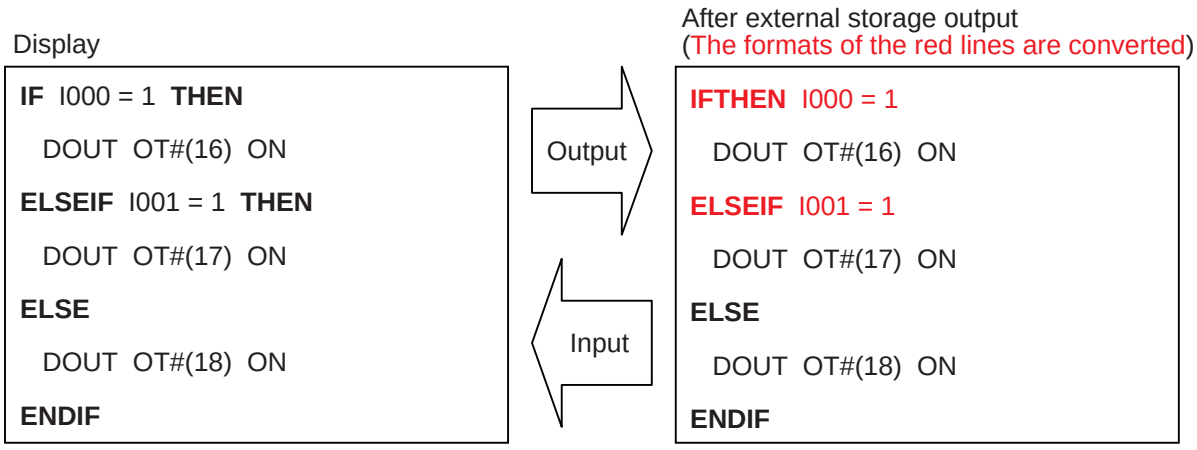
e.g.) In the case of a job that includes the repetitive processing, “FOR”

Because there is no existing instruction named “FOR” (that is, conflict does not occur, and therefore, conversion is not necessary), the job is output in the same format.

4.1 IF ~ ENDIF Statements

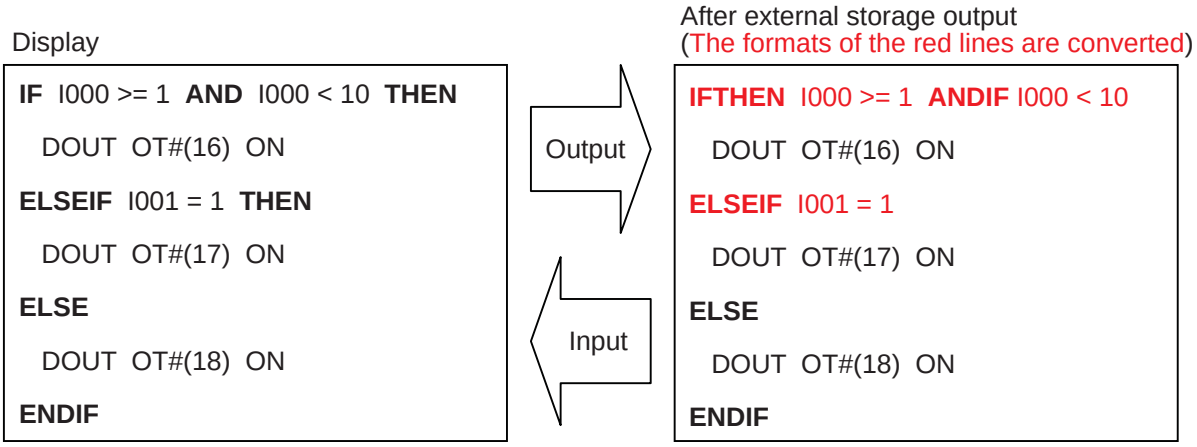
4.1.1 IF ~ ENDIF Statements (SINGLE Condition)

1. "IF ~ THEN" → Converted into "IFTHEN ~"
2. "ELSEIF ~ THEN" → Converted into "ELSEIF ~"
3. "ELSE" → No changes
4. "ENDIF" → No changes



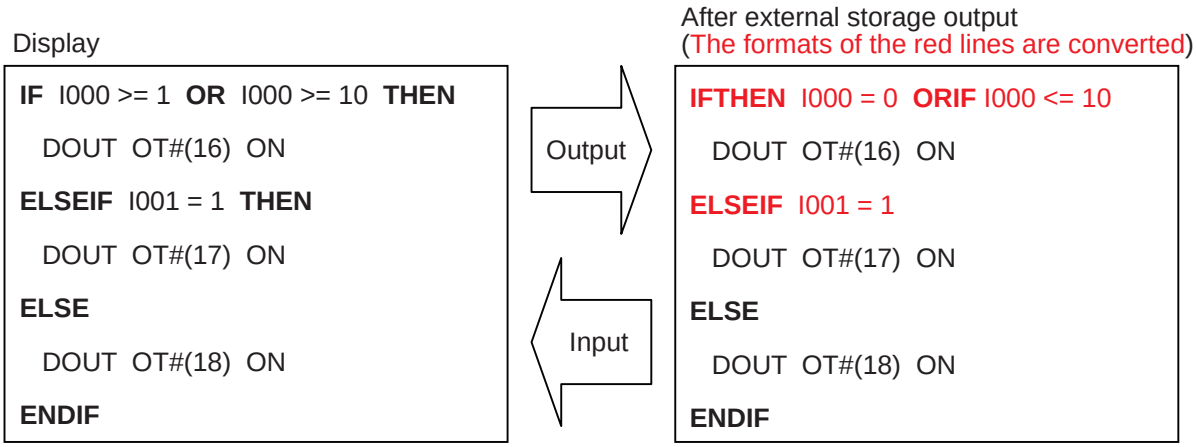
4.1.2 IF ~ ENDIF Statements (AND Condition)

1. "IF ~ AND ~ THEN" → Converted into "IFTHEN ~ ANDIF"
2. "ELSEIF ~ THEN" → Converted into "ELSEIF ~"
3. "ELSE" → No changes
4. "ENDIF" → No changes



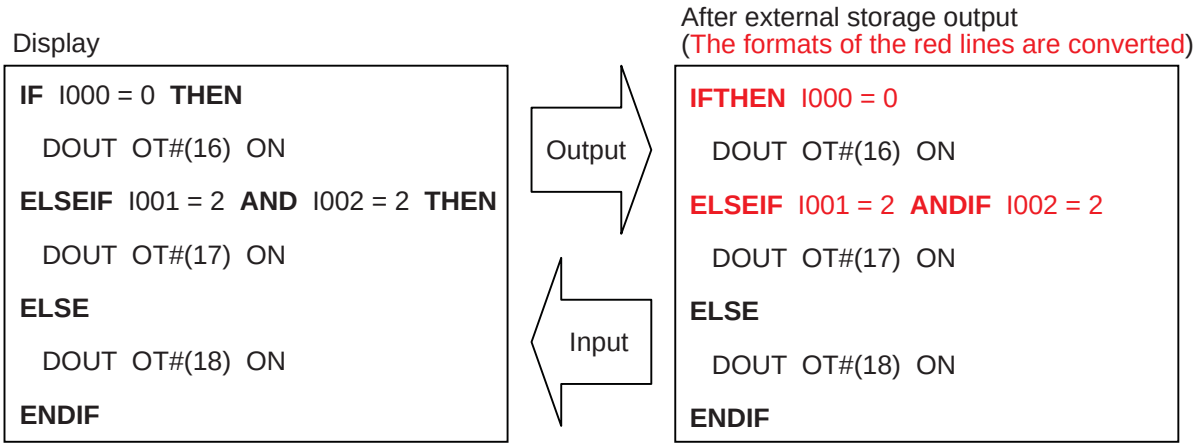
4.1.3 IF ~ ENDIF Statements (OR Condition)

1. "IF ~ OR ~ THEN" → Converted into "IFTHEN ~ ORIF"
2. "ELSEIF ~ THEN" → Converted into "ELSEIF ~"
3. "ELSE" → No changes
4. "ENDIF" → No changes



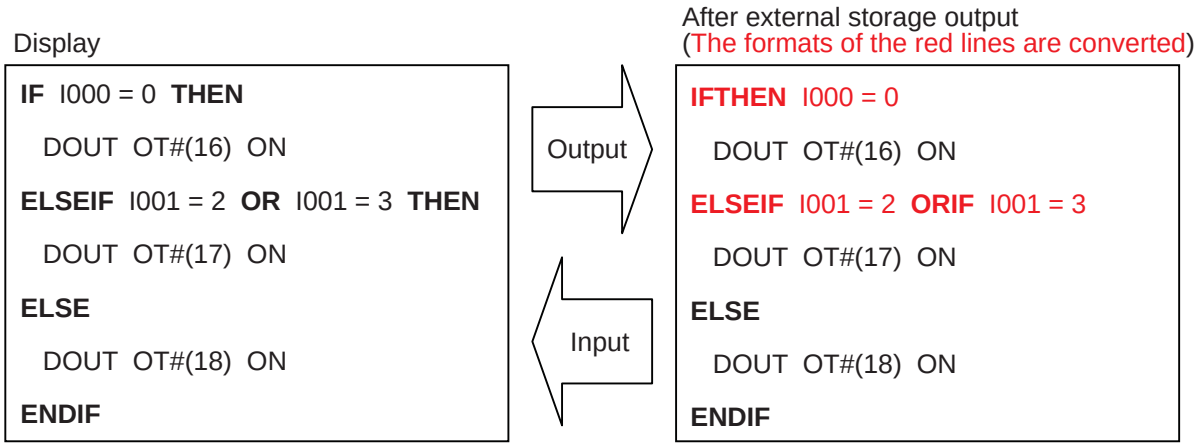
4.1.4 IF ~ ENDIF Statements (AND Condition of ELSEIF)

1. "IF ~ THEN" → Converted into "IFTHEN ~"
2. "ELSEIF ~ AND ~ THEN" → Converted into "ELSEIF ~ ANDIF ~"
3. "ELSE" → No changes
4. "ENDIF" → No changes



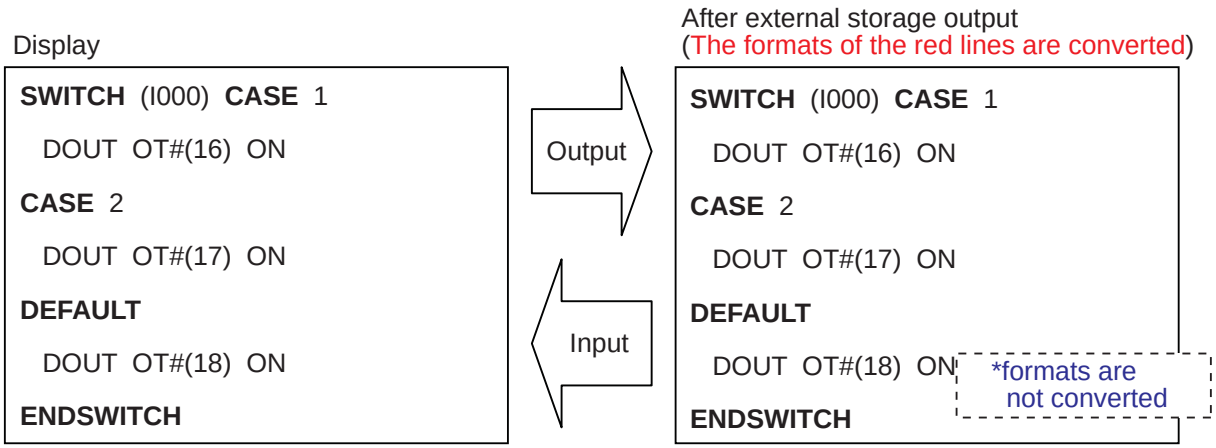
4.1.5 IF ~ ENDIF Statements (OR Condition of ELSEIF)

1. "IF ~ THEN" → Converted into "IFTHEN ~"
2. "ELSEIF ~ OR ~ THEN" → Converted into "ELSEIF ~ ORIF ~"
3. "ELSE" → No changes
4. "ENDIF" → No changes



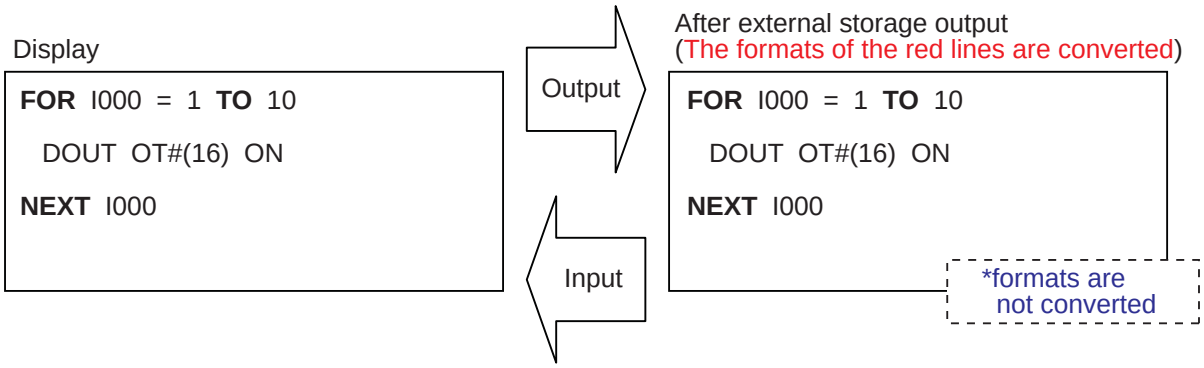
4.2 SWITCH ~ ENDSWITCH Statements

* Because the commands in the structured language used between the SWITCH and ENDSWITCH statements do not conflict with the existing instructions, the formats are not converted at external output.



4.3 FOR ~ NEXT Statements

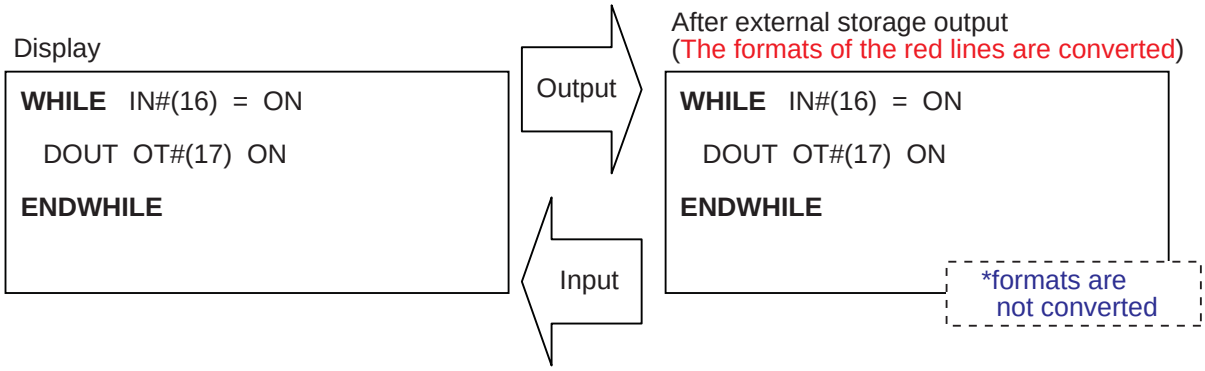
* Because the commands in the structured language used between the FOR and NEXT statements do not conflict with the existing instructions, the formats are not converted at external output.



4.4 WHILE ~ ENDWHILE Statements

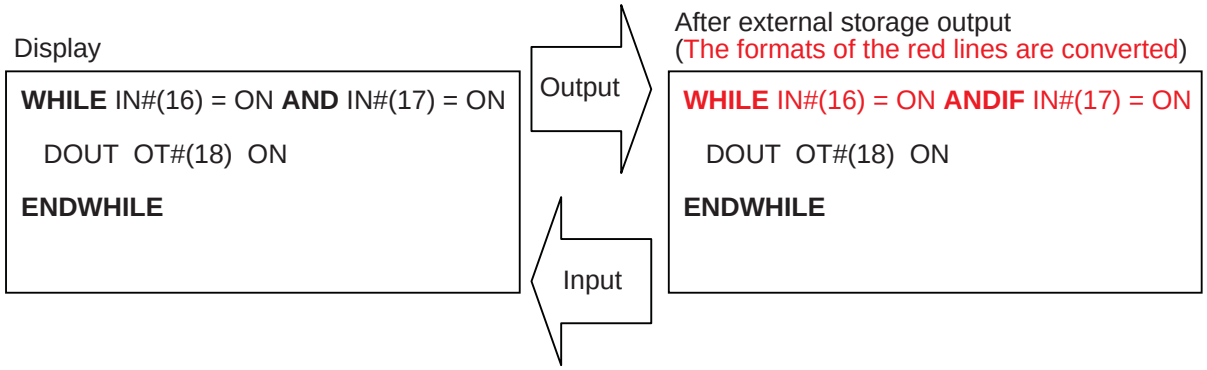
4.4.1 WHILE ~ ENDWHILE Statements (SINGLE Condition)

* Because the commands in the structured language used between the WHILE and ENDWHILE statements (single condition) do not conflict with the existing instructions, the formats are not converted at external output.



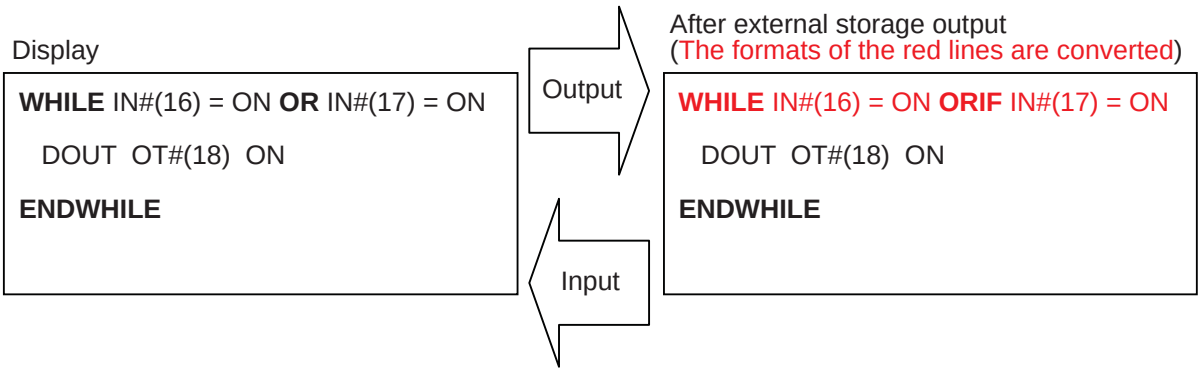
4.4.2 WHILE ~ ENDWHILE Statements (AND Condition)

1. "WHILE ~ AND" → Converted into "WHILE ~ ANDIF ~"
2. "ENDWHILE" → No changes



4.4.3 WHILE ~ ENDWHILE Statements (OR Condition)

1. "WHILE ~ OR ~" → Converted into "WHILE ~ ORIF ~"
2. "ENDWHILE" → No changes



5 Related Parameter

S2C693: Indent level for structured language nesting
Unit: [Number of characters]

FS100 OPTIONS
INSTRUCTIONS

INFORM EXTENSION FUNCTION
STRUCTURED PROGRAM LANGUAGE

HEAD OFFICE
2-1 Kurosakishiroishi, Yahatanishi-ku, Kitakyushu 806-0004 Japan
Phone +81-93-645-7745 Fax +81-93-645-7746

YASKAWA America Inc. MOTOMAN Robotics Division
805 Liberty Lane, West Carrollton, OH 45449, U.S.A.
Phone +1-937-847-6200 Fax +1-937-847-6277

YASKAWA Nordic AB
Verkstadsgatan 2, PO Box 504,SE-385 25 Torsås, Sweden
Phone +46-480-417-800 Fax +46-486-414-10

YASKAWA Europe GmbH Robotics Div.
Kammerfeldstr. 1, 80591 Allershausen, Germany
Phone +49-8166-90-0 Fax +49-8166-90-103

YASKAWA Electric Korea Co., Ltd
9F, KYOBO Securities Bldg., 26-4, Yeoido-Dong Yeoungeungpo-ku,Seoul, KOREA
Phone +82-2-784-7844 Fax +82-2-784-8495

YASKAWA Electric (Singapore) PTE Ltd.
151 Lorong Chuan, #04-02A, New Tech Park, Singapore 556741
Phone +65-6282-3003 Fax +65-6289-3003

YASKAWA Electric (Thailand) Co., Ltd.
252/246, 4th Floor. Muang Thai-Phatra Office Tower II Rachadaphisek Road, Huaykwang Bangkok, 10320 Thailand
Phone +66-2-693-2200 Fax +66-2-693-4200

Shougang MOTOMAN Robot Co. Ltd.
No.7, Yongchang-North Road, Beijing E&T Development Area, China 100176
Phone +86-10-6788-0548 Fax +86-10-6788-0548-813

YASKAWA ELECTRIC (SHANGHAI) Co., Ltd.
No.18Xizang Zhong Road, 17F, Harbour Ring Plaza, Shanghai 200001, CHINA
Phone +86-21-5385-0655 Fax +86-21-5385-2770

YASKAWA Robotics India Ltd.
#426, Udyog Vihar, Phase- IV, Gurgaon, Haryana, India
Phone +91-124-475-8500 Fax +91-124-414-8016

YASKAWA ELECTRIC CORPORATION

© Printed in Japan September 2011 11-09

MANUAL NO.
HW1480732