

**REVISIONS**

\* The manual number is given on the bottom left of the back cover.

Printe Date	* Manual Number	Revision
Sep. 1996	IB (NA) -66695-A	First printing

This manual does not imply guarantee or implementation right for industrial ownership or implementation of other rights. Mitsubishi Electric Corporation is not responsible for industrial ownership problems caused by use of the contents of this manual.

## About this manual

The manuals relating to this product are given in the table below. When necessary refer to the following table to order the manuals.

### Related manuals

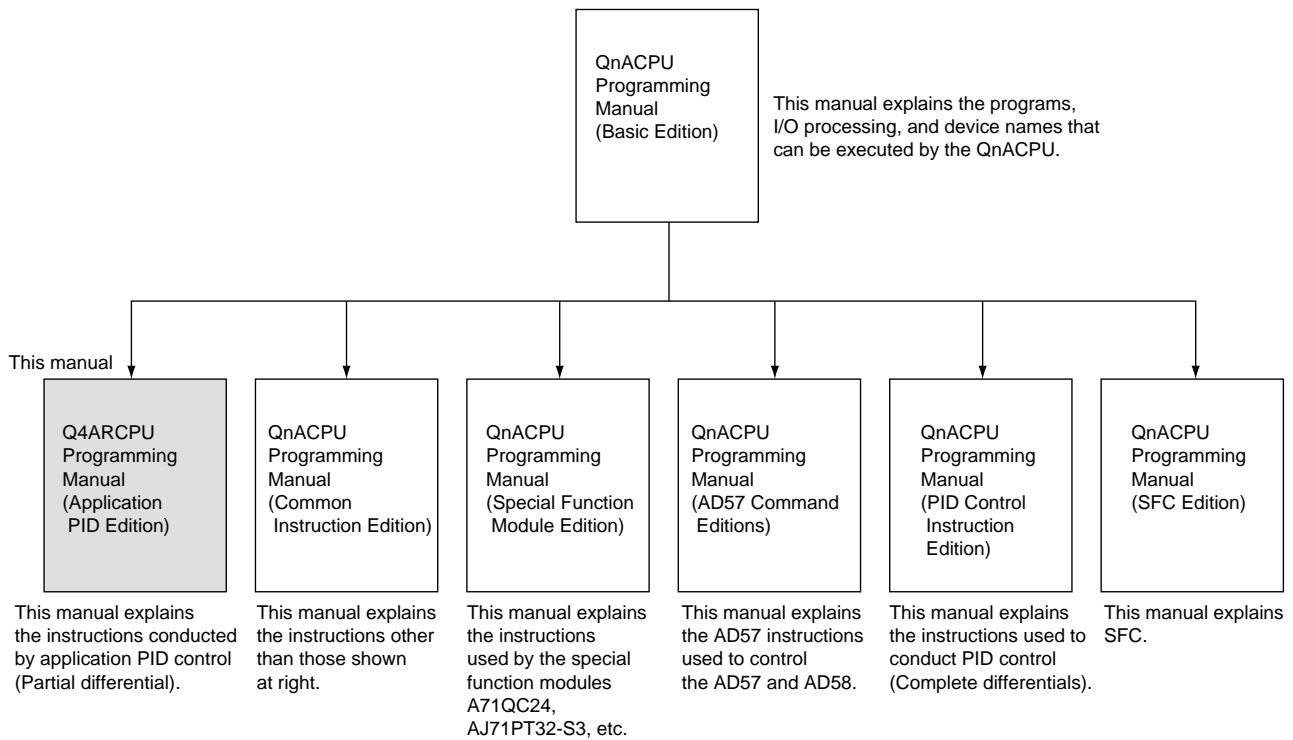
Manual name	Manual No. (Model name code)
Q4ARCPU User's Manual (Detailed Edition) This manual explains the items relating to the Q4ARCPU performance, functions, and handling and the bus switching module, system control module, power supply module, memory card, and base unit specifications and handling. (Sold separately)	IB-66685 (13J852)
MELSECNET/10 Network System Reference Manual for QnA/Q4AR This manual explains the network overview, specifications, and programming method, etc., that use the QnACPU and the Q4ARCPU. (Sold separately)	IB-66690 (13JF78)
Type SW0IVD-GPPQ GPP Software package OPERATING MANUAL (Offline) This manual explains the offline functions such as the SW0NX-GPPQ program creation method, print out method, and file maintenance. (Included in packaging)	IB-66623 (13JF12)
Type SW0IVD-GPPQ GPP Software package OPERATING MANUAL (Online) This manual explains the online functions such as SW0NX-GPPQ monitor method and debugging method. (Included in packaging)	IB-66624 (13JF13)
Type SW0NX-GPPQ GPP Software package OPERATING MANUAL (SFC) This manual explains the SFC functions such as the SFC program editing methods and monitoring methods. (included in packaging)	IB-66625 (13JF14)

## Related Programming Manuals

In addition to this manual there are related programming manuals that explain all of the Q4ARCPU instructions.

- QnACPU Programming Manual (Basic Edition)
- QnACPU Programming Manual (Common Instruction Edition)
- QnACPU Programming Manual (Special Function Module Edition)
- QnACPU Programming Manual (PID Control Instruction Edition)
- QnACPU Programming Manual (AD75 Command Edition)
- QnACPU Programming Manual (SFC Edition)

Before reading this manual please read the QnACPU Programming Manual (Basic Edition) and (Common Instruction Edition) to gain a basic knowledge of this product.



# Introduction

Thank you for purchasing the Mitsubishi General Purpose PC MELSEC-QnA series.

Before use read this manual carefully and correctly use the equipment after fully understanding the QnA series sequence functions and performance.

Please put this manual in a location accessible to the end user.

## Table of Contents

### **1 GENERAL DESCRIPTION**

1-1~1-2

#### **1.1 Features**

1-1

### **2 Data Configuration**

2-1~2-10

#### **2.1 Comprehensive Data Configuration**

2-1

#### **2.2 Local Work Memory**

2-3

#### **2.3 Process Control Instruction Configuration**

2-4

**2.3.1** Block Memory .....  
2-4

**2.3.2** Operation Constant.....  
2-4

**2.3.3** Loop Memory.....  
2-5

**2.3.4** Loop Tag Memory Allocation Contents .....  
2-6

**2.3.5** Loop RUN/STOP .....  
2-8

**2.3.6** Trucking Function .....  
2-9

**2.3.7** Cascade Loop Trucking.....  
2-9

### **3 Execution Time and Control Time**

3-1~3-2

#### **3.1 Time and Operation**

3-1

#### **3.2 Execution time and control time data storage location**

3-2

#### **3.3 Example Program in the Timing Relationship**

3-2

## **4** PID Control System

### **4-1~4-6**

#### **4.1** PID Control Overview

4-1

#### **4.2** PID Control

4-2

<b>4.2.1</b>	Operation Method .....	4-2
<b>4.2.2</b>	Direct Action and Reverse Action .....	4-2
<b>4.2.3</b>	Proportional action (P action) .....	4-3
<b>4.2.4</b>	Integral action (I action) .....	4-4
<b>4.2.5</b>	Derivative Operation (D Operation) .....	4-5
<b>4.2.6</b>	PID Operation .....	4-6

## **5** Instruction List

### **5-1~5-3**

#### **5.1** How to Read the Instruction List Table

5-1

#### **5.2** I/O Control Instructions

5-2

#### **5.3** Control Operation Instructions

5-2

#### **5.4** Correction Operation Instructions

5-2

#### **5.5** Arithmetic Operation Instruction

5-3

## **6** Instruction Configuration 6-1~6-3

### **6.1** Instruction configuration 6-1

### **6.2** Method for specifying the data in a device 6-1

6-1	<b>6.2.1</b>	For Bit Data.....	
6-2	<b>6.2.2</b>	For Word (16-bit) Data.....	
6-2	<b>6.2.3</b>	For Real Number Data (Floating Point Data).....	
6-3	<b>6.2.4</b>	Process Control Function Operation Error.....	
6-3	<b>6.2.5</b>	Instruction Execution Conditions .....	
6-3	<b>6.2.6</b>	Number of Steps .....	
6-3	<b>6.2.7</b>	Index Qualification .....	

## **7** How to Read Instructions 7-1~7-3

## **8** Process Control Instruction 8-1~8-40

### **8.1** I/O Instructions 8-1

8-1	<b>8.1.1</b>	Analog Input Processing.....	S. IN
8-5	<b>8.1.2</b>	Output Processing-1 with Mode Switching .....	S. OUT1

### **8.2** Control Operation Instruction 8-9

8-9	<b>8.2.1</b>	Basic PID .....	S. PID
8-15	<b>8.2.2</b>	Process High Alarm Process Low Alarm .....	S. PHPL
8-20	<b>8.2.3</b>	Lead Lag.....	S. LLAG
8-22	<b>8.2.4</b>	Integral.....	S. I
8-24	<b>8.2.5</b>	Derivative.....	S. D

8.2.6 Dead Time .....S. DED  
8-26

**8.3** Correction Operation Instruction  
8-29

8.3.1 Function Generator .....S. FG  
8-29

8.3.2 Inverse Function Generator .....S. IFG  
8-31

8.3.3 Standard Filter .....S. FLT  
8-33

8.3.4 Engineering Value Conversion .....S. ENG  
8-35

8.3.5 Inverse Engineering Value Conversion .....S. IENG  
8-37

**8.4** Arithmetic Operation  
8-39

8.4.1 Absolute Value .....S. ABS  
8-39

**9** Error Code  
9-1~9-2

**9.1** How to Read Error Codes  
9-1

**9.2** Error Code List  
9-2

**Appendix**  
**Appendix-1~Appedix-6**

Appendix 1 Example Program

Appendix 2 Operation Processing Time

# GENERAL DESCRIPTION

This manual explains the process control instructions used to conduct application PID control in the Q4ARCPU.

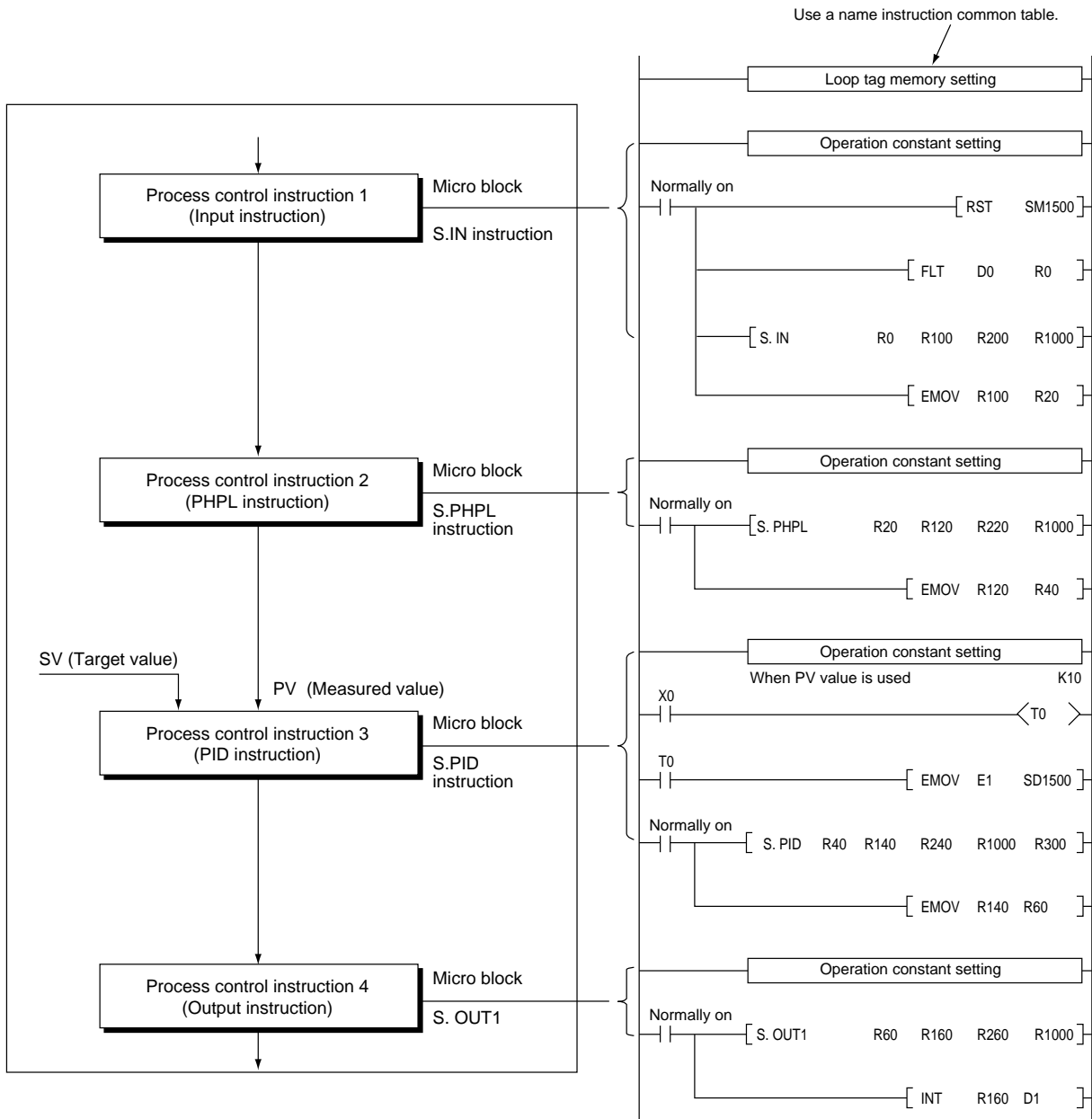
## 1.1 Features

The process control instructions have the following features.

**Because the CPU contains a floating point operation processor, floating point real number operations for PID processing can be processed at high speed.**

**Because floating point real number data is handled, a wide range of operations can be executed at high accuracy.**

**PID control is used to make micro block units independent and to improve maintainability.**





**A wider application is possible to make build up as an option possible.**

Precise control makes it possible to easily build up input signal conversion, correction operation, and retentive and alarm checks and output signal processing instructions to conduct display and adjustment control operations for digital adjustment modules, etc.

The following is one usage example.

**S.LLAG (Lead lag):**

This is used as an operation characteristic compensation for conducting correction operations required before disturbance information can have an effect on the control system.

**S.DED (Dead time):**

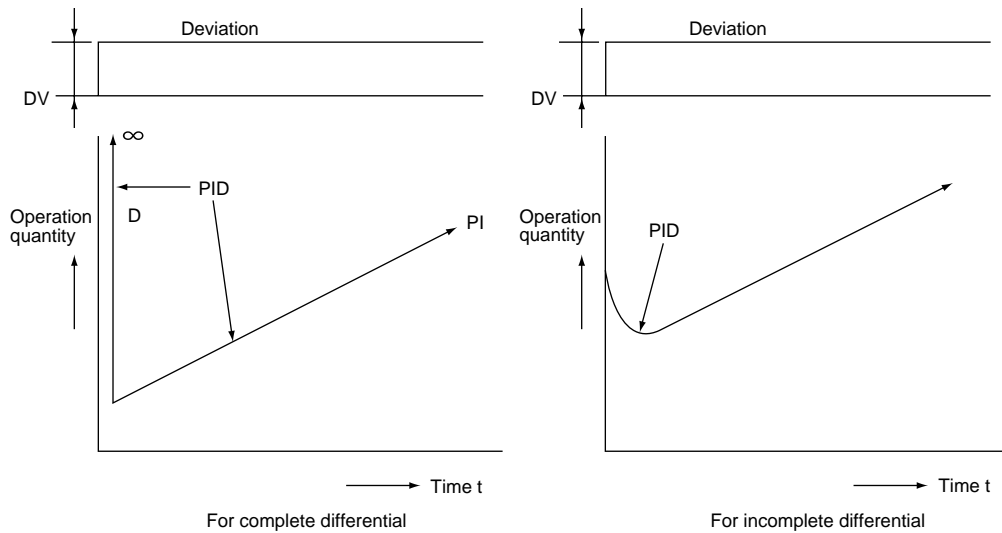
A variety of different dead times exist. There are dead times, such as the delayed transport of physical equipment and the delay caused by detectors, such as analyzers, and this kind of dead time is generally difficult to control. In this case, the contents of the operation control signal can be used to delay the output of the input data by only the dead time.

**A save system can be developed because various warning information can be automatically detected by the system.**

**PID algorithm using a velocity type incomplete differential format**

Partial differential has the following advantages over the complete differential format.

- 1 The differential gain is 1/ and the limit value can be set.
- 2 The output contains time amplitude, so the system actually responds to the operation edge so the differential operation makes the movement valid.



# Data Configuration

## 2.1 Comprehensive Data Configuration

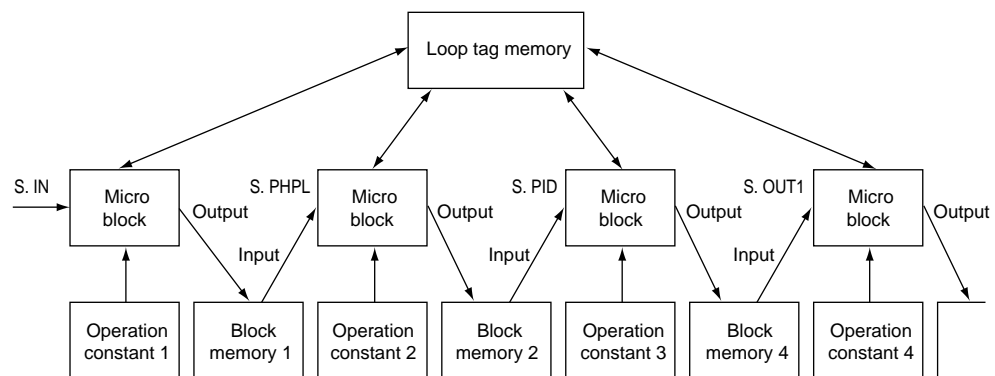
A variety of control equipment, such as detectors, regulators, and operating equipment, must be used to conduct PID control.

For this reason, the comprehensive data configuration (data flow) that conducts the operation equipment groups using the process control instructions is explained below.

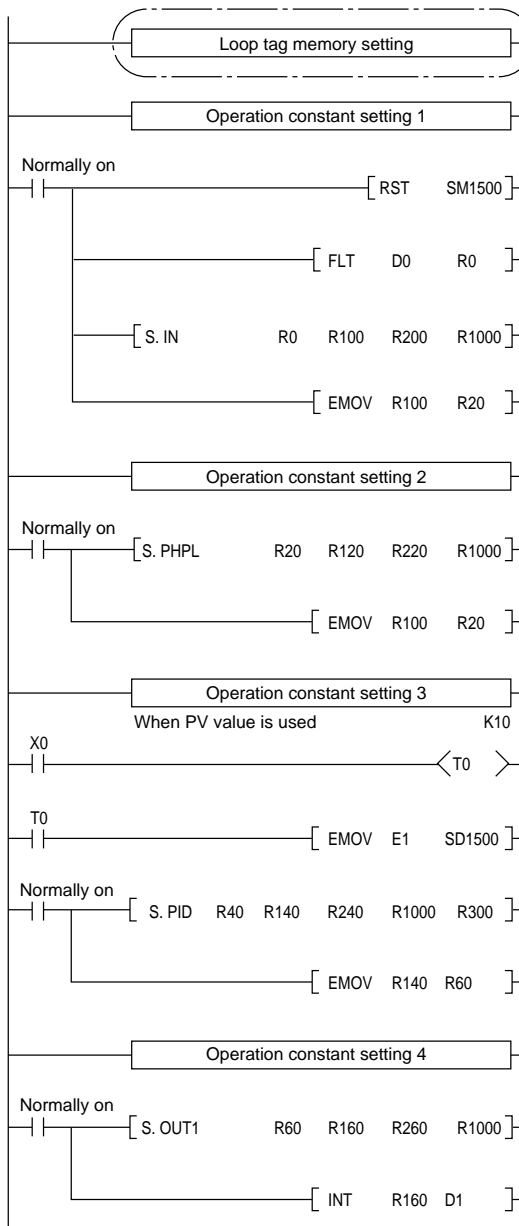
Configuration when using loop tag

- 1 The loop units have common storage areas that show the control information. This collection of common information is called a loop tag and the storage memory is called the loop tag memory.
- 2 By monitoring this the loop (Control unit) monitoring can be tuned.

Block diagram



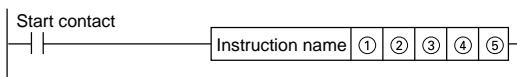
Ladder diagram



Loop tag memory (96W)

Instruction used	Item returned	Standard value setting	Data type
+0			Word
+1	MODE	8H	Word
+3	ALM	4000H	Word
+4	INH	4000H	Word
+10	S. PHPL	PV	Floating point
+12	S. OUT1	MV	Floating point
+14	S. PID	SV	Floating point
+16	S. PID	DV	Floating point
+18	S. OUT1	MH	Floating point
+20	S. OUT1	ML	Floating point
+22	S. PHPL	RH	Floating point
+46	S. PID	CT	Floating point
+48	S. OUT1	DML	Floating point
+50	S. PID	DVL	Floating point
+52	S. PID	P	Floating point
+54	S. PID	I	Floating point
+56	S. PID	D	Floating point
+58	S. PID	GW	Floating point
+60	S. PID	GG	Floating point
+62	S. OUT1	MVPL	Floating point
to		to	to
+93		0	Floating point
+94		0	Floating point
+95		0	Floating point

The meanings of the codes in the ladder diagram are given below.



Instruction name	S. IN	S. PHPL	S. PID	S. OUT1
1 Input block header device	R0	R20	R40	R60
2 Block memory header device	R100	R120	R140	R160
3 Operation constant header device	R200	R220	R240	R260
4 Loop tag memory header device	R1000			
5 Target value header device	—	—	R300	—

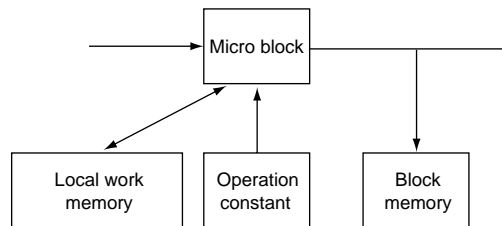
## 2.2 Local Work Memory

Memory (Local memory) is sometimes required to store the instruction intermediate operation results of some process control instructions.

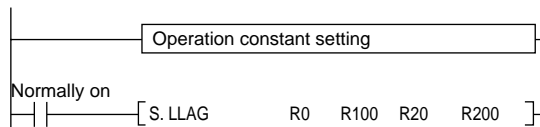
The instructions that can use local work memory are shown below.

Instruction name	Remarks
S. LLAG (Lead lag) S. D (Differential) S. DED (Dead time) S. FLT (Standard filter)	This stores the interim operation results for the OS itself. (Cannot be used by the user.)
S. FG (Function Generator) S. IFG (Inverse Function Generator)	This stores the broken line coordinate value (Xn, Yn) used by the user. Operations are conducted based on this.

Block diagram



Ladder diagram



Instruction name	S.LLAG (Lead lag)
Input block header address	R0
Block memory header address	R100
Operation constant header address	R20
Local work memory header address	R200

## 2.3 Process Control Instruction Configuration

Process control instruction performs processing by receiving and processing data from the operation constant based on the PV (Process value) and SV (Set value) as well as processing of the data exchange with the loop memory. In addition, it outputs alarm signals based on the processing contents.

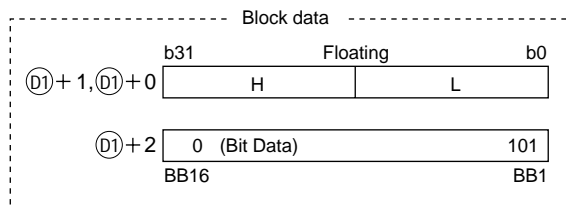
### 2.3.1 Block Memory

Processing results are stored in block memory as output values and these output values are connected to loops to create the following process control instruction input values.

This is the output result storage area for each instruction. This makes it possible to monitor the interim results and the auxiliary output results.

**The data is configured from word data and bit data.**

**The storage area is specified by each instruction.**



The data transferred to the next instruction is used as this data.

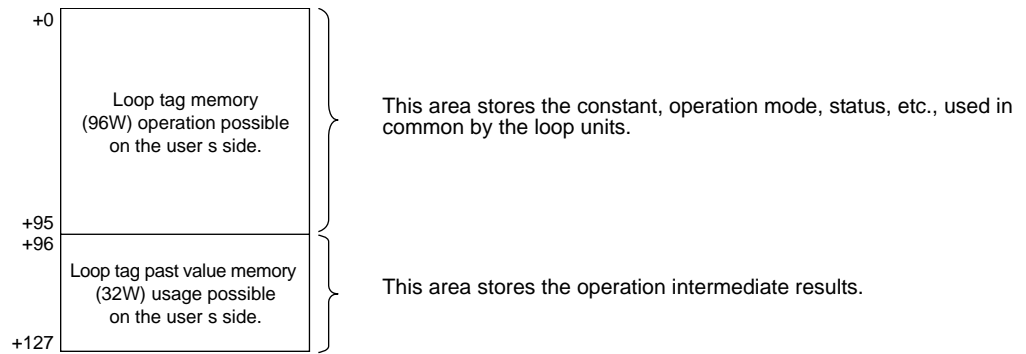
Stores the bit output for each instruction. Mainly used as auxiliary information for alarms, etc.

### 2.3.2 Operation Constant

The operation constant used for the unit is stored for each instruction. The storage area and data contents differ depending on the instruction. (Refer to Section 8)

### 2.3.3 Loop Memory

Data is configured from loop tag memory and loop tag past value memory. (1 loop 128W)



**When operation starts write the standard value setting in the file register using the sequence program.**

- Loop tag memory : Enter the standard value settings written in the explanation portion for each instruction in Section 8 Process Control Instructions.
- Loop tag past value memory : Write "0" to clear the past values only when operation has started.

#### Key Points

When the user is using the loop tag past value memory, the system will not operate normally, so take due precautions.

2.3.4 Loop Tag Memory Allocation Contents

The loop tag memory allocation contents are shown below.

Instructions used in loop tag      Show the number of words from the loop tag header      Abbreviated name of each item      After setting some values are changed by the numbers from the operation results. (Highlighted areas)

Instruction used	Offset	Item	Setting range	Standard value setting	Data type
	+0				
	+1	MODE	0 to FFFFH	8H	Word
	+3	ALM	0 to FFFFH	4000H	Word
	+4	INH	0 to FFFFH	4000H	Word
S. PHPL	+10	PV	RL* to RH* *1	0	Floating point
S. OUT1	+12	MV	-10 to 110	0	Floating point
S. PID	+14	SV	RL* to RH* *1	0	Floating point
S. PID	+16	DV	-110 to 110	0	Floating point
S. OUT1	+18	MH	-10 to 110	100	Floating point
S. OUT1	+20	ML	-10 to 110	0	Floating point
S. PHPL	+22	RH	-999999 to 999999	100	Floating point
S. PHPL	+24	RL	-999999 to 999999	0	Floating point
S. PHPL	+26	PH	RL* to RH* *1	100	Floating point
S. PHPL	+28	PL	RL* to RH* *1	0	Floating point
S. PHPL	+30	HH	RL* to RH* *1	100	Floating point
S. PHPL	+32	LL	RL* to RH* *1	0	Floating point
S. IN	+38	α	0 to 1	0.2	Floating point
S. PHPL	+40	HS	0 to 999999	0	Floating point
S. PHPL	+42	CTIM	0 to 999999	0	Floating point
S. PHPL	+44	DPL	0 to 100	100	Floating point
S. PID	+46	CT	0 to 999999	1	Floating point
S. OUT1	+48	DML	0 to 100	100	Floating point
S. PID	+50	DVL	0 to 100	100	Floating point
S. PID	+52	P	0 to 999999	1	Floating point
S. PID	+54	I	0 to 999999	10	Floating point
S. PID	+56	D	0 to 999999	0	Floating point
S. PID	+58	GW	0 to 100	0	Floating point
S. PID	+60	GG	0 to 999999	1	Floating point
S. OUT1	+62	MVP	-999999 to 999999 *2	0	Floating point

For PID control (SPID loop)  
All commonly set in the same loop tag

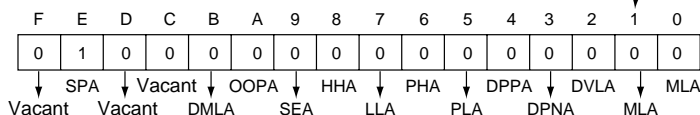
Sets the offset position for each instruction

Remarks

- 1 \*1 means the following value.  
 $PL* = RL - 0.05 (RH - RL)$   
 $RH* = RH + 0.05 (RH - RL)$
- 2 \*2 means the maximum value and minimum value that can be found using the floating point data.
- 3 The CPU does not conduct data range checks for other than the floating point data maximum and minimum.

Shows the contents of the bit pack using the loop tag data.

ALM



The standard value setting 4000H is shown when manual operation is conducted using the loop step status. Use 0000H for automatic alarms.

Table 2.1 ALM details list

U: Established by user selection.

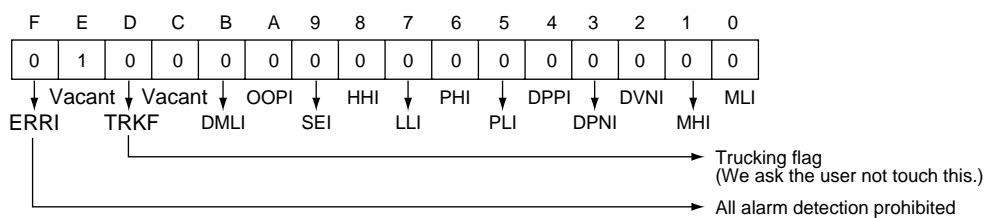
0: Established automatically by internal

conditions.

Name	Abbreviation	Description	Flag establishment conditions
Step alarm	SPA	Shows the loop stop status. Changes the loop mode to manual. Conducts stop alarm processing for the output value (BW) and alarm signal.	U
Output change rate limit alarm	DMLA	Conducts the change rate limiter for the input data and outputs the change rate alarm. (For the output change upper limit value/control value)	0
Output open alarm	OOPA	Shows that it has changed to open status when the operation output signal has become disconnected, etc.	0
Sensor alarm	SEA	Sensor error alarm	0
Upper upper limit alarm	HHA	Checks the upper limit value of the process equipment upper limit and if the measurement value is higher than the upper limit value outputs an alarm.	0
Lower lower limit alarm	LLA	Checks the lower limit value of the process equipment limit and if the lower limit value is lower than the measured value limit then outputs an alarm.	0
Upper limit alarm	PHA	Checks the measurement value upper limit value and if higher than the upper limit outputs an alarm.	0
Lower limit alarm	PLA	Checks the measurement value lower limit value and if lower than the lower limit value outputs an alarm.	0
Positive direction change rate alarm	DPPA	Outputs an alarm if the change rate is higher than the increasing trend change rate range.	0
Negative direction change rate alarm	DPNA	Outputs an alarm if the change rate is lower than the lower trend change rate range.	0
Deviation size alarm	DVLA	Conducts an error check and then outputs an alarm if over. In addition, if the error check determines that the deviation is completely less than the alarm value and the error is reduced by a set value from the warning value then the error size alarm will be released.	0
Output upper limit alarm	MHA	Conducts a check using the upper and lower limit limiter and if the limiter results are larger than the input upper limit value an alarm is output.	0
Output lower limit alarm	MLA	A check is conducted by an upper and lower limit limiter and if the limiter results are smaller than the input lower limit value an alarm is output.	0

INH

This prohibits alarm detection for each item. In addition, the alarms prohibited by INH are not



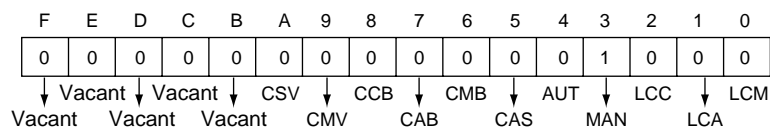
detected. (Bit 0 to B is compared to each ALM bit.)



MODE

The loop tag control mode adds modes that satisfy the following operations to systems connected to the operator station, PC, upper level computer, equipment operation panels, etc. (Only the chain selector is installed.)

- 1 Monitoring and control is possible from the operator station. (MAN, AUT, CAS)
- 2 Operation and monitoring in the control mode from the operator station is possible using loop control from an upper level computer. (CSV, CMV)
- 3 When a computer problem occurs during loop control from an upper level computer, the system switches to backup from a previously specified operator station, so this status changes. (CMB, CAB, CCB)
- 4 During plant startup operation startup from a loop display, etc., from other than the operator station is required, so monitoring in the control mode from the operator station can be done. (LCM, LCA, LCC)



For MODE make one of them a 1 bit only flag 1.

**2.3.5 Loop RUN/STOP**

When there is a problem with loop configuration equipment, such as detectors as operation terminals other than the PC, only the corresponding loop stops and RUN/STOP can be set for each loop to conduct hardware maintenance.

**Basic operation during loop STOP**

- 1 Output state hold (The S.PID instruction is output=0)
- 2 Alarm undetected (Process alarm)
- 3 Make the control mode MAN.

### 2.3.6 Trucking Function

The trucking function is:

Prevents the manipulated value (MV) output step change and switches the MV output bumplessly and smoothly when there is a large change in the manipulated value when switching between the automatic mode and the manual mode.

Restricts changes in the MV in the micro function output processing area after there has been a switch from the manual mode to the automatic mode making it possible to make the MV output switch smoothly.

The trucking function is a function that contains both the above (a) bumpless function and (b) output limiter processing operation.

#### Bumpless function

The bumpless function prevents manipulated value (MV) output stepping changes when switching from the automatic mode to the manual mode and continuously and smoothly controls MV output.

#### Output limiter processing function

The output limiter processing function limits the upper limit and lower limit of the manipulated value (MV) output by the PID operation during the automatic mode. This output limiter processing function is only valid in the automatic mode and is not executed for manual data. In addition, when the parameter trucking function execution validity is set to not valid when in the automatic mode the output limiter processing function will not execute.

#### Micro block for which the trucking operation is possible

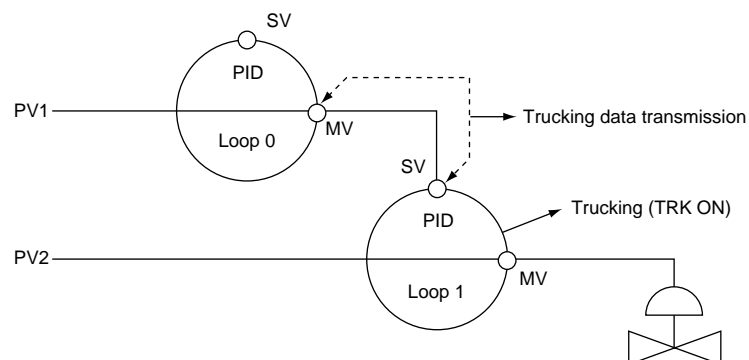
S.OUT1

### 2.3.7 Cascade Loop Trucking

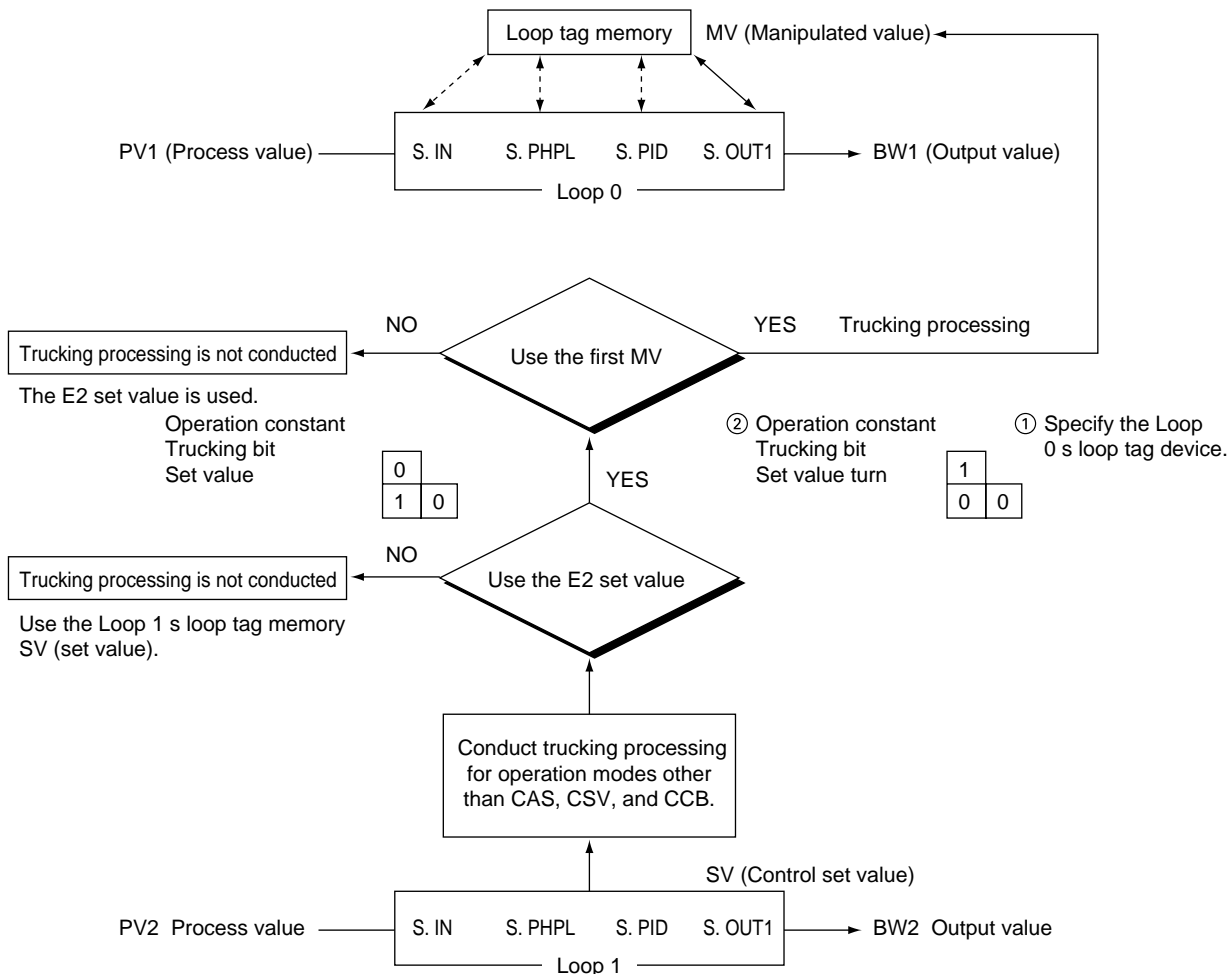
The process control loops that comprise the cascade loop, when the secondary loop (Loop 1) control mode is switched the primary loop (Loop 0) manipulated value (MV) is used as the secondary loop (Loop 1) control set value (SV) to prevent a rapid change in the control set value (SV).

The cascade PID loop trucking process is shown in the diagram below.

Processing concept diagram



When the control mode is switched to another mode other than CAS, CSV, or CCB, trucking is processed under the following conditions.



PID trucking

- 1 During cascade operation : Loop 0's MV value—Loop 1's SV value
- 2 When not conducting cascade operation : Loop 1's SV value—Loop 0's MV value  
(Trucking to the Loop 1's SV input terminal specified source.)

Micro block for which trucking operation is possible  
S.PID

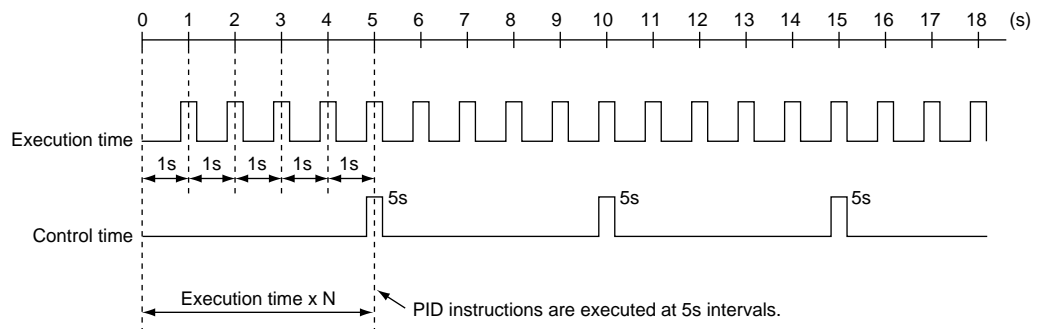
## Execution Time and Control Time

The process control instruction global subroutine program is started up during the set time by sink link processing.

The control time (CT) is the PID control operation execution time. Therefore, it is set differently than that above.

### 3.1 Time and Operation

Example: Basic PID control is executed for every monitor 1 second and control every 5 seconds.



**The startup request uses the timer to take measurements and executes the process control instruction when the timer is time up. Set the timer setting time and SD1500, SD1501 (special registers) execution time to the same time.**

**The control time is the execution time of the S.PID instruction, etc. set in each process control instruction loop, so set the execution time multiple. When the control time (CT) is not a multiple of the execution time (T), calculate  $CT/T$  and round down the decimal.**

Example:

For  $CT=2.5$  sec  $T=1.0$  sec setting

Operate at  $2.5/1.0=3.0$ .

**The execution time setting can be set in multiples of 10 msec.**

### 3.2 Execution time and control time data storage location

Example: For S.PID

Loop tag memory		Special register
①+ 1	Operation mode	MODE
+ 3	Alarm detection	ALM
+ 4	Alarm detection prohibited	INH
+ 14	Set value	SV
+ 16	Deviation	DV
.....		
+ 46	Control time	CT

Floating

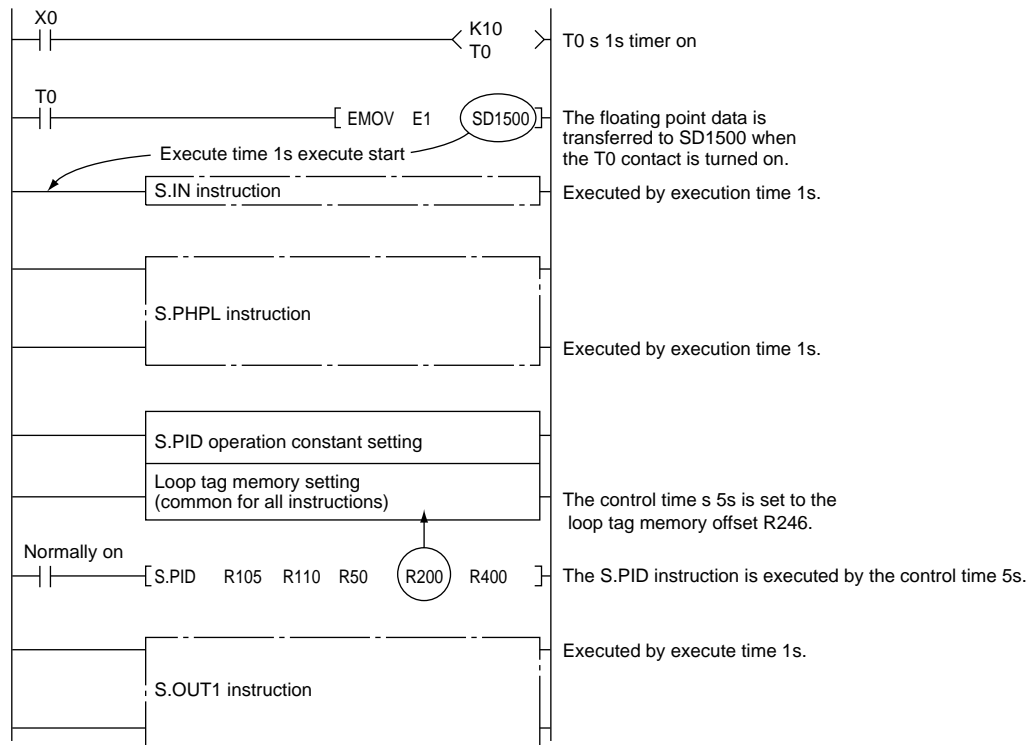
SD1500	Execution time (sampling time)( $\Delta T$ )	Floating
SD1501	(SEC)	

**Point**

When execution time is used by the process control instruction, the data from SD1500 and SD1501 is used. Therefore, when executing the S.PID instruction, be sure to set the execution time in SD1500 and SD1501 in advance.

### 3.3 Example Program in the Timing Relationship

Example 1: When using execute time 1s and control time 5s in the SPID loop



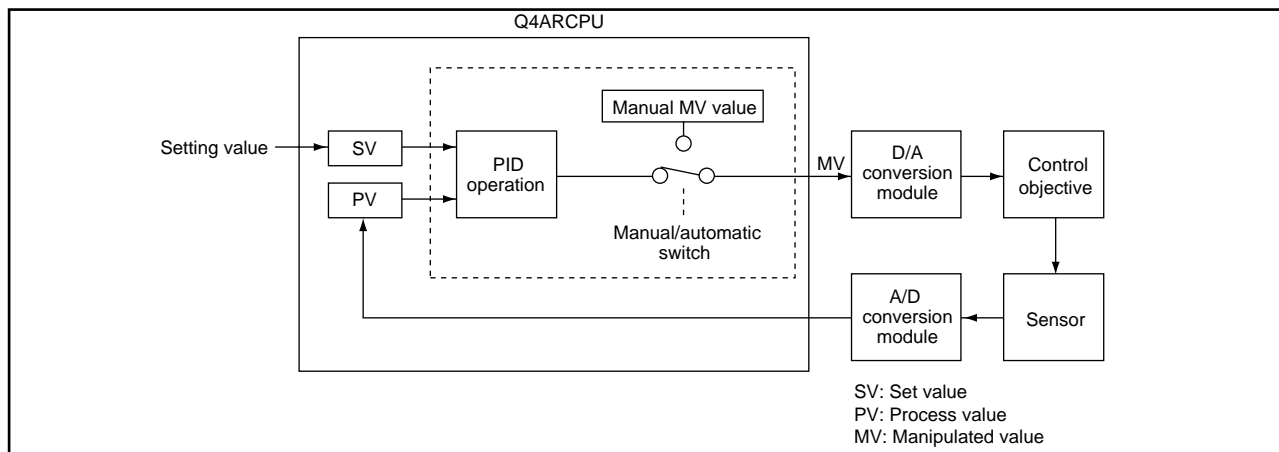
# PID Control System

This section explains the PID control method using the process control instruction.

## 4.1 PID Control Overview

PID control can be used to control processes such as flow volume, speed, air amount, temperature, tension, and combining, and is configured as shown in Figure 4.1 to maintain the values set for the objective control.

Figure 4.1 Example application for process control



PID control compares the measured value (process value) from the detectors and the previously set value (set value) and adjusts the output value (manipulated value) to remove any difference between the process value and the set value.

PID control operations combine proportional operations (P), integral operations (I), and derivative operations (D) to calculate the manipulated value to quickly and accurately make the process value the same as the setting value. In other words, if the difference between the process value and the setting value is large, the manipulated value will be made large to quickly approach the setting value, and if the difference between the operation value and the set value is small, the manipulated value is made smaller to slowly and accurately adjust the manipulated value to make it the same as the set value.

## 4.2 PID Control

The operation method used for a process control instruction's PID control are the velocity type and the differential type. The velocity type and the differential type conduct the following control.

### 4.2.1 Operation Method

#### Velocity type operation

While the position type finds each manipulated value (MV), the velocity type calculates the portion (  $\Delta$  MV) to add to the previous MV value.

The two different operations have their various characteristics, but the velocity type makes it easy to use integrals and conduct switching processing, so currently it is the most possible.

#### Process value differential type

PID control conducts operations based on the difference between the set value (SV) and the process value (PV) and this is called deviation differential. However, in this case if the set value (SV) changes rapidly, the differential operation will become large causing the manipulated value to change rapidly. Therefore, if only the control variable is used, the effect on the change in set value is much less so this is called the process value differential type.

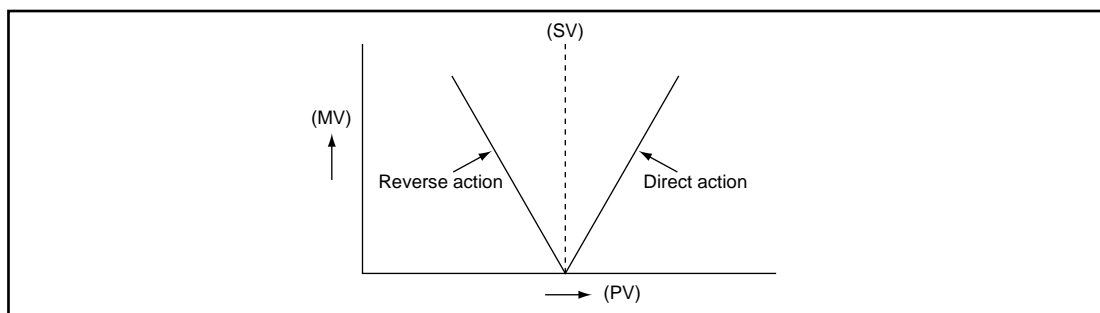
### 4.2.2 Direct Action and Reverse Action

**Direct action is the action that increases the manipulated value when the process value increases more than the set value.**

**Reverse action is the action that increases the manipulated value when the process value is decreasing more than the set value.**

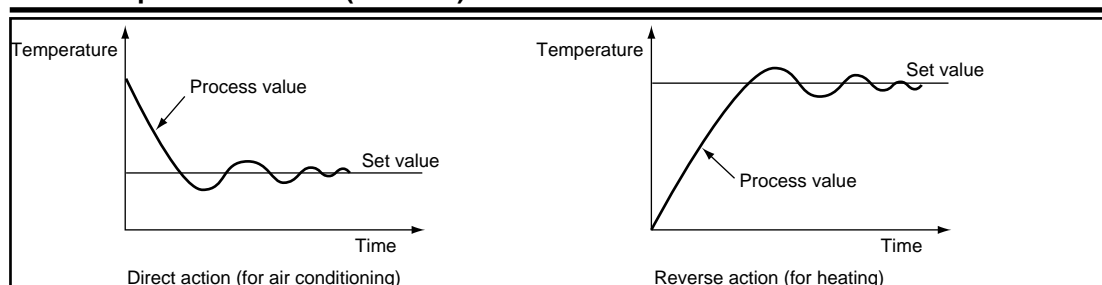
**Direct action reverse action make the manipulated value larger as the difference between the set value and the process value becomes larger.**

The direct action and reverse action use the manipulated value (MV), process value (PV), and set value (SV) as shown in the following diagram.



An example of process control using direct action and reverse action is shown in the following diagram.

### 4.2.3 Proportional action (P action)



This section explains the control method using proportional action.

**Proportional action is the action that compares the deviation (difference between the set value and the process value) to find the manipulated value.**

**The change in relationship between deviation (DV) and manipulated value (MV) using proportional action is shown using the following mathematical formula.**

$$\boxed{MV = K_P DV}$$

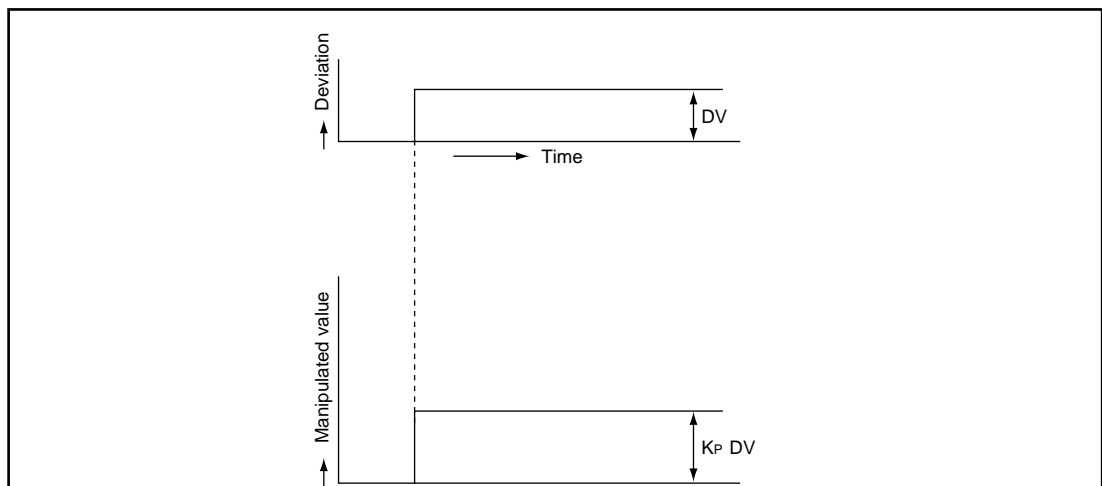
$K_P$  is called the proportional gain or proportional constant.

**The proportional action when the deviation is a constant step response is shown in Figure 4.2.**

**Figure 4.2 Proportional action when deviation is constant**

**The manipulated value fluctuates between -10 and 110.**

As  $K_P$  becomes larger the manipulated value corresponding to the deviation also becomes larger

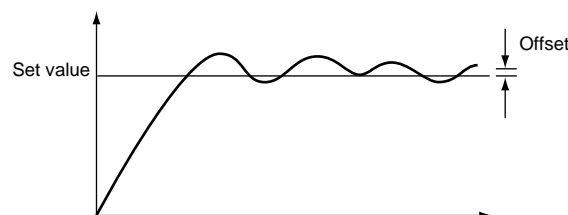


making the correction action stronger.

**Offset occurs in proportional action.**

#### **Remarks**

Offset generates a constant error for the set value even when the control objective keeps the system in a stable state. This error is called offset.





#### 4.2.4 Integral action (I action)

This section explains the control method using integral action.

**Integral action is the action that continuously changes the manipulated value to eliminate deviation when there is deviation.**

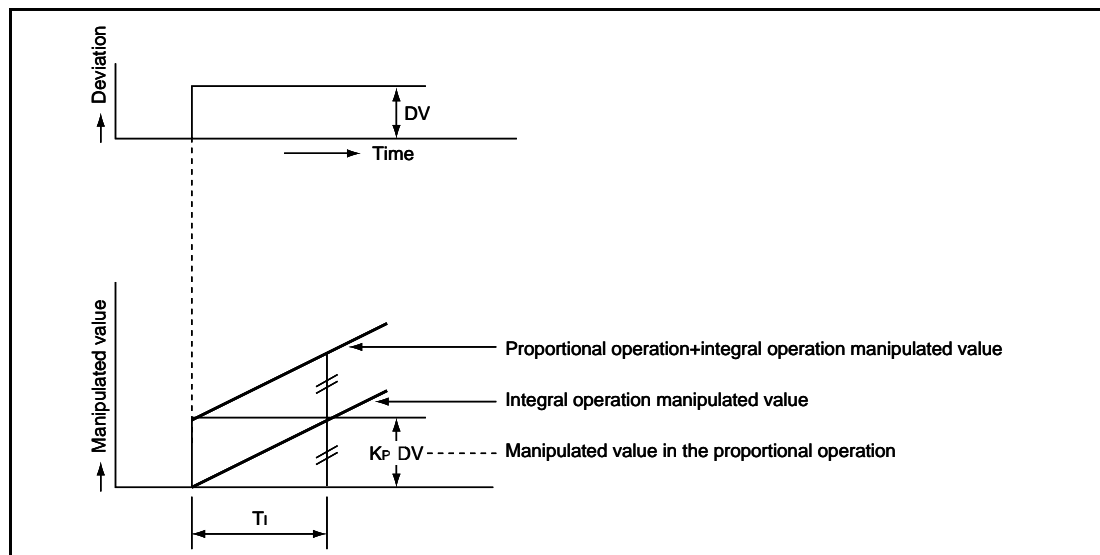
This can eliminate the offset that occurs from the proportional operation.

**In the integral operation, the time from when a deviation occurs to when the integral operation manipulated value becomes the proportional operation manipulated value is called the integral time and is shown as  $T_I$**

The smaller  $T_I$  becomes the stronger the integral operation.

The integral operation when the deviation is a constant value step response is shown in Figure 4.3.

Figure 4.3 Integral operation when the deviation is a constant



**The integral operation is used as the PI operation that is combined with the proportional operation or as the PID operation that is combined with the proportional operation and the derivative operation.**

The integral operation cannot be used by itself.

- 1 As the integral time lengthens the integral effect weakens. (It takes time for the effect to settle down.)
- 2 As the integral time shortens the integral effect strengthens and makes the correction action stronger, so sometimes the hunting becomes large. (Same as during proportional.)

### 4.2.5 Derivative Operation (D Operation)

This section explains the control method using the derivative operation.

**The derivative operation is an operation that adds the proportional manipulated value to the change speed to eliminate deviation when a deviation has occurred.**

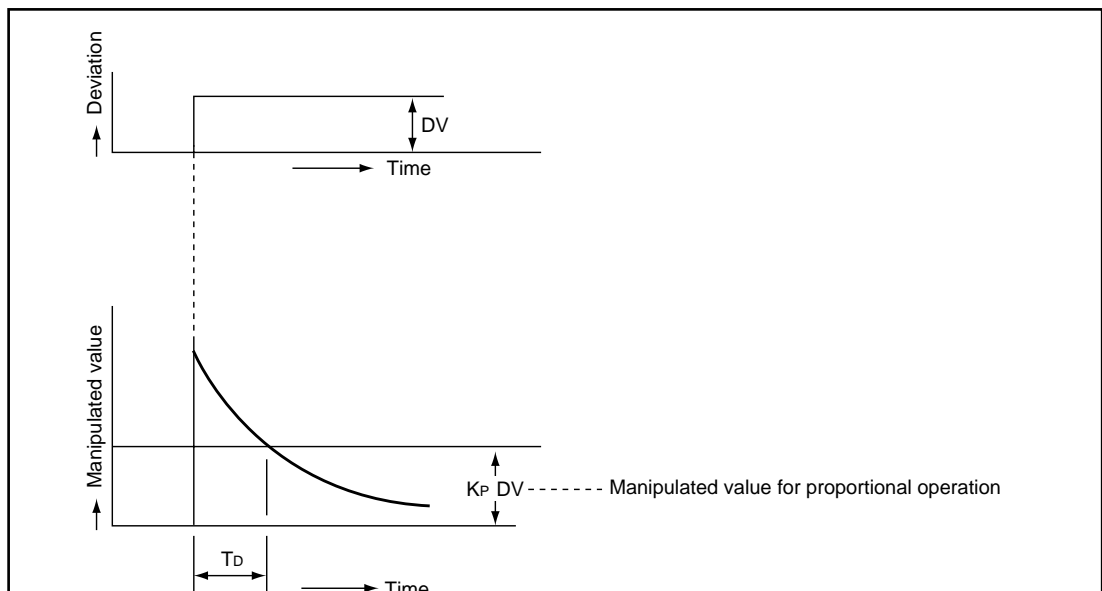
The derivative operation can prevent large changes in the object control from disturbances.

**For derivative operation, the time from when a deviation occurs until the derivative operation manipulated value becomes the manipulated value is called the derivative operation and is shown as  $T_D$ .**

The larger  $T_D$  is the stronger the derivative operation.

**The derivative operation when the deviation is a constant value step response is shown in Figure 4.4.**

Figure 4.4 Derivative operation when deviation is constant



**The derivative operation can be used as PD operation in combination with a proportional operation or as a PID operation in combination with the proportional operation and integral operation.**

The derivative operation cannot be used by itself.

- 1 As the derivative time shortens the derivative effect weakens. (The derivation only has an effect for a short time.)
- 2 As the derivative time lengthens the derivative effect strengthens, so hunting and the system can sometimes become unstable.

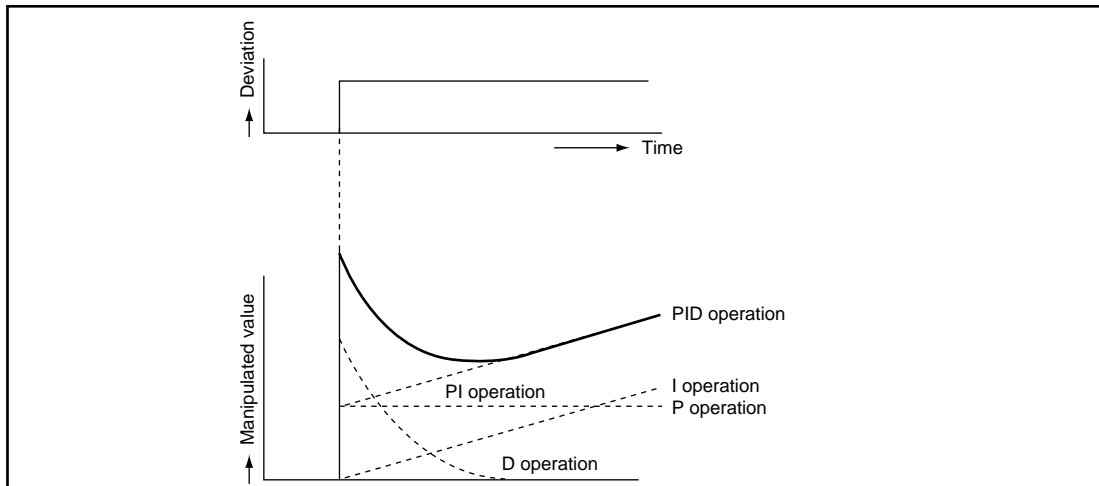
### 4.2.6 PID Operation

This section explains the control operation using combinations of proportional operation (P operation), integral operation (I operation), and derivative operation (D operation).

The PID operation controls the calculated manipulated value using (P+I+D) operation.

The PID operation when the deviation is a constant value step response is shown in Figure 4.5.

Figure 4.5 PID operation when deviation is constant

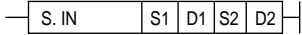
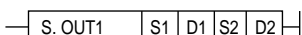


# Instruction List

The process control instruction is largely divided into the I/O control instructions, control operation instructions, correction operation instructions, and arithmetic operation instructions.

## 5.1 How to Read the Instruction List Table

Table 5.1 How to read the instruction list

Classification	Instruction code	Symbol	Processing description	Basic number of steps	Explanation page
I/O control instruction	S. IN		Conducts the input data (PV value) upper and lower limit check, input limiter processing, engineering value conversion, and digital filter processing.	7	8-1
	S. OUT1		Calculates the MV value (0 to 100%) from the input data ( $\Delta MV$ ), and conducts upper and lower limit and change rate limiter processing and output conversion.	8	8-5

①
②
③
④
⑤
⑥

**Explanation**

- 1 Classifies the instructions by application.
- 2 Shows the instruction signal used by the program.
- 3 Shows the symbol diagram used in the circuit.

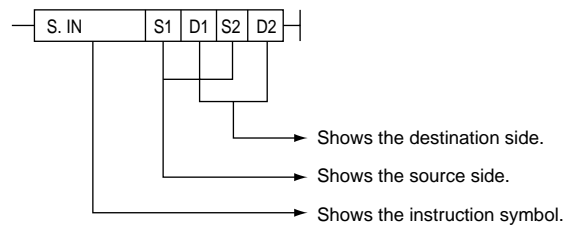


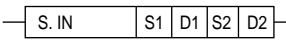
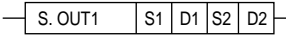
Figure 5.1 Symbols in the circuit

Destination: Shows the destination of the data after operation.  
 Source: Stores the data before the operation.

- 4 Shows the processing contents of each instruction.
- 5 Shows the number of steps for each instruction. For information regarding the number of steps refer to Item 6.2.
- 6 Shows the explanation page for each instruction.

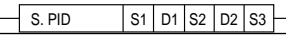
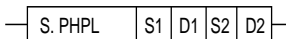
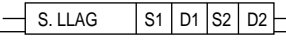
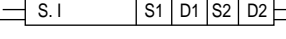
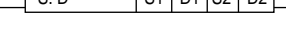
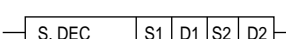
## 5.2 I/O Control Instructions

Table 5.2 I/O instructions

Classification	Instruction code	Symbol	Processing description	Basic number of steps	Explanation page
I/O control instruction	S. IN		Conducts the input data (PV value) upper and lower limit check, input limiter processing, engineering value conversion, and digital filter processing.	7	8-1
	S. OUT1		Calculates the MV value (0 to 100%) from the input data (MV), processes the upper and lower limit and change rate limiter processing, and conducts output conversion.	8	8-5

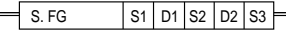
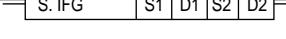
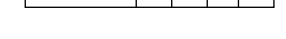

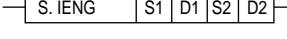
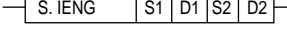
## 5.3 Control Operation Instructions

Table 5.3 Control Operation Instructions

Classification	Instruction code	Symbol	Processing description	Basic number of steps	Explanation page
Control operation instructions	S. PID		Conducts process value derivative type (partial derivative) PID operations.	9	8-9
	S. PHPL		Conducts an upper limit value/lower limit value check of the PV value output by the S.IN instruction.	8	8-15
	S. LLAG		Conducts lead lag compensation for input data and outputs the operation results.	8	8-20
	S. I		Conducts integral operations on the input data and outputs the operation results.	7	8-22
	S. D		Conducts differential operations on the input data and outputs the operation results.	7	8-24
	S. DED		Delays the input data by the specified dead time and then outputs it.	8	8-26

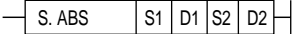
## 5.4 Correction Operation Instructions

Table 5.4 Correction operation instructions

Classification	Instruction code	Symbol	Processing description	Basic number of steps	Explanation page
Correction operation instructions	S. FG		This outputs the input data in accordance with the specified function generator pattern.	7	8-29
	S. IFG		This outputs the specified data in accordance with the specified Inverse function generator pattern.	8	8-31
	S. FLT		This outputs the sampled n units data average value at the specified data collection interval.	8	8-33
	S. ENG		Converts the input percentage (%) data to the specified range value.	8	8-35
	S. IENG		Converts the input data to percentage (%) and outputs it.	8	8-37
					

## 5.5 Arithmetic Operation Instruction

Table 5.5 Arithmetic operation instruction

Classification	Instruction code	Symbol	Processing description	Basic number of steps	Explanation page
Arithmetic operation instruction	S. ABS		Outputs the absolute value of the input data.	8	8-39

# Instruction Configuration

## 6.1 Instruction configuration

The instructions that can be used by the process control instructions can be divided into the instruction section and device section.

The instruction section and device applications are as follows.

Instruction section: This shows the functions for these instructions.

Device instruction: This shows the data required for operations and the storage destination of the stored operation results.

The device section is classified as the source device and destination device.

### Source (S)

The source is where the data used for the operation is stored.

The devices specified by each instruction are as follows.

The header device stored that stores the data used by the operation is specified.

The data must be stored in the specified device by the time the operation is executed. The data used by the instruction can be changed by so specifying during program execution.

### Destination (D)

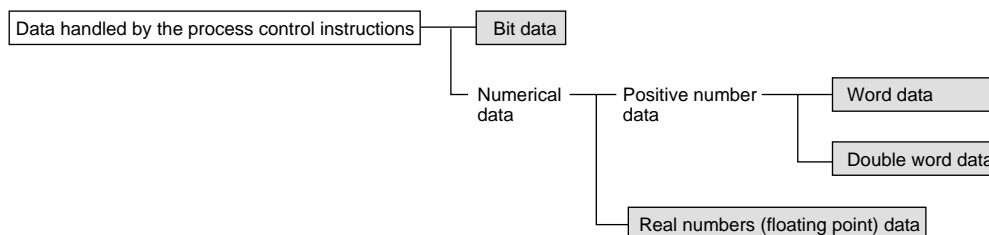
Destination is where the data is stored after operation.

However, depending on the instruction the data to be used by the operation must also be stored in the destination before the operation.

Sets the device for which the data will be stored in the destination.

## 6.2 Method for specifying the data in a device

The following 4 types of data can be used by the process control instructions.

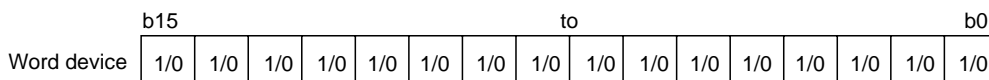


### 6.2.1 For Bit Data

The bit data is the data that handles 1-bit units for alarm condition, selection, etc.

#### When word devices are used

Can use the specified bit No.'s 1/0 as bit data by specifying bit No.

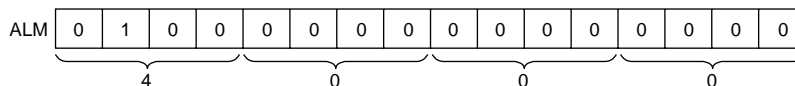


**6.2.2 For Word (16-bit) Data**

Word data is the 16-bit numerical data that is used for the loop tag memory bit pack contents and operation constants, etc.

Hexadecimal constant: 0000H to FFFFH

Example: For the loop tag memory ALM (standard value setting 4000H)

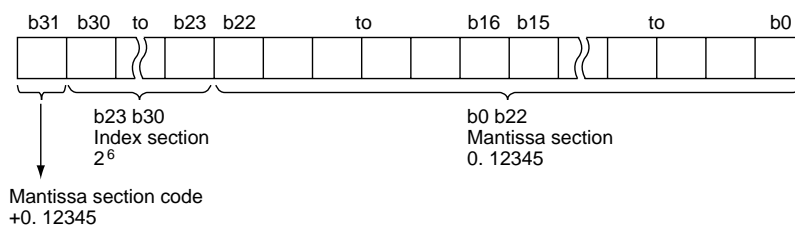


**6.2.3 For Real Number Data (Floating Point Data)**

The data required for operations and the operation results are 32-bit floating point data. Floating point data is displayed as follows using 2 word devices.

0.12345 (mantissa section) x 2<sup>6</sup> [index section]

The bit configuration when the floating point data is expressed internally and its meaning are as follows.



Mantissa section code This shows the mantissa section code in b31.  
 0: Positive  
 1: Negative

Index section This shows the 2<sup>n</sup>'s n and b23 to b30. The n from b23 to b30's BIN value is as follows.

b23 b30	FFH	FEH	FDH		81H	80H	7FH	7EH	
n	Unused	127	126		2	1	0	1	

Mantissa section This shows the value of 1234 as 0.12345 in binary for the b0 to b22's 23 bit.

**Point**

The process control instruction floating point data can be monitored using the peripheral equipment monitor function.  
 When displayed as 0, b0 to b31 are all 0.



### 6.2.4 Process Control Function Operation Error

The process control instruction function is the basic function for conducting floating point operations. The process control instructions can be realized while using these functions.

Operation errors from these process control instruction functions are stored in the following special registers. For information regarding other than operation errors, refer to the error codes listed in the QnACPU (Common Instruction Edition). (The error codes are stored in special register SD0.)

#### Remarks

The following contents for errors other than operation errors are stored in the special register.

Error No. 4100: When there is data that cannot be handled.

4300: When the specified instruction is incorrect.

4301: When the extension instruction number of devices is incorrect.

4302: When a device that cannot be specified is specified.

**For error No. 4100, the detailed information is stored in special registers SD1502 to SD1503. At times other than when a process control instruction function operation error occurs, SD1502 and SD1503 are set to 0.**

SD1502: This shows the error code when an error occurs in the process control instruction function.

SD1053: This shows the instruction process No. when an error occurs.

For an explanation of the error contents refer to the appendix.

### 6.2.5 Instruction Execution Conditions

The process control instructions are instructions that are executed while the input condition is on.

### 6.2.6 Number of Steps

The number of process control instruction steps differs depending upon the number of instruction characters, the device used, and whether or not an indirect setting is valid.

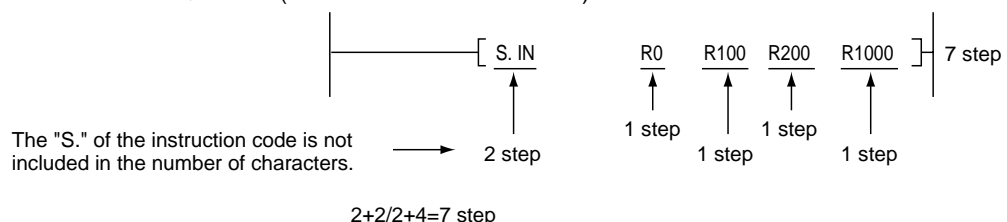
The basic number of steps for the extension instruction are as follows.

$$\text{Number of steps for the extension instruction} = 2 + \frac{\text{number of instruction characters}^{\text{Note 1}}}{2} + \text{number of devices}$$

Note 1:

The number of characters is calculated by adding 1 when the number is odd. (For example when rounding up the results of a division.)

For details refer to QnACPU (Common Instruction Edition).



### 6.2.7 Index Qualification

The index qualification that can be used by the extension instruction is the same as that that can be used by the QnACPU fundamental instruction.

# How to Read Instructions

The following format will be used to explain how to read instructions presented hereafter.

① → 8. Process Control Instruction — S.IN — MELSEC QnA

② → 8 Process Control Instruction

③ →

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc\c		Special function module Uc\Gc	Index register Zc	Constant	Other
	Bit	Word		Bit	Word				
⑤	—	○							
⑥	—	○							
⑦	—	○							
⑧	—	○							

④ →

⑤ →

Setting data	Description	Data format
⑤	Input block header address	Real number
⑥	Block memory header device	Device name
⑦	Operation constant header device	Device name
⑧	Loop tag memory header device	Device name

⑥ → **Function**

This function reads the digital value converted by the A/D converter and conducts range check, input limiter, engineering value conversion, and digital filter processing.

8-1

- Shows the instruction symbol.
- Shows the item No. in the instruction summary.
- b is added to devices that can be used by the instruction. The usage classifications for devices that can be used is shown below.

Device classifications	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc\c		Special function module Uc\Gc	Index register Zc	Constant	Other
	Bit	Word		Bit	Word				
Usable devices	FX, FY, S, SM, X, Y, M, L, F, B, SB	A, VD, SD T, C, D, W, SW, ST	R, ZR	Jc\X Jc\Y Jc\B Jc\SB	Jc\W Jc\SW	Uc\Gc	Z	Decimal constant Hexadecimal constant Real number constant Character string constant	P, I, J, U, DX, DY, N, BL, TR, BL\S

An asterisk by a constant or other use shows what device can be used. For constants a decimal constant is shown by a K, hexadecimal constant by H, real number constant by E, and a character string constant by \$.

8. Process Control Instruction — S. IN — MELSEC QnA

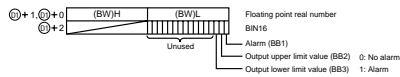
(1) Handling data

(a) Input data

Stores the input value (E1) in ⑤.  
 ⑤+1, ⑤+0 (E1)H (E1)L Floating point real number

(b) Block memory

The output value (BW), alarm (BB1), input upper limit value (BB2), and input lower limit value (BB3) are stored in ⑥.  
 ⑥+2's BB4 to BB16 bits are not used.



(c) Shows the contents of the operation constant set in the [S2] device.

Item name	Item	Settable range	Standard value setting
⑤+1, ⑤+0 Engineering conversion upper limit	EMAC	-99999 to 99999	100.0
⑤+3, ⑤+2 Engineering conversion lower limit	EMIN	-99999 to 99999	0.0
⑤+5, ⑤+4 Input upper limit	NMAX	-99999 to 99999	100.0
⑤+7, ⑤+6 Input lower limit	NMIN	-99999 to 99999	0.0
⑤+9, ⑤+8 Upper limit side range error occurrence	HH	-99999 to 99999	110.0
⑤+11, ⑤+10 Upper limit side range error return	H	-99999 to 99999	100.0
⑤+13, ⑤+12 Lower limit side range error return	L	-99999 to 99999	0.0
⑤+15, ⑤+14 Lower limit side range error occurrence	LL	-99999 to 99999	10.0

(d) Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

(e) Shows the loop tag memory used by [D2].

Item name	Item	Settable range	Standard value setting
⑤+1 Operation mode	MODE	0 to FFFFH	8H
⑤+3 Alarm detection	ALM	0 to FFFFH	4000H
⑤+4 Alarm detection prohibited	INH	0 to FFFFH	4000H
⑤+39, ⑤+38 Filter coefficient		0 to 1	0.2

Error ← • When an overflow occurs during an operation. (Error code: 4100)

8-2

4 This shows the expression and instruction execution conditions in the circuit mode.

Execution conditions	Normal execution	Executed during on	Executed once during on	Executed once during off
Displays the No. of the explanation page	Nothing recorded		Nothing recorded	Nothing recorded

5 This shows the instruction setting data explanations and data formats.

Data format	Description
BIN16	Shows how each BIN 16-bit and word device header No. is handled.
Real number	Shows how the floating point data is handled.
Device name	Shows how the device name is handled.
Dummy	Shows how the dummy device is handled.

6 This shows the function executing the instruction.

7 : This shows word data.

: This shows the floating point real number data.

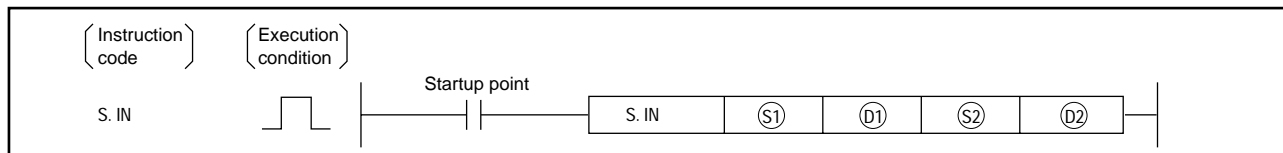
8 This shows the conditions and error No. created by an error.

# Process Control Instruction

## 8.1 I/O Instructions

### 8.1.1 Analog Input Processing

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b	—		—	—	—	—	
t	—	b	—		—	—	—	—	
p	—	b	—		—	—	—	—	
u	—	b	—		—	—	—	—	

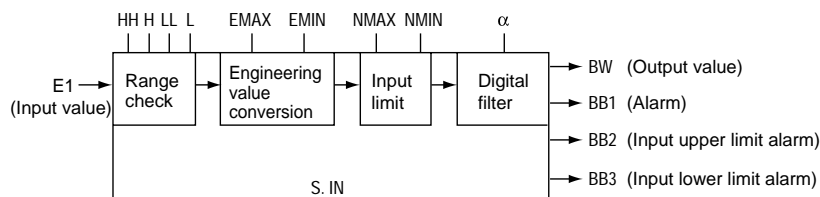


#### Set data

Setting data	Description	Data format
o	Input block header address	Real number
t	Block memory header device	Device name
p	Operation constant header device	Device name
u	Loop tag memory header device	Device name

#### Function

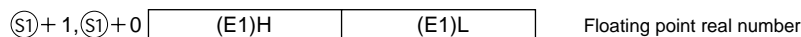
This function reads the digital value converted by the A/D converter and conducts range check, input limiter, engineering value conversion, and digital filter processing.



**Handling data**

Input data

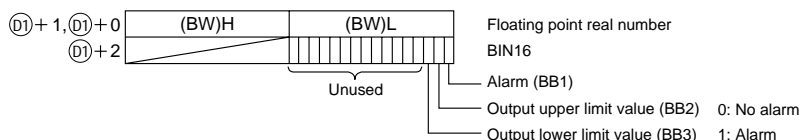
Stores the input value (E1) in o.



Block memory

The output value (BW), alarm (BB1), input upper limit value (BB2), and input lower limit value (BB3) are stored in t.

t+2's BB4 to BB16 bits are not used.



Shows the contents of the operation constant set in the [S2] device.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Engineering conversion upper limit	EMAC	-999999 to 999999	100.0
p+3, p+2	Engineering conversion lower limit	EMIN	-999999 to 999999	0.0
p+5, p+4	Input upper limit	NMAX	-999999 to 999999	100.0
p+7, p+6	Input lower limit	NMIN	-999999 to 999999	0.0
p+9, p+8	Upper limit side range error occurrence	HH	-999999 to 999999	110.0
p+11, p+10	Upper limit side range error return	H	-999999 to 999999	100.0
p+13, p+12	Lower limit side range error return	L	-999999 to 999999	0.0
p+15, p+14	Lower limit side range error occurrence	LL	-999999 to 999999	10.0

Execution time ( T )

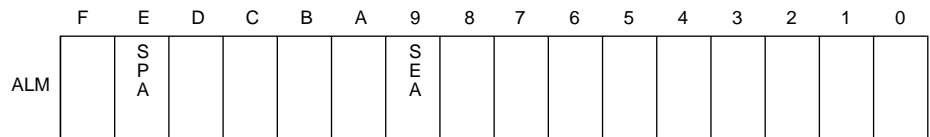
Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

Shows the loop tag memory used by [D2].

	Item name	Item	Settable range	Standard value setting
u+1	Operation mode	MODE	0 to FFFFH	8H
u+3	Alarm detection	ALM	0 to FFFFH	4000H
u+4	Alarm detection prohibited	INH	0 to FFFFH	4000H
u+39, u+38	Filter coefficient		0 to 1	0.2

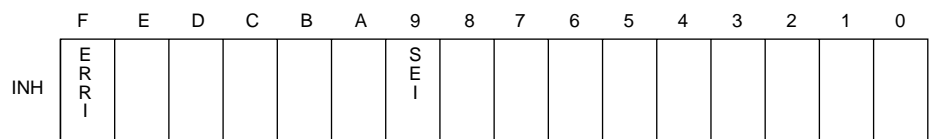
The bit used by the alarm detection (ALM) is shown below.

SPA can be set by the user, and is 1 when SEA outputs an alarm.



The bit used by alarm detection prohibition (INH) is shown below.

ERRI and SEI can be set by the user.



**Processing explanation**

Loop STOP processing

- 1 When the alarm detection (ALM) SPA is 1, the following process is conducted and ended.
  - a BW holds the previous BW value.
  - b The operation mode is changed to MAN (MANUAL).
  - c BB's BB1 to BB3 is made to 0.
  - d Alarm detection (ALM)'s SEA is made to 0.

- 2 When the alarm detection (ALM)'s SPA is 0, the following processing is conducted.

a Range check

This conducts an upper and lower limit check for the input value (E1) and outputs an alarm.

Upper and lower limit checklist

Upper limit check

Condition	BB2
Input value HH	1
Input value H	0
H<Input value<HH	Previous value

HH : Upper limit range error occurred  
 H : Upper limit range error return

Lower limit check

Condition	BB3
Input value LL	1
Input value L	0
LL<Input value<L	Previous value

LL : Lower limit range error occurred  
 L : Lower limit range error return

Note 1:

When the alarm detection prohibition permitted (INH)'s ERR1 or SE1 is 1, the alarm is stopped, so SEA, BB1, BB2, and BB3 are made 0.

Note 2:

It is made so that BB1=SEA=BB2/VBB3.

Previous value hold processing.

This processing specifies whether the output value is held or whether operation continues as is when a range over occurs during a range check.

The following processing is conducted and ended when there is a hold mode (SM1500=1) and a sensor error occurs during the range check (BB1=1).

- 1 BW holds the previous BW value.
- 2 BB is set to the error information at that time.

Input limiter

This sets the upper and lower limit limiter for the input value (E1).

For example, if the upper limit value (NMAX) is set to 50 and the input value is 55, the value actually used is 50.

Condition	Result (T1)
Input value NMAX	NMAX
Input value NMIN	NMIN
NMIN<Input value<NMAX	Input (E1)

Engineering value conversion

Engineering value conversion converts the count value using engineering value conversion when the A/D converter count value is the IN instruction input value.

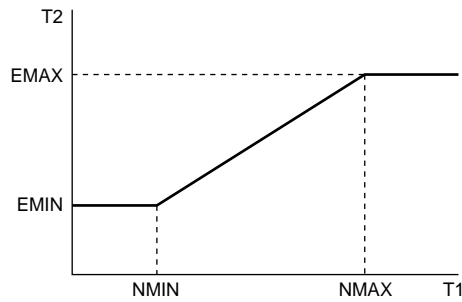
Engineering value conversion

$$T2 = (EMAX - EMIN) \frac{T1 - NMIN}{NMAX - NMIN} + EMIN$$

EMAX: Engineering conversion upper limit      NMAX: Input upper limit

EMIN: Engineering conversion lower limit      NMIN: Input lower limit

T1: Input limiter results      T2: Engineering value conversion results



Digital filter

A digital operation is conducted for the input value to reduce the effects of noise.

Digital filter

$$BW = T2 + (\text{Previous BW value} - T2)$$

: Filter coefficient      T2: Engineering value conversion results

**Error**

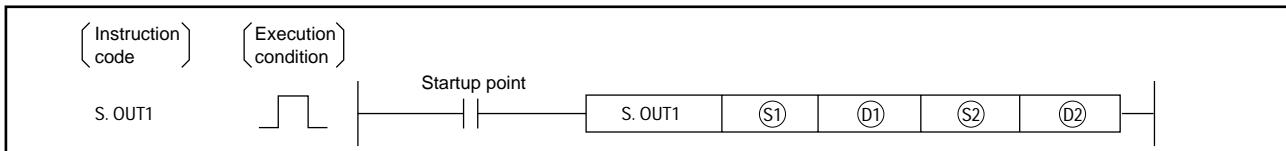
When an overflow occurs during an operation.

(Error code: 4100)



8.1.2 Output Processing-1 with Mode Switching

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b			—		—	—	
t	—	b			—		—	—	
p	—	b			—		—	—	
u	—	b			—		—	—	

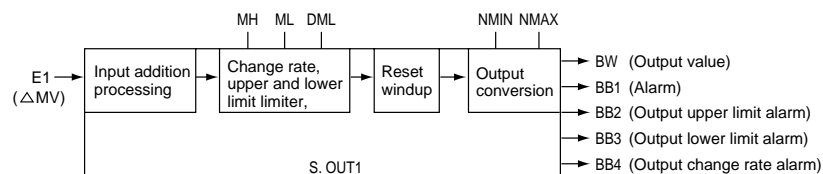


Set data

Setting data	Description	Data format
o	Input block header address	Real number
t	Block memory header device	Device name
p	Operation constant header device	Device name
u	Loop tag memory header device	Device name

Function

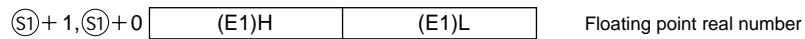
This function has an automatic/manual switching function and switches the output method in accordance with each operation mode. The E1 (input value) is read and input addition processing, change rate, lower and upper limit limiter, and reset windup output conversion processing is conducted and output to BW.



**Handling data**

Input data

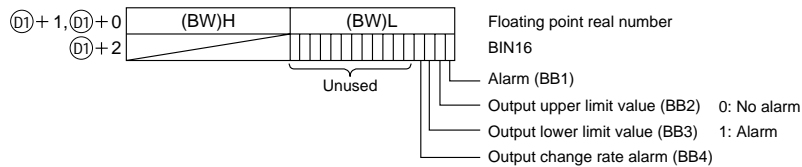
Stores the input value (E1= MV) in o.



Block memory

The output value (BW), alarm (BB1), input upper limit value (BB2), input lower limit value (BB3), and output change rate alarm (BB4) are stored in t.

t+2's BB5 to BB16 are not used.



Shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Output conversion upper limit	NMAX	-999999 to 999999	100.0
p+3, p+2	Output conversion lower limit	NMIN	-999999 to 999999	0.0

Execution time ( T )

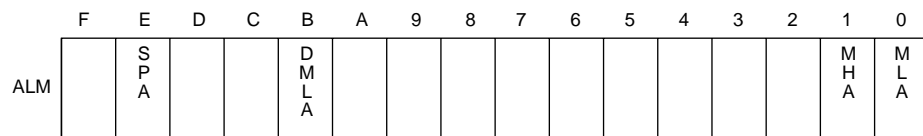
Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

Shows the loop tag memory used by u.

	Item name	Item	Settable range	Standard value setting
u+1	Operation mode	MODE	FFFFH	8H
u+3	Alarm detection	ALM	FFFFH	4000H
u+4	Alarm detection prohibited	INH	FFFFH	4000H
u+13, u+12	Manipulated value	MV	-10 to 110	0.0
u+19, u+18	Output upper limit value	MH	-10 to 110	100.0
u+21, u+20	Output lower limit value	ML	-10 to 110	0.0
u+49, u+48	Output change upper limit value	DML	0 to 100	100.0
u+55, u+54	Integral constant	I	0 to 999999	10.0
u+63, u+62	MV internal operation value	MVP	-999999 to 999999	0.0

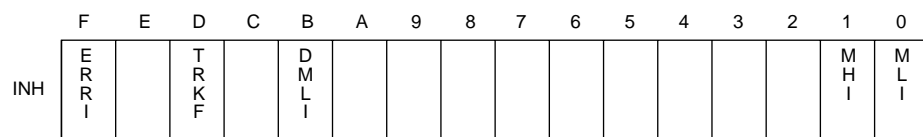
The bit used by the alarm detection (ALM) is shown below.

SPA can be set by the user, and corresponding bit is 1 when DMLA, MHA, MLA outputs an alarm.



The bit used by alarm detection prohibition (INH) is shown below.

ERRI, DMLI, MHI, MHL can be set by the user.



**Processing explanation**

Loop STOP processing

- 1 When the alarm detection (ALM) SPA is 1, the following process is conducted and ended.
  - a BW holds the previous BW value.
  - b The operation mode (MODE) is changed to MAN (MANUAL).
  - c BB's BB1 to BB4 is made to 0.
  - d Alarm detection (ALM)'s MHA, MLA, DMLA is made to 0.

Mode determination

The following processing is conducted by the operation mode (MODE).

- 1 When the operation mode (MODE) is MAN, CMB, CMV, or LCM (alarm clear processing)
  - a The alarm detection (ALM)'s MHA, MLA, and DMLA are made to 0.
  - b BB's BB1 to BB4 is made to 0.
  - c The alarm detection prohibition (INH)'s TRKF is made to 1.
  - d Output conversion processing is conducted and ended.
- 2 When the operation mode is (MODE) AUT, CAB, CAS, CCB, CSV, LCA, or LCC  
 Processing after (c) input addition is conducted. However, when the alarm detection (ALM)'s SEA is 1 or when there is a hold (SM1501) then BB1 to BB4 is made to 0 and ended.

Input addition processing

The estimated MV value (T) is calculated based on the input value (change value: E1= MV).

- 1 The following processing is conducted when the alarm detection prohibition (INH)'s TRKF is 1.
  - a The loop tag memory MVP=MV.
  - b The input value (E1) is made to 0. ( MV=0)
  - c The alarm detection prohibition (INH)'s TRKF is made to 0.
  - d The estimated MV value is found using the following formula.

$T = E1 + MVP$
$MVP = T$

E1: Input value      MVP: MV internal operation value

- 2 This finds the estimated MV value when the alarm detection prohibition (INH)'s TRKF is 0.
  - a Conducts processes 1's d.

Lower and upper limit and change rate limiter.

This conducts a check of the change rate and upper and lower limit for the input value (E1) and conducts data and alarm output after limiter processing is finished.

The change rate limiter conducts the following operation and outputs the results to BB4 and DMLA.

Condition	BB4, DMLA	T1 (Results)	
$ T - MV  \geq DM$	0	T	
$T - MV > DML$	1	MV + DML	0: No alarm 1: Alarm
$T - MV < -DML$	1	MV - DML	

T: Estimated MV value      MV: Manipulated value  
 DML: Output change upper limit value

Note 1:

When the alarm detection prohibition (INH)'s DMLI and ERRI are 1, the alarm detection (ALM)'s DMLA and BB4 are made to 0.

The upper and lower limit delimiter conducts the following operations and outputs the results to BB2, BB3, MLA, and MHA.

Condition	BB3, MLA	BB2, MHA	MV
T1>MH	0	1	MH
T1<ML	1	0	ML
ML T1 MH	0	0	T1

0: No alarm  
1: Alarm

MH: Output upper limit value      T1: Change rate limiter results  
ML: Output lower limit value

**Note 1:**

When the alarm detection prohibited (INH)'s MHI and ERR1 are 1, the alarm detection (ALM)'s MHA and BB2 are made to be 0.

**Note 2:**

When the alarm detection prohibited (INH)'s MLI and ERR1 are 1, the alarm detection (ALM)'s MLA and BB3 are made to be 0.

**Reset windup**

When the MV exceeds the upper or lower limit value the following operations are performed so that the upper and lower limit values are transferred and a response is made when a deviation is returned.

Condition	Formula
MHA=1 and $\frac{T}{TI} = 1$	$MVP = \frac{T}{TI} (MH - T) + T$
MLA=1 and $\frac{T}{TI} = 1$	$MVP = \frac{T}{TI} (ML - T) + T$

However, when TI=0, reset windup processing is not conducted.

MHA: Output upper limit alarm      MH: Output upper limit value  
MLA: Output lower limit alarm      ML: Output lower limit value  
T: Execution time                      T: Estimated MV value  
TI: Change rate limiter results

**Output conversion processing**

Output conversion conducts the following processing.

Output conversion processing

$$BW \frac{NMAX - NMIN}{100} MV - NMIN$$

NMAX: Input upper limit      NMIN: Input lower limit      MV: Manipulated value

**Hold processing**

This processing specifies whether to hold or to continue processing as is for the output value from the OUT1 instruction when a sensor error occurs (detected by IN instruction) by loop STOP determination processing.

Selection can be made by setting SM1501 to hold or not hold the MV value when a sensor alarm occurs.

- SM1501=0: No hold
- SM1501=1: Hold

**Error**

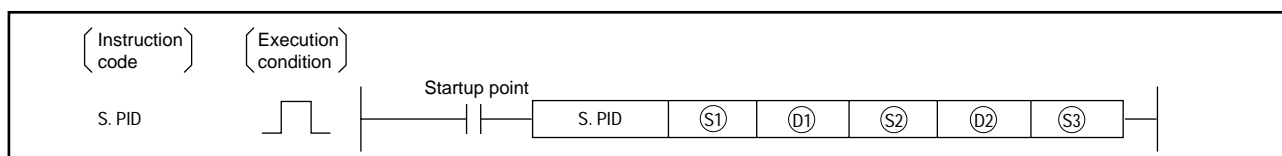
When an overflow occurs during an operation.

(Error code: 4100)

## 8.2 Control Operation Instruction

### 8.2.1 Basic PID

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b					—	—	
t	—	b					—	—	
p	—	b					—	—	
u	—	b					—	—	
q	—	b					—	—	

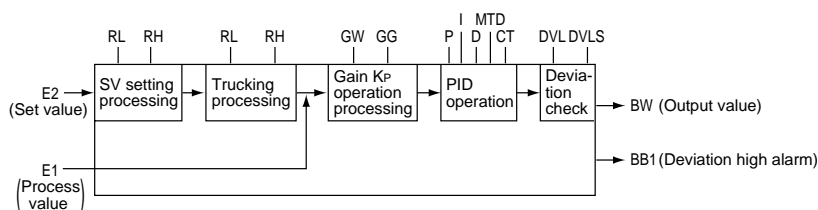


#### Set data

Setting data	Description	Data format
o	Input block header address	Real number
t	Block memory header device	Device name
p	Operation constant header device	Device name
u	Loop tag memory header device	Device name
q	Input block header address or first MV address (during use)	Real number

#### Function

When the control time is reached SV setting processing, trucking processing, gain Kp operation processing, PID operation, and deviation check are conducted.



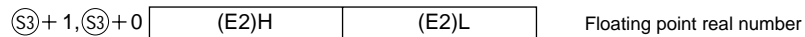
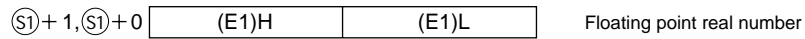
**Data handled**

Input data

- 1 The input value (E1) is stored in o.
- 2 The q set value (E2) can be used when the set value (E2) is set (0 bit=1) by the operation constant set value parameter.

For other cases set the dummy device (SD1506).

In addition, when the set value (E2) is set by the first loop tag memory MV value, set the device (+12: MV value) set by the first loop tag memory MV value.

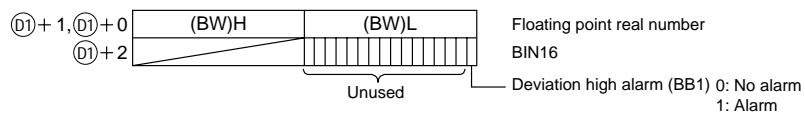


**Block memory**

The output value ( MV) and deviation high alarm (BB1) are stored in t.

t+2's BB2 to BB16 are not used.

The output value is made to 0 when an error occurs.



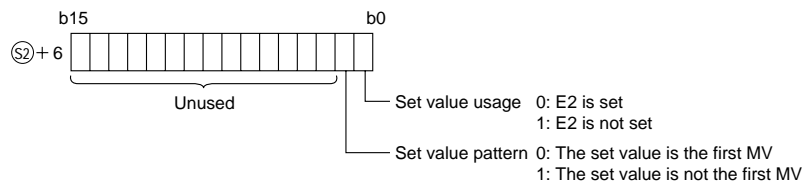
This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Derivative gain	MTD	0 to 999999	8.0
p+3, p+2	Deviation high alarm hysteresis	DVLS	0 to 100	2.0
p+4	Reverse action, forward action	PN	0 to 1	0
p+5	Trucking bit	TRK	0 to 1	0
p+6	Set value pattern	SVPTN	0 to 3	3

0: Reverse action  
1: Forward action

0: Not trucked  
1: Trucked

The set value pattern (SVPTN) is a device that sets whether the set value is set by q and whether that set value is set by the first loop device (+12: MV value). The set value pattern (SVPTN) cannot use bits 2 through 15.

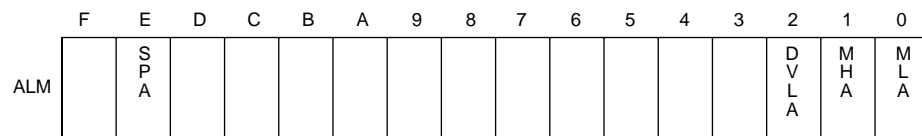


Shows the loop tag memory used by u.

	Item name	Item	Settable range	Standard value setting
u+1	Operation mode	MODE	0 to FFFFH	8H
u+3	Alarm detection	ALM	0 to FFFFH	4000H
u+4	Alarm detection prohibited	INH	0 to FFFFH	4000H
u+15, u+14	Set value	SV	RH* (RL*) to RL* (RH*)	0.0
u+17, u+16	Deviation	DV	-110 to 110	0.0
u+47, u+46	Control time (sec)	CT	0 to 999999	1.0
u+51, u+50	Change rate limit value	DVL	0 to 100	100.0
u+53, u+52	Gain	P	0 to 999999	10.0
u+55, u+54	Integral constant (sec)	I	0 to 999999	0.0
u+57, u+56	Derivative constant (sec)	D	0 to 999999	0.0
u+59, u+58	Gap width	GM	0 to 100	1.0
u+61, u+60	Gap gain	GG	0 to 999999	0.0
u+63, u+62	MV internal operation value	MVP	-999999 to 999999	0.0

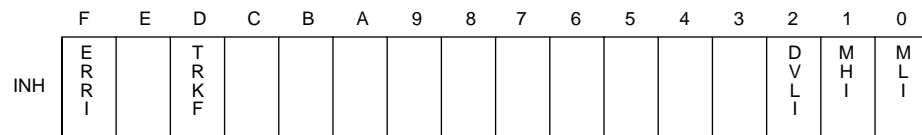
The bit used by the alarm detection (ALM) is shown below.

SPA can be set by the user, and corresponding bit is 1 when DVLA, MHA, MLA outputs an alarm.



The bit used by alarm detection prohibition (INH) is shown below.

ERRI, DVLI, MHI, MLI can be set by the user.



Loop tag pasted value memory

This shows the contents of the loop tag passed value memory used by the PID instruction. The user does not need to set the contents. However, for the initial state it must be cleared by the sequence.

	Description
u+96	Control time counter initial set completed flag
u+97	Control time counter
u+103, u+102	Bn-1
u+105, u+104	PV'n (Process value)
u+107, u+106	PV'n-1 (Previous process value)
u+109, u+108	PV'n-2 (Process value before last)
u+111, u+110	DVn-1

Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

**Processing explanation**

Loop STOP processing

- 1 When the alarm detection (ALM) SPA is 1, the following process is conducted and ended.
  - a BW outputs 0.
  - b Alarm detection (ALM)'s DVLA becomes 0.
  - c The operation mode (MODE) is changed to MAN (MANUAL).
  - d BB's BB1 to BB4 is made to 0.
 

When the alarm detection (ALM) SPA is 0, processing is conducted from the (b) control time determination.

Control time determination

This determines whether the control time from the control time (CT) has been reached and conducts the following processing.

- 1 If the control time has not been reached
 

The BW value is made to 0 and processing is ended.
- 2 If the control time has been reached
 

Processing is conducted from the (c) SV setting.

SV setting processing

The following processing is conducted by the operation mode (MODE).

- 1 When the operation mode (MODE) is either CAS, CCB, or CSV (when the input value is used as the set value)
  - a When the set value (E2) is not set, processing is conducted from the (d) trucking processing.
  - b When q set value (E2) is set, processing is conducted from the (d) trucking processing after the following engineering value conversion has been conducted.

Engineering value conversion

$$SV_n = \frac{RH - RL}{100} E2 + RL$$

RH: Engineering value upper limit    RL: Engineering value lower limit    E2: Set value

- 2 When the operation mode (MODE) is MAN, AUT, CMV, CMB, CAB, LCM, LCA, or LCC
  - a Processing is conducted from the (d) trucking processing.

Trucking processing

This conducts the engineering value conversion (SVn').

Inverse engineering value conversion

$$SV_n' = \frac{100}{RH - RL} (SV_n - RL)$$

Trucking processing is conducted when the following conditions occur.

- 1 When the operation constant's TRK is 1.
- 2 When the set value (E2) is used.
- 3 When the mode is not CAS, CCB, or CSV.

Trucking processing stores the set value (E2) after the above engineering value conversion (SVn') is conducted.

**E2=SVn'**

In addition, when the set value (E2) is the first loop tag memory MVn, the first loop tag memory alarm detection prohibition (INH)'s TRKF is made to be 1.



Gain K<sub>P</sub> operation processing

The deviation (DV) is calculated using the following conditions.

Condition	Calculation results
Forward action (PN=0)	DV=E1 - SVn'
Reverse action (PN=1)	DV=SVn' - E1

DV: Deviation      SVn': Engineering value conversion processing results  
E1: Process value

Next the PID value final output value's output gain (K) is calculated using the following conditions.

Condition	Formula
When  DV  ≤ GW	K=GG
When  DV  > GW	$K=1 \cdot \frac{(1 - GG) \cdot GW}{ DV }$

DV: Deviation      K: Output gain      GW: Gap width  
GG: Gap gain      KP=K x gain (P)

The PID calculation is found using the following formula.

Item	For forward action (PN=1)	For reverse action (PN=0)
B <sub>n</sub>	$B_{n-1} + \frac{MD \cdot TD}{MD \cdot CT \cdot TD}$ $\{ (PV_n - 2 \cdot PV_{n-1} + PV_{n-2}) \cdot \frac{CT \cdot B_{n-1}}{TD} \}$	$B_{n-1} + \frac{MD \cdot TD}{MD \cdot CT \cdot TD}$ $\{ (PV_n - 2 \cdot PV_{n-1} + PV_{n-2}) \cdot \frac{CT \cdot B_{n-1}}{TD} \}$
BW ( MV)	$K_P \cdot \{ (DV_n - DV_{n-1}) + \frac{CT}{TI} \cdot DV_n + B_n \}$	

K<sub>P</sub>: K x gain (P)                      M<sub>d</sub>: Derivative gain (MTD)  
T<sub>i</sub>: Integral constant (I)              C<sub>T</sub>: Control time  
T<sub>d</sub>: Derivative constant (D)          P<sub>V<sub>n</sub></sub>: Process value (E1)  
P<sub>V<sub>n-1</sub></sub>: Previous process value      P<sub>V<sub>n-2</sub></sub>: Process value before last

However, special processing is done for the following cases so take due precaution.

Condition	Measures
When either 1 or 2 below 1. T <sub>D</sub> =0 2. When the operation mode (MODE) is either MAN, LCM, or CMV	B <sub>n</sub> =0 (However, passed value set is conducted.)
For either 1, 2, or 3 below 1. T <sub>i</sub> =0 2. When MVP > MH when an MH or ML error occurs $\frac{CT}{TI} \cdot DV_n = 0$ 3. When MVP < MH when an MH or ML error occurs $\frac{CT}{TI} \cdot DV_n = 0$	$\frac{CT}{TI} \cdot DV_n = 0$

When the PID operation is ended the PV passed value memory data is overwritten with new data.  
P<sub>V<sub>n-2</sub></sub> P<sub>V<sub>n-1</sub></sub> P<sub>V<sub>n-1</sub></sub> P<sub>V<sub>n</sub></sub> P<sub>V<sub>n</sub></sub> E1

## Deviation check

A deviation check is conducted under the following conditions and the results are output to DVLA and BB1.

Conditions	Results
$DVL <  DV $	DVLA=BB1=1
$(DVL - DVLS) <  DV  < DVL$	DVLA=BB1=Previous process value status hold
$ DV  < (DVL - DVLS)$	DVLA=BB1=0

DV: Deviation      DVL: Change rate limit value

DVLS: Deviation size alarm hysteresis

## Note 1:

When the DVLI or ERRI are 1, the DVLA and BB1 become 0.

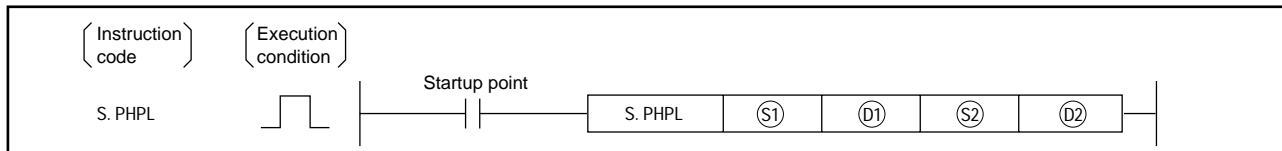
**Error**

When an overflow occurs during an operation.

(Error code: 4100)

8.2.2 Process High Alarm Process Low Alarm

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b	—		—	—	—	—	
t	—	b	—		—	—	—	—	
p	—	b	—		—	—	—	—	
u	—	b	—		—	—	—	—	

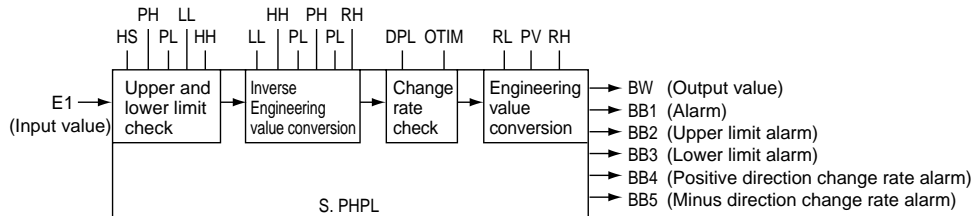


Set data

Setting data	Description	Data format
o	Input block header address	Real number
t	Block memory header device	Device name
p	Dummy device	Dummy
u	Loop tag memory header device	Device name

Function

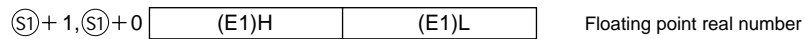
An upper and lower limit check is conducted for the input (E1) value and then alarm is output.



**Handling data**

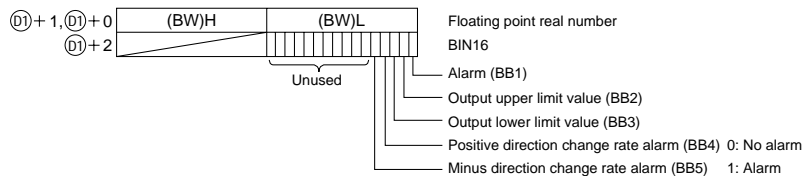
Input data

Stores the input value (E1) in o.



Block memory

The output value (BW), alarm (BB1), upper limit alarm (BB2), lower limit alarm (BB3), positive direction change rate alarm (BB4), and negative direction change rate alarm (BB5) are stored in t. t+2's BB6 to BB16 bits are not used.



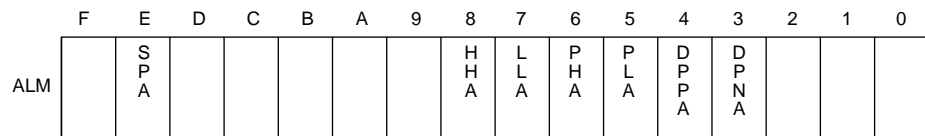
Set the dummy device (SD1506) in p.

Shows the loop tag memory used by u.

	Item name	Item	Settable range	Standard value setting
u+1	Operation mode	MODE	0 to FFFFH	8H
u+3	Alarm detection	ALM	0 to FFFFH	4000H
u+4	Alarm detection prohibited	INH	0 to FFFFH	4000H
u+11, u+10	Process value	PV	RH* (RL*) to RL* (RH*)	0.0
u+23, u+22	Engineering range upper limit	RH	-999999 to 999999	100.0
u+25, u+24	Engineering range lower limit	RL	-999999 to 999999	0.0
u+27, u+26	Upper limit alarm value	PH	RH* (RL*) to RL* (RH*)	100.0
u+29, u+28	Lower limit alarm value	PL	RH* (RL*) to RL* (RH*)	0.0
u+31, u+30	Upper upper limit alarm value	HH	RH* (RL*) to RL* (RH*)	100.0
u+33, u+32	Lower lower limit alarm value	LL	RH* (RL*) to RL* (RH*)	0.0
u+41, u+40	Upper and lower limit alarm hysteresis	HS	0 to 999999	0.0
u+43, u+42	Change rate alarm check time	CTIM	0 to 999999	0.0
u+45, u+44	Change rate alarm value	DPL	0 to 100	100.0

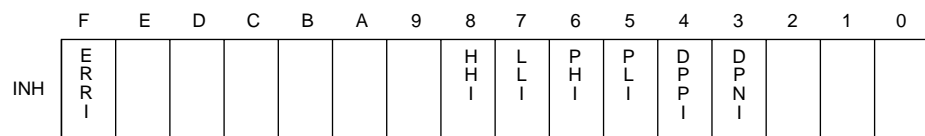
The bit used by the alarm detection (ALM) is shown below.

SPA can be set by the user, and corresponding bit is 1 when HHA, LLA, PHA, PLA, DPPA, DPNA outputs an alarm.



The bit used by alarm detection prohibition (INH) is shown below.

ERRI, HHI, LLI, PHI, PLI, DPPI, DPNI can be set by the user.



Loop tag pasted value memory

This shows the contents of the loop tag pasted value memory used by the PHPL instruction. The user does not need to set the contents. However, for the initial state it must be cleared by the sequence.

	Description
u+124	Change rate monitor counter initial set completed flag
u+125	Change rate monitor counter
u+127, u+126	Xn-m

**Processing explanation**

Loop stop processing

The following processing is conducted when the alarm detection (ALM) is one.

1

Inverse engineering value conversion
$BW \frac{100}{RH - RL} (PV \ RL)$

RH: Engineering value upper limit      RL: Engineering value lower limit  
 PV: Process value

2 Changes BB's BB1 to BB5 to 0.

3 Changes all alarm detection (ALM)'s DPNA, DPPA, LLA, HHA, PLA, and PHA to 0.

When the loop STOP processing is not conducted, processing is conducted from the (b) engineering value reverse conversion processing.

Inverse engineering value conversion processing

The following calculation is conducted to make the input value (E1) and engineering value PH, PL, HH, and LL ranges match.

$PH' \frac{100}{RH - RL} (PH \ RL)$	$PL' \frac{100}{RH - RL} (PL \ RL)$
$HH' \frac{100}{RH - RL} (HH \ RL)$	$LL' \frac{100}{RH - RL} (LL \ RL)$

PH: Upper limit value alarm      HH: Upper upper limit value alarm  
 PL: Lower limit value alarm      LL: Lower lower limit value alarm

## Upper and lower limit check

This conducts an upper and lower limit check for the value from the input value (PV) process by engineering value reverse conversion.

Check item	Input value (E1) condition	Alarm (ALM)	BB2
Upper limit check	Input value>PH'	PHA=1	1
	Input value PH' HS	PHA=0	0
	Other	PHA hold	Hold

## Note 1:

When the alarm detection prohibition (INH)'s ERRI and PHI are 0 the PHA and BB2 become 0.

Check item	Input value (E1) condition	Alarm (ALM)	BB3
Lower limit check	Input value<PL'	PLA=1	1
	Input value PL' HS	PLA=0	0
	Other	PLA hold	Hold

## Note 1:

When the alarm detection prohibition (INH)'s ERRI and PHI are 0 the PLA and BB3 become 0.

Check item	Input value (E1) condition	Alarm (ALM)
Upper upper limit check	Input value>HH'	HHA=1
	Input value HH' HS	HHA=0
	Other	HHA hold

## Note 1:

When the alarm detection prohibition (INH)'s ERRI and HHI are 0 the HHA become 0.

Check item	Input value (E1) condition	Alarm (ALM)
Lower lower limit check	Input value<LL'	LLA=1
	Input value LL' HS	LLA=0
	Other	LLA hold

## Note 1:

When the alarm detection prohibition (INH)'s ERRI and LLI are 0 the LLA become 0.

Change rate check

This function compares the input data change with the change rate alarm value (DPL) for each execution time between the time width CTIM (sec) specified by the loop tag and conducts the change rate alarm check using the following formula.

$$m = \frac{CTIM}{T}$$

m: Change rate monitoring counter      T: Execution time

CTIM: Change rate alarm check time

Check item	Input value (E1) condition	Alarm (ALM)	BB4
Change rate check	X <sub>n+m</sub> X <sub>n</sub> DPL	DPPA=1	1
	Other	DPPA=0	0

Note 1:

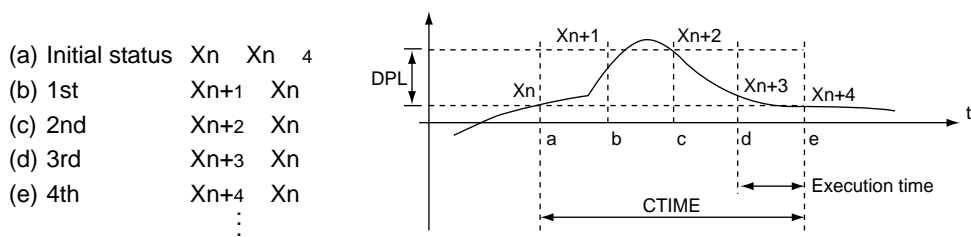
When either of the alarm detection prohibition (INH)'s ERRI or DPPI are 1, then DPPA and BB4 become 0.

Check item	Input value (E1) condition	Alarm (ALM)	BB5
Change rate check	X <sub>n+m</sub> X <sub>n</sub> DPL	DPNA=1	1
	Other	DPNA=0	0

Note 1:

When either of the alarm detection prohibition (INH)'s ERRI or DPNI are 1, then DPNA and BB5 become 0.

m is changed from 1 to m. For example, when m = 4 then it becomes as shown below.



- (a) Initial status X<sub>n</sub> X<sub>n</sub> 4
- (b) 1st X<sub>n+1</sub> X<sub>n</sub>
- (c) 2nd X<sub>n+2</sub> X<sub>n</sub>
- (d) 3rd X<sub>n+3</sub> X<sub>n</sub>
- (e) 4th X<sub>n+4</sub> X<sub>n</sub>

However, when m = 0 (integer portion) then no processing is conducted.

Engineering value conversion

$$PV = \frac{RH - RL}{100} (E1 - RL)$$

RH: Engineering value limit      RL: Engineering value lower limit      E1: Input value

BB1 output

The BB2 to BB5's OR signal is output to BB1.

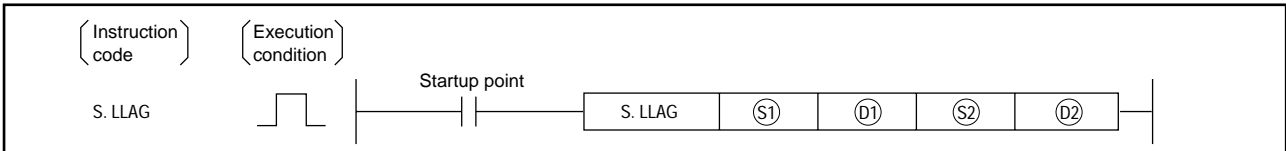
**Error**

When an overflow occurs during an operation.

(Error code: 4100)

**8.2.3 Lead Lag**

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b	—		—	—	—	—	
t	—	b	—		—	—	—	—	
p	—	b	—		—	—	—	—	
u	—	b	—		—	—	—	—	

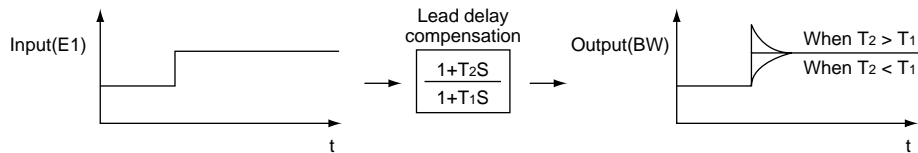


**Set data**

Setting data	Description	Data format
o	Input block header address	Device name
t	Block memory header device	Real number
p	Operation constant header device	Device name
u	Local work memory header device	Device name

**Function**

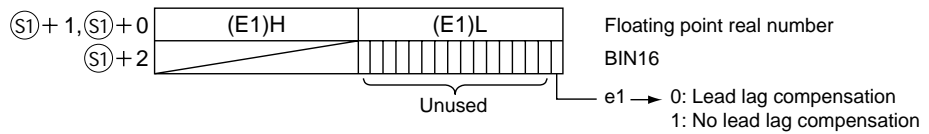
This function executes the lead delay operation using the operation constant setting or the action signal e1 and outputs it.



**Data handling**

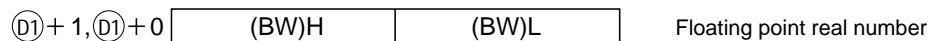
**Input data**

Stores the input value (E1) in o.  
o + 2's 1 through 15 bits are not used.



**Block memory**

The BW (Output value) is stored in t.  
BB is not used.





Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Lag time	T <sub>1</sub>	0 to 999999	1.0
p+3, p+2	Lead time	T <sub>2</sub>	0 to 999999	1.0

Local work memory

This shows the contents of the local work memory used by the LLAG instruction.

The user does not need to set the contents.

However, clear must be conducted by sequence in the initial state.

	Description
u+1, u+0	E1 <sub>n-1</sub>

Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

**Processing explanation**

This operation is executed using the following o + 2 conditions.

Condition	Output value (BW)
e1=0	$BW = \frac{1}{T_1} \cdot \frac{1}{T} \cdot \{T_2 \cdot (E1 - E1_{n-1}) + T_1 \cdot BW_{\text{previous value}} + T \cdot E1\}$ However, when $T_1 + T = 0$ , $BW = 0$ .
e1=1	BW=E1 (The input value is output as is.)

T<sub>1</sub>: Lag time      T: Execution time      T<sub>2</sub>: Lead time      E1: Input data

E1<sub>n-1</sub>: Input passed value data

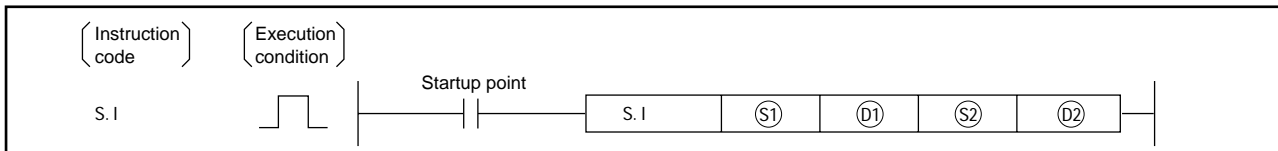
**Error**

When an overflow occurs during an operation.

(Error code: 4100)

8.2.4 Integral

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b			—		—	—	
t	—	b			—		—	—	
p	—	b			—		—	—	
u	—	b			—		—	—	

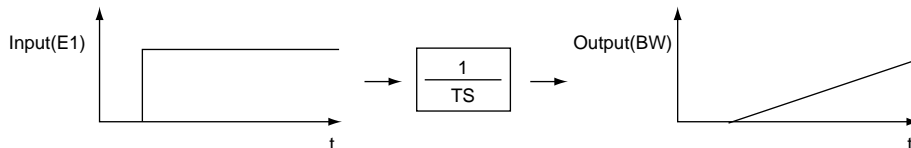


Set data

Setting data	Description	Data format
o	Input block header address	Device name
t	Block memory header device	Real number
p	Operation constant header device	Device name
u	Dummy device	Dummy

Function

The integral operation is executed by the action control signal (e1).

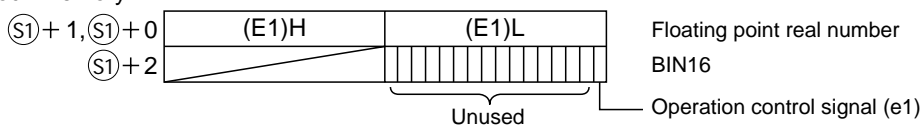


Data handling

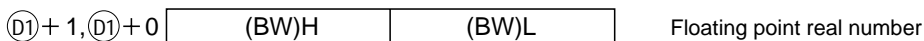
Input data

Stores the input value (E1) in o.  
o + 2's 1 through 15 bits are not used.

Block memory



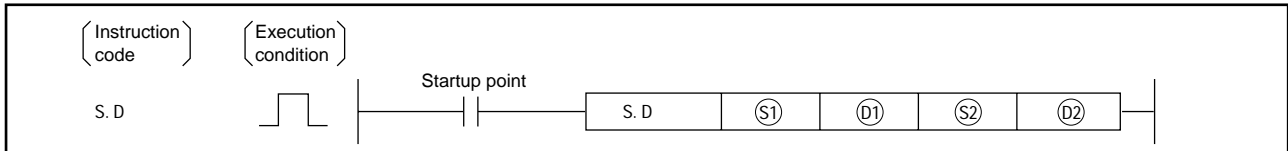
The BW (Output value) is stored in t.  
BB is not used.





**8.2.5 Derivative**

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b	—	—	—	—	—	—	
t	—	b	—	—	—	—	—	—	
p	—	b	—	—	—	—	—	—	
u	—	b	—	—	—	—	—	—	

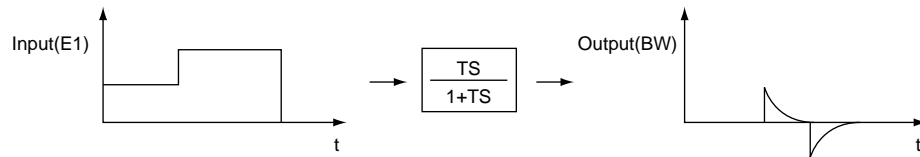


**Set data**

Setting data	Description	Data format
o	Input block header address	Device name
t	Block memory header device	Real number
p	Operation constant header device	Device name
u	Local work memory header device	Device name

**Function**

The derivative operation is executed by the action control signal (e1).

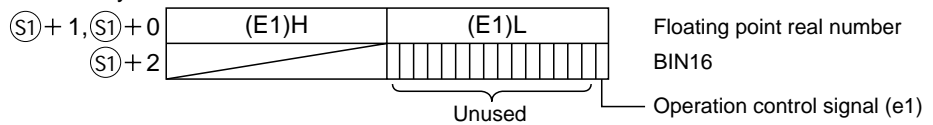


**Data handling**

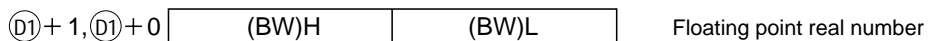
Input data

Stores the input value (E1) in o.  
o + 2's 1 through 15 bits are not used.

Block memory



The BW (Output value) is stored in t.  
BB is not used.



Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Derivative time (sec)	T	0 to 999999	1.0
p+3, p+2	Output initial value	Y <sub>S</sub>	-999999 to 999999	0.0

Local work memory

This shows the contents of the local work memory used by the D instruction.

The user does not need to set the contents.

However, clear must be conducted by sequence in the initial state.

	Description
u+1, u+0	Previous input value (E1')

Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

**Processing explanation**

This operation is executed using the following o + 2 conditions.

e1	Calculation results (BW)
0	$BW = \frac{T}{T} \frac{T}{T} (Y_{n-1} E1' + E1)$ However, when T + T = 0, BW = 0.
1	BW=Y <sub>S</sub>

E1: This input value      T: Execution time      E1': Previous output value

T: Derivative constant      Y<sub>n-1</sub>: Previous output value

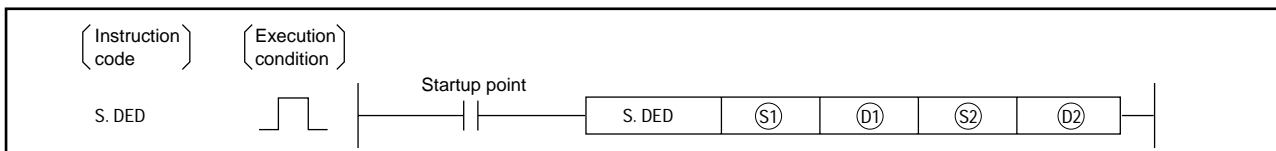
**Error**

When an overflow occurs during an operation.

(Error code: 4100)

**8.2.6 Dead Time**

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b			—		—	—	
t	—	b			—		—	—	
p	—	b			—		—	—	
u	—	b			—		—	—	

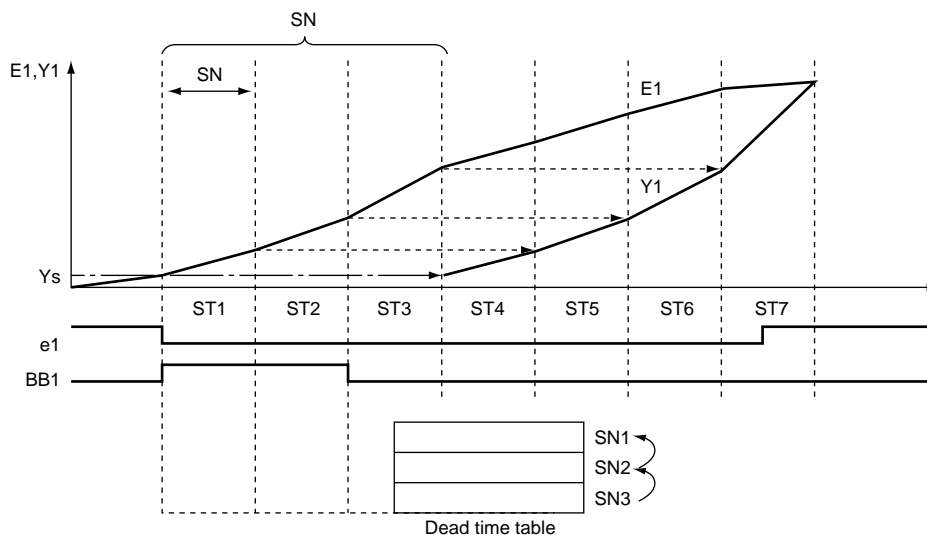


**Set data**

Setting data	Description	Data format
o	Input block header address	Device name
t	Block memory header device	Device name
p	Operation constant header device	Device name
u	Local work memory header device	Device name

**Function**

The input data is only delayed by the dead time by using the operation control signal code (E1) contents.



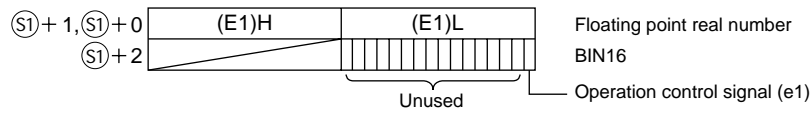
SN: Number of samples      E1: Input value  
 ST: Data collection interval      Ys: Output initial value

**Data handling**

Input data

Stores the input value (E1) in o.

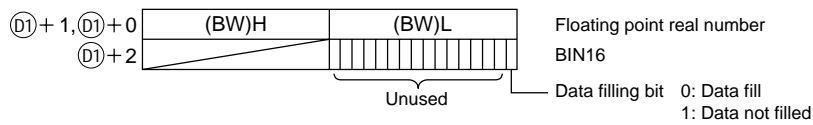
o + 2's 1 through 15 bits are not used.



Block memory

The BW (Output value) is stored in t.

t+2's BB2 to BB16 are not used.



Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Data collection interval (sec)	ST	0 to 999999	1.0
p+2	Number of samples (n)	SN	0 to 48	0
p+4, p+3	Output initial value	Y <sub>S</sub>	-999999 to 999999	0.0
p+5	Output switch during initial	0CHG	0.1	0

Local work memory

This shows the contents of the local work memory used by u.

The user does not need to set the contents.

However, clear must be conducted by sequence in the initial state.

	Description
u+0	Previous value input (e1')
u+1	Period counter
u+2	Dead time table number of storage tables
u+4, u+3	Dead time table 1
u+6, u+5	Dead time table 2
⋮	⋮
⋮	⋮
u+2 <sub>n+4</sub> , u+2 <sub>n+3</sub>	Dead time table 48

Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

**Processing explanation**

This operation is executed using the following o + 2 conditions.

e1	0CHG	Dead time	Calculation results (BW)	
0	Free	None	E1	
1 0	0	ST SN	Until the SNth time	E1 when e1 is changed from 1 to 0.
			After the SNth time	Oldest data*
	1		Until the SNth time	Ys
			After the SNth time	Oldest data*
0 0	Free	ST SN	Oldest data*	

Note 1:

When the dead time table data is not filled, BB1 is turned on.

Note 2:

An error occurs when the number of samples<0 or number of samples>48.

Note 3:

When SN = 0, BB1 = 0 and BW = E1.

ST: Data collection interval (sec) SN: Number of samples

e1': Previous output value Ys: Output initial value

\*: The oldest data is the E1 after the SNth time.

**Error**

When an overflow occurs during an operation.

(Error code: 4100)

When the number of samples<0 or number of samples>48

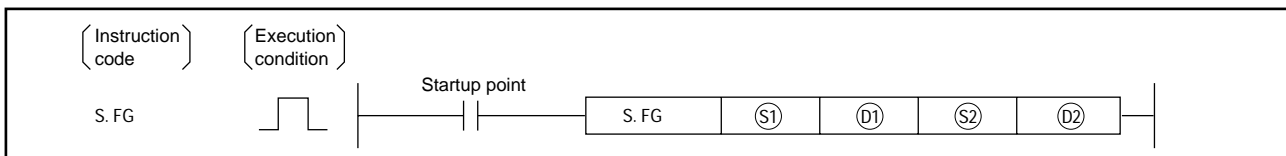
(Error code 4100)



### 8.3 Correction Operation Instruction

#### 8.3.1 Function Generator

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b					—	—	
t	—	b					—	—	
p	—	b					—	—	
u	—	b					—	—	

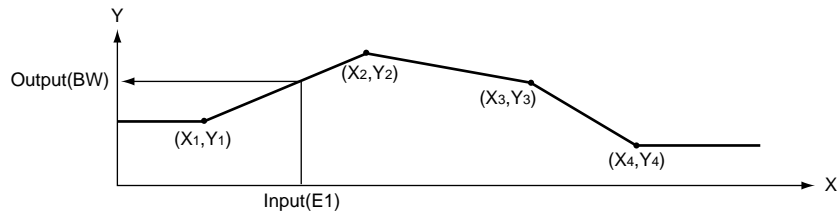


**Set data**

Setting data	Description	Data format
o	Input block header address	Real number
t	Block memory header device	Real number
p	Operation constant header device	Device name
u	Local work memory header device	Device name

**Function**

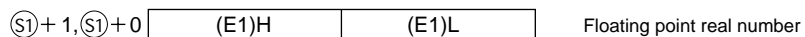
This function conducts the output for input (E1) following the broken line pattern from the nth break points specified by the operation constant.



**Data handling**

**Input data**

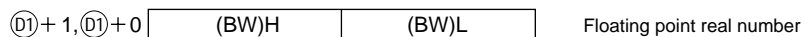
Stores the input value (E1) in o.



**Block memory**

The BW (Output value) is stored in t.

BB is not used.



Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+0	Number of broken point units (n)	SN	0 to 48	0

Local work memory

This shows the contents of the local work memory used by u.

Xn and Yn both become the respective break point coordinates, so the user needs to set them.

However, clear must be conducted by sequence in the initial state.

$(D2+1, D2+0)$	(X1)H	(X1)L	X1
$(D2+3, D2+2)$	(Y1)H	(Y1)L	Y1
$(D2+5, D2+4)$	(X2)H	(X2)L	X2
$(D2+7, D2+6)$	(Y2)H	(Y2)L	Y2
$(D2+4n-2, D2+4n-3)$	(Xn)H	(Xn)L	Xn
$(D2+4n-1, D2+4n-1)$	(Yn)H	(Yn)L	Yn

Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

### Processing explanation

This executes the following operation.

Condition	Output value (BW)
$E1 < X_1$	$BW=Y_1$
$X_{i-1} < E1$ $X_i$ (i=2 to n)	$BW = \frac{Y_i}{X_i} \frac{Y_{i-1}}{X_{i-1}} (E1 - X_{i-1}) + Y_{i-1}$
$X_n < E1$	$BW=Y_n$

However, when  $n = 0$  there is no processing.

When  $X_{i-1} > X_i$ , the value is cut off to  $n = i - 1$  (Data after that is ignored.)

When there are multiple  $Y_i$  for the same  $X_i$ , the newest  $i$  is selected.

When  $n < 0$  or  $n > 48$  an error occurs.

### Error

When an overflow occurs during an operation.

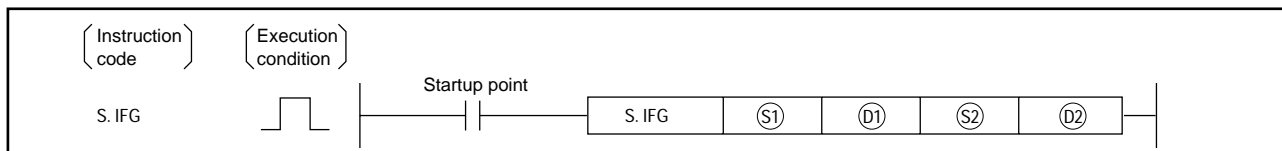
(Error code: 4100)

When  $n < 0$  or  $n > 48$

(Error code: 4100)

**8.3.2 Inverse Function Generator**

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b	—		—	—	—	—	
t	—	b	—		—	—	—	—	
p	—	b	—		—	—	—	—	
u	—	b	—		—	—	—	—	

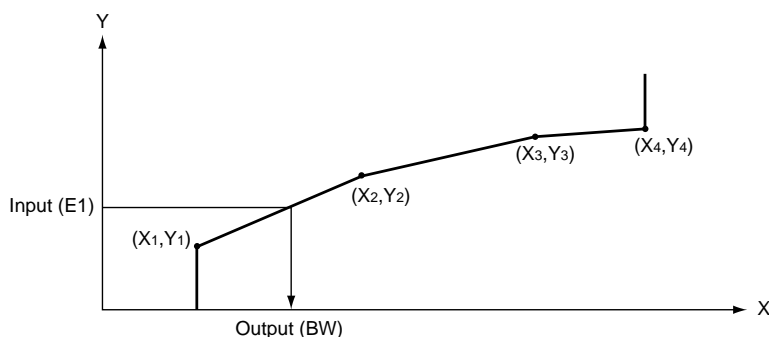


**Set data**

Setting data	Description	Data format
o	Input value (E1)	Real number
t	Block memory header device	Real number
p	Operation constant header device	Device name
u	Local work memory header device	Device name

**Function**

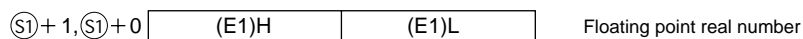
This function conducts the output for input (E1) following the broken point pattern from the nth break points specified by the operation constant.



**Data handling**

Input data

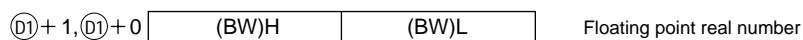
Stores the input value (E1) in o.



Block memory

The BW (Output value) is stored in t.

BB is not used.



Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+0	Number of broken point units (n)	SN	0 to 48	0

Local work memory

This shows the contents of the local work memory used by u.

Xn and Yn both become the respective break point coordinates, so the user needs to set them.

However, clear must be conducted by sequence in the initial state.

$(D2)+1, (D2)+0$	(X1)H	(X1)L	X1
$(D2)+3, (D2)+2$	(Y1)H	(Y1)L	Y1
$(D2)+5, (D2)+4$	(X2)H	(X2)L	X2
$(D2)+7, (D2)+6$	(Y2)H	(Y2)L	Y2
$(D2)+4n-2, (D2)+4n-3$	(Xn)H	(Xn)L	Xn
$(D2)+4n, (D2)+4n-1$	(Yn)H	(Yn)L	Yn

Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

**Processing explanation**

This executes the following operation.

Condition	Output value (BW)
$E1 < Y_1$	$BW=X1$
$Y_{i-1} < E1 < Y_i$ (i=2 to n)	$BW = \frac{X_i - X_{i-1}}{Y_i - Y_{i-1}} (E1 - Y_{i-1}) + X_{i-1}$
$Y_n < E1$	$BW=Xn$

However, when n = 0 there is no processing.

When  $Y_{i-1} > Y_i$ , the value is cut off to n = i - 1 (Data after that is ignored.)

When there are multiple Xi for the same Yi, the newest i is selected.

When n<0 or n>48 an error occurs.

**Error**

When an overflow occurs during an operation.

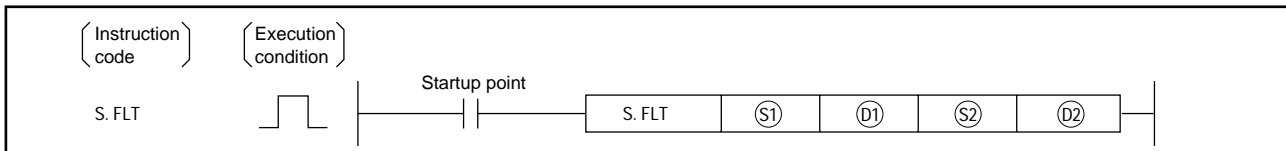
(Error code: 4100)

When n<0 or n>48

(Error code: 4100)

8.3.3 Standard Filter

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b			—		—	—	
t	—	b			—		—	—	
p	—	b			—		—	—	
u	—	b			—		—	—	



Set data

Setting data	Description	Data format
o	Input value (E1)	Real number
t	Block memory header device	Device name
p	Operation constant header device	Device name
u	Local work memory header device	Device name

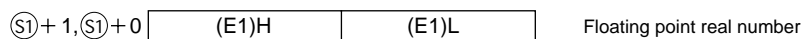
Function

The sampled SN units of data are stored in the dead time table at the table collection interval (ST), those SN units data are averaged, and output.

Data handling

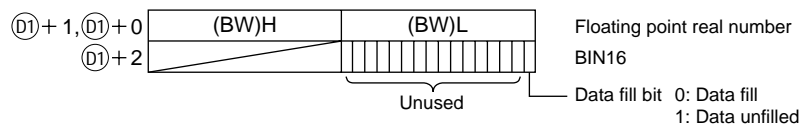
Input data

Stores the input value (E1) in o.



Block memory

The BW (Output value) is stored in t. t+2's BB2 to BB16 are not used.



Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Data collection interval	ST	0 to 999999	1.0
p+2	Number of samples	SN	0 to 48	0

Local work memory

This shows the contents of the local work memory used by FLT.

The user does not need to set the contents.

However, clear must be conducted by sequence in the initial state.

$(D2)+1, (D2)+0$	(ST)H	(ST)L	Previous data collection interval (ST)
$(D2)+2$		(SN)	Previous number of samples (SN)
$(D2)+3$		(i)	Period counter (i)
$(D2)+4$		(n1)	Storage data number (n1)
$(D2)+5$		(n2)	Storage address (n2)
$(D2)+7, (D2)+4n+6$			
$(D2)+9, (D2)+4n+8$	(1)H	(1)L	Dead time table 1
$(D2)+11, (D2)+4n+10$	(2)H	(2)L	Dead time table 2
$(D2)+2n+7, (D2)+2n+6$	(SN)H	(SN)L	Dead time table SN

Execution time ( T )

Set the execution time in SD1500 and SD1501. (Refer to Section 3.)

**Processing explanation**

This executes the following operation.

- 1 The data update time is  $\frac{ST}{T}$  (The decimal is rounded down.)
- 2 BB1 becomes 0 when the data buffer SN units of data is filled.

In addition, it becomes 1 when it is not filled.

Note 1:

When SN is 0, BW and BB are cleared and ended.

Note 2:

When  $SN < 0$  or  $SN > 48$  an error occurs.

Note 3:

The data portion is averaged and output until the data buffer is filled by data.

Note 4:

Processed using  $ST = n \times T$  (n is an integral).

**Error**

When an overflow occurs during an operation.

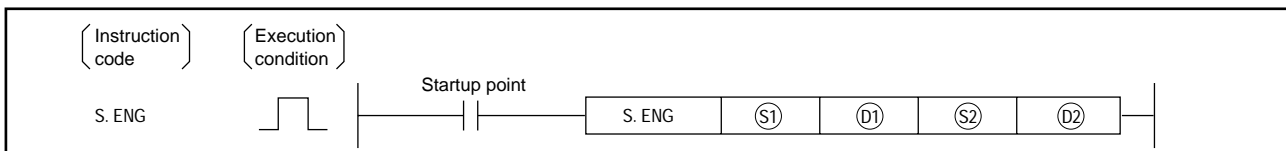
(Error code: 4100)

When  $SN < 0$  or  $SN > 48$

(Error code: 4100)

**8.3.4 Engineering Value Conversion**

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b	—	—	—	—	—	—	
t	—	b	—	—	—	—	—	—	
p	—	b	—	—	—	—	—	—	
u	—	b	—	—	—	—	—	—	

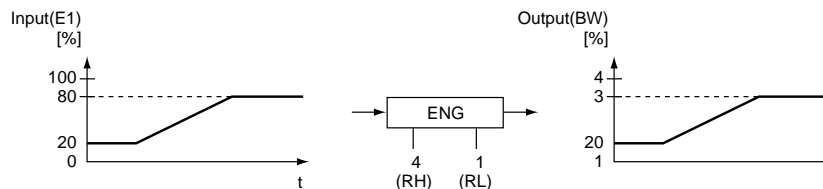


**Set data**

Setting data	Description	Data format
o	Input value (E1)	Real number
t	Block memory header device	Real number
p	Operation constant header device	Device name
u	Dummy device	Dummy

**Function**

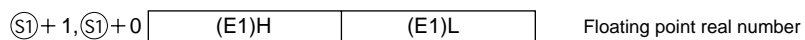
The input value (E1) is output by the engineering value conversion.



**Data handling**

**Input data**

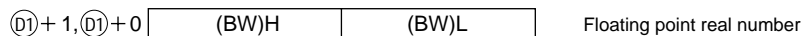
Stores the input value (E1) in o.



**Block memory**

The BW (Output value) is stored in t.

BB is not used.



Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Engineering value upper limit	RH	-999999 to 999999	100.0
p+3, p+2	Engineering value lower limit	RL	-999999 to 999999	0.0

#### Processing explanation

This executes the following operation.

Engineering value conversion

$$BW \frac{RH - RL}{100} (E1 \quad RL) (E1=0 \text{ to } 100\%)$$

RH: Engineering value upper limit      RL: Engineering value lower limit

#### Error

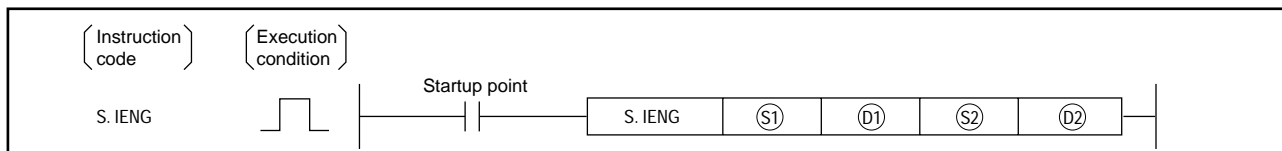
When an overflow occurs during an operation.

(Error code: 4100)



**8.3.5 Inverse Engineering Value Conversion**

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b	—		—	—	—	—	
t	—	b	—		—	—	—	—	
p	—	b	—		—	—	—	—	
u	—	b	—		—	—	—	—	

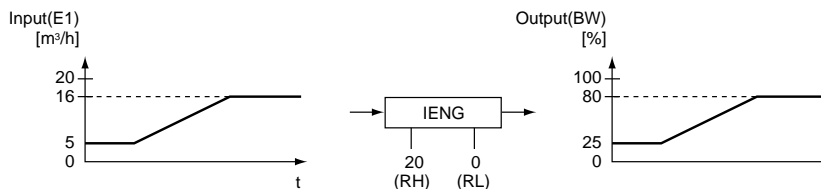


**Set data**

Setting data	Description	Data format
o	Input value (E1)	Real number
t	Block memory header device	Real number
p	Operation constant header device	Device name
u	Dummy device	Dummy

**Function**

The input value (E1) is converted to % value and output.



**Data handling**

Input data

Stores the input value (E1) in o.

(S1)+1, (S1)+0 | (E1)H | (E1)L | Floating point real number

Block memory

The BW (Output value) is stored in t.

BB is not used.

(D1)+1, (D1)+0 | (BW)H | (BW)L | Floating point real number

Operation constant

This shows the contents of the operation constant set in p.

	Item name	Item	Settable range	Standard value setting
p+1, p+0	Engineering value upper limit	RH	-999999 to 999999	100.0
p+3, p+2	Engineering value lower limit	RL	-999999 to 999999	0.0

#### Processing explanation

This executes the following operation.

Inverse engineering value conversion

$$BW = \frac{100}{RH - RL} (E1 - RL) (\%)$$

RH: Engineering value upper limit      RL: Engineering value lower limit

Set so that RH>RL.

Even if RH = RL, continue processing as is, but engineering value reverse conversion will not be conducted.

When RH=RL, BW=0.

#### Error

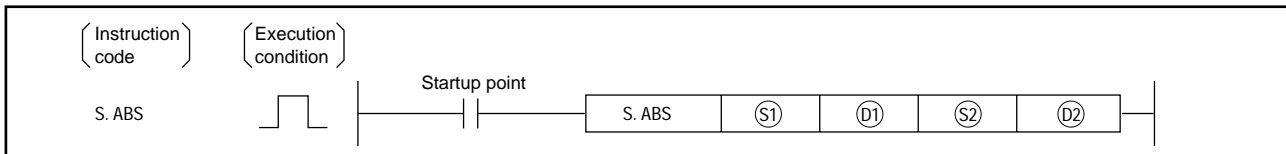
When an overflow occurs during an operation.

(Error code: 4100)

## 8.4 Arithmetic Operation

### 8.4.1 Absolute Value

Setting data	Usable Devices								
	Internal devices (System, user)		File register	MELSECNET/10 direct Jc/c		Special function module Uc/Gc	Index register Zn	Constant	Other
	Bit	Word		Bit	Word				
o	—	b			—		—	—	
t	—	b			—		—	—	
p	—	b			—		—	—	
u	—	b			—		—	—	

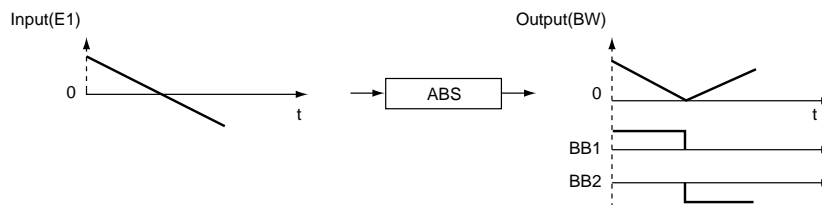


#### Set data

Setting data	Description	Data format
o	Input value (E1)	Real number
t	Block memory header device	Device name
p	Dummy device	Dummy
u	Dummy device	Dummy

#### Function

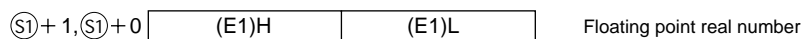
The input value (E1) absolute value is output.



#### Data handling

Input data

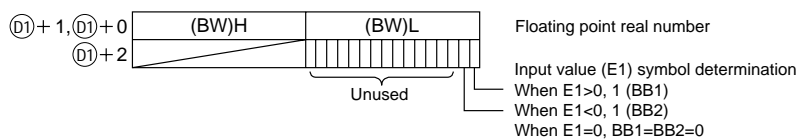
Stores the input value (E1) in o.



Block memory

The BW (Output value) is stored in t.

t+2's BB3 to BB16 are not used.



**Processing explanation**

This executes the following operation.

$$\boxed{BW=|E1|}$$

At the same time this determines the input value (E1) symbol and outputs it to BB1 and BB2.

When  $E1 > 0$ ,  $BB1=1$ ,  $BB2=0$

When  $E1 < 0$ ,  $BB1=0$ ,  $BB2=1$

When  $E1=0$ ,  $BB1=BB2=0$

**Error**

When an overflow occurs during an operation.

(Error code: 4100)

# Error Code

This section explains the contents and countermeasures for errors generated by the Q4ARCPU.

## 9.1 How to Read Error Codes

---

When an error occurs, the error code or error message can be read using the GPP function peripheral equipment.

For details regarding the GPP function peripheral equipment operation method, refer to the Peripheral Equipment SWaIVD-GPPQ Operating Manual (Online Edition).

## 9.2 Error Code List

The application PID instruction errors are as follows.

**Error occurs during an operation**

**(Error No.: 4100)**

**CPU internal error**

**(Error No.: 1206)**

In addition, when an operation error occurs the detail information is stored in SD1502 to SD1503.

(At times other than when an application PID instruction function operation error occurs SD1502 is 0.)

SD1502: This stores the error code that occurs for the application PID instruction function.

(Refer to Table 1)

SD1503: This stores instruction processing Nos. 1 to 8 for when an error occurred.

(Refer to Table 2)

**Table 1 Error codes that occur in a process control instruction function**

Error code	Error description	Cause
1	There is either a non-numeric or non-correct number.	There is a problem with the set data such as the operation constant, loop tag memory, loop tag history value memory, or execution time. (The set data must be checked.)
2	Symbol error (The number is negative)	
3	Number error (The number is outside the range).	
4	Integer range is exceeded	
5	Tried to divide by 0.	
6	An overflow occurred.	
16	DSP hardware error	CPU internal error
17	Command code error	
18	Data error	
19	Time-out	

**Table 2 Instruction processing Nos. for which an error occurred**

When the following instruction errors occur the process No. becomes 1.

Process No. / Instruction	Process							
	1	2	3	4	5	6	7	8
S. IN	Range check	Input limiter	Engineering value conversion	Digital filter				
S. OUT1	Input addition processing	Change rate upper and lower limit limiter	Reset windup	Output conversion				
S. PID	Execution time monitoring	SV setting processing	Trucking processing	Gain Kp operation	PID operation	Deviation check		
S. PHPL	Engineering value reverse conversion	Upper and lower limit check	Change rate check	Engineering value conversion	Loop stop			

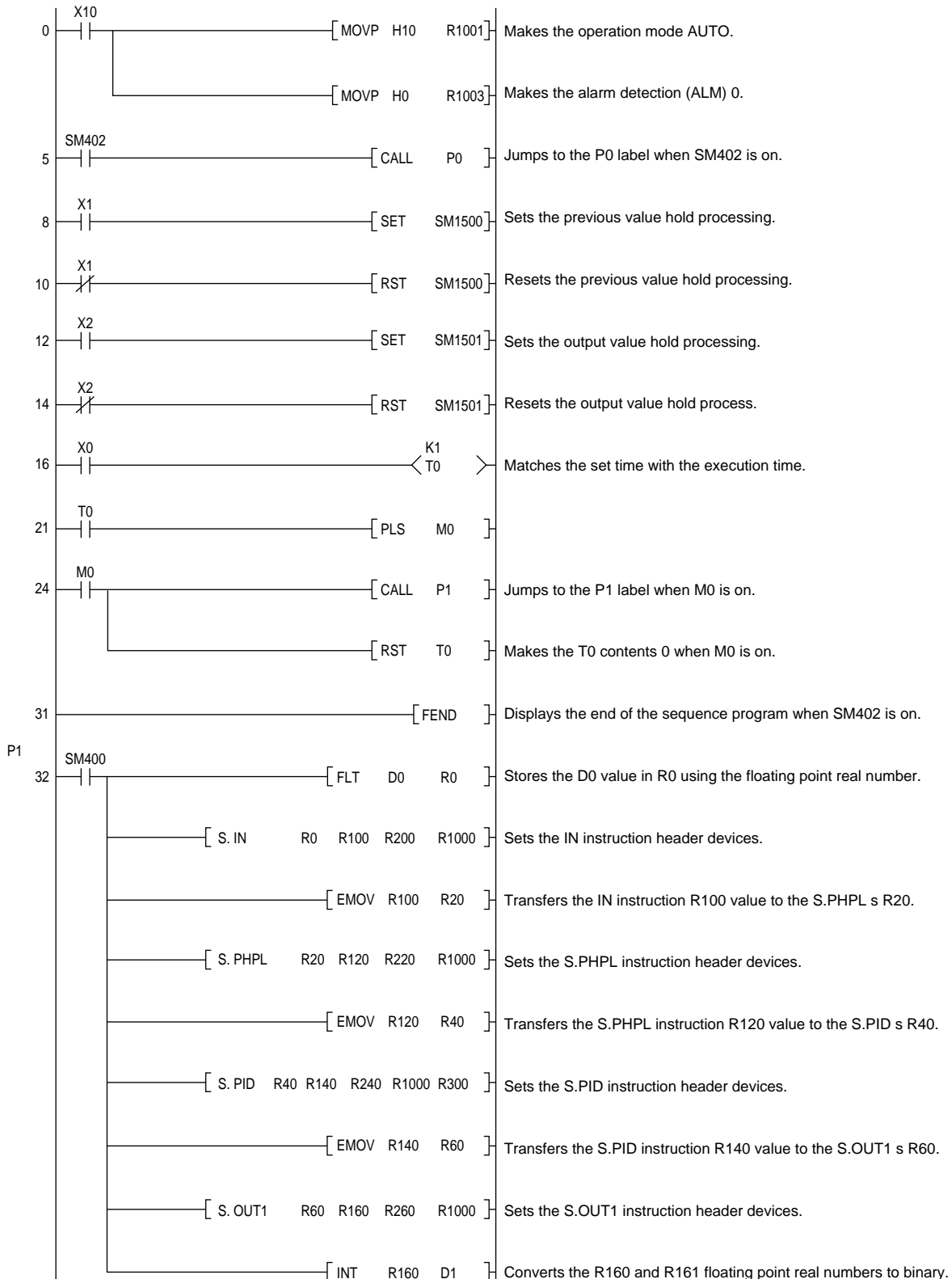
**Point**

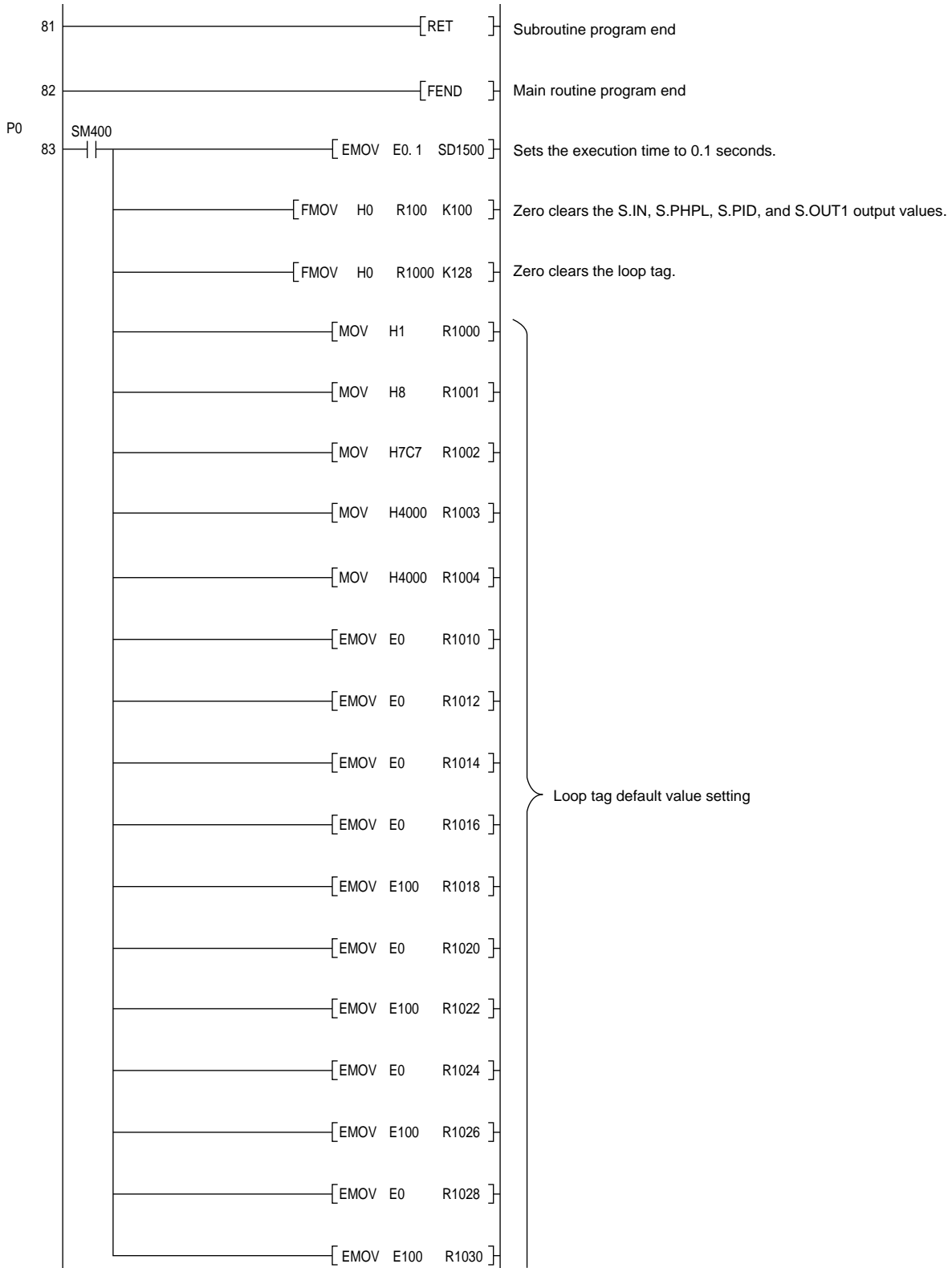
When an error other than the application PID instruction has occurred, refer to the QnACPU Programming Manual (Common Instruction Edition).

# Appendix

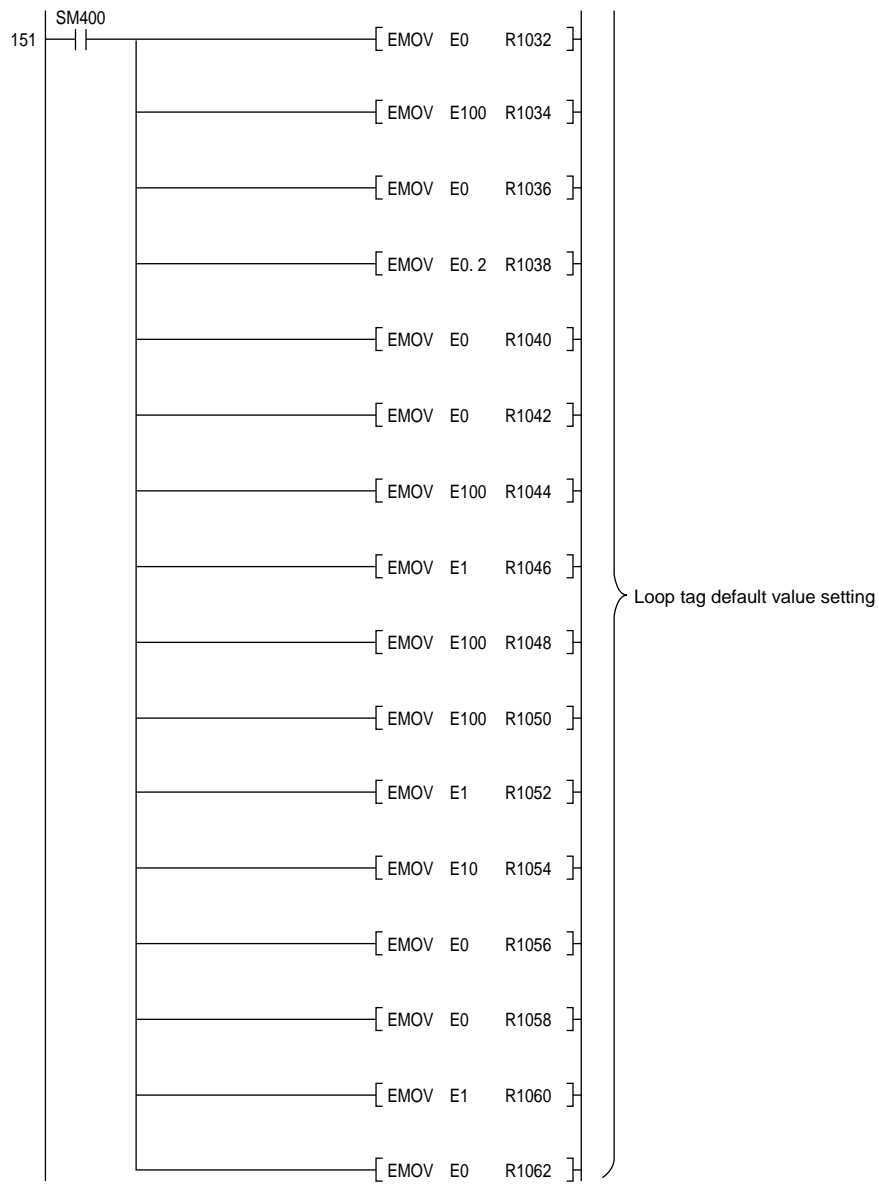
## Appendix 1 Example Program

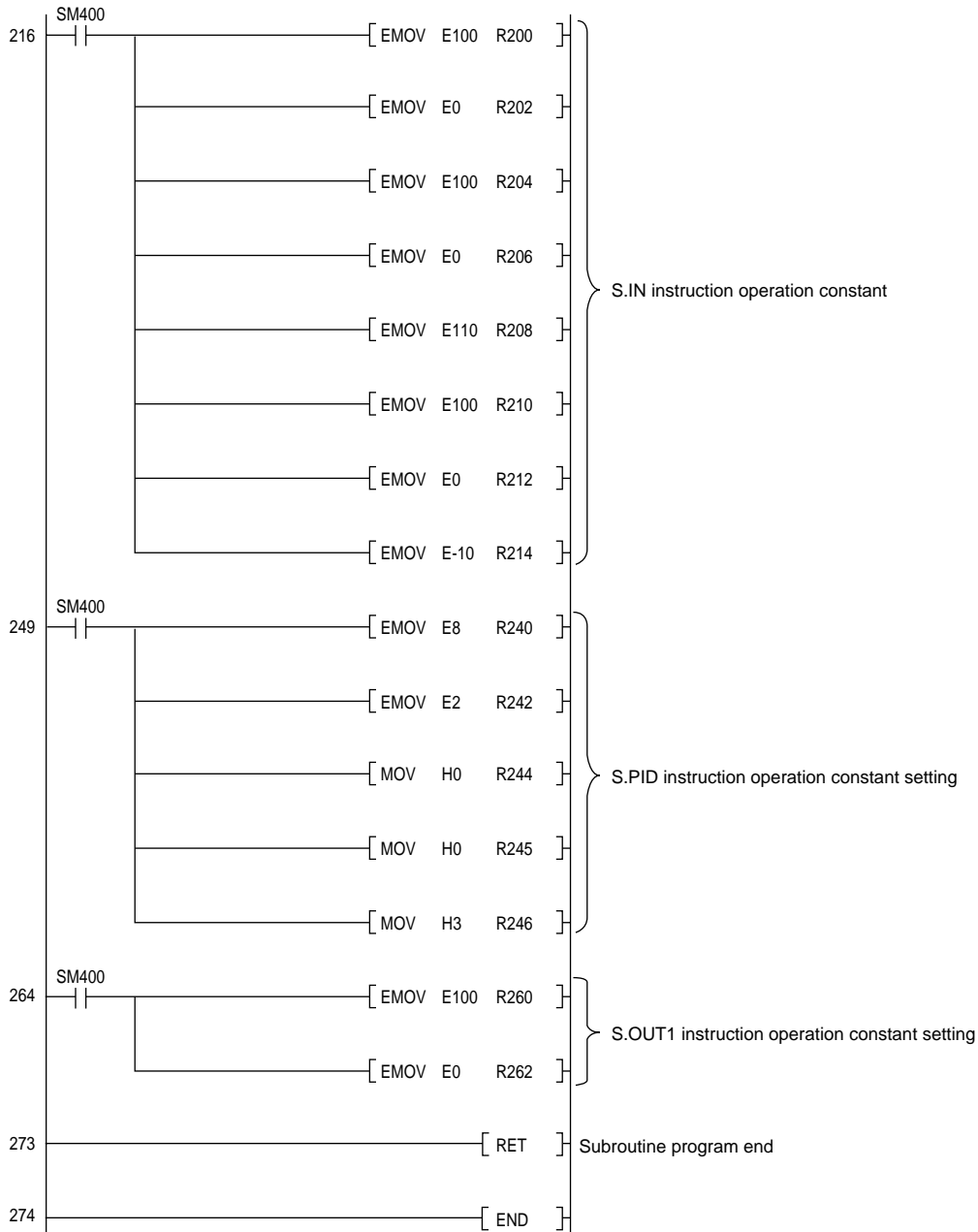
This section explains a program that conducts PID control in the Q4ARCPU.











## Appendix 2 Operation Processing Time

This section shows an example processing time for when an actual number is entered into the instruction operation constant and loop tag memory.

Loop type: .....SPID

Instruction used: .....S.IN, S.PHPL, S.PID, S.OUT1

## IN instruction operation constant

Item name	Item	Number
Engineering value conversion upper limit	EMAX	E100
Engineering value conversion lower limit	EMIN	E0
Input upper limit	NMAX	E100
Input lower limit	NMIN	E0
Upper limit side range error occurrence	HH	E95
Upper limit side range error return	H	E80
Lower limit side range error return	L	E20
Lower limit side range error occurrence	LL	E5

No operation constant for PHPL.

## PID instruction operation constant

Item name	Item	Number
Derivative gain	MTD	E4
Deviation high alarm hysteresis	DVLS	E3
Reverse action, forward action	PN	K0
Trucking bit	TRK	K0
Set value pattern	SVPTN	K3

## S.OUT1 instruction operation constant

Item name	Item	Number
Input upper limit	NMAX	E100
Input lower limit	NMIN	E0

## Loop tag memory

Offset	Item	Lower limit	Upper limit	Number
+0	FUNC	0	15	
+1	MODE	0	HFFFF	H10
+2	MDIH	0	HFFFF	
+3	ALM	0	HFFFF	H0
+4	INH	0	HFFFF	H0
+5	ALML	0	HFFFF	
+6	CTNO	0	32	
+7	CTFN	0	HFFFF	
+8	UNIT	0	127	
+9	N	0	4	
+10	PV	RL* (RH*)	RL* (RH*)	E0
+12	MV	-10	110	E0
+14	SV	RL* (RH*)	RL* (RH*)	E55
+16	DV	-110	110	E7
+18	MH	-10	110	E100
+20	ML	-10	110	E0
+22	RH	-999999	999999	E100
+24	RL	-999999	999999	E0
+26	PH	RL* (RH*)	RL* (RH*)	E80
+28	PL	RL* (RH*)	RL* (RH*)	E20
+30	HH	RL* (RH*)	RL* (RH*)	E90
+32	LL	RL* (RH*)	RL* (RH*)	E10
+34	SH	RL* (RH*)	RL* (RH*)	
+36	SL	RL* (RH*)	RL* (RH*)	
+38		0	1	E0
+40	HS	0	999999	E3
+42	CTIM	0	999999	E8
+44	DPL	0	100	E30
+46	CT	0	999999	E1
+48	DML	0	100	E100
+50	DVL	0	100	E25
+52	P	0	999999	E3
+54	I	0	999999	E8
+56	D	0	999999	E5
+58	GW	0	100	E15
+60	GG	0	999999	E2
+62	MVP	FMIN	FMAX	E0.25

## Instruction processing time

S. IN .....204.38 s

S. PHPL .....437.79 s

S. PID.....466.89 s

S. OUT1 .....227.94 s

The SPID loop type processing time is 1337 s.

## Index

- [A]
  - Arithmetic Operation Instruction  
5-3
- [B]
  - Bit Data  
6-1
  - Block Data  
2-4
  - Block Memory  
2-4
  - Bumpless Function  
2-9
- [C]
  - Cascade Loop Trucking  
2-9
  - Control Operation Instruction  
5-2
  - Correction Operation Instruction  
5-2
- [D]
  - Data Configuration  
2-1
  - Derivative Operation (D)  
4-5
- [E]
  - Execution Time and Operation Time  
3-1
  - Execution Time and Control Time Data Storage Location  
3-2
- [F]
  - Forward Action and Reverse Action  
4-2
- [H]
  - How to View the Instruction  
5-1
- [I]
  - I/O Control Instruction  
5-2
  - Index Qualification  
6-3
  - Instruction Configuration  
6-1
  - Instruction Execution Conditions  
6-3
  - Integral Action (I)  
4-4
- [L]
  - Local Work Memory  
2-3
  - Loop Memory  
2-5
  - Loop Tag Past value Memory  
2-5
  - Loop RUN/STOP  
2-8
  - Loop Tag Memory Allocation Contents  
2-6
- [M]
  - Micro Block  
2-1
- [N]
  - Number of Steps  
6-3
- [O]
  - Offset  
4-3
  - Operation Constant  
2-4
  - Output Limiter Processing Function  
2-9
- [P]
  - PID control System Overview  
4-1
  - PID Operation  
4-6
  - Process Control Function Operation Error  
6-3
  - Process Control Instruction Configuration  
2-4
  - Process Value Derivative Type  
4-2
  - Proportional Action (P)  
4-3
- [S]
  - S.ABS  
8-39
  - S.D  
8-24

S.DED  
8-26  
S.ENG  
8-35  
S.FG  
8-29  
S.FLT  
8-33  
S.I  
8-22  
S.IENG  
8-37  
S.IFG  
8-31  
S.IN  
8-1  
S.LLAG  
8-20  
S.OUT1  
8-5

S.PHPL  
8-15  
S.PID  
8-9

[T]  
Time and Operation  
3-1  
Trucking Function  
2-9

[V]  
Velocity Type Incomplete Differential Format  
1-2

[W]  
Word Data  
6-2