

Chapter 4. CPU MODULE

4.1 Performance specifications

The following shows the general specifications of the GLOFA-GM series.

Items			Specifications			Remarks
			GM6-CPUA	GM6-CPUB	GM6-CPUC	
Operation method			Cyclic operation of stored program, Interrupt task operation			
I/O control method			Scan synchronized batch processing method(Refresh method)			
Programming language			Ladder Diagram(LD) Instruction List(IL) Sequential Function Chart(SFC)			
Number of instructions	Operator		LD : 13, IL : 21			
	Basic function		194			
	Basic function block		11			
	Special function block		Each special module have their own special function blocks			
Processing speed	Operator		Refer to Appendix 3.			
	Basic function					
	Basic function block					
Programming memory capacity			68 k bytes(17 k steps)			
I/O points			256 points			
Data memory	Direct variable area		2 to 8 k bytes			
	Symbolic variable area		30 k bytes – Direct variable area			
Timer			No limitations in points. Time range : 0.01 to 4294967.29 sec(1193 hours)			1 point occupies 20 bytes of symbolic variable area.
Counter			No limitations in points Counting range: -32768 to +32767			1 point occupies 8 bytes of symbolic variable area.
Program types	Numbers of program blocks		100			
	Initialization programs		1 (_INIT)			
	Task Programs	Time driven tasks	0 ~ 8			Total : 8 (The type of task is variable, however, total numbers of tasks is 8.)
		External interrupt tasks	0 ~ 8			
		Internal task	0 ~ 8			
Operation modes			RUN, STOP, PAUSE and DEBUG			
Restart modes			Cold, Warm			
Self-diagnostic functions			Watch dog timer, Memory error detection, I/O error detection, Battery error detection, Power supply error detection, etc.			
Data protection method at power failure			Set to 'Retain' variables at data declaration.			
Built-in special functions			RS-232C	RS-422/485 RTC PID control	RS-232C RTC PID control High Speed Counter	
Internal current consumption			170mA	210mA	170mA	
Weight			0.11Kg	0.11 Kg	0.12Kg	

4.2 Operation Processing

4.2.1 Operation Processing Method

1) Cyclic operation

A PLC program is sequentially executed from the first step to the last step, which is called scan.

This sequential processing is called cyclic operation. Cyclic operation of the PLC continues as long as conditions do not change for interrupt processing during program execution.

This processing is classified into the following stages.

Stages	Processing
Operation Start	-
Initialization	<ul style="list-style-type: none"> Stage for the start of a scan processing. it is executed only one time when the power is applied or reset is executed. It executes the following processing. <ul style="list-style-type: none"> I/O modules reset Execution of self-diagnosis Data clear I/O module address allocation or type registration
Input image area refresh	<ul style="list-style-type: none"> Input module conditions are read and stored into the input image area before operation processing of a program.
Program operation processing Program start ~ Program end	<ul style="list-style-type: none"> Program is sequentially executed from the first step to the last step
Output image area refresh	<ul style="list-style-type: none"> The contents stored in the output image area is output to output modules when operation processing of a program is finished.
END processing	<ul style="list-style-type: none"> Stage for return processing after the CPU module has finished 1 scan. The following processing are executed. <ul style="list-style-type: none"> Self-diagnosis Change of the present values of timer and counter, etc. Processing data communications between computer link module and communications module. Checking the switch for mode setting.

2) Time driven interrupt operation method

In time driven interrupt operation method, operations are processed not repeatedly but at every pre-set interval. Interval, in the GM6 CPU module, can be set to between 0.01 to 4294967.29 sec. This operation is used to process operation with a constant cycle.

3) Event driven interrupt operation method

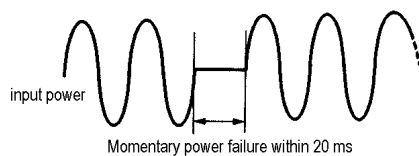
If a situation occurs which is requested to be urgently processed during execution of a PLC program, this operation method processes immediately the operation which corresponds to interrupt program. The signal which informs the CPU module of those urgent conditions is called interrupt signal. The GM6 CPU module has two kind of interrupt operation methods, which are internal and external interrupt signal methods.

4.2.2 Operation processing at momentary power failure occurrence

The CPU module detects any momentary power failure when the input line voltage to the power supply module falls down below the defined value.

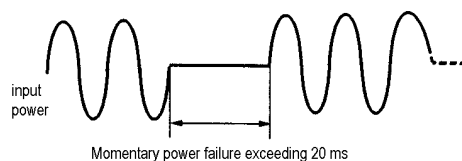
When the CPU module detects any momentary power failure, the following operations will be executed.

1) Momentary power failure within 20 ms



- (1) The operation processing is stopped with the output retained.
- (2) The operation processing is resumed when normal status is restored.
- (3) The output voltage of the power supply module retains the defined value.
- (4) The watch dog timer(WDT) keeps timing and interrupt timing normally while the operations is at a stop.

2) Momentary power failure exceeding 20 ms



- The re-start processing is executed as the power is applied.

REMARK

1) Momentary power failure

The PLC defining power failure is a state that the voltage of power has been lowered outside the allowable variation range of it. The momentary power failure is a power failure of short interval(several to tens ms).

4.2.3 Scan Time

The processing time from a 0 step to the next 0 step is called scan time.

1) Expression for scan time

Scan time is the addition value of the processing time of scan program that the user has written, of the task program processing time and the PLC internal processing time.

(1) Scan time = Scan program processing time + Task program processing time + PLC internal processing time

- Scan program processing time = The processing time used to process a user program that is not specified to a task program.
- Task program processing time = Total of the processing times of task programs executed during one scan.
- PLC internal processing time = Self-diagnosis time + I/O refresh time + Internal data processing time + Communications service processing time

(2) Scan time differs in accordance with the execution or non-execution of task programs and communications processing, etc.

2) Flag

(1) Scan time is stored in the following system flag area.

- `_SCAN_MAX` : Maximum scan time (unit : 1 ms)
- `_SCAN_MIN` : Minimum scan time (unit : 1 ms)
- `_SCAN_CUR` : Current scan time (unit : 1 ms)

4.2.4 Scan Watchdog Timer

1) Watchdog timer is used to detect a delay of abnormal operation of sequence program.

(Watchdog time is set in menu of basic parameter of GMWIN.)

2) When watchdog timer detects an exceeding of preset watchdog time, the operation of PLC is stopped immediately and all output is off.

3) If an exceeding of preset watchdog time is expected in sequence program, use 'WDT_RST' function.

'WDT_RST' function make elapsed watchdog time as zero.

4) In order to clear watchdog error, using manual reset switch, restarting the PLC and mode change to STOP mode are available.

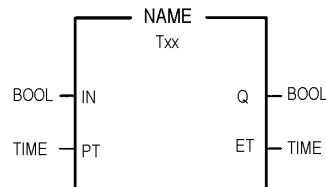
REMARK

Setting range of watchdog : 1 ~ 65,535ms(1ms base)

4.2.5 Timer Processing

The CPU module timer is an incremental timer which increases its present value according to the measuring time. Three types of On Delay Timer (TON), Off Delay Timer (TOF) and Pulse Timer (TP) are available.

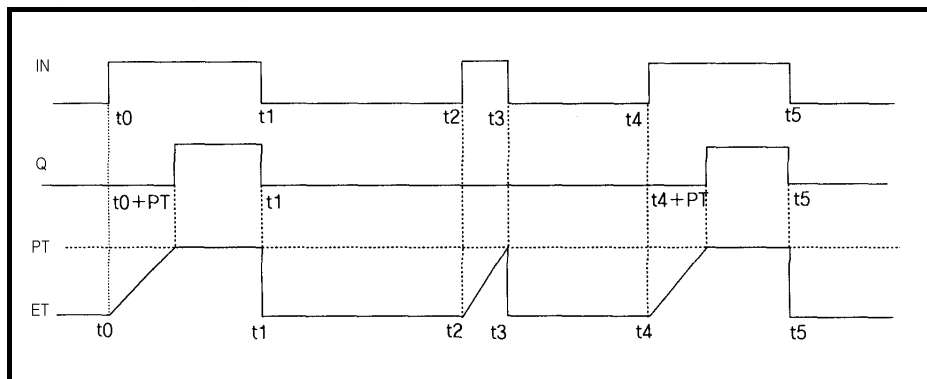
Its measuring range is 0.001 to 4,294,967,295 sec (1,193 hours) by 1 ms. For details, refer to 'GLOFA-GM Programming'.



1) On Delay Timer Process Time Change and Contact On/Off

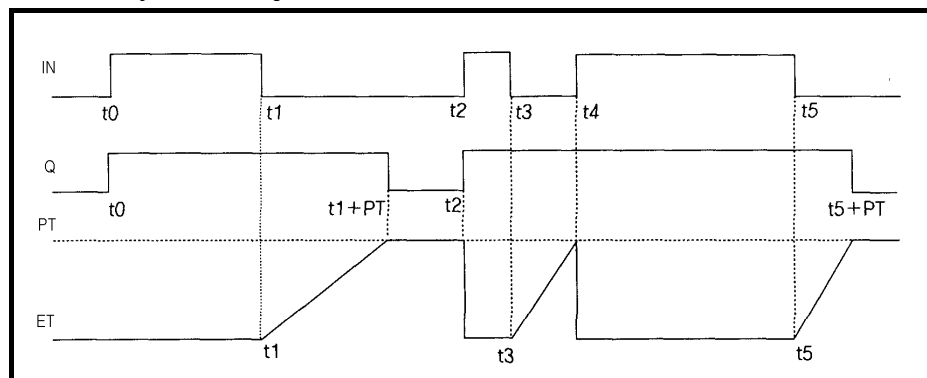
Timer Process time is newly changed when the timer function block is executed. When the process time reaches the setting time (process time = setting time), the Timer output contact turns on.

On Delay Timer Timing Diagram is shown as below.



2) Off Delay Timer Process Time Change and Contact On/Off

- If input condition turns on, timer output contact (Q) turns on. If input condition turns off, timer process time change starts.
- The process time is newly changed when the timer function block is executed. When the process time reaches the setting time (process time = setting time), the contact (Q) turns off. The following diagram shows Off Delay Timer Timing.



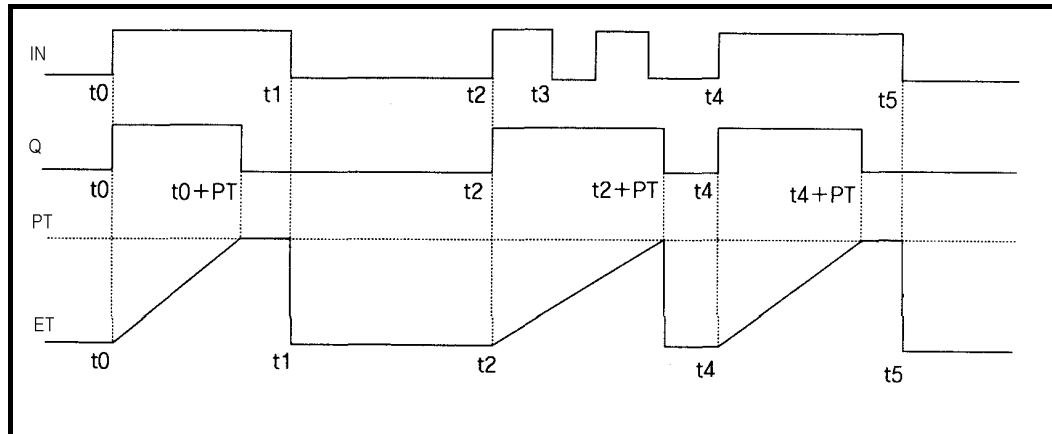
3) Pulse Timer Process Time Change and Contact On/Off

If input condition turns on, output contact (Q) turns on.

The process time is newly changed when the timer function block is executed. When the process time reaches the setting time (process time = setting time), the contact (Q) turns off.

The contact turns off after the setting time regardless of input condition off status.

The following diagram shows pulse timer timing.



4) Timer error

The maximum timer error is "1 scan time + time from the start of scan to execution of the timer function block".

4.2.6 Counter Processing

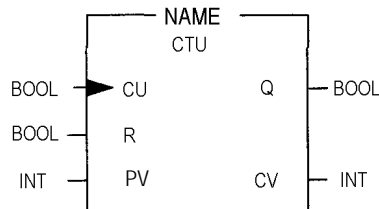
The CPU module counter increment/decrement the present counting value by the detection of rising edge(off→on) of input signal. Three types of counter are increment counter, Decrement counter and Increment-Decrement Counter. For details, refer to 'GLOFA – GM Programming'.

- The Increment counter is a counter which increment the present counting value
- The Decrement counter is a counter which decrement the present counting value
- The Increment-Decrement counter is a counter which compares the counting values of two input conditions.

1) Counter Present Value Change and Contact On/Off

(1) Increment Counter

- It should have Input condition (CU), reset condition (R) and setting value (PV).

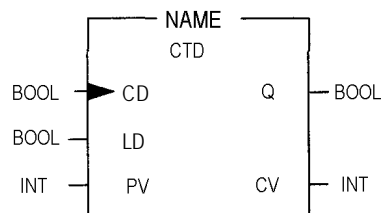


- If the counting value (CV) increments and reaches the setting value(PV) the output contact (Q) turns on.

When the reset signal is turn on, the counting value is set to '0' and the output contact (Q) turns off.

(2) Decrement Counter

- It should have inputcondition (CD), load (LD) and setting value (PV).

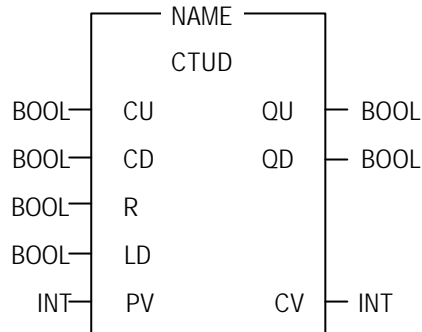


- If the counting value (CV) decrements and reaches '0', the output contact (Q) turns on.

If the load(LD) signal is turned on, the counting value is set to the setting value and the output contact (Q) turns off.

(3) Increment/Decrement Counter

- It should have Increment input condition (CU), Decrement input condition (CD), load (LD) and setting value (PV).



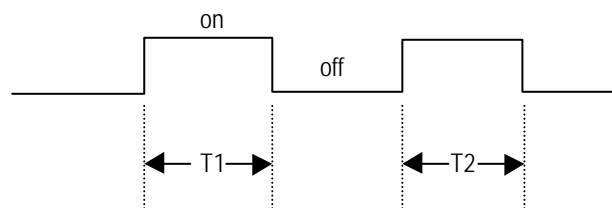
- If reset signal(R) turns on, counting value (CV) is set to '0'.
- If load signal(LD) turns on, counting value is set to setting value(PV).
- It is increased by 1 at the rising edge of increment input(CU) and decreased by 1 at the edge of decrement input(CD). If counting value(CV) is equal or larger than setting value(PV), QU will be on, and if counting value(CV) is equal or less than setting value(PV), QD will be on.

2) Counting speed

- The counting speed is decided by scan time and it will be counted when on time or off time of input condition is larger than each scan time.

$$\text{Max. Counting speed (Cmax.)} = n / 100 \times 1 / t_s \text{ [pps]} \quad [n : \text{Duty(\%)}, t_s : \text{scan time(s)}]$$

- Duty is percent of on time / off time.



$$T1 \leq T2 : n = T1 / (T1+T2) \times 100 [\%]$$

$$T1 > T2 : n = T2 / (T1+T2) \times 100 [\%]$$

4.3 Program

4.3.1 Program Configuration

A program consists of all of the function elements that is needed to execute a particular control. It is to be stored in the internal RAM of the CPU module or the flash memory of the memory module.

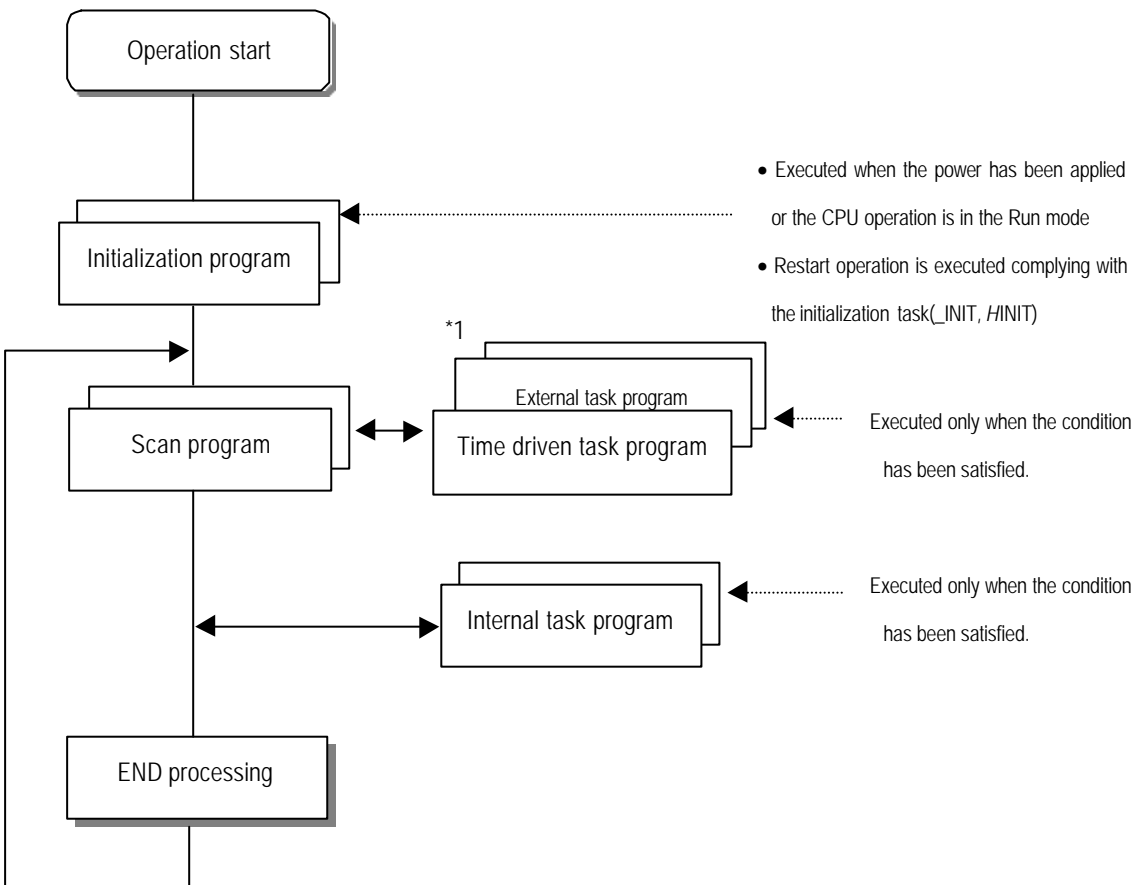
The function elements are classified as below.

Function Elements	Processing Operation
Initialization program	<ul style="list-style-type: none">• Executed when the power is applied or the CPU operation is transited to the RUN mode.• Executes the initial/fixed data setting for execution of scan program and the initialization of peripheral devices on special modules.
Scan program	<ul style="list-style-type: none">• Processes the constantly repeated signals which are executed every scan.
Time driven task program	<ul style="list-style-type: none">• When the following time conditional processing is required the program is executed complying with the time interval setting.<ul style="list-style-type: none">▶ In case that the processing need a shorter interval than that of average one scan processing time.▶ In case that the processing need a longer interval than that of average one scan processing time.▶ In case that the processing should be executed by the specified time interval.
Event driven task program	<ul style="list-style-type: none">• A shorter processing is executed for internal or external interrupt.

4.3.2 Program Execution Procedure

The followings explain the program execution procedure when the power is applied or the mode setting switch of CPU module is in the RUN status.

Program operation processing is executed as the procedure given below



REMARK

- 1) *1 : In the GLOFA PLC, the time driven interrupt task programs and event driven interrupt task programs are called task program.
Event driven programs are classified into single task(internal interrupt) or interrupt task (external interrupt) according to the S/W and H/W interrupt signaling method.

1) Initialization program

(1) Function

- The Initialization program initializes the program to execute scan and task programs.
- The initialization can be executed with the restart mode which has been specified for program.

(2) Restart mode execution conditions

- The initialization tasks can be specified as below complying with the purpose of the initialization task.
 - Program for Cold/ Warm restart started by the _INIT task

(3) Cold/ Warm Restart program

- The initialization program specified to _INIT task is executed with cold or warm restart mode when the operation starts.
- This initialization program executes the operations repeatedly until the setting conditions are satisfied(that is, until the Flag_INIT_DONE in the initialization program turns on). However, the I/O refresh is still executed.

(4) Flag

- _INIT_RUN flag is on during executing the initialization program.

2) Scan program

(1) Function

- In order to process signals which repeats constantly, the program executes its sequential operation repeatedly from the first step to the end step.
- If the interrupt task execution condition has been satisfied by a time driven task or event driven task module during scan program execution, the program that is under execution will be temporary stopped and the corresponding task program will be executed.
- If the scan program has been completely executed, the single task(internal interrupt) execution condition will be checked and the corresponding task program will be executed.

(2) configuration

- Up to 100 scan programs can be used.
(If task programs are used, the usable number is reduced as many as that of the used task programs)
- Program has been not specified to initialization or task program when writing that program, it will be automatically specified to scan program.
- Scan program has lowest execution priority and the priorities of scan program are determined their registration sequence in the GMWIN screen when writing those programs.

3) Task program

(1) Function

- In order to process internal/ external signal which occurs periodically or non-periodically, the task program temporarily stop the operation of scan program and processes first the corresponding function

(2) Types

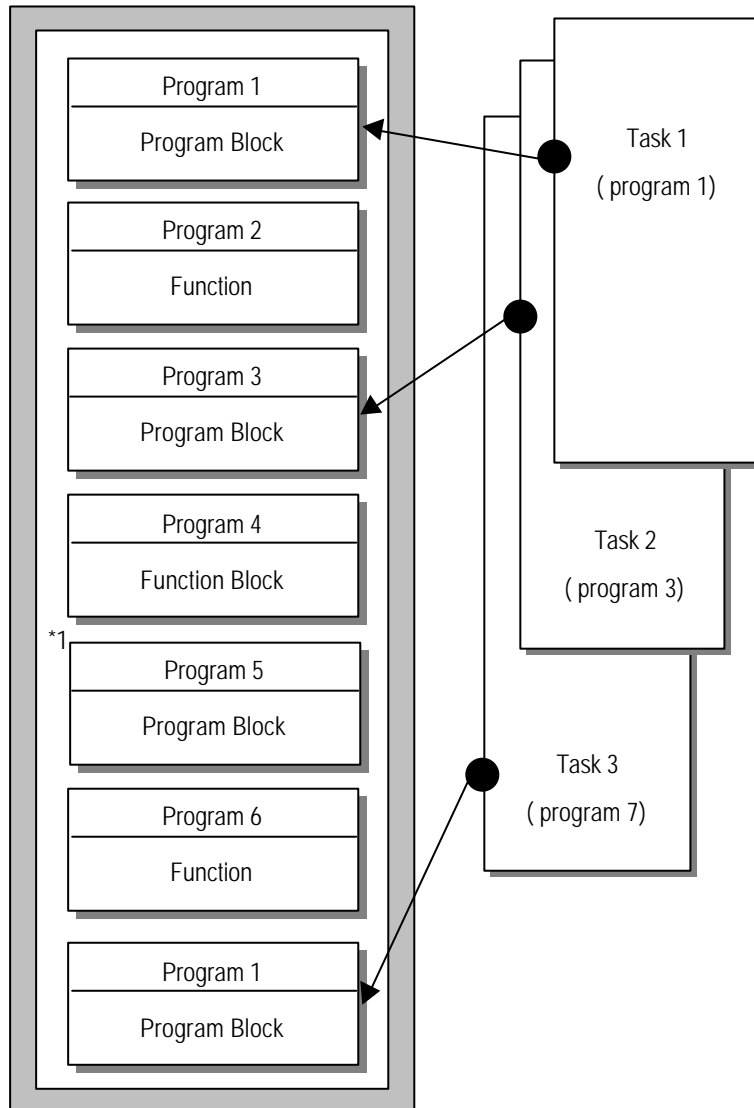
- Task programs are classified into the three types as below.
 - Time driven task program : Up to 8 programs are applicable
 - Single (internal) task program : Up to 8 programs are applicable
 - Interrupt (external) task program : Up to 8 programs are applicable
- Time driven task program
 - The program is executed by the time interval set before
- Single (internal) task program
 - The corresponding program will be executed at the rising edge and on state of internal contact in the program.
 - The detection of the start up condition will be executed after the scan program has been processed.
- Interrupt (external) task program
 - The program is executed according to the external signal a input to the interrupt module

REMARK

- 1) Refer to section 4.3.3 task for details of task program.
- 2) For interrupt signal processing, the GM6 series use general digital input module instead of external interrupt input module. Refer 4.3.3. task for details.

4.3.3 Task

The followings explain the program structure and tasks of the GMWIN, that is, the GLOFA-GM programming S/W, in order to give an understanding of the task function



REMARK

1) A task executes the some function as the control panel which are used to execute programs. Each task consists of one or more program blocks in the three types of program. Those programs are called task programs A program to which a task has not been specified as marked with '*1' will be automatically specified to scan program

1) Task types and functions

The following table show the types and functions of tasks

Type Specifications	Time driven task	External interrupt task	Internal interrupt task
Number ¹⁾	8	8	8
Start up condition	Time driven interrupt (up to 4,294,967.29sec by the 10msec)	At the rising edge of input contact on the designated slot	The rising edge or on state of the BOOL variable data which has been specified of buffer data
Detection and execution	Executed periodically as setting time	Immediately executed when an edge occurs in the interrupt module	Executed with edge detection after scan program has been finished
Detection delay time	Up to 1msec delay	Maximum 1msec delay + Input module delay(Within 3msec)	Delayed for the same time as maximum scan time
Execution priority	Level 0 to 7 (Level 0 has highest priority)	Level 0 to 7	Level 0 to 7

* 1) Up to 8 task programs are available.

2) Task program processing Method

The following explains the common processing method and instructions for task programs

(1) Task program characteristics

- The task program will be executed when a execution condition is satisfied while the scan program is repeatedly processed at every scan. Be sure to consider that point when writing a task program
- For example, if a timer and a counter have been used in a 10 sec cycle time driven task program, the timer can occur up to 10 sec error and an input which has been changed within 10 sec will not be counted because the counter checks its input status every 10 sec

(2) Execution priority

- The higher priority task program will be executed firstly.
- If a newly invoked task has higher priority than that of existing tasks which are under execution, they are temporary stopped and task has higher priority will be executed.
- When determining the priority of a task program, consider the characteristics, importance and urgency of the program

(3) Processing delay time

The following factors influence on the processing delay of task program, consider the characteristics, importance and urgency of the program

- Task detection delay (Refer to the detailed description of each task)
- Execution delay due to the execution of prior task programs
- Delay due to the execution of higher priority task programs while executing task programs

(4) Relationship of task program to initialization or scan program

- User defined tasks will not start while the initialization task program is being executed.
- As scan program has the lowest priority, if a task is invoked the scan program will be stopped and the task programs will be processed prior to them. Therefore, if tasks are invoked many times or concentrated sometimes the scan time may be extended abnormally. Be cautious when setting task conditions.

(5) Protection of the programs under execution from task programs

- If problems can occur in case that program lose its execution continuousness by the task programs which have higher priorities, the execution of task programs can be partly perverted. For program protection, use the DI function(Task program start-up disable) or EI function(task program start-up enable)

3) Time driven task program processing method

The followings explain the processing method of a task program when its task condition(start-up condition) has been set to be driven by time.

(1) Settings that have to be set for the task

- Set the task execution cycle and its priority which are used as start-up conditions for the task programs to be executed. Priority number will be assigned as task number automatically.

(2) Time driven task processing

- The corresponding time driven interrupt task program will be executed every setting time interval (execution cycle).

(3) Precautions for using the time driven task program

- While a time driven task program is being executed or ready for its execution, if a same priority task program has been invoked to be executed the newly invoked task will be ignored, the representative task collision warning flag (`_TASK_ERR`) will be set to ON, the detailed system error flag(`_TC_BMAP[n]`) will be set to ON at its corresponding location and occurrence time of the time driven tasks whose execution requests have been ignored will be written at its corresponding location of the flag `_TC_CNT[n]`.
- The timer that invokes the execution request for time driven task programs will be incremented only when the operation mode is in the RUN mode

If the RUN mode has been changed into the PAUSE mode while operating with the RUN mode, and then the operation mode has been changed again into the RUN mode, the operation time spent with the PAUSE mode will be ignored.

- When setting the execution cycle for a time driven task program, be cautious that execution requests for many time driven task programs can occur. If four time driven task programs of cycle 2, 4, 10 and 20sec are used, four execution requests will occur every 20 sec and scan time can be momentarily extended.

4) External contact program processing method

The following explains in the case that the task(start-up condition) of a task program has been set to an external input signal.

(1) Settings that have to be set for the input module

- A contact of input module can be used as interrupt input.

(2) Settings that have to be set for the task

- Set the contact No. of input module and priority for the task that will be used as start-up conditions of the task programs to be executed. Priority will be the task number.

(3) External contact task processing

- The CPU module checks the occurrence of interrupt input every 1ms and executes the task program which are designated by the contact at which the signal has been occurred.

(4) Precautions for using an external contact task.

- While a task program which are designated by an input module having interrupt input, contact is being executed or ready for its execution, if an execution request of a task program has been occurred to the same input contact then the newly invoked task will be ignored, the representative task collision warning flag(_TASK_ERR) will be set to ON, the detailed system error flag(_TC_BAMP[n]) will be set to ON at its corresponding location and the occurrence time of the external task whose execution request has been congested.
- Execution request for a task program can be accepted only when the operation mode is in the RUN mode. That is, if the RUN mode has been changed into the PAUSE mode while operating with the RUN mode and the operation mode has been changed into the RUN mode again, all execution requests occurred during the operation with the PAUSE mode will be ignored.

5) Internal task program processing method

The following explains the processing method when the task (start-up condition) of a task program has been set to the contact of direct variable area(I, Q or M) or automatic variable area.

(1) Settings that have to be set for the task.

- Set the contact No. of input module and priority for the task that will be used as start-up conditions of the task programs to be executed. Priority will be the task number.

(2) Internal contact task processing

- After the execution of scan program has been completed in the CPU module, the internal contacts that are the start-up conditions of the task program will be checked and the internal task programs where rising edge or on state has been occurred will be executed in accordance with its parameter.

(3) Precautions when using an internal task program.

- The internal task program is executed when scan program has finished its execution. Therefore, though the execution condition for the internal task program has been invoked in the scan program or task program(time driven, external) the task (start-up condition) will not be immediately executed but will be executed when scan program has finished its execution.

- If execution of an internal task program is requested, the execution conditions will be checked when scan program has finished its execution. Therefore, if an internal task execution conditions, during 'one' scan, has been occurred and disappeared (if the specified contact has been turned from OFF to ON, and then from ON to OFF) by scan program or (time driven or external) task program the task will not be executed as the execution condition can not be detected at the time that execution conditions are being checked.

6) Task processing at momentary power failure

- In case of the power failure of 20 ms or less, the ready tasks before the power failure will be executed, a time driven task will be invoked with calculation of the power failure time, and time driven tasks invoked repeatedly before the power failure will be ignored.

7) Examination on task program

After writing down a task program, be sure to examine the following items.

(1) Task setting has been correctly done?

If tasks are invoked more frequently than necessary or several tasks are invoked simultaneously within one scan, the scan time become longer and irregular. In case that the task setting cannot be changed, check the maximum scan time.

(2) Task priorities are properly arranged?

The lower priority tasks still may not be processed after its time due to delay by higher priority tasks. In some cases, if the prior tasks have been delayed and next task occurs task collision can occur. Set the priority with due consideration of items such as urgency and execution time of a task.

(3) Task programs are written as shortly as possible?

If execution time of a task program is long, the scan time may become longer and irregular and also collision of task programs may occur. Therefore, write task programs as shortly as possible.

(4) Protection of lower priority programs against higher priority program isn't needed during execution of those programs.

If the priority of a task program (or a scan program) has been set to lower priority and other tasks must not interrupt during its execution, use the function 'DI' and 'EI' to protect the program partly. When processing global variables used commonly in other programs, special modules or communications modules, problems can occur.

REMARK

- 1) For examination on processing speed of scan program and task program, refer to the 'Scan time Calculation Example in the Section 4.2.3 'Scan Time'.

8) Example of program configuration and processing

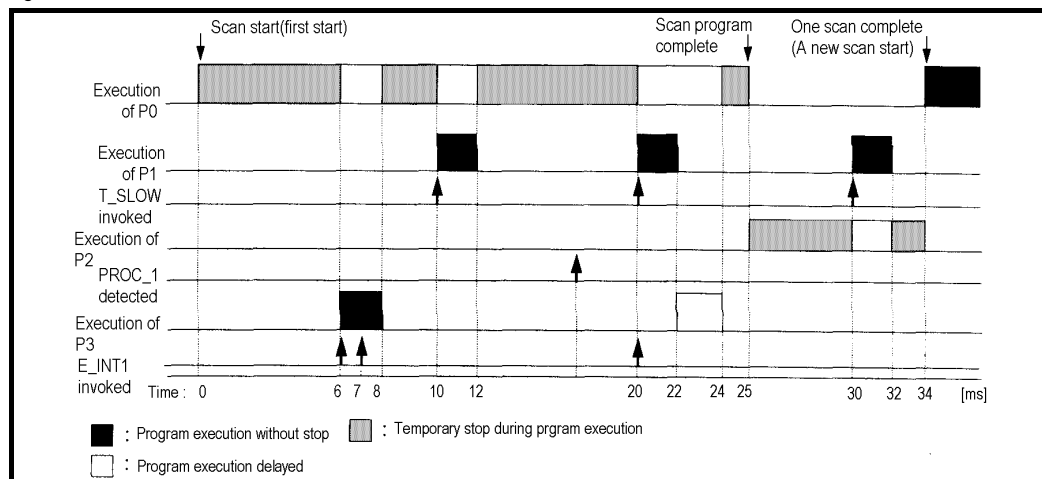
When the task and program have been registered as below,

- Task registration : T_SLOW (interval : T#10ms, priority : = 0)
PROC_1 (single : %MX0, priority : = 3)
E_INT1 (interrupt : %IX0.0.1, priority : = 0)
- program registration : program → P0
program → P1 with the task T_SLOW
program → P2 with the task PROC_1
program → P3 with the task E_INT1

If program execution time is equal to external interrupt occurrence time :

- Execution time for each program : P0 = 17 ms, P1 = 2 ms, P2 = 7 ms, P3 = 2 ms
- Interrupt E_INT occurrence time : Occurred at the 6, 7, 20 ms after the operation started.
- PROC_1 : Invoked during execution of scan program

Program execution is shown as below.



- Processing with time
 - 0 [ms] : Scan starts and the scan program P0 starts its execution.
 - 0 to 6 [ms] : The program P0 is being executed.
 - 6 to 8 [ms] : Execution request for P3 is input, and P0 is stopped and P3 is executed. Execution request for P1 by E_INT1 at the 7 [ms] is ignored as the P2 is being executed.
 - 8 to 10 [ms] : P3 finishes its execution and the P0 stopped continues its execution.
 - 10 to 12 [ms] : P0 is stopped and P1 is executed due to execution request for P1.
 - 12 to 20 [ms] : P2 finishes its execution and the P0 stopped continues its execution.
 - 20 [ms] : Execution requests for P1 and P3 are simultaneously exist, but the higher priority P1 is executed and P3 is ready for its execution.
 - 20 to 22 [ms] : P0 is stopped and P1 is executed.
 - 22 to 24 [ms] : P1 finishes its execution and the higher priority P3 is executed before P0.
 - 24 to 25 [ms] : P3 finishes its execution and the P0 stopped completes its execution.
 - 25 [ms] : Execution request for P2 is checked at the finish time of the scan program (P0) and P2 is executed.
 - 25 to 30 [ms] : The program P2 is executed.
 - 30 to 32 [ms] : Execution request for P1 is input and P2 is stopped and P1 finishes its execution.
 - 32 to 34 [ms] : P1 finishes its execution and the P2 stopped finishes its execution.
 - 34 [ms] : A new scan starts. (P0 starts its execution.)

4.3.4 Error Handling

1) Error Classification

Errors occur due to various causes such as PLC system defect, system configuration fault or abnormal operation result. Errors are classified into fatal error mode, which stops system operation for system stability, and ordinary error mode, which continues system operation with informing the user of its error warning.

The main factors that occurs the PLC system error are given as followings.

- PLC hardware defect
- System configuration error
- Operation error during execution of the user programs
- External device malfunction

2) Operation mode at error occurrence

In case of error occurrence, the PLC system write the error contents the corresponding flags and stops or continues its operation complying with its operation mode.

(1) PLC hardware defect

The system enters into the STOP state if a fatal error such as the CPU module defect has occurred, and continues its operation if an ordinary error such as battery error has occurred.

(2) System configuration error

This error occurs when the PLC hardware configuration differs from the configuration defined in the software. The system enter into the STOP state.

(3) Operation error during execution of the user programs

If the numeric operation error of these errors occurs during execution of the user programs, its contents are marked on the error flags and the system continues its operation. If operation time overruns the watch dog time or I/O modules loaded are not normally controlled, the system enters into the STOP state.

(4) External device malfunction

The PLC user program detects malfunctions of external devices. If a fatal error is detected the system enters into the STOP state, and if an ordinary error is detected the system continues its operation.

REMARK

- 1) In occurrence of a fatal error the state is to be stored in the representative system error flags, and an ordinary error in the representative system warning flags.
- 2) For details of flags, refer to Appendix 2. Flag List.

4.3.5 Precautions when using special modules

This system offers convenience and high performance in using special modules compared with the existing methods. Therefore, take some precautions when composing the system. Check the system after the following items have been thoroughly understood.

1) Special module programming

- (1) Special function block is offered for each special module to make programs concise and to prevent errors in writing down the user program.
- (2) Function blocks are largely of two types. 'Initialization' function block for initializing special modules and 'control' function block for control of the operations of special modules. Function block functions as an interface between the user program data and the special modules. As it includes the function that watches the operation status of special modules and indicates the error status, other separate error detection program does not have to be written.

(For detailed description of function block, refer to the User's Manuals of special modules and GLOFA-GM instructions.)

2) Special Module Initialization

This means to define the operations of a special module. It is done with 'initialization' function block. Generally, it specifies the data range to used channel, resolution or filtering method, etc. It defines the hardware characteristics and only one time execution at system start is sufficient.

REMARK

- 1) As the initialization should be finished before the scan program starts its execution, its program should be written in the restart program (initialization task program).

3) Control of special modules

In control the operations of special modules, write the program using function blocks which correspond to the operations that have to be controlled. These function blocks can locate at any place within the program.

REMARK

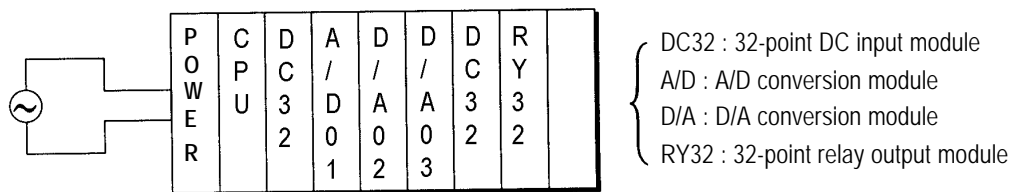
- 1) If a power failure occurs in the base unit where special units are loaded, special modules data are removed. Therefore, data should be newly written down in the program.

4) Restart Program Example

(1) System Configuration

The followings give an example for writing the initialization program of the system where a special module has been loaded onto its basic base unit shown as below figure.

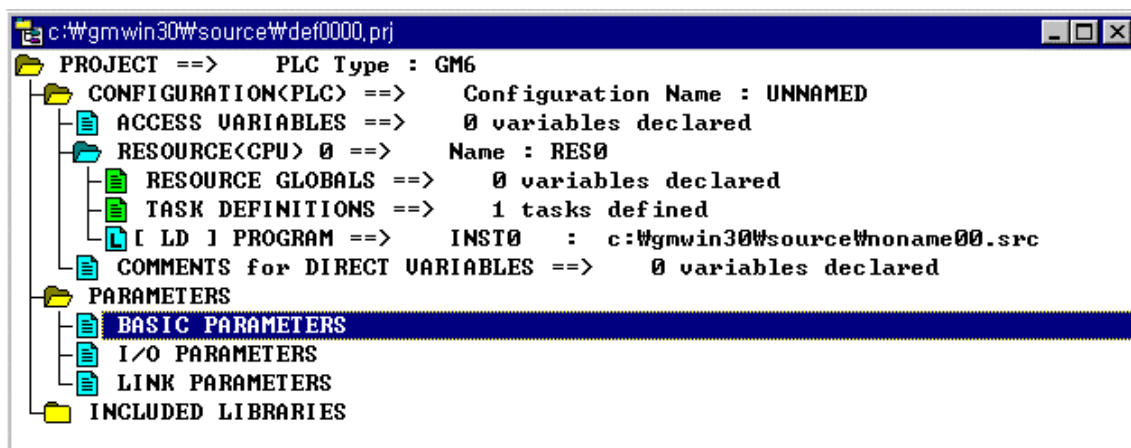
The followings describe an example for writing the 'cold/warm restart program' and 'scan program' for the scan program where the 'D/A 02' outputs data every scan and the 'D/A 03' outputs data only when the data has been changed.



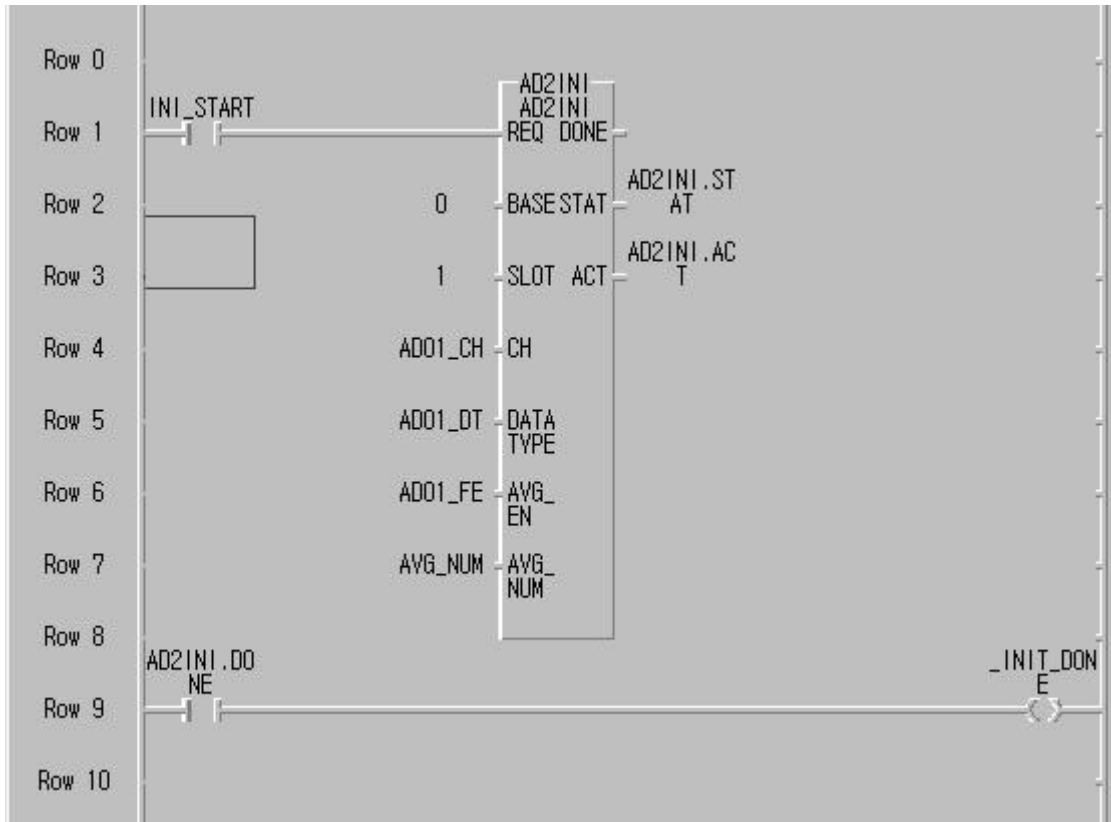
- As cold/warm restart makes the whole system restart, the 'cold/warm restart program' consists of only initialization program of special module.

(2) program

- Project Configuration : Restart.prj

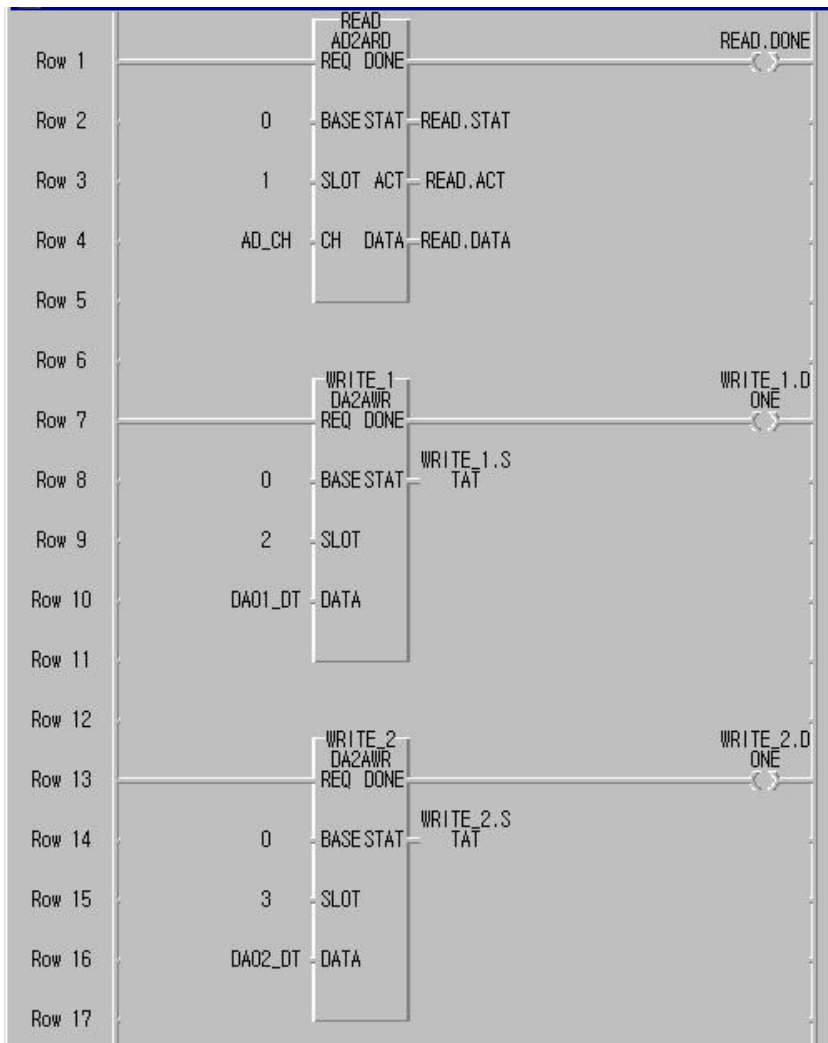


- Program : cw_rst.src (cold/warm restart initialization program)



Variable Name	Variable type	Data type	Initial value	Description
INI_START	VAR	BOOL	-	Start condition of initialization
AD2INI.ACT	VAR	ARRAY[4] OF BOOL	-	Shows active channel
AD01_DT	VAR	ARRAY[4] OF BOOL	Set by parameter	Select digital output type
AD01_CH	VAR	ARRAY[4] OF BOOL	Set by parameter	Select channel to be used
AD2INI	VAR	FB Instance	-	
AD2INI.STAT	VAR	USINT	-	Shows error status
AD01_FE	VAR	ARRAY[4] OF BOOL	-	Enable/Disable average function
AVG_NUM	VAR	ARRAY[4] OF BOOL	Set by parameter	

- Program : scan.src (scan program)



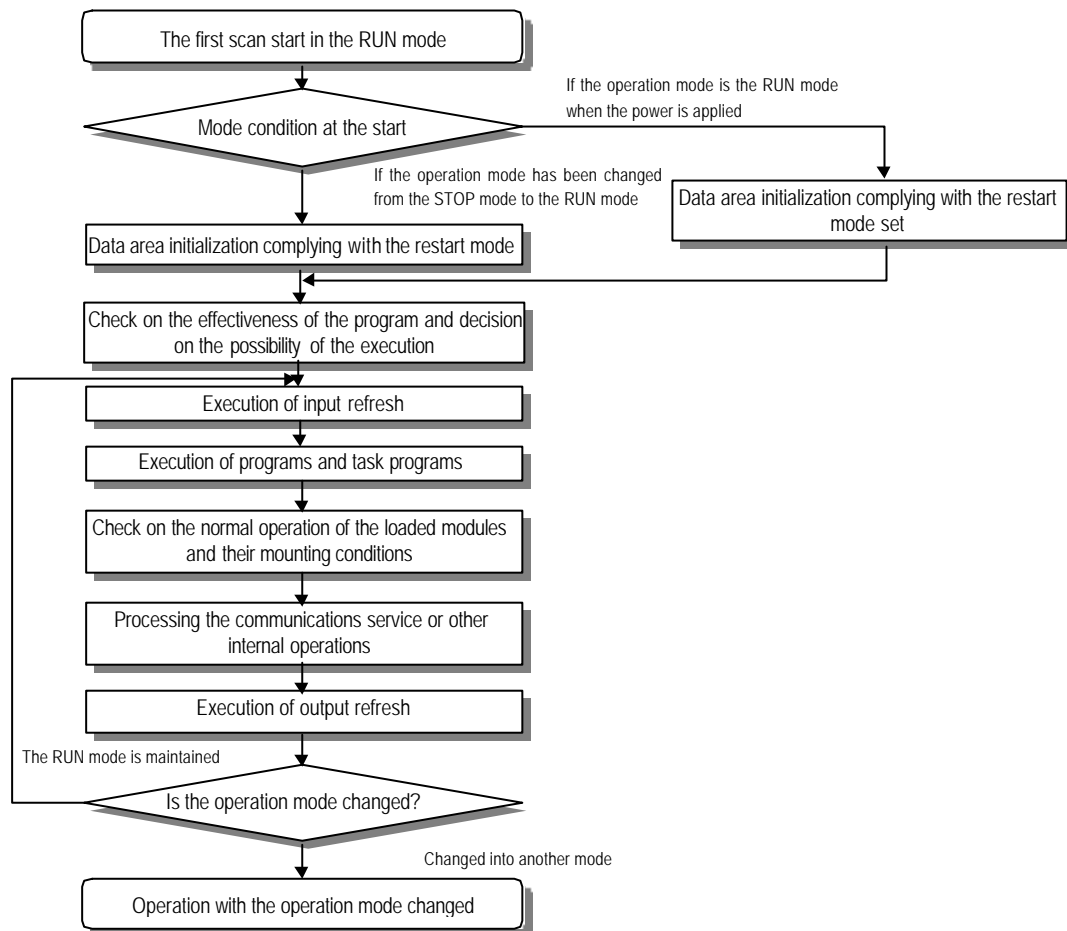
Variable Name	Variable type	Data type	Description
READ	VAR	FB Instance	
AD_CH	VAR	ARRAY[4] OF BOOL	Assign a channel of AD module to be used
READ.DONE	VAR	ARRAY[4] OF BOOL	Indicates the reading operation is completed
READ.STAT	VAR	USINT	Shows the error status of AD read FB
READ.ACT	VAR	ARRAY[4] OF BOOL	Shows the error status of AD read FB
READ.DATA	VAR	ARRAY[4] OF INT	Digital data converted from analog input
WRITE_1	VAR	FB Instance	
DA01_DT	VAR	ARRAY[4] OF INT	Digital data to be output
WRITE_1.DONE	VAR	BOOL	Indicates the write operation is completed
WRITE_1.STAT	VAR	USINT	Shows the error status of DA write FB
WRITE_2	VAR	FB Instance	
DA02_DT	VAR	ARRAY[4] OF INT	Digital data to be output
WRITE_2.DONE	VAR	BOOL	Indicates the write operation is completed
WRITE_2.STAT	VAR	USINT	Shows the error status of DA write FB

4.4 Operation Modes

The CPU module operates in one of the four modes - the RUN, STOP, PAUSE and DEBUG mode. The following describes the PLC operation processing in each operation mode.

4.4.1 RUN mode

In this mode, programs are normally operated.



1) Processing when the operation mode changes.

Initialization of data area is executed when the first scan starts.

- (1) If the PLC is in the RUN mode when applying the power :
- (2) If the operation mode has been changed into from the STOP mode into the RUN mode : the initialization is executed complying with the restart mode set. (cold / warm)
- (3) The possibility of execution of the program is decided with check on its effectiveness.

2) Operation processing contents

I/O refresh and program operation are executed.

- (1) Task programs are executed with the detection of their start-up conditions.
- (2) Normal or abnormal operation and mounting conditions of the loaded module are checked.
- (3) Communications service or other internal operations are processed.

4.4.2 STOP mode

In this mode, programs are not operated.

1) Processing when the operation mode changes

The output image area is cleared and output refresh is executed.

2) Operation processing contents

- (1) I/O refresh is executed.
- (2) Normal or abnormal operation and mounting conditions of the loaded module are checked.
- (3) Communications service or other internal operations are processed.

4.4.3 PAUSE mode

In this mode, the program operation is temporarily stopped. If it returns to the RUN mode, the operation continues from the state before the stop.

1) Processing when the operation mode changes

Data area clear and input image clear are not executed and the operating conditions just before the mode change is maintain.

2) Operation processing contents

- (1) I/O refresh is executed.
- (2) Normal or abnormal operation and mounting conditions of the loaded module are checked.
- (3) Communications service or other internal operations are processed.

4.4.4 DEBUG mode

In this mode, errors of a program are searched and the operation sequence is traced. Changing into this mode is only possible in the STOP mode. In this mode, a program can be checked with examination on its execution state and contents of each data.

1) Processing when the operation mode changes

- (1) Data area is initialized at the starting time of the mode change complying with the restart mode, which has been set on the parameters.
- (2) The output image area is cleared and output refresh is executed.

2) Operation processing contents

- (1) I/O refresh is executed by one time every scan.
- (2) Communications service or other internal operations are processed.

3) Debug operation conditions

- Two or more of the following four operation conditions can be simultaneously specified.

Operation conditions	Description
Executed by the one operation unit, (step over)	If an operation command is ordered, the system operates one operation unit and stops.
Executed to the specified breakpoint.	<ul style="list-style-type: none"> If break step is specified in the program, the operation stops at those step before execution. Up to 8 breakpoints can be specified.
Executed according to the contact state	If the contact area to be watched and the condition (Read, Write, Value) where the operation has to stop are specified, the operation stops when the specified operation occurs at the specified contact.(after execution)
Executed by the specified scan number.	If the number of scan that will be operated is specified, the operation stops after it has operated by the specified scan number.

4) Operation method

- Execute the operation after the debug operation conditions have been set in the GMWIN.
- In task programs, each task can be specified to operation enable/disable.(For detailed operation method, refer to the GMWIN User's Manual Chapter 9.

4.4.5 Operation mode change

1) Operation mode change methods

The following method are used to change the operation mode.

- Change by the mode setting switch of CPU module.
- Change by the GMWIN connected with the CPU module communications port.
- Change by the GMWIN connected to the remote CPU module through Fnet.
- Change by the user's command using FAM or computer link module, etc.
- Change by the 'STOP function', 'ESTOP function' during program execution.

2) Operation mode change by the mode setting switch of CPU module

The following shows the operation mode change by the mode setting switch of CPU module.

Mode setting switch position	Operation mode
RUN	Local RUN
STOP	Local STOP
STOP → PAU/REM	Remote STOP
PAU/REM → RUN ¹⁾	Local RUN
RUN → PAU/REM ²⁾	Local PAUSE / Remote RUN
PAU/REM → STOP	Local STOP

REMARK

- If the operation mode changes from RUN mode to local RUN mode by the mode setting switch, the PLC operates continuously without stop.
- If Local PAUSE disable(or Local PAUSE enable) is set by parameter in GMWIN, it operated as Remote RUN(or Local PAUSE).

3) Remote operation mode change

Remote operation mode change is available only when the operation mode is set to the remote STOP mode (i.e., the mode setting switch position is in the STOP →PAU/REM).

Mode setting switch position	Mode Change	Mode change by the GMWIN	Mode change using FAM or computer link, etc.
PAU/REM	Remote STOP → Remote RUN	○	○
	Remote STOP → Remote PAUSE	×	×
	Remote STOP → DEBUG	○	○
	Remote RUN → Remote PAUSE	○	○
	Remote RUN → Remote STOP	○	○
	Remote RUN → DEBUG	×	×
	Remote PAUSE → Remote RUN	○	○
	Remote PAUSE → Remote STOP	○	○
	Remote PAUSE → Remote DEBUG	×	×
	DEBUG → Remote STOP	○	○
	DEBUG → Remote RUN	×	×
	DEBUG → Remote PAUSE	×	×

4) Remote operation mode change enable/disable

It is possible to disable the mode change for system protection so that some parts of the operation mode sources cannot change the mode. If remote operation mode change has been disabled, the operation mode change is possible only by the mode setting switch and GMWIN. To enable the remote operation change, set the parameter 'Enabling the PLC control by communications' to enable. (For details, refer to the Appendix 1. System Definitions)

4.5 Functions

4.5.1 Restart mode

The restart mode defines how to initialize variables and the system and how to operate in the RUN mode when the system starts its operation with the RUN mode by re-application of the power or mode change. Two restart modes, cold and warm restart are available and the execution condition for each restart mode is given below.

(For details, refer to the '4.5.1 Basic Parameters Edit' of the GMWIN User's Manual Section 4.5 Parameters Edit.

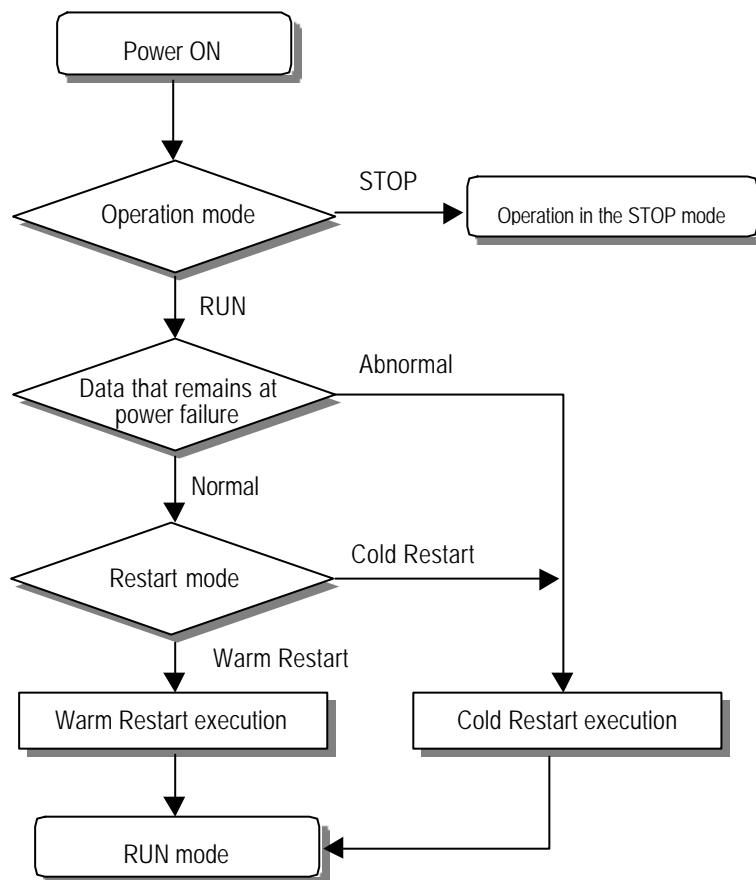
1) Cold Restart

- (1) It is executed when the restart mode parameter has been set to the cold restart mode.
- (2) All data are cleared with '0' and only the variables to which their initial value has been defined will be set to their initial value.
- (3) Though the parameter has been set to the warm restart mode, cold restart will be executed at the first execution of a program after it has been changed.
- (4) In case of selection 'Reset' command in the GMWIN, it restarts in accordance with setting in parameter and in case of selection 'Overall Reset' command, it restarts as cold restart mode.

2) Warm Restart

- (1) It is executed when the restart mode parameter has been set to the warm restart mode.
- (2) A data which set as retain & initial will be retain and a data which set as initial value will be set with default value during the warm restart. All other data will be cleared with '0'.
- (3) Though the parameter has been set to the warm restart mode, cold restart will be executed at the first execution of a program after it has been stopped due to its download or error.
- (4) Though the parameter has been set to the warm restart mode, cold restart will be executed if data contents are abnormal (i.e., the data does not remain at a power failure)

- Restart mode is executed as the figure given below when the power has been re-applied during execution of the CPU module



4) Data initialization according to the restart mode

The variables relating to the restart mode are classified into three types, i.e., default variable, initialization variable and retain variable. The following table shows the initialization method for each type variable.

Variable type \ Mode	Cold	Warm
Default	Initialized with '0'	Initialized with '0'
Retain	Initialized with '0'	Previous value is retained
Initialization	Initialized with the user defined value	Initialized with the user defined value
Retain & Initialization	Initialized with the user defined value	Previous value is retained

REMARK

1) Definitions

- (1) Default variable : A variable whose initial value is not defined or previous value will not be retained.
- (2) Initialization variable : A variable whose initial value is defined.
- (3) Retain variable : A variable whose previous value will be retained.

4.5.2 Self-diagnosis

1) Functions

- (1) The self-diagnosis function permits the CPU module to detect its own errors.
- (2) Self-diagnosis is carried out when the PLC power supply is turned on and when an error occurs the PLC is in the RUN state. If an error is detected, the system stops operation to prevent faulty PLC operation.

2) Error flag

If an error occurs, it will be stored to the following flags and the STOP LED flickers.

- Representative system error flag : _CNT_ER
- Representative system warning flag : _CNF_WAR

REMARK
1) Refer to 12.5 Error Code List of Chapter 12. Troubleshooting for details of contents of self-diagnosis and corrective actions.

4.5.3 Remote function

The CPU module can be controlled by external operations (from GMWIN and computer link module, etc.). For remote operation, set the mode setting switch of CPU module to remote position.

1) Remote RUN/STOP

- (1) The remote RUN/STOP permits external operations to RUN/STOP the CPU module under the condition that the mode setting switch of CPU module is in the remote position.
- (2) This function is convenient when the CPU module is located on the place where it is difficult to control the CPU module or the user want to control the CPU module in the control panel from outside.

2) Remote PAUSE

- (1) The remote PAUSE permits external operations to execute PAUSE operations under the condition that the mode setting switch of CPU module is in the remote position. The PAUSE operations stop the CPU module operation processing while maintaining the On/Off state of the output module.
- (2) This function is convenient when the user wants to maintain the ON state of the output module under the condition the CPU module has been stopped.

3) Remote DEBUG

- (1) This function permits external operations to execute DEBUG operations under the condition that the mode setting switch of CPU module is in the remote position. The DEBUG operations execute programs complying with the specified operation conditions.
- (2) This function is convenient when program execution or contents of any data are checked for debugging of the program.

4) Remote reset

- (1) This function permits remote operations to reset the CPU module, which locates in the place where direct operations cannot be applied, when an error has occurred.

REMARK

- 1) For remote function operations, refer to the GMWIN User's Manual Chapter 7. On-line.

4.5.4 I/O Force On/Off function

1) Force On/Off setting method

Force on/off setting is applied to input area and output area.

Force on/off should be set for each input and output, the setting operates from the time that 'Force I/O setting enable' is set.

This setting can be done when I/O modules are not really loaded.

2) Force on/off Processing timing and method

(1) Force Input

- After data have been read from input modules, at the time of input refresh the data of the junctions which have been set to force on/off will be replaced with force setting data to change the input image area. And then, the user program will be executed with real input data and force setting data.

(2) Force output

- When a user program has finished its execution the output image area has the operation results. At the time of output refresh the data of the junctions which have been set to force on/off will be replaced with force setting data and the replaced data will be output. However, the force on/off setting does not change the output image area data while it changes the input image area data.

(3) Force on/off processing area

- Input/output areas for force on/off setting are larger than the real I/O areas. If remote I/O is specified using this area, the force on/off function is as just available in it as in the basic I/O areas.

(4) Precautions

- Turning the power off and on, change of the operation mode or operation by reset switch(GM3) does not change the previous force on/off setting data. They remain within the CPU module and operation is executed with the same data.
- Force I/O data will not be cleared even in the STOP mode.
- If a program is downloaded or its backup breaks, the force on/off setting data will be cleared. The operating program in memory differs from the program in the flash memory so that if operation restarts with the program in the flash memory the on/off setting data will be also cleared.
- When setting new data, disable every I/O settings using the setting data 'clear' function and set the new data.

REMARK
1) For detailed operation, refer to the GMWIN User's Manual Chapter 7 'Force I/O setting.'

4.5.5 History Log-In

The GM6 CPU stores 3 operation histories such as error occurrence, mode change, and power shut-down. Each history log-in contains the last 16 operation histories.

1) Error occurrence

- Record occurrence time and error code when an error occurred while the CPU is in RUN mode.

2) Mode change

- Record the mode change time, operation mode, and restart mode when a operation mode is changed.

3) Power failure

- Record the occurrence time and total occurrence number when the AC failure occur while the CPU is in RUN mode.

4.5.6 External Device Error Diagnosis function

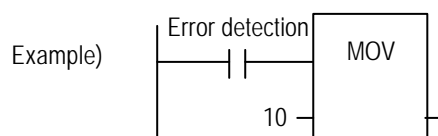
Flags are given for the user to implement easily the program in which the error detection of external devices and system stop and warning are coded. By use of these flags, error indication of external devices is possible without complex programming and monitoring of the error location can be done without special tools (GMWIN, etc.) or source programs.

1) External device fault detection and classification

- (1) The user program detects external device faults. The faults are classified into fatal fault(error), where the PLC stops its operation, and ordinary fault(warning), where operation continues.
- (2) The flag ANC_ERR[n] is used to indicate error. The flag ANC_WN[n] is used to indicate warning.

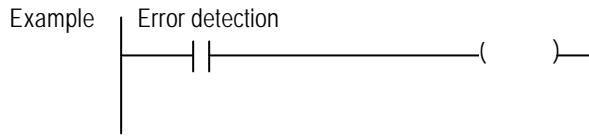
2) External Device Fatal-fault (Error) Processing.

- (1) If an error of external device is detected and the error type, where other value than 0 is used, is written to the system flag ANC_ERR[n], the flag will checked at the time that scan program finishes its execution. If an error is indicated on the flag, it will be also indicated on the _ANNUN_ER of the representative system error flag _CNF_ER, the PLC turns all output modules off and the error state will be same as the PLC self-diagnosis.
- (2) The user can know the cause of error by use of the GMWIN, and also by direct monitoring of the flag _ANC_ERR[n].
- (3) As the flag _ANC_ERR[n] has sixteen elements(n : 0 to 15), the user can classify error states largely. User defined error No. can be written to the elements. A number of 1 to 65535 is usable.



3) External device Ordinary-fault (Warning) Processing.

- (1) If a warning of external device is detected and the corresponding flag of the system flag `_ANC_WB[n]` is set to on, the flag will be checked from the `_ANC_WB[0]` at the time that scan program finishes its execution. If an error is indicated on the flag, it will be also indicated on the `_ANNUN_WR` of the representative system warning flag `_CNF_WAR`. External device warning numbers will be written to from `_ANC_WAR[0]` to `_ANC_WAR[7]` according to occurrence sequence.
- (2) The user can know the cause of error by use of the GMWIN, and also by direct monitoring of the flags `_ANC_WAR[n]` and `_ANC_WB[n]`.
- (3) If an external device warning is removed, that is, the elements of `_ANC_WAR[n]` are released from warning, the corresponding `_ANC_WAR[n]` will be automatically cleared. If all element flags are cleared, the flag `_ANNUN_WR` of the system flag `_CNF_WAR` will be reset.



_ANNUN_WR	= 1	If the user program had detected a system fault and set _ANC_WB[10] to ON, the states of _ANNUN_WR and _ANN_WAR [0..7] will be shown as left after the scan has been finished	
	= 10		
_ANC	= 0		
_WAR[0]	= 0		
	= 0		
	= 0		
	= 0		
_ANC	= 0		
_WAR[1]	= 0		
_ANNUN_WR	= 1	After the next scan has been finished, if the numbers 1, 2, 3, 10, 15 , 40 , 50, 60 and 75 of _ANC_WB[n] are tuned on _ANC_WAR[n] will be shown as left	
	= 10		
_ANC	= 0		
_WAR[0]	= 0		
	= 0		
	= 0		
	= 0		
_ANC	= 0	As the number 10 has turned on (has occurred) in the previous scan, though the number 10 has lower priority than the numbers 1, 2 and 3, it will be the lower element of _ANC_WAR[n]. The _ANC_WB[75] is not indicated as it is turned on and the warning that occurred before has written to the _ANC_WAR[n].	
_WAR[1]	= 0		
_ANNUN_WR	= 1		After the next scan has been finished, if the numbers 1, 2, 3, 10, 15 , 40 , 50, 60 and 75 of _ANC_WB[n] are tuned on _ANC_WAR[n] will be shown as left.
	= 10		
_ANC	= 0		
_WAR[0]	= 0		
	= 0		
	= 0		
	= 0		
_ANC	= 0	The No. 10 warning has been released the content of _ANC_WAR[0] will be cleared and the contents of _ANC_WAR[1..7] will shift into the lower elements. The content of _ANC_WAR[7] will has been cleared by the shifting and the content of _ANC_WB[75] will be written to _ANC_WAR[7].	
_WAR[1]	= 0		
_ANNUN_WR	= 1		If all warnings indicated on the _ANC_WB[n] are released during operation, the _ANNUN_WR and _ANC_WAR[n] will be shown as left.
	= 10		
_ANC	= 0		
_WAR[0]	= 0		
	= 0		
	= 0		
	= 0		
_ANC	= 0		
_WAR[1]	= 0		

4.6 Memory Configuration

The CPU module includes two types of memory that are available by the user. One is program memory which is used to store the user programs written to implement a system by the user. The other is data memory which stores data during operation.

1) Program memory configuration

The table given below shows the contents to be stored and the storage capacity of program memory.

Item	Memory Capacity
Overall program memory area	68 k bytes
Parameter area <ul style="list-style-type: none">• Basic parameter area• I/O parameter area• High speed link parameter area• Interrupt setting information area	2 k bytes
Program area <ul style="list-style-type: none">• Scan program area• Task program area• User defined function/function block area• Standard library area• Access variable are• Variable initialization information area• Protective variable specification information area	66 k bytes

2) Data memory Configuration

The table given below shows the contents to be stored and the storage capacity of program memory.

Item	Memory Capacity
Overall data memory area	32 k bytes
System area <ul style="list-style-type: none">• I/O information table• Force I/O table	1 k bytes
System flag area	1.5 k bytes
Input image area (%IX)	128 bytes
Output image area (%QX)	128 bytes
Direct variable area (%M)	2 to 8 k bytes
Symbolic variable area (maximum)	29 k bytes – the size of direct variable area
Stack area	3 k bytes

3) Purpose

(1) System area

it used to store the self-created data of the CPU module for system management and GMWIN system control data.

(2) System flag area

it used to user flags and system flags. The user operates it with flag name.

(3) Input image area

It used to store input data read from input modules. Overall size is %IX0.0.0 to %IX1.7.63. The redundant area (Actual input module is not installed) can be used as auxiliary relay in user program. Especially, it is convenient to use the data storing area of remote input through high speed link.

(4) Output image area

It used to store operation results. The stored data are automatically output to output modules. Overall size is %QX0.0.0 to %QX1.7.63. The redundant area (Actual output module is not installed) can be used as auxiliary relay in user program. Especially, it is convenient to use the data storing area of remote output through high speed link.

(5) Direct variable area

The user can use this area to access direct memory data through the variable names such as %MX0, %MB0, %MW0 and %MD0, which was pre-defined by the system. Memory size is defined when program is made by user and it refers to 'App1. System Definitions'.

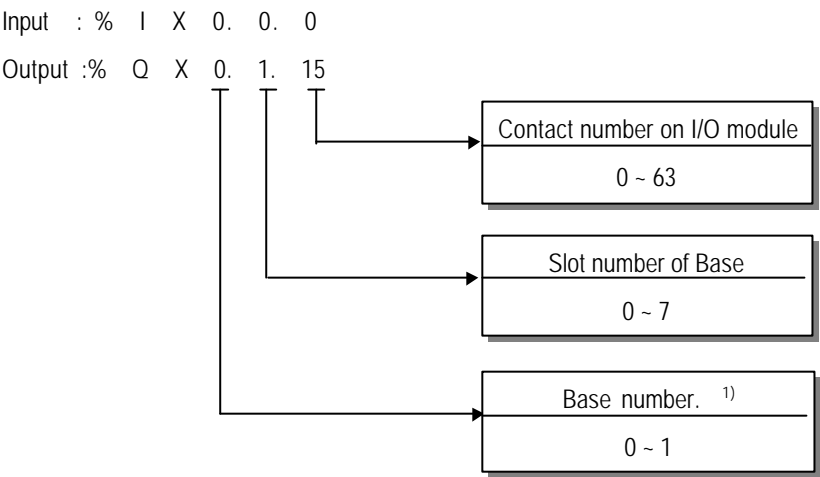
(6) Symbolic variable area

It used to store the variables that the user created, that is, whose names the user defined when writing a program. Global variables and instance memory are located in this area. The variables used in program blocks locates in the 'PB instance memory' of the program, and the memory used in function block locates in the 'FB instance memory'.

The maximum size of the PB instance memory is 32 Kbytes. If the used size overruns the maximum size, divide the program blocks or use global variables.

4.7 I/O No. Allocation Method

- 1) I/O No. allocation means to give an address to each module in order to read data from input modules and output data to output modules.
- 2) Fixed 64 points are allocated to each module for I/O points.
- 3) Fixed 64 points are allocated regardless of mounting/dismounting or type of modules.
- 4) The following shows I/O No. allocation method.

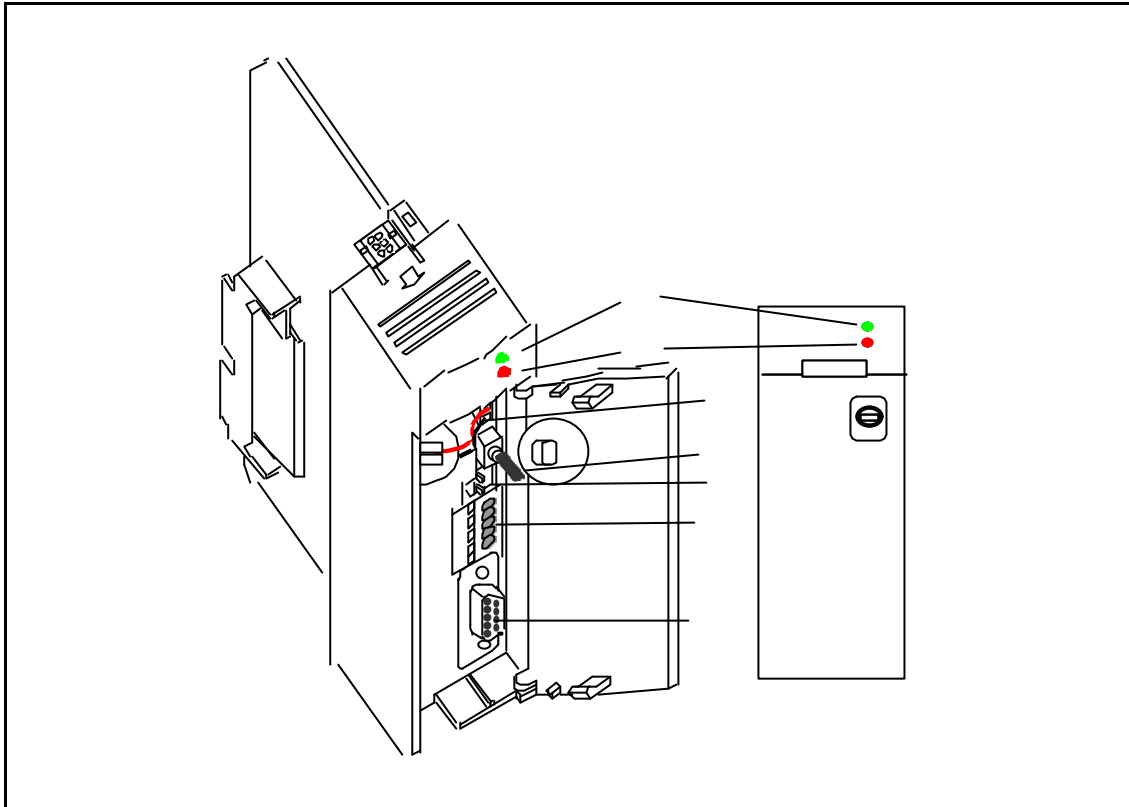


REMARK

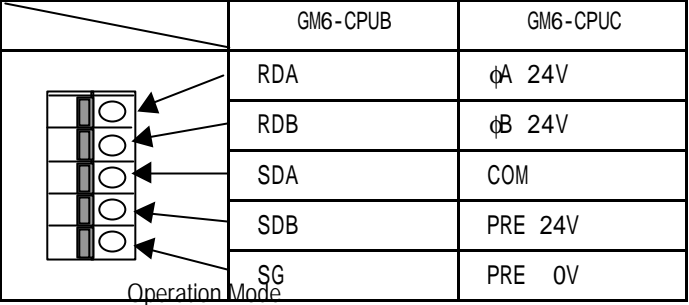
1) Although there is no expansion base, A base having more than 8 I/O slot which has a plan to develop set by 1 as base number.

4.8 Names of Parts

The following describes the names and functions of parts of the CPU module.



No.	Name	Function
1	RUN LED	<p>Indicates the operation status of the CPU module.</p> <ul style="list-style-type: none"> • On : when the CPU module operates with the mode setting switch in the local or remote RUN state. • Off : when the followings occur <ul style="list-style-type: none"> The voltage is not normally supplied to the SPU module. The mode setting switch is in the STOP or PAU/REM state. An error which makes operation stop is detected.
2	STOP LED	<ul style="list-style-type: none"> • On : when the mode setting switch is in the local or remote STOP state. • Off : when the followings occur <ul style="list-style-type: none"> The mode setting switch is in the local RUN or local PAUSE state. The operation state is in the RUM/PAUSE/DEBUG state. • Flickering : when an error is detected by self-diagnosis during operation.
3	Battery installing connector	It used to connect to the backup battery.
4	Mode setting switch	<p>Sets the operation mode of the CPU module. .</p> <ul style="list-style-type: none"> • RUN : Program operation is executed. • STOP : Program operation is temporarily stopped. • PAU/REM : <ul style="list-style-type: none"> PAUSE : Program operation is temporarily stopped. REMOTE: Used for the remote operation
5	DIP S/W for flash memory	See chap 6.

No.	Name	Function																		
6	Terminal block for built-in special function	<p>GM6-CPUA : N/A (The terminal block is not installed) GM6-CPUB : RS-422/485 interface terminal block GM6-CPUC : High speed counter input terminal block</p>  <table border="1"> <thead> <tr> <th></th><th>GM6-CPUB</th><th>GM6-CPUC</th></tr> </thead> <tbody> <tr> <td>RDA</td><td></td><td>φA 24V</td></tr> <tr> <td>RDB</td><td></td><td>φB 24V</td></tr> <tr> <td>SDA</td><td></td><td>COM</td></tr> <tr> <td>SDB</td><td></td><td>PRE 24V</td></tr> <tr> <td>SG</td><td></td><td>PRE 0V</td></tr> </tbody> </table>		GM6-CPUB	GM6-CPUC	RDA		φA 24V	RDB		φB 24V	SDA		COM	SDB		PRE 24V	SG		PRE 0V
	GM6-CPUB	GM6-CPUC																		
RDA		φA 24V																		
RDB		φB 24V																		
SDA		COM																		
SDB		PRE 24V																		
SG		PRE 0V																		
6	RS-232C connector	<p>It used to connect to peripheral devices(GMWIN, etc.) GM6-CPUA and GM6-CPUC have built-in RS-232C interface function, and it shares the RS-232C connector with peripheral device interface. (Refer the chapter 13 for details)</p>																		

REMARK

The followings shows the LED status complying with the operation mode, and the operation mode complying with the position of the mode setting switch.

1) LED status complying with the operation mode

	LED Status	
	RUN	STOP
Local Run	On	Off
Local Stop	Off	On
Local Pause	Off	Off
Remote Run	On	Off
Remote Stop	Off	On
Remote Pause, Remote Debug	Off	Off

2) Operation mode complying with the position of the mode setting switch.

Position of Mode switch	Operation Mode
STOP → PAU/REM →	Remote Stop
PAU/REM → RUN	Local Run
RUN → PAU/REM	Local Pause ¹⁾

- Change of remote mode is available only after the operation mode has entered into the remote STOP mode.

caution 1) In case of local pause disable, it operated as Remote Run.