

### 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.

<b>REMARK</b>
---------------

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 Direct I/O Operation function

This function is usefully available when an input junction state is directly read during execution of a program and used in the operation, or the operation result is directly output to an output junction.

##### 1) Direct input

- Direct input is executed by use of the 'DIRECT\_IN' function. If this function is used, the input image area will be directly updated and applied to the continuing operations.

##### 2) Direct output

- Direct output is executed by use of the 'DIRECT\_O' function. If this function is used, the data of the output image area, which has the operation results by the time, will be directly output to the direct output module.

##### 3) Force on/off

- Force on/off settings are still effective when processing direct I/O.

#### REMARK

- 1) For detailed direct I/O functions, refer to the GLOFA-GM commands.

#### 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.

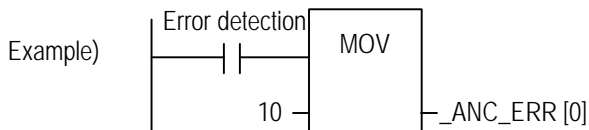
##### 1) External device fault detection and classification

- (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.

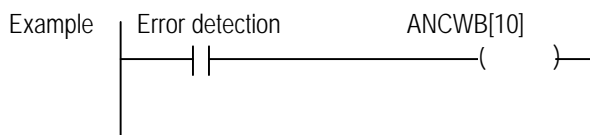
- (1) If an error of external device is detected and the error type, where other value than 0 is used, is written to the system flag ANC\_ERR[n], the flag will be checked at the time that scan program finishes its execution. If an error is indicated on the flag, it will be also indicated on the \_ANNUN\_ER 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.



<pre>  ANNUN_WR = 1  ANC_WAR[0] = 10  ANC_WAR[1] = 0  ANC_WAR[2] = 0  ANC_WAR[3] = 0  ANC_WAR[4] = 0  ANC_WAR[5] = 0  ANC_WAR[6] = 0  ANC_WAR[7] = 0 </pre>	<p>If the user program had detected a system fault and set <code>_ANC_WB[10]</code> to ON, the states of <code>_ANNUN_WR</code> and <code>_ANN_WAR [0..7]</code> will be shown as left after the scan has been finished</p>
---	---

<pre>  ANNUN_WR = 1  ANC_WAR[0] = 10  ANC_WAR[1] = 0  ANC_WAR[2] = 0  ANC_WAR[3] = 0  ANC_WAR[4] = 0  ANC_WAR[5] = 0  ANC_WAR[6] = 0  ANC_WAR[7] = 0 </pre>	<p>After the next scan has been finished, if the numbers 1, 2, 3, 10, 15, 40, 50, 60 and 75 of <code>_ANC_WB[n]</code> are tuned on <code>_ANC_WAR[n]</code> will be shown as left</p> <p>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 <code>_ANC_WAR[n]</code>. The <code>_ANC_WB[75]</code> is not indicated as it is turned on and the warning that occurred before has written to the <code>_ANC_WAR[n]</code>.</p>
---	---

<pre>  ANNUN_WR = 1  ANC_WAR[0] = 10  ANC_WAR[1] = 0  ANC_WAR[2] = 0  ANC_WAR[3] = 0  ANC_WAR[4] = 0  ANC_WAR[5] = 0  ANC_WAR[6] = 0  ANC_WAR[7] = 0 </pre>	<p>After the next scan has been finished, if the numbers 1, 2, 3, 10, 15, 40, 50, 60 and 75 of <code>_ANC_WB[n]</code> are tuned on <code>_ANC_WAR[n]</code> will be shown as left.</p> <p>The No. 10 warning has been released the content of <code>_ANC_WAR[0]</code> will be cleared and the contents of <code>_ANC_WAR[1..7]</code> will shift into the lower elements. The content of <code>_ANC_WAR[7]</code> will has been cleared by the shifting and the content of <code>_ANC_WB[75]</code> will be written to <code>_ANC_WAR[7]</code>.</p>
---	--

<pre>  ANNUN_WR = 1  ANC_WAR[0] = 10  ANC_WAR[1] = 0  ANC_WAR[2] = 0  ANC_WAR[3] = 0  ANC_WAR[4] = 0  ANC_WAR[5] = 0  ANC_WAR[6] = 0  ANC_WAR[7] = 0 </pre>	<p>If all warnings indicated on the <code>_ANC_WB[n]</code> are released during operation, the <code>_ANNUN_WR</code> and <code>_ANC_WAR[n]</code> will be shown as left.</p>
---	---

## 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"> <li>• Basic parameter area</li> <li>• I/O parameter area</li> <li>• High speed link parameter area</li> <li>• Interrupt setting information area</li> </ul>	2 k bytes
Program area <ul style="list-style-type: none"> <li>• Scan program area</li> <li>• Task program area</li> <li>• User defined function/function block area</li> <li>• Standard library area</li> <li>• Access variable are</li> <li>• Variable initialization information area</li> <li>• Protective variable specification information area</li> </ul>	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"> <li>• I/O information table</li> <li>• Force I/O table</li> </ul>	1 k bytes
System flag area	1.5 k bytes
Input image area (%IX)	128 k bytes
Output image area (%QX)	128 k 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.

#### (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.

#### (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'.

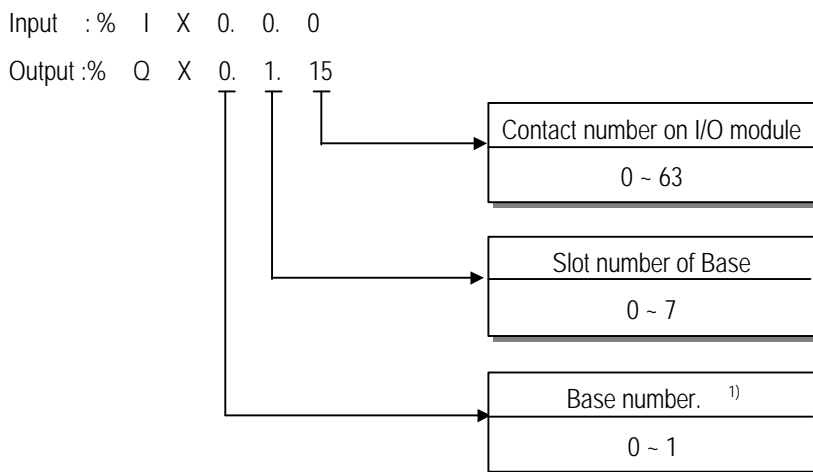
#### (6) Symbolic variable area

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.

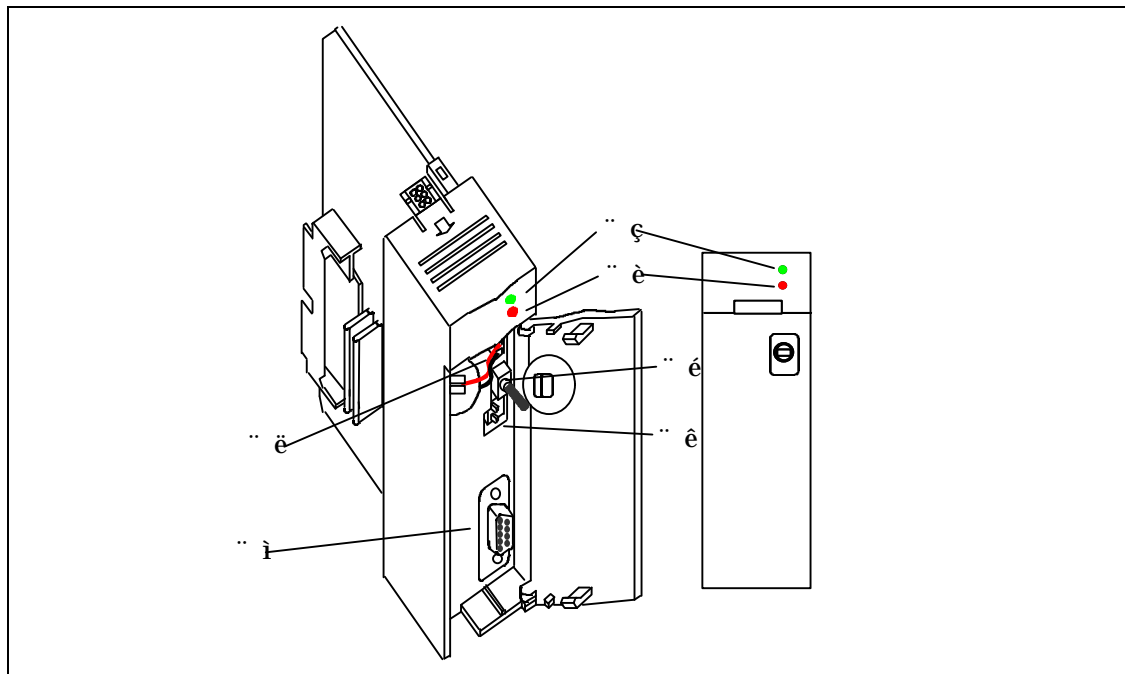


**REMARK**

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"> <li>• On : when the CPU module operates with the mode setting switch in the local or remote RUN state.</li> <li>• Off : when the followings occur                             <ul style="list-style-type: none"> <li>The voltage is not normally supplied to the SPU module.</li> <li>The mode setting switch is in the STOP or PAU/REM state.</li> <li>An error which makes operation stop is detected.</li> </ul> </li> </ul>
2	STOP LED	<ul style="list-style-type: none"> <li>• On : when the mode setting switch is in the local or remote STOP state.</li> <li>• Off : when the followings occur                             <ul style="list-style-type: none"> <li>The mode setting switch is in the local RUN or local PAUSE state.</li> <li>The operation state is in the RUM/PAUSE/DEBUG state.</li> </ul> </li> <li>• Flickering : when an error is detected by self-diagnosis during operation.</li> </ul>
3	Mode setting switch	<p>Sets the operation mode of the CPU module. .</p> <ul style="list-style-type: none"> <li>• RUN : Program operation is executed.</li> <li>• STOP : Program operation is temporarily stopped.</li> <li>• PAU/REM :                             <ul style="list-style-type: none"> <li>PAUSE : Program operation is temporarily stopped.</li> <li>REMOTE: Used for the remote operation</li> </ul> </li> </ul>
4	DIP S/W for flash memory	See chap 6.
5	Battery installing connector	It used to connect to the backup battery.
6	RS-232C connector	It used to connect to peripheral devices(GMWIN, etc.)

**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

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

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 <sup>1)</sup>
PAU/REM → STOP	Local Stop

- 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.