

FUJI可程式化控制器

MICREX-SX 系列

使用者手冊 說明

(針對 "SX程式設計師標準類
型：NP4H-SWN")

本「使用者手冊」將為您介紹SPH的系統組態、記憶體及語言。請仔細閱讀本手冊，以確保您能夠正確操作。在使用模組或周邊裝置時，請務必先仔細閱讀下面列出的對應使用者手冊。


標題	手冊編號	內容
使用者手冊硬體，MICREX-SX系列SPH	FEH201	說明MICREX-SX系列產品中模組的系統組態、規格及操作方式。
MICREX-SX 系列 SX-Programer Standard 的延伸 FB使用者手冊	FEH589	說明了SX-Programer Standard 的延伸FB 之規格及操作方式（安裝程序等）。
MICREX-SX 系列 SX-Programer Standard 使用者手冊	FEH590	說明了SX-Programer Standard的功能表、圖示及操作方式。


注意

1. 未經製造商事先書面之核准，不得以任何形式複製本手冊全部或部分內容。
2. 本手冊之內容(包括規格)如有變更，恕不另行通知。
3. 如果您在本手冊中發現了任何有歧意或不正確的說明，請將其(以及封面上的手冊編號)記下來，然後與FUJI聯絡。

安全注意事項

在使用模組之前，請務必仔細閱讀「安全注意事項」。
本手冊中，安全注意事項項目分為「警告」與「小心」兩類。

 **警告**：錯誤處理本裝置將會導致死亡或嚴重的傷害。

 **小心**：錯誤處理本裝置將會導致人體輕微的傷害。

即使是標示為「小心」的某些項目也可能導致發生嚴重的意外。
這兩種安全指示類別皆提供了重要資訊。請務必嚴格遵循這些指示。

警告

◇ 為安全起見，請將緊急停止電路、聯鎖電路或類似電路置於PLC之外。PLC錯誤可能會導致機器發生故障或出現問題。

小心

◇ 在操作時進行程式變更、強制輸出、開機、停機或其他任何動作之前，請確定安全無虞。
錯誤操作將會導致機器發生故障或出現問題。

* 手冊編號在封面上。

出版日期	* 手冊編號	修訂內容
2004年三月	FEH588	第一版
2004年六月	FEH588	位址與指令的表示會根據SX-Programmer standard的升級而改變

前言

安全注意事項

修訂

目錄

頁碼

第 1 節	程式設計概觀	1-1
1-1	程式設計程序	1-1
第 2 節	規格	2-1
2-1	效能規格	2-1
2-1-1	高效能 CPU (SPH300)	2-1
2-1-2	標準 CPU (SPH200)	2-2
2-2	記憶體	2-3
2-2-1	記憶配置表	2-3
2-2-2	記憶體說明	2-8
2-3	輸入 / 輸出位址指派	2-33
2-3-1	位址指派範例	2-33
2-3-2	位址指派慣例	2-33
2-4	工作	2-34
2-4-1	工作規格	2-34
2-4-2	工作的類型與作業	2-34
2-4-3	週期工作作業範例	2-35
2-4-4	事件工作作業的範例	2-36
2-4-5	工作中斷處理	2-37
2-5	程式類型	2-38
2-6	行事曆功能	2-39
2-7	操作流程圖	2-41
第 3 節	指令	3-1
3-1	指令類型	3-1
3-1-1	指令類型	3-1
3-1-2	指令清單	3-2
3-2	順序指令	3-19
3-2-1	基本指令	3-19
3-2-2	程式控制指令	3-32
3-2-3	轉換指令	3-35
3-2-4	算數運算指令	3-42
3-2-5	位元字串運算指令	3-66
3-2-6	選擇及比較指令	3-73
3-2-7	字元字串指令	3-83
3-2-8	時間指令	3-99
3-2-9	原始功能	3-108
3-2-10	IEC 標準功能區塊	3-126
3-2-11	原始功能區塊	3-132

目錄

頁碼

第 4 節	系統定義	4-1
4-1	系統定義摘要	4-1
4-2	在系統定義中設定	4-2
4-3	系統屬性	4-5
4-3-1	系統作業定義	4-5
4-3-2	系統備援定義	4-6
4-3-3	系統軟故障啟動	4-7
4-4	CPU 參數	4-9
4-4-1	CPU 作業定義	4-9
4-4-2	CPU 記憶體大小定義	4-11
4-4-3	I/O 群組設定	4-13
4-4-4	軟故障執行	4-19
4-5	輸入 / 輸出參數	4-22
4-5-1	輸入過濾時間	4-22
4-5-2	輸出保留定義	4-23
4-5-3	系統輸出定義	4-24
第 5 節	CPU 雙工系統	5-1
5-1	雙工模式中的系統作業	5-1
5-1-1	1 對 1 雙工模式	5-1
5-1-2	N 對 1 雙工模式	5-2
5-2	操作及等待 CPU 與效能之間的對換條件	5-3
5-2-1	對換的條件	5-3
5-2-2	雙工模式下的系統效能	5-3
5-2-3	多 CPU 傳遞切換	5-4
5-2-4	資料等化	5-5
5-2-5	操作與等待 CPU 之間對換時的記憶體作業	5-9
5-2-6	CPU 模組 LED 與輸出至顯示系統	5-10
5-2-7	雙工系統的應用程式	5-11
附錄		
附錄 1	設定高效能 CPU Takt 週期	APP.1-1
附錄 2	設定標準 CPU Takt 週期	APP.2-1
附錄 3	存取處理器匯流排	APP.3-1
附錄 4	建立使用者 FB 的程序	APP.4-1
附錄 5	設定字元字串資料	APP.5-1

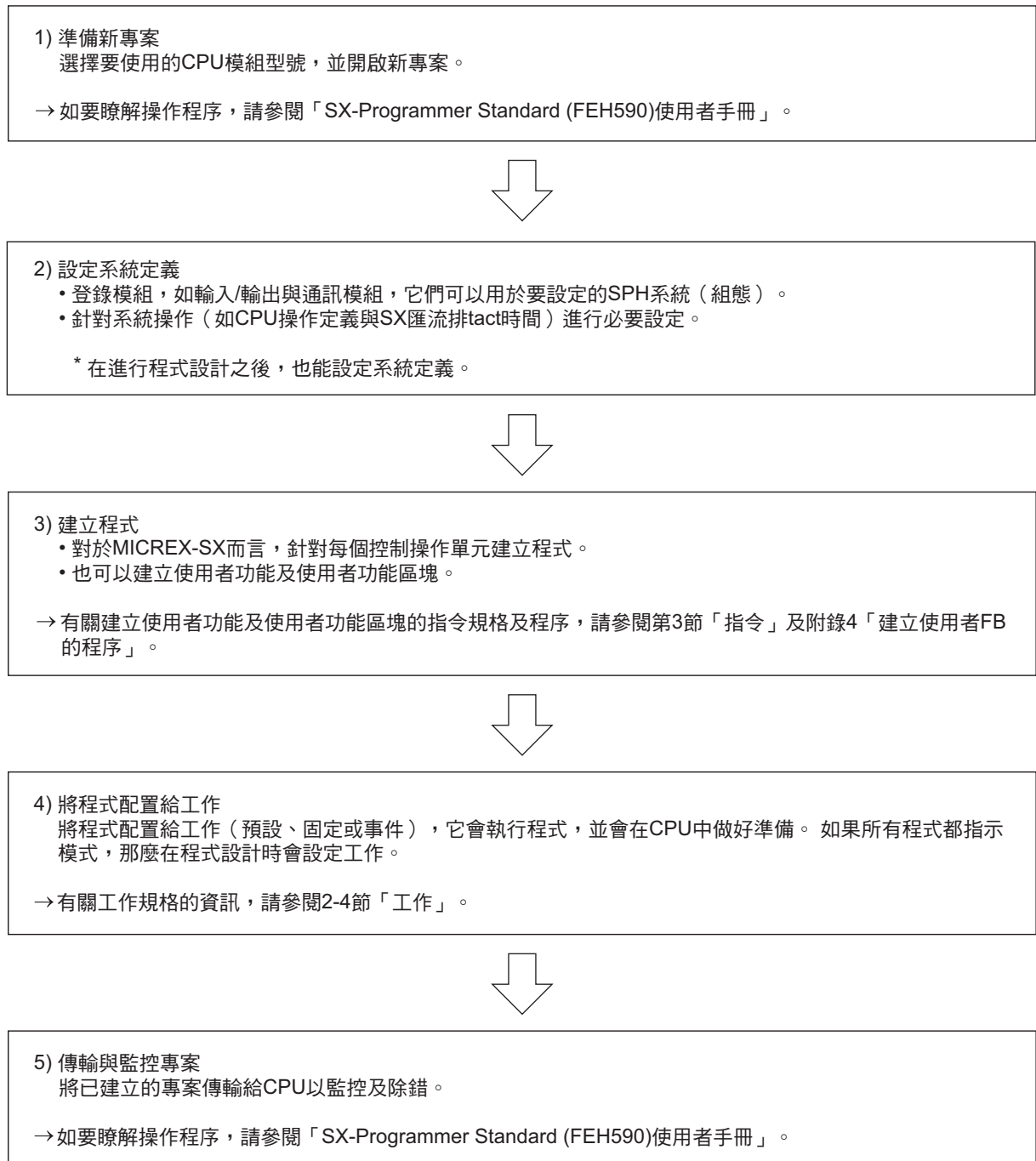
第1節 程式設計概觀

	頁碼
1-1 程式設計程序	1-1

第 1 節 程式設計概觀

1-1 程式設計程序

程式設計的一般程序如下所示。



第2節 規格

	頁碼
2-1 效能規格	2-1
2-1-1 高效能CPU (SPH300)	2-1
2-1-2 標準CPU (SPH200)	2-2
2-2 記憶體	2-3
2-2-1 記憶配置表	2-3
(1) NP1PS-32/NP1PS-32R (高效能CPU)	2-3
(2) NP1PS-74/NP1PS-74R (高效能CPU)	2-4
(3) NP1PS-117/NP1PS-117R (高效能CPU)	2-5
(4) NP1PH-16 (標準CPU)	2-6
(5) NP1PH-08 (標準CPU)	2-7
2-2-2 記憶體說明	2-8
(1) 輸入/輸出記憶體區域 (512字)	2-8
(2) 標準記憶體區域 (M)	2-9
(3) 保留記憶體區域 (L)	2-10
(4) 使用者FB記憶體區域	2-11
(5) 系統FB記憶體區域	2-12
(6) 初始區域	2-13
(7) 系統記憶體區域 (512字)	2-14
2-3 輸入/輸出位址指派	2-33
2-3-1 位址指派範例	2-33
2-3-2 位址指派慣例	2-33
2-4 工作	2-34
2-4-1 工作規格	2-34
2-4-2 工作的類型與作業	2-34
2-4-3 週期工作作業範例	2-35
2-4-4 事件工作作業的範例	2-36
2-4-5 工作中斷處理	2-37
2-5 程式類型	2-38
2-6 行事曆功能	2-39
(1) 行事曆的值範圍	2-39
(2) 行事曆精確度	2-39
(3) 從載入器中監控與設定行事曆時鐘	2-39
(4) 從應用程式中監控與設定行事曆時鐘	2-40
(5) 時間調整功能	2-40
2-7 操作流程圖	2-41

第2節 規格

2-1 效能規格

2-1-1 高效能CPU (SPH300)

項目		規格			備註	
CPU型號		NP1PS-32/ NP1PS-32R	NP1PS-74/ NP1PS-74R	NP1PS-117/ NP1PS-117R		
I/O 記憶體	X, WX, DX	8192點 (包括遠端I/O點)			固定值	
	Y, WY, DY					
標準記憶體	M, WM, DM	高速	2K字		固定值	
		標準	6K字	30K字	126K字	預設值 (註1)
保留記憶體	L, WL, DL	4K字	16K字	32K字	預設值 (註1)	
用於使用者FB的記憶體	F, WF, DF	4K字	16K字	32K字	預設值 (註1)	
用於系統FB的記憶體	邊緣偵測		1024點 (2K字)	4096點 (8K字)	4096點 (8K字)	預設值 (註1)
	計時器	T	512點 (4K字)	2048點 (16K字)	2048點 (16K字)	
	整合計時器	TR	128點 (1K字)	512點 (4K字)	512點 (4K字)	
	計數器	C	256點 (2K字)	1024點 (8K字)	1024點 (8K字)	
	其他		8K字	32K字	32K字	
系統記憶體	SM, WSM, DSM	512字			固定值	
參數	V, WV, DV	-			(註2)	
暫存區	8K字 (使用者規格已停用)			固定值		
工作	循環掃描 (預設工作) : 1個工作 固定工作+事件工作 : 總共4個工作					
可以登錄的程式數目	最多246個程式 (每個工作最多128個程式)					
使用者FB數目	512					
使用者FB的階層數	127階層					
使用者功能數目	512					
使用者功能的階層數	127階層					

注意：1) 透過設定系統定義的CPU記憶體大小定義，可以增加或減少不是位於高速區域內的標準記憶體、保留記憶體、用於使用者FB的記憶體以及用於系統FB的記憶體。此清單列出了預設值。
2) 使用者FB的參數 (V、WV及DV) 會配置給用於使用者FB的記憶體。

2-1 效能規格

2-1-2 標準CPU (SPH200)

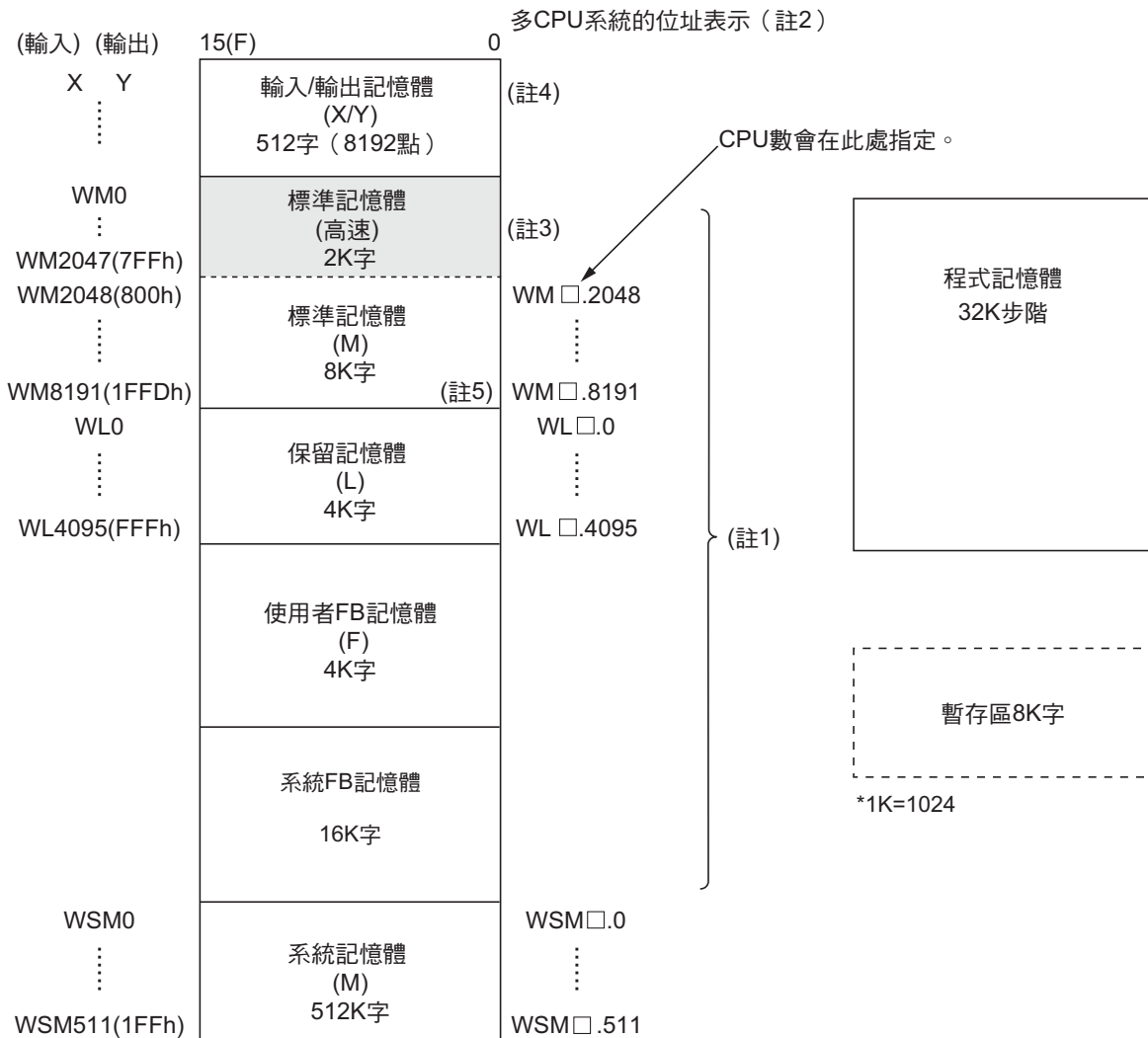
項目		規格		備註	
CPU型號		NP1PH08	NP1PH16		
I/O記憶體	X, WX, DX	8192點 (包括遠端I/O點)		固定值	
	Y, WY, DY				
標準記憶體	M, WM, DM	4K字	8K字	預設值 (註1)	
保留記憶體	L, WL, DL	2K字	4K字	預設值 (註1)	
用於使用者FB的記憶體	F, WF, DF	2K字	4K字	預設值 (註1)	
用於系統FB的記憶體	邊緣偵測		512點 (1K字)	256點 (0.5K字)	預設值 (註1)
	計時器	T	128點 (1K字)	256點 (2K字)	
	整合計時器	TR	32點 (256K字)	64點 (512K字)	
	計數器	C	64點 (512K字)	128點 (1K字)	
	其他		2K字	4K字	
系統記憶體	SM, WSM, DSM	512字			
參數	V, WV, DV	-		固定值	
暫存區	4K字 (使用者規格已停用)			(註2)	
工作	循環掃描 (預設工作) : 1個工作 固定工作+事件工作 : 總共4個工作			固定值	
可以登錄的程式數目	最多64個程式 (每個工作最多64個程式)				
使用者FB數目	256				
使用者FB的階層數	64階層				
使用者功能數目	256				
使用者功能的階層數	64階層				

注意：1) 透過設定系統定義的CPU記憶體大小定義，可以增加或減少不是位於高速區域內的標準記憶體、保留記憶體、用於使用者FB的記憶體以及用於系統FB的記憶體。此清單列出了預設值。

2) 使用者FB的參數 (V、WV及DV) 會配置給用於使用者FB的記憶體。

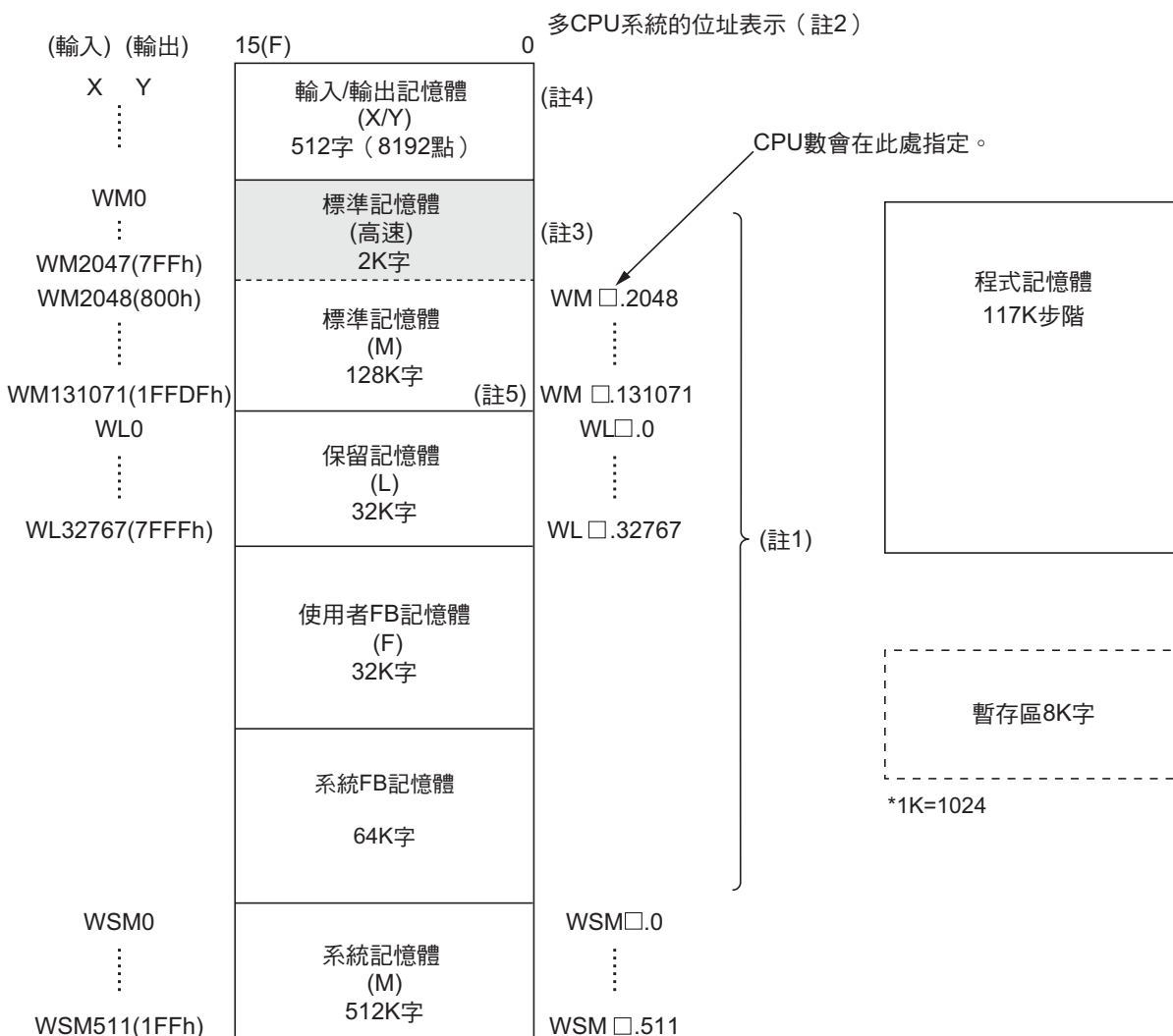
2-2-1 記憶配置表

(1) NP1PS-32/NP1PS-32R (高效能CPU)



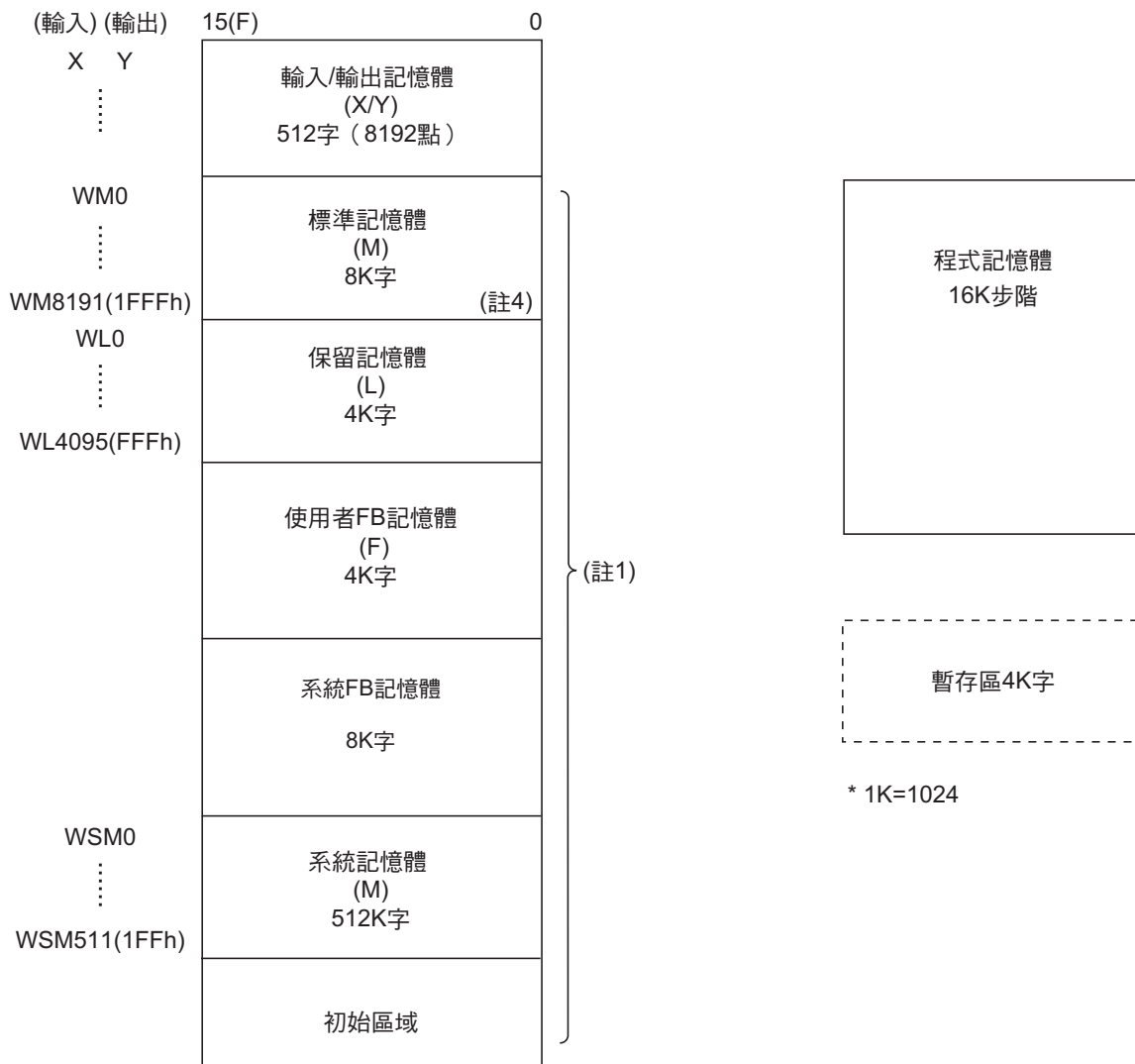
- 注意：1) 標準記憶體 (不包括高速記憶體)、保留記憶體、使用者FB記憶體及系統FB記憶體的大小可能會依載入器設定值而增加或減少。在這些記憶體中，已經儲存了預設值。相關的詳細資訊，請參見「4-4-2定義CPU記憶體大小」。
- 2) 當在多CPU系統中存取其他CPU記憶體時，會在□中指定CPU數。對於存取自己的記憶體而言，這是沒有必要的。
- 3) 標準記憶體 (M) 的前2K字是由以較快速度存取的高速記憶體所組成的。其他CPU無法以通用記憶體的方式來存取此區域。其大小不會改變。
- 4) 如需輸入/輸出位址指派詳細內容，請參閱「2-3輸入/輸出位址指派」。
- 5) 標準記憶體的最後兩個字會由系統保留起來。

(3) NP1PS-117/NP1PS-117R (高效能CPU)



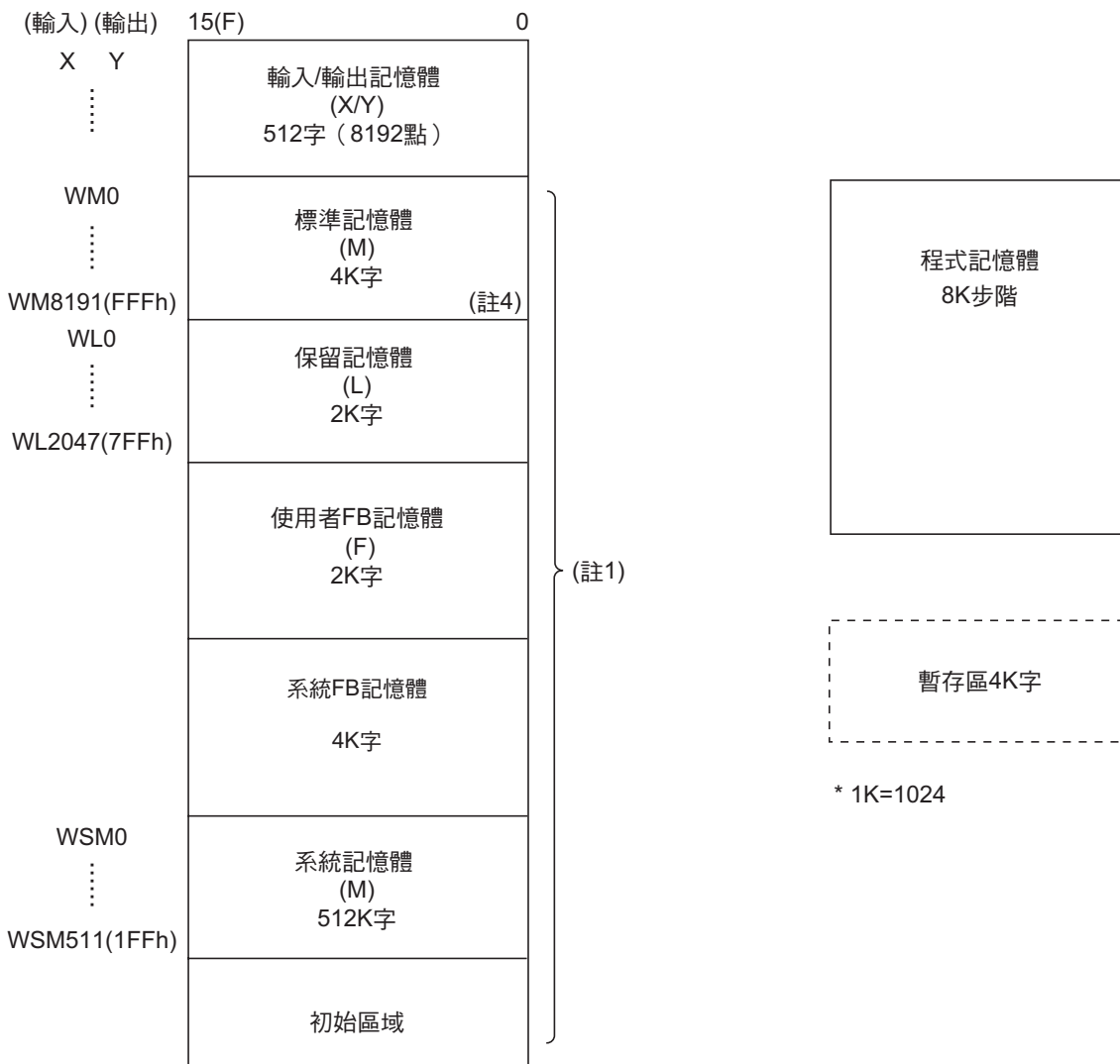
- 注意：1) 標準記憶體 (不包括高速記憶體)、保留記憶體、使用者FB記憶體及系統FB記憶體的大小可能會依載入器設定值而增加或減少。在這些記憶體中，已經儲存了預設值。相關的詳細資訊，請參見「4-4-2定義CPU記憶體大小」。
- 2) 當在多CPU系統中存取其他CPU記憶體時，會在□中指定CPU數。對於存取自己的記憶體而言，這是沒有必要的。
- 3) 標準記憶體 (M) 的前2K字是由以較快速度存取的高速記憶體所組成的。其他CPU無法以通用記憶體的方式來存取此區域。其大小不會改變。
- 4) 如需輸入/輸出位址指派的詳細內容，請參閱「2-3輸入/輸出位址指派」。
- 5) 標準記憶體的最後兩個字會由系統保留起來。

(4) NP1PH-16 (標準CPU)



- 注意：1) 標準記憶體、保留記憶體、使用者FB記憶體、系統FB記憶體及初始區域的大小可能會依載入器設定值而增加或減少。在這些記憶體中，已經儲存了預設值。相關的詳細資訊，請參見「4-4-2定義CPU記憶體大小」。
- 2) 使用標準CPU模組無法建立多系統。
- 3) 如需輸入/輸出位址指派的詳細內容，請參閱「2-3輸入/輸出位址指派」。
- 4) 標準記憶體的最後兩個字會由系統保留起來。

(5) NP1PH-08 (標準CPU)



- 注意： 1) 標準記憶體、保留記憶體、使用者FB記憶體、系統FB記憶體及初始區域的大小可能會依載入器設定值而增加或減少。在這些記憶體中，已經儲存了預設值。相關的詳細資訊，請參見「4-4-2定義CPU記憶體大小」。
- 2) 使用標準CPU模組無法建立多系統。
- 3) 如需輸入/輸出位址指派的詳細內容，請參閱「2-3輸入/輸出位址指派」。
- 4) 標準記憶體的最後兩個字會由系統保留起來。

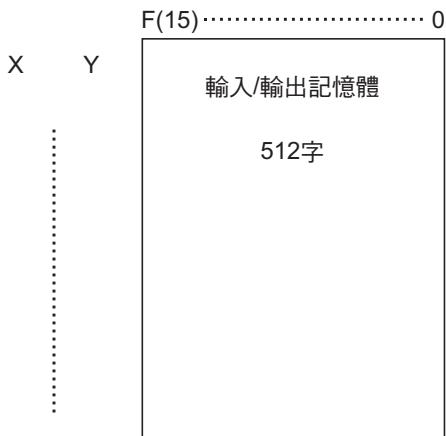
2-2-2 記憶體說明

(1) 輸入/輸出記憶體區域 (512字)

輸入/輸出記憶體區域是一個視窗，CPU與外部裝置之間會透過它來交換資料。它是由輸入裝置（如用來將資料傳送到CPU中的按鈕、開關及感應器）與輸出裝置（如用來顯示程式執行結果的繼電器、螺線管及指示器）所使用的。

重點：

- (1) 輸入由X表示，輸出由Y（前置詞）表示。
X/Y：位元，WX/WY：字，DX/DY：雙字
詳細資訊，請參考「2-3輸入/輸出位址指派」。
- (2) 直接連接到SX匯流排的I/O及遠端I/O如連結等都會被配置到這個區域。
高效能CPU需要81個工作在本區域與外部設備（如POD）之間執行資料讀取或寫入作業的一次循環。建議您在從外部設備讀取或寫入資料到外部設備時存取一次內部記憶體。
- (3) 輸入與輸出無法存在於同一個字中。

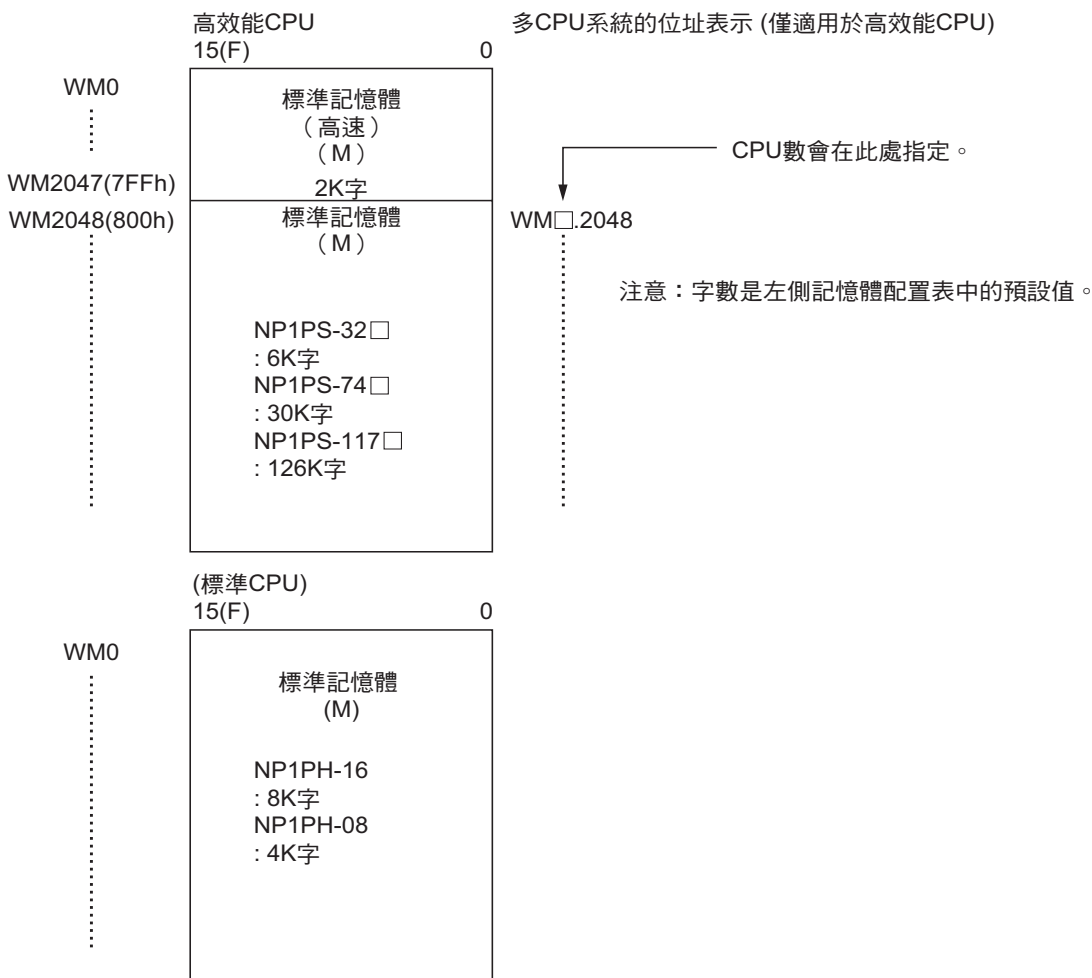


(2) 標準記憶體區域 (M)

標準記憶體區域可做為在PLC內部使用的輔助繼電器。

重點：

- 1) 位址指定是以M (M：位元、WM：字、DM：雙字) 表示的。
 - 2) 當啟動自身的PLC時，指定的記憶體區域會被重設為 (0) 零。
 - 3) 對高效能CPU而言，從標準記憶體上方開始的2K字就是在以高速處理的CPU中資料存取的區域。也就是說，當從外部裝置 (如POD) 存取資料時，作業需要81個工作來執行資料讀取或寫入的一個循環。
 - 4) 在多CPU系統中，其他區域可以從其他任何CPU中以通用記憶體區域的方式存取 (僅適用於高效能CPU及SPS)。
 - 5) 如果將其他區域也考慮進去的話，標準記憶體區域的大小可以修改。請注意，快速存取記憶體的大小是2K字，這在高效能CPU中是固定的，且無法修改 (僅適用於SPH)。
- 關於記憶體大小的修改，請參考「4-4-2 CPU記憶體大小定義」。
- 6) 對高效能CPU而言，不允許連續存取快速存取記憶體區域及介於其他區域之間的界限。例如，陣列或結構不可以跨越界限。



(3) 保留記憶體區域 (L)

這是輔助繼電器的記憶體區域，其用於PLC內部的屬性在停電時仍能保存下來。

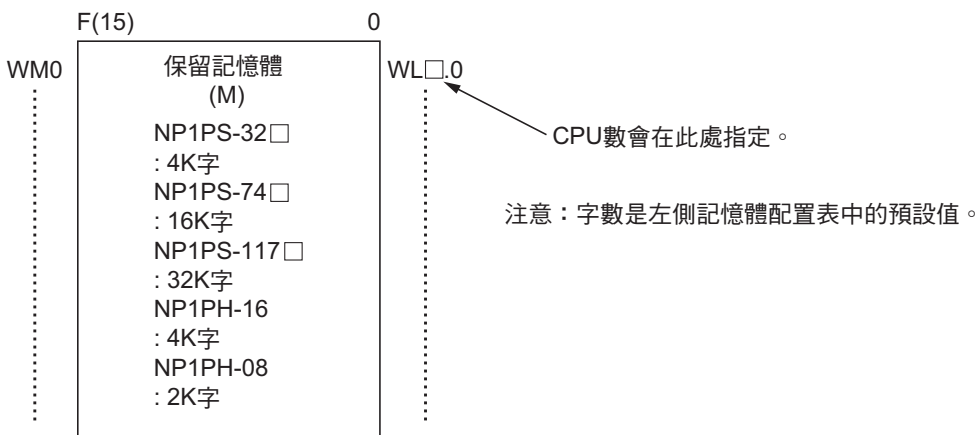
重點：

- (1) 位址規格以L表示。
L：位元，WL：字，DL：雙字
- (2) 以下過程是在冷啟動或熱啟動時執行的。

	冷啟動	熱啟動
保留記憶體	重設為0 (零)	保留舊值
初始化的保留記憶體	寫入指定的初始值	保留舊值

- (3) 當傳輸專案時，您可以選擇是否在傳輸專案時清除區域。如果您選擇「清除」，那麼系統便會冷啟動，如果您選擇「不清除」，系統則會熱啟動。
- (4) 在多CPU系統中，保留記憶體區域可以從其他任何CPU中以通用記憶體區域的方式存取（僅適用於高效能CPU）。
- (5) 如果將其他區域也考慮進去的話，保留記憶體區域的大小可以修改。關於記憶體大小的修改，請參考「4-4-2 CPU記憶體大小定義」（僅適用於SPH）。

多CPU系統的位址表示（僅適用於功能強大的CPU）



注意：當從載入器初始啟動時，系統會冷啟動，而當電源開啟或從載入器啟動時，系統則會熱啟動。

(4) 使用者FB記憶體區域

使用者FB記憶體區域是用來處理在PLC內部使用的每個使用者FB的記憶體區域。

重點：

- (1) 請勿從應用程式或其他載入器中將資料從實例記憶體區域中讀取出來，或將資料寫入到其中。
如果您忽略了此建議，使用者FB可能會無法正確操作。
- (2) 如果將其他區域的大小也考慮進去，此區域是可以修改的。
關於記憶體大小的修改，請參考「4-4-2 CPU記憶體大小定義」。

WF0

⋮

使用者FB記憶體 (F)
NP1PS-32 <input type="checkbox"/> : 4K字
NP1PS-74 <input type="checkbox"/> : 16K字
NP1PS-117 <input type="checkbox"/> : 32K字
NP1PH-16 : 4K字
NP1PH-08 : 2K字

注意：字數是左側記憶體配置表中的預設值。

(5) 系統FB記憶體區域

系統FB記憶體區域是用來處理在PLC內部使用的每個系統FB（如計時器、計數器及不同指令）的記憶體區域。

重點：

- (1) 請勿從應用程式或載入器中將資料從記憶體區域中讀取出來，或將資料寫入到其中。
如果您忽略了此建議，使用者FB可能會無法正確操作。
- (2) 啟動PC時，會執行預先定義的初始動作。（舊值會被保留或重設為0（零）。
請注意，即使是在通常保留舊值的區域內，無論是否傳輸專案，這些值都會被重設為0（零）。
例如）計數器與計時器總數目前的值以及邊緣偵測指令計數器的舊值都會被保留，而計時器目前的值（不包括總數）則會被重設為0（零）。
- (3) 會使用八字/計時器、四字/計數器及二字/邊緣偵測指令計數器。
- (4) 如果將其他區域也考慮進去的話，系統FB記憶體區域的大小可以修改。
關於記憶體大小的修改，請參考「4-4-2 CPU記憶體大小定義」。
- (5) 依照預設值，計時器數、計時器總數、計數器以及邊緣偵測指令計數器已針對每一個CPU預先設定。如有必要，這些數目也可以修改。

	計時器(T)	計時器總數 (TR)	計數器 (C)	邊緣偵測	其他
NP1PS-32	512	128	256	1024	8192 字
NP1PS-74 /117	2048	512	1024	4096	32768 字
NP1PH-16	256	64	128	512	4096 字
NP1PH-08	128	32	64	256	2048 字

要設定計時器、計時器總數、計數器、邊緣偵測及其他系統FB區域，應該符合以下條件。

$$(\text{計時器數}) \times 8\text{字} + (\text{計數器數}) \times 4\text{字} + (\text{邊緣偵測計數器}) \times 2\text{字} + \text{其他} \leq \text{系統FB記憶體區域的預設大小}$$

邊緣偵測
NP1PS-32□: 2K字
NP1PS-74□/117□: 8K字
NP1PH-16: 1K字
NP1PH-08: 0.5K字

計數器
NP1PS-32□: 1K字
NP1PS-74□/117□: 4K字
NP1PH-16: 0.5K字
NP1PH-08: 0.25K字

計時器總數
NP1PS-32□: 1K字
NP1PS-74□/117□: 4K字
NP1PH-16: 0.5K字
NP1PH-08: 0.25K字

計時器
NP1PS-32□: 4K字
NP1PS-74□/117□: 16K字
NP1PH-16: 2K字
NP1PH-08: 1K字

其他
NP1PS-32□: 8K字
NP1PS-74□/117□: 32K字
NP1PH-16: 4K字
NP1PH-08: 2K字

注意：字數是左側記憶體配置表中的預設值。

(6) 初始區域

只有在標準CPU中指定的初始區域才可儲存使用者功能區塊（FB）的初始值及變數。

重點：

- (1) 儲存區域的大小是由底下運算式計算出來的。
 (初始區域) = (使用者FB區域中的字數) x 9/8 + (初始值被設定的變數數目) x 5 (字)
- (2) 預設值已經被預設了，如下所示。

	初始區域（完整）	初始使用者FB值 儲存區域	初始變數值儲存區域
NP1PH-16	7K 字	4608 字	2560 字
NP1PH-08	3K 字	2304 字	768 字

從上表可以看到，初始值能夠被設定的變數數目（預設值）是由底下運算式計算出來的。

NP1PH-16：2560 / 5 = 512（小數點後面的數字都會被捨去） 512

NP1PH-08：768 / 5 = 153（小數點後面的數字都會被捨去） 153

- (3) 使用者FB初始區域的大小需要：預設使用者FB區域的字數 x 9/8

初始區域

初始使用者FB值
 儲存區域
 NP1PH-16:
 4608 字
 (預設值)
 NP1PH-08:
 2304 字
 (預設值)

初始變數值儲存區域
 NP1PH-16:
 2560 字
 (預設值)
 NP1PH-08:
 768 字
 (預設值)

(7) 系統記憶體區域 (512字)

系統記憶體區域會配置給指出了MICREX-SX系列操作狀態或錯誤狀態的旗標。此區域為專用區域。

< 系統記憶體 >

資源會被定義為由一個CPU模組與二或多個I/O模組所設定的一個CPU系統。

WSM0	資源操作狀態
WSM1	資源開關資訊/使用者ROM狀態
WSM2	資源致命錯誤因素
WSM3	未使用
WSM4	資源非致命錯誤因素
WSM5	未使用
WSM6	CPU錯誤因素
WSM7	未使用
WSM8, 9	記憶體錯誤因素
WSM10(Ah) WSM11(Bh)	SX匯流排錯誤因素
WSM12(Ch)	應用程式錯誤因素 (致命錯誤)
WSM13(Dh)	應用程式錯誤因素 (非致命錯誤)
WSM14(Eh) WSM16(10h)	使用者致命錯誤因素0 - 因素47
WSM17(11h)	未使用
WSM18(12h) WSM20(14h)	使用者致命錯誤因素0 - 因素47
WSM21(15h)	未使用
WSM22(16h) WSM29(1Dh)	系統定義錯誤因素
WSM30(1Eh) WSM37(25h)	未使用
WSM38(26h) WSM39(27h)	應用程式錯誤因素
WSM40(28h) WSM41(29h)	未使用
WSM42(2Ah) WSM43(2Bh)	傳播器傳遞
WSM44(2Ch) WSM45(2Dh)	未使用
WSM46(2Eh)	雙工主控傳播器
WSM47(2Fh)	雙工作業模式
WSM48,49(30h,31h)	資源操作/執行模式
WSM50,51(32h,33h)	資源組態/錯誤資訊
WSM52(34h) WSM67(43h)	SX匯流排組態資訊 (SPH系統組態資訊)
WSM68(44h) WSM83(53h)	SX匯流排錯誤資訊 (SPH系統錯誤資訊)
WSM84(54h) WSM99(63h)	SX匯流排連接模組軟故障模式資訊
WSM100(64h) WSM127(7Fh)	未使用

WSM128(80h) WSM135(87h) WSM136(88h) WSM143(8Fh) WSM144(90h) WSM151(97h) WSM152(98h) WSM159(9Fh) WSM160(A0h) WSM167(A7h) WSM168(A8h) WSM175(AFh) WSM176(B0h) WSM183(B7h) WSM184(B8h) WSM191(BFh) WSM192(C0h) WSM199(C7h) WSM200(C8h) WSM207(CFh) WSM208(D0h) WSM215(D7h) WSM216(D8h) WSM223(DFh) WSM224(E0h) WSM231(E7h) WSM232(E8h) WSM239(EFh) WSM240(F0h) WSM247(F7h) WSM248(F8h) WSM255(FFh) WSM256(100h) WSM507(1FBh) WSM508(1FCh) WSM511(1FFh)	遠端I/O master 0 I/O模組組態資訊
	遠端I/O master 0 I/O模組錯誤資訊
	遠端I/O master 1 I/O模組組態資訊
	遠端I/O master 1 I/O模組錯誤資訊
	遠端I/O master 2 I/O模組組態資訊
	遠端I/O master 2 I/O模組錯誤資訊
	遠端I/O master 3 I/O模組組態資訊
	遠端I/O master 3 I/O模組錯誤資訊
	遠端I/O master 4 I/O模組組態資訊
	遠端I/O master 4 I/O模組錯誤資訊
	遠端I/O master 5 I/O模組組態資訊
	遠端I/O master 5 I/O模組錯誤資訊
	遠端I/O master 6 I/O模組組態資訊
	遠端I/O master 6 I/O模組錯誤資訊
	遠端I/O master 7 I/O模組組態資訊
	遠端I/O master 7 I/O模組錯誤資訊
	未使用
	SX匯流排傳送錯誤率資訊

* 圓括號內部是以十六進位來表示的。

1) 資源操作狀態WSM0 (唯讀)

下表顯示了資源 (CPU 模組) 操作狀態及操作模式。

位址	名稱	說明	SPH 300	SPH 200
SM00	執行	當CPU執行時，設定為「開啟」。	○	○
SM01	停止	當CPU當機時，設定為「開啟」。	○	○
SM02	致命狀態	當已發生致命資源錯誤時設定為「開啟」。	○	○
SM03	非致命狀態	當已發生非致命資源錯誤時設定為「開啟」。	○	○
SM04	雙工操作站	當操作CPU在雙工模式下執行時設定為「開啟」。	○	-
SM05	雙工等待站	當等待CPU在雙工模式下執行時設定為「開啟」。	○	-
SM06	1對1雙工	當系統處於1對1雙工模式下時設定為「開啟」。	○	-
SM07	N對1雙工	當系統處於N對1雙工模式下時設定為「開啟」。	○	-
SM08	非自動作業模式	當處於非自動作業模式下時設定為「開啟」。	○	○
SM09	自動作業模式	當處於自動作業模式下時設定為「開啟」。	○	○
SM0A	優先狀態模式	當處於優先狀態模式下時設定為「開啟」。	○	○
SM0B	低電量模式	當處於低電量模式下時設定為「開啟」。	○	○
SM0C	未使用		-	-
SM0D (註)	SX匯流排連接模組軟故障模式	當針對連接到SX匯流排或單獨重設的所有模組執行軟故障時，設定為「開啟」。	○	○
SM0E	處理器匯流排主控	當CPU模組控制處理器匯流排時設定為「開啟」。	○	○
SM0F	SX匯流排主控	當CPU模組控制SX匯流排時設定為「開啟」。	○	○

○: 支援 -: 不支援

非自動作業模式

當系統電源開啟，且CPU模組前面板上的主開關被設定為 "RUN" 或 "TERM" 時，CPU將不會開始作業的模式。這是由CPU模組的系統定義設定的。

自動作業模式

當系統電源開啟，且CPU模組前面板上的主開關被設定為 "RUN" 或 "TERM" 時，CPU將會開始作業的模式。這是由CPU模組的系統定義設定的。此模式會在電源開啟時在系統資源組態中啟用。(依預設值，自動模式是開啟的。)

優先狀態模式

當系統電源開啟，且CPU模組前面板上的主開關被設定為 "RUN" 時，CPU將會開始作業的模式；當系統電源開啟，且開關被設定為 "TERM" 時，CPU將會進入優先狀態 (執行或停止)，這是在優先執行中電源遭到關閉時所建立的。

低電量作業模式

系統電源開啟時，全部記憶體都會被初始化為初始值或全部為零。請注意，既未執行電池連接檢查也沒有執行電壓檢查。此模式會在電源開啟時在系統組態中啟用。當在低電量模式中開啟了優先狀態模式時，CPU會進入自動作業模式。對標準CPU而言，除非插入了使用者ROM卡，否則不會啟用此模式。

注意：<當SM013 (0DH)關閉時 (SX匯流排上的模組對於退化 (degenerate) 來講是無效的) >

對一般模組而言，不會執行軟故障，對於連接到SX匯流排的I/O模組而言，即使使用載入器來啟用，也不會被執行。

<當SM0D開啟時 (SX匯流排上的模組對於退化 (degenerate) 來講是有效的) >

對一般模組而言，會執行軟故障，對於連接到SX匯流排的I/O模組而言，由於已經使用載入器來啟用了，因此會被執行。

2) 資源開關/使用者ROM狀態WSM1 (唯讀)

此區域指出了控制資源的CPU模組開關的狀態。

位址	名稱	說明	SPH 300	SPH 200
SM10 SM13	CPU數	指出使用CPU模組前方面板上的CPU數設定開關設定的4位元數字。允許0-7的數字範圍。	0	-
SM14 SM15	未使用		-	-
SM16	使用者ROM卡連接狀態	1: 已連接 0: 未連接	0 (註1)	0
SM17	使用者ROM卡防寫保護	1: 防寫保護 0: 允許寫入 (當SM16開啟時有效)	0 (註1)	0
SM18	STOP位置	當主開關位於STOP位置時設定為「開啟」。	0	0
SM19	TERM位置 (底部)	當主開關位於TERM位置時設定為「開啟」。(底部)	0	0
SM1A	TERM位置 (上方) (註2、3)	當主開關位於TERM位置時設定為「開啟」。(上方)	0	0
SM1B	RUN位置	當主開關位於RUN位置時設定為「開啟」。	0	0
SM1C SM1F	未使用		-	-

注意：1) 僅適用使用者ROM卡 (CF卡) 適用之機型。

2) 當主開關處於未知狀態時，TERM位置旗標也會開啟。

3) 使用採用了高效能CPU模組的使用者ROM卡，當主開關被設定為UROM_TERM位置時，它會被設定為「開啟」。

3) 資源致命錯誤因素WSM2 (唯讀)

此區域指出將會停止資源 (單一CPU系統) 的致命錯誤因素。

位址	名稱	說明	SPH 300	SPH 200
SM20	CPU錯誤	當已在CPU模組中發生致命錯誤時，設定為「開啟」。	0	0
SM21	電源供應錯誤	當已經發生電源關閉狀況時，設定為「開啟」。	0	0
SM22	記憶體錯誤	當CPU模組中的記憶體中已經發生錯誤時，設定為「開啟」。	0	0
SM23	SX匯流排錯誤	當發生SX匯流排錯誤時，例如纜線或迴接插頭中斷連接時，設定為「開啟」。	0	0
SM24	應用程式錯誤	當已在應用程式或系統定義中發現錯誤時，設定為「開啟」。	0	0
SM25	I/O模組錯誤	當已在由自身CPU模組控制的任一I/O模組中發生錯誤，且軟故障也已經「停用」時，設定為「開啟」。 當軟故障已經「啟用」時，即使I/O模組中已經發生錯誤，整個系統還會繼續正常作業。	0	0
SM26	一般模組錯誤	當連接到SX匯流排的任何一個一般模組 (不包括自身模組) 中已經發生錯誤時，設定為「開啟」。	0	0
SM27	繼電器切換錯誤	當在雙工作業模式下無法執行繼電器切換時，設定為「開啟」。	0	-
SM28 SM29	未使用		-	-
SM2A	遠端I/O主控板上的遠端I/O模組錯誤	當系統由於遠端I/O裝置或模組上的錯誤而關閉時，設定為「開啟」。	-	-
SM2B	驅動程式錯誤		-	-
SM2C	未使用		-	-
SM2D	其他硬體錯誤	當CPU數選擇開關中已經發生錯誤時，設定為「開啟」。	0	0
SM2E	未使用		-	-
SM2F	使用者致命錯誤	當使用者程式開啟其中一個使用者致命錯誤旗標 (SM140至SM16F) 時，設定為「開啟」。	0	0

4) 資源非致命錯誤因素WSM4 (唯讀)

此區域指出了允許資源繼續處理的錯誤因素。

位址	名稱	說明	SPH 300	SPH 200
SM40 SM41	未使用		-	-
SM42	記憶體錯誤	當自身的CPU模組的記憶體中已經發生錯誤時，設定為「開啟」。	○	○
SM43	SX匯流排錯誤	當SX匯流排中已經發生錯誤時，設定為「開啟」。	○	○
SM44	應用程式錯誤	當已在應用程式或系統定義中發現錯誤時，設定為「開啟」。	○	○
SM45	I/O模組錯誤	當已在由自身CPU模組控制的任一I/O模組中發生錯誤，且軟故障也已經「啟用」時，設定為「開啟」。(註)	○	○
SM46	一般模組錯誤 (註*)	當連接到SX匯流排的任一I/O模組(不包括自身CPU)中已經發生錯誤時，設定為「開啟」。	○	○
SM47 SM49	未使用		-	-
SM4A	遠端I/O模組錯誤	當遠端I/O單元或模組已發生錯誤時，設定為「開啟」。	-	-
SM4B	未使用		-	-
SM4C	使用者ROM卡 - CPU不相符	當使用者ROM卡的內容與CPU的內容不符時，設定為「開啟」。驗證目標是系統定義、專案與密碼。	○ (註1)	-
SM4D	其他硬體錯誤	當任一主開關或載入器/一般目的通訊選擇開關中發生錯誤時，設定為「開啟」。如果在主開關中偵測到錯誤，CPU模組會藉由假設已啟用「TERM」來進行作業。如果在載入器/一般目的通訊選擇開關中偵測到了錯誤，它會假設已經選擇了載入器這一邊。	○	○
SM4E	電池錯誤	當資料備份電池的電壓降低低於臨界值或電池電量已經耗盡時，設定為「開啟」。	○	○
SM4F	使用者非致命錯誤	當使用者程式開啟其中一個使用者非致命錯誤旗標(SM180至SM20F)時，設定為「開啟」。	○	○

注意：1) 僅限使用者ROM卡(CF卡)適用之機型

* 一般模組就是指那些已連接到SX匯流排但卻不佔用I/O區域的模組(例如CPU模組、通訊模組等)。

5) CPU錯誤因素WSM6 (唯讀) (不受SPS支援)

位址	名稱	說明
SM60	算數處理器錯誤	CPU模組中算數LSI中的硬體錯誤
SM61	OS處理器錯誤	CPU模組中OS控制LSI中的硬體錯誤
SM62 SM6F	未使用	

6) 記憶體錯誤因素WSM8、WSM9 (唯讀) (不受SPS支援)

位址	名稱	說明	程度
SM80	系統ROM錯誤	當CPU模組中的系統ROM中已經發生錯誤時，設定為「開啟」。	致命錯誤
SM81	系統RAM錯誤	當CPU模組中的系統RAM中已經發生錯誤時，設定為「開啟」。	致命錯誤
SM82	應用程式ROM錯誤	當CPU模組中的使用者程式ROM中已經發生錯誤時，設定為「開啟」。	致命錯誤 (註1)
SM83	應用程式RAM錯誤	當CPU模組中的使用者程式RAM中已經發生錯誤時，設定為「開啟」。	致命錯誤
SM84 SM8E	未使用		
SM8F	記憶體備份錯誤	當未保留停電時間資料時，設定為「開啟」。	致命錯誤 (註2)
SM90 SM9E	未使用		
SM9F	記憶體備份錯誤	當未保留停電時間資料時，設定為「開啟」。可以使用應用程式來清除錯誤情形。	非致命錯誤 (註2)

注意：1) 當使用者ROM卡中已經發生錯誤時，設定為「開啟」。

2) 對於高效能CPU而言，當記憶體備份錯誤已經發生時，視模組版本而定，位元會設定為「開啟」。

V**.25之前版本：SM8F V10.30或更新版本：SM9F

已經發生記憶體錯誤之後的系統作業

任何記憶體備份錯誤都會將整個使用者記憶體區域重設為0(零)。請注意在大多數情況下，由於在發生硬體錯誤時SM8.0至SM8.3會被設定為「開啟」，因此，循環電源可能會導致記憶體錯誤重複發生。

7) SX匯流排錯誤因素WSM10 (Ah)，WSM11 (Bh)

位址	名稱	說明	程度	SPH 300	SPH 200
SM100	SX匯流排LSI錯誤	當控制SX匯流排的LSI中已經發生錯誤時，設定為「開啟」。	致命錯誤		○
SM101	站台數目雙重指定	當相同的SX匯流排站台數目被指定給SPH系統中的一個以上的模組時，設定為「開啟」。	致命錯誤		○
SM102	超出模組計數	當連接到SX匯流排的模組數超過254時，設定為「開啟」。	致命錯誤		○
SM103 SM10C	未使用			-	-
SM10D	SX匯流排傳送錯誤	當經由SX匯流排傳送過程中發生錯誤時，設定為「開啟」。	致命錯誤		○
SM10E	處理器匯流排存取錯誤	當存取處理器匯流排時發生錯誤時，設定為「開啟」。 (存取的錯誤要歸因於自身的CPU模組)	致命錯誤		○
SM10F	I/O重新整理減慢	當SX匯流排中的輸入/輸出資料已經更新超過128毫秒的時間時，設定為「開啟」。	致命錯誤		○
SM110 SM11D	未使用			-	-
SM11E	處理器匯流排存取錯誤	當處理器匯流排中發生錯誤時設定為「開啟」(錯誤起因在於存取的模組)。或許可以使用應用程式來清除錯誤情形。	非致命錯誤	○	○
SM11F	未使用			-	-

8) 應用程式錯誤因素WSM12 (Ch) , WSM13 (Dh) (唯讀)

位址	名稱	說明	程度	SPH 300	SPH 200
SM120	系統定義錯誤	當在系統定義中發現錯誤時，設定為「開啟」。	致命錯誤		○
SM121	應用程式錯誤	當在「應用程式」定義中發現錯誤時，設定為「開啟」。	致命錯誤		○
SM122 SM12F	未使用			-	-
SM130	未使用			-	-
SM131	應用程式錯誤	當已在「應用程式」中發現錯誤時，設定為「開啟」。	非致命錯誤		○
SM132 SM13F	未使用			-	-

9) 使用者致命錯誤WSM14 (Eh) 至 WSM16 (10h)

位址	名稱	說明	SPH 300	SPH 200
SM140 SM14F	使用者致命錯誤因素0 使用者致命錯誤因素15	致命錯誤已經發生，且當這些位元中的任何一個被應用程式設定為「開啟」時，CPU都會停止。		○
SM150 SM15F	使用者致命錯誤因素16 使用者致命錯誤因素31			
SM160 SM16F	使用者致命錯誤因素32 使用者致命錯誤因素47			

10) 使用者非致命錯誤WSM18 (12h) 至WSM20 (14h)

位址	名稱	說明	SPH 300	SPH 200
SM180 SM18F	使用者非致命錯誤因素0 使用者非致命錯誤因素15	非致命錯誤已經發生，且當這些位元中的任何一個被應用程式設定為「開啟」時，CPU都會繼續執行。 如果應用程式將位元設定從「開啟」改為「關閉」，系統將會從非致命錯誤狀態中恢復。		○
SM190 SM19F	使用者非致命錯誤因素16 使用者非致命錯誤因素31			
SM200 SM20F	使用者非致命錯誤因素32 使用者非致命錯誤因素47			

11) 系統定義錯誤因素SM22 (16h) 至SM29 (1Dh) (唯讀)

位址	名稱	說明	程度	SPH 300	SPH 200
SM220	未使用			-	-
SM221	系統定義錯誤	當CPU模組中系統定義的內容與實際系統組態不符時，設定為「開啟」。	致命錯誤	○	○
SM222	系統作業定義錯誤	當系統中「工作」時間被設定為0.5毫秒，且在此系統中二或多個一般模組在一個SPH系統中連接起來時，設定為「開啟」。	致命錯誤	○	○
SM223	系統DO選擇錯誤	當在系統DO（輸出）中定義的SX匯流排直接連接模組不是數位輸出模組時，設定為「開啟」。	致命錯誤	○	○
SM224	雙工選擇錯誤	當在等值範圍規格中發現錯誤時，設定為「開啟」。	致命錯誤	○	-
SM225	軟故障啟動選擇錯誤	如果任一模組都不適用於系統中的軟故障，那麼當啟用軟故障啟動時，設定為「開啟」。	致命錯誤	○	○
SM226 SM229	未使用			-	-
SM22A	CPU作業定義錯誤	當CPU模組中的開關設定與系統定義中的CPU數設定不同時，設定為「開啟」。	致命錯誤	○	○
SM22B	CPU記憶體界限定義錯誤	當應用程式使用的記憶體空間超過了總記憶體容量時，設定為「開啟」。	致命錯誤	○	○
SM22C SM22F	未使用			-	-
SM230	CPU I/O群組定義錯誤（針對預設工作）	當針對輸出模組定義輸入模組時，設定為「開啟」。	致命錯誤	○	○
SM231	CPU I/O群組定義錯誤（針對階層0工作）				
SM23	CPU I/O群組定義錯誤（針對階層1工作）				
SM233	CPU I/O群組定義錯誤（針對階層2工作）				
SM234	CPU I/O群組定義錯誤（針對階層3工作）				
SM235	直接I/O連接軟故障定義錯誤	當針對除輸入/輸出模組之外的模組設定軟故障時，設定為「開啟」。	致命錯誤	○	○
SM236	遠端I/O master 0軟故障定義錯誤	當在軟故障定義中發現錯誤時，設定為「開啟」。	致命錯誤	○	○
SM237	遠端I/O master 1軟故障定義錯誤				
SM238	遠端I/O master 2軟故障定義錯誤				
SM239	遠端I/O master 3軟故障定義錯誤				
SM23A	遠端I/O master 4軟故障定義錯誤				
SM23B	遠端I/O master 5軟故障定義錯誤				
SM23C	遠端I/O master 6軟故障定義錯誤				
SM23D	遠端I/O master 7軟故障定義錯誤				
SM23E SM23F	未使用			-	-

SM240	I/O模組保持定義錯誤	當針對除輸出模組或為系統DO設定的輸出模組之外的模組定義保持時，設定為「開啟」。	致命錯誤	○	○
SM241	I/O初始錯誤	當在連接到SX匯流排的模組的作業設定中發現錯誤時，設定為「開啟」。	致命錯誤	○	○
SM242 SM24F	未使用			-	-
SM250	遠端I/O master 0初始錯誤	當在遠端I/O 主機初始中發現錯誤時，設定為「開啟」。	致命錯誤	○	○
SM251	遠端I/O master 1初始錯誤				
SM252	遠端I/O master 2初始錯誤				
SM253	遠端I/O master 3初始錯誤				
SM254	遠端I/O master 4初始錯誤				
SM255	遠端I/O master 5初始錯誤				
SM256	遠端I/O master 6初始錯誤				
SM257	遠端I/O master 7初始錯誤				
SM258 SM25F	未使用			-	-
SM260	處理器連結0初始錯誤	當在P/PE-link/FL-net初始中發現錯誤時，設定為「開啟」。處理器連結1與連結號碼「9」的模組相對應，處理器連結0則與行號「8」的模組相對應。	致命錯誤	○	○
SM261	處理器連結1初始錯誤				
SM262 SM29F	未使用			-	-

注意：系統定義錯誤因素也包括在正常作業期間沒有發生的錯誤。

12) 應用程式錯誤因素WSM38 (26h) , WSM39 (27h)

位址	名稱	說明	程度	SPH 300	SPH 200
SM380	應用程式WDT錯誤	當預設工作的執行時間超過了監視器 (watchdog) 計時器的預設值時，設定為「開啟」。	致命錯誤	○	○
SM381	應用程式執行錯誤	當在導致「超過暫存區大小」的使用者程式執行期間發生錯誤時，設定為「開啟」。	致命錯誤	○	○
SM382 SM38A	未使用			-	-
SM38B	FB實例設定錯誤	當未找到指定的儲存位址時，設定為「開啟」。	致命錯誤	○	○
SM38C	初始設定值錯誤	當預設值超過儲存區域的定義範圍時，設定為「開啟」。	致命錯誤	○	○
SM38D	SFM界限定義錯誤	當定義了大於最大SFM容量值的大小時，設定為「開啟」。	致命錯誤	○	○
SM38E	程式指令錯誤	當在程式指令中發現錯誤時，設定為「開啟」。	致命錯誤	○	○
SM38F	工作登錄錯誤	當發現工作登錄錯誤時設定為「開啟」。	致命錯誤	○	○
SM390	遺失階層0工作	當遺失工作時，設定為「開啟」。 可以使用應用程式來清除錯誤情形。	非致命錯誤	○	○
SM391	遺失階層1工作				
SM392	遺失階層2工作				
SM393	遺失階層3工作				
SM394	階層0工作減慢	當程式執行延遲，且沒有維持預先定義的週期時間時，設定為「開啟」。	非致命錯誤	○	○
SM395	階層1工作減慢				
SM396	階層2工作減慢				
SM397	階層3工作減慢				
SM398 SM39E	未使用			-	-
SM39F	工作週期監控錯誤	當「工作」週期與系統定義不符時，設定為「開啟」。	非致命錯誤	○	○

注意：系統定義錯誤因素也包括在正常作業期間沒有發生的錯誤，這是因為載入器禁止這些錯誤發生（例如，編譯檢查）。

13) 傳播器傳遞WSM42 (2Ah) , SM43 (2Bh)

位址	名稱	說明	SPH 300	SPH 200
SM420	初始旗標	在下載程式之後第一次啟動且在初始啟動（冷啟動）時，設定為「開啟」。在作業期間，此旗標絕不會被設定為「關閉」。	○	○
SM421	關閉電源旗標	當在之前作業階段中已發生電源關閉狀況時，設定為「開啟」。	○	○
SM422 SM42D	未使用		-	-
SM42E	虛擬模組旗標	當已在一個SPH系統中安裝了多個虛擬模組時，設定為「開啟」。	○	○
SM42F	處理器匯流排存取停用旗標	當處理器匯流排停用時設定為「開啟」。	○	○
SM430	階層0啟動旗標	在第一次執行階層0工作期間設定為開啟。	○	○
SM431	階層1啟動旗標	在第一次執行階層1工作期間設定為開啟。	○	○
SM432	階層2啟動旗標	在第一次執行階層2工作期間設定為開啟。	○	○
SM433	階層3啟動旗標	在第一次執行階層3工作期間設定為開啟。	○	○
SM434 SM43E	未使用		-	-
SM43F	預設工作啟動旗標	在第一次執行預設工作期間設定為開啟。	○	○

14) 雙工傳播器傳遞WSM46 (2Eh) , 雙工作業模式WSM47 (2Fh) (唯讀) (不受SPH200X支援)

位址	名稱	說明
SM460	雙工持續啟動旗標	在雙工模式下的作業期間，且當作業系統切換為等待時，設定為「開啟」。 (從等待模式改變為執行模式的CPU)
SM461 SM46F	未使用	
SM470 SM473	雙工邏輯CPU數	指出雙工作業模式下的4位元CPU邏輯數(0-7)。 可讓您識別由等待CPU所接管的預設操作CPU。除「雙工」模式之外，其他模式中都不會定義它。
SM474 SM477	未使用	
SM478	雙工傳播器傳遞模式0	在雙工模式下，當在作業期間已針對一對CPU 0與1啟用傳播器傳遞時，設定為「開啟」。
SM479	雙工傳播器傳遞模式1	在雙工模式下，當在作業期間已針對一對CPU 2與3啟用傳播器傳遞時，設定為「開啟」。
SM47A	雙工傳播器傳遞模式2	在雙工模式下，當在作業期間已針對一對CPU 4與5啟用傳播器傳遞時，設定為「開啟」。
SM47B	雙工傳播器傳遞模式3	在雙工模式下，當在作業期間已針對一對CPU 6與7啟用傳播器傳遞時，設定為「開啟」。
SM47C SM47F	未使用	

15) 資源組態/作業資訊WSM48 (30h) , WSM49 (31h) (唯讀) (僅適用於SPH300)

此資訊可以用來識別雙工或單一模式下系統 (CPU模組) 作業的目前狀態。資源組態資訊只能用於雙工模式。

當與資源作業/錯誤資訊關聯的位元 (WSM50 (32) , WSM51 (33)) 被設定為「開啟」之後, 底下列出的狀態類型是有效的。

< 在雙工模式下 >

資源作業資訊	資源執行資訊	資源狀態
關閉	關閉	等待CPU停止
開啟	關閉	操作CPU停止
開啟	開啟	操作CPU執行
關閉	開啟	等待CPU執行

< 資源作業資訊 >

位址	名稱	說明
SM480	操作CPU 0執行	當操作CPU在雙工模式下 (在除「雙工」模式之外的其他模式下, 未定義) 執行時, 設定為「開啟」。
SM481	操作CPU 1執行	
SM482	操作CPU 2執行	
SM483	操作CPU 3執行	
SM484	操作CPU 4執行	
SM485	操作CPU 5執行	
SM486	操作CPU 6執行	
SM487	操作CPU 7執行	
SM488 SM48F	未使用	

< 資源執行資訊 >

位址	名稱	說明
SM490	CPU 0 執行	當連接到SX匯流排的關聯CPU模組執行時, 設定為「開啟」。
SM491	CPU 1 執行	
SM492	CPU 2 執行	
SM493	CPU 3 執行	
SM494	CPU 4 執行	
SM495	CPU 5 執行	
SM496	CPU 6 執行	
SM497	CPU 7 執行	
SM498 SM49F	未使用	

16) 資源組態/錯誤資訊WSM50 (32h) , WSM51 (33h) (唯讀)

此資訊可藉由使用應用程式來識別其他資源 (CPU模組) 的狀態。

資源作業資訊	資源錯誤資訊	資源狀態
關閉	關閉	不存在
開啟	關閉	正常 (執行或停止)
開啟	開啟	非致命錯誤 (執行或停止)
關閉	開啟	致命錯誤 (停止或中斷)

< 資源組態資訊 >

位址	名稱	說明	SPH 300	SPH 200
SM500	CPU 0組態	當已經找到連接至SX匯流排的關聯CPU模組，且資源作業狀態為「正常」或「非致命錯誤」時，設定為「開啟」。 注意：對於SPH 200而言，只有CPU0才是目標。	○	○
SM501	CPU 1組態			
SM502	CPU 2組態			
SM503	CPU 3組態			
SM504	CPU 4組態			
SM505	CPU 5組態			
SM506	CPU 6組態			
SM507	CPU 7組態			
SM508 SM50F	未使用		-	-

< 資源錯誤資訊 >

位址	名稱	說明	SPH 300	SPH 200
SM510	CPU 0錯誤	當已經找到連接至SX匯流排的關聯CPU模組，且資源作業狀態為「正常」或「非致命錯誤」時，設定為「開啟」。 注意：對於SPH 200而言，只有CPU0才是目標。	○	○
SM511	CPU 1錯誤			
SM512	CPU 2錯誤			
SM513	CPU 3錯誤			
SM514	CPU 4錯誤			
SM515	CPU 5錯誤			
SM516	CPU 6錯誤			
SM517	CPU 7錯誤			
SM518 SM51F	未使用		-	-

17) SX匯流排組態資訊WSM52 (34h) 至wsm67 (43h) (唯讀)

當SX匯流排中存在模組且正常執行中或存在非致命錯誤時，此模組的SX站台數位元會被設定為「開啟」。模組是正常還是處於非致命錯誤狀態下是透過組態錯誤資訊項目的結合來識別的。

SX匯流排組態資訊	SX匯流排錯誤資訊	模組狀態
關閉	關閉	不存在
開啟	關閉	正常
開啟	開啟	非致命錯誤
關閉	開啟	致命錯誤或連接中斷

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0 ←位元位址
WSM52(34h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	← 未使用
WSM53(35h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM54(36h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM55(37h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM56(38h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM57(39h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM58(3Ah)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM59(3Bh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112
WSM60(3Ch)	143	142	141	140	139	138	137	136	135	134	133	132	131	130	129	128
WSM61(3Dh)	159	158	157	156	155	154	153	152	151	150	149	148	147	146	145	144
WSM62(3Eh)	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160
WSM63(3Fh)	191	190	189	188	187	186	185	184	183	182	181	180	179	178	177	176
WSM64(40h)	207	206	205	204	203	202	201	200	199	198	197	196	195	194	193	192
WSM65(41h)	223	222	221	220	219	218	217	216	215	214	213	212	211	210	209	208
WSM66(42h)	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224
WSM67(43h)		254	253	252	251	250	249	248	247	246	245	244	243	242	241	240

18) SX匯流排錯誤資訊WSM68 (44h) 至WSM83 (53h) (唯讀)

當SX匯流排中存在模組，且受致命或非致命錯誤影響的時候，與此模組的SX匯流排站數相對應位元會被設定為「開啟」。

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM68(44h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		←未使用
WSM69(45h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM70(46h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM71(47h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM72(48h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM73(49h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM74(4Ah)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM75(4Bh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	
WSM76(4Ch)	143	142	141	140	139	138	137	136	135	134	133	132	131	130	129	128	
WSM77(4Dh)	159	158	157	156	155	154	153	152	151	150	149	148	147	146	145	144	
WSM78(4Eh)	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	
WSM79(4Fh)	191	190	189	188	187	186	185	184	183	182	181	180	179	178	177	176	
WSM80(50h)	207	206	205	204	203	202	201	200	199	198	197	196	195	194	193	192	
WSM81(51h)	223	222	221	220	219	218	217	216	215	214	213	212	211	210	209	208	
WSM82(52h)	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	
WSM83(53h)		254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	

19) SX匯流排連接模組軟故障資訊WSM84 (54h) 至WSM99 (63h) (唯讀)

當連接至SX匯流排的任一模組的軟故障或個別重設無法完成時，此模組的SX站台數位元會被設定為「開啟」。

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM84(54h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		←未使用
WSM85(55h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM86(56h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM87(57h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM88(58h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM89(59h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM90(5Ah)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM91(5Bh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	
WSM92(5Ch)	143	142	141	140	139	138	137	136	135	134	133	132	131	130	129	128	
WSM93(5Dh)	159	158	157	156	155	154	153	152	151	150	149	148	147	146	145	144	
WSM94(5Eh)	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	
WSM95(5Fh)	191	190	189	188	187	186	185	184	183	182	181	180	179	178	177	176	
WSM96(60h)	207	206	205	204	203	202	201	200	199	198	197	196	195	194	193	192	
WSM97(61h)	223	222	221	220	219	218	217	216	215	214	213	212	211	210	209	208	
WSM98(62h)	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	
WSM99(63h)		254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	

20) 遠端I/O master 0 I/O模組組態/錯誤資訊WSM128 (80h) 至WSM143 (8Fh) (唯讀)

當存在位於遠端I/O主機0控制之下的模組是正常或處於非致命錯誤狀態下時，相關模組的SX站台數位元會被設定為「開啟」。

遠端組態資訊	遠端錯誤資訊	模組狀態
關閉	關閉	不存在
開啟	關閉	正常
開啟	開啟	非致命錯誤
關閉	開啟	致命錯誤或連接中斷

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM128(80h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM129(81h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM130(82h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM131(83h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM132(84h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM133(85h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM134(86h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM135(87h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

當存在位於遠端I/O主機0控制之下的模組處於致命或非致命錯誤狀態下時，與模組的遠端站台數相對應的位元會被設定為「開啟」。

<錯誤資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM136(88h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM137(89h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM138(8Ah)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM139(8Bh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM140(8Ch)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM141(8Dh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM142(8Eh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM143(8Fh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

第21) 段到第27) 段的解釋與第20) 段相同。

21) 遠端I/O 主機 1 I/O模組組態/錯誤資訊WSM144 (90h) 至WSM159 (9Fh) (唯讀)

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM144(90h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM145(91h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM146(92h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM147(93h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM148(94h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM149(95h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM150(96h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM151(97h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<錯誤資訊>

WSM152(98h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM153(99h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM154(9Ah)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM155(9Bh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM156(9Ch)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM157(9Dh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM158(9Eh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM159(9Fh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

22) 遠端I/O 主機 2 I/O模組組態/錯誤資訊WSM160 (A0h) 至WSM175 (AFh) (唯讀)

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM160(A0h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM161(A1h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM162(A2h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM163(A3h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM164(A4h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM165(A5h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM166(A6h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM167(A7h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<錯誤資訊>

WSM168(A8h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM169(A9h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM170(AAh)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM171(ABh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM172(ACCh)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM173(ADh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM174(AEh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM175(AFh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

23) 遠端I/O 主機 3 I/O模組組態/錯誤資訊WSM176 (B0h) 至WSM191 (BFh) (唯讀)

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM176(B0h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM177(B1h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM178(B2h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM179(B3h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM180(B4h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM181(B5h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM182(B6h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM183(B7h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<錯誤資訊>

WSM184(B8h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM185(B9h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM186(9Ah)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM187(BBh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM188(BCh)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM189(BDh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM190(BEh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM191(BFh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

24) 遠端I/O 主機 4 I/O模組組態/錯誤資訊WSM192 (C0h) 至WSM207 (CFh) (唯讀)

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM192(C0h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM193(C1h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM194(C2h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM195(C3h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM196(C4h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM197(C5h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM198(C6h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM199(C7h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<錯誤資訊>

WSM200(C8h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM201(C9h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM202(CAh)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM203(CBh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM204(CCh)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM205(CDh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM206(CEh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM207(CFh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

25) 遠端 I/O master 5 I/O 模組組態/錯誤資訊 WSM208 (D0h) 至 WSM223 (DFh) (唯讀)

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM208(D0h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM209(D1h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM210(D2h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM211(D3h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM212(D4h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM213(D5h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM214(D6h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM215(D7h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<錯誤資訊>

WSM216(D8h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM217(D9h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM218(DAh)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM219(DBh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM220(DCh)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM221(DDh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM222(DEh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM223(DFh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

26) 遠端 I/O 主機 6 I/O 模組組態/錯誤資訊 WSM224 (E0h) 至 WSM239 (EFh) (唯讀)

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM224(E0h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM225(E1h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM226(E2h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM227(E3h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM228(E4h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM229(E5h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM230(E6h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM231(E7h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<錯誤資訊>

WSM232(E8h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM233(E9h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM234(EAh)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM235(EBh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM236(ECh)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM237(EDh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM238(EEh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM239(EFh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

27) 遠端I/O 主機 7 I/O模組組態/錯誤資訊WSM240 (F0h) 至WSM255 (FFh) (唯讀)

<組態資訊>

字位址 ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←位元位址
WSM240(F0h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM241(F1h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM242(F2h)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM243(F3h)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM244(F4h)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM245(F5h)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM246(F6h)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM247(F7h)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<錯誤資訊>

WSM248(F8h)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM249(F9h)	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM250(FAh)	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM251(FBh)	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM252(FCh)	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM253(FDh)	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM254(FEh)	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM255(FFh)	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

28) SX匯流排傳送錯誤率資訊WSM508 (1FCh) 至WSM511 (1FFh) (唯讀)

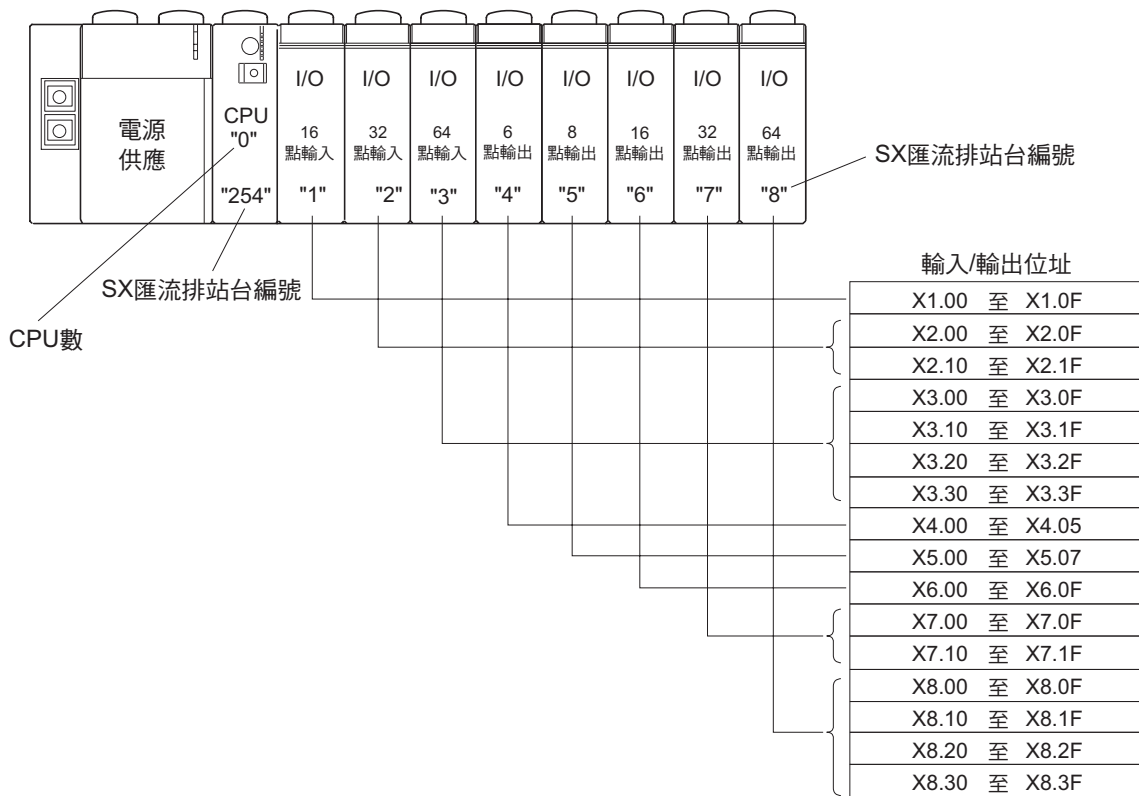
執行100,000個Takt，發生SX匯流排錯誤的Takt數以ppm表示。對已執行的Takt而言，如果即使有一個Takt是錯誤的，此值也會變成「10」。這些資料會每100,000個Takt更新一次。

位址	名稱	說明
WSM508(1FCh)	最大值 (小寫)	設定為由自身CPU模組偵測的SX匯流排的錯誤率值的最大值。
WSM509(1FDh)	最大值 (大寫)	
WSM510(1FEh)	目前值 (小寫)	設定為由自身CPU模組偵測的SX匯流排的錯誤率值的目前值。
WSM511(1FFh)	目前值 (大寫)	

注意：系統記憶體區域的各種類型的系統旗標資訊都可以從應用程式中參照。請勿使用「事件變數」資訊，它會啟動應用程式的事件工作（否則，某些變數可能不會啟動關聯的工作）。

2-3-1 位址指派範例

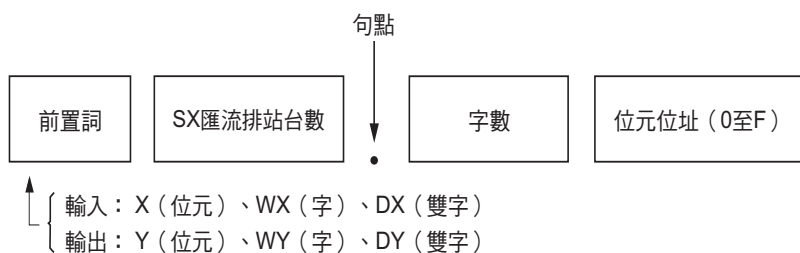
本小節將為您列舉一個位址指派的範例。樣本系統組態如下所示。



2-3-2 位址指派慣例

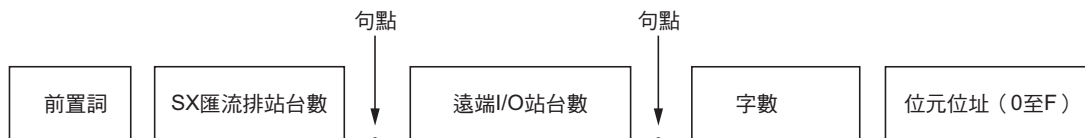
以下慣例可以用來將輸入/輸出位址指派給MICREX-SX SPH系列CPU模組：

1) SX匯流排直接連接I/O



2) 遠端I/O

針對遠端I/O的輸入/輸出，例如T-link、OPCN-1、DeviceNet、PROFI-BUS等，如下所示。



工作就是程式執行的時間表。

工作會決定程式執行的順序（時間表）。MICREX-SX系列CPU模組使用了三種類型的工作：用於循環處理的預設工作、週期工作及事件工作。

您必須指派工作給會一直執行的程式，以便確定它們的執行順序。

2-4-1 工作規格

項目	規格
工作類型	預設工作（循環處理） 週期工作 事件工作
工作數	1（預設值）+ 4（週期與事件工作）
工作優先權	0 > 1 > 2 > 3 > 預設值

2-4-2 工作的類型與作業

1) 預設工作

- 與「工作」同步時始終重複執行。「指派程式」（Assign Programs）在計算作業中不需要信賴性及週期性。
- 預設工作可能會被指派二或多個「程式」。

注意：使用者WDT就是監控預設工作執行時間的計時器。它會在執行過程結束之後檢查時間。當沒有使用預設工作時，CPU會處理與預設工作相同的工作，以執行內部處理，如使用者WDT檢查。

2) 週期工作

- 週期中斷工作會以預先確定的時間間隔（0.5毫秒、1毫秒到32秒）來執行一次。它會被指定給程式與過濾器，它們都需要高信賴性以調整為控制目標的速度，且例如整合工具之類的程式都需要以預定的間隔時間來執行。
 - 週期中斷工作擁有0到3的優先權（0是最高優先權）。
 - 可以將多個程式指定給一個工作。
 - 可以將二或多個程式指定給週期中斷工作（僅適用於SPH）。
- 當使用SPS時，固定的循環工作會異步地中斷目前執行的工作。

注意：一個Takt週期就是一個SX匯流排通訊週期。對於「Takt週期」而言，可以設定0.5毫秒、1毫秒、.....及10毫秒。請注意，在使用標準CPU時，無法指定0.5毫秒。Takt週期要依據系統組態（I/O點數、遠端I/O master模組、通訊模組及CPU模組）的範圍以及應用程式的執行步驟數而定。如果擁有單獨一個功能強大的CPU、連接到SX匯流排的256或更少的I/O模組，且沒有遠端I/O及通訊模組，則可以執行0.5毫秒的Takt週期。在標準CPU系統中，將Takt週期設定為0.5毫秒將會導致發生「系統作業定義錯誤」，從而使CPU由於致命錯誤而關閉。（如需瞭解如何計算Takt週期，請參閱適當的附錄。）

3) 事件工作

- 事件工作會在每次指定的位元裝置變為「1」時被執行一次。它會被指定給處理通訊模組或高速計數器模組中的中斷的程式。
- 二或多個程式可能會被指定給事件工作。

2-4-3 週期工作作業範例

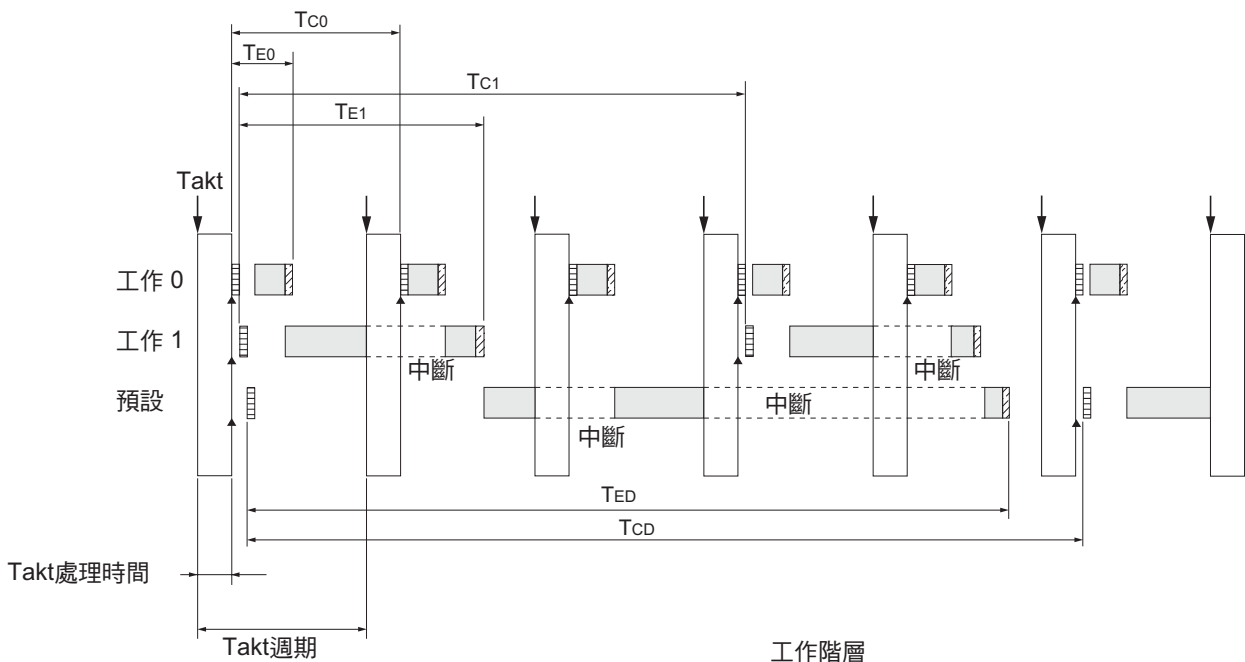
我們將以底下操作條件為您說明使用SPH時固定工作的作業範例。

<操作條件>

工作優先權: 工作0 > 工作1 > 預設工作 (循環)
 Takt週期: 1毫秒
 工作類型: 工作0: 固定工作 (1Takt週期)
 工作1: 固定工作 (3Takt週期)
 預設工作: 與Takt週期異步

注意: 固定工作的工作週期必須設定為SX匯流排Takt週期的整數倍數。

<工作作業>



工作階層
 工作 0 > 工作 1 > 預設

- TE0: 工作0執行時間
- TC0: 工作0執行週期
- TE1: 工作1執行時間
- TC1: 工作1執行週期
- TED: 預設工作執行時間
- TCD: 預設工作執行週期
- ▲ : 工作啟動要求
- ▤ : 資料輸入處理
- ▨ : 資料輸出處理

預設工作會在沒有操作固定工作或在事件工作作業中時執行。(它們與Takt週期同步啟動。)請務必調整執行時間以啟動上層工作的週期,以便能夠為預設工作保留執行時間。(否則,使用者WDTUP或上層工作可能會延遲。)

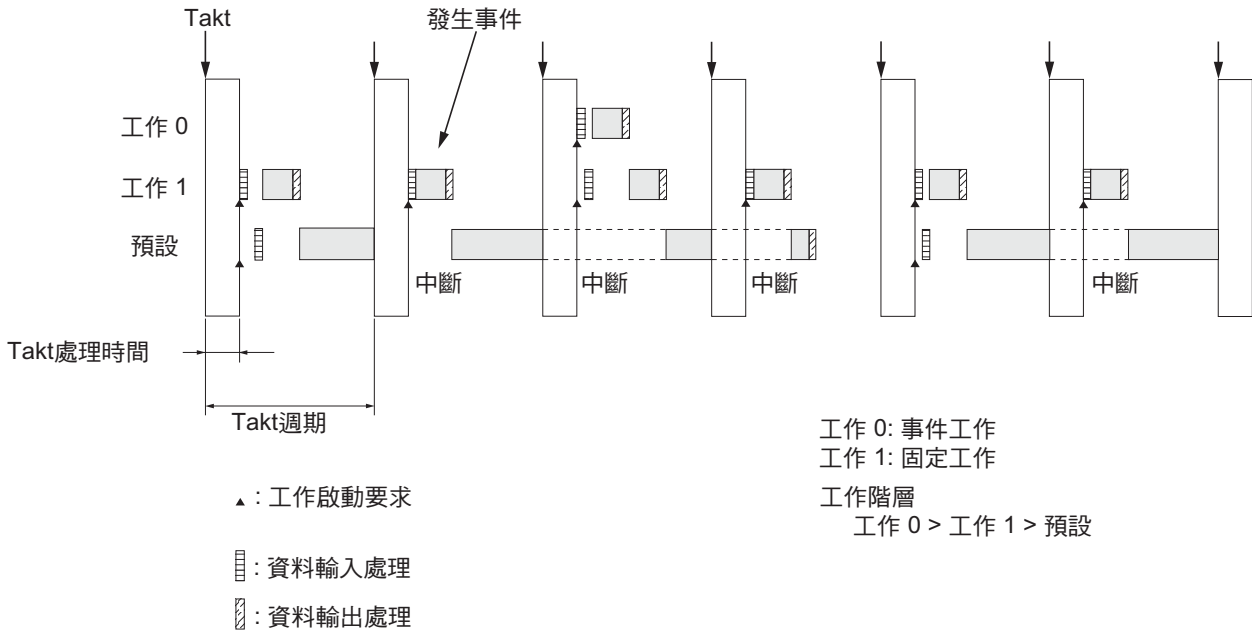
2-4-4 事件工作作業的範例

事件工作的作業範例如下所示。

<操作條件>

工作優先權: 工作0 > 工作1 > 預設工作 (循環)
 Takt週期: 1毫秒
 工作類型: 工作0: 事件工作
 工作1: 固定工作 (1 Takt週期)
 預設工作: 與Takt週期不同步

<工作作業>



當事件發生，但卻在識別事件之後開始下一個Takt週期時，事件工作不會立即啟動。

監控工作的執行時間與執行週期

一個工作的執行時間與執行週期可以根據「載入器資源控制」對話方塊中顯示的資源資訊畫面來監控。

工作執行時間：直到從完成工作後輸出為止才會開始輸入至工作之後的時間。

工作執行週期：直到輸入到下一個工作啟動為止才會開始輸入至工作之後的時間。

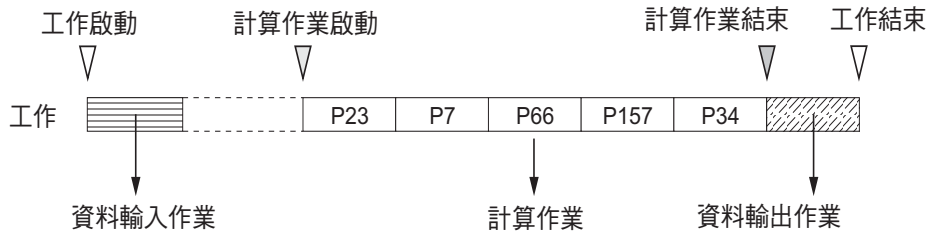
在執行兩次工作之後，工作執行週期會顯示在開啟資源資訊畫面中。在雙工模式下，當CPU停止/啟動或CPU在操作與等待兩邊間切換且資源資訊畫面已開啟時，測量會暫時停止，然後會重新啟動。

在執行週期能夠顯示在畫面上之前，必須已經在CPU停止/啟動之後執行了兩次工作，或已在操作與等待兩邊間切換了工作。

2-4-5 工作中斷處理

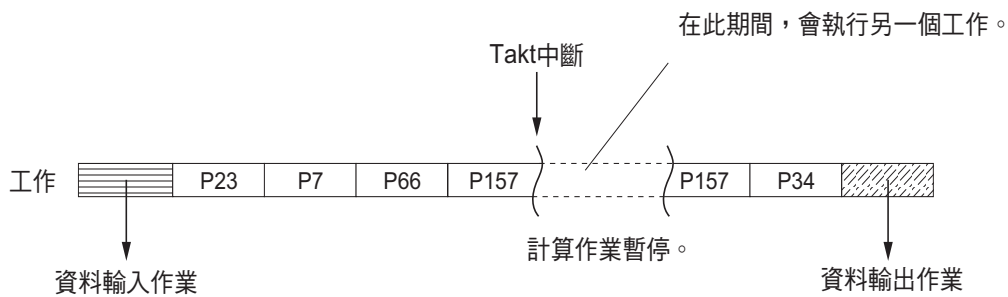
工作的處理可分為資料輸入處理、計算作業以及資料輸出處理，如下所示。當所有這些作業都完成之後，可以將工作作業順序視為終止。

在計算與資料輸出作業期間，Takt中斷可以經由SX匯流排產生（在資料輸入作業期間，不會發生Takt中斷）。

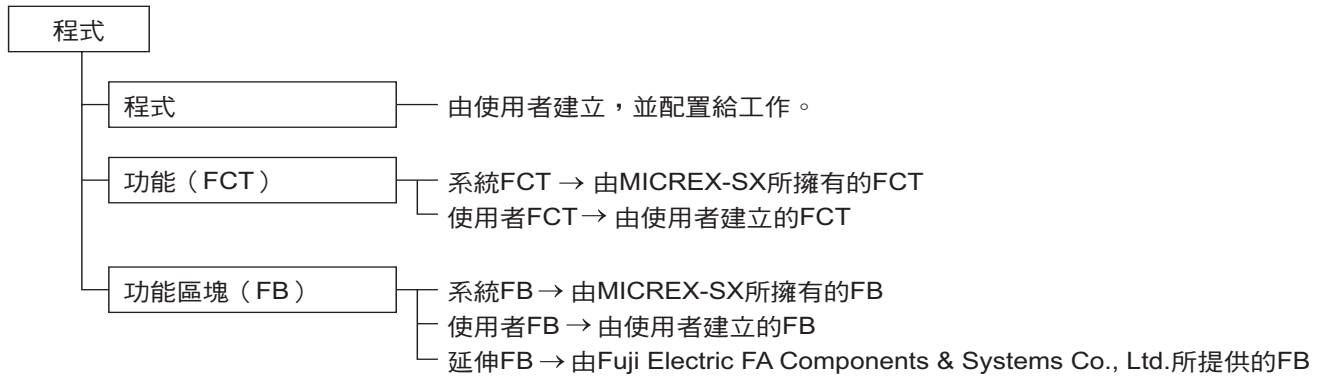


當在計算作業期間發生Takt中斷時，作業會如下圖所示一樣被中斷。當發生Takt中斷時，系統會檢查新工作的啟動條件。如果存在工作啟動要求，系統會針對該工作執行資料輸入作業，並會啟動擁有最高優先權的工作。因此，另一個工作可能會在目前工作暫停時被執行，如下圖所示。

當計算作業結束時，系統會檢查時間，直到下一個Takt中斷發生為止，而且，如果時間比執行資料輸出作業所需的時間長，系統會執行資料輸出處理並終止工作。如果下一個Takt中斷的時間比執行資料輸出作業所需的時間短，系統便會讓工作一直暫停，並不會針對此工作執行輸出作業。資料輸出作業只會在系統執行了下一個Takt處理之後才執行。由於到下一個Takt中斷的時間是在計算作業結束時計算出來的，因此在執行資料輸出作業期間不會發生Takt中斷。



程式分為程式、FCT及FB三類。



<關於呼叫功能與FB>

呼叫程式的類型	可被呼叫的程式類型
程式	系統FCT、使用者FCT、系統FB、使用者FB及延伸FB
使用者FB	系統FCT、使用者FCT、系統FB、使用者FB及延伸FB
使用者FCT	系統FCT與使用者FCT

* 如需瞭解建立使用者FB的程序，請參見附錄4「建立使用者FB的程序」。

MICREX-SX系列CPU模組擁有一個提供行事曆功能的時鐘。行事曆的值可以從載入器中監控及設定，也可以從應用程式中監控與設定。

(1) 行事曆的值範圍

行事曆可以測量的值範圍是從1970年1月1日，00:00:00到2069年12月31日23:59:59。

注意：如果時間比2069年12月31日23:59:59再多一秒，日期與時間都將返回到1970年1月1日。

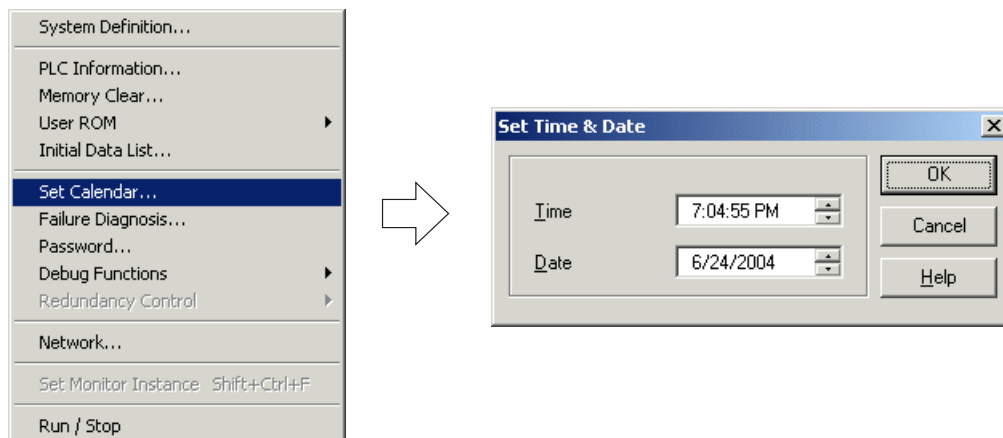
(2) 行事曆精確度

CPU中行事曆（時鐘）的精確度為每月27秒（在25° C的環境下）。

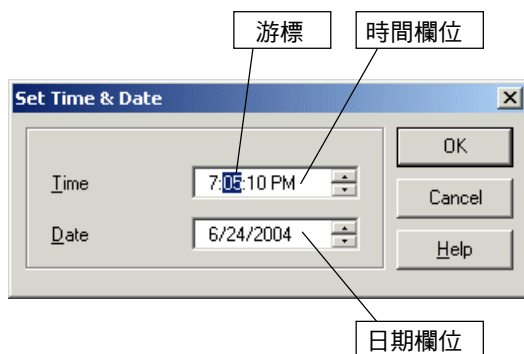
注意：行事曆時鐘的精確度會依據環境條件（如周圍環境溫度）的改變而變化。要在需要高行事曆精確度的系統中使用CPU模組時，請測量實際行事曆精確度並檢查檢驗（時鐘校準）週期。

(3) 從載入器中監控與設定行事曆時鐘

1) 執行 [PLC功能] 功能表中的 [設定行事曆...] 來開啟 [設定時間與日期] 對話方塊。



2) 將游標移動到要變更行事曆的位置，並設定數值。



時間：可讓您設定小時、分、秒。
日期：可讓您設定年、月、日。

3) 在設定了時間與日期之後，請按一下 [確定] 按鈕將設定資料登錄到CPU模組中的行事曆中。

(4) 從應用程式中監控與設定行事曆時鐘**1) HW_RTC (硬體RTC) - 原始FB**

使用硬體RTC功能區塊 (HW_RTC) 可從應用程式中監控與設定行事曆時鐘。相關的詳細說明請參閱HW_RTC FB的使用手冊。

注意) DT類型的 (資料與時間型別) 資料應該針對HW_RTC來指定。DT資料的範圍是1970年1月1日00:00:00到2160年2月7日 6:18:15。而HW_RTC所允許的資料範圍則是從1970年1月1日00:00:00到12月31日23:59:59。(如果指定了任何日期與時間, 預期的值都會被設定。)

當使用外部裝置提供的資料來設定行事曆時鐘時, 需要將輸入資料轉換為DT類型。DT類型的資料相等於秒數的32位元不帶正負號的整數, 其始自1970年1月1日00:00:00。

範例:

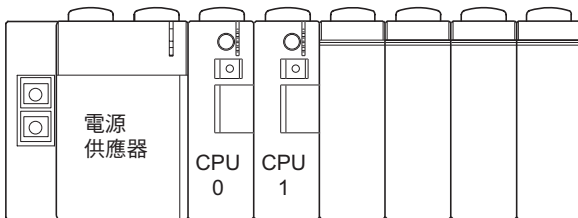
- | | |
|--|------------------|
| (1) 1970年1月1日, 12:34:54至DINT#45296 | WORD#16#0000B0F0 |
| (2) 1998年1月1日, 00:00:00至DINT#883612800 | WORD#16#34AADC80 |

2) RTC (即時時鐘)

RTC無法用來設定行事曆時鐘。如果使用RTC來設定行事曆時鐘, 會有一個相關值儲存在行事曆時鐘要執行的區域內。

(5) 時間調整功能

在多CPU系統中, MICREX-SX系列CPU模組提供了一個功能, 可以自動調整其內部時鐘 (即時時鐘) 的時間。

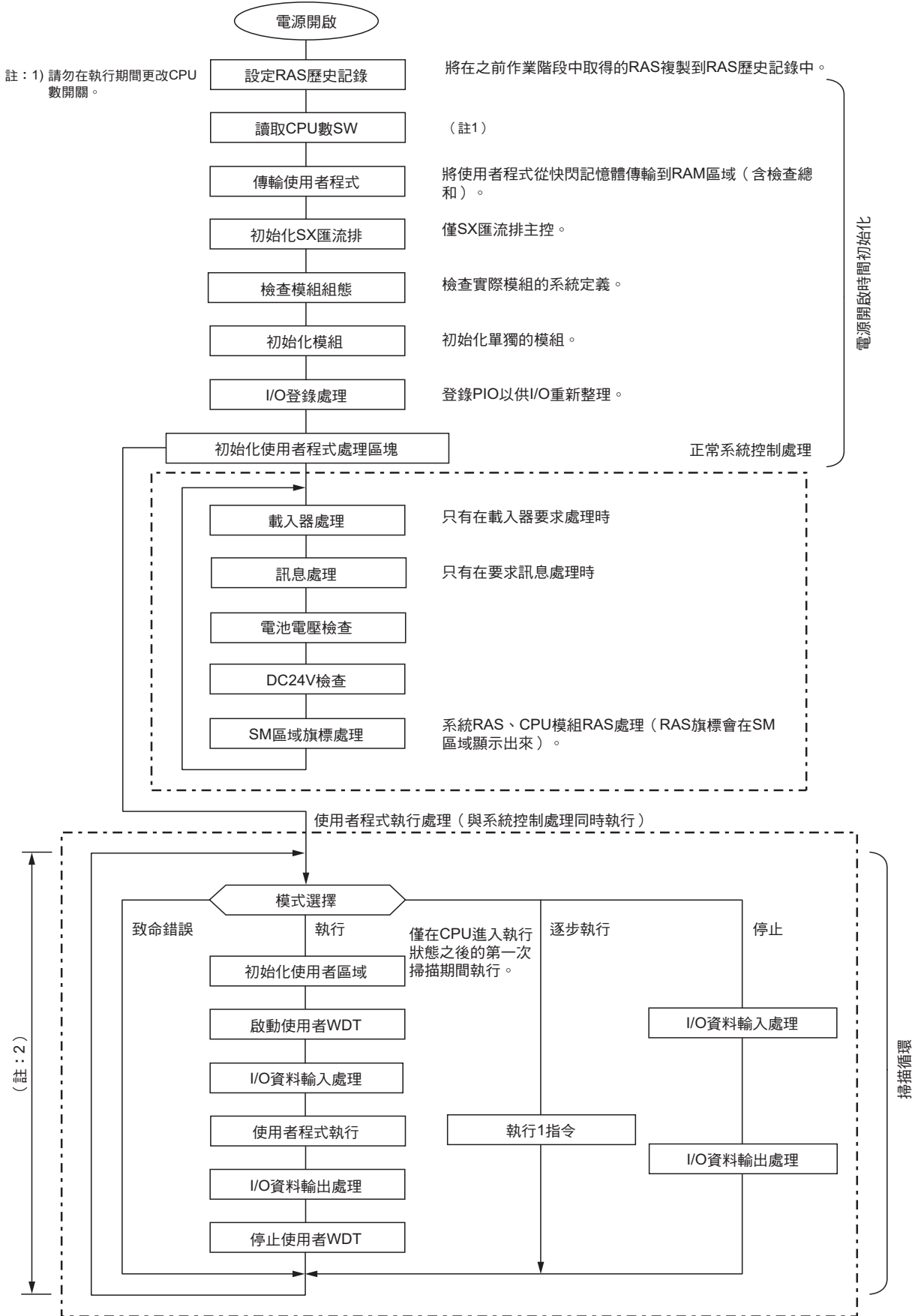
**1) 時間調整管理**

數字「0」CPU模組可調整其他CPU模組的即時時鐘。如果CPU0當掉, 另一個CPU模組就會被指定來調整其他CPU模組的即時時鐘。

2) 調整時間的時機

- 在開啟了系統電源開啟後接著會依據預先確定的時間間隔 (每分鐘) 來進行調整。
- 在從載入器或應用程式中更新即時時鐘時。

下面的操作流程圖顯示了MICREX-SX系列SPH系統的電源開啟順序及後續的作業順序。



註：2) 由程式設計載入器指定的預設工作的執行時間，請參閱本節的時間部分。
 3) 有時在已經發生模式轉變時，完成與指令相關的訊息會花費一些時間。

3-1-1 指令類型

MICREX-SX系列指令可分為下列幾種類型。

指示類型	說明	相關章節
基本指令	讓基本元素設定階梯順序（例如接觸點與線圈）的指令。	3-2-1
程式控制指令	控制程式執行（跳躍、迴圈等）的指令。	3-2-2
轉換指令	轉換裝置資料類型的指令。	3-2-3
數值運算指令	算數運算、三角函數及轉換相關等指令。	3-2-4
位元字串運算指令	邏輯、移位及旋轉運算等指令。	3-2-5
選擇/比較指令	選擇與比較運算的指令。	3-2-6
字元字串指令	從字元字串資料和控制字元字串資料中取得字元的指令，例如連接、取代、比較等。	3-2-7
時間指令	控制時間相關資料的指令。	3-2-8
原始指令	MICREX-SX SPH系列的原始指令。	3-2-9
IEC標準功能區塊	由與程式語言相關的IEC 61131-3國際標準所指定的功能區塊。	3-2-10
原始指令區塊	MICREX-SX SPH系列的原始功能區塊。	3-2-11
使用者功能/使用者功能區塊	使用者建立的指令。	附錄4

< 關於指令描述欄位中的內容 >

「可用的裝置」是每一個指令的說明。裝置的可用性將依建立實際應用程式時是否可用來決定。對非真實的裝置而言，即使可由PLC程式設計師指定，也會表示為“-”（不可用）。

(說明範例)

符號

迴圈

迴圈結束

S: 迴圈 (0至2147483647)

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	
S	-	-	-	-	-	-	-	-	-	-	○	○	-	-	-	○	○

↑
*1

↑
*2



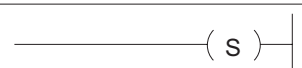




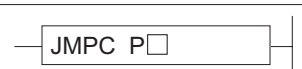

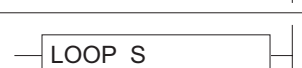
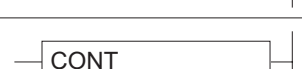

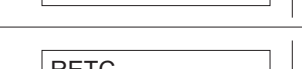
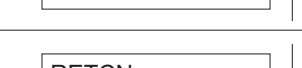
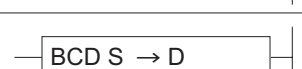
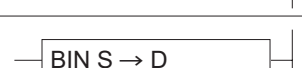
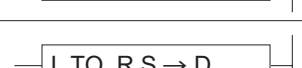

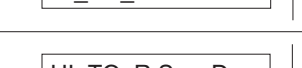
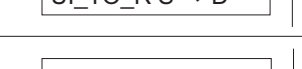
↑
*3

*1. 輸入/輸出記憶體X與Y會表示為不可用，因為除非模組的確存在，否則它們基本上不可用。
*2. 系統記憶體SM是含有定義用法的記憶體。它會表示為不可用，因為使用它做為應用程式的輔助記憶體是不實際的。
*3. 目前計時器數值T與TR會被指定為不可用，因為要用它們來控制迴圈計數很困難。

3-1 指令類型

3-1-2 指令清單

類型	指令	符號	功能	步驟數	相關頁數
基本指令	LD		指定從匯流排正常開啟接觸點	1	3-19
	LD+		指定從匯流排正常開啟接觸點 (上升緣變異)	8	3-20
	LD-		指定從匯流排正常開啟接觸點 (下降緣變異)	8	3-21
	LDI		指定從匯流排正常關閉接觸點	1	3-19
	LDI+		指定從匯流排正常關閉接觸點 (上升緣變異)	8	3-20
	LDI-		指定從匯流排正常關閉接觸點 (下降緣變異)	8	3-21
	AND		以序列方式連接正常開啟接觸點	1	3-19
	AND+		以序列方式連接正常開啟接觸點 (上升緣變異)	8	3-20
	AND-		以序列方式連接正常開啟接觸點 (下降緣變異)	8	3-21
	ANI		以序列方式連接正常關閉接觸點	1	3-19
	ANI+		以序列方式連接正常關閉接觸點 (上升緣變異)	8	3-20
	ANI-		以序列方式連接正常關閉接觸點 (下降緣變異)	8	3-21
	OR		以平行方式連接正常開啟接觸點	1	3-19
	OR+		以平行方式連接正常開啟接觸點 (上升緣變異)	8	3-20
	OR-		以平行方式連接正常開啟接觸點 (下降緣變異)	8	3-21
	ORI		以平行方式連接正常關閉接觸點	1	3-19
	ORI+		以平行方式連接正常關閉接觸點 (上升緣變異)	8	3-20
	ORI-		以平行方式連接正常關閉接觸點 (下降緣變異)	8	3-21
	ANB		以序列方式連接邏輯區塊	1	3-22
	ORB		以平行方式連接邏輯區塊	1	3-22
MPS	未標記於階梯圖表	將運算結果寫入登錄中。	4	3-24	

類型	指令	符號	功能	步驟數	相關頁數
基本指令	MRD	未標記於階梯圖表	從登錄中讀取運算結果	3	3-24
	MPP	未標記於階梯圖表	讀取並重設運算結果	3	3-24
	OUT		線圈	1	3-26
	OUTI		反向線圈	1	3-26
	SET		設定線圈	1	3-27
	RST		重設線圈	1	3-27
	NOT		反向	1	3-28
程式控制指令	JEND		跳躍結束	1	3-29
	JMP		無條件跳躍	2	3-29
	JMPC		有條件跳躍	2	3-29
	JMPCN		負條件跳躍	2	3-29
	LOOP		迴圈	14	3-30
	CONT		迴圈結束	3	3-30
	RET		無條件返回	3	3-31
	RETC		有條件返回	3	3-31
	RETCN		負條件返回	3	3-31
轉換指令	BCD		BIN至BCD	6	3-32
	BIN		BCD至BIN	6	3-33
	I_TO_R		整數至實數（有正負號的整數16位元）	6	3-34
	DI_TO_R		整數至實數（有正負號的整數32位元）	6	3-34
	UI_TO_R		整數至實數（無正負號的整數16位元）	6	3-34
	UDI_TO_R		整數至實數（無正負號的整數32位元）	6	3-34

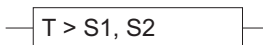
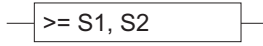
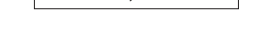

3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
轉換指令	R_TO_I	$\boxed{R_TO_I\ S \rightarrow D}$	實數至整數（有正負號的整數16位元）	6	3-35
	R_TO_DI	$\boxed{R_TO_DI\ S \rightarrow D}$	實數至整數（有正負號的整數32位元）	6	3-35
	R_TO_UI	$\boxed{R_TO_UI\ S \rightarrow D}$	實數至整數（無正負號的整數16位元）	6	3-35
	R_TO_UDI	$\boxed{R_TO_UDI\ S \rightarrow D}$	實數至整數（無正負號的整數32位元）	6	3-35
	TRUNC_I	$\boxed{TRUNC_I\ S \rightarrow D}$	實數至整數（有正負號的整數16位元，小數部分捨去）	6	3-36
	TRUNC_DI	$\boxed{TRUNC_DI\ S \rightarrow D}$	實數至整數（有正負號的整數32位元，小數部分捨去）	6	3-36
	TRUNC_UI	$\boxed{TRUNC_UI\ S \rightarrow D}$	實數至整數（無正負號的整數16位元，小數部分捨去）	6	3-36
	TRUNC_UDI	$\boxed{TRUNC_UDI\ S \rightarrow D}$	實數至整數（無正負號的整數32位元，小數部分捨去）	6	3-36
數值運算指令	ADD	$\boxed{ADD\ S1, S2 \rightarrow D}$	加（有正負號）	6	3-37
	ADD_UAI	$\boxed{ADD_UAI\ S1, S2 \rightarrow D}$	加（無正負號）	6	3-37
	ADD_R	$\boxed{ADD_R\ S1, S2 \rightarrow D}$	加（實數）	6	3-37
	SUB	$\boxed{SUB\ S1, S2 \rightarrow D}$	減（有正負號）	6	3-38
	SUB_UAI	$\boxed{SUB_UAI\ S1, S2 \rightarrow D}$	減（無正負號）	6	3-38
	SUB_R	$\boxed{SUB_R\ S1, S2 \rightarrow D}$	減（實數）	6	3-38
	MUL	$\boxed{MUL\ S1, S2 \rightarrow D}$	乘（有正負號）	6	3-39
	MUL_UAI	$\boxed{MUL_UAI\ S1, S2 \rightarrow D}$	乘（無正負號）	6	3-39
	MUL_R	$\boxed{MUL_R\ S1, S2 \rightarrow D}$	乘（實數）	6	3-39
	DIV	$\boxed{DIV\ S1, S2 \rightarrow D}$	除（有正負號）	6	3-40
	DIV_UAI	$\boxed{DIV_UAI\ S1, S2 \rightarrow D}$	除（無正負號）	6	3-40
	DIV_R	$\boxed{DIV_R\ S1, S2 \rightarrow D}$	除（實數）	6	3-40
	MOD	$\boxed{MOD\ S1, S2 \rightarrow D}$	除餘（有正負號）	6	3-41
	MOD_UAI	$\boxed{MOD_UAI\ S1, S2 \rightarrow D}$	除餘（無正負號）	6	3-41

類型	指令	符號	功能	步驟數	相關頁數
數值運算指令	EXPT	EXPT S1, S2 → D	指數	6	3-42
	ABS	ABS S → D	絕對值	6	3-43
	ABS_R	ABS_R S → D	絕對值 (實數)	6	3-43
	SQRT	SQRT S → D	平方根	6	3-44
	LN	LN S → D	自然對數	6	3-45
	LOG	LOG S → D	常用對數	6	3-46
	EXP	EXP S → D	指數	6	3-47
	SIN	SIN S → D	正弦	6	3-48
	COS	COS S → D	餘弦	6	3-49
	TAN	TAN S → D	正切	6	3-50
	ASIN	ASIN S → D	反正弦	6	3-51
	ACOS	ACOS S → D	反餘弦	6	3-52
	ATAN	ATAN S → D	反正切	6	3-53
轉換指令	MOVE	MOVE S → D	搬移	7	3-54
	NEG	NEG S → D	否定	6	3-55
	NEG_R	NEG_R S → D	否定 (實數)	6	3-55
	BMOV	BMOV S, D, N	區塊搬移	7	3-56
	FMOV	FMOV S, D, N	填滿搬移	19	3-57
	XCH	XCH, D1, D2	交換	11	3-58
	BDMPX	BDMPX S1, D, S2, N	間接放置 (區塊搬移)	12	3-59
	BMPX	BMPX S1, S2, D, N	間接取出 (區塊搬移)	12	3-60

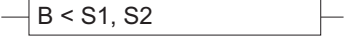
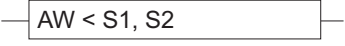
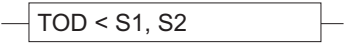
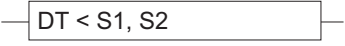


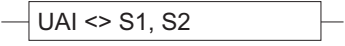
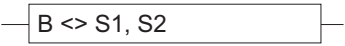
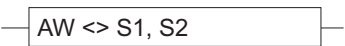
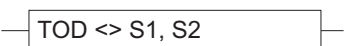
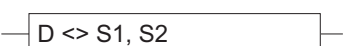
3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
位元串運算指令	AND_AW	— AND_AW S1, S2 → D	邏輯AND	6	3-61
	OR_AW	— OR_AW S1, S2 → D	邏輯OR	6	3-62
	XOR_B	— XOR_B S1, S2 → D	邏輯互斥OR (位元)	6	3-63
	XOR_AW	— XOR_AW S1, S2 → D	邏輯互斥OR (字)	6	3-63
	XORN_B	— XORN_B S1, S2 → D	邏輯互斥NOR (位元)	6	3-64
	XORN_AW	— XORN_AW S1, S2 → D	邏輯互斥NOR (字)	6	3-64
	NOT_AW	— NOT_AW S → D	邏輯NOT	6	3-65
	ROR	— ROR S, N → D	向右旋轉	6	3-66
	ROL	— ROL S, N → D	向左旋轉	6	3-66
	SHR	— SHR S, N → D	向右移動	6	3-67
	SHL	— SHL S, N → D	向左移動	6	3-67
選擇/比較指令	LD > AND > OR >	— > S1, S2	比較 (S1 > S2)	3	3-68
	LD_UAI > AND_UAI > OR_UAI >	— UAI > S1, S2		3	3-68
	LD_R > AND_R > OR_R >	— R > S1, S2		3	3-68
	LD_B > AND_B > OR_B >	— B > S1, S2		3	3-68
	LD_AW > AND_AW > OR_AW >	— AW > S1, S2		3	3-68
	LD_TOD > AND_TOD > OR_TOD >	— TOD > S1, S2		3	3-68
	LD_DT > AND_DT > OR_DT >	— DT > S1, S2		3	3-68
	LD_D > AND_D > OR_D >	— D > S1, S2		3	3-68

類型	指令	符號	功能	步驟數	相關頁數
選擇/比較指令	LD_T > AND_T > OR_T >		比較 (S1 > S2)	3	3-68
	LD >= AND >= OR >=		比較 (S1 ≥ S2)	3	3-69
	LD_UAI >= AND_UAI >= OR_UAI >=			3	3-69
	LD_R >= AND_R >= OR_R >=			3	3-69
	LD_B >= AND_B >= OR_B >=			3	3-69
	LD_AW >= AND_AW >= OR_AW >=			3	3-69
	LD_TOD >= AND_TOD >= OR_TOD >=			3	3-69
	LD_DT >= AND_DT >= OR_DT >=			3	3-69
	LD_D >= AND_D >= OR_D >=			3	3-69
	LD_T >= AND_T >= OR_T >=			3	3-69
	LD = AND = OR =		比較 (S1 = S2)	3	3-70
	LD_UAI = AND_UAI = OR_UAI =			3	3-70
	LD_R = AND_R = OR_R =			3	3-70
	LD_B = AND_B = OR_B =			3	3-70

3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
選擇/比較指令	LD_AW = AND_AW = OR_AW =		比較 (S1 = S2)	3	3-70
	LD_TOD = AND_TOD = OR_TOD =			3	3-70
	LD_DT = AND_DT = OR_DT =			3	3-70
	LD_D = AND_D = OR_D =			3	3-70
	LD_T = AND_T = OR_T =			3	3-70
	LD <=		比較 (S1 ≦ S2)	3	3-71
	LD_UAI <=			3	3-71
	LD_R <=			3	3-71
	LD_B <=			3	3-71
	LD_AW <=			3	3-71
	LD_TOD <=			3	3-71
	LD_DT <=			3	3-71
	LD_D <=			3	3-71
	LD_T <=			3	3-71
	LD <			比較 (S1 < S2)	3
LD_UAI <		3	3-72		

類型	指令	符號	功能	步驟數	相關頁數
選擇/比較指令	LD_R < AND_R < OR_R <		比較 (S1 < S2)	3	3-72
	LD_B < AND_B < OR_B <			3	3-72
	LD_AW < AND_AW < OR_AW <			3	3-72
	LD_TOD < AND_TOD < OR_TOD <			3	3-72
	LD_DT < AND_DT < OR_DT <			3	3-72
	LD_D < AND_D < OR_D <			3	3-72
	LD_T < AND_T < OR_T <			3	3-72
	LD <> AND <> OR <>		比較 (S1 <> S2)	3	3-73
	LD_UAI <> AND_UAI <> OR_UAI <>			3	3-73
	LD_R <> AND_R <> OR_R <>			3	3-73
	LD_B <> AND_B <> OR_B <>			3	3-73
	LD_AW <> AND_AW <> OR_AW <>			3	3-73
	LD_TOD <> AND_TOD <> OR_TOD <>			3	3-73
	LD_DT <> AND_DT <> OR_DT <>			3	3-73
LD_D <> AND_D <> OR_D <>		3	3-73		
LD_T <> AND_T <> OR_T <>		3	3-73		

3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
選擇/比較指令	SEL_B	— SEL_B S1, S2, S3 → D	選擇 S1：「開啟」時，S3 → D 「關閉」時，S2 → D	12	3-74
	SEL	— SEL S1, S2, S3 → D		12	3-74
	SEL_UAI	— SEL_UAI S1, S2, S3 → D		12	3-74
	SEL_R	— SEL_R S1, S2, S3 → D		12	3-74
	SEL_AW	— SEL_AW S1, S2, S3 → D		12	3-74
	SEL_T	— SEL_T S1, S2, S3 → D		12	3-74
	SEL_STR	— SEL_STR S1, S2, S3 → D		12	3-74
	MAX	— MAX S1, S2 → D	最大數值	6	3-75
	MAX_UAI	— MAX_UAI S1, S2 → D		6	3-75
	MAX_R	— MAX_R S1, S2 → D		6	3-75
	MIN	— MIN S1, S2 → D	最小數值	6	3-76
	MIN_UAI	— MIN_UAI S1, S2 → D		6	3-76
	MIN_R	— MIN_R S1, S2 → D		6	3-76
LIMIT	— LIMIT S1, S2, S3 → D	限制 S2<S1時，S1 → D S1≤S2≤S3時，S2 → D S2>S3時，S3 → D	9	3-77	
LIMIT_UAI	— LIMIT_UAI S1, S2, S3 → D		9	3-77	
LIMIT_R	— LIMIT_R S1, S2, S3 → D		9	3-77	
字元字串指令	LEN	— LEN S → D	取得長度	6	3-78
	LEFT	— LEFT S, N → D	取得左子字串	8	3-79
	RIGHT	— RIGHT S, N → D	取得右子字串	8	3-80
	MID	— MID S, N1, N2 → D	取得中字字串	9	3-81
	CONCAT	— CONCAT S1, S2 → D	毗連	9	3-82
	INSERT	— INSERT S1, S2, N → D	插入字串	9	3-83
	DELETE	— DELETE S, N1, N2 → D	刪除字串	9	3-84

類型	指令	符號	功能	步驟數	相關頁數
字元字串指令	REPLACE	— REPLACE S1, S2, N1, N2 → D	取代字串	10	3-85
	FIND	— FIND S1, S2 → D	尋找字串	7	3-86
	LD_STR> AND_STR> OR_STR>	— STR> S1, S2	比較字串 (S1>S2)	4	3-87
	LD_STR>= AND_STR>= OR_STR>=	— STR>= S1, S2	比較字串 (S1≥S2)	4	3-88
	LD_STR= AND_STR= OR_STR=	— STR= S1, S2	比較字串 (S1=S2)	4	3-89
	LD_STR<= AND_STR<= OR_STR<=	— STR<= S1, S2	比較字串 (S1≤S2)	4	3-90
	LD_STR< AND_STR< OR_STR<	— STR< S1, S2	比較字串 (S1<S2)	4	3-91
	LD_STR<> AND_STR<> OR_STR<>	— STR<> S1, S2	比較字串 (S1<>S2)	4	3-92
	MOVE_STR	— MOVE_STR S → D	移動字串	8	3-93
時間指令	ADD_T_T	— ADD_T_T S1, S2 → D	加時間	8	3-94
	ADD_TD_T	— ADD_TD_T S1, S2 → D	加時間 (期間 + 時間)	15	3-94
	ADD_DT_T	— ADD_DT_T S1, S2 → D	加時間 (日期與時間 + 時間)	10	3-94
	SUB_T_T	— SUB_T_T S1, S2 → D	減時間	8	3-96
	SUB_D_D	— SUB_D_D S1, S2 → D	減時間 (日期 - 日期)	10	3-96
	SUB_TD_T	— SUB_TD_T S1, S2 → D	減時間 (期間 - 時間)	13	3-96
	SUB_TD_TD	— SUB_TD_TD S1, S2 → D	減時間 (期間 - 期間)	10	3-96
	SUB_DT_T	— SUB_DT_T S1, S2 → D	減時間 (日期與時間 - 時間)	10	3-96
	SUB_DT_DT	— SUB_DT_DT S1, S2 → D	減時間 (日期與時間 - 日期與時間)	10	3-96
	MUL_T_UDI	— MUL_T_UDI S1, S2 → D	乘時間	8	3-98
	MUL_T_R	— MUL_T_R S1, S2 → D	乘時間 (時間 x 實數)	9	3-98

3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
時間指令	DIV_T_UDI	$\text{DIV_T_UDI } S1, S2 \rightarrow D$	除時間	8	3-99
	DIV_T_R	$\text{DIV_T_R } S1, S2 \rightarrow D$	除時間 (時間 / 實數)	9	3-99
	CONCAT_D_D	$\text{CONCAT_D_D } S1, S2 \rightarrow D$	毗連時間	8	3-101
	DT_TO_TOD	$\text{DT_TO_TOD } S \rightarrow D$	日期與時間 - 期間轉換	9	3-102
	DT_TO_D	$\text{DT_TO_D } S \rightarrow D$	日期與時間 - 日期轉換	10	3-102
原始指令	SBIT	$\text{SBIT } S, N \rightarrow D$	設定位元	6	3-103
	RBIT	$\text{RBIT } S, N \rightarrow D$	重設位元	6	3-104
	TBIT	$\text{TBIT } S, N \rightarrow D$	測試位元	6	3-105
	DECODE	$\text{DECODE } S \rightarrow D$	解碼	6	3-106
	ENCODE	$\text{ENCODE } S \rightarrow D$	編碼	6	3-106
	BITCOUNT	$\text{BITCOUNT } S \rightarrow D$	位元計數	6	3-107
	STR_TO_UI	$\text{STR_TO_UI } S \rightarrow D$	將字串轉換為數字	6	3-108
	UI_TO_STR	$\text{UI_TO_STR } S \rightarrow D$	將數字轉換為字串	7	3-109
	SJ_TO_STR	$\text{SJ_TO_STR } S \rightarrow D$	將shift-JIS轉換為字串	8	3-109
	STR_TO_SJ	$\text{STR_TO_SJ } S \rightarrow D$	將字串轉換為shift-JIS	7	3-110
	BYTE_LEN	$\text{BYTE_LEN } S \rightarrow D$	位元長度	6	3-111
	DBAND	$\text{DBAND } S1, S2 \rightarrow D$	遲滯	8	3-112
	DBAND_R	$\text{DBAND_R } S1, S2 \rightarrow D$	遲滯 (實數)	8	3-112
	BIAS	$\text{BIAS } S1, S2 \rightarrow D$	偏差	8	3-113
	BIAS_R	$\text{BIAS_R } S1, S2 \rightarrow D$	偏差 (實數)	8	3-113
	SC_COIL	$\text{SC_COIL } C, N1, N2 \rightarrow D$	步驟順序線圈	9	3-114
	SC	$\text{SC } C, N1, N2 \rightarrow D$	步驟順序位元	9	3-114
SLC	$\text{SLC } S1, S2 \rightarrow D$	移動左32位元含進位	8	3-115	

類型	指令	符號	功能	步驟數	相關頁數
原始指令	SRC	— SRC S1, S2 → D	移動右32位元含進位	8	3-115
	SLCO	— SLCO S → D	移動左32位元進位	7	3-116
	SRCO	— SRCO S → D	移動右32位元進位	7	3-116
	ADC	— ADC S1, S2, S3 → D	32位元加法含進位	9	3-117
	ADCO	— ADCO S1, S2, S3 → D	32位元加法進位	9	3-117
	SBB	— SBB S1, S2, S3 → D	32位元減法含借位	9	3-118
	SBBO	— SBBO S1, S2, S3 → D	32位元減法借位	9	3-118
	MULL	— MULL S1, S2 → D	64位元乘法（較低順序數字）	8	3-119
	MULU	— MULU S1, S2 → D	64位元乘法（較高順序數字）	8	3-119
	DIVL	— DIVL S1, S2, S3 → D	64位元除法（較低順序數字）	9	3-120
	DIVU	— DIVU S1, S2, S3 → D	64位元除法（較高順序數字）	9	3-120
IEC標準功能區塊	SR	— SR S1 — PARA S2 — PARA S3 — PARA D	設定重設正反器	9	3-121
	RS	— RS S1 — PARA S2 — PARA S3 — PARA D	重設設定正反器	9	3-121
	R_TRIG	— R_TRIG S1 — PARA S2 — PARA D	上升緣觸發器	7	3-122
	F_TRIG	— F_TRIG S1 — PARA S2 — PARA D	下降緣觸發器	7	3-122

3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
IEC標準功能區塊	CTU		往上計數器	13	3-123
	CTD		往下計數器	13	3-123
	TP		脈衝	10	3-124
	TON		開啟延遲計時器	10	3-125
	TOF		關閉延遲計時器	10	3-125
	RTC		即時時鐘	10	3-126
原始指令區塊	RCT		環計數器	13	3-127
	TMR		整合計時器	14	3-128
	MR		可重複觸發計時器	10	3-129
	M_OPEN		開啟頻道	32	3-130
	M_SEND		傳送訊息	23	3-133

類型	指令	符號	功能	步驟數	相關頁數
原始指令區塊	M_RECEIVE		接收訊息	23	3-135
	READ_W READ_B		直接讀取	24 20	3-137
	WRITE_W WRITE_B		直接寫入	24 20	3-139
	R_READ		遠端資料讀取	32	3-141
	R_WRITE		遠端資料寫入	32	3-144

3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
原始指令區塊	F_READ		檔案資料讀取	28	3-148
	F_WRITE		檔案資料寫入	26	3-150
	EXT_T_S		擴充測試與設定	19	3-152
	FFST		順序檔案儲存	24	3-154
	FIFO		順序檔案先載	24	3-156

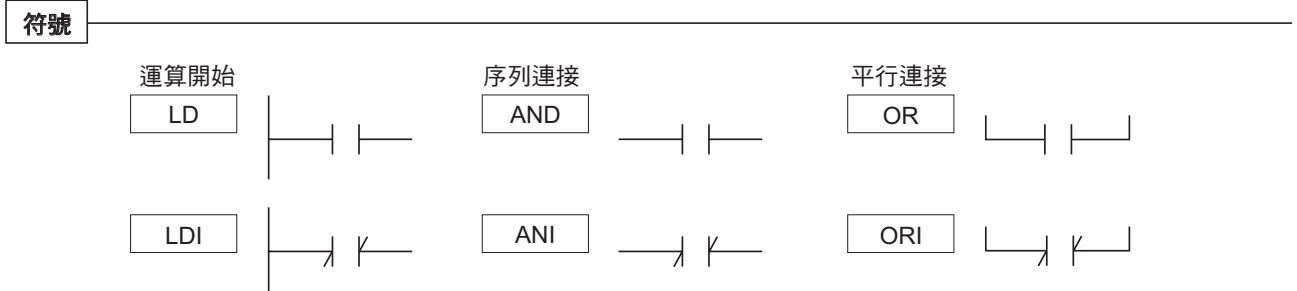
類型	指令	符號	功能	步驟數	相關頁數
原始指令區塊	FILO		順序檔案後載	24	3-158
	FILTER_DI FILTER_R		過濾	12 12	3-160
	INT_DI INT_R		整合	20 20	3-162
	DIF_DI DIF_R		變異	12 12	3-164
	PULSE_CNT		脈衝計數	16	3-166

3-1 指令類型

類型	指令	符號	功能	步驟數	相關頁數
原始指令區塊	PULSE_OUT		脈衝輸出	16	3-167
	PWM		調制脈衝寬度	13	3-168
	HW_RTC		硬體RTC	13	3-169
	T_S		測試與設定	12	3-170
	BANK_CHG		變更頻帶	19	3-171

3-2-1 基本指令

(1) 運算開始、序列連接及平行連接

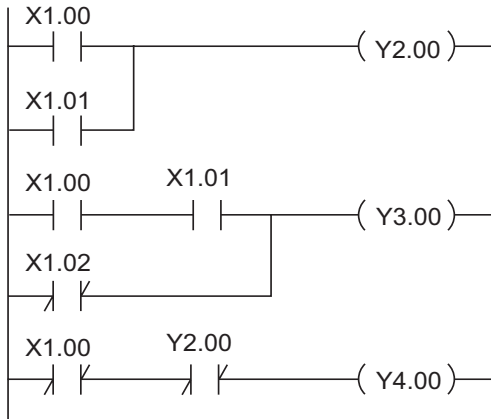


可用裝置										
位元裝置										
X	Y	M	L	SM	T	TR	C	F	V	
○	○	○	○	○	○	○	○	○	○	○

- | 功能 | | |
|-----|-----|---|
| LD | LDI | 1) LD指令是開始運算的NO接觸點指令。LDI指令是開始運算的NC接觸點。每一個指令都會取得指定裝置的「開啟」或「關閉」狀態做為運算結果。 |
| AND | ANI | 1) AND指令會以序列方式連接NO接觸點。ANI指令會以序列方式連接NC接觸點。這些指令會AND指定裝置的「開啟」或「關閉」狀態與之前運算的結果。此AND運算的結果即為指令的結果。 |
| OR | ORI | 1) OR指令會以平行方式連接NO接觸點。ORI指令會以平行方式連接NC接觸點。這些指令會OR指定裝置的「開啟」或「關閉」狀態與之前運算的結果。此OR運算的結果即為指令的結果。 |

程式範例

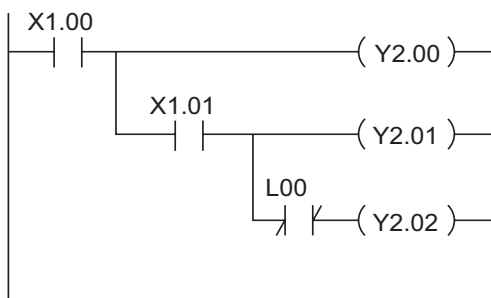
1) 階梯圖表表示法



(助記表示法)

指令	裝置		
LD	X1.00		
OR	X1.01		
OUT	Y2.00		
LD	X1.00		
AND	X1.01		
OUT	Y3.00		
LDI	X1.00		
ANI	Y2.00		
OUT	Y4.00		

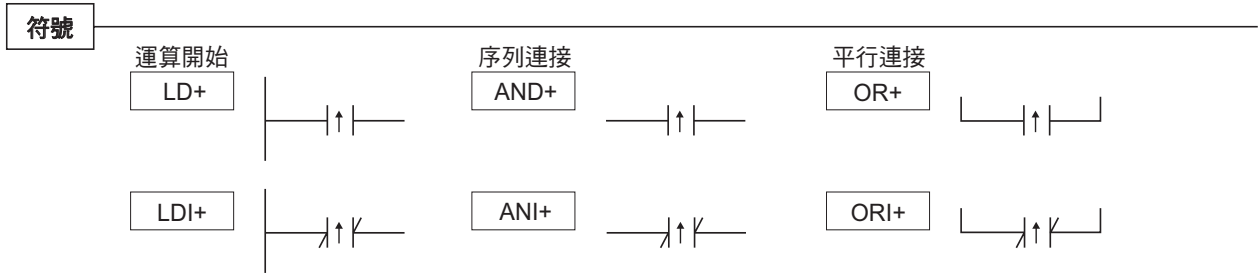
2) 階梯圖表表示法



(記憶表示法)

指令	裝置		
LD	X1.00		
OUT	Y2.00		
AND	X1.01		
OUT	Y2.01		
ANI	L00		
OUT	Y2.02		

(2) 上升緣變異



可用裝置										
位元裝置										
X	Y	M	L	SM	T	TR	C	F	V	
○	○	○	○	○	○	○	○	○	○	○

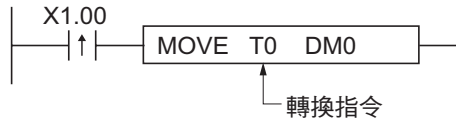
指出上升緣變異正常開啟接觸點FB與無正負號實數（UNIT形式）的實例數。

- | 功能 | | |
|------|------|---|
| LD+ | LDI+ | 1) LD+指令是開始運算的「關閉」對「開啟」變異NO接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。LDI+指令是開始運算的「關閉」對「開啟」變異NC接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「關閉」。 |
| AND+ | ANI+ | 1) AND+指令是以序列方式連接NO接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「開啟」，然後它會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。ANI+指令是以序列方式連接NC接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。 |
| OR+ | ORI+ | 1) OR+指令是以平行方式連接NO接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「開啟」，然後它會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。ORI+指令是以平行方式連接NC接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。 |

程式範例

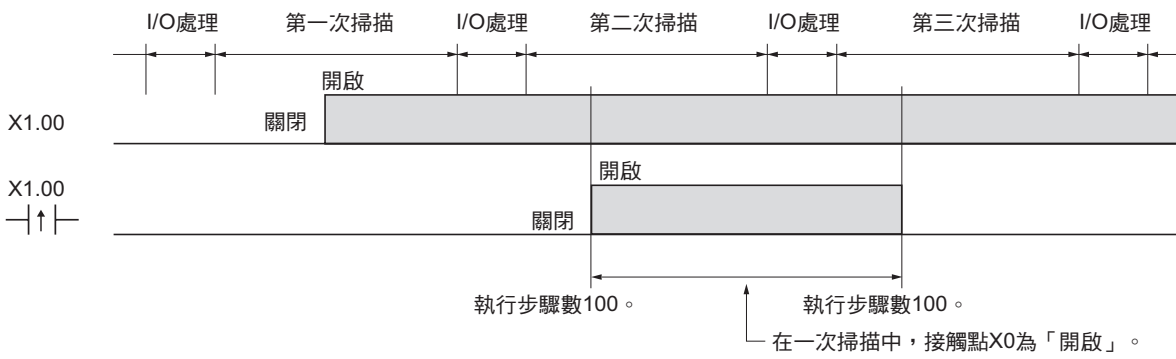
1) 當接觸點X1.00設定為「開啟」時，T0（計時器目前數值）只會傳輸一次至DM0。

(階梯圖表表示法)

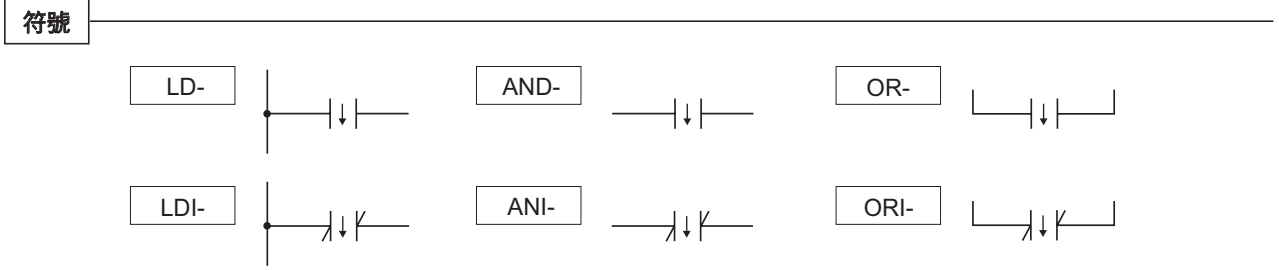


(記憶表示法)

指令	裝置		
LD+	X1.00	0	
MOVE	T0	DM0	



(3) 下降緣變異



可用裝置

位元裝置										
X	Y	M	L	SM	T	TR	C	F	V	
○	○	○	○	○	○	○	○	○	○	○

指出下降緣變異正常開啟接觸點FB與無正負號實數（UNIT形式）的實例數。

- 功能**
- LD-, LDI- 1) LD-指令是開始運算的「開啟」對「關閉」變異NO接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。LDI-指令是開始運算的「關閉」對「開啟」變異NC接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「關閉」。
 - AND-, ANI- 1) AND-指令是以序列方式連接NO接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。然後這個指令會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。ANI-指令是以序列方式連接NC接觸點的「開啟」對「關閉」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。
 - OR-, ORI- 1) OR-指令是以平行方式連接NO接觸點的「開啟」對「關閉」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。然後這個指令會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。ORI-指令是以平行方式連接NC接觸點的「開啟」對「關閉」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。

程式範例

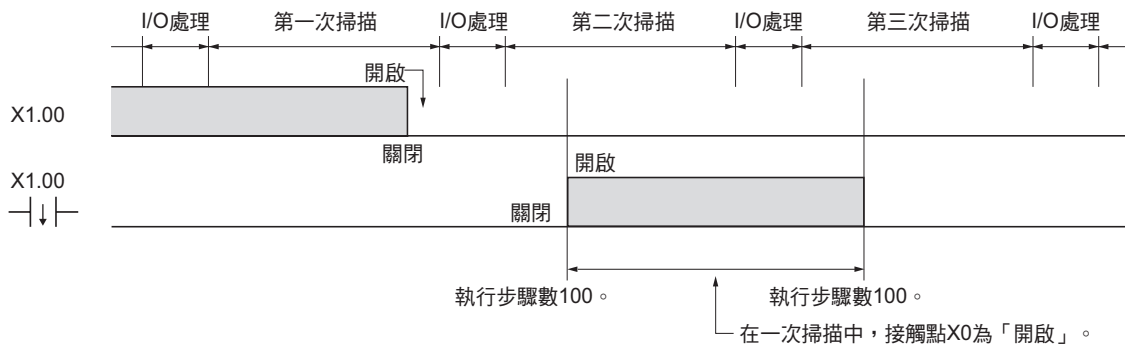
1) 當接觸點X1.00設定為「關閉」時，C0（計數器目前數值）只會傳輸一次至D0。

（階梯圖表表示法）



（記憶表示法）

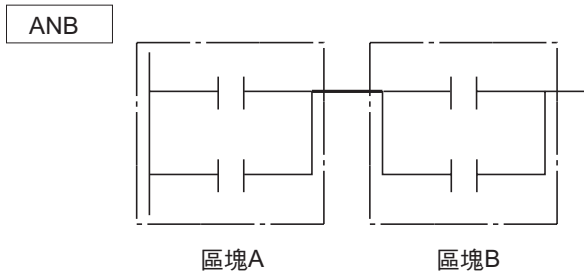
指令	裝置	
LD-	X1.00	0
MOVE	C0	DM0



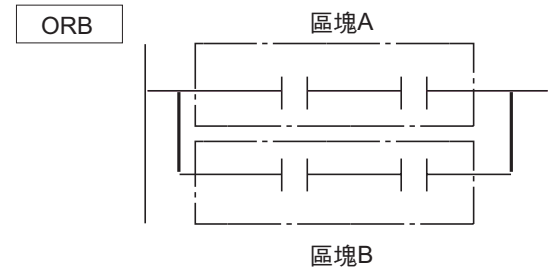
(4) 電路區塊的序列連接與平行連接

符號

電路區塊的序列連接



電路區塊的平行連接



功能

ANB

1) ANB指令可AND區塊A與B。此AND運算的結果會成為指令的結果。

ORB

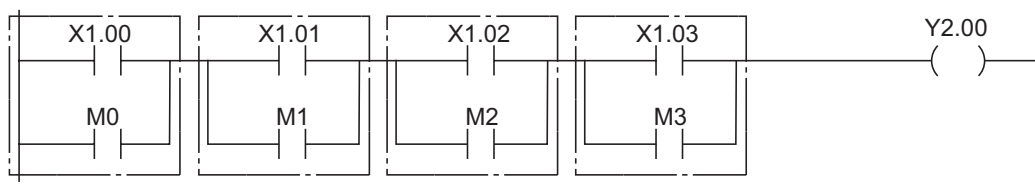
1) ORB指令可OR區塊A與B。此OR運算的結果會成為指令的結果。

2) ORB會以平行方式連接有二個以上接觸點的電路區塊。只有一個接觸點的電路區塊可以使用OR或ORI指令以平行方式連接。ORB指令不適用此用途。

程式範例

ANB

(階梯圖表表示法)



有兩種記憶語言程式設計可以序列方式連續連接電路區塊。如果選擇方法2)，所能使用的ANB指令有限。

(記憶表示法1)

指令	裝置		
LD	X1.00		
OR	M0		
LD	X1.01		
OR	M1		
ANB			
LD	X1.02		
OR	M2		
ANB			
LD	X1.03		
OR	M3		
ANB			
OUT	Y2.00		

* 可以使用的ANB指令數無限。

(記憶表示法2)

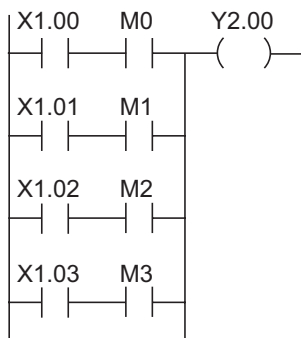
指令	裝置		
LD	X1.00		
OR	M0		
LD	X1.01		
OR	M1		
LD	X1.02		
OR	M2		
LD	X1.03		
OR	M3		
ANB			
ANB			
ANB			
OUT	Y2.00		

* 最多可連續寫入的ANB指令為七個 (八個區塊)。

ORB

有兩種記憶語言程式設計可以平行方式連續連接電路區塊。

(階梯圖表表示)



(記憶表示法1)

指令	裝置		
LD	X1.00		
AND	M0		
LD	X1.01		
AND	M1		
ORB			
LD	X1.02		
AND	M2		
ORB			
LD	X1.03		
AND	M3		
ORB			
OUT	Y2.00		

* 可以使用的ORB指令數無限。

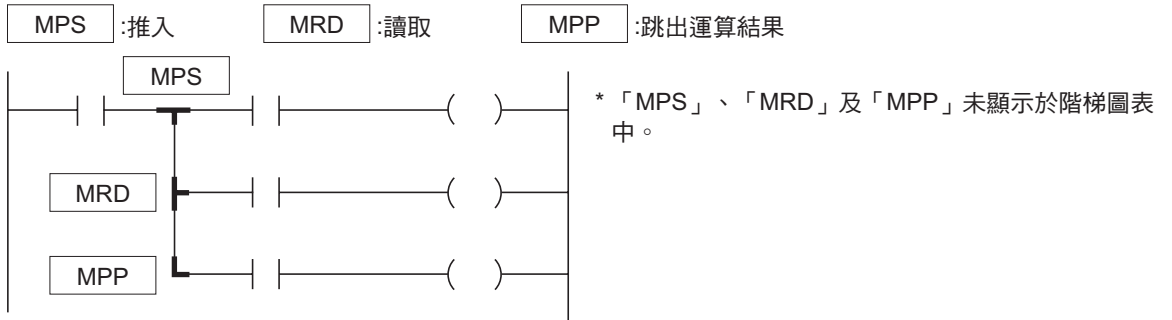
(記憶表示法2)

指令	裝置		
LD	X1.00		
AND	M0		
LD	X1.01		
AND	M1		
LD	X1.02		
AND	M2		
LD	X1.03		
AND	M3		
ORB			
ORB			
ORB			
OUT	Y2.00		

* 最多可連續寫入的ORB指令為七個 (八個區塊)。

(5) 運算結果的推入、讀取及跳出

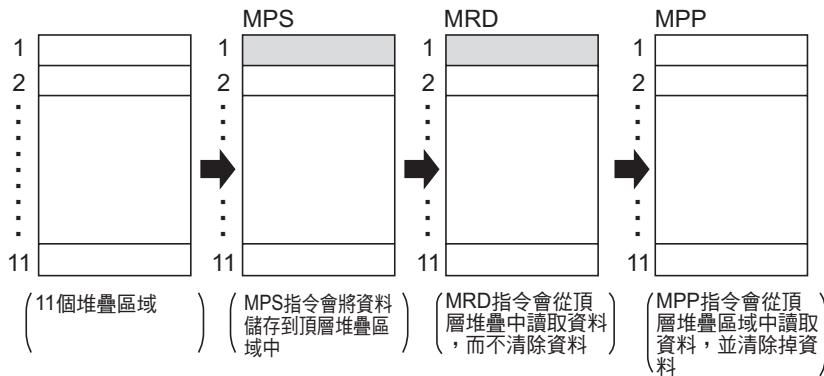
符號



功能

- MPS**
- 1) MPS指令會在MPS指令之前儲存運算結果（「開啟」或「關閉」狀態）。
 - 2) 最多可連續寫入11個MPS指令。
- MRD**
- 1) MRD指令會讀取由MPS指令所儲存的運算結果。然後它會根據讀取運算的結果繼續從下一個步驟運算。
- MPP**
- 1) MPP指令會讀取由MPS指令所儲存的運算結果。然後它會根據讀取運算的結果繼續從下一個步驟運算。
 - 2) MPP指令會清除由MPS指令所儲存的結果。

<Strage (堆疊) 區域運算>



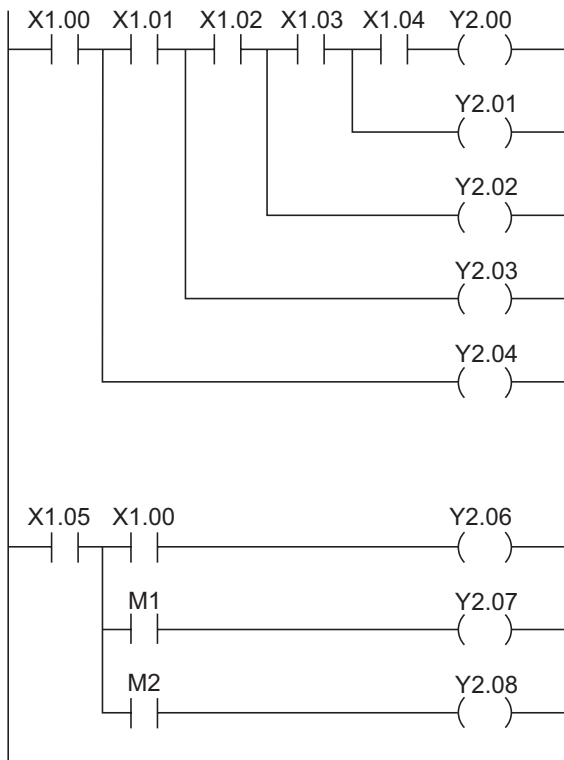
程式範例

MPS

MRD

MPP

(階梯圖表表示法)



重點

在電路中，MPS指令數必須與MPP指令數相同。如果指令數不同，則會偵測到程式錯誤，PC將無法正常作業。

(記憶表示法)

指令	裝置		
LD	X1.00		
MPS			
AND	X1.01		
MPS			
AND	X1.02		
MPS			
AND	X1.03		
MPS			
AND	X1.04		
OUT	Y2.00		
MPP			
OUT	Y2.01		
MPP			
OUT	Y2.02		
MPP			
OUT	Y2.03		
MPP			
OUT	Y2.04		
LD	X1.05		
MPS			
AND	M0		
OUT	Y2.06		
MRD			
AND	M1		
OUT	Y2.07		
MPP			
AND	M2		
OUT	Y2.08		

(6) 線圈、反向線圈

符號



可用裝置

位元裝置									
X	Y	M	L	SM	T	TR	C	F	V
-	○	○	○	○	-	-	-	○	○

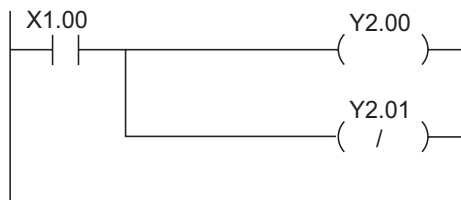
功能

- OUT** 可將上至「OUT」指令的運算結果輸出到指定裝置。
- OUTI** 可反轉上至「OUTI」指令的運算結果，並將結果輸出到指定裝置。

程式範例

OUT , **OUTI** 當X1.00「開啟」時，Y2.00也會「開啟」，而Y2.01則會「關閉」。

(階梯圖表表示法)



(記憶表示法)

指令	裝置		
LD	X1.00		
OUT	Y2.00		
OUTI	Y2.01		

(7) 設定、重設



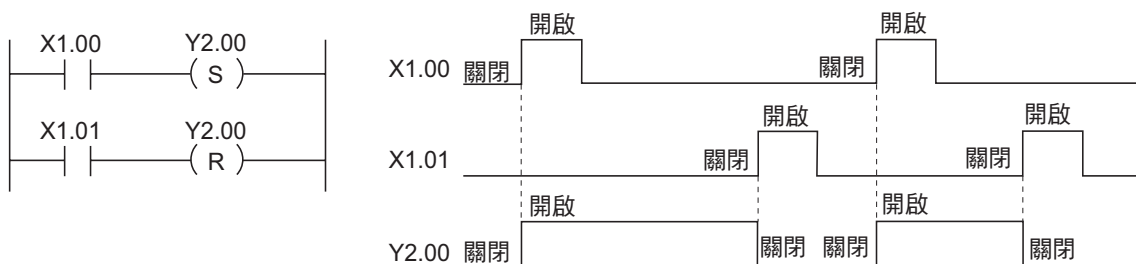
可用裝置

位元裝置									
X	Y	M	L	SM	T	TR	C	F	V
-	○	○	○	○	-	-	-	○	○

功能

SET

- 1) 當「SET」輸入「開啟」時，指定裝置也會「開啟」。
- 2) 設定好的裝置可使用RTS指令來設定為「關閉」。



RST

- 1) 當「RST」輸入「開啟」時，指定裝置則會「關閉」。

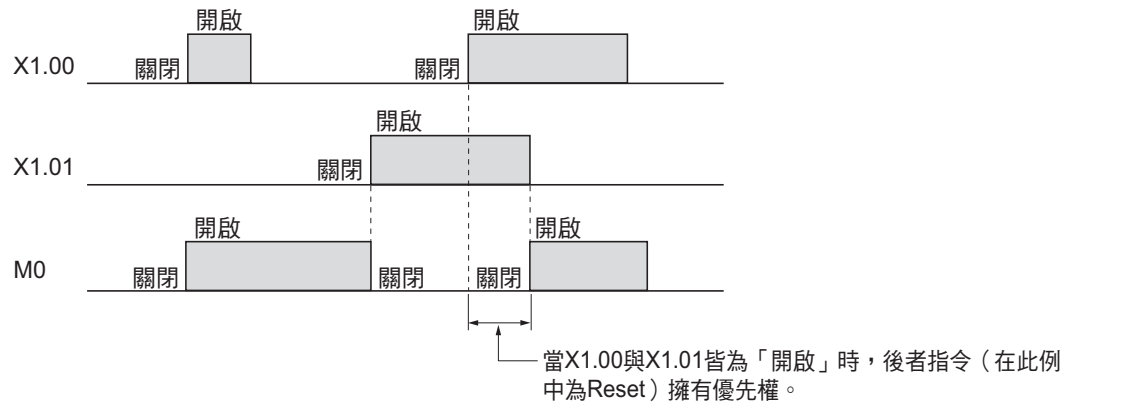
程式範例

1) 當X1.00「開啟」時，將M0設定為「開啟」，當X1.01為「開啟」時，將M0重設為「關閉」。

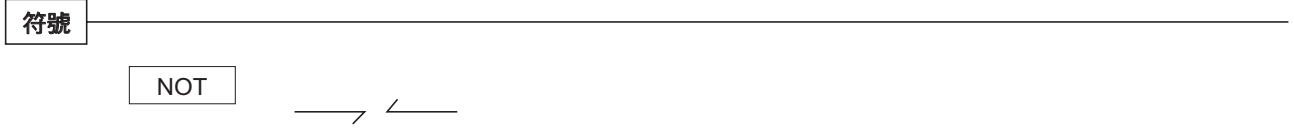
(階梯圖表表示法)

(記憶表示法)

指令	裝置		
LD	X1.00		
SET	M0		
LD	X1.01		
RST	M0		



(8) 反轉

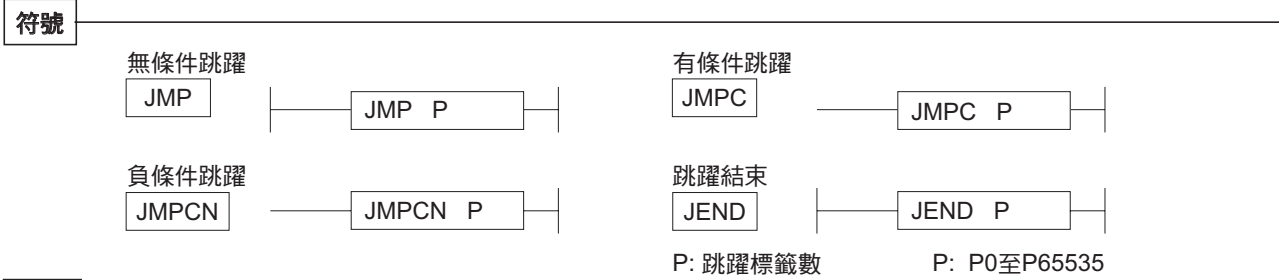


功能

NOT NOT指令可將位於左方的邏輯線路數值反轉，並將結果傳送到位於其右的邏輯線路中。

3-2-2 程式控制指令

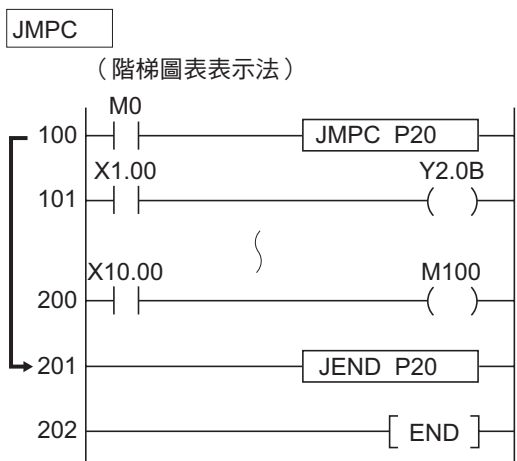
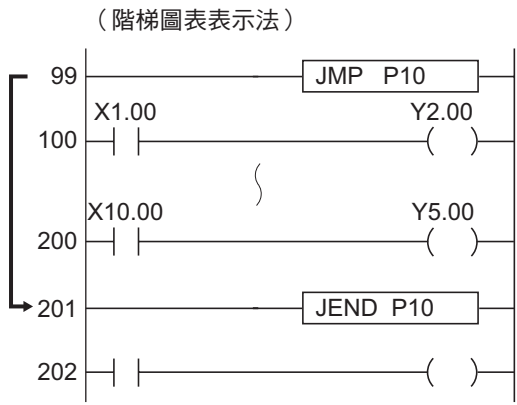
(1) 跳躍 / 跳躍結束



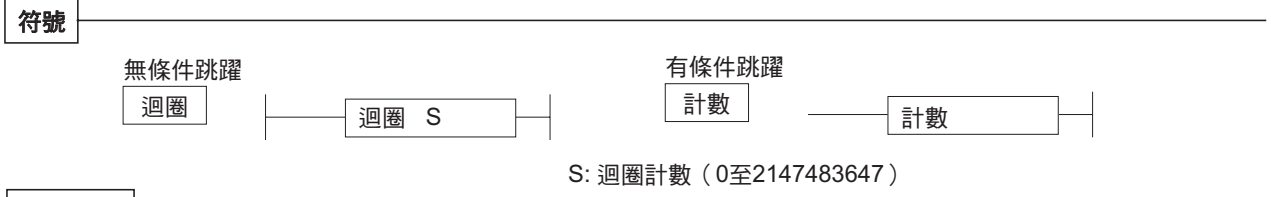
- | 功能 | |
|-------|---|
| JMP | 跳到運算元中所指定的標籤位置。被跳過的指令將不執行。 |
| JMPC | 當上至目前時間的運算結果為「開啟」時，可跳至運算元中所指定的標籤位置。
當上至目前時間的運算結果為「關閉」時，則執行下一個指令。 |
| JMPCN | 當上至目前時間的運算結果為「關閉」時，可跳至運算元中所指定的標籤位置。
當上至目前時間的運算結果為「開啟」時，則執行下一個指令。 |
| JEND | 將跳躍標籤加到任意位置上。這必須跟各個JMP配對使用。 |

程式範例

JMP 無條件在程式中從第100個電路跳到第200個電路。



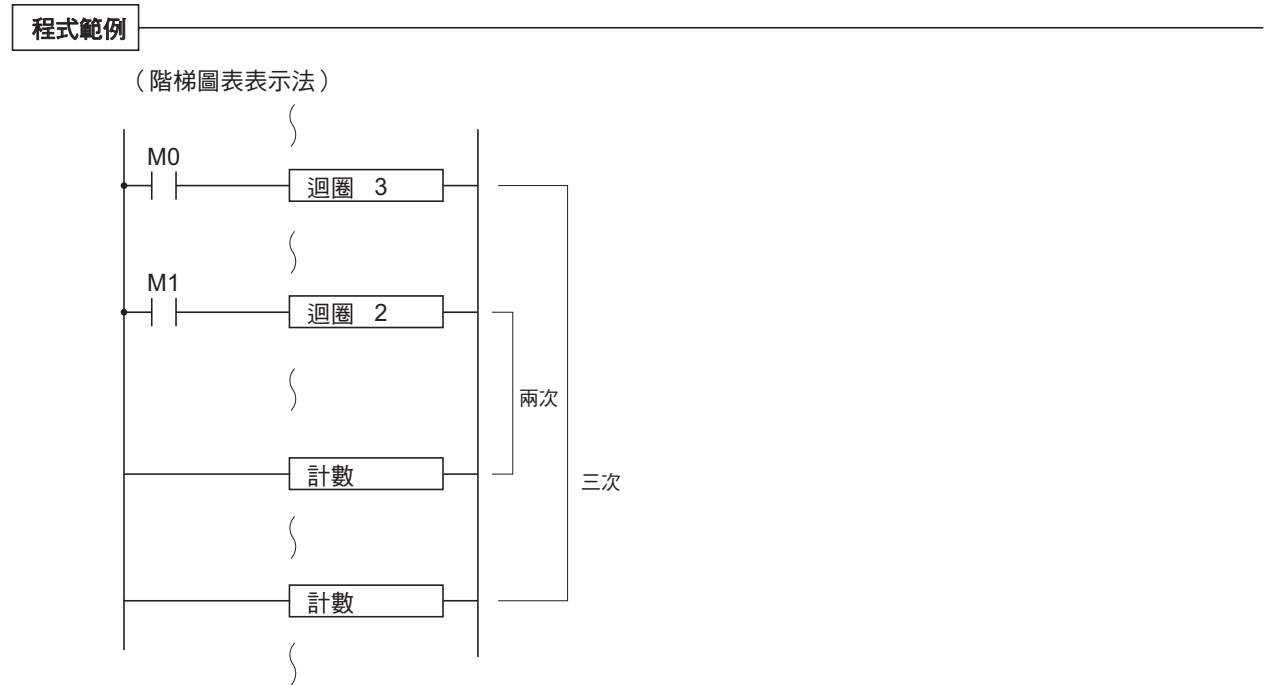
(2) 迴圈、迴圈結束



可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	
S	-	-	-	-	-	-	-	-	-	-	○	○	-	-	-	○	○

- 功能**
- 1) 在執行LOOP至CONT指令 "S"次之後 (次數以S表示)，則執行CONT之後的步驟。
 - 2) 允許的巢狀結構層數最高8層。
 - 3) "S"的允許範圍從0到2147483647。在 "S" 中指定負值時，則假定S=1。
 - 4) 如果在LOOP指令之前放了一個CONT指令、不存在CONT指令、在FEND或END指令之後放了一個CONT指令、LOOP指令數與CONT指令數不符或發生超過層數的情形時，都會發生程式錯誤。
 - 5) 在執行 "LOOP - CONT" 的過程中，更改迴圈計數是無效的。
- 註：當運算花很長時間進行高迴圈計數時，便可能會發生監視計時器錯誤。



(3) 返回指令

符號

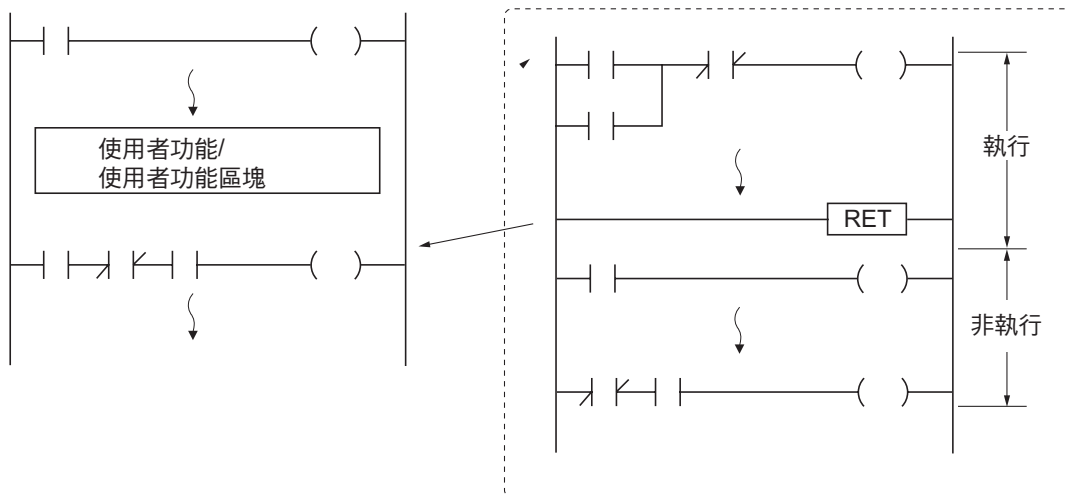


功能

- RET**
- 1) 在使用者功能/使用者功能區塊中使用它時，會無條件返回到功能呼叫位置之後的步驟。
 - 2) 在程式中使用它時，會無條件返回到程式的最後位置。
- RETC**
- 1) 上至目前時間的運算結果為「開啟」時，則執行 "- [RET] -"。
 - 2) 上至目前時間的運算結果為「關閉」時，則執行下一個指令。
- RETCN**
- 3) 上至目前時間的運算結果為「關閉」時，則執行 "- [RET] -"。
 - 4) 上至目前時間的運算結果為「開啟」時，則執行下一個指令。

<運算>

執行RET指令



3-2-3 轉換指令

(1) BIN 至 BCD (BCD)

符號

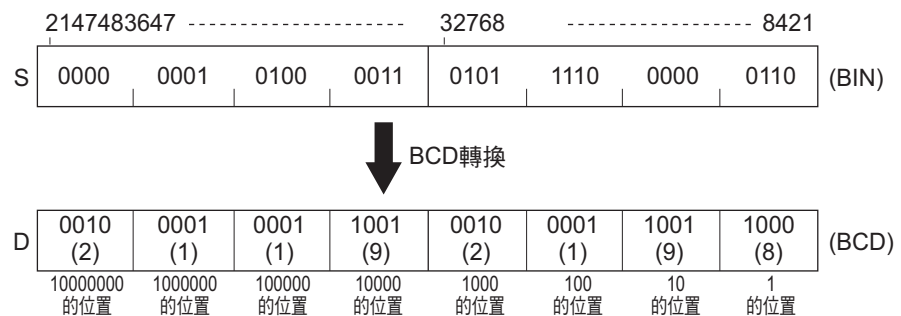


可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

BCD 1) 將指定於 "S" 中的裝置之BIN資料 (16或32位元) 轉換為BCD資料, 並將結果輸出至指定於 "D" 中的裝置。



- 2) 指定於 "S" 與 "D" 中的數值必須擁有相同的位元寬度。
- 3) 資料範圍是：
 指定字裝置時：0至9999
 指定雙字裝置時：0至99999999
 如果BIN資料超出指定的資料範圍，則會輸出 "9999" (指定字裝置時) 或 "99999999" (指定雙字裝置時)。

(2) BCD 至 BIN (BIN)

符號

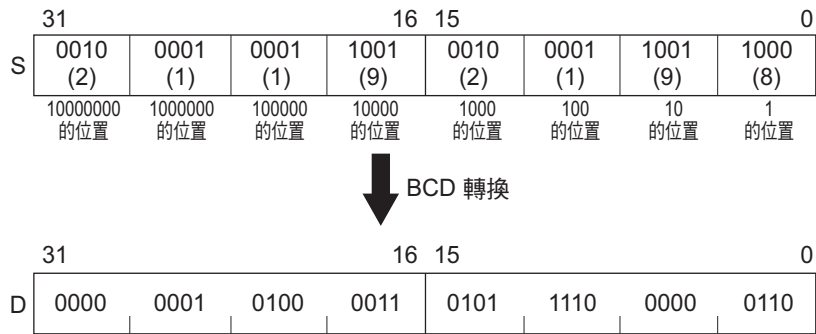


可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—	

功能

1) 轉換 "S" 中指定的裝置中的BCD資料 (16位元資料為0至9999，或32位元資料為0至99999999)，並將結果輸出至在 "D" 中所指定的裝置中。



2) 指定於 "S" 與 "D" 中的數值必須擁有相同的位元寬度。

(3) BIN 至 real (I_TO_R)

符號

- I_TO_R S → D : 含正負號的整數16位元
 - DI_TO_R S → D : 含正負號的整數32位元
 - UI_TO_R S → D : 不含正負號的整數16位元
 - UDI_TO_R S → D : 不含正負號的整數32位元
- S : 儲存被轉換的BIN資料的裝置
D : 儲存轉換結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

- 1) 將在指定裝置 "S" 中的BIN整數資料轉換為實數類型 (REAL) 資料，並將轉換結果輸出到在 "D" 中所指定的裝置中。
- 2) 轉換之後的有效數字是6

<運算>

I_TO_R , DI_TO_R
UI_TO_R , UDI_TO_R

<p>S (BIN) 轉換 D (實數)</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">32767</div> ⇒ <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">3.2767E + 04</div> </div> <p style="margin-left: 150px;">ENO = 1</p>	<p>S (BIN) 轉換 D (實數)</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">65535</div> ⇒ <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">6.5535E + 04</div> </div> <p style="margin-left: 150px;">ENO = 1</p>
<p>S (BIN) 轉換 D (實數)</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">-32768</div> ⇒ <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">-3.2768E + 04</div> </div> <p style="margin-left: 150px;">ENO = 1</p>	<p>S (BIN) 轉換 D (實數)</p> <div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">0</div> ⇒ <div style="border: 1px solid black; padding: 5px; margin-right: 10px;">-0.0E + 00</div> </div> <p style="margin-left: 150px;">ENO = 1</p>

[參照]

資料範圍如下：

- BIN 整數 (含正負號的 16 位元) : -32,768 至 32,767
- BIN 整數 (不含正負號的 16 位元) : 0 至 65,535
- BIN 整數 (含正負號的 32 位元) : -2,147,483,648 至 2,147,483,647
- BIN 整數 (不含正負號的 32 位元) : 0 至 4,294,967,295
- 實數 : $-2^{128} < N \leq -2^{-126}$ 、 0 、 $2^{-126} \leq N < 2^{128}$

(4) 實數至 BIN 整數 (R_TO_I)

符號

- $R_TO_I \quad S \rightarrow D$: 含正負號的整數16位元
 - $R_TO_DI \quad S \rightarrow D$: 含正負號的整數32位元
 - $R_TO_UI \quad S \rightarrow D$: 不含正負號的整數16位元
 - $R_TO_UDI \quad S \rightarrow D$: 不含正負號的整數32位元
- S: 儲存被轉換的實數資料的裝置
D: 儲存轉換結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	-	-	-	-	-	-	-	-	○	○	○	○	○	○	○	○	○	○	○
D	○	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	○	-

功能

- 1) 將在指定裝置 "S" 中的BIN整數資料轉換為實數類型 (REAL) 資料，並將轉換結果輸出到在 "D" 中所指定的裝置中。
- 2) 當從實數類型資料轉換出來的資料超過在 "D" 中指定的裝置資料範圍時，會將資料範圍的邊界數值輸出至 "D"。
- 3) 捨去小數點以下的小數。

<運算>

R_TO_I

- 當運算結果在 INT 類型範圍以內時



- 當運算結果超出 INT 類型範圍時



R_TO_UI

- 當運算結果在 UINT 類型範圍以內時



- 當運算結果超出 UINT 類型範圍時



R_TO_DI

- 當運算結果在 DINT 類型範圍以內時



- 當運算結果超出 DINT 類型範圍時



R_TO_UDI

- 當運算結果在 UDINT 類型範圍以內時



- 當運算結果超出 UDINT 類型範圍時



[參照]

資料範圍如下：

- BIN 整數（含正負號的 16 位元）：-32,768 至 32,767
- BIN 整數（不含正負號的 16 位元）：0 至 65,535
- BIN 整數（含正負號的 32 位元）：-2,147,483,648 至 2,147,483,647
- BIN 整數（不含正負號的 32 位元）：0 至 4,294,967,295
- 實數： $-2^{128} < N \leq -2^{-126}$ 、 0 、 $2^{-126} \leq N < 2^{128}$

(5) 實數至整數 (TRUNC_I)

符號

- TRUNC_I S → D : 含正負號的整數16位元
 - TRUNC_DI S → D : 含正負號的整數32位元
 - TRUNC_UI S → D : 不含正負號的整數16位元
 - TRUNC_UDI S → D : 不含正負號的整數32位元
- S : 儲存被轉換的實數資料的裝置
D : 儲存轉換結果的裝置

可用裝置

	字裝置 (W*)							雙字裝置 (D*)							常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T		TR	F
S	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	○	○	○	○	-	○	○	-	○	○	○	-	○	○	○	○	-

功能

- 1) 將在指定裝置 "S" 中的BIN整數資料轉換為實數類型 (REAL) 資料，並將轉換結果輸出到在 "D" 中所指定的裝置中。
- 2) 當從實數類型資料轉換出來的資料超過在 "D" 中指定的裝置資料範圍時，會將資料範圍的邊界數值輸出至 "D"。
- 3) 捨去小數點以下的小數。

<運算>

TRUNC_I

- 當運算結果在 INT 類型範圍以內時



- 當運算結果超出 INT 類型範圍時



TRUNC_UI

- 當運算結果在 UINT 類型範圍以內時



- 當運算結果超出 UINT 類型範圍時



TRUNC_DI

- 當運算結果在 DINT 類型範圍以內時



- 當運算結果超出 DINT 類型範圍時



[參照]

資料範圍如下：

- BIN 整數 (含正負號的 16 位元) :
-32,768 至 32,767
- BIN 整數 (含正負號的 32 位元) :
-2,147,483,648 至 2,147,483,647
- 實數 : $-2^{128} < N \leq -2^{-126}$, $0, 2^{-126} \leq N < 2^{128}$

TRUNC_UDI

- 當運算結果在 UDINT 類型範圍以內時



- 當運算結果超出 UDINT 類型範圍時

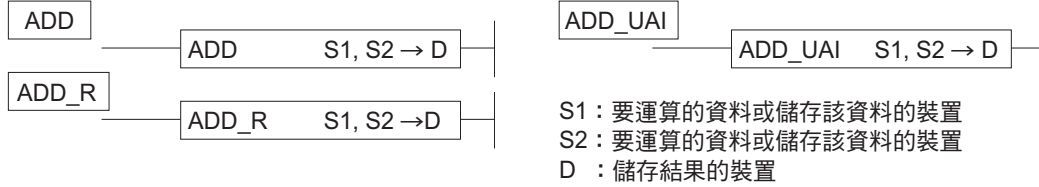


- BIN 整數 (不含正負號的 16 位元) :
0 至 65,535
- BIN 整數 (不含正負號的 32 位元) :
0 至 4,294,967,295

3-2-4 算數運算指令

(1) 加 (ADD)

符號



ADD : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
ADD_UAI : 整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
ADD_R : 實數類型 (單精準浮點類型) [REAL]

可用裝置

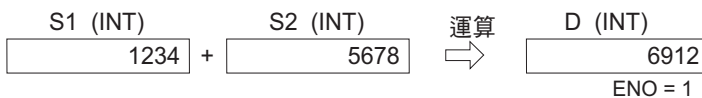
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

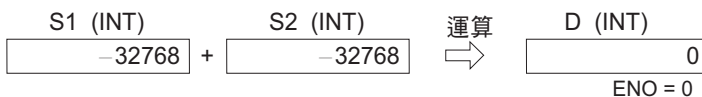
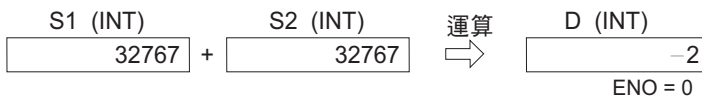
- 1) 將輸入資料 "S1" 與 "S2" 相加，並將結果輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 當運算結果超出資料類型範圍時，ENO會變為0。
- 4) 當資料類型為實數，且非常接近0，而無法以實數資料表示時，輸出會變為0，ENO變為1。

<運算>

- 當運算結果在資料類型範圍以內時



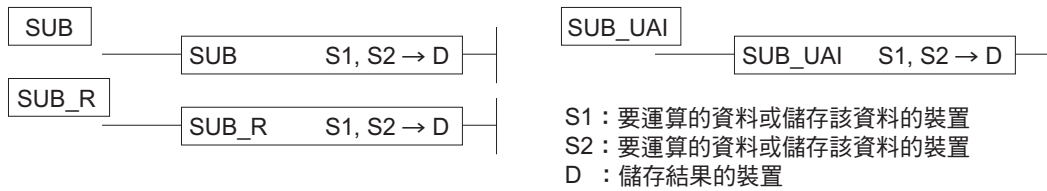
- 當運算結果超出資料類型範圍時



註：當非實數類型的資料超出資料類型範圍時，不會執行邊界數值處理。
請小心不要讓運算結果超出資料型範圍。實數類型資料則會執行邊界數值處理。

(2) 減 (SUB)

符號



SUB : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
SUB_UAI : 整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
SUB_R : 實數類型 (單精準浮點類型) [REAL]

可用裝置

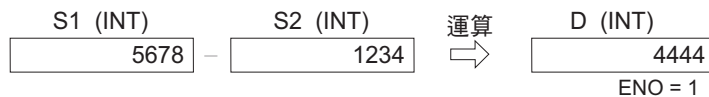
	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-

功能

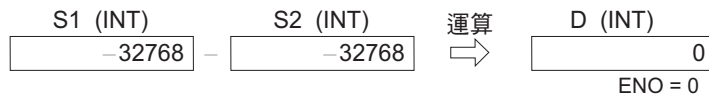
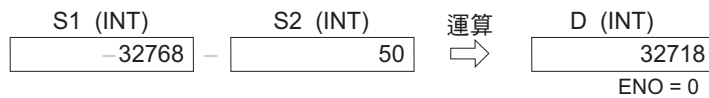
- 1) 從輸入資料 "S1" 中減去 "S2", 並將結果輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2", 且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 當運算結果超出資料類型範圍時, ENO會變為0。
- 4) 當資料類型為實數, 且非常接近0, 而無法以實數資料表示時, 輸出會變為0, ENO變為1。

<運算>

- 當運算結果在資料類型範圍以內時



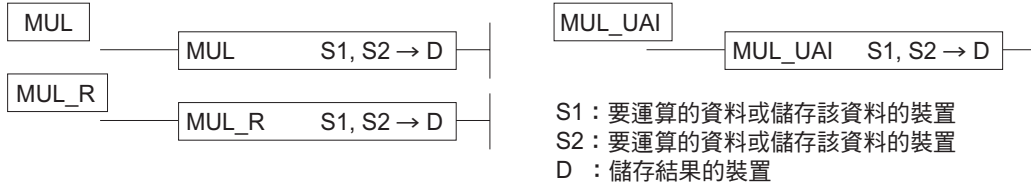
- 當運算結果超出資料類型範圍時



註：當非實數類型的資料超出資料類型範圍時, 不會執行邊界數值處理。
請小心不要讓運算結果超出資料型範圍。實數類型資料則會執行邊界數值處理。

(3) 乘 (MUL)

符號



- MUL : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
- MUL_UAI : 整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
- MUL_R : 實數類型 (單精準浮點類型) [REAL]

可用裝置

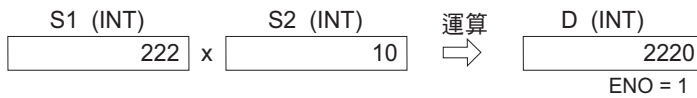
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

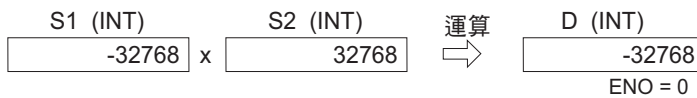
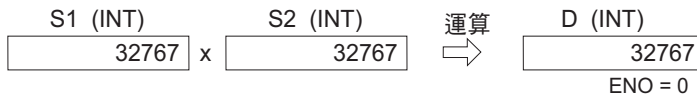
- 1) 將輸入資料 "S1" 與 "S2" 相乘，並將結果輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 當運算結果超出資料類型的邊界數值時，會輸出資料類型的邊界數值，且 ENO 會變為 0。
- 4) 當資料類型為實數，且非常接近 0，而無法以實數資料表示時，輸出會變為 0，ENO 變為 1。

<運算>

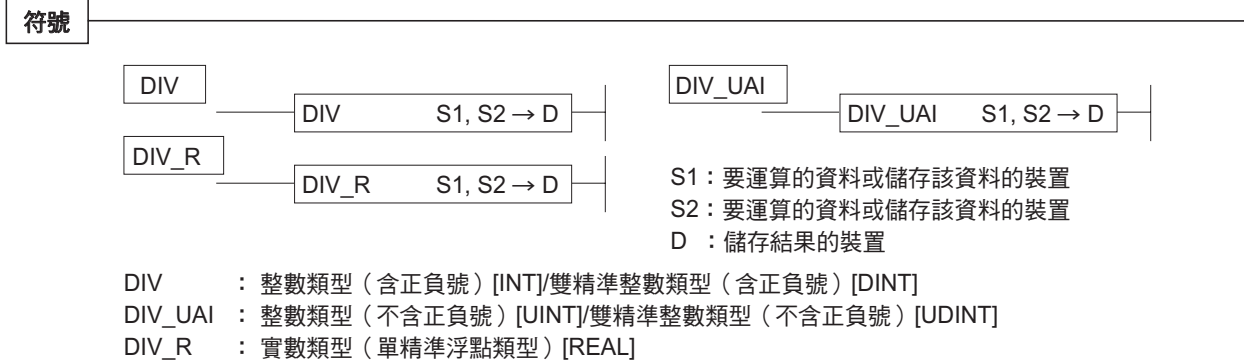
- 當運算結果在資料類型範圍以內時



- 當運算結果超出資料類型範圍時



(4) 除 (DIV)



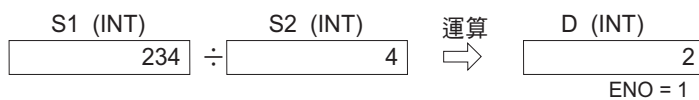
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—

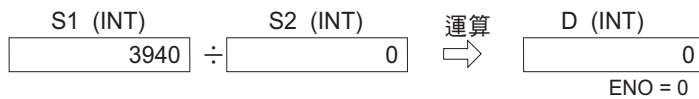
- 功能**
- 1) 以 "S2" 除輸入資料 "S1", 並將結果輸出至 "D"。
 - 2) 輸入裝置 "S1" 與 "S2", 且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
 - 3) 當運算結果超出資料類型的邊界數值時, 會輸出資料類型的邊界數值, 且ENO會變為0。
 - 4) 當資料類型為實數, 且非常接近0, 而無法以實數資料表示時, 輸出會變為0, ENO變為1。
 - 5) 當除數為0時, 會輸出符合被除數正負號的最大數值, 且ENO會變為0。

<運算>

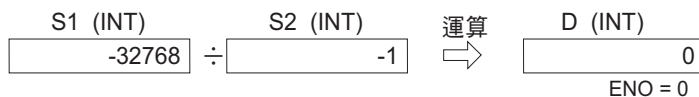
- 當運算結果在資料類型範圍以內時



- 當除數為 0 時



- 當商數超出資料類型範圍時



(5) 除餘 (MOD)

符號



S1：要運算的資料或儲存該資料的裝置
 S2：要運算的資料或儲存該資料的裝置
 D：儲存結果的裝置

MOD：整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
 MOD_UAI：整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
 MOD_R：實數類型 (單精準浮點類型) [REAL]

可用裝置

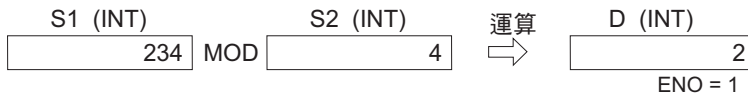
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

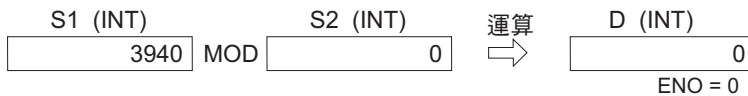
- 1) 以 "S2" 除輸入資料 "S1"，並將餘數輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 如果商數超出資料類型的邊界數值，輸出會變為0，且ENO變為0。
- 4) 當除數為0時，會輸出符合被除數正負號的最大數值，且ENO會變為0。
- 5) 運算的執行會使 (除數) x (商數) + (餘數) = (被除數)。

<運算>

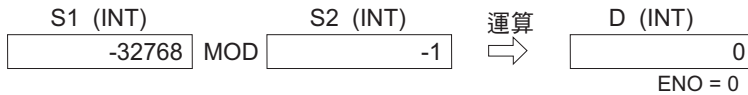
- 當運算結果在資料類型範圍以內時



- 當除數為0時



- 當商數超出資料類型範圍時



(6) 指數 (EXPT)

符號



S1：要運算的資料或儲存該資料的裝置
 S2：要運算的資料或儲存該資料的裝置
 D：儲存結果的裝置

可用裝置

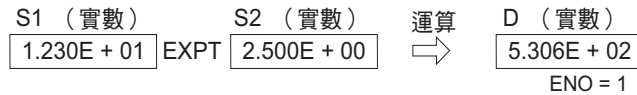
	字裝置 (W*)							雙字裝置 (D*)							常數				
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T		TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	-	○	○	-	○	○	○	-	-	-	○	○	-	

功能

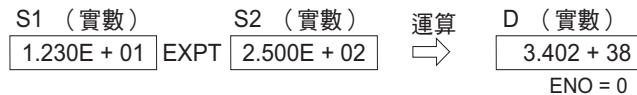
- 1) 將輸入資料 "S1" (基數) 由 "S2" (指數) 取冪，並將結果輸出至 "D"。
- 2) 當運算結果超出實數類型的邊界數值，輸出會變為0，且ENO變為0。
- 3) 基數 ≥ 0。當基數 < 0 時，輸出會變為0，且ENO變為0。
- 4) 輸出中的有效數字是4。
- 5) 當輸出數值超出實數類型的邊界數值時，ENO會變為0。當輸出數值非常接近0，使其無法以實數資料來表示時，輸出會變為0，且ENO變為1。
- 6) 當基數 = 0且指數 = 0 (0fl) 時，輸出會變為1，且ENO變為1。

<運算>

- 當運算結果在資料類型範圍以內時



- 當運算結果超出資料類型範圍時



(7) 絕對值 (ABS)

符號



ABS : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
 ABS_R : 實數類型 (單精準浮點類型) [REAL]

S : 要運算的資料或儲存該資料的裝置
 D : 儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	○	○	○	○	-	○	○	○	○	○	○	○	-	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-

功能

- 1) 取得輸入資料 "S" 的絕對值並將結果輸出至 "D"。
- 2) 當輸入數值為最大負值時，輸出該輸出資料類型的最大正值。
- 3) 輸入裝置 "S"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。

<運算>

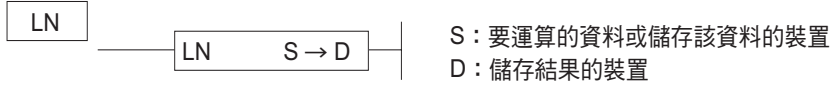


- 當輸入數值為最大負值時 (-32768)



(9) 自然對數 (LN)

符號



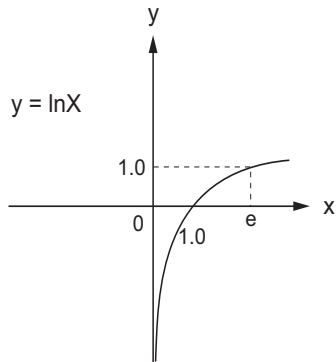
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

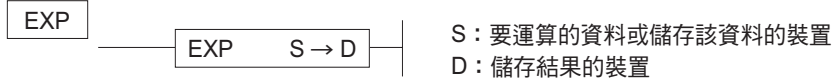
- 1) 取得輸入資料"S"的平方根 (實數類型, (單精準浮點類型) [REAL]) 並將結果輸出至"D"。
- 2) 當輸入數值為負時, 輸出0且ENO變為0。
- 3) 當輸入數值為0時, 輸出會變為最大負值, 且ENO變為0。
- 4) 輸出的有效位數是4。

<運算>



(11) 指數 (EXP)

符號



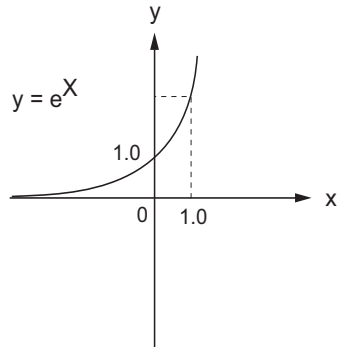
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

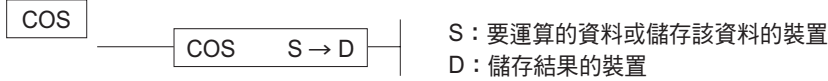
- 1) 以2.718281為基數 (e) 對"S"執行指數運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至"D"。
- 2) 當運算結果超出實數類型的邊界數值時, 便輸出邊界數值。
- 3) 輸出的有效位數:
 - 當輸入/輸出在-64至64的範圍內時... 4
 - 以上情況以外... 錯誤將會增加。
- 4) 當輸出數值超出實數類型的邊界數值時, ENO會變為0。當輸出數值非常接近0, 使其無法以實數資料來表示時, 輸出會變為0, 且ENO變為1。

<運算>



(13) 餘弦 (COS)

符號



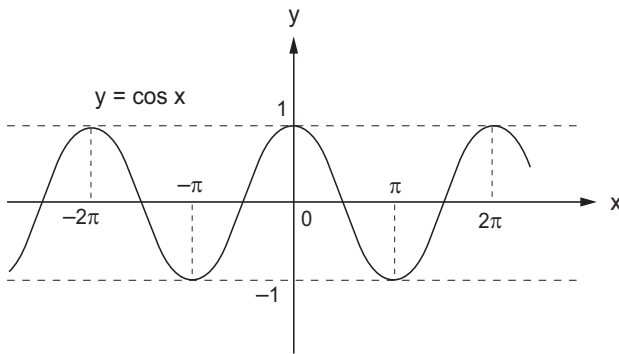
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

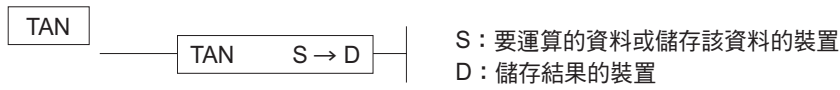
- 1) 對 "S" 執行餘弦運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸入的單位為弧度。
- 3) 當輸入在 -2π 至 2π 的範圍以內時，輸出的有效位數為5 (最多四位小數)。當輸入的絕對值為 2π 或以上時，將會執行運算，但錯誤將會增加。
- 4) 當 (輸入) $< -2^{31}$ 或 (輸入數值) $> 2^{31} - 1$ 時，輸出數值會變為0，且 ENO 變為0。

<運算>



(14) 正切 (TAN)

符號



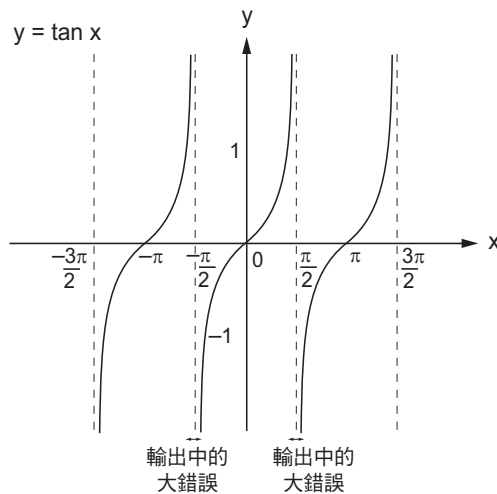
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

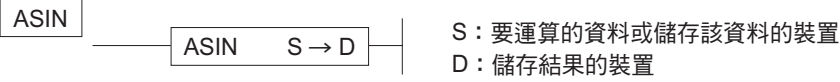
- 1) 對 "S" 執行正切運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸入的單位為弧度。
- 3) 當輸入在 -2π 至 2π 的範圍以內時, 輸出的有效位數為4, 但輸出中的錯誤將會增加, 接近 $\pi/2$ 的整數倍。當輸入的絕對時為 2π 或以上時, 將會執行運算, 但錯誤將會增加。
- 4) 當 (輸入) $< -2^{31}$ 或 (輸入數值) $> 2^{31} - 1$ 時, 輸出數值會變為0, 且ENO變為0。

<運算>



(15) 反正弦 (ASIN)

符號



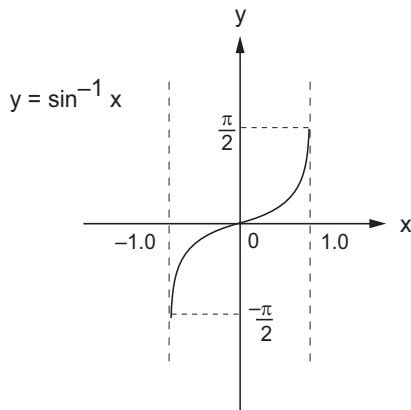
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

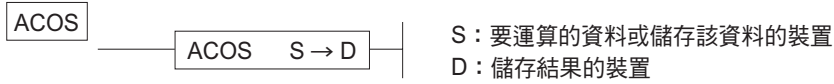
- 1) 對 "S" 執行反正弦運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸出的單位為弧度。
- 3) 輸入的範圍為 -1.0 至 +1.0，輸出範圍為 $-\pi/2$ 至 $\pi/2$ 。當輸入超出此範圍時，輸出會變為 0，且 ENO 變為 0。
- 4) 輸出的有效位數：
 - 當 |輸入數值| = 1.0 或 0.998999 或更少時... 4 位數
 - 當 |輸入數值| = 0.999 至 0.999999 時... 錯誤將會增加。

<運算>



(16) 反餘弦 (ACOS)

符號



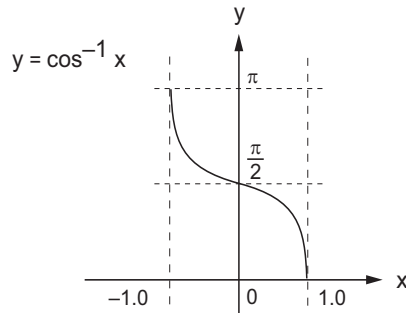
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

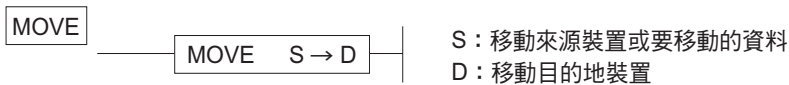
- 1) 對 "S" 執行反餘弦運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸出的單位為弧度。
- 3) 輸入的範圍為-1.0至+1.0，輸出範圍為 π 至0。當輸入超出此範圍時，輸出會變為0，且ENO變為0。
- 4) 輸出的有效位數：
 當 |輸入數值| = 1.0或0.998999或更少時... 4位數
 當 |輸入數值| = 0.999至0.999999時... 錯誤將會增加

<運算>



(18) 移動 (MOVE)

符號



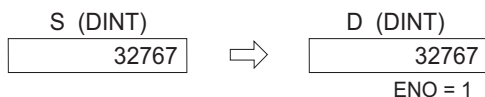
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

MOVE 將輸入資料 "S" 直接移動到 "D"。

<運算>



註：輸入裝置 "S" 與輸出裝置 "D" 中的資料必須有相同的位元長度。

(19) 負 (NEG)

符號



NEG : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
 NEG_R : 實數類型 (單精準浮點類型) [REAL]

S : 輸入資料或儲存輸入資料的裝置
 D : 儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	○	-

功能

- 1) 反轉輸入資料 "S" 的正負號，並將結果輸出至"D"。
- 2) 可用資料類型為整數類型、雙整數類型及實數類型。
- 3) 當資料類型為整數類型或雙整數類型，且運算結果超出資料類型範圍時，ENO會變為0。

<運算>



● 針對雙整數類型或整數類型的負邊界數值



(20) 分塊移動 (BMOV)

符號



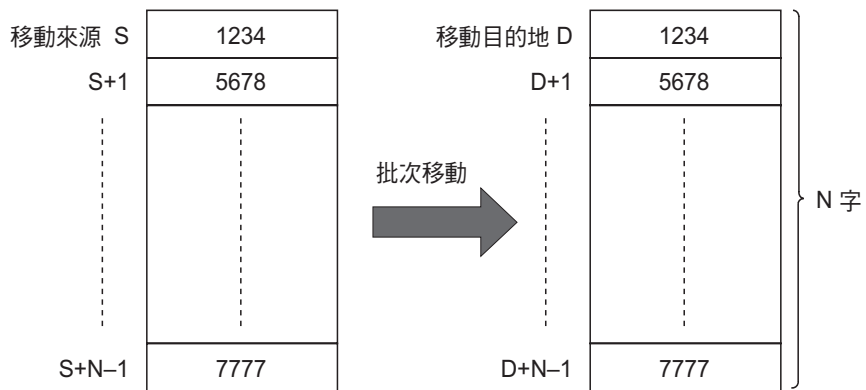
S : 要從其中移出的資料的標頭裝置
 D : 要移至的資料的標頭裝置
 N : 要移動的字數或儲存這些字的裝置 (範圍1至32767)

可用裝置

	位元裝置										字裝置 (W*)						雙字裝置 (D*)						常數					
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L		SM	T	TR	F	V
S1	○	○	○	○	○	—	—	—	○	○	○	○	○	○	○	—	○	○	○	○	○	○	○	—	—	○	○	—
S1	—	○	○	○	—	—	—	—	○	○	—	○	○	○	—	—	○	○	—	○	○	○	—	—	—	○	○	—
D	—	—	—	—	—	—	—	—	—	—	○	—	○	○	—	○	○	○	—	—	○	○	—	—	—	○	○	○

功能

BMOV 1) 一次從在"S"中指定的裝置中將"N"字的資料 (要移動的字數) 移至在"D"中指定的裝置上開始的N字區域。



2)"S" (移動來源資料) 與 "D" (移動目的地資料) 中的資料必須有相同的位元寬度。

註：指定要移動的字數，這樣一來用於其他目的的裝置便不會被重複寫入。當間接指定要移動的數時，需要特別小心。

(21) 填滿移動 (FMOV)

符號

FMOV S, D, N

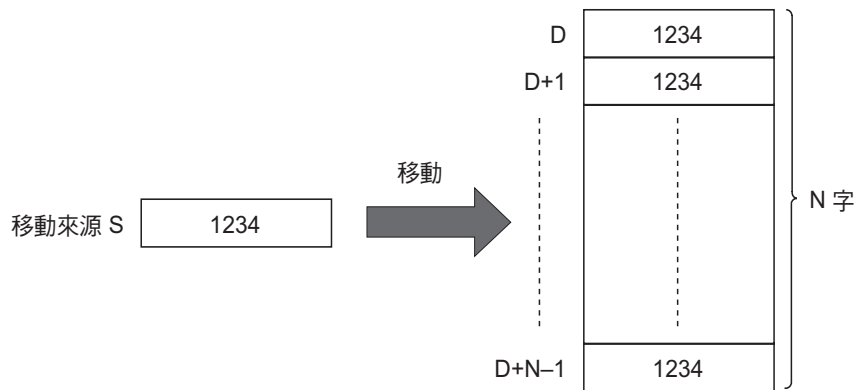
S：要移動的資料或儲存資料的裝置
 D：移動目的地的標頭裝置
 N：要移動的字數或儲存這些字的裝置（範圍1至32767）

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	-	○	○	○	-	-	○	○	-	-	-	-	-	-	-	-	-	-	-
D	○	○	○	○	○	○	○	○	-	-	○	○	-	-	-	○	○	○	○

功能

FMOV 將在"D"中指定的裝置內容移動到在"D"中指定的裝置上開始的N字區域。



註：指定要移動的字數，這樣一來用於其他目的的裝置便不會被重複寫入。
 當間接指定要移動

(22) 交換 (XCH)

符號

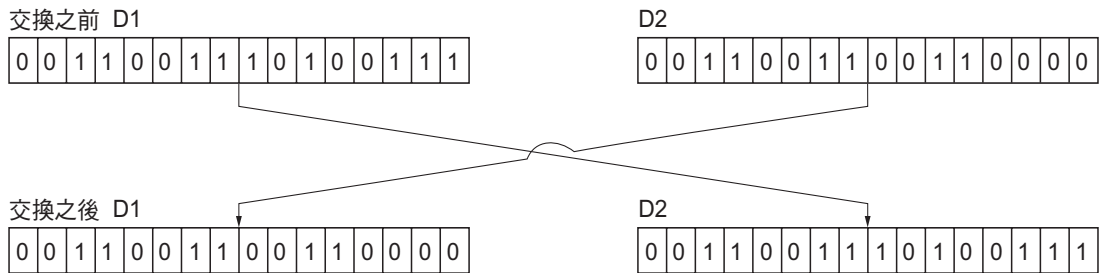


可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	○	○	○	○	—	○	○	○	○	○	○	○	—	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—

功能

XCH 交換"D1"中的資料與"D2"中的資料。



"D1"與"D2"中的資料必須有相同的位元寬度。

(23) 間接放置 (BDMPX)

符號

BDMPX — BDMPX S1, D, S2, N

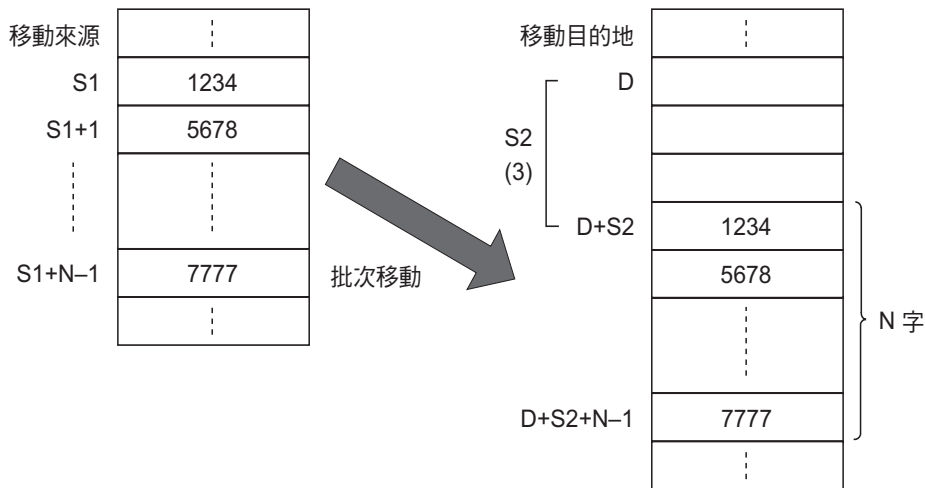
S1：移動來源裝置
 D：移動目的地裝置（參照位址）
 S2：資料指標
 N：要移動的字數（範圍：1至32767）

可用裝置

	位元裝置										字裝置 (W*)						雙字裝置 (D*)						常數					
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L		SM	T	TR	F	V
S1	○	○	○	○	○	—	—	—	○	○	○	○	○	○	○	—	○	○	○	○	○	○	○	—	—	○	○	○
D	—	○	○	○	—	—	—	○	○	—	○	○	○	—	—	○	○	—	○	○	○	—	—	—	○	○	—	
S2	—	—	—	—	—	—	—	—	—	○	—	○	○	—	○	○	○	○	—	○	○	—	—	—	○	○	○	
N	—	—	—	—	—	—	—	—	—	○	—	○	○	—	○	○	○	—	—	○	○	—	—	—	○	○	○	

功能

BDMPX 1) 從在"S1"中指定的裝置將"N"字（要移動的字數）的資料移動到從在"D"中指定的裝置（參照位址）"S2"中指定的裝置中的資料（資料指標）所取代的裝置裡。



2) "S1"（移動來源資料）與"D"（移動目的地資料）中的資料必須有相同的位元寬度。

註：指定要移動的字數，這樣一來用於其他目的的裝置便不會被重複寫入。
 當間接指定要移動的字數時，需要特別小心。

(24) 間接取出 (BMPX)

符號



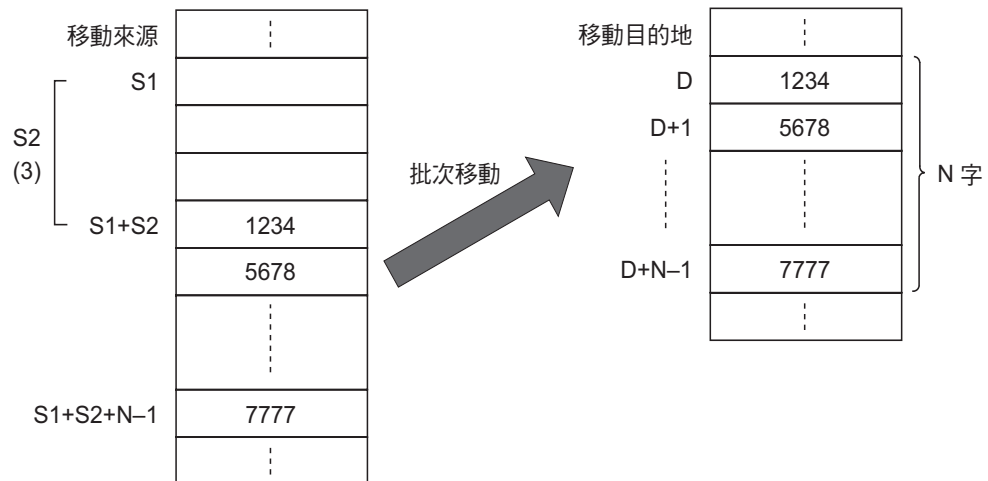
S1：移動來源裝置（參照位址）
 S2：資料指標
 D：移動目的地裝置
 N：要移動的字數（範圍：1至32767）

可用裝置

	位元裝置										字裝置 (W*)						雙字裝置 (D*)						常數					
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L		SM	T	TR	F	V
S1	○	○	○	○	○	-	-	-	○	○	○	○	○	○	○	-	○	○	○	○	○	○	○	-	-	○	○	-
S2	-	-	-	-	-	-	-	-	-	-	○	-	○	○	-	○	○	○	-	-	○	○	-	-	-	○	○	○
D	-	○	○	○	-	-	-	-	-	-	-	○	○	○	-	-	○	○	○	○	○	○	-	-	-	○	○	-
N	-	-	-	-	-	-	-	-	-	-	○	-	○	○	-	○	○	○	-	-	○	○	-	-	-	○	○	○

功能

BDMPX 1) 從在"S1"中指定的裝置將"N"字（要移動的字數）的資料移動到從"D"中指定的裝置（參照位址）中的"S2"中指定的裝置中的資料（資料指標）所取代的裝置裡。



2) "S"（移動來源資料）與 "D"（移動目的地資料）中的資料必須有相同的位元寬度。

註：指定要移動的字數，這樣一來用於用於其他目的的裝置便不會被重複寫入。
 當間接指定要移動

3-2-5 位元字串運算指令

(1) 邏輯 AND (AND_AW)

符號



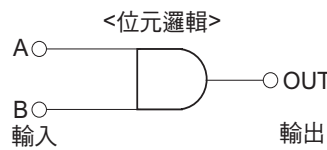
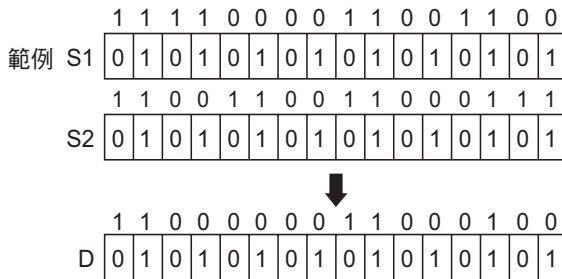
S1：儲存要進行邏輯AND的資料的裝置
 S2：要進行邏輯AND的資料或儲存資料的裝置
 D：儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

1) 對在"S1"中指定的裝置的資料及在"S2"中指定的裝置的資料（或常數）執行逐位元的邏輯AND，並將結果輸出到在"D"中指定的裝置。



A	B	OUT
0	0	0
1	0	0
0	1	0
1	1	1

2) "S1"與"S2"中的資料必須有相同的位元寬度。

(2) 邏輯 OR (OR_AW)

符號



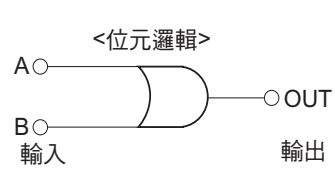
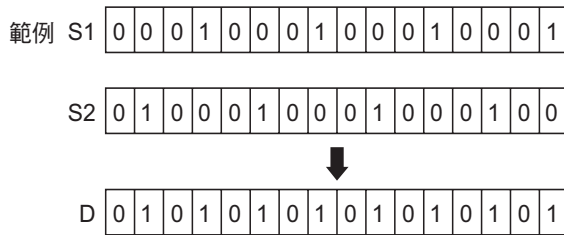
S1 : 儲存要進行邏輯OR的資料的裝置
 S2 : 要進行邏輯OR的資料或儲存資料的裝置
 D : 儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

1) 對在"S1" 中指定的裝置的資料及在"S2"中指定的裝置的資料 (或常數) 執行逐位元的邏輯OR，並將結果輸出到在"D"中指定的裝置。



A	B	OUT
0	0	0
1	0	1
0	1	1
1	1	1

2) "S1"與"S2"中的資料必須有相同的位元寬度。

(3) 邏輯互斥 OR (XOR)

符號

位元類型：

XOR_B

XOR_B S1, S2 → D

字/雙字類型：

XOR_AW

XOR_AW S1, S2 → D

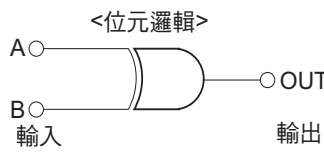
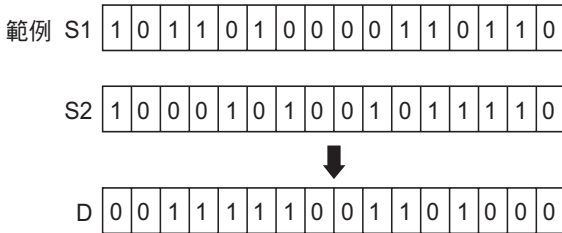
S1：儲存要進行邏輯互斥OR的資料的裝置
 S2：要進行邏輯互斥OR的資料或儲存資料的裝置
 D：儲存結果的裝置

可用裝置

	位元裝置										字裝置 (W*)						雙字裝置 (D*)						常數						
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L		SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	○	-	-	-	○	○	-	○	○	-	-	○	○	-	○	○	○	-	○	○	○	○	-	-	

功能

1) 對在"S1"中指定的裝置的資料及在"S2"中指定的裝置的資料（或常數）執行逐位元的邏輯互斥OR，並將結果輸出到在"D"中指定的裝置。



A	B	OUT
0	0	0
1	0	1
0	1	1
1	1	0

2) "S1 與"S2"中的資料必須有相同的位元寬度。

(4) 邏輯互斥 NOR (XORN)

符號

位元類型：



字/雙字類型：



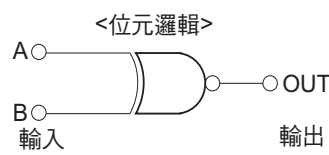
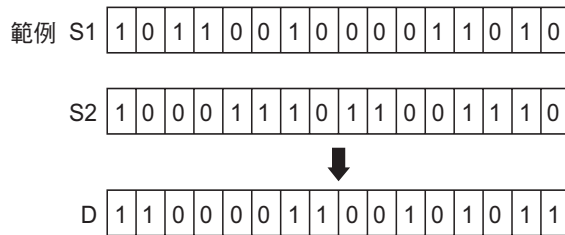
S1：儲存要進行邏輯互斥OR的資料的裝置
 S2：要進行邏輯互斥OR的資料或儲存資料的裝置
 D：儲存結果的裝置

可用裝置

	位元裝置										字裝置 (W*)							雙字裝置 (D*)							常數			
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T		TR	F	V
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	○	-	-	-	○	○	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-

功能

1) 對在"S1"中指定的裝置的資料及在"S2"中指定的裝置的資料（或常數）執行逐位元的邏輯互斥NOR，並將結果輸出到在"D"中指定的裝置。

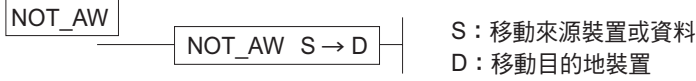


A	B	OUT
0	0	1
1	0	0
0	1	0
1	1	1

2) "S1"與"S2"中的資料必須有相同的位元寬度。

(5) 邏輯 NOT (NOT)

符號



可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

- 1) 反轉在"S"中指定的裝置中的資料的每一個位元，並將結果輸出至在"D"中指定的裝置中。

移動來源：S

0 0 1 1 0 1 0 0 1 1 1 0 0 0 1 1

↓ 反轉及移動

移動目的地：D

1 1 0 0 1 0 1 1 0 0 0 1 1 1 0 0

- 2) "S"與"D"中的資料必須有相同的位元寬度。

(6) 旋轉 (ROR、ROL)

符號

向右旋轉：

向左旋轉：

S：要旋轉的裝置
N：要旋轉的位元數 (N = 0至31)
D：儲存旋轉資料的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	-	-	-	-	-	-	-	-	-	○	
D	-	○	○	○	-	-	○	○	-	○	○	○	-	-	-	○	○	-	

功能

ROR 1) 以在"N (要旋轉的位元數) 中所指定的數值向右旋轉輸入資料"S (要旋轉的裝置)，並將結果輸出到"D"。

例如：向右旋轉3位元 (N = 3)

ROL 1) 以在"N (要旋轉的位元數) 中所指定的數值向左旋轉輸入資料"S (要旋轉的裝置)，並將結果輸出到"D"。

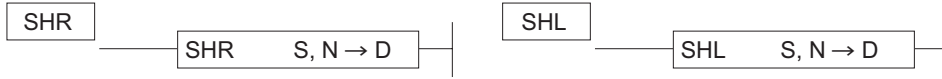
例如：向左旋轉3位元 (N = 3)

<關於要旋轉的位元數 "N" 的說明>
 當位元寬度為16位元時，低順序的四個位元對"N"是有效的。
 例如，當N = 16時，旋轉大小為0位元；當N = 17時，旋轉大小為1位元。當位元寬度為32位元時，低順序的五個位元對"N"是有效的。
 例如，當N = 32時，旋轉大小為0位元；當N = 33時，旋轉大小為1位元。

* "S"與"D"中的資料必須有相同的位元寬度。

(7) 移位 (SHR、SHL)

符號



S：要移位的裝置
N：要移位的位元數 (N = 0至31)
D：儲存移位資料的裝置

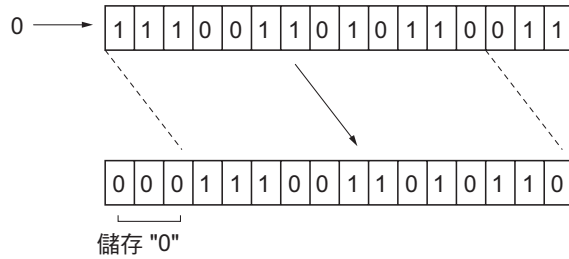
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	-	-	-	-	-	-	-	-	-	○	
D	-	○	○	○	-	-	○	○	-	○	○	○	-	-	-	○	○	-	

功能

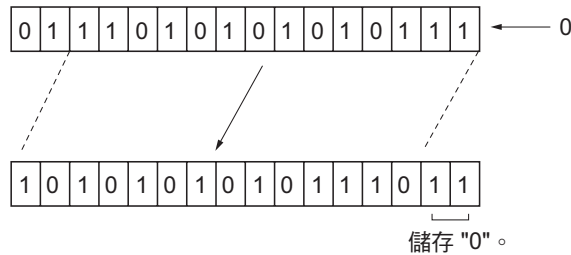
SHR 1) 將輸入資料"S"往右移位"N"位元，在由移位所清空的位元中放置 0，然後將結果輸出到在"D"中指定的裝置中。

例如：向右移位3位元 (N = 3)



SHL 1) 將輸入資料"S"往左移位"N"位元，在由移位所清空的位元中放置 0，然後將結果輸出到在"D"中指定的裝置中。

例如：向左旋轉3位元 (N = 3)



<關於要旋轉的位元數 "N" 的說明>

當位元寬度為16位元時，低順序的四個位元對 "N"是有效的。

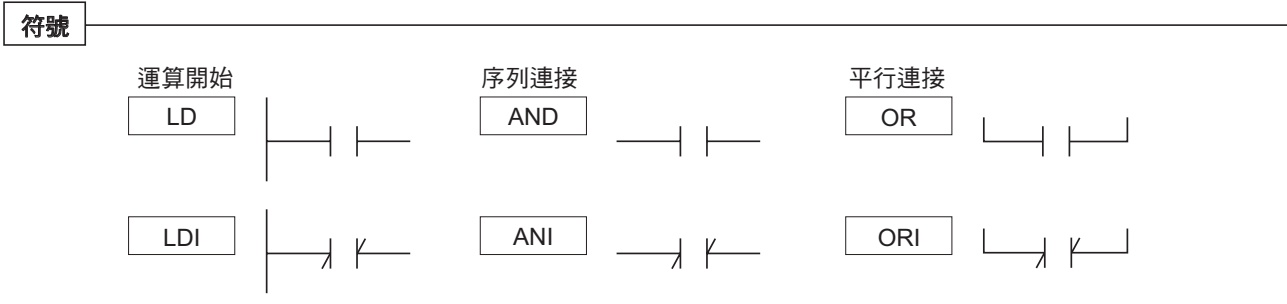
例如，當N = 16時，旋轉大小為0位元；當N = 17時，旋轉大小為1位元。當位元寬度為32位元時，低順序的五個位元對"N"是有效的。

例如，當N = 32時，旋轉大小為0位元；當N = 33時，旋轉大小為1位元。

* "S"與"D"中的資料必須有相同的位元寬度。

3-2-1 基本指令

(1) 運算開始、序列連接及平行連接

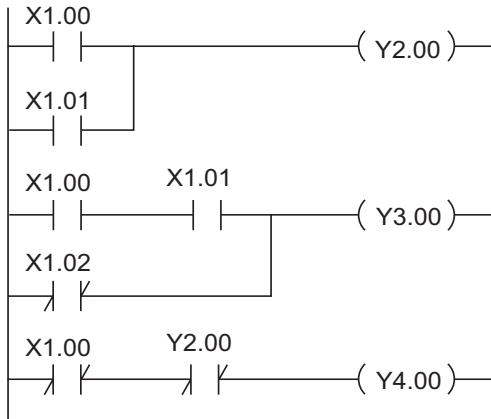


可用裝置										
位元裝置										
X	Y	M	L	SM	T	TR	C	F	V	
○	○	○	○	○	○	○	○	○	○	○

- | 功能 | | |
|-----|-----|---|
| LD | LDI | 1) LD指令是開始運算的NO接觸點指令。LDI指令是開始運算的NC接觸點。每一個指令都會取得指定裝置的「開啟」或「關閉」狀態做為運算結果。 |
| AND | ANI | 1) AND指令會以序列方式連接NO接觸點。ANI指令會以序列方式連接NC接觸點。這些指令會AND指定裝置的「開啟」或「關閉」狀態與之前運算的結果。此AND運算的結果即為指令的結果。 |
| OR | ORI | 1) OR指令會以平行方式連接NO接觸點。ORI指令會以平行方式連接NC接觸點。這些指令會OR指定裝置的「開啟」或「關閉」狀態與之前運算的結果。此OR運算的結果即為指令的結果。 |

程式範例

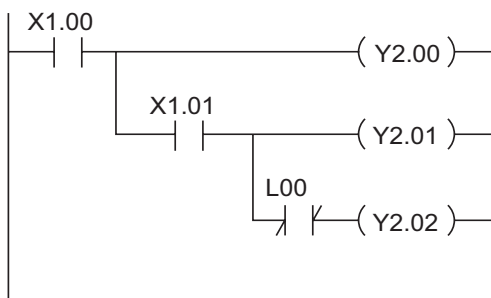
1) 階梯圖表表示法



(助記表示法)

指令	裝置		
LD	X1.00		
OR	X1.01		
OUT	Y2.00		
LD	X1.00		
AND	X1.01		
OUT	Y3.00		
LDI	X1.00		
ANI	Y2.00		
OUT	Y4.00		

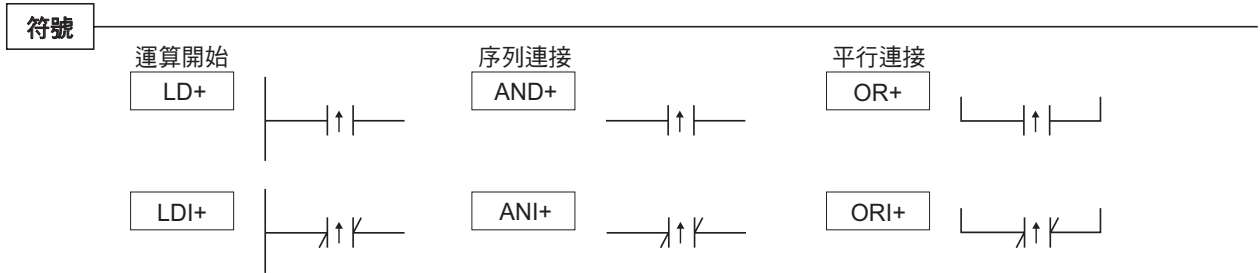
2) 階梯圖表表示法



(記憶表示法)

指令	裝置		
LD	X1.00		
OUT	Y2.00		
AND	X1.01		
OUT	Y2.01		
ANI	L00		
OUT	Y2.02		

(2) 上升緣變異



可用裝置										
位元裝置										
X	Y	M	L	SM	T	TR	C	F	V	
○	○	○	○	○	○	○	○	○	○	○

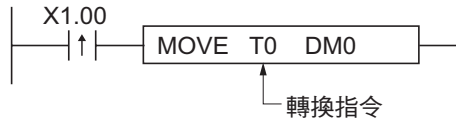
指出上升緣變異正常開啟接觸點FB與無正負號實數（UNIT形式）的實例數。

- | 功能 | | |
|------|------|---|
| LD+ | LDI+ | 1) LD+指令是開始運算的「關閉」對「開啟」變異NO接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。LDI+指令是開始運算的「關閉」對「開啟」變異NC接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「關閉」。 |
| AND+ | ANI+ | 1) AND+指令是以序列方式連接NO接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「開啟」，然後它會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。ANI+指令是以序列方式連接NC接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。 |
| OR+ | ORI+ | 1) OR+指令是以平行方式連接NO接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「開啟」，然後它會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。ORI+指令是以平行方式連接NC接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「關閉」對「開啟」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。 |

程式範例

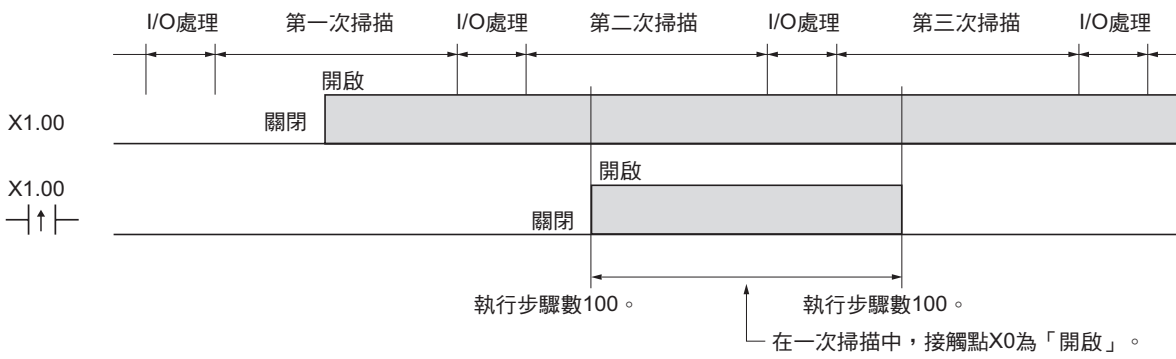
1) 當接觸點X1.00設定為「開啟」時，T0（計時器目前數值）只會傳輸一次至DM0。

(階梯圖表表示法)



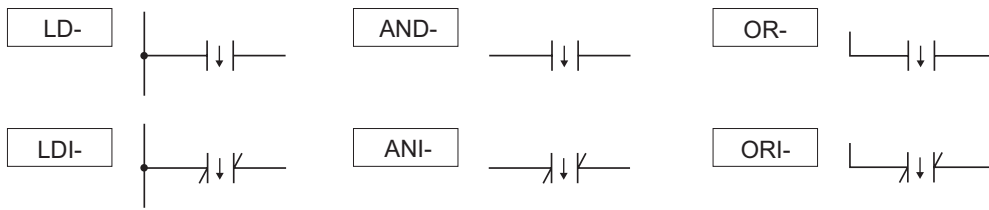
(記憶表示法)

指令	裝置		
LD+	X1.00	0	
MOVE	T0	DM0	



(3) 下降緣變異

符號



可用裝置

位元裝置									
X	Y	M	L	SM	T	TR	C	F	V
○	○	○	○	○	○	○	○	○	○

指出下降緣變異正常開啟接觸點FB與無正負號實數（UNIT形式）的實例數。

功能

- | | | |
|------|------|--|
| LD- | LDI- | 1) LD-指令是開始運算的「開啟」對「關閉」變異NO接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。LDI-指令是開始運算的「關閉」對「開啟」變異NC接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「關閉」。 |
| AND- | ANI- | 1) AND-指令是以序列方式連接NO接觸點的「關閉」對「開啟」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。然後這個指令會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。ANI-指令是以序列方式連接NC接觸點的「開啟」對「關閉」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會AND執行程式的結果與之前運算的結果。此AND運算的結果會成為指令的結果。 |
| OR- | ORI- | 1) OR-指令是以平行方式連接NO接觸點的「開啟」對「關閉」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「開啟」。然後這個指令會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。ORI-指令是以平行方式連接NC接觸點的「開啟」對「關閉」變異接觸點指令。此指令可於偵測到來自指定裝置的「開啟」對「關閉」轉換時為一次程式掃描循環設定指定的接觸點「關閉」，然後它會OR執行程式的結果與之前運算的結果。此OR運算的結果會成為指令的結果。 |

程式範例

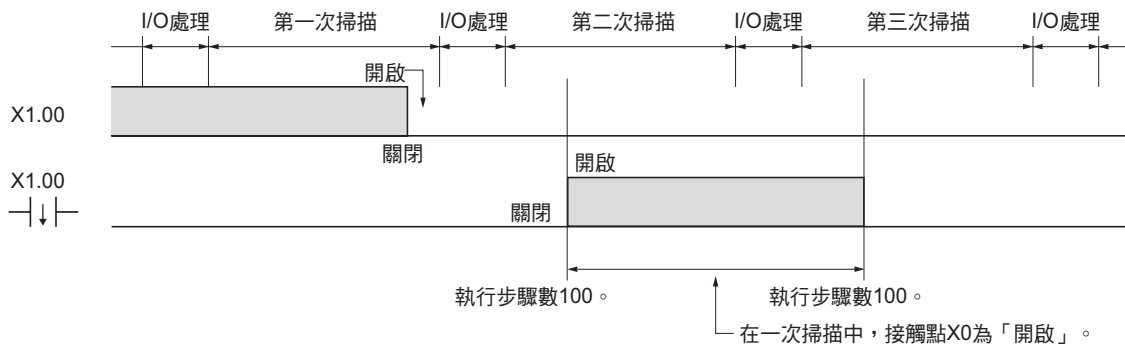
1) 當接觸點X1.00設定為「關閉」時，C0（計數器目前數值）只會傳輸一次至D0。

（階梯圖表表示法）



（記憶表示法）

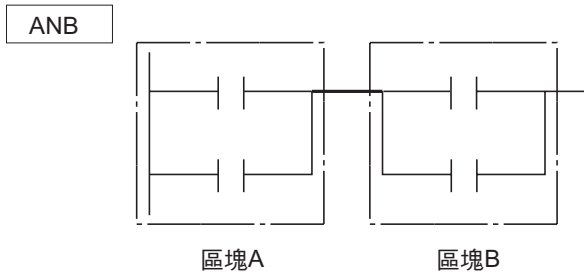
指令	裝置	
LD-	X1.00	0
MOVE	C0	DM0



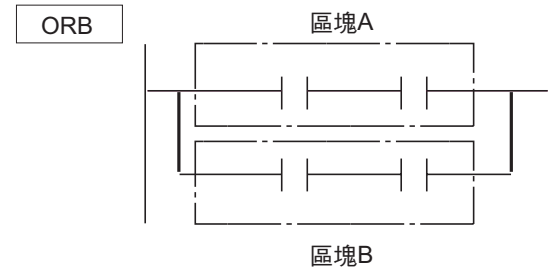
(4) 電路區塊的序列連接與平行連接

符號

電路區塊的序列連接



電路區塊的平行連接



功能

ANB

1) ANB指令可AND區塊A與B。此AND運算的結果會成為指令的結果。

ORB

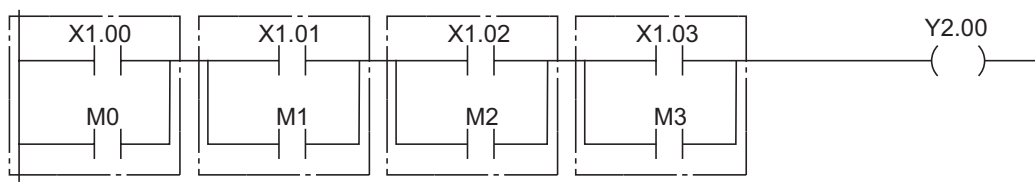
1) ORB指令可OR區塊A與B。此OR運算的結果會成為指令的結果。

2) ORB會以平行方式連接有二個以上接觸點的電路區塊。只有一個接觸點的電路區塊可以使用OR或ORI指令以平行方式連接。ORB指令不適用此用途。

程式範例

ANB

(階梯圖表表示法)



有兩種記憶語言程式設計可以序列方式連續連接電路區塊。如果選擇方法2)，所能使用的ANB指令有限。

(記憶表示法1)

指令	裝置		
LD	X1.00		
OR	M0		
LD	X1.01		
OR	M1		
ANB			
LD	X1.02		
OR	M2		
ANB			
LD	X1.03		
OR	M3		
ANB			
OUT	Y2.00		

* 可以使用的ANB指令數無限。

(記憶表示法2)

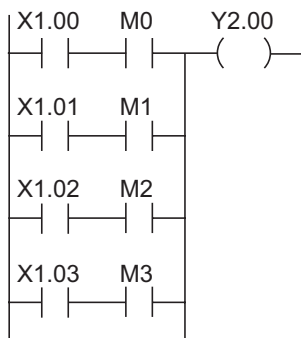
指令	裝置		
LD	X1.00		
OR	M0		
LD	X1.01		
OR	M1		
LD	X1.02		
OR	M2		
LD	X1.03		
OR	M3		
ANB			
ANB			
OUT	Y2.00		

* 最多可連續寫入的ANB指令為七個 (八個區塊)。

ORB

有兩種記憶語言程式設計可以平行方式連續連接電路區塊。

(階梯圖表表示)



(記憶表示法1)

指令	裝置		
LD	X1.00		
AND	M0		
LD	X1.01		
AND	M1		
ORB			
LD	X1.02		
AND	M2		
ORB			
LD	X1.03		
AND	M3		
ORB			
OUT	Y2.00		

* 可以使用的ORB指令數無限。

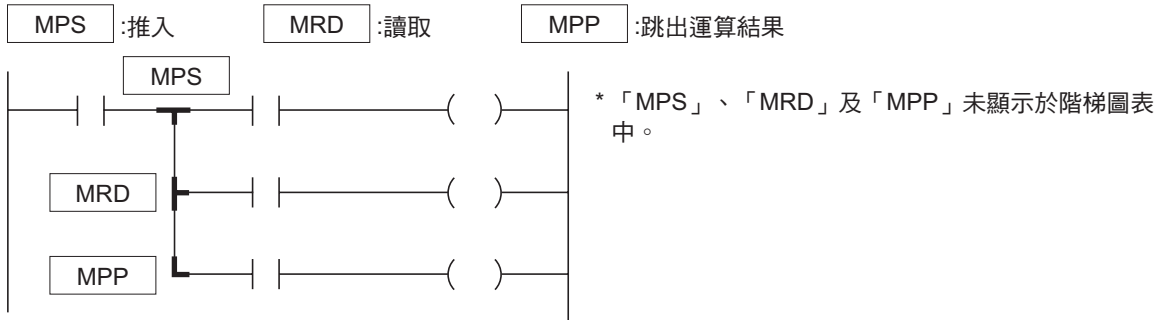
(記憶表示法2)

指令	裝置		
LD	X1.00		
AND	M0		
LD	X1.01		
AND	M1		
LD	X1.02		
AND	M2		
LD	X1.03		
AND	M3		
ORB			
ORB			
OUT	Y2.00		

* 最多可連續寫入的ORB指令為七個 (八個區塊)。

(5) 運算結果的推入、讀取及跳出

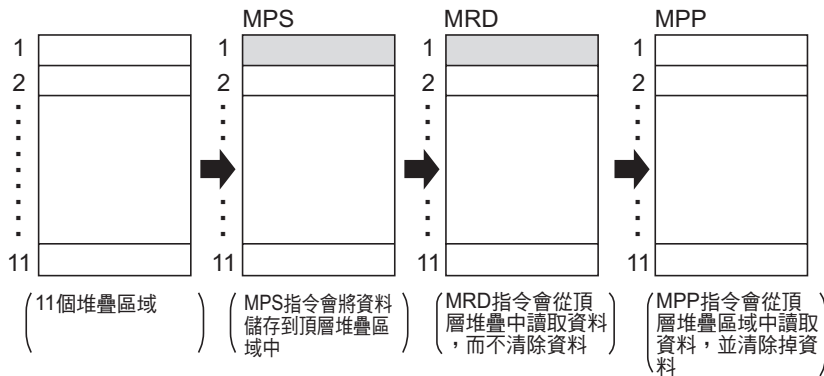
符號



功能

- MPS** 1) MPS指令會在MPS指令之前儲存運算結果（「開啟」或「關閉」狀態）。
2) 最多可連續寫入11個MPS指令。
- MRD** 1) MRD指令會讀取由MPS指令所儲存的運算結果。然後它會根據讀取運算的結果繼續從下一個步驟運算。
- MPP** 1) MPP指令會讀取由MPS指令所儲存的運算結果。然後它會根據讀取運算的結果繼續從下一個步驟運算。
2) MPP指令會清除由MPS指令所儲存的結果。

<Strage (堆疊) 區域運算>



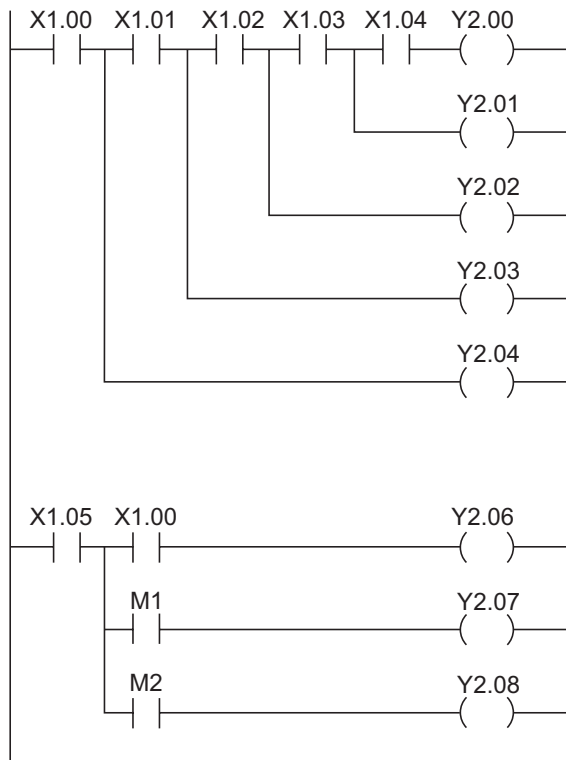
程式範例

MPS

MRD

MPP

(階梯圖表表示法)



重點

在電路中，MPS指令數必須與MPP指令數相同。如果指令數不同，則會偵測到程式錯誤，PC將無法正常作業。

(記憶表示法)

指令	裝置		
LD	X1.00		
MPS			
AND	X1.01		
MPS			
AND	X1.02		
MPS			
AND	X1.03		
MPS			
AND	X1.04		
OUT	Y2.00		
MPP			
OUT	Y2.01		
MPP			
OUT	Y2.02		
MPP			
OUT	Y2.03		
MPP			
OUT	Y2.04		
LD	X1.05		
MPS			
AND	M0		
OUT	Y2.06		
MRD			
AND	M1		
OUT	Y2.07		
MPP			
AND	M2		
OUT	Y2.08		

(6) 線圈、反向線圈

符號



可用裝置

位元裝置										
X	Y	M	L	SM	T	TR	C	F	V	
-	○	○	○	○	-	-	-	○	○	

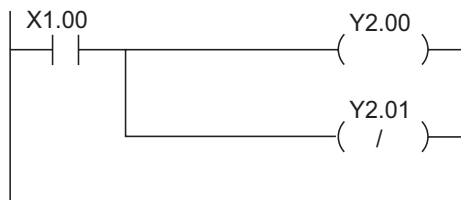
功能

- OUT** 可將上至「OUT」指令的運算結果輸出到指定裝置。
- OUTI** 可反轉上至「OUTI」指令的運算結果，並將結果輸出到指定裝置。

程式範例

OUT , **OUTI** 當X1.00「開啟」時，Y2.00也會「開啟」，而Y2.01則會「關閉」。

(階梯圖表表示法)



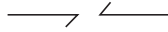
(記憶表示法)

指令	裝置		
LD	X1.00		
OUT	Y2.00		
OUTI	Y2.01		

(8) 反轉

符號

NOT



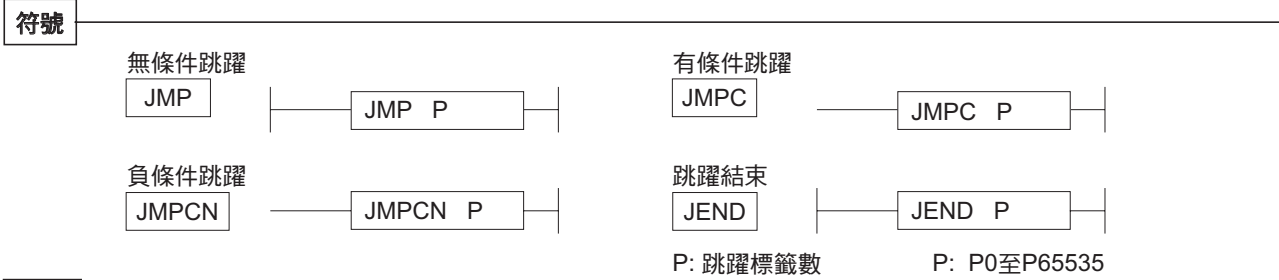
功能

NOT

NOT指令可將位於左方的邏輯線路數值反轉，並將結果傳送到位於其右的邏輯線路中。

3-2-2 程式控制指令

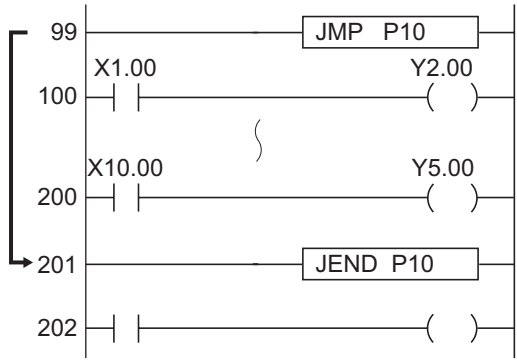
(1) 跳躍 / 跳躍結束



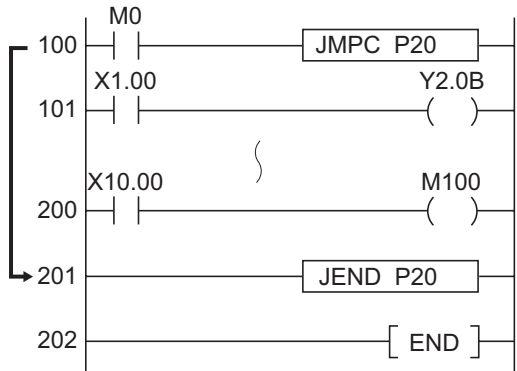
- | 功能 | |
|-------|---|
| JMP | 跳到運算元中所指定的標籤位置。被跳過的指令將不執行。 |
| JMPC | 當上至目前時間的運算結果為「開啟」時，可跳至運算元中所指定的標籤位置。
當上至目前時間的運算結果為「關閉」時，則執行下一個指令。 |
| JMPCN | 當上至目前時間的運算結果為「關閉」時，可跳至運算元中所指定的標籤位置。
當上至目前時間的運算結果為「開啟」時，則執行下一個指令。 |
| JEND | 將跳躍標籤加到任意位置上。這必須跟各個JMP配對使用。 |

程式範例

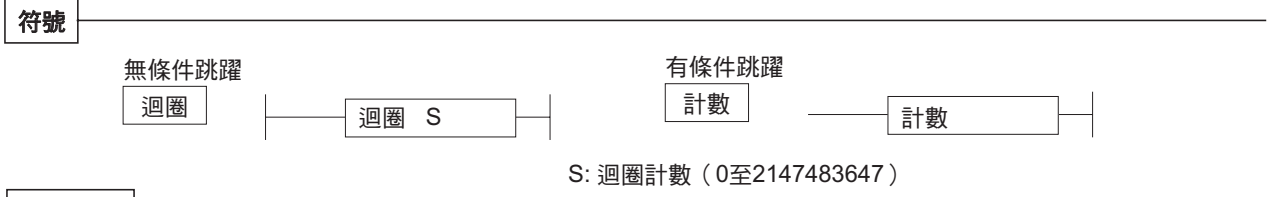
JMP 無條件在程式中從第100個電路跳到第200個電路。
(階梯圖表表示法)



JMPC
(階梯圖表表示法)



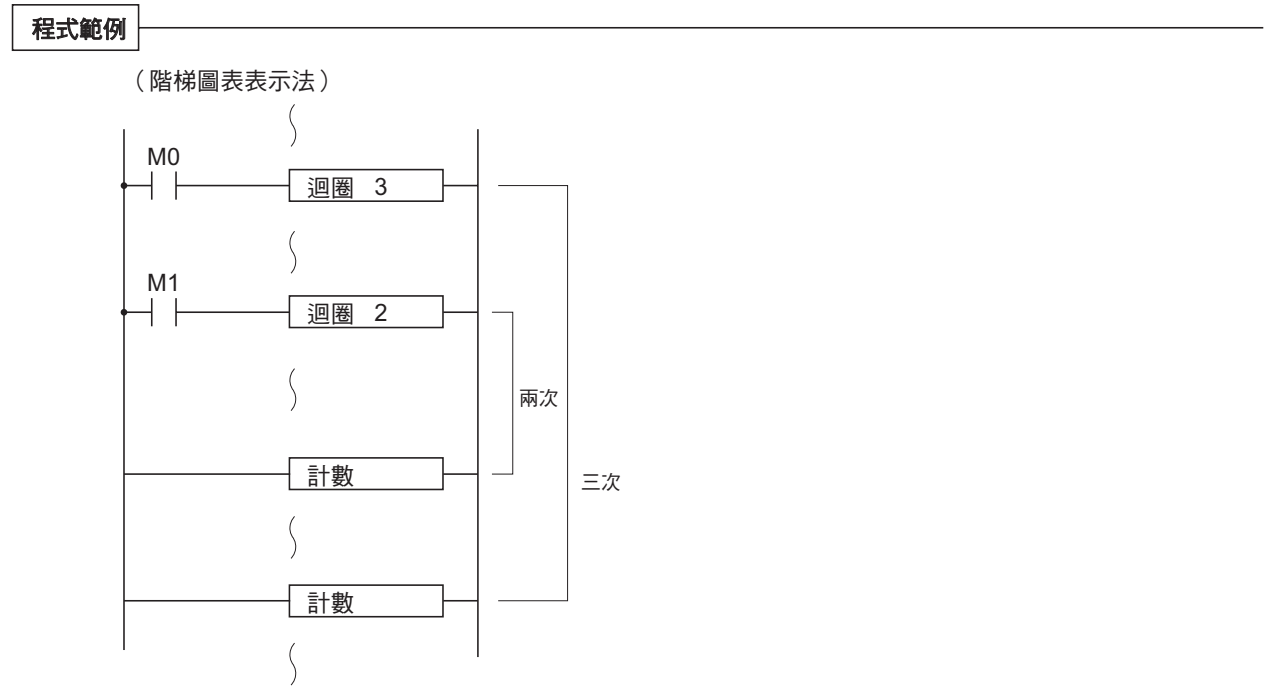
(2) 迴圈、迴圈結束



可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	
S	-	-	-	-	-	-	-	-	-	-	○	○	-	-	-	○	○

- 功能**
- 1) 在執行LOOP至CONT指令 "S"次之後 (次數以S表示)，則執行CONT之後的步驟。
 - 2) 允許的巢狀結構層數最高8層。
 - 3) "S"的允許範圍從0到2147483647。在 "S" 中指定負值時，則假定S=1。
 - 4) 如果在LOOP指令之前放了一個CONT指令、不存在CONT指令、在FEND或END指令之後放了一個CONT指令、LOOP指令數與CONT指令數不符或發生超過層數的情形時，都會發生程式錯誤。
 - 5) 在執行 "LOOP - CONT" 的過程中，更改迴圈計數是無效的。
- 註：當運算花很長時間進行高迴圈計數時，便可能會發生監視計時器錯誤。



(3) 返回指令

符號

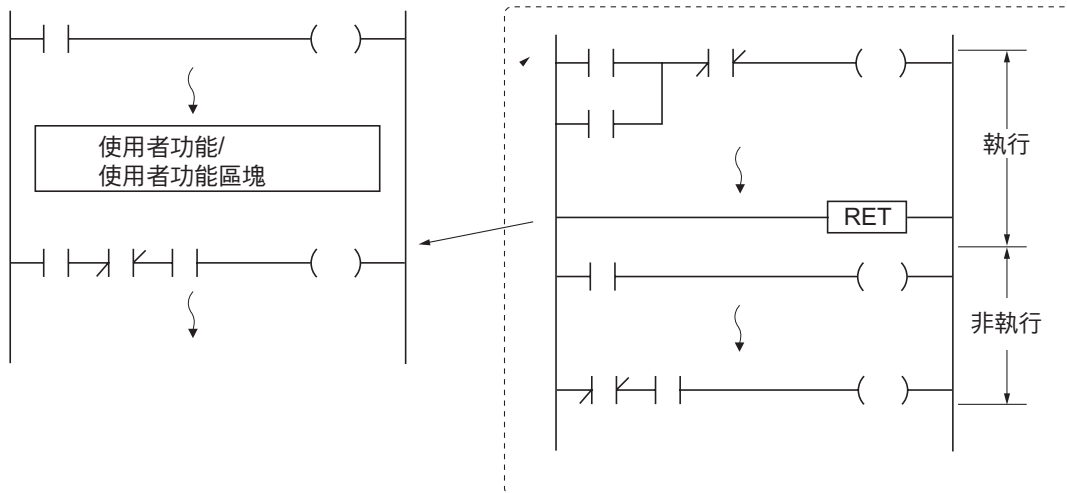


功能

- RET**
- 1) 在使用者功能/使用者功能區塊中使用它時，會無條件返回到功能呼叫位置之後的步驟。
 - 2) 在程式中使用它時，會無條件返回到程式的最後位置。
- RETC**
- 1) 上至目前時間的運算結果為「開啟」時，則執行 "- [RET] -"。
 - 2) 上至目前時間的運算結果為「關閉」時，則執行下一個指令。
- RETCN**
- 3) 上至目前時間的運算結果為「關閉」時，則執行 "- [RET] -"。
 - 4) 上至目前時間的運算結果為「開啟」時，則執行下一個指令。

<運算>

執行RET指令



3-2-3 轉換指令

(1) BIN 至 BCD (BCD)

符號

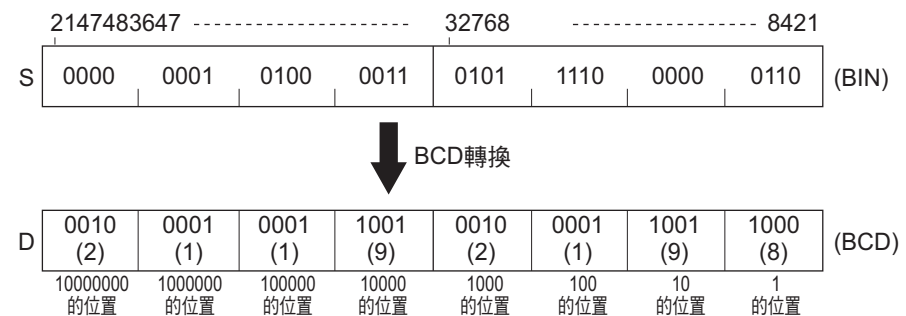


可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—	

功能

BCD 1) 將指定於 "S" 中的裝置之BIN資料 (16或32位元) 轉換為BCD資料, 並將結果輸出至指定於 "D" 中的裝置。



- 2) 指定於 "S" 與 "D" 中的數值必須擁有相同的位元寬度。
- 3) 資料範圍是：
 指定字裝置時：0至9999
 指定雙字裝置時：0至99999999
 如果BIN資料超出指定的資料範圍，則會輸出 "9999" (指定字裝置時) 或 "99999999" (指定雙字裝置時)。

(2) BCD 至 BIN (BIN)

符號

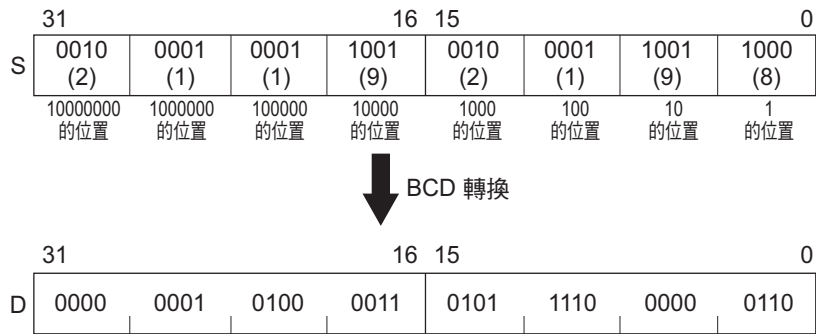


可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—	

功能

1) 轉換 "S" 中指定的裝置中的BCD資料 (16位元資料為0至9999，或32位元資料為0至99999999)，並將結果輸出至在 "D" 中所指定的裝置中。



2) 指定於 "S" 與 "D" 中的數值必須擁有相同的位元寬度。

(3) BIN 至 real (I_TO_R)

符號

- I_TO_R S → D : 含正負號的整數16位元
 - DI_TO_R S → D : 含正負號的整數32位元
 - UI_TO_R S → D : 不含正負號的整數16位元
 - UDI_TO_R S → D : 不含正負號的整數32位元
- S : 儲存被轉換的BIN資料的裝置
D : 儲存轉換結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

- 1) 將在指定裝置 "S" 中的BIN整數資料轉換為實數類型 (REAL) 資料，並將轉換結果輸出到在 "D" 中所指定的裝置中。
- 2) 轉換之後的有效數字是6

<運算>

I_TO_R	,	DI_TO_R		UI_TO_R	,	UDI_TO_R
S (BIN)	轉換	D (實數)		S (BIN)	轉換	D (實數)
32767	⇒	3.2767E + 04 ENO = 1		65535	⇒	6.5535E + 04 ENO = 1
S (BIN)	轉換	D (實數)		S (BIN)	轉換	D (實數)
-32768	⇒	-3.2768E + 04 ENO = 1		0	⇒	-0.0E + 00 ENO = 1

[參照]

資料範圍如下：

- BIN 整數 (含正負號的 16 位元) : -32,768 至 32,767
- BIN 整數 (不含正負號的 16 位元) : 0 至 65,535
- BIN 整數 (含正負號的 32 位元) : -2,147,483,648 至 2,147,483,647
- BIN 整數 (不含正負號的 32 位元) : 0 至 4,294,967,295
- 實數 : $-2^{128} < N \leq -2^{-126}$ 、 0 、 $2^{-126} \leq N < 2^{128}$

(4) 實數至 BIN 整數 (R_TO_I)

符號

- $R_TO_I \quad S \rightarrow D$: 含正負號的整數16位元
 - $R_TO_DI \quad S \rightarrow D$: 含正負號的整數32位元
 - $R_TO_UI \quad S \rightarrow D$: 不含正負號的整數16位元
 - $R_TO_UDI \quad S \rightarrow D$: 不含正負號的整數32位元
- S: 儲存被轉換的實數資料的裝置
D: 儲存轉換結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	-	-	-	-	-	-	-	-	○	○	○	○	○	○	○	○	○	○	○
D	○	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	○	-

功能

- 1) 將在指定裝置 "S" 中的BIN整數資料轉換為實數類型 (REAL) 資料，並將轉換結果輸出到在 "D" 中所指定的裝置中。
- 2) 當從實數類型資料轉換出來的資料超過在 "D" 中指定的裝置資料範圍時，會將資料範圍的邊界數值輸出至 "D"。
- 3) 捨去小數點以下的小數。

<運算>

R_TO_I

- 當運算結果在 INT 類型範圍以內時



- 當運算結果超出 INT 類型範圍時



R_TO_UI

- 當運算結果在 UINT 類型範圍以內時



- 當運算結果超出 UINT 類型範圍時



R_TO_DI

- 當運算結果在 DINT 類型範圍以內時



- 當運算結果超出 DINT 類型範圍時



R_TO_UDI

- 當運算結果在 UDINT 類型範圍以內時



- 當運算結果超出 UDINT 類型範圍時



[參照]

資料範圍如下：

- BIN 整數（含正負號的 16 位元）：-32,768 至 32,767
- BIN 整數（不含正負號的 16 位元）：0 至 65,535
- BIN 整數（含正負號的 32 位元）：-2,147,483,648 至 2,147,483,647
- BIN 整數（不含正負號的 32 位元）：0 至 4,294,967,295
- 實數： $-2^{128} < N \leq -2^{-126}$ 、 0 、 $2^{-126} \leq N < 2^{128}$

(5) 實數至整數 (TRUNC_I)

符號

- TRUNC_I S → D : 含正負號的整數16位元
 - TRUNC_DI S → D : 含正負號的整數32位元
 - TRUNC_UI S → D : 不含正負號的整數16位元
 - TRUNC_UDI S → D : 不含正負號的整數32位元
- S : 儲存被轉換的實數資料的裝置
D : 儲存轉換結果的裝置

可用裝置

	字裝置 (W*)							雙字裝置 (D*)							常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T		TR	F
S	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	○	○	○	○	-	○	○	-	○	○	○	-	○	○	○	○	-

功能

- 1) 將在指定裝置 "S" 中的BIN整數資料轉換為實數類型 (REAL) 資料，並將轉換結果輸出到在 "D" 中所指定的裝置中。
- 2) 當從實數類型資料轉換出來的資料超過在 "D" 中指定的裝置資料範圍時，會將資料範圍的邊界數值輸出至 "D"。
- 3) 捨去小數點以下的小數。

<運算>

TRUNC_I

- 當運算結果在 INT 類型範圍以內時



- 當運算結果超出 INT 類型範圍時



TRUNC_UI

- 當運算結果在 UINT 類型範圍以內時



- 當運算結果超出 UINT 類型範圍時



TRUNC_DI

- 當運算結果在 DINT 類型範圍以內時



- 當運算結果超出 DINT 類型範圍時



[參照]

資料範圍如下：

- BIN 整數 (含正負號的 16 位元) :
-32,768 至 32,767
- BIN 整數 (含正負號的 32 位元) :
-2,147,483,648 至 2,147,483,647
- 實數 : $-2^{128} < N \leq -2^{-126}$, $0, 2^{-126} \leq N < 2^{128}$

TRUNC_UDI

- 當運算結果在 UDINT 類型範圍以內時



- 當運算結果超出 UDINT 類型範圍時

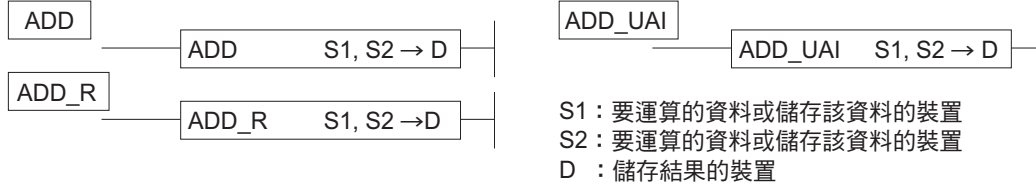


- BIN 整數 (不含正負號的 16 位元) :
0 至 65,535
- BIN 整數 (不含正負號的 32 位元) :
0 至 4,294,967,295

3-2-4 算數運算指令

(1) 加 (ADD)

符號



ADD : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
ADD_UAI : 整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
ADD_R : 實數類型 (單精準浮點類型) [REAL]

可用裝置

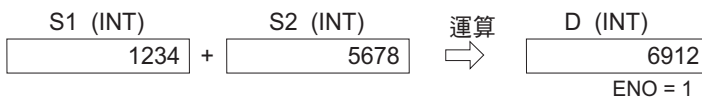
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

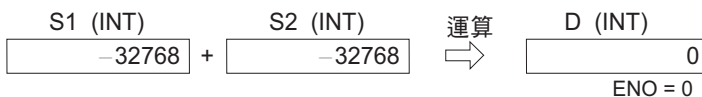
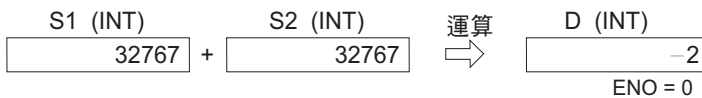
- 1) 將輸入資料 "S1" 與 "S2" 相加，並將結果輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 當運算結果超出資料類型範圍時，ENO會變為0。
- 4) 當資料類型為實數，且非常接近0，而無法以實數資料表示時，輸出會變為0，ENO變為1。

<運算>

- 當運算結果在資料類型範圍以內時



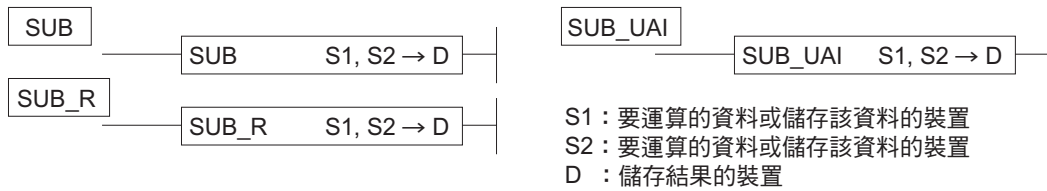
- 當運算結果超出資料類型範圍時



註：當非實數類型的資料超出資料類型範圍時，不會執行邊界數值處理。
請小心不要讓運算結果超出資料型範圍。實數類型資料則會執行邊界數值處理。

(2) 減 (SUB)

符號



SUB : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
 SUB_UAI : 整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
 SUB_R : 實數類型 (單精準浮點類型) [REAL]

可用裝置

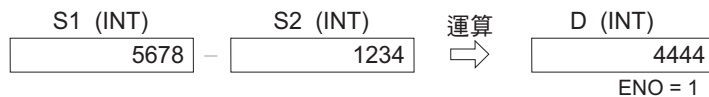
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

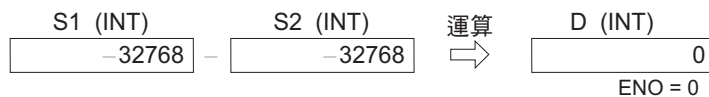
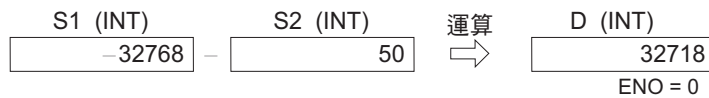
- 1) 從輸入資料 "S1" 中減去 "S2"，並將結果輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 當運算結果超出資料類型範圍時，ENO會變為0。
- 4) 當資料類型為實數，且非常接近0，而無法以實數資料表示時，輸出會變為0，ENO變為1。

<運算>

- 當運算結果在資料類型範圍以內時



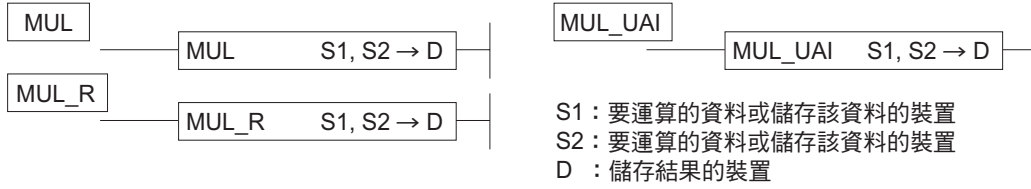
- 當運算結果超出資料類型範圍時



註：當非實數類型的資料超出資料類型範圍時，不會執行邊界數值處理。
 請小心不要讓運算結果超出資料型範圍。實數類型資料則會執行邊界數值處理。

(3) 乘 (MUL)

符號



- MUL : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
- MUL_UAI : 整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
- MUL_R : 實數類型 (單精準浮點類型) [REAL]

可用裝置

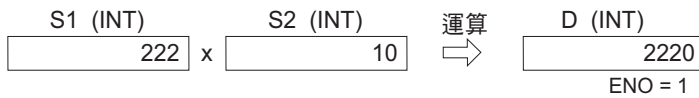
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

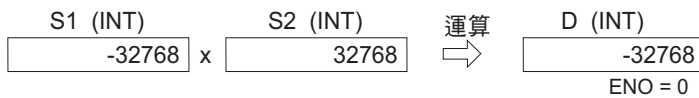
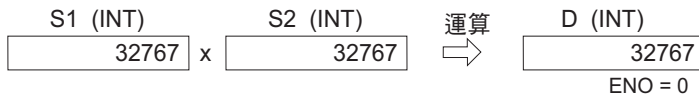
- 1) 將輸入資料 "S1" 與 "S2" 相乘，並將結果輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 當運算結果超出資料類型的邊界數值時，會輸出資料類型的邊界數值，且 ENO 會變為 0。
- 4) 當資料類型為實數，且非常接近 0，而無法以實數資料表示時，輸出會變為 0，ENO 變為 1。

<運算>

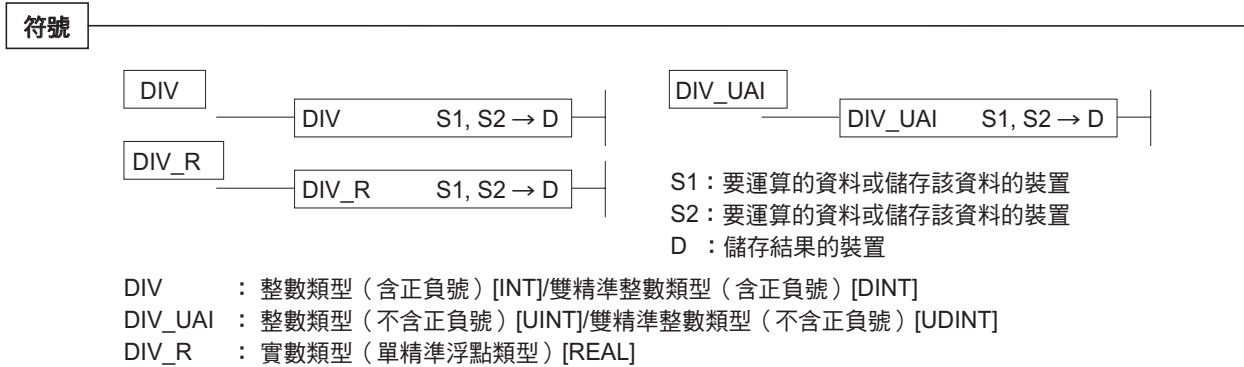
- 當運算結果在資料類型範圍以內時



- 當運算結果超出資料類型範圍時



(4) 除 (DIV)



可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—

- 功能**
- 1) 以 "S2" 除輸入資料 "S1"，並將結果輸出至 "D"。
 - 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
 - 3) 當運算結果超出資料類型的邊界數值時，會輸出資料類型的邊界數值，且 ENO 會變為 0。
 - 4) 當資料類型為實數，且非常接近 0，而無法以實數資料表示時，輸出會變為 0，ENO 變為 1。
 - 5) 當除數為 0 時，會輸出符合被除數正負號的最大數值，且 ENO 會變為 0。

<運算>

- 當運算結果在資料類型範圍以內時

$$\begin{array}{ccc}
 \begin{array}{c} S1 (INT) \\ \boxed{234} \end{array} & \div & \begin{array}{c} S2 (INT) \\ \boxed{4} \end{array} & \xrightarrow{\text{運算}} & \begin{array}{c} D (INT) \\ \boxed{2} \end{array} \\
 & & & & \text{ENO} = 1
 \end{array}$$

- 當除數為 0 時

$$\begin{array}{ccc}
 \begin{array}{c} S1 (INT) \\ \boxed{3940} \end{array} & \div & \begin{array}{c} S2 (INT) \\ \boxed{0} \end{array} & \xrightarrow{\text{運算}} & \begin{array}{c} D (INT) \\ \boxed{0} \end{array} \\
 & & & & \text{ENO} = 0
 \end{array}$$

- 當商數超出資料類型範圍時

$$\begin{array}{ccc}
 \begin{array}{c} S1 (INT) \\ \boxed{-32768} \end{array} & \div & \begin{array}{c} S2 (INT) \\ \boxed{-1} \end{array} & \xrightarrow{\text{運算}} & \begin{array}{c} D (INT) \\ \boxed{0} \end{array} \\
 & & & & \text{ENO} = 0
 \end{array}$$

(5) 除餘 (MOD)

符號



S1：要運算的資料或儲存該資料的裝置
 S2：要運算的資料或儲存該資料的裝置
 D：儲存結果的裝置

MOD：整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
 MOD_UAI：整數類型 (不含正負號) [UINT]/雙精準整數類型 (不含正負號) [UDINT]
 MOD_R：實數類型 (單精準浮點類型) [REAL]

可用裝置

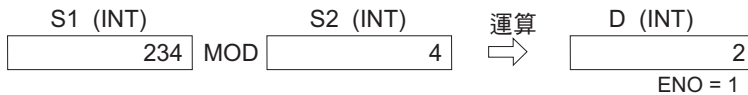
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

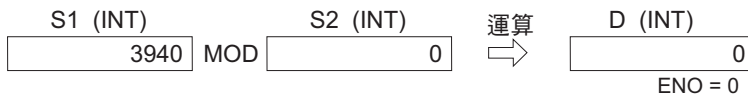
- 1) 以 "S2" 除輸入資料 "S1"，並將餘數輸出至 "D"。
- 2) 輸入裝置 "S1" 與 "S2"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。
- 3) 如果商數超出資料類型的邊界數值，輸出會變為0，且ENO變為0。
- 4) 當除數為0時，會輸出符合被除數正負號的最大數值，且ENO會變為0。
- 5) 運算的執行會使 (除數) x (商數) + (餘數) = (被除數)。

<運算>

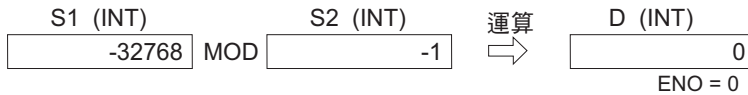
- 當運算結果在資料類型範圍以內時



- 當除數為0時



- 當商數超出資料類型範圍時



(6) 指數 (EXPT)

符號



S1：要運算的資料或儲存該資料的裝置
 S2：要運算的資料或儲存該資料的裝置
 D：儲存結果的裝置

可用裝置

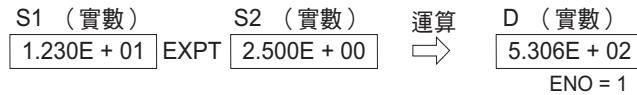
	字裝置 (W*)							雙字裝置 (D*)							常數				
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T		TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	-	○	○	-	○	○	○	-	-	-	○	○	-	

功能

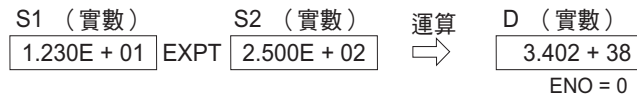
- 1) 將輸入資料 "S1" (基數) 由 "S2" (指數) 取冪，並將結果輸出至 "D"。
- 2) 當運算結果超出實數類型的邊界數值，輸出會變為0，且ENO變為0。
- 3) 基數 ≥ 0。當基數 < 0 時，輸出會變為0，且ENO變為0。
- 4) 輸出中的有效數字是4。
- 5) 當輸出數值超出實數類型的邊界數值時，ENO會變為0。當輸出數值非常接近0，使其無法以實數資料來表示時，輸出會變為0，且ENO變為1。
- 6) 當基數 = 0且指數 = 0 (0fl) 時，輸出會變為1，且ENO變為1。

<運算>

- 當運算結果在資料類型範圍以內時

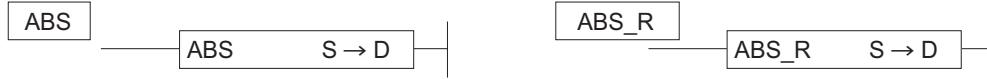


- 當運算結果超出資料類型範圍時



(7) 絕對值 (ABS)

符號



ABS : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
 ABS_R : 實數類型 (單精準浮點類型) [REAL]

S : 要運算的資料或儲存該資料的裝置
 D : 儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	○	○	○	○	-	○	○	○	○	○	○	○	-	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-

功能

- 1) 取得輸入資料 "S" 的絕對值並將結果輸出至 "D"。
- 2) 當輸入數值為最大負值時，輸出該輸出資料類型的最大正值。
- 3) 輸入裝置 "S"，且輸出裝置 "D" 必須有相同的資料類型及相同的位元寬度。

<運算>



- 當輸入數值為最大負值時 (-32768)



(8) 平方根 (SQRT)

符號



可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	○	-	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

- 1) 取得輸入資料 "S" 的平方根 (實數類型, (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 當輸入數值為負時, 輸出0且ENO變為0。
- 3) 輸出的有效位數是5。

<運算>

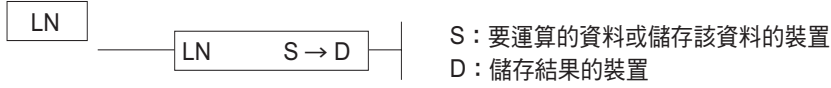


- 當輸入數值為負時



(9) 自然對數 (LN)

符號



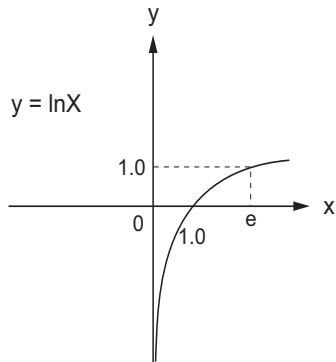
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

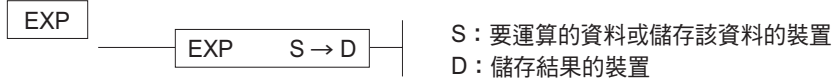
- 1) 取得輸入資料"S"的平方根 (實數類型, (單精準浮點類型) [REAL]) 並將結果輸出至"D"。
- 2) 當輸入數值為負時, 輸出0且ENO變為0。
- 3) 當輸入數值為0時, 輸出會變為最大負值, 且ENO變為0。
- 4) 輸出的有效位數是4。

<運算>



(11) 指數 (EXP)

符號



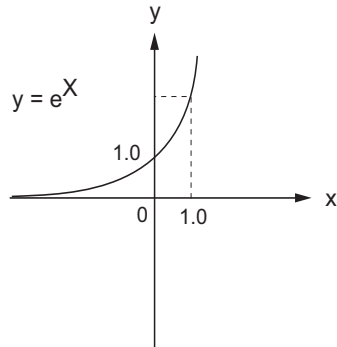
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

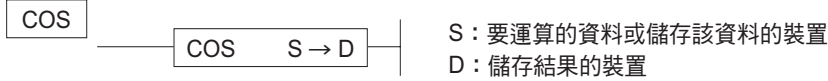
- 1) 以2.718281為基數 (e) 對"S"執行指數運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至"D"。
- 2) 當運算結果超出實數類型的邊界數值時, 便輸出邊界數值。
- 3) 輸出的有效位數:
 - 當輸入/輸出在-64至64的範圍內時... 4
 - 以上情況以外... 錯誤將會增加。
- 4) 當輸出數值超出實數類型的邊界數值時, ENO會變為0。當輸出數值非常接近0, 使其無法以實數資料來表示時, 輸出會變為0, 且ENO變為1。

<運算>



(13) 餘弦 (COS)

符號



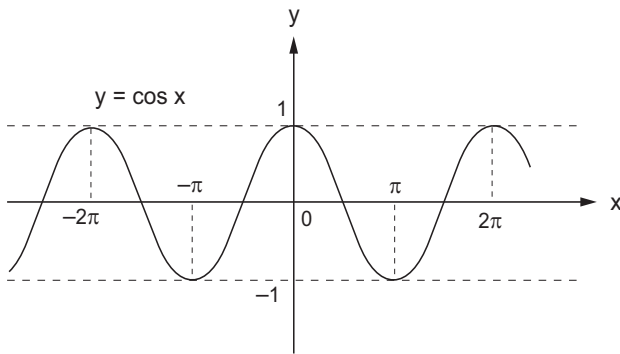
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

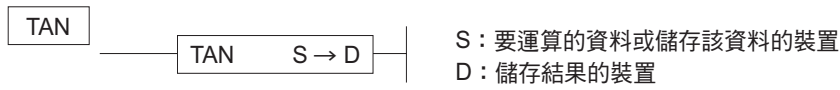
- 1) 對 "S" 執行餘弦運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸入的單位為弧度。
- 3) 當輸入在 -2π 至 2π 的範圍以內時，輸出的有效位數為5 (最多四位小數)。當輸入的絕對值為 2π 或以上時，將會執行運算，但錯誤將會增加。
- 4) 當 (輸入) $< -2^{31}$ 或 (輸入數值) $> 2^{31} - 1$ 時，輸出數值會變為0，且 ENO 變為0。

<運算>



(14) 正切 (TAN)

符號



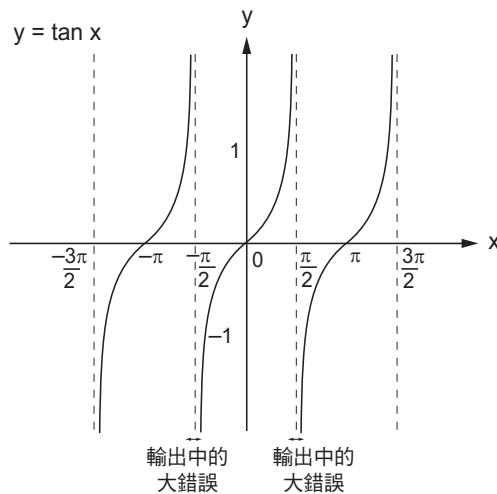
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

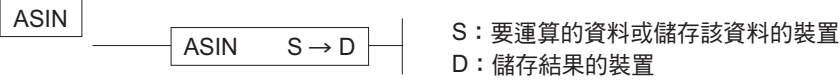
- 1) 對 "S" 執行正切運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸入的單位為弧度。
- 3) 當輸入在 -2π 至 2π 的範圍以內時, 輸出的有效位數為4, 但輸出中的錯誤將會增加, 接近 $\pi/2$ 的整數倍。當輸入的絕對時為 2π 或以上時, 將會執行運算, 但錯誤將會增加。
- 4) 當 (輸入) $< -2^{31}$ 或 (輸入數值) $> 2^{31} - 1$ 時, 輸出數值會變為0, 且 ENO 變為0。

<運算>



(15) 反正弦 (ASIN)

符號



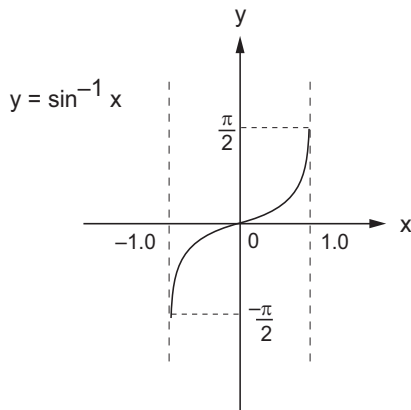
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

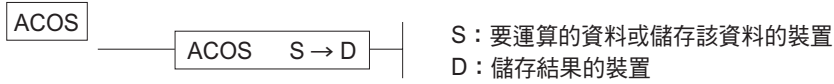
- 1) 對 "S" 執行反正弦運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸出的單位為弧度。
- 3) 輸入的範圍為 -1.0 至 +1.0，輸出範圍為 $-\pi/2$ 至 $\pi/2$ 。當輸入超出此範圍時，輸出會變為 0，且 ENO 變為 0。
- 4) 輸出的有效位數：
 - 當 |輸入數值| = 1.0 或 0.998999 或更少時... 4 位數
 - 當 |輸入數值| = 0.999 至 0.999999 時... 錯誤將會增加。

<運算>



(16) 反餘弦 (ACOS)

符號



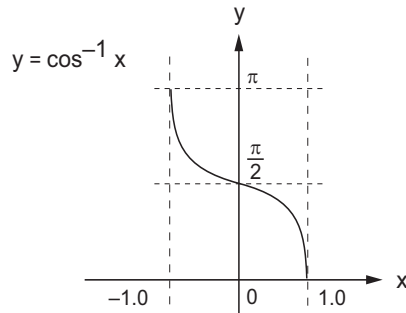
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

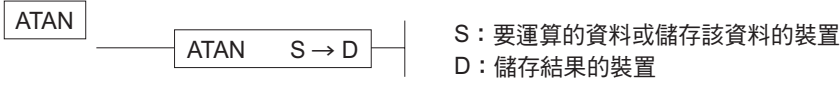
- 1) 對 "S" 執行反餘弦運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸出的單位為弧度。
- 3) 輸入的範圍為-1.0至+1.0，輸出範圍為 π 至0。當輸入超出此範圍時，輸出會變為0，且ENO變為0。
- 4) 輸出的有效位數：
 當 |輸入數值| = 1.0或0.998999或更少時... 4位數
 當 |輸入數值| = 0.999至0.999999時... 錯誤將會增加

<運算>



(17) 反正切 (ATAN)

符號



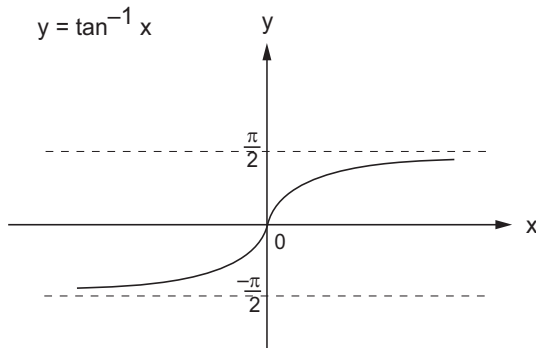
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	-	-	-	-	-	-	-	-	○	○	○	○	-	○	○	○	○	○
D	-	-	-	-	-	-	-	-	-	○	○	○	-	-	-	○	○	-

功能

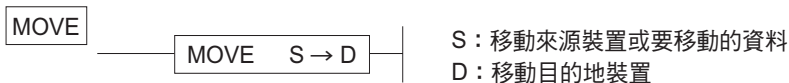
- 1) 對 "S" 執行反正切運算 (實數類型 (單精準浮點類型) [REAL]) 並將結果輸出至 "D"。
- 2) 輸出的單位為弧度。
- 3) 輸出範圍從最大負值至最大正值，輸出範圍為 $-\pi/2$ 至 $\pi/2$ 。
- 4) 輸出的有效位數是5。

<運算>



(18) 移動 (MOVE)

符號



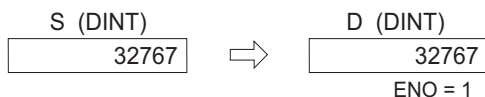
可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

MOVE 將輸入資料 "S" 直接移動到 "D"。

<運算>



註：輸入裝置 "S" 與輸出裝置 "D" 中的資料必須有相同的位元長度。

(19) 負 (NEG)

符號



NEG : 整數類型 (含正負號) [INT]/雙精準整數類型 (含正負號) [DINT]
 NEG_R : 實數類型 (單精準浮點類型) [REAL]

S : 輸入資料或儲存輸入資料的裝置
 D : 儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

- 1) 反轉輸入資料 "S" 的正負號，並將結果輸出至"D"。
- 2) 可用資料類型為整數類型、雙整數類型及實數類型。
- 3) 當資料類型為整數類型或雙整數類型，且運算結果超出資料類型範圍時，ENO會變為0。

<運算>



● 針對雙整數類型或整數類型的負邊界數值



(20) 分塊移動 (BMOV)

符號



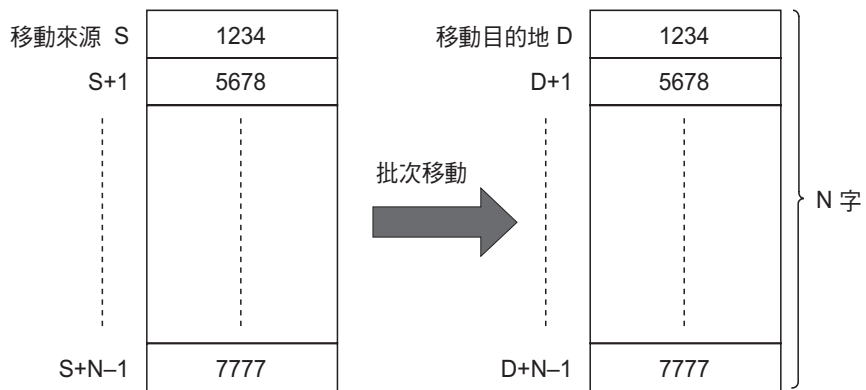
S : 要從其中移出的資料的標頭裝置
 D : 要移至的資料的標頭裝置
 N : 要移動的字數或儲存這些字的裝置 (範圍1至32767)

可用裝置

	位元裝置										字裝置 (W*)						雙字裝置 (D*)						常數					
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L		SM	T	TR	F	V
S1	○	○	○	○	○	-	-	-	○	○	○	○	○	○	○	-	○	○	○	○	○	○	○	-	-	○	○	-
S1	-	○	○	○	-	-	-	-	○	○	-	○	○	○	-	-	○	○	-	○	○	○	-	-	-	○	○	-
D	-	-	-	-	-	-	-	-	-	-	○	-	○	○	-	○	○	○	-	-	○	○	-	-	-	○	○	○

功能

BMOV 1) 一次從在"S"中指定的裝置中將"N"字的資料 (要移動的字數) 移至在"D"中指定的裝置上開始的N字區域。



2)"S" (移動來源資料) 與 "D" (移動目的地資料) 中的資料必須有相同的位元寬度。

註：指定要移動的字數，這樣一來用於其他目的的裝置便不會被重複寫入。當間接指定要移動的數時，需要特別小心。

(21) 填滿移動 (FMOV)

符號

FMOV S, D, N

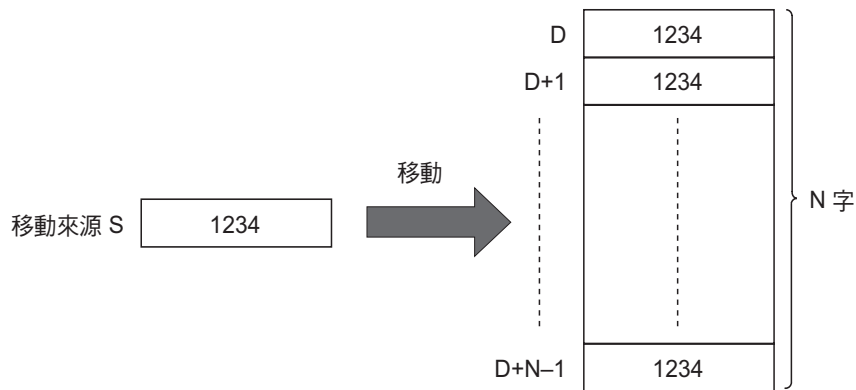
S：要移動的資料或儲存資料的裝置
 D：移動目的地的標頭裝置
 N：要移動的字數或儲存這些字的裝置（範圍1至32767）

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	-	○	○	○	-	-	○	○	-	-	-	-	-	-	-	-	-	-	-
D	○	○	○	○	○	○	○	○	-	-	○	○	-	-	-	○	○	○	○

功能

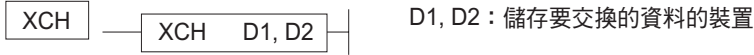
FMOV 將在"D"中指定的裝置內容移動到在"D"中指定的裝置上開始的N字區域。



註：指定要移動的字數，這樣一來用於其他目的的裝置便不會被重複寫入。
 當間接指定要移動

(22) 交換 (XCH)

符號

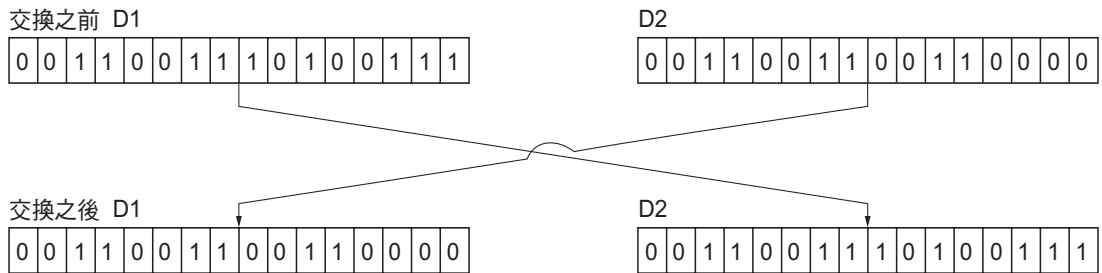


可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	○	○	○	○	—	○	○	○	○	○	○	○	—	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—

功能

XCH 交換"D1"中的資料與"D2"中的資料。



"D1"與"D2"中的資料必須有相同的位元寬度。

(23) 間接放置 (BDMPX)

符號

BDMPX — BDMPX S1, D, S2, N

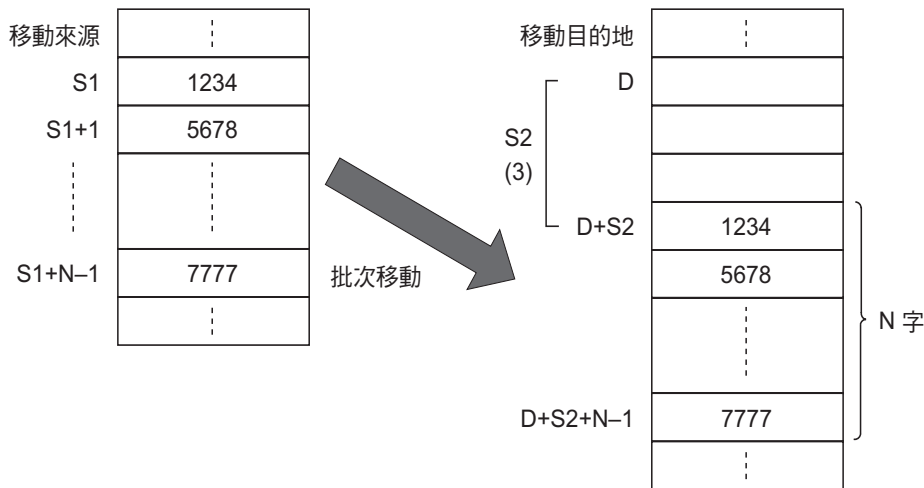
S1：移動來源裝置
 D：移動目的地裝置（參照位址）
 S2：資料指標
 N：要移動的字數（範圍：1至32767）

可用裝置

	位元裝置										字裝置 (W*)						雙字裝置 (D*)						常數					
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L		SM	T	TR	F	V
S1	○	○	○	○	○	—	—	—	○	○	○	○	○	○	○	—	○	○	○	○	○	○	○	—	—	○	○	○
D	—	○	○	○	—	—	—	—	○	○	—	○	○	○	—	—	○	○	—	○	○	○	—	—	—	○	○	—
S2	—	—	—	—	—	—	—	—	—	○	○	—	○	○	—	○	○	○	○	—	○	○	—	—	—	○	○	○
N	—	—	—	—	—	—	—	—	—	○	○	—	○	○	—	○	○	○	—	—	○	○	—	—	—	○	○	○

功能

BDMPX 1) 從在"S1"中指定的裝置將"N"字（要移動的字數）的資料移動到從在"D"中指定的裝置（參照位址）"S2"中指定的裝置中的資料（資料指標）所取代的裝置裡。



2) "S1"（移動來源資料）與"D"（移動目的地資料）中的資料必須有相同的位元寬度。

註：指定要移動的字數，這樣一來用於其他目的的裝置便不會被重複寫入。
 當間接指定要移動的字數時，需要特別小心。

(24) 間接取出 (BMPX)

符號



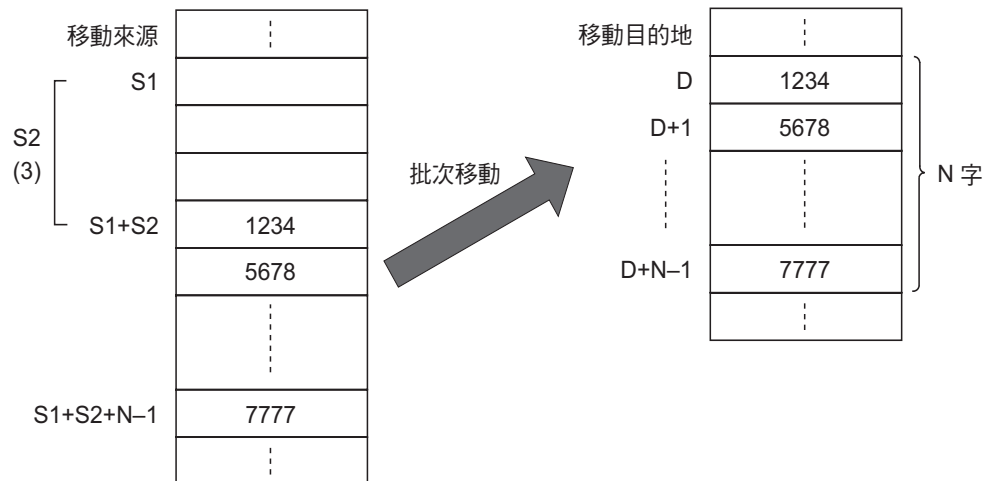
S1：移動來源裝置（參照位址）
 S2：資料指標
 D：移動目的地裝置
 N：要移動的字數（範圍：1至32767）

可用裝置

	位元裝置										字裝置 (W*)							雙字裝置 (D*)							常數			
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T		TR	F	V
S1	○	○	○	○	○	-	-	-	○	○	○	○	○	○	○	-	○	○	○	○	○	○	○	-	-	○	○	-
S2	-	-	-	-	-	-	-	-	-	-	○	-	○	○	-	○	○	○	-	-	○	○	-	-	-	○	○	○
D	-	○	○	○	-	-	-	-	-	-	-	○	○	○	-	-	○	○	○	○	○	○	-	-	-	○	○	-
N	-	-	-	-	-	-	-	-	-	-	○	-	○	○	-	○	○	○	-	-	○	○	-	-	-	○	○	○

功能

BDMPX 1) 從在"S1"中指定的裝置將"N"字（要移動的字數）的資料移動到從"D"中指定的裝置（參照位址）中的"S2"中指定的裝置中的資料（資料指標）所取代的裝置裡。



2) "S"（移動來源資料）與 "D"（移動目的地資料）中的資料必須有相同的位元寬度。

註：指定要移動的字數，這樣一來用於用於其他目的的裝置便不會被重複寫入。
 當間接指定要移動

3-2-5 位元字串運算指令

(1) 邏輯 AND (AND_AW)

符號



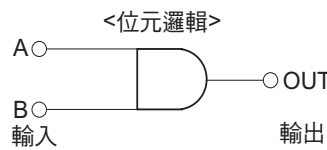
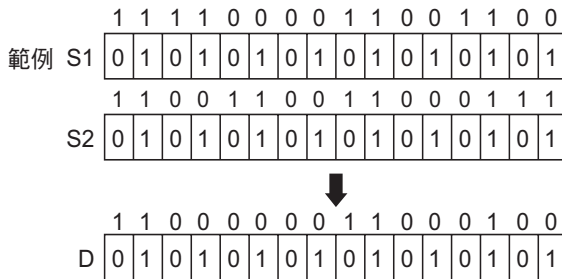
S1：儲存要進行邏輯AND的資料的裝置
 S2：要進行邏輯AND的資料或儲存資料的裝置
 D：儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-	

功能

1) 對在"S1"中指定的裝置的資料及在"S2"中指定的裝置的資料（或常數）執行逐位元的邏輯AND，並將結果輸出到在"D"中指定的裝置。



A	B	OUT
0	0	0
1	0	0
0	1	0
1	1	1

2) "S1"與"S2"中的資料必須有相同的位元寬度。

(2) 邏輯 OR (OR_AW)

符號

OR_AW

OR_AW

S1, S2 → D

S1：儲存要進行邏輯OR的資料的裝置
 S2：要進行邏輯OR的資料或儲存資料的裝置
 D：儲存結果的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—	

功能

- 1) 對在"S1"中指定的裝置的資料及在"S2"中指定的裝置的資料（或常數）執行逐位元的邏輯OR，並將結果輸出到在"D"中指定的裝置。

範例 S1

0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

S2

0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

↓

D

0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

<位元邏輯>

A ○
B ○
輸入

○ OUT
輸出

A	B	OUT
0	0	0
1	0	1
0	1	1
1	1	1

- 2) "S1"與"S2"中的資料必須有相同的位元寬度。

(3) 邏輯互斥 OR (XOR)

符號

位元類型：

XOR_B

XOR_B S1, S2 → D

字/雙字類型：

XOR_AW

XOR_AW S1, S2 → D

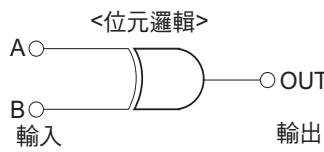
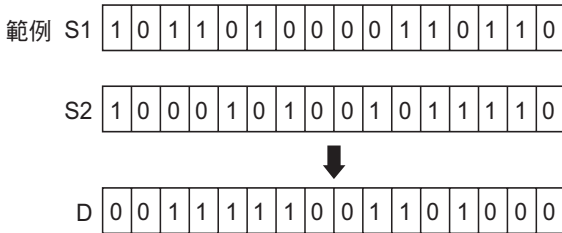
S1：儲存要進行邏輯互斥OR的資料的裝置
 S2：要進行邏輯互斥OR的資料或儲存資料的裝置
 D：儲存結果的裝置

可用裝置

	位元裝置										字裝置 (W*)						雙字裝置 (D*)						常數						
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L		SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	○	-	-	-	○	○	-	○	○	-	-	○	○	-	○	○	○	-	○	○	○	○	-	-	

功能

1) 對在"S1"中指定的裝置的資料及在"S2"中指定的裝置的資料（或常數）執行逐位元的邏輯互斥OR，並將結果輸出到在"D"中指定的裝置。



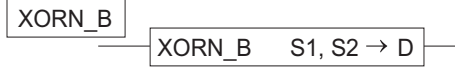
A	B	OUT
0	0	0
1	0	1
0	1	1
1	1	0

2) "S1 與"S2"中的資料必須有相同的位元寬度。

(4) 邏輯互斥 NOR (XORN)

符號

位元類型：



字/雙字類型：



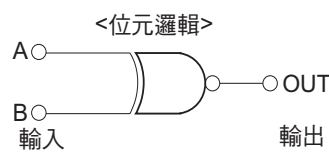
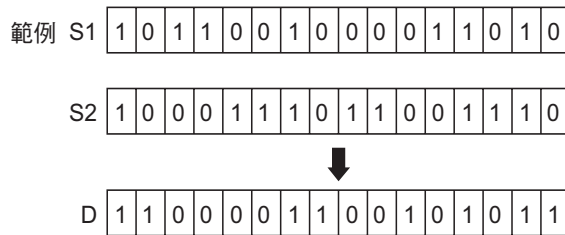
S1：儲存要進行邏輯互斥OR的資料的裝置
 S2：要進行邏輯互斥OR的資料或儲存資料的裝置
 D：儲存結果的裝置

可用裝置

	位元裝置										字裝置 (W*)							雙字裝置 (D*)							常數			
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T		TR	F	V
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	○	-	-	-	○	○	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-

功能

1) 對在"S1"中指定的裝置的資料及在"S2"中指定的裝置的資料（或常數）執行逐位元的邏輯互斥NOR，並將結果輸出到在"D"中指定的裝置。

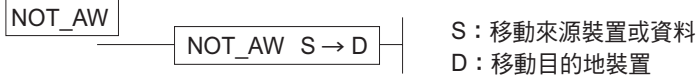


A	B	OUT
0	0	1
1	0	0
0	1	0
1	1	1

2) "S1"與"S2"中的資料必須有相同的位元寬度。

(5) 邏輯 NOT (NOT)

符號



可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數	
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	-	○	○	○	-	○	○	○	-	○	○	○	-	○	○	○	○	-

功能

1) 反轉在"S"中指定的裝置中的資料的每一個位元，並將結果輸出至在"D"中指定的裝置中。

移動來源：S

0 0 1 1 0 1 0 0 1 1 1 0 0 0 1 1

↓ 反轉及移動

移動目的地：D

1 1 0 0 1 0 1 1 0 0 0 1 1 1 0 0

2) "S"與"D"中的資料必須有相同的位元寬度。

(6) 旋轉 (ROR、ROL)

符號

向右旋轉：

向左旋轉：

S：要旋轉的裝置
N：要旋轉的位元數 (N = 0至31)
D：儲存旋轉資料的裝置

可用裝置

	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	-	-	-	-	-	-	-	-	-	○	
D	-	○	○	○	-	-	○	○	-	○	○	○	-	-	-	○	○	-	

功能

ROR 1) 以在"N (要旋轉的位元數)"中所指定的數值向右旋轉輸入資料"S (要旋轉的裝置)"，並將結果輸出到"D"。

例如：向右旋轉3位元 (N = 3)

ROL 1) 以在"N (要旋轉的位元數)"中所指定的數值向左旋轉輸入資料"S (要旋轉的裝置)"，並將結果輸出到"D"。

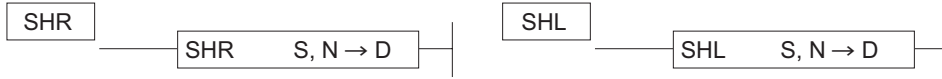
例如：向左旋轉3位元 (N = 3)

<關於要旋轉的位元數 "N" 的說明>
 當位元寬度為16位元時，低順序的四個位元對"N"是有效的。
 例如，當N = 16時，旋轉大小為0位元；當N = 17時，旋轉大小為1位元。當位元寬度為32位元時，低順序的五個位元對"N"是有效的。
 例如，當N = 32時，旋轉大小為0位元；當N = 33時，旋轉大小為1位元。

* "S"與"D"中的資料必須有相同的位元寬度。

(7) 移位 (SHR、SHL)

符號



S：要移位的裝置
N：要移位的位元數 (N = 0至31)
D：儲存移位資料的裝置

可用裝置

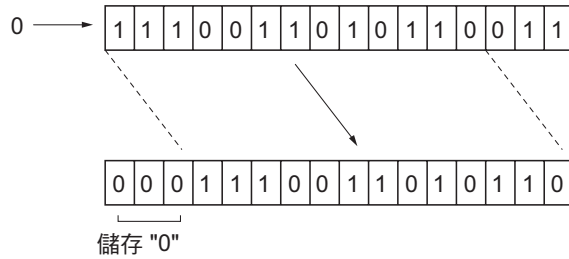
	字裝置 (W*)								雙字裝置 (D*)								常數		
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F		V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S1	○	○	○	○	○	○	○	○	-	-	-	-	-	-	-	-	-	○	
D	-	○	○	○	-	-	○	○	-	○	○	○	-	-	-	○	○	-	

功能

SHR

1) 將輸入資料"S"往右移位"N"位元，在由移位所清空的位元中放置 0，然後將結果輸出到在 "D"中指定的裝置中。

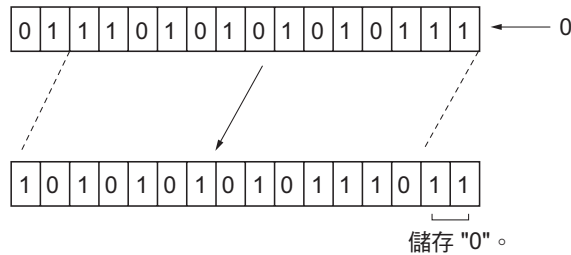
例如：向右移位3位元 (N = 3)



SHL

1) 將輸入資料"S"往左移位"N"位元，在由移位所清空的位元中放置 0，然後將結果輸出到在 "D"中指定的裝置中。

例如：向左旋轉3位元 (N = 3)



<關於要旋轉的位元數 "N" 的說明>

當位元寬度為16位元時，低順序的四個位元對 "N"是有效的。

例如，當N = 16時，旋轉大小為0位元；當N = 17時，旋轉大小為1位元。當位元寬度為32位元時，低順序的五個位元對"N"是有效的。

例如，當N = 32時，旋轉大小為0位元；當N = 33時，旋轉大小為1位元。

* "S"與"D"中的資料必須有相同的位元寬度。

第 4 節 系統定義

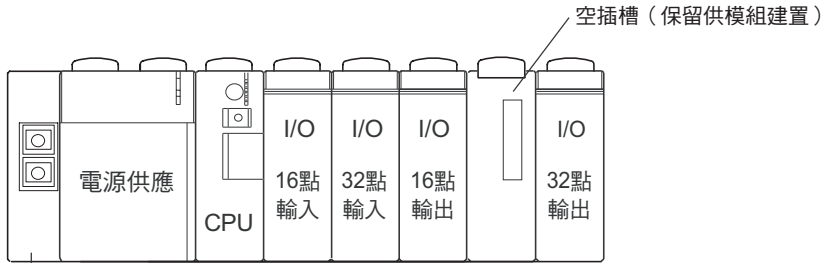
	頁碼
4-1 系統定義摘要	4-1
4-2 在系統定義中設定	4-2
4-3-1 系統作業定義	4-5
(1) SX 匯流排 Takt 時間	4-5
(2) 組態檢查等待時間	4-5
(3) 初始方法	4-5
4-3 系統屬性	4-5
4-3-2 系統備援定義	4-6
(1) 1 對 1 備援系統	4-6
(2) N 對 1 備援系統	4-6
4-3-3 系統軟故障啟動	4-7
(1) 軟故障停用	4-7
(2) 擁有 SX 匯流排站台數的模組的部分軟故障啟動	4-7
4-4 CPU 參數	4-9
4-4-1 CPU 作業定義	4-9
(1) 監視器計時器	4-9
(2) 電源開啟時間作業	4-9
(3) 低電量作業	4-9
4-4-2 CPU 記憶體大小定義	4-11
4-4-3 I/O 群組設定	4-13
4-4-4 軟故障執行	4-19
4-5 輸入 / 輸出參數	4-22
4-5-1 輸入過濾時間	4-22
4-5-2 輸出保留定義	4-23
4-5-3 系統輸出定義	4-24

名稱	定義	參照	識別計時	
			高效能CPU	標準CPU
系統組態定義	SPH系統中的模組登錄及SX匯流排站台數	4-2	SPH系統重設，下載	SPH系統重設
系統作業定義	SX匯流排Takt期間，組態檢查等待時間，初始方法	4-3-1	SPH系統重設	SPH系統重設
雙工模式定義	雙工模式啟用/停用，1對1雙工模式，N對1雙工模式	4-3-2, 5	SPH系統重設	未支援
系統軟故障啟用定義	軟故障啟動啟動SX匯流排站台數	4-3-3	SPH系統重設	未支援
CPU作業定義	監視器計時器，低電量作業，電源開啟時作業	4-4-1	SPH系統重設	SPH系統重設
CPU記憶體大小定義	資料記憶體大小，AT指定範圍，保留記憶體	4-4-2	SPH系統重設	SPH系統重設，下載
I/O群組	I/O群組登錄	4-4-3	SPH系統重設	SPH系統重設
軟故障設定	針對一般模組以外的其他模組（I/O模組等）進行軟故障啟用/停用	4-4-4	SPH系統重設	SPH系統重設
輸入過濾時間	數位輸入模組（DC輸入裝置）的輸入過濾時間	4-5-1	下載	SPH系統重設
輸出保留定義	針對輸出模組保留/重設登錄	4-5-2	下載	SPH系統重設
系統輸出定義	系統輸出模組	4-5-3	下載	SPH系統重設
T連結主模組參數	單獨輸出保留站台定義	4-6	SPH系統重設	SPH系統重設

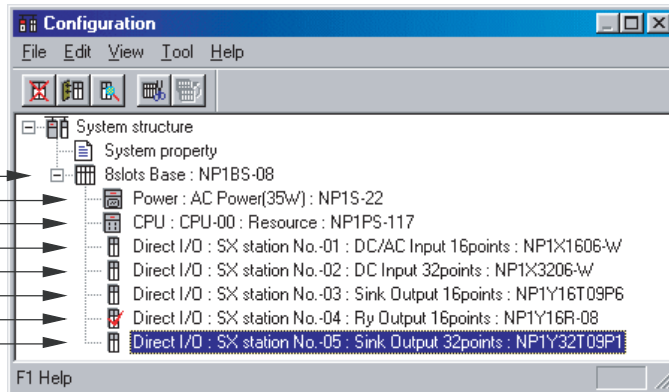
註： 1) 重設包含所有重設（SPH系統重設）、單獨重設（資源重設）及電源開啟。

對於MICREX-SX，有必要將所有要使用模組登錄在一個SPH系統底下。
 底下將說明使用下述範本系統組態的模組登錄程序。

< 範本系統組態 >

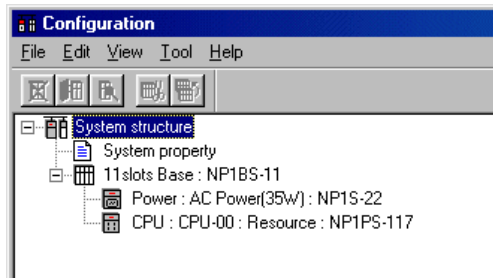


< 系統組態登錄畫面 >



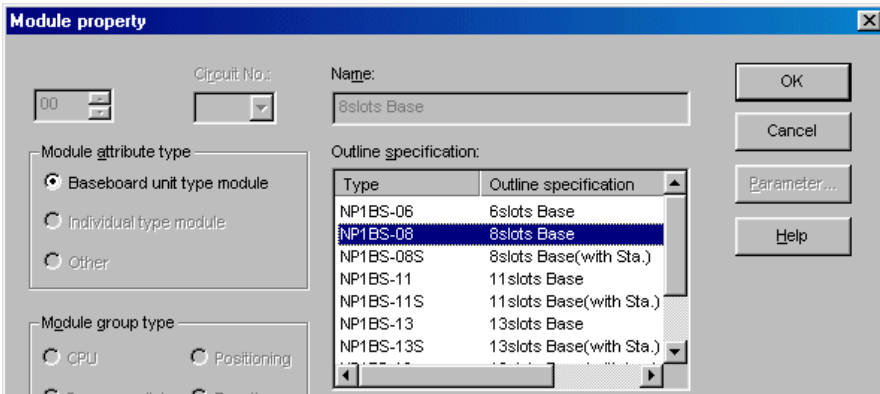
< 登錄方法 >

執行專案樹狀結構中的「PLC功能」功能表 → [系統定義]。此時您會看到系統定義螢幕。

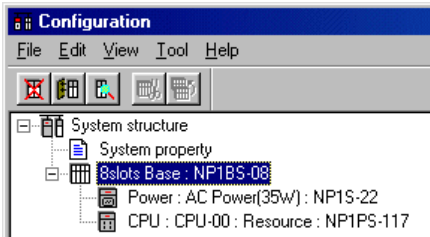


將基板變更為8插槽以便實際使用。選擇基板，然後按下 [內容] 按鈕。此時您會看到「模組內容」對話框。





從 [概略規格] 清單框中選擇「NP1BS-08 8插槽基板」然後按下 [確定] 按鈕。基板會變成8插槽基板。



登錄模組的登錄順序從基板左邊開始。由於按照預設值會登錄電源供應器與 CPU 模組，因此請登錄 16 點輸入模組。要在 CPU 模組底下登錄（新增）模組，請選擇 CPU 模組，然後按下 [插入] 按鈕。此時您會看到「模組插入」對話框。



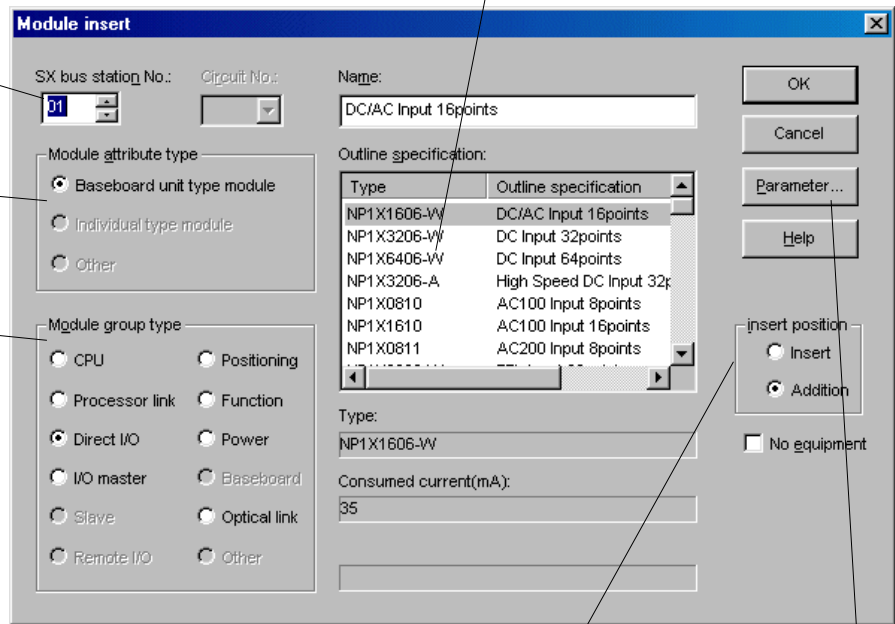
在選取的模組之下插入。



選擇 SX 匯流排站數。至於 CPU 與處理器連結模組，請選擇 CPU 數。

選擇模組屬性。

選擇模組類型。



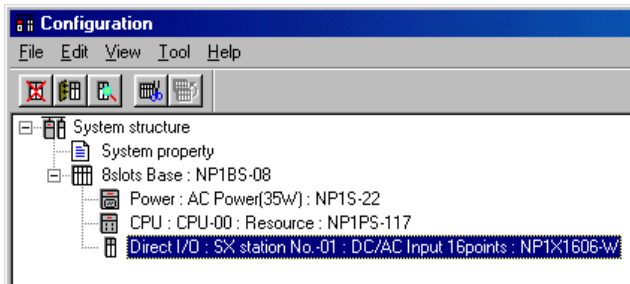
選擇模組。

選擇模組的插入位置。

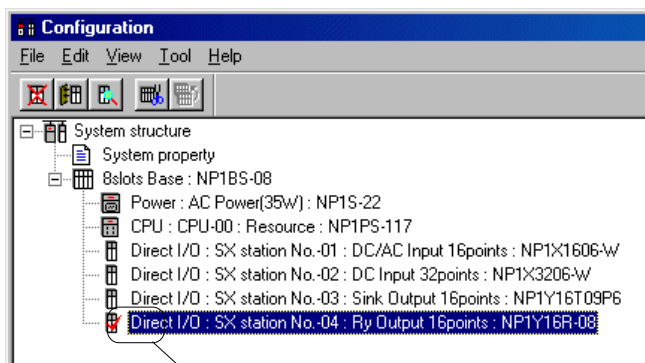
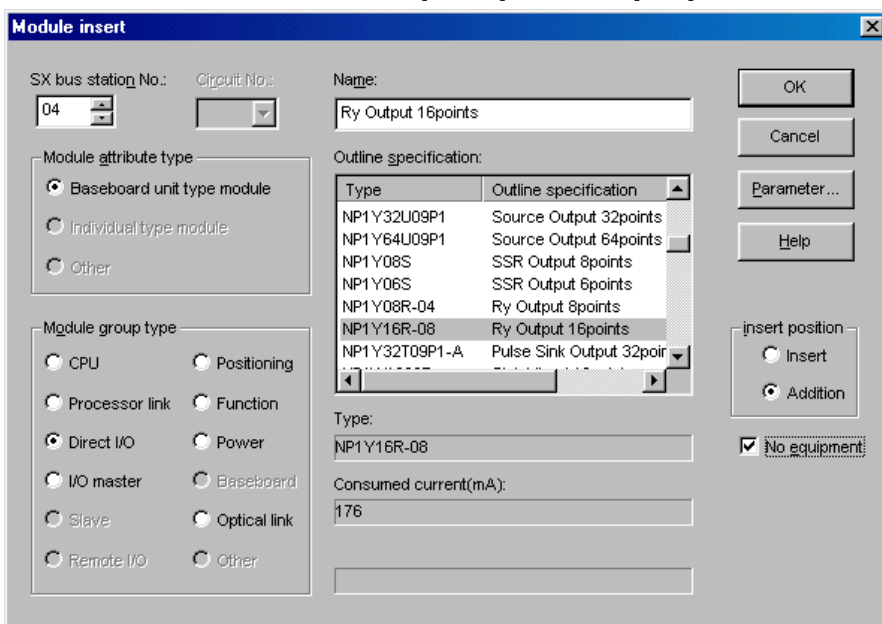
設定模組參數。

根據實際組態選擇模組 (NP1X1606-W DC/AC 輸入 16 點)。針對 SX 匯流排站台數，可指定從 1 到 238 之間任何想要的數字，但是 CPU 模組會從右到左指定為數字 1、2、3 等。

當您選擇模組並按下 [確定] 按鈕時，模組會被登錄。



請以相同方式登錄模組。如果您未來要加載模組，並如系統組態所示保持插槽不使用的狀態，請以與其他模組相同的方式指定 SX 匯流排站台數及要加載的模組，並核取 [無設備]，然後按下 [確定] 按鈕。



模組會被登錄，其圖示會標記上「未加載」。

4-3-1 系統作業定義

(1) SX 匯流排 Takt 時間

SX 匯流排 Takt 時間會被定義為一個期間，在這段期間內，會在連接到 SX 匯流排的模組（如輸入/輸出模組）之間交換資料。Takt 時間的設定範圍介於 0.5 毫秒至 10.0 毫秒之間，每段 0.5 毫秒（當 CPU 韌體版本為 V50 或更新版本時）。至於較舊的 CPU 韌體版本，Takt 時間可被設定為 0.5 毫秒、1.0 毫秒、2.0 毫秒.....（從 1.0 毫秒開始，每段 1 毫秒）。預設值為 1.0 毫秒。

- 註：1) 0.5 毫秒的 Takt 期間可在例如擁有功能強大的 CPU、256 或較少直接連接的 I/O 及無遠端 I/O 及通訊模組等情況下執行。
 2) 至於 CPU 韌體版本為 V34 或更早或 V3A 至 V3Z 的產品，Takt 時間可設定為最大 20 毫秒，每段 1 毫秒。
 3) V2.2 或更新的版本支援以每段 0.5 毫秒來設定 Takt 時間的功能。

(2) 組態檢查等待時間

電源開啟時，CPU 模組會初始化它本身。當初始化完成後將開始針對一個組態中的模組組態進行檢查。請針對「組態檢查等待時間」設定組態檢查的時間。

(3) 初始方法

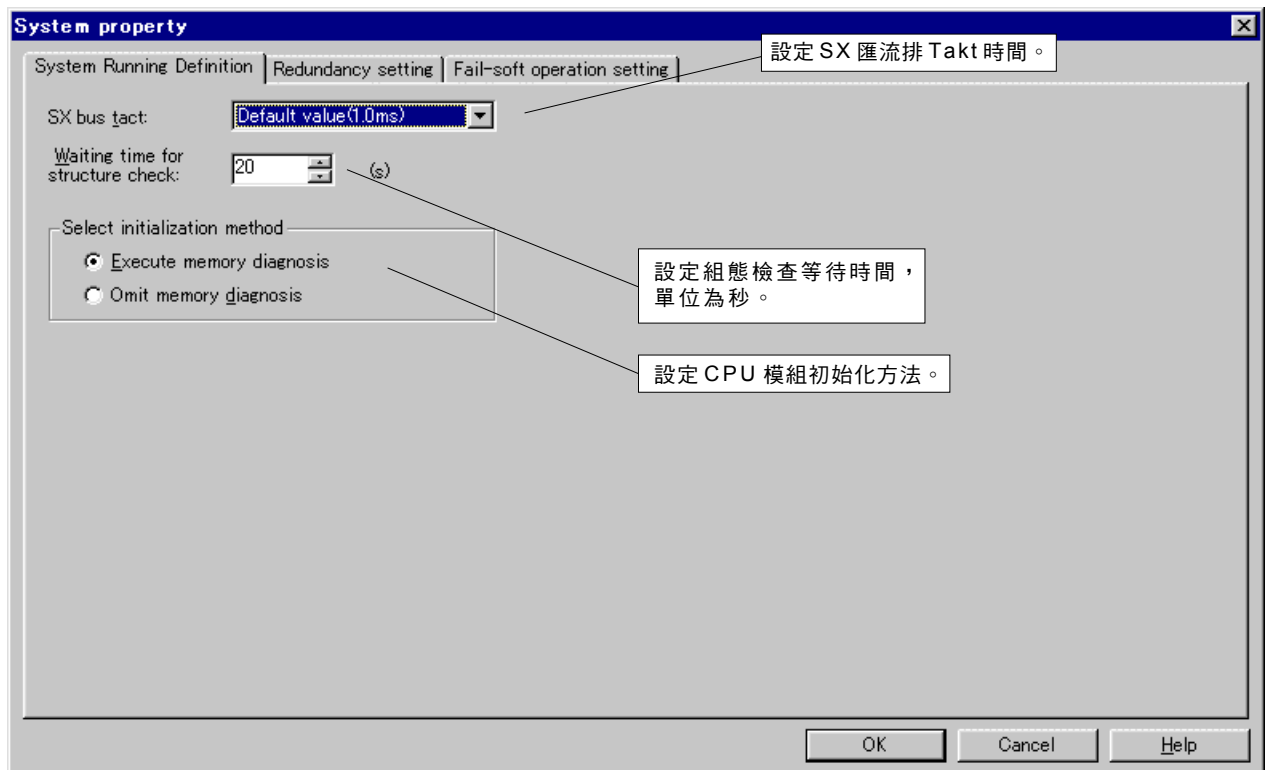
當系統電源開啟時，CPU 模組會初始化它本身。使用者可以指定 CPU 是否要在 CPU 的內部記憶體上執行診斷。當執行記憶體診斷時，CPU 初始化時間約為 4.5 秒，而停用記憶體診斷時，約為 2.5 秒。

- 註：1) 記憶體診斷會執行裝置讀取/寫入檢查。
 2) 標準 CPU 不支援此項目。（不會執行裝置讀取/寫入檢查。）

< 設定程序 >

從系統定義畫面中選擇系統屬性並按下 [內容] 按鈕，此時您會看到 [系統屬性] 對話框的 [系統執行定義] 畫面。

- 1) 您會看到「系統屬性」對話框。



- 2) 當填入必要的欄位後，請按下 [確定] 按鈕。

4-3-2 系統備援定義

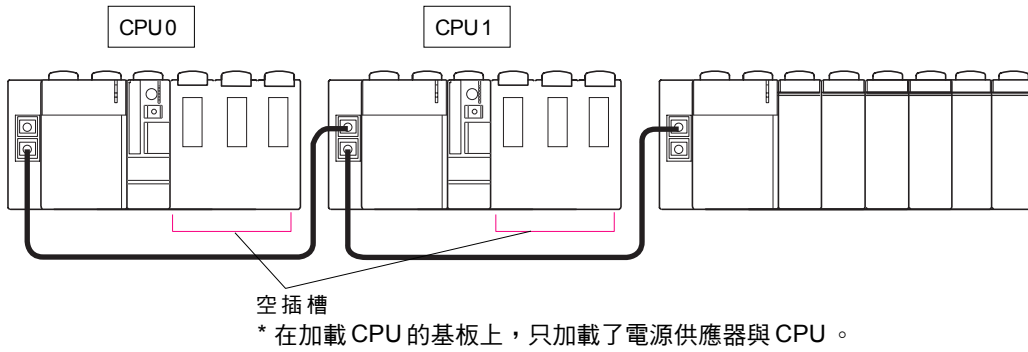
如果在「主動 CPU」模組中發生故障，配對的「待機 CPU」會繼續執行 PLC 系統作業。這種組態就稱為 CPU 備援。

(1) 1 對 1 備援系統

每個備援系統中，一個主動 CPU 都會配有一個待機 CPU。CPU0-CPU1、CPU2-CPU3、CPU4-CPU5 及 CPU6- CPU7 每一對都代表一對主動 CPU 及待機 CPU。因此，當建立多 CPU 1 對 1 備援系統時，最多可以安裝四個 CPU。在此情況下，會使用相同的應用程式。

< 備援系統組態範例 >

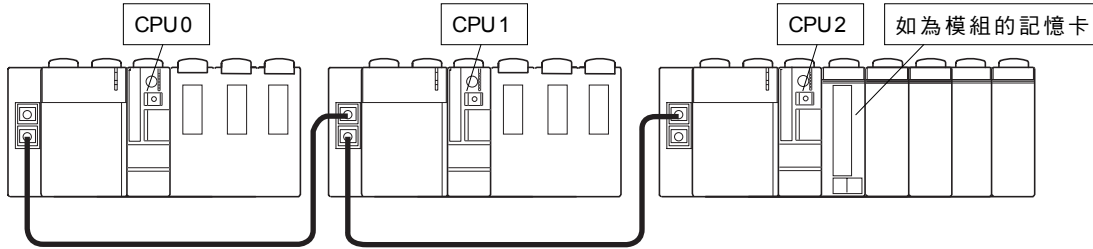
一般而言，主動 CPU 與待機 CPU 會加載到不同的基板上，而由 CPU 所控制的其他模組則會加載到其他基板上，如下所示。如果主動 CPU 故障而啟動了待機 CPU，這個組態便可於待機 CPU 作業期間取代故障的 CPU。



(2) N 對 1 備援系統

針對多個（2 至 7 個）主動 CPU 配備一個待機 CPU 的備援系統。每一個組態最多可以定義兩對 N 對 1 的備援群組。在登錄群組中由最大數目的 CPU 作為待機 CPU 使用的 CPU 模組。

< N 對 1 備援系統組態範例 >



* 關於雙工系統的詳細規格與設定程序，請參考「第 5 節 CPU 雙工系統」。

4-3-3 系統軟故障啟動

(1) 軟故障停用

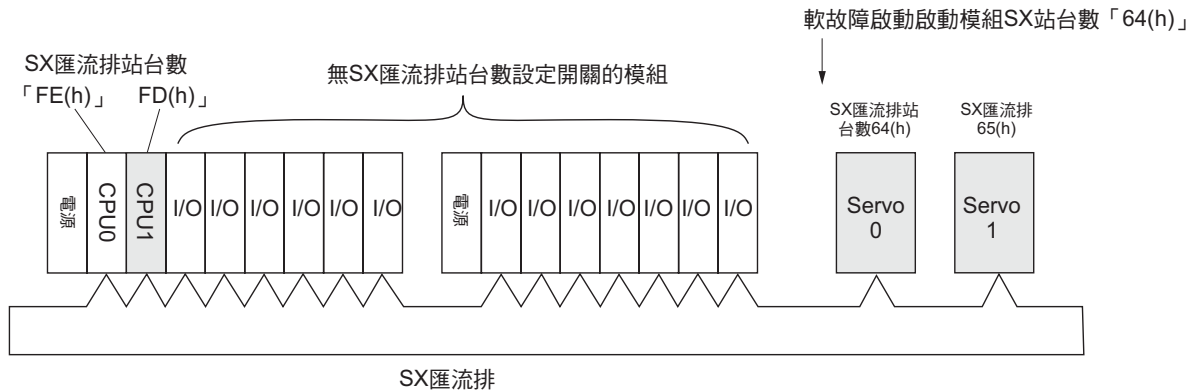
如果登錄於系統定義中的所有模組並未在系統組態檢查等待時間（預設值：20 秒）之內啟動，便會發生系統錯誤（CPU ALM 變亮）。

(2) 擁有 SX 匯流排站台數的模組的部分軟故障啟動

在 MICREX-SX 系統啟動時，如果偵測到了電源未開啟的某些模組，系統便會重新啟動，並於系統組態檢查等待時間結束之後排除掉這些模組。

系統會在非致命故障之下執行（CPU 模組「RUN」為開啟，「ALM」為開啟）。

< 軟故障啟動時的系統組態範例 >



在以上範例中，要部分軟故障啟動的模組在 中為「servo 0」、「servo 1」及「CPU 1」。

- 1) 軟故障啟動啟動模組的範圍位於 SX 匯流排站台數中，其已設定為要軟故障啟動至 253 的第一個。註 1) 設定數字不含要軟故障啟動的模組的 SX 匯流排站台數的啟動數。
- 2) 務必在宣告為啟動數字之後僅將 SX 站台數指定給含有 SX 匯流排站台數設定開關的模組。註 2) 您不需要為它們指定連續的數字。
- 3) 當在軟故障啟動（在非致命故障狀態之下）之後開啟了要軟故障啟動的模組時，模組便會被啟動，系統也會正常執行（「RUN」為開啟，「ALM」為關閉）。

註：1) CPU 模組、PE 連結模組及 P 連結模組也都是軟故障啟動的。注意，CPU0 已排除。

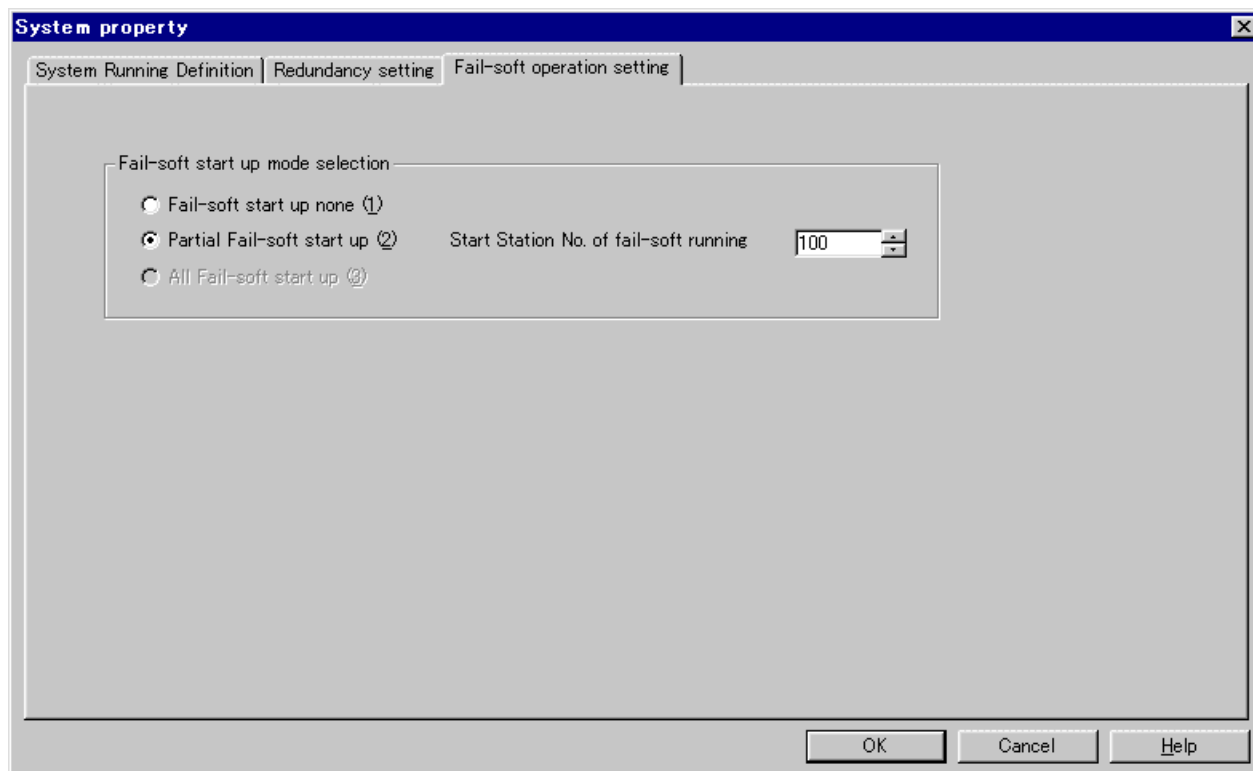
2) 如果偵測到不含 SX 匯流排站台數設定開關的任何模組，系統致命故障（在初始啟動時）或非致命故障（在接續啟動時）便會發生。

3) 務必僅在軟故障相容模組（版本 20** 或以上）中使用系統軟故障啟動模式。

如果在初始啟動或接續啟動時偵測到了任何與軟故障不相容的模組，便會發生系統致命故障。

< 設定系統軟故障啟動 >

1) 呼叫「系統屬性」對話框，並使用滑鼠左鍵按下「軟故障作業設定」標籤。



2) 選擇已啟用的「部分軟故障啟動」(2) 並輸入軟故障啟動數字。

3) 當填入欄位之後，使用滑鼠左鍵來按下 [確定] 按鈕。

4-4-1 CPU 作業定義

CPU 作業定義包含監視器計時器、電源開啟時間作業及低電量作業。

(1) 監視器計時器

使用者可以設定 1 毫秒至 4,095 毫秒之間的監視器計時器預設值。預設值為 4,095 毫秒。

(2) 電源開啟時間作業

此參數指定了在 CPU 模組前方面板上的主開關設定為「RUN」或「TERM」時，開啟系統電源後 CPU 模組會執行的作業。下表表示 CPU 模組主開關位置與 CPU 模組作業之間的關係。預設值為「RUN = 執行中 / TERM = 執行中」。

<CPU 作業與主開關位置>

系統定義設定	作業	
	RUN	TERM
RUN=執行/TERM=執行	執行	執行
RUN=執行/TERM=先前狀態	執行	先前狀態（註）
RUN=停止/TERM=停止	停止	停止

註：先前狀態是在關閉系統電源之前的 CPU 建立狀態。如果 CPU 正在執行中，此狀態會被設定為執行，而如果 CPU 已停止，則會設定為停止。

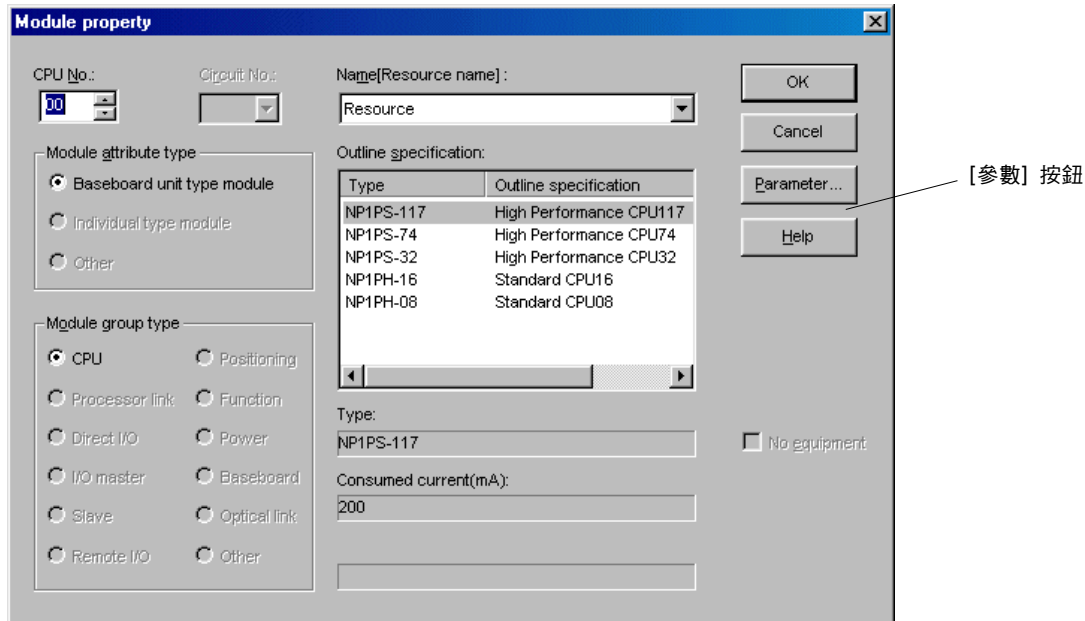
(3) 低電量作業

在低電量執行時，當啟動 CPU 模組時，CPU 模組會初始化它的記憶體以開始作業；不會偵測到任何資料備份錯誤，預設值為停用低電量執行。至於標準 CPU，如果它們沒有插入使用者 ROM 卡（MP8PMF-16）則不適用低電量執行。

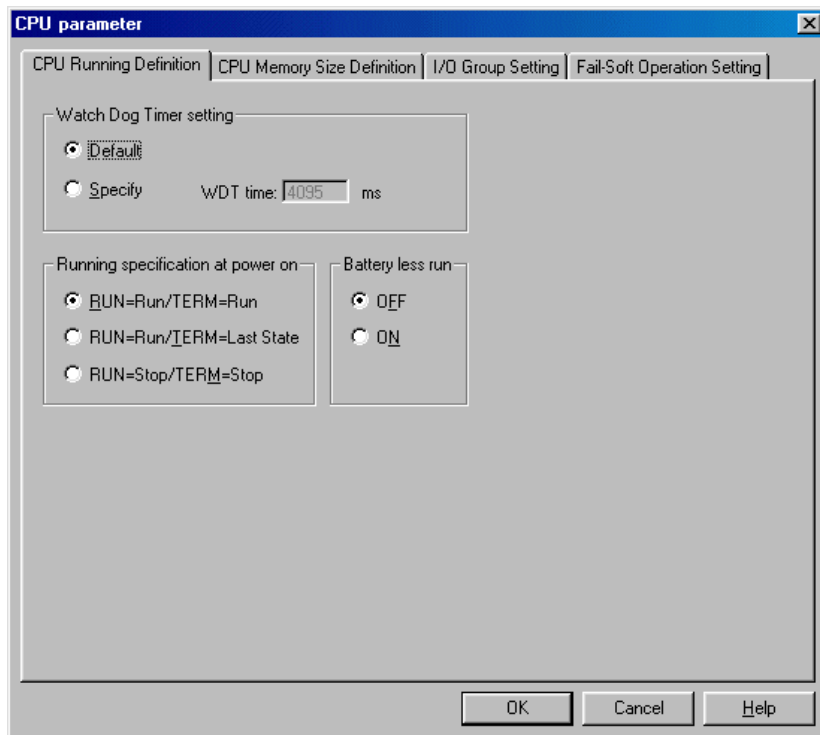
4-4 CPU 參數

< 設定程序 >

從系統組態畫面中選擇 CPU，然後按下 [內容] 按鈕。此時您會看到 CPU 的「模組內容」對話框。



按下 [參數] 按鈕。此時您會看到「CPU 參數」對話框。
對話框包含四個標籤頁，先顯示的是「CPU 執行定義」標籤頁。



當設定好必要的項目之後，使用滑鼠左鍵按下 [確定] 按鈕。

4-4-2 CPU 記憶體大小定義

使用者可以設定 CPU 模組中的資料記憶體大小。雖然使用者記憶體區域的大小為 32K 字，且每一個區域都以預設大小所定義，但使用者仍可視需要更改每一個區域的大小。每一個區域的大小皆可以每段增加 0.5K 字的差距來更改。

< 允許的記憶體區域大小範圍 >

使用者記憶體區域的大小能夠以底下顯示的範圍進行變更。注意，使用者記憶體的總字數是固定的。例如，要為高效能 CPU (NP1PS-32) 的標準記憶體設定 32K 字，請將其他記憶體區域全部指定為 0s。

高效能 CPU (NP1PS-32/NP1PS-32R)

記憶體類型		最小數值	初始數值	最大數值
標準記憶體 (非保留記憶體)	(WM)	2K字	8K字	32K字
保留記憶體	(WL) (WK)	0K字	4K字	30K字
使用者FB記憶體		0K字	4K字	14K字
系統FB記憶體		0K字	16K字	30K字

高效能 CPU (NP1PS-74/NP1PS-742R)

記憶體類型		最小數值	初始數值	最大數值
標準記憶體 (非保留記憶體)	(WM)	2K字	32K字	128K字
保留記憶體	(WL) (WK)	0K字	16K字	126K字
使用者FB記憶體		0K字	16K字	113K字
系統FB記憶體		0K字	64K字	126K字

高效能 CPU (NP1PS-117R/NP1PS-117R)

記憶體類型		最小數值	初始數值	最大數值
標準記憶體 (非保留記憶體)	(WM)	2K字	128K字	256K字
保留記憶體	(WL) (WK)	0K字	32K字	254K字
使用者FB記憶體		0K字	32K字	113K字
系統FB記憶體		0K字	64K字	254K字

標準 CPU (NP1PH-16)

記憶體類型		最小數值	初始數值	最大數值
標準記憶體 (非保留記憶體)	(WM)	0K字	8K字	31K字
保留記憶體	(WL) (WK)	0K字	4K字	31K字
使用者FB記憶體		0K字	4K字	14928K字
系統FB記憶體		0K字	8K字	31K字
初始數值設定區域		0K字	7K字	31K字

標準 CPU (NP1PH-08)

記憶體類型		最小數值	初始數值	最大數值
標準記憶體 (非保留記憶體)	(WM)	0K字	4K字	15K字
保留記憶體	(WL) (WK)	0K字	2K字	15K字
使用者FB記憶體		0K字	4K字	7216K字
系統FB記憶體		0K字	2K字	15K字
初始數值設定區域		0K字	3K字	15K字

4-4 CPU 參數

< 設定程序 >

按下 [CPU 記憶體大小定義] 標籤。此時您會看到「CPU 記憶體大小定義」。

The screenshot shows the 'CPU parameter' dialog box with the 'CPU Memory Size Definition' tab selected. The dialog box has a title bar with a close button (X). Below the title bar are four tabs: 'CPU Running Definition', 'CPU Memory Size Definition', 'I/O Group Setting', and 'Fail-Soft Operation Setting'. The main area contains several input fields and labels:

- Non retain memory: 128.0 KW. Range of word address: WM000000 - WM01FFFD.
- Retain memory: 32.0 KW. Range of word address: WL000000 - WL007FFF.
- User FB memory: 32.0 KW.
- System FB memory: 64.0 KW.
- Initial data: 3200. A 'Default' button is next to it.
- Detail of system FB memory section:
 - Edge detection: 4096 Point x 2W. 8192 W.
 - Counter: 1024 Point x 4W. 4096 W.
 - Additional timer: 512 Point x 8W. 4096 W.
 - Timer: 2048 Point x 8W. 16384 W.
 - Other system FB area: 32768 W.

At the bottom of the dialog box are three buttons: 'OK', 'Cancel', and 'Help'.

設定好記憶體容量之後，請按下 [確定] 按鈕。

註：標準 CPU 的初始資料之數目會因「非保留記憶體」、「保留記憶體」、「使用者 FB 記憶體」及「系統 FB 記憶體」的設定而不同。（使用者 FB 初始數值區域也會自動計算。）

4-4-3 I/O 群組設定

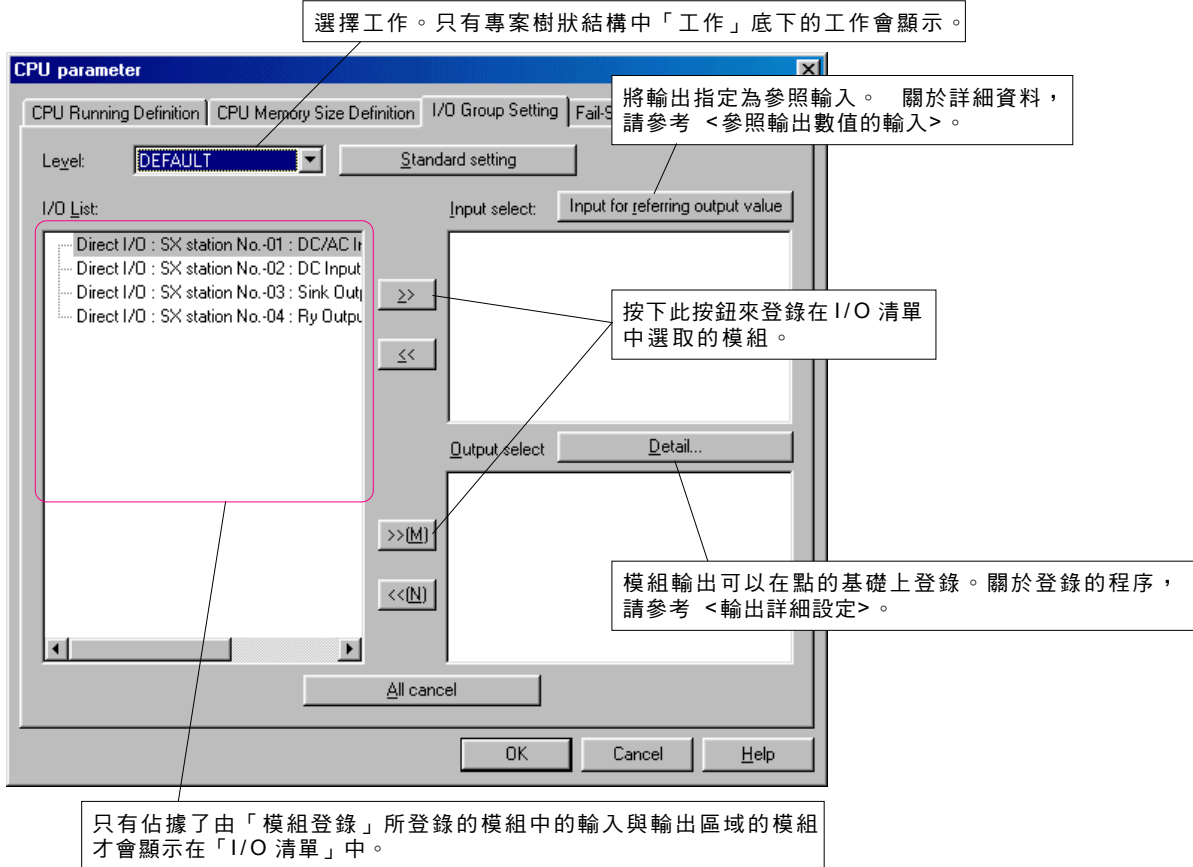
此設定指定在 SPH 系統內控制輸入/輸出模組的 CPU 與工作。

對於要存取模組輸入與輸出的 CPU 模組而言，此設定是很重要的。

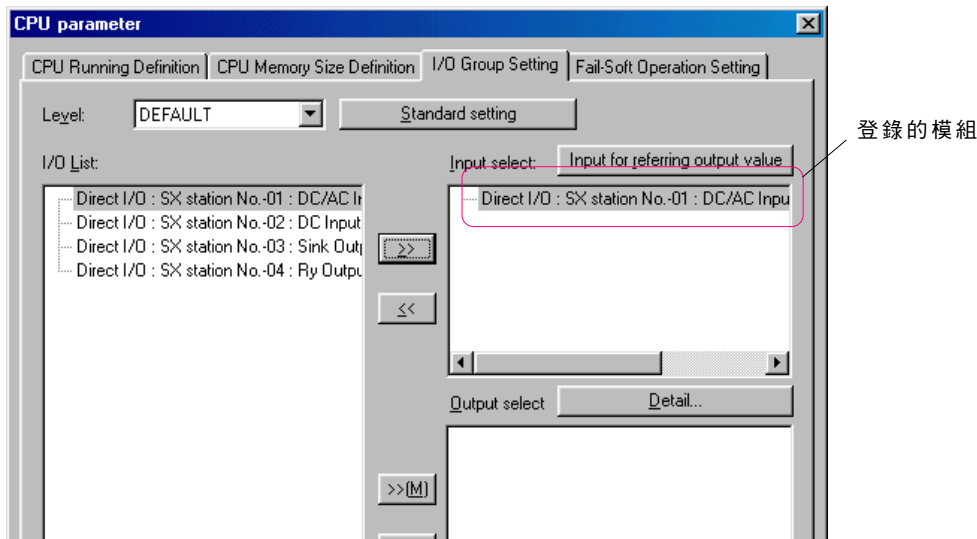
* 在版本 V1.3 或更新版本中，當系統定義在設定沒有 I/O 群組的情況下終止時，設定值會自動登錄到預設工作中。當系統定義終止時，會顯示出警告訊息。

<標準設定 (1)>

按下「CPU 參數」對話框中的 [I/O 群組設定] 標籤。此時您會看到「I/O 群組設定」的標籤頁。



從「I/O 清單」中選擇（按下）模組，然後按下登錄按鈕，選取的模組會被登錄。

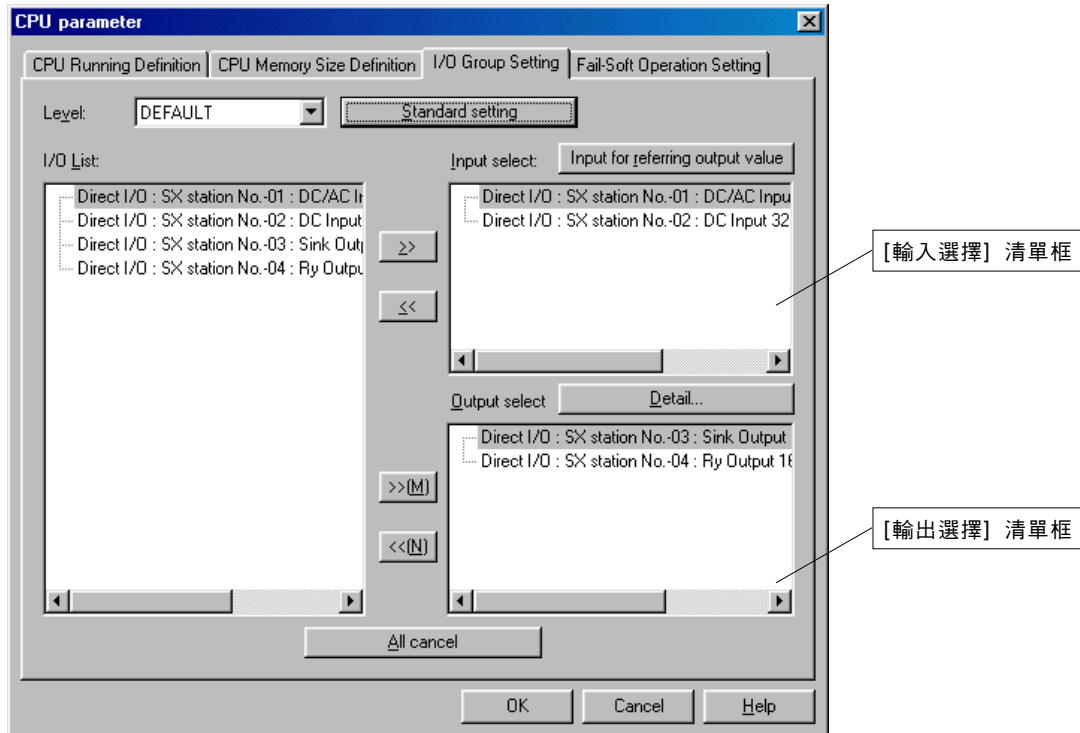


登錄要控制的所有模組後按下 [確定] 按鈕。

<標準設定 (2)>

只有當一個 CPU 登錄於組態中，且只有「預設」工作控制 I/O 時，模組才能一次登錄。

選擇「預設」作為「階層」，然後按下 [標準設定] 按鈕。模組會自動登錄到輸入選擇框及輸出選擇框中。



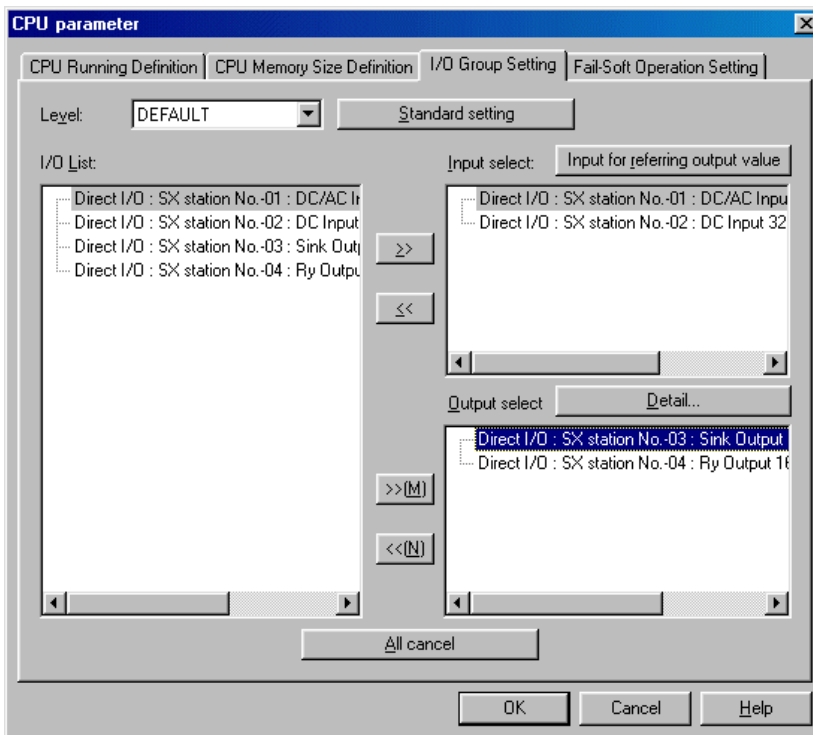
註 1：必須要在 [輸入選擇] 清單框及 [輸出選擇] 清單框中以輸入/輸出區域來登錄模組，例如高速計數器及定位模組。

註 2：要在多 CPU 系統組態中參照為其他 CPU 模組所登錄的輸出模組，必須在 [輸入選擇] 清單框中登錄輸出模組。

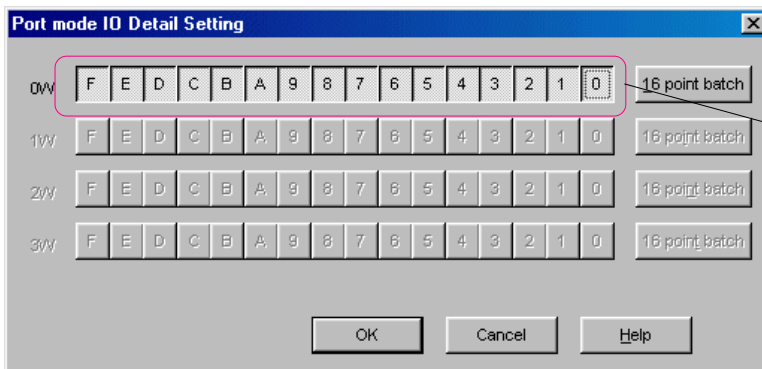
在 [輸出選擇] 清單框中針對其他 CPU 模組登錄的輸出模組不能夠以複製的方式登錄。

<標準設定 (3)>

在多 CPU 系統組態之下，您可以用位元或字的基礎來指定控制模組輸出的 CPU。
從 [輸出選擇] 中為「輸出詳細設定」選擇一個模組後按下 [詳細資料] 按鈕。

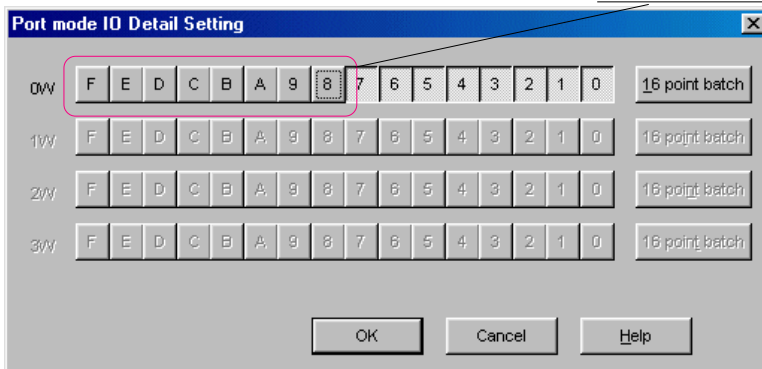


當選取數位輸出模組之後，此時您會看到「連接埠模組 IO 詳細設定」對話框。



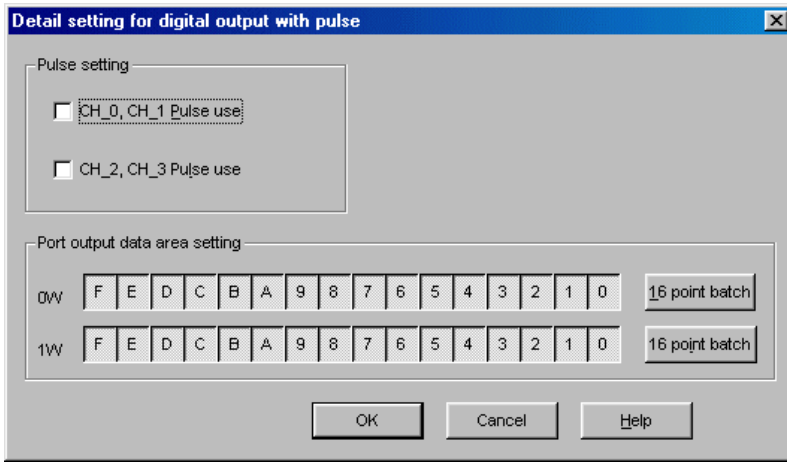
按照預設值，所有位元都會被選取。

按下相對於要由其他 CPU 所控制的位元的按鈕來將它設定為「關閉」。

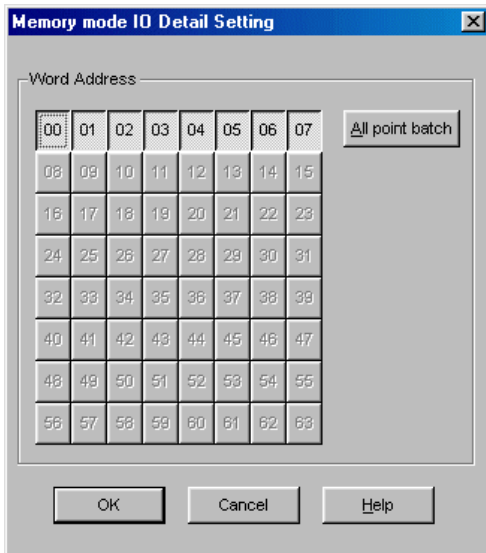


註1：根據選取的模組，當您按下 [詳細資料] 按鈕時所顯示的對話框可能會有所不同。

< 當選取含有脈衝串列輸出的數位輸出模組時 >



< 當選取類比輸出模組時 >

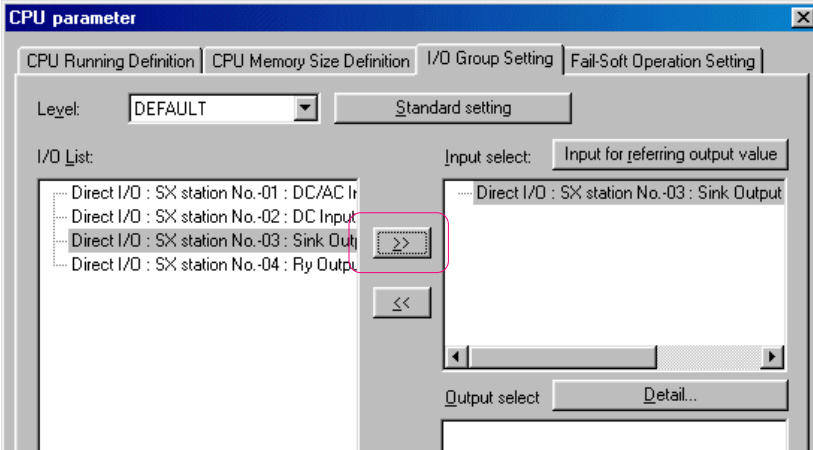


註2：如果您選擇無法執行「輸出詳細資料設定」的模組，即使您按下 [詳細資料] 按鈕也不會顯示詳細資料定的對話框。

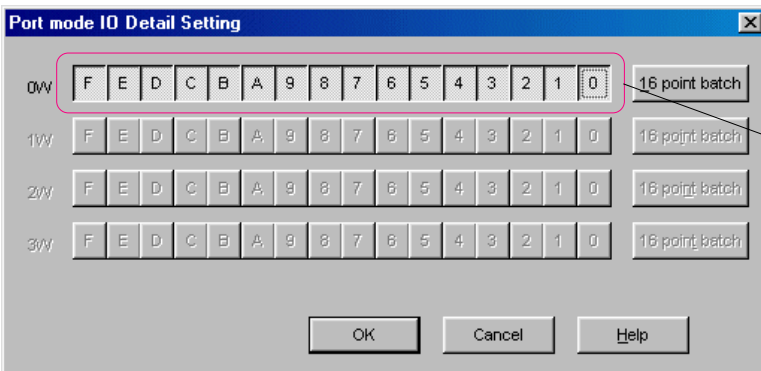
<標準設定 (4)>

在多 CPU 系統組態之下，如果您要使用以本地 CPU 的應用程式、將其他 CPU 用來作為「輸入」的輸出模組之輸出，請針對本地 CPU 的 I/O 群組將位元或字登錄至「輸入選擇」。

選擇要參照的輸出模組，按下 [>>] 按鈕，然後將它登錄至「輸入選擇」。

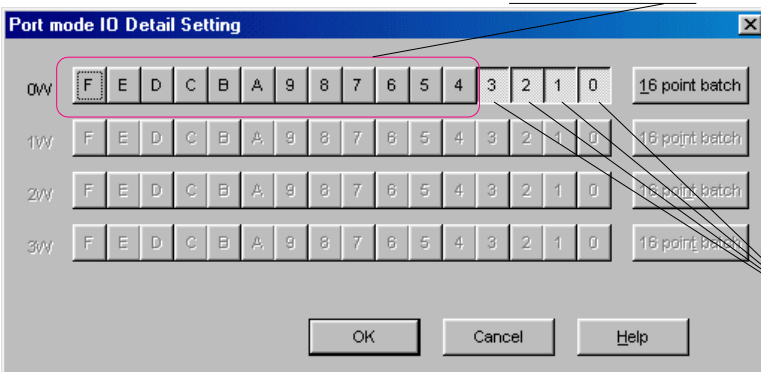


按下 [輸入以參照輸出數值] 按鈕。此時您會看到「連接埠模式 IO 詳細設定」對話框。



按照預設值，所有位元都會被選取。

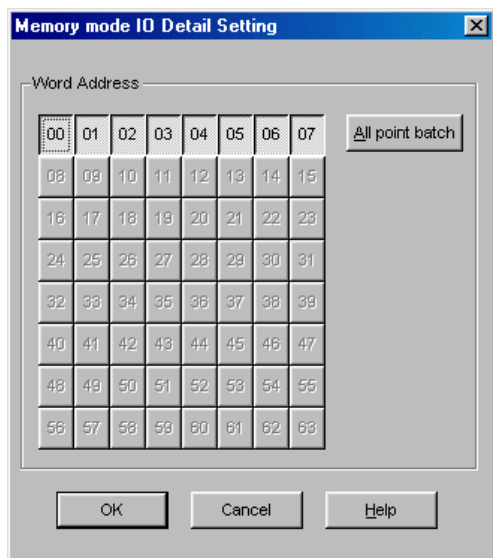
按下相對於無輸出數值參照的位元的按鈕，將它設定為「關閉」。



含有輸出數值參照的位元

按下 [確定] 按鈕。

註：在參照類比輸出或其他輸出數值時，必須要在字的基礎上設定。
在此情況下您會看到「記憶體模式IO詳細設定」對話框。



4-4-4 軟故障執行

即使在啟用軟故障時，在模組中發生故障且系統執行過程中當機，整個系統仍然可以繼續執行。

<MICREX-SX 系統的軟故障作業>

1) 模組當機

即使啟用了軟故障的模組發生故障，並於系統執行時當機，CPU 仍可在非致命系統故障的情況下繼續執行（「RUN」為「開啟」，「ALM」為開啟）。如果停用了軟故障的模組發生故障且在系統執行時當機，CPU 會在致命系統故障的情況下停止執行（「RUN」為「關閉」，「ALM」為開啟）。

2) 模組復原

當當機的模組復原時，系統會恢復至其正常狀態。注意，如果有一個以上的模組當機了，除非所有當機的模組都復原了，否則將無法啟動任何模組。

3) 啟用軟故障的模組

不含I/O區域的一般模組	CPU、P/PE連結及一般通訊模組	無條件的情況下，軟故障適用
含I/O區域的模組	數位I/O、類比I/O及AS-I主模組	登錄啟用軟故障的模組

註：如果您要在軟故障無條件適用的情況下在模組中發生故障時停止系統執行，請監控系統記憶體（WSM44 至 WSM53）中的 SPH 系統故障資訊，並將使用者故障旗標（WSME 至 WSM10）設定為「開啟」。

4) I/O 群組及軟故障登錄

模組 CPU 無法控制未登錄於 I/O 群組中的模組（已指定 I/O 區域）。即使模組當機，CPU 仍可繼續正常執行。

建立軟故障系統的考量因素

底下列出支援軟故障的模組及基板單元版本。

- 已安裝內部韌體的模組 1030
- 未安裝內部韌體的模組 10（如基板單元）

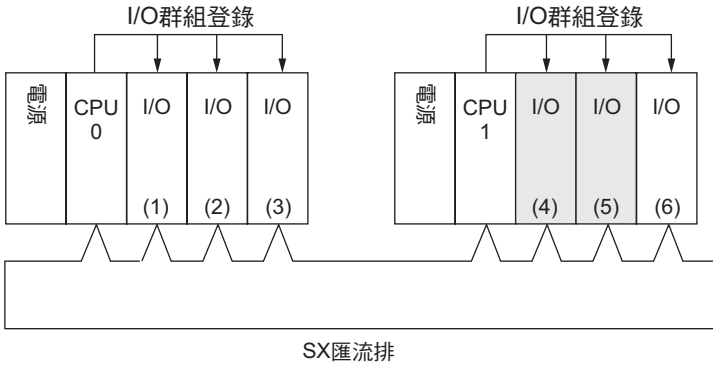
當任何早於以上列出的版本之模組已連接至 SX 匯流排時，不會有軟故障建置於其上。在系統記憶體中，SX 匯流排連接模組軟故障模式旗標（SMC）會被設定為「關閉」。

註：即使是在已設定了與軟故障不相容的模組的系統中，仍可使用遠端 I/O 軟故障功能。

5) 多 CPU 系統的軟故障登錄

針對多 CPU 系統，請在 I/O 群組中登錄由自我 CPU 所控制的模組，並針對軟故障予以登錄。系統在故障時將依下述方式操作。

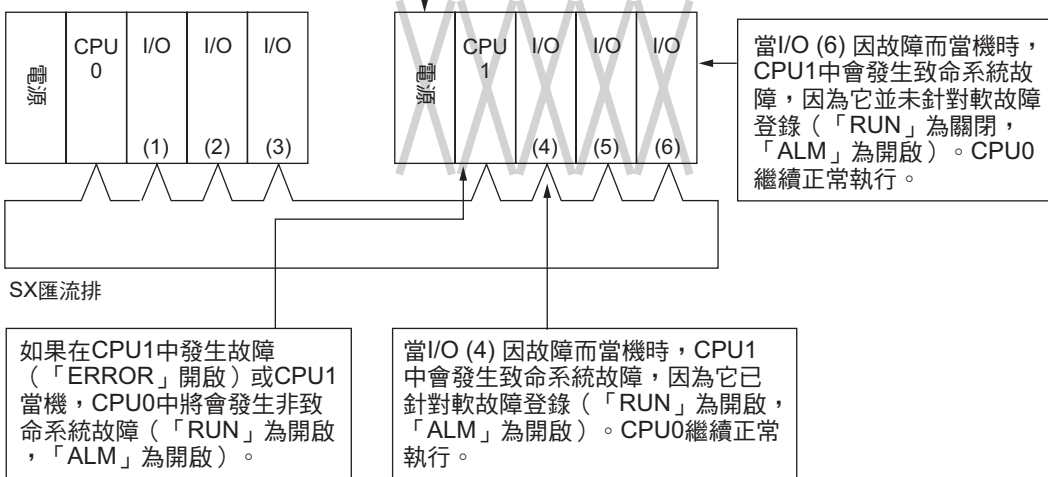
<系統組態範例>



在CPU0中，將I/O (1)、(2) 與 (3) 一起登錄為一個I/O群組，並針對軟故障予以登錄。

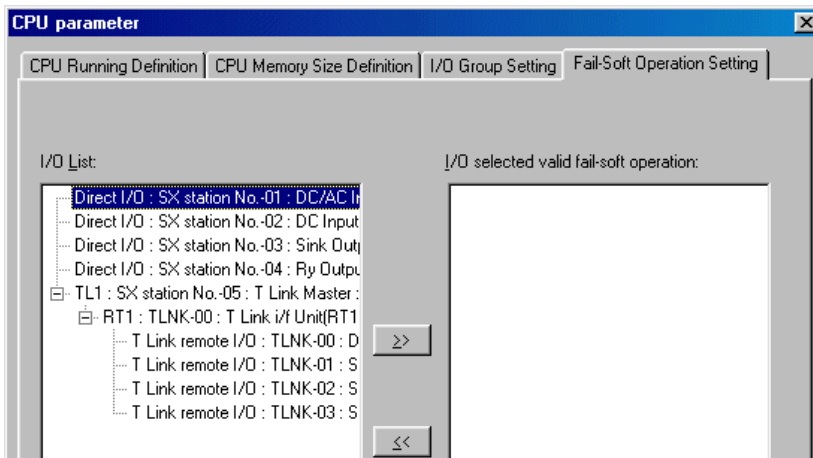
在CPU1中，將I/O (4)、(5) 與 (6) 一起登錄為一個I/O群組，並針對軟故障登錄 (4) 與 (5)。

<出現任何故障的系統作業>

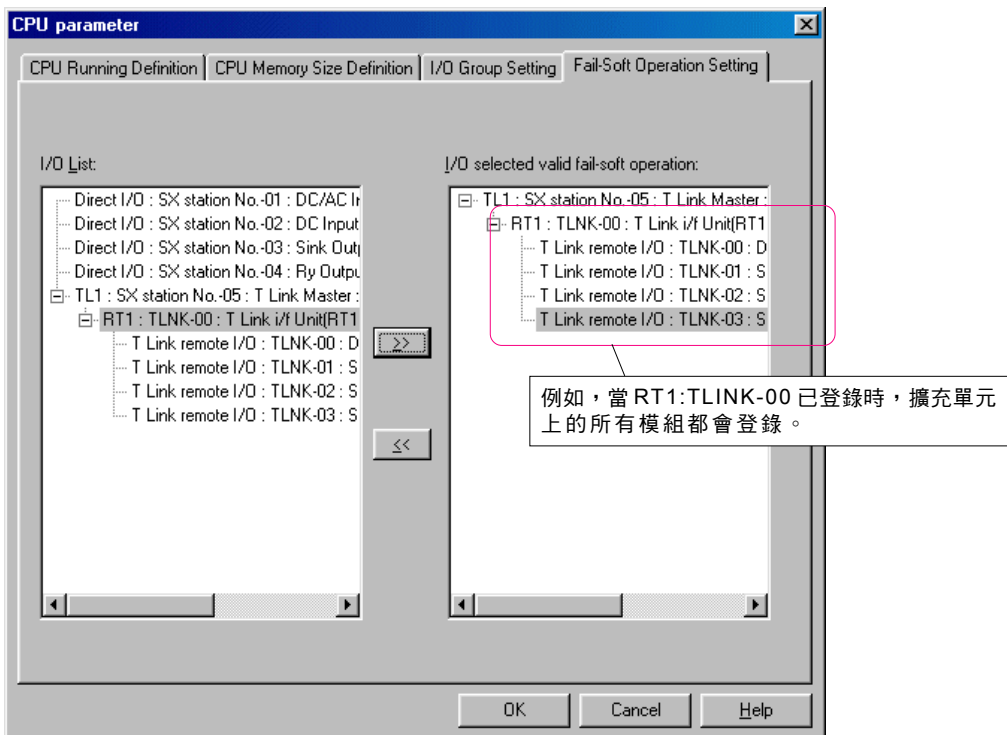


<設定程序>

按下「CPU 參數」對話框中的 [軟故障作業設定] 標籤。此時您會看到「軟故障作業設定」標籤頁。



選擇軟故障作業設定的模組，然後按下 [>>] 按鈕。



按下 [確定] 按鈕。

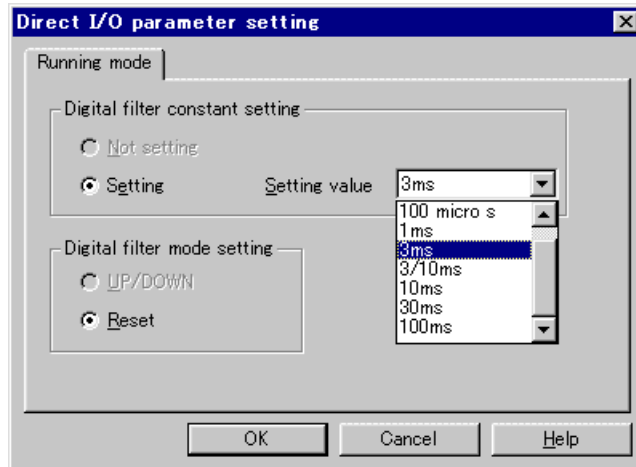
註：對於遠端I/O上的擴充單元來說，會針對每個單獨單元登錄軟故障作業。針對每個模組登錄是不可能的。

4-5-1 輸入過濾時間

輸入過濾時間是針對 DC 型數位輸入模組設定的。時間是以（「關閉」-至-「開啟」時間）-（「開啟」-至-「關閉」時間）的格式設定。使用者可以選擇的數值有 1-1 毫秒、3-3 毫秒（預設）、3-10 毫秒、10-10 毫秒、30-30 毫秒、100-100 毫秒及無過濾。只有針對快速輸入模組（NP1X3206-A）才可以設定無過濾器或 100-100 微秒輸入過濾時間。

< 設定程序 >

- 1) 在系統定義樹狀結構中選擇要設定輸入過濾時間的數位輸入模組，並開啟參數設定對話框。



- 2) 指定過濾時間後用滑鼠左鍵按下 [確定] 按鈕。

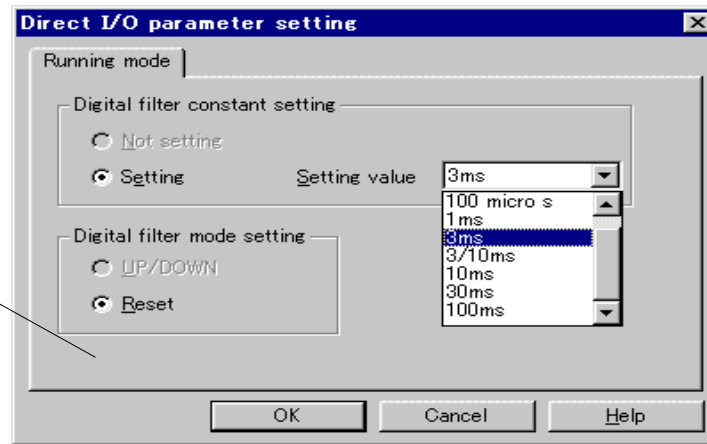
4-5-2 輸出保留定義

輸出保留定義可以用來在系統錯誤發生或 CPU 模組關機前立即保留所建立的輸出狀態，或在 CPU 停止且被保留為停止狀態之前立即保留所建立的輸出狀態。

<設定程序>

- 1) 在系統定義樹狀結構中選擇要設定輸出保留選項的數位輸出模組，並開啟參數設定對話框。

用滑鼠左鍵按下「保留模式 (H)」。



- 2) 選擇保留模式後用滑鼠左鍵按下 [確定] 按鈕。

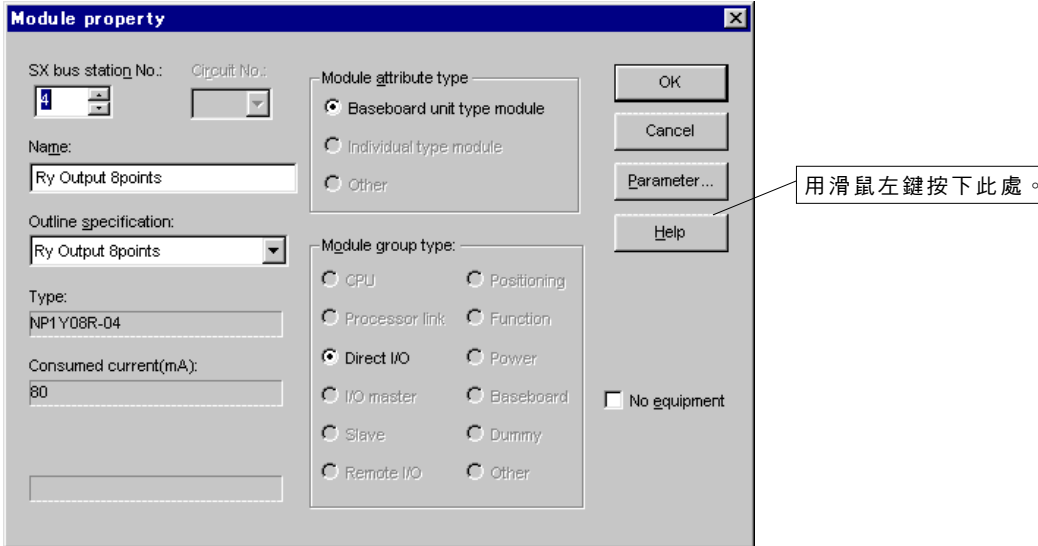
註：無法設定 OPCN-1 上的遠端 I/O 之輸出保留。

4-5-3 系統輸出定義

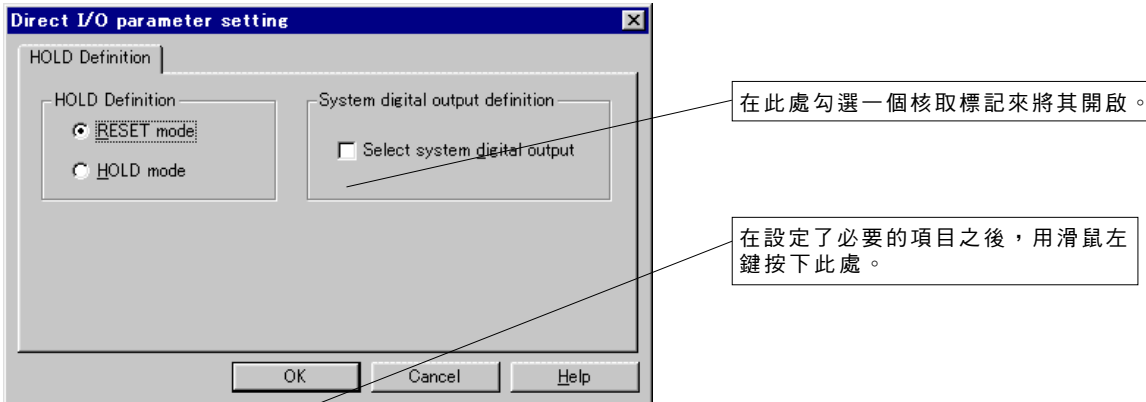
使用者可以針對每個SPH系統定義一個位元，來指出系統作業狀態，無論應用程式為何。當整個系統正常執行時，此位元會被設定為「開啟」，如果系統出現錯誤，則會設定為「關閉」。可以被設定的位元是輸出模組的位元 0。

< 設定程序 >

- 1) 選擇要設定系統輸出的數位輸出模組。
- 2) 此時您會看到「模組內容」對話框。



- 3) 此時您會看到「直接 I/O 參數設定」對話框。



- 註： 1) 要指定做為系統輸出的模組的位元 0 無法登錄在 I/O 群組定義中。
 2) 要指定做為系統輸出的模組的位元 0 無法登錄在除 CPU0 之外的任一 CPU 之 I/O 群組定義中。
 3) 「保留」模式與軟故障都無法設定成指定做為系統輸出的模組。

第 5 節 CPU 雙工系統

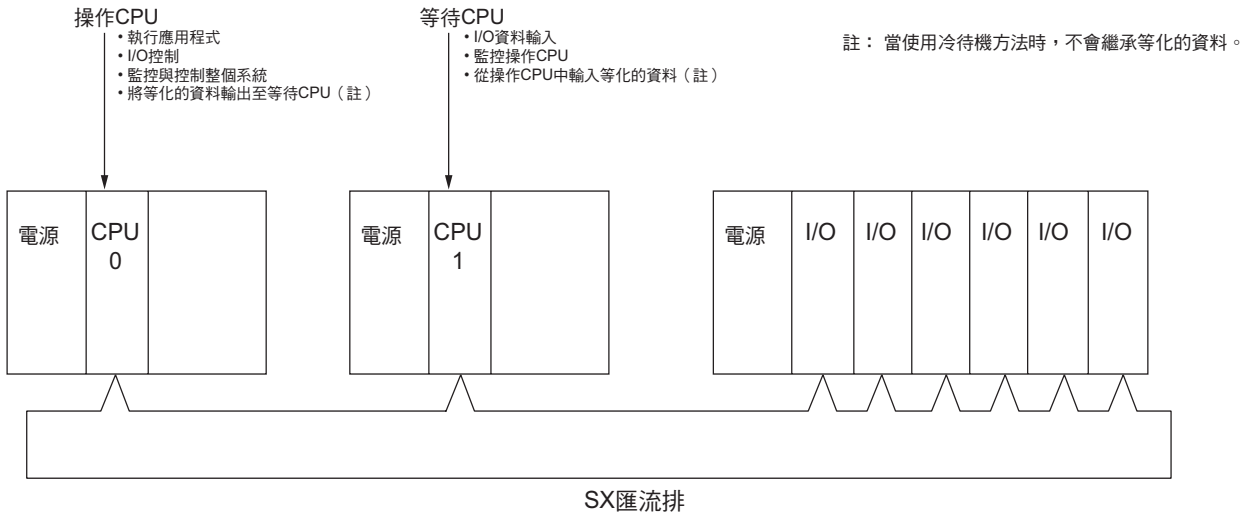
	頁碼
5-1 雙工模式中的系統作業	5-1
5-1-1 1 對 1 雙工模式	5-1
(1) 系統作業	5-1
(2) 將錯誤 CPU 更換為新 CPU	5-1
5-1-2 N 對 1 雙工模式	5-2
(1) 系統作業	5-2
(2) 將錯誤 CPU 更換為新 CPU	5-2
5-2 操作及等待 CPU 與效能之間的對換條件	5-3
5-2-1 對換的條件	5-3
5-2-2 雙工模式下的系統效能	5-3
5-2-3 多 CPU 傳遞切換	5-4
5-2-4 資料等化	5-5
(1) 由使用者指定的等化資料	5-5
(2) 由系統自動等化的區域	5-6
(3) 等化的資料大小	5-7
(4) 資料等化	5-8
5-2-5 操作與等待 CPU 之間對換時的記憶體作業	5-9
5-2-6 CPU 模組 LED 與輸出至顯示系統	5-10
5-2-7 雙工系統的應用程式	5-11
(1) 連續啟動 CPU 模組	5-11
(2) 1 對 1 雙工系統	5-11
(3) N 對 1 雙工系統	5-11
(4) 系統定義	5-11

它被稱為雙工模式是因為在控制系統中使用了兩個裝置來改善系統安全性及可依賴性。在 MICREX-SX 系列中，電源模組與 CPU 模組都可以內建到雙工系統中。我們將在本節為您介紹如何將 CPU 模組內建到雙工系統中。MIXREX-SX 的 CPU 雙工模式中包含 1 對 1 與 N 對 1 兩種類型。

5-1-1 1 對 1 雙工模式

在此模式中，一個操作 CPU 與一個等待 CPU 是相關聯的。CPU0-CPU1、CPU2-CPU3、CPU4-CPU5 與 CPU6-CPU7 等的組合被建立做為一對的操作與等待 CPU。在此情況下，會使用相同的應用程式。

< 1 對 1 雙工系統組態範例 >



(1) 系統作業

在系統電源開啟時，系統會開始執行，假設指定的含偶數 CPU 數的 CPU 模組位於操作一邊，而含奇數 CPU 數的 CPU 模組則位於等待一邊。(在上述範例中，CPU0 就是操作 CPU，而 CPU1 則是等待 CPU。)

當操作 CPU 發生錯誤且當機時，等待 CPU 會開始執行。

1 對 1 雙工模式包含兩種類型，暖待機與冷待機，在暖待機中，等待 CPU 會繼承操作 CPU 中的資料，而在冷待機中，等待 CPU 卻不會這樣做。由等待 CPU 所繼承的資料叫做等化資料，其範圍是在系統定義中指定的。

(2) 將錯誤 CPU 更換為新 CPU

在上述範例中，由於只有在加載了 CPU 的基板單元上安裝了電源模組，因此，當 CPU1 執行時，CPU0 會被替換，以取代當機的 CPU0。要將 CPU0 更換為新的 CPU，請關閉 CPU0 電源，並將其更換為新的 CPU0，然後再開啟電源。會假設新的 CPU0 位於等待一邊。

當在兩個 CPU (分別位於操作一邊與等待一邊) 中都發生錯誤時，將兩個系統的電源都關閉，然後再重新啟動它們。

重點

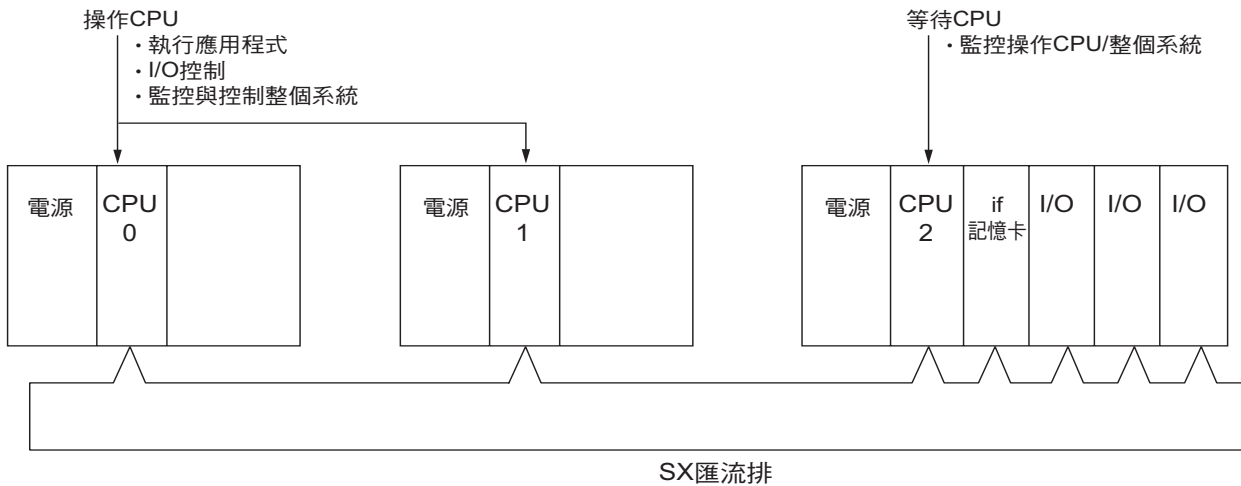
- 在操作與等待CPU中，要安裝相同的應用程式。
- 在暖待機及冷待機其中任一方法中，都會繼承I/O資料。
- 使用載入器可以讓您在操作CPU與等待CPU之間切換。

對於雙工組態中的 CPU 模組來說，必須使用軟體版本 V40 或更高版本的高效能 CPU。V40 之前的版本或標準 CPU 無法用來設定雙工系統。

5-1-2 N 對 1 雙工模式

在這種類型的系統中，多個 (2-7) 操作 CPU 會與一個等待 CPU 相關聯。每一個組態最多可以定義兩組 N 對 1 的雙工群組。在群組中的各 CPU 中，含最大 CPU 數的 CPU 就是等待 CPU。

<2:1 雙工系統組態範例>



(1) 系統作業

在系統電源開啟時，在 N 對 1 雙工系統中，含最大 CPU 數的 CPU 模組會被假設為是等待 CPU。（在上述範例中，CPU0 與 CPU1 是操作 CPU，而 CPU2 則是等待 CPU。）

由於 CPU0 或 CPU1 中發生錯誤而導致系統當機時，等待 CPU 會從記憶卡介面模組下載錯誤 CPU 的程式，並會開始執行。在 N 對 1 雙工模式中，只適用冷待機方法。不會從操作 CPU 中繼承資料。

(2) 將錯誤 CPU 更換為新 CPU

在上述範例中，由於只有在加載了 CPU 的基板單元上安裝了電源模組，因此，當 CPU2 執行時，CPU 會被替換，以取代當機的 CPU1。

要將 CPU1 更換為新的 CPU1，請關閉 CPU1 電源，並將其更換為新的 CPU1，然後再開啟電源。新的 CPU 模組會被假設為位於等待一邊，且正在等待來自載入器或系統電源重設的切換指令。這表示系統沒有位於 N 對 1 雙工系統中。

重點

- N (操作 CPU 數) 應用程式需要儲存在記憶卡介面模組中。
- 在 N 對 1 雙工模式中，只能應適用冷待機方法。不會繼承內部資料與 I/O 資料。
- 使用載入器可以讓您在操作 CPU 與等待 CPU 之間切換。當更換了錯誤 CPU 時，操作 CPU 需要切換為等待 CPU。
- 在 N 對 1 雙工系統中，禁止由模組中的 if 記憶卡前面板上的切換開關來執行程式讀取/寫入作業。請勿使用記憶卡 if 模組來從在 CPU 上執行的應用程式中針對檔案記憶體讀取/寫入存取的 N 對 1 雙工系統執行應用程式的儲存。請為檔案讀取/寫入準備另一個記憶卡 if 模組。如果將檔案記憶體與記憶卡 if 模組一起使用，就會發生存取爭用 (contention)，且在操作 CPU 與等待 CPU 之間會執行對換。

5-2-1 對換的條件

操作 CPU 與等待 CPU 之間會發生對換的條件如下所示。對 1 對 1 與 N 對 1 模式來說，條件都是相同的。

○：對換 -：無對換

操作CPU中 發生致命錯誤	CPU中發生錯誤	<ul style="list-style-type: none"> 應用程式作業處理器中發生錯誤 OS處理器中發生錯誤 	○
	記憶體中發生錯誤	<ul style="list-style-type: none"> 系統記憶體（ROM/RAM）中發生錯誤 應用程式記憶體（ROM/RAM）中發生錯誤 記憶體電池備份中發生錯誤 	○
	SX匯流排中發生錯誤	<ul style="list-style-type: none"> SX匯流排控制LSI中發生錯誤 處理器匯流排存取中發生錯誤（由自身模組引起） 	○
<ul style="list-style-type: none"> 重複的站台數 超出所連接的模組數 SX匯流排傳輸中出現錯誤 I/O重新整理中發生延遲 		○ (註1)	
操作資源中 發生致命錯誤	電源失敗	<ul style="list-style-type: none"> 基板電源關閉 	○
	應用程式錯誤	<ul style="list-style-type: none"> 使用者程式錯誤 應用程式WDT錯誤 應用程式執行錯誤 	○
	I/O模組中發生錯誤	<ul style="list-style-type: none"> 由自身CPU模組控制的連接了SX匯流排的I/O中發生錯誤 及遠端I/O模組（軟故障停用） 	-
	使用者致命錯誤	<ul style="list-style-type: none"> 偵測到使用者致命錯誤 	-
透過載入器的 對換指令	透過載入器在操作CPU與等待CPU之間對換		○
多CPU 傳遞切換	在多CPU雙工系統中（1對1模式），一個CPU會由於一個錯誤被切換，然後是另一個CPU。		○

註： 1) 由於 SX 匯流排是整個系統的共用資源，因此當在兩個 CPU 中都發生錯誤時，無法在操作 CPU 與等待 CPU 之間完成對換。

2) 故意停止的 CPU 不算在對換條件中。

5-2-2 雙工模式下的系統效能

	切換時間	Takt時間	掃描時間
1對1雙工模式冷待機	在130毫秒內 (註2)	與在普通多CPU系統中的Takt時間相同	與在普通多CPU系統中的掃描時間相同
1對1雙工模式暖待機	在130毫秒內 (註2)	普通多CPU系統中的Takt時間 + 1到3毫秒	普通多CPU系統中的掃描時間 + 數毫秒-數十毫秒 (註1)
N對1雙工模式冷待機	數十秒	與在普通多CPU系統中的Takt時間相同	與在普通多CPU系統中的掃描時間相同

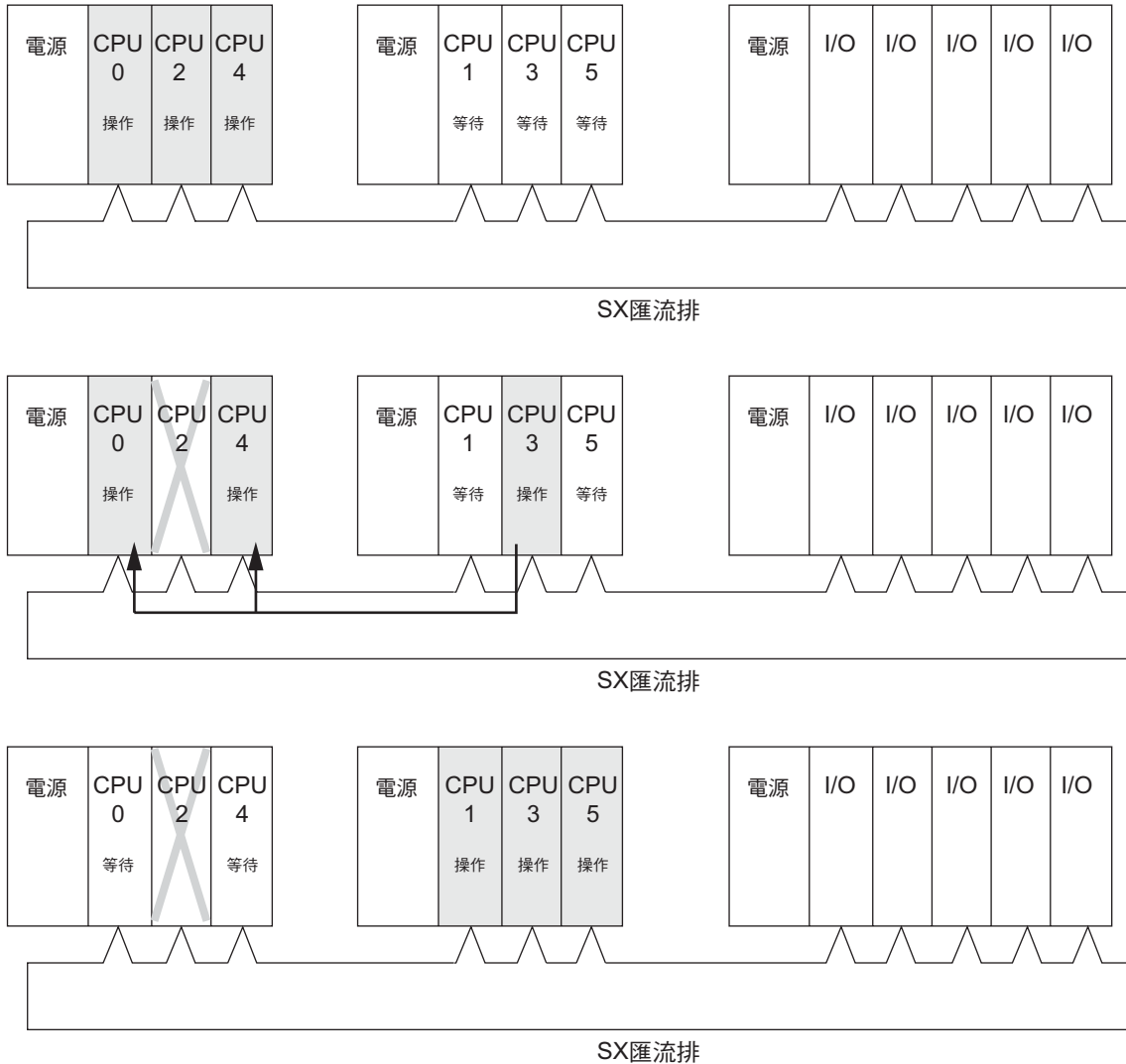
註： 1) 依據等化資料的數量而定。

2) 如果在從操作 CPU 切換為等待 CPU 之後，直接使用與訊息有關的 FB，那麼忙碌狀態將會持續到訊息關閉過程結束為止（100 至 600 毫秒）。

5-2-3 多 CPU 傳遞切換

通常，在 1 對 1 雙工模式下對換是在一對操作與等待 CPU 中完成的。在多 CPU 系統中，當您想要在一對操作與等待 CPU 的對換之後，在另外一對 CPU 之間切換時，便需要設定傳遞切換模式。這可以讓系統自動在兩個 CPU 之間切換。

< 在 1 對 1 雙工模式下三對傳遞切換系統組態的範例 >



在上述範例中，當 CPU2 由於錯誤而停止執行時，CPU3 會開始執行。（切換的時間最多為 130 毫秒。）然後 CPU3 會向 CPU0 與 CPU4 發出切換命令。當接收到命令時，CPU0 與 CPU4 會進入等待模式，而同時 CPU1 與 CPU5 則會被切換到操作模式。（切換的時間最多為 130 毫秒。）

- 註：
- 1) 當 CPU2 在上述系統組態中發生錯誤時，如果在操作 CPU 中發生錯誤，系統則無法切換到等待 CPU。整個系統都會由於致命錯誤而當機。
 - 2) 當操作系統與等待系統都正常執行時，可以完成傳遞切換。
例如，當在初始化（作業一邊與等待一邊都未定義）期間發生對換，即使含啟用傳遞切換的 CPU 也無法切換至另一個 CPU。也就是說，操作 CPU 與等待 CPU 在 CPU 群組中都會針對已經啟用傳遞切換者同時執行。
傳遞切換是否已經正常操作可以藉由檢驗資源執行資訊（WSM48（30h））位元來判斷，設定為「1」，則表示已啟用傳遞切換的 CPU 群組。

5-2-4 資料等化

1 對 1 雙工暖待機組態可以讓您使操作 CPU 的內部資料與待機 CPU 的內部資料相同，這就叫做資料等化。資料等化的指定方式如下所示。

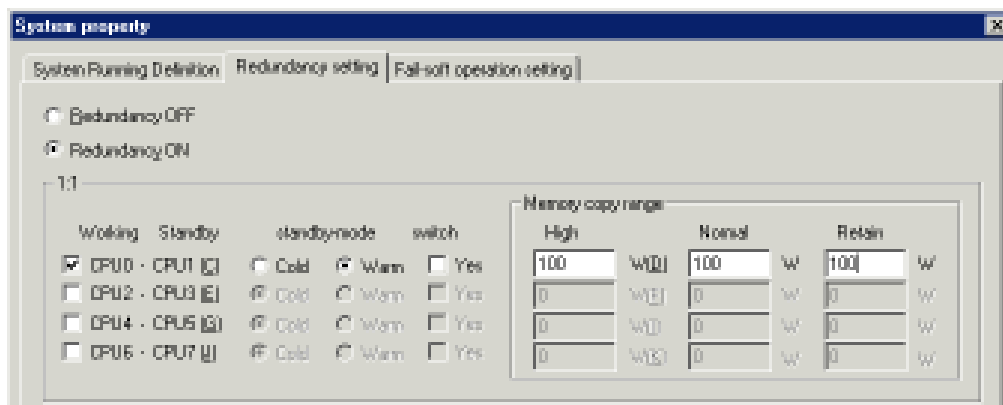
- | | |
|----------------|--|
| ● 透過使用者指定等化的區域 | → 標準記憶體（高速）、標準記憶體與保留記憶體區域
（依指定方式而定（註）） |
| ● 由系統自動等化的區域 | → 在使用者 FB 中指定的含保留的裝置
系統 FB 中含保留屬性的區域
系統 FB 中的計時器目前數值區域 |

註：詳細資料，請參閱<設定使用者指定的等化區域的大小>

等化記憶體的大小會影響預設工作啟動計時。當在雙工系統中等化資料時，需要計算要被等化的資料的大小，並檢查預設工作啟動計時。

(1) 由使用者指定的等化資料

原則上，使用者可以將等化區域指定為從指定區域開始的字數。



來自「備援」設定的數值設定會決定使用者指定的等化大小。

等化大小： S_{PG}

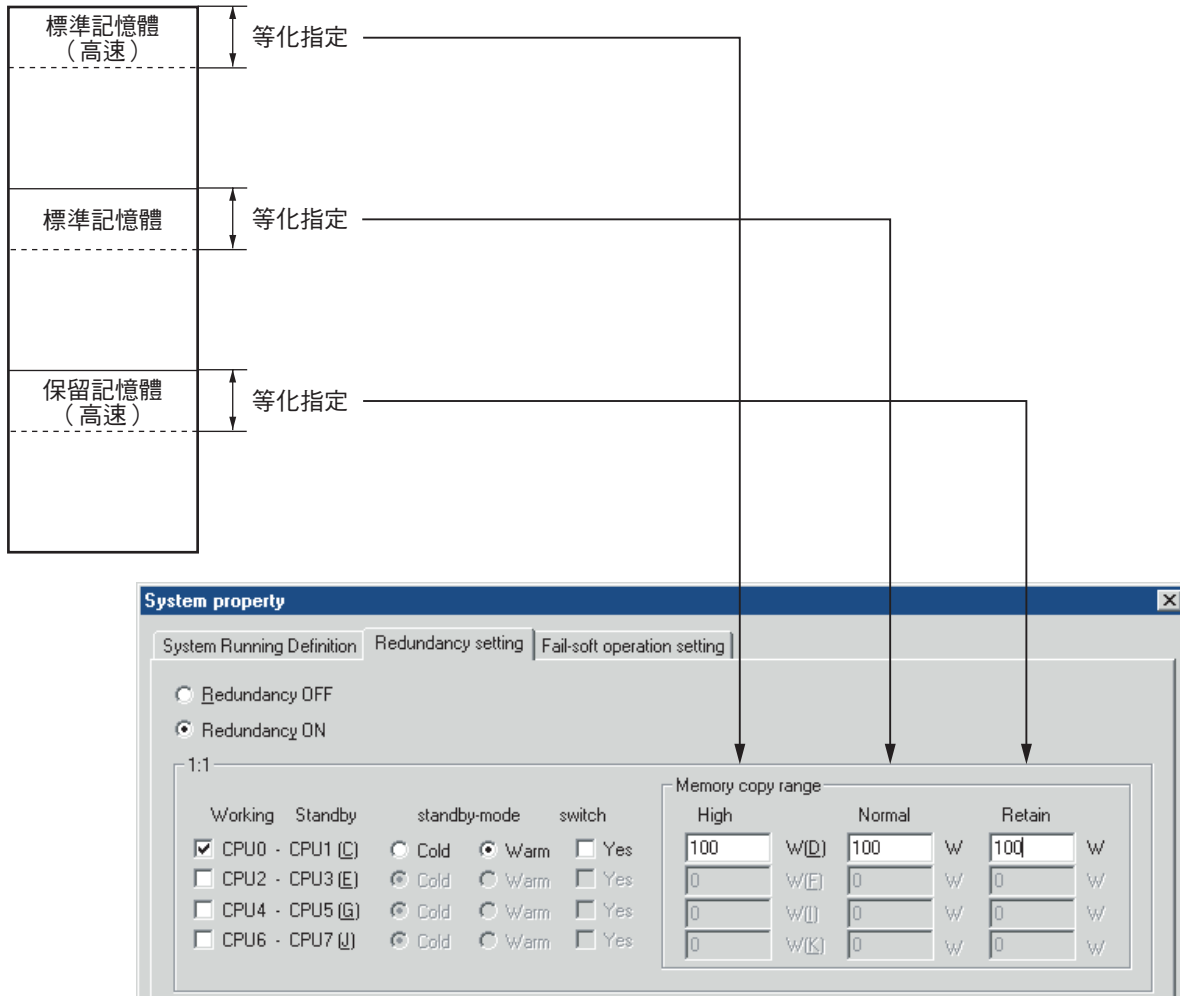
$$S_{PG} = (\text{針對高速記憶體指定的字數}) + (\text{針對標準記憶體指定的字數}) + (\text{針對保留記憶體指定的字數}) (\text{字})$$

在上述範例中，等化大小如下所示。

$$\begin{aligned} (\text{使用者指定的等化大小}) &= 100 + 100 + 100 \\ &= 300 (\text{字}) \end{aligned}$$

< 設定使用者指定的等化區域大小 >

使用 [備援設定] 畫面，由使用者指定等化區域。對每個標準記憶體（高速）、標準記憶體與保留記憶體而言，含有從 [備援設定] 畫面指定的字數之等化區域會從記憶體的開頭開始保留。



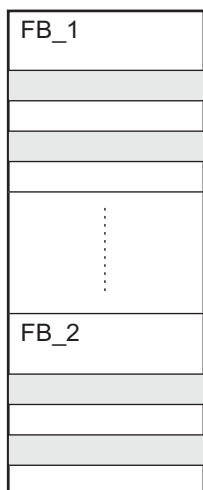
(2) 由系統自動等化的區域

系統會自動等化以下區域：

- 在使用者FB中指定的含保留的記憶體
- 系統FB中含保留屬性與計時器目前數值區域的記憶體

1) 在使用者FB中指定的含保留的記憶體

在使用者FB中含有在初始數值設定清單中檢查的保留記憶體之裝置會自動變為等化區域。



■：含保留指定的裝置
 裝置大小會依照以下方式來處理：
 • 16位元資料類型的裝置 → 1字

* 根據上述內容，計算使用者FB中含保留記憶體指定的裝置之總大小。

註：在所有使用者FB的實例記憶體中，可以指定為保留記憶體的區域總大小被限制為 2048字。

等化大小： S_{UFB} (字)
 S_{UFB} = 含已檢查的保留記憶體之裝置

2) 系統 FB 中含保留屬性與計時器目前數值區域的記憶體

使用者 FB 中含保留屬性的記憶體也會自動成為等化區域，相關的 FB 如下所示。

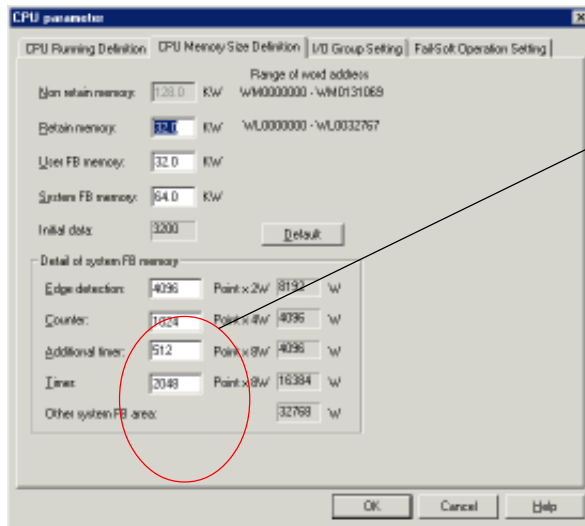
等化的系統FB	等化的實例元素	字數
邊緣偵測 (R_TRIG, F_TRIG) [上升緣聯絡, 下降緣聯絡]	舊輸入數值	2
計數器 (CTU、CTD、CTUD、RCT)	目前計數器數值 舊輸入數值	2
計時器, 計時器總數 (TP、TON、TOF、MR、TMR)	目前計時器數值、舊輸入數值、 時鐘旗標	4

等化大小 S_{SFB}

$S_{SFB} = \text{邊緣偵測指令數} \times 2 + \text{計數器指令數} \times 2 + \text{計時器指令數} \times 4$ (註 1)

註： 1) 包含總計時器數。

2) 等化的指令數指出了在 CPU 記憶體大小定義中 (而不是在程式中) 定義的每個 FB 中的可用指令數。



(3) 等化的資料大小

可以被等化的資料大小是在上述 (1) 與 (2) 中計算出的總字數，在一對的 1 對 1 雙工系統中它被限制為 8192 字。

$$8192 \geq S_{PG} + S_{UFB} + S_{SFB} \text{ (字)}$$

註： 當在 1 對 1 雙工暖待機組態中使用 NP1PS-74 □、NP1PS-117 □ (高效能 CPU 74K 或 117K 產品) 時，請務必考慮記憶體定義中的等化資料。依預設，在記憶體定義中系統 FB 實例記憶體的等化將需要多於 8192 的字。

(4) 資料等化

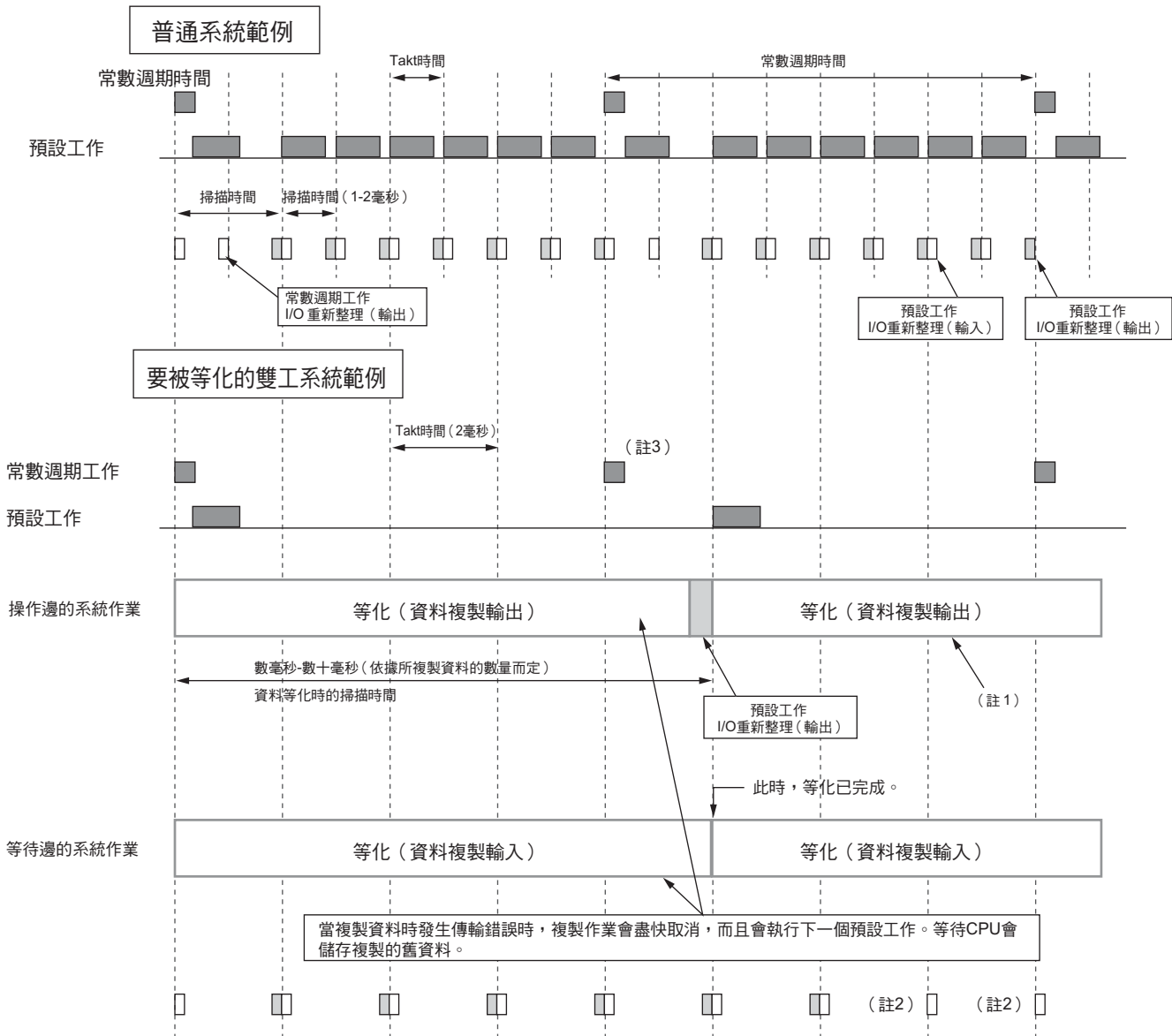
在計算了等化資料的大小之後，可以計算出 CPU 掃描時間。由於資料等化會在執行預設工作之後發生，因此等化資料的大小會影響到預設工作執行間隔（掃描時間）。可以使用下面的公式計算出掃描時間。

雙工系統的掃描時間

$$= (\text{普通系統的掃描時間}) + \left(1 + \frac{\text{等化的總字數}}{512\text{字}}\right) \times \text{tact時間 (毫秒)}$$

捨去的小數

< 等化的計時 >



- 註：
- 1) 當資料複製期間發生對換事件時，等待 CPU 會使用之前複製的等化資料來啟動作業。
 - 2) 當發生對換事件時，資料輸出會從下一個 tact 停止。在等待 CPU 啟動作業之前，I/O 模組輸出不會更新。
 - 3) 固定工作可以設定為比掃描時間更短的時間。但是由於等化處理是在完成預設工作之後執行的，因此，對於循環比掃描時間（預設工作執行計時）短的工作所使用的記憶體來說，等化可能會停用。

5-2-5 操作與等待 CPU 之間對換時的記憶體作業

系統狀態	記憶體或旗標	1對1雙工模式		N對1雙工模式
		冷待機	暖待機	
系統電源開啟 (暖執行)	標準記憶體	已清除		
	保留記憶體	保留的舊數值		
	I/O記憶體	重設保留指定		
	預設工作啟動旗標	開啟		
	初始旗標	關閉		
操作 => 等待	標準記憶體	在執行期間，會保留數值		
	保留記憶體			
	I/O記憶體			
等待 => 操作	標準記憶體	已清除	要被等化的區域以外的區域都會遭到清除，且要被等化的區域中的資料會由操作CPU來繼承。	已清除
	保留記憶體	會保留舊數值。	會保留舊數值，而且要被等化的區域中的資料會由操作CPU來繼承。	已清除
	I/O記憶體	資料仍未變更	資料仍然未變更	重設/保留
	預設工作啟動旗標	開啟	開啟	開啟
	初始旗標	關閉 (註1)	關閉 (註1)	開啟
	操作/等待對換旗標	開啟	開啟	開啟
冷執行 (在程式下載或透過載入器初始啟動時)	標準記憶體	已清除		
	保留記憶體	已清除		
	I/O記憶體	已清除		
	預設工作啟動旗標	開啟		
	初始旗標	開啟		

- 註： 1) 當在程式下載之後等待 CPU 沒有執行時，初始旗標會被設定為「1」。
 2) 在雙工系統中，要在操作 CPU 與等待 CPU 之間對換時，將資料傳送給用於信號的位元變數：
- 將信號的位元變數指定給標準記憶體區域，以避免等化。
 (在對換時，它們會被重設為0。)
 - 當發生對換時，請嘗試再次獲取信號。

重點

<對換時的系統作業>

- 1對1雙工模式待機系統
 系統會以與單一CPU暖執行的系統相同的方式來操作，但是I/O記憶體中的資料不會被繼承。
- 1對1雙工暖待機系統
 這是加上了資料等化的冷待機系統。
- N對1雙工系統
 由於程式是從記憶卡介面模組下載的，因此系統會使用冷待機方法。

5-2-6 CPU 模組 LED 與輸出至顯示系統

○: 開啟, -: 關閉, △: 閃爍

錯誤模組			整個系統會 正常操作	整個系統會 正常停止	操作CPU模組中 發生致命錯誤	等待CPU模組 中發生致命 錯誤	操作CPU模組中 發生致命錯誤	I/O錯誤 (軟故障啟用)	I/O錯誤 (軟故障停用)
模組									
操作 CPU	LED 指示	開啟	○	○	-	○	○	○	○
		錯誤	-	-	○	-	-	-	-
		執行	○	-	-	○	-	○	-
		警示	-	-	○	○	○	○	○
	旗標		正常執行	正常執行	自身CPU資源中 發生致命錯誤	等待CPU中 發生致命錯誤	自身CPU資源中 發生致命錯誤	I/O模組中發生錯誤 自身CPU資源中發 生非致命錯誤	I/O模組中發生錯誤 自身CPU資源中發 生致命錯誤
等待 CPU	LED 指示	開啟	○	○	○	-	○	○	○
		錯誤	-	-	-	○	-	-	-
		執行	△	-	○	-	○	△	-
		警示	-	-	○	○	-	○	○
	旗標		正常執行	正常執行	操作CPU中發生 致命錯誤	自身CPU中發 生致命錯誤	操作CPU中發生 致命錯誤	I/O模組中發生錯誤 自身CPU資源中發 生非致命錯誤	I/O模組中發生錯誤 自身CPU資源中發 生致命錯誤
系統 DO	LED 指示	開啟	○	○	○	○	○	○	○
		錯誤	-	-	-	-	-	-	○
		系統DO	○	-	○	○	○	○	-
	聯絡輸出		開啟 (執行)	關閉 (停止)	開啟 (執行)	開啟 (執行)	開啟 (執行)	開啟 (執行)	關閉 (停止)

5-2-7 雙工系統的應用程式

(1) 連續啟動 CPU 模組

在雙工模式中，除 CPU0 之外的 CPU 都可以在沒有啟用軟故障的情況下連續啟動。請注意在系統可以啟動之前必須先加載 CPU0。

註：如果所有 CPU 模組都尚未設定，請使用載入器逐一將程式下載到它們之中。

(2) 1 對 1 雙工系統

1) 已設定操作 CPU 或等待 CPU

即使已設定了非預設等待 CPU (奇數 CPU 數)，系統也會在擁有非致命系統錯誤的情況下啟動。

2) 如果尚未設定操作 CPU 或等待 CPU

在 1 對 1 雙工系統中，即使操作 CPU 與等待 CPU 都已設定，系統依舊會使用剩餘 CPU 在有非致命錯誤的情況下啟動。

註：如果啟用了傳遞切換，系統中會發生致命錯誤。

(3) N 對 1 雙工系統

1) 尚未設定部份操作 CPU 或等待 CPU

1) 在設定了非預設等待 CPU (擁有最大被指定數的 CPU) 的情況下，系統會以非致命錯誤啟動。

2) 在未設定部份預設操作 CPU 的情況下，系統會以非致命錯誤啟動。

2) 尚未設定部份 CPU 與等待 CPU

系統會以含有非致命錯誤的剩餘 CPU 啟動。

(4) 系統定義

1) 在雙工系統中 (1 對 1 與 N 對 1)，針對系統定義中所有 CPU 指定相同的資料。系統定義資訊會由一個下載作業傳輸到所有 CPU 中。

2) 當實際組態與系統定義中的組態不同時，整個系統中都會發生系統定義錯誤。如果某些 CPU 擁有與其他 CPU 不同的資訊，在那些 CPU 中就會發生系統定義錯誤。

附錄 1 設定高效能 CPU Takt 週期

- (1) 從系統組態資料中計算 Takt 週期的近似公式 附錄 1-1
- (2) 由 Takt 週期掃描時計算效能的公式 附錄 1-2
- (3) 樣本時間計算 附錄 1-2
- (4) 預估 1 對 1 暖待機雙工系統中的 Takt 週期 附錄 1-3

附錄 1 設定高效能 CPU Takt 週期

在 SX 匯流排上，資料與 Takt 週期同步在 CPU 模組與 I/O 模組之間進行交換。CPU 模組上的應用程式會執行 1) I/O 資料輸入更新、2) 算數運算、3) 在被稱為工作（預設工作、週期工作與事件工作）的執行單元中的 I/O 資料輸出更新。這些作業是與 SX 匯流排中的資料交換同時執行的。

本附錄將會為您介紹針對每個 takt 週期來計算掃描時間的近似公式。基本上，SX 匯流排的 Takt 週期是依系統組態而定的。在需要以 Takt 週期為基礎的工作週期的系統中，Takt 週期要依據應用程式的步驟數而定。要取得精確時間，必須在實際機器上測量。

< 依 Takt 週期而定的系統組態 >

- I/O 點數
- CPU 數
- 遠端 I/O 主模組數
- 通訊模組數

(1) 從系統組態資料 Takt 週期 T (微秒) 中計算 Takt 週期的近似公式

1) 1 CPU + 直接 I/O 組態: $T = T_b$ [以直接連接輸入 / 輸出點數為基礎的基本時間 (T_b)]

直接連接 I/O 點數 (點)	0	32	128	256	512	1024	2048	3072	4096	6144	8192
基本時間 T_b (微秒)	418	504	507	510	556	695	1042	1388	1520	1711	1911

以上數目的基礎都是假設 I/O 輸入與 I/O 輸出之間的比率為 1 比 1。輸出越多，基本時間就越長，反之亦然。波動範圍約為 $\pm 20\%$ 。

- 註：1) 在存在單一 CPU、直接連接 I/O 點數是 256 或更少且沒有使用通訊模組的情況下，才有可能是 0.5 毫秒 takt 週期。
2) 使用者可以選擇的 Takt 週期有 0.5、1、2、3、...、18、19 及 20 毫秒。使用者應該選擇透過將上表中所列出的相對應數值向上捨去所取得的數值。

2) 多 CPU 組態:[CPU 數:n]

- $T = T_b + 210n$ (直接連接輸入 / 輸出點數:2048 或更少)
- $T = T_b + 200n + 190$ (直接連接輸入 / 輸出點數:多於 2048)

3) 單一 CPU + 遠端 I/O:[遠端 I/O 主模組數:m]

- $T = T_b + 250m + 430$ (直接連接輸入 / 輸出點數:2048 或更少)
- $T = T_b + 280m + 730$ (直接連接輸入 / 輸出點數:多於 2048)

4) 多 CPU + 遠端 I/O:[CPU 數:n, 遠端 I/O 主模組數:m]

- $T = T_b + 340n + 200m + 400$ (直接連接輸入 / 輸出點數:2048 或更少)
- $T = T_b + 405n + 260m + 340$ (直接連接輸入 / 輸出點數:多於 2048)

5) 當通訊模組被加到 1 CPU + 直接連接 I/O (1)中時:[通訊模組數:p]

- $T = T_b + 40p + 250$

6) 當通訊模組被加到組態 ((2)至(4)) 時:[通訊模組數:p]

- $T =$ (針對 ((2)至(4)) 計算的時間 + 85p [當不存在遠端 master 模組時])
- $T =$ (針對 ((2)至(4)) 計算的時間 + 128p [當存在遠端 master 模組時])

註: 3) 2048 點 / 1 行是針對遠端 I/O 假設的。

(2) 當由 Takt 週期掃描時，計算效能的公式

計算時間

$$= [\text{Takt週期}] - [\text{SBM 負擔時間 (200 微秒)}] - [\text{I/O 重新整理時間}] - [\text{POU 控制時間}]$$

- I/O 重新整理時間 = $(2n + m + 60)$ 微秒 [n : I/O 模組數，m : I/O 總字數]

- POU 控制時間

$$= \text{PG 控制時間} + \text{使用者 FB 控制時間} + \text{使用者 FCT 控制時間}$$

$$= (4a + 6b + 7c) \text{ 微秒}$$

[a:PG 數，b:使用者 FB 呼叫數，c:使用者 FCT 呼叫數]

計算時間

$$= [\text{Takt週期}] - (2n + m + 60) - (4a + 6b + 7c) \text{ 微秒}$$

$$\text{程式步驟數} = [\text{計算時間} / \text{單一指令執行時間} / 1024] \text{ k步}$$

$$\text{程式步驟數} = [\text{計算時間} / 20.48] \text{ k步} \text{ [當單一指令執行時間} = 20\text{ns時]}$$

$$= [\text{計算時間} / 61.44] \text{ k步} \text{ [當單一指令執行時間} = 60\text{ns時]}$$

註：1) 有關單獨指令的執行時間，請參考附錄 1，「指令處理速度表」。

2) 指令執行時間會依照存取記憶體所需的時間而變化，要處理的變數會被指定給此記憶體。因此，您必須根據指令存取的變數數目將以下存取時間增加量增加到指令執行時間。

記憶體存取時間是按照如下方式計算出來的：

- 1) I/O 記憶體與標準記憶體 (高速) :基本時間 (20ns)
- 2) 標準記憶體、保留記憶體、使用者 FB 記憶體、系統 FB 記憶體、系統記憶體:增加 40ns。
- 3) 經由處理器匯流排的另一個 CPU 存取中的記憶體：3 微秒

(3) 樣本時間計算

1) 單一 CPU

掃描時間 (Takt時間)	系統組態	POU控制時間 (PG/FB/FCT數)	程式可執行時間 (每段20 ns)
0.5毫秒	CPU... 1模組 (通訊模組停用) 直接連接I/O : 256點	68微秒 (4/4/4)	124微秒 (6k步驟)
1毫秒	CPU... 1模組 直接連接I/O : 1024點	136微秒 (8/8/8)	412微秒 (20k步驟)
2毫秒	CPU... 1模組 直接連接I/O : 2048點	480微秒 (16/32/32)	876微秒 (42k步驟)
	CPU... 1模組 遠端I/O ...1模組 : 2048點	480微秒 (16/32/32)	876微秒 (42k步驟)
	CPU... 1模組 遠端I/O ...2模組 : 4096點 直接連接I/O : 2048點	480微秒 (16/32/32)	492微秒 (24k步驟)

2) 多 CPU 組態

掃描時間 (Takt時間)	系統組態	POU控制時間 (PG/FB/FCT數)	程式可執行時間 (每段20 ns)
4毫秒	CPU... 4模組 遠端I/O ...2模組 : 4096點 直接連接I/O : 2048點	480微秒 (16/32/32)	2492微秒 (484k步驟 = 121k × 4)
	CPU... 4模組 遠端I/O ...2模組 : 4096點 直接連接I/O : 1024點 通訊模組... 2模組	480微秒 (16/32/32)	2620微秒 (508k步驟 = 127k × 4)

(4) 預估 1 對 1 暖待機雙工系統中的 Takt 週期

在 1 對 1 暖待機系統中，Takt 時間比普通多 CPU 系統中的時間長。這是因為在此系統中，等化的資料應該在操作 CPU 與等待 CPU 之間傳輸。預估的公式將陳述於下。在此公式中，大的 Takt 時間用於雙工系統 Takt 時間 1 (T_{R1}) 與雙工系統 Tact 時間 2 (T_{R2})。

雙工系統 Takt 時間 1 : T_{R1} [微秒]

T_{R1} = 通常 Takt 時間註記) + $596 \times N + 430$ (直接連接的 I/O 數:2048 或更少)

T_{R1} = 通常 Takt 時間註記) + $626 \times N + 730$ (直接連接的 I/O 數:多於 2048)

N : 雙工系統中的配對數

註：通常的 Takt 時間是透過公式(1)計算出來的。使用者可使用雙工系統中的 CPU 配對數計算出 CPU 數。

雙工系統 Takt 時間 2 : T_{R2} [微秒]

T_{R2} = (I/O 重新整理時間) + $T_{DMA} + T_{CPY} + 200$ [微秒]

● I/O 重新整理時間: $(2n + m + 60)$ [É s] [n:I/O 模組數, m:I/O 總字數]

● T_{DMA} = [(包含 CPU 的 SX 匯流排模組數) + (CPU 模組數) $\times 2$ + (遠端主模組數 $\times 55$) + (所有連接的 / 遠端 I/O 的總字數) + 512] $\times 0.5$ [微秒]

T_{CPY} = (在高速、標準、保留記憶體區域中等化的變數之字數) $\times 0.3$

+ (使用者保留變數的字數) $\times 0.35$

+ (邊緣偵測指令與計數器指令數) $\times 0.3$

+ (計時器指令數) $\times 0.45$

+ 10 [微秒]

附錄 2 設定標準 CPU Takt 週期

- (1) 根據 SX 匯流排效能計算的 Takt 週期 附錄 2-1
- (2) 以系統軟體執行時間為基礎的必要 Takt 週期 附錄 2-1

附錄 2 設定標準 CPU Takt 週期

在 SX 匯流排上，資料與 Takt 週期同步在 CPU 模組與 I/O 模組之間進行交換。對 CPU 模組中的應用程式而言，I/O 資料輸入更新、算數運算及 I/O 資料輸出更新都是在每個工作（處理單元）中執行的，與 SX 匯流排中的資料交換相同。

標準 CPU 會將一個 Takt 期間中的系統軟體分為在每個 Takt 週期中執行的程序，以及執行應用程式或預設工作已完成時所執行的程序。載入器過程是在一個過度的 Takt 週期中執行的。每個 Takt 週期的掃描時間的預估公式將陳述於下。Takt 週期依據系統組態而定。執行系統軟體也依據 Takt 週期而定。因此，可執行的 Takt 週期必須由系統軟體來設定。

< 依據 Takt 週期而定的系統組態元件 >

- I/O 點數
- 遠端 I/O 模組數：
- 同時發出載入器命令的站台數

可以從 1、2、3、...9、10 毫秒中選擇 Takt 週期。(1) 所計算的 Takt 週期是以 SX 匯流排效能 (I/O 模組數) 為基礎的，(2) 以系統軟體執行時間為基礎的必要 Takt 週期，無論哪個比較大，都是由捨去來確定的

(1) 根據 SX 匯流排效能計算的 Takt 週期

Takt 週期 T (微秒)

1) 一個 CPU + 連接的 I/O : $T = T_b$

直接連接 I/O 點數 (點)	0	32	128	256	512	1024	2048	3072	4096	6144	8192
基本時間 T_b (微秒)	418	504	507	510	556	695	1042	1388	1520	1711	1911

上面所列出的時間是在 I/O 輸入與輸出比例 = 1 : 1 的情況下計算出來的。時間會隨著所增加的輸出數而增加，而且會增加所減少的輸出數而減少。其波動範圍為 + 20% 到 -20%。

2) 一個 CPU + 遠端 I/O 主模組：[f 遠端 I/O 主模組]

- $T = T_b + 250 m + 430$ (直接連接輸入 / 輸出點數：2048 或更少)
- $T = T_b + 280 m + 730$ (直接連接輸入 / 輸出點數：多於 2048)

註：2048 點 / 1 行是針對遠端 I/O 假設的。

(2) 以系統軟體執行時間為基礎的必要 Takt 週期

系統軟體會處理 I/O 傳送、工作及載入器命令。這其中的任一處理過程都應該在一個 Takt 週期中完成。根據同時發出載入器命令的站台數，Takt 週期的建議預設時間及計算時間 (應用程式處理時間) 如下表所示。

建議的 Takt 時間 (毫秒)	1	2	3	4	5	6	7	8
同時發出載入器命令的站台數	1	4	16	27	27	27	27	27
計算時間 (微秒)	409	1209	2009	2809	3609	4409	5209	6009

註：同時發出載入器命令的站台包括載入器、POD 及與訊息有關的指令相關聯的模組 (如果模組有兩個連接埠，它們都會被計數。)

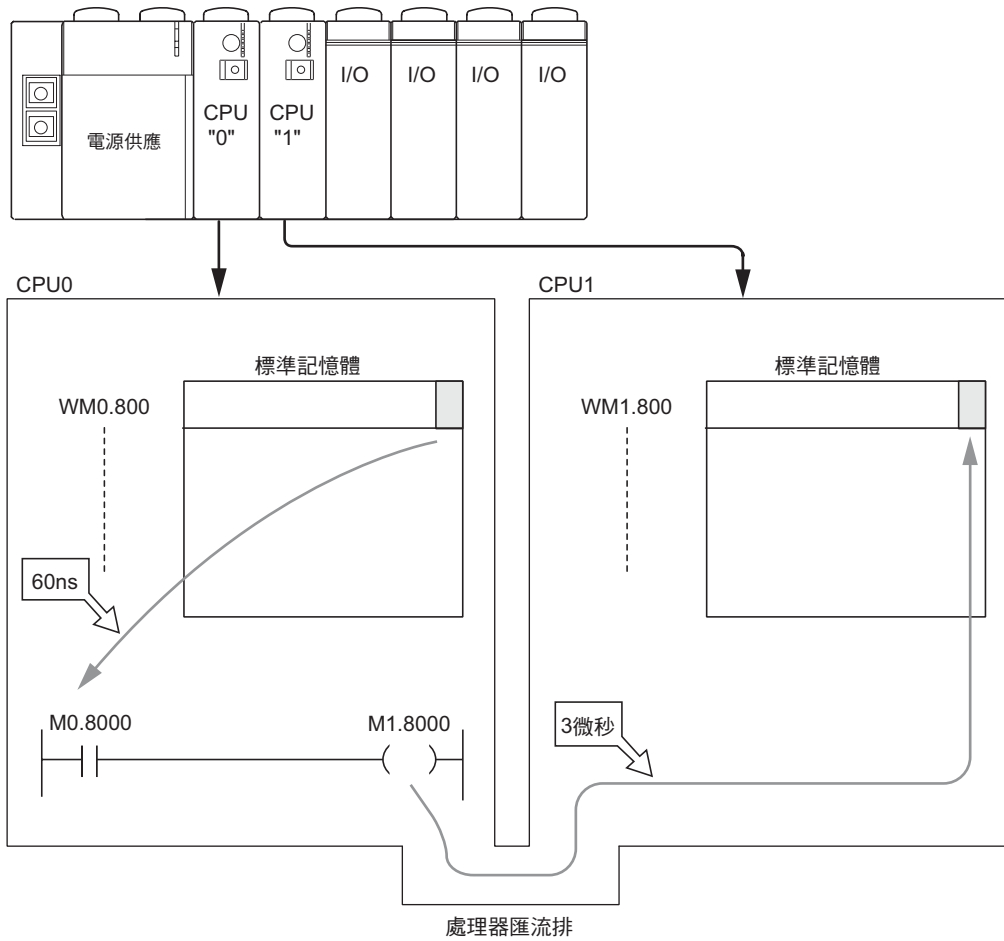
重點

- 要與 Takt 週期同步執行工作處理，應該撰寫應用程式才不會使用比下面所列時間更長的計算時間。
- 與所建議的 Takt 時間相比較，即使是在同時發出載入器命令的站台數較小的情況下，計算時間也不會變化。(載入器命令的責任會加大。)
- 與所建議的 Takt 時間相比較，當同時發出載入器命令的站台數較大時，計算時間會減少。(載入器命令的責任會減少。)

附錄 3 存取處理器匯流排

附錄 3 存取處理器匯流排

處理器匯流排可以用來讀取與寫入多 CPU 系統中各 CPU 之間的記憶體，而且可以用來讀取與寫入 CPU 與 P/PE 連結記憶體之間的記憶體。



< 從應用程式 (CPU) 中存取每個記憶體的時間 >

從 CPU 存取進或存取出每個記憶體的時間如下所示。

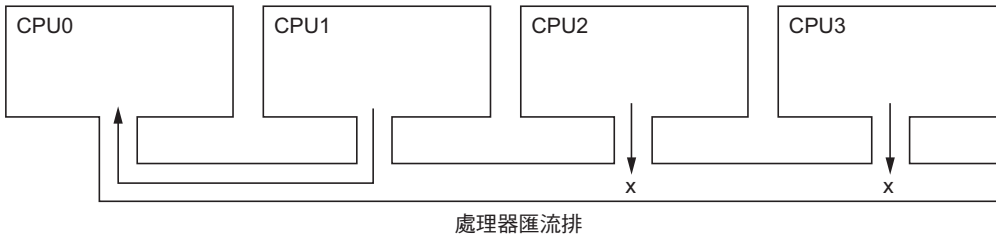
存取的記憶體	存取時間 (/字)
自身CPU中的高速記憶體 (%MW1.0至%MW1.2047)	20ns
除自身CPU中記憶體之外的記憶體	60ns
I/O區域 (X/Y)	20ns
多CPU系統中任何其他CPU中的記憶體 (註1)	3微秒 (註2)
P/PE連結記憶體	3微秒 (註2)

註： 1) 高速記憶體區域無法在任何其他 CPU 中存取。

2) 通過處理器匯流排的存取時間。

< 在處理器匯流排中讀取 / 寫入記憶體的考量因素 >

處理器匯流排無法由多個 CPU 同時存取。如果多個 CPU 同時存取處理器匯流排，CPU 會從被指定最高優先權的匯流排開始按順序使用它。擁有較低優先權的 CPU 會等待很長一段時間。任何此類延遲都會影響 CPU 處理速度。

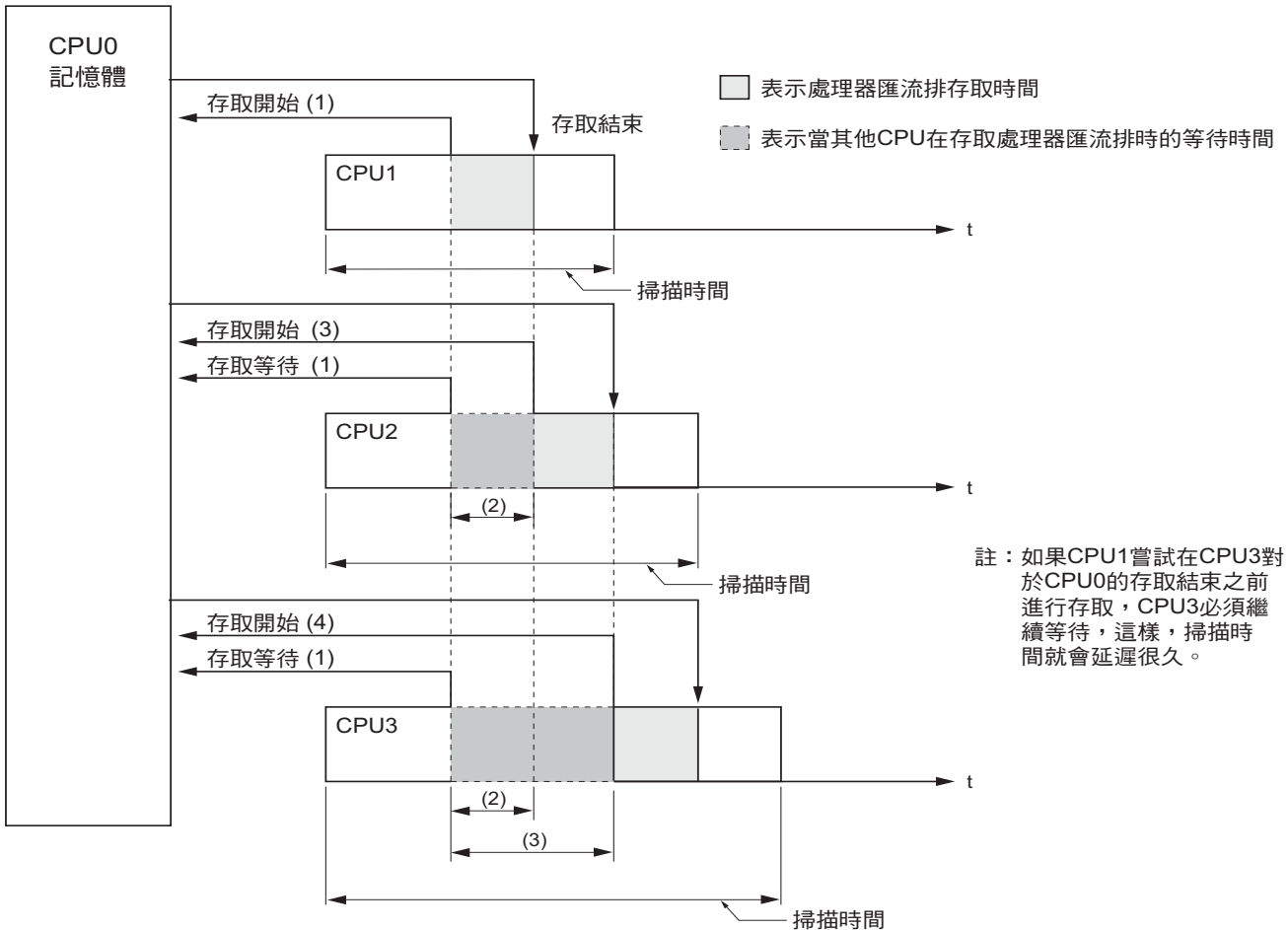


註：優先權是透過以升序排列的數字來判斷的，最高優先權會指定給最小的數。

(範例)

在下面的例子中，多 CPU 系統是由四個 CPU 所組成的。此範例說明了處理器匯流排存取與 CPU 掃描時間之間的關係。

- 1) 當 CPU1、CPU2 與 CPU3 同時存取 CPU0 記憶體時，優先權最高的第一個 CPU1 擁有對於它的初始存取權。CPU2 與 CPU3 會保持等待狀態。
- 2) CPU2 與 CPU3 的掃描時間會被 CPU1 存取時間延遲。
- 3) 當 CPU1 存取結束後，擁有第二高優先權的 CPU2 便會嘗試取得對 CPU0 記憶體的存取權。CPU3 必須再等待。
- 4) 當 CPU2 存取結束後，CPU3 會開始取得對記憶體的存取權。(註)



重點：

- 在多CPU系統與P/PE連結系統中，設計系統應使存取處理器匯流排的次數減到最少。我們推薦的存取數為128/毫秒。

附錄 4 建立使用者 FB 的程序

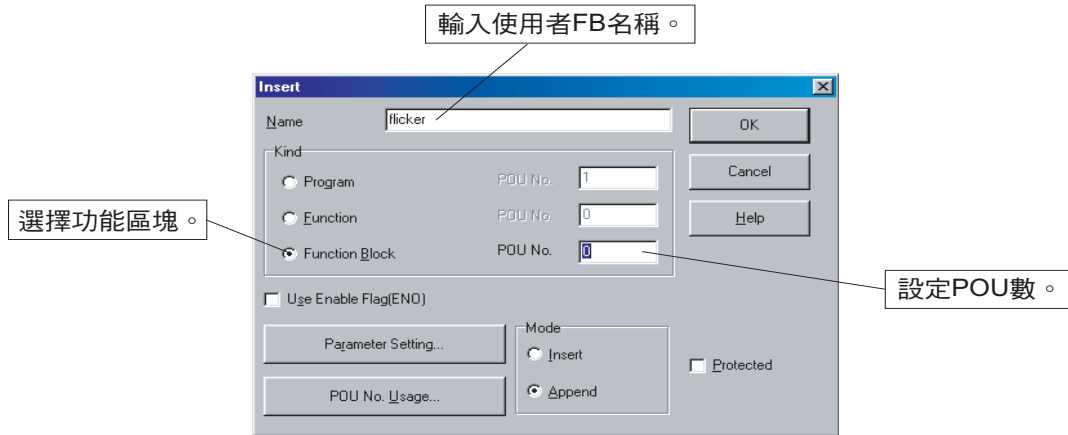
- (1) 建立程序 附錄 4-1
- (2) 讀取使用者 FB 附錄 4-3
- (3) 使用由其他專案建立的使用者 FB 附錄 4-4

附錄 4 建立使用者 FB 的程序

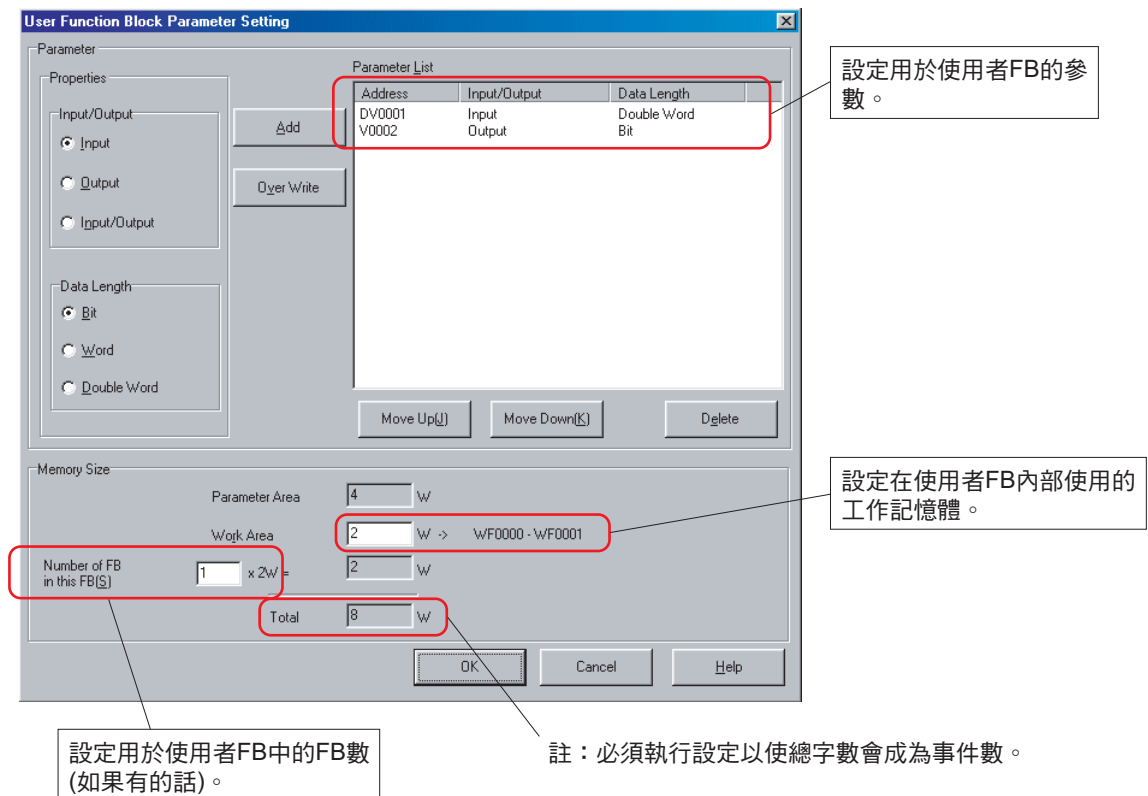
本節以建立輸出閃爍訊號的 FB 為例來為您介紹如何建立與使用使用者 FB。

(1) 建立程序

- ◇ 為使用者 FB 增加 POU。在 [程式組態] 圖示上或專案樹狀結構上已存在的 POU 上按一下按下滑鼠右鍵，並執行蹦式功能表中的 [插入] 命令，以顯示 [插入] 對話方塊。
- ◇ 在類型欄位勾點選「功能區塊」並輸入使用者 FB 名稱與 POU 數。

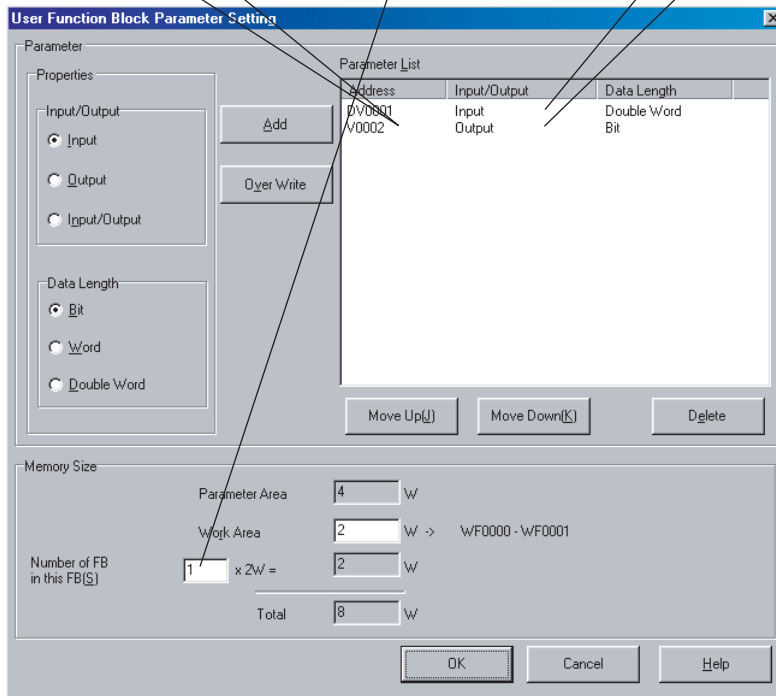
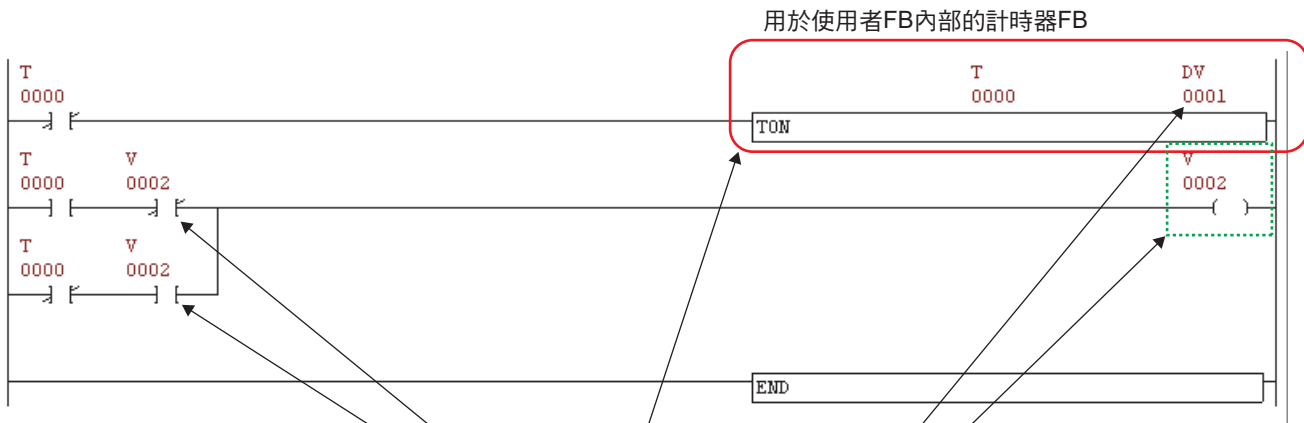


- ◇ 然後，設定使用者 FB 參數。按一下按下 [參數設定...] 按鈕來顯示 [使用者功能區塊參數設定] 對話方塊。



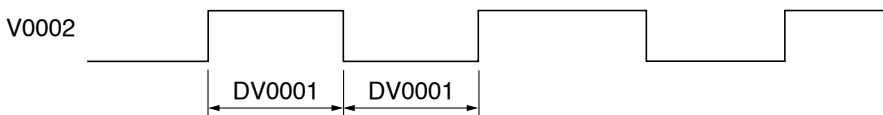
- ◇ 設定好必要的項目之後，請按一下按下 [確定] 按鈕。

◇ 建立程式。



<FB 作業>

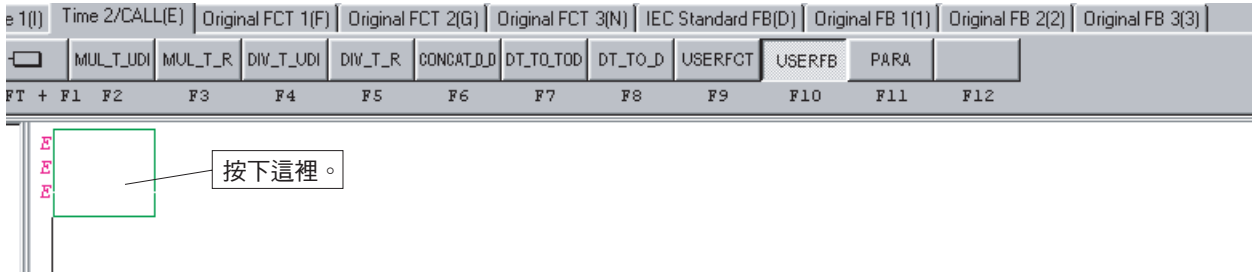
指定閃爍「開啟 / 關閉」脈衝時間來輸入參數「DV0001」，閃爍訊號會輸出至輸出參數「V0002」。



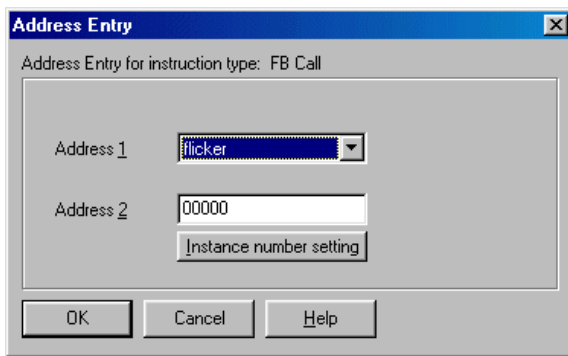
(2) 讀取使用者 FB

本段將為您說明如何使用在建立了 FB 的專案之 POU 中的使用者 FB。

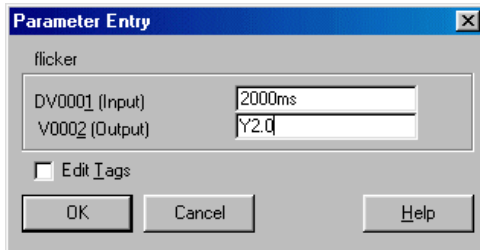
- ◇ 按一下按下 [USERFB] 按鈕之後，按一下按下使用者 FB 插入位置來顯示 [位址輸入] 對話方塊。



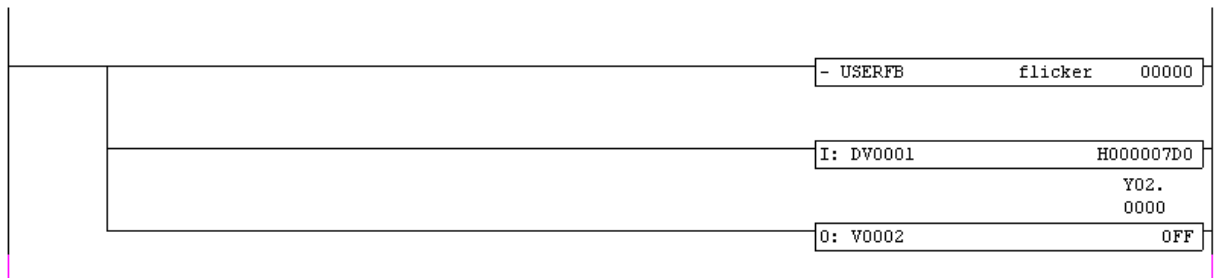
- ◇ 設定要使用的 FB 及其實例數，並按一下按下 [確定] 按鈕。



- ◇ 此時您會看到 [參數輸入] 對話方塊。輸入必要的參數，如裝置與常數後按一下按下 [確定] 按鈕。



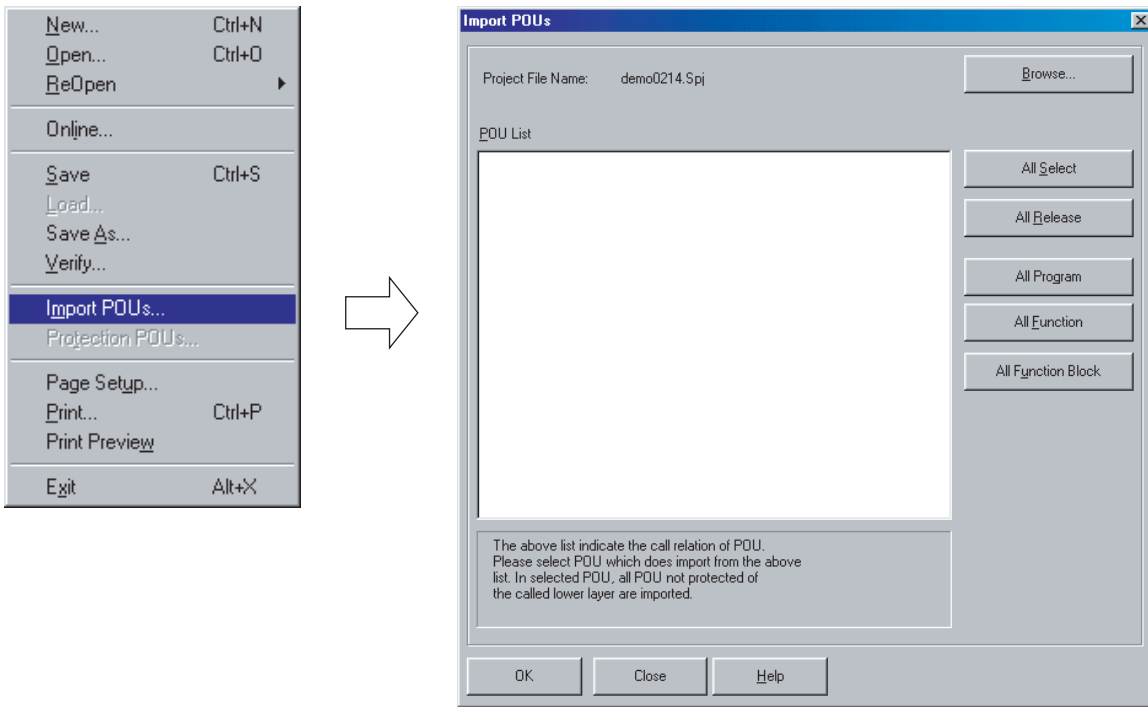
- ◇ 此時您會看到使用者 FB，如下所示。



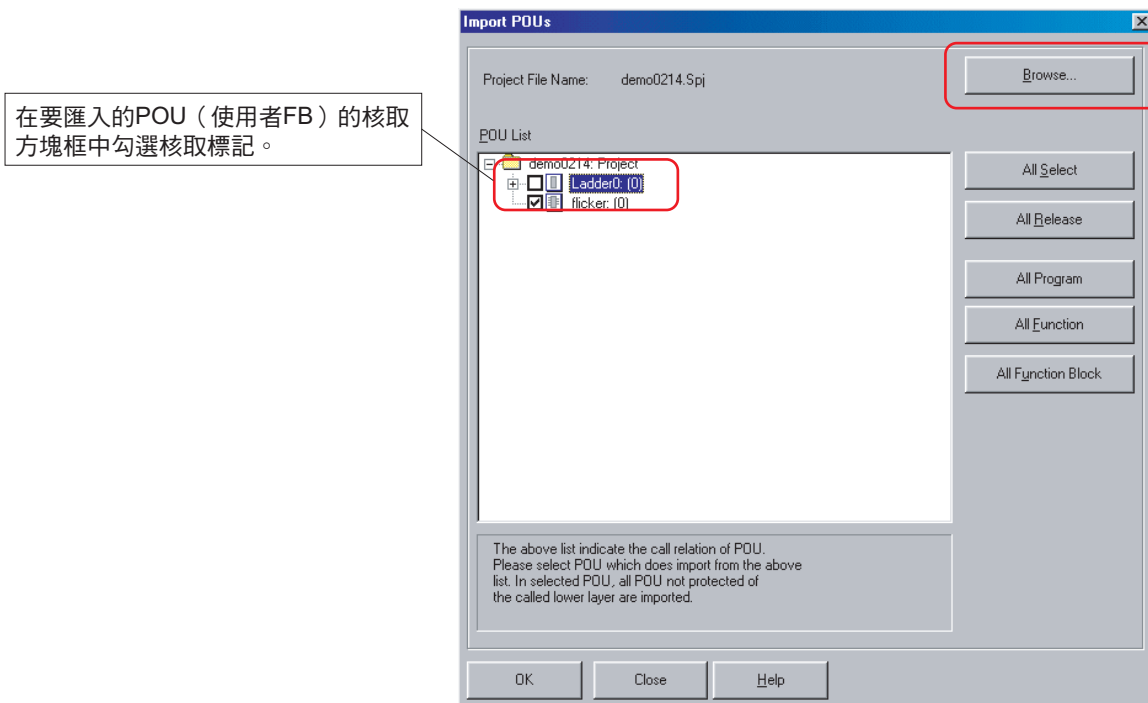
(3) 使用由其他專案建立的使用者FB

本段將為您說明如何使用在其他專案中建立的使用者FB。

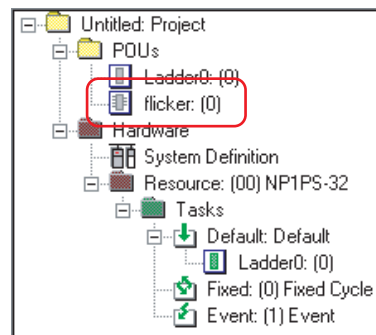
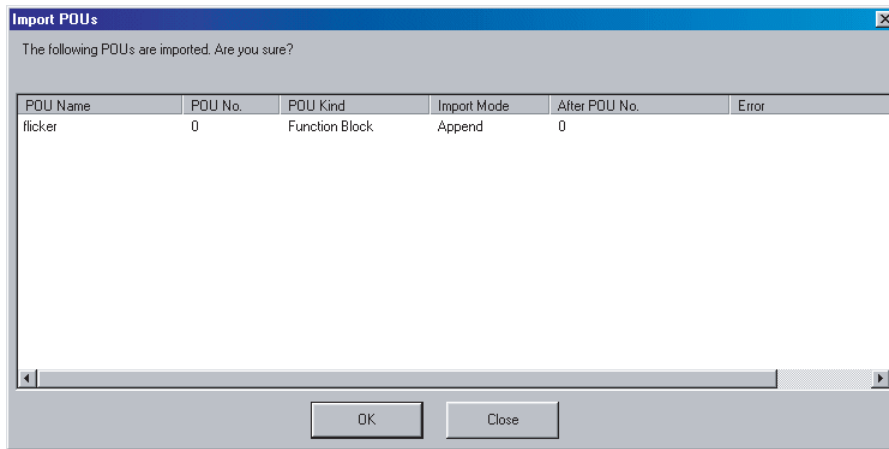
◇ 執行 [檔案] 功能表中的 [匯入POU...]，來顯示 [匯入POU] 對話方塊。



◇ 按一下按下 [瀏覽...] 按鈕並選擇要匯入使用者FB的專案。



- ◇ 選擇了要匯入的POU之後，按一下按下 [確定] 按鈕來顯示底下所示的確認對話方塊框。按一下按下 [確定] 按鈕來匯入所選擇的POU。



* 可用與上述 (2) 中說明的相同來使用匯入的使用者FB。

註：需要操作指定POU的所有POU都會被匯入。

附錄 5 設定字元字串資料

附錄 5 設定字元字串資料

字元字串是以 Fuji Electric 的原始程式碼表示，它以 Shift JIS 編碼系統為基礎。由於 Shift JIS 字串混合了 8 位元與 16 位元程式碼，因此，此編碼系統將 8 位元程式碼延伸為了 16 位元程式碼，以使單一字元的長度能夠固定為 16 位元。字串的長度可以變化，最長可為 64 字元。

NULL 程式碼 ('X'0000') 會自動附加到每個字串的結尾。因此，當宣告字元字串時，會保留與字元字串長度相等的記憶體空間再加上一個字元，也就是 65 字。

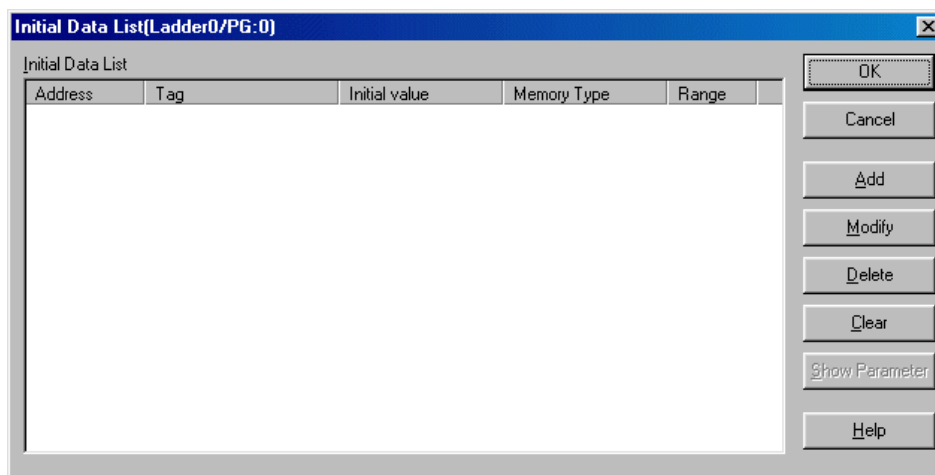
例如：字元字串「ABCDEF」而言的Shift JIS程式碼與字串資料型別會表示為：

字元字串		Shift JIS程式碼	字元字串資料型別
A	→	41	00 41
B	→	42	00 42
C	→	43	00 43
D	→	44	00 44
E	→	45	00 45
F	→	46	00 46
			00 00 →NULL程式碼

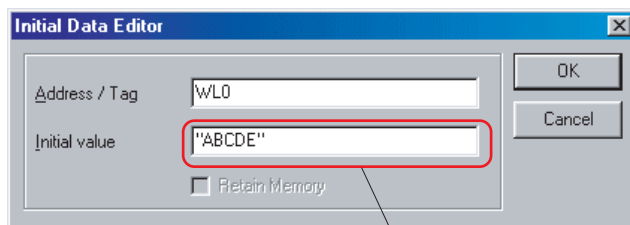
< 使用載入器設定字元字串資料 >

如果將要在應用程式中使用字元字串資料，在裝置中將資料設定為初始數值是有幫助的。您可以使用以下程序來設定它。

- ◇ 執行 [PLC功能] 功能表中的 [初始資料清單...] 來顯示 [初始資料清單] 對話方塊框。

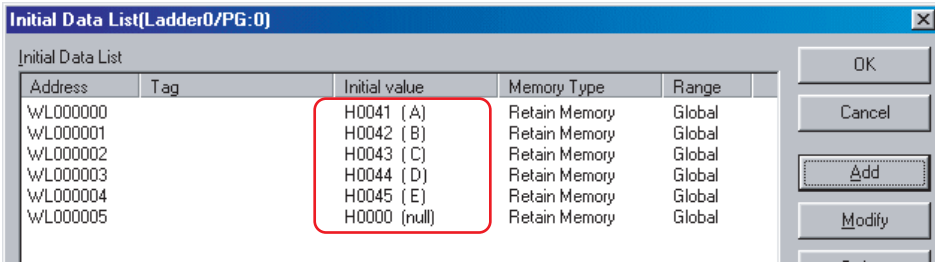


- ◇ 按一下按下 [新增] 按鈕來顯示 [初始資料編輯器] 對話方塊框。在此對話方塊框中，輸入要在其中設定初始數值的裝置（開頭位址）及字元字串初始數值。

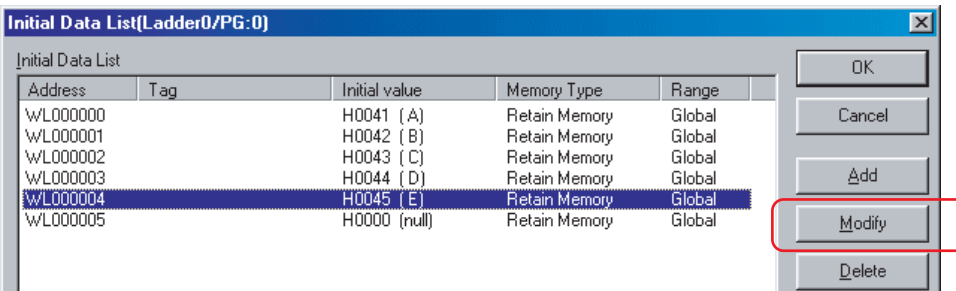


當設定字元字串時，要用一對1位元組的雙引號（"）將其括起來。

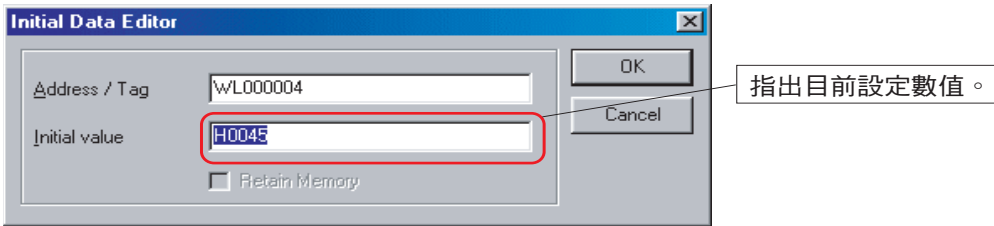
- ◇ 在設定完裝置與字元字串之後，按一下按下 [確定] 按鈕來將設定字元字串寫入到 [初始資料清單] 中。



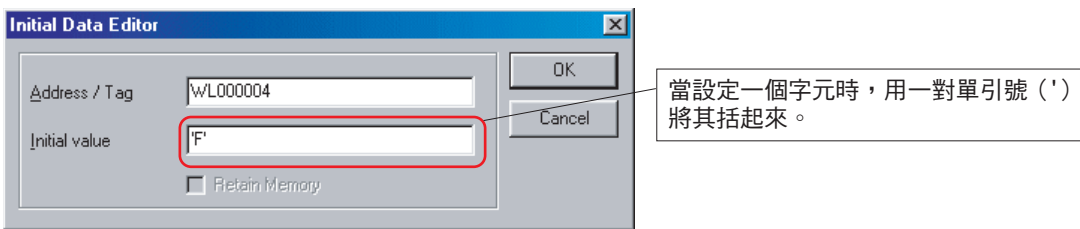
- ◇ 要更改設定字元字串中的字元，請選擇該字元並按一下按下 [修改] 按鈕。



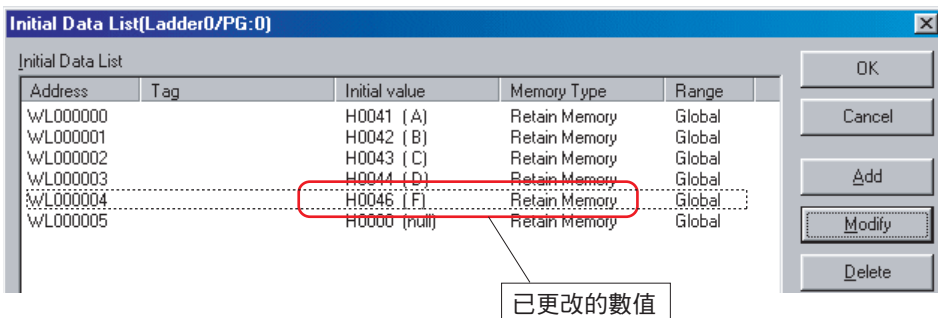
- ◇ 此時您會看到 [初始資料編輯器] 對話方塊框。



- ◇ 輸入括有一對 1 位元組單引號 (') 的字元。



- ◇ 設定完成之後，請按一下按下 [確定] 按鈕。變更會顯示在清單中。



Fuji Electric FA Components & Systems Co., Ltd.

Gate City Ohsaki, East Tower, 11-2, Osaki 1-chome, Shinagawa-ku, Tokyo, 141-0032, Japan

電話: +81-3-5435-7135 ~ 8

傳真: +81-3-5435-7456 ~ 9

網址 <http://www.fujielectric.co.jp/fcs/eng/>