Cat. No. W447-TW5-01

SYSMAC

CX-Programmer

6.1 版

WS02-CXPC1-E-V61

CS1-H、CJ1-H、CJ1M、CP1H CPU 模組

NSJ、FQM1

功能區塊
中文版操作手冊

OMRON

CX-Programmer

6.1 版
WS02-CXPC1-E-V61

CS1-H、CJ1-H、CJ1M、CP1H CPU 模組
NSJ、FQM1

Function blocks 中文版操作手冊

注意：

OMRON 產品須由合格操作人員依照正常操作步驟來使用，而且僅能用於本手冊所說明之用途。

下列符號用語是用來分類及說明本手冊中的注意事項，用戶必須注意這些資訊。如忽略這些注意事項可能會導致人員受傷或資產受損。

⚠ **危險** 表示即將發生危險，如未避免，將導致死亡或造成嚴重傷害。此外，也可能造成嚴重的財物損失。

⚠ **警告** 表示有潛在的危險，如未避免，可能會導致死亡或造成嚴重傷害。此外，也可能造成嚴重的財物損失。

⚠ **注意** 表示有潛在的危險，如未避免，可能會導致輕微或中度傷害，或造成財物損失。

OMRON 產品之參考說明

所有 OMRON 產品在本手冊中均以大寫字母表示。當以“模組(Unit)”表示 OMRON 產品時，也會以大寫字母表示，不論是否列出產品的正式名稱。

出現在某些顯示以及某些 OMRON 產品上的縮寫字母"Ch"通常表示"word"，而在文件中則以縮寫"Wd"來表示。

縮寫字母"PLC"表示可程式控制器。

閱讀輔助

下列位於手冊左欄的標題可以幫助您找到不同類型的資訊。

備註 表示可以讓產品方便而有效運作的特殊重要資訊。

1,2,3… 1. 用來列舉說明，例如程序步驟、檢查表等等。

目錄

關於這本手冊：

本手冊說明搭配 CP1H CPU 模組及模組版本 3.0 以上的 CS1-H、CJ1-H、及 CJ1M CPU 模組使用的 CX-Programmer 6.1 版的 function blocks (功能區塊)及相關功能，包括下一頁所述的章節。CX-Programmer 6.1 版是一個可以使用個人電腦來做為功能區塊設定裝置的軟體，並且只能用於支援功能區塊的 SYSMAC CS 系列及 CJ 系列 CPU 模組。

CX-Programmer 6.1 版 function blocks (功能區塊)的功能已經強化。本手冊將指說明與功能區塊有關的 CX-Programmer 6.1 版的操作。關於與功能區塊無關的操作，請參考 *CX-Programmer 操作手冊*(隨附，Cat. No. W437)。本手冊也只提供與 CS1-H、CJ1-H、CJ1M、及 CP1H CPU 模組的功能區塊有關的資訊。其他的資訊，請參考 CS/CJ/CP 系列的手冊。

在嘗試安裝或操作 CX-Programmer 6.1 版或 CS1-H、CJ1-H、CJ1M、或 CP1H CPU 模組之前，請詳細閱讀本手冊及相關手冊並確定您已瞭解手冊所提供的資訊。請確實閱讀下節所提供的注意事項。

與 CX-Programmer 6.1 版有關的手冊

名稱	Cat. No.	目錄
SYSMAC WS02-CXPC1-E-V60 CX-Programmer 6.1 版操作手冊功能區塊 (CS1G-CPU□□H, CS1H-CPU□□H, CJ1G-CPU□□H, CJ1H-CPU□□H, CJ1M-CPU□□, CP1H-X□□□□-□, CP1H-XA□□□□-□, CP1H-Y□□□□-□ CPU 模組)	W447 (本手冊)	說明 CX-Programmer 6.1 版及 CP 系列 CPU 模組或模組版本 3.0 以上的 CS/CJ 系列 CPU 模組以功能區塊為基礎的獨特功能。與 CX-Programmer 相同的功能則在 W446 (隨附)中說明。
SYSMAC WS02-CXPC1-E-V60 CX-Programmer 操作手冊	W446	提供有關如何使用 CX-Programmer 除功能區塊以外的所有功能的資訊。
CX-Net 操作手冊	W362	有關網路設定如設定資料連結、路由表及模組設定的資訊。
SYSMAC CXONE-AL□□C-E CX-Integrator 操作手冊	W445	說明 CX-Integrator 的操作程序。

與 CS1-H、CJ1-H、CJ1M CPU 模組相關手冊

名稱	Cat. No.	目錄
SYSMAC CS 系列 CS1G/H-CPU□□-EV1, CS1G/H-CPU□□H 可程式控制器 操作手冊	W339	提供 CS 系列 PLC 的簡介並說明相關的設計、安裝、維護及其他基本操作。 包括下列資訊： 概述及特性 系統組態 安裝及配線 I/O 記憶體配置 故障排除 請搭配 W394 一起使用本手冊。
SYSMAC CJ 系列 CJ1G-CPU□□, CJ1G/H-CPU□□H, CJ1G-CPU□□P, CJ1M-CPU□□ 可程式控制器 操作手冊	W393	提供 CJ 系列 PLC 的簡介並說明相關的設計、安裝、維護及其他基本操作。 包括下列資訊： 概述及特性 系統組態 安裝及配線 I/O 記憶體配置 故障排除 請搭配 W394 一起使用本手冊。
SYSMAC CS/CJ 系列 CS1G/H-CPU□□-EV1, CS1G/H-CPU□□H, CJ1G-CPU□□, CJ1G/H-CPU□□H, CJ1G-CPU□□P, CJ1M-CPU□□ 可程式控制器 程式編輯手冊	W394	說明程式編輯及使用 CS/CJ 系列 PLC 功能的其他方法。 包括下列資訊： 程式編輯 Task (工件) 檔案記憶體 其他功能 請搭配 W339 或 W393 一起使用本手冊。
SYSMAC CS/CJ 系列 CS1G/H-CPU□□-EV1, CS1G/H-CPU□□H, CJ1G-CPU□□, CJ1G/H-CPU□□H, CJ1G-CPU□□P, CJ1M-CPU□□ 可程式控制器 指令參考手冊	W340	說明 CS/CJ 系列 PLC 所支援的階梯圖程式編輯指令。 在編寫程式時，請搭配操作手冊(CS1：W339 或 CJ1：W393)及*程式編輯手冊*(W394)一起使用本手冊。
SYSMAC CS/CJ 系列 CS1G/H-CPU□□-EV1, CS1G/H-CPU□□H, CS1W-SCB21-V1/41-V1, CS1W-SCU21/41, CJ1G-CPU□□, CJ1G/H-CPU□□H, CJ1G-CPU□□P, CJ1M-CPU□□, CJ1W-SCU21-V1/41-V1 通訊指令 參考手冊	W342	說明可以用來進行 CS/CJ 系列 CPU 模組編址的通訊指令。 包括下列資訊： C 系列(Host Link)指令 FINS 命令 備註：本手冊說明可以經由 CPU 模組上的序列通訊埠、序列通訊模組/卡上的通訊埠、或者任何其他通訊模組上的通訊埠傳送給 CPU 模組而與通訊路徑無關的指令。

NSJ 系列 NSJ 控制器手冊

關於本手冊未提供的 NSJ 系列 NSJ 控制器規格及使用方法，請參閱下列手冊。

Cat. No.	型號	名稱	說明
W452	NSJ5-TQ□□(B)-G5D NSJ5-SQ□□(B)-G5D NSJ8-TV□□(B)-G5D NSJ10-TV□□(B)-G5D NSJ12-TS□□(B)-G5D	NSJ 系列 操作手冊	提供有關 NSJ 系列 NSJ 控制器的下列資訊： 概述及特性 指定系統組態 安裝及配線 I/O 記憶體配置 故障排除及維護 本手冊請搭配下列手冊使用：SYSMAC CS 系列操作手冊(W339)、SYSMAC CJ 系列操作手冊(W393)、SYSMAC CS/CJ 系列程式編輯手冊(W394)、及 NS-V1/-V2 系列設定手冊(V083)

FQM1 系列手冊(模組版本 3.0 以上)

本手冊未提供的 FQM1 系列(模組版本 3.0 (FQM1-CM002/MMP22/MMA22))規格及使用方法，請參考下列手冊。

Cat. No.	型號	名稱	說明
O012	FQM1-CM002 FQM1-MMP22 FQM1-MMA22	FQM1 系列 操作手冊	提供有關 FQM1 系列模組(模組版本 3.0)的下列資訊： 概述及特性 指定系統組態 安裝及配線 I/O 記憶體配置 故障排除及維護
O013	FQM1-CM002 FQM1-MMP22 FQM1-MMA22	FQM1 系列 指令 參考手冊	個別說明關於 FQM1 程式編輯的指令。 在編寫程式時請搭配 *FQM1 系列操作手冊*(O012)使用本手冊。

關於從 CX-One FA 整合工具套件安裝 CX-Programmer 的程序的詳細資訊，請參考隨 CX-One 提供的 *CX-One 設定手冊*。

Cat. No.	型號	手冊名稱	目錄
W444	CXONE-AL□□C-E	CX-One 設定手冊	CX-One FA 整合工具套件的安裝與概要說明。

內容概述

注意事項 提供有關 CX-Programmer 6.1 版使用上的一般注意事項。

第 1 章 介紹 CX-Programmer 功能區塊的功能並解說不包含於非功能區塊版本的 CX-Programmer 中的特性。

第 2 章 提供在使用功能區塊時可供參考的規格，包括有關功能區塊、實例、及相容 PLC 的規格、以及使用注意事項及指導說明。

第 3 章 說明在 CX-Programmer 上建立功能區塊的程序。

附錄 提供有關資料類型、結構文字規格、及外部變數的資訊。

⚠ 警告 若不先閱讀及瞭解本手冊所提供的資訊，可能會導致人員傷亡、產品受損或故障。進行任何步驟或操作之前，請先詳細閱讀本手冊內的每個區段，同時務必瞭解內容與及相關區段所提供的資訊。

請閱讀並瞭解本手冊的內容

請在使用產品前先閱讀及瞭解本手冊的內容。如有任何問題或意見，請與您的 OMRON 業務人員聯繫。

保固與責任範圍

保固

OMRON 為其產品提供售出後一年(或另行指定的期間)內，材質與製品上的無瑕疵擔保。

OMRON 不以明示或暗示的方法來保證或表示其產品無侵權、適合銷售或適合特殊用途。買主或用戶都必須瞭解，買主或用戶需自行認定該產品可符合其用途需求。OMRON 皆不負其他明示或暗示的保證責任。

責任範圍

與本產品有關之特殊、間接或衍生損害、盈虧或商業損失，無論這些索賠主張係基於合約、保固、疏失或絕對法律責任，OMRON 概不負責。

無論在任何情況下，OMRON 對產品所負之責任不得超過產品的單價。

無論在任何情況下，OMRON 對產品保固、維修或其他產品相關的索賠概不負責，除非經 OMRON 分析證實本產品確實受正確操作、存放、安裝及保養，而且未遭受污染、濫用、誤用或不當改造或維修。

應用的考量因素

適用性

OMRON 對於客戶在應用或使用產品時是否遵循產品組合適用的標準、法律或法規，概不負責。

如客戶要求，OMRON 將提供適用的協力廠商認證文件，註明本產品所適用的額定值與限制。這項資訊本身並不足以完全認定該產品適合與終端產品、機械、系統或其他應用或用途搭配使用。

以下是一些必須特別注意的應用範例。此處並未詳細列出本品的所有可能用途，也非暗示所列出的用途適合這些產品：

- 戶外使用、涉及化學污染或電子干擾的使用，或本手冊未載明的環境或用途。
- 核能控制系統、燃燒系統、鐵路系統、飛航系統、醫療設備、遊戲機器、車輛、安全裝置，以及受個別產業與政府規範的安裝。
- 可能危害生命或財產安全的系統、機器及設備。

請瞭解並遵守本產品在使用上的所有禁止規定。

如未確保系統整體的設計目的可應付危險、且 OMRON 產品的額定值與安裝方式皆符合設備或系統整體的使用目的時，若應用涉及嚴重危害生命或財產，切勿使用本產品。

可程式產品

OMRON 對用戶之可程式產品的程式設計或後續的任何結果，概不負責。

免責聲明

規格更換

產品規格與附件隨時都可能因改良或其他原因而更換。

依照我們的慣例，當已發行之額定值或特性更換，或架構大幅變動時，就會更改型號。然而，有些產品規格可能不經通知而更換，恕不另行通知。若有疑慮，如您提出要求，特別型號的產品可送修或建立您的應用程式的重要規格。您可以隨時洽詢您的 OMRON 代表，確認您所購買之產品的實際規格。

尺寸與重量

即使有列出容許誤差，尺寸與重量皆為額定值，不得做為製造之用途。

效能資料

本手冊所載明的效能資料，其用意在協助使用者判斷產品的適用性，而非提供產品保證。該資料可能包含產品在 OMRON 測試環境下所得到的測試結果，用戶必須考量實際的應用需求。實際的效能表現會受到 OMRON 保固與責任範圍的限制。

錯誤與疏漏

本手冊內的資訊已經經過仔細的檢核，以確保其精確性；然而，若有筆誤、印刷或校對錯誤或遺漏，OMRON 恕不負責。

注意事項

本章提供有關使用 CX-Programmer 6.0 版及可程式邏輯控制器的一般注意事項。

本章所包含的資訊對於安全且可靠的應用 CX-Programmer 6.0 版及可程式控制器非常重要。在嘗試設定或操作 CX-Programmer 6.0 版及可程式控制器之前，您必須詳細閱讀本章並瞭解所含的資訊。

1 目標讀者

本手冊針對下列具有電氣系統知識之人員(電氣工程師或具相同資格的人員)。

- 負責安裝 FA 系統之人員。
- 負責設計 FA 系統之人員。
- 負責管理 FA 系統及設備之人員。

2 一般注意事項

使用者必須按照操作手冊內所說明的性能規格來操作本產品。

將本產品用於本手冊未載明之環境或將產品用於核能控制系統、鐵路系統、飛航系統、車輛、燃燒系統、醫療設備、遊戲機器、安全裝置，或其它使用不當可能會嚴重影響生命和財產的系統、機器之前，請先與 OMRON 業務人員洽詢。

請確定本產品的額定值與效能特性符合系統的要求，並務必為系統、機械及裝置提供雙重的安全機制。

本手冊提供程式設計和操作使用之資訊。在開始使用本產品之前，務必先閱讀本手冊，並將手冊放置於隨手可得之處，供操作時參閱。

⚠ **警告** 將 PLC 和所有 PLC 模組應用於特定用途且在特定環境時，特別是會直接或間接影響生命安全的應用環境時，請務必注意。在將 PLC 系統應用於上述情況之前，必須先與 OMRON 業務人員洽詢。

3 安全注意事項

⚠ **警告** 將 I/O 記憶體區域的狀態從 CX-Programmer 6.0 版傳送到 CPU 模組之前，請確定是否有足夠的安全性。不論 CPU 模組的操作模式為何，連接到輸出模組的裝置都有發生故障的可能。

- 使用 *PLC Memory (PLC 記憶體)*視窗，從 CX-Programmer 傳送到 CPU 模組中的實際 I/O (CIO 區域)。
- 使用 *Memory Card (記憶卡)*視窗，從檔案記憶體傳送到 CPU 模組中的實際 I/O (CIO 區域)。

⚠ **注意** 變數必須以 AT 設定(或外部變數)來指定，或者變數的大小必須與在指定指令運算元中的多個字組的第一或最後位址時要由該指令處理的資料大小相同。

1. 如果一個非陣列變數有不同的資料大小且沒有指定 AT 設定，則 CX-Programmer 會在匯集時輸出一個錯誤。
2. 陣列變數規格

- 當要由指令運算元處理的大小固定時:陣列元件數必須與要由該指令處理的元件數相同。否則，CX-Programmer 會在匯集時輸出一個錯誤。
- 當要由指令運算元處理的大小不固定時:陣列元件數必須大於或等於其他運算元中所指定的大小。
  - 如果其他運算元指定一個大小是一個常數，則 CX-Programmer 6.0 版會在匯集時輸出一個錯誤。
  - 如果其他運算元指定一個大小是一個變數，則 CX-Programmer 6.0 版不會在匯集時一個錯誤，即使陣列變數的大小與其他運算元(變數)所指定的不同。不過，它會顯示一個警告訊息。尤其是，如果陣列元件數小於其他運算元所指定的大小(例如指令運算元的大小是 16，而實際變數表中所登錄的元件數是 10)，則指令會針對超出元件數的部份執行讀取/寫入處理。例如，讀取/寫入處理會針對實際變數表中所登錄的元件數之後的 6 個字組執行。如果這些字組用於其他指令(包括內部變數配置)，則會發生非預期的操作，這可能會導致嚴重的意外事故。
    在開始操作 PLC 之前，如果運算元中所指定的變數大小小於運算元定義中的大小，請檢查系統是否不會受到不良的影響。

⚠ **注意** 將程式傳送至另一個節點或更換 I/O 記憶體區域的內容之前，必須先確認目的節點的安全性。如在進行上述兩項作業之前未確認安全性，可能會有人員受傷的危險。

⚠ **注意** 只有在確認延長循環時間不會產生不利影響之後，才可以執行線上編輯。否則輸入訊號可能無法讀取。

⚠ **注意** 在 Ladder Section 視窗監控電流量及顯示數值狀態之前，或是在 Watch 視窗監控當前數值時，請先確認有足夠的安全性。如果不慎按到快速鍵而執行強制設定/重置或設定/重置作業，不論 CPU 模組的操作模式為何，連接到輸出模組的裝置都可能會發生故障。

4 使用注意事項

使用 CX-Programmer 時，請遵循下列注意事項。

- 使用者程式不能上傳到 CX-Programmer。
- 在啟動 CX-Programmer 之前，請遵守下列注意事項。
  - 關閉所有和 CX-Programmer 沒有直接關聯的應用程式。特別是螢幕保護程式、病毒掃描程式、電子郵件或其他通訊軟體等，以及行事曆或其他會定期或自動啟動的應用程式。
  - 取消在任何網路上與其他電腦共用硬碟、印表機或其他裝置的共享功能。

  - 在某些筆記型電腦上，RS-232C 連接埠會依預設值配置給數據機或紅外線裝置。請依照您的電腦的使用說明，將 RS-232C 連接埠啟用為一般的序列埠。
  - 在某些筆記型電腦上，預設的省電設定並不會供電給 RS-232C 連接埠。Windows 及特定的電腦公用程式與 BIOS 都可能設定省電功能。請依照您的電腦的使用說明，關閉所有省電設定。
- 請勿關閉 PLC 的電源供應器，或在 CX-Programmer 與 PLC 連線時拔掉連接線。電腦可能會因此而故障。
- 務必確認系統不會產生不利影響後，再進行下列任一程序。否則可能會導致不可預期的操作錯誤。
  - 更換 PLC 的操作模式。
  - 強制設定/強制重置記憶體中的任一個位元。
  - 更換任一 word 的當前數值或記憶體中的任一設定數值。
- 實際在模組上執行程式之前，請先檢查用戶程式能確實執行。如未檢查程式，可能會導致不可預期的操作錯誤。
- 在執行線上編輯時，使用者程式及 CS1-H、CJ1-H、CJ1M、及 CP1H CPU 模組中的參數區域資料會備份在內建的快閃記憶體中。備份時，CPU 模組前面的 BKUP 指示燈會亮起。當 BKUP 指示燈亮起時，請勿關閉 CPU 模組的電源，否則資料將無法成功備份。要顯示寫入到 CX-Programmer 上的快閃記憶體中的狀態，請選取 PLC 屬性中的 *Display dialog to show PLC Memory Backup Status (顯示對話來顯示 PLC 記憶體備份狀態)*然後從檢視主選單中選取 ***Windows - PLC Memory Backup Status (PLC 記憶體備份狀態)***。
- 包含功能區塊(階梯圖程式編輯語言或結構化文字(ST)語言)的程式可以透過和不包含功能區塊的標準程式相同的方式來下載或上傳。不過，包含功能區塊的工作不能下載到工作模組中(可以上傳)。
- 如果將一個在 CX-Programmer 6.0 版以上建立的包含有功能區塊的使用者程式下載到一個並不支援功能區塊的 CPU 模組上(模組版本 2.0 以下的 CS/CJ 系列 CPU 模組)，則所有實例都會被視為不合規定的指令且不能進行編輯或執行使用者程式。
- 如果輸入變數資料不是 Boolean 格式，且數值(例如 20)只輸入到參數中，則會轉送 CIO 區域位址中的實際值(例如 0020)。因此，請確定在輸入數值之前加上一個&、#、或+、-的字首。
- 位址可以設定在輸入參數中，但位址本身不能做為一個輸入變數轉送。(即使一個位址被設定為一個輸入參數，轉送給功能區塊的值也會是輸入變數中的資料大小。)因此，當運算元指定有多重字組的第一或最後元件時，輸入變數不能用來做為功能區塊中的指令的運算元。請使用有 AT 設定的內部變數。或者，請在一個內部陣列變數中指定第一或最後元件。

- 在執行運算之前，數值會以批次方式從輸入參數轉送給輸入變數(與執行的運算中的指令並不同時)。因此，在執行功能區塊運算中的一個指令時若要將數值從一個參數轉送給一個輸入變數，請使用一個內部變數或外部變數來取代一個輸入變數。同樣的情形也適用於從輸出變數將數值寫入到參數時。
- 在下列情況下請務必使用有 AT 設定的內部變數。
  - 分配給基本 I/O 模組、特殊 I/O 模組及 CPU 匯流排模組的位址不能登錄給全域符號，且這些變數不能指定為外部變數(例如全域變數的資料集可能不可靠)。
  - 在將預先登錄給外部變數以外的附屬區域位元登錄給全域符號且這些變數每有指定為外部變數時，請使用內部變數。
  - 在針對網路上的其他節點指定 PLC 位址時，請使用內部變數：例如，遠端節點的第一目的字組使用 SEND(090)而遠端節點的第一來源字組使用 RECV(098)。
  - 當透過一個指令運算元指定多重字組的第一及最後字組且該運算元不能指定為一個內部陣列變數(例如不能指定陣列元件數)時，請使用內部變數。

章節 1
簡介

本章將介紹 CX-Programmer 功能區塊的功能並解說不包含於非功能區塊版本的 CX-Programmer 中的特性。

1-1 功能區塊介紹(Introducting the Function Blocks)

1-1-1 概覽及功能

CX-Programmer 5.0 版(及更新版本)是一種可使用 IEC 61131-3 標準功能區塊的編程裝置。CX-Programmer 5.0 版功能區塊由 CP1H CPU 模組、NSJ 系列 NSJ 控制器、FQM1 Flexible Motion 控制器，以及 3.0 版或更新版的 CS/CJ 系列 CPU 模組所支援，並具有下列功能：

- 使用者定義程序可透過功能區塊轉換為區塊格式。
- 功能區塊演算可寫入階梯圖編程語言或結構化文字(ST)語言。(見備註)
  - 使用階梯圖編程時，非 CX-Programmer 4.0 版或更早版本所建立的階梯圖程式，皆可利用複製及貼上的方式重複使用。
  - 使用 ST 語言時，原本利用階梯圖編程而難以輸入的演算式程序，將更容易編寫。

  備註 ST 語言為工業控制(主要為可程式邏輯控制器)的進階語言，定義於 IEC 61131-3。CX-Programmer 所支援的的 ST 語言符合 IEC 61131-3 標準。
- 由於變數不必利用文字宣告，因此功能區塊的建立將更為容易。這些變數登錄於變數表當中;登錄於變數表之後，被登錄的變數即可輸入到階梯圖程式。
- 單一功能區塊可以轉換為單一檔案的程式庫功能檔，因此有助於重複使用功能區塊，以進行標準程序。
- 可以針對單一功能區塊執行程式檢查，以便於確認功能區塊作為程式庫功能檔的可靠性。
- 包括功能區塊(階梯圖編程語言或結構化文字(ST)語言)的程式，可比照不含功能區塊的標準程式，以相同方式進行下載或上傳。但是，包括功能區塊的工作則無法在工作單元中下載(但可能可以上傳)。
- 支援一維陣列變數，可更加容易地進行多個應用程式的資料處理。

  備註 IEC6131 標準係為國際電工委員會(IEC)所制定的國際性可程式邏輯控制器標準，此標準可分為七部份，PLC 編程相關規格定義於*第三部份：文字化程式語言(IEC 61131-3)*。
- 您可從另一功能區塊(階梯圖編程語言或結構化文字(ST)語言)呼叫功能區塊(階梯圖編程語言或結構化文字(ST)語言)。功能區塊的巢狀結構可達 8 層，階梯圖/ST 語言的功能區塊還可以自由組合。

1-1-2 功能區塊規格

關於下表中所沒有列出的規格，請參考 *CX-Programmer 6.1 版操作手冊* (W437)。

項目		規格
型號號碼		WS02-CXPC1-E-V5□
安裝光碟		CD-ROM
相容的 CPU 模組		相容於模組版本 3.0 以上的 CS/CJ 系列 CS1-H、CJ1-H、及 CJ1M CPU 模組。 置類型　CPU 類型 • CS1G-H　CS1G-CPU42H/43H/44H/45H • CS1H-H　CS1H-CPU63H/64H/65H/66H/67H • CJ1G-H　CJ1G-CPU42H/43H/44H/45H • CJ1H-H　CJ1H-CPU65H/66H/67H • CJ1M　CJ1M-CPU11/12/13/21/22/23 下列 CP 系列 CPU 模組相容。 • CP1H　CP1H-X40*/XA40* **備註** 如果將一個在 CX-Programmer 5.0 版以上建立的包含有功能區塊的使用者程式下載到一個並不支援功能區塊的 CPU 模組上(模組版本 2.0 以下的 CS/CJ 系列 CPU 模組)，則所有實例都會被視為不合規定的指令且不能進行編輯或執行使用者程式。 • NSJ　G5D (用於 NSJ5-TQ0□-G5D, NSJ5-SQ0□-G5D, NSJ8-TV0□-G5D, NSJ10-TV0□-G5D 及 NSJ12-TS0□-G5D) • FQM1-CM　FQM1-CM002 • FQM1-MMA　FQM1-MMA22 • FQM1-MMP　FQM1-MMP22
		CS/CJ/CP 系列功能限制 • 功能區塊定義中不支援的指令 區塊程式指令(BPRG 及 BEND)、副常式指令(SBS、GSBS、RET、MCRO、及 SBN)、跳越指令(JMP、CJP、及 CJPN)、步驟階梯指令(STEP 及 SNXT)、立即更新指令(!)、I/O 更新(IORF)、ONE-MS 計時器(TMHH) 詳細資訊，請參考 *2-3 功能區塊限制*。
相容的 CPU	電腦	IBM PC/AT 或相容的電腦
	CPU	使用 Windows 98、98SE、或 NT 4.0 (Service Pack 6 以上)應為 133 MHz Pentium 以上
	OS	Microsoft Windows 95、98、98SE、Me、2000、XP、或 NT 4.0 (Service Pack 6 以上)
	記憶體	使用 Windows 98、98SE、或 NT 4.0 (Service Pack 6 以上)應為至少 64 MB 詳細資訊，請參考 *CX-Programmer 5.0 版操作手冊* (W437)。
	硬碟大小	至少 100 MB 可用硬碟空間
	監控	至少 SVGA (800 x 600 pixels) **備註** 字形大小請使用"小字形"。
	光碟機	至少一個光碟機裝置
	COM 埠	至少一個 RS-232C 埠

<table>
<tr><th colspan="3">項目</th><th colspan="2">規格</th></tr>
<tr><td rowspan="11">CX-Programmer 版本 4.0 以下不支援的功能。</td><td rowspan="8">定義及建立功能區塊</td><td>功能區塊定義數</td><td colspan="2">CS1-H/CJ1-H CPU 模組：
• CPU44H/45H/64H/65H/66H/67H 字尾：每個 CPU 模組最多 1,024
• Suffix-CPU42H/43H/63H：每個 CPU 模組最多 128
CJ1M CPU 模組：
• CJ1M-CPU11/12/13/21/22/23：每個 CPU 模組最多 128
CP1H CPU 模組：
• 所有型號：每個 CPU 模組最多 128
NSJ 控制器：
• 所有型號：每個控制器最多 1,024
FQM1 彈性運動控制器：
• FQM1-CM002/MMA22/MMP22：每個控制器最多 128</td></tr>
<tr><td>功能區塊名稱</td><td colspan="2">最多 64 個字元</td></tr>
<tr><td rowspan="5">變數</td><td>變數名稱</td><td>最多 30,000 個字元</td></tr>
<tr><td>變數類別</td><td>輸入、輸出及外部</td></tr>
<tr><td>功能區塊定義中的 I/O 變數</td><td>最多 64 (不包括 EN 及 ENO)</td></tr>
<tr><td>變數所使用的位址分配</td><td>自動分配(分配範圍可以由使用者設定。)</td></tr>
<tr><td>實際位址規格</td><td>支援</td></tr>
<tr><td>陣列規格</td><td>支援(僅限於一次元陣列)</td></tr>
<tr><td>語言</td><td colspan="2">功能區塊可使用階梯圖程式編輯語言或結構化文字(ST，請參閱備註)建立。</td></tr>
<tr><td rowspan="2">建立實例</td><td>實例數</td><td colspan="2">CS1-H/CJ1-H CPU 模組：
• CPU44H/45H/64H/65H/66H/67H 字尾：每個 CPU 模組最多 2,048
• Suffix-CPU42H/43H/63H：每個 CPU 模組最多 256
CJ1M CPU 模組：
• CJ1M-CPU11/12/13/21/22/23：每個 CPU 模組最多 256
CP1H CPU 模組：
• 所有型號：每個 CPU 模組最多 256
NSJ 控制器：
• 所有型號：每個控制器最多 2,048
FQM1 彈性運動控制器：
• FQM1-CM002/MMA22/MMP22：每個控制器最多 256</td></tr>
<tr><td>實例名稱</td><td colspan="2">最多 30,000 個字元</td></tr>
<tr><td rowspan="3">將功能區塊儲存為檔案</td><td>專案檔案</td><td colspan="2">包含功能區塊定義及實例的專案檔案(.cxp/cxt)。</td></tr>
<tr><td>程式檔案</td><td colspan="2">包括功能區塊定義及實例的檔案記憶體程式檔案(*.obj)。</td></tr>
<tr><td>功能區塊資料庫檔案</td><td colspan="2">每個功能區塊定義都可以儲存為一個單獨檔案(.cxf)可重複使用於其他專案。</td></tr>
</table>

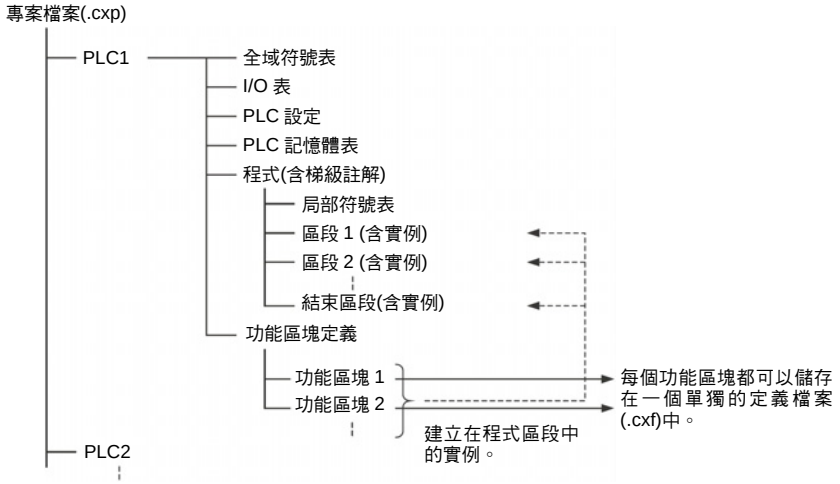
備註 結構化文字(ST 語言)符合 IEC 61131-3 標準，但 CX-Programmer 5.0 版只支援指定敘述、選擇敘述(CASE 及 IF 敘述)、反覆敘述(FOR、WHILE、REPEAT、及 EXIT 敘述)、RETURN 敘述、算術運算子、邏輯運算子、比較函數、數值函數、及註解。詳細資訊，請參考*附錄 B 結構化文字(ST 語言)規格*。

1-1-3 以 CX-Programmer 6.0 版建立的檔案

專案檔案(*.cxp)及檔案記憶體程式檔案(*.obj)

使用 CX-Programmer 所建立的專案，包含功能區塊定義及含有各種實例的專案、儲存在同一個標準專案檔案(*.cxp)及檔案記憶體程式檔案(*.obj)中。

下圖顯示一個專案的內容。功能區塊定義會建立在與相關 PLC 目錄中的程式相同的目錄層級上。



功能區塊資料庫檔案(*.cxf)

一個以 CX-Programmer 6.0 版建立在一個專案中的功能區塊定義可以儲存為一個檔案(1 個定義=1 個檔案)，讓這些定義可以載入到程式中重複使用。

備註 如果功能區塊進行巢狀時，則所有巢狀的(目的地)功能區塊定義都會包含在這個功能區塊資料庫檔案(.cxf)中。

包含功能區塊的專案文字檔(*.cxt)

相當於以 CX-Programmer 6.0 版建立的專案檔案(*.cxp)中的資料的相關資料可以儲存為 CXT 文字檔(*.cxt)。

1-1-4 CX-Programmer 5.0 版(及以上版本)中的功能區塊主選單

下表所列為與 CX-Programmer 5.0 版及以上版本中的功能區塊有關的主選單。關於所有主選單的詳細資訊，請參考 *CX-Programmer 5.0 版操作手冊* (W437)。

主選單

主選單	Submenu		快速鍵	功能
檔案	功能區塊	從檔案載入功能區塊	---	讀取儲存的功能區塊資料庫檔案(*.cxf)。
		將功能區塊儲存為檔案	---	將所建立的功能區塊定義儲存到一個檔案中(*[功能區塊資料庫檔案]*.cxf)。
編輯	更新功能區塊		---	當一個功能區塊定義的 I/O 變數在建立一個實例後已經改變時，則會使該實例的左匯流排以紅色顯示來表示錯誤。這個指令可以將該實例更新為新的資訊並清除錯誤。
	到下一層		---	跳越到所選取的實例的功能區塊定義。

主選單	Submenu		快速鍵	功能
檢視	監控 FB 階梯實例		---	在線上監控程式時，會監控該實例中的階梯圖的 I/O 位元及字組狀態(I/O 位元監控)。 (只有 CX-Programmer 6.0 版及以上版本支援)。
	監控 FB 實例		---	在線上監控程式時，會監控該實例中的階梯圖的 ST 變數狀態以及 I/O 位元及字組狀態(I/O 位元監控)。 (只有 CX-Programmer 6.1 版及以上版本支援)。
	到下一層		---	在右側顯示所選取的實例的功能區塊定義內容。(只有 CX-Programmer 6.0 版及以上版本支援。)
	到上一層		---	返回到呼叫實例(階梯圖或 ST)。(只有 CX-Programmer 6.0 版及以上版本支援。)
	視窗	FB 實例檢視器	---	顯示 FB 實例檢視器。(在巢狀時，顯示畫面會顯示諸如實例巢狀層級與實例中分配的變數位址之間的關係的詳細資訊。)
插入	引用功能區塊		F	在程式(區段)中的目前游標位置上建立一個實例。
	功能區塊參數		P	當游標在一個輸入變數的左側或一個輸出變數的右側時，可設定該變數的輸入或輸出參數。
PLC	功能區塊記憶體	功能區塊記憶體配置	---	設定內部分配給選定的實例變數的位址範圍(功能區塊實例區)。
		功能區塊記憶體統計	---	檢查內部分配給選定的實例變數的位址的狀態。
		功能區塊實例位址	---	檢查內部分配給選定的實例的每個變數的位址。
		功能記憶體最佳化	---	最佳化分配內部分配給各變數的位址。

主選單	Submenu		快速鍵	功能
工具	模擬	中止點 | 設定/清除中止點	---	設定或清除一個中止點(break point)。
		中止點 | 清除所有中止點	---	清除所有中止點。
		模式 | 執行(監控模式)	---	執行連續掃描。(將階梯執行引擎的執行模式設定為"監控"模式。)
		模式 | 停止(程式模式)	---	將模擬器的操作模式設定為"程式"模式。
		模式 | 暫停	---	暫停模擬器的作業。
		步驟執行	---	只執行模擬器程式的一個步驟。
		步驟執行 | 步驟進入	---	當有一個功能區塊呼叫指令時，這個指令可以進入執行內部程式步驟。
		步驟執行 | 步驟進出	---	在執行一個功能區塊的內部程式步驟時，這個指令可以回到上一層(呼叫來源)並暫停執行。
		步驟執行 | 連續步驟執行	---	連續執行各個步驟達一段固定的時間長度。
		步驟執行 | 掃描執行	---	執行一個循環並暫停執行。
		永遠顯示目前的執行點	---	搭配"步驟執行"或"連續步驟執行"指令使用來自動捲動顯示畫面並永遠顯示暫停點。
		中止點清單	---	顯示一個已經設定的中止點的清單。(作業可以跳越到一個指定點。)

主快顯主選單(Main Pop-up Menus)

功能區塊定義快顯主選單

快顯主選單		功能
插入功能區塊	階梯圖	建立一個採用階梯圖程式編輯語言演算法的功能區塊定義。
	結構化文字	建立一個採用 ST 語言演算法的功能區塊定義。
	從檔案	從一個功能區塊資料庫檔案(*.cxf)讀取一個功能區塊定義。

插入的功能區塊的快顯主選單

快顯主選單	功能
開啟	在視窗右側顯示所選定的功能區塊定義的內容。
儲存功能區塊檔案	將所選定的功能區塊定義儲存在一個檔案中。
編譯	匯集所選定的功能區塊定義。

功能區塊變數表快顯主選單

快顯主選單		功能
編輯		編輯變數。
插入垂直線		新增一個變數到最後一行。
插入垂直線	上方	在目前游標位置上方插入變數。
	下方	在目前游標位置上方插入變數。
剪下		剪下變數。
複製		複製變數。

快顯主選單	功能
貼上	貼上變數。
搜尋	搜尋變數。可以搜尋變數名稱、變數註解、或所有(字串)。
取代	取代變數。
刪除	刪除變數。
重新命名	只變更變數的名稱。

實例快顯主選單

快顯主選單	功能
編輯	變更實例名稱。
引用更新	當一個功能區塊定義的 I/O 變數在建立一個實例後已經改變時，則會使該實例的左匯流排以紅色顯示來表示錯誤。這個指令可以將該實例更新為新的資訊並清除錯誤。
監控 FB 階梯實例	在線上監控程式時，會監控該實例中的階梯圖的 I/O 位元及字組狀態(I/O 位元監控)。 (只有 CX-Programmer 6.0 版及以上版本支援)。
監控 FB 實例	在線上監控程式時，會監控該實例中的階梯圖的 ST 變數狀態以及 I/O 位元及字組狀態(I/O 位元監控)。 (只有 CX-Programmer 6.1 版及以上版本支援)。
登錄到監控視窗中	顯示 *FB 變數登錄*對話框以便從選定的實例登錄一個變數到*監控視窗*中。
功能區塊定義	在視窗右側顯示所選定的實例的功能區塊定義。

快速鍵(Shortcut Keys)

F 鍵:將功能區塊定義貼到程式中　將游標移到可在階梯區段視窗中產生複製的功能區塊實例的位置，並按一下 **F** 鍵。這項操作與選取*插入-引用功能區塊*相同。

P 鍵：輸入參數　將游標放在輸入變數的左側、或輸出變數的右側並按一下 **P 鍵**。這項操作與選取*插入-參數功能區塊*相同。

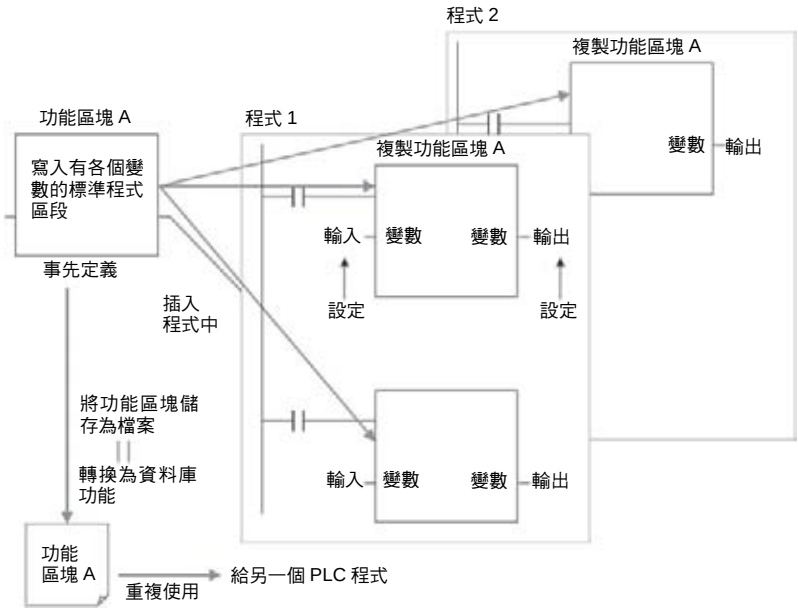
1-2 功能區塊(Function Blocks)

1-2-1 概要

一個功能區塊就是一個包含有事先定義的標準處理功能的基本程式元件。在定義一個功能區塊後，使用者只需將功能區塊插入程式中並設定 I/O 以便使用這項功能即可。

做為一個標準處理功能，功能區塊並沒有包含實際位址，而是包含變數。使用者可在這些變數中設定位址或常數。這些位址或常數稱為參數。變數本身所使用的位址會由 CX-Programmer 針對每個程式自動分配。

透過 CX-Programmer，一個功能區塊可以儲存為一個個別檔案並可重複使用在其他 PLC 程式中，所以標準處理功能可以組成資料庫。



1-2-2 功能區塊的優點

功能區塊讓各種複雜的程式編輯單位可以簡便的重複使用。一旦在一個功能區塊中建立標準程式編輯並且儲存在一個檔案中，它就可以重複使用，只需要將這個功能區塊放到一個程式中並設定功能區塊的 I/O 參數即可。可以重複使用現有功能區塊的能力，將可以顯著的節省程式建立/除錯的時間、減少編寫錯誤、並且可以讓程式更容易瞭解。

結構化的程式編輯

使用功能區塊所建立的結構化程式具有更佳的設計品質並且只需要更少的開發時間。

簡單易讀的"黑盒子"設計

I/O 運算元會在程式中顯示為變數名稱，因此程式就像在輸入或讀取程式時的一個"黑盒子"，不需要浪費時間來嘗試瞭解內部的演算法則。

將一個功能區塊用在多個處理上

利用標準程序中的參數做為輸入變數(例如計時器 SV、控制常數、速度設定、以及移動距離)，只需要根據一個功能區塊就可以很容易的建立許多不同的程序。

減少編寫錯誤

由於所有的區塊都已經除錯而且可以重複使用，因此可以減少編寫錯誤。

資料保護

功能區塊中的變數不能直接從外部存取，因此可以保護資料。(資料不會被不慎變更。)

透過變數設定改善重複使用性

功能區塊的 I/O 會輸入做為變數，所以在重複使用時並不需要變更區塊中的資料位址。

建立資料庫

獨立且可重複使用的程序(例如個別步驟、機器、設備、或控制系統的程序)可以儲存為功能區塊定義並轉換為資料庫功能。

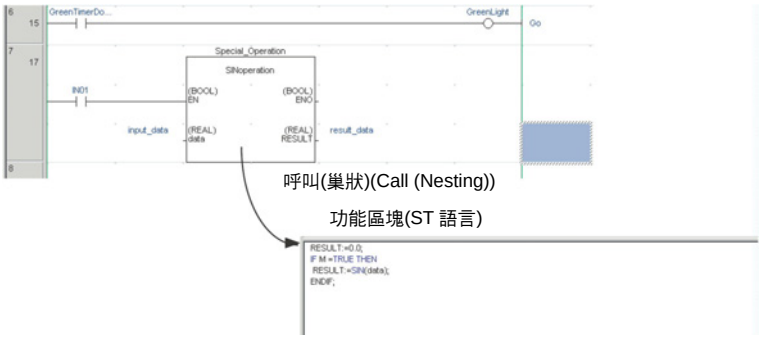
功能區塊採用變數名稱建立，不受實際位址約束，所以可以很方便開發新的程式，只需要從檔案讀取定義並將它們放到新的程式中即可。

支援巢狀及多重語言 (Supports Nesting and Multiple Languages)

數學運算式可以輸入結構化文字(ST)語言中。

透過 CX-Programmer 6.0 版及以上的版本，各個功能區塊可以進行巢狀的呼叫。功能區塊巢狀功能可以只在巢狀在一個階梯語言功能區塊中的 ST 語言功能區塊中執行特殊處理。

功能區塊(階梯語言)

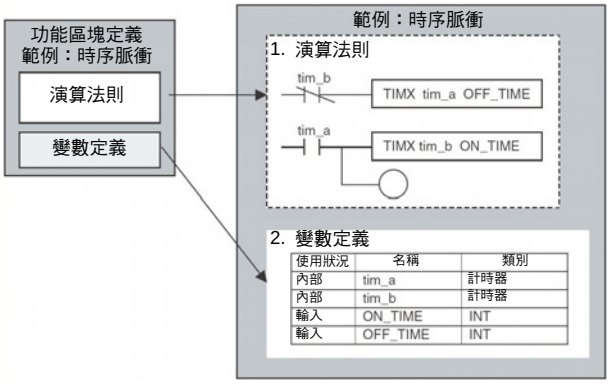


1-2-3 功能區塊結構(Function Block Structure)

功能區塊包括事先建立的功能區塊定義以及插入到程式中的功能區塊實例。

功能區塊定義

功能區塊定義也就是包含在功能區塊中的程式。每個功能區塊定義都包含有演算法則及變數定義，如下圖所示。



1. 演算法則

標準化的程式編輯是以變數名稱而不是以實際 I/O 記憶體位址撰寫的。在 CX-Programmer 中，演算法則可以以階梯圖程式編輯或者結構化文字撰寫。

2. 變數定義

變數表會表列每個變數的用法(輸入、輸出、或內部)及屬性(資料類型等)。詳細資訊，請參考 *1-3 變數*。

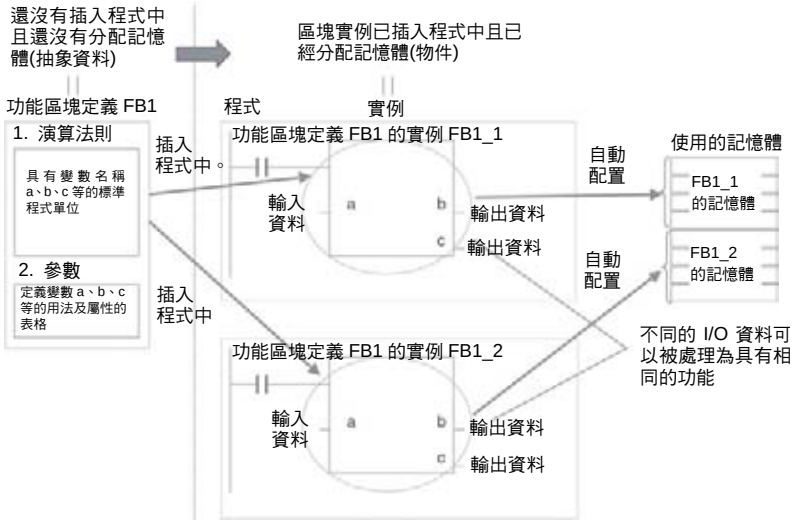
功能區塊定義數

根據 CPU 模組的型式，可以針對一個 CPU 模組建立的最大功能區塊定義數是 128 或 1,024。

實例

要在一個程式中使用一個實際功能區塊定義時，請建立一個功能區塊圖的拷貝並將它插入程式中。每個插入程式中的功能區塊定義即稱為一個“實例"或者“功能區塊實例"。每個實例會指定一個識別碼，稱為“實例名稱"。

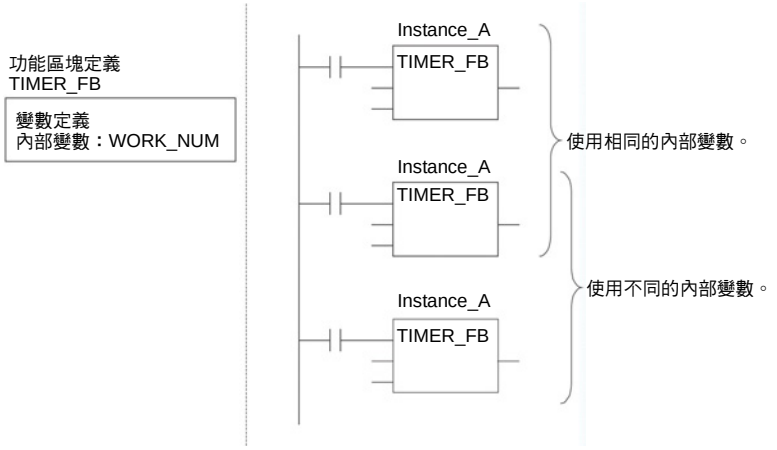
透過實例的產生，一個單一功能區塊定義可以用來處理具有相同功能的不同 I/O 資料。



備註 各個實例以名稱來進行管理。也可以在程式中插入一個以上具有相同名稱的實例。如果有兩個以上的實例具有相同的名稱，它們會使用相同的內部變數。不同名稱的實例會有不同的內部變數。

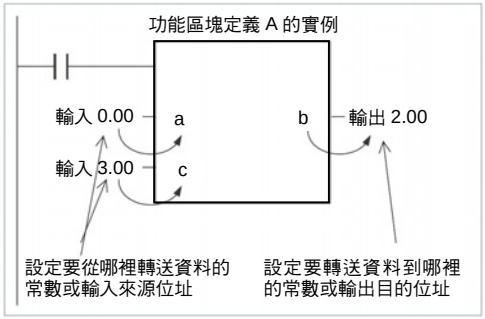
例如，假設有多個功能區塊使用一個計時器做為內部變數。在這種情況下，所有實例都必須賦予不同的名稱。如果有一個以上的實例使用相同的名稱，則同一個計時器會被使用在多個位置，而導致計時器重複使用。

不過，如果沒有使用內部變數或者只是暫時使用且會在下次執行一個實例時進行起始化，則可以使用相同的實例名稱來儲存記憶。



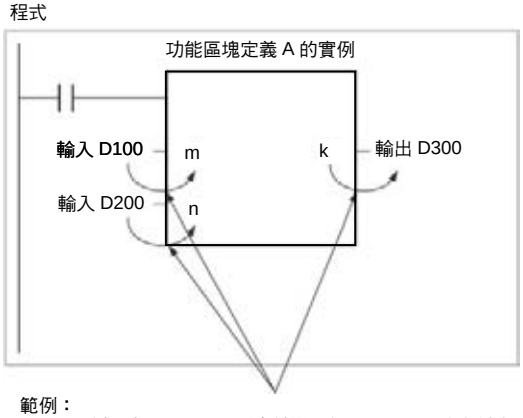
實例數 可以從一個單一功能區塊定義建立多個實例。根據 CPU 模組的型式，最多可以針對一個 CPU 模組建立 256 或 2,048 的實例。容許的實例數與功能區塊定義數及實例所插入的 Task (工件)數無關。

參數 每次建立一個實例時，必須設定實際 I/O 記憶體位址或所使用的 I/O 變數的常數來將輸入資料值轉送給實例並從實例取得輸出資料值。這些位址或常數稱為參數。



在這裡，它並不是輸入來源位址本身，而是形式和大小由轉送給功能區塊的變數資料類型所指定的輸入位址的內容。同樣的，它也不是輸出目的位址本身，而是形式和大小由從功能區塊所轉送的變數資料類型所指定的輸出位址的內容。

即使一個輸入來源位址(即，一個輸入參數)或一個輸出目的位址(即，一個輸出參數)是一個文字位址，所轉送的資料也會是形式和大小由源自指定文字位址的變數資料類型所指定的資料。



範例：
如果 m 的類別是 WORD，則會轉送一個取自 D100 的資料字組到這個變數。
如果 n 的類別是 DWORD，則會轉送兩個取自 D200 至 D201 的資料字組到這個變數。
如果 k 的類別是 LWORD，則會轉送四個取自 D300 至 D303 的資料字組到這個變數。

備註 (1) 只有下列區域中的位址可以用來做為參數：CIO 區域、附屬區域、DM 區域、EM 區域(記憶庫 0 到 C)、保留區域、及工作區域。

下列則不能使用：索引及資料暫存器(包括直接和間接規格)和 DM 區域及 EM 區域的間接位址(包括二進位及 BCD 模式)。

(2) 使用者程式中的區域和全域符號也可以指定做為參數。不過，要指定時，區域或全域符號的資料大小必須與功能區塊變數的資料大小相同。

(3) 在執行一個實例時，輸入值會在演算法則處理之前從參數轉送給輸入變數。輸出值則會在演算法則剛處理後從輸出變數轉送給參數。如果必須在演算法則執行循環中讀取或寫入一個值，則不可將這個值轉送給參數或從參數轉送這個值。請將這個值指定為一個內部變數並使用一個 AT 設定(指定的位址)。

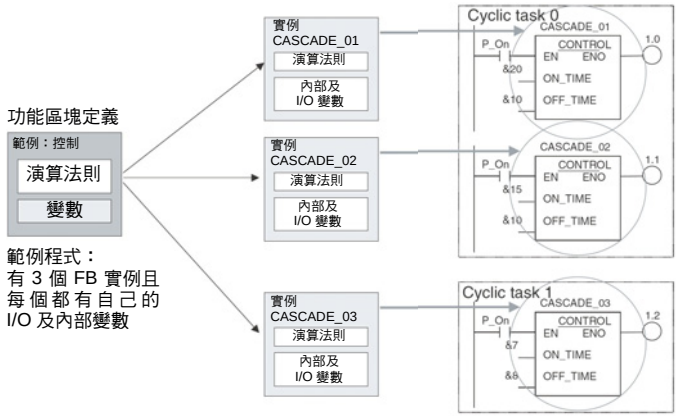
⚠ **注意** 如果在一個輸入參數中指定一個位址，則會將位址中的值轉送給輸入變數。實際位址資料本身則不能轉送。

⚠ **注意** 參數不能用來在演算法則執行循環中讀取或寫入值。請使用有 AT 設定(指定的位址)的內部變數。或者，可引用一個全域符號做為外部變數。

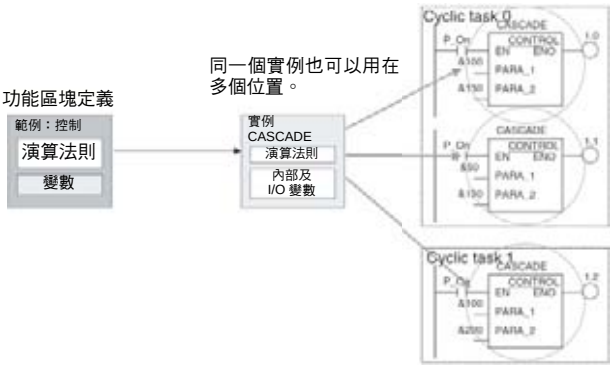
■ **參考資訊**

可以從一個單一功能區塊利用類參數元件(例如固定值)做為輸入變數並針對每個實例變更轉送給輸入變數的值即可很容易的建立各種程序。

範例：從 1 個功能區塊定義建立 3 個實例



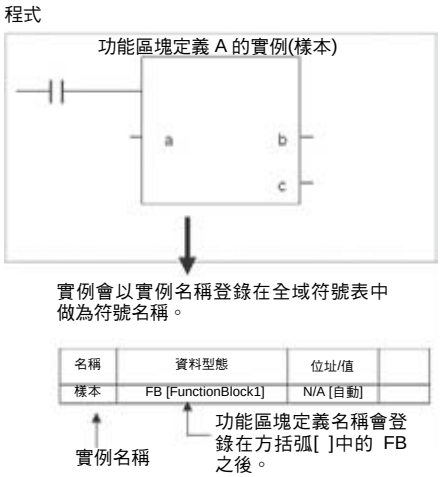
如果沒有使用內部變數、如果處理將不受影響、或如果將內部變數用在其他位置，則相同的實例名稱可以用在程式中的多個位置。



在使用相同的記憶區時，必須注意一些事項。例如，如果包含有一個計時器指令的相同實例用在一個以上的程式位置上，則會使用相同的計時器號碼而造成循環重複，且如果兩個指令都執行，計時器將無法正常作用。

實例的登錄

每個實例名稱會登錄在全域符號表中做為檔案名稱。

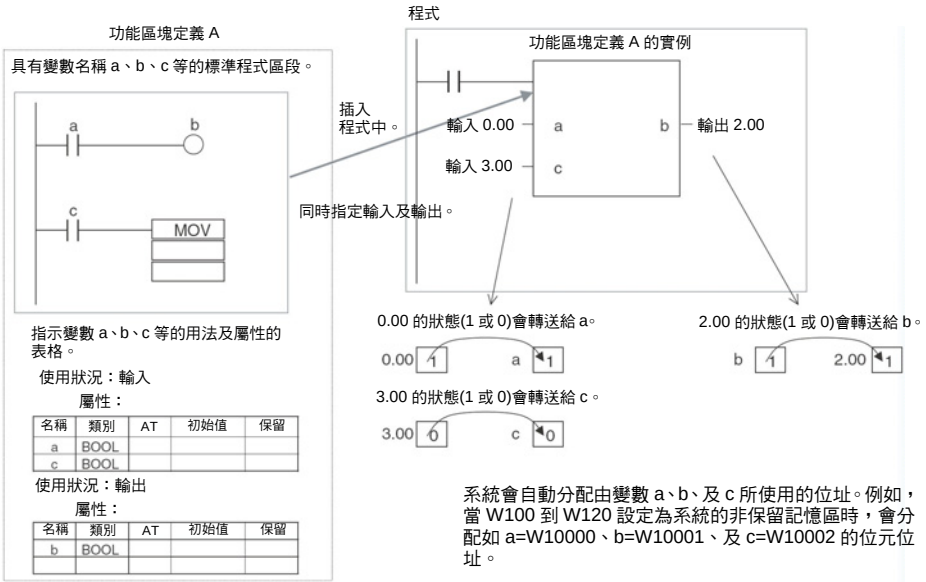


1-3 變數

1-3-1 簡介

在一個功能區塊中，位址(請參閱備註)並不會輸入做為實際 I/O 記憶體位址，它們都會輸入做為變數名稱。每次建立一個實例時，變數所使用的實際位址會由 CX-Programmer 自動分配在指定的 I/O 記憶區中。因此，使用者並不須要知道功能區塊中所使用的實際 I/O 記憶體位址，就像他並不須要知道電腦中的實際記憶體配置一樣。就這方面來說，一個功能區塊和一個副常式並不相同，也就是說，功能區塊會使用就像是"黑盒子"的變數和位址。

範例程式：



備註 常數不能登錄做為變數。請直接在指令運算元中輸入常數。

- 階梯圖程式編輯語言：在#後輸入十六進位數值以及在&後十進位數值。
- 結構化文字(ST 語言)：在 16#後輸入十六進位數值以及在同樣符號後輸入十進位數值。

例外：請直接在指令運算元中輸入針對指標暫存器 IR0 到 IR15 及資料暫存器 DR0 到 DR15 所直接或間接指定的位址。

1-3-2 變數用法及屬性

變數用法

支援下列變數類別(用法)。

內部： 內部變數只用在一個實例中。它們不能用來直接將資料轉送給 I/O 參數或從 I/O 參數轉送資料。

輸入： 輸入變數可以從實例以外的輸入參數輸入資料。預設的輸入變數是一個 EN (Enable)變數，它會轉送輸入條件資料。

輸出： 輸出變數可以輸出資料到實例以外的輸出參數。預設的輸出變數是一個 ENO (Enable Out)變數，它會轉送實例的執行狀態。

外部： 外部變數是事先透過 CX-Programmer 登錄的由系統所定義的變數(例如條件旗標及某些附屬區域位元)，或者是用於實例中由使用者所定義的全域符號。

關於變數用法的詳細資訊，請參考 *2-1-2 功能區塊元件*的*變數定義*項目下的*變數類別(用法)*。

下表顯示可以使用的變數數和針對每一種變數用法預設建立的變數種類。

1-3-3 變數屬性

變數具有下列屬性。

變數名稱

變數名稱用來在功能區塊識別變數。如果在其他功能區塊中使用相同的名稱並沒有關係。

備註

變數名稱長度最多可達 30,000 個字元，但開頭必須不是數字。同時，名稱在同一行中不能包含 2 個底線字元。字元串不能與一個指標暫存器相同，例如 IR0 到 IR15。關於其他限制的詳細資訊，請參考 *2-1-2 功能區塊元件*中的*變數定義*。

資料型態

請為變數選取下列一個資料類型：

BOOL, INT, UINT, DINT, UDINT, LINT, ULINT, WORD, DWORD, LWORD, REAL, LREAL, TIMER, COUNTER

關於變數資料類型的詳細資訊，請參考 *2-1-2 功能區塊元件*中的*變數定義*。

AT 設定(分配給一個實際位址)

也可以針對一個特定的 I/O 記憶體位址設定一個變數而不由系統自動分配。要指定一個特定位址時，使用者可以在這個屬性中輸入想要的 I/O 記憶體位址。這個屬性只能針對內部變數設定。即使設定有一個指定的位址，演算法則仍必須使用變數名稱。

關於 AT 設定的詳細資訊，請參考 *2-1-2 功能區塊元件*中的*變數定義*；關於使用 AT 設定的詳細資訊，請參考 *2-4-3 內部變數的 AT 設定*。

陣列設定

一個變數可以視為一個具有相同屬性的資料陣列。要將一個變數轉換為一個陣列時，必須指定它是一個陣列並指定最大元件數。

這個屬性只能針對內部變數設定。CX-Programmer 5.0 版以上的版本只支援一次元陣列。

- 設定程序
  按一下 **Advanced (進階)**按鈕，選取 *Array Variable (陣列變數)*選項，並輸入最大元件數。
- 在功能區塊定義的演算法則中輸入一個陣列變數名稱時，請在方括弧中的變數號碼後輸入陣列指標號碼。

關於陣列設定的詳細資訊，請參考 *2-1-2 功能區塊元件*中的*變數定義*。

初始值

這是在實例首次執行之前設定在一個變數中的起始值。之後，這個值可能會因為實例的執行而改變。

例如，將一個 Boolean (BOOL)變數(位元)設定為 1 (TRUE)或 0 (FALSE)。將一個 WORD 變數設定為一個介於 0 與 65,535 之間的值(十六進位則介於 0000 與 FFFF 之間)。

如果沒有設定起始值，則變數會被設定為 0。例如，一個 Boolean 變數會是 0 (FALSE)而一個 WORD 變數會是 0000 (十六進位)。

保留

如果您想要在 PLC 再次開啟(ON)時及 PLC 開始操作時保留一個變數的資料，請選取 *Retain Option (保留選項)*。

- 設定程序
  選取 *Retain Option (保留選項)*。

1-3-4 變數屬性及變數用法

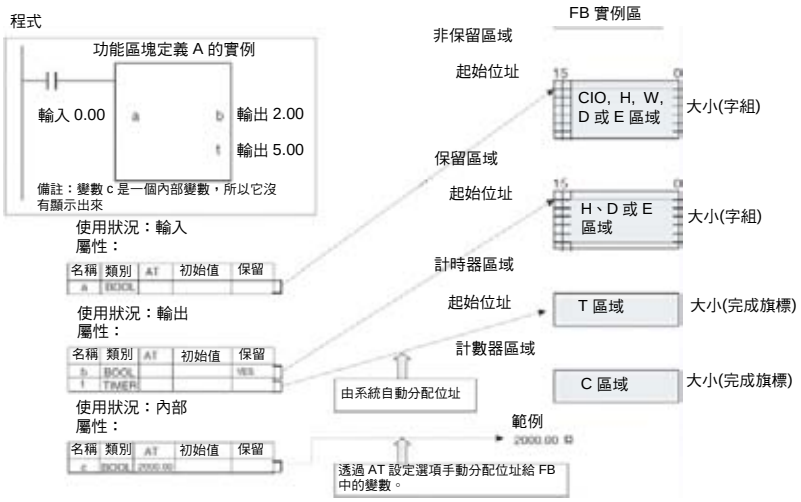
下表顯示根據變數的用法哪些屬性必須設定、可以設定、及不能設定。

Property	變數用法		
	內部	輸入	輸出
名稱	必須設定	必須設定	必須設定
資料型態	必須設定	必須設定	必須設定
AT (指定的位址)	可以設定	無法設定	無法設定
初始值	可以設定	可以設定 (請參閱備註。)	可以設定
保留	可以設定	可以設定 (請參閱備註。)	可以設定

備註 各項輸入可以被設定為起始值，但實際輸入參數的值會被賦予屬性。

1-3-5 變數位址的內部分配

在從一個功能區塊定義建立一個實例時，CX-Programmer 會內部分配位址給各個變數。除了已經以 *AT 設定*屬性指定實際位址的變數之外，位址會分配給所有登錄在功能區塊定義中的變數。



設定變數的內部分配區域

使用者可設定由系統內部分配位址的功能區塊實例區。變數會由系統自動分配給由使用者設定的適當實例區。

設定程序

從 *PLC* 主選單中選取 ***Function Block Memor (功能區塊記憶體) - Function Block Memory Allocation (功能區塊記憶體配置)***。在功能區塊記憶體配置對話框中設定記憶區。

功能區塊實例區

3.0 版以上的 CS/CJ 系列 CPU 模組及 NSJ 控制器

FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留	H512	H1407	896	CIO, WR, HR, DM, EM
保留	H1408	H1535	128	HR, DM, EM
計時器	T3072	T4095	1024	TIM
計數器	C3072	C4095	1024	CNT

FQM1 彈性運動控制器

FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留	5000	5999	1000	CIO, WR, DM
保留	無			
計時器	T206	T255	50	TIM
計數器	C206	C255	50	CNT

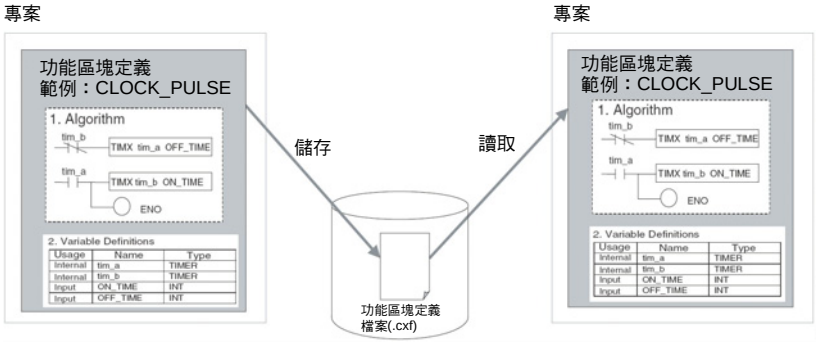
功能區塊保留區域字組 (H512 至 H1535)

功能區塊保留區域字組會被指定為從 H512 到 H1535。這些字組與用於程式的標準保留區域(H000 到 H511)不同，並且只用於功能區塊實例區(內部分配的變數區)。這些字組不能指定為指令運算元。如果在一個功能區塊沒有建立時輸

入，它們會顯示為紅色。雖然字組可以在建立一個功能區塊時輸入，但在檢查程式時會出現一個錯誤。如果這個區域在功能區塊記憶體配置對話框中被指定為不要保留，則在開始作業時會開啟/關閉電源或清除這個區域而不會保留數值。

1-4 將功能區塊定義轉換為資料庫檔案

一個使用 CX-Programmer 建立的功能區塊定義可以儲存為一個稱為功能區塊定義檔案的檔案，檔案的副檔名為*.cxf。這些檔案可以在其他專案(PLC)中重複使用。



1-5 使用程序

一旦建立一個功能區塊定義且已經建立一個演算法則的實例後，這個實例可以在執行時將它叫出來使用。同時，建立的功能區塊定義可以儲存在一個檔案中使它可以重複使用在其他專案(PLC)中。

1-5-1 建立功能區塊及執行實例

下列程序概要說明建立及執行一個功能區塊所需的步驟。

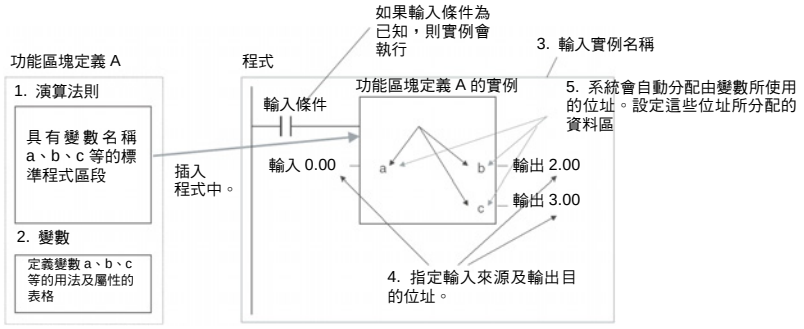
1,2,3… 1. 首先，在階梯圖程式或 ST 語言中建立包括演算法則及變數定義的功能區塊定義。或者，插入一個已經事先備妥的功能區塊資料庫檔案。

備註 (a) 請建立包含完整變數名稱的演算法則。

(b) 在階梯圖程式編輯語言中輸入演算法則時，以 5.0 版以下的 CX-Programmer 版本所建立的專案檔案可以透過將專案檔案讀入 5.0 版以上的 CX-Programmer 並複製及貼上有用的部份來重複使用。

2. 在建立程式時，插入完成的功能區塊定義的拷貝。這個步驟可以建立功能區塊的實例。
3. 為每個實例輸入一個實例名稱。
4. 將變數的輸入來源位址及/或常數以及輸出目的位址及/或常數設定為參數來轉送每個實例的資料。
5. 選取建立的實例，從 *PLC* 主選單中選取 ***Function Block Memory (功能區塊記憶體) - Function Block Memory Allocation (功能區塊記憶體配置)***，並針對每一種變數設定內部資料區。
6. 將程式傳送到 CPU 模組。

7. 啟動 CPU 模組中的程式執行，如果實例的輸入條件為 ON，它們會被叫出並執行。

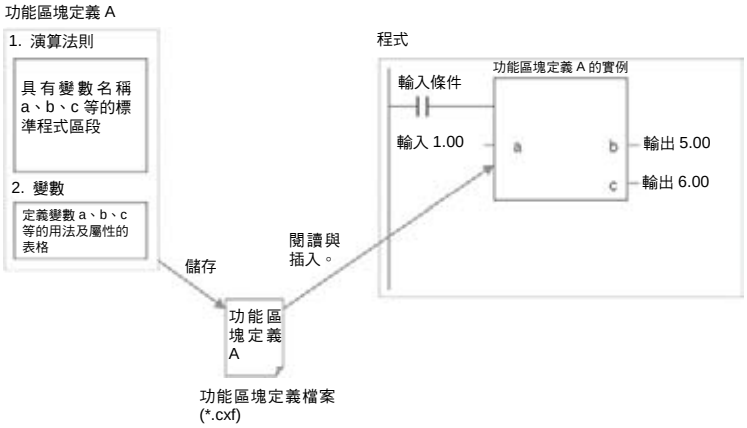


1-5-2 重複使用功能區塊

利用下列程序來將一個功能區塊定義儲存為一個檔案並將它使用於另一個 PLC 的程式中。

1,2,3…

1. 選取您要儲存的功能區塊並將它儲存為功能區塊定義檔案(*.cxf)。
2. 開啟另一個 PLC 的專案並開啟/讀取先前儲存的功能區塊定義檔案(*.cxf)。
3. 在建立新程式時將功能區塊定義插入程式中。



備註 在 CX-Programmer 5.0 版中，每個功能區塊定義可以向一個程式般進行匯集及檢查。我們建議在儲存及重複使用檔案之前進行匯集來針對每個功能區塊定義檔案執行程式檢查。

1-6 版本升級資訊

本節將概要說明 CX-Programmer 從 5.0 版升及到 6.0 版以及從 6.0 版升及到 6.1 版的功能改善。

6.0 到 6.1 版的升級資訊

適用 PLC 機型

6.0 版到 6.1 版的升及適用於下列 PLC 型號。

- CP-系列 CP1H (CP1H-XA, CP1H-X 與 CP1H-Y)

支援 NSJ 系列之 NSJ 控制器

PLC 機型(“裝置類型”)可以設定為“NSJ”，CPU 類型也可以設定為 G5D。

支援 FQM1 模組 3.0 版　　現在支援 FQM1 Flexible Motion Controller (彈性位置控制器)的新機型(也就是 FQM1-CM002 協調模組和 FQM1-MMA22/MMP22 運動控制模組)。

實例 ST/階梯圖程式模擬功能

先前版本(6.0 版)	新版本(6.1 版)
CX-Simulator 可以用來執行階梯圖程式之步驟(Step Run)、連續執行步驟(Continuous Step Run)、執行單一循環(Scan Run)及設定 I/O 中斷點條件。	Step Run、Continuous Step Run、Scan Run 及設定/清除中斷點功能都可以像 CXammer 功能般執行。 所有上述功能都可以在功能區塊中和階梯圖程式及階梯/ST 程式一起使用。 **備註** 必須安裝 CX-Simulator 1.6 版(另售)才能使用這些功能。 **備註** 無法設定 I/O 中斷點條件。

功能區塊之改良功能

<u>在功能區塊中監控 ST 程式</u>

先前版本(6.0 版)	新版本(6.1 版)
監控其他線上程式時，無法監控功能區塊 instances (實例)內的 ST 程式作業。 (可以檢查功能區塊定義之程式的內容及監控功能區塊 instance 之階梯圖的 I/O 狀態。)	在監控程式時，可以監控功能區塊 instance 之 ST 程式的狀態。 如要監控 ST 程式的狀態，請按兩下功能區塊 instance，或以滑鼠右鍵點選該 instance，並從快顯主選單中選擇 ***Monitor FB Instance (監控 FB Instance)***。此時，將可以變更 PV 及強制設定/重置位元。 **備註** 不支援線上編輯。

<u>功能區塊的密碼保護</u>

先前版本(6.0 版)	新版本(6.1 版)
用戶可以設定功能區塊的屬性，以防止功能區塊定義之程式出現在畫面上。	有兩種密碼保護方式可供設定。 • 同時限制讀與寫的密碼保護。 • 只限制讀取的密碼保護。

5.0 到 6.0 版的升級資訊

巢狀功能區塊

先前版本(5.0 版)	新版本(6.0 版)
功能區塊無法呼叫另一個功能區塊(不支援巢狀呼叫。)	功能區塊可呼叫另一個功能區塊(巢狀呼叫)。最多支援 8 層巢狀呼叫。 呼叫功能的語言和被呼叫的功能區塊可以使用階梯語言或 ST 語言。 功能區塊之間的巢狀關係可以用目錄樹狀格式來顯示。當功能區塊進行巢狀時，只會針對呼叫功能區塊和它所叫出的(巢狀的)功能區塊定義儲存一個功能區塊資料庫檔案(副檔名為.cxf)。

功能區塊中的階梯圖程式的 I/O 位元監控支援

先前版本(5.0 版)	新版本(6.0 版)
功能區塊 instance 的階梯圖之 I/O 狀態，無法在線上監控程式時一併監看。 (只能在功能區塊定義中檢查程式。)	功能區塊 instance 的階梯圖之 I/O 狀態，可在線上監控程式時一併監看。 如欲監看 I/O 狀態，可以雙擊功能區塊 instance，或以滑鼠右鍵點選該 instance，並從快顯主選單中選擇 ***Monitor FB Ladder Instance***。這個時候就能監看 I/O 位元的狀態及 Words 的內容、更改 PVs、強制設定/重置位元及監看位元的變異(ON/OFF 轉換)。 **備註** 不支援線上編輯，也無法更改計時器/計數器 SVs。

在 Watch (查看)視窗中登錄及監控功能區塊 Instance 的變數

先前版本(5.0 版)	新版本(6.0 版)
如要在 Watch 視窗中登錄功能區塊 instance 的變數，請務必先開啟 Watch 視窗、按兩下該視窗，然後從下拉式清單中選擇您要的變數。	在 Watch 視窗中，可以輕鬆地一併登錄功能區塊 instance 中的數個變數。下面幾種方式可以顯示 *FB variables registration (功能區塊變數登錄)*對話框，變數也可以在對話框中登錄。 • 以滑鼠右鍵點選功能區塊 instance，然後從快顯主選單中選擇 Register in Watch Window (在 Watch 視窗中登錄)。 • 在程式或變數表中選擇您要的功能區塊 instance，然後利用複製/貼上或拖/放的方式，將 instance 放入 Watch 視窗中。 • 將游標移到 Watch 視窗的空白列上，然後從快顯主選單中選擇 Register in Watch Window (在 Watch 視窗中登錄)。

其他功能區塊的改善

- 功能區塊內的階梯圖程式支援參照快顯功能。
- ST 語言的說明程式可以從 ST 編輯器的快顯主選單中開啟。
- 只要按兩下功能區塊 instance，就能開啟功能區塊的定義。
- 功能區塊 instance 的輸入參數獲得確認後，游標就會自動往下移。

章節 2
規格

本章提供在使用功能區塊時可供參考的規格，包括有關功能區塊、實例、及相容 PLC 的規格、以及使用注意事項及指導說明。

2-1 功能區塊規格

2-1-1 功能區塊規格

項目	說明
功能區塊定義數	CS1-H/CJ1-H CPU 模組： • Suffix-CPU44H/45H/64H/65H/66H/67H 字尾：每個 CPU 模組最多 1,024 • Suffix-CPU42H/43H/63H：每個 CPU 模組最多 128 CJ1M CPU 模組： • CJ1M-CPU11/12/13/21/22/23：每個 CPU 模組最多 128 CP1H CPU 模組： • CP1H-XA/X/Y：每個 CPU 模組最多 128 NSJ 控制器： • 所有型號：每個控制器最多 1,024 FQM1 彈性位置控制器： • FQM1-CM002/MMA22/MMP22：每個控制器最多 128
實例數	CS1-H/CJ1-H CPU 模組： • Suffix-CPU44H/45H/64H/65H/66H/67H 字尾：每個 CPU 模組最多 2,048 • Suffix-CPU42H/43H/63H：每個 CPU 模組最多 256 CJ1M CPU 模組： • CJ1M-CPU11/12/13/21/22/23：每個 CPU 模組最多 256 CP1H CPU 模組： • CP1H-XA/X/Y：每個 CPU 模組最多 256 NSJ 控制器： • 所有型號：每個控制器最多 2,048 FQM1 彈性位置控制器： • FQM1-CM002/MMA22/MMP22：每個控制器最多 256
實例巢狀層級數	• CX-Programmer 5.0 版： 不支援巢狀。 • CX-Programmer 6.0 版及以上版本： 支援最多達 8 層的巢狀。(從程式叫出的實例以一個巢狀層計算。)
I/O 變數數	每個功能區塊定義最多 64 的變數

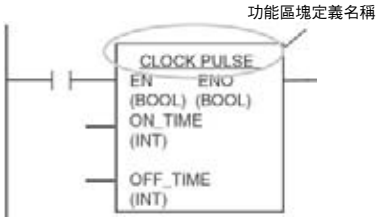
2-1-2 功能區塊元素

下表顯示在定義功能區塊時使用者所必須輸入的項目。

項目	說明
功能區塊定義名稱	功能區塊定義的名稱
語言	功能區塊定義中所使用的程式編輯語言。請選取階梯圖程式編輯或結構化文字
變數定義	變數設定，功能區塊執行時所需的如運算元及回歸值 • 變數的類別(用法) • 變數的名稱 • 資料型態變數 • 起始值的變數
演算法則	在階梯或結構化文字中輸入程式編輯邏輯。 • 輸入使用這些變數的程式編輯邏輯。 • 直接輸入常數而不要登錄到變數中。
註解	功能區塊可以加上註解。

功能區塊定義名稱

每個功能區塊定義都有一個名稱。名稱長度最多可為 64 個字元且沒有禁止使用的字元。預設的功能區塊名稱為 FunctionBlock□，其中□為號碼(依序指定)。



語言

選取階梯圖程式編輯語言或結構化文字(ST 語言)。

備註 (1) 關於 ST 語言的詳細資訊，請參考*附錄 B 結構化文字(ST 語言)規格*。

(2) 在進行巢狀時，使用 ST 語言及階梯語言的功能區塊可以自由結合(僅限於 6.0 版以上)。

變數定義

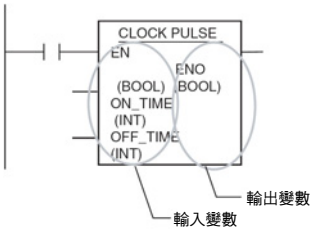
定義用於功能區塊定義中的運算元及變數。

變數名稱

- 變數名稱長度最多可達 30,000 個字元。
- 變數名稱不能包含空格或下列任何字元：
  ! " # $ % & ' ( ) = - ~ ^ \ | ' @ { [+ ; * :] } < , > . ? /
- 變數名稱不能以數字(0 到 9)開頭。
- 變數名稱不能在同一行中包含 2 個底線字元。
- 下列字元不能用來表示 I/O 記憶體中的位址。

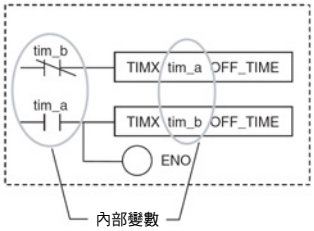
  A、W、H (或 HR)、D (或 DM)、E (或 EM)、T (或 TIM)、C (或 CNT)後隨數值(文字位址)

變數表示



變數表

使用狀況	名稱	類別
內部	tim_a	計時器
內部	tim_b	計時器
輸入	ON_TIME	INT
輸入	OFF_TIME	INT



變數類別(用法)

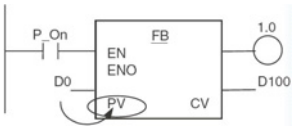
項目	變數類別			
	輸入	**輸出**	**內部**	**外部**
定義	給實例的運算元	來自實例的回歸值	只使用在實例中的變數	事先以 CX-Programmer 登錄為變數的全域符號或使用者所定義的全域符號。
下次執行時的值的狀態	數值不轉送給下次執行。	數值轉送給下次執行。	數值轉送給下次執行。	數值轉送給下次執行。
顯示	顯示在實例的左側。	顯示在實例的右側。	不顯示。	不顯示。
容許數目	每個功能區塊最多 64 (不包括 EN)	每個功能區塊最多 64 (不包括 ENO)	不限定	不限定
AT 設定	無	無	支援	無
陣列設定	無	無	支援	無
保留設定	無	支援	支援	無
預設建立的變數	EN (Enable)： 接收一個輸入條件。	ENO (Enable 輸出)： 輸出功能區塊的執行狀態。	無	在 CX-Programmer 中事先登錄為變數的預先定義符號，如條件旗標及某些附屬區域位元。

備註 關於外部變數的詳細資訊，請參考*附錄 C 外部變數*。

■ 輸入變數

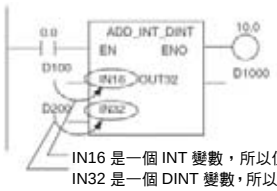
輸入變數可將外部運算元轉送給實例中。輸入變數會顯示在實例的左側。

輸入來源的值(只在叫出實例之前包含在指定的參數中的資料)會被轉送給輸入變數。

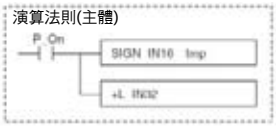


指定為輸入(值為 D0)的參數的值會被轉送給實例的輸入變數(PV)。

範例

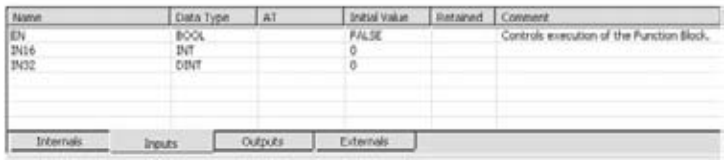


IN16 是一個 INT 變數，所以使用 D100 的內容。
IN32 是一個 DINT 變數，所以使用 D200 與 D201 的內容。



變數表

使用狀況	名稱	類別
內部	tmp	DINT
輸入	EN	BOOL
輸入	IN16	INT
輸入	IN32	DINT
輸出	ENO	BOOL
輸出	OUT32	DINT



備註 1. 同一個名稱不能指定給一個輸入變數及輸出變數。如果輸入變數及輸出變數需要有相同的變數，請以不同的名稱登錄變數並將輸入變數的值以一個諸如 MOV 的指令轉存到功能區塊中的輸出變數。

2. 在執行實例時，輸入值會在演算法則處理之前從參數轉送給輸入變數。因此，在演算法則中數值不能從參數讀入到輸入變數。如果必須在演算法則執行循環中讀取一個值，請不要從一個參數轉送數值。請將這個值指定為一個內部變數並使用一個 AT 設定(指定的位址)。或者，可引用一個全域符號做為外部變數。

初始值

起始值可以針對輸入變數設定，但輸入參數的值將會被啟用(輸入參數值會在輸入變數 EN 的參數轉為 ON 且實例執行時設定)。

備註 在使用 CXProgrammer 時不能忽略輸入參數設定。

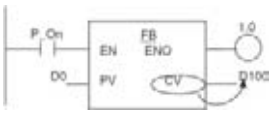
EN (Enable)變數

在建立一個輸入變數時，預設的輸入變數是 EN 變數。實例將會在輸入變數 EN 的參數為 ON 時執行。

■ 輸出變數

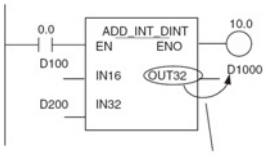
輸出變數會將回歸值從實例轉送給外部應用上。輸出變數會顯示在實例的右側。

在實例執行後，輸出變數的值會轉送給指定的參數。

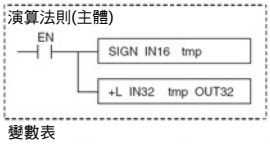


輸出變數(CV)的值會轉送給指定為輸出目的地的參數，在這個範例中是 D100。

範例

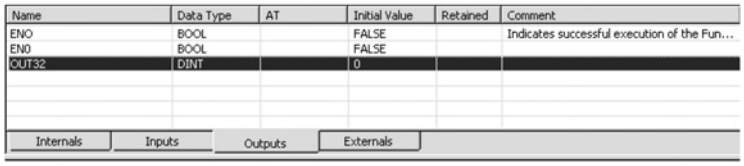


OUT32 是一個 DINT 變數，所以變數的值會轉送給 D1000 及 D1001。



變數表

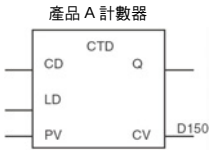
使用狀況	名稱	類別
內部	tmp	DINT
輸入	EN	BOOL
輸入	IN16	INT
輸入	IN32	DINT
輸出	ENO	BOOL
輸出	OUT32	DINT



就像內部變數一樣，輸出變數的值會保留直到下次實例執行時(即，當 EN 轉為 OFF 時，輸出變數的值會被保留)。

範例程式：

在下列範例中，輸出變數 CV 的值會被保留直到下次實例執行時。



備註

1. 同一個名稱不能指定給一個輸入變數及輸出變數。如果輸入變數及輸出變數需要有相同的變數，請以不同的名稱登錄變數並將輸入變數的值以一個諸如 MOV 的指令轉存到功能區塊中的輸出變數。
2. 在執行實例時，輸出變數會在演算法則處理之後轉送給相應的參數。因此，在演算法則中數值不能從輸出變數寫入到參數中。如果必須在演算法則執行循環中寫入一個值，請不要將值寫入到一個參數中。請將這個值指定為一個內部變數並使用一個 AT 設定(指定的位址)。

初始值

可以針對一個沒有被保留的輸出變數設定一個起始值，例如，當"保留選項"沒有選取時。如果"保留選項"已經選取則不能針對輸出變數設定一個起始值。

如果 IOM 保留位元(A50012)為 ON，則起始值將不會被寫入輸出變數。

附屬區域控制位元		初始值
IOM 保留位元(A50012)	ON	起始值將不會設定。

ENO (Enable 輸出)變數

ENO 變數會建立做為預設輸出變數。ENO 輸出變數在叫出實例時不會轉為 ON。使用者可以變更這個值。ENO 輸出變數可以用來做為一個旗標來檢查實例的執行是否已經正常完成。

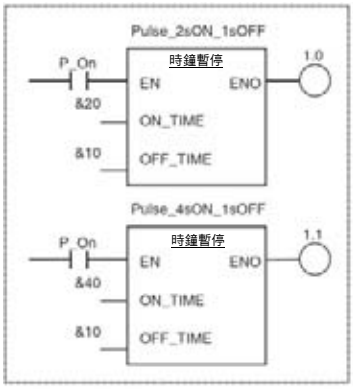
■ 內部變數

內部變數只用在一個實例中。這些變數是在每個實例的內部。它們不能從實例的外部進行參照且不會顯示在實例中。

內部變數的值會保留直到下次實例執行時(即，當 EN 轉為 OFF 時，內部變數的值會被保留)。因此，即使相同功能區塊定義的實例以相同的 I/O 參數執行，結果也不必然會相同。

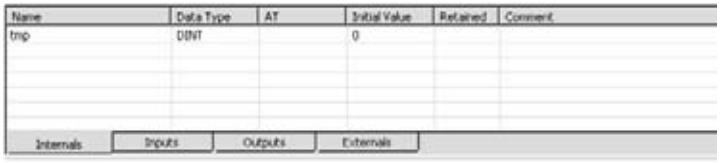
範例程式：

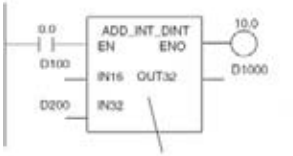
實例 Pulse_2sON_1sOFF 中的內部變數 tim_a 與實例 Pulse_4sON_1sOFF 中的內部變數 tim_a 不同，因此實例不能參照且不會影響其他的每個 tim_a 的值。

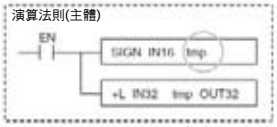


變數表

使用狀況	名稱	資料型態
內部	tim_a	計時器
內部	tim_b	計時器
輸入	ON_TIME	INT
輸入	OFF_TIME	INT







內部變數 tmp 不會顯示出來。

變數表

使用狀況	名稱	資料型態
內部	tmp	DINT
輸入	EN	BOOL
輸入	IN16	INT
輸入	IN32	DINT
輸出	ENO	BOOL
輸出	OUT32	DINT

保留資料渡過電力中斷及作業開始

內部變數會保留最後一次實例被叫出時的值。另外，也可以選取"保留選項"來使內部變數在電力中斷或作業開始時也可以保留它的值(模式會從 PROGRAM (程式)轉變為 RUN (執行)或 MONITOR (監控)模式)。

當"保留選項"有選取時，變數的值會在電力中斷或作業開始時被保留下來直到 CPU 模組備份電池沒電為止。如果 CPU 模組的電池不良，則數值將會不穩定。

變數	條件	狀態
變數設定為 *Retain (保留)*	作業開始	保留
	電源開啟	保留

當"保留選項"沒有選取時，在電力中斷或作業開始時將不會保留變數的值。不過，即使變數沒有保留，仍可以在作業開始時透過將 IOM 保留位元(A50012)轉為 ON 來保留，且可以透過設定"PLC 設定"在電力中斷時保留，如下表所示。

變數	條件	IOM 保留位元(A50012)狀態		
		OFF	ON	
			啟動時 IOM 保留位元狀態 (PLC 設定)有選取	啟動時 IOM 保留位元狀態 (PLC 設定)無選取
變數不設定為 *Retain (保留)*	作業開始	沒有保留	保留	保留
	電源開啟	沒有保留	保留	沒有保留

備註 支援 IOM 保留位元(A50012)以便與過去的機型相容。不過，要保留功能區塊中的變數值時，請使用*"保留選項"*而不是 IOM 保留位元。

初始值

可以針對一個沒有被保留的輸入變數設定一個起始值(例如，當"保留選項"沒有選取時)。如果"保留選項"已經選取則不能針對輸入變數設定一個起始值。

沒有被保留的內部變數將會被起始化而成為 0。

如果 IOM 保留位元(A50012)為 ON，則起始值將不會被寫入內部變數。

附屬區域控制位元		初始值
IOM 保留位元(A50012)	ON	起始值將不會設定。
	OFF	起始值將會設定。

■ **外部變數**

外部變數就是已經事先登錄到 CX-Programmer 中的系統定義變數，或者是全域符號表中外部參照使用者定義變數的變數。

- 關於系統定義變數的詳細資訊，請參考*附錄 C 外部變數*。
- 要參照全域符號表中的使用者定義變數時，變數必須利用與外部變數相同的變數名稱及資料類型登錄到全域符號表中。

變數屬性

變數名稱

變數名稱用來在功能區塊識別變數。名稱長度最多可達 30,000 個字元。相同的名稱可以用在其他功能區塊中。

備註 即使有 AT 設定(指定的位址)，也必須針對變數輸入一個變數名稱。

資料型態

可以使用下列任何類型。

資料型態	目錄	大小	輸入	輸出	內部
BOOL	位元資料	1 位元	OK	OK	OK
INT	整數	16 位元	OK	OK	OK
UNIT	無正負號整數	16 位元	OK	OK	OK
DINT	二位整數	32 位元	OK	OK	OK
UDINT	無正負號雙重整數	32 位元	OK	OK	OK
LINT	長(4 個字)整數	64 位元	OK	OK	OK
ULINT	無正負號長整數(4 個字)	64 位元	OK	OK	OK
WORD	16 位元資料	16 位元	OK	OK	OK
DWORD	32 位元資料	32 位元	OK	OK	OK
LWORD	64 位元資料	64 位元	OK	OK	OK
REAL	實數	32 位元	OK	OK	OK
LREAL	長實數	64 位元	OK	OK	OK
TIMER	計時器(請參閱備註 1.)	旗標：1 位元 PV：16 位元	不支援	不支援	OK
COUNTER	計數器(請參閱備註 2。)	旗標：1 位元 PV：16 位元	不支援	不支援	OK

備註 (1) TIMER (計時器)資料類型可用來在計時器指令(TIM、TIMH 等)的運算元中輸入計時器號碼(0 到 4095)的變數。當這個變數用在另一個指令時，則會指定計時器完成旗標(1 位元)或計時器現行值(16 位元) (根據指令運算元)。TIMER 資料類型不能用在結構化文字功能區塊中。

(2) COUNTER (計數器)資料類型可用來在計數器指令(CNT、CNTR 等)的運算元中輸入計數器號碼(0 到 4095)的變數。當這個變數用在另一個指令時，則會指定計數器完成旗標(1 位元)或計數器現行值(16 位元) (根據指令運算元)。COUNTER 資料類型不能用在結構化文字功能區塊中。

AT 設定(分配給一個實際位址)

在內部變數方面，也可以針對一個特定的 I/O 記憶體位址設定變數而不由系統自動分配。要指定一個特定位址時，使用者可以在這個屬性中輸入想要的 I/O 記憶體位址。即使指定有一個特定位址，仍需要在程式編輯使用變數名稱。

備註 (1) AT 屬性只能針對內部變數設定。

(2) AT 設定只能搭配 CIO (核心 I/O 區域)、A (附屬區域)、D (資料記憶區)、E (執行的記憶區)、H (保留繼電器區域)、W (內部繼電器區域)使用。
下列記憶區中不能設定 AT 屬性：

- 指標暫存器及資料暫存器區域(直接/間接指定)
- 間接指定 DM/EM (：二進位模式，*：BCD 模式)

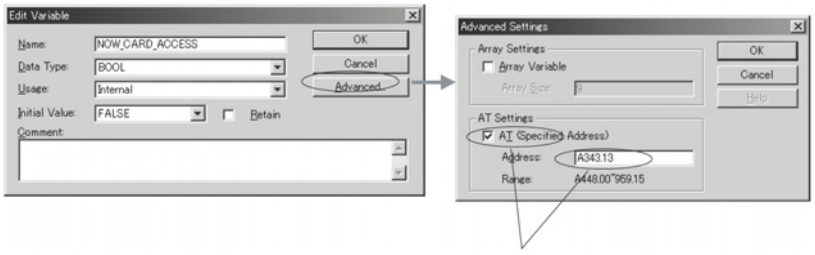
(3) AT 設定可以用於下列分配。

- 基本 I/O 模組、CPU 匯流排模組、或特殊 I/O 模組的位址
- 沒有事先登錄做為外部變數的附屬區域位元
- 網路中其他節點的 PLC 位址
- 指令運算元會指定多重字組的開頭字組(或節尾字組)。

範例程式：

如果"讀取資料檔案"指令(FREAD)用在功能區塊定義中且需要檢查檔案記憶體作業旗標(A34313)，請在 AT 設定中使用一個內部變數並指定旗標的位址。

請登錄一個內部變數，選取 AT 設定選項，並指定 A34313 做為位址。檔案記憶體作業旗標的狀態可以透過這個內部變數來檢查。



位址 A34313 會被分配給名稱為
NOW_CARD_ACCESS 的 Boolean 內部變數。

在使用 AT 設定時，功能區塊會失去它的彈性。這個功能應只在必要時使用。

陣列設定

在內部變數方面，可以將一個變數定義為一個陣列。

備註 CX-Programmer 版本只支援一次元陣列。

在陣列設定方面，可以只登錄一個變數來使用大量具有相同屬性的變數。

- 一個陣列可以有 1 到 32,000 個陣列元素。
- 陣列設定只能針對內部變數設定。
- 可以針對一個陣列變數指定任何資料類型，只要它是一個內部變數即可。
- 在功能區塊定義的演算法則中輸入一個陣列變數名稱時，請在方括弧中的變數名稱輸入陣列指標號碼。下列 3 種方法可以用來指定指標。(在這個例子中陣列變數是一個[]。)
  - 直接使用號碼(階梯或 ST 語言程式編輯)
    範例：a[2]
  - 使用一個變數(階梯或 ST 語言程式編輯)
    範例：[n]，其中 n 是一個變數

  備註 INT、DINT、LINT、UINT、UDINT、或 ULINT 可以用來做為變數資料類型。

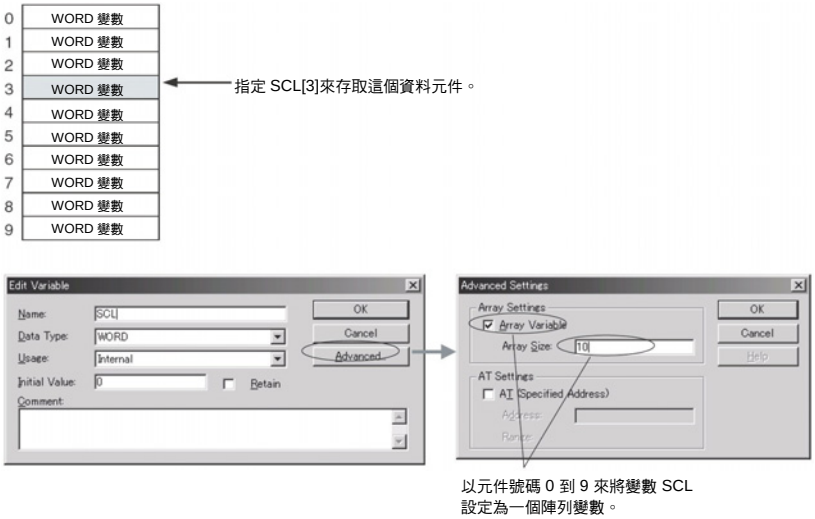
  - 使用一個算式(僅限於 ST 語言程式編輯)
    範例：a[b+c]，其中 b 和 c 是一個變數

  備註 算式可以包含任何算術運算子(+、-、*、及/)。

一個陣列就是一個相同資料類型的資料元件集。每個陣列元件會指定有相同的變數名稱及一個獨特的指標。(指標包括元件在陣列中的位置。)

一次元陣列就是只有一個指標號碼的陣列。

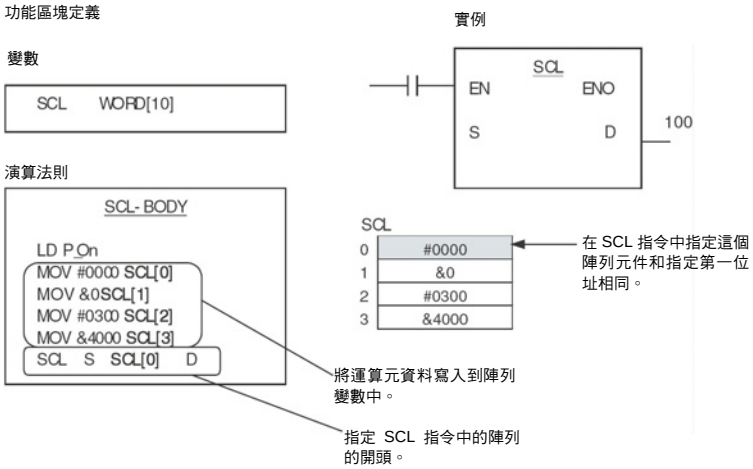
範例：當一個名稱為 SCL 的內部變數設定做為有 10 個元件的陣列變數時，可以使用下列 10 種變數：
SCL[0]、SCL[1]、SCL[2]、SCL[3]、SCL[4]、SCL[5]、SCL[6]、SCL[7]、SCL[8]及 SCL[9]



備註 如果一個具有 AT 屬性的內部變數不能針對運算元設定且無法設定一個外部變數，在指定一個指令運算元中的多重字組的第一或最後一個元件時，請使用一個內部陣列變數以便可以重複使用該功能區塊。準備一個元件數為所需大小的

內部陣列變數，並在每個陣列元件中設定資料後，指定運算元的陣列變數中的第一或最後一個元件。

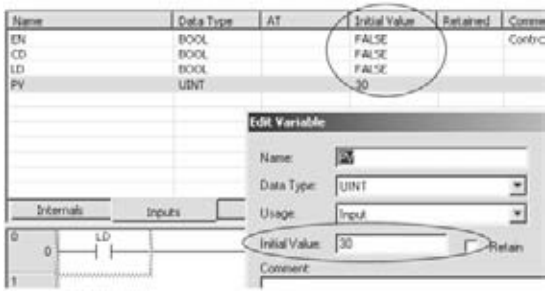
範例程式：



備註 詳細資訊，請參考 *2-5 以運算元指定多重字組的開頭或結尾的指令的注意事項*。

起始值

當一個實例首次執行時，可以針對輸入變數、內部變數、及輸出變數設定起始值。詳細資訊，請參考前述輸入變數、內部變數、及輸出變數下的*起始值*。



保留資料渡過電力中斷及作業開始

內部變數的值可以保留渡過電力中斷及作業開始。當"保留選項"有選取時，變數會被分配給一個在電力中斷及 PLC 作業開始時會保留的記憶區。

演算法則

輸入使用登錄的變數的程式編輯邏輯。

運算元輸入限制

位址不能直接輸入功能區塊中的指令運算元中。直接輸入的位址會被視為變數名稱。

備註 例外：請針對指標暫存器 IR0 到 IR15 直接或間接輸入以及針對資料暫存器 DR0 到 DR15 直接輸入指定的位址到指令運算元中。不要輸入變數。

請直接輸入常數到指令運算元中。

- 階梯圖程式編輯語言：在#後輸入十六進位數值以及在&後十進位數值。

- 結構化文字(ST 語言)：在 16#後輸入十六進位數值以及在同樣符號後輸入十進位數值。

註解

一個註解的長度最多可輸入 30,000 個字元。

2-2 實例規格

2-2-1 一個實例的組成

下表列出使用者在登錄一個實例時必須設定的項目。

項目	說明
實例名稱	實例的名稱
語言變數定義	程式編輯及變數與功能區塊定義完全相同。
功能區塊實例區	變數所使用的位址的範圍
註解	每個實例可以輸入一個註解。

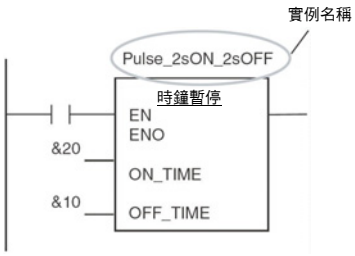
實例名稱

這是實例的名稱。

- 實例名稱長度最多可達 30,000 個字元。
- 實例名稱不能包含空格或下列任何字元：
  ! " # $ % & ' ( ) = - ~ ^ \ | ' @ { [+ ; * :]] < , > . ? /
- 實例名稱不能以數字(0 到 9)開頭。

此外沒有其他的限制。

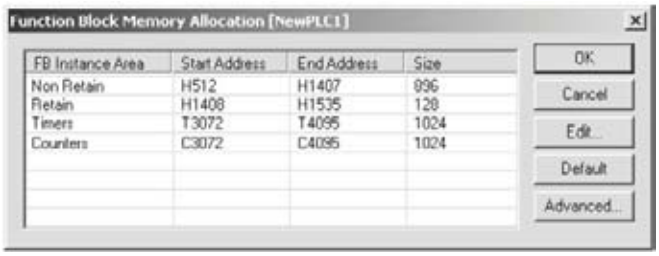
實例名稱顯示在圖中的實例上方。



功能區塊實例區

要使用一個功能區塊時，系統會要求記憶體儲存實例的內部變數及 I/O 變數。這些區域稱為功能區塊實例區，使用者必須指定第一位址和這些區域的大小。第一位址和區域大小可以以 1 個字組的單位來指定。

當 CX-Programmer 匯集這個功能時，如果使用者程式中有任何指令惠存取這些區域中的字組，它會輸出一個錯誤。



版本 3.0 以上的 CS/CJ 系列 CPU 模組及 NSJ 控制器

FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留	H512	H1407	896	CIO, WR, HR, DM, EM
保留	H1408	H1535	128	HR, DM, EM
計時器	T3072	T4095	1024	TIM
計數器	C3072	C4095	1024	CNT

FQM1 彈性位置控制器

FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留	5000	5999	1000	CIO, WR, DM
保留	無			
計時器	T206	T255	50	TIM
計數器	C206	C255	50	CNT

功能區塊實例區類別

下列設定必須在功能區塊實例區中進行：

版本 3.0 以上的 CS/CJ 系列 CPU 模組

非保留區域

項目	內容
分配的變數	電力中斷及作業開始的保留屬性被設定為不保留的變數(請參閱備註 1)。
適用區域	H (功能區塊特別保留區域)、I/O (CIO 區域)、H (保留區域)、W (內部繼電器區域)、D (資料記憶區) (請參閱備註 2)、E (擴充資料記憶區) (請參閱備註 2 及 3。)
設定單位	字組的設定
分配的字組(預設)	H512 至 H1407

備註 (1) 當資料類型被設定為 TIMER (計時器)或 COUNTER (計數器)時除外。

(2) 即使指定DM或EM區域用於非保留區域或保留區域，位元資料仍可以存取。

(3) 如果指定 EM 區域用於非保留區域或保留區域，則相同的記憶庫號碼不能指定為使用者程式中的目前記憶庫。

保留區域

項目	內容
分配的變數	電力中斷及作業開始的保留屬性被設定為不保留的變數(請參閱備註 1)。
適用區域	H (功能區塊特別保留區域)、H (保留區域)、D (資料記憶區) (請參閱備註 2)、E (擴充資料記憶區) (請參閱備註 2 及 3。)
設定單位	字組的設定
分配的字組(預設)	H1408 至 H1535

備註 (1) 當資料類型被設定為 TIMER (計時器)或 COUNTER (計數器)時除外。

(2) 即使指定DM或EM區域用於非保留區域或保留區域，位元資料仍可以存取。

(3) 如果指定 EM 區域用於非保留區域或保留區域，則相同的記憶庫號碼不能指定為使用者程式中的目前記憶庫。

計時器區域

項目	內容
分配的變數	將 TIMER(計時器)設定為資料類型的變數。
適用區域	T (計時器區域)計時器旗標(1 位元)或計時器 PV (16 位元)
分配的字組(預設)	T3072 至 T4095 計時器旗標(1 位元)或計時器 PV (16 位元)

計數器區域

項目	內容
分配的變數	將 COUNTER (計數器)設定為資料類型的變數。
適用區域	C (計數器區域)計數器旗標(1 位元)或計數器 PV (16 位元)
分配的字組(預設)	C3072 至 C4095 計數器旗標(1 位元)或計數器 PV (16 位元)

功能區塊保留區域(H512 至 H1535)

設定為保留及非保留字組的預設功能區塊保留區域配置字組為 H512 到 H1535。這些字組與用於程式的標準保留區域(H000 到 H511)不同，並且只用於功能區塊實例區(內部分配的變數區)。

- 這些字組不能在內部變數的 AT 設定中指定。
- 這些字組不能指定為指令運算元。
  - 如果在一個功能區塊沒有建立時輸入這些字組，它們會顯示為紅色。
  - 雖然字組可以在建立一個功能區塊時輸入，但在檢查程式時會出現一個錯誤。
- 如果這個區域指定為非保留，則在開始作業時會開啟/關閉電源或清除這個區域而不會保留數值。

備註 為避免實例區位址與程式中所使用的位址重疊，請為非保留區域及保留區域設定 H512 到 H1535 (功能區塊保留區域字組)。

FQM1 彈性位置控制器

FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留	5000	5999	1000	CIO, WR, DM
保留	無			
計時器	T206	T255	50	TIM
計數器	C206	C255	50	CNT

非保留區域

項目	內容
分配的變數	電力中斷及作業開始的保留屬性被設定為不保留的變數(請參閱備註 1)。
適用區域	I/O (CIO)、W (工作區域)、及 D (DM 區域) (請參閱備註 2)。
設定單位	字組的設定
分配的字組(預設)	CIO 5000 至 CIO 5999

備註 (1) 當資料類型被設定為 TIMER (計時器)或 COUNTER (計數器)時除外。

(2) 即使指定 DM 區域用於非保留區域，位元資料仍可以存取。

保留區域

無

計時器區域

項目	內容
分配的變數	將 TIMER(計時器)設定為資料類型的變數。
適用區域	T (計時器區域)計時器旗標(1 位元)或計時器 PV (16 位元)
分配的字組(預設)	T206 至 T255 計時器旗標(1 位元)或計時器 PV (16 位元)

計數器區域

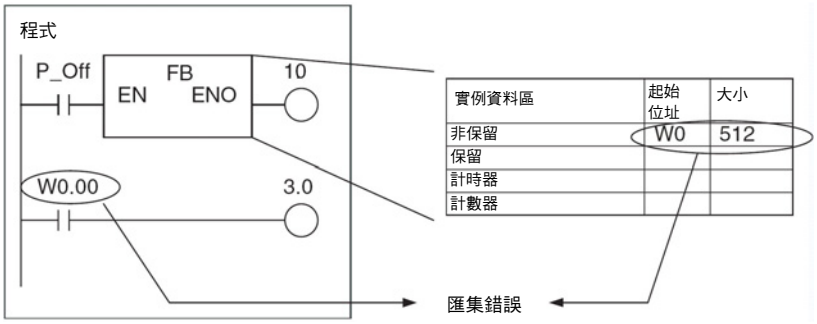
項目	內容
分配的變數	將 COUNTER (計數器)設定為資料類型的變數。
適用區域	C (計數器區域)計數器旗標(1 位元)或計數器 PV (16 位元)
分配的字組(預設)	C206 至 C255 計數器旗標(1 位元)或計數器 PV (16 位元)

從使用者程式存取功能區塊實例區

如果使用者程式包含有一個指令會存取功能區塊實例區，如果嘗試進行下列操作時，CX-Programmer 的輸出視窗的"匯集"索引標籤中會顯示一個錯誤。

- 嘗試在線上編輯中寫入(不可寫入)
- 執行程式檢查(選取從程式主選單 Compile (匯集)或從 PLC 主選單選取 Compile All PLC Programs (匯集所有 PLC 程式))

範例：如果將 W0 到 W511 指定為功能區塊實例區的非保留區域並將 W0.00 用於階梯圖程式中，則在匯集時會出現一個錯誤並且會顯示為"ERROR：[省略]…-位址 - W0.00 is reserved for Function Block use (保留供功能區塊使用)]。



備註 功能區塊實例區中的變數配置會在新增或刪除變數時自動重新分配。不過，一個實例會要求位址必須依照順序，所以如果無法取得符合順序的位址，則所有變數都會被分配不同的位址。因此，會建立未使用的區域。如果這種情況發生，請執行最佳化作業來有效的使用分配的區域並移除未使用的區域。

註解

一個註解的長度最多可輸入 30,000 個字元。

建立多個實例

叫出相同的實例

一個實例可以從多個位置叫出。在這種情況下，會共用內部變數。

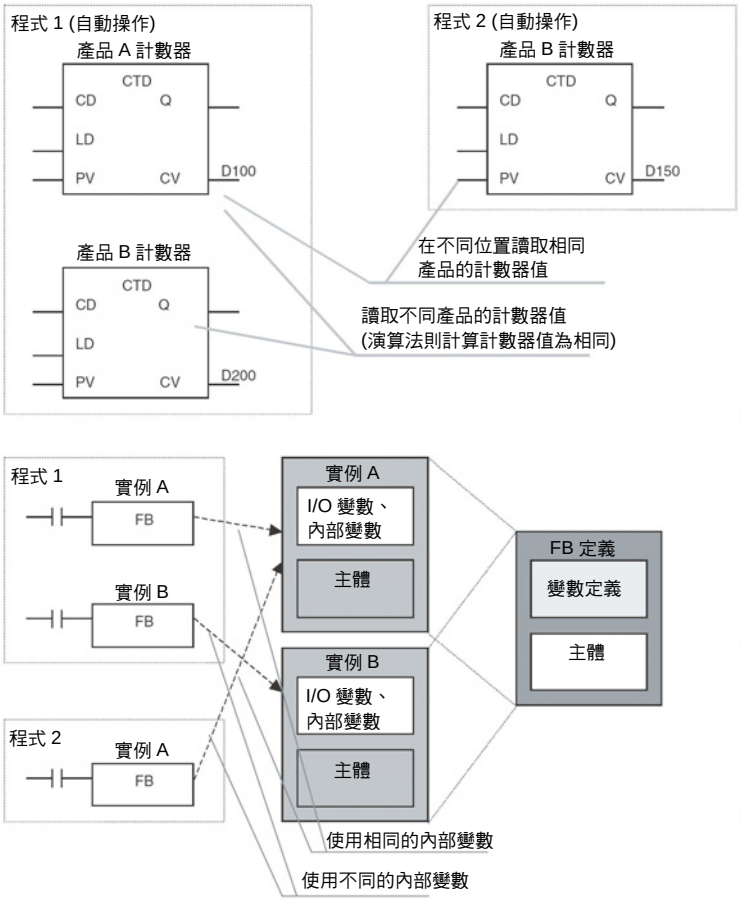
建立多個實例

可以從一個單一功能區塊定義建立多個實例。在這種情況下，每個實例中的內部變數的值都會不同。

範例：計數產品 A 及產品 B

準備一個稱為“倒數計數器"(CTD)的功能區塊定義，針對產品 A 及產品 B 設定計數器。有兩種程式一種用來進行自動操作而另一種用來進行手動操作。使用者可以切換到適當的操作模式。

在這種情況下，會根據一個功能區塊建立多個實例。同一個實例必須可從多個位置叫出。



2-2-2 參數規格

可以由使用者在輸入參數及輸出參數中設定的資料如下：

項目	適用資料
輸入參數	值(請參閱備註 1)、位址、及程式符號(全域符號及區域符號) (請參閱備註 2)。 **備註** 會從參數轉送給輸入變數的資料，是輸入變數資料大小的實際值。(即使參數中設定有一個位址，位址本身也不會轉送。) **備註** 必須設定輸入參數。如果即使有一個輸入參數沒有設定，也會發生致命的錯誤並且輸入參數將不會傳送給實際的 PLC。
輸出參數	位址、程式符號(全域符號、區域符號) (請參閱備註 2。)

備註 (1) 如下表顯示在參數中輸入值的方法。

輸入變數資料類型	目錄	大小	參數值輸入方法	設定範圍
BOOL	位元資料	1 位元	P_Off, P_On	0 (FALSE)、1 (TRUE)
INT	整數	16 位元	正值：&或+後隨整數 負值：-後隨整數	-32,768 至 32,767
DINT	二位整數	32 位元		-2,147,483,648 至 2,147,483,647
LINT	長(8 位元)整數	64 位元		-9,223,372,036,854,775,808 至 9,223,372,036,854,775,807
UINT	無正負號整數	16 位元	正值：&或+後隨整數	&0 至 65,535
UDINT	無正負號雙重整數	32 位元		&0 至 4,294,967,295
ULINT	無正負號長(8 個字)整數	64 位元		&0 至 18,446,744,073,709,551,615
REAL	實數	32 位元	正值：&或+後隨實數(有小數點) 負值：-後隨實數(有小數點)	-3.402823 × 10^{38} 至−1.175494 × 10^{-38}、0、1.175494 × 10^{-38} 至 3.402823 × 10^{38}
LREAL	Long 實數	64 位元		-1.79769313486232 × 10^{308} 至 −2.22507385850720 × 10^{-308}、0、2.22507385850720 × 10^{-308}、1.79769313486232 × 10^{308}
WORD	16 位元資料	16 位元	#後隨十六進位數(最大 4 位數) &或+後隨十進位數	#0000 至 FFFF 或&0 至 65,535
DWORD	32 位元資料	32 位元	#後隨十六進位數(最大 8 位數) &或+後隨十進位數	#00000000 至 FFFFFFFF 或&0 至 4,294,967,295
LWORD	64 位元資料	64 位元	#後隨十六進位數(最大 16 位數) &或+後隨十進位數	#0000000000000000 至 FFFFFFFFFFFFFFFF 或&0 至 18,446,744,073,709,551,615

(2) 功能區塊輸入變數及輸出變數的大小必須符合程式符號(全域及區域)，如下表所示。

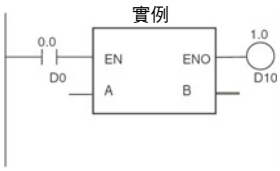
大小	功能區塊變數資料類型	程式符號(全域、區域)資料類型
1 個位元	BOOL	BOOL
16 個位元	INT, UINT, WORD	INT, UINT, UINT BCD, WORD
32 個位元	DINT, UDINT, REAL, DWORD	DINT, UDINT, UDINT BCD, REAL, DWORD
64 個位元	LINT, ULINT, LREAL, LWORD	LINT, ULINT, ULINT BCD, LREAL, LWORD
1 個位元以上	非 Boolean	途徑、號碼(請參閱備註)

備註 程式符號號碼只能設定在輸入參數中。輸入的值必須在功能區塊變數資料類型的大小範圍內。

2-2-3 作業規格

叫出實例

使用者可以從任何位置叫出一個實例。實例會在轉送給 EN 的輸入為 ON 時執行。

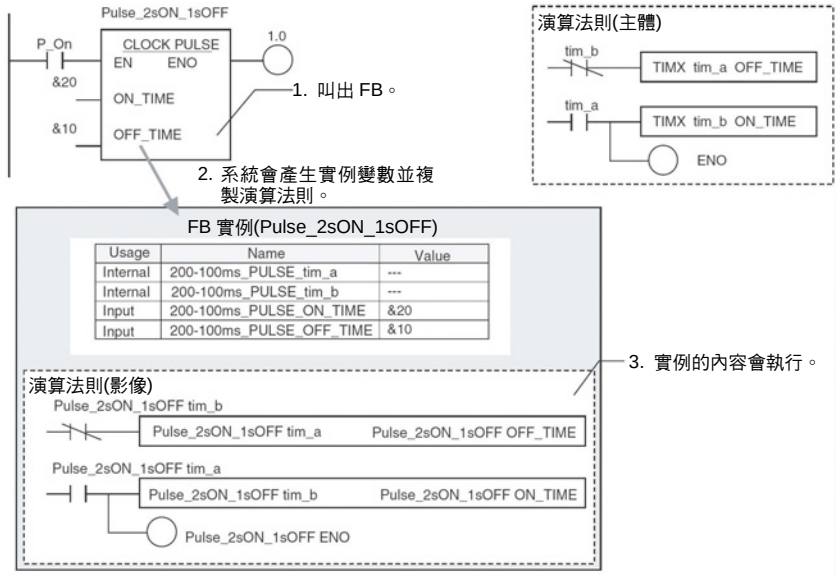


在這種例子中，轉送給 EN 的輸入是圖左的位元 0.0。

- 當轉送給 EN 的輸入為 ON 時，實例會執行且執行的結果會反映在位元 1.0 及字組 D10 中。
- 當轉送給 EN 的輸入為 OFF 時，實例不會執行，位元 1.0 會轉為 OFF，且 D10 的內容不會改變。

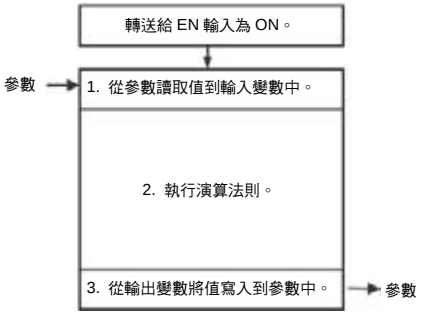
實例執行時的操作

系統會在轉送給功能區塊 EN 輸入變數的輸入為 ON 時叫出一個功能區塊。當功能區塊被叫出時，系統會產生實例的變數並複製登錄在功能區塊中的演算法則。實例接著就會執行。



執行的順序如下：

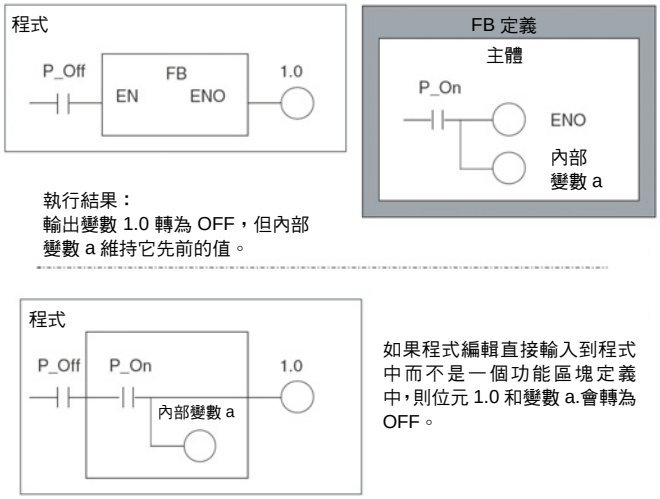
1. 從參數讀取資料到輸入變數中。
2. 執行演算法則。
3. 從輸出變數將資料寫入到參數中。



資料不能在演算法則本身中與參數交換。另外，如果一個輸出變數沒有因執行演算法則而改變，則輸出參數會維持先前的值。

實例不執行時的操作

當轉送給功能區塊 EN 輸入變數的輸入為 OFF 時，功能區塊不會被叫出，所以實例的內部變數不會改變(值會被保留)。同樣的，當 EN 為 OFF 時，輸出變數也不會改變(值會被保留)。

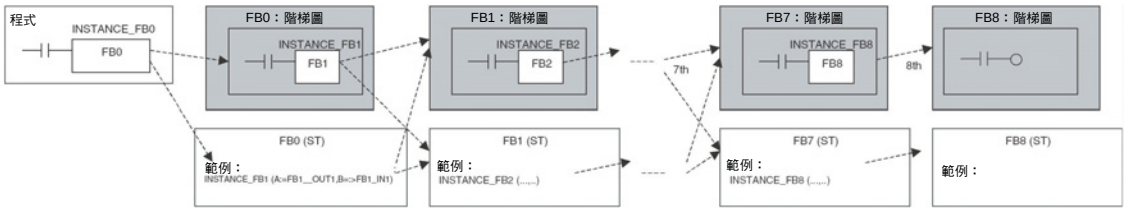


⚠ **注意** 一個實例在它的 EN 輸入變數為 OFF 時將不會被執行，所以“區別"和“計時器"指令在 EN 為 OFF 時都不會被起始化。如果使用“區別"或“計時器"指令，EN 的輸入條件請務必使用 ON 旗標(P_On)並包括功能區塊定義範圍內的指令輸入條件。

巢狀

在 CX-Programmer 6.0 版以上的版本方面，一個功能區塊可以從其他的功能區塊叫出，即支援巢狀。功能區塊最多可以巢狀 8 層(包括從程式叫出的功能區塊)。

呼叫功能區塊和被叫出的功能區塊可以是 ST 語言、階梯語言、或兩者的組成。



“INSTANCE_FB1”、“INSTANCE_FB2”等是“功能區塊”資料類型實例名稱。
備註：被呼叫功能區塊與呼叫功能區塊之間可以使用任何階梯圖與結構化文字的程式編輯組成。

功能區塊的巢狀階層也可以顯示在一個具有 FB 實例檢視器功能的樹狀目錄格式中。

巢狀的功能區塊的功能區塊定義，會包括在包含有呼叫功能區塊定義的功能區塊資料庫檔案(.cxf)中。

2-3 功能區塊限制

階梯圖程式編輯限制

階梯圖程式中所使用的指令有一些限制。

功能區塊定義中禁止使用的指令

下列指令不能用在功能區塊定義中。如果使用這些指令中的任何一個，則會出現一個匯集錯誤。

- 區塊程式編輯指令(所有指令，包括 BPRG 及 BEND)
- 副程式指令(SBS、GSBS、RET、MCRO、SBN、GSBN、及 GRET)
- 跳越指令(JMP、CJP、CJPN、及 JME)
- 步驟指令(STEP 與 SNXT)
- 立即更新指令(!)
- I/O 更新指令(IORF)
- TMHH 和 TMHHX 教學
- CV 位址轉換指令(FRMCV 及 TOCV)
- 處理記錄位置的指令(PUSH、FIFO、LIFO、SETR、及 GETR)
- 失效點偵測指令(FPD)
- 將計時器/計數器 PV 移到暫存器指令(MOVRW)

AT 設定限制 (不支援的資料區)

下列區域中的位址不能用於 AT 設定。

- 指標暫存器(不支援間接及直接編址)及資料暫存器

  備註 請直接輸入位址，而不是 AT 設定。
- DM 或 EM 區域位址的間接編址(不支援二進位模式也不支援 BCD 模式間接編址。)

指令運算元中的 I/O 記憶體的直接編址

- 位址(而不是變數)可以直接輸入到指標暫存器(包括間接及直接編址)和資料暫存器中。
  下列值可以輸入到指令運算元中：
  直接編址：IR0 到 IR15；間接編址：,IR0 到,IR15；常數差值(範例)：+5,IR0；DR 差值：DR0、IR0；自動增量：,IR0++；自動減量：--,IR0
- I/O 記憶體中的任何其他區域不支援指令運算元中的直接編址。

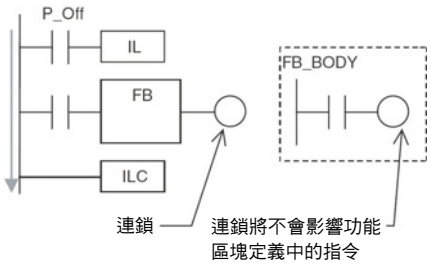
I/O 變數限制 (不支援的資料區)

下列資料區中的位址不能用來做為輸入及輸出變數的參數。

- 指標暫存器(不支援間接及直接編址)及資料暫存器
- DM 或 EM 區域位址的間接編址(不支援二進位模式也不支援 BCD 模式間接編址。)

連鎖限制

當一個功能區塊被從一個連鎖的程式區段叫出時，功能區塊定義的內容將不會被執行。連鎖的功能區塊的行為將只會像是一個連鎖的副程式一樣。

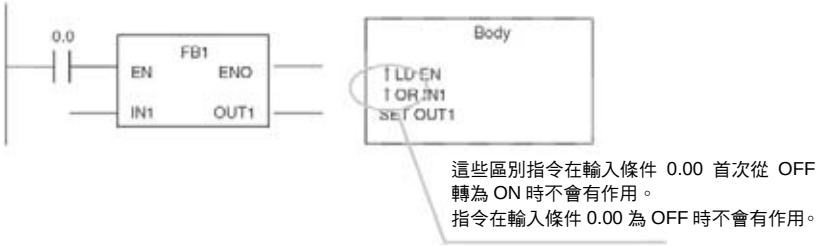


功能區塊定義中的區別指令

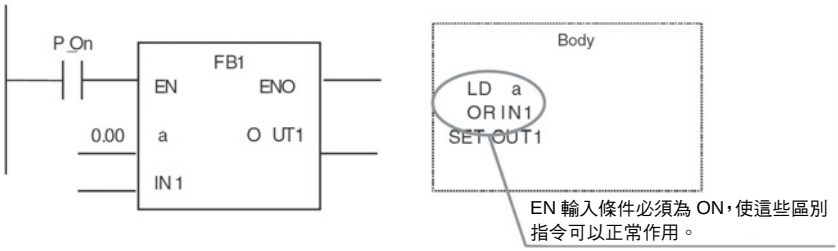
一個實例在它的 EN 輸入變數為 OFF 時將不會被執行，所以在使用功能區塊定義中的一個區別指令時，必須注意下列注意事項。(區別指令包括 DIFU、DIFD、及任何有@或%字首的指令。)

- 只要實例的 EN 輸入變數為 OFF，執行條件將會維持它先前的狀態(最後一次 EN 輸入變數為 ON 時的狀態)且區別指令將不會有作用。
- 當實例的 EN 輸入變數轉為 ON 時，現行的執行狀況的狀態將不會與上一個循環的狀態進行比較。現行執行狀況將會與最後一次 EN 輸入變數為 ON 時的狀況進行比較，所以區別指令將不會正常作用。(如果 EN 輸入變數保持為 ON，則區別指令會在下次上升界限或下降界限發生時正確作用。)

範例程式：



如果使用“區別"指令，EN 的輸入條件請務必使用 ON 旗標(P_On)並包括功能區塊定義範圍內的指令輸入條件。



- 在指定下列指令的第一運算元時，請在“#"後輸入一個十進位數值。
  MILH(517), MILR(518), MILC(519), DIM(631), MSKS(690), MSKR(692), CLI(691), FAL(006), FALS(007), TKON(820), TKOF(821)

  備註 不支援“&"。

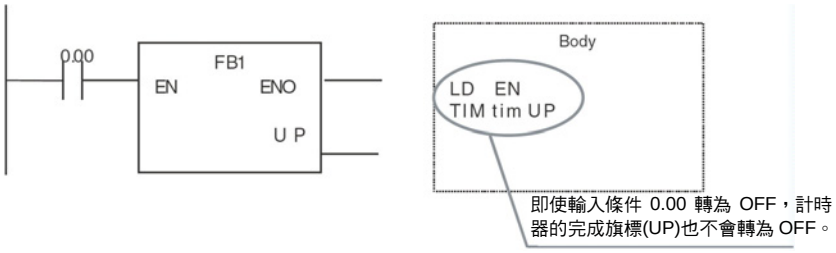
- CNR(545)、CNRX(547) (重置計時器/計數器)指令不能用來同時重置一個功能區塊中的多個計時器和計數器。

  第一運算式(計時器/計數器號碼 1)及第二運算式(計時器/計數器號碼 2)請務必指定相同的變數。第一及第二運算式不能指定不同的變數。

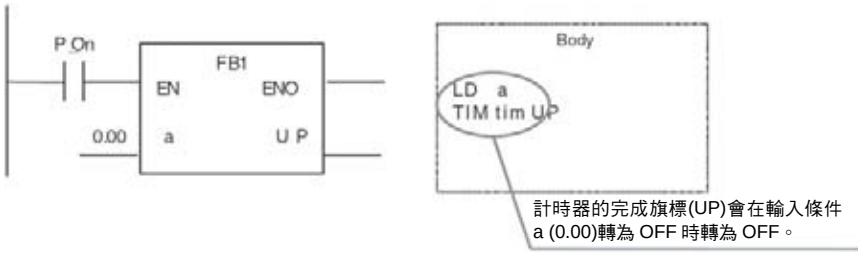
功能區塊定義中的計時器指令

一個實例在它的 EN 輸入變數為 OFF 時將不會被執行，所以在使用功能區塊定義中的一個計時器指令時，必須注意下列注意事項。

即使實例的 EN 輸入變數轉為 OFF，計時器指令也不會被起始化。因此，如果 EN 輸入變數在計時器開始作用後轉為 OFF，計時器的完成旗標將不會轉為 OFF。



如果使用"計時器"指令，EN 的輸入條件請務必使用 ON 旗標(P_On)並包括功能區塊定義範圍內的指令輸入條件。



- 如果包含有一個計時器的同一個實例同時用於多個位置，則計時器會被複製。

ST 程式編輯限制

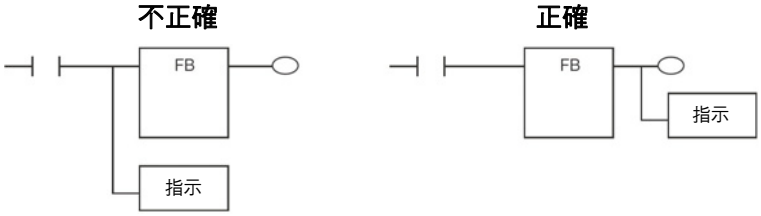
- 僅支援下列敘述及運算數。
  - 指定敘述
  - 選擇敘述(CASE 及 IF 敘述)
  - 反覆敘述(FOR、WHILE、REPEAT、及 EXIT 敘述)
  - RETURN 敘述
  - 功能區塊呼叫敘述
  - 算術運算數
  - 邏輯運算數
  - 比較運算數
  - 數值函數
  - 算術函數
  - 註解
- 不能使用計時器及計數器資料類型。

更詳細的資訊，請參考*附錄 B 結構化文字(ST 語言)規格*。

程式結構注意事項

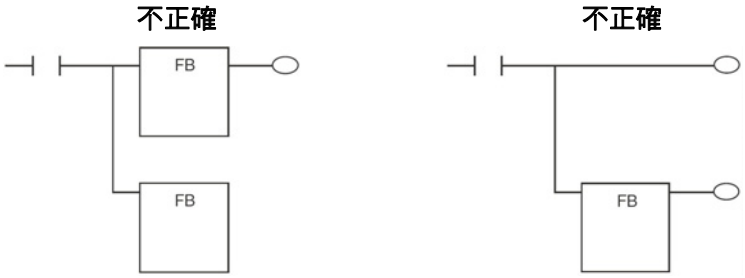
分支不能在實例的左側

分支不允許在實例的左側。分支允許在右側。



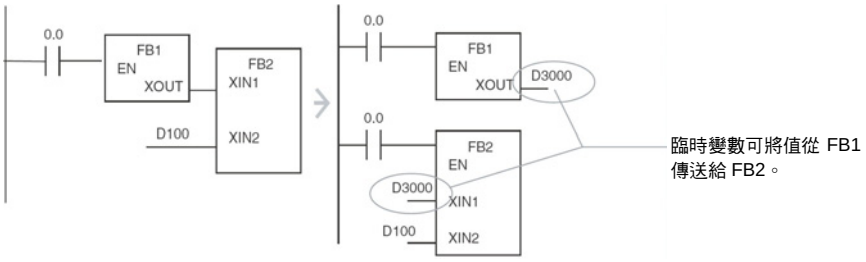
每個梯級只有一個實例

一個程式梯級不能有一個以上的實例。



沒有功能區塊連結

一個功能區塊的輸入不能連結到其他功能區塊的輸出。在這種情況下，必須登錄一個變數來從第一個功能區塊的輸出傳送執行狀態給第二個功能區塊的輸入。



下載到工作模組中

包括功能區塊的工作不能下載到工作模組中，但可以上傳。

程式設計控制台顯示

當一個以 CX-Programmer 建立的使用者程式下載到 CPU 模組並由一個程式設計控制台讀取時，實例都會顯示為一個問號。(實例名稱不會顯示出來。)

線上編輯限制

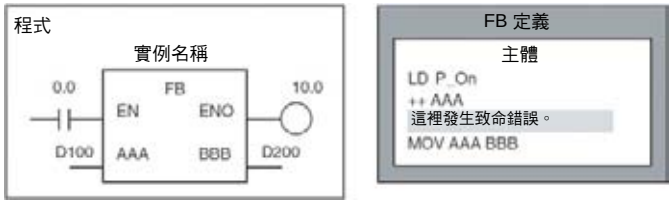
下列線上編輯作業不能在 CPU 模組中的使用者程式上執行。

- 變更或刪除功能區塊定義(變數表或演算法則)
- 插入實例或變更實例名稱

備註 實例的 I/O 參數不能變更，實例可以刪除，且一個實例以外的指令可以變更。

與錯誤有關的限制

如果在一個功能區塊定義被執行時 CPU 模組發生一個致命錯誤，則階梯圖程式的執行將會停止在發生錯誤的位置上。



在這種情況下，MOV AAA BBB 指令將不會執行且輸出變數 D200 會維持與功能區塊執行前相同的值。

禁止存取 FB 實例區

要使用一個功能區塊時，系統會要求記憶體區域儲存實例的內部變數及 I/O 變數。

版本 3.0 以上的 CS/CJ 系列 CPU 模組及 NSJ 控制器

功能區塊實例區	開始位址的起始值	起始值的尺寸	容許的資料區
非保留	H512	896	CIO, WR, HR, DM, EM
保留	H1408	128	HR, DM, EM
計時器	T3072	1,024	TIM
計數器	C3072	1,024	CNT

FQM1 彈性位置控制器

FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留	5000	5999	1000	CIO, WR, DM
保留	無			
計時器	T206	T255	50	TIM
計數器	C206	C255	50	CNT

如果使用者程式中有一個指令會存取一個 FB 實例區中的位址，則 CX-Programmer 會在下列情況下輸出一個錯誤。

- 當使用者從程式主選單選取 ***Program (程式) - Compile (匯集)***或從 *PLC* 主選單選取 ***Compile All Programs (匯集所有程式)***來執行程式檢查時。
- 當嘗試透過線上編輯來寫入程式時(無法寫入)。

2-4 功能區塊應用準則

本節說明透過 CX-Programmer 使用功能區塊的準則。

2-4-1 根據變數資料類型

整數資料類型
(1、2 或 4 字組資料)

在處理 1、2 或 4 字組單位中的一個數字時，請使用下列資料類型。

- INT 與 UINT
- DINT 與 DINT
- LINT 與 ULINT

備註 如果所使用的數字符合範圍，則使用含正負號的整數。

文字資料類型 (1、2 或 4 字組資料)

在處理 1、2 或 4 字組單位中的資料組群(非數字資料)時，請使用下列資料類型。

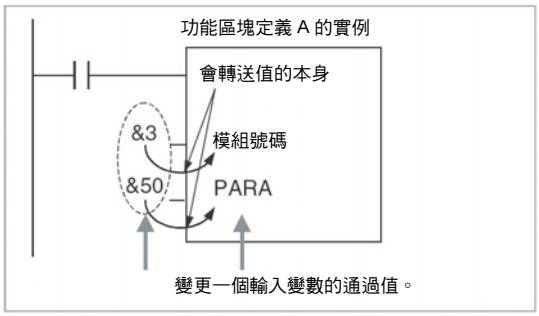
- WORD
- DWORD
- LWORD

2-4-2 決定變數類別(輸入、輸出、外部及內部)

使用輸入變數來變更轉送值

要將一個功能區塊貼到一個程式中，然後變更要針對每個實例轉送給功能區塊的值(不是位址本身)時，請使用一個輸入變數。

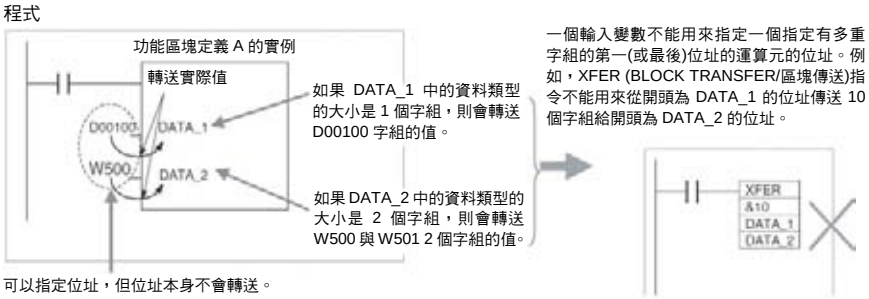
程式



適用下列兩項限制。

- 可以在一個輸入參數中設定一個位址，但一個位址本身不能轉送給一個輸入變數(即使輸入參數中設定有一個位址，也會轉送輸入變數資料類型的大小值給功能區塊)。因此，當功能區塊中的指令運算元中指定有多重字組的第一及最後元件時，不能將輸入變數用於運算元。請指定使用有 AT 設定的內部變數(指定一個內部陣列變數中的第一或最後元件)，或者使用一個外部變數(如 *2-4-4 內部變數的陣列設定*中所述)。

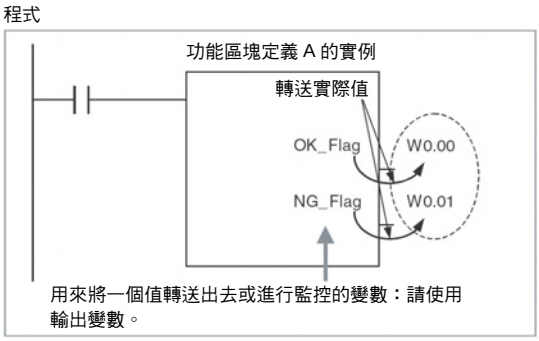
程式



- 在執行運算之前，數值會以批次方式從輸入參數轉送給輸入變數(與執行的運算中的指令並不同時)。因此，在執行功能區塊運算中的一個指令時若要將數值從一個參數轉送給一個輸入變數，請使用一個內部變數或外部變數來取代一個輸入變數。

從輸出變數轉送值或監控輸出變數

要貼到程式中然後將值從功能區塊針對每個實例轉送出去(到程式以外)，或者要監控數值時，請使用輸出變數。



適用下列限制。

- 值只有在演算法則執行後才會從輸出變數轉送給輸出參數。

外部變數：程式中的條件旗標、時序脈衝、附屬區域位元、全域符號

用在程式中的條件旗標(例如永遠 ON 旗標、相等旗標)、時序脈衝(例如 1.0 秒時序脈衝位元)、預先登錄附屬區域位元(例如第一循環旗標)、及全域符號全部都是由系統所定義的外部變數。

內部變數：內部分配的變數及需要 AT 設定的變數

沒有被指定為輸入、輸出或外部的變數都屬於內部。內部變數包括有內部分配位址的變數、要求有 AT 設定的位址(例如 I/O 配置位址、針對特殊 I/O 模組特別分配的位址)的變數、或者要求陣列設定的變數。關於要求 AT 設定或陣列設定的條件的詳細資訊，請參考 *2-4-3 內部變數的 AT 設定*、及 *2-4-4 內部變數的陣列設定*。

2-4-3 內部變數的 AT 設定

在下列情況下請務必指定內部變數的 AT 設定。

- 當使用有分配給基本 I/O 模組、特殊 I/O 模組、或 CPU 匯流排模組的位址且這些位址都登錄為不能指定為外部變數的全域符號(例如全域符號的資料集會不穩定)時。

  備註 指定特殊 I/O 模組配置位址的指標暫存器的方法會要求針對配置區域的第一位址指定 AT 設定。(詳細資訊，請參考 *2-4-5 指定分配給特殊 I/O 模組的位址*。)

- 在使用沒有預先登錄為外部變數的附屬區域位元，且這些位元登錄為沒有指定為外部變數的全域符號時。
- 在設定遠端節點的第一目的字組使用 SEND(090)而本地節點的第一來源字組使用 RECV(098)時。
- 在指令運算元指定有多重字組的第一或最後一字組，且內部陣列變數不能針對運算元指定(例如不能指定陣列元件數)時。

2-4-4 內部變數的陣列設定

使用陣列變數來指定多重字組運算元的第一或最後一字組

在指定一個指令運算元(請參閱備註)中的一個字組範圍的第一或最後一字組時，指令會在 AT 指定或內部分配後根據位址作用。(因此，變數的資料類型及變數的元件數與指令的操作無關。)請務必指定一個有 AT 設定的變數或一個元件數符合要由指令處理的資料大小的陣列變數。

備註 有一些範例是 XFER(070) (BLOCK TRANSFER/區塊傳送)指令的第一來源字組或第一目的字組、SEND(090)的第一來源字組、或者相關指令的控制資料。

詳細資訊，請參考 *2-5 以運算元指定多重字組的開頭或結尾的指令的注意事項*。請使用下列方法來指定一個陣列變數。

1,2,3... 1. 準備一個有所需元件數的內部陣列變數。

備註 請確定要由指令處理的資料大小是否與元件數相同。關於由每個指令處理的資料大小的詳細資訊，請參考 *2-6 指令支援及運算元限制*。

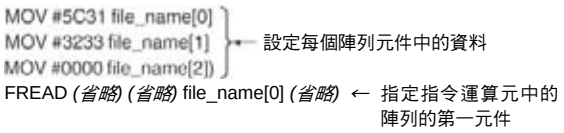
2. 利用功能區塊定義中的 MOV 指令來設定每個陣列元件中的資料。
3. 指定運算元的陣列變數的第一(或最後)元件。這可以指定一個字組範圍內的第一(或最後)位址。

範例如以下所示。

<u>處理多重字組中的資料的單一字串</u>

在這個範例中，一個陣列包含有一個 FREAD 指令的目錄和檔案名稱(運算元 S2)。

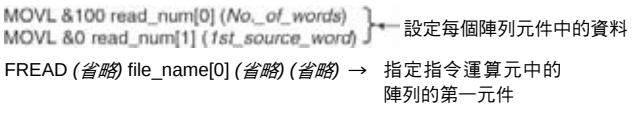
- 變數表
  內部變數、資料類型=WORD、陣列設定有 10 個元件、變數名稱=檔名[0]到檔名[9]
- 階梯圖程式編輯



<u>處理多重字組中的控制資料</u>

在這個範例中，一個陣列包含有一個 FREAD 指令的字組數及第一來源字組(運算元 S1)。

- 變數表
  內部變數、資料類型=DINT、陣列設定有 3 個元件、變數名稱=read_num[0]至 read_num[9]
- 階梯圖程式編輯



處理一個多重字組中的讀取資料的區塊

容許的讀取資料量必須事先決定且必須準備一個可以處理最大資料量的陣列。在這個範例中，一個陣列會接收 FREAD 指令的讀取資料(運算元 D)。

- 變數表
  內部變數、資料類型=WORD、陣列設定有 100 個元件、變數名稱=read_data[0] to read_data[99]
- 階梯圖程式編輯

FREAD *(省略)* *(省略)* *(省略)* read_data[0]

利用整數陣列變數的除法(僅限於階梯圖程式編輯)

一個有 2 個元件的陣列可以用來儲存一個階梯圖程式的含正負號二進位除法(/)指令的結果。指令的結果是 D (商數)和 D+1 (餘數)。這個方法可以用來從階梯圖程式編輯中的一個除法運算取得餘數。

備註 在使用 ST 語言時，並不須要使用一個陣列來接受除法運算的結果。同時，餘數也不能直接在 ST 語言中運算。餘數必須以下列方式計算：
餘數=被除數-(除數 x 商數)

2-4-5 指定分配給特殊 I/O 模組的位址

請使用指標暫存器 IR0 到 IR15 (間接指定常數差值)來根據功能區塊定義中針對做為輸入參數的模組號碼所轉送的值指定分配給特殊 I/O 模組的位址，如下列範例所示。

備註 關於使用功能區塊中的指標暫存器的詳細資訊，請參考 *2-4-6 使用指標暫存器*。

範例

範例 1：指定一個功能區塊中的 CIO 區域(DM 區域也相同)

特殊 I/O 模組

變數：使用模組號碼做為一個輸入變數，並指定地一分配位址做為一個內部變數，將 AT 設定為 CIO 2000。

程式：使用下列程序。

1,2,3...

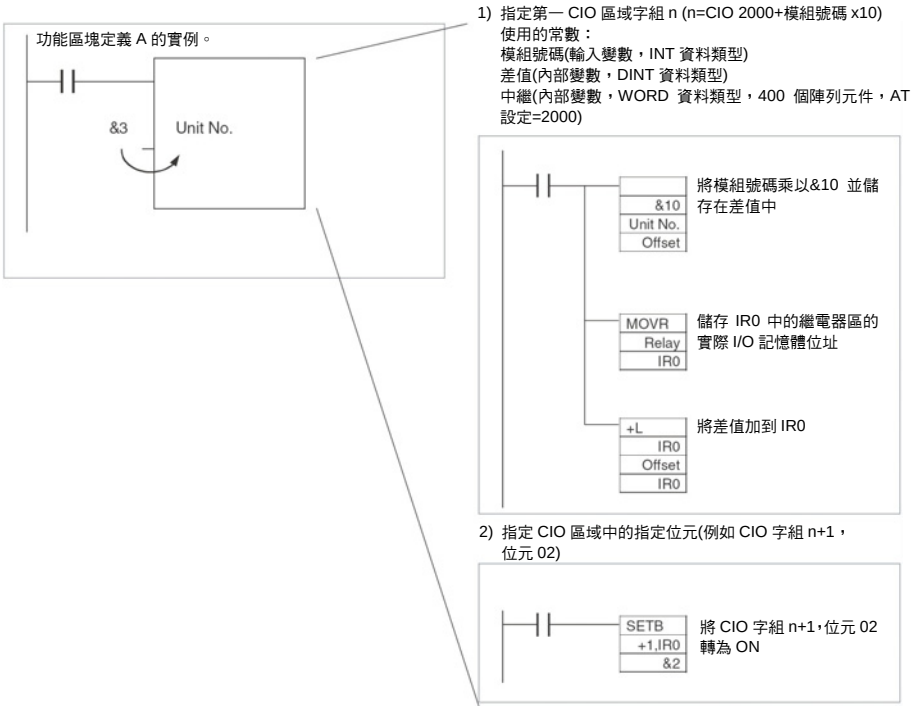
1. 將模組號碼(輸入變數)乘以&10，並建立模組號碼差值(內部變數，DINT 資料類型)。
2. 使用 MOVR(560) (移到暫存器)指令來儲存指標暫存器(例如 IR0)中的第一分配位址(內部變數，AT=CIO 2000)的實際 I/O 記憶體位址。
3. 將模組號碼差值加到指標暫存器中的實際 I/O 記憶體位址(例如 IR0)。

範例 2：指定 CIO 區域中的指定位元(例如 CIO 字組 n+a，位元 b)

程式：使用下列任一種方法。

- 字組位址：利用一個間接指定(例如+a,IR0)來指定指標暫存器的常數差值。
- Bit Addresses (位元位址)：指定一個可以在一個字組中指定一個位元位址的指令(例如寫入時的 SETB 指令以及讀取時的 TST 指令的第二運算元中的&b)。

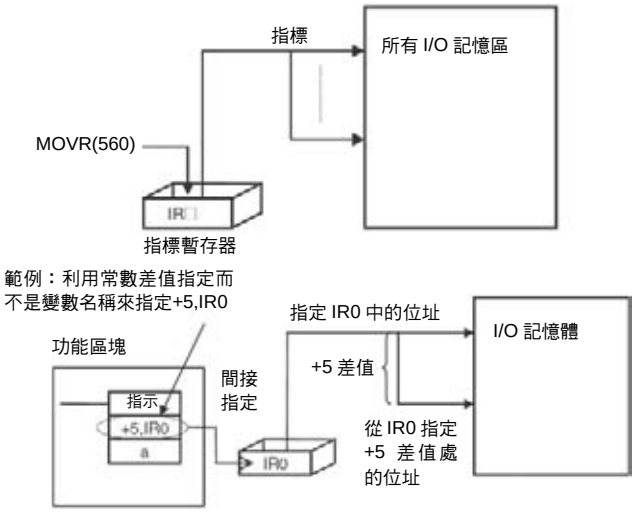
範例：特殊 I/O 模組



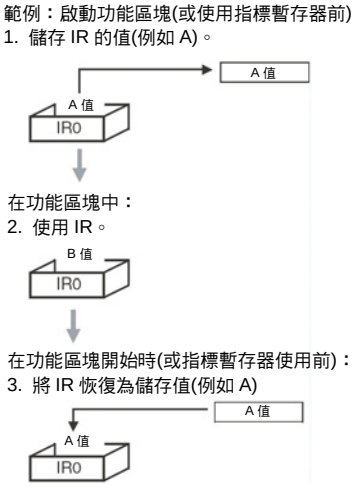
2-4-6 使用指標暫存器

指標暫存器 IR0 到 IR15 的功能是做為指定 I/O 記憶體位址時的指標。這些指標暫存器可以用在功能區塊中來利用 IR0 到 IR15 而不是變數名稱直接指定位址(指標暫存器直接指定：IR0 到 IR15；直接指定中的指標暫存器：,IR0 到,IR15)

備註 在利用 MOVR(560)指令將實際 I/O 記憶體位址儲存在指標暫存器中之後，指標暫存器可以利用一般指令來間接指定。這可讓所有的 I/O 記憶區可以動態指定。



備註 (1) 當指標暫存器 IR0 到 IR15 用在功能區塊中時，使用其他功能區塊中或功能區塊以外的程式中的相同指標暫存器會在兩個實例之間產生競爭且程式將不會正常執行。因此，在使用指標暫存器(IR0 到 IR15)時，請務必將指標暫存器的值儲存在功能區塊開始(或指標暫存器被使用前)的一點，並在功能區塊完成時(或指標暫存器被使用後)，合併程式中的處理來將指標暫存器恢復為儲存值。

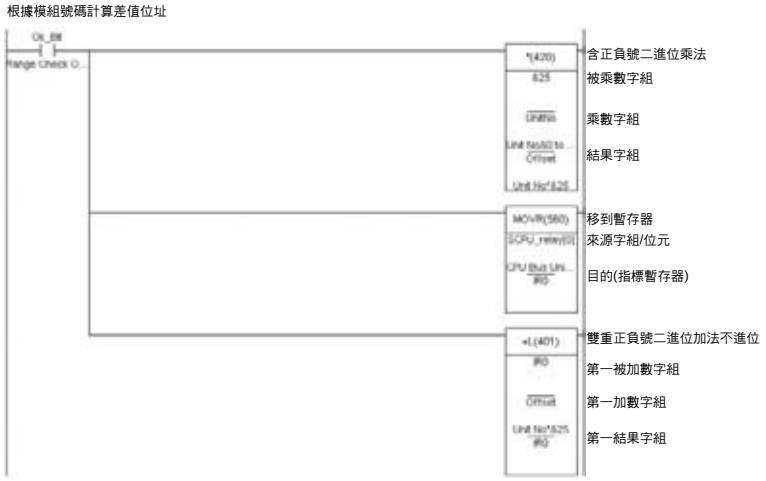


(2) 務必在使用指標暫存器之前設定相關值。如果使用指標暫存器而值沒有設定，操作將會不穩定。

應用範例

以下是使用功能區塊中的指標暫存器 IR0 到 IR15 的範例。

範例	Details (詳細資料)
在使用指標暫存器之前儲存指標暫存器的值 將 IR0 暫時儲存在備份緩衝區中  	當指標暫存器用在這個功能區塊中時，儲存指標暫存器值的處理會在功能開始時(或指標暫存器使用前)執行來讓值可以在功能區塊完成後(或指標暫存器使用後)恢復為原指標暫存器值。 範例：儲存指標暫存器 IR0 的內容，將它儲存在 *SaveIR[0]*(內部變數，資料類型 DINT，1 個陣列元件)中。

範例	Details (詳細資料)
使用指標暫存器 1) 設定指標暫存器中的值。(儲存第一 CIO 區域字組 n 的實際 I/O 記憶體位址) 	範例：以轉送自功能區塊的 CPU 匯流排模組的模組號碼(&0 到&15)為基礎、分配在 CPU 匯流排模組配置區中的 CIO 1500+模組號碼 x25 的第一字組的實際 I/O 記憶體位址會儲存在 IR0 中。 程序： 假設模組號碼&0 到&15 已經(從功能區塊外)輸入到 *UnitNo* (輸入變數，INT 資料類型)中。 1. 將 *UnitNo* 乘以*&25*，並儲存在*差值*中(內部變數，DINT 資料類型 2. 將 *SCPU_Relay* (內部變數，WORD 資料類型，(必要時，指定陣列為 400 個元件(請參閱備註)，AT 設定=1500))的實際 I/O 記憶體位址儲存在指標暫存器 IR0 中。 **備註** 指定一個 SCPU_relay 的陣列，如 *SCPU_relay[2]*(例如)，來讓位址 CIO 1500 + (*UnitNo* x &25) + 2 可以被指定。這也適用於以下範例 2。 3. 以變數*差值*(變數 *UnitNo* x &25)的值增加指標暫存器 IR0 中的實際 I/O 記憶體位址。
2) 指定指標暫存器的常數差值(指定一個介於 CIO n+0 到 n+24 之間的位元)	CIO 1500 + (*UnitNo* x &25)的實際 I/O 記憶體位址會透過上述步驟 1 中的處理儲存在指標暫存器 IR0 中。因此字組的位址會利用來自 IR0 的常數差值指定。 例如，指定+2,IR0 會指定 CIO 1500 + (*UnitNo* x &25) + 2。 **備註** 也可以透過利用有 *SCPU_relay* 的陣列設定指定 *SCPU_relay [2]*來指定 CIO 1500 + (*UnitNo* x &25) + 2。 利用可以指定字組中的位元位址(例如 TST(350/351)/SETB(532)指令的第二運算元)的指令來指定位元位址。
檢查本地節點資料連結關係 	範例：變數 *NodeSelf_OK* 會在 *NetCheck_OK* (內部變數，BOOL 資料類型)為 ON 且來自 IR0 的+6 差值處的字組的位元 15 (CIO 1500 + *UnitNo* x &25 +6)為 ON 時轉為 ON。
將指標暫存器恢復為先前的值 將資料從臨時備份緩衝區重新儲存到 IR0 	指標暫存器會在這個功能區塊完成後(或指標暫存器使用後)恢復為原來的值。 範例：所儲存的變數 *SaveIR[0]*的值會儲存在指標暫存器 IR0 中，而這個值會恢復為這個功能開始時(或使用指標暫存器之前)的內容。

2-5 以運算元指定多重字組的開頭或結尾的指令的注意事項

在使用階梯圖程式編輯來建立指令運算元指定有一個字組範圍的第一或最後一個字組的功能區塊時，在指定運算元的變數時適用下列注意事項。

當運算元指定有多重字組的第一或最後一個字組時，指令會根據針對 AT 設定(或外部變數設定)所內部分配的位址作用。因此，變數資料類型及陣列元件數與指令的操作無關。指定一個有 AT 設定的變數，或指定一個大小符合要由指令處理的資料大小的陣列變數。

關於一個 AT 設定(或外部變數設定)或一個有一些元件的陣列設定是否需要指定指令運算元中的一個字組範圍的第一位址的詳細資訊，請參考 *2-6 指令支援及運算元限制*。

備註 要指定一個指令運算元中的多重字組的第一或最後一字組時，請務必指定一個有 AT 設定的變數(或一個外部變數)，或一個大小與要在指令中處理的資料大小相同的變數。適用下列注意事項。

1,2,3... 1. 如果指定一個非陣列變數而沒有 AT 設定且沒有相符的資料大小，當編輯時 CX-Programmer 會在匯集時輸出一個錯誤。

2. 在指定一個陣列變數時，適用下列注意事項。

要在指令運算元中處理的大小是固定的

請確定陣列中的元件數是否與要由指令處理的大小相同。否則，CX-Programmer 會在匯集時輸出一個錯誤。

要在指令運算元中處理的大小是不固定的

請確定陣列中的元件數是否等於或大於由一個其他運算元所指定的大小。

其他運算元指定大小：常數

CX-Programmer 會在匯集時輸出一個錯誤。

其他運算元指定大小：變數

即使陣列中的元件數與一個其他運算元中所指定的大小不符(變數)，CX-Programmer 也不會在匯集時輸出一個錯誤(會顯示一個警告訊息)。

尤其是，當陣列中的元件數小於由一個其他運算元所指定的大小時，(例如當指令處理的大小是 16 而實際登錄在變數表中的元件數是 10 時)，指令會在超出元件數的區域中執行讀取/寫入處理。(在這個範例中，讀取/寫入處理會在元件數登錄到實際變數表中以後針對下 6 個字組執行。)如果相同的區域也由其他指令使用(包括內部變數配置)，則可能會發生非預期的操作，這可能會導致嚴重的意外事故。

不要使用大小與指定一個字組範圍的第一位址(或最後位址)的運算元中要由指令處理的資料大小不符的變數。請務必使用大小與指令所要求的資料大小相同的非陣列變數資料類型或元件數與指令所要求的資料大小相同的陣列變數。否則，會發生下列錯誤。

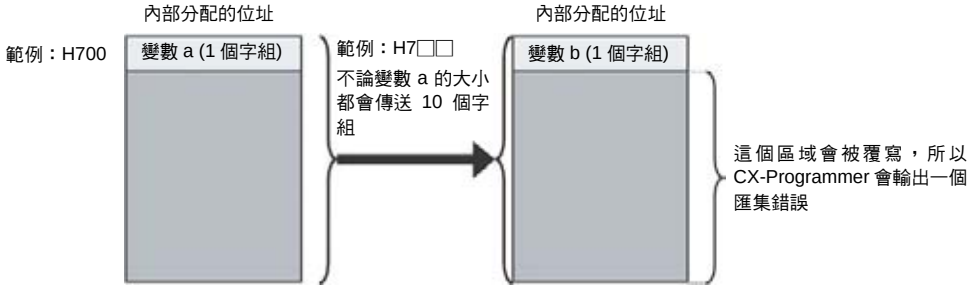
沒有相符的資料大小且沒有 AT 設定的非陣列變數

如果指定有多重字組的第一位址(或最後位址)的運算元使用一個大小與指令所要求的資料大小不符並且沒有使用 AT 設定的非陣列變數資料類型，則 CX-Programmer 會輸出一個匯集錯誤。

範例：區塊傳送(070)指令：XFER W S D

(W：字組數，S：第一來源字組；D：第一目的字組)
當 W 指定為*&10*、S 指定為資料類型為 WORD 的變數 *a*、而 D 指定為資料類型為 WORD 的變數 *b* 時，即為：XFER &10 a b。XFER(070)指令會傳送開頭來自變數 *a* 中自動分配的位址的 10 個字組中的資料給到開頭為變數 *b* 中自動分配的位址的 10 個字組。因此，CX-Programmer 會輸出一個匯集錯誤。

範例：XFER &10 a b
(變數 a 和 b 為 WORD 資料類型)



陣列變數

結果取決於下列條件。

要由指令處理的大小是固定的

如果要由指令處理的大小是一個固定的運算元，且這個大小與陣列元件數不符，則 CX-Programmer 會輸出一個匯集錯誤。

範例：行到欄(064)指令；COLM S D N

(S：位元數，D：第一目的字組，N：來源字組)

E.g., COLM a b[0] c

如果 D 指定為一個有 10 個陣列元件的 WORD 資料類型陣列而它應為 16 個陣列元件時，CX-Programmer 會在匯集時輸出一個錯誤。

要由指令處理的大小是不固定的

當要由指令處理的運算元大小不是固定的(大小為指令中的其他運算元所指定)時，請確定陣列元件數是否等於或大於其他運算元中所指定的大小(即，要由指令處理的大小)。

其他運算元指定大小：常數

CX-Programmer 會在匯集時輸出一個錯誤。

範例：區塊傳送：XFER W S D

(W：字組數，S：第一來源字組；D：第一目的字組)

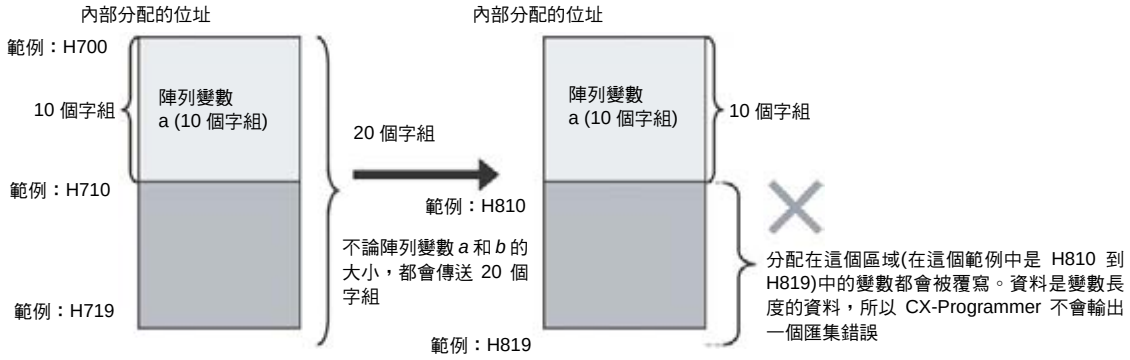
當 W 指定為*&20*、S 指定為 WORD 資料類型且有 10 個元件的陣列變數 *a*、而 D 指定為 WORD 資料類型且有 10 個元件的陣列變數 *b* 時：

XFER　&20　a[0]　b[0]

雖然陣列變數 a[0]和 b[0]都是 10 個字組，XFER(070)指令仍會執行 W 所指定的 20 個字組的傳送處理。因此，XFER(070)指令會針對隨在所分配的陣列元件數後的 I/O 記憶區執行讀取/寫入處理，如下圖所示。

因此，如果 a[10 個元件]是內部分配的字組(例如 H700 到 H709)，而 b[10 個元件]是內部分配的字組(例如 H800 到 H809)，XFER(070)會傳送字組 H700 到 H719 中的資料給字組 H800 到 H819。在這項操作中，如果另一個內部分配的變數(例如 c)是 H810 到 H819 中分配的字組，則這個字組會被覆寫而導致發生非預期的操作。要傳送 20 個字組時，請確定兩個陣列變數 *a* 和 *b* 的元件數是否指定為 20 個元件。

XFER　&20　a[0]　b[0]
變數 *a* 和 *b* 都使用 10 個元件的 WORD 資料類型：
要傳送 20 個字組時，陣列變數 *a* 和 *b* 請務必都指定為 20 個元件。



其他運算元指定大小：變數

即使陣列元件數與其他運算元(變數)中所指定的大小(即，要由指令處理的大小)不符，CX-Programmer 也不會在匯集時輸出一個錯誤。指令會根據運算元所指定的大小執行，而與陣列變數中的元件數無關。

尤其是如果陣列中的元件數少於其他運算元(變數)所指定的大小(即，要由指令處理的大小)，其他變數會被影響且可能會發生非預期的操作。

2-6　指令支援及運算元限制

本附錄中的表格顯示可以用在以階梯圖程式編輯語言所建立的功能區塊中的指令、它們和運算元所適用的限制，包括變數(不論是否為陣列變數及是否要求 AT 設定或外部變數指定，以及可以使用哪些資料類型)。

指令支援

不支援由 CX-Programmer、CP 系列 CPU 模組、以及模組版本 3.0 的 CS/CJ 系列 CPU 模組用於功能區塊定義中的指令會在*符號*欄中標示為*功能區塊中不支援*。

運算元限制

- 指定多重字組的第一或最後一字組並因此要求 AT 設定或陣列變數指定的運算元，如以下 *AT 設定或陣列要求*欄中所示。
  是：運算元必須指定一個 AT 設定(或外部指定)或陣列變數才能指定多重字組的第一或最後一字組。
  - 括弧中的值是由指令用來進行讀取、寫入、或其他處理的固定大小。這個大小表示資料類型的大小或文字單位中所指定的陣列變數所需的大小。在陣列變數方面，這個大小必須與元件數相同。否則，CX-Programmer 會在匯集時輸出一個錯誤。
  - 如果括弧中顯示"不固定"，則由指令用來進行讀取、寫入、或其他處理的大小可能會改變。請確定是否提供陣列元件數所需的最大規模(大小)。即使一個大小不固定的運算元中的陣列元件數與一個其他運算元中所指定的大小不符，CX-Programmer 也不會在匯集時輸出一個錯誤。指令會根據其他運算元中所指定的大小作用，而與陣列變數元件數無關。

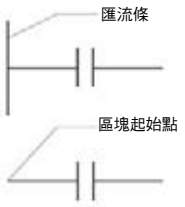
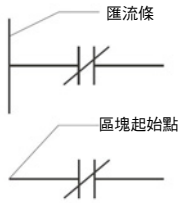
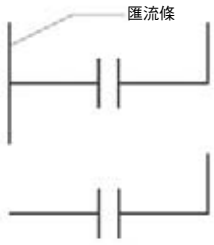
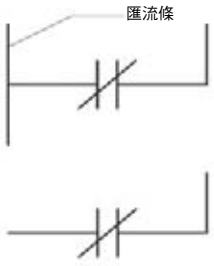
  ---：不要求 AT 設定或陣列變數指定的運算元。

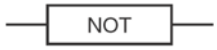
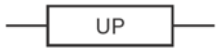
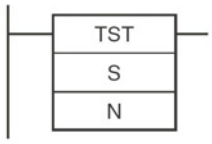
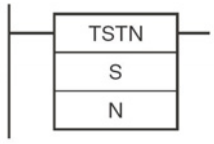
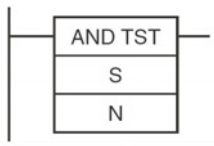
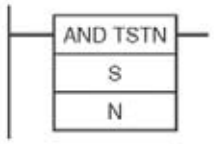
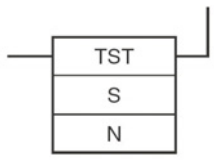
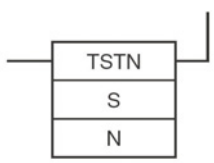
  備註 當一個指令運算元中指定有多重字組的第一或最後一字組時，輸入參數不能用來轉送資料給變數或從變數轉送資料。必須準備一個 AT 設定或一個有所需元件數的陣列變數，並且在陣列資料設定到功能區塊定義中以後，必須針對運算元指定陣列變數中的第一或最後一個元件。
- 任何必須針對網路中的一個遠端節點上的一個 I/O 記憶體位址指定一個 AT 設定的運算元都會在 *AT 設定或陣列要求*欄中標示為*指定有 AT 設定的遠端節點的位址*。

下表列出 CS/CJ 系列 CPU 模組、NSJ 系列 NSJ 控制器、及 FQM1 彈性動作控制器(模組版本 3.0 以上)所支援的所有指令。

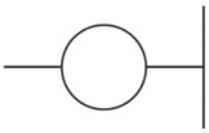
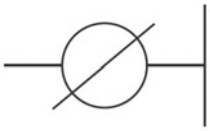
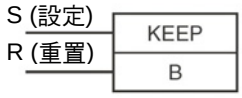
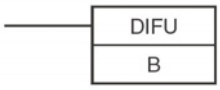
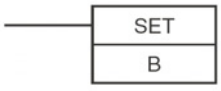
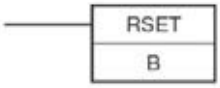
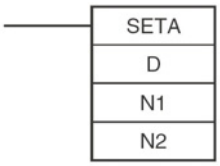
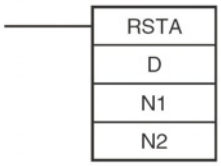
- 某些指令只有 FQM1 彈性動作控制器(模組版本 3.0 以上)有支援。這些指令會在數值指令下標示為"僅限於 FQM1"。
- 也有一些指令只有 CS/CJ 系列 CPU 模組和 NSJ 系列 NSJ 控制器有支援，也就是說，不能用於 FQM1 彈性動作控制器(模組版本 3.0 以上)。請參考 *FQM1 指令參考手冊*(Cat. No. O013)來確認有支援的指令。

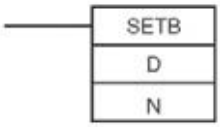
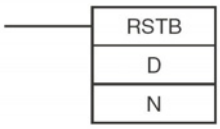
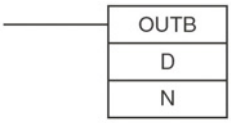
2-6-1 可程式控制器輸入指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
LOAD	LD @LD %LD !LD !@LD !%LD			B：位元	BOOL	---
LOAD NOT	LD NOT !LD NOT @LD NOT %LD NOT !@LD NOT !%LD NOT			B：位元	BOOL	---
AND	AND @AND %AND !AND !@AND !%AND			B：位元	BOOL	---
AND NOT	AND NOT !AND NOT @AND NOT %AND NOT !@AND NOT !%AND NOT			---	BOOL	---
OR	OR @OR %OR !OR !@OR !%OR			---	BOOL	---
OR NOT	OR NOT !OR NOT @OR NOT %OR NOT !@OR NOT !%OR NOT			---	BOOL	---
AND LOAD	AND LD			---	---	---

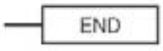
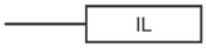
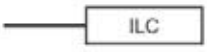
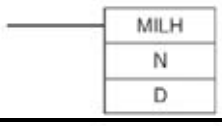
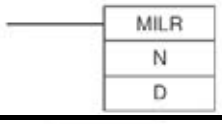
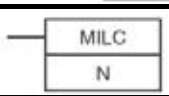
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
OR LOAD	OR LD			---	---	---
NOT	NOT	520		---	---	---
CONDITION ON (條件 ON)	UP	521		---	---	---
CONDITION OFF (條件 OFF)	DOWN	522		---	---	---
BIT TEST (位元測試)	LD TST	350		S：來源字組 N：位元號碼	WORD UINT	--- ---
BIT TEST (位元測試)	LD TSTN	351		S：來源字組 N：位元號碼	WORD UINT	--- ---
BIT TEST (位元測試)	AND TST	350		S：來源字組 N：位元號碼	WORD UINT	--- ---
BIT TEST (位元測試)	AND TSTN	351		S：來源字組 N：位元號碼	WORD UINT	--- ---
BIT TEST (位元測試)	OR TST	350		S：來源字組 N：位元號碼	WORD UINT	--- ---
BIT TEST (位元測試)	OR TSTN	351		S：來源字組 N：位元號碼	WORD UINT	--- ---

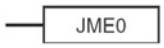
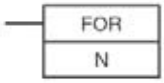
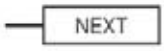
2-6-2 可程式控制器輸出指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
OUTPUT (輸出)	OUT !OUT			---	BOOL	---
OUTPUT NOT (輸出 NOT)	OUT NOT !OUT NOT			---	BOOL	---
KEEP (保持)	KEEP !KEEP	011		---	BOOL	---
DIFFERENTIATE UP (區別 UP)	DIFU !DIFU	013		---	BOOL	---
DIFFERENTIATE DOWN (區別 DOWN)	DIFD !DIFD	014		---	BOOL	---
SET (設定)	SET @SET %SET !SET !@SET !%SET			B：位元	BOOL	---
RESET (重置)	RSET @RSET %RSET !RSET !@RSET !%RSET			B：位元	BOOL	---
MULTIPLE BIT SET (多重位元設定)	SETA @SETA	530		D：開頭字組 N1：開頭位元 N2：位元數	UINT UINT UINT	是(不固定) --- ---
MULTIPLE BIT RESET (多重位元重置)	RSTA @RSTA	531		D：開頭字組 N1：開頭位元 N2：位元數	UINT UINT UINT	是(不固定) --- ---

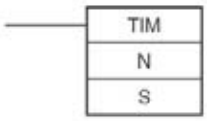
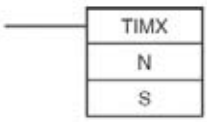
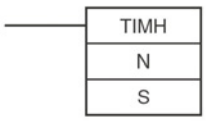
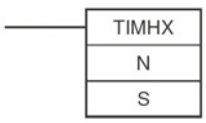
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SINGLE BIT SET (單一位元設定)	SETB @SETB !SETB	532		D：字組位址	UINT	---
				N：位元號碼	UINT	---
SINGLE BIT RESET (單一位元重置)	RSTB @RSTB !RSTB	533		D：字組位址	UINT	---
				N：位元號碼	UINT	---
SINGLE BIT OUTPUT (單一位元輸出)	OUTB @OUTB !OUTB	534		D：字組位址	UINT	---
				N：位元號碼	UINT	---

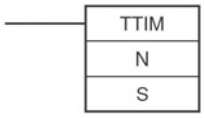
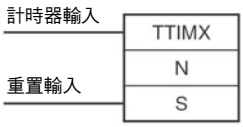
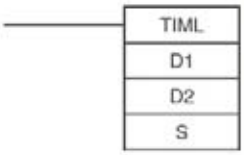
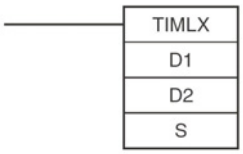
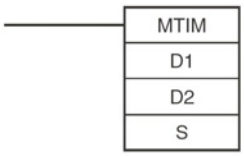
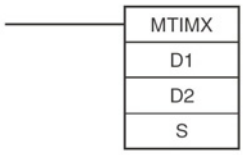
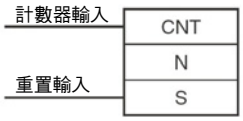
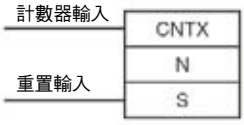
2-6-3 順序控制指令

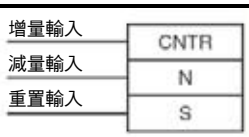
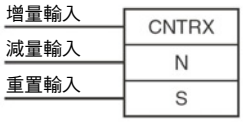
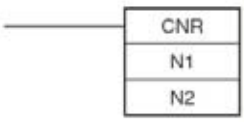
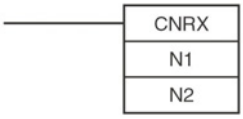
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
END (結束)	END	001		---	---	---
NO OPERATION (無運算)	NOP	000	---	---	---	---
INTERLOCK (連鎖)	IL	002		---	---	---
INTERLOCK CLEAR (清除連鎖)	ILC	003		---	---	---
MULTI-INTERLOCK DIFFERENTIATION HOLD (多重連鎖區別保留)	MILH	517		N：連鎖號碼	#僅限於#正的 10 進制值	---
				D：連鎖狀態位元	BOOL	---
MULTI-INTERLOCK DIFFERENTIATION RELEASE (多重連鎖區解除)	MILR	518		N：連鎖號碼	#僅限於#正的 10 進制值	---
				D：連鎖狀態位元	BOOL	---
MULTI-INTER LOCK CLEAR (多重連鎖區清除)	MILC	519		N：連鎖號碼	#僅限於#正的 10 進制值	---
JUMP (跳越)	JMP	004	功能區塊中不支援	N：跳越號碼	---	---
JUMP END (跳越結束)	JME	005	功能區塊中不支援	N：跳越號碼	---	---
CONDITIONAL JUMP (條件跳越)	CJP	510	功能區塊中不支援	N：跳越號碼	---	---
CONDITIONAL JUMP (條件跳越)	CJPN	511	功能區塊中不支援	N：跳越號碼	---	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
MULTIPLE JUMP (多重跳越)	JMP0	515		---	---	---
MULTIPLE JUMP END (多重跳越結束)	JME0	516		---	---	---
FOR-NEXT LOOPS (下一迴路)	FOR	512		N：迴路數	UINT	---
BREAK LOOP (中斷迴路)	BREAK	514		---	---	---
FOR-NEXT LOOPS (下一迴路)	NEXT	513		---	---	---

2-6-4 計時器與計數器指令

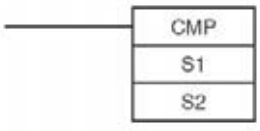
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
TIMER (計時器)	TIM (BCD)			N：計時器輸入	TIMER	---
				S：設定值	WORD	---
	TIMX (BIN)	550		N：計時器輸入	TIMER	---
				S：設定值	UINT	---
HIGH-SPEED TIMER (高速計時器)	TIMH (BCD)	015		N：計時器輸入	TIMER	---
				S：設定值	WORD	---
	TIMHX (BIN)	551		N：計時器輸入	TIMER	---
				S：設定值	UINT	---
ONE-MS TIMER (ONE-MS 計時器)	TMHH (BCD)	540	功能區塊中不支援	N：計時器輸入	TIMER	---
				S：設定值	WORD	---
	TMHHX (BIN)	552	功能區塊中不支援	N：計時器輸入	TIMER	---
				S：設定值	UINT	---

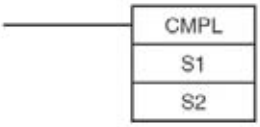
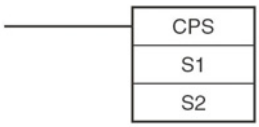
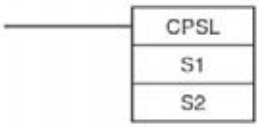
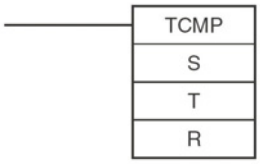
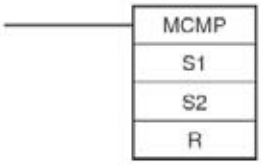
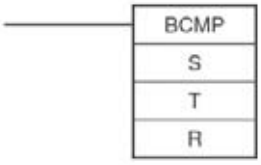
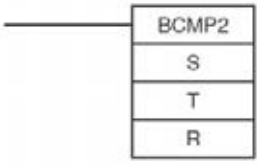
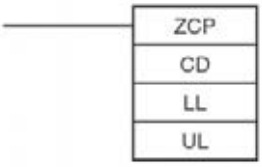
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
ACCUMULATIVE TIMER (累進計時器)	TTIM (BCD)	087		N：計時器輸入	TIMER	---
				S：設定值	WORD	---
	TTIMX (BIN)	555		N：計時器輸入	TIMER	---
				S：設定值	UINT	---
LONG TIMER (LONG 計時器)	TIML (BCD)	542		D1：完成旗標	WORD	---
				D2：PV 字組	DWORD	---
				S：SV 字組	DWORD	---
	TIMLX (BIN)	553		D1：完成旗標	UINT	---
				D2：PV 字組	UDINT	---
				S：SV 字組	UDINT	---
MULTI-OUTPUT TIMER (多重輸出計時器)	MTIM (BCD)	543		D1：完成旗標	UINT	---
				D2：PV 字組	WORD	---
				S：第 1 SV 字組	WORD	是(8)
	MTIMX (BIN)	554		D1：完成旗標	UINT	---
				D2：PV 字組	UINT	---
				S：第 1 SV 字組	WORD	是(8)
COUNTER (計數器)	CNT (BCD)			N：計數器號碼	COUNTER	---
				S：設定值	WORD	---
	CNTX (BIN)	546		N：計數器號碼	OUNTER	---
				S：設定值	UINT	---

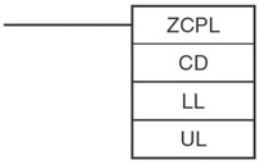
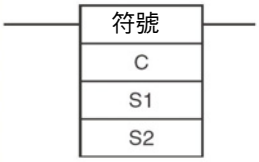
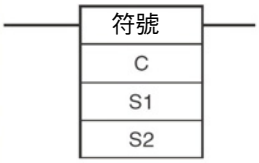
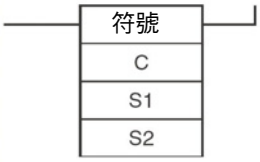
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
REVERSIBLE COUNTER (可逆計數器)	CNTR (BCD)	012		N：計數器號碼	COUNTER	---
				S：設定值	WORD	---
	CNTRX (BIN)	548		N：計數器號碼	COUNTER	---
				S：設定值	UINT	---
RESET TIMER/ COUNTER (重置計時器/計數器)	CNR @CNR (BCD)	545		N1：範圍中的第一個數字	計時器或計數器(請參閱備註)	---
				N2：範圍中的最後一個數字	計時器或計數器(請參閱備註)	---
	CNRX @CNRX (BIN)	547		N1：範圍中的第一個數字	計時器或計數器(請參閱備註)	---
				N2：範圍中的最後一個數字	計時器或計數器(請參閱備註)	---

備註 在 N1 及 N2 指定相同的變數時啟用。

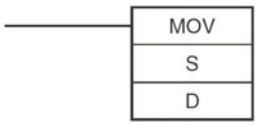
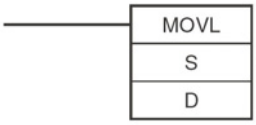
2-6-5 比較指令

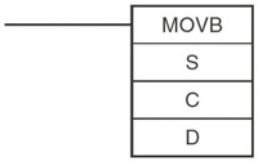
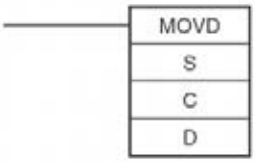
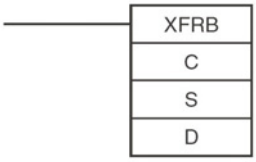
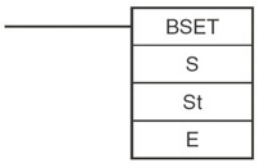
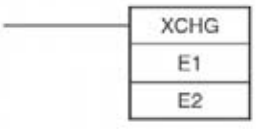
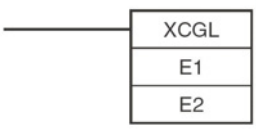
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
符號比較 (無正負號)	LD,AND, OR + =, <>, <, <=, >, >=	300 (=) 305 (<>) 310 (<) 315 (<=) 320 (>) 325 (>=)	使用 LD：  使用 AND：  使用 OR：  	S1：比較資料 1	UINT	---
				S2：比較資料 2	UINT	---
符號比較 (雙字組，無正負號)	LD,AND, OR + =, <>, <, <=, >, >= + L	301 (=) 306 (<>) 311 (<) 316 (<=) 321 (>) 326 (>=)	---	S1：比較資料 1	UDINT	---
				S2：比較資料 2	UDINT	---
符號比較 (含正負號)	LD,AND, OR + =, <>, <, <=, >, >= + S	302 (=) 307 (<>) 312 (<) 317 (<=) 322 (>) 327 (>=)	---	S1：比較資料 1	INT	---
				S2：比較資料 2	INT	---
符號比較 (雙字組，含正負號)	LD,AND, OR + =, <>, <, <=, >, >= + SL	303 (=) 308 (<>) 313 (<) 318 (<=) 323 (>) 328 (>=)	---	S1：比較資料 1	DINT	---
				S2：比較資料 2	DINT	---
UNSIGNED COMPARE (無正負號比較)	CMP !CMP	020	 	S1：比較資料 1	UINT	---
				S2：比較資料 2	UINT	---

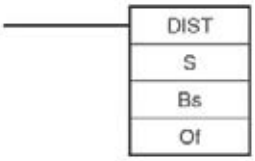
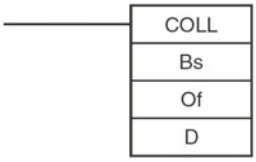
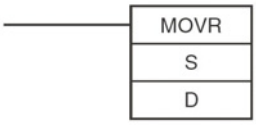
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE UNSIGNED COMPARE (雙重無正負號比較)	CMPL	060		S1：比較資料 1	UDINT	---
				S2：比較資料 2	UDINT	---
SIGNED BINARY COMPARE (含正負號二進位比較)	CPS !CPS	114		S1：比較資料 1	INT	---
				S2：比較資料 2	INT	---
DOUBLE SIGNED BINARY COMPARE (雙重含正負號二進位比較)	CPSL	115		S1：比較資料 1	DINT	---
				S2：比較資料 2	DINT	---
TABLE COMPARE (表格比較)	TCMP @TCMP	085		S：來源資料	WORD	
				T：表格的第 1 個字組	WORD	是(16)
				R：結果字組	UINT	---
MULTIPLE COMPARE (多重比較)	MCMP @MCMP	019		S1：設定 1 的第 1 個字組	WORD	是(16)
				S2：設定 2 的第 1 個字組	WORD	是(16)
				R：結果字組	UINT	
UNSIGNED BLOCK COMPARE (無正負號區塊比較)	BCMP @BCMP	068		S：來源資料	WORD	---
				T：表格的第 1 個字組	WORD	是(32)
				R：結果字組	UINT	---
EXPANDED BLOCK COMPARE (擴充區塊比較)	BCMP2 @BCMP2	502		S：來源資料	WORD	---
				T：區塊的第 1 個字組	WORD	是(不固定)
				R：結果字組	WORD	---
AREA RANGE COMPARE (區域範圍比較)	ZCP	088		CD：比較資料 (1 個字組)	UINT	---
				LL：範圍下限	UINT	---
				UL：範圍上限	UINT	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE AREA RANGE COMPARE (雙重區域範圍比較)	ZCPL	116		CD：比較資料(2 個字組)	UDINT	---
				LL：範圍下限	UDINT	---
				UL：範範圍上限	UDINT	---
時間比較	LD, AND, OR + =DT, <> DT, <DT, <=DT, >DT, >=DT	341 (=DT) 342 (<>DT) 343 (<DT) 344 (<=DT) 345 (>DT) 346 (>=DT)	LD (載入)：  AND：  OR： 	C：控制字組	WORD	---
				S1:現行時間的第 1 個字組	WORD	是(3)
				S2:比較時間的第 1 個字組	WORD	是(3)

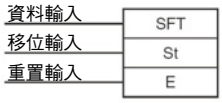
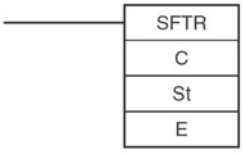
2-6-6 資料移動指令

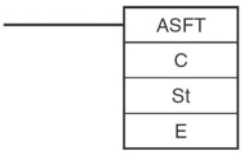
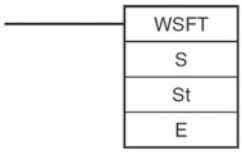
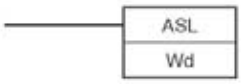
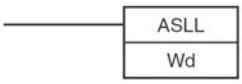
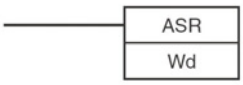
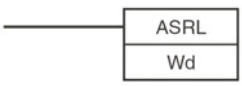
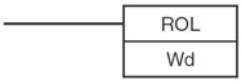
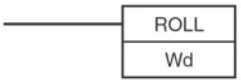
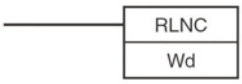
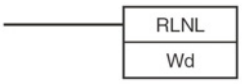
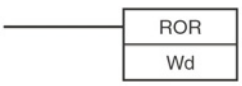
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
MOVE (移動)	MOV @MOV !MOV !@MOV	021		S：來源	WORD	---
				D：目的	WORD	---
DOUBLE MOVE (雙重移動)	MOVL @MOVL	498		S：第 1 來源字組	DWORD	---
				D：第 1 目的字組	DWORD	---

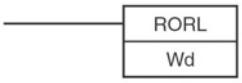
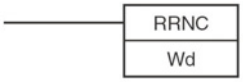
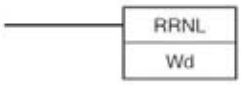
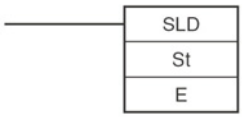
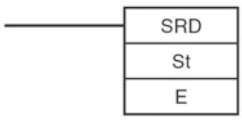
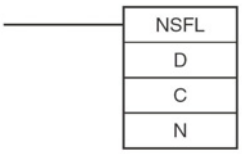
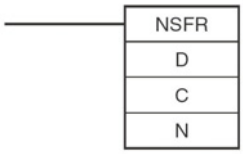
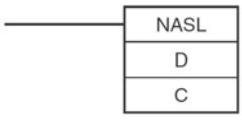
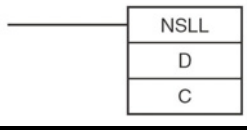
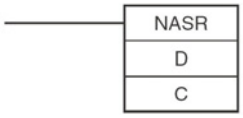
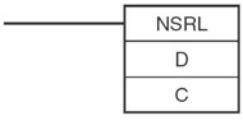
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
MOVE NOT (移動 NOT)	MVN @MVN	022		S：來源	WORD	---
				D：目的	WORD	---
DOUBLE MOVE NOT (雙重移動 NOT)	MVNL @MVNL	499		S：第 1 來源字組	DWORD	---
				D：第 1 目的字組	DWORD	---
MOVE BIT (移動位元)	MOVB @MOVB	082		S：來源字組或資料	WORD	---
				C：控制字組	UINT	---
				D：目的字組	WORD	---
MOVE DIGIT (移動位數)	MOVD @MOVD	083		S：來源字組或資料	WORD	---
				C：控制字組	UINT	---
				D：目的字組	UINT	---
MULTIPLE BIT TRANSFERÅ@ (多重位元傳輸 Å@)	XFRB @XFRB	062		C：控制字組	UINT	---
				S：第 1 來源字組	WORD	是(不固定)
				D：第 1 目的字組	WORD	是(不固定)
BLOCK TRANSFER (區塊傳送)	XFER @XFER	070		N：字組數	UINT	---
				S：第 1 來源字組	WORD	是(不固定)
				D：第 1 目的字組	WORD	是(不固定)
BLOCK SET (區塊設定)	BSET @BSET	071		S：來源字組	WORD	---
				St：起始字組	WORD	是(不固定)
				E：結束字組	WORD	是(不固定)
DATA EXCHANGE (資料交換)	XCHG @XCHG	073		E1：第 1 交換字組	WORD	---
				E2：第 2 交換字組	WORD	---
DOUBLE DATA EXCHANGE (雙重資料交換)	XCGL @XCGL	562		E1：第 1 交換字組	DWORD	---
				E2：第 2 交換字組	DWORD	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SINGLE WORD DISTRIBUTE (單一字組分配)	DIST @DIST	080		S：來源字組	WORD	---
				Bs：目的資料庫位址	WORD	是(不固定)
				Of：差值	UINT	---
DATA COLLECT (資料收集)	COLL @COLL	081		Bs：來源資料庫位址	WORD	是(不固定)
				Of：差值	WORD	---
				D：目的字組	WORD	---
MOVE TO REGISTER (移到暫存器)	MOVR @MOVR	560		S：來源(想要的字組軌跡)	BOOL	---
				D：目的(指標暫存器)	WORD	---
MOVE TIMER/ COUNTER PV TO REGISTER (將計時器/計數器 PV 移到暫存器)	MOVRW @MOVRW	561	功能區塊中不支援	S:來源(想要的 TC 號碼)	---	---
				D：目的(指標暫存器)	---	---

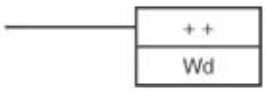
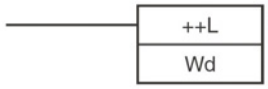
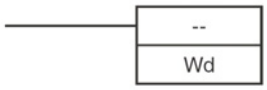
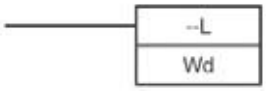
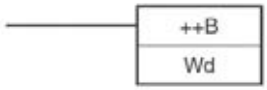
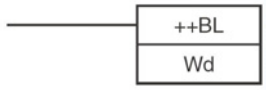
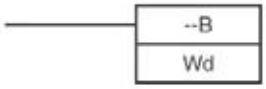
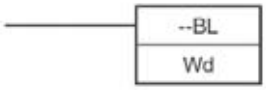
2-6-7 資料移位指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SHIFT REGISTER (移位暫存器)	SFT	010		St：起始字組	UINT	是(不固定)
				E：結束字組	UINT	在需要陣列變數時 D1 和 D2 必須是相同的陣列變數。
REVERSIBLE SHIFT REGISTER (可逆移位暫存器)	SFTR @SFTR2	084		C：控制字組	UINT	---
				St：起始字組	UINT	是(不固定)
				E：結束字組	UINT	在需要陣列變數時 D1 和 D2 必須是相同的陣列變數。

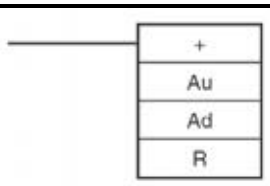
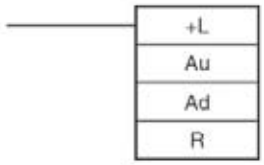
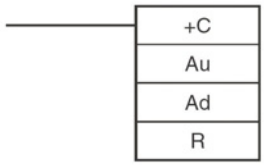
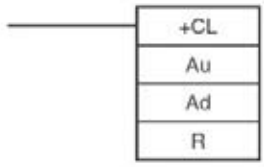
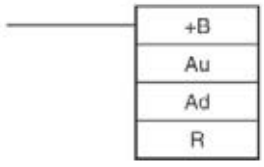
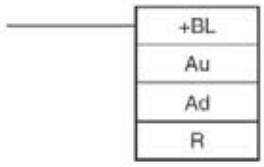
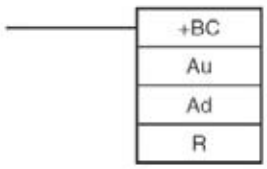
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
ASYNCHRONOUS SHIFT REGISTER (非同步移位暫存器)	ASFT @ASFT	017		C：控制字組	UINT	---
				St：起始字組	UINT	是(不固定)
				E：結束字組	UINT	在需要陣列變數時 D1 和 D2 必須是相同的陣列變數。
WORD SHIFT (字組移位)	WSFT @WSFT	016		S：來源字組	UINT	---
				St：起始字組	WORD	是(不固定)
				E：結束字組	UINT	在需要陣列變數時 D1 和 D2 必須是相同的陣列變數。
ARITHMETIC SHIFT LEFT (算術移位，左)	ASL @ASL	025		Wd：字組	UINT	
DOUBLE SHIFT LEFT (雙重移位，左)	ASLL @ASLL	570		Wd：字組	UDINT	
ARITHMETIC SHIFT RIGHT (算術移位，右)	ASR @ASR	026		Wd：字組	UINT	
DOUBLE SHIFT RIGHT (雙重移位，右)	ASRL @ASRL	571		Wd：字組	UDINT	
ROTATE LEFT (輪調，左)	ROL @ROL	027		Wd：字組	UINT	
DOUBLE ROTATE LEFT (雙重輪調，左)	ROLL @ROLL	572		Wd：字組	UDINT	
ROTATE LEFT WITHOUT CARRY (輪調，左，不進位)	RLNC @RLNC	574		Wd：字組	UINT	
DOUBLE ROTATE LEFT WITHOUT CARRY (雙重輪調，左，不進位)	RLNL @RLNL	576		Wd：字組	UDINT	
ROTATE RIGHT (輪調，右)	ROR @ROR	028		Wd：字組	UINT	

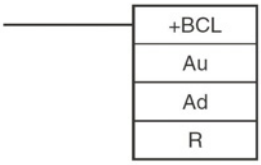
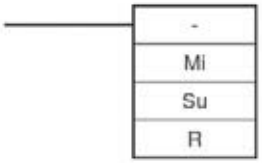
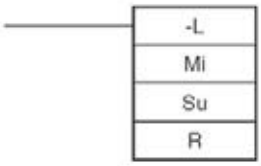
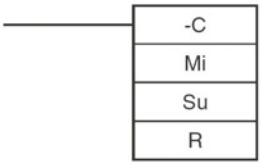
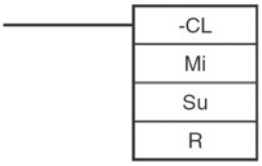
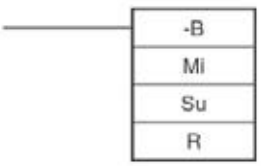
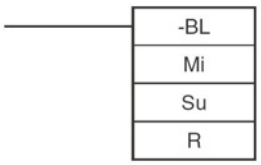
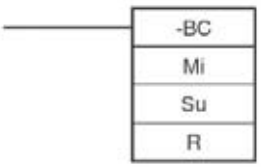
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE ROTATE RIGHT (雙重輪調，右)	RORL @RORL	573		Wd：字組	UDINT	---
ROTATE RIGHT WITHOUT CARRY (輪調，右，不進位)	RRNC @RRNC	575		Wd：字組	UINT	---
DOUBLE ROTATE RIGHT WITHOUT CARRY (雙重輪調，右，不進位)	RRNL @RRNL	577		Wd：字組	UDINT	---
ONE DIGIT SHIFT LEFT (一位數移位，左)	SLD @SLD	074		St：起始字組 E：結束字組	UINT UINT	是(不固定) 是(不固定)
ONE DIGIT SHIFT RIGHT (一位數移位，右)	SRD @SRD	075		St：起始字組 E：結束字組	UINT UINT	是(不固定) 是(不固定)
SHIFT N-BIT DATA LEFT (移位 N 位元資料，左)	NSFL @NSFL	578		D：移位的開頭字組 C：開頭位元 N：移位資料長度	UINT UINT UINT	是(不固定) --- ---
SHIFT N-BIT DATA RIGHT (移位 N 位元資料，右)	NSFR @NSFR	579		D：移位的開頭字組 C：開頭位元 N：移位資料長度	UINT UINT UINT	是(不固定) --- ---
SHIFT N-BITS DATA LEFT (移位 N 位元資料，左)	NASL @NASL	580		D：移位字組 C：控制字組	UINT UINT	--- ---
DOUBLE SHIFT N-BITS LEFT (雙重移位 N 位元資料，左)	NSLL @NSLL	582		D：移位字組 C：控制字組	UDINT UINT	--- ---
SHIFT N-BITS RIGHT (移位 N 位元，右)	NASR @NASR	581		D：移位字組 C：控制字組	UINT UINT	--- ---
DOUBLE SHIFT N-BITS RIGHT (雙重移位 N 位元資料，右)	NSRL @NSRL	583		D：移位字組 C：控制字組	UDINT UINT	--- ---

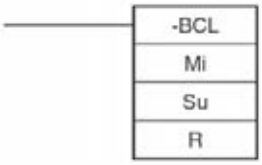
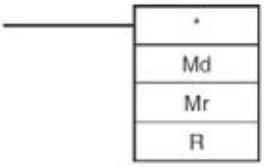
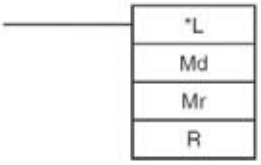
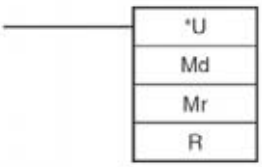
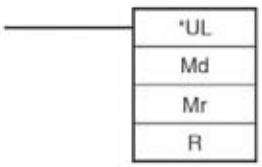
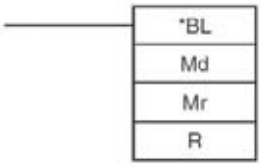
2-6-8 增量/減量指令

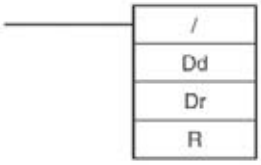
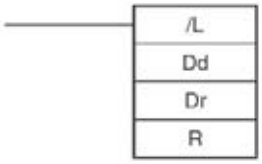
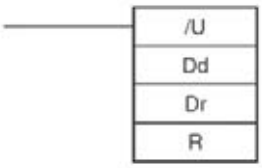
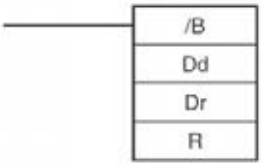
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
INCREMENT BINARY (增量二進位)	++ @++	590		Wd：字組	UINT	---
DOUBLE INCREMENT BINARY (雙重增量二進位)	++L @++L	591		Wd：字組	UDINT	---
DECREMENT BINARY (減量二進位)	-- @--	592		Wd：字組	UINT	---
DOUBLE DECREMENT BINARY (雙重減量二進位)	--L @--L	593		Wd：第 1 個字組	UDINT	---
INCREMENT BCD (增量 BCD)	++B @++B	594		Wd：字組	WORD	---
DOUBLE INCREMENT BCD (雙重增量 BCD)	++BL @++BL	595		Wd：第 1 個字組	DWORD	---
DECREMENT BCD (減量 BCD)	--B @--B	596		Wd：字組	DWORD	---
DOUBLE DECREMENT BCD (雙重減量 BCD)	--BL @--BL	597		Wd：第 1 個字組	WORD	---

2-6-9 符號數學指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SIGNED BINARY ADD WITHOUT CARRY (含正負號二進位除法不進位)	+ @+	400		Au：被加數字組	INT	---
				Ad：加數字組	INT	---
				R：結果字組	INT	---
DOUBLE SIGNED BINARY ADD WITHOUT CARRY (雙重含正負號二進位加法，不進位)	+L @+L	401		Au：第 1 加數字組	DINT	---
				Ad：第 1 加數字組	DINT	---
				R：第 1 結果字組	DINT	---
SIGNED BINARY ADD WITH CARRY (含正負號二進位除法進位)	+C @+C	402		Au：被加數字組	INT	---
				Ad：加數字組	INT	---
				R：結果字組	INT	
DOUBLE SIGNED BINARY ADD WITH CARRY (雙重含正負號二進位加法，進位)	+CL @+CL	403		Au：第 1 加數字組	DINT	---
				Ad：第 1 加數字組	DINT	---
				R：第 1 結果字組	DINT	---
BCD ADD WITHOUT CARRY (BCD 加法不進位)	+B @+B	404		Au：被加數字組	WORD	---
				Ad：加數字組	WORD	---
				R：結果字組	WORD	---
DOUBLE BCD ADD WITHOUT CARRY (雙重 BCD 加法，不進位)	+BL @+BL	405		Au：第 1 加數字組	DWORD	---
				Ad：第 1 加數字組	DWORD	---
				R：第 1 結果字組	DWORD	---
BCD ADD WITH CARRY (BCD 加法進位)	+BC @+BC	406		Au：被加數字組	WORD	---
				Ad：加數字組	WORD	---
				R：結果字組	WORD	---

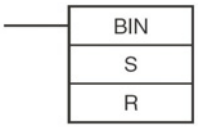
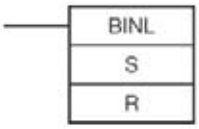
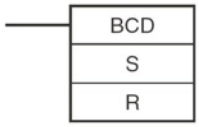
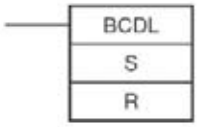
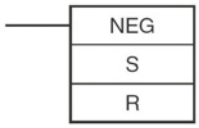
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE BCD ADD WITH CARRY (雙重 BCD 加法，進位)	+BCL @+BCL	407		Au：第 1 加數字組 Ad：第 1 加數字組 R：第 1 結果字組	DWORD DWORD DWORD	--- --- ---
SIGNED BINARY SUBTRACT WITHOUT CARRY (含正負號二進位減法不進位)	- @-	410		Mi：被減數字組 Su：減數字組 R：結果字組	INT INT INT	--- --- ---
DOUBLE SIGNED BINARY SUBTRACT WITHOUT CARRY (雙重含正負號二進位減法，不進位))	-L @-L	411		Mi：被減數字組 Su：減數字組 R：結果字組	DINT DINT DINT	--- --- ---
SIGNED BINARY SUBTRACT WITH CARRY (含正負號二進位減法進位)	-C @-C	412		Mi：被減數字組 Su：減數字組 R：結果字組	INT INT INT	--- --- ---
DOUBLE SIGNED BINARY WITH CARRY (雙重含正負號二進位，進位)	-CL @-CL	413		Mi：被減數字組 Su：減數字組 R：結果字組	DINT DINT DINT	--- --- ---
BCD SUBTRACT WITHOUT CARRY (BCD 減法不進位)	-B @-B	414		Mi：被減數字組 Su：減數字組 R：結果字組	WORD WORD WORD	--- --- ---
DOUBLE BCD SUBTRACT WITHOUT CARRY (雙重 BCD 減法，不進位)	-BL @-BL	415		Mi：第 1 被減數字組 Su：第 1 減數字組 R：第 1 結果字組	DWORD DWORD DWORD	--- --- ---
BCD SUBTRACT WITH CARRY (BCD 減法進位)	-BC @-BC	416		Mi：被減數字組 Su：減數字組 R：結果字組	WORD WORD WORD	--- --- ---

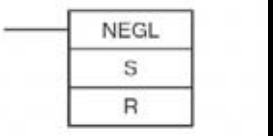
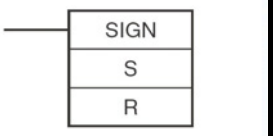
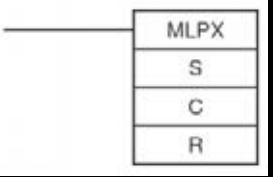
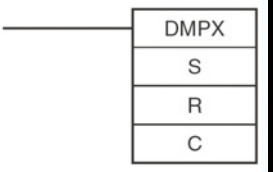
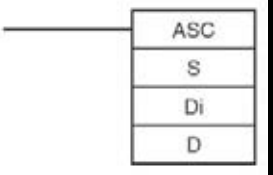
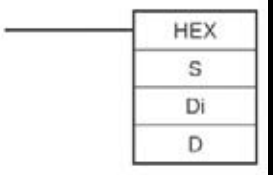
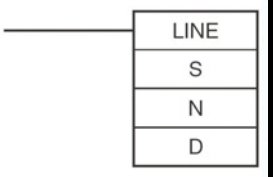
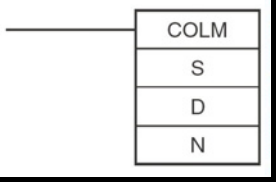
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE BCD SUBTRACT WITH CARRY (雙重 BCD 減法，進位)	-BCL @-BCL	417		Mi：第 1 被減數字組	DWORD	---
				Su：第 1 減數字組	DWORD	---
				R：第 1 結果字組	DWORD	---
SIGNED BINARY MULTIPLY (含正負號二進位乘法)	* @*	420		Md：被乘數字組	INT	---
				Mr：乘數字組	INT	---
				R：結果字組	DINT	---
DOUBLE SIGNED BINARY MULTIPLY (雙重含正負號二進位乘法)	*L @*L	421		Md：第 1 被乘數字組	DINT	---
				Mr：第 1 乘數字組	DINT	---
				R：第 1 結果字組	LINT	---
UNSIGNED BINARY MULTIPLY (無正負號二進位乘法)	*U @*U	422		Md：被乘數字組	UINT	---
				Mr：乘數字組	UINT	---
				R：結果字組	UINT	---
DOUBLE UNSIGNED BINARY MULTIPLY (雙重無正負號二進位乘法)	*UL @*UL	423		Md：第 1 被乘數字組	UDINT	---
				Mr：第 1 乘數字組	UDINT	---
				R：第 1 結果字組	ULINT	---
BCD MULTIPLY (BCD 乘法)	*B @*B	424		Md：被乘數字組	WORD	---
				Mr：乘數字組	WORD	---
				R：結果字組	DWORD	---
DOUBLE BCD MULTIPLY (雙重 BCD 乘法)	*BL @*BL	425		Md：第 1 被乘數字組	DWORD	---
				Mr：第 1 乘數字組	DWORD	---
				R：第 1 結果字組	LWORD	---

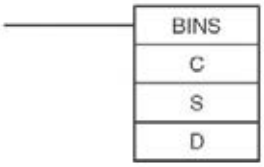
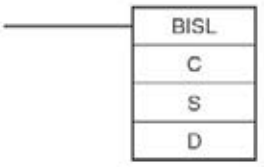
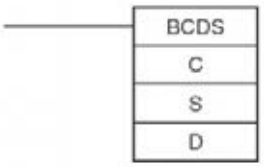
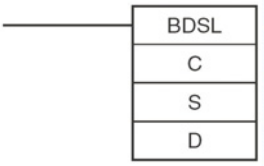
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SIGNED BINARY DIVIDE (含正負號二進位除法)	/ @/	430		Dd：被除數字組	INT	---
				Dr：除數字組	INT	---
				R：結果字組	DWORD	是(2) 在需要陣列變數時必須使用 INT
DOUBLE SIGNED BINARY DIVIDE (雙重含正負號二進位除法)	/L @/L	431		Dd：第 1 被除數字組	DINT	---
				Dr：第 1 除數字組	DINT	---
				R：第 1 結果字組	LWORD	是(2) 在需要陣列變數時必須使用 DINT
UNSIGNED BINARY DIVIDE (無正負號二進位除法)	/U @/U	432		Dd：被除數字組	UINT	---
				Dr：除數字組	UINT	---
				R：結果字組	DWORD	是(2) 在需要陣列變數時必須使用 UINT
DOUBLE UNSIGNED BINARY DIVIDE (雙重無正負號二進位除法)	/UL @/UL	433		Dd：第 1 被除數字組	UDINT	---
				Dr：第 1 除數字組	UDINT	---
				R：第 1 結果字組	LWORD	是(2) 在需要陣列變數時必須使用 UDINT
BCD DIVIDE (BCD 除法)	/B @/B	434		Dd：被除數字組	WORD	---
				Dr：除數字組	WORD	---
				R：結果字組	DWORD	是(2) 在需要陣列變數時必須使用字組

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE BCD DIVIDE (雙重 BCD 除法)	/BL @/BL	435		Dd：第 1 被除數字組	DWORD	---
				Dr：第 1 除數字組	DWORD	---
				R：第 1 結果字組	LWORD	是(2) 在需要陣列變數時必須使用 DWORD

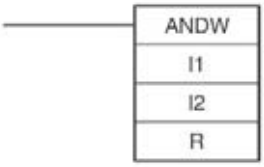
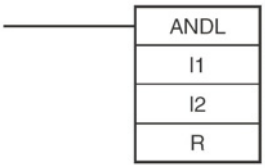
2-6-10 轉換指令

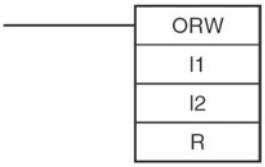
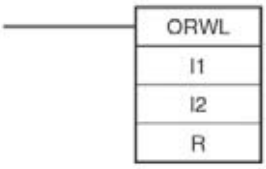
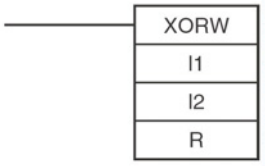
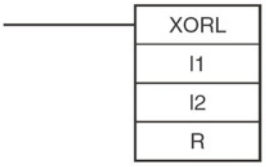
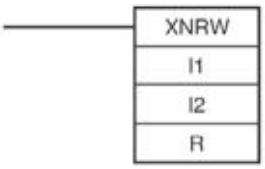
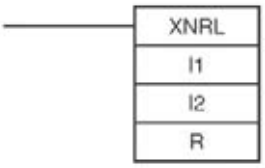
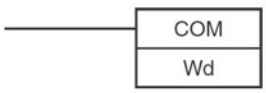
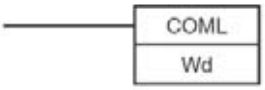
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
BCD-TO-BINARY (BCD 對二進位)	BIN @BIN	023		S：來源字組	WORD	---
				R：結果字組	UINT	---
DOUBLE BCD-TO-DOUBLE BINARY (雙重 BCD 對雙浮點二進位)	BINL @BINL	058		S：第 1 來源字組	DWORD	---
				R：第 1 結果字組	UDINT	---
BINARY-TO-BCD (二進位對 BCD)	BCD @BCD	024		S：來源字組	UINT	---
				R：結果字組	WORD	---
DOUBLE BINARY-TO-DOUBLE BCD (雙重二進位對雙重 BCD)	BCDL @BCDL	059		S：第 1 來源字組	UDINT	---
				R：第 1 結果字組	DWORD	---
2'S COMPLEMENT (2 的補數)	NEG @NEG	160		S：來源字組	WORD	---
				R：結果字組	UINT	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE 2'S COMPLEMENT (雙重 2 的補數)	NEGL @NEGL	161		S：第 1 來源字組	DWORD	---
				R：第 1 結果字組	UDINT	---
16-BIT TO 32-BIT SIGNED BINARY (16 位元到 32 位元含正負號二進位)	SIGN @SIGN	600		S：來源字組	WORD	---
				R：第 1 結果字組	DINT	---
DATA DECODER (解碼器)	MLPX @MLPX	076		S：來源字組	UINT	---
				C：控制字組	UINT	---
				R：第 1 結果字組	UINT	是(不固定)
DATA ENCODER (編碼器)	DMPX @DMPX	077		S：第 1 來源字組	UINT	是(不固定)
				R：結果字組	UINT	---
				C：控制字組	UINT	---
ASCII CONVERT (ASCII 轉換)	ASC @ASC	086		S：來源字組	UINT	---
				Di：位數指示符	UINT	---
				D：第 1 目的字組	UINT	是(3)
ASCII TO HEX (ASCII 對十六進位)	HEX @HEX	162		S：第 1 來源字組	UINT	是(2)
				Di：位數指示符	UINT	---
				D：目的字組	UINT	是(不固定)
COLUMN TO LINE (欄到行)	LINE @LINE	063		S：第 1 來源字組	WORD	是(16)
				N：位元號碼	UINT	---
				D：目的字組	UINT	---
LINE TO COLUMN (行到欄)	COLM @COLM	064		S：來源字組	WORD	---
				D：第 1 目的字組	WORD	是(16)
				N：位元號碼	UINT	---

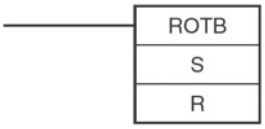
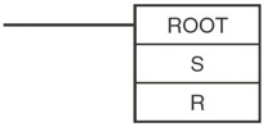
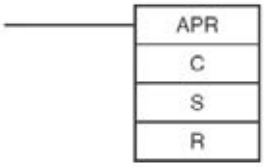
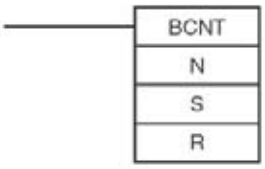
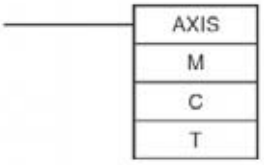
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SIGNED BCD-TO-BINARY (含正負號 BCD 對二進位)	BINS @BINS	470		C：控制字組	UINT	---
				S：來源字組	WORD	---
				D：目的字組	INT	---
DOUBLE SIGNED BCD-TO-BINARY (雙重含正負號 BCD 對二進位)	BISL @BISL	472		C：控制字組	UINT	---
				S：第 1 來源字組	DWORD	---
				D：第 1 目的字組	DINT	---
SIGNED BINARY-TO-BCD (含正負號二進位對 BCD)	BCDS @BCDS	471		C：控制字組	UINT	---
				S：來源字組	INT	---
				D：目的字組	WORD	---
DOUBLE SIGNED BINARY-TO-BCD (雙重含正負號二進位對 BCD)	BDSL @BDSL	473		C：控制字組	UINT	---
				S：第 1 來源字組	DINT	---
				D：第 1 目的字組	DWORD	---

2-6-11 邏輯指令

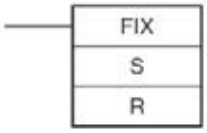
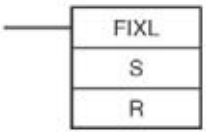
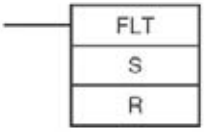
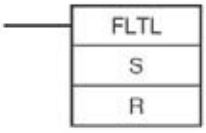
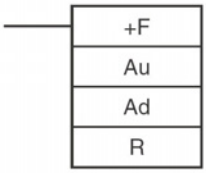
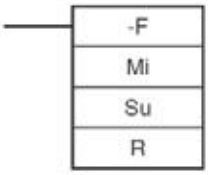
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
LOGICAL AND (邏輯 AND)	ANDW @ANDW	034		I1：輸入 1	WORD	---
				I2：輸入 2	WORD	---
				R：結果字組	WORD	---
DOUBLE LOGICAL AND (雙重邏輯 AND)	ANDL @ANDL	610		I1：輸入 1	DWORD	---
				I2：輸入 2	DWORD	---
				R：結果字組	DWORD	---

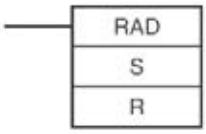
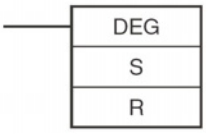
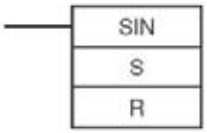
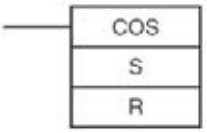
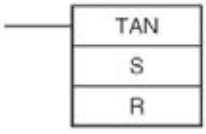
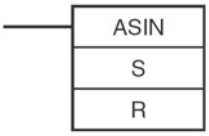
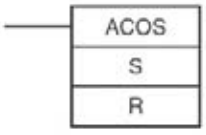
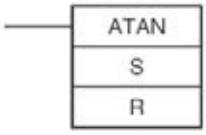
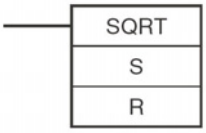
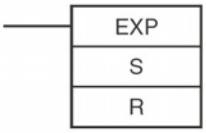
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
LOGICAL OR (邏輯 OR)	ORW @ORW	035		I1：輸入 1 I2：輸入 2 R：結果字組	WORD WORD WORD	--- --- ---
DOUBLE LOGICAL OR (雙重邏輯 OR)	ORWL @ORWL	611		I1：輸入 1 I2：輸入 2 R：結果字組	DWORD DWORD DWORD	--- --- ---
EXCLUSIVE OR (互斥 OR)	XORW @XORW	036		I1：輸入 1 I2：輸入 2 R：結果字組	WORD WORD WORD	--- --- ---
DOUBLE EXCLUSIVE OR (雙重互斥 OR)	XORL @XORL	612		I1：輸入 1 I2：輸入 2 R：結果字組	DWORD DWORD DWORD	--- --- ---
EXCLUSIVE NOR (互斥 NOR)	XNRW @XNRW	037		I1：輸入 1 I2：輸入 2 R：結果字組	WORD WORD WORD	--- --- ---
DOUBLE EXCLUSIVE NOR (雙重互斥 NOR)	XNRL @XNRL	613		I1：輸入 1 I2：輸入 2 R：結果字組	DWORD DWORD DWORD	--- --- ---
COMPLEMENT (互補)	COM @COM	029		Wd：字組	WORD	---
DOUBLE COMPLEMENT (雙重互補)	COML @COML	614		Wd：字組	DWORD	---

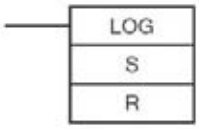
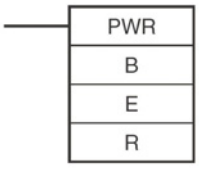
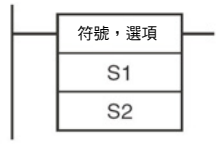
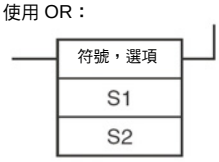
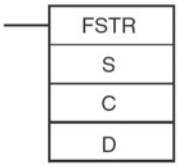
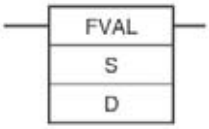
2-6-12 特殊數學指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
BINARY ROOT (二進位根)	ROTB @ROTB	620		S：第 1 來源字組	UDINT	---
				R：結果字組	UINT	---
BCD SQUARE ROOT (BCD 平方根)	ROOT @ROOT	072		S：第 1 來源字組	DWORD	---
				R：結果字組	WORD	---
ARITHMETIC PROCESS (算術步驟)	APR @APR	069		C：控制字組	UINT	是(不固定)
				S：來源資料	WORD	---
				R：結果字組	WORD	---
FLOATING POINT DIVIDE (浮點除法)	FDIV @FDIV	079		Dd：第 1 被除數字組	UDINT	---
				Dr：第 1 除數字組	UDINT	---
				R：第 1 結果字組	UDINT	---
BIT COUNTER (位元計數器)	BCNT @BCNT	067		N：字組數	UINT	---
				S：第 1 來源字組	UINT	是(不固定)
				R：結果字組	UINT	---
VIRTUAL AXIS (虛擬軸)	AXIS (僅限於FQM1)	981		M：模式名稱	WORD	---
				C：處理循環	WORD	---
				T：第 1 設定表字組	WORD	是(27)

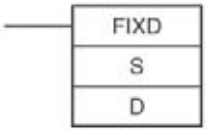
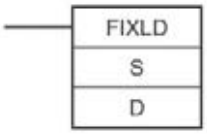
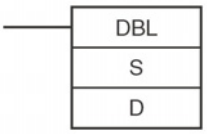
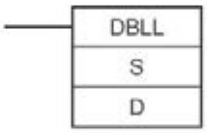
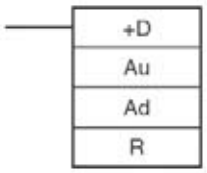
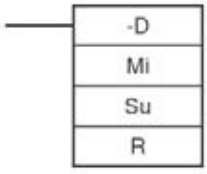
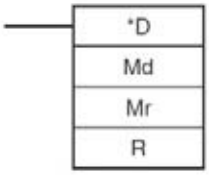
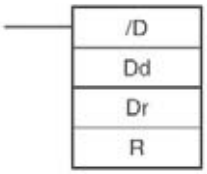
2-6-13 浮點數學指令

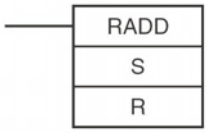
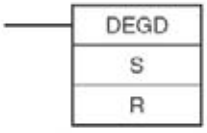
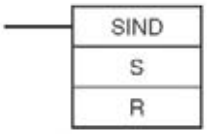
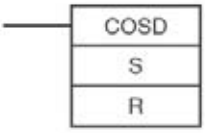
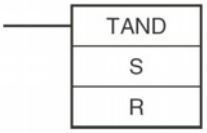
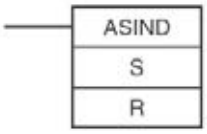
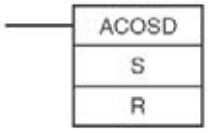
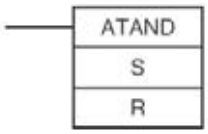
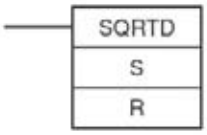
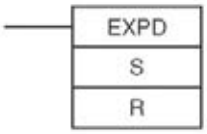
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
FLOATING TO 16-BIT (浮點對 16 位元)	FIX @FIX	450		S：第 1 來源字組	REAL	---
				R：結果字組	INT	---
FLOATING TO 32-BIT (浮點對 32 位元)	FIXL @FIXL	451		S：第 1 來源字組	REAL	---
				R：結果字組	DINT	---
16-BIT TO FLOATING (16 位元對浮點)	FLT @FLT	452		S：來源字組	INT	---
				R：第 1 結果字組	REAL	---
32-BIT TO FLOATING (32 位元對浮點)	FLTL @FLTL	453		S：第 1 來源字組	DINT	---
				R：結果字組	REAL	---
FLOATING-POINT ADD (浮點加法)	+F @+F	454		Au：第 1 加數字組	REAL	---
				Ad：第 1 加數字組	REAL	---
				R：第 1 結果字組	REAL	---
FLOATING-POINT SUBTRACT (浮點減法)	-F @-F	455		Mi：第 1 被減數字組	REAL	---
				Su：第 1 減數字組	REAL	---
				R：第 1 結果字組	REAL	---
FLOATING-POINT MULTIPLY (浮點乘法)	*F @*F	456		Md：第 1 被乘數字組	REAL	---
				Mr：第 1 乘數字組	REAL	---
				R：第 1 結果字組	REAL	---
FLOATING-POINT DIVIDE (浮點除法)	/F @/F	457		Dd：第 1 被除數字組	REAL	---
				Dr：第 1 除數字組	REAL	---
				R：第 1 結果字組	REAL	---

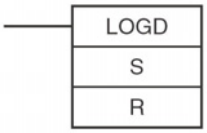
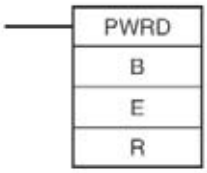
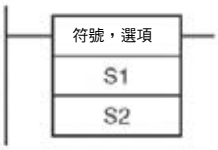
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DEGREES TO RADIANS (角度對弧度)	RAD @RAD	458		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
RADIANS TO DEGREES (弧度對角度)	DEG @DEG	459		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
SINE (正弦)	SIN @SIN	460		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
COSINE (餘弦)	COS @COS	461		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
TANGENT (正切)	TAN @TAN	462		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
ARC SINE (ARC 正弦)	ASIN @ASIN	463		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
ARC COSINE (ARC 餘弦)	ACOS @ACOS	464		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
ARC TANGENT (ARC 正切)	ATAN @ATAN	465		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
SQUARE ROOT (平方根)	SQRT @SQRT	466		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
EXPONENT (指數)	EXP @EXP	467		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
LOGARITHM (對數)	LOG @LOG	468		S：第 1 來源字組	REAL	---
				R：第 1 結果字組	REAL	---
EXPONENTIAL POWER (指數乘冪)	PWR @PWR	840		B：第 1 基數字組	REAL	---
				E：第 1 交換字組	REAL	---
				R：第 1 結果字組	REAL	---
浮點符號比較	LD, AND, OR + =F, <>F, <F, <=F, >F, >=F	329 (=F) 330 (<>F) 331 (<F) 332 (<=F) 333 (>F) 334 (>=F)	使用 LD：  使用 AND：  使用 OR： 	S1：比較資料 1	REAL	---
				S2：比較資料 2	REAL	---
FLOATING-POINT TO ASCII (浮點對 ASCII)	FSTR @FSTR	448		S：第 1 來源字組	REAL	---
				C：控制字組	UINT	是(3)
				D：目的字組	UINT	是(不固定)
ASCII TO FLOATING-POINT (ASCII 對雙浮點)	FVAL @FVAL	449		S：來源字組	UINT	是(不固定)
				D：第 1 目的字組	REAL	---

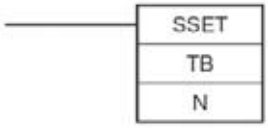
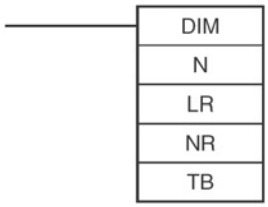
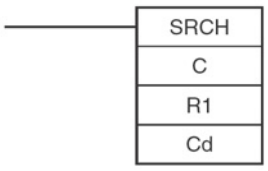
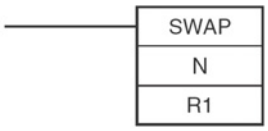
2-6-14 雙精密浮點指令

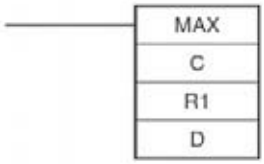
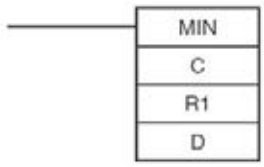
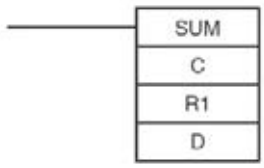
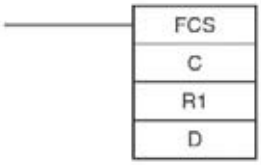
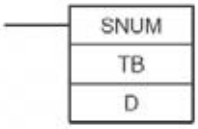
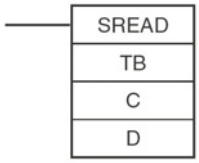
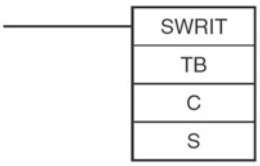
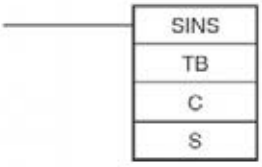
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE FLOATING TO 16-BIT BINARY (雙浮點對 16 位元二進位)	FIXD @FIXD	841		S：第 1 來源字組	LREAL	---
				D：目的字組	INT	---
DOUBLE FLOATING TO 32-BIT BINARY (雙浮點對 32 位元二進位)	FIXLD @FIXLD	842		S：第 1 來源字組	LREAL	---
				D：第 1 目的字組	DINT	---
16-BIT BINARY TO DOUBLE FLOATING (16 位元二進位對雙浮點)	DBL @DBL	843		S：來源字組	INT	---
				D：第 1 目的字組	LREAL	---
32-BIT BINARY TO DOUBLE FLOATING (32 位元二進位對雙浮點)	DBLL @DBLL	844		S：第 1 來源字組	DINT	---
				D：第 1 目的字組	DINT	---
DOUBLE FLOATING-POINT ADD (雙浮點加法)	+D @+D	845		Au：第 1 加數字組	LREAL	---
				Ad：第 1 加數字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE FLOATING-POINT SUBTRACT (雙浮點減法)	-D @-D	846		Mi：第 1 被減數字組	LREAL	---
				Su：第 1 減數字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE FLOATING-POINT MULTIPLY (雙浮點乘法)	*D @*D	847		Md：第 1 被乘數字組	LREAL	---
				Mr：第 1 乘數字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE FLOATING-POINT DIVIDE (雙浮點除法)	/D @/D	848		Dd：第 1 被除數字組	LREAL	---
				Dr：第 1 除數字組	LREAL	---
				R：第 1 結果字組	LREAL	---

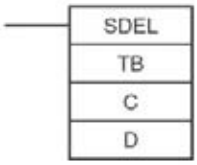
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE DEGREES TO RADIANS (雙重角度對弧度)	RADD @RADD	849		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE RADIANS TO DEGREES (雙重弧度對角度)	DEGD @DEGD	850		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE SINE (雙重正弦)	SIND @SIND	851		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE COSINE (雙重餘弦)	COSD @COSD	852		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE TANGENT (雙重正切)	TAND @TAND	853		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE ARC SINE (雙重 ARC 正弦)	ASIND @ASIND	854		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE ARC COSINE (雙重 ARC 餘弦)	ACOSD @ACOSD	855		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE ARC TANGENT (雙重 ARC 正切)	ATAND @ATAND	856		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE SQUARE ROOT (雙重 BCD 平方根)	SQRTD @SQRTD	857		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE EXPONENT (雙重指數)	EXPD @EXPD	858		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DOUBLE LOGARITHM (雙重對數)	LOGD @LOGD	859		S：第 1 來源字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE EXPONENTIAL POWER (雙重指數乘冪)	PWRD @PWRD	860		B：第 1 基數字組	LREAL	---
				E：第 1 交換字組	LREAL	---
				R：第 1 結果字組	LREAL	---
DOUBLE SYMBOL COMPARISON (雙重符號比較)	LD, AND, OR + =D, <>D, <D, <=D, >D, >=D	335 (=D) 336 (<>D) 337 (<D) 338 (<=D) 339 (>D) 340 (>=D)	使用 LD：  使用 AND：  使用 OR： 	S1：比較資料 1	LREAL	---
				S2：比較資料 2	LREAL	---

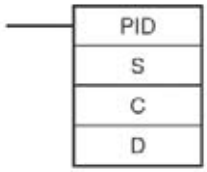
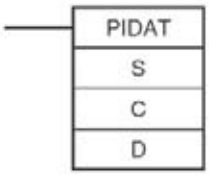
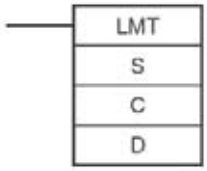
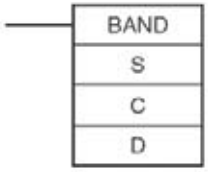
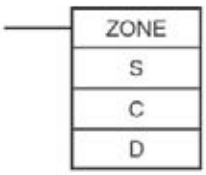
2-6-15 Table 資料處理指令

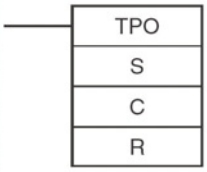
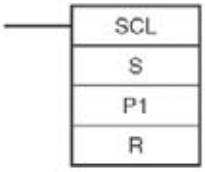
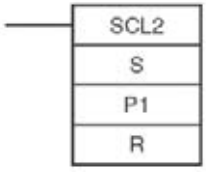
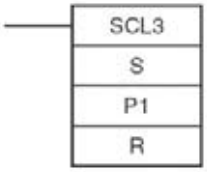
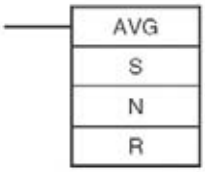
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SET STACK (設定堆疊)	SSET @SSET	630		TB：第 1 堆疊位址	UINT	是(不固定)
				N：字組數	UINT	---
PUSH ONTO STACK (堆到堆疊上)	PUSH @PUSH	632	功能區塊中不支援	TB：第 1 堆疊位址	---	---
				S：來源字組	---	---
FIRST IN FIRST OUT (先進先出)	FIFO @FIFO	633	功能區塊中不支援	TB：第 1 堆疊位址	---	---
				D：目的字組	---	---
LAST IN FIRST OUT (後進先出)	LIFO @LIFO	634	功能區塊中不支援	TB：第 1 堆疊位址	---	---
				D：目的字組	---	---
DIMENSION RECORD TABLE (記錄表長度)	DIM @DIM	631		N：表格號碼	#僅限於#正的 10 進制值	---
				LR：每個記錄的長度	UINT	---
				NR：記錄數	UINT	---
				TB：第 1 表格字組	UINT	是(不固定)
SET RECORD LOCATION (設定記錄位置)	SETR @SETR	635	功能區塊中不支援	N：表格號碼	---	---
				R：記錄號碼	---	---
				D：目的指標暫存器	---	---
GET RECORD NUMBER (取得記錄號碼)	GETR @GETR	636	功能區塊中不支援	N：表格號碼	---	---
				IR：包含指標	---	---
				D：目的字組	---	---
DATA SEARCH (資料搜尋)	SRCH @SRCH	181		C：第 1 控制字組	UDINT	---
				R1：範圍中的第 1 個字組	UINT	是(不固定)
				Cd：比較資料	WORD	---
SWAP BYTES (置換位元組)	SWAAP @SWP	637		字組數	UINT	---
				R1：範圍中的第 1 個字組	UINT	是(不固定)

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
FIND MAXIMUM (搜尋最大值)	MAX @MAX	182		C：第 1 控制字組	UDINT	---
				R1：範圍中的第 1 個字組	UINT	是(不固定)
				D：目的字組	UINT	---
FIND MINIMUM (搜尋最小值)	MIN @MIN	183		C：第 1 控制字組	UDINT	---
				R1：範圍中的第 1 個字組	UINT	是(不固定)
				D：目的字組	UINT	---
SUM (總和)	SUM @SUM	184		C：第 1 控制字組	UDINT	---
				R1：範圍中的第 1 個字組	UINT	是(不固定)
				D：第 1 目的字組	UDINT	---
FRAME CHECK SUM (框校對總和)	FCS @FCS	180		C：第 1 控制字組	UDINT	---
				R1：範圍中的第 1 個字組	UINT	是(不固定)
				D：第 1 目的字組	UINT	---
STACK SIZE READ (堆疊大小讀取)	SNUM @SNUM	638		TB：第一堆疊位址	UINT	是(不固定)
				D：目的字組	UINT	---
STACK DATA READ (堆疊資料讀取)	SREAD @SREAD	639		TB：第一堆疊位址	UINT	是(不固定)
				C：差值	UINT	---
				D：目的字組	UINT	---
STACK DATA OVERWRITE (堆疊資料過長)	SWRIT @SWRIT	640		TB：第一堆疊位址	UINT	是(不固定)
				C：差值	UINT	---
				S：來源資料	UINT	---
STACK DATA INSERT (堆疊資料插入)	SINS @SINS	641		TB：第一堆疊位址	UINT	是(不固定)
				C：差值	UINT	---
				S：來源資料	UINT	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
STACK DATA DELETE (堆疊資料刪除)	SDEL @SDEL	642		TB：第一堆疊位址	UINT	是(不固定)
				C：差值	UINT	---
				D：目的字組	UINT	---

2-6-16 資料控制指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
PID CONTROL (PID 控制)	PID	190		S：輸入字組	UINT	---
				C：第 1 參數字組	WORD	是(39)
				D：輸出字組	UINT	---
PID CONTROL WITH AUTO TUNING (自動調諧 PID 控制)	PIDAT	191		S：輸入字組	UINT	---
				C：第 1 參數字組	WORD	是(41)
				D：輸出字組	UINT	---
LIMIT CONTROL (超限控制)	LMT @LMT	680		S：輸入字組	INT	---
				C：第 1 限制字組	DINT	是(2)
				D：輸出字組	INT	---
DEAD BAND CONTROL (無效帶控制)	BAND @BAND	681		S：輸入字組	INT	---
				C：第 1 限制字組	UINT	是(2)
				D：輸出字組	UINT	---
DEAD ZONE CONTROL (無效區控制)	ZONE @ZONE	682		S：輸入字組	INT	---
				C：第 1 限制字組	UDINT	是(2)
				D：輸出字組	UINT	---

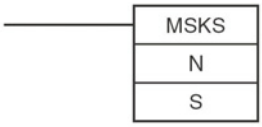
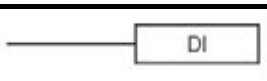
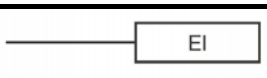
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
TIME-PROPORTIONAL OUTPUT (時間比例輸出)	TPO	685		S：輸入字組	UINT	---
				C：第 1 參數字組	WORD	是(7)
				R：脈衝輸出位元	BOOL	---
SCALING (定比)	SCL @SCL	194		S：輸入字組	UINT	---
				P1：第 1 參數字組	LWORD	是(4)
				R：結果字組	WORD	---
SCALING 2 (定比 2)	SCL2 @SCL2	486		S：輸入字組	INT	---
				P1：第 1 參數字組	WORD	是(3)
				R：結果字組	WORD	---
SCALING 3 (定比 3)	SCL3 @SCL3	487		S：輸入字組	WORD	---
				P1：第 1 參數字組	WORD	是(5)
				R：結果字組	INT	---
AVERAGE (平均)	AVG	195		S：輸入字組	UINT	---
				N：循環數	UINT	---
				R：結果字組	UINT	是(不固定)

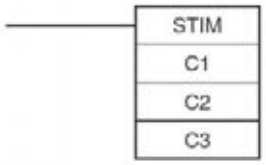
2-6-17 副程式編號

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SUBROUTINE CALL (副程式呼叫)	SBS @SBS	091	功能區塊中不支援	N：副程式號碼	---	---
SUBROUTINE ENTRY (副程式進入)	SBN	092	功能區塊中不支援	N：副程式號碼	---	---

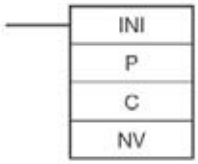
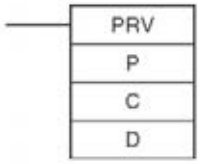
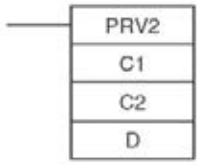
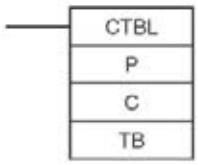
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SUBROUTINE RETURN (副程式返回)	RET	093	功能區塊中不支援		---	---
MACRO (巨集)	MCRO @MCRO	099	功能區塊中不支援	N：副程式號碼	---	---
				S：第 1 輸入參數字組	---	---
				D：第 1 輸出參數字組	---	---
GLOBAL SUBROUTINE CALL (全域副程式呼叫)	GSBS 2GSBS	750	功能區塊中不支援	N：副程式號碼	---	---
GLOBAL SUBROUTINE ENTRY (全域副程式進入)	GSBN	751	功能區塊中不支援	N：副程式號碼	---	---
GLOBAL SUBROUTINE RETURN (全域副程式返回)	GRET	752	功能區塊中不支援		---	---
JUMP TO SUBROUTINE (跳越到副程式)	JSB (僅限於FQM1)	982	功能區塊中不支援	---	---	---

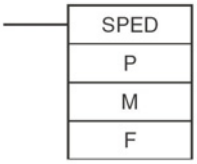
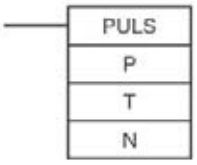
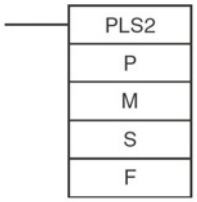
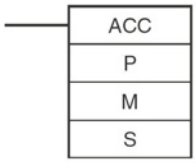
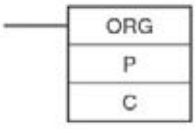
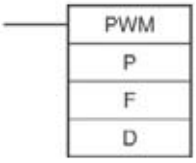
2-6-18 中斷控制指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SET INTERRUPT MASK (設定中斷遮罩)	MSKS @MSKS	690		N：中斷識別碼	#僅限於#正的 10 進制值	---
				S：中斷資料	UINT	---
READ INTERRUPT MASK (讀取中斷遮罩)	MSKR @MSKR	692		N：中斷識別碼	#僅限於#正的 10 進制值	---
				D：目的字組	UINT	---
CLEAR INTERRUPT (清除中斷)	CLI @CLI	691		N：中斷識別碼	#僅限於#正的 10 進制值	---
				S：中斷資料	UINT	---
DISABLE INTERRUPTS (取消中斷)	DI @DI	693		---	---	---
ENABLE INTERRUPTS (啟動中斷)	EI	694		---	---	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
INTERVAL TIMER (區間計時器)	STIM @STIM (僅限於FQM1)	980		C1：控制資料#1	WORD	---
				C2：控制資料#2	INT	---
				C3：控制資料#3	INT	---

2-6-19 高速計數器及脈衝輸出指令(僅限於 CJ1M-CPU21/22/23)

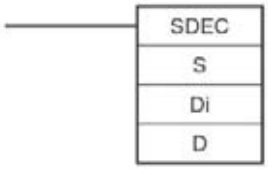
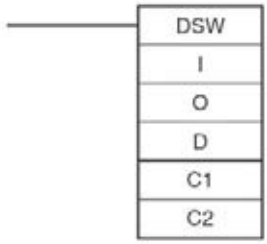
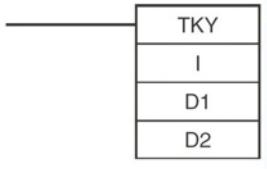
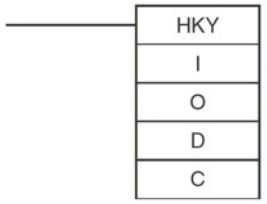
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
MODE CONTROL (模式控制)	INI @INI	880		P：連接埠指定	WORD	---
				C：控制資料	UINT	---
				NV:有新 PCV 的第 1 個字組	DWORD	---
HIGH-SPEED COUNTER PV READ (高速計數器 PV (現在值)讀取)	PRV @PRV	881		P：連接埠指定	#僅限於#正的 10 進制值	---
				C：控制資料	#僅限於#正的 10 進制值	---
				D：第 1 目的字組	WORD	是(1 或 2)
COUNTER FREQUENCY CONVERT (計數器頻率轉換) (僅限於 CJ1M CPU 模組版本 2.0 以上)	PRV2	883		C1：控制資料	WORD	---
				C2：脈衝/轉數	UINT	---
				D：第 1 目的字組	UDINT	---
COMPARISON TABLE LOAD (比較表載入)	CTBL @CTBL	882		P：連接埠指定	#僅限於#正的 10 進制值	---
				C：控制資料	#僅限於#正的 10 進制值	---
				TB：第 1 比較表字組	LWORD	是(不固定)

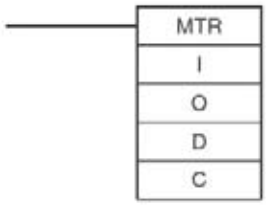
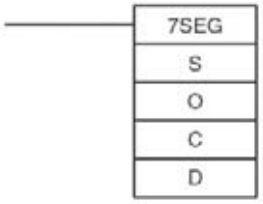
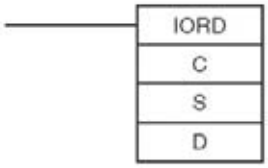
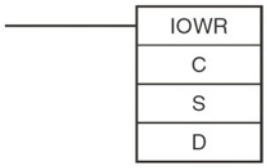
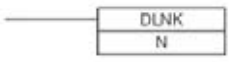
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SPEED OUTPUT (速度輸出)	SPED @SPED	885		P：連接埠指定	UINT	---
				M：輸出模式	WORD	---
				F：第 1 脈衝頻率字組	UDINT	---
SET PULSES (設定脈衝)	PULS @PULS	886		P：連接埠指定	UINT	---
				T：脈衝類型	UINT	---
				N：脈衝數	DINT	---
PULSE OUTPUT (脈衝輸出)	PLS2 @PLS2	887		P：連接埠指定	#僅限於#正的 10 進制值	---
				M：輸出模式	#僅限於#正的 10 進制值	---
				S：設定表的第 1 個字組	WORD	是(6)
				F：起始頻率的第 1 個字組	UDINT	---
ACCELERATION CONTROL (加速控制)	ACC @ACC	888		P：連接埠指定	#僅限於#正的 10 進制值	---
				M：輸出模式	#僅限於#正的 10 進制值	---
				S：設定表的第 1 個字組	WORD	是(3)
ORIGIN SEARCH (原點搜尋)	ORG @ORG	889		P：連接埠指定	#僅限於#正的 10 進制值	---
				C：控制資料	#僅限於#正的 10 進制值	---
PULSE WITH VARIABLE DUTY FACTOR (有可變作用因數的脈衝)	PWM @PWM	891		P：連接埠指定	#僅限於#正的 10 進制值	---
				F：頻率	#僅限於#正的 10 進制值	---
				D：作用因數	#僅限於#正的 10 進制值	---

2-6-20 步驟指令

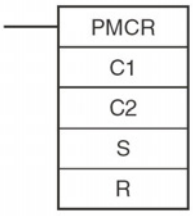
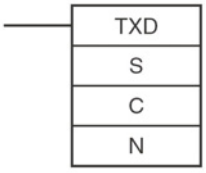
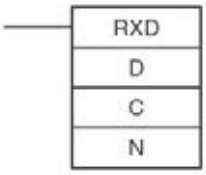
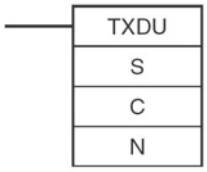
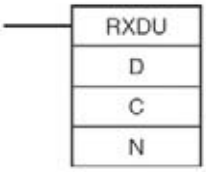
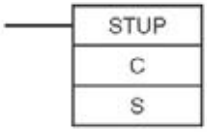
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
STEP DEFINE (步驟定義)	STEP	008	功能區塊中不支援	B：位元	---	---
STEP START (步驟開始)	SNXT	009	功能區塊中不支援	B：位元	---	---

2-6-21 基本 I/O 模組指令

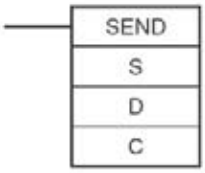
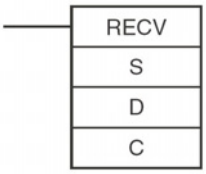
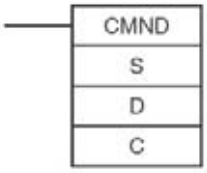
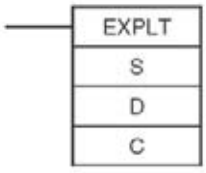
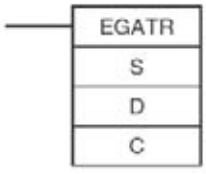
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
I/O REFRESH (I/O 更新)	IORF @IORF	097	功能區塊中不支援	St：起始字組	---	---
				E：結束字組	---	---
7-SEGMENT DECODER (7 段解碼器)	SDEC @SDEC	078		S：來源字組	UINT	---
				Di：Digit designator	UINT	---
				D：第 1 目的字組	UINT	是(不固定)
DIGITAL SWITCH INPUT (數位交換輸入)	DSW	210		I：資料輸入字組(D0 至 D3)	UINT	---
				O：輸出字組	UINT	---
				D：第 1 結果字組	WORD	---
				C1：位數	UINT	---
				C2：系統字組	WORD	---
TEN KEY INPUT (十鍵輸入)	TKY	211		I：資料輸入字組	UINT	---
				D1：第 1 暫存器字組	UDINT	---
				D2：按鍵輸入字組	UINT	---
HEXADECIMAL KEY INPUT (十六進位鍵輸入)	HKY	212		I：資料輸入字組	UINT	---
				O：輸出字組	UINT	---
				D：第 1 暫存器字組	WORD	是(3)
				C：系統字組	WORD	---

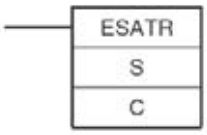
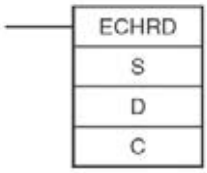
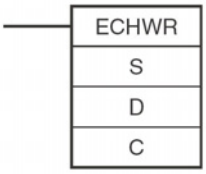
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
MATRIX INPUT (矩陣輸入)	MTR	213		I：資料輸入字組	UINT	---
				O：輸出字組	UINT	---
				D：第 1 目的字組	ULINT	是(4)
				C：系統字組	WORD	---
7-SEGMENT DISPLAY OUTPUT (7 段顯示器輸出)	7SEG	214		S：第 1 來源字組	WORD	是(2)
				O：輸出字組	UINT	---
				C：控制資料	#僅限於#正的 10 進制值	---
				D：系統字組	WORD	---
INTELLIGENT I/O READ (智慧 I/O 讀取)	IORD @IORD	222		C：控制資料	UINT	---
				S：傳送來源及字組數	UDINT	是(2) 在需要陣列變數時必須使用 UINT
				D：傳送目的及字組數	UINT	是(不固定)
INTELLIGENT I/O WRITE (智慧 I/O 編寫)	IOWR @IOWR	223		C：控制資料	UINT	---
				S：傳送來源及字組數	WORD	是(不固定)
				D：傳送目的及字組數	UINT	是(2) 在需要陣列變數時必須使用 UINT
CPU BUS UNIT I/O REFRESH (CPU 匯流排模組 I/O 更新)	DLNK @DLNK	226		N：模組號碼	UINT	---

2-6-22 序列通訊指令

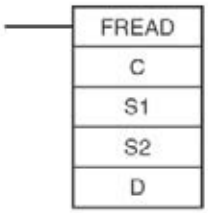
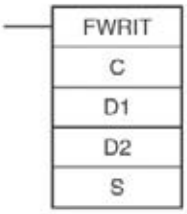
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
PROTOCOL MACRO (通訊協定巨集)	@PMCR PMCR	260		C1：控制字組 1	UINT	---
				C2：控制字組 2	UINT	---
				S：第 1 傳送字組	UINT	是(不固定)
				R：第 1 接收字組	UINT	是(不固定)
TRANSMIT (傳送)	TXD @TXD	236		S：第 1 來源字組	UINT	是(不固定)
				C：控制字組	UINT	---
				N：0000 到 0100 十六進位(0 到 256 十進位)位元組數	UINT	---
RECEIVE (接收)	RXD @RXD	235		D：第 1 目的字組	UINT	是(不固定)
				C：控制字組	UINT	---
				N：0000 到 0100 十六進位(0 到 256 十進位)位元組數	UINT	---
TRANSMIT VIA SERIAL COMMUNICATIONS UNIT (經由序列通訊模組傳送)	TXDU @TXDU	256		S：第 1 傳送字組	UINT	是(不固定)
				C：第 1 控制字組	UDINT	---
				N：傳送位元組數(4 位數 BCD)	UINT	---
RECEIVE VIA SERIAL COMMUNICATIONS UNIT (經由序列通訊模組接收)	RXDU @RXDU	255		D：第 1 目的字組	UINT	是(不固定)
				C：第 1 控制字組	UDINT	---
				N：儲存位元組數	UINT	---
CHANGE SERIAL PORT SETUP (變更序列埠設定)	@STUP STUP	237		C：控制字組(埠)	UINT	---
				S：第 1 傳送字組	UINT	是(不固定)

2-6-23 網路指令

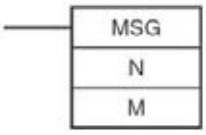
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
NETWORK SEND (網路傳送)	SEND @SEND	090		S：第 1 來源字組	UINT	是(不固定)
				D：第 1 目的字組	UINT	指定有 AT 設定的遠端節點的位址
				C：第 1 控制字組	WORD	是(5)
NETWORK RECEIVE (網路接收)	RECV @RECV	098		S：第 1 來源字組	UINT	指定有 AT 設定的遠端節點的位址
				D：第 1 目的字組	UINT	是(不固定)
				C：第 1 控制字組	WORD	是(5)
DELIVER COMMAND (遞送指令)	CMND @CMND	490		S：第 1 指令字組	UINT	是(不固定)
				D：第 1 回應字組	UINT	是(不固定)
				C：第 1 控制字組	WORD	是(6)
EXPLICIT MESSAGE SEND (明示訊息傳送)	EXPLT	720		S：傳送訊息的第 1 個字組	WORD	是(不固定)
				D：接收訊息的第 1 個字組	WORD	是(不固定)
				C：第 1 控制字組	LWORD	是(4) 在需要陣列變數時必須使用字組
EXPLICIT GET ATTRIBUTE (明示取得屬性)	EGATR	721		S：傳送訊息的第 1 個字組	ULINT	是(4) 在需要陣列變數時必須使用字組
				D：接收訊息的第 1 個字組	WORD	是(不固定)
				C：第 1 控制字組訊息	LWORD	是(4)

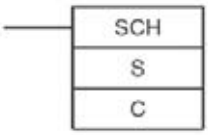
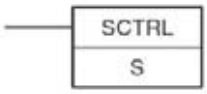
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
EXPLICIT SET ATTRIBUTE (明示設定屬性)	ESATR	722		S：傳送訊息的第 1 個字組	WORD	是(不固定)
				C：第 1 控制字組	WORD	是(3)
EXPLICIT WORD READ (明示字組讀取)	ECHRD	723		S：遠端 CPU 模組中的第 1 來源字組	UINT	使用 AT 設定時指定目的位址
				D：本地 CPU 模組中的第 1 目的字組	UINT	是(2)
				C：第 1 控制字組	WORD	是(5)
EXPLICIT WORD WRITE (明示字組編寫)	ECHWR	724		S：本地 CPU 模組中的第 1 來源字組	UINT	是(不固定)
				D：遠端 CPU 模組中的第 1 目的字組	UINT	使用 AT 設定時指定目的位址
				C：第 1 控制字組	WORD	是(5)

2-6-24 檔案記憶體指令

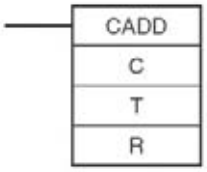
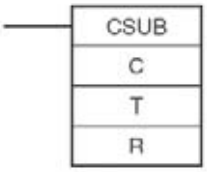
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
READ DATA FILE (讀取資料檔案)	FREAD @FREAD	700		C：控制字組	UINT	---
				S1：第 1 來源字組	LWORD	是(2) 在需要陣列變數時必須使用 DWORD
				S2：檔名	UINT	是(39)
				D：第 1 目的字組	UINT	是(不固定)
WRITE DATA FILE (編寫資料檔案)	@FWRIT FWRIT	701		C：控制字組	LWORD	---
				D1：第 1 目的字組	UINT	是(2) 在需要陣列變數時必須使用 DWORD
				D2：檔名	UINT	是(39)
				S：第 1 來源字組	UINT	是(不固定)

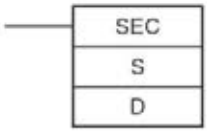
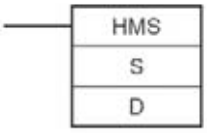
2-6-25 顯示器指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
DISPLAY MESSAGE (顯示器訊息)	MSG @MSG	046		N：訊息號碼	UINT	---
				M：第 1 訊息字組	UINT	是(16)

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SEVEN-SEGMENT LED WORD DATA DISPLAY (7 段 LED 文字資料顯示器)	SCH @SCH	047		S：訊息資料	WORD	---
				C：最左/最右 2 位數指定	UINT	---
SEVEN-SEGMENT LED CONTROL (7 段 LED 控制)	SCTRL @SCTRL	048		S：控制資料	UINT	---

2-6-26 時鐘指令

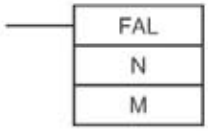
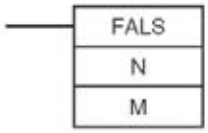
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
CALENDAR ADD (日曆加法)	CADD @CADD	730		C：第 1 日曆字組	WORD	是(3)
				T：第 1 時間字組	DWORD	是(2) 在需要陣列變數時必須使用字組
				R：第 1 結果字組	WORD	是(3)
CALENDAR SUBTRACT (日曆減法)	CSUB @CSUB	731		C：第 1 日曆字組	WORD	是(3)
				T：第 1 時間字組	DWORD	是(2) 在需要陣列變數時必須使用字組
				R：第 1 結果字組	WORD	是(3)

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
HOURS TO SECONDS (小時換算成秒)	SEC @SEC	065		S：第 1 來源字組	DWORD	是(2) 在需要陣列變數時必須使用字組
				D：第 1 目的字組	DWORD	是(2) 在需要陣列變數時必須使用字組
SECONDS TO HOURS (秒換算成小時)	HMS @HMS	066		S：第 1 來源字組	DWORD	是(2) 在需要陣列變數時必須使用字組
				D：第 1 目的字組	DWORD	是(2) 在需要陣列變數時必須使用字組
CLOCK ADJUSTMENT (時鐘調整)	DATE @DATE	735		S：第 1 來源字組	LWORD	是(2) 在需要陣列變數時必須使用字組

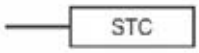
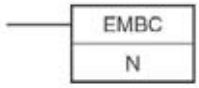
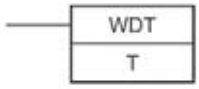
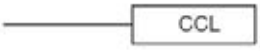
2-6-27 除錯指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
TRACE MEMORY SAMPLING (追蹤記憶體取樣)	TRSM	045		---	---	---

2-6-28 故障診斷指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
FAILURE ALARM (故障警報)	FAL @FAL	006		N：FAL 號碼	#僅限於#正的 10 進制值	---
				M：產生的第 1 訊息字組或錯誤碼 (#0000 到#FFFF)	WORD	---
SEVERE FAILURE ALARM (嚴重故障警報)	FALS	007		N：FALS 號碼	#僅限於#正的 10 進制值	---
				M：產生的第 1 訊息字組或錯誤碼 (#0000 到#FFFF)	WORD	---
FAILURE POINT DETECION (故障點偵測)	FPD	269	功能區塊中不支援	C：控制字組	---	---
				T：監控時間	---	---
				R：第 1 暫存器字組	---	---

2-6-29 其他指令

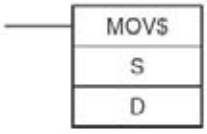
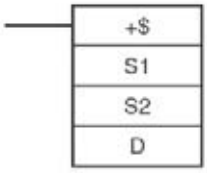
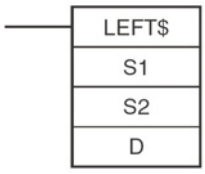
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
SET CARRY (設定進位)	STC @STC	040		---	---	---
CLEAR CARRY (清除進位)	CLC @CLC	041		---	---	---
SELECT EM BANK (選取 EM 記憶庫)	EMBC @EMBC	281		N：EM 記憶庫號碼	UINT	---
EXTEND MAXIMUM CYCLE TIME (延長最大循環時間)	WDT @WDT	094		T：計時器設定	僅限常數	---
SAVE Condition FlagS (SAVE 條件旗標)	CCS @CCS	282		---	---	---
LOAD Condition FlagS (LOAD 條件旗標)	CCL @CCL	283		---	---	---
CONVERT ADDRESS FROM CV (從 CV 轉換位址)	FRMCV @FRMCV	284	功能區塊中不支援	S：包含 CV 系列記憶體位址的字組	---	---
				D：目的指標暫存器	---	---
CONVERT ADDRESS TO CV (轉換位址至 CV)	TOCV @TOCV	285	功能區塊中不支援	S：包含 CS 系列記憶體位址的指標暫存器	---	---
				D：目的字組	---	---
DISABLE PERIPHERAL SERVICING (停用週邊服務)	IOSP @IOSP	287		---	---	---
ENABLE PERIPHERAL SERVICING (啟動週邊服務)	IORS	288		---	---	---

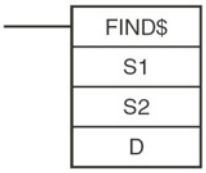
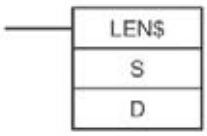
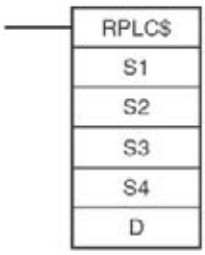
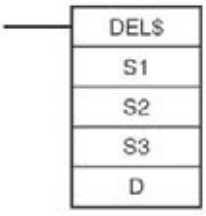
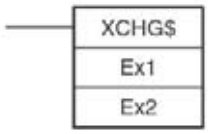
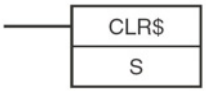
2-6-30 區塊程式編輯指令

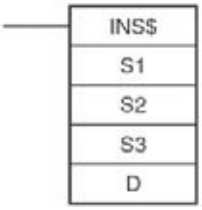
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求(要求的字組資料大小顯示在括弧中)
BLOCK PROGRAM BEGIN (區塊程式開始)	BPRG	096	功能區塊中不支援	N：區塊程式號碼	---	---
BLOCK PROGRAM END (區塊程式結束)	BEND	801	功能區塊中不支援	---	---	---
BLOCK PROGRAM PAUSE (區塊程式暫停)	BPPS	811	功能區塊中不支援	N：區塊程式號碼	---	---
BLOCK PROGRAM RESTART (區塊程式 RESTART)	BPRS	812	功能區塊中不支援	N：區塊程式號碼	---	---
CONDITIONAL BLOCK EXIT (條件式區塊退出)	CONDITION EXIT	806	功能區塊中不支援	---	---	---
CONDITION EXIT (條件退出)	EXIT Bit operand	806	功能區塊中不支援	B：位元運算元	---	---
CONDITIONAL BLOCK EXIT (NOT) (條件式區塊退出)	EXIT NOT Bit operand	806	功能區塊中不支援	B：位元運算元	---	---
CONDITIONAL BLOCK BRANCHING (條件式區塊分支)	CONDITION IF	802	功能區塊中不支援	---	---	---
CONDITIONAL BLOCK BRANCHING (條件式區塊分支)	IF Bit operand	802	功能區塊中不支援	B：位元運算元	---	---
CONDITIONAL BLOCK BRANCHING (NOT) (條件式區塊分支)	IF NOT Bit operand	802	功能區塊中不支援	B：位元運算元	---	---
CONDITIONAL BLOCK BRANCHING (ELSE) (條件式區塊分支)	ELSE	803	功能區塊中不支援	---	---	---
CONDITIONAL BLOCK BRANCHING END (條件式區塊分支結束)	IEND	804	功能區塊中不支援	---	---	---
ONE CYCLE AND WAIT (一個循環並等待)	CONDITION WAIT	805	功能區塊中不支援	---	---	---
ONE CYCLE AND WAIT (一個循環並等待)	WAIT Bit operand	805	功能區塊中不支援	B：位元運算元	---	---
ONE CYCLE AND WAIT (NOT) (一個循環並等待)	WAIT NOT Bit operand	805	功能區塊中不支援	B：位元運算元	---	---
TIMER WAIT (計時器等待)	TIMW (BCD)	813	功能區塊中不支援	N：計時器號碼	---	---
				SV：設定值	---	---
	TIMWX (BIN)	816	功能區塊中不支援	N：計時器號碼	---	---
				SV：設定值	---	---
COUNTER WAIT (計數器等待)	CNTW (BCD)	814	功能區塊中不支援	N：計數器號碼	---	---
				SV：設定值	---	---
				I：計數器輸入	---	---
	CNTWX (BIN)	817	功能區塊中不支援	N：計數器號碼	---	---
				SV：設定值	---	---
				I：計數器輸入	---	---

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求(要求的字組資料大小顯示在括弧中)
HIGH-SPEED TIMER WAIT (高速計時器等待)	TMHW (BCD)	815	功能區塊中不支援	N：計時器號碼	---	---
				SV：設定值		
	TMHWX (BIN)	818	功能區塊中不支援	N：計時器號碼	---	---
				SV：設定值	---	---
LOOP (迴路)	LOOP	809	功能區塊中不支援	---	---	---
LEND (借出)	LEND	810	功能區塊中不支援	---	---	---
LEND (借出)	LEND Bit operand	810	功能區塊中不支援	B：位元運算元	---	---
LEND NOT (不借出)	LEND NOT Bit operand	810	功能區塊中不支援	B：位元運算元	---	---

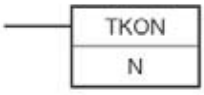
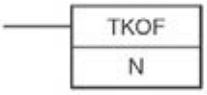
2-6-31 本文字串處理指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
MOV STRING (移動字串)	MOV$ @MOV$	664		S：第 1 來源字組	UINT	是(不固定)
				D：第 1 目的字組	UINT	是(不固定)
CONCATENATE STRING (串接字串)	+$ @+$	656		S1：本文字串 1	INT	是(不固定)
				S2：本文字串 2	INT	是(不固定)
				D：第 1 目的字組	INT	是(不固定)
GET STRING LEFT (自左取得字串)	LEFT$ @LEFT$	652		S1：本文字串第一個字組	UINT	是(不固定)
				S2：字元數	UINT	---
				D：第 1 目的字組	UINT	是(不固定)
GET STRING RIGHT (自右取得字串)	RGHT$ @RGHT$	653		S1：本文字串第一個字組	UINT	是(不固定)
				S2：字元數	UINT	---
				D：第 1 目的字組	UINT	是(不固定)

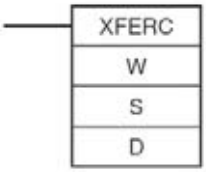
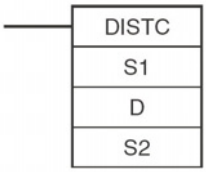
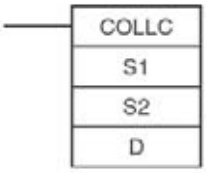
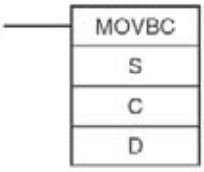
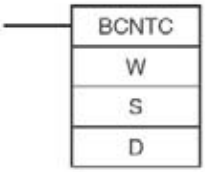
指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
GET STRING MIDDLE (自中間取得字串)	MID$ @MID$	654		S1：本文字串第一個字組	UINT	是(不固定)
				S2：字元數	UINT	---
				S3：開始位置	UINT	---
				D：第一目的字組	UINT	是(不固定)
FIND IN STRING (在字串中搜尋)	FIND$ @FIND$	660		S1：來源本文字串第一個字組	UINT	是(不固定)
				S2：找到本文字串第一個字組	UINT	是(不固定)
				D：第一目的字組	UINT	---
STRING LENGTH (字串長度)	LEN$ @LEN$	650		S：本文字串第一個字組	UINT	是(不固定)
				D：第一目的字組	UINT	---
REPLACE IN STRING (在字串中取代)	RPLC$ @RPLC$	661		S1：本文字串第一個字組	UINT	是(不固定)
				S2：取代本文字串第一個字組	UINT	是(不固定)
				S3：字元數	UINT	---
				S4：開始位置	UINT	---
				D：第一目的字組	UINT	是(不固定)
DELETE STRING (刪除字串)	DEL$ @DEL$	658		S1：本文字串第一個字組	UINT	是(不固定)
				S2：字元數	UINT	---
				S3：開始位置	UINT	---
				D：第一目的字組	UINT	是(不固定)
EXCHANGE STRING (交換字串)	XCHG$ @XCHG$	665		第 1 交換字組 1	UINT	是(不固定)
				Ex2:第 1 交換字組 2	UINT	是(不固定)
CLEAR STRING (清除字串)	CLR$ @CLR$	666		S：本文字串第一個字組	UINT	是(不固定)

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求(要求的字組資料大小顯示在括弧中)
INSERT INTO STRING (插入到字串中)	INS$ @INS$	657		S1：基本本文字串第一個字組	UINT	是(不固定)
				S2：插入的本文字串第一個字組	UINT	是(不固定)
				S3：開始位置	UINT	---
				D：第一目的字組	UINT	是(不固定)
String Comparison (字串比較)	LD,AND, OR + =$,<>$,<$,<=$,>$,>=$	670 (=$) 671 (<>$) 672 (<$) 673 (<=$) 674 (>$) 675 (>=$)		S1：本文字串 1	UINT	是(不固定)
				S2：本文字串 2	UINT	是(不固定)

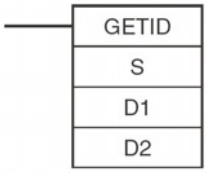
2-6-32 工作控制指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求(要求的字組資料大小顯示在括弧中)
TASK ON (工作 ON)	TKON @TKON	820		N：工作號碼	#僅限於#正的 10 進制值	---
TASK OFF (工作 OFF)	TKOF @TKOF	821		N：工作號碼	#僅限於#正的 10 進制值	---

2-6-33 型號轉換指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
BLOCK TRANSFER (區塊傳送)	XFERC @XFERC	565		W：字組數(BCD)	WORD	---
				S：第 1 來源字組	WORD	是(不固定)
				D：第 1 目的字組	WORD	是(不固定)
SINGLE WORD DISTRIBUTE (單一字組分配)	DISTC @DISTC	566		S1：來源字組	WORD	---
				D：目的資料庫位址	WORD	是(不固定)
				S2：差值(BCD)	WORD	---
DATA COLLECT (資料收集)	COLLC @COLLC	567		S1：來源資料庫位址	WORD	是(不固定)
				S2：差值(BCD)	WORD	---
				D：目的字組	WORD	---
MOVE BIT (移動位元)	MOVBC @MOVBC	568		S：來源字組或資料	WORD	---
				C：控制字組(BCD)	WORD	---
				D：目的字組	WORD	---
BIT COUNTER (位元計數器)	BCNTC @BCNTC	621		W：字組數(BCD)	WORD	---
				S：第 1 來源字組	UINT	是(不固定)
				D：結果字組	WORD	---

2-6-34 功能區塊的特殊指令

指示	數值指令	功能碼	符號	運算元	支援的變數資料類型	AT 設定或陣列變數要求 (要求的字組資料大小顯示在括弧中)
GET VARIABLE ID (取得變數 ID)	GETID @GETIC	286		來源變數或位址	WORD	---
				變數類別(I/O 記憶區)碼	WORD	---
				字組位址	WORD	---

2-7 CPU 模組功能區塊規格

版本 3.0 以上的 CS/CJ 系列 CS1-H、CJ1- H、及 CJ1M CPU 模組中所使用的功能區塊的規格如下表所示。其他規格請參考 CS/CJ 系列的其他操作手冊。

2-7-1 規格

CS1-H CPU 模組

項目		規格								
型號		CS1H-CPU67H	CS1H-CPU66H	CS1H-CPU65H	CS1H-CPU64H	CS1H-CPU63H	CS1G-CPU45H	CS1G-CPU44H	CS1G-CPU43H	CS1G-CPU42H
I/O 點		5,120						1,280	960	
程式容量(步驟)		250K	120K	60K	30K	20K	60K	30K	20K	10K
資料記憶體		32K 個字組								
擴充資料記憶體		32K 個字組 x 13 個記憶庫 E0_00000 到 E0_32767	32K 個字組 x 7 個記憶庫 E0_00000 到 E6_32767	32K 個字組 x 3 個記憶庫 E0_00000 到 E2_32767	32K 個字組 x 1 個記憶庫 E0_00000 到 E0_32767		32K 個字組 x 3 個記憶庫 E0_00000 到 E2_32767	32K 個字組 x 1 個記憶庫 E0_00000 到 E0_32767		
功能區塊	最大定義數	1,024	1,024	1,024	1,024	128	1,024	1,024	128	128
	最大實例數	2,048	2,048	2,048	2,048	256	2,048	2,048	256	256
快閃記憶體	功能區塊程式記憶體 (Kbytes)	1,664	1,664	1,024	512	512	1,024	512	512	512
	註解檔案 (Kbytes)	128	128	64	64	64	64	64	64	64

項目		規格								
快閃記憶體	程式索引檔案 (Kbytes)	128	128	64	64	64	64	64	64	64
	變數表 (Kbytes)	128	128	128	64	64	128	64	64	64

CJ1-H CPU 模組

項目		規格						
型號		CJ1H-CPU67H	CJ1H-PU66H	CJ1H-CPU65H	CJ1G-CPU45H	CJ1G-CPU44H	CJ1G-CPU43H	CJ1G-CPU42H
I/O 點		2,560			1,280		960	
程式容量(步驟)		250K	120K	60K	60K	30K	20K	10K
資料記憶體		32K 個字組						
擴充資料記憶體		32K 個字組 x13 個記憶庫 E0_00000 到 E0_32767	32K 個字組 x7 個記憶庫 E0_00000 到 E6_32767	32K 個字組 x3 個記憶庫 E0_00000 到 E2_32767	32K 個字組 x3 個記憶庫 E0_00000 到 E2_32767	32K 個字組 x13 個記憶庫 E0_00000 到 E0_32767		
功能區塊	最大定義數	1,024	1,024	1,024	1,024	1,024	128	128
	最大實例數	2,048	2,048	2,048	2,048	2,048	256	256
快閃記憶體	功能區塊程式記憶體 (Kbytes)	1,664	1,664	1,024	1,024	512	512	512
	註解檔案 (Kbytes)	128	128	64	64	64	64	64
	程式索引檔案 (Kbytes)	128	128	64	64	64	64	64
	變數表 (Kbytes)	128	128	128	128	64	64	64

CJ1M CPU 模組

項目	規格					
	有內部 I/O 功能的模組			沒有內部 I/O 功能的模組		
型號	CJ1M-CPU23	CJ1M-CPU22	CJ1M-CPU21	CJ1M-CPU13	CJ1M-CPU12	CJ1M-CPU11
I/O 點	640	320	160	640	320	160
程式容量(步驟)	20K	10K	5K	20K	10K	5K
擴充底板(Rack)數	1 (最大)	不支援擴充		1 (最大)	不支援擴充	
資料記憶體	32K 個字組					
擴充資料記憶體	無					

項目		規格			
		有內部 I/O 功能的模組		沒有內部 I/O 功能的模組	
脈衝開始時間		46 μs (無加速/減速) 70 μs (有加速/減速)	63 μs (無加速/減速) 100 μs (有加速/減速)	---	
排定中斷數		2	1	2	1
PWM 輸出		2	1	無	
最大副程式號碼值		1,024	256	1,024	256
JMP 指令中最大跳越號碼值		1,024	256	1,024	256
內部輸入		10 點 • 4 個中斷輸入(脈衝擷取) • 2 個高速計數器輸入(50-kHz 差相/或 100-kHz 單相)		---	
內部輸出		6 點 • 2 個脈衝輸出(100 kHz) • 2 個 PWM 輸出	6 點 • 2 個脈衝輸出(100 kHz) • 2 個 PWM 輸出	---	
功能區塊	最大定義數	128			
	最大實例數	256			
快閃記憶體	功能區塊程式記憶體(Kbytes)	256			
	註解檔案(Kbytes)	64			
	程式索引檔案(Kbytes)	64			
	變數表(Kbytes)	64			

CP1H CPU 模組

項目	X 型號	XA 型號	Y 型號
型號	CP1H-X40DR-A CP1H-X40DT-D CP1H-X40DT1-D	CP1H-XA40DR-A CP1H-XA40DT-D CP1H-XA40DT1-D	CP1H-Y20DT-D
最大 I/O 點數	320 點(40 個內建點 + 40 個點/擴充底板(Rack) x 7 個底板(Rack))		300 點(20 個內建點 + 40 個點/擴充底板(Rack) x 7 個底板(Rack))
程式容量(步驟)	20K		
資料記憶體	32K 個字組		

項目		X 型號	XA 型號	Y 型號
功能區塊	最大定義數	128		
	最大實例數	256		
快閃記憶體	功能區塊程式記憶體(Kbytes)	256		
	註解檔案(Kbytes)	64		
	程式索引檔案(Kbytes)	64		
	變數表(Kbytes)	64		

NSJ-series NSJ Controllers

型號		NSJ5-TQ0□-G5D, NSJ5-SQ0□-G5D, NSJ8-TV0□-G5D, NSJ10-TV0□-G5D, NSJ12-TS0□-G5D,
最大 I/O 點數		1,280
程式容量(步驟)		60K
資料記憶體		32K 個字組
功能區塊	最大定義數	1024
	最大實例數	2048
快閃記憶體	功能區塊程式記憶體(Kbytes)	1024
	註解檔案(Kbytes)	64
	程式索引檔案(Kbytes)	64
	變數表(Kbytes)	128

FQM1 彈性位置控制器

項目		CPU 模組	Motion 控制模組	
型號		FQM1-CM002	FQM1-MMA22	FQM1-MMP22
最大 I/O 點數		344 點(24 個內建點 + 基本 I/O 模組上的 320 個點)	20 個內建點	
程式容量(步驟)		10K		
資料記憶體		32K 個字組		
功能區塊	最大定義數	128		
	最大實例數	256		
快閃記憶體	功能區塊程式記憶體(Kbytes)	256		
	註解檔案(Kbytes)	64		
	程式索引檔案(Kbytes)	64		
	變數表(Kbytes)	64		

2-7-2 計時器指令的作用

CP1H CPU 模組和模組版本為 3.0 版以上的 CS1H、CJ1-H、及 CJ1M CPU 模組的 PLC 屬性中有一個選項稱為*套用與 TO-2047 到 T2048-4095 相同的規格*。這個設定會如本節所述影響計時器的作用。

選取這個選項(預設)

如果選取這個選項，則所有計時器都會進行相同的作用而不論它的計時器號碼，如下表所示。

計時器號碼 T0000 至 T4095 的計時器作用

更新	說明
當指令執行時	PV 會在每次指令執行時更新。 如果 PV 為 0，則完成旗標會轉為 ON。如果它不是 0，則完成旗標會轉為 OFF。
當所有工作的執行都完成時	所有 PV 會在每個循環更新一次。
每隔 80 ms	如果周期時間超過 80 ms，則所有 PV 會每隔 80 ms 更新一次。

沒有選取這個選項

如果沒有選取這個選項，則計時器號碼 T0000 到 T2047 的計時器指令更新會與計時器號碼 T2048 到 T4095 不同，如以下所示。這樣的行為與不支援功能區塊的 CPU 模組相同。(詳細資訊請參考 *CS/CJ 系列指令參考*中的個別指令的說明。)

計時器號碼 T0000 至 T2047 的計時器作用

更新	說明
當指令執行時	PV 會在每次指令執行時更新。 如果 PV 為 0，則完成旗標會轉為 ON。如果它不是 0，則完成旗標會轉為 OFF。
當所有工作的執行都完成時	所有 PV 會在每個循環更新一次。
每隔 80 ms	如果循環時間超過 80 ms，則所有 PV 會每隔 80 ms 更新一次。

計時器號碼 T2048 至 T4095 的計時器作用

更新	說明
當指令執行時	PV 會在每次指令執行時更新。 如果 PV 為 0，則完成旗標會轉為 ON。如果它不是 0，則完成旗標會轉為 OFF。
當所有工作的執行都完成時	PV 沒有更新。
每隔 80 ms	即使循環時間超過 80 ms，PV 也沒有更新。

在使用由功能區塊變數所預設分配的計時器號碼(T3072 到 T4095)時，請選取"*套用與 TO-2047 到 T2048-4095 相同的規格*"選項來確保一致的作用。

2-8 功能區塊程式步驟數與實例執行時間

2-8-1 功能區塊程式步驟數

本節只適用於模組版本 1.0 版以上的 CP 系列 CPU 模組及模組版本 3.0 版以上的 CS/CJ 系列 CPU 模組、NSJ 控制器以及 FQM1 彈性 Motion 控制器。

當功能區塊定義已經建立且實例複製到 CPU 模組的使用者程式中以後，請使用下列等式來計算程式步驟數。

步驟數
= 實例數 x (呼叫部份容量 m + I/O 參數傳送部份容量 n x 參數數) + 功能區塊定義中的指令步驟數 p (請參閱備註。)

備註 當相同的功能區塊定義複製到多個位置(即，多個實例)時，子順序實例中的功能區塊定義中的指令步驟數(p)並不會減少。因此，在上述等式中，實例數並沒有乘以功能區塊定義中的指令步驟數(p)。

下表只適用於模組版本 Ver. 1.0 以上的 CP 系列 CPU 模組及模組版本 Ver. 3.0 以上的 CS/CJ 系列 CPU 模組、NSJ 控制器、以及 FQM1 彈性 Motion 控制器。

目錄			步驟數
m	呼叫部份		57 步驟
n	I/O 參數傳送部份 資料類型顯示在括弧中	1 位元 I/O 變數(BOOL)	6 步驟
		1 字組 I/O 變數(INT、UINT、WORD)	6 步驟
		2 字組 I/O 變數(DINT, UDINT, DWORD, REAL)	6 步驟
		4 字組 I/O 變數(LINT, ULINT, LWORD, LREAL)	12 步驟
p	功能區塊定義中的指令步驟數	總計指令步驟數(與標準使用者程式相同) + 27 個步驟。	

範例程式：
有 1 個字組資料類型(INT)的輸入變數：5
有 1 個字組資料類型(INT)的輸出變數：5
功能區塊定義區段：100 步驟
1 個實例的步驟數 = 57 + (5 + 5) x 6 個步驟 + 100 個步驟 + 27 個步驟
= 244 個步驟

2-8-2 功能區塊實例執行時間

本節只適用於模組版本 1.0 版以上的 CP 系列 CPU 模組及模組版本 3.0 版以上的 CS/CJ 系列 CPU 模組、NSJ 控制器、以及 FQM1 彈性 Motion 控制器。

當功能區塊定義已經建立且實例複製到 CPU 模組的使用者程式中以後，請使用下列等式來計算針對循環時間的實例執行效果。

針對循環時間的實例執行效果
= 啟動時間(A)
+ I/O 參數傳輸處理時間(B)
+功能區塊定義中的指令執行時間(C)

下表顯示 A、B、和 C 的時間長度。

操作			CPU 模組型號		
			CS1H-CPU6□H CJ1H-CPU6□H	**CS1G-CPU4□H CJ1G-CPU4□H NSJ**	**CJ1M-CPU□□ CP1H-X□□□-□ CP1H-A□□□-□**
A	啟動時間	不包括 I/O 參數傳輸的啟動時間	6.8 μs	8.8 μs	15.0 μs
B	I/O 參數傳輸處理時間 資料類型表示在括弧中	1 位元 I/O 變數(BOOL)	0.4 μs	0.7 μs	1.0 μs
		1 字組 I/O 變數(INT、UINT、WORD)	0.3 μs	0.6 μs	0.8 μs
		2 字組 I/O 變數(DINT, UDINT, DWORD, REAL)	0.5 μs	0.8 μs	1.1 μs
		4 字組 I/O 變數(LINT, ULINT, LWORD, LREAL)	1.0 μs	1.6 μs	2.2 μs
C	功能區塊定義指令執行時間	總計指令處理時間(與標準使用者程式相同)			

範例：CS1H-CPU63H
有 1 個字組資料類型(INT)的輸入變數：3
有 1 個字組資料類型(INT)的輸出變數：2
功能區塊定義區段中的總計指令處理時間：10 μs
1 個實例的執行時間 = 6.8 μs + (3 + 2) x 0.3 μs + 10 μs = 18.3 μs

備註 當相同的功能區塊定義複製到多個位置時，執行時間會隨著多重實例數而增加。

章節 3
建立功能區塊

本章說明在 CX-Programmer 上建立功能區塊的程序。

3-1 程序流程

下列程序可用來建立功能區塊、將它們儲存在檔案中、將它們傳送到 CPU 模組上、監控它們、以及進行除錯。

建立功能區塊

建立一個專案

詳細資訊請參考 *3-2-1 建立一個專案*。

■ **建立一個新的專案**

1,2,3…

1. 啟動 CX-Programmer 並從檔案主選單選取 ***New***。
2. 選取一個*裝置類型*：CS1G-H、CS1H-H、CJ1G-H、CJ1H-H、CJ1M 或 CP1H、NSJ 或 FQM1-CM (MMA/MMP)。

■ **重複使用一個現有的 CX-Programmer 專案**

1,2,3…

1. 啟動 CX-Programmer，並從檔案主選單選取檔案來讀取利用 CX-Programmer 版本 4.0 以下所建立的現有專案檔案(.cxp)。
2. 選取一個*裝置類型*：CS1H-H、CS1G-H、CJ1G-H、CJ1H-H、CJ1M 或 CP1H、NSJ 或 FQM1-CM (MMA/MMP)。

建立一個功能區塊定義

詳細資訊請參考 *3-2-2 建立一個新的功能區塊定義*。

1,2,3…

1. 在專案工作區中選取 *Function Blocks (功能區塊)*並按一下滑鼠右鍵。
2. 從快顯主選單選取 ***Insert Function Block (插入功能區塊) - Ladder (階梯)*** 或 ***Insert Function Blocks (插入功能區塊) - Structured Text (結構化文字)***。

定義功能區塊

詳細資訊請參考 *3-2-3 定義使用者所建立的功能區塊*。

■ **在輸入階梯圖程式或 ST 程式之前登錄變數**

1,2,3…

1. 將變數登錄到變數表中。
2. 建立階梯圖程式或 ST 程式。

■ **在輸入階梯圖程式或 ST 程式時視需要登錄變數**

1,2,3…

1. 建立階梯圖程式或 ST 程式。
2. 需要時在變數表中登錄一個變數。

從功能區塊定義建立一個實例

詳細資訊請參考 *3-2-4 從功能區塊定義建立實例*。

■ **插入實例到階梯區段視窗中然後輸入實例名稱**

1,2,3…

1. 將游標放在要建立一個功能區塊實例(也就是一個拷貝)的位置上並按 **F 鍵**。
2. 輸入實例的名稱。
3. 選取要複製的功能區塊定義。

■ **將實例名稱登錄到全域符號表中然後在插入時選取實例名稱**

1,2,3…

1. 選取 *Function Block (功能區塊)*做為全域符號表中的變數的資料類型。
2. 按階梯區段視窗中的 **F 鍵**。

3. 從 *FB Instance (FB 實例)*欄位上的下拉式主選單選取所登錄的實例名稱。

分配外部 I/O 給功能區塊

詳細資訊請參考 *3-2-5 利用 Enter 鍵設定功能區塊參數*。

1,2,3…

1. 將游標放在輸入變數或輸出變數的位置並按 **P 鍵**。
2. 輸入輸入變數的來源位址或輸出變數的目的位址。

設定功能區塊記憶體配置 (實例區)

詳細資訊請參考 *3-2-6 設定 FB 實例區*。

1,2,3…

1. 選取實例並從 PLC 主選單選取 ***Function Block Memory (功能區塊記憶體) - Function Block Memory Allocation (功能區塊記憶體配置)***。
2. 設定功能區塊記憶體配置。

列印、儲存、及重複使用功能區塊檔案

匯集功能區塊定義並將它儲存為一個資料庫檔案

詳細資訊請參考 *3-2-12 匯集功能區塊定義(檢查程式)*及 *3-2-15 儲存及重複使用功能區塊定義檔案*。

1,2,3…

1. 匯集已經儲存的功能區塊。
2. 列印功能區塊。
3. 將功能區塊儲存為一個功能區塊定義檔案(.cxf)。
4. 將檔案讀入另一個 PLC 專案中。

將程式傳送到 PLC

請參考 *3-2-16 下載/上傳程式到實際 CPU 模組*。

監控及進行功能區塊除錯

請參考 *3-2-17 監控及進行功能區塊除錯*。

3-2 程序

3-2-1 建立一個專案

以 CX-Programmer 建立新的專案

1,2,3… 1. 啟動 CX-Programmer 並從檔案主選單選取 ***New***。

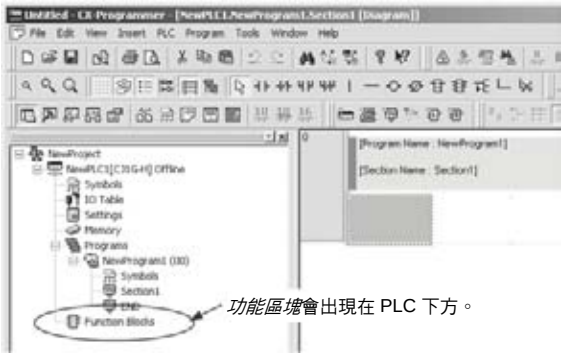
2. 在變更 PLC 視窗中，選取一個支援功能區塊的 *Device Type (裝置類型)*。這些類型如下表所列。

裝置	CPU
CS1G-H	CPU42H/43H/44H/45H
CS1H-H	CPU63H/64H/65H/66H/67H
CJ1G-H	CPU42H/43H/44H/45H
CJ1H-H	CPU65H/66H/67H
CJ1M	CPU11/12/13/21/22/23
CP1H	XA/X
NSJ	G5D (用於 NSJ5-TQ0□-G5D, NSJ5-SQ0□-G5D, NSJ8- TV0□-G5D, NSJ10-TV0□-G5D 及 NSJ12-TS0□-G5D)
FQM1-CM	FQM1-CM002
FQM1-MMA	FQM1-MMA22
FQM1-MMP	FQM1-MMP22

按 Settings (設定)按鈕並選取 *CPU 型式*。關於其他設定的詳細資訊，請參考 *CX-Programmer 5.0 版操作手冊(W414)*。

3-2-2 建立一個新的功能區塊定義

1,2,3… 1. 在建立一個專案時，專案工作區中會出現一個 *Function Blocks (功能區塊)* 圖示按鈕，如以下所示。



2. 將功能區塊定義插入到功能區塊圖示之後來建立功能區塊定義。

建立功能區塊定義

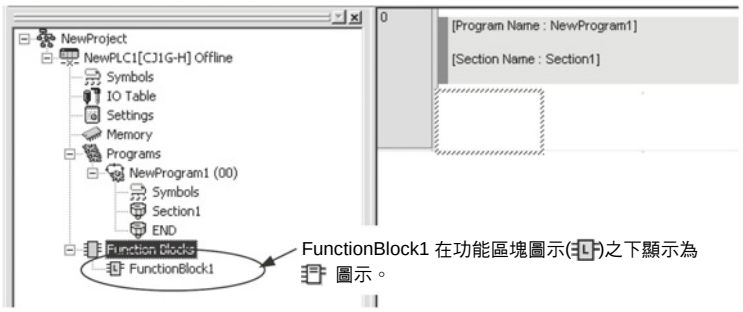
功能區塊可以由使用者利用階梯圖程式編輯或結構化文字來定義。

建立(插入)有階梯圖的功能區塊定義

1. 在專案工作區中選取 ***Function Blocks (功能區塊)***，按一下滑鼠右鍵，並從快顯主選單選取 ***Insert Function Blocks (插入功能區塊) - Ladder (階梯圖)***。(或者從插入主選單選取***功能區塊 - 階梯圖***。)

建立(插入)有結構化文字的功能區塊定義

1. 在專案工作區中選取 ***Function Blocks (功能區塊)***，按一下滑鼠右鍵，並從快顯主選單選取 ***Insert Function Blocks (插入功能區塊) - Structured Text (結構化文字)***。(或者從插入主選單選取***功能區塊 - 結構化文字***。)

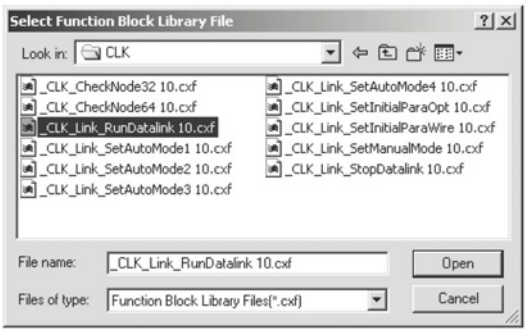


2. 一個稱為FunctionBlock1的功能區塊會自動插入到階梯圖程式編輯語言(預設)的 之後或者ST語言的 之後。這個圖示包含有新建立的(插入的)功能區塊的定義。
3. 在建立一個功能區塊定義後，會自動指定 FunctionBlock□的名稱，其中□是一個流水號碼。這些名稱可以變更。所有的名稱都必須不超過 64 個字元。

使用 OMRON FB 資料庫檔案

請利用下列程序來插入 OMRON FB 資料庫檔案(.cxf)。

1. 在專案工作區中選取 ***Function Blocks (功能區塊)***，按一下滑鼠右鍵，並從快顯主選單選取 ***Insert Function Blocks (插入功能區塊) - Library File (資料庫檔案)***。(或者從插入主選單選取***功能區塊 - 資料庫檔案***。)
2. 接著會顯示下列"選取功能區塊資料庫檔案"對話框。



備註 要在"功能區塊資料庫檔案"對話框中指定預設資料夾(檔案位置)時，請選取 ***Tools (工具) - Options (選項)***，按一下 **General (一般)** 索引標籤並選取 *OMRON FB library storage location (OMRON FB 資料庫儲存位置)*欄中的預設檔案。

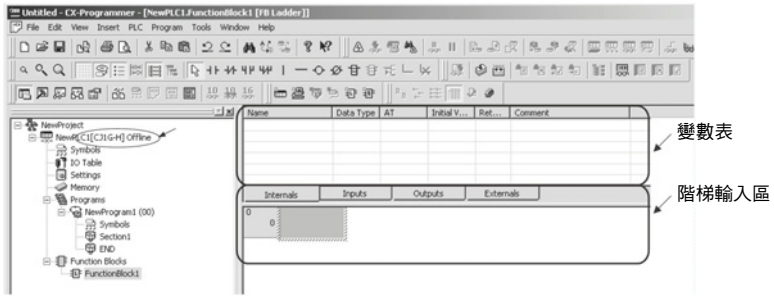
3. 指定 OMRON FB 資料庫檔案所在的資料夾，選取資料庫檔案，並按一下 **Open (開啟)**按鈕。資料庫檔案就會插入 之後做為一個功能區塊定義。

功能區塊定義

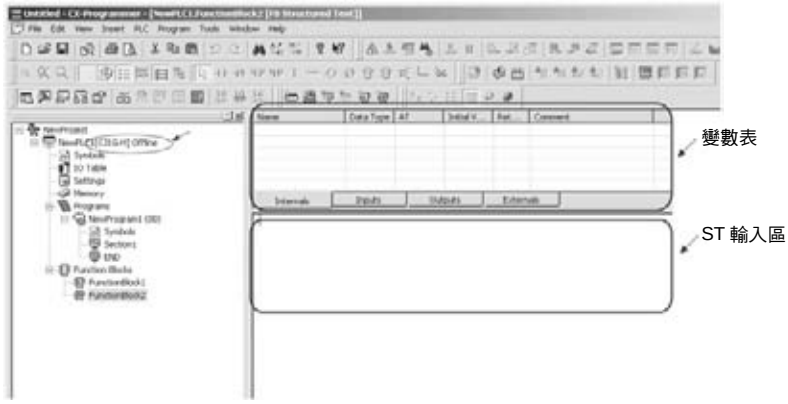
建立功能區塊定義

按兩下新建立的功能區塊 1 圖示(或者按一下滑鼠右鍵並從快顯主選單中選取 ***Open***)時，會顯示下列視窗中的一個視窗。一個功能區塊中所使用的變數的變數表會顯示在上方而一個供階梯圖程式或結構化文字使用的輸入區會顯示在下方。

階梯圖程式



結構化文字



如圖所示，包括有一個變數表的功能區塊定義會用來做為一個介面，而階梯圖程式或結構化文字則會用來做為一個演算法則。

變數表做為介面

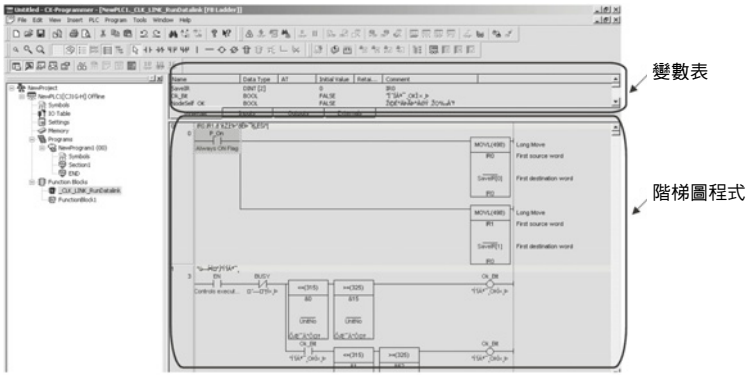
此時，變數表會是空的，因為它還沒有被分配 PLC 中的 I/O 記憶體位址的變數。

階梯圖程式或結構文字做為演算法則

- 除了某些例外之外，功能區塊的階梯圖程式可能包含有任何用於一般程式中的指令。關於可能使用的指令的限制，請參考 *2-3 功能區塊限制*。
- 結構化文字可以根據 IEC61131-3 中所定義的 ST 語言輸入。

使用 OMRON FB 資料庫檔案

按兩下插入的功能區塊資料庫(或按一下滑鼠右鍵並從快顯主選單選取 ***Open***)來將已經完成建立的變數表顯示在右上方視窗，以及將已經完成建立的階梯圖程式顯示在右下方視窗中。這兩個視窗都會顯示為灰色而不能編輯。



備註 功能區塊定義並不會顯示在 OMRON FB 資料庫檔案(.cxf)的預設設定中。要顯示定義時，請選取功能區塊屬性中的 ***Display the inside of FB (顯示 FB 的內容)***選項。(在專案工作區中選取 OMRON FB 資料庫檔案，按一下滑鼠右鍵，選取 ***Properties (屬性)***，並選取一般索引標籤中的 ***Display the inside of FB (顯示 FB 的內容)***選項。)

3-2-3 定義使用者所建立的功能區塊

一個功能區塊可透過登錄變數及建立一個演算法則來加以定義。有兩種方法可以進行這個作業。

- 先登錄變數然後再輸入階梯圖程式或結構文字。
- 在輸入階梯圖程式或結構文字時再登錄所需的變數。

先登錄變數

在變數表中登錄變數

變數表中的變數會根據類別分為 4 個表單：內部、輸入、輸出、及外部。

在登錄或顯示變數時，必須開啟這些表單。

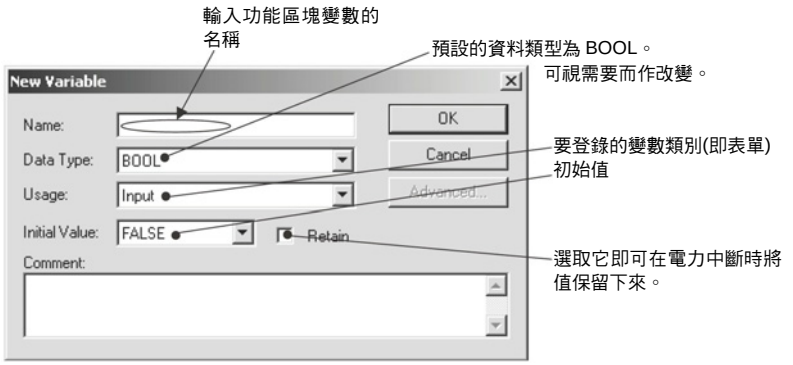
1,2,3...

1. 在變數表中選取要登錄的變數類別的表單做為現行表單。(請參閱備註。)將游標放在表單中，按一下滑鼠右鍵，並執行下列任一項操作：
   - 要新增一個變數到最後一行時，請從快顯主選單選取 ***Insert Variable (插入變數)***。
   - 要將變數新增到清單中的上一行或下一行時，請從快顯主選單選取 ***Insert Variable (插入變數) - Above (上)***或 ***Below (下)***。

   備註 在插入一個變數時，透過設定用法也可以開啟變數所要登錄的表單(N：內部，I：輸入，O：輸出，E：外部)。

接著會顯示如以下所示的"新變數"對話框。

- **名稱：**輸入變數的名稱。
- **資料型態：**選取資料類型。
- **使用狀況：**選擇變數類型。
- **起始值：**選取在作業開始時的變數起始值。
- **保留：**選取在電源開啟時或者在操作模式從 PROGRAM 或 MONITOR 模式變更為 RUN 模式時變數的值是否要保留。如果沒有選取 *Retain (保留)*，則變數的值在這些時候會被清除。

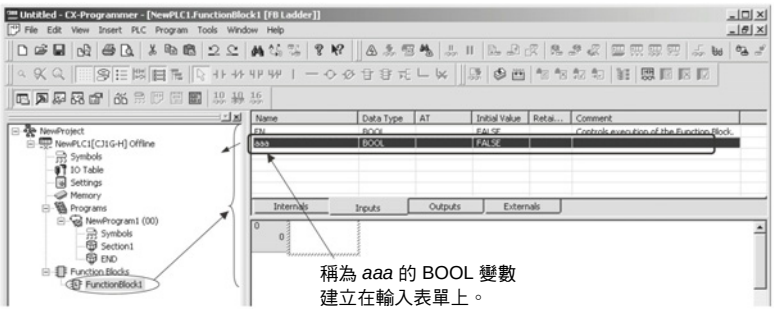


備註 (a) 在使用者定義的外部變數方面，可以透過在全域符號表中登錄相同的變數名稱來瀏覽全域符號表。

(b) 由系統定義的外部變數會事先登錄在外部變數表中。

2. 例如，輸入“aaa"做為變數名稱並按一下 **OK** 按鈕。

   如下圖所示，一個稱為 *aaa* 的 BOOL 變數會建立在變數表的 Inputs (輸入) 表單上。



備註 (1) 在新增一個變數後，可以選取它來讓它反白顯示，然後利用拖放操作來將它移到另一行。要選取一個變數進行拖放時，請選取除 *名稱欄*以外的任何一欄中的變數。

(2) 在輸入一個變數後，要移到所登錄的表單時，請按兩下，並開啟 *Usage (用法)*欄中的設定(N：內部，I：輸入，O：輸出，E：外部)。變數也可以在內部、外部、輸入、及輸出變數的表單之間進行複製或移動。請選取變數，按一下滑鼠右鍵，並從快顯主選單中選取 ***Copy (複製)***或 ***Cut (剪下)***，然後選取 ***Paste (貼上)***。

(3) 以 AT (分配實際位址)設定指定的變數也必須輸入變數名稱。

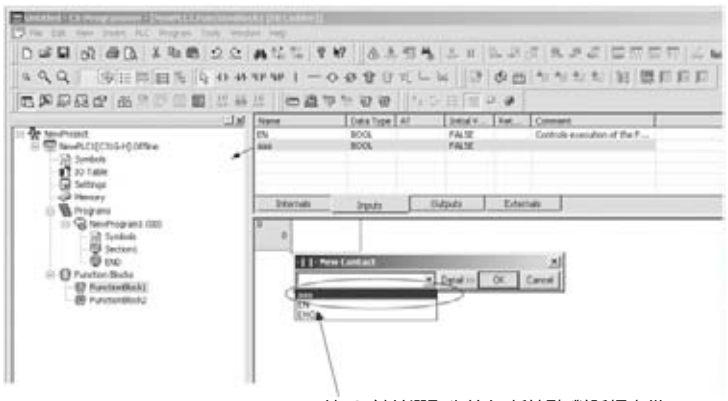
(4) 下列文字是用來顯示 PLC 中的 I/O 記憶體位址的，因此不能輸入到功能區塊變數表中做為變數名稱。

- A、W、H、HR、D、DM、E、EM、T、TIM、C、或 CNT 後隨一個數值

建立演算法則

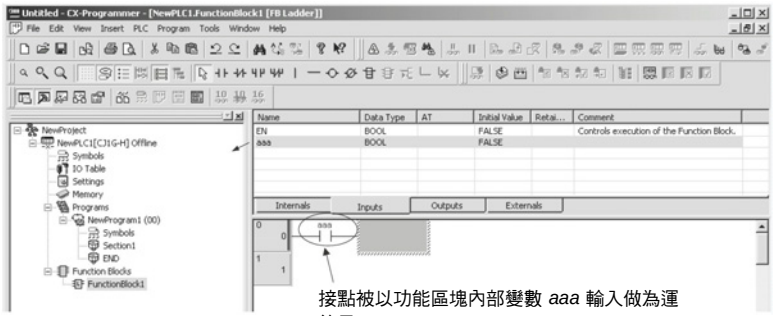
利用一個階梯圖程式

1,2,3… 1. 按 **C 鍵**並選取先前在“新接點”對話框中從下拉式主選單所登錄的 *aaa*。



按 C 鍵並選取先前在“新接點”對話框中從下拉式主選單所登錄的 *aaa*。

2. 按一下 **OK** 按鈕，一個接點就會以功能區塊內部變數 *aaa* 輸入做為運算元 (變數類別：內部)。



接點被以功能區塊內部變數 *aaa* 輸入做為運算元。

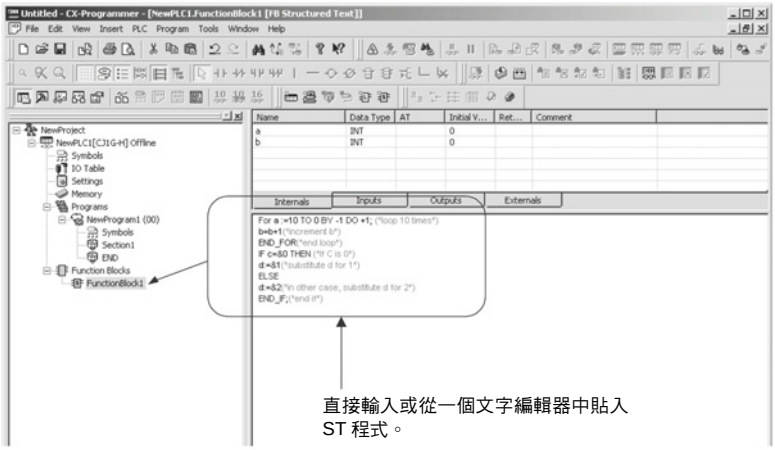
其他的階梯圖程式可透過 CX-Programmer 以和標準程式完全相同的方法輸入。

備註 位址不能直接輸入功能區塊中的指令運算元中。只有指標暫存器(IR)及資料暫存器(DR)可以輸入直接如下(不做為變數)：位址 DR0 到 DR5，直接指定 IR0 到 IR15，並間接指定,IR0 到,IR15。

利用結構化文字

可以直接在 ST 輸入區中輸入一個 ST 語言程式(請參閱備註)，或可以複製一個輸入到一般用途文字編輯器中的程式然後利用 Edit (編輯)主選單的 *Paste (貼上)*指令將它貼入 ST 輸入區中。

備註 ST 語言符合 IEC61131-3 規定。詳細資訊請參考*附錄 B 結構化文字(ST 語言)規格*。



備註 (1) 可以輸入定位點(Tab 鍵)或空格(空間棒)來產生縮排。這些並不會影響演算法則。

(2) 顯示的大小可以按住 **Ctrl 鍵**並轉動滑鼠上的捲動輪來變更。

(3) 當一個 ST 語言程式輸入或貼入 ST 輸入區時，語法關鍵字組的保留字組會自動以藍色顯示，註解以綠色、錯誤以紅色、其他的則以黑色顯示。

(4) 要變更字形大小或顏色時，請從工具主選單選取 ***Options (選項)***然後在 Appearance (外觀)索引標籤頁上按一下 **ST Font (ST 字形)**按鈕。然後就可以變更字形名稱、字形大小(預設為 8 point)和顏色。

(5) 關於結構化文字規格的詳細資訊，請參考*附錄 B 結構化文字(ST 語言)規格*。

視需要登錄變數

階梯圖程式或結構化文字程式可以先行輸入並在需要時才登錄變數。

使用一個階梯圖程式

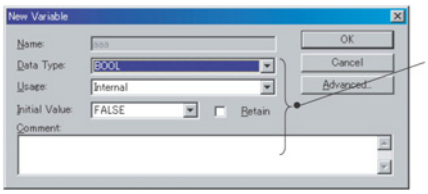
在使用一個階梯圖時，在輸入一個沒有登錄過的變數名稱時，會顯示一個對話框來登錄變數。此時即可登錄變數。

使用下列程序。

1,2,3... 1. 按 **C 鍵**並在“新接點”對話框中輸入一個沒有登錄過的變數名稱，例如 *aaa*。

備註 位址不能直接輸入功能區塊中的指令運算元中。只有指標暫存器(IR)及資料暫存器(DR)可以直接輸入如下(不做為變數)：位址 DR0 到 DR5，直接指定 IR0 到 IR15，和間接指定,IR0 到,IR15。

2. 按一下 **OK** 按鈕，接著會顯示如所示的“新變數”對話框。利用特殊指令，會針對指令中的每個運算元顯示一個“新變數”對話框。



所有輸入變數的屬性都會在一開始時就顯示，如下：

- 使用狀況：內部
- 資料型態：接點為 BOOL，路徑為 WORD (字組)
- 起始值：該資料類型的預設值。
- 保留：無選取。

3. 作完任何必要的變更後並按一下 **OK** 按鈕。
4. 如下圖所示，所登錄的變數會顯示在程式上方的變數表中。

5. 如果所輸入的一個變數的類別或屬性不正確，請按兩下變數表中的變數並進行必要的修正。

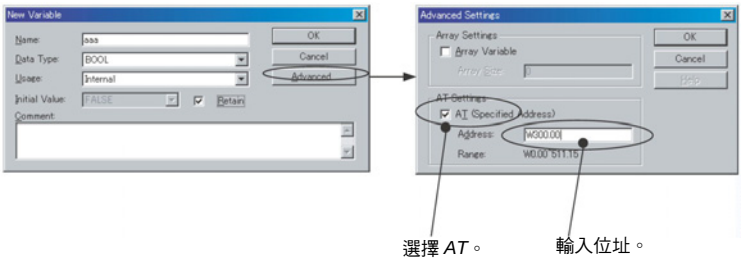
■ 參考資訊

AT 設定(指定的位址)

可以在變數屬性中進行 AT 設定來指定基本 I/O 模組、特殊 I/O 模組、或 CPU 匯流排模組的配置位址，或沒有利用 CX-Programmer 登錄的附屬區域位址。必須要有一個變數名稱才能這樣做。請利用下列程序來指定位址。

1,2,3...

1. 在新變數對話框中輸入變數名稱後，請按一下 **Advanced (進階)**按鈕。接著會顯示進階設定對話框。
2. 選取 *AT Settings* 下的 *AT (指定的位址)*並輸入想要的位址。



即使變數已經有一個用於 AT 設定的指定位址，變數名稱仍可以用來將變數輸入到功能區塊定義中的演算法則中(和沒有指定位址的變數相同)。
例如，如果一個名為 *Restart* 的變數有一個用於 AT 設定的指定位址 A50100，則 *Restart* 會被指定用於指令運算元。

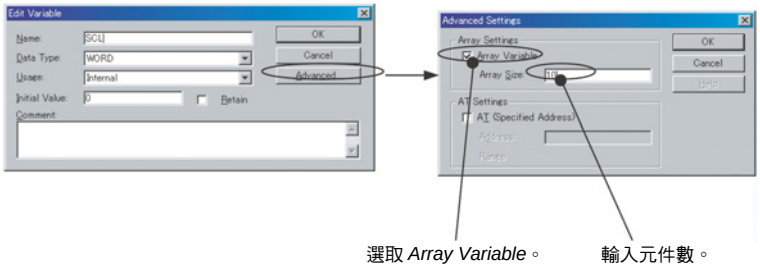
陣列設定

可以指定一個陣列來將相同的資料屬性用於一個以上的變數並以一個群組來管理變數。

請利用下列程序來設定一個陣列。

1,2,3...

1. 在新變數對話框中輸入變數名稱後，請按一下 **Advanced (進階)**按鈕。接著會顯示進階設定對話框。
2. 在 *Array Settings (陣列設定)*中選取 *Array Variable (陣列變數)*並輸入陣列中的最大元件數。



在將一個陣列變數的名稱輸入到功能區塊定義中的演算法則中時，陣列名稱之後會出現內有索引的方括弧。

例如，如果您建立一個名為 PV，最多有 3 個元件的變數，則可以指定 PV[0]、PV[1]、及 PV[2]做為指令運算元。

有 3 種方法可以用來指定索引。

- 直接使用數字，例如上述範例中的 PV[1](階梯圖程式編輯或 ST 語言程式編輯)
- 使用一個變數，例如上述範例中的 PV[a]，其中“a”是一個資料類型為 INT 的變數的名稱(階梯圖程式編輯或 ST 語言程式編輯)
- 使用一個等式，例如上述範例中的 PV[a+b]或 PV[a+1]，其中“a”和“b”是一個資料類型為 INT 的變數的名稱(僅限於 ST 語言程式編輯)

利用結構化文字

在使用結構化文字時，在輸入一個沒有登錄過的變數名稱時，並不會顯示一個對話框來登錄變數。請確定務必在您需要時或者在程式匯集之後登入用於變數表中標準文字程式編輯中的變數。(將游標放在登錄變數的索引標籤頁中，按一下滑鼠右鍵，並從快顯主選單選取 ***Insert Variable (插入變數)***。)

備註 關於結構化文字規格的詳細資訊，請參考*附錄 B 結構化文字(ST 語言)規格*。

複製使用者程式迴路並貼入功能區塊定義的階梯圖程式編輯中

可以複製使用者程式中的一個迴路或多個迴路並將它貼入到功能區塊定義的階梯圖程式編輯中。不過，這項操作會有下列限制。

來源指令運算元：只能使用位址

位址並沒有登錄到功能區塊定義的變數表中。在貼上之後，位址會在運算元中顯示為紅色。請按兩下指令並將變數名稱輸入到運算元中。

備註 不過，指標暫存器(IR)和資料暫存器(DR)在貼上後並不須要修改，運算元中的功能也照舊。

來源指令運算元：位址及 I/O 註解

已選取“工具”主選單的“選項”下的 Symbols (符號)索引標籤中的 *Automatically generate symbol name (自動產生符號名稱)*選項

使用者程式符號名稱(只在全域符號表中)會自動產生為 AutoGen_+*位址*(如果取消選取這個選項，則符號名稱會被移除)。

範例 1：如果位址是 100.01，則符號名稱會顯示為 AutoGen_100_01。

範例 2：如果位址是 D0，則符號名稱會顯示為 AutoGen_D0。

如果將使用者程式中的迴路複製並貼到功能區塊定義程式中，符號會自動登錄到功能區塊定義符號表中(與複製迴路同時)做為符號名稱“AutoGen_*位址*”而“I/O 註解”會登錄做為*註解*。這個功能可以讓設定的迴路方便的重複使用在功能區塊中做為位址和 I/O 註解。

備註 字首 AutoGen_並不會被加到指標暫存器(IR)和全域資料暫存器(DR)上，而且它們不能登錄到原來的全域符號表中。

不選取“工具”主選單的“選項”下的 Symbols (符號)索引標籤中的 Automatically generate symbol name (自動產生符號名稱)選項

位址和 I/O 註解不會被登錄到功能區塊定義變數表中。位址會在運算元中顯示為紅色。I/O 註解會流失。請按兩下指令並將符號名稱輸入到運算元中。

不過，指標暫存器(IR)和資料暫存器(DR)在貼上後並不須要修改，運算元中的功能也照舊。

來源指令運算元：符號

使用者程式符號會自動登錄在功能區塊定義變數表的內部變數中。不過，這項操作會有下列限制。

Addresses (位址)

符號位址不會登錄。請使用 AT 設定來指定相同的位址。

符號資料類型

符號資料類型會在從使用者程式貼入功能區塊定義時轉換，如下表所示。

使用者程式中的符號資料類型	→	貼入功能區塊程式後的變數資料類型
CHANNEL	→	WORD
NUMBER	→	變數將不會登錄，而它的值(數字)會被直接貼到運算元中做為一個常數。
UINT BCD	→	WORD
UDINT BCD	→	DWORD
ULINT BCD	→	LWORD

不過，符號資料類型 CHANNEL、NUMBER、UINT BCD、UDINT BCD 或 ULINT BCD 不能從符號表(而不是程式)複製然後再貼到功能區塊定義的變數表中。

備註 自動產生符號名稱(AutoGen_+*位址*)的符號不能從全域符號表複製然後再貼到功能區塊定義的符號表中。

3-2-4 從功能區塊定義建立實例

如果全域符號表中登錄有一個功能區塊定義，則可以利用下列任一個方法來建立實例。

方法 1：選取功能區塊定義，將它插入程式中，並輸入一個新的實例名稱。實例就會自動登錄到全域符號表中。

方法 2：在全域符號表中設定“FUNCTION BLOCK (功能區塊)”的資料類型，指定要使用的功能區塊定義，並輸入實例的名稱來登錄。

備註 若使用 ST 語言時，可以利用想要的實例名稱透過選取“FUNCTION BLOCK (功能區塊)”做為變數的資料類型來叫出一個功能區塊，並輸入一個功能區塊呼叫 ST 的內文

■ **方法 1：在階梯區段視窗中利用 F 鍵並輸入實例名稱**

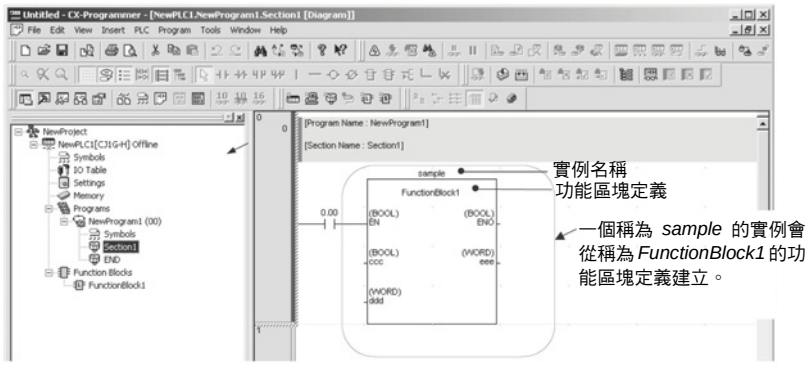
1,2,3…

1. 在階梯區段視窗中，將游標放在實例要插入的程式中並按 **F 鍵**。(或者，從“插入”主選單選取***引用功能區塊***。)接著會顯示“引用新功能區塊”對話框。

   若使用 ST 語言時，可以利用想要的實例名稱透過選取“FUNCTION BLOCK (功能區塊)”做為變數的資料類型來叫出一個功能區塊，並輸入下一個功能區塊呼叫敘述。在實例名稱後的括弧中指定引數(來將輸入變數的值從呼叫功能區塊轉送給叫出的功能區塊中的輸入變數)同時指定回歸值(來接收來自叫出的功能區塊給中呼叫功能區塊的輸出變數的輸出變數值)。實例名稱可以透過“FUNCTION BLOCK”資料類型設定到任何內部變數上。

2. 輸入實例名稱，選取要從它建立一個實例的功能區塊，並按一下 **OK** 按鈕。

3. 例如，在 *FB Instance (FB 實例)*欄中將實例名稱設定為 ***sample***，在 *FB Definition (FB 定義)*欄中將功能區塊設定為 ***FunctionBlock1***，並按一下 **OK** 按鈕。如下圖所示，它會以 *sample* 的實例名稱建立一個稱為 *FunctionBlock1* 的功能區塊定義拷貝。



實例會以 *sample* 的實例名稱和 *FUNCTION BLOCK (功能區塊)*的資料類型自動登錄到全域符號表中。

■ **方法 2：事先在全域符號表中登錄實例名稱然後再選取實例名稱**

如果實例名稱事先登錄在全域符號表中，則可以從全域符號表選取實例名稱來建立其他實例。

1,2,3... 1. 在階梯圖方面，請在全域符號表中選取 *Function block (功能區塊)*的資料類型，輸入實例名稱，並登錄這個實例。

在 ST 方面，請選取 *Function block (功能區塊)*的資料類型，使用這個實例名稱，並使用一個功能區塊的呼叫敘述來叫出功能區塊，如下：
輸入實例名稱(任何有功能區塊資料類型的內部變數名稱)後隨括弧中的引數(即，指定要轉送給叫出的功能區塊輸入變數的呼叫功能區塊輸入變數值)。同時包括回歸值(即，指定要回送給呼叫功能區塊輸出變數中所叫出的功能區塊輸出變數值)。

2. 按階梯區段視窗中的 **F 鍵**。接著會顯示"引用功能區塊"對話框。
3. 選取先前從 *FB 實例*欄的下拉式主選單登錄的實例名稱。接著就會建立實例。

限制

在建立實例時請遵守下列限制。關於相關細節，請參考 *2-3 功能區塊限制*。

- 每個程式迴路中不能建立超過一個以上的功能區塊。
- 梯級不能在實例的左側分支。
- 實例不能直接連接到左側匯流排，即，必須隨時插入一個 EN。

備註 如果對一個功能區塊定義的變數表中的 I/O 變數進行變更，則所有從該功能區塊定義所建立的實例左側的匯流排都會顯示為紅色來表示錯誤。當這種情況發生時，請選取功能區塊，按一下滑鼠右鍵，並選取 ***Update Invocation (更新引用)***。實例就會針對功能區塊定義中所進行的任何變更來進行更新並清除表示錯誤的紅色匯流排顯示。

3-2-5 利用 Enter 鍵設定功能區塊參數

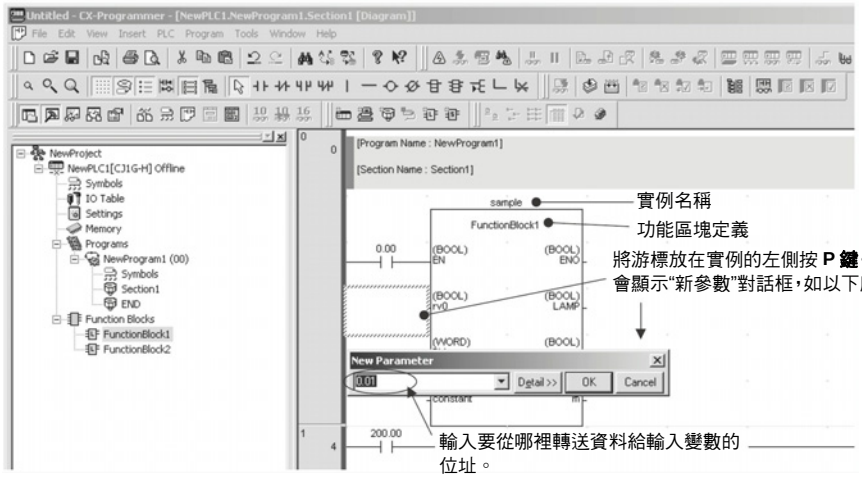
在一個功能區塊的實例建立後，必須設定輸入變數的輸入參數且必須設定輸出變數的輸出參數來啟用外部 I/O。

- 輸入參數中可以設定包括值、位址、及程式符號(全域符號及區域符號)。(請參閱備註 a。)
- 輸出參數中可以設定位址及程式符號(全域符號及區域符號)。(請參閱備註 b。)

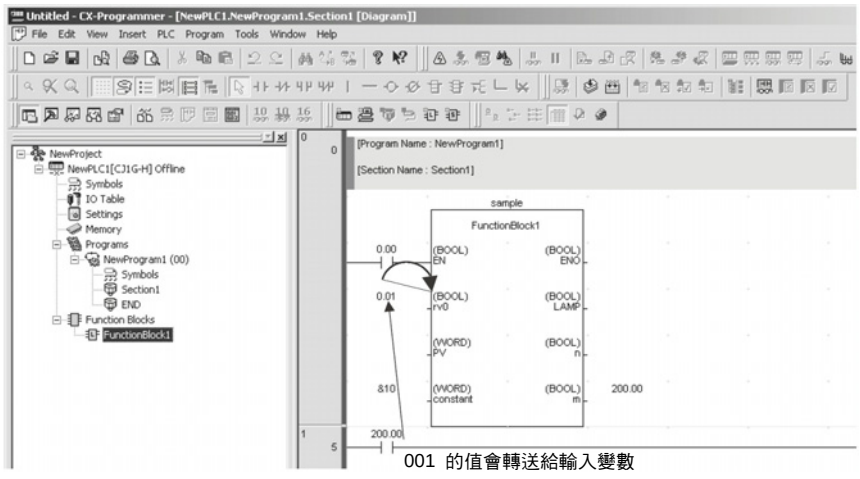
備註 (a) 功能區塊的輸入變數資料大小與程式的符號資料大小必須相符。

(b) 功能區塊的輸出變數資料大小與程式的符號資料大小必須相符。

1,2,3… 1. 輸入是在實例的左側而輸出則在右側。將游標放在要設定參數的位置並按 **Enter 鍵**。(或者，從插入主選單選取 ***Function Block Parameter (功能區塊參數)***。)接著會顯示"新參數"對話框，如以下所示。



2. 設定要從哪裡轉送位址給輸入變數的來源位址。同時設定要從輸出變數轉送位址資料給哪裡的目的位址。



備註 設定所有輸入參數中的資料。如果即使只有一個輸入參數保留空白，實例的左側匯流排也會顯示為紅色來表示錯誤。如果有這種情況發生，程式不能傳送給 CPU 模組。

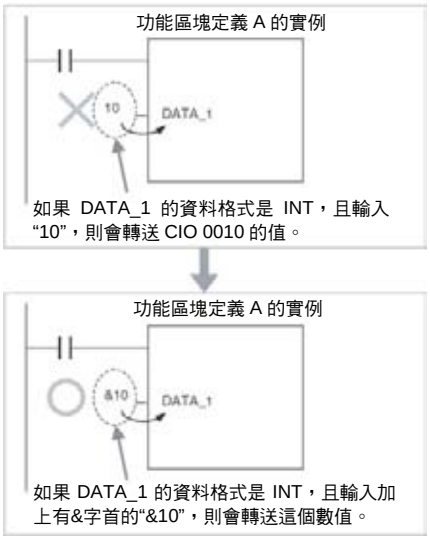
輸入參數中的值

下表列出在參數中輸入值的方法。

輸入變數資料類型	目錄	大小	輸入方法	設定範圍
BOOL	位元資料	1 位元	P_Off，P_On	0 (FALSE), 1 (TRUE)
INT	整數	16 位元	正值：&或+後隨整數 負值：-後隨整數	-32768 至+32767
DINT	二位整數	32 位元		-2147483648 至+2147483647
LINT	長(4 個字)整數	64 位元		-9223372036854775808 至 +9223372036854775807
UINT	無正負號整數	16 位元	正值：&或+後隨整數	&0 至 65535
UDINT	無正負號雙重整數	32 位元		&0 至 4294967295
ULINT	無正負號長(4 個字)整數	64 位元		&0 至 18446744073709551615
REAL	實數	32 位元	正值：&或+後隨實數(有小數點) 負值：- 後隨實數(有小數點)	-3.402823×10^{38} 至$-1.175494 \times 10^{-38}$、0、$+1.175494 \times 10^{-38}$ 至 $+3.402823 \times 10^{38}$
LREAL	長實數	64 位元		$-1.79769313486232 \times 10^{308}$ 至 $-2.22507385850720 \times 10^{-308}$、0、$+2.22507385850720 \times 10^{-308}$ 至 $+1.79769313486232 \times 10^{308}$
WORD	16 位元資料	16 位元	#後隨十六進位數(最大 4 位數) &或+後隨十進位數	#0000 至 FFFF 或&0 至 65535
DWORD	32 位元資料	32 位元	#後隨十六進位數(最大 8 位數) &或+後隨十進位數	#00000000 至 FFFFFFFF 或&0 至 4294967295
LWORD	64 位元資料	64 位元	#後隨十六進位數(最大 16 位數) &或+後隨十進位數	#0000000000000000 至 FFFFFFFFFFFFFFFF 或&0 至 18446744073709551615

備註 如果輸入變數使用一個非 boolean 的資料類型且只輸入一個數值(例如 20)，則會轉送 CIO 區域位址的值(例如 CIO 0020)，而不是這個數值。要設定一個數值時，請務必在輸入數值前插入一個&、#、+或-的字首。

範例程式：



如果輸入變數的資料類型是 boolean 且參數中只輸入一個數值(例如 0 或 1)，則會轉送 CIO 000000 (0.00)或 CIO 000001 (0.01)的值。請務必將 0(OFF)輸入為 P_Off 及將 1(ON)輸入為 P_On。

3-2-6 設定 FB 實例區

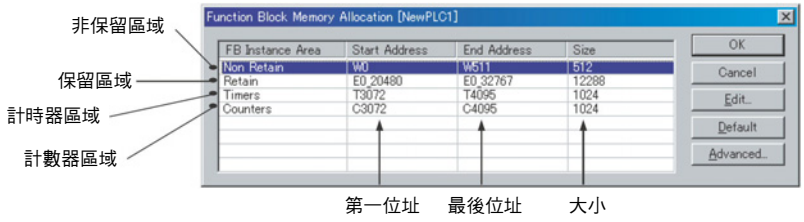
可以設定功能區塊中所使用的變數位址所在的區域。這些區域稱為功能區塊實例區。

1,2,3...

1. 在階梯區段視窗中或在全域符號表中選取實例，然後從 PLC 主選單選取 ***Function Block Memory (功能區塊記憶體) - Function Block Memory Allocation (功能區塊記憶體配置)***。

   接著會顯示如下所示的“功能區塊記憶體配置”對話。

2. 設定 FB 實例區。



非保留及保留區域以字組設定。計時器及計數器區域則以時間及計數器號碼設定。

相關預設值如下：

版本 3.0 以上的 CS/CJ 系列 CPU 模組及 NSJ 控制器

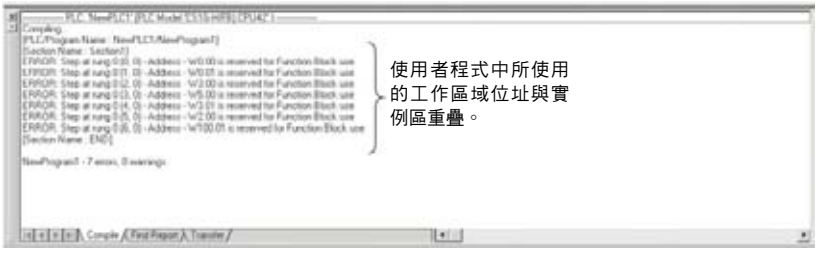
FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留(請參閱備註 1 及 3)	H512 (請參閱備註 2)	H1407 (請參閱備註 2)	896	CIO, WR, HR, DM, EM
保留(請參閱備註 1)	H1408 (請參閱備註 2)	H1535 (請參閱備註 2)	128	HR, DM, EM
計時器	T3072	T4095	1024	TIM
計數器	C3072	C4095	1024	CNT

備註 (1) 即使指定DM或EM區域用於非保留區域或保留區域，位元資料仍可以存取。

(2) 功能區塊保留區域字組會被指定為在 H512 到 H1535。使用者程式中的指令運算元中不能指定這些字組。內部變數的AT設定中也不可以指定這些字組。

(3) 保留區域中包含有字組 H512 到 H1535，但設定為非保留的位址會在電源關閉並再次開啟時或在作業開始時被清除。

(4) 為避免實例區位址與程式中所使用的位址重疊，請為非保留區域及保留區域設定 H512 到 H1535 (功能區塊保留區域字組)。如果設定另一個區域，則位址可能會與使用者程式中所使用的位址重疊。如果功能區塊實例區中的位址與使用者程式中所使用的任一個位址重疊，則會在匯集時發生一個錯誤。這個錯誤也會發生在下載一個程式、進行線上編輯、或由使用者檢查時。



如果複製位址並發生一個錯誤，請變更功能區塊實例區或者使用者程式中所使用的位址。

FQM1 彈性位置控制器

FB 實例區	預設值			適用記憶區
	起始位址	結束位址	大小	
非保留	5000	5999	1000	CIO, WR, DM
保留	無			
計時器	T206	T255	50	TIM
計數器	C206	C255	50	CNT

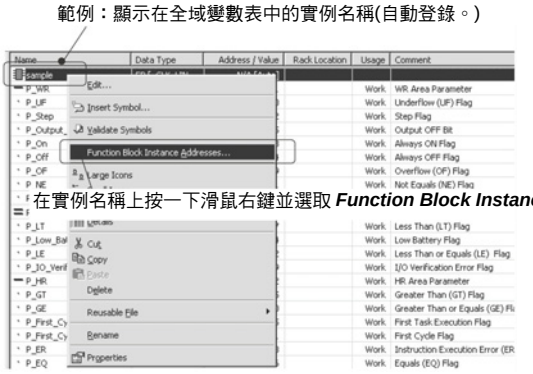
備註 即使指定 DM 區域用於非保留區域，位元資料仍可以存取。

3-2-7 檢查變數的內部位址配置

下列程序可以用來檢查內部分配給變數的 I/O 記憶體位址。

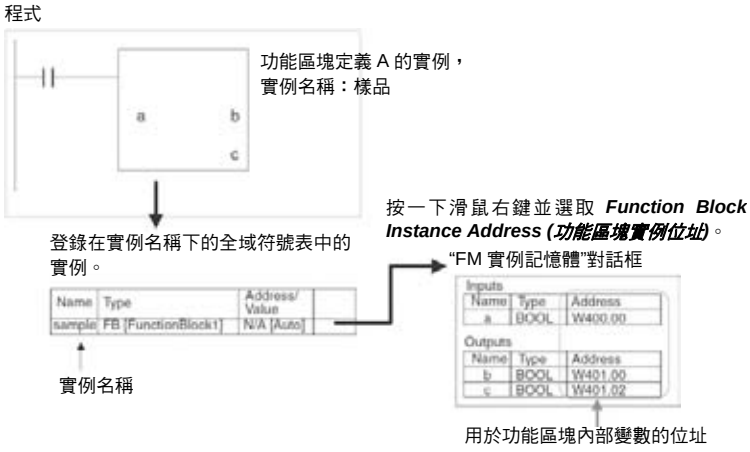
1,2,3... 1. 選取 ***View (檢視) - Symbols (符號) - Global (全域)***。

2. 在全域符號表中選取實例，按一下滑鼠右鍵，並從快顯主選單選取***Function Block Memory Address (功能區塊記憶體位址)***。(或者，從 PLC 主選單選取 ***Function Block Memory (功能區塊記憶體) - Function Block Memory Address (功能區塊記憶體位址)***。)



3. 接著會顯示"FB 介面記憶體"對話框。在這裡檢查內部分配給變數的 I/O 記憶體位址。

用來檢查內部分配給變數的位址的方法

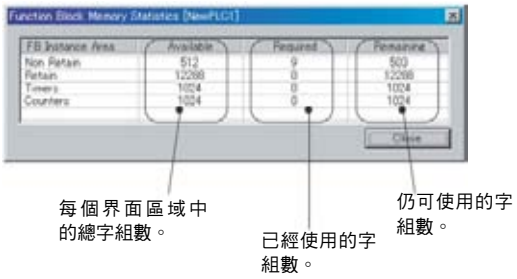


檢查內部分配給變數的位址的狀態

下列程序可以用來檢查分配給變數的位址數以及功能區塊實例區中仍可用於分配的位址數。

1,2,3...

1. 在“階梯區段”視窗中選取實例，按一下滑鼠右鍵，並從 PLC 主選單選取 ***Function Block Memory (功能區塊記憶體) - Function Block Memory Statistics (功能區塊記憶體統計)***。
2. 接著會顯示“功能區塊記憶體統計”對話框，如以下所示。在這裡檢查位址的使用情形。



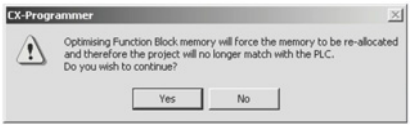
功能記憶體最佳化

在新增或刪除一個變數時，變數實例區中的位址會自動重新分配。每個實例都需要連續的位址，所以如果原來的位址區塊無法容納變數的變更，則所有的變數都會被分配到一個不同的位址區塊。這會導致產生一個未使用的位址區塊。下列程序可以用來剔除記憶體中未使用的區域使記憶體可以更有效率的使用。

1,2,3...

1. 在“階梯區段”視窗中選取實例，按一下滑鼠右鍵，並從 PLC 主選單選取 ***Function Block Memory (功能區塊記憶體) - Optimize Function Memory (功能記憶體最佳化)***。

   將會出現下列對話框。



2. 按一下 **OK** 按鈕，功能區塊實例區的配置就會最佳化。

3-2-8 複製及編輯功能區塊定義

請利用下列作業來複製及編輯已經建立的功能區塊定義。

1. 選取要複製的功能區塊，按一下滑鼠右鍵，並從快顯主選單選取 ***Copy (複製)***。
2. 將游標放在專案目錄中的 PLC 下的功能區塊項目上，按一下滑鼠右鍵並從快顯主選單選取 ***Paste (貼上)***。
3. 功能區塊定義接著就會被複製(在複製來源處的功能區塊定義名稱前會顯示“copy”字樣)。
4. 要變更功能區塊名稱時，按一下滑鼠左鍵或按一下滑鼠右鍵並從快顯主選單選取 ***Rename (重新命名)***。
5. 按兩下功能區塊定義可進行編輯。

3-2-9 從一個實例檢查來源功能區塊定義

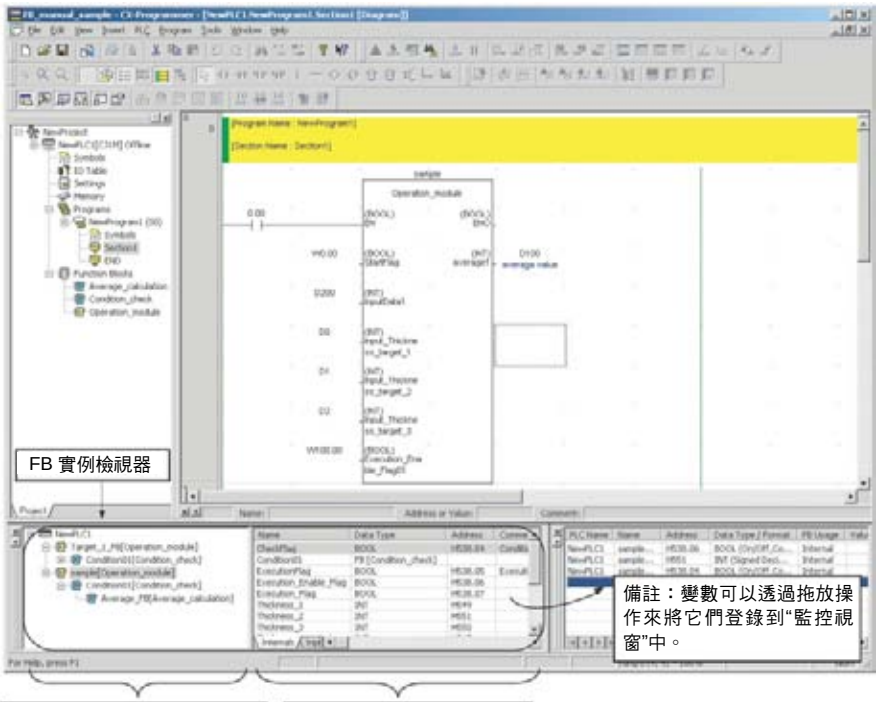
請利用下列程序來檢查一個實例所建立的功能區塊定義。

按兩下實例或在實例上按一下滑鼠右鍵並從快顯主選單選取 ***Go To (到) - Function Block Definition (功能區塊定義)***。功能區塊定義接著就會顯示出來。

3-2-10 檢查實例資訊(如巢狀層級)

當功能區塊巢狀到建立的程式中，可以透過從“檢視”主選單選取 ***Windows (視窗) - FB Instance Viewer (FB 實例檢視器)***來檢查巢狀層級的結構。功能區塊的關係會以樹狀目錄的格式顯示，其中呼叫功能區塊在較高的層級而被呼叫的功能區塊在較低的層級。

“FB 實例檢視器”視窗也會提供其他資訊，諸如所使用的陣列變數及分配給變數的內部位址，如下圖所示。可以透過將變數從用於實例中的變數清單中拖曳出來並將變數放入“監控視窗”中即可將變數登錄到“監控視窗”。

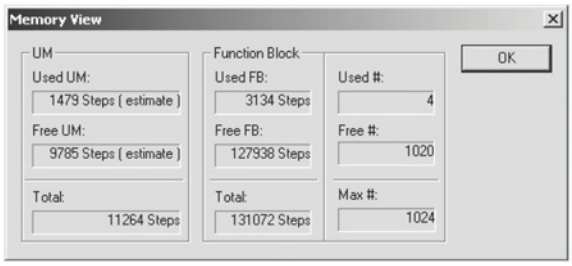


3-2-11 檢查功能區塊定義的大小

CX-Programmer 可以用來檢查利用類似方法建立的功能區塊定義的大小以便檢查程式的容量。程序如下：

1. 從“檢視”主選單選取 ***Memory View (記憶體檢視)***。

2. 功能區塊定義的大小和功能區塊定義數接著會顯示在“記憶體檢視”對話框中，如以下所示。



- *Function Block (功能區塊) 下方的 Used #、Free #及 Max #*欄位，所指的是功能區塊定義的數目。

3-2-12 匯集功能區塊定義(檢查程式)

可以匯集一個功能區塊定義來對它執行程式檢查。使用下列程序。

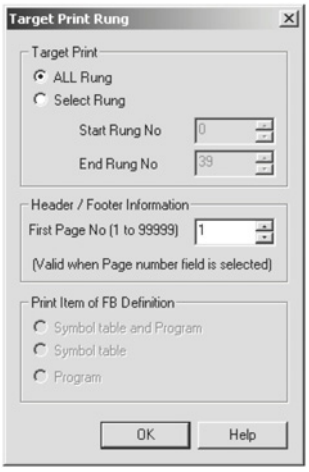
1,2,3… 選取功能區塊定義，按一下滑鼠右鍵，並從快顯主選單選取 ***Compile (匯集)***。(或者，按 **Ctrl+F7 鍵**。)

功能區塊接著就會被匯集且程式檢查的結果會自動顯示在“輸出”視窗的“Compile Table”(匯集表)頁面。

3-2-13 列印功能區塊定義

請利用下列程序來列印功能區塊定義。

1,2,3… 1. 按兩下要列印的功能區塊定義，並在變數表及演算法則出來時，從“檔案”主選單選取 ***Print (列印)***。接著會顯示下列“Target Print Rung (目標列印階梯圖)”對話框。



2. 選取 ***All Rung (所有梯級)***或 ***Select Rung (選取梯級)***選項。若選取 Select Rung (選取梯級)選項，請指定開始梯級和截止梯級的號碼。如果已經在 *File (檔案) - Page Setup (頁面設定)*中的頁首及頁尾欄中指定有一個頁碼，則可以指定第一頁的頁碼。
3. 針對功能區塊列印範圍選取下列任一個選項。
   - 符號表及程式(預設)
   - 符號表
   - 程式
4. 按一下 **OK** 按鈕，並顯示"列印"對話框。在設定印表機、要列印的項目數、以及頁面設定後，按一下 OK 按鈕。
5. 後隨演算法則(例如階梯圖程式編輯語言)的下列變數表接著就會列印出來。

備註 關於列印設定的詳細資訊，請參考 *CX-Programmer 5.0 版操作手冊*(W437)中有關列印的章節。

3-2-14 功能區塊定義的密碼保護

概覽

一個專案中的功能區塊定義可以設定密碼來限制存取以提供保護。可以根據適當情況來設定下列兩個層級的密碼保護。

同時針對寫入及讀取的密碼保護 這個密碼保護層級現限制寫入(變更)及顯示功能區塊定義的內容。

要設定讀取/寫入保護時，請在功能區塊的屬性中選取 *Prohibit writing and display (禁止寫入及顯示)*做為*保護類別*。這個保護層級可以防止不小心的程式變更/修改並且也可以保護程式內容被盜用。

僅針對寫入的密碼保護

這個密碼保護層級現限制寫入(變更)及顯示功能區塊定義的內容。

要設定寫入保護時，請在功能區塊的屬性中選取 *Prohibit writing (禁止寫入)*做為*保護類別*。這個保護層級可以防止不小心的程式變更/修改。

設定密碼保護

這項作業只能離線執行。

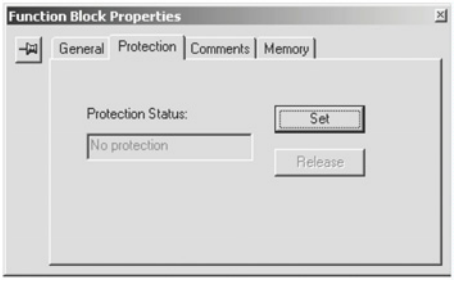
密碼保護可以套用到個別功能區塊定義上或多個功能區塊定義上。

保護個別功能區塊定義

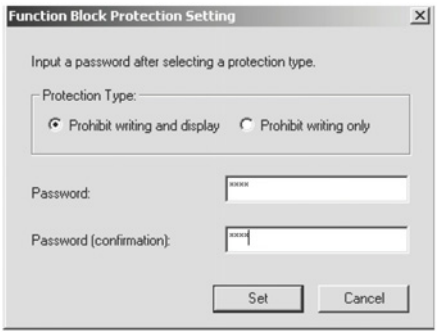
請利用下列程序來針對一個個別功能區塊定義設定密碼保護。

1,2,3...

1. 在專案工作區中，選取功能區塊定義，按一下滑鼠右鍵，並從快顯主選單選取 ***Properties (屬性)***。(或者，從"檢視"主選單選取 ***Properties (屬性)***。)
2. 接著會顯示"功能區塊屬性"對話框。按一下 **Protection (保護)**索引標籤並按一下 **Set (設定)**按鈕。



3. 接著會顯示"功能區塊保護設定"對話框。在 *Protection Type (保護類別)*欄中選取保護層級。



下表顯示每個保護層級中受限的功能。

功能	保護類別	
	禁止寫入及顯示	禁止寫入
顯示功能區塊內容	禁止	允許
列印功能區塊內容		
編輯功能區塊內容		禁止
儲存/載入到功能區塊資料庫檔案	允許	允許

4. 在"功能區塊保護設定"對話框的 *Password (密碼)*欄中輸入密碼。在確認欄中再次輸入相同的密碼來確定密碼並按一下 **Set (設定)**按鈕。
密碼長度最多可為 8 個字元，且只能使用數字與文字字元。
5. 當一個功能區塊定義設定有密碼保護後，這個功能區塊定義的圖示會變更來表示它已經受到保護。圖示也會顯示保護層級，如以下所示。

   ：禁止寫入及顯示(階梯及 ST 都相同)

   ：禁止寫入(ladder)

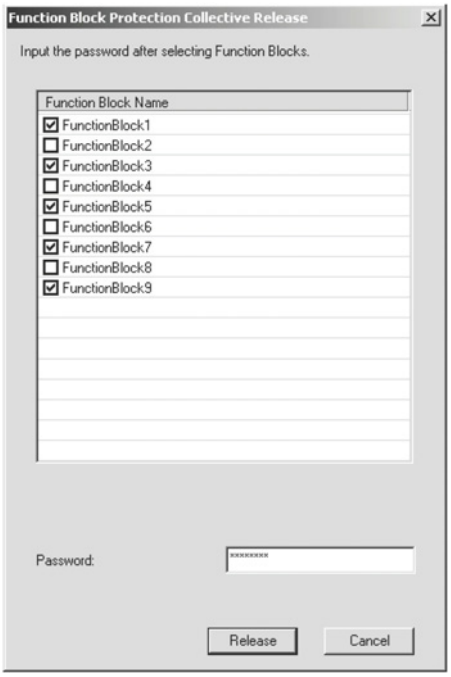
   ：禁止寫入(ST)

保護數個功能區塊定義

請利用下列程序來同時針對兩個以上的功能區塊定義設定密碼保護。

1,2,3...

1. 在專案工作區中選取 ***Function Blocks (功能區塊)***，按一下滑鼠右鍵，並從快顯主選單選取 ***Function Block Protection (功能區塊保護) - Set (設定)***。
2. 接著會顯示"功能區塊保護集團設定"對話框。選取您要保護的功能區塊的名稱，選取 *Protection Type* (保護層級)，輸入密碼，並按一下 **Set (設定)**按鈕。



3. 所選取的功能區塊定義接著就會有密碼保護。

清除密碼保護

這項作業只能離線執行。

可以清除個別功能區塊定義或多個功能區塊定義的密碼保護。

清除個別功能區塊的密碼保護

請利用下列程序來清除一個個別功能區塊定義的密碼保護。

1,2,3…

1. 在專案工作區中，選取功能區塊定義，按一下滑鼠右鍵，並從快顯主選單選取 ***Properties (屬性)***。(或者，從"檢視"主選單選取 ***Properties (屬性)***。)
2. 接著會顯示"功能區塊屬性"對話框。按一下 **Protection (保護)**索引標籤並按一下 **Release (解除)**按鈕。
3. 接著會顯示"功能區塊保護解除"對話框。在 *Password (密碼)*欄中輸入密碼並按一下 **Release (解除)**按鈕。
4. 如果密碼正確，保護就會被清除且專案工作區中的功能區塊定義的圖示會變更為正常的圖示。

清除多個功能區塊的密碼保護

請利用下列程序來同時清除兩個以上的功能區塊定義的密碼保護。

1,2,3…

1. 在專案工作區中選取 ***Function Blocks (功能區塊)***，按一下滑鼠右鍵，並從快顯主選單選取 ***Function Block Protection (功能區塊保護) - Release (解除)***。
2. 接著會顯示"功能區塊保護集團解隱"對話框。選取您要解除保護的功能區塊的名稱，輸入密碼，並按一下 **Release (解除)**按鈕。
3. 如果輸入的密碼與所選取的功能區塊的密碼相符，則所有功能區塊定義的保護都會立即清除。

3-2-15 儲存及重複使用功能區塊定義檔案

已經建立的功能區塊定義可以獨立儲存為一個功能區塊資料庫檔案(*.cxf)以便可以重複使用在其他專案中。

備註

(1) 在儲存檔案、或重複使用在另一個專案之前，請匯集功能區塊定義並執行一次程式檢查。

(2) 在巢狀功能區塊時，被呼叫(巢狀)的功能區塊的功能區塊定義會包括並儲存在功能區塊資料庫檔案中。

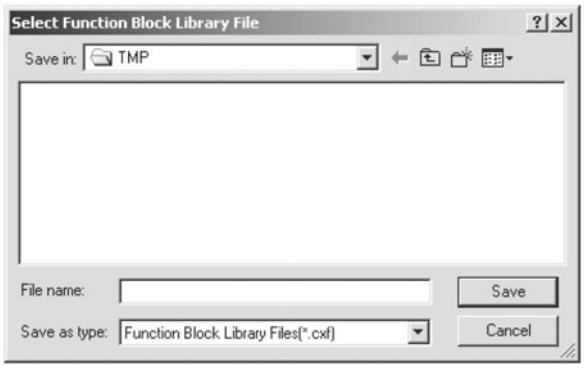
儲存一個功能區塊資料庫檔案

請利用下列程序來將一個功能區塊定義儲存為一個功能區塊資料庫檔案。

1,2,3…

1. 選取功能區塊定義，按一下滑鼠右鍵，並從快顯主選單選取 ***Save Function Block to File (將功能區塊儲存為檔案)***。(或者，從"檔案"主選單選取 ***Function Block (功能區塊) - Save Function Block to File (將功能區塊儲存為檔案)***。)

2. 將會出現下列對話框。輸入檔案名稱。應選取*功能區塊資料庫檔案(*.cxf)*做為檔案類型。

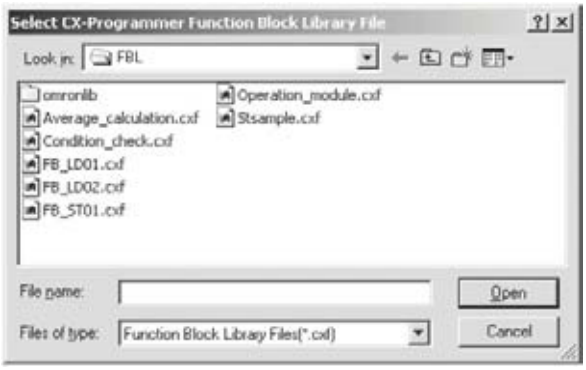


將功能區塊資料庫檔案讀入其他專案

請利用下列程序來將一個功能區塊資料庫檔案(*.cxf)讀入一個專案中。

1,2,3…

1. 選取專案工作區中的 PLC 目錄下的功能區塊定義項目，按一下滑鼠右鍵，並從快顯主選單選取 ***Insert Function Block (插入功能區塊) - From File (從檔案)*** (或選取 ***File (檔案) - Function Block (功能區塊) - Load Function Block from File (從檔案載入功能區塊)***)。
2. 將會出現下列對話框。選取一個功能區塊資料庫檔案(*.cxf)並按一下 **Open (開啟)**按鈕。



3. 一個稱為 FunctionBlock1 的功能區塊，接著會自動插入到功能區塊圖示之後。這個圖示包含有功能區塊的定義。
4. 接著會顯示變數表及演算法則。按兩下 **FunctionBlock1** 圖示。

3-2-16 下載/上傳程式到實際 CPU 模組

在一個包含有功能區塊的程式建立後，可以將它從 CX-Programmer 下載到一個線上連結的實際 CPU 模組上。程式也可以從實際 CPU 模組上傳。同時，也可以檢查 CX-Programmer (個人電腦)上和實際 CPU 模組中的程式是否相同。不過，如果程式包含有功能區塊，則不能下載到 TASK (工件)中(但可以上傳)。

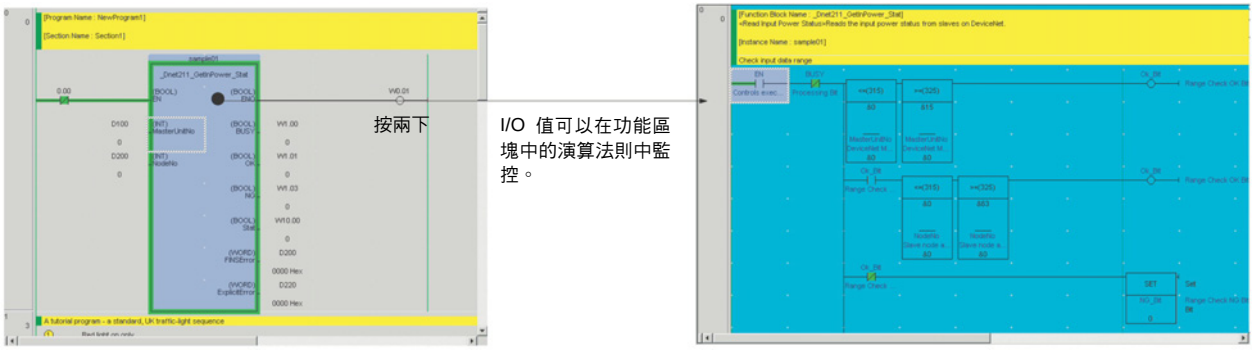
3-2-17 監控及進行功能區塊除錯

下列程序可以用來監控包含有功能區塊的程式。

監控實例中的階梯圖程式的 I/O

在 CX-Programmer 版本 6.0 以上的版本方面，可以在監控程式時監控一個實例中的階梯圖程式的位元狀態和字組內容。要監控 I/O 位元及字組(I/O 位元監控)時，請按兩下實例或在實例上按一下滑鼠右鍵並從快顯功能表選取 ***Monitor FB Ladder Instance(監控 FB 階梯實例)***。此時，可以監控位元和字組、變更 PV、強制設定/重置位元、以及執行區別監控。

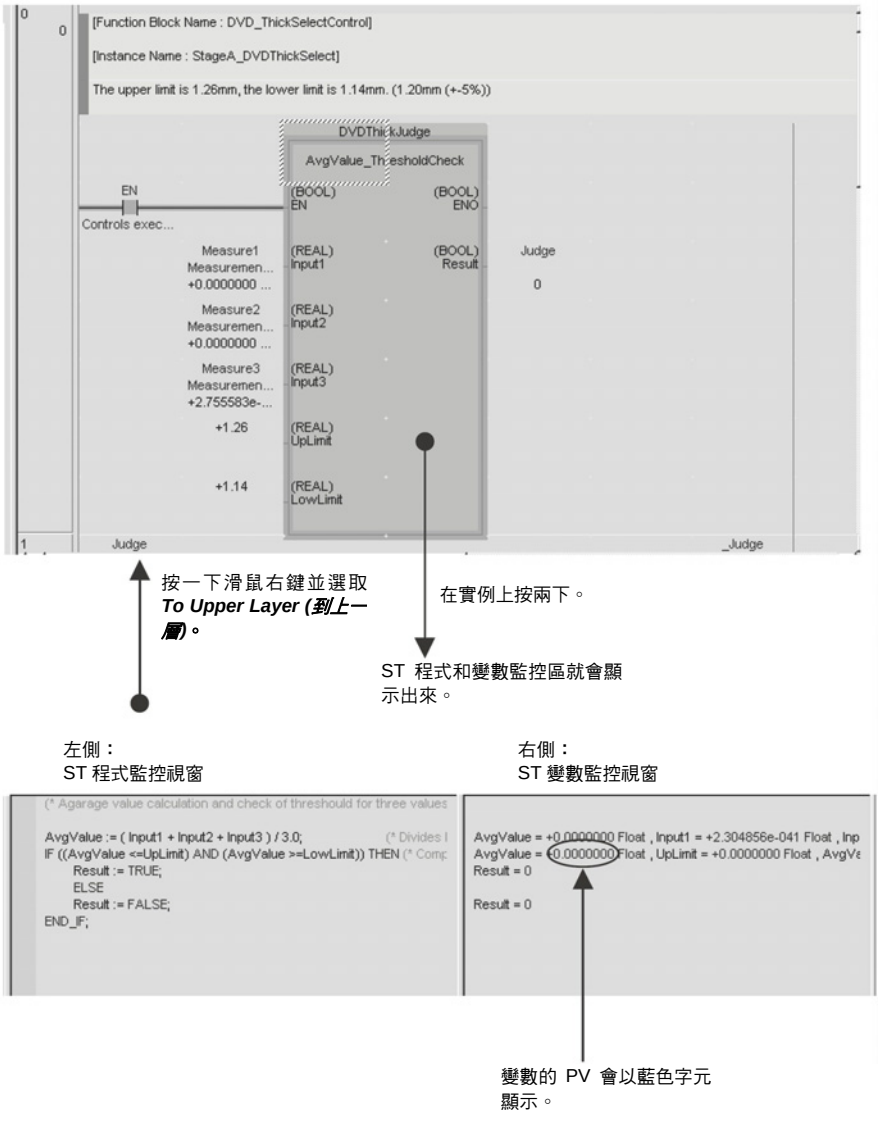
備註 不能執行線上編輯或變更計時器/計數器 SV。



監控實例中的 ST 程式的變數

在 CX-Programmer 版本 6.1 以上的版本方面，可以在監控程式時監控一個實例中的 ST 程式。要監控 I/O 位元及字組(I/O 位元監控)時，請按兩下實例或在實例上按一下滑鼠右鍵並從快顯功能表選取 ***Monitor FB Instance (監控 FB 實例)***。

如果要回到原來的實例上，請在 ST 程式監控視窗中按一下滑鼠右鍵並從快顯功能表選取 ***To Upper Layer (到上一層)***。



ST 程式會顯示在視窗的左側(稱為 ST 程式監控視窗)。

ST 程式中所使用的變數的值會顯示在視窗的右側(稱為 ST 變數監控視窗)。

此時，可以監控變數值、變更 PV、強制設定或強制重置位元、及在監控視窗中進行變數的複製/貼上。(這些這些操作將在以下說明。)

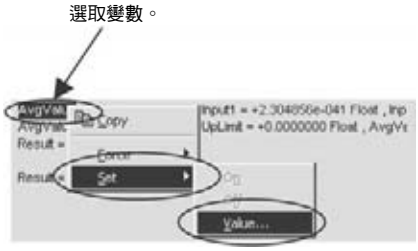
備註 ST 程式不能進行線上編輯。

監控變數

變數值會在 ST 變數監控視窗中以藍色顯示。

變更 PVs

要變更一個 PV 時，請在 ST 變數監控視窗中選取想要的變數(被選取時會以反白顯示)，按一下滑鼠右鍵，並從快顯功能表選取 ***Set(設定) - Value(值)***。



接著會顯示"設定新值"對話框。請在 *Value(值)*的欄位中輸入新的值。

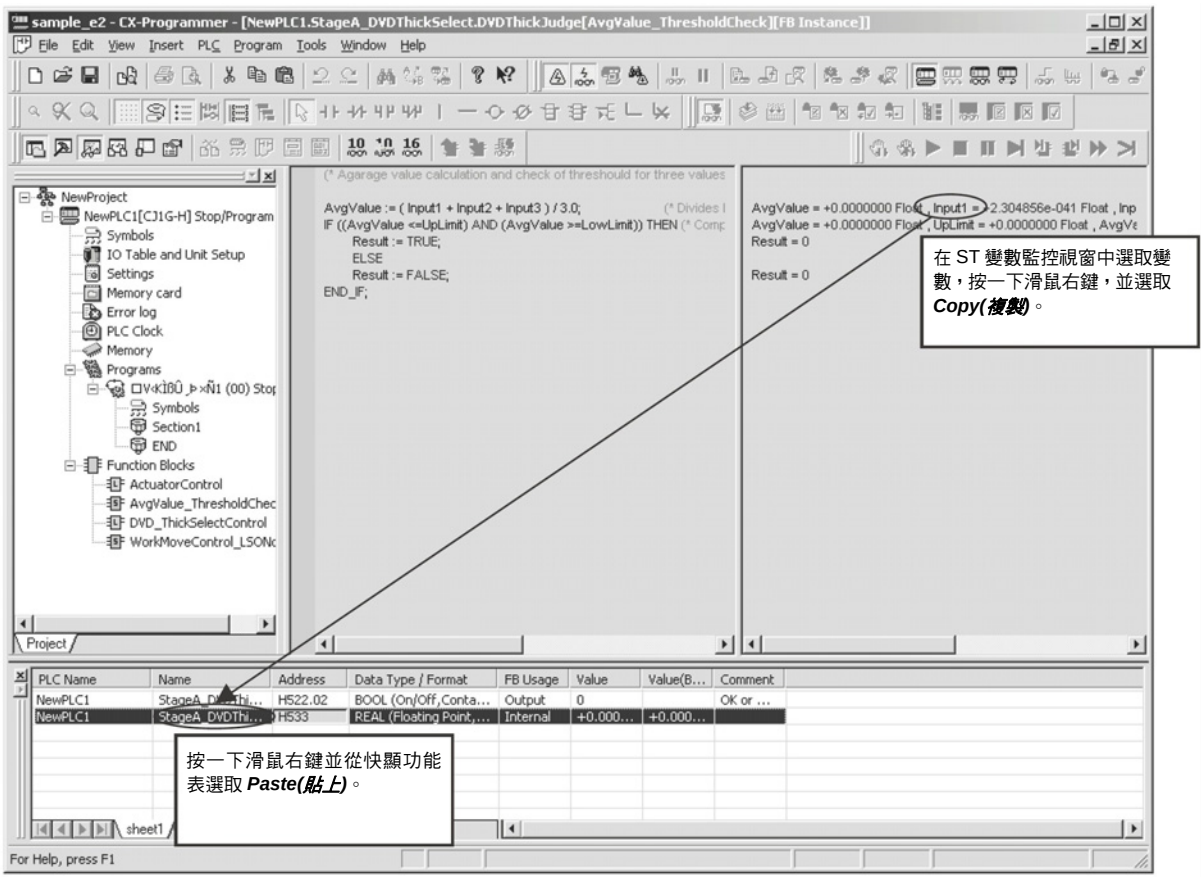
強制設定及強制重置位元

要強制設定、強制重置、或清除強制狀態時，請在 ST 變數監控視窗中選取想要的變數(被選取時會以反白顯示)，按一下滑鼠右鍵，並從快顯功能表選取 ***Force(強制) - On***、***Force(強制) - Off***、***Force(強制) - Cancel(取消)***、或 ***Force(強制) - Cancel All Forces(取消所有強制)***。

在監控視窗進行複製及貼上

1,2,3…

1. 要複製一個變數到監控視窗時，請在 ST 變數監控視窗中選取想要的變數(被選取時會以反白顯示)，按一下滑鼠右鍵，並從快顯功能表選取 ***Copy(複製)***。
2. 在監控視窗中按一下滑鼠右鍵並從快顯功能表選取 ***Paste(貼上)***。



檢查功能區塊定義中的程式

利用下列程序可在監控時檢查一個實例的功能區塊定義中的程式。

1,2,3… 在實例上按一下滑鼠右鍵並從快顯功能表選取 ***To Lower Layer(到下一層)***。功能區塊定義接著就會顯示出來。

在監控視窗中監控實例變數

請利用下列程序來監控實例變數。

1,2,3…

1. 選取 ***View(檢視) - Window(視窗) - Watch(監控)***。

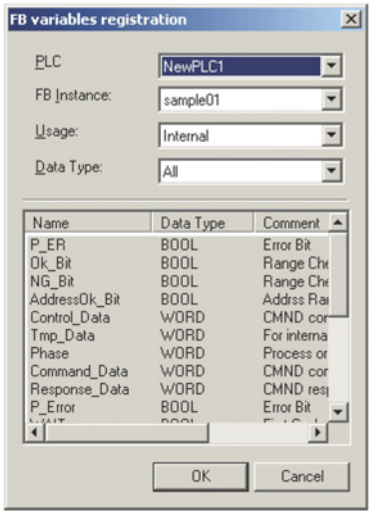
   接著會顯示一個監控視窗。

2. 起使用下列 3 種方法中的任一種方法來顯示*"FB 變數登錄"*對話框。

   a. 在實例上按一下滑鼠右鍵並從快顯功能表選取 ***Register in Watch Windows(登錄到監控視窗中)***。

   b. 複製實例並將它貼到監控視窗中。

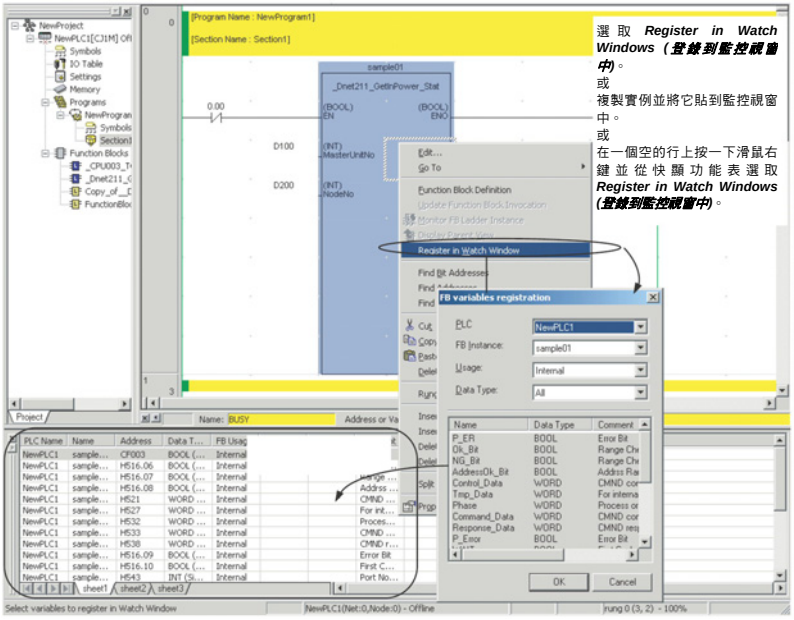
   c. 在監控視窗中的一個空的行上按一下滑鼠右鍵並從快顯功能表選取 ***Register in Watch Windows(登錄到監控視窗中)***。



3. 選取 ***Usage(用法) - Data Type(資料類型)***。也可以選取 ***FB 實例***設定。預設的*用法*是 ***N：內部***而其他可用的選擇是 ***I：輸入***、***O：輸出***、及 ***E：外部***。

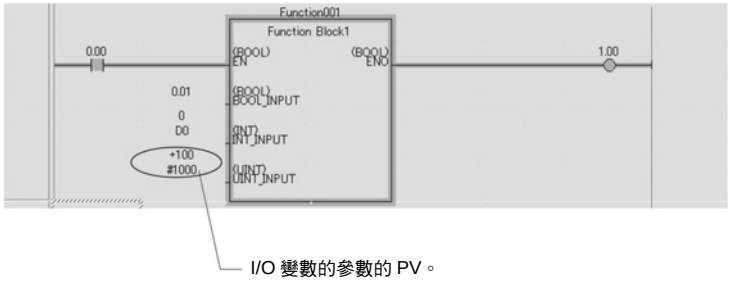
   預設的資料類型為 ***A：全部。***也可以選取特殊資料類型 ***BOOL*** 及 ***INT***。

4. 按一下 **OK** 按鈕，所選取的變數就會被登錄到監控視窗中並且會顯示它的值，如以下所示。



監控實例 I/O 變數

I/O 變數的參數的現行值會顯示在參數的下方。



線上編輯功能區塊定義程式

使用功能區塊的程式可以線上編輯。同時也可以對實例進行變更。

- 實例的參數可以變更，實例可以刪除，而實例中除此以外的其他指令可以變更。
- 實例不能新增，實例名稱不能變更，而功能區塊定義中的演算法則和變數表也不能變更。

模擬實例中的階梯/ST 程式

CX-One 版本 1.1(CX-Programmer 版本 6.1)以上的版本有一個模擬功能可以模擬一個功能區塊實例中的階梯圖程式或 ST 程式的作用。它同時支援步驟執行和中止點操作如果要回到原來的實例上，請在 ST 程式監控視窗中按一下滑鼠右鍵並從快顯功能表選取 ***To Upper Layer (到上一層)***。。

■ 啟用模擬功能

請利用下列程序來啟用模擬功能。

1,2,3...

1. 開啟包含有要進行除錯的實例的程式。
2. 選擇 ***View (檢視)—Toolbars (工具列)***，再選取 **Toolbars (工具列)**標籤中的 *Simulator Debug (模擬器除錯)*選項。
3. 從 CX-Programmer 的 *PLC* 主選單中選擇 ***Work Online Simulator (與模擬器連線工作)***選項，然後將程式傳送給電腦中的 CX-Simulator。

備註 步驟 2 和 3 可以以相反的順序執行。

■ **步驟執行(Step Run)**

依漸進的步驟(指令)執行程式。當實例停止時，這個功能可以移到這個實例中的階梯或 ST 程式的第一個步驟(指令)。

實例中的程式可以透過 Step Run(步驟執行)或 Continuous Step Run(連續步驟執行)的方法來執行(請參閱附註)。

備註 選取 CX-Programmer 的 ***Tools(工具) - Options(選項)***指令並在 PLCs 索引標籤頁面上設定 *Continuous Step Interval(連續步驟間隔)*，來設定"連續步驟執行"作業的 Step Run 持續時間。

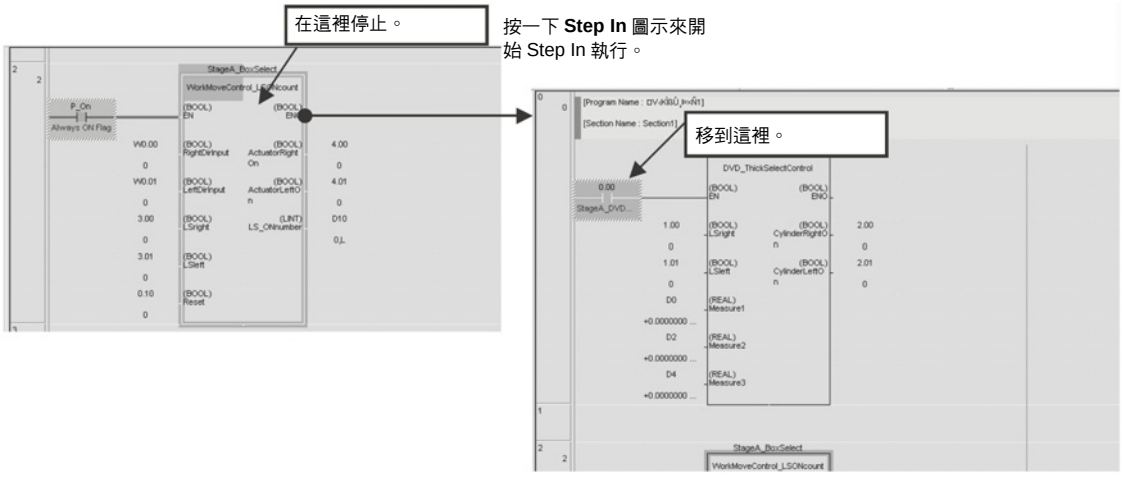
Step In (步驟執行開始)

請利用下列程序來開始一個實例中的一個階梯/ST 程式的步驟執行(稱為 Step Run 作業)。

1,2,3… 1. 暫停實例的執行。(請參閱備註)

2. 按一下 **Step In** 圖示或選取 ***Tools(工具) - Simulation(模擬) - Mode(模式) - Step In (步驟執行開始)***。

範例：從實例 Step In 到內部階梯圖程式



範例：從 SP 程式 Step In 到內部階梯圖程式

備註 當程式在一個功能區塊實例以外的點上執行時，相關的處理會與一般 Step Run 作業相同。

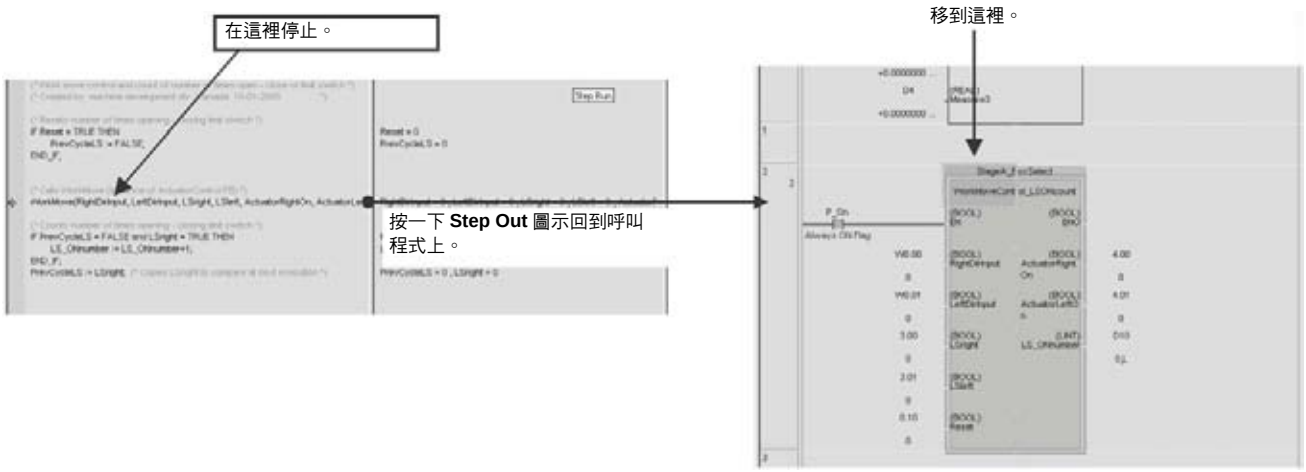
Step Out (步驟執行結束)

請利用下列程序來暫停一個實例中的一個階梯/ST 程式的步驟執行(Step Run 作業)並回到程式中較高的一層(呼叫來源的程式或實例)。

1,2,3… 1. 在 Step Run 作業中，將游標移到實例中的任何停止點上。

2. 按一下 **Step Out** 圖示或選取 ***Tools(工具) - Simulation(模擬) - Mode(模式) - Step* Out *(步驟執行結束)***。

範例程式：
從一個 ST 程式回到呼叫程式或實例



備註 Step Out 指令只能在一個實例中的一個階梯/ST 程式內執行。

■ **由模擬功能暫停作業時的顯示**

游標(或 ST 程式中的箭頭)的顏色會顯示"模擬功能"視窗中的一項作業是否已經暫停、以及是哪一項作業已經暫停。

除錯作業	顏色(預設)	程式執行狀態	詳細資料
步驟執行或連續步驟執行	粉紅色	模擬器暫停狀態	由步驟執行作業或透過 **Pause** 按鈕暫停
	正常顏色	由於連鎖或其他功能而沒有執行	步驟因為一個指令(如 IL、MILR/ MILH、JMP0、或 FOR/BREAK)而沒有執行
中止點	藍色	模擬器指令中止	由一個中止點暫停(中止狀態)

備註 (1) 在選取 ***Tools(工具) - Simulation(模擬) - Always Display Current Execution Point(永遠顯示目前執行點)***後，模擬器會在執行步驟執行或連續步驟執行作業時自動捲動顯示畫面來顯示實例中的暫停點。

(2) 在"模擬功能"視窗中用來表示一項作業暫停時的游標(或 ST 程式中的箭頭)的顏色可以改變為預設以外的顏色。
要變更顏色時，請選取 ***Tools(工具) - Options(選項)*** 並按一下 **Appearance(外觀)** 索引標籤。選取 *Pause Simulator(暫停模擬器)*、*Simulator Instruction Break(模擬器指令中止)*、或 *Simulator IO Break(模擬器 IO 中止)*，並變更相關狀況的顏色。

■ **一個實例中的中止點操作**

執行可以在實例中的預設中止點上暫停。(在這種情況下，Step In 作業不能使用。)

備註 當一個中止點針對一個實例設定時，中止點只對這個實例有效。(這個中止點對根據同一個功能區塊定義所建立的其他實例無效。)

附錄 A
資料型態

基本資料類型

資料型態	目錄	大小	值的範圍
BOOL	位元資料	1	0 (FALSE)、1 (TRUE)
INT	整數	16	-32,768 至+32,767
DINT	二位整數	32	-2,147,483,648 至+2,147,483,647
LINT	長(8 位元)整數	64	-9,223,372,036,854,775,808 至+9,223,372,036,854,775,807
UINT	無正負號整數	16	&0 至 65,535
UDINT	無正負號雙重整數	32	&0 至 4,294,967,295
ULINT	無正負號長(8 個字)整數	64	&0 至 18,446,744,073,709,551,615
REAL	實數	32	-3.402823×10^{38} 至$-1.175494 \times 10^{-38}$，0，$+1.175494 \times 10^{-38}$ 至 $+3.402823 \times 10^{38}$
LREAL	Long 實數	64	$-1.79769313486232 \times 10^{308}$ 至$-2.22507385850720 \times 10^{-308}$，0，$2.22507385850720 \times 10^{-308}$ 至 $1.79769313486232 \times 10^{308}$
WORD	16 位元資料	16	#0000 至 FFFF 或&0 至 65,535
DWORD	32 位元資料	32	#00000000 至 FFFFFFFF 或&0 至 4,294,967,295
LWORD	64 位元資料	64	#0000000000000000 至 FFFFFFFFFFFFFFFF 或 &0 至 18,446,744,073,709,551,615
計時器 (請參閱備註)	計時器(請參閱備註)	旗標：1 位元 PV：16 位元	計時器號碼：0 至 4095 完成旗標：0 或 1 計時器 PV：0 到 9999 (BCD)，0 到 65535 (二進位)
計數器 (請參閱備註)	計數器(請參閱備註)	旗標：1 位元 PV：16 位元	計數器號碼：0 至 4095 完成旗標：0 或 1 計數器 PV：0 至 9999 (BCD)，0 至 65535 (二進位)
功能區塊	功能區塊實例	---	---

備註 TIMER(計時器)和 COUNTER(計數器)資料類型不能用在結構化文字功能區塊中。

衍生資料類型

陣列	1 次元陣列；最多 32,000 個元件

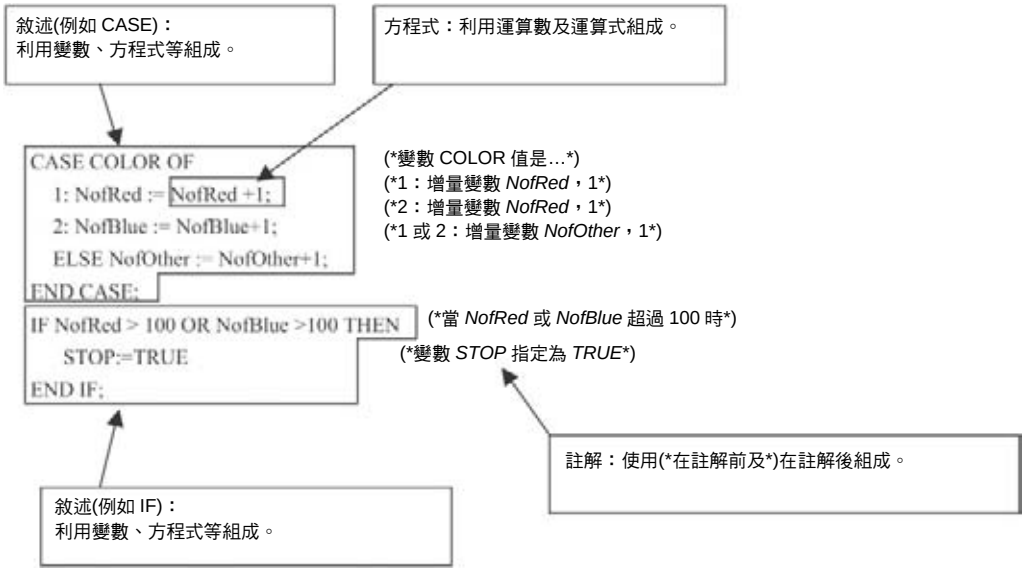
附錄 B
結構化文字(ST 語言)
規格

結構化文字

結構化文字(也稱為 ST 語言)是一種類似於使用諸如選擇及反覆敘述的語言結構的 PASCAL 的高階程式語言。使用結構化文字編寫的程式會利用敘述來組成。敘述會由變數及方程式組成。

- 方程式也就是包含有運算數和運算式(變數或常數)的數列。運算數包括算術運算數、比較運算數、以及邏輯運算數。
- 敘述包括指定或控制敘述。指定敘述會儲存由變數中的方程式所得的計算結果。控制敘述則包括選擇敘述及反覆敘述。

結構化文字範例



限制

敘述分界符號

- 敘述(指定及控制敘述)必須永遠以一個分號(;)結尾。敘述不能只利用換行來完成。
- 不要在一個敘述中使用一個分號(;)來做為分界符號(諸如跟在保留字組、值、或方程式之後)。在一個敘述結尾以外的地方插入一個分號會導致語法錯誤。

註解

- 註解會以括弧和星號框起來，即：*(*註解*)*。除括弧及星號以外的任何字元都可以用在註解中。註解中不支援套疊。

表示	範例
*(*註解*)*	(*這是一個註解*)

備註：註解中不能進行套疊，即，(*(*不支援這種形式的套疊*)*)

空格、換行、定位點

- 任何數量的空格、換行、及定位點或它們的組合都可以用在敘述中的任何地方。因此，請在保留字組及方程式之間使用空格、換行、及定位點來讓它們更方便檢讀。
- 空格、換行、及定位點不能用在下列記號(匯集時最小的有意義單位)之間，在這樣的情況下它們稱為記號分隔符號。
  記號：保留字組、變數名稱、特殊字元、常數(數值)

  保留字組(大寫或小寫)：AND、CASE、DO、ELSE、FOR、IT、NOT、OF、OR、REPEAT、THEN、TO、UNTIL、WHILE、XOR、TRUE、FALSE、ELSIF、BY、EXIT、RETURN

  變數名稱：　任何沒有保留字組的文字都會被視為一個變數名稱。

  特殊字元：　<=、>=、<>、:=、.., &、(*, *)

  常數(數值)：　數值僅限於十進位數
  16# 後隨十六進位數的數值
  2# 後隨二進位數的數值
  8# 後隨八進位數的數值

  如果一個空格、換行、或定位點用在上述任何記號之間，則不論在任一側的記號部份都會被視為分隔記號。因此，請確定空格、換行、或定位點是否沒有用在一個個別記號中。
- 請務必在保留字組與變數名稱之間使用空格、換行、定位點、或其他記號分隔符號。其他記號組合之間可選擇性使用記號分隔符號。
  在下列範例中，方塊(□)表示那裡需要一個空格、換行、定位點、或其他記號分隔符號。

```
IF□A>0THEN□X=10;
ELSE□
  X:=0;
END_IF;
```

大寫及小寫

- 保留字組和變數名稱並沒有區分大小寫(都可以使用)。

禁止使用於變數名稱的字元

- 下列框在方括弧中的字元不能用在變數名稱中。
  [!]、[“]、[#]、[$]、[%]、[&]、[‘]、[(]、[)]、[、[-]、[=]、[^]、[~]、[\]、[|]、[@]、[‘]、[[]、[{]、[;]、[+]、[:]、[*]、[]]、[}]、[,]、[<]、[.]、[>]、[/]、[?]
- 數字 0 到 9 不能用來做為變數名稱的第一個字元。
- 一個底線不能緊隨在變數名稱中的另一個底線之後。
- 空格不能用在變數名稱中。

如果有任何這些字元以上述的方式使用，會出現一個錯誤訊息。

輸入常數(數值)

- 數值可以以十進位、十六進位、八進位、或二進位制來表示，如下列範例所示。

	表示方法	範例(十進位值 12)
十進位：	只有數值	12
十六進位：	16# 後隨數值	16#C
八進位：	8# 後隨數值	8#14
二進位：	2# 後隨數值	2#1100

運算數優先順序

- 請考慮結構化文字語法中的運算數的優先順序，或者將需要優先處理的運算框在括弧中。
  範例：AND 的優先順序優於 OR。因此，在 X OR Y AND Z 的範例中，Y AND Z 會優先處理。

CX-Programmer 的 ST 輸入畫面顯示

文字顯示顏色

在文字輸入或貼入 ST 輸入畫面時，CX-Programmer 會自動以下列顏色顯示文字。

- 文字關鍵字組(保留字組)：藍
- 註解：綠
- 錯誤：紅
- 其他：空白

變更字形

樣變更字形大小或顯示顏色時，請選取 ***Tools(工具) - Options(選項)***，按一下 **Appearance(外觀)**索引標籤，然後按一下 **ST Font (ST 字形)**按鈕。然後就可以變更字形名稱、字形大小(預設為 8 point)和顏色。

敘述

敘述	功能	範例
敘述的結尾	結束敘述	;
註解	所有(*與*)之間的文字都會被視為一個註解。	(**註解*)
指定	將表示式的結果、變數、或變數右側的值代入左側。	A:=B;

IF, THEN, ELSIF, ELSE, END_IF	在條件為真時評估一個表示式。	IF (condition_1) THEN
		(expression 1);
		ELSIF (condition_2) THEN
		(expression 2);
		ELSE
		(expression 3);
		END_IF;

CASE, ELSE, END_CASE	根據一個變數的值來評估一個表示。	CASE *(variable)* OF 1: *(expression 1)*; 2: *(expression 2)*; 3: *(expression 3)*; ELSE *(expression 4)*; END_CASE;
FOR, TO, BY, DO, END_FOR	根據起始值、最終值、及增量反覆評估一個表示式。	FOR *(identifier)* := *(initial_value)* TO *(final_value)* BY *(increment)* DO *(expression)*; END_FOR;
WHILE, DO, END_WHILE	只要條件為真時反覆評估一個表示式。	WHILE *(condition)* DO *(expression)*; END_WHILE;
REPEAT, UNTIL, END_REPEAT	反覆評估一個表示式直到條件為真。	REPEAT *(expression)*; UNTIL *(condition)* END_REPEAT;
關閉	停止反覆的處理。	EXIT;

敘述	功能	範例
RETURN	返回到程式中一個功能區塊被叫出的點。	RETURN;
功能區塊實例呼叫	叫出另一個功能區塊定義。	功能區塊資料類型的變數名稱(被呼叫的功能區塊定義的輸入變數名稱 := 呼叫功能區塊定義的變數名稱或常數，...，被呼叫的功能區塊定義的輸出變數名稱或常數 => 呼叫功能區塊定義的輸出變數名稱，...);

運算數

操作	符號	運算數所支援的資料類型	優先順序 1：Lowest 11：Highest
括弧及方括弧	(*表示式*)，*陣列*[*索引*]		1
功能評估	*識別碼*	取決於功能(請參考 *2-6 指令支援及運算式限制*)	2
指數	**	REAL, LREAL	3
互補	NOT	BOOL, WORD, DWORD, LWORD	4
乘法	*	INT, DINT, UINT, UDINT, ULINT, REAL, LREAL	5
除法	/	INT, DINT, LINT, UINT, UDINT, ULINT, REAL, LREAL	5
加法	+	INT, DINT, LINT, UINT, UDINT, ULINT, REAL, LREAL	6
減法	-	INT, DINT, LINT, UINT, UDINT, ULINT, REAL, LREAL	6
比較	<, >, <=, >=	BOOL, INT, DINT, LINT, UINT, UDINT, ULINT, WORD, DWORD, LWORD, REAL, LREAL	7
等於	=	BOOL, INT, DINT, LINT, UINT, UDINT, ULINT, WORD, DWORD, LWORD, REAL, LREAL	8
不等於	<>	BOOL, INT, DINT, LINT, UINT, UDINT, ULINT, WORD, DWORD, LWORD, REAL, LREAL	8
Boolean AND	&	BOOL, WORD, DWORD, LWORD	9
Boolean AND	AND	BOOL, WORD, DWORD, LWORD	9
Boolean 互斥 OR	XOR	BOOL, WORD, DWORD, LWORD	10
Boolean OR	OR	BOOL, WORD, DWORD, LWORD	11

備註 運算會根據資料類型來執行。
因此，(例如)INT 資料的加法結果，必須是一個使用 INT 資料類型的變數。當一個整數類型變數的運算中發生進位或借位時，必須特別注意。例如，使用整數類型變數 A=3 及 B= 2，如果執行(A/B)*2 的運算，A/B 的結果是 1(1.5，但小數點後的值捨去)，所以(A/B)*2 = 2。

功能

功能	語法
數值函數	絕對值，三角函數等。
算術函數	指數(EXPT)
資料類型轉換函數	*來源資料類型*_TO_*新資料類型*(*變數名稱*)

數值函數

下列數值函數可以用在結構化文字中。

數值函數	引數資料類型	回歸值資料類型	目錄	範例
ABS (*引數*)	INT, DINT, LINT, UINT, UDINT, ULINT, REAL, LREAL	INT, DINT, LINT, UINT, UDINT, ULINT, REAL, LREAL	絕對值[*引數*]	a: = ABS (b) (*變數 *b* 的絕對值儲存在變數 *a* 中*)
SQRT (*引數*)	REAL, LREAL	REAL, LREAL	平方根： √ 引數	a: = SQRT (b) (*變數 *b* 的平方根儲存在變數 *a* 中*)
LN (*引數*)	REAL, LREAL	REAL, LREAL	自然對數：LOG_e 引數	a: = LN (b) (*變數 *b* 的自然對數儲存在變數 *a* 中*)
LOG (*引數*)	REAL, LREAL	REAL, LREAL	常用對數： LOG_{10} 引數	a: = LOG (b) (*變數 *b* 的常用對數儲存在變數 *a* 中*)
EXP (*引數*)	REAL, LREAL	REAL, LREAL	自然指數函數：$e^{引數}$	a: = EXP (b) (*變數 *b* 的自然指數函數儲存在變數 *a* 中*)
SIN (*引數*)	REAL, LREAL	REAL, LREAL	正弦：SIN 引數	a: = SIN (b) (*變數 *b* 的正弦儲存在變數 *a* 中*)
COS (*引數*)	REAL, LREAL	REAL, LREAL	餘弦：COS 引數	a: = COS (b) (*變數 *b* 的餘弦儲存在變數 *a* 中*)
TAN (*引數*)	REAL, LREAL	REAL, LREAL	正切：TAN 引數	a: = TAN (b) (*變數 *b* 的正切儲存在變數 *a* 中*)
ASIN (*引數*)	REAL, LREAL	REAL, LREAL	ARC 正弦：SIN^{-1} 引數	a: = ASIN (b) (*變數 *b* 的 ARC 正弦儲存在變數 *a* 中*)
ACOS (*引數*)	REAL, LREAL	REAL, LREAL	ARC 餘弦：COS^{-1} 引數	a: = ACOS (b) (*變數 *b* 的 ARC 餘弦儲存在變數 *a* 中*)
ATAN (*引數*)	REAL, LREAL	REAL, LREAL	ARC 正切：TAN^{-1} 引數	a: = ATAN (b) (*變數 *b* 的 ARC 正切儲存在變數 *a* 中*)

備註 針對數值函數所回歸的資料類型會與引數中所使用的相同。因此，代入函數回歸值的變數必須與引數的資料類型相同。

算術函數

下列一般指數函數可以用在結構化文字中。

數值函數	引數資料類型	回歸值資料類型	目錄	範例
EXPT(*基數*、*指數*)	基本：REAL, LREAL 指數：INT, DINT, LINT, UINT, UDINT, ULINT	REAL, LREAL	指數：$基礎^{指數}$	a: = EXPT (b,c) (*以變數 *b* 做為基數而變數 *c* 做為指數的指數函數儲存在變數 *a* 中*)

備註 針對一般指數函數所回歸的資料類型會與引數中所使用的相同。因此，代入函數回歸值的變數必須與引數的資料類型相同。

資料類型轉換函數

下列資料類型轉換函數可以用在結構化文字中。

語法

*來源資料類型*_TO_*新資料類型(變數名稱)*

範例：REAL_TO_INT (C)

在這個範例中，變數 *C* 的資料類型將會從 REAL 變更為 INT。

資料類型組合

可以轉換的資料類型組合如下表所示。
(YES=可轉換，No=不可轉換)

從	至											
	BOOL	**INT**	**DINT**	**LINT**	**UINT**	**UDINT**	**ULINT**	**WORD**	**DWORD**	**LWORD**	**REAL**	**LREAL**
BOOL	否	否	否	否	否	否	否	否	否	否	否	否
INT	否	否	是	是	是	是	是	是	是	是	是	是
DINT	否	是	否	是	是	是	是	是	是	是	是	是
LINT	否	是	是	否	是	是	是	是	是	是	是	是
UINT	否	是	是	是	否	是	是	是	是	是	是	是
UDINT	否	是	是	是	是	否	是	是	是	是	是	是
ULINT	否	是	是	是	是	是	否	是	是	是	是	是
WORD	否	是	是	是	是	是	是	否	是	是	否	否
DWORD	否	是	是	是	是	是	是	是	否	是	否	否
LWORD	否	是	是	是	是	是	是	是	是	否	否	否
REAL	否	是	是	是	是	是	是	否	否	否	否	是
LREAL	否	是	是	是	是	是	是	否	否	否	是	否

敘述細節

指定

總結

敘述的左側(變數)以敘述的右側(方程式、變數、或常數)代入。

保留字組

:=

冒號(:)與等號(=)的組合。

敘述語法

變數：=方程式、*變數*、*或常數*；

使用狀況

使用指定敘述將值輸入變數中。這是一個用於控制敘述之前或之中的基本敘述。這個敘述可以用來設定起始值、儲存計算結果、及增大或減小變數。

說明

代入(儲存)*變數*的一個*方程式*、*變數*、或*常數*。

範例

範例 1：以方程式 X+1 的結果代入變數 A。

```
A:=X+1;
```

範例 2：以變數 B 代入變數 A。

```
A:=B;
```

範例 3：以常數 10 代入變數 A。

```
A:=10;
```

注意事項
要指定的方程式、變數、或常數的資料類型必須與要代入的變數的資料類型相同。否則，會發生語法錯誤。

控制敘述

IF 敘述(單一條件)

總結
這個敘述用於在符合指定的條件時執行一個表示式。如果條件不符，則會執行不同的表示式。

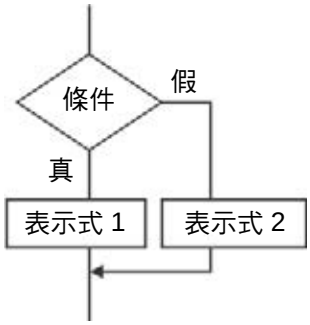
保留字組
IF, THEN, (ELSE), END_IF

備註 ELSE 可以省略。

敘述語法

```
IF <condition> THEN
   <expression_1>;
ELSE
   <expression_2>;
END_IF;
```

處理流程圖



使用狀況
根據是否符合一個單一條件(條件方程式)使用 IF 敘述來執行一個不同的運算。

說明
條件 = 如果為真，則執行*表示式_1*

條件 = 如果為假，則執行*表示式_2*

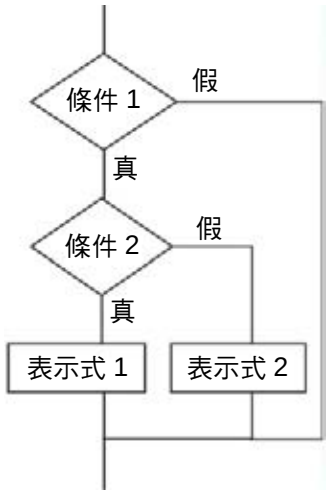
注意事項

- IF 必須與 END_IF 一起使用。
- *條件*必須包括一個用來評估結果的真或假方程式。
  範例：IF(A>10)
  *條件*也可以只指定為一個不是方程式的 Boolean 變數。因此，變數值將會是 1 (ON) = 真的結果、0 (OFF) = 假的結果。
- 可以用在*表示式_1* 及*表示式_2* 中的敘述是指定敘述、IF、CASE、FOR、WHILE、或 REPEAT。

範例程式：

```
IF <condition_1> THEN
 IF <condition_2> THEN
  <expression_1>;
 ELSE
  <expression_2>;
  END_IF;
END_IF;
```

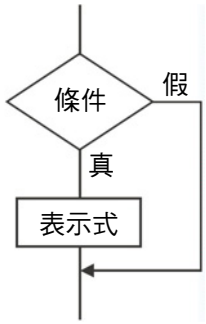
處理流程圖如下：



ELSE 相當於就在它之前的 THEN，如上圖所示。

- 可以在*表示式_1* 及*表示式_2* 中執行多個敘述。請確定在一個*表示式*中的多個敘述之間使用一個分號(;)的分界符號。
- ELSE 敘述可以省略。當省略 ELSE 時，如果*條件*方程式的結果為假，則不會執行運算。

處理流程圖



範例

範例 1：如果變數 A>0 為真，變數 X 將會代入數值 10。如果 A>0 為假，變數 X 將會代入數值 0。

```
IF A>0 THEN
  X:=10;
ELSE
  X:=0;
END_IF;
```

範例 2：如果變數 A>0 及變數 B>1 皆為真，變數 X 將會代入數值 10，且變數 Y 將會代入數值 20。如果變數 A>0 及變數 B>1 皆為假，則變數 X 及變數 Y 都會代入數值 0。

```
IF A>0 AND B>1 THEN
  X:=10; Y:=20;
ELSE
  X:=0; Y:=0;
END_IF;
```

範例 3：如果 Boolean(BOOL 資料類型)變數 A=1(ON)，變數 X 將會代入數值 10。如果變數 A=0(OFF)，變數 X 將會代入數值 0。

```
IF A THEN X:=10;
ELSE X:=0;
END_IF;
```

IF 敘述(多個條件)

總結

這個敘述用於在符合指定的條件時執行一個表示式。如果不符合第一個條件，但符合另一個條件，則會執行一個相應的表示式。如果條件都不符合，則會執行不同的表示式。

保留字組

IF, THEN, ELSIF, (ELSE)

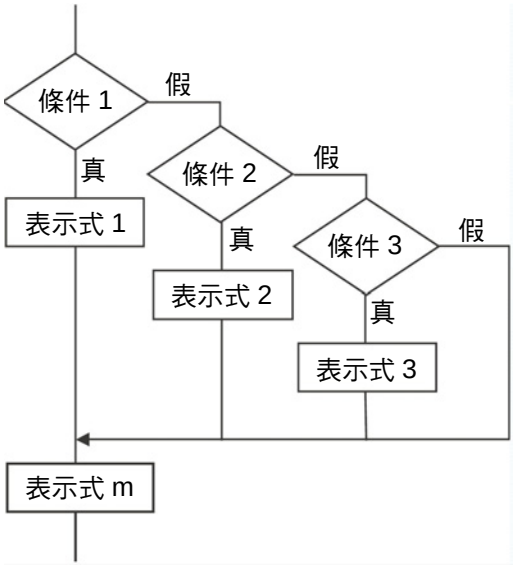
備註 ELSE 可以省略。

敘述語法

```
IF <condition_1> THEN <expression_1>;
     ELSIF <condition_2> THEN <expression_2>;
     ELSIF <condition_3> THEN <expression_3>;

           ...
     ELSIF <condition_n> THEN <expression_n>;
ELSE <expression_m>;
END_IF;
```

處理流程圖



使用狀況

根據多個條件(*條件*方程式)中符合哪個條件，使用 IF 敘述來執行不同的運算。

說明

條件 1 = 如果為真，則執行*表示式 1*

條件 *1* = 如果為假，

條件 2 = 如果為真，則執行*表示式 2*

條件 2 = 如果為假，

條件 3 = 如果為真，則執行*表示式 3*

等

條件 n = 如果為真，則執行*表示式 n*

如果這些條件都沒有一個符合，則會執行*條件 m*。

注意事項

- IF 必須與 END_IF 一起使用。
- *條件*_□包含有方程式(例如 IF(A>10))的真或假的結果。
  也可以只指定一個 Boolean (BOOL 資料類型)變數做為不是方程式的*條件*。
  在 Boolean 條件方面，當變數值是 1 (ON)時結果為真，當它是 0 (OFF)時結果為假。
- 可以用在*表示式*_□中的敘述是指定敘述、IF、CASE、FOR、WHILE、或 REPEAT。
- 可以在*表示式*_□中執行多個敘述。請確定在一個*表示式*中的多個敘述之間使用一個分號(;)的分界符號。
- ELSE 敘述可以省略。當省略 ELSE 時，如果*條件*方程式的結果為假，則不會執行運算。

範例

範例 1：如果變數 A>0 為真，變數 X 將會代入數值 10。
如果 A>0 為假，但變數 B=1，則變數 X 將會代入數值 1。
如果 A>0 為假，但變數 B=2，則變數 X 將會代入數值 2。
如果這些條件中有任一個符合，變數 X 將會代入數值 0。

```
IF A>0 THEN X:=10;
          ELSIF B=1 THEN X:=1;
          ELSIF B=2 THEN X:=2;
ELSE X:=0;
END_IF;
```

CASE 敘述

總結

這個敘述會執行一個包含有與來自一個整數方程式的值相符的一個選定的整數的表示式。如果選定的整數值不同，則不會執行表示式或會執行一個指定的表示式。

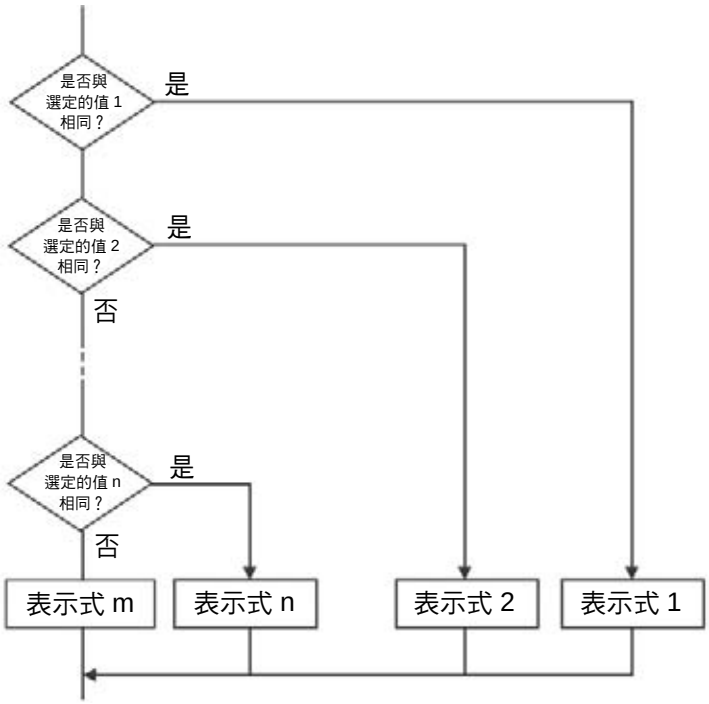
保留字組

CASE

敘述語法

```
CASE <integer_equation> OF
     <integer_equation_value_1>:<expression_1>;
     <integer_equation_value_2>:<expression_2>;
                    . . .
     <integer_equation_value_n>:<expression_n>;
ELSE <expression_m>;
END_CASE;
```

處理流程圖



使用狀況
使用 CASE 敘述來根據指定的整數值執行不同的運算。

說明
如果*整數_方程式*符合*整數_方程式_值_n*，則會執行*表示式_n*。
如果*整數_方程式*與任一個*整數_方程式_值_n* 都不符，則會執行*表示式_m*。

注意事項
- CASE 必須與 END_CASE 一起使用。
- *整數_方程式*的結果必須為整數格式(INT、DINT、LINT、UINT、UDINT、或 ULINT)。
- 可以用在*表示式_*□中的敘述是指定敘述、IF、CASE、FOR、WHILE、或 REPEAT。
- 可以在*表示式_*□中執行多個敘述。請確定在一個*表示式*中的多個敘述之間使用一個分號(;)的分界符號。
- *整數_方程式*中可以指定整數格式的變數(INT、DINT、LINT、UINT、UDINT、或 ULINT)，或者會回送整數值的方程式。
- 當 OR 邏輯用於*整數_方程式_值_n* 中的多個整數時，請使用一個逗號的分界符號來隔開數值。要指定一個整數數列時，請使用兩個句號(..)做為第一和最後一個整數之間的分界符號。

範例
範例 1：如果變數 A 是 1，變數 X 會代入數值 1。如果變數 A 是 2，變數 X 會代入數值 2。如果變數 A 是 3，變數 X 會代入數值 3。如果這些情況有任一個符合，變數 Y 將會代入 0。

```
CASE A OF
        1:X:=1;
        2:X:=2;
        3:X:=3;
ELSE Y:=0;
END_CASE;
```

範例 2：如果變數 A 是 1，變數 X 會代入數值 1。如果變數 A 是 2 或 5，變數 X 會代入數值 2。如果變數 A 是 6 與 10 之間的一個值，變數 X 會代入數值 3。如果變數 A 是 11、12 或 15 或 20，變數 X 會代入數值 4。如果這些情況有任一個符合，變數 Y 將會代入 0。

```
CASE A OF
1:X:=1;
2,5:X:=2;
6..10:X:=3;
11,12,15..20:X:=4;
ELSE Y:=0;
END_CASE;
```

FOR 敘述

總結

這個敘述會用來重複執行一個指定的表示式值到一個變數(在這裡稱為反覆變數)達到一個指定的值為止。

保留字組

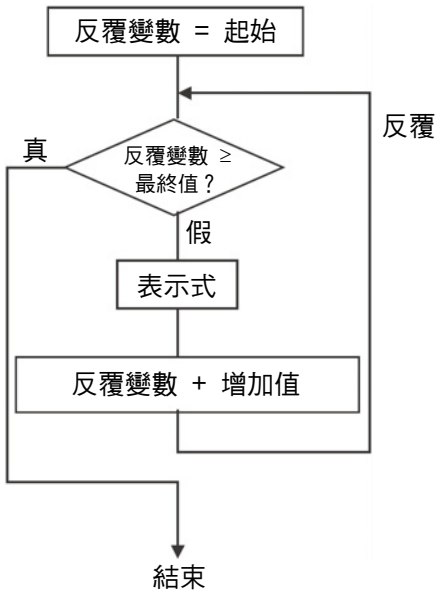
FOR, TO, (BY), DO, END_FOR

備註 BY 可以省略。

敘述語法

```
FOR <iteration_variable>:= <initial_value> TO <final_value_equation> BY
<increment_value_equation>
DO
  <expression>;
END_FOR;
```

處理流程圖



使用狀況

當反覆次數已事先決定時請使用 FOR 敘述。FOR 在根據一個指定的反覆變數的值來變換一個陣列變數中的元件數時特別好用。

說明

當*反覆_變數*是*起始_值*時，會執行*表示式*。在執行後，從*增量_方程式*取得的值會加到*反覆_變數*中，且如果*反覆_變數* < *最終_值_方程式*(請參閱備註 1)，則會執行*表示式*。在執行後，從*增量_方程式*取得的值會加到*反覆_變數*中，且如果*反覆_變數* < *最終_值_方程式*(請參閱備註 1)，則會執行*表示式*。這個處理會一直重複。如果*反覆_變數* ≥ *最終_值_方程式*(請參閱備註 2)，則處理會結束。

備註 (1) 如果取自*增量_方程式*的值為負數，條件為*反覆_變數* > *最終_值_方程式*的值。

(2) 如果取自*增量_方程式*的值為負數，條件為*反覆_變數* ≤ *最終_值_方程式*。

注意事項

- *增量_方程式*可以指定一個負數值
- FOR 必須結合 END_FOR 一起使用。
- *起始_值*、最終_*值_方程式*及*最終_值_方程式*必須是一個整數的資料類型(INT、DINT、LINT、UINT、UDINT、或 ULINT)。
- 在以最終值執行處理後，反覆值會增加到最終值+1 上並結束反覆處理。
  範例：在下列結構化文字中，“a”的值變為 TRUE。

```
FOR i:=0 TO 100 DO
  array[i]:=0;
END_FOR;


IF i=101 THEN
   a:=TRUE;
ELSE
   a:=FALSE;
END_IF;
```

- 在反覆變數直接變更的不要使用 FOR 敘述。這麼做會導致非預期的操作。
  範例程式：

```
FOR i:=0 TO 100 BY 1 DO
  array[i]:=0;
  i:=i+5;
END_FOR;
```

- 可以用在*表示式*中的敘述是指定敘述、IF、CASE、FOR、WHILE、或 REPEAT。
- 可以在*表示式*中執行多個敘述。請確定在一個*表示式*中的多個敘述之間使用一個分號(;)的分界符號。
- BY *增量_方程式*可以省略。若省略時，BY 會被視為 1。
- *起始_值*、*最終_值_方程式*、及*增量_方程式*中可以指定整數資料類型的變數(INT、DINT、LINT、UINT、UDINT、或 ULINT)，或者會回送整數值的方程式。
  範例 1：反覆會在反覆變數 n = 0 到 50 (增量為 5)時執行，且陣列變數 SP[n]會代入 100。

```
FOR n:=0     TO 50  BY 5 DO
  SP[n]:=100;
END_FOR;
```

範例 2：會計算陣列變數 DATA[n]的元件 DATA[1]到 DATA[50]的總計值，並且代入變數 SUM 中。

```
FOR n:=0      TO 50  BY 1 DO
  SUM:=SUM+DATA[n];
END_FOR;
```

範例 3:會偵測陣列變數 DATA[n]的元件從 DATA[1]到 DATA[50]的最大值及最小值。最大值會代入變數 MAX 中而最小值會代入變數 MIN 中。DATA[n]的值介於 0 與 1000 之間。

```
MAX:=0;
MIN:=1000;
FOR n:=1      TO 50  BY 1 DO
  IF DATA[n]>MAX THEN
  MAX:=DATA[n];
  END IF;
  IF DATA[n]<MIN THEN
  MIN:=DATA[n];
  END IF;
END_FOR;
```

WHILE 敘述

總結

這個敘述用來重複執行一個指定的表示式，只要指定的條件為真。

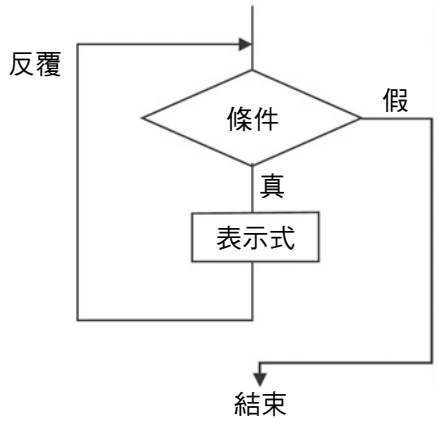
保留字組

WHILE, DO, END_WHILE

敘述語法

```
WHILE <condition> DO
 <expression>;
END_WHILE;
```

處理流程圖



使用狀況

在反覆次數沒有事先決定(根據符合的條件)時請使用 WHILE 敘述在符合條件的期間重複指定的處理。這個敘述可以用來在只有條件方程式為真時執行處理(測試前循環)。

說明

在執行*表示式*之前，會評估*條件*。

如果*條件*為真，則會執行*表示式*。之後，會再次評估*條件*。這個處理會一直重複。如果*條件*為假，則不會執行*表示式*且結束評估*條件*。

注意事項

- WHILE 必須結合 END_WHILE 一起使用。
- 在執行*表示式*之前，如果*條件*方程式為假，則會結束處理而不會執行*表示式*。
- 可以用在*表示式*中的敘述是指定敘述、IF、CASE、FOR、WHILE、或 REPEAT。
- 可以在*表示式*中執行多個敘述。請確定在一個*表示式*中的多個敘述之間使用一個分號(;)的分界符號。
- *條件*也可以只指定為一個不是方程式的 Boolean 變數(BOOL 資料類型)。

範例

範例 1：會計算超過 1000、增量為 7 的值必代入變數 A 中。

```
A:=0;
WHILE A>=1000 DO
 A:=A+7;
END_WHILE;
```

範例 2：當 X<3000 時，X 的值會乘以 2，並且將值代入陣列變數 DATA[1]中。將著再次將 X 的值乘以 2，並將值代入陣列變數 DATA[2]中。這個處理會一直重複。

```
n:=1'
WHILE X<3000 DO
 X:=X*2;
 DATA[n]:=X;
 n:=n+1;
END_WHIE;
```

REPEAT 敘述

總結

這個敘述用於重複執行一個表示式直到指定的條件為真時為止。

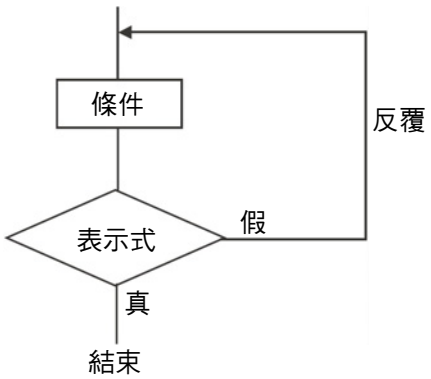
保留字組

REPEAT, UNTIL, END_REPEAT

敘述語法

```
REPEAT
 <expression>;
UNTIL <condition>
END_REPEAT
```

處理流程圖



使用狀況
當反覆次數沒有事先決定(根據是否符合條件)時，請使用 REPEAT 敘述在指定的處理後只要符合一個條件即重複處理。這個敘述可以用來根據指定處理的執行結果來決定是否進行重複處理(測試後循環)。

說明
*表示式*會先不根據條件執行一次。之後，它會評估*條件*方程式。如果*條件*為假，則*表示式*會再次執行。如果*條件*為真，則處理會結束而不會執行*表示式*。

注意事項
- REPEAT 必須與 END_REPEAT 一起使用。
- 在*表示式*執行之前即使*條件*方程式為"真"，也會執行*表示式*。
- 可以用在*表示式*中的敘述是指定敘述、IF、CASE、FOR、WHILE、或 REPEAT。
- 可以在*表示式*中執行多個敘述。請確定在一個*表示式*中的多個敘述之間使用一個分號(;)的分界符號。
- *條件*也可以只指定為一個不是方程式的 Boolean 變數(BOOL 資料類型)。

範例
範例 1：從 1 到 10 的數值會遞增並將總值代入變數 TOTAL 中。

```
A:=1;
TOTAL:=0;
REPEAT
  TOTAL:=TOTAL+A;
  A:=A+1;
UNTIL A>10
END_REPEAT;
```

EXIT 敘述

總結
這個敘述只用在反覆敘述(FOR、WHILE、REPEAT)中來強制結束一個反覆敘述。這個敘述也可以用在一個 IF 敘述中在符合一個指定的條件時強制結束一個反覆敘述。

保留字組
EXIT

敘述語法(例如：使用在 IF 敘述中)

```
FOR (WHILE, REPEAT) expression
                                ...
IF <condition> THEN EXIT;
END_IF;

                                ...
END_FOR (WHILE, REPEAT);
```

使用狀況
使用 EXIT 敘述可在符合結束條件之前強制結束反覆處理。

說明(例如：使用在 IF 敘述中)
當*條件*方程式為真時，反覆敘述(FOR、WHILE、REPEAT)會被強制結束，並且在 EXIT 之後的任何敘述都不會執行。

備註 (1) *條件*也可以只指定為一個不是方程式的 Boolean 變數(BOOL 資料類型)。

(2) 在*表示式*執行之前即使*條件*方程式為真，也會執行*表示式*。

範例
處理會從變數 n = 1 時一直重複直到 50 為止(增量為 1)且 n 會加到陣列變數 DATA[n]中。不過，如果 DATA[n]超過 100，則會結束處理。

```
FOR n:=1; TO 50 BY 1 DO
  DATA[n]:=DATA[n]+n;
  IF DATA[n]>100 THEN EXIT;
  END_IF;
END_FOR;
```

RETURN 敘述

總結
這個敘述用來在結構化文字中的功能區塊必須在完成之前強制結束時執行緊隨在程式中叫出功能區塊的位置之後的下一個指令。

保留字組
RETURN

敘述語法

```
RETURN;
```

使用狀況
在一個功能區塊已經被強制結束時請使用 RETURN 敘述。

功能區塊呼叫敘述

總結
這個敘述可用來叫出另一個功能區塊定義。

保留字組
無

敘述語法
在指定實例名稱(請參閱備註)之後，請在括弧中指定引數(要轉送被呼叫功能區塊的輸入變數的呼叫功能區塊輸入變數)及回歸值(要由呼叫功能區塊的輸出變數接收的被呼叫功能區塊輸出變數)。

備註 實例名稱可以是具有*功能區塊*資料類型的任何內部變數名稱。

可使用下列兩個方法中的任一個來輸入功能呼叫敘述。

1. 指定方法 A：指定被呼叫與呼叫功能區塊的變數名稱

   實例名稱(被呼叫功能區塊定義的輸入變數名稱 := 呼叫功能區塊定義的變數名稱或常數, …, 被呼叫功能區塊定義的輸出變數名稱或常數 => 呼叫功能區塊定義的輸出變數名稱, …)；

   備註 所有輸入變數的指定(被呼叫功能區塊定義的輸入變數名稱 := 呼叫功能區塊定義的變數名稱或常數)必須以逗號隔開。只有所需的輸出變數指定(被呼叫功能區塊定義的輸出變數名稱或常數 => 呼叫功能區塊定義的輸出變數名稱)必須以逗號隔開。

2. 指定方法 B：只指定呼叫功能區塊的變數名稱(或常數)並省略被呼叫功能區塊的變數名稱

   實例名稱(呼叫功能區塊定義的變數名稱或常數, …, 呼叫功能區塊定義的輸出變數名稱, …)；

備註 當被呼叫功能區塊定義的輸入及輸出變數名稱如以上所示被省略時，會依變數登錄在變數表中的順序自動將呼叫功能區塊的輸入變數(或常數)的值轉送給被呼叫功能區塊的輸入變數。透過相同的方式，被呼叫功能區塊的輸出變數也會依它們登錄在變數表中的順序自動回送給呼叫功能區塊的輸出變數。

使用狀況

使用功能區塊呼叫敘述來從一個 ST 語言程式叫出一個功能區塊定義(ST 或階梯語言)。

說明

1. 下列實例會登錄在變數表的內部變數中。

內部變數元件	目錄	範例
名稱	任何實例名稱	資料型態
功能區塊	功能區塊	功能區塊
FB 定義	選取被呼叫功能區塊定義。	計算

2. 將在變數之間轉送的值會在登錄的實例名稱之後的括弧內指定(在這個範例中是 Calcu_execute)並且以一個分號來標示敘述的結尾，如以下範例所示。

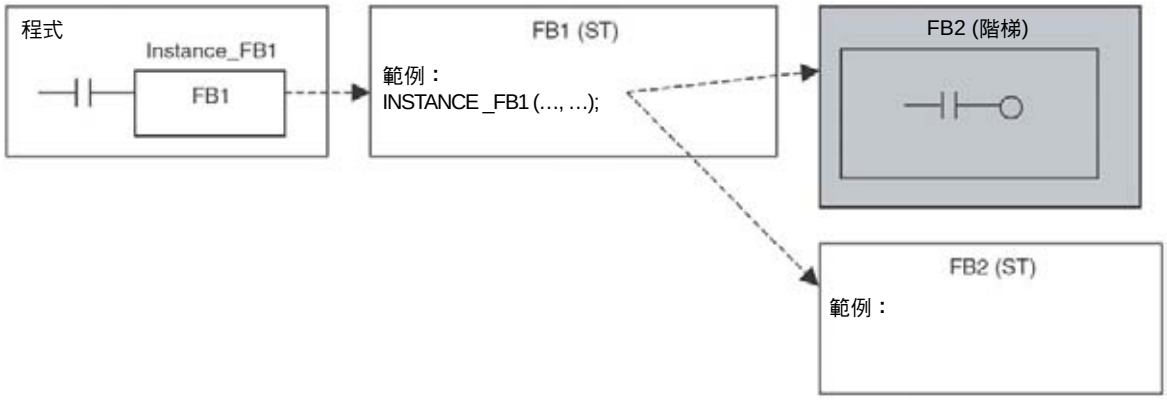
   Calcu_execute (A:=B,C=>D) ;

   輸入變數 B (在呼叫功能區塊中)的值會轉送給輸入變數 A (在被呼叫功能區塊中)。同樣的，輸出變數 C (在被呼叫功能區塊中)的值會回送給輸出變數 D(在呼叫功能區塊中)。

範例

在下列範例中，一個功能區塊被從功能區塊 1 (FB1)叫出。

- 功能區塊 1 是以 ST 語言編寫的。
- 功能區塊 2 可能以階梯或 ST 語言編寫。



INSTANCE _FB1 是一個資料類型為 FUNCTION BLOCK 的實例名稱。

- 下表顯示功能區塊 1 中的變數以及功能區塊 2 中用來接收/轉送資料的對應變數。

變數類別	變數名稱	轉送給 FB1/從 FB2 轉送
輸入變數	FB1_IN1	轉送給 FB2_IN1
	FB1_IN2	轉送給 FB2_IN2
	FB1_IN3	轉送給 FB2_IN3

變數類別	變數名稱	轉送給 FB1/從 FB2 轉送
輸出變數	FB1_OUT1	從 FB2_OUT1 接收
	FB1_OUT2	從 FB2_OUT2 接收
	FB1_OUT3	從 FB2_OUT3 接收
內部變數	A 備註：資料類型 = Bool	轉送給 EN
	B 備註：資料類型 = Bool	從 ENO 接收
內部變數(實例)	Instance_FB2 備註：資料類型 = 功能區塊	被呼叫功能區塊定義： 功能區塊 2

- 下表顯示功能區塊 2 中的變數以及功能區塊 1 中用來接收/轉送資料的對應變數。

變數類別	變數名稱	轉送給 FB1/從 FB2 轉送
輸入變數	FB2_IN1	從 FB1_IN1 接收
	FB2_IN2	從 FB1_IN2 接收
	FB2_IN3	從 FB1_IN3 接收
輸出變數	FB2_OUT1	轉送給 FB1_OUT1
	FB2_OUT2	轉送給 FB1_OUT2
	FB2_OUT3	轉送給 FB1_OUT3

範例 1：**指定方法 A (同時只定 FB1 及 FB2 變數)**

Instance_FB2 (EN:=A,FB2_IN1:=FB1_IN1,FB2_IN2:=FB1_IN2,FB2_IN3:=FB1_IN3,FB2_OUT1=> FB1_ OUT1,FB2_OUT2=>FB1_OUT2,FB2_OUT3=>FB1_OUT3,ENO=>B)

- 它適用於要以不規則順序列出的引數及回歸值。
- 輸入變數的引數必須在清單的開頭處，或者如果列出 EN 變數的話，必須緊接在 EN 變數之後。
- 指定方法 B 不能與指定方法 A 一起使用在同一個功能區塊呼叫敘述中。

其他敘述變化：

- 不使用 EN 的敘述

  Instance_FB2(FB2_IN1:=FB1_IN1,FB2_IN2:=FB1_IN2,FB2_IN3:=FB1_IN3,FB2_OUT1=>FB1_OUT1, FB2_OUT2=>FB1_OUT2,FB2_OUT3=>FB1_OUT3,ENO=>B)

- 不使用 EN 和 ENO 的敘述

  Instance_FB2(FB2_IN1:=FB1_IN1,FB2_IN2:=FB1_IN2,FB2_IN3:=FB1_IN3,FB2_OUT1=>FB1_OUT1, FB2_OUT2=>FB1_OUT2,FB2_OUT3=>FB1_OUT3)

- 不使用 ENO 的敘述

  Instance_FB2(EN:=A,FB2_IN1:=FB1_IN1,FB2_IN2:=FB1_IN2,FB2_IN3:=FB1_IN3,FB2_OUT1=>FB1 _ OUT1,FB2_OUT2=>FB1_OUT2,FB2_OUT3=>FB1_OUT3)

- 不使用 FB2_OUT2 (不需要 FB2_OUT2 資料)的敘述。

  Instance_FB2(EN:=A,FB2_IN1:=FB1_IN1,FB2_IN2:=FB1_IN2,FB2_IN3:=FB1_IN3,FB2_OUT1=>FB1 _ OUT1,FB2_OUT3=>FB1_OUT3,ENO=>B)

- 不使用 FB2_OUT2 (不需要 FB2_OUT2 資料)的敘述。

  Instance_FB2(EN:=A,FB2_IN1:=FB1_IN1,FB2_IN2:=FB1_IN2,FB2_IN3:=FB1_IN3,FB2_OUT1=>FB1 _ OUT1,FB2_OUT3=>FB1_OUT3,ENO=>B)

- 採用不規則順序的敘述

  Instance_FB2(EN:=A,FB2_IN1:=FB1_IN1,FB2_OUT1=>FB1_OUT1,FB2_IN2:=FB1_IN2,FB2_OUT2 => FB1_OUT2,FB2_IN3:=FB1_IN3,FB2_OUT3=>FB1_OUT3,ENO=>B)

範例 2：指定方法 B(只指定 FB1 變數)

Instance_FB2(FB1_IN1,FB1_IN2,FB1_IN3,FB1_OUT1,FB1_OUT2,FB1_OUT3)

Instance_FB2(FB1_IN1,FB1_IN2,FB1_IN3,FB1_OUT1)

- 引數及回歸值必須以固定順序列出。

  輸入變數 1, 輸入變數 2, …, 輸出變數 1, 輸出變數 2, …

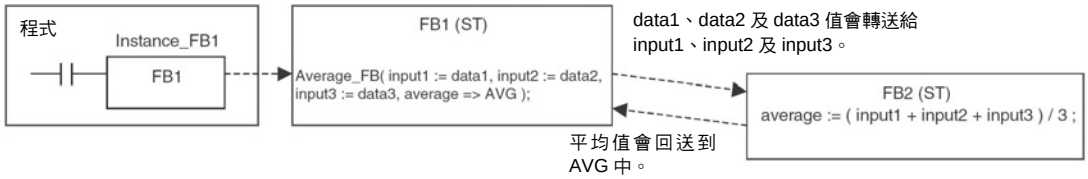
- 輸入變數的引數必須在清單的開頭處，或者如果列出 EN 變數的話，必須緊接在 EN 變數之後。
- 一個輸出變數如果資料沒有被實際用到且輸出變數不是在輸出變數清單的中間，則可以省略。

  範例：顯示_FB2 (FB1_IN1, FB1_IN2, FB1_IN3, FB1_OUT1, B1_OUT3)
  在這種情況下，在清單末尾的 FB1_OUT3 會從 FB2_OUT2 回送它的值

- EN 和 ENO 資料不能輸入做為一個引數或回歸值。
- 指定方法 A 不能與指定方法 B 一起使用在同一個功能區塊呼叫敘述中。

範例 3：平均值計算功能區塊

在下列範例中，功能區塊 1 會呼叫負責計算平均值的功能區塊 2。



Average_FB 是一個資料類型為 FUNCTION BLOCK 的實例名稱。

功能區塊 1

- 變數表

變數類別	變數名稱	資料型態	轉送給 FB1/從 FB2 轉送
輸入變數	EN	BOOL	---
輸入變數	data1	INT	轉送給 input1
輸入變數	data2	INT	轉送給 input2
輸入變數	data3	INT	轉送給 input3
輸入變數	bCheck	BOOL	---
輸出變數	ENO	BOOL	---
輸出變數	AVG	INT	從 average 接收
內部變數	Average_FB	功能區塊 被呼叫功能區塊定義：功能區塊 2	---

- ST 語言演算法則

  如果 bCheck 為真，功能區塊 2 會被叫出來計算平均值。data1、data2 及 data3 的 3 個值會分別轉送給功能區塊 2 的輸入變數 input1、input2 及 input3。計算的結果*(平均值)*會回送給 AVG。

  備註 下圖顯示 Average_FB 功能區塊以指定方法 A 被叫出(列出兩個功能區塊的變數)。

```
IF bCheck = TRUE THEN
   Average(input1:=data1,input2:=data2,input3:=data3,average=>AVG);
ELSE
   RETURN;
END_IF;
```

功能區塊 2

- 變數表

變數類別	變數名稱	資料型態	轉送給 FB1/從 FB2 轉送
輸入變數	EN	BOOL	---
輸入變數	input1	INT	從 data1 接收
輸入變數	input2	INT	從 data2 接收
輸入變數	input3	INT	從 data3 接收
輸出變數	ENO	BOOL	---
輸出變數	average	INT	轉送給 AVG

- ST 語言演算法則

  計算 input1、input2 和 input3 的平均值並將結果儲存到 *average* 中。

```
average:=(input1+input2+input3)/3;
```

結構化文字程式編寫範例

範例 1：將 BCD 資料(#0000-#9999)轉換為 BIN 資料

(*檢查輸入參數“Input_BCD”(BCD 資料)*)

```
IF (Input_BCD>=0 & Input_BCD<=16#9999) THEN
   ENO:=true;
ELSE
   ENO:=false;
   RETURN;
END_IF;
```

(*BCD 資料除以 16 四次來獲得從 BCD 資料轉換的 BIN 資料的每個位數*)

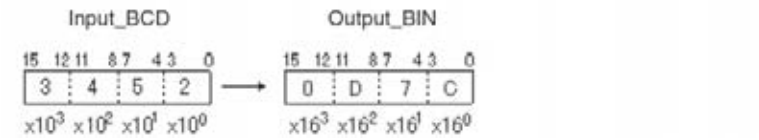
```
DIV_1:=Input_BCD/16;
DIV_2:=DIV_1/16;
DIV_3:=DIV_2/16;
DIV_4:=DIV_3/16;
```

(*計算從 BCD 資料轉換的 BIN 資料的每個位數*)

```
BIN_1:=Input_BCD-16*DIV_1;  (*一個 16^0 位數的數字*)
BIN_2:=DIV_1-16*DIV_2;  (*一個 16^1 位數的數字*)
BIN_3:=DIV_2-16*DIV_3;  (*一個 16^2 位數的數字*)
BIN_4:=DIV_3-16*DIV_4;  (*一個 16^3 位數的數字*)
```

(*計算 BIN 資料“Output_BIN”(輸出參數)*)

```
Output_BIN:=BIN_1+BIN_2*10+BIN_3*10*10+BIN_4*10*10*10;
```



範例 2：將 BIN 資料(#0000-#9999)轉換為 BCD 資料

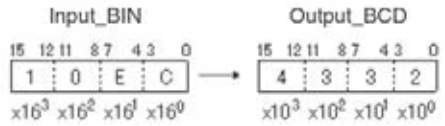
(*檢查輸入參數“Input_BIN”(BIN 資料)*)

```
IF (Input_BIN>=0 & Input_BIN<=16#FFFF) THEN
   ENO:=true;
ELSE
   ENO:=false;
   RETURN;
END_IF;
```

(*BIN 資料除以 10 四次來獲得從 BIN 資料轉換的 BCD 資料的每個位數*)

```
DIV_1:=Input_BIN/10;
DIV_2:=DIV_1/10;
DIV_3:=DIV_2/10;
DIV_4:=DIV_3/10;
```

```
(*計算從 BIN 資料轉換的 BCD 資料的每個位數*)
BCD_1:=Input_BIN-10*DIV_1; (*一個 10^0 位數的數字*)
BCD_2:=DIV_1-10*DIV_2; (*一個 10^1 位數的數字*)
BCD_3:=DIV_2-10*DIV_3; (*一個 10^2 位數的數字*)
BCD_4:=DIV_3-10*DIV_4; (*一個 10^3 位數的數字*)
(*計算 BCD 資料“Output_BCD”(輸出參數)*)
Output_BCD:=BCD_1+BCD_2*16+BCD_3*16*16+BCD_4*16*16*16;
```



限制

套疊

可以用在 IF、CASE、FOR、WHILE 或 REPEAT 敘述中的套疊數沒有限制。

資料類型限制

- 資料類型為 WORD、DWORD、INT、DINT、UINT、UDINT 或 ULINT 的變數只能指定為整數。例如，如果 A 是一個 INT 資料類型，則可以是 A:=1; 。如果值不是一個整數資料類型，則會發生語法錯誤。例如，如果 A 是一個 INT 資料類型，則 A:=2.5;會發生語法錯誤。
- 如果是一個實數(浮點十進位資料)，則只能分配給資料類型為 REAL 及 UREAL 的變數。例如，如果 A 是一個 REAL 資料類型，則可以是 A:=1.5;。如果沒有使用一個變數，則會發生語法錯誤。如，如果 A 是一個 REAL 資料類型，則 A:=2;會發生語法錯誤。使用 A:=2.0;。
- 位元(TRUE、FALSE)只能分配給 BOOL 資料類型的變數。例如，如果 A 是一個 BOOL 資料類型，則可以是 A:=FALSE;。如果沒有使用一個 BOOL 資料類型，則會發生語法錯誤。例如，如果 A 是一個 REAL 資料類型，則 A:=2;會發生語法錯誤。例如，如果 A 是一個 INT 資料類型，則 A:=FALSE;會發生語法錯誤。
- 結構化文字中的資料類型必須完全一致。例如，如果 A、B、及 C 是 INT 資料類型，則可以是 A:=B+C;。不過，如果 A 和 B 是 INT 資料類型，但 C 是一個 REAL 資料類型或 LINT 資料類型，則 A:=B+C;會發生語法錯誤。

結構化文字錯誤

錯誤訊息

錯誤訊息	錯誤原因	範例
%s'的輸入變數不能指定一個值	對一個輸入變數代入一個值	
%s'資料類型不支援%s 的運算數	使用了一個運算數不支援的資料類型的數值或變數	A:=B+1;(*A 和 B 是 WORD 類型的變數*)
%s'的變數有一個唯讀記憶體 AT 位址，不能指定一個值	對一個分配給唯讀記憶體位址(唯讀附屬區域位址或條件旗標)的變數代入一個值	
陣列指標超出範圍	指定了一個大於陣列大小的陣列指標	Array[100]：=10; (**Array* 是一個陣列大小為 100 的陣列變數*)

錯誤訊息	錯誤原因	範例
轉換不能從%s 轉換為%s	一個數值方程式中的運算結果的資料類型與代入目的地的變數不符以及代入與資料類型不同的變數	Y:=ABS(X); (*X 是一個 INT 類型的變數，Y 是一個 UINT 類型的變數*)
除以零	數值表示式包含除以 0	
找不到註解的結尾	註解沒有一個與註解的開始括弧和星號“*)”對應的完結括弧和星號“(*”	(**註解*
無效的文字格式'%s'	數值格式不合規定	X:=123_; (*底線之後沒有數字*) X:=1__23; (*底線後緊隨有另一個底線*) X:=2#301; Y:=8#90; (*使用了一個不能用於二進位或八進位值的數值*) **備註** 底線可以插入數字之間以方便檢讀。請在數值開頭處加上 2#、8#、及 16#來分別表示數值為二進位、八進位、及十六進位的值
無效的文字值	變數格式不合規定。	X:=1e2; (*一個指標用於不是 REAL 資料類型的數值中*) **備註** “e”表示 10 的指數。
無效的陣列指標	一個數值方程式有非整數類型的運算結果或陣列指標中指定一個非整數的變數	Array[Index]：=10; (**Index* 是一個 WORD 類型的變數*)
無效的常數	一個數值方程式有一個非整數類型的運算結果，或一個 CASE 敘述的整數方程式中指定一個非整數的變數或數字值	CASE A OF (*A 是一個 REAL 類型的變數*) 1: X:=1; 2: X:=2; END_CASE;
無效的表示式	數值方程式不合規定。例如，整數方程式或條件方程式不合規定或沒有在語法(IF、WHILE、REPEAT、FOR、CASE)中指定	WHILE DO (*WHILE 敘述沒有包含一個條件方程式*) X:=X+1; END_CASE;
FOR 循環宣告中有無效的參數	一個 FOR 敘述中的變數使用一個資料類型為 INT、DINT、LINT、UINT、UDINT、或 ULINT 以外的變數	FOR I:=1 TO 100 DO (*I 是一個 WORD 類型的變數*) X:=X+1; END_FOR;
無效的敘述	敘述不合規定。例如，(IF、WHILE、REPEAT、FOR、CASE、REPEAT)敘述沒有在語法中分別包含 IF、WHILE、REPEAT、FOR、CASE、或 REPEAT	X:=X+1; (*語法中沒有 REPEAT*) UNTIL X>10 ENG_REPEAT;

錯誤訊息	錯誤原因	範例
無效的功能輸出變數	針對功能輸出指定的變數不合規定(指定一個非 Boolean (BOOL) 的變數或數值做為 ENO 的傳送目的地)	Y:=SIN(X1, ENO=>1);
遺漏(	一個資料格式轉換指令或功能的呼叫沒有包含一個“(”(開始括弧)	Y:=INT_TO_DINT X);
遺漏)	運算數括弧或資料格式轉換指令或功能的呼叫沒有包含一個對應於“(”(開始括弧)的“)”(結尾括弧)	Y:=(X1+X2/2
遺漏:	CASE 敘述中的整數方程式沒有跟隨一個“:”(冒號)	CASE A OF 1 X:=1; END_CASE;
遺漏:=	指定方程式中沒有包括“:=”	
遺漏;	敘述沒有以一個“;”(分號)結尾	
遺漏 DO	FOR 或 WHILE 敘述中沒有加上“DO"	
遺漏 END_CASE	CASE 敘述的末尾沒有加上“END_CASE”	
遺漏 END_FOR	FOR 敘述的末尾沒有加上“END_FOR”。	
遺漏 END_IF	IF 敘述的末尾沒有加上“END_IF”	
遺漏 END_REPEAT	REPEAT 敘述的末尾沒有加上“END_REPEAT”	
遺漏 END_WHILE	WHILE 敘述的末尾沒有加上“END_WHILE”	
遺漏輸入參數，必須設定所有的輸入變數	沒有指定函數引數或不完整	Y:=EXPT(X);
遺漏 OF	CASE 敘述中沒有包括“OF”	
遺漏 THEN	IF 敘述中沒有包括“THEN”	
遺漏 TO	FOR 敘述中沒有包括“TO”	
遺漏 UNTIL	REPEAT 敘述中沒有包括“UNTIL”	
遺漏[	沒有指定陣列變數的陣列指標	X:=Array; (*_Array_ 是一個陣列變數*)
遺漏]	沒有指定陣列變數的陣列指標	X:=Array[2; (*_Array_ 是一個陣列變數*)

錯誤訊息	錯誤原因	範例
漏失常數	CASE 敘述的整數方程式中沒有提供一個常數	CASE A OF 2..; X:=1; 2..; X:=2; END_CASE;
NOT 運算沒有文字數字支持	NOT 運算數用於一個數值	Result:=NOT 1;
%s 資料類型不支援負值	資料類型不支援負數值的變數(UINT、UDINT、ULINT)前使用一個負號	Y:=-X; (*X 是一個 UINT 類型的變數，Y 是一個 INT 類型的變數*)
必須有一行有效碼(不包括註解)	沒有一行有效碼(不包括註解)	
針對函數指定有太多變數	針對函數指定有太多參數設定	Y:=SIN(X1,X2);
未定義的識別碼'%s'	使用一個沒有在變數表中定義的變數	
非預期的語法'%s'	一個關鍵字(保留字組)或變數的使用不合規定。	FOR I:=1 TO 100 DO BY -1 (*DO 的位置不合規定*) X;=X+1; END_FOR;
函數變數中的用法不符	函數參數的使用不合規定	Y:=SIN(X1,EN=>BOOL1); (*輸入參數 EN 被用來做為一個輸出參數*)
值超出範圍	一個值超出代入變數中的變數資料類型的範圍	X:=32768; (*X 是一個 INT 類型的變數*)
變數'%s'不是一個函數參數	參數中指定有一個不能在函數參數中指定的變數	Y:=SIN(Z:=X); (*X 和 Y 是 REAL 類型的變數，而 Z 不是一個 SIN 的函數參數*)

警告訊息

警告訊息	警告原因	範例
關鍵字'%s'為冗餘	關鍵字被用在無效的位置。例如，使用超出一個循環語法的 EXIT 敘述	
從'%s'轉換為'%s'，可能會流失資料	資料可能會因為從一個資料大小較大的資料類型轉換為一個資料大小較小的資料類型而流失	Y:=DINT_TO_INT(X); (*X 是一個 DINT 類型的變數，Y 是一個 INT 類型的變數*)

常見的問題

Q：如何表示一個十六進位值？

A：在數值前加上“16#”，例如 16#123F。

也可以加上 8# 和 2# 的字首來分別表示八進位和二進位數。沒有這些字首的數都會被視為十進位數。

Q：FOR 可以使用多少次？

A：在下列範例中，FOR 敘述的內容會執行 101 次。循環處理會在“i”的值等於 101 時結束。

```
FOR i:=0 TO 100 BY 1 DO
  a:=a+1;
END_FOR;
```

Q：超過陣列下標時會發生什麼狀況？

A：在有 10 個元件的陣列變數 INT[10]方面，下列敘述類別將不會偵測到錯誤。如果超過這個敘述則運算會不可靠。

```
i:=15;
INT[i]:=10;
```

Q：結構化文字編輯器中的變數是否會自動登錄到變數表中？

A：否。請在使用變數前將它們登錄到變數表中。

Q：階梯程式編寫指令能否直接叫出？

A：否。

附錄 C
外部變數

分類	名稱	CX-Programmer 中的外部變數	資料型態	位址
條件旗標	大於或等於(GE)旗標	P_GE	BOOL	CF00
	不等於(NE)旗標	P_NE	BOOL	CF001
	小於或等於(LE)旗標	P_LE	BOOL	CF002
	指令執行錯誤(ER)旗標	P_ER	BOOL	CF003
	進位(CY)旗標	P_CY	BOOL	CF004
	大於(GT)旗標	P_GT	BOOL	CF005
	等於(EQ)旗標	P_EQ	BOOL	CF006
	小於(LT)旗標	P_LT	BOOL	CF007
	負值(N)旗標	P_N	BOOL	CF008
	溢位(OF)旗標	P_OF	BOOL	CF009
	下溢(UF)旗標	P_UF	BOOL	CF010
	存取錯誤旗標	P_AER	BOOL	CF011
	永遠 OFF 旗標	P_Off	BOOL	CF114
	永遠 ON 旗標	P_On	BOOL	CF113
時鐘暫停	0.02 秒時序脈衝位元	P_0_02s	BOOL	CF103
	0.1 秒時序脈衝位元	P_0_1s	BOOL	CF100
	0.2 秒時序脈衝位元	P_0_2s	BOOL	CF101
	1 分鐘時序脈衝位元	P_1mim	BOOL	CF104
	1 秒鐘時序脈衝位元	P_1s	BOOL	CF102
附屬區域旗標/位元	第一循環旗標	P_First_Cycle	BOOL	A200.11
	步驟旗標	P_Step	BOOL	A200.12
	第一工作執行旗標	P_First_Cycle_Task	BOOL	A200.15
	最大循環時間	P_Max_Cycle_Time	UDINT	A262
	現行掃描時間	P_Cycle_Time_Value	UDINT	A264
	循環時間錯誤旗標	P_Cycle_Time_Error	BOOL	A401.08
	低電量旗標	P_Low_Battery	BOOL	A402.04
	I/O 驗證錯誤旗標	P_IO_Verify_Error	BOOL	A402.09
	輸出 OFF 位元	P_Output_Off_Bit	BOOL	A500.15
OMRON FB 資料庫字組(請參閱備註)	CIO 區域指定	P_CIO	WORD	A450
	HR 區域指定	P_HR	WORD	A452
	WR 區域指定	P_WR	WORD	A451
	DM 區域指定	P_DM	WORD	A460
	EM0 至 C 區域指定	P_EM0 至 P_EMC	WORD	A461 至 A473

備註 這些字組是 OMRON FB 資料庫的外部變數。請不要使用這些字組來建立功能區塊。

各位OMRON產品愛用者

選購時的注意事項

首先感謝您平時對OMRON產品的支持與愛護。
各位根據型錄購買本公司控制器產品(以下稱為「本公司產品」時，敬請確認以下內容。

1. 保固內容：
保固期間
本公司的產品保固期間為購買產品後抑或是將產品交貨至指定地點後一年內。
保固範圍
於上述的保固期間內，若產品因非人為因素而發生故障，本公司將於原購買地點提供免費的代替品更換與維修等服務。
但下列故障原因不在保固範圍內：
a) 不在本目錄或規格書內所規定之條件、環境的使用下所造成的故障
b) 非產品本身原因所造成的故障
c) 非經由本公司所進行的改裝或維修所造成的故障
d) 未依照原本設計之使用方式所造成的故障
e) 出貨時之科技水準所無法預測之原因所造成的故障
f) 其它天災、災害等不可抗力所造成的故障
此外，上述保固僅限於本公司產品本身，因產品故障所導致之相關損失並不包含在本保固範圍內。

2. 責任限制
關於因本公司產品所引發之一切特別損害、間接損害、消極損害(應得利益之喪失)，本公司不負任何責任。
關於本公司之可程式化產品，針對非經本公司之技術人員所執行之程式或因其所造成之結果，本公司不負任何責任。

3. 選購時，應符合用途條件
將本公司商品與其他搭配使用時，請確認是否符合顧客所需之規格、法規或限制等。
此外，請顧客自行確認目前所使用的系統、機械或是裝置是否適用於本公司商品。
再者，請顧客自行確認本公司商品是否符合目前所使用的系統、機械或是裝置。
如未確認是否符合或適用時，本公司無須對本公司商品的適用性負責。
使用於以下用途時，敬請於洽詢本公司業務人員後根據規格書等進行確認，同時注意安全措施，例如使用的額定電壓、性能要盡量低於限制範圍以策安全；或是採用在發生故障時可將危險程度降至最小的安全回路等。
a) 用於戶外、會遭受潛在化學污染、電力會遭受妨礙的用途、或是在本型錄未記載的條件或環境下使用。
b) 核能控制設備、焚燒設備、鐵路、航空、車輛設備、醫用機器、娛樂用途機械設備、安全裝置以及遵照政府機構或個別業界規定的設備。
c) 危及生命或財產的系統、機械、裝置。
d) 瓦斯、水/供電系統，或是系統穩定性有特殊要求的設備。
e) 其他符合a)~d)、需要有高度安全性的用途。
當顧客將本公司商品使用於可能嚴重危害生命、財產等用途時，敬請務必事先確認系統整體有危險告示、 並採用備援設計等可確保安全性，以及本公司產品針對整體設備的特定用途上的配電與設置適當。
由於本型錄所記載的應用程式範例屬於參考性質，如需直接採用時，使用前請先確認機械、裝置的功能與安全性。
敬請顧客務必以正確的方法來使用本公司產品，並了解使用時的禁止事項與注意事項，以免不當的使用而造成他人意外的損失。

4. 規格變更
本型錄所記載的規格以及附屬品，可能會在必要時、進行改良時或其他事由而變更。敬請洽詢本公司或特約店之營業人員，以確認本公司商品的實際規格。

台灣歐姆龍股份有限公司

http://www.omron.com.tw

OMRON 產品技術客服中心

OMRON 鈴鈴鈴 支援我

0800-000-705

國際電話，行動電話請改撥付費電話：(02)8768-2568

【產業自動化】
產品技術諮詢服務
・服務時間・
週一 ~ 週五
9:00 ~12:00 / 13:00 ~18:00
・FAX諮詢專線・
(02) 8768-3705
・E-mail諮詢・
www.omron.com.tw

■台北營業所：台北市復興北路363號6樓(弘雅大樓)
電話：02-2715-3331 傳真：02-2712-6712

■桃園營業所：桃園縣蘆竹鄉南崁路一段83號11F-5
電話：03-212-0677 傳真：03-212-0003

■台中營業所：台中市中港路一段345號27樓之3(中港高峰大樓)
電話：04-2325-0834 傳真：04-2325-0734

■台南營業所：台南市大同路二段615號17樓
電話：06-290-3797 傳真：06-290-3796

特約店

註：規格可能改變，恕不另行通知，最終以產品說明書為準。

Cat. No. W447-TW5-01

2006-05-1000 Printed in Taiwan