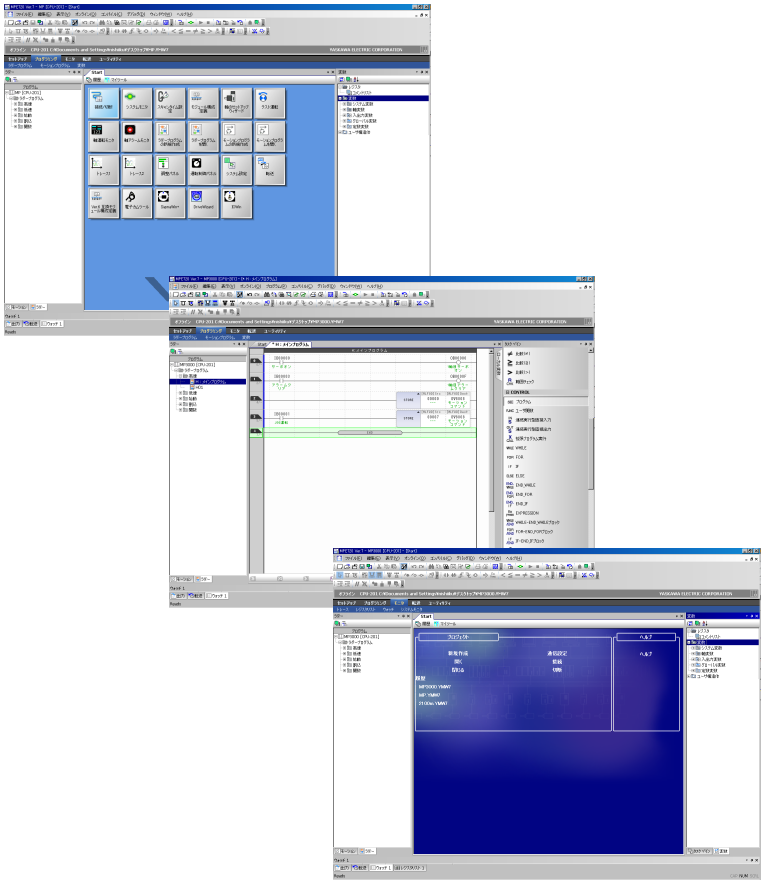
YASKAWA

運動控制器　MP3000系列

階梯圖程式
程式編寫手冊



資料編號　YTWMNCO-15004A

前言

本手冊將針對 MP3000 系列運動控制器的階梯圖程式等相關資訊進行說明。

使用前請詳閱本手冊之說明，以期能正確使用本運動控制器系統，並善用系統功能以作為控制生產線系統之用。

此外，請妥善保管本手冊，以利在需要時能隨時參閱。

本手冊的使用方式

◆ 簡稱及代號

本手冊使用下述簡稱及代號。

簡稱及代號	代表意義
運動控制器	MP3000 系列運動控制器
MPE720	可程式裝置專用軟體或已安裝該軟體之可程式化裝置 (PC)
PLC	可程式控制器
MP3200	電源元件、CPU 單元、基本元件、擴充槽 I/F 元件等的總稱
MP3300	CPU 模組、基本元件等的總稱

◆ 本手冊中所使用的開發工具

本手冊引用 MPE720 Ver. 7 的畫面作為說明之用。

◆ 反轉訊號名稱標記

凡本手冊文章中所出現的反轉訊號名稱 (L 值有效訊號) 皆會在訊號名稱的最前面附加反斜線 (/) 的標記。

標記範例

- $\overline{\text{S-ON}}$ = /S-ON
- $\overline{\text{P-CON}}$ = /P-CON

◆ 關於專有名詞 「轉矩」

旋轉型伺服馬達通常使用 「轉矩」，而線性伺服馬達則使用 「推力」，本手冊則統一採用 「轉矩」一詞來標示 (參數名稱除外)。

◆ 註冊商標等

- MECHATROLINK 係 MECHATROLINK 協會之註冊商標。
- Ethernet 係 Xerox 公司之註冊商標。
- 其他本手冊中所刊載之產品名稱、公司名稱等專用名稱分別為該公司之商標、註冊或商品名稱。本手冊並未在各公司的註冊商標或商標上標註 TM、® 標誌。

◆ 圖示符號

本手冊採用下列圖示符號，並標示在文章中重要的位置，目的在於明確地區別所說明的內容。


重要

包含務必遵守的注意事項或限制事項等。
代表雖然會發出警報，但還不至於造成裝置損壞程度之注意事項。


註記

包含注意或是避免操作錯誤之註記事項。

範例　包含操作或設定範例等。

補充　包含補充說明或是有用的輔助資訊等。


專有名詞解說

說明範圍包含不容易瞭解的專有名詞或是文章中所出現未預先說明的名詞。

相關使用手冊

表示下表相關之使用手冊，請依實際需要作為參考之用。

請在確實瞭解產品規格及使用限制條件的前提下，使用本產品。

分類	資料名稱	資料編號	內容
基本功能	運動控制器 MP2000/MP3000 系列 運動控制器系統 設定手冊	SIJP C880725 00	說明 MP2000/MP3000 系列運動控制器安裝 / 連接、設定、試運轉、程式編寫 / 除錯及各項功能。
	運動控制器 MP3000 系列 MP3200/MP3300 故障排除手冊	SIJP C880725 01	說明 MP3000 系列運動控制器如何除錯。
	運動控制器 MP3000 系列 MP3200 使用手冊	SIJP C880725 10	說明 MP3000 系列 MP3200 規格、系統架構及 CPU 單元功能。
	運動控制器 MP3000 系列 MP3300 產品手冊	YTWMNCO-14008A	說明 MP3000 系列 MP3300 的規格、系統架構及 CPU 模組功能。
通訊功能	運動控制器 MP3000 系列 通訊功能 使用手冊	YTWMNCO-15003A	包含 MP3000 系列乙太網路通訊規格、系統架構及通訊連線方法等相關內容。
運動控制功能	運動控制器 MP3000 系列 運動控制功能 使用手冊	YTWMNCO-14013A	包含 MP3000 系列運動控制功能 (SVC32/SVR32) 之規格、系統架構及使用方法等相關內容。
程式	運動控制器 MP3000 系列 運動程式 程式編寫手冊	YTWMNCO-15005A	包含 MP3000 系列運動程式及序列程式之規格及指令等相關內容。
	運動控制器 MP2000 系列 使用手冊 階梯圖程式篇	SI-C887-1.2	包含 MP2000 系列階梯圖程式適用之演算指令等相關內容。
開發工具	運動控制器 MP2000/MP3000 系列 系統整合開發工具 MPE720 Ver. 7 使用手冊	SIJP C880761 03	說明 MPE720 Ver. 7 的操作方法。

安全注意事項

本手冊採用下列標誌，提醒使用者注意使用安全。

攸關安全的標誌上所刊載的文字係為重要內容，請務必嚴格遵守之。

⚠ 警告 操作錯誤時，恐將引發危險狀況，甚至有可能導致死亡或重傷等意外發生。

⚠ 注意 一旦操作錯誤時，恐將引發危險狀況，甚至有可能導致中度傷害或輕傷等意外發生，或者有可能僅造成物品損壞。

此外，即使是已刊載於 ⚠注意 上之事項，在某些狀況下亦有可能因此導致嚴重後果。

禁止 表示禁止 (嚴格禁止) 事項。例如：以 標誌表示嚴禁煙火。

強制 表示強制 (絕對必要) 事項。例如：以 標誌表示必須接地。

以下所示為存放、搬運、安裝、配線、運轉、維護、檢查及報廢時務必遵守之重要注意事項。

◆ 一般

⚠ 警告

- **本產品應由專業技術人員妥善進行設置。**
  否則恐將造成觸電或人身傷害等意外發生。
- **連接裝置運轉時，應保持隨時可以緊急停止之狀態。**
  否則恐將造成人身傷害等意外發生。
- **運轉時因瞬間停電而重置時，裝置有可能突然重新啟動，此時請勿靠近裝置。重新啟動時，請採取安全防護措施，以維護人身安全。**
  否則恐將造成人身傷害等意外發生。
- **嚴禁碰觸產品內部。**
  否則恐將造成觸電意外。
- **請勿在通電狀態下卸除前方護蓋、纜線、接頭及配件等。**
  否則恐將造成觸電、故障或產品損壞等意外發生。
- **請勿刮傷纜線，強力拉扯、不當施力、在纜線上放置重物或是擠壓纜線。**
  否則恐將造成觸電或產品停止動作，甚至損毀等意外發生。
- **嚴禁改造本產品。**
  否則恐將造成裝置的損壞。

◆ 存放 / 搬運

⚠ 注意

- **本產品必須存放於下述環境。**
  - ・ 不會受到陽光直射之場所
  - ・ 環境溫度未超過所規定的存放溫度條件之場所
  - ・ 相對濕度未超過所規定的存放濕度條件之場所
  - ・ 溫度不會急遽變化或結露之場所
  - ・ 不含腐蝕性或可燃性氣體之場所
  - ・ 粉塵、灰塵、鹽分或金屬粉末含量較低之場所
  - ・ 不會碰觸水、油或藥品等之場所
  - ・ 產品不會受到震動或撞擊力影響之場所

  否則恐將造成火災、觸電或裝置損壞等意外發生。
- **搬運時，請搬妥產品的主體部分。**
  搬運產品時，若僅搬運纜線或接頭部位，可能會造成接頭損壞或纜線斷線，並成人身傷害等意外發生。
- **請勿重覆堆疊本產品 (請依照標示)。**
  否則恐將造成產品的故障。
- **運送時，請勿讓本產品暴露於含有鹵素 (氟素、氯、溴或碘) 等氣體的環境。**
  可能會造成產品的故障或損壞。
- **如需針對包裝用的木質材料 (木框、合板或棧板等) 進行消毒或除蟲時，請務必採用煙燻以外的方式。**
  例：熱處理 (中心溫度大於 56°C，持續 30 分鐘以上)
  此外，請在包裝前的材料階段進行處理，而非包裝完成後再整體處理。
  若使用經過煙燻處裡的木質材料包裝電子產品 (單機或是裝載在機器等的產品上)，材料所散發出來的氣體或蒸氣，恐將嚴重損壞電子零件。尤其像是鹵素類消毒劑 (氟素、氯、溴或碘等) 可能會造成電容器內部的腐蝕。

◆ 安裝

⚠ 注意

- **請勿將本產品設置於下列環境中。**
  - 不會受到陽光直射之場所
  - 環境溫度不會超過所規定的設置溫度條件之場所
  - 相對濕度不會超過所規定的存放濕度條件之場所
  - 溫度不會急遽變化或結露之場所
  - 不含腐蝕性或可燃性氣體之場所
  - 粉塵、灰塵、鹽分或金屬粉末含量較低之場所
  - 不會碰觸水、油或藥品等之場所
  - 產品不會受到震動或撞擊力影響之場所

  否則恐將造成火災、觸電或裝置損壞等意外發生。
- **設置時，請勿讓本產品暴露於含有鹵素 (氟素、氯、溴或碘) 等氣體的環境。**
  可能會造成產品的故障或損壞。
- **請勿坐在本產品上，或是在產品上放置重物。**
  否則恐將造成產品的故障。
- **請勿覆蓋住進氣及排氣孔，此外，也請勿讓異物進入產品內部。**
  否則恐將造成內部元件品質劣化，甚至造成火災或產品故障。
- **請務必遵守本手冊所規定之安裝方向。**
  否則恐將造成產品故障。
- **設置產品、控制面板內面及其他裝置時，必須依照規定的間隔來設置。**
  否則恐將造成火災或產品故障。
- **請勿強力撞擊本產品。**
  否則恐將造成產品故障。
- **安裝電池時，應由專業的技術人員妥善進行安裝作業。**
  否則恐將造成觸電、人身傷害或裝置損壞等意外發生。
- **請勿碰觸電池的電極部分。**
  否則靜電恐將造成產品損壞。

◆ 配線

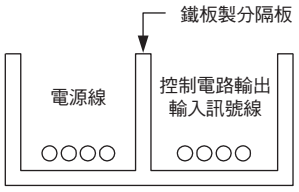
⚠ 注意

- **配線作業需正確且確實實施。**
  否則恐將造成馬達爆震，或是人身傷害、產品故障等意外發生。
- **使用時，請依指定的電源電壓正確使用。**
  否則恐將造成火災或產品故障。
- **在電源狀態不佳的場所使用本產品時，請於輸入功率在規定的電壓變動範圍內可供電的狀態下使用。**
  否則恐將造成裝置損壞。
- **請設置斷路器等安全裝置，以利外部配線短路時之用。**
  否則恐將引發火災等意外發生。
- **在以下場所使用本產品時，請分別採取適當的遮蔽措施。**
  - 因靜電而發生干擾之場所
  - 產生強力電場及磁場之場所
  - 有可能暴露於幅射源之場所
  - 電源線從附近經過之場所

  否則恐將造成裝置損壞。
- **相較於輸出輸入專用的 24 V 電源，本產品採用 CPU 單元 /CPU 模組必須先通電的電路架構。如需進一步瞭解電路相關資訊，請參閱以下的使用手冊。**
  MP3000 系列　CPU 單元　操作手冊 (資料編號：TOBP C880725 16)
  MP3000 系列 MP3300　CPU 模組　操作手冊 (資料編號：TOBP C880725 23)

  若先將輸出輸入專用的 24 V 電源等外部電源通電後，再對 CPU 單元 /CPU 模組通電，恐造成 CPU 單元 / CPU 模組的輸出瞬間啟動，並因無法預期的動作，造成人身傷害或裝置損壞等意外發生。
- **請利用產品外部的控制電路來架構攸關安全防護的緊急停止電路、互鎖電路及限制電路等。**
  否則恐將造成裝置的損壞。
- **使用 MECHATROLINK 輸出輸入模組時，必須先建立 MECHATROLINK 通訊，以作為互鎖輸出的條件。**
  否則恐將造成裝置損壞。
- **連接電池時，必須連接至正確的極性。**
  否則恐將造成電池損壞或爆炸。
- **選擇用來連接產品及外部裝置的輸出輸入訊號線 (外部配線) 時，必須考量以下事項。**
  - 機械強度
  - 干擾影響
  - 配線距離
  - 訊號電壓
- **如欲降低電源纜線所造成之干擾影響，控制面板內部和外部在進行控制電路的輸出輸入訊號纜線配線及設置時，均必須和電源線分隔。**
  若纜線未確實分隔時，可能會造成本產品的故障。

  配線分隔範例



◆ 運轉

⚠ 注意

- **請依照產品相對應的使用手冊上所刊載之步驟及指示來進行運轉及試運轉作業。**
  若在伺服馬達與機器連接狀態下發生操作錯誤，不但將造成機器損壞，有時甚至會導致人身傷害等意外發生。
- **請在產品外部加裝互鎖訊號等安全電路，如此即使在下述狀況發生時，仍能確保整個系統的安全性。**
  - 產品故障或因外部因素而造成異常發生時
  - 本產品藉由自動診斷功能檢測異常發生，停止運轉，並將輸出訊號 OFF(或是維持) 時
  - 因輸出繼電器熔接、燒毀或電晶體損壞，使得產品的輸出變成 ON 或 OFF 時
  - 本產品的 DC 24 V 輸出因過負載或短路狀態，以致電壓降低，而無法輸出訊號時
  - 因本產品的自動診斷功能無法檢測電源、輸出輸入部位或記憶體異常，而發生無預期輸出時

  以上狀況皆有可能導致人身傷害、裝置損壞或燒毀等意外。

◆ 維護 / 檢查

⚠ 注意

- **請勿自行拆解或維修本產品。**
  否則恐將造成觸電、人身傷害或裝置損壞等意外發生。
- **請勿在通電的狀態下更改配線。**
  否則恐將造成觸電、人身傷害或裝置損壞等意外發生。
- **更換電池時，應由專業的技術人員進行。**
  否則恐將造成觸電、人身傷害或裝置損壞等意外發生。
- **更換電池時，MP3□00 務必處於供電狀態。**
  在 MP3□00 供電中斷的狀態下交換電池時，儲存於 MP3□00 記憶體內的資料可能會被刪除。
- **更換電池時，請勿碰觸電極部位。**
  否則靜電恐將造成產品損壞。
- **更換 CPU 單元 /CPU 模組時，請勿遺漏下述作業。**
  - 為即將更換的 CPU 單元 /CPU 模組的程式及參數進行備份。
  - 將備份好的程式及參數傳送到新的 CPU 單元 /CPU 模組。

  未先傳送資料而運作 CPU 單元 /CPU 模組時，可能會因無法預期的動作而造成人身傷害或裝置損壞等意外發生。
- **通電過程中或電源關閉後請勿立刻碰觸 CPU 單元 /CPU 模組的散熱裝置。**
  若散熱裝置處於高溫狀態，恐將造成燙傷等意外發生。

◆ 廢棄

⚠ 注意

- **本產品應依照一般工業廢棄物標準進行處理。**
- **使用過的電池必須依照當地的規定進行處理。**

◆ 一般注意事項

使用時請注意。
● 本手冊上所刊載的插圖之目的在於說明，因此所顯示的圖面有可能為已卸除護蓋或安全防護品之狀態。操作本產品前，請務必依規定將護蓋或防護品恢復原狀，並依照使用手冊上之說明進行操控。 ● 本手冊上所刊載的插圖僅為代表性範例，與實際交貨的產品有可能會有部分差異。 ● 損壞或遺失本手冊時，如欲重新訂購，請聯絡本公司經銷商或本手冊封底所刊載本公司最近的業務單位，並告知欲訂購之資料編號。

關於保固

◆ 保固內容

■ 保固期間

本公司對客戶所購買的產品 (以下簡稱交貨產品) 自交貨至客戶指定的地點該日起提供 1 年的產品保固，或是自本公司工廠出貨後 18 個月，以先到期者為準。

■ 保固範圍

保固期限內一旦發生故障，且可歸咎於本公司之責時，本公司將免費提供替代品或故障品維修的服務。
如因交貨產品的使用壽命已到而造成產品故障或需更換消耗品、週期性零件時，則不在保固範圍內。

此外，倘若故障原因符合下列因素時，亦不在本公司保固範圍內。

- 未依照型錄、手冊或另行取得的規格書上所刊載的適當條件、環境或操作方式來使用本產品
- 非交貨產品所造成時
- 產品改造或維修並非出自本公司時
- 產品的使用方式不符規定時
- 依本公司出貨時之科技、技術水準所無法預知之事由時
- 其他如天災、災害等非可歸咎於本公司責任之原因時

◆ 免責事項

- 對於因交貨產品故障所造成之損害以及客戶在機會上的損失等，本公司一律不負相關責任。
- 由非本公司人員對可程式化產品執行程式 (包含各種參數設定) 所導致的任何結果，本公司一律不負相關責任。
- 型錄或手冊上所刊載的所有資訊係以客戶能依用途而選擇適合的產品為目的。不保證其使用上對本公司及第三人之智慧財產權或其他權利不會造成侵害或表示同意其實施。
- 因使用型錄或手冊上的資訊而發生侵害第三人智慧財產權或其他相關權利，本公司一律不負責任。

◆ 確認適用用途及條件

・如將本公司產品搭配其他產品使用時，請客戶先行確認適用規格、應遵守之法律或規定等。
・請客戶自行確認您所使用的系統、機器、裝置與本公司產品之間之適用性。
・本產品如欲作為下述用途，請先洽詢本公司後，再判斷是否可行。如判斷可行，建議採用額定規格、性能餘裕下使用，並採行安全措施，以便將故障時所發生的危險降至最低。
  ･在戶外或是受到潛在化學污染、電磁波干擾，或型錄、手冊上所未刊載的條件或環境下使用
  ･核能控制設備、燃燒設備、鐵路、航空、車輛設備、醫療裝置、娛樂設備，或是需要遵守行政機構或各業界規定之設備
  ･有危害生命或財產之虞的系統、機器或裝置
  ･瓦斯、自來水、電力供應系統或 24 小時連續運轉系統等需要高可靠性之系統
  ･其他符合上述條件且需要極高安全性之系統
・若將本產品用於對生命或財產有重大危害之虞的用途時，請利用危險警告標誌或冗餘設計方式，來確保必要之安全性，此外，也必須事先確認本公司產品的配電和設置是否適當。
・型錄或手冊上所刊載的電路實例或其他應用實例僅為參考之用，引用前，必須先確認您所要使用的機器、裝置的功能及安全性。
・請正確瞭解所有使用時之禁止事項及注意事項，並正確使用本公司的產品，避免造成第三人意外損害。

◆ 規格變更

本公司因改善產品或其他事由有權變更型錄或手冊上所刊載的產品品名、規格、外觀或附件，且不另行通知。變更後，本公司將更新型錄或手冊上的資料編號，並發行修訂版。當您詢問或訂購型錄上所刊載的產品時，請事先和本公司業務人員確認。

目　錄

階梯圖程式特色及概述

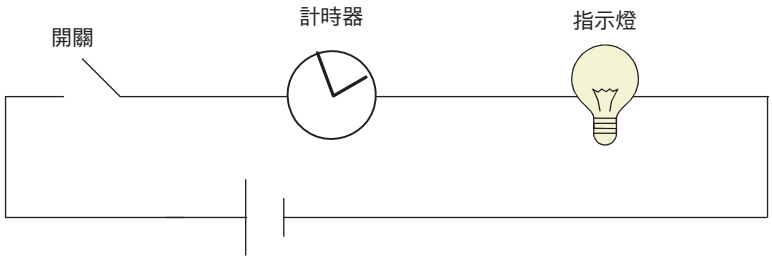
1

本章將針對階梯圖程式特色及其概要進行說明。

1.1 何謂「階梯圖程式」

所謂「階梯圖程式」就是利用各種階梯圖語言指令及暫存器，將開關、計時器、指示燈等所構成的電路加以圖形化。

<電路示意圖>

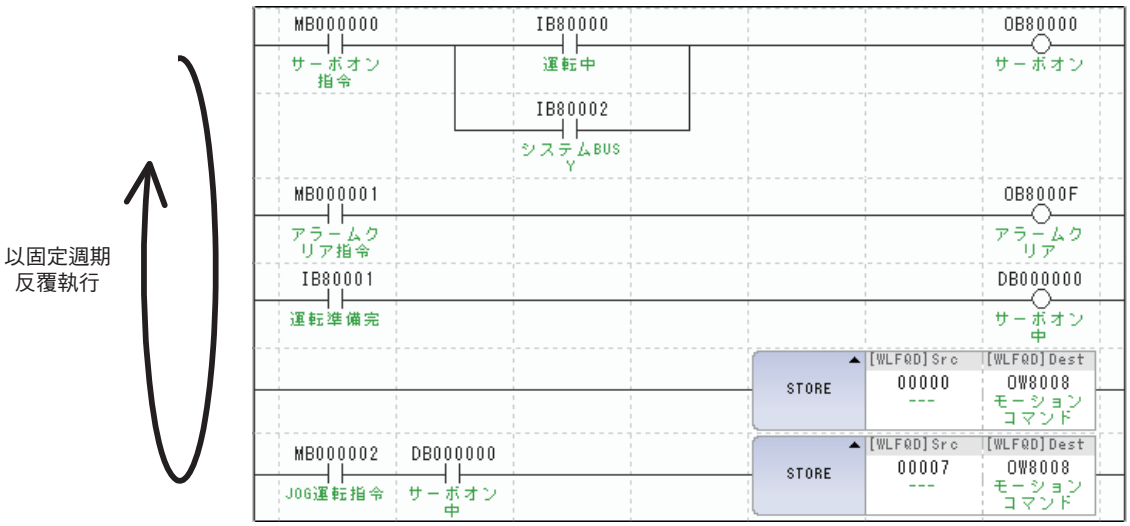


<階梯圖程式>

即使是複雜的電路或大型電路也能利用階梯圖程式輕鬆編寫出來。

階梯圖程式編寫完成後，會先進行一次掃描，接下來就會以固定週期的方式反覆執行。

<階梯圖程式範例>



1.2 特色

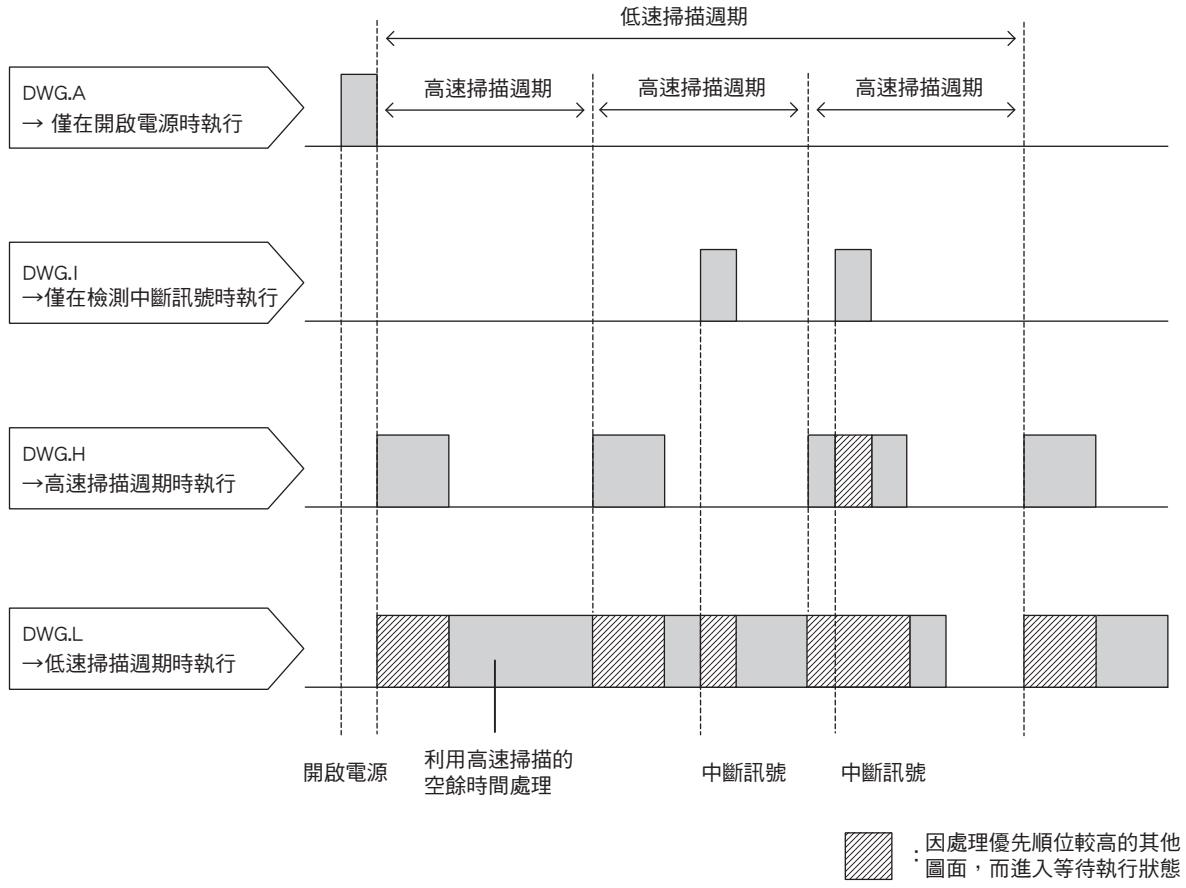
本節將說明階梯圖程式之特色。

各種動作時序之階梯圖圖面

階梯圖程式係以圖面 (DWG) 作為管理單位，因此，該圖面就稱之為「階梯圖圖面」。

如下圖所示，MP3000 系列依動作時序不同，階梯圖圖面亦各異。

使用者只要根據每個階梯圖面的執行時序來編寫處理程序，就會依不同的時序執行所需的處理作業。



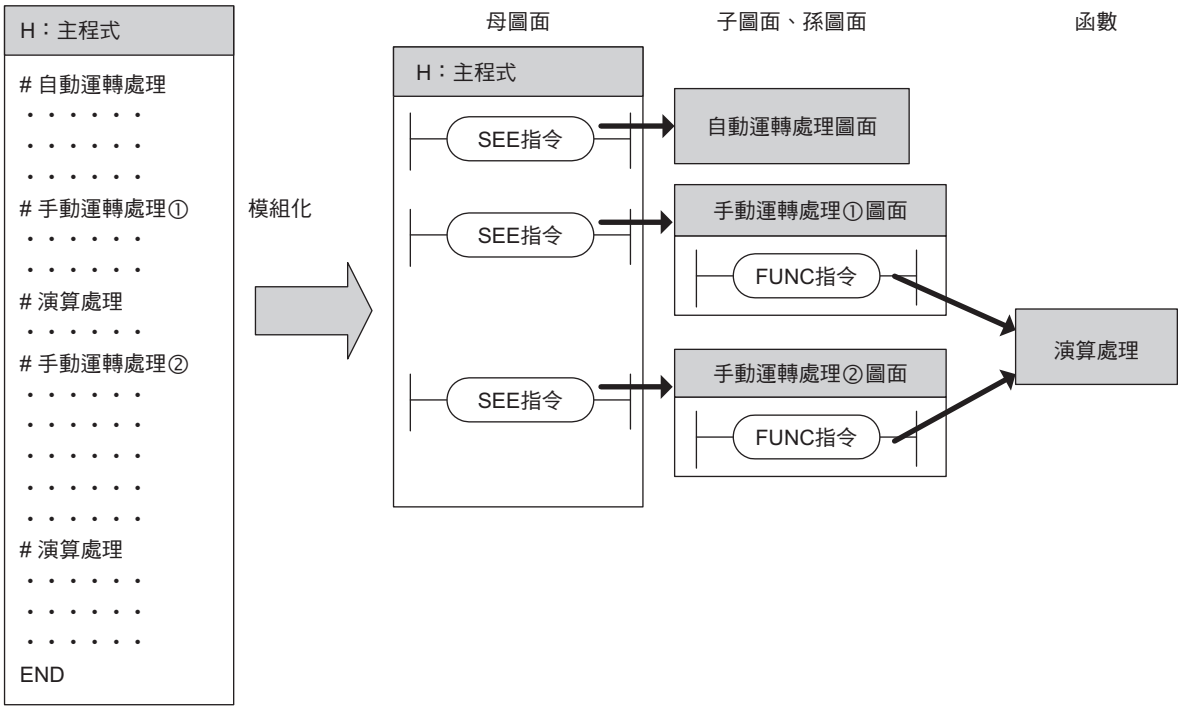
下表所示為每個圖面的執行時序。

階梯圖圖面	優先順位 *	執行時序 (處理範例)
DWG.A	1 (高)	僅在開啟電源時執行一次 (資料初始化)
DWG.I	2 (↑)	檢測中斷訊號時執行 (外部訊號中斷處理)
DWG.H	3 (↑)	高速掃描週期時執行 (運動控制)
DWG.L	4 (低)	低速掃描週期時執行 (觸控面板顯示處理)

* 數字愈小，優先順位愈高，也愈優先處理。

程式模組化

將主程式分為子圖面、孫圖面及函數等處理程序，再加以模組化，如此就能讓程式更一目瞭然。下圖為程式模組化之範例。

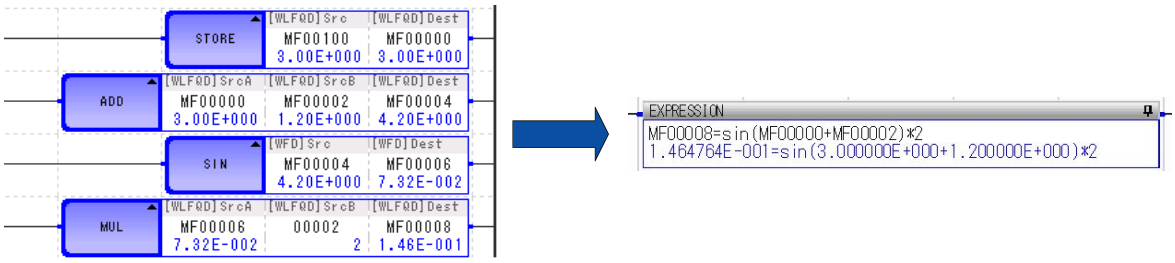


輕鬆編寫複雜的數值演算式

就連需要編寫多行的複雜算式，只要透過一個 EXPRESSION 指令，就能輕鬆編寫完成。

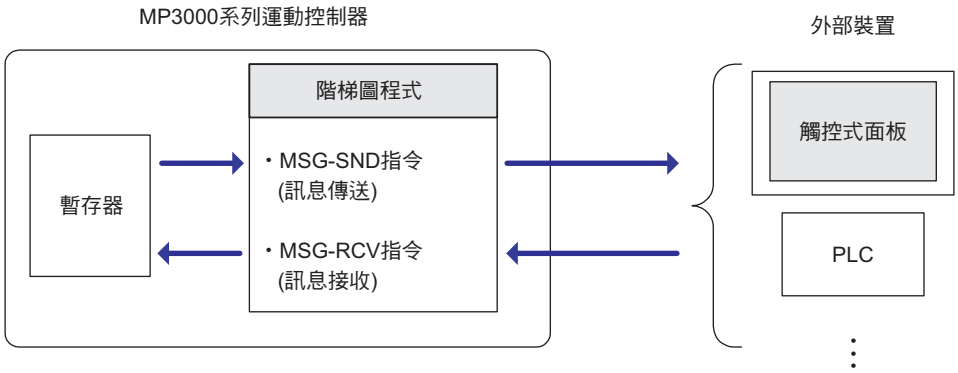
不但如此，編寫變數、構造體、sin、cos 等基本函數時，就像使用 C 語言來編寫一樣。

而且，此種指令和階梯圖語言的指令一樣，可顯示現在值。



對外部裝置進行通訊控制

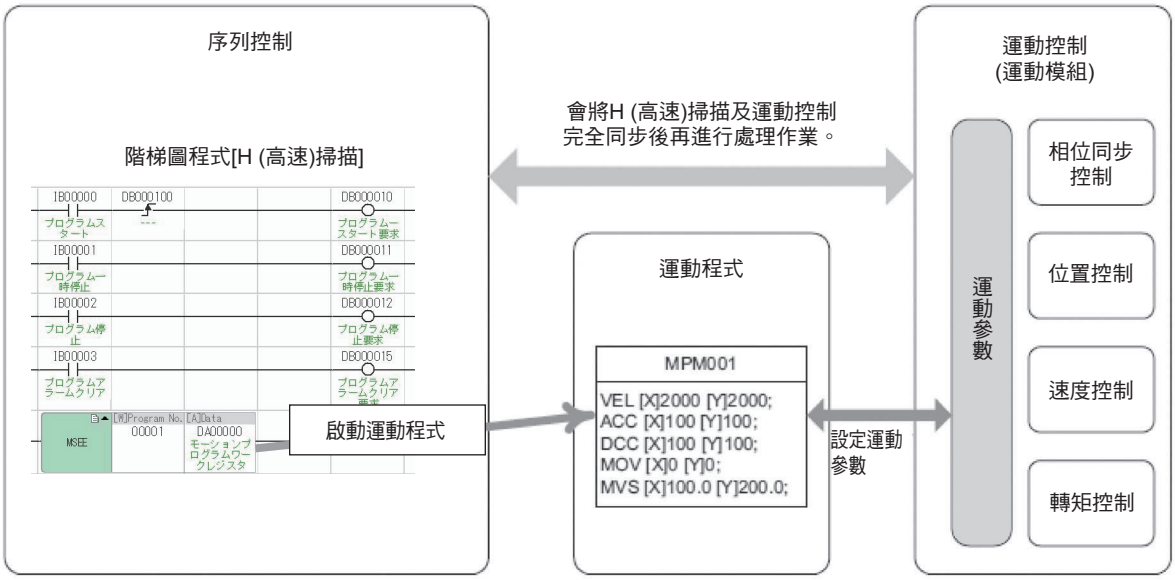
通訊時所使用的階梯圖語言指令 (MSG-SND 指令、MSG-RCV 指令) 適用多種通訊協定，可對觸控式面板、上位 PLC 等多種外部裝置進行通訊控制。如此一來，使用者便能藉由外部裝置來存取控制器內部的暫存器。



補充　若不透過階梯圖程式來進行通訊控制，則可利用 I/O 訊息通訊功能及自動接收功能。
詳情請參閱以下使用手冊。
MP3000 系列　通訊功能　使用手冊 (資料編號：YTWMNCO-15003A)

運動控制完全同步化

H (高速) 掃描模式下，階梯圖程式啟動後會和運動控制完全進行同步，再執行處理作業。另外，此功能還能叫出運動程式，以執行複雜的運動控制，然後再和階梯圖程式互相同步。

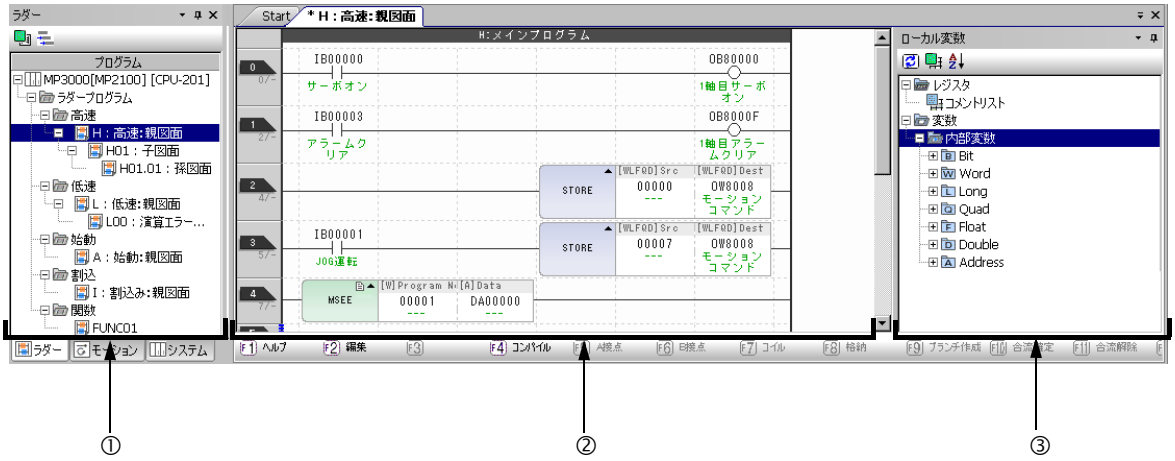


1.3 概要

本節將說明階梯圖程式之概要。

階梯圖程式編輯器

MPE720 Ver. 7 利用以下視窗來編寫或是編輯階梯圖程式。



① 階梯圖子視窗

階梯圖程式顯示時係以圖面為單位。

如欲進一步瞭解圖面，請參閱以下項目之說明。

☞ 階梯圖圖面 (第 1-7 頁)

② 階梯圖程式編輯視窗

此視窗可用來編輯階梯圖程式。

③ 變數子視窗

用來顯示變數。如欲進一步瞭解暫存器，請參閱以下章節之說明。

☞ 第 3 章 暫存器

除了本書所介紹的視窗外，還附有各種視窗及工具列可供使用。

階梯圖圖面

階梯圖程式係以圖面 (階梯圖圖面) 為管理單位，並以圖面編號 (DWG 編號) 作為區別之用。
此圖面為基本階梯圖程式。

類型及階層架構

接下來將說明階梯圖圖面的類型及階層架構。

◆ 類型

依處理目的不同，階梯圖圖面可分為以下 4 種類型。

- DWG.A (啟動處理圖面)
  可用來設定任意的暫存器資訊。在高速掃描處理圖面及低速掃描處理圖面出現前進行處理。
- DWG.I (中斷處理圖面)
  可用來優先處理選配模組所輸入的訊號。
  無論掃描週期為何，本圖面的執行順位皆優於高速掃描處理圖面。
- DWG.H (高速掃描處理圖面)
  可用來執行運動控制或是高速 I/O 控制等作業。
- DWG.L (低速掃描處理圖面)
  可用來執行 HMI 與外部裝置之間通訊或是一般 I/O 控制用途。

下表所示為各種圖面之優先順位、執行條件及最大圖面數。

圖面類型	優先順位*	執行條件	最大圖面數
DWG.A (啟動處理圖面)	1	開啟電源 (僅在開啟電源時執行一次)	64
DWG.I (中斷處理圖面)	2	外部中斷訊號 (選配模組輸出 DI 中斷、計數器模組皆中斷訊號時執行)	64
DWG.H (高速掃描處理圖面)	3	固定週期啟動 (每次進行高速掃描時執行)	200
DWG.L (低速掃描處理圖面)	4	固定週期啟動 (每次進行低速掃描時執行)	1000

* 數字愈小，優先順位愈高。

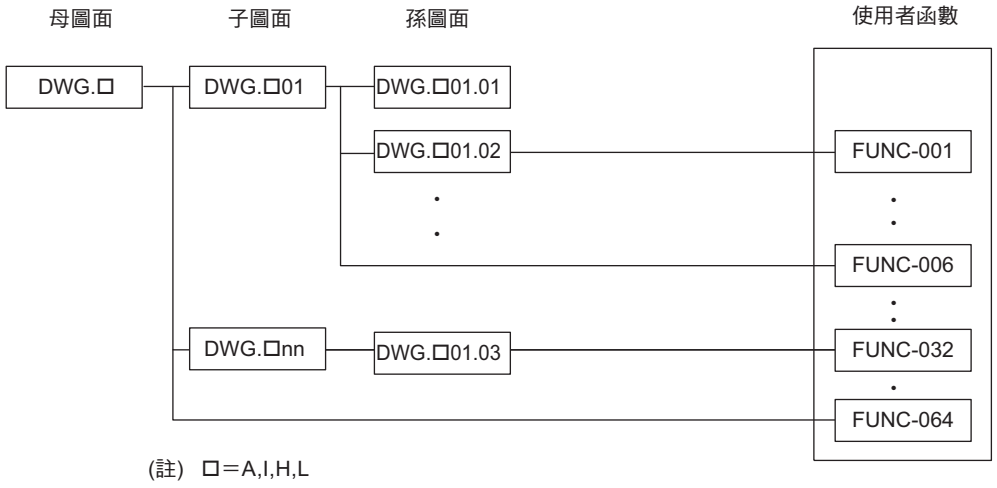
◆ 階層結構

階梯圖圖面係由母圖面、子圖面、孫圖面及演算錯誤處理圖面所組成。

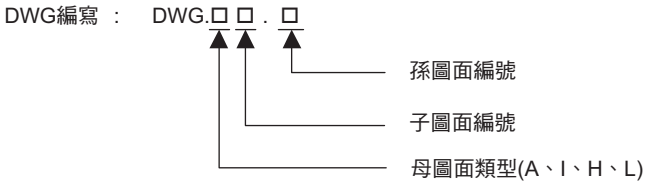
- 母圖面
  當「執行條件」成立，系統程式就會自動執行。
- 子圖面
  母圖面利用 SEE 指令進行參照時執行。
- 孫圖面
  子圖面利用 SEE 指令進行參照時執行。
- 演算錯誤處理圖面
  發生演算錯誤時，系統程式即自動執行。

母圖面不得參照不同類型的子圖面，同樣地，子圖面亦無法參照不同類型的孫圖面。此外，亦無法從母圖面直接參照孫圖面。採用母圖面僅能參照子圖面，而子圖面也僅可參照孫圖面的結構，這就是圖面的「階層結構」。

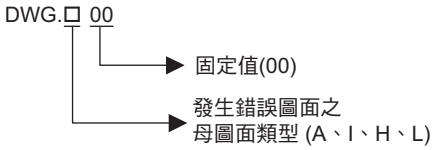
下圖係利用母圖面－子圖面－孫圖面的顯示方式，將所有處理程式階層化。



(註) □=A,I,H,L



(註) 以下名稱僅適用於演算錯誤圖面。



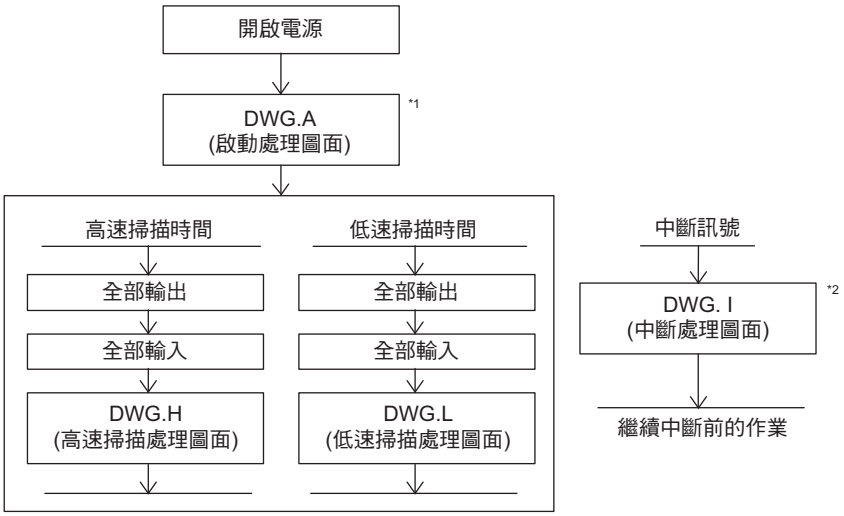
下表為各種階梯圖圖面之圖面數的明細。

圖面	圖面數			
	DWG.A	**DWG.I**	**DWG.H**	**DWG.L**
母圖面	1	1	1	1
演算錯誤處理圖面	1	1	1	1
子圖面	總和最大為 62	總和最大為 62	總和最大為 998	總和最大為 1998
孫圖面				

補充　除了圖面外，還提供了函數功能，讓使用者可從各種圖面自由參照資料。進入母圖面、子圖面或孫圖面並利用 FUNC 指令來參照，即可執行函數。此外，最多可編寫 2000 個函數。

圖面執行控制

各種階梯圖圖面將依照不同的優先順位，執行以下的處理作業。

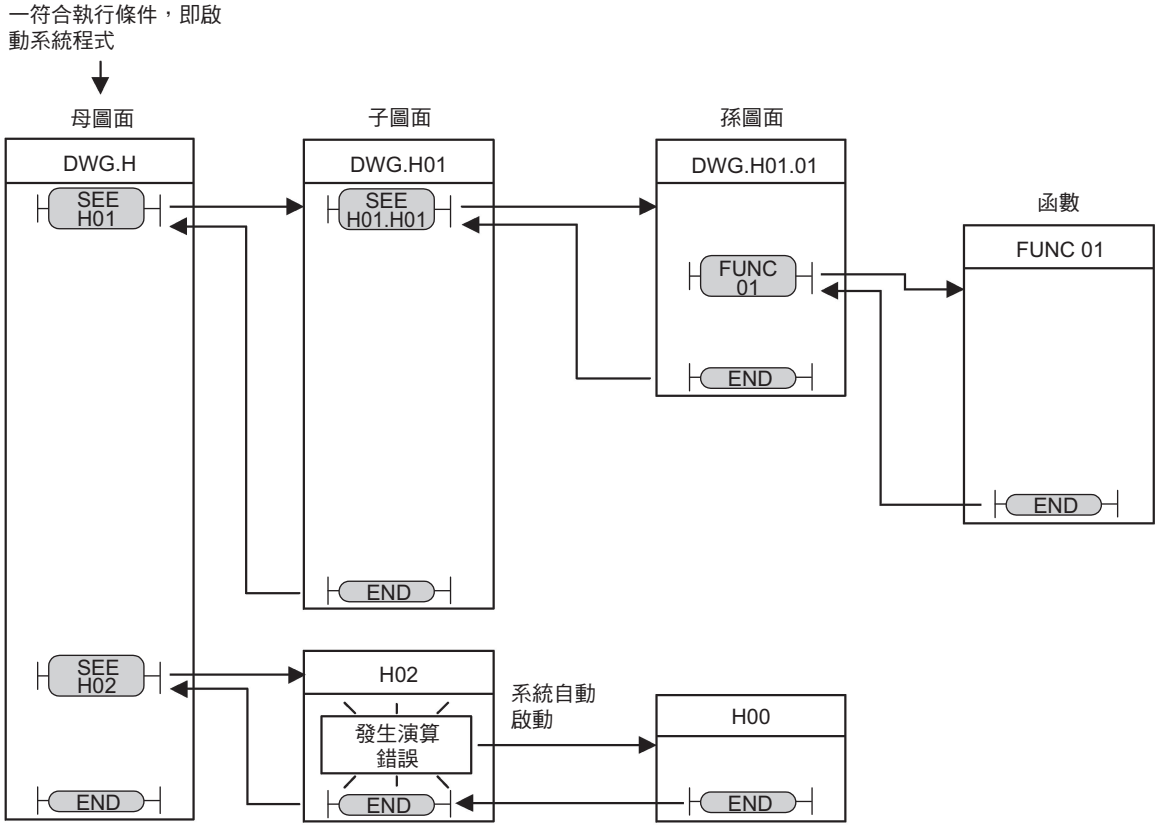


*1. 若編寫的圖面為 DWG.A，則會在開啟電源後執行本作業。
*2. 當中斷訊號輸入時，即使正在執行 DWG.H、DWG.L，仍會優先執行 DWG.I。

(註)所有圖面的母圖面皆由系統自動叫出後執行。

◆ 圖面執行處理方式

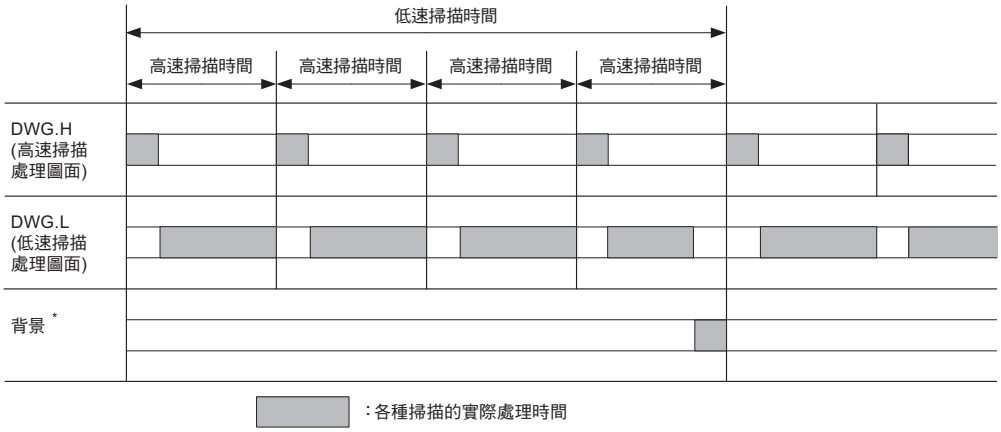
採用階層化結構的圖面係透過上位圖面參照下位圖面的方式來執行處理作業。下圖係以 DWG.H (高速掃描處理圖面) 為例，說明圖面的執行處理方式。



(註)1. 母圖面係由系統自動叫出後執行。
進入子圖面和孫圖面、母圖面和子圖面並編寫 SEE 指令後即可執行作業。
2. 所有的圖面皆可參照函數。此外，函數和函數之間亦可互相參照。
3. 發生演算錯誤時，就會自動啟動各個圖面相對應之演算錯誤處理圖面。
4. 使用演算錯誤處理圖面時，圖面編號必須指定為 00。

◆ 高速 / 低速掃描處理圖面之執行排程

DWG.H (高速掃描處理圖面) 和 DWG.L (低速掃描處理圖面) 無法同時執行。
DWG.L 會利用 DWG.H 空餘時間執行。
DWG.H 執行週期稱為 「高速掃描時間」，而低速掃描處理圖面的執行週期則稱之為 「低速掃描時間」。



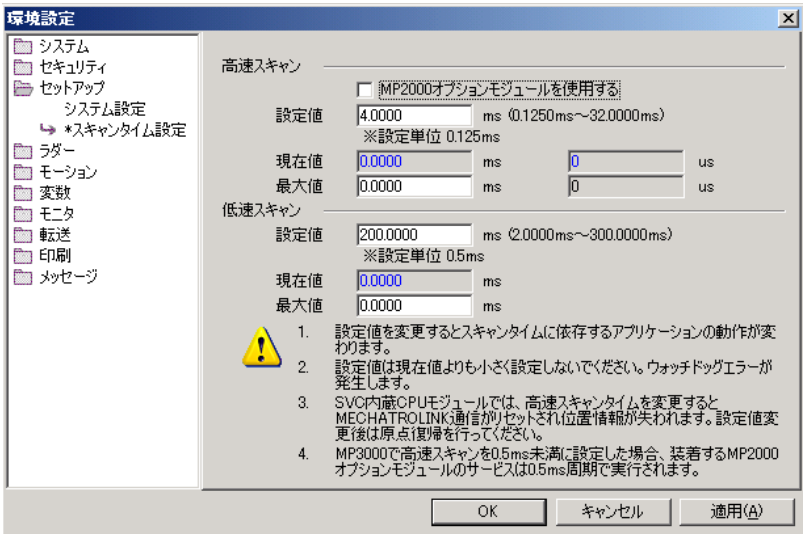
* 目的在於執行系統內部處理 (通訊處理等) 作業。

◆ 設定高速 / 低速掃描時間

請使用 MPE720 Ver. 7 並依照以下步驟來設定高速掃描時間及低速掃描時間。

1. **請從主選單上依序選擇 [檔案] － [環境設定](或是從我的工具視窗中點擊 [系統設定] 鍵) 後，畫面上就會出現 [環境設定] 對話框。**

2. **依序選擇 [設定] － [設定掃描時間] 後，即可叫出以下的對話框。**



設定值：輸入各種掃描時間。
現在值：離線時顯示為 「0.0 ms」。離線狀態下，畫面上將顯示各種掃描實際的處理時間。
最大值：畫面上將顯示各種掃描實際處理時間之最大值。此外，當實際數值超過您所設定的數值時，將維持最大值。

3. 請在 [高速掃描] 的 [設定值] 窗格中輸入高速掃描時間，並在 [低速掃描] 的 [設定值] 窗格中輸入低速掃描時間。

下表為各種掃描時間之數值設定範圍及初始值。

項目	數值的設定範圍	初始值
高速掃描時間	0.125 ms ~ 32 ms (以 0.125 ms 為單位)	4.0 ms
低速掃描時間	2.0 ms ~ 300.0 ms (以 0.5 ms 為單位)	200.0 ms

(註)數值設定範圍及初始值等依機型而異。詳情請參閱您所使用模組之使用手冊。

重要

設定高速掃描及低速掃描時間時，請注意以下重點。

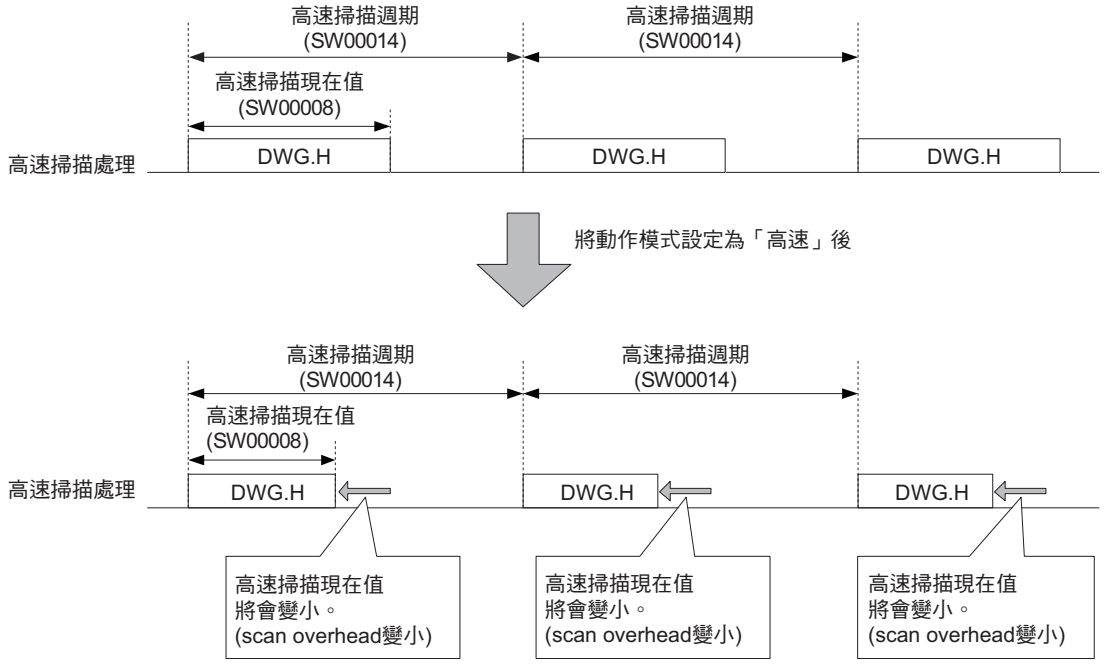
1. 設定掃描值時，設定值必須大於掃描最大值的 1.25 倍以上。
   若您所設定的掃描設定值過於趨近掃描最大值，MPE720 視窗更新速度將會變得非常緩慢，甚至還會發生通訊逾時等情形。此外，一旦掃描最大值大於掃描設定值，因監視逾時，可能會造成運動控制器系統當機。
2. 使用 MECHATROLINK-II、MECHATROLINK-III 必須分別將通訊週期及傳送週期設定為整數倍。變更通訊週期及傳送週期後，請確認掃描設定值。
3. 請勿在伺服裝置啟動狀態下變更掃描設定值。尤其是在轉軸移動 (馬達旋轉) 狀態下，可能會造成馬達旋轉動作異常，因此嚴禁變更設定值。
4. 掃描時間設定 / 變更完成後，請務必執行快閃記憶體儲存功能，以便將資料儲存於快閃記憶體中。

◆ 設定高速圖面動作模式

所謂「高速圖面動作模式」就是針對 DWG.H (高速掃描處理圖面) 所進行之模式設定。

若不使用 DWG.I (中斷處理圖面)，請選擇高速模式。可讓 DWG.H 處理效率達到最佳化。

使用 DWG.I (中斷處理圖面) 前，請先選擇一般模式。選擇高速模式時，將不會執行 DWG.I。



補充

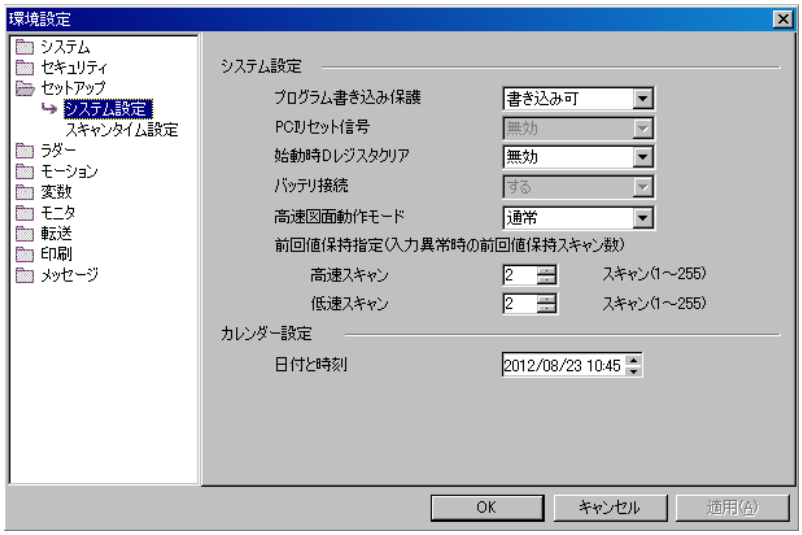
・DWG.A (啟動處理圖面)、DWG.I (中斷處理圖面)、DWG.L (低速掃描處理圖面) 等未配備動作模式變更功能。
・只要多使用以下指令，即可讓 DWG.H 處理作業達到最佳化效果。

類型	功能
繼電器電路指令	上升 A 接點
	下降 A 接點
	上升 B 接點
	下降 B 接點
	上升脈衝
	下降脈衝
	線圈
	反轉型線圈
	上升變化檢測用線圈
	下降變化檢測用線圈
	設定線圈
	重置線圈
數值演算指令	+1遞增
	−1遞減

設定高速圖面動作模式時，必須使用 MPE720 Ver. 7，並依照以下步驟來進行。

1. **請從主選單上依序選擇 [檔案] － [環境設定](或是從我的工具視窗中點擊 [系統設定] 鍵) 後，畫面上就會出現 [環境設定] 對話框。**

2. **依序選擇 [設定] － [系統設定] 後，即可叫出以下的對話框。**



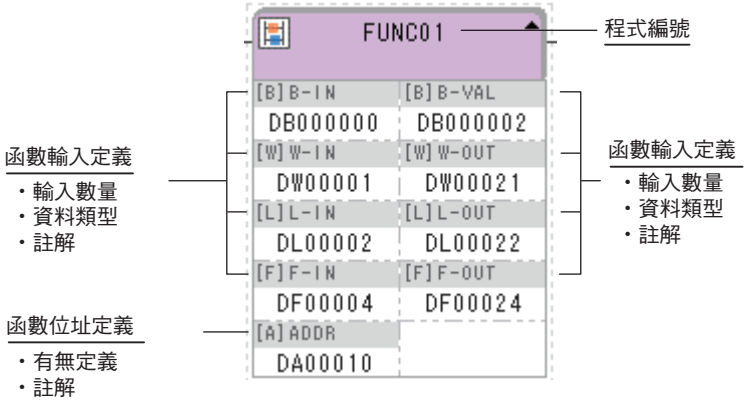
3. **在 [高速圖面動作模式] 窗格中選擇 [高速] 或 [一般]。**

使用者函數

何謂「使用者函數」

使用者函數就是由使用者自由編寫函數定義 (程式編號及輸出輸入定義) 和處理內容的函數。

下圖為函數定義之內容。



使用者函數概述

使用者函數的主要處理內容係利用階梯圖程式所編寫而成。

使用者只要進入母圖面、子圖面或是孫圖面，並利用 FUNC 指令來參照，即可執行使用者函數。

以下為可供參照的使用者函數。

・ 可從任何一個圖面任意參照。

・ 可從不同類型或階層的圖面同時參照同一個使用者函數。

・ 可從某個使用者函數參照其他的使用者函數。

・ 可使用不同的程式多次參照。

使用者函數具有以下優點。

・ 輕鬆即可將使用者程式零件化

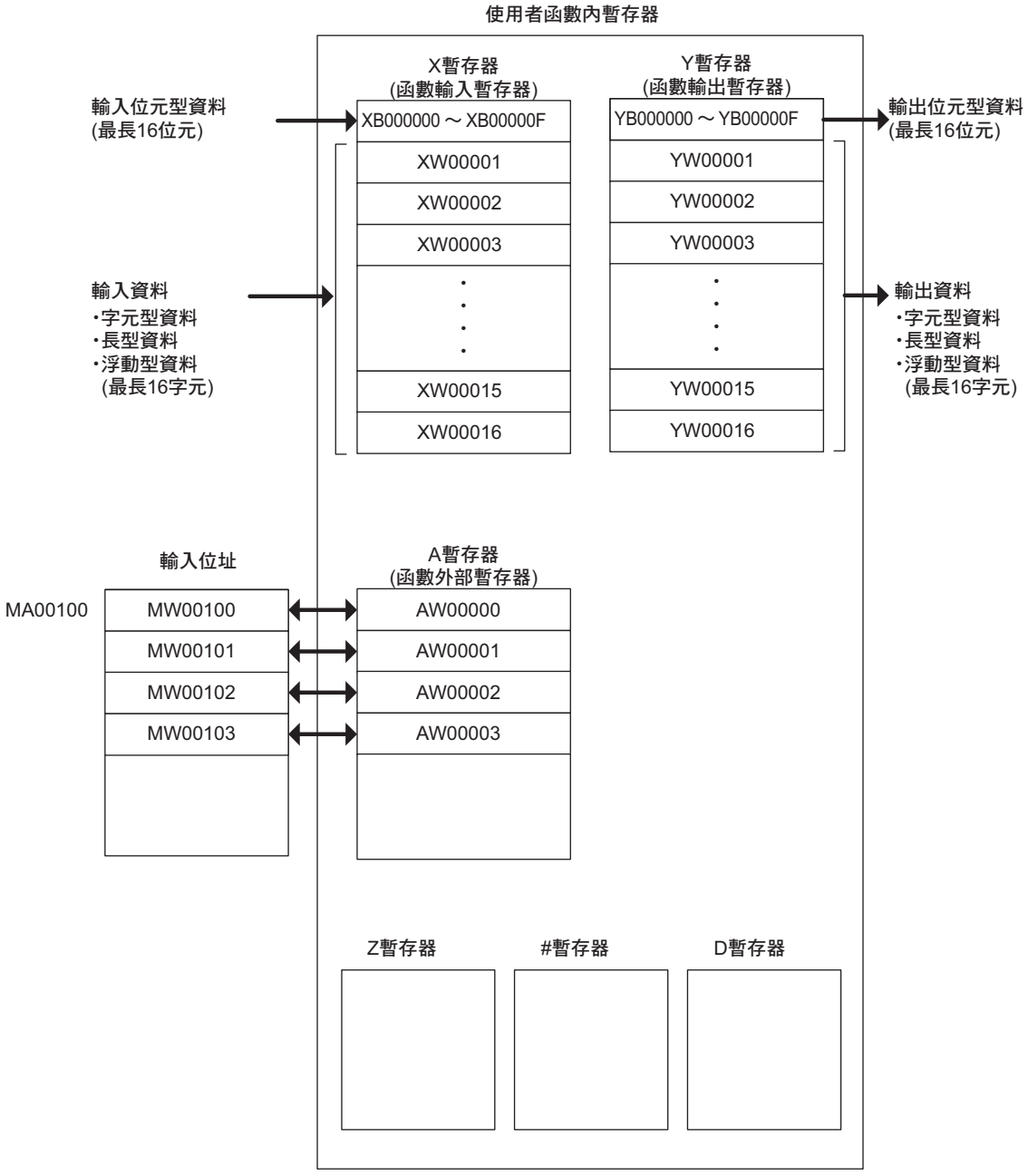
・ 編寫及維護使用者程式時更輕鬆

重要

叫出使用者函數時，必須考量每個使用者函數的暫存器現在值，然後再進行初始化等處理作業。
如欲進一步瞭解，請參閱以下項目之說明。

☞ **第 3 章 暫存器 – ■ 使用者函數內局部暫存器之使用注意事項 (第 3-5 頁)**

下圖為使用者函數的輸出輸入資料與使用者函數內暫存器之間的關係。



補充

1. 使用者函數在叫出 X 暫存器、Y 暫存器、Z 暫存器或 D 暫存器時之初始值計算方式各不相同。如欲進一步瞭解，請參閱以下項目之說明。

☞ 第 3 章 暫存器 – ■ 使用者函數內局部暫存器之使用注意事項 (第 3-5 頁)

2. S、M、I、O、C 暫存器可由函數內參照。

使用者函數之編寫

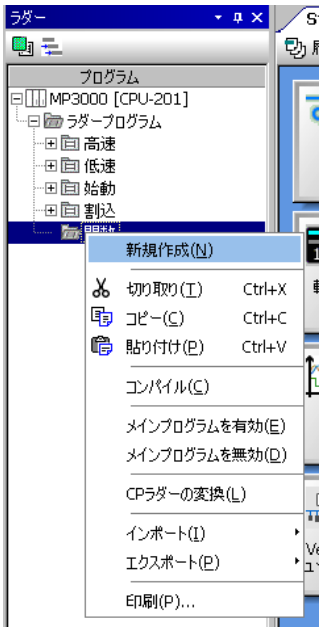
本節將以下列使用者函數為例，說明使用者函數之編寫步驟。

函數定義	名稱	備註
程式編號	FUNC01	–
函數輸入值	IN	整數型
函數輸出值 1	OUT1	整數型
函數輸出值 2	OUT2	整數型
處理內容		
將函數輸入值 (IN) 增加 2 倍後，再輸出為函數輸出值 1 (OUT1)。 將函數輸入值 (IN) 增加 3 倍後，再輸出為函數輸出值 2 (OUT2)。		

◆ 編寫步驟

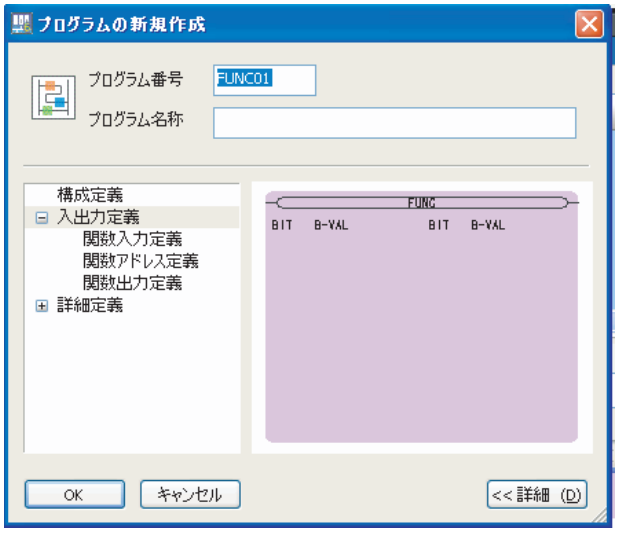
1. **進入啟動程式，並依序選擇 [編寫程式] － [階梯圖程式]。**
   畫面上將出現階梯圖子視窗。

2. **在階梯圖程式的 [函數] 上按一下右鍵，然後再選擇 [開新檔案]。**

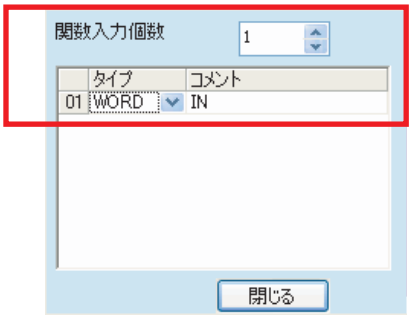


   畫面上將出現 [開啟新程式] 對話框。

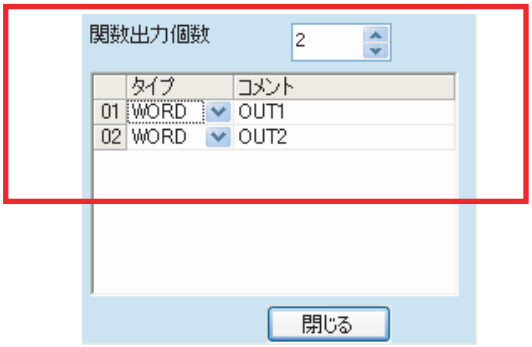
3. **請在 [程式編號] 窗格中輸入「FUNC01」。**



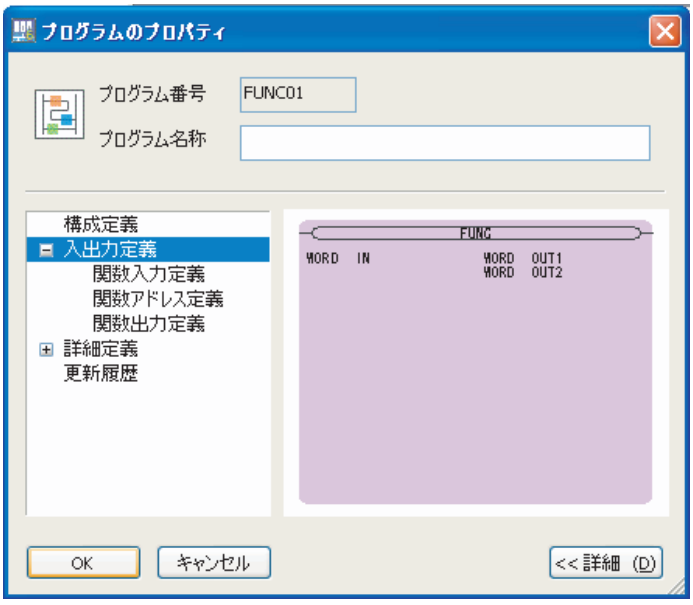
4. 依序選擇 [輸出輸入定義] － [函數輸入定義] 後，即可依照下圖所示開始編寫。



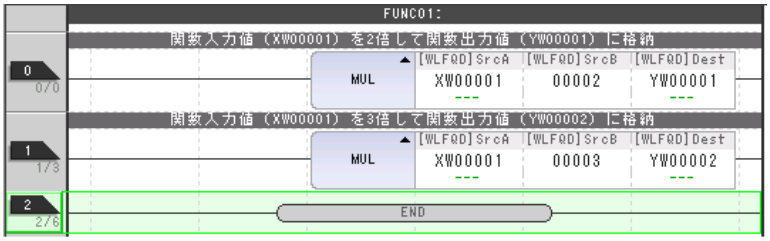
5. 依序選擇 [輸出輸入定義] － [函數輸出定義] 後，即可依照下圖所示開始編寫。



6. 請點擊 [OK] 鍵。以上為所有的函數定義設定步驟。



7. 進入步驟 5 所編寫的使用者函數「FUNC01」圖面，並依照下圖所示來編寫階梯圖程式。



8. 叫出階梯圖程式後，請進入主選單並依序選擇 [編譯] － [編譯] 後，即開始進行編譯作業。
編譯完成後，階梯圖程式會自動被儲存。

重要

執行程式編譯時，一旦輸出子視窗顯示錯誤訊息，階梯圖程式就不會被儲存起來。

以上為使用者函數之編寫步驟。

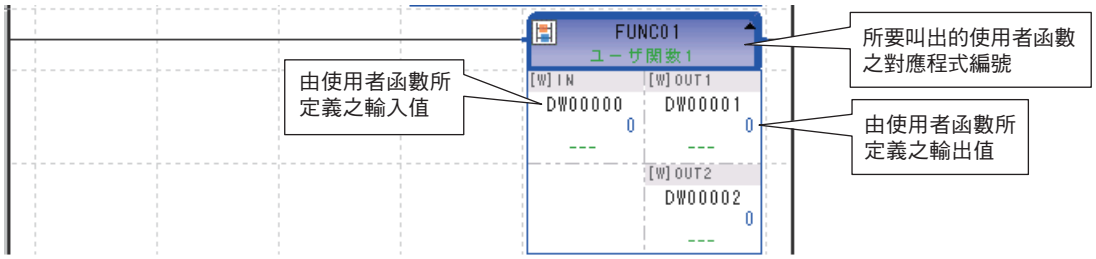
叫出使用者函數

利用階梯圖圖面的 FUNC 指令，即可叫出使用者函數。

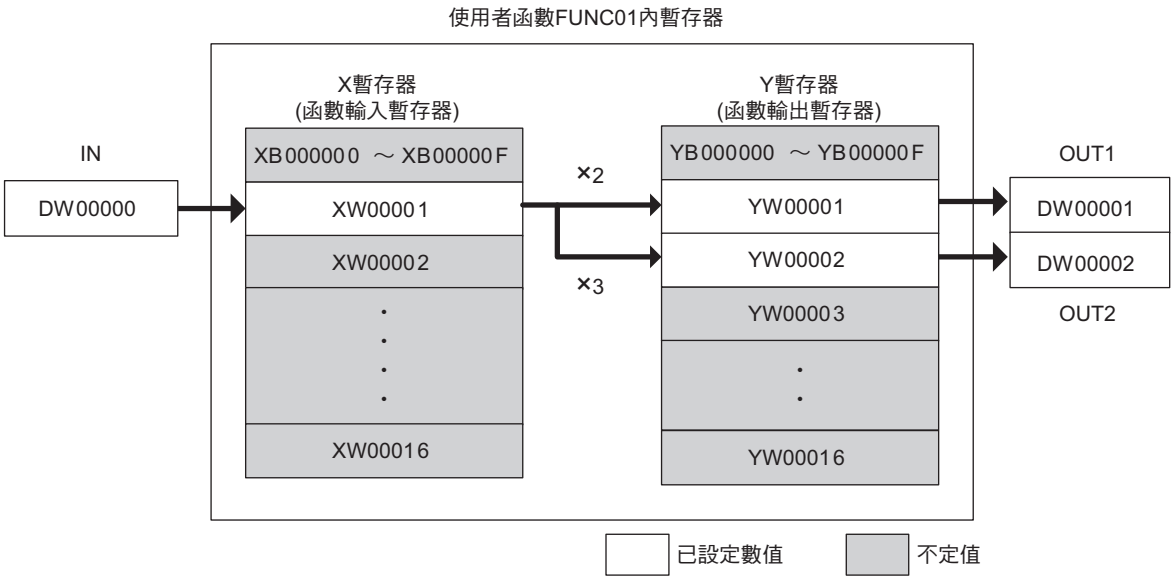
接下來將說明如何利用編寫完成的使用者函數來叫出 DWG.H (高速掃描處理畫面) 之範例。

■ 利用 DWG.H 叫出使用者函數 FUNC01 之範例

請依照下圖所示，將 FUNC 指令寫入 DWG.H 裡。



依照上圖所示的內容來編寫程式後，即進入下述狀態。

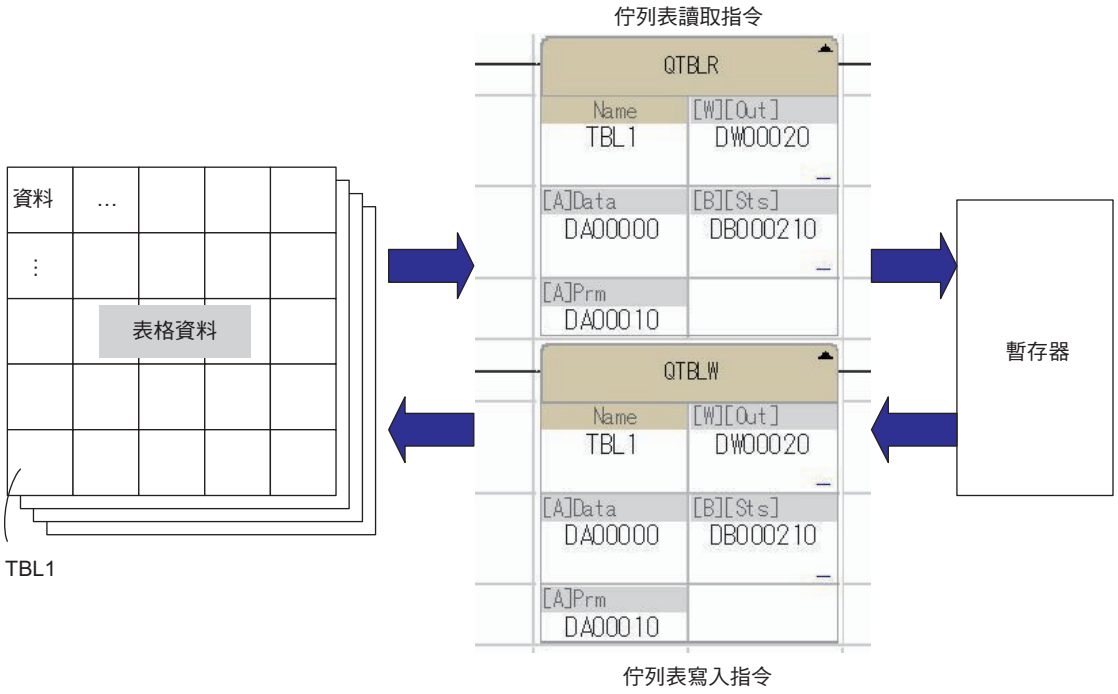


利用 DWG.H，將 DW00000 設定為 10 後，請確認 DW00001 是否為 20，且 DW00002 是否為 30，以證明是否正確叫出您所編寫的使用者函數。

表格資料

所謂「表格資料」

所謂「表格資料」就是以表格型式來管理的一種資料類型。此類資料和暫存器所儲存的位置不同。
利用階梯圖程式來執行資料表操作指令後，表格資料就會被展開並顯示於暫存器上，或是將表格資料儲存於暫存器中。當暫存器的空間不足時，此功能亦可當作資料救援用途。



製作表格資料

編寫表格資料前，必須先設定好下述表格定義資訊及列屬性等內容。

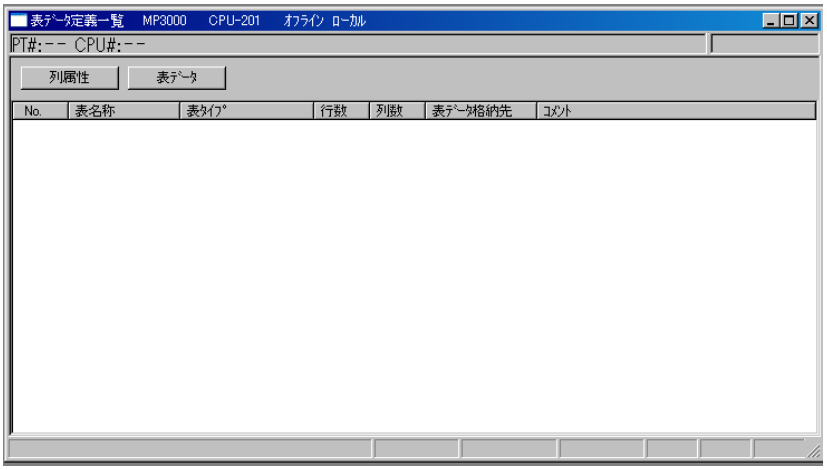
表格定義資訊	內容
表格名稱	用來顯示表格名稱。
表格類型	用來選擇陣列型 / 記錄型。 ・陣列型：此種表格類型所有列的列屬性皆相同。 ・記錄型：此種表格類型各列的列屬性皆各異。
列數	用來顯示表格的列數。(最多為 10000 列)
行數	用來顯示表格的行數。(最多為 10000 行)
表格註解	用來顯示表格之註解。
表格資料儲存目的地	用來選擇一般 / 電池組備用。 ・一般：每個表格的最大容量為 5 MB。 ・電池組備用： 每個表格的最大容量為 3 MB。 如欲瞭解表格資料的最大容量及適合電池組備用的機型，請參閱以下章節。 ☞ 附錄 E 運動控制器規格一覽表

列屬性	內容
列名稱	用來顯示列名稱。
資料類型	包含整數、長整數、實數及文字列等類型。
尺寸	用來顯示資料類型的長度。
顯示類型	包含 2 進位制、10 進位制、16 進位制、實數及文字列等類型。
列註解	用來顯示列之註解。

■ 編寫步驟

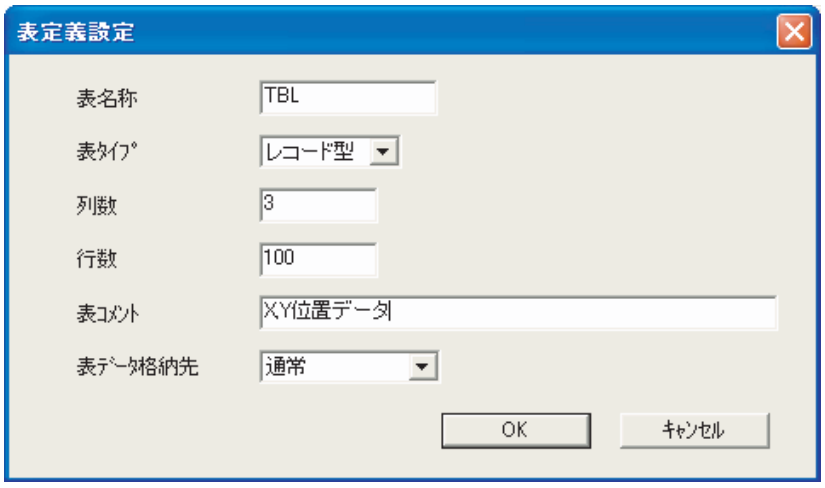
1. **進入啟動程式，並依序選擇 [公用程式] － [Engineering Manager]。**
   啟動 Engineering Manager。

2. **從 Engineering Manager 的主選單上，依序選擇 [檔案] － [開啟] － [定義表格資料] － [表格資料清單]。**
   畫面上將出現 [表格資料清單] 對話框。



3. **讓 [表格資料清單] 對話框留在畫面上，然後再利用 Engineering Manager 的主選單依序選擇 [檔案] － [開新檔案]。**
   畫面上將出現 [表格定義設定] 對話框。

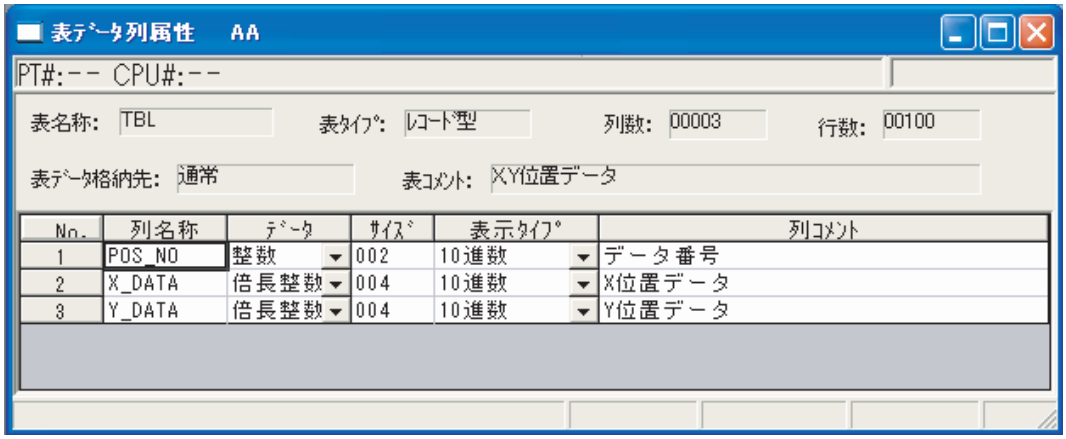
4. **表格定義資訊設定完成後，請點擊 [OK] 鍵。**



即可叫出 [表格資料列屬性] 對話框。

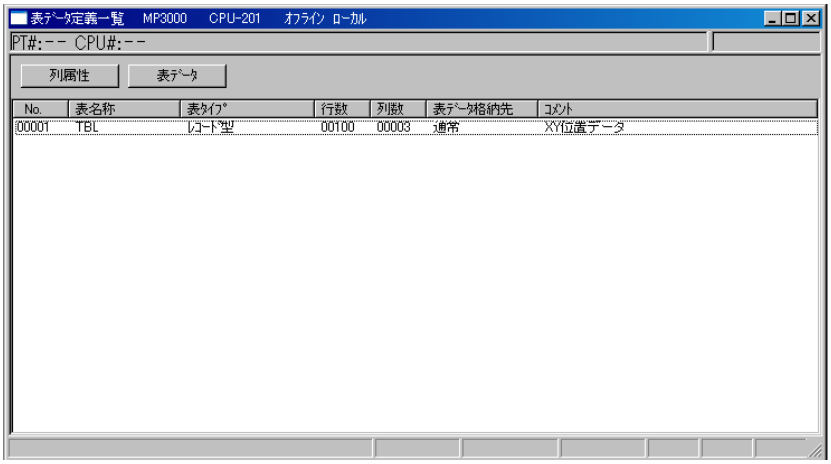
5. 設定表格資料列屬性。

(註)設定表格定義資料時，若選擇陣列型作為表格類型，則只要設定列屬性一個項目。



6. 從主選單上依序選擇 [檔案] － [儲存]，以儲存設定資料。

接著，畫面上將會出現步驟 2 所曾經出現過的 [表格定義清單] 對話框編寫完成後之表格。



以上為表格資料之編寫步驟。

補充

1. 編寫表格資料時，資料內容將被初始化為 0。
2. 選擇 [表格定義清單] 對話框中所編寫的表格資料，然後再點擊 [表格資料] 後，即可開始讀取 / 寫入表格資料。
3. 利用階梯圖程式來處理表格資料時，必須使用資料表操作指令。

表格資料

階梯圖程式開發流程

2

本章將針對階梯圖程式開發流程進行說明。

2.1 概要

接下來將利用以下流程來說明階梯圖程式開發流程。

① 備妥連線裝置

備妥連線裝置並加以配線。
將 MPE720 Ver. 7 安裝於電腦中。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.2 **備妥連線裝置 – 連接硬體** (第 2-3 頁)
☞ 2.2 **備妥連線裝置 – 安裝** MPE720 Ver. 7(第 2-4 頁)

▼

② 製作專案

製作新的專案以作為開發階梯圖程式之前置準備工作。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.3 **製作專案** (第 2-5 頁)

▼

③ 自動配置

執行自動配置程式，以啟動系統。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.4 **自動配置** (第 2-6 頁)

▼

④ 在線連線

設定和運動控制器通訊時所需的通訊條件。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.5 **在線連線** (第 2-7 頁)

▼

⑤ 編寫階梯圖程式

在階梯圖編輯器中編寫階梯圖程式。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.6 **編寫階梯圖程式** (第 2-8 頁)

▼

⑥ 寫入階梯圖程式

將編寫完成的階梯圖程式傳送至運動控制器。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.7 **寫入階梯圖程式** (第 2-12 頁)

▼

⑦ 確認階梯圖程式的動作

確認編寫完成的階梯圖程式動作。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.8 **確認階梯圖程式的動作** (第 2-14 頁)

▼

⑧ 將階梯圖程式儲存至快閃記憶體

將除錯完成的階梯圖程式儲存於快閃記憶體中。
如欲進一步瞭解，請參閱以下章節之說明。
☞ 2.9 **將階梯圖程式儲存至快閃記憶體** (第 2-18 頁)

(註)本頁所示的流程僅為階梯圖程式的某個設計範例。將程式嵌入實際的裝置時，必須和外部裝置進行進一步的設定。

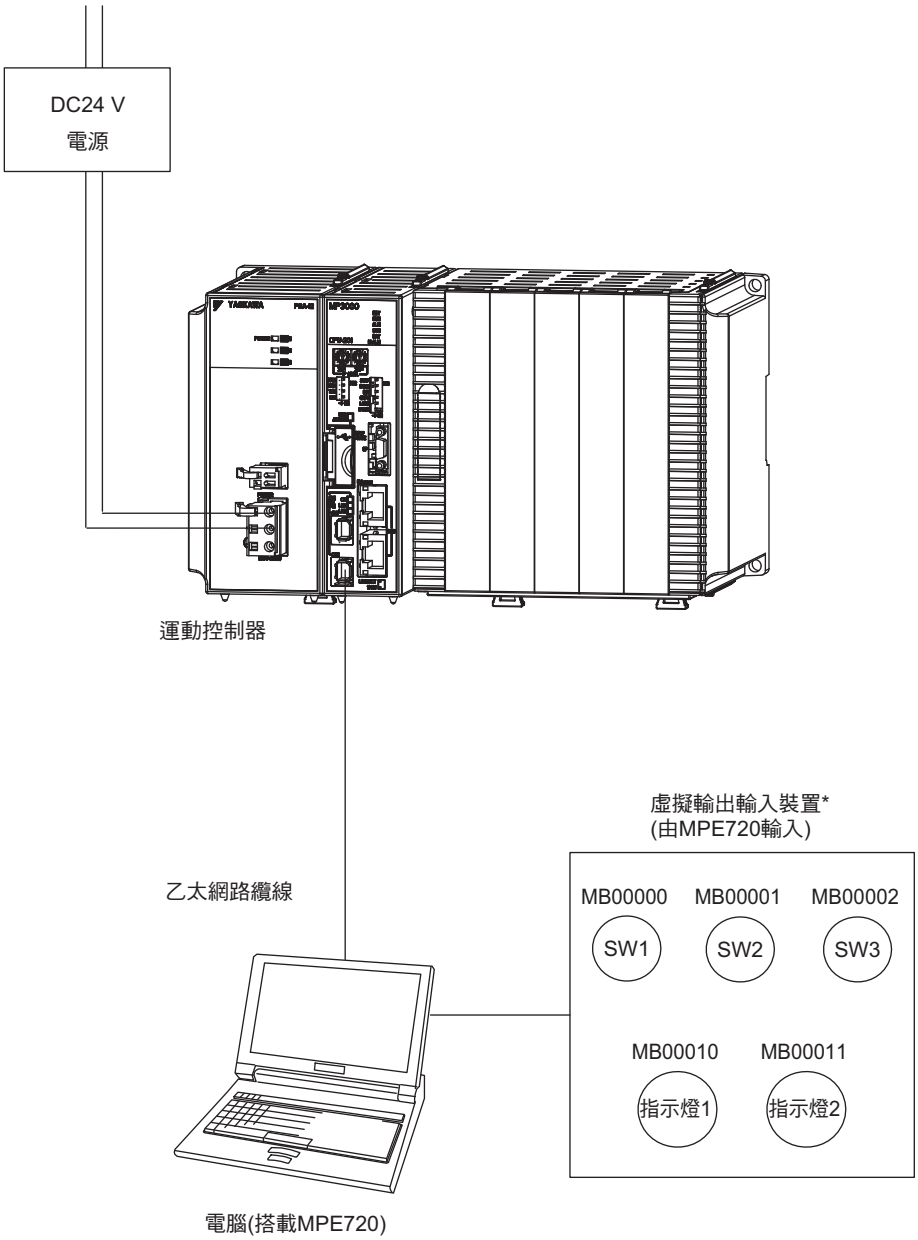
2.2 備妥連線裝置

本節將針對硬體連線及安裝 MPE720 Ver. 7 等相關內容進行說明。

連接硬體

備妥連線裝置並加以配線。

接下來，將採用以下的系統架構模組來進行說明。



* 上圖所示範例係使用運動控制器內部的 M 暫存器做為虛擬輸出輸入裝置。
實際使用時，請將輸出輸入訊號連接至運動控制器的 I/O 模組，然後再使用 I、O 暫存器來編寫階梯圖程式。

安裝 MPE720 Ver. 7

將 MPE720 Ver. 7 安裝至電腦中。

安裝步驟請參閱以下手冊。

MP2000/MP3000 系列　運動控制器系統　安裝手冊 (資料編號：SIJP C880725 00)

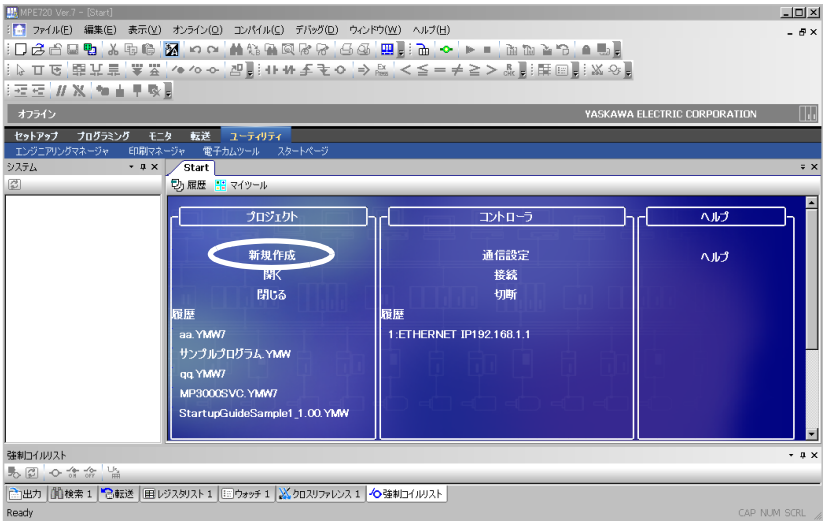
2.3 製作專案

請依照下列步驟來編寫程式。

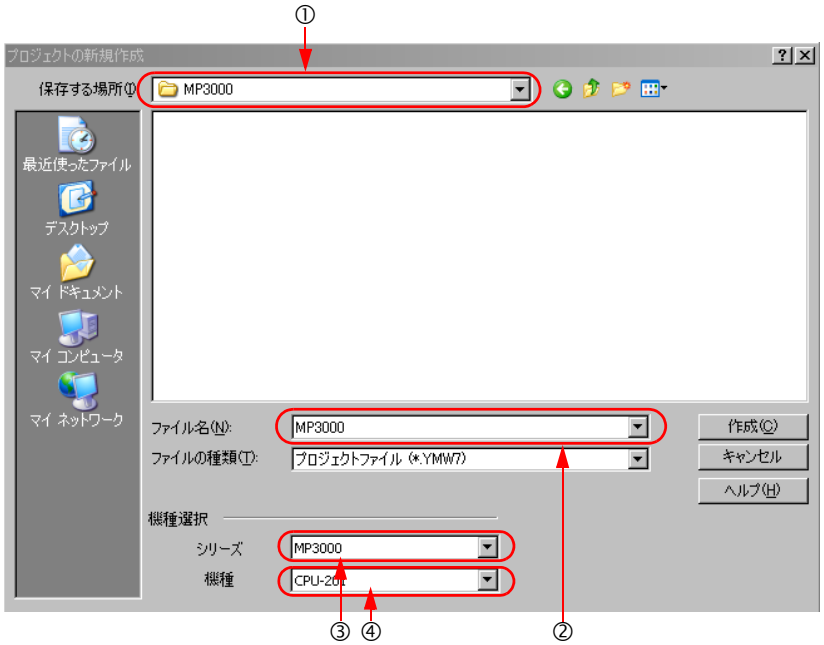
1. 請從電腦桌面畫面雙擊以下圖示，以啟動 MPE720 Ver. 7。

MPE720 Ver.7

2. 找到 [開始] 分頁，並點擊 [開新檔案]。



3. 指定您所需要的檔案名稱、檔案儲存位置、運動控制器的系列名稱及機型。



①在 [儲存位置] 窗格中指定檔案的儲存位置。

②在 [檔案名稱 (N)：] 窗格中輸入檔案名稱。

③從 [系列名稱] 窗格中選擇相對應的系列名稱。

④從 [機型] 窗格中選擇相對應的機型名稱。

4. 點擊 [製作] 鍵。

2.4 自動配置

使用自動配置功能來啟動系統。所謂「自動配置」就是一種可自動辨識運動控制器所裝載的模組以及以 MECHATROLINK 連接器所連接裝置的功能。透過此一功能，即可輕鬆並且在最短時間內啟動系統。自動配置功能可透過運動控制器的 DIP 開關或 MPE720 來執行。

自動配置步驟請參閱以下手冊。

- MP3000 系列　MP3200 使用手冊 (資料編號：SIJP C880725 10)
- MP3000 系列　MP3300 使用手冊 (資料編號：YTWMNCO-14008A)

2.5 在線連線

使用者必須設定好運動控制器和電腦通訊時所需之通訊條件。
通訊設定步驟請參閱以下手冊。

MP2000/MP3000 系列　運動控制器系統　安裝手冊 (資料編號：SIJP C880725 00)

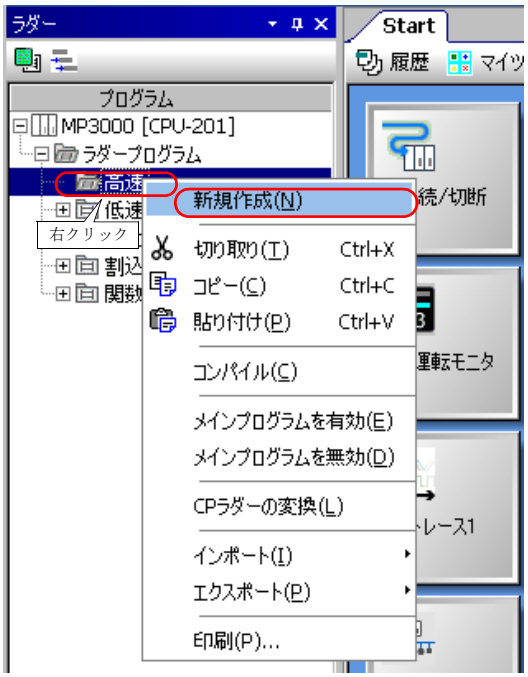
2.6 編寫階梯圖程式

請依照以下步驟來編寫階梯圖程式。

> 註記
>
> 本節所介紹的是「高速」程式之編寫步驟，編寫「低速」及「啟動」程式之步驟亦同。

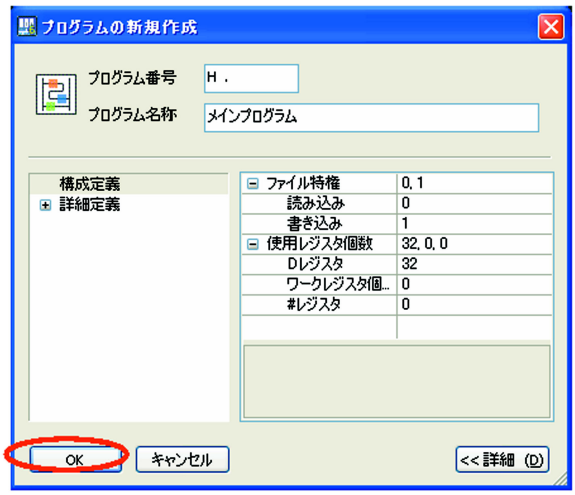
1. **進入啟動程式，並依序選擇 [編寫程式] － [階梯圖程式]。**
   畫面上將出現階梯圖子視窗。

2. **在階梯圖程式的 [高速] 上按一下右鍵，然後再選擇 [開新檔案]。**



   畫面上將出現 [開啟新程式] 對話框。

3. **請點擊 [OK] 鍵。**



   啟動階梯圖編輯器。

4. **編寫階梯圖程式。**
階梯圖程式係依照插入指令集、插入指令以及填寫指令參數等順序編寫而成。如欲進一步瞭解，請參閱以下項目之說明。
☞ 階梯圖程式編寫範例 (第 2-9 頁)

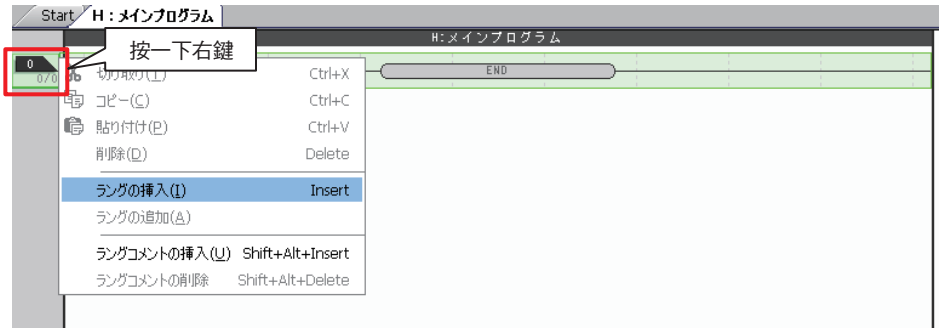
5. **叫出階梯圖程式後，請進入主選單並依序選擇 [編譯] － [編譯] 後，即開始進行編譯作業。**
編譯完成後，階梯圖程式會自動被儲存。

> 重要
> 執行程式編譯時，一旦輸出子視窗顯示錯誤訊息，階梯圖程式就不會被儲存起來。

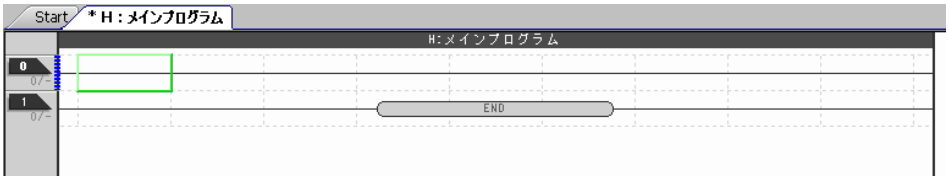
階梯圖程式編寫範例

以下將介紹插入 NOC 指令之範例。

1. **在顯示行編號的位置按一下右鍵，然後再選擇 [插入指令集]。**

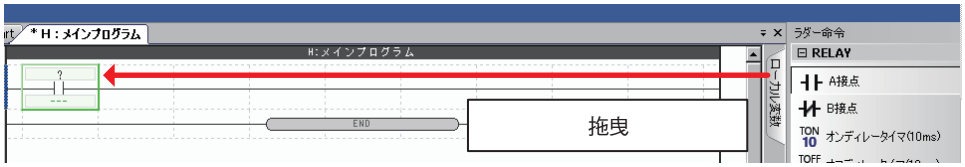


插入指令集。

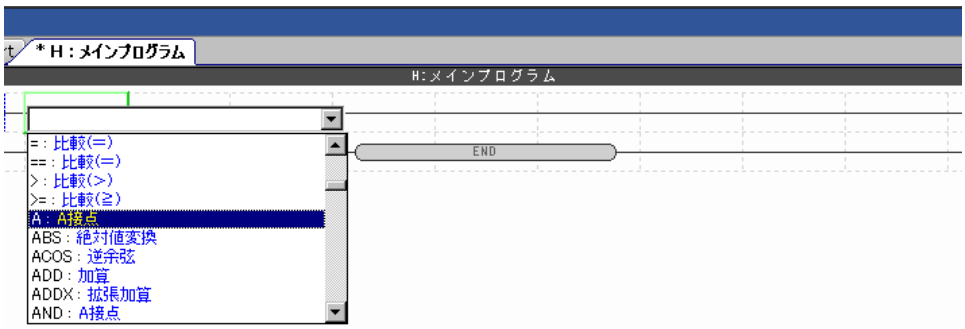


2. **您可利用以下任一種方法來編寫 NOC 指令。**

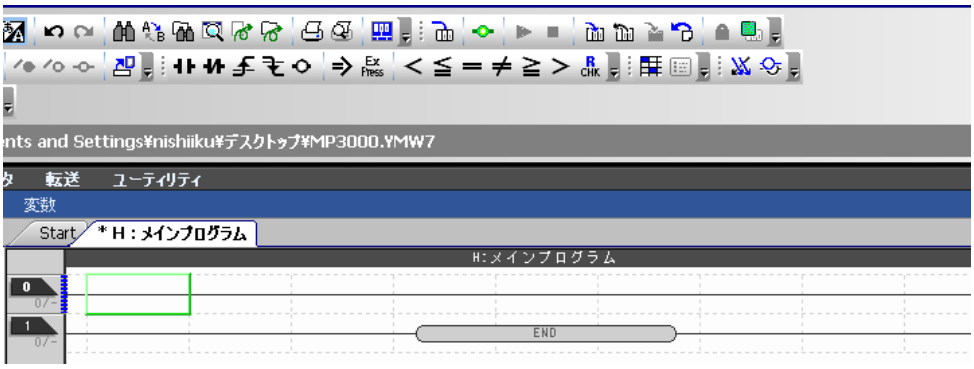
・ 在工作窗格中依序選擇 [RELAY] － [A 接點]，然後再拖曳到所插入指令集上。



・ 在插入指令集上雙擊您想要編寫 NOC 指令的位置，接著再從畫面上所顯示的清單中選擇 [A：A 接點]。



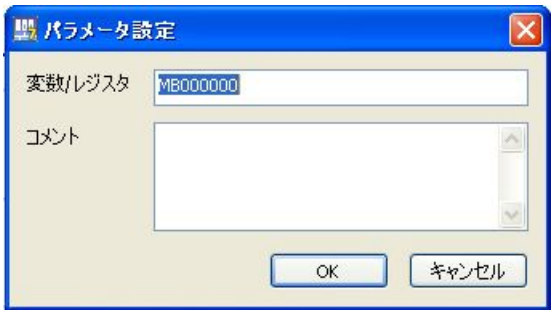
·在插入指令集上選擇您所要編寫 NOC 指令的位置，並點擊 [NOC 指令] 鍵。



3. **雙擊標示為「？」的部分。**

畫面上將出現 [參數設定] 對話框。

4. **在 [變數 / 暫存器] 窗格中輸入暫存器「MB000000」，然後再點擊 [OK] 鍵。**



「MB000000」就會被顯示在 NOC 指令中。

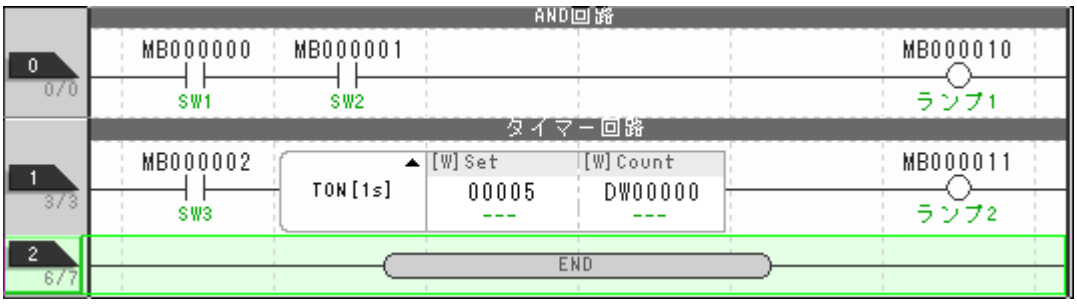
(註)適用之暫存器類型及資料類型依指令而異。如欲進一步瞭解各種指令相關內容，請參閱以下章節。

☞ **第 4 章 階梯圖程式語言指令**

補充　在您想要插入註解的該行的行編號顯示位置按一下右鍵，然後再選擇 [插入指令集註解]，即可插入註解。

5. 請依照步驟 1 ~ 4 來編寫階梯圖程式。以下所示僅為階梯圖程式及時序圖的其中一個範例。

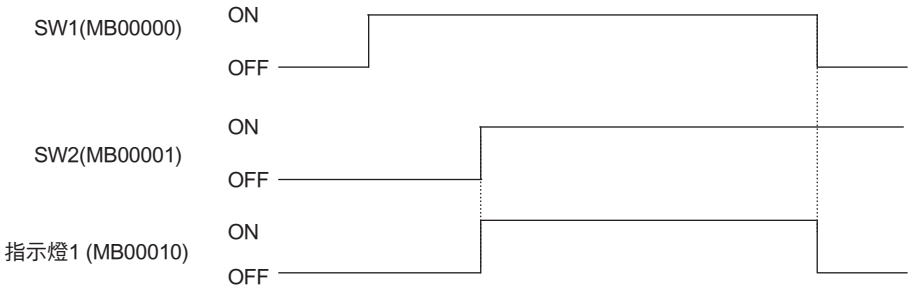
< 階梯圖程式範例 >



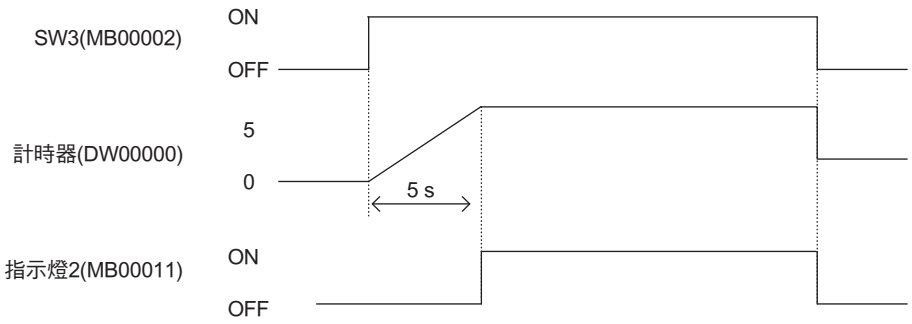
(註)上圖所示的階梯圖程式係利用 M 暫存器來編寫開關、指示燈之範例。
使用者在編寫階梯圖程式時，必須配合實際的裝置，並使用 I、O 暫存器來編寫。

<時序圖範例>

AND電路之動作



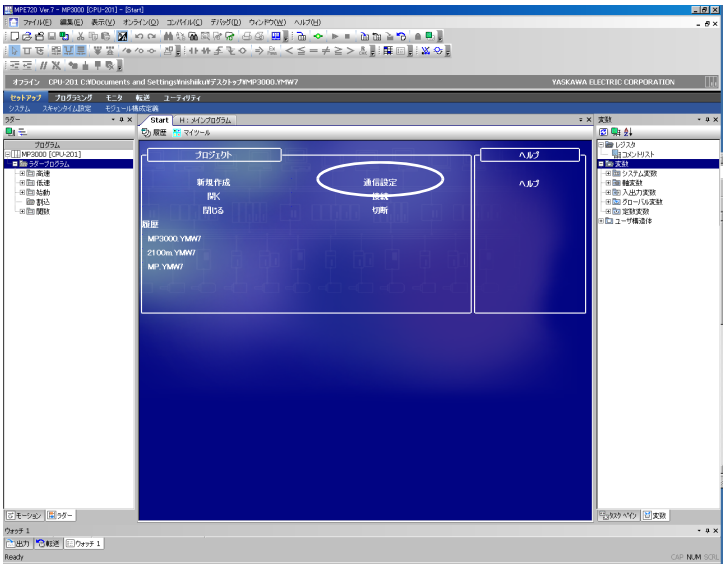
計時器電路之動作



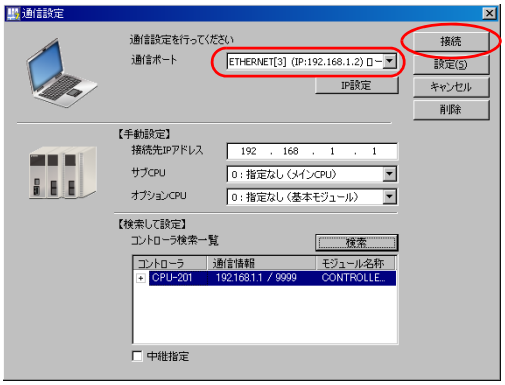
2.7 寫入階梯圖程式

請依照下列步驟，將階梯圖程式寫入運動控制器中。不過，若要在連線模式下編寫階梯圖程式，則不需要本節所述之步驟。

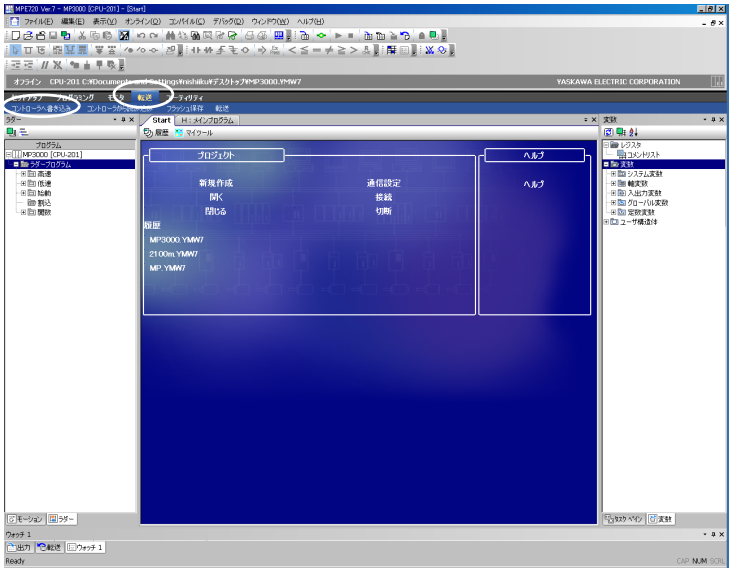
1. 找到 [開始] 分頁，並點擊 [通訊設定]。



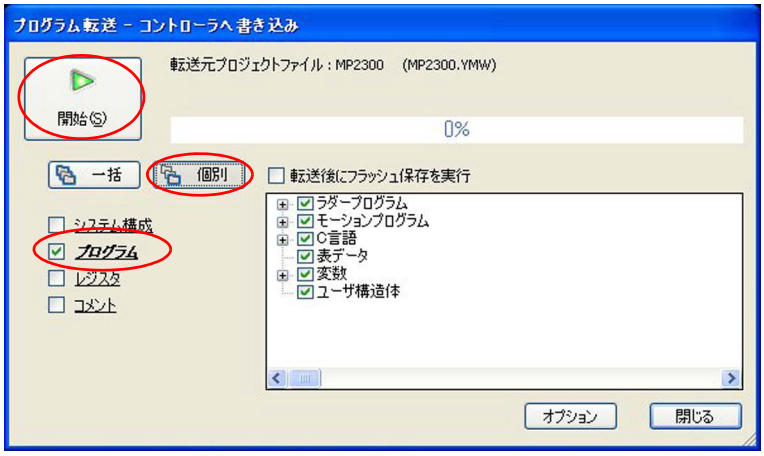
2. 從 [通訊設定] 對話框的 [通訊埠] 窗格中選擇您所設定的通訊埠，然後再點擊 [連線] 鍵。



3. 利用啟動程式，依序選擇 [傳送] － [寫入控制器]。

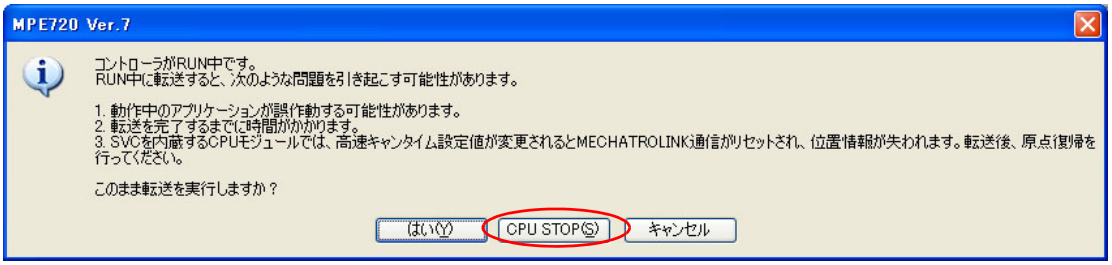


4. **點擊 [個別] 鍵，而且只要點擊 [程式] 核取方塊。請點擊 [開始] 鍵。**



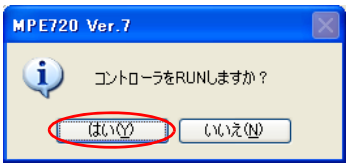
(註)1. 選擇個別傳送時，控制器裡的同一個檔案會被您所選擇的專案檔案資料所覆寫上去。
2. 若選擇全部傳送，運動控制器的 RAM 資料會在傳送前被清除，而所有的專案檔案資料則會同時被寫入。

5. **請點擊 [CPU STOP] 鍵。**



即開始傳送資料。

6. **請在以下的對話框中，點擊 [是] 鍵。**



運動控制器將會開始 RUN。

2.8 確認階梯圖程式的動作

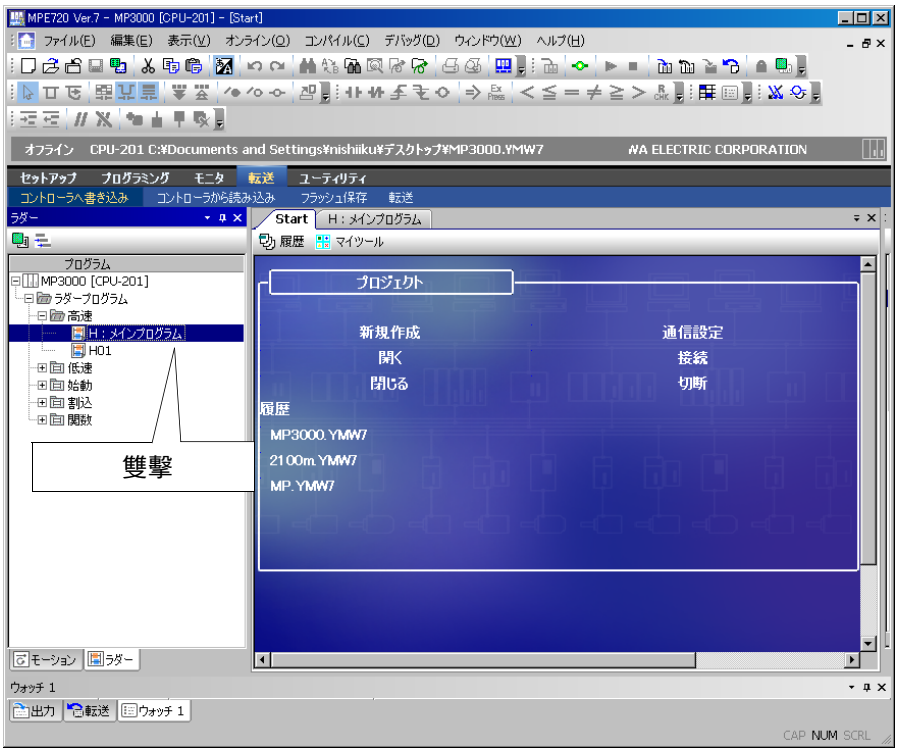
請依照以下步驟來確認編寫完成的階梯圖程式動作。

利用暫存器清單來操控暫存器，再透過暫存器清單及階梯圖編輯器的執行監控程式來確認程式動作是否正常。

動作確認的前置準備

請依照以下步驟，做好動作確認的準備。

1. 雙擊以階梯圖子視窗所編寫完成的階梯圖程式。



2. 點擊 [暫存器清單 1] 索引標籤。

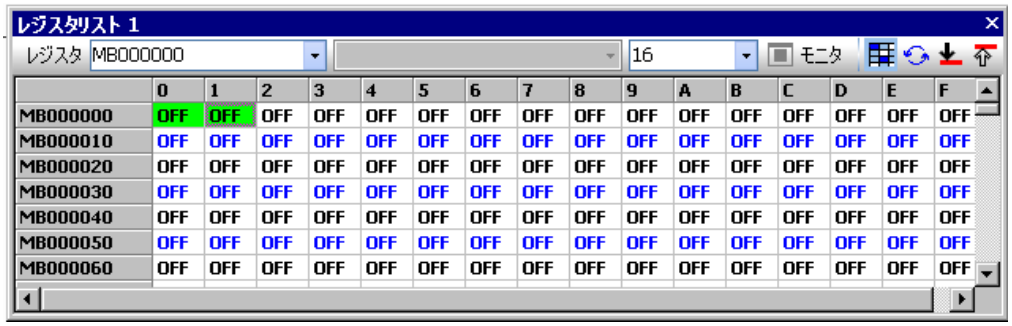
畫面上將出現 [暫存器清單 1] 對話框。

補充

[若畫面上未出現 [暫存器清單 1]，請利用以下任一種方法叫出 [暫存器清單 1] 對話框。

- 利用主選單，依序點選 [顯示] － [暫存器清單] － [暫存器清單 1]。
- 利用啟動程式，依序選擇 [監控] － [暫存器清單]。

3. **請在 [暫存器] 窗格中輸入「MB000000」。**
畫面上將會開啟以下的暫存器清單。



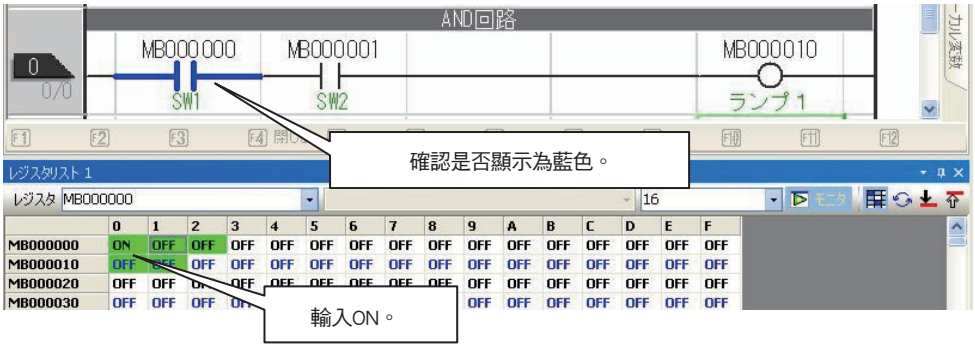
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
MB000000	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
MB000010	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
MB000020	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
MB000030	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
MB000040	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
MB000050	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
MB000060	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

確認第 0000 行 (AND 電路) 的動作

請依照以下步驟來確認第 0000 行的動作。

1. 利用暫存器清單，將 MB000000 設定為 ON。此時，請利用階梯圖編輯器確認 MB000000 的 A 接點是否顯示為藍色。

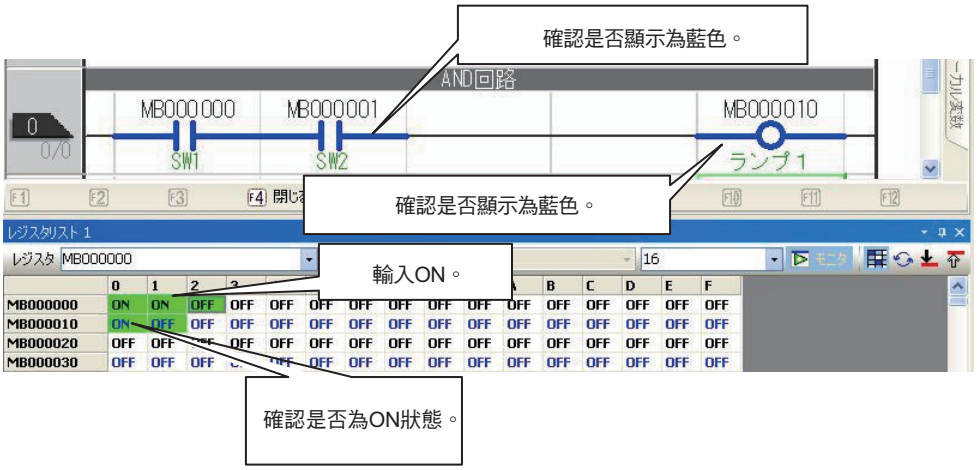
(註)若線圈或接點顯示為藍色，表示兩者目前處於 ON 狀態。



2. 利用暫存器清單，將 MB000001 設定為 ON。此時必須確認以下重點。

・ 利用階梯圖編輯器確認 MB000001 的 A 接點和 MB000010 線圈是否顯示為藍色

・ 利用暫存器清單確認 MB000010 是否處於 ON 狀態

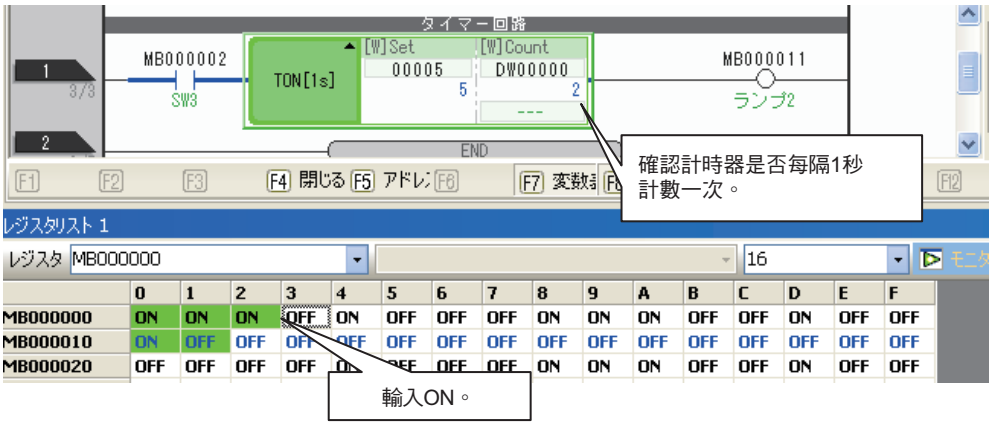


確認第 0001 行 (計時器電路) 的動作

請依照以下步驟來確認第 0001 行的動作。

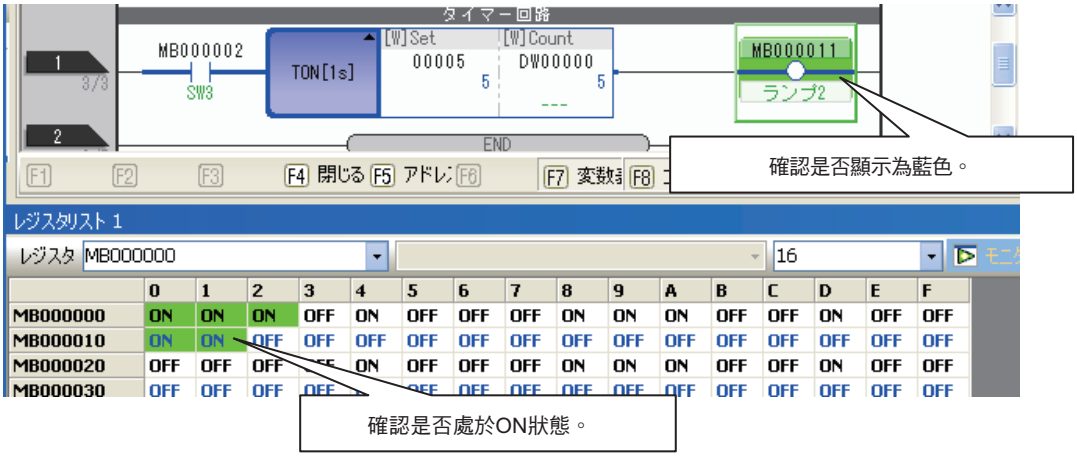
利用暫存器清單，將 MB000002 設定為 ON。此時必須確認以下重點。

・計時器 (DW00000) 必須每隔 1 秒計數一次



・5 秒後，階梯圖編輯器上的 MB000011 線圈必須顯示為藍色

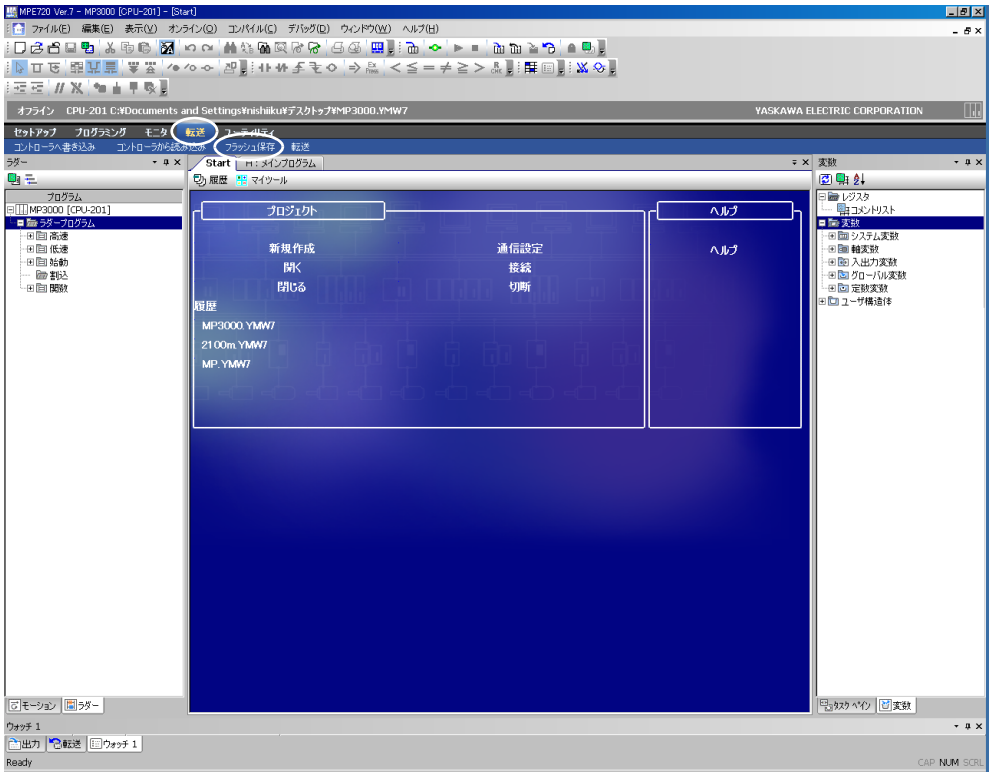
・利用暫存器清單確認 MB000011 是否處於 ON 狀態



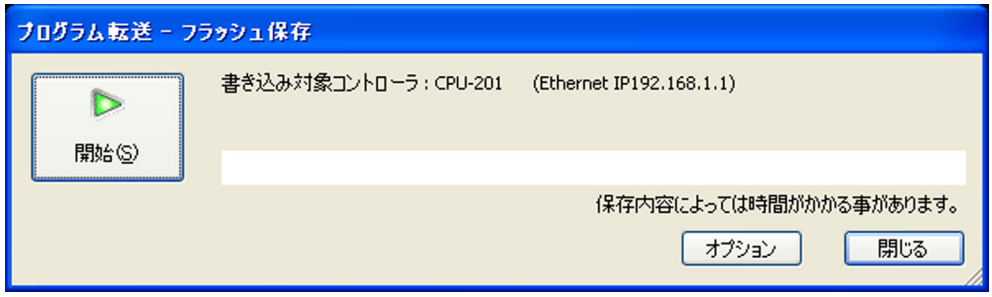
2.9 將階梯圖程式儲存至快閃記憶體

請依照以下步驟，將運動控制器 RAM 裡的資料儲存於運動控制器的快閃記憶體中。

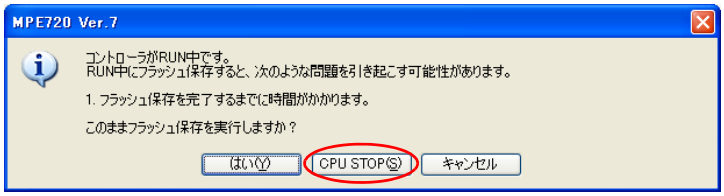
1. **利用啟動程式依序選擇 [傳送] – [儲存於快閃記憶體]。**



2. **請點擊 [開始] 鍵。**

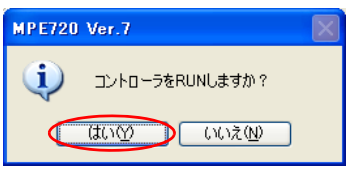


3. **請點擊 [CPU STOP] 鍵。**



程式就會被儲存在快閃記憶體中。

4. **點擊 [是] 鍵。**



運動控制器將會開始 RUN。


重要

將資料寫入運動控制器後，務必將該資料儲存於快閃記憶體中。
不儲存即再次啟動電源時，將會回復到快閃記憶體最後的儲存狀態，此時寫入的資料將會消失。

確認第 0001 行 (計時器電路) 的動作

暫存器

3

本章將針對暫存器進行相關說明。

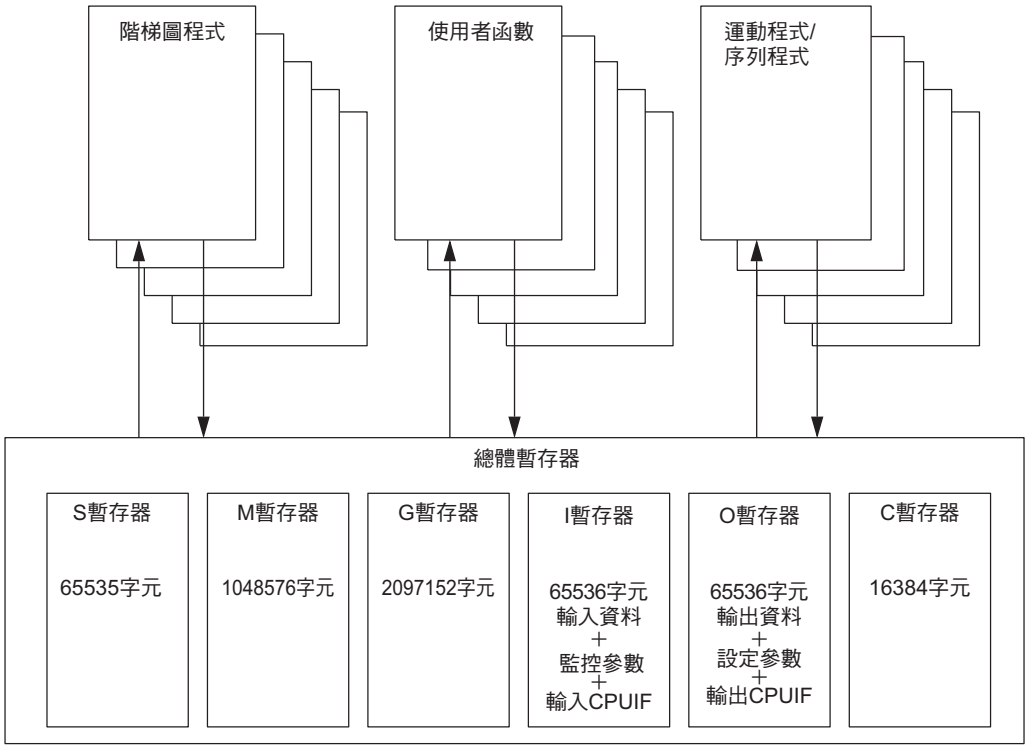
暫存器

所謂「暫存器」就是運動控制器內部用來儲存資料的一個區域。所謂「變數」就是暫存器之對應名稱 (變數名稱)。

暫存器可分為所有程式皆可共用的總體暫存器，以及適合所有程式單獨使用的局部暫存器。

總體暫存器

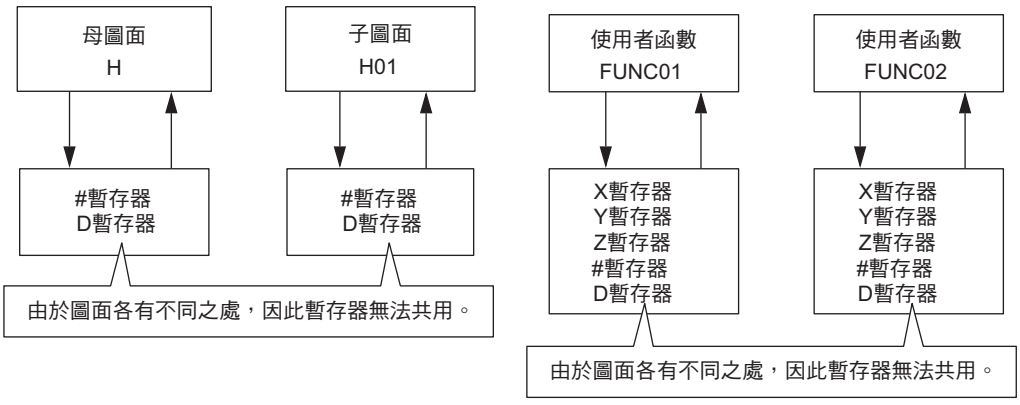
「總體暫存器」就是階梯圖程式、使用者函數、運動程式和序列程式等各種程式皆可共用的一種暫存器。總體暫存器的大小依暫存器不同，由系統各自決定。



局部暫存器

「局部暫存器」就是適用於各種圖面的一種暫存器。其他圖面無法用來啟動暫存器。

< 階梯圖程式示意圖 >



暫存器編號的架構

暫存器編號= S W 00000
- 00000：範圍(位數依暫存器類型、資料類型各有不同)。
- W：資料類型
- S：暫存器類型

補充　可以組合作為變數使用的索引暫存器或陣列暫存器，亦可用來指定暫存器。
如欲進一步瞭解，請參閱以下項目之說明。

☞ 索引暫存器 (i、j)(第 3-10 頁)
☞ 陣列暫存器 ([ ])(第 3-12 頁)

暫存器類型

接下來將說明總體暫存器及局部暫存器的各種類型。

◆ 總體暫存器

「總體暫存器」就是階梯圖程式、使用者函數、運動程式和序列程式等各種程式皆可共用的一種暫存器。換句話說，也就是某個階梯圖程式的演算結果將適用於其他使用者函數、運動程式及序列程式等。

類型	名稱	指定方法	適用範圍	內容
S	系統暫存器 (S 暫存器)	SBnnnnnh， SWnnnnn， SLnnnnn， SQnnnnn， SFnnnnn， SDnnnnn， SAnnnnn	SW00000 ～ SW65534	系統所預設之暫存器，可用來報告運動控制器的狀態等。 系統啟動時，SW00000 ~ SW00049 將會被歸零。 將以電池組備用。
M	資料暫存器 (M 暫存器)	MBnnnnnnnh， MWnnnnnnn， MLnnnnnnn， MQnnnnnnn， MFnnnnnnn， MDnnnnnnn， MAnnnnnnn	MW0000000 ～ MW1048575	適用於不同程式 I/F 的暫存器。 將以電池組備用。
G	G 暫存器	GBnnnnnnnh, GWnnnnnnn, GLnnnnnnn, GQnnnnnnn, GFnnnnnnn, GDnnnnnnn, GAnnnnnnn,	GW0000000 ～ GW2097151	適用於不同程式 I/F 的暫存器。 未以電池組備用。
I	輸入暫存器 (I 暫存器)	IBhhhhhh， IWhhhhh， ILhhhhh， IQhhhhh， IFhhhhh， IDhhhhh， IAhhhhh，	IW00000 ～ IW07FFF, IW10000 ～ IW17FFF	適用於輸入資料的暫存器。
			IW08000 ～ IW0FFFF, IW18000 ～ IW1FFFF	為運動監控參數。 適用於運動模組的暫存器。
			IW20000 ～ IW23FFF	適用於 CPUIF 輸入資料的暫存器。

(續下頁)

(續上頁)

類型	名稱	指定方法	適用範圍	內容
O	輸出暫存器 (O 暫存器)	OBhhhhhh， OWhhhhh， OLhhhhh， OQhhhhh， OFhhhhh， ODhhhhh， OAhhhhh，	OW00000 ～ OW07FFF, OW10000 ～ OW17FFF	適用於輸出資料的暫存器。
			OW08000 ～ OW0FFFF, OW18000 ～ OW1FFFF	為運動設定參數。 適用於運動模組的暫存器。
			OW20000 ～ OW23FFF	適用於 CPUIF 輸出資料的暫存器。
C	常數暫存器 (C 暫存器)	CBnnnnnh， CWnnnnn， CLnnnnn， CQnnnnn, CFnnnnn， CDnnnnn, CAnnnnn	CW00000 ～ CW16383	僅能在程式內部參照的暫存器。 利用 MPE720 即可設定數值。

(註)n：10 進位制、h：16 進位制

◆ 局部暫存器

每個程式所預設的暫存器。無法參照其他程式的局部暫存器。

適用範圍必須由使用者利用 MPE720 來指定。

類型	名稱	指定方法	內容	特性
#	# 暫存器	#Bnnnnnh， #Wnnnnn， #Lnnnnn， #Qnnnnn， #Fnnnnn， #Dnnnnn， #Annnnn	僅能在程式內部參照的暫存器。 利用 MPE720 即可設定數值。	單一程式
D	D 暫存器	DBnnnnnh， DWnnnnn, DLnnnnn， DQnnnnn， DFnnnnn， DDnnnnn， DAnnnnn	程式內部所通用的暫存器。 預設值為每個程式可儲存 32 個字元。 重新啟動電源時之初始值取決於 [啟動時清除 D 暫存器] 選項之設定內容。 如欲進一步瞭解，請參閱以下項目之說明。 ☞ **•[啟動時清除 D 暫存器] 選項設定方法** (**第 3-6 頁**)	
X	函數輸入暫存器	XBnnnnnh， XWnnnnn， XLnnnnn， XQnnnnn， XFnnnnn， XDnnnnn	輸入函數 ・輸入位元：XB000000 ~ XB00000F ・輸入整數：XW00001 ~ XW00016 ・長整數：XL00001 ~ XL00015 ・4 長整數：XQ00001 ~ XQ00013 ・實數：XF00001 ~ XF00015 ・倍精度實數：XD00001 ~ XD00013	單一函數
Y	函數輸出暫存器	YBnnnnnh， YWnnnnn, YLnnnnn， YQnnnnn， YFnnnnn， YDnnnnn	從函數輸出 ・輸出位元：YB000000 ~ YB00000F ・輸出整數：YW00001 ~ YW00016 ・長整數：YL00001 ~ YL00015 ・4 長整數：YQ00001 ~ YQ00013 ・實數：YF00001 ~ YF00015 ・倍精度實數：YD00001 ~ YD00013	

(續下頁)

(續上頁)

類型	名稱	指定方法	內容	特性
Z	函數內部暫存器	ZBnnnnnh， ZWnnnnn, ZLnnnnn， ZQnnnnn， ZFnnnnn， ZDnnnnn	每個函數所預設的內部暫存器。適合用來執行函數內部處理作業。 ・位元：ZB000000 ~ ZB00063F ・整數：ZW00000 ~ ZW00063 ・長整數：ZL00000 ~ ZL00062 ・4 長整數：ZQ00000 ~ ZQ00060 ・實數：ZF00000 ~ ZF00062 ・倍精度實數：ZD00000 ~ ZD00060	單一函數
A	函數外部暫存器	ABnnnnnh， AWnnnnn, ALnnnnn， AQnnnnn， AFnnnnn， ADnnnnn	將位址輸入值視為基底位址的外部暫存器。 為函數**叫出**來源，只要附加 M 暫存器及 D 暫存器的位址輸入值，函數內部即可根據該位址來參照函數**叫出**來源的暫存器。	

(註)n：10 進位制、h：16 進位制

重要

不同的程式可多次參照使用者函數。

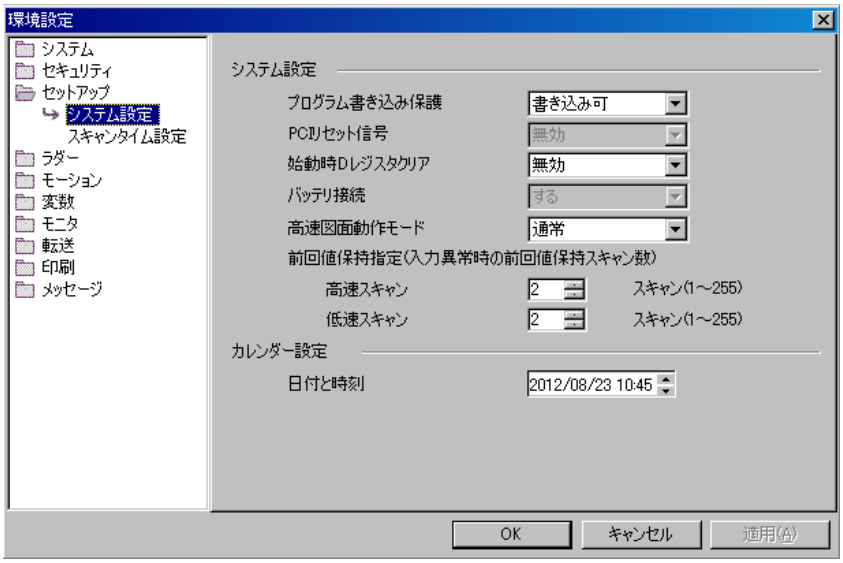
■ 使用者函數內局部暫存器之使用注意事項

叫出使用者函數時，應考量局部暫存器的現在值，然後再進行初始化等處理作業。

名稱	注意事項
X 暫存器 (函數輸入暫存器)	若未設定輸入值，將造成數值不穩定。 請勿使用超出輸入定義功能指定範圍的 X 暫存器。
Y 暫存器 (函數輸出暫存器)	若未設定輸出值，將造成數值不穩定。 Y 暫存器符合輸出定義功能所指定的範圍，因此必須設定好數值。
Z 暫存器 (函數內部暫存器)	**叫出**函數時，前一次的設定值將會消失，並造成數值不穩定的情形。 此函數不適合必須維持前一次數值的指令。 使用前請先設定好函數內的初始值。
# 暫存器	本函數為常數暫存器，因此數值不會改變。
D 暫存器	**叫出**函數時，也會記憶前一次設定值。 若不使用前一次的數值，就必須設定初始值，或是使用 Z 暫存器。 D 暫存器會一直保存資料直到電源關閉為止。 重新啟動電源時之初始值取決於 [啟動時清除 D 暫存器] 等選項之設定內容。如欲進一步瞭解，請參閱以下項目之說明。 ☞ **•[啟動時清除 D 暫存器] 選項設定方法 (第 3-6 頁)**

· **[啟動時清除 D 暫存器] 選項設定方法**

1. **請在 MPE720 Ver. 7 視窗中，依序選擇 [檔案] － [環境設定]。**
2. **點選 [設定] － [系統設定]。**
3. **選擇啟動時清除 D 暫存器功能 [關閉 / 開啟]。**
   關閉：初始值不定
   開啟：初始值 0



資料類型

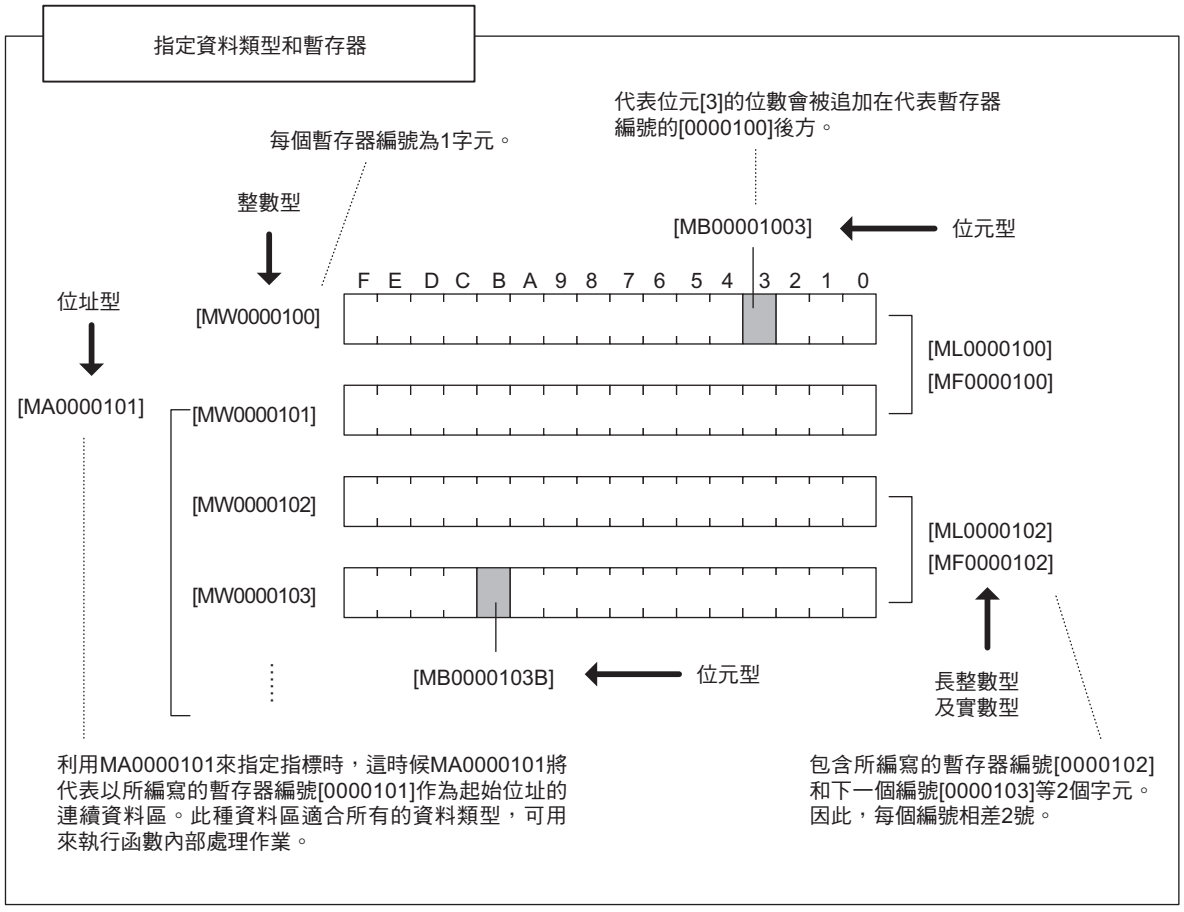
資料類型依目的不同，可分為下表所示的位元型、整數型、長整數型、4 長整數型、實數型、倍精度實數型及位址型等。

符號	資料類型	數值範圍	資料大小	備註
B	位元	1 (ON)、0 (OFF)	–	用來判定繼電器電路或 ON/OFF 條件。
W	整數	-32,768 ～ 32,767 (8000H ～ 7FFFH)	1 字元	作為數值演算之用。左欄 ( ) 內所示為使用邏輯演算方式時之數值。
L	長整數	-2,147,483,648 ～ 2,147,483,647 (80000000H ～ 7FFFFFFFH)	2 字元	作為數值演算之用。左欄 ( ) 內所示為使用邏輯演算方式時之數值。
Q	4 長整數 *1	-9223372036854775808 ～ 9223372036854775807 （8000000000000000H ～ 7FFFFFFFFFFFFFFFH）	4 字元	作為數值演算之用。左欄 ( ) 內所示為使用邏輯演算方式時之數值。
F	實數	± (1.175E-38 ～ 3.402E+38)，0	2 字元	適合高階的數值演算用途。*2
D	倍精度實數 *1	± (2.225E-308 ～ 1.798E+308)，0	4 字元	適合高階的數值演算用途。*2
A	位址	0 ～ 2,097,152	–	僅使用於指定指標。

*1. 不適合用來間接指定運動程式。
*2. 符合 IEEE754 規範。

重要

MP3000 系列並未根據不同的資料類型內置適用的暫存器。如下圖所示，只要位址相同，即使資料類型不同，也能存取同一個暫存器。
例如，位元型 MB00001003 和整數型 MW0000100 的資料類型雖然不同，卻同樣能存取 MW0000100。



專有名詞解說

指定指標

將位址當作引數傳給函數，就稱為「指定指標」。

指定指標後，以所編寫的暫存器編號作為起始位址的連續資料區將適合所有的資料類型，作為函數內部處理之用。

◆ 不同資料類型之演算注意事項

對不同資料類型進行演算時，結果將依所儲存的暫存器資料類型而有以下差異，使用時請特別注意。

· **將實數型資料儲存於整數型暫存器**

MW0000100 = MF0000200；將實數值轉換為整數並加以儲存。

(00001)　　(1.234)

(註)將實數資料儲存於整數型暫存器時，需特別注意捨入誤差。

將實數化為整數時之捨入方法 (無條件捨去 / 四捨五入) 必須利用圖面的性質設定功能。

☞ ■ **實數轉換 (CAST) 時的動作設定 (第 3-9 頁)**

MW0000100 = MF0000200 + MF0000202；

(0124)　(123.48)　(0.02)　演算結果依所演算的變數值而異。

(0123)　(123.49)　(0.01)

· **將實數型資料儲存於長整數型暫存器**

ML0000100 = MF0000200； 將實數值轉換為整數並加以儲存。

(65432)　(65432.1)

· **將長整數型資料儲存於整數型暫存器**

MW0000100 = ML0000200；將直接儲存長整數型資料的低階 16 位元。

(-00001)　(65535)

· **將整數型資料儲存於長整數型暫存器**

ML0000100 = MW0000200；整數型資料會被轉換為長整數型資料再加以儲存。

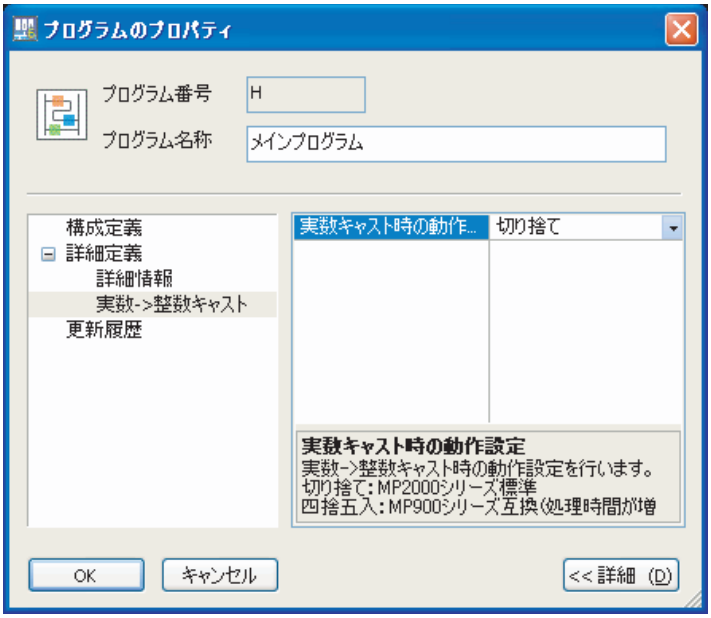
(0001234)　(1234)

■ 實數轉換 (CAST) 時的動作設定

利用程式的性質詳細定義功能，即可設定實數轉換 (CAST) 時之動作 (無條件捨去 / 四捨五入)。
請依不同圖面分別設定實數轉換 (CAST) 時之動作。

下圖所示為 [程式性質] 對話框的顯示步驟。

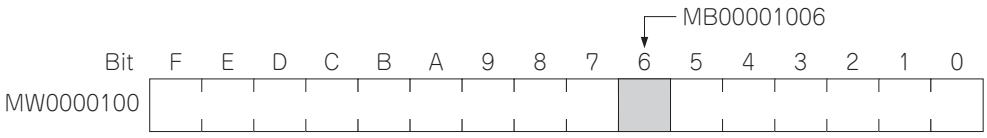
1. **進入階梯圖子視窗，即可選擇您要顯示出內容的階梯圖程式檔案。**
2. **在您所選擇的程式上按一下右鍵，然後再從畫面上出現的彈出式選單中選擇 [性質]。**
   畫面上將出現 [程式性質] 對話框。



補充

以下範例係以「 little-endian」的方式來表示資料排列結構。

・ MB00001006 的情形



・ MW0000100 = 1234H 的情形

MW0000100	1234H

・ ML0000100 = 12345678H 的情形

MW0000100	5678H	ML0000100
MW0000101	1234H	

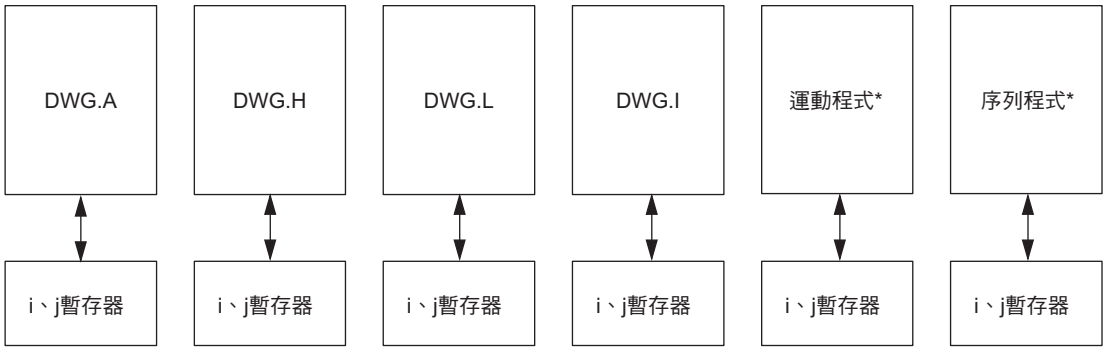
・ MQ0000100 = 123456789ABCDEF0H 的情形

MW0000100	DEF0H	MQ0000100
MW0000101	9ABCH	
MW0000102	5678H	
MW0000103	1234H	

索引暫存器 (i、j)

這是一種用來修飾繼電器編號及暫存器編號的專用暫存器，內置 i 和 j 等 2 種暫存器。i 和 j 的功能完全相同。暫存器編號可當作變數使用。

下圖所示為不同程式類型的索引暫存器。



* 依任務不同，運動程式及序列程式可分別建置 i 或 j 暫存器。

(註)函數可參照您所叫出的圖面專用 i、j 暫存器。
例如，由 DWG.H 叫出的函數可參照 DWG.H 專用的 i、j 暫存器。

接下來將針對不同的暫存器資料類型，舉例說明索引暫存器的動作。

◆ 位元型附加索引

就像暫存器編號被加上 i 或 j 的數值一樣。

例如，當 i = 2 時，MB00000000i 和 MB00000002 相同。

```
i = 2;
DB000000 = MB00000000i;   等價 ⟷   DB000000 = MB00000002;
```

◆ 整數型附加索引

就像暫存器編號被加上 i 或 j 的數值一樣。

例如，當 j = 30 時，MW0000001j 和 MW0000031 相同。

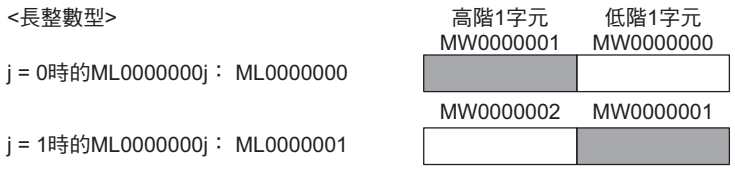
```
j = 30;
DW00000 = MW0000001j;   等價 ⟷   DW00000 = MW0000031;
```

◆ 長整數型及實數型附加索引

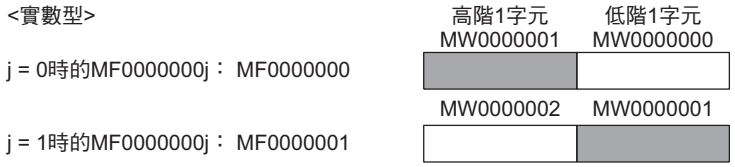
就像暫存器編號被加上 i 或 j 的數值一樣。

例如，當 j = 1 時，ML0000000j 和 ML0000001 相同。此外，當 j = 1 時，MF0000000j 和 MF0000001 相同。

<長整數型>



<實數型>



註記

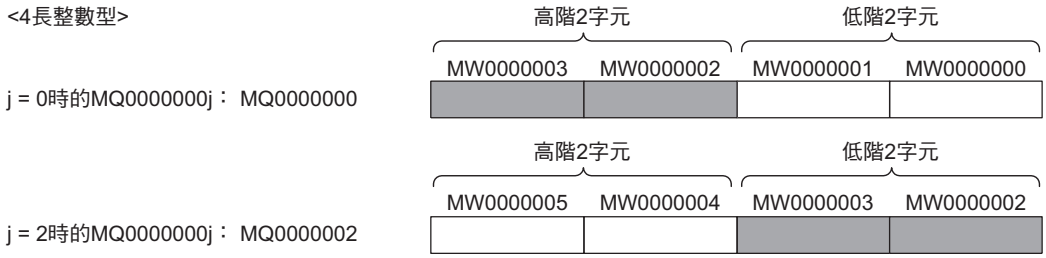
若資料為長整數型或實數型時，所使用的區域相當於 2 字元。例如，j = 0 時的 ML0000000j 和 j = 1 時的 ML0000000j，兩者的資料區將和 MW0000001 的 1 字元互相重複。
若長整數型和實數型附加索引，則必須注意資料區是否重複。

◆ 4 長整數型及倍精度實數型附加索引

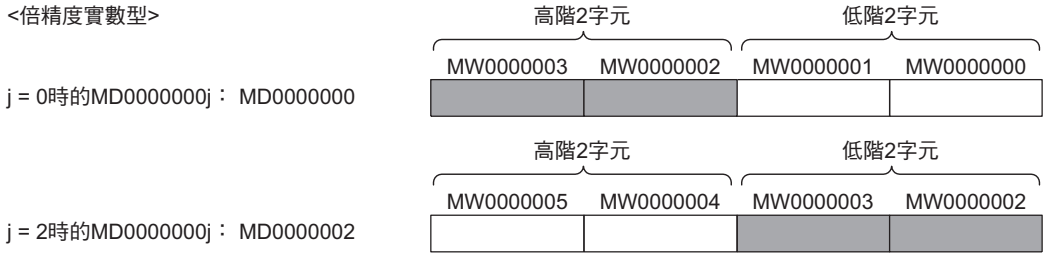
就像暫存器編號被加上 i 或 j 的數值一樣。

例如，當 j = 2 時，MQ0000000j 和 MQ0000002 相同。此外，當 j = 2 時，MD0000000j 和 MD0000002 相同。

<4長整數型>



<倍精度實數型>



註記

若資料為 4 長整數型或倍精度實數型時，所使用的區域相當於 4 字元。例如，j = 0 時的 MQ0000000j 和 j = 2時的MQ0000000j，兩者的資料區將和MW0000002、MW0000003的2字元互相重複。若4長整數型和倍精度實數型附加索引，則必須注意資料區是否重複。

陣列暫存器 ([ ])

所謂「陣列暫存器」就是用來修飾暫存器編號的一種專用暫存器，以符號 [ ] 來表示。

適合以暫存器編號為變數時使用。

暫存器編號像索引暫存器一樣，會被加上偏移值。

◆ 為位元型附加陣列

就像暫存器編號被加上陣列數值一樣。

例如，當 DW00000 = 2 時，MB00000000 [DW00000] 和 MB00000002 相同。

DW00000 = 2;

DB000020 = MB00000000[DW00000]; 等價 DB000020 = MB00000002;

◆ 為位元型以外類型附加陣列

就像暫存器編號被加上 (陣列數值 x 資料型字元大小) 一樣。

例如，當 DW00000 = 30 時，ML0000002 [DW00000] 和 ML0000062 相同。

DL00002 = ML00000 (30×2 + 2) = ML0000062

DW00000 = 30;

DL00002 = ML0000002[DW00000]; 等價 DL00002 = ML0000062;

階梯圖程式語言指令

4

本章將針對階梯圖語言指令的詳細內容進行說明。

4.1 概要

本節將針對階梯圖語言指令的類型、功能及指令判讀方法等加以說明。

階梯圖語言指令一覽表

下表為各種階梯圖語言指令。

類型	指令	代表意義
繼電器電路指令	NOC	A 接點
	ONP-NOC	上升 A 接點
	OFFP-NOC	下降 A 接點
	NCC	B 接點
	ONP-NCC	上升 B 接點
	OFFP-NCC	下降 B 接點
	TON (1 ms)	通電延遲計時器 (1 ms)
	TOFF (1 ms)	斷電延遲計時器 (1 ms)
	TON (10 ms)	通電延遲計時器 (10 ms)
	TOFF (10 ms)	斷電延遲計時器 (10 ms)
	TON (1 s)	通電延遲計時器 (1 s)
	TOFF (1 s)	斷電延遲計時器 (1 s)
	ON-PLS	上升脈衝
	OFF-PLS	下降脈衝
	COIL	線圈
	REV-COIL	反轉型線圈
	ONP-COIL	上升變化檢測用線圈
	OFFP-COIL	下降變化檢測用線圈
	S-COIL	設定線圈
	R-COIL	重置線圈
數值運算指令	STORE	儲存
	ADD(+)	加法
	ADDX(+ +)	加法擴充
	SUB (–)	減法
	SUBX(– –)	減法擴充
	MUL(×)	乘法
	DIV(÷)	除法
	MOD	整數型餘數
	REM	實數型餘數
	INC	遞增
	DEC	遞減
	TMADD	增加時間
	TMSUB	減少時間
	SPEND	經過時間
	INV	符號反轉
	COM	1 的補數
	ABS	絕對值轉換
	BIN	轉換為 2 進位制
	BCD	BCD 轉換
	PARITY	同位轉換
	ASCII	ASCII 轉換 1
	BINASC	ASCII 轉換 2
	ASCBIN	ASCII 轉換 3

(續下頁)

(續上頁)

類型	指令	代表意義
邏輯運算指令	AND	邏輯積
	OR	邏輯和
	XOR	互斥或
	<	比較
	≦	比較
	=	比較
	≠	比較
	≧	比較
	>	比較
	RCHK	檢查範圍
程式控制指令	SEE	叫出圖面
	MSEE	叫出運動程式
	FUNC	叫出使用者函數
	INS	連續執行型直接輸入
	OUTS	連續執行型直接輸出
	XCALL	執行擴充程式
	WHILE END_WHILE	WHILE 陳述式
	FOR END_FOR	FOR 陳述式
	IF END_IF	IF 陳述式
	IF ELSE END_IF	IF-ELSE 陳述式
	EXPRESSION	算式編寫
基本函數指令	SQRT	平方根
	SIN	正弦
	COS	餘弦
	TAN	正切
	ASIN	反正弦
	ACOS	反餘弦
	ATAN	反正切
	EXP	指數
	LN	自然對數
	LOG	常用對數
資料操控指令	ROTL	位元向左旋轉
	ROTR	位元向右旋轉
	MOVB	位元傳送
	MOVW	字元傳送
	XCHG	置換傳送
	SETW	資料表初始化
	BEXTD	位元組 → 字元展開
	BPRESS	字元 → 位元組壓縮
	BSRCH	資料檢索
	SORT	排序
	SHFTL	位元左移
	SHFTR	位元右移
	COPYW	複製字元
	BSWAP	位元組調換

(續下頁)

(續上頁)

類型	指令	代表意義
DDC 指令	DZA	死區 A
	DZB	死區 B
	LIMIT	上下限值
	PI	PI 控制
	PD	PD 控制
	PID	PID 控制
	LAG	1 次延遲
	LLAG	相位前進延遲
	FGN	函數產生器
	IFGN	反函數產生器
	LAU	直線加速減速器 1
	SLAU	直線加速減速器 2
	PWM	脈衝幅度調變
資料表操控指令	TBLBR	讀取區塊
	TBLBW	寫入區塊
	TBLSRL	行搜尋 (垂直方向)
	TBLSRC	列搜尋 (水平方向)
	TBLCL	刪除區塊
	TBLMV	傳送資料表區塊
	QTBLR	讀取佇列表
	QTBLRI	讀取佇列表 (指標步進)
	QTBLW	寫入佇列表
	QTBLWI	寫入佇列表 (指標步進)
	QTBLCL	清除佇列指標
系統標準函數指令	COUNTER	計數器
	FINFOUT	先進 / 先出
	TRACE	追蹤
	DTRC-RD	讀取資料追蹤
	MSG-SND	傳送訊息
	MSG-SNDE	傳送訊息 (擴充)
	MSG-RCV	接收訊息
	MSG-RCVE	接收訊息 (擴充)
	MLNK-SVW	將參數寫入伺服驅動器
	MOTREG-W	寫入運轉暫存器
	MOTREG-R	讀取運轉暫存器資料
	IMPORT/IMPORTL	匯入
	EXPORT/EXPORTL	匯出

階梯圖語言指令的判讀方法

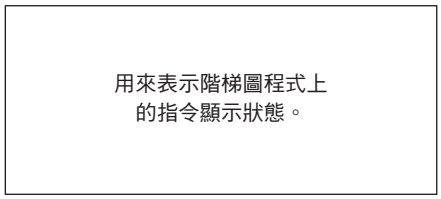
本章將採用以下結構來說明各種指令。

同時說明指令的動作。

如有需要，將以圖示方式來說明指令的動作。

格式

以下將說明指令的格式。



圖示：表示MPE720上的圖示。

按鍵輸入：表示階梯圖程式編輯器所使用的快捷鍵。

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸出輸入項目 (階梯圖程式所使用之標示法)	×	○	○	×	○	×	×	○	○

(註)1. × … 該資料類型不適用。
○… 該資料類型可適用於所有的暫存器。
2. 如欲瞭解資料類型之相關說明，請參閱以下章節。
☞ 第 3 章 暫存器

程式範例

表示使用各種指令之階梯圖程式範例。

補充事項

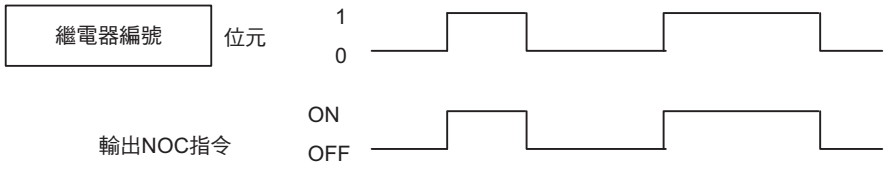
需要補充說明之事項。若該指令不需要特別補充說明，則不會刊載此事項。

4.2 繼電器電路指令

A 接點 (NOC)

當繼電器編號的位元為 1 時，繼電器將輸出 ON。

位元為 0 時，繼電器將輸出 OFF。



格式

所採用之格式如下。

圖示 ：

按鍵輸入 ：][

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
繼電器編號	○	×	×	×	×	×	×	×	×

程式範例

使用 NOC 指令讓繼電器 (DB000000) ON 時，輸出線圈 (DB000001) 也會變為 ON。

上升 A 接點 (ONP-NOC)

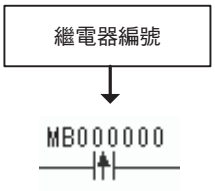
當位元輸入由 0 變成 1 時，只有在執行一次掃描時，才會輸出 ON。

NOC 指令與 ON-PLS 指令會互相搭配執行動作。

補充　所執行的動作和 OFFP-NCC 指令相同。

格式

所採用之格式如下。



圖示　：

按鍵輸入 ：]P[

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
繼電器編號	○*	×	×	×	×	×	×	×	×

* #，C 暫存器屬於常數暫存器，數值不會改變，因此也不會執行標的動作。

程式範例

使用 NOC 指令讓繼電器 (DB000000) 由 OFF 變為 ON 時，輸出線圈 (DB000001) 也會變為 ON。

下降 A 接點 (OFFP-NOC)

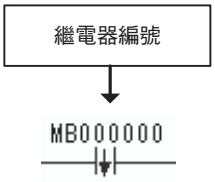
當位元輸入由 1 變成 0 時，只有在執行一次掃描時，才會輸出 ON。

NOC 指令與 OFF-PLS 指令會互相搭配執行動作。

補充　所執行的動作和 ONP-NCC 指令相同。

格式

所採用之格式如下。



圖示　：

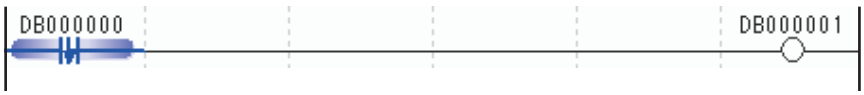
按鍵輸入：]N[

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
繼電器編號	○*	×	×	×	×	×	×	×	×

* #，C 暫存器屬於常數暫存器，數值不會改變，因此也不會執行標的動作。

程式範例

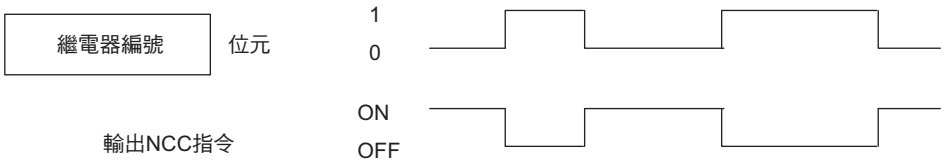
使用 NOC 指令讓繼電器 (DB000000) 由 ON 變為 OFF 時，輸出線圈 (DB000001) 將變為 ON。



B 接點 (NCC)

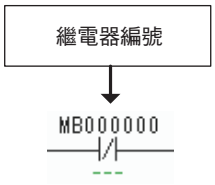
當繼電器編號的位元為 1 時，繼電器將輸出 OFF。

位元為 0 時，繼電器將輸出 ON。



格式

所採用之格式如下。



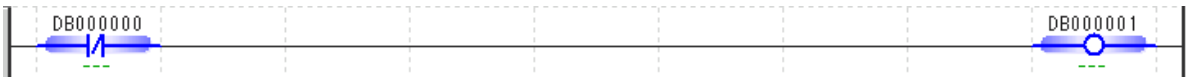
圖示　：

按鍵輸入：]/

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
繼電器編號	○	×	×	×	×	×	×	×	×

程式範例

使用 NCC 指令讓繼電器 (DB000000) 變為 OFF 時，線圈 (DB000001) 將變為 ON。



上升 B 接點 (ONP-NCC)

當位元輸入由 1 變成 0 時，只有在執行一次掃描時，才會輸出 ON。

NCC 指令會和 ON-PLS 指令互相搭配執行動作。

補充　所執行的動作和 OFFP-NOC 指令相同。

格式

所採用之格式如下。

圖示　：

按鍵輸入：]P/

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
繼電器編號	○*	×	×	×	×	×	×	×	×

* #，C 暫存器屬於常數暫存器，數值不會改變，因此也不會執行標的動作。

程式範例

使用 NCC 指令讓繼電器 (DB000000) 由 ON 變為 OFF 時，輸出線圈 (DB000001) 將變為 ON。

下降 B 接點 (OFFP-NCC)

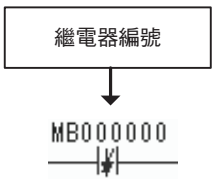
當位元輸入由 0 變成 1 時，只有在執行一次掃描時，才會輸出 ON。

NCC 指令與 OFF-PLS 指令會互相搭配執行動作。

補充　所執行的動作和 ONP-NOC 指令相同。

格式

所採用之格式如下。



圖示　：

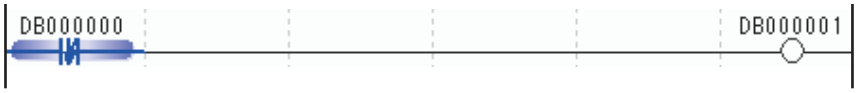
按鍵輸入：]N/

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
繼電器編號	○*	×	×	×	×	×	×	×	×

* #，C 暫存器屬於常數暫存器，數值不會改變，因此也不會執行標的動作。

程式範例

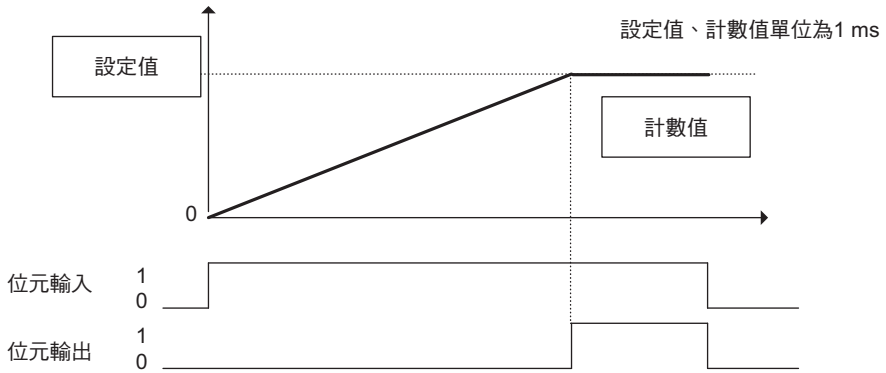
使用 NCC 指令讓繼電器 (DB000000) 由 OFF 變為 ON 時，輸出線圈 (DB000001) 也會變為 ON。



通電延遲計時器 (TON (1 ms))

當計時器的位元輸入為 1 時，就會執行時間計數。當計數值等於設定值時，位元輸出就會變為 1。

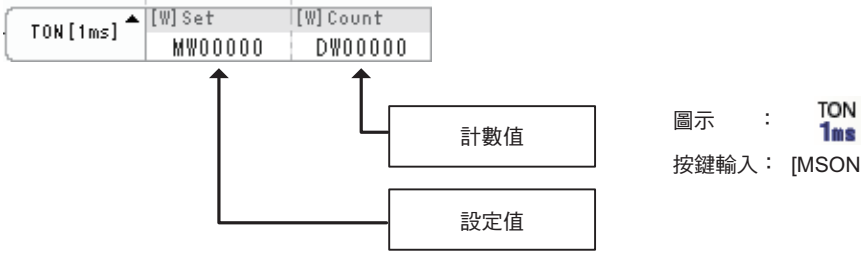
若在計數過程中，位元輸入變為 0，則計時器將停止動作。若位元輸入再次變為 1，將重新 (0) 開始計數。此外，實際的計數時間 (單位為 1 ms) 值將會被儲存在計數專用暫存器中。



(註)計算誤差小於 1 ms。

格式

所採用之格式如下。



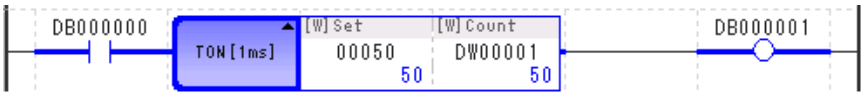
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
設定值 (Set)	×	○	×	×	×	×	×	×	○
計數值 (Count)	×	○*	×	×	×	×	×	×	×

* C、# 暫存器除外

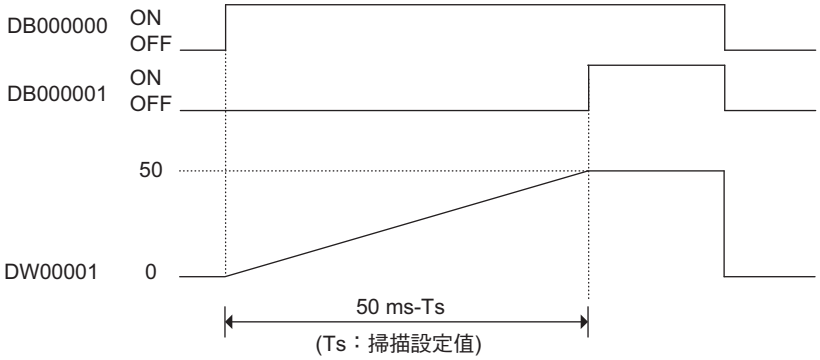
程式範例

以下所示為通電延遲計時器的設定值被設定為 50，且將計數值儲存於 DW00001 時之程式範例。

當繼電器 (DB000000) ON 50 ms 後，線圈 (DB000001) 就會 ON。



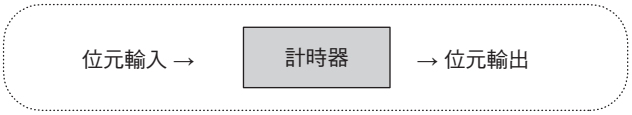
以下為時序圖。

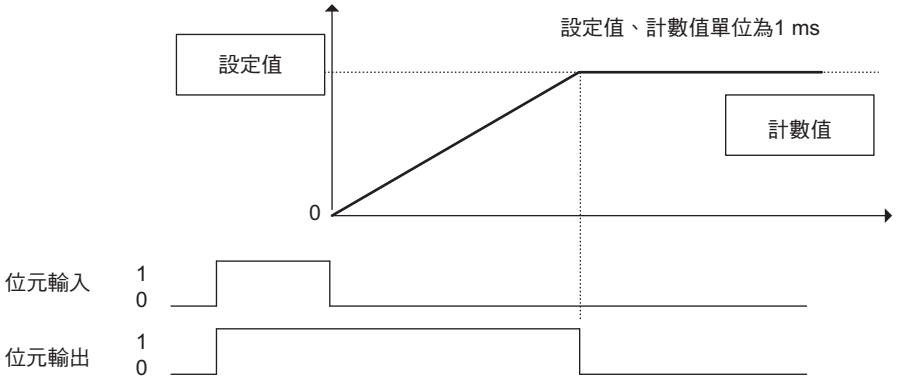


斷電延遲計時器 (TOFF (1 ms))

當計時器的位元輸入為 0 時，就會執行時間計數。當計數值等於設定值時，位元輸出就會變為 0。

若在計數過程中，位元輸入變為 0，則計時器將停止動作。若位元輸入再次變為 1，將重新 (0) 開始計數。此外，實際的計數時間 (單位為 1 ms) 值將會被儲存在計數專用暫存器中。

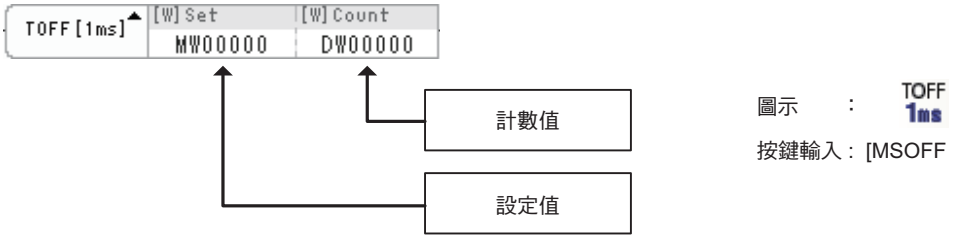




(註)計算誤差小於 1 ms。

格式

所採用之格式如下。



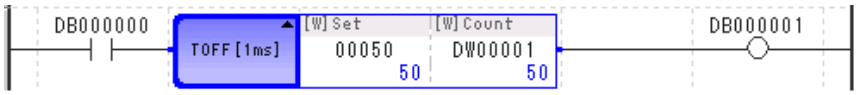
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
設定值 (Set)	×	○	×	×	×	×	×	×	○
計數值 (Count)	×	○*	×	×	×	×	×	×	×

* C、# 暫存器除外

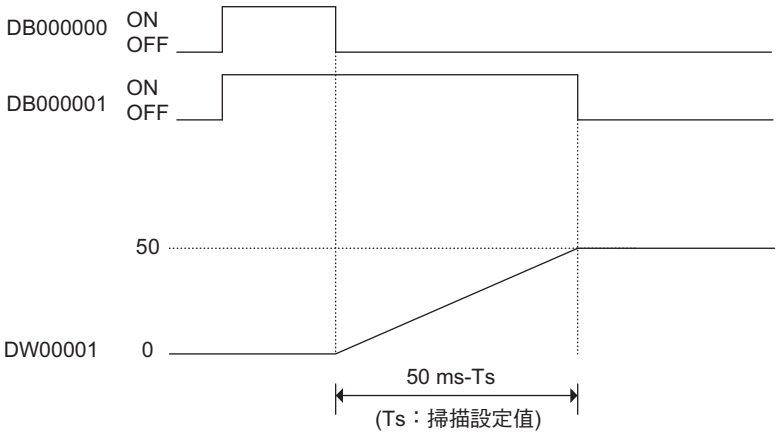
程式範例

以下所示為斷電延遲計時器被設定為 50，且將計數值儲存於 DW00001 時之程式範例。

當繼電器 (DB000000) OFF 50 ms 後，線圈 (DB000001) 就會 OFF。



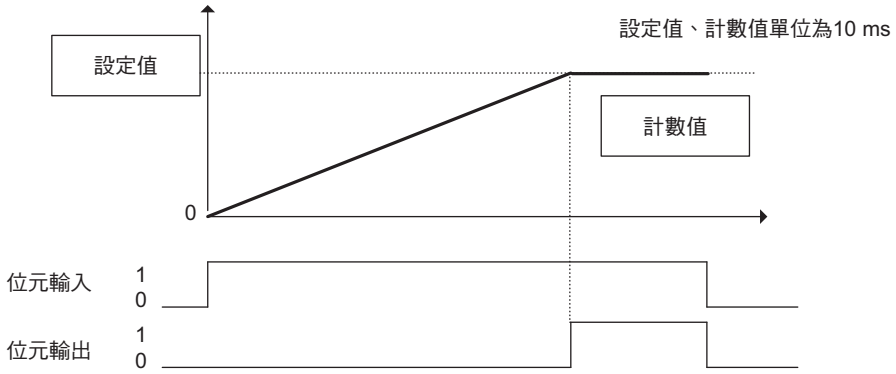
以下為時序圖。



通電延遲計時器 (TON (10 ms))

當計時器的位元輸入為 1 時，就會執行時間計數。當計數值等於設定值時，位元輸出就會變為 1。

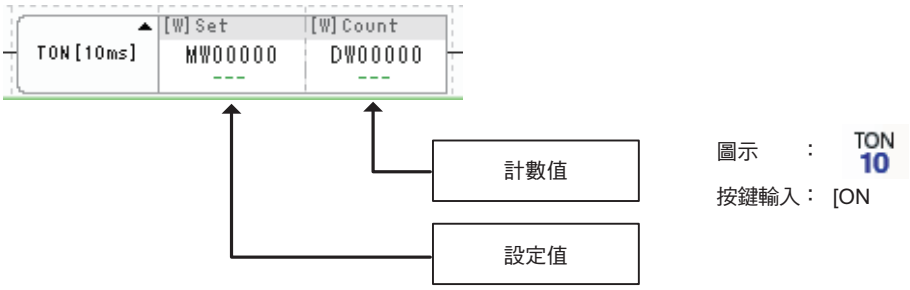
若在計數過程中，位元輸入變為 0，則計時器將停止動作。若位元輸入再次變為 1，將重新 (0) 開始計數。此外，實際的計數時間 (單位為 10 ms) 值將會被儲存在計數專用暫存器中。



(註)計算誤差小於 10 ms。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
設定值 (Set)	×	○	×	×	×	×	×	×	○
計數值 (Count)	×	○*	×	×	×	×	×	×	×

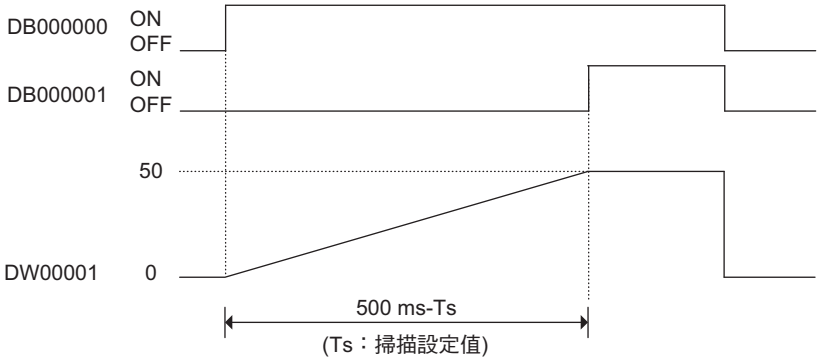
* C、# 暫存器除外

程式範例

以下所示為通電延遲計時器的設定值被設定為 50，且將計數值儲存於 DW00001 時之程式範例。

當繼電器 (DB000000) ON 500 ms 後，線圈 (DB000001) 就會 ON。

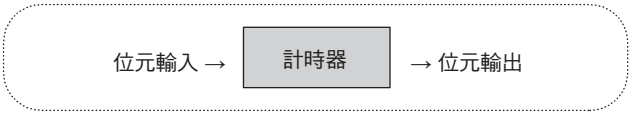
以下為時序圖。

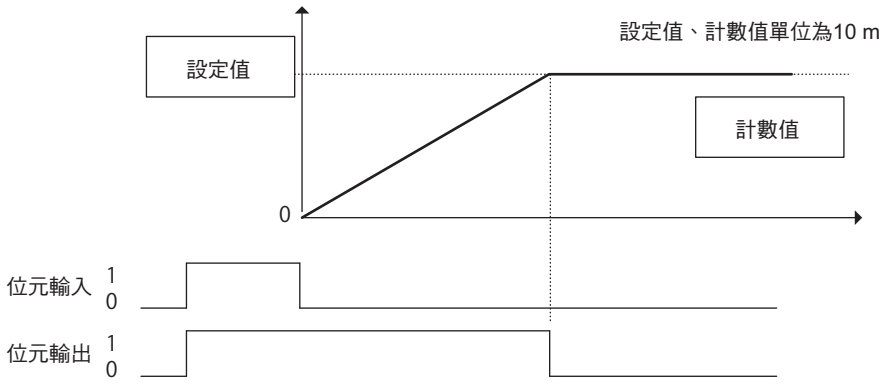


斷電延遲計時器 (TOFF (10 ms))

當計時器的位元輸入為 0 時，就會執行時間計數。當計數值等於設定值時，位元輸出就會變為 0。

若在計數過程中，位元輸入變為 1，則計時器將停止動作。若位元輸入再次變為 0，將重新 (0) 開始計數。此外，實際的計數時間 (單位為 10 ms) 值將會被儲存在計數專用暫存器中。

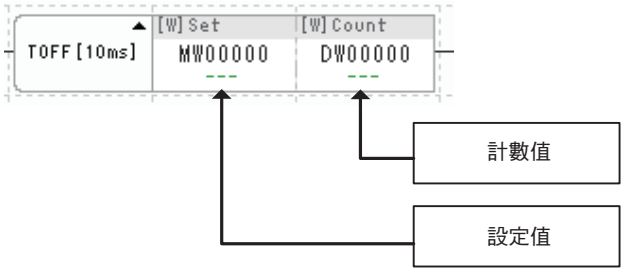




(註)計算誤差小於 10 ms。

格式

所採用之格式如下。



圖示 ： TOFF 10

按鍵輸入： [OFF

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
設定值 (Set)	×	○	×	×	×	×	×	×	○
計數值 (Count)	×	○*	×	×	×	×	×	×	×

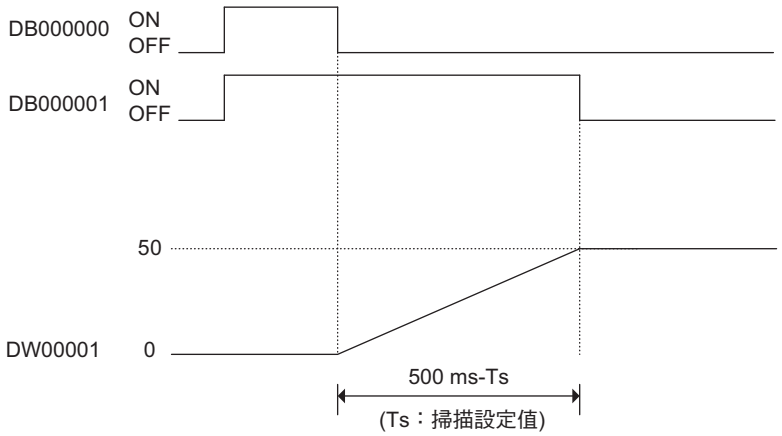
* C、# 暫存器除外

程式範例

以下所示為斷電延遲計時器被設定為 50，且將計數值儲存於 DW00001 時之程式範例。

當繼電器 (DB000000) OFF 500 ms 後，線圈 (DB000001) 就會 OFF。

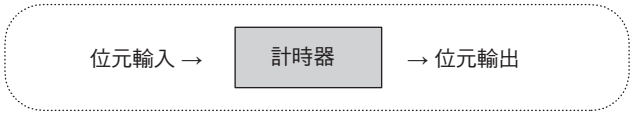
以下為時序圖。

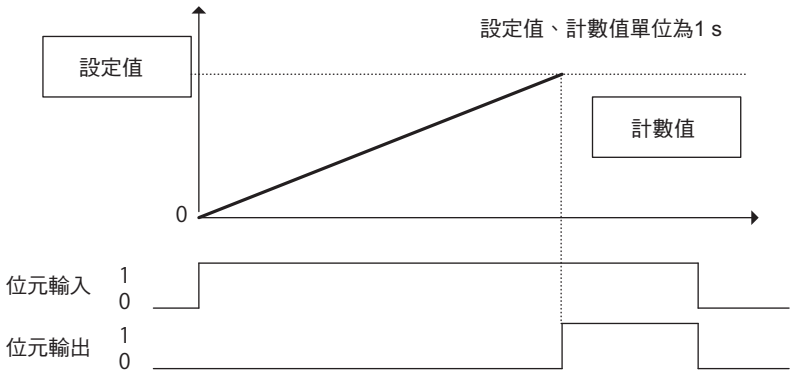


通電延遲計時器 (TON (1 s))

當計時器的位元輸入為 1 時，就會執行時間計數。當計數值等於設定值時，位元輸出就會變為 1。

若在計數過程中，位元輸入變為 0，則計時器將停止動作。若位元輸入再次變為 1，將重新 (0) 開始計數。此外，實際的計數時間 (單位為 1 s) 值將會被儲存在計數專用暫存器中。

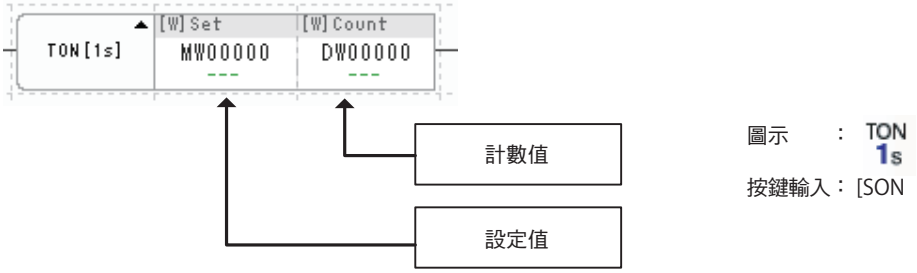




(註)計算誤差小於 1 s。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
設定值 (Set)	×	○	×	×	×	×	×	×	○
計數值 (Count)	×	○*	×	×	×	×	×	×	×

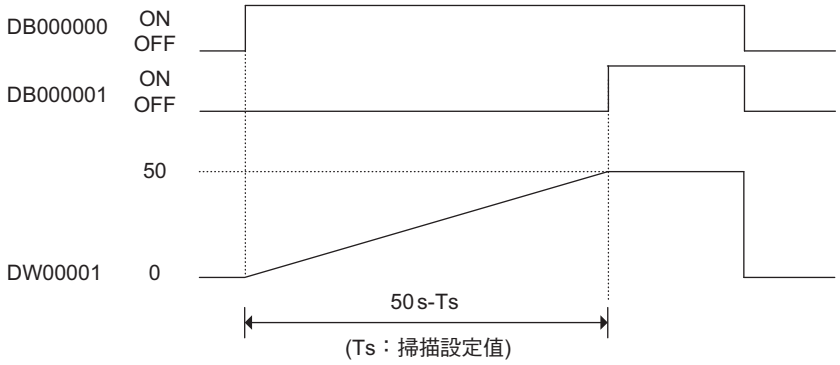
* C、# 暫存器除外

程式範例

以下所示為通電延遲計時器的設定值被設定為 50，且將計數值儲存於 DW00001 時之程式範例。

當繼電器 (DB000000) ON 50 秒後，線圈 (DB000001) 就會 ON。

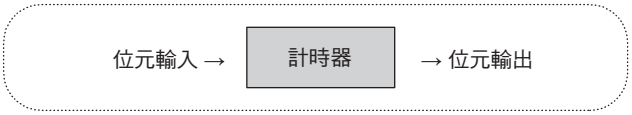
以下為時序圖。

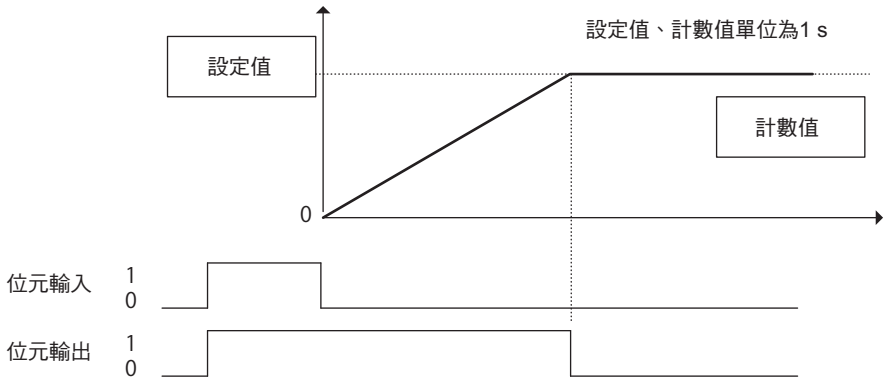


斷電延遲計時器 (TOFF (1 s))

當計時器的位元輸入為 0 時，就會執行時間計數。當計數值等於設定值時，位元輸出就會變為 1。

若在計數過程中，位元輸入變為 1，則計時器將停止動作。若位元輸入再次變為 0，將重新 (0) 開始計數。此外，實際的計數時間 (單位為 1 s) 值將會被儲存在計數專用暫存器中。

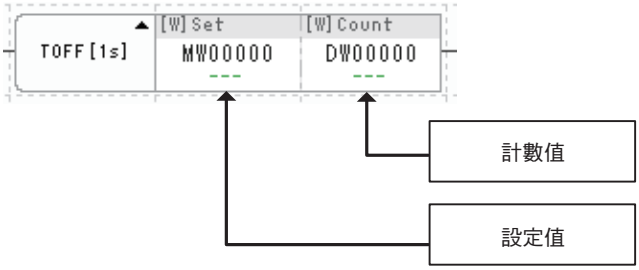




(註)計算誤差小於 1 s。

格式

所採用之格式如下。



圖示 ： TOFF 1s

按鍵輸入： [SOFF

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
設定值 (Set)	×	○	×	×	×	×	×	×	○
計數值 (Count)	×	○*	×	×	×	×	×	×	×

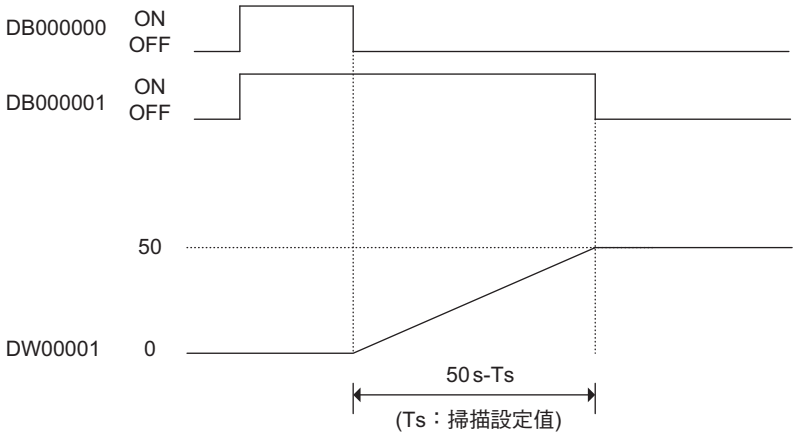
* C、# 暫存器除外

程式範例

以下所示為斷電延遲計時器被設定為 50，且將計數值儲存於 DW00001 時之程式範例。

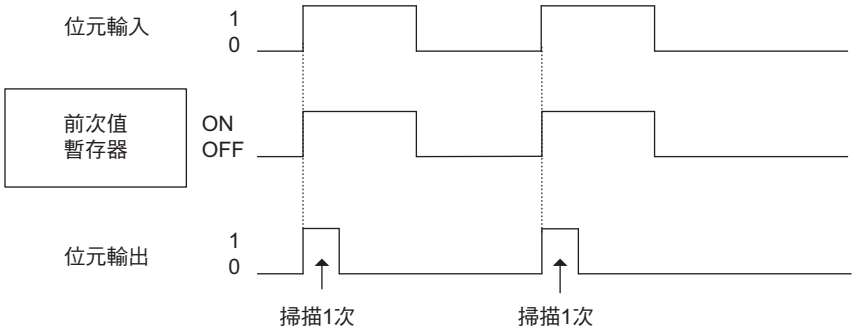
當繼電器 (DB000000) OFF 50 秒後，線圈 (DB000001) 就會 OFF。

以下為時序圖。



上升脈衝 (ON-PLS)

當位元輸入由 0 變為 1 時，只有在執行一次掃描時，位元輸出才會變為 1。位元輸入時的前次值會被儲存於具備 ON-PLS 指令的前次值暫存器中。



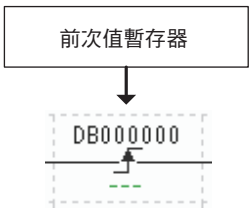
以下將利用真值表來表示使用 ON-PLS 指令時位元輸入與前次值暫存器、位元輸出之間的關係。

位元輸入	前次值暫存器	ON-PLS 指令	位元輸出
0	OFF	→	0
0	ON	→	0
1	OFF	→	1
1	ON	→	0

從表格第 3 行可知，當位元輸入為 1 且前次值暫存器為 0 時，也就是當位元輸入由 0 變為 1 時，ON-PLS 指令的位元輸出就會變成 1。

格式

所採用之格式如下。



圖示　：

按鍵輸入：]P

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
前次值暫存器	○*	×	×	×	×	×	×	×	×

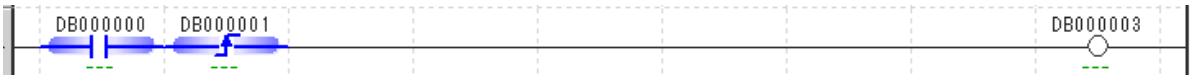
* C、# 暫存器除外

(註)前次值暫存器是用來儲存所輸入的位元前次值的暫存器。請勿使用其他指令來設定數值。

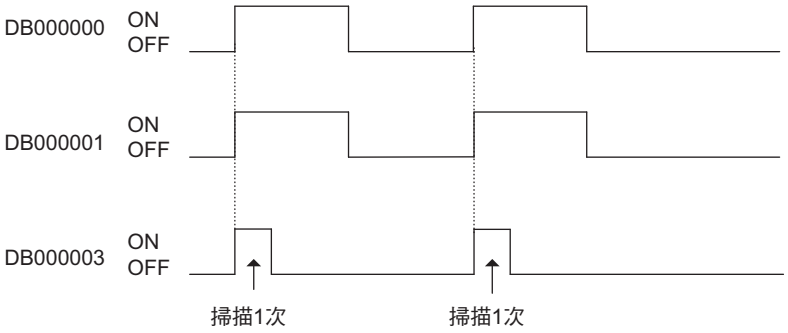
程式範例

當繼電器 (DB000000) 由 OFF 變成 ON 時，只有在執行一次掃描時，線圈 (DB000003) 才會 ON。DB000001 可用來儲存 DB000000 的前次值。

請勿在單一圖面中多次使用前次值儲存用暫存器。

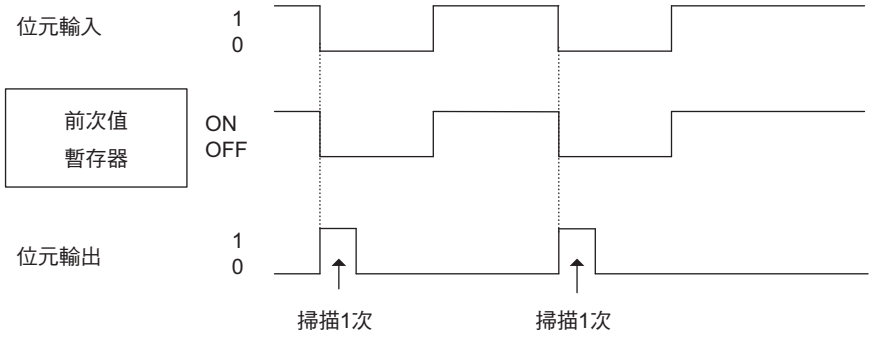


以下為時序圖。



下降脈衝 (OFF-PLS)

當位元輸入由 1 變為 0 時，只有在執行一次掃描時，位元輸出才會變為 1。位元輸入時的前次值會被儲存於具備 OFF-PLS 指令的前次值暫存器中。



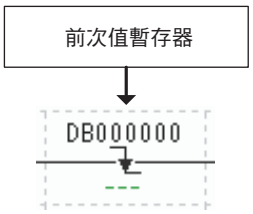
以下將利用真值表來表示使用 OFF-PLS 指令時位元輸入與前次值暫存器、位元輸出之間的關係。

位元輸入	前次值暫存器	OFF-PLS 指令	位元輸出
0	OFF	→	0
0	ON	→	1
1	OFF	→	0
1	ON	→	0

從表格第 2 行可知，當位元輸入為 0 且前次值暫存器為 1 時，也就是當位元輸入由 1 變為 0 時，OFF-PLS 指令的位元輸出就會變成 1。

格式

所採用之格式如下。



圖示　：

按鍵輸入：]N

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
前次值暫存器	O*	×	×	×	×	×	×	×	×

* C、# 暫存器除外

(註)前次值暫存器是用來儲存所輸入的位元前次值的暫存器。請勿使用其他指令來設定數值。

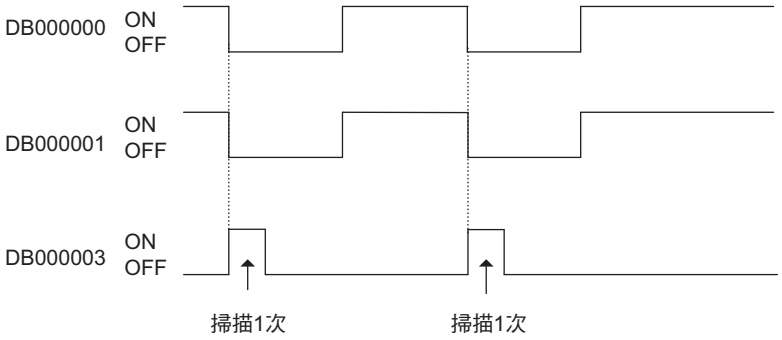
程式範例

當繼電器 (DB000000) 由 ON 變為 OFF 時，只有在執行一次掃描時，線圈 (DB000003) 才會 ON。
DB000001 可用來儲存 DB000000 的前次值。


註記

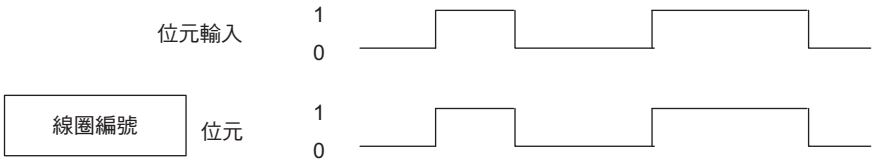
請勿在單一圖面中多次使用前次值儲存用暫存器。

以下為時序圖。



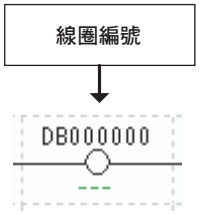
線圈 (COIL)

當位元輸入為 1 時，線圈編號的位元將變為 1。當位元輸入為 0 時，線圈編號的位元將變為 0。



格式

所採用之格式如下。



圖示 ：

按鍵輸入：@

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
線圈編號	○*	×	×	×	×	×	×	×	×

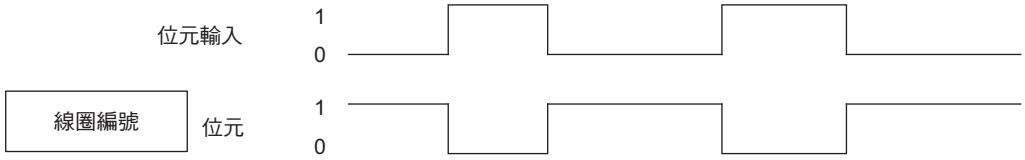
* C、# 暫存器除外

程式範例

當繼電器 (DB000001) ON 時，線圈 (DB000000) 也變為 ON。

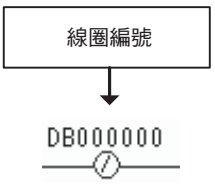
反轉型線圈 (REV-COIL)

當位元輸入為 0 時，線圈編號的位元將變為 1。當位元輸入為 1 時，線圈編號的位元將變為 0。



格式

所採用之格式如下。



圖示 ：

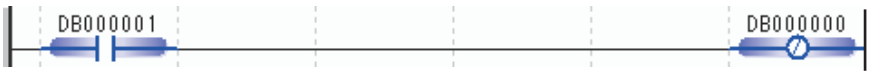
按鍵輸入： @RV

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
線圈編號	○*	×	×	×	×	×	×	×	×

* #，C 暫存器除外

程式範例

當繼電器 (DB000001) ON 時，線圈 (DB000000) 變為 OFF。



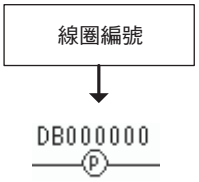
上升變化檢測用線圈 (ONP-COIL)

當位元輸入由 0 變為 1 時，只有在掃描一次時，線圈編號的位元才會變為 1。

與結合 ON-PLS 指令和 COIL 指令時的動作相同。

格式

所採用之格式如下。



圖示 ：

按鍵輸入： @P

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
線圈編號	○*	×	×	×	×	×	×	×	×

* #，C 暫存器除外

程式範例

使用 NOC 指令，讓繼電器 (DB000000) 由 OFF 變為 ON 時，上升變化檢測線圈 (DB000001) 也會變 ON。

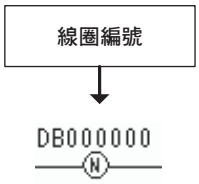
下降變化檢測用線圈 (OFFP-COIL)

當位元輸入由 1 變為 0 時，只有在掃描一次時，線圈編號的位元才會變為 1。

與結合 OFF-PLS 指令和 COIL 指令時的動作相同。

格式

所採用之格式如下。



圖示　：

按鍵輸入： @N

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
線圈編號	○*	×	×	×	×	×	×	×	×

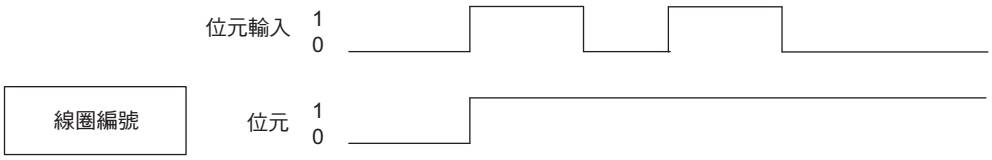
* #，C 暫存器除外

程式範例

使用 NOC 指令讓繼電器 (DB000000) 由 ON 變為 OFF 時，下降變化檢測線圈 (DB000001) 將變為 ON。

設定線圈 (S-COIL)

當位元輸入為 1 時，線圈編號的位元也會變為 1，此時設定線圈將維持 ON 狀態。



格式

所採用之格式如下。

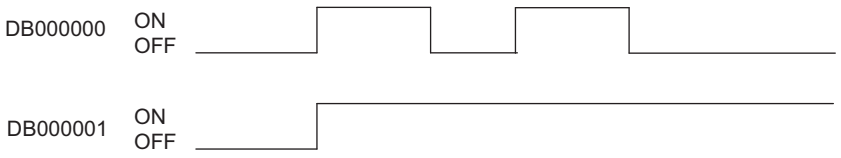
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
線圈編號	O*	×	×	×	×	×	×	×	×

* C、# 暫存器除外

程式範例

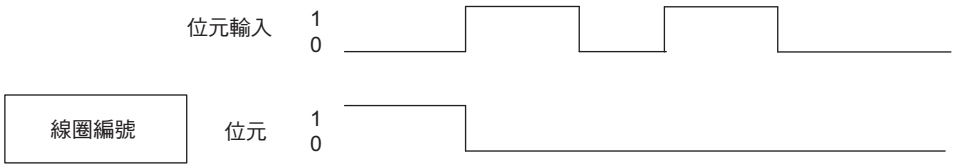
當繼電器 (DB000000) ON 時，設定線圈 (DB000001) 將維持 ON 狀態。

以下為時序圖。



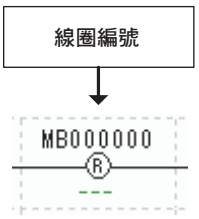
重置線圈 (R-COIL)

當位元輸入為 1 時，重置線圈編號的位元將變為 0，此時重置線圈將維持 OFF 狀態。



格式

所採用之格式如下。



圖示 ： Ⓡ

按鍵輸入： @R

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
線圈編號	O*	×	×	×	×	×	×	×	×

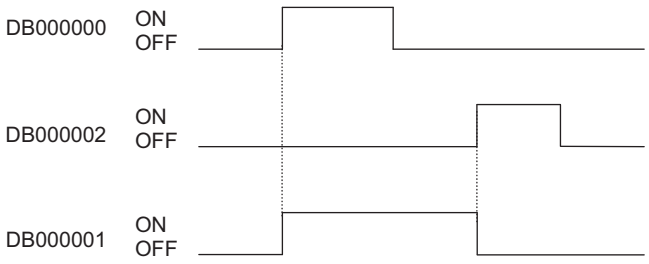
* C、# 暫存器除外

程式範例

以下所示為使用重置線圈，將目前處於 ON 狀態的第 1 行設定線圈 OFF 時之程式範例。

當第 1 行的設定線圈 (DB000001) ON 時，繼電器 (DB000002) 也將變為 ON，這時候第 2 行的重置線圈 (DB000001) 將變為 ON，而設定線圈 (DB000001) 則變為 OFF。

以下為時序圖。



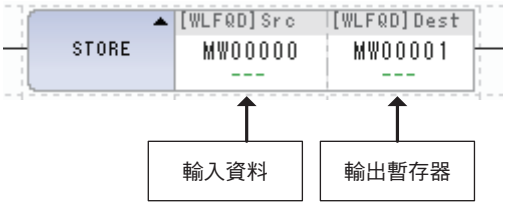
4.3 數值運算指令

儲存 (STORE)

可將輸入資料儲存於輸出暫存器中。

格式

所採用之格式如下。



圖示 ： =>

按鍵輸入： ;

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	O	O	O	O	O	O	O	O
輸出暫存器 (Dest)	×	O*	O*	O*	O*	O*	O*	O	×

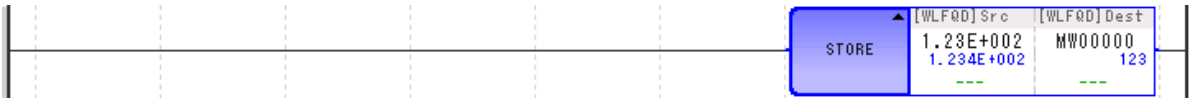
* C、# 暫存器除外

程式範例

以下所示為輸入資料被儲存至輸出暫存器時之程式範例。

· **將輸入資料 12345 (整數) 儲存至輸出暫存器 (MW00000) 中時**

· **將輸入資料 123.45 (實數) 儲存至輸出暫存器 (MW00000) 中時**



· **將輸入資料 H89ABCDEF (長整數) 儲存至輸出暫存器 (MW00000) 中時**
將長整數的低階字元 -12817 (CDEFH) 儲存至 MW00000。

· **將輸入資料 1234 (整數) 儲存至輸出暫存器 (MF00000) 中時**

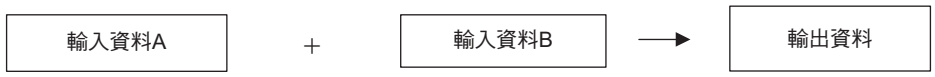
補充　為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ **第 3 章 暫存器 – 不同資料類型之演算注意事項 (第 3-8 頁)**

加法 (ADD (+))

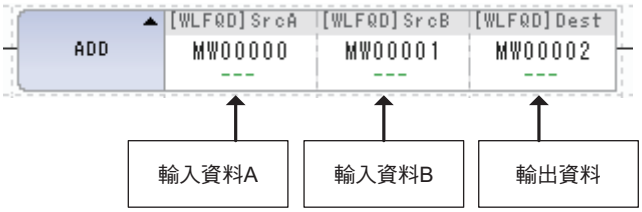
將輸入資料 A 和輸入資料 B 相加後，再將結果儲存為輸出資料。

發生溢位 (Overflow) 及欠位 (Underflow) 時，就會出現運算錯誤。



格式

所採用之格式如下。



圖示 ： ✚

按鍵輸入： ＋

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	O	O	O	O	O	×	O	O
輸入資料 B (SrcB)	×	O	O	O	O	O	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	O*	O*	×	O	×

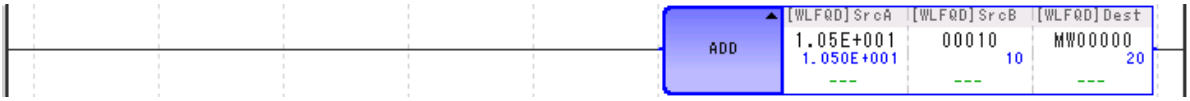
* C、# 暫存器除外

程式範例

以下所示為輸入資料 A 和輸入資料 B 相加，然後再將結果儲存為輸出資料時之程式範例。

· **輸出資料 A 為 100、輸出資料 B 為 200，而且將輸出資料儲存在 MW00000 時**
100 + 200 → MW00000 = 300

· **輸出資料 A 為 10.5、輸出資料 B 為 10，而且將輸出資料儲存在 MW00000 時**
10.5 + 10 → MW00000 = 20 (設定為小數點以下無條件捨去)



· **當輸出資料 A (MW00002) 為 20000、輸出資料 B (MW00003) 為 30000，而且將輸出資料儲存於 ML00000 時**
MW00002(20000) + MW00003(30000) → ML00000 = 32767*

* 輸入資料 A、B 皆為整數型時，會在整數型範圍內進行運算，因此上述範例發生了溢位的情形。

補充事項

若資料為整數型時，一旦運算結果超過 32767 就會發生溢位，若小於 -32768 則會發生欠位等運算錯誤。

若資料為長整數型時，一旦運算結果超過 2147483647 就會發生溢位，若小於 -2147483648 則會發生欠位等運算錯誤。

補充

為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ **第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)**

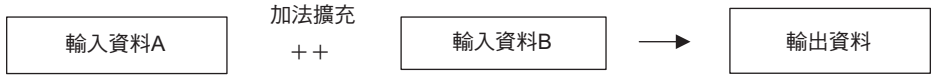
長整數型的加減運算指令 (+、-、++、--) 通常利用 32 位元來進行運算。
不過，只有在使用餘數修正運算式 [前方接 MUL(x) 指令，後方接 DIV (÷) 指令] 時，才會利用 64 位元來進行運算。

加法擴充 (ADDX (++))

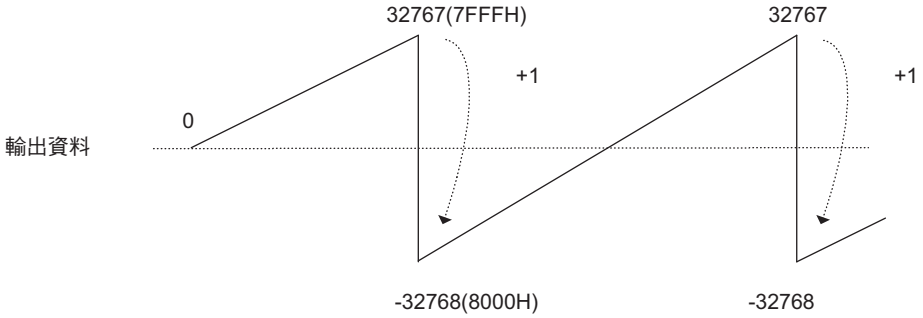
將輸入資料 A 和輸入資料 B 相加後，再將結果儲存為輸出資料。

發生溢位時，將會回到負方向最大值繼續運算，而不會出現運算錯誤。

發生欠位時，將會回到正方向最大值繼續運算，而不會出現運算錯誤。



下圖為輸出資料之變化圖。

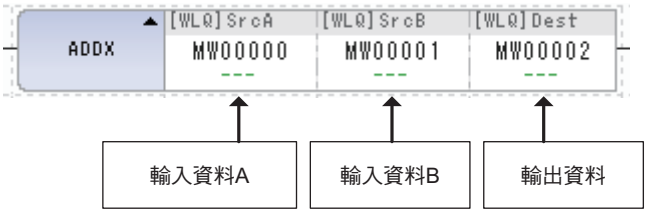


(註)1. 上圖所示的輸出資料為整數型資料。若資料為長整數型時，只要將 2147483647 (7FFFFFFFH) 加上 1，就會變為 -2147483648 (80000000)。

2. 所使用的運算方法有別於 ADD (+) 指令、SUB (-) 指令或 EXPRESSION 指令，不會發生溢位或欠位等錯誤。

格式

所採用之格式如下。



圖示 ： ✚✚

按鍵輸入：＋＋

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	×	×	×	○	○
輸入資料 B (SrcB)	×	○	○	○	×	×	×	○	○
輸出資料 (Dest)	×	○*	○*	○*	×	×	×	○	×

* C、# 暫存器除外

程式範例

以下所示為輸入資料 A 和輸入資料 B 加法擴充，然後再將結果儲存為輸出資料時的程式範例。

· **輸出資料 A 為 32760、輸出資料 B 為 10，而且將輸出資料儲存在 MW00000 時**
32760 ++ 10 → MW00000 = –32766

· **當輸出資料 A (MW00002) 為 20000、輸出資料 B (MW00003) 為 30000，而且將輸出資料儲存於 ML00000 時**
MW00002(20000) ++ MW00003(30000) → ML00000 = –15536*

* 輸入資料 A、B 皆為整數型時，會在整數型範圍內進行運算，以上述範例來說，就是 ML00000=50000。

· **輸出資料 A 為 2147483647、輸出資料 B 為 2，而且將輸出資料儲存在 ML00000 時**
2147483647 ++ 2 → ML00000 = –241783647

· **輸出資料 A 為 -32768、輸出資料 B 為 -1，而且將輸出資料儲存在 MW00000 時**
–32768 ++ –1 → MW00000 = 32767

補充

為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

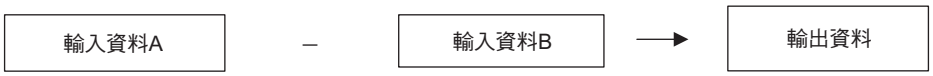
☞ **第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)**

長整數型的加減運算指令 (+、-、++、--) 通常利用 32 位元來進行運算。
不過，只有在使用餘數修正運算式 [前方接 MUL(x) 指令，後方接 DIV (÷) 指令] 時，才會利用 64 位元來進行運算。

減法 (SUB (–))

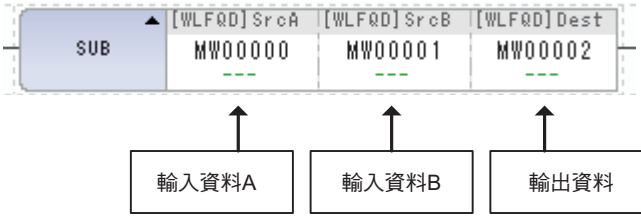
將輸入資料 A 和輸入資料 B 相減，再將結果儲存為輸出資料。

發生溢位 (Overflow) 及欠位 (Underflow) 時，就會出現運算錯誤。



格式

所採用之格式如下。



圖示 ： ▬

按鍵輸入： -

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○
輸出資料 (Dest)	×	○*	○*	○*	○*	○*	×	○	×

* C、# 暫存器除外

程式範例

以下所示為輸入資料 A 和輸入資料 B 相減，然後再將結果儲存為輸出資料之程式範例。

· **輸出資料 A 為 100、輸出資料 B 為 200，而且將輸出資料儲存在 MW00000 時**
100–200 → MW00000 = –100

· **輸出資料 A 為 10.5、輸出資料 B 為 10，而且將輸出資料儲存在 MW00000 時**
10.5 – 10 → MW00000 = 0 (設定為小數點以下無條件捨去)

· **當輸出資料 A (MW00002) 為 -20000、輸出資料 B (MW00003) 為 30000，而且將輸出資料儲存於 ML00000 時**
MW00002(–20000)–MW00003(30000) → ML00000 = –32768*

* 輸入資料 A、B 皆為整數型時，會在整數型範圍內進行運算，因此上述範例發生了欠位的情形。

補充事項

若資料為整數型時，一旦運算結果超過 32767 就會發生溢位，若小於 -32768 則會發生欠位等運算錯誤。

若資料為長整數型時，一旦運算結果超過 2147483647 就會發生溢位，若小於 -2147483648 則會發生欠位等運算錯誤。

補充

為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ **第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)**

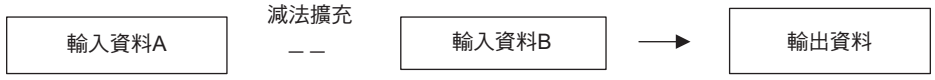
長整數型的加減運算指令 (+、-、++、--) 通常利用 32 位元來進行運算。
不過，只有在使用餘數修正運算式 [前方接 MUL(x) 指令，後方接 DIV (÷) 指令] 時，才會利用 64 位元來進行運算。

減法擴充 (SUBX (– –))

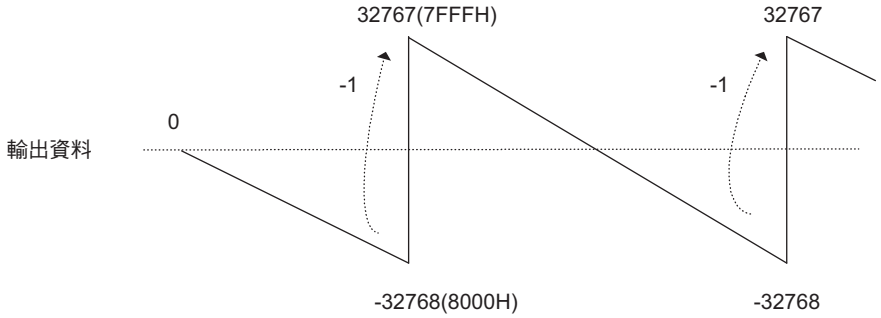
將輸入資料 A 和輸入資料 B 相加後，再將結果儲存為輸出資料。

發生溢位時，將會回到負方向最大值繼續運算，而不會出現運算錯誤。

發生欠位時，將會回到正方向最大值繼續運算，而不會出現運算錯誤。



以下為輸出資料變化圖。

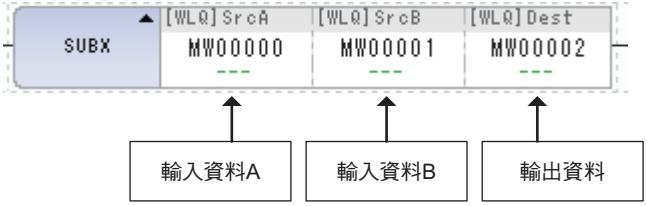


(註)1. 上圖所示的輸出資料為整數型資料。資料為長整數型時，只要在運算時將 -2147483648 (80000000H) 減 1，就會變為 2147483647 (7FFFFFFFH)。

2. 所使用的運算方法有別於 ADD (+) 指令、SUB (-) 指令或 EXPRESSION 指令，不會發生溢位或欠位等錯誤。

格式

所採用之格式如下。



圖示 ：

按鍵輸入： --

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	×	×	×	○	○
輸入資料 B (SrcB)	×	○	○	○	×	×	×	○	○
輸出資料 (Dest)	×	○*	○*	○*	×	×	×	○	×

* C、# 暫存器除外

程式範例

以下所示為輸入資料 A 和輸入資料 B 減法擴充後，然後再將結果儲存為輸出資料時的程式範例。

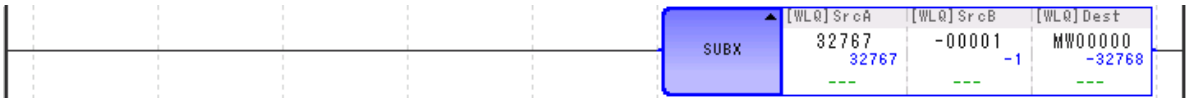
· **輸出資料 A 為 -32760、輸出資料 B 為 10，而且將輸出資料儲存在 MW00000 時**
–32760 – – 10 → MW00000 = 32766

· **當輸出資料 A (MW00002) 為 -20000、輸出資料 B (MW00003) 為 30000，而且將輸出資料儲存於 ML00000 時**
MW00002(–20000) – – MW00003(30000) → ML00000 = 15536*

* 輸入資料 A、B 皆為整數型時，會在整數型範圍內進行運算，以上述範例來說，就是 ML00000=-50000。

· **輸出資料 A 為 -2147483648、輸出資料 B 為 2，而且將輸出資料儲存在 ML00000 時**
–2147483648 – – 2 → ML00000 = 241783646

· **輸出資料 A 為 32767、輸出資料 B 為 -1，而且將輸出資料儲存在 MW00000 時**
32767 – – –1 → MW00000 = –32768



補充

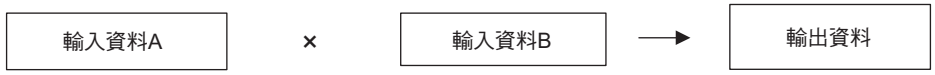
為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ **第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)**

長整數型的加減運算指令 (+、-、++、--) 通常利用 32 位元來進行運算。
不過，只有在使用餘數修正運算式 [前方接 MUL(x) 指令，後方接 DIV (÷) 指令] 時，才會利用 64 位元來進行運算。

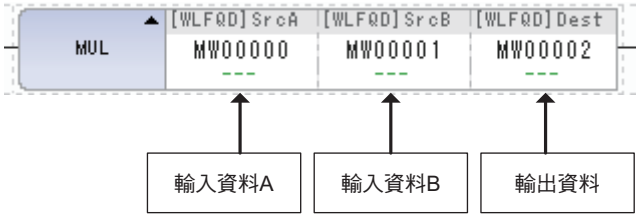
乘法 (MUL (×))

將輸出資料 A 和輸出資料 B 相乘後，再將結果儲存為輸出資料。



格式

所採用之格式如下。



圖示　：

按鍵輸入：*

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○
輸出資料 (Dest)	×	○*	○*	○*	○*	○*	×	○	×

* C、# 暫存器除外

程式範例

以下所示為輸入資料 A 和輸入資料 B 相乘，然後再將結果儲存為輸出資料時之程式範例。

· **輸出資料 A 為 100、輸出資料 B 為 200，而且將輸出資料儲存在 MW00000 時**
100 × 200 → MW00000 = 20000

· **當輸出資料 A (MW00002) 為 200、輸出資料 B (MW00003) 為 300，而且將輸出資料儲存於 ML00000 時**
MW00002(200) × MW00003(300) → ML00000 = 60000

· **當輸出資料 A (ML00000) 為 -200、輸出資料 B (MW00003) 為 300，而且將輸出資料儲存於 MW00002 時**
−200 × 300 → MW00002 = 5536*

* 當輸入資料包含長整數型資料時，會在長整數型資料的範圍內進行運算，若輸出資料為整數型資料時，只要運算結果超過整數型資料的範圍，原本運算結果的低階 16 位元就會被儲存為輸出資料。

補充

為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ **第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)**

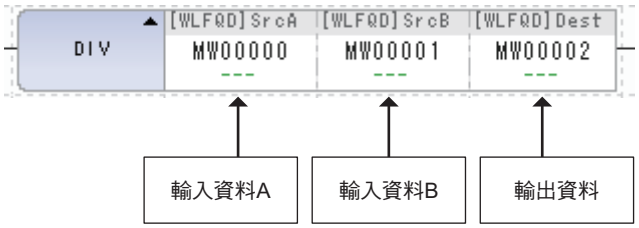
長整數型的加減運算指令 (+、-、++、--) 通常利用 32 位元來進行運算。
不過，只有在使用餘數修正運算式 [前方接 MUL (x) 指令，後方接 DIV (÷) 指令] 時，才會利用 64 位元來進行運算。

除法 (DIV (÷))

將輸入資料 A 和輸入資料 B 相除，然後再將結果儲存為輸出資料。

格式

所採用之格式如下。



圖示 ： ÷

按鍵輸入：/

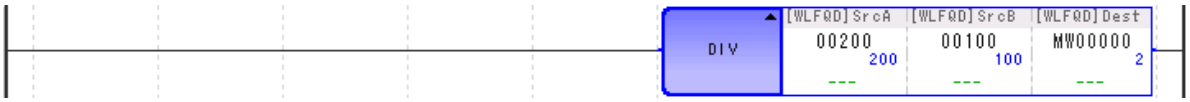
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○
輸出資料 (Dest)	×	○*	○*	○*	○*	○*	×	○	×

* C、# 暫存器除外

程式範例

以下所示為輸入資料 B 除以輸入資料 A，然後再將結果儲存為輸出資料時的程式範例。

· **輸出資料 A 為 200、輸出資料 B 為 100，而且將輸出資料儲存在 MW00000 時**
200 ÷ 100 → MW00000 = 2



· **輸出資料 A 為 200、輸出資料 B 為 1000，而且將輸出資料儲存在 ML00000 時**
200 ÷ 1000 → ML00000 = 0

· **輸出資料 A 為 200、輸出資料 B 為 1000，而且將輸出資料儲存在 MF00000 時**
200 ÷ 1000 → MF00000 = 0.2

補充

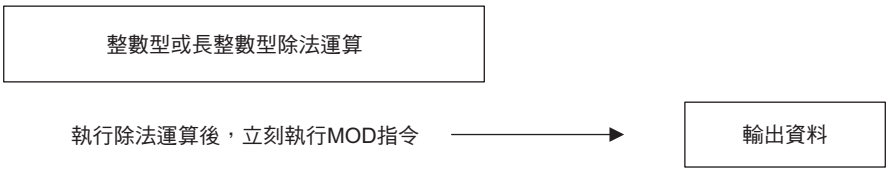
為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ **第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)**

長整數型的加減運算指令 (+、-、++、--) 通常利用 32 位元來進行運算。
不過，只有在使用餘數修正運算式 [前方接 MUL(x) 指令，後方接 DIV (÷) 指令] 時，才會利用 64 位元來進行運算。

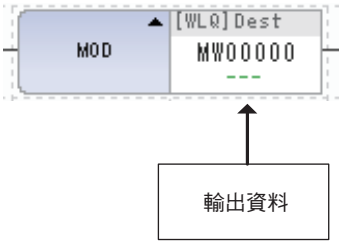
整數型餘數 (MOD)

可將前一項執行過的整數型及長整數型除法運算的餘數儲存為輸出資料。請在執行 DIV (÷) 指令後，立刻執行 MOD 指令。若無法在執行 DIV (÷) 指令後立刻執行 MOD 指令，將不保證下一個數值運算指令出現前的運算結果。



格式

所採用之格式如下。



圖示　：MOD

按鍵輸入：MOD

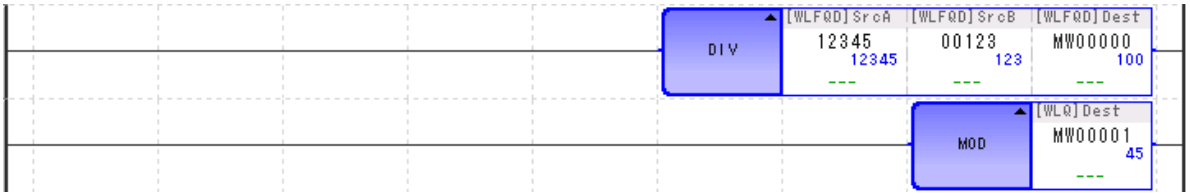
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸出資料 (Dest)	×	O*	O*	O*	×	×	×	O	×

* C、# 暫存器除外

程式範例

以下所示為輸入資料 A 除以輸入資料 B，再將餘數儲存為輸出資料之程式範例。

· **前一項除法運算如下時**
12345 ÷ 123 → MW00000 = 100
立刻執行 MOD 指令 → MW00001 = 45



· **前一項除法運算如下時**
123456789 ÷ 12345 → ML00000 = 10000
立刻執行 MOD 指令 → ML00002 = 6789

補充　為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。
☞ **第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)**

實數型餘數 (REM)

可將實數型除法運算的餘數儲存為輸出資料。所謂的「餘數」就是輸入資料反覆地減去基底值後所得到之剩餘數值。

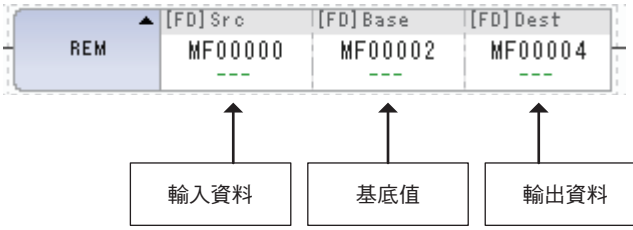
換句話說，當輸入資料減去基底值 n 次後所得到的數值 (輸入資料 - 基底值 x n) 小於基底值時，就會輸出餘數。

n 值依序增加為 0、1、2、3... 等數值，並且滿足以下計算公式時，就會利用 n 值來運算輸出資料。

(輸入資料 － 基底值 x n) < 基底值

格式

所採用之格式如下。



圖示　：REM

按鍵輸入：REM

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	×	×	×	○	○	×	×	○
基底 (Base)	×	×	×	×	○	○	×	×	○
輸出資料 (Dest)	×	×	×	×	○*	○*	×	○	×

* C、# 暫存器除外

程式範例

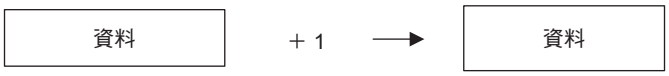
以下所示為輸入資料減去基底值 n 次後，再將所得到的餘數儲存為輸出資料之程式範例。

· **當輸入資料為 5.0、基底值為 2.0，且將輸出資料儲存於 MF00000 時**
5.0–2.0–2.0 = 1.0 < 基底值 (2.0) → MF00000 = 1.0

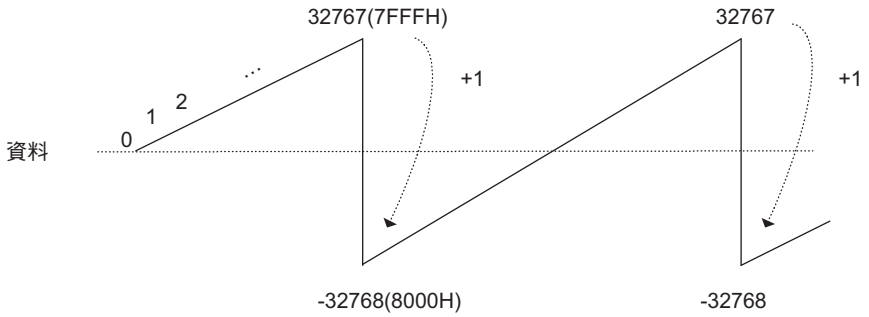
· **當輸入資料為 3000.0、基底值為 3.0，且將輸出資料儲存於 MF00000 時**
3000.0–3.0–3.0 … = 0.0 < 基底值 (3.0) → MF00000 = 0.0

遞增 (INC)

可將整數型及長整數型的資料加 1。無論是整數型或是長整數型資料，皆不會發生溢位及欠位的情形。運算方法和 ADDX (++) 指令相同。



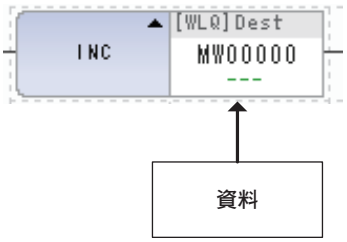
以下為執行 INC 指令時之資料變化圖。



(註) 上圖係以整數型資料為例。若資料為長整數型時，只要將 2147483647 (7FFFFFFFH) 加上 1，就會變為 -2147483648 (80000000H)。

格式

所採用之格式如下。



圖示　：**+1**

按鍵輸入： INC

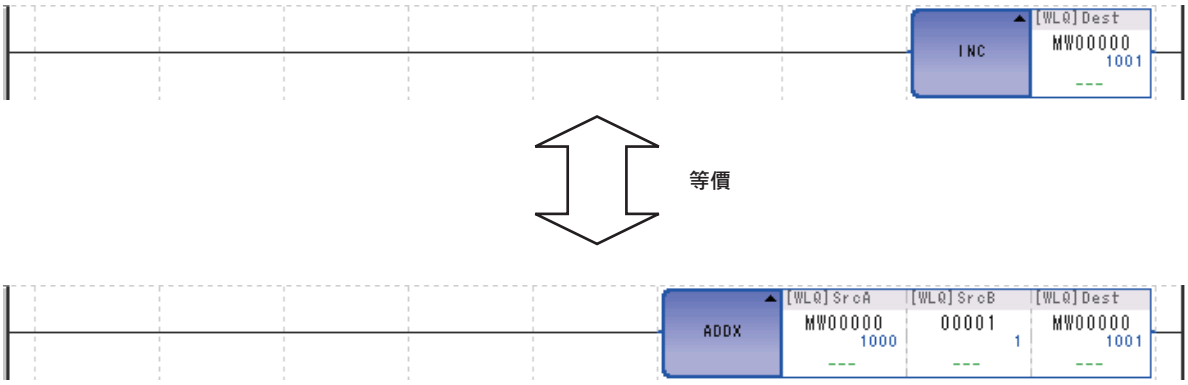
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
資料 (Dest)	×	○	○	○*	×	×	×	○	×

* C、# 暫存器除外

程式範例

以下所示為使用 INC 指令及 ADDX (++) 指令之程式範例。

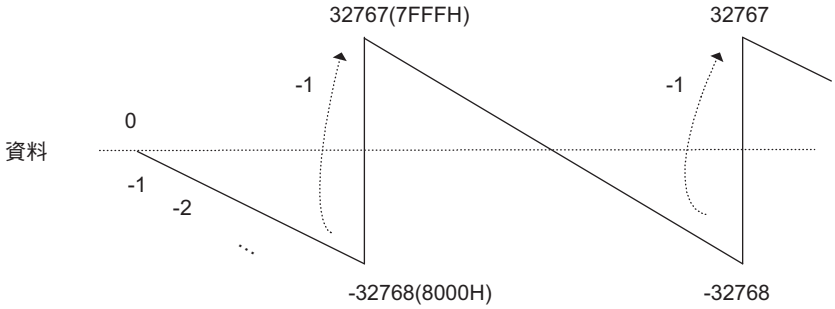
此指令等價於使用 ADDX (++) 指令，將資料 (MW00000) 1000 加 1。



遞減 (DEC)

將整數型及長整數型的資料減 1。無論是整數型或是長整數型資料，皆不會發生溢位及欠位的情形。運算方法與 SUBX (--) 指令相同。

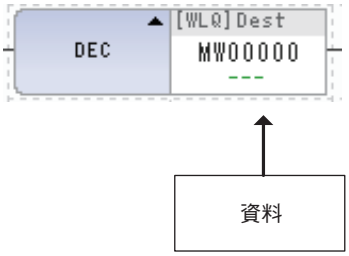
以下為執行 DEC 指令時之資料變化圖。



(註)上圖係以整數型資料為例。+ 若資料為長整數型時，只要在運算時將 -2147483648 (80000000H) 減 1，就會變為 2147483647 (7FFFFFFFH)。

格式

所採用之格式如下。



圖示　：-1

按鍵輸入： DEC

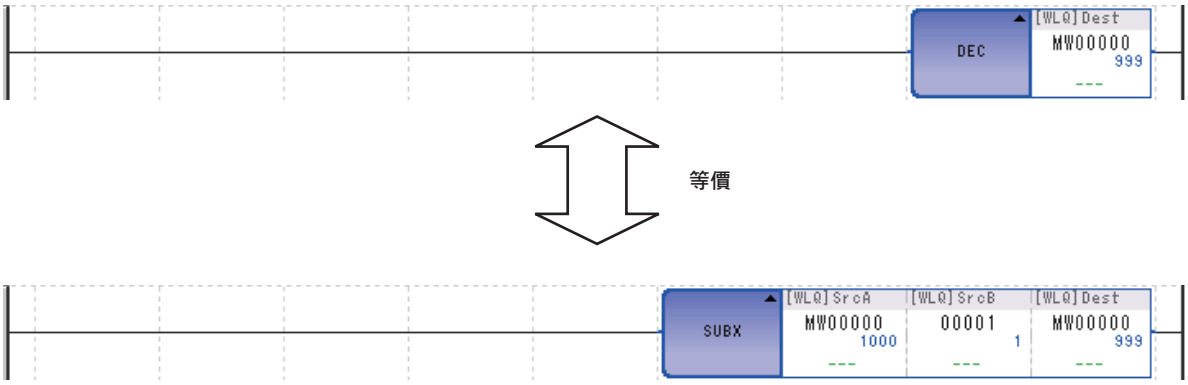
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
資料 (Dest)	×	○	○	○*	×	×	×	○	×

* C、# 暫存器除外

程式範例

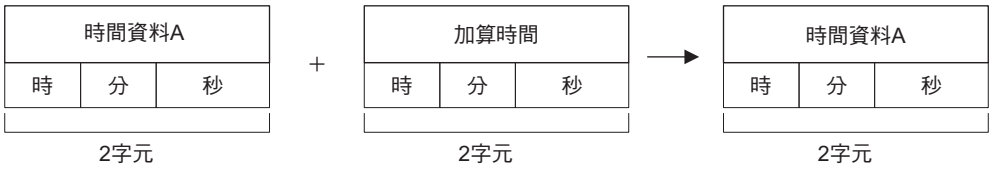
以下為同時使用 DEC 指令和 SUBX (--) 指令之程式範例。

此指令等價於使用 SUBX (--) 指令，將資料 (MW00000) 1000 減 1。



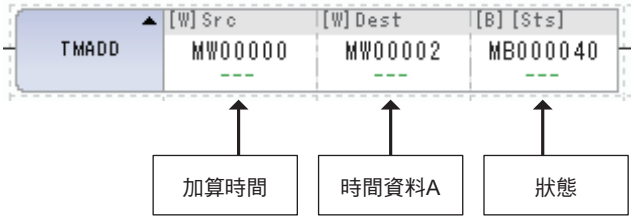
增加時間 (TMADD)

可將二個時間資料 (時 / 分 / 秒) 相加。將時間資料 A 加上加算時間後，結果將被儲存於時間資料 A。時間資料相當於 2 個字元。



格式

所採用之格式如下。



圖示 ：

按鍵輸入： TMADD

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
加算時間 (Src)	×	○*2	×	×	×	×	×	×	×
時間資料 A (Dest)	×	○*2	×	×	×	×	×	×	×
狀態 (Sts)*1	○*2	×	×	×	×	×	×	×	×

*1. 可省略
*2. C、# 暫存器除外

以下所示為時間資料的格式。

偏移值	資料內容	資料範圍 (BCD)
0	時 / 分	高位元組 (時)：00 ~ 23 低位元組 (分)：00 ~ 59
1	秒	0000 ~ 0059

若運算結果超出上述資料範圍，就會輸出 9999 作為秒數的資料，且狀態位元會變為 1，此時將不更新時間資料 A。

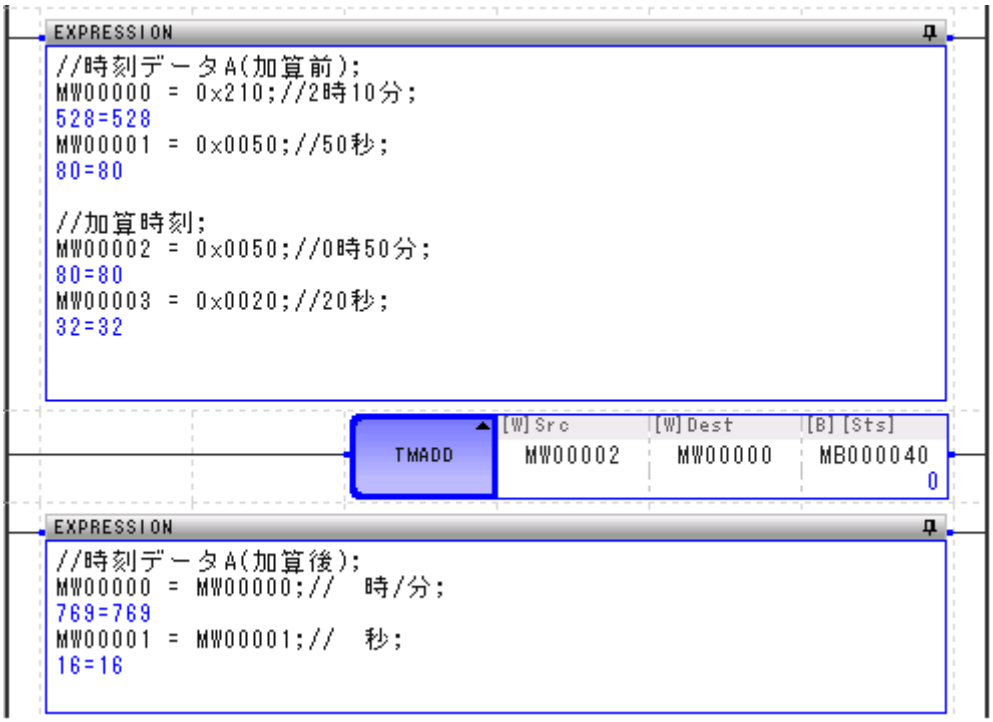
若運算結果符合所規定的範圍，則狀態位元將變為 0。

程式範例

下表為使用 TMADD 指令來編寫階梯圖程式之條件範例。表格裡包含了執行指令前的時間資料 A 和所要加上的時間。

時間	執行指令前的時間資料 A	加算時間
時 / 分	MW00000 = 0210H (2 時 10 分)	MW00002 = 0050H (0 時 50 分)
秒	MW00001 = 0050H (50 秒)	MW00003 = 0020H (20 秒)

以下為根據上表所示的條件加上時間，然後再將結果儲存為時間資料 A 之程式範例。

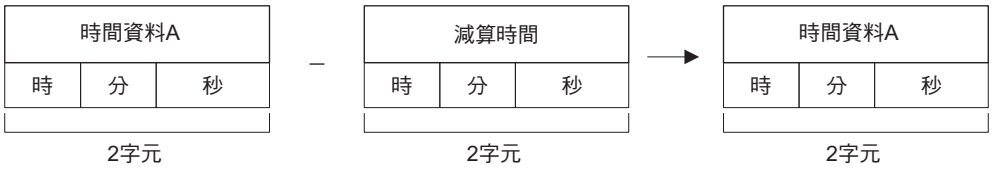


下表為執行指令前的時間資料 A 加上加算時間後所得到的結果。

時間	執行指令後之時間資料 A
時 / 分	MW00000 = 769 = 0301H (3 時 01 分)
秒	MW00001 = 16 = 0010H (10 秒)

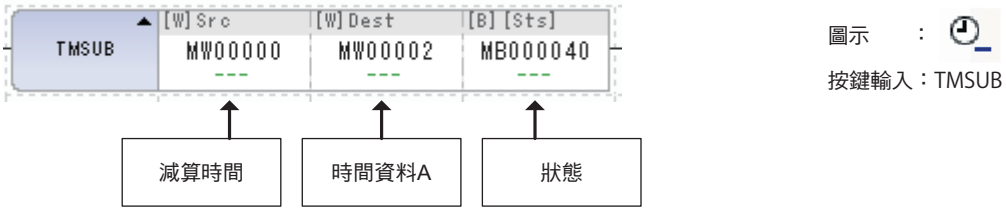
減少時間 (TMSUB)

可將 2 個時間資料 (時 / 分 / 秒) 相減。時間資料 A 減掉所要減去的時間後，結果會被儲存為時間資料 A。時間資料相當於 2 個字元。



格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
減算時間 (Src)	×	○*2	×	×	×	×	×	×	×
時間資料 A (Dest)	×	○*2	×	×	×	×	×	×	×
狀態 (Sts)*1	○*2	×	×	×	×	×	×	×	×

*1. 可省略
*2. C、# 暫存器除外

以下所示為時間資料的格式。

偏移值	資料內容	資料範圍 (BCD)
0	時 / 分	高位元組 (時)：00 ~ 23 低位元組 (分)：00 ~ 59
1	秒	0000 ~ 0059

若運算結果超出上述資料範圍，就會輸出 9999 作為秒資料，且狀態位元會變為 1，此時將不更新時間資料 A。

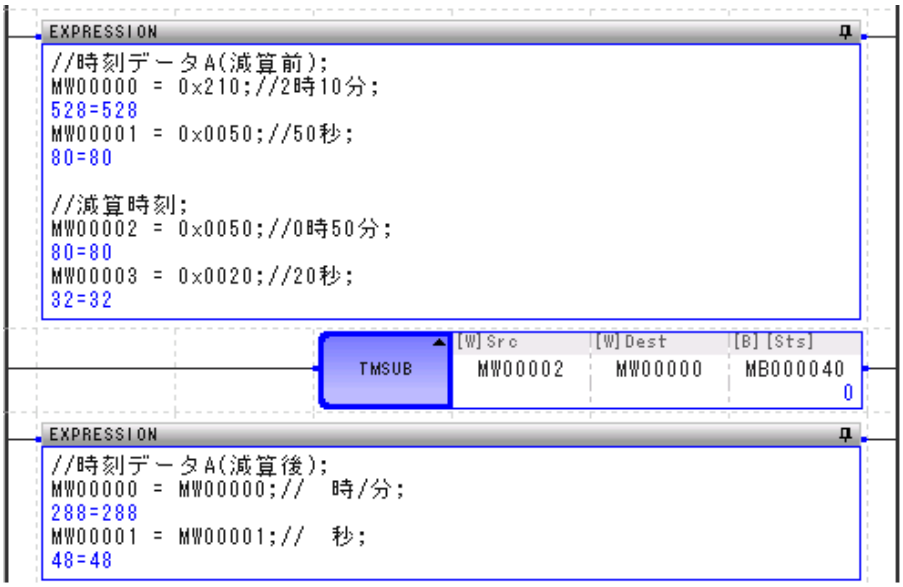
若運算結果符合所規定的範圍，則狀態位元將變為 0。

程式範例

下表為使用 TMSUB 指令來編寫階梯圖程式之條件範例。表格裡包含了執行指令前的時間資料 A 以及所要減去的時間。

時間	執行指令前的時間資料 A	減算時間
時 / 分	MW00000 = 0210H (2 時 10 分)	MW00002 = 0050H (0 時 50 分)
秒	MW00001 = 0050H (50 秒)	MW00003 = 0020H (20 秒)

以下為利用上述條件減去時間，然後再將結果儲存為時間資料 A 之程式範例。



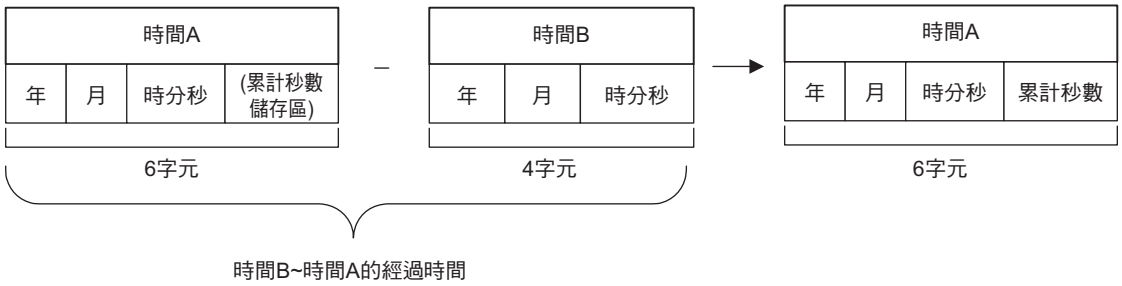
下表為執行指令前的時間資料 A 減掉所要減去的時間後所得到的結果。

時間	執行指令後之時間資料 A
時 / 分	MW00000 = 288 = 0120H (1 時 20 分)
秒	MW00001 = 48 = 0030H (30 秒)

經過時間 (SPEND)

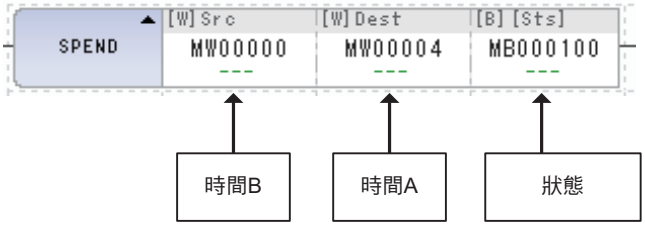
將 2 項資料 (年 / 月 / 日 / 時 / 分 / 秒) 作減法運算，並計算所經過的時間。將時間 A 減去時間 B (也就是計算時間 B~ 時間 A 所經過的時間)，再將結果儲存至時間 A。

時間資料約佔 4 個字元。



格式

所採用之格式如下。



圖示 ： 

按鍵輸入： SPEND

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
時間 B (Src)	×	O*2	×	×	×	×	×	×	×
時間 A (Dest)	×	O*2	×	×	×	×	×	×	×
狀態 (Sts)*1	O*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

下表為時間 B 的格式。

偏移值	資料內容	資料範圍 (BCD)	輸出輸入
0	年 (BCD)	0000 ~ 0099	IN
1	月 / 日 (BCD)	高位元組 (月)：01 ~ 12 低位元組 (日)：01 ~ 31	IN
2	時 / 分 (BCD)	高位元組 (時)：00 ~ 23 低位元組 (分)：00 ~ 59	IN
3	秒 (BCD)	0000 ~ 0059	IN

下表為時間 A 的格式。

偏移值	資料內容	資料範圍 (BCD)	輸出輸入
0	年 (BCD)	0000 ~ 0099	IN/OUT
1	月 / 日 (BCD)	高位元組 (月)：01 ~ 12 低位元組 (日)：01 ~ 31	IN/OUT
2	時 / 分 (BCD)	高位元組 (時)：00 ~ 23 低位元組 (分)：00 ~ 59	IN/OUT
3	秒 (BCD)	0000 ~ 0059	IN/OUT
4	累計秒數	可將年 / 月 / 日 / 時 / 分等運算結果變更為秒數 (長整數)	IN/OUT
5			

當運算結果超過上表所示的範圍，將會輸出 9999 作為秒資料，而不會更新時間 A，此時狀態位元將變為 1。

若運算結果符合所規定的範圍，則狀態位元將變為 0。

補充　一年以 365 天為計算單位。不考慮閏年。
而且不計算月數和日數。僅進行日數計算。

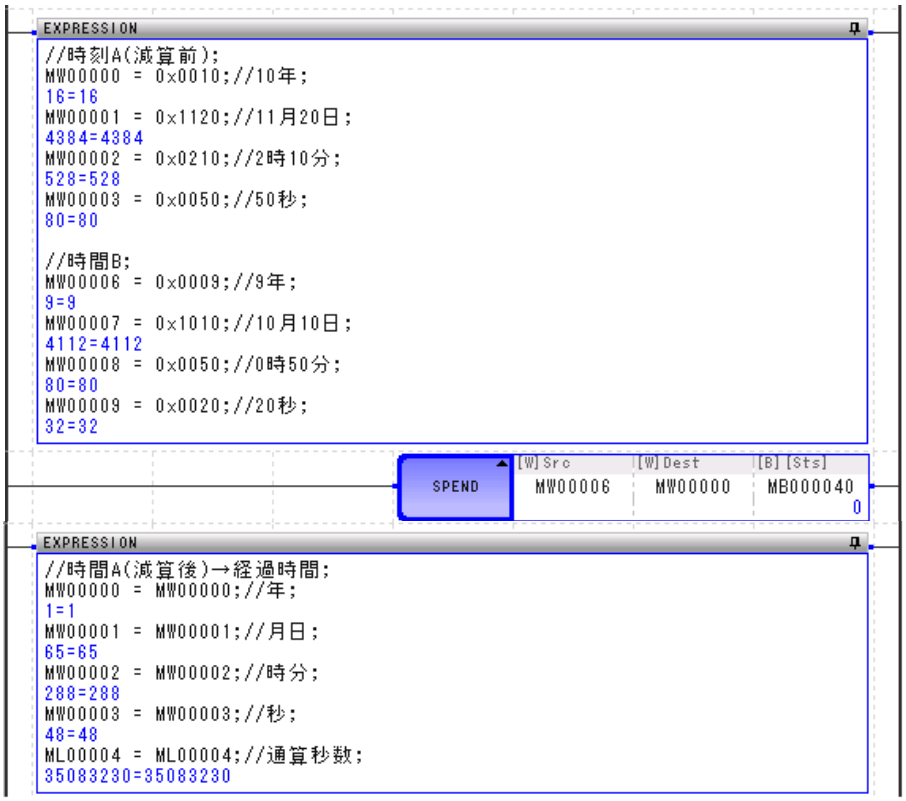
程式範例

下表為使用 SPEND 指令來編寫階梯圖程式之條件範例。

下表包含了執行指令前的時間 A (10 年 11 月 20 日 2 時 10 分 50 秒) 及時間 B (9 年 10 月 10 日 0 時 50 分 20 秒)。

	指令執行前的時間 A	時間 B
年	MW00000 = 0010H (10 年)	MW00006 = 0009H (9 年)
月 / 日	MW00001 = 1120H (11 月 20 日)	MW00007 = 1010H (10 月 10 日)
時 / 分	MW0002 = 0210H (2 時 10 分)	MW00008 = 0050H (0 時 50 分)
秒	MW00003 = 0050H (50 秒)	MW00009 = 0020H (20 秒)

以下為根據上述條件將時間 A 減去時間 B，然後再將所經過的時間儲存為時間 A 之程式範例。



下表為使用 SPEND 指令後所得到的結果。

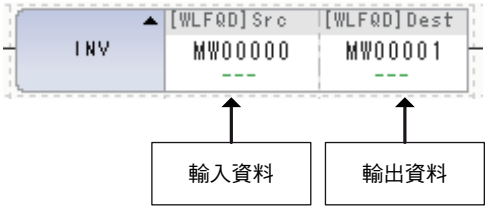
	執行指令後的時間 A
年	MW00000 = 1 = 0001H (1 年)
月 / 日	MW00001 = 65 = 0041H (0 月 41 日)
時 / 分	MW00002 = 288 = 0120H (1 時 20 分)
秒	MW00003 = 48 = 0030H (30 秒)
累計秒數	ML00004 = 35083230

符號反轉 (INV)

可將輸出資料的符號反轉，並將結果儲存為輸出資料。

格式

所採用之格式如下。



圖示　：INV
按鍵輸入：INV

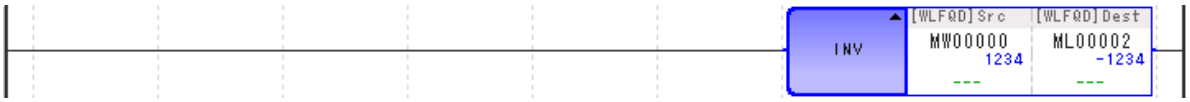
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	O	O	O	O	O	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	O*	O*	×	O	×

* C、# 暫存器除外

程式範例

以下為假設輸出資料 A (MW00000) 為 1234 時，如何將符號反轉，並將結果儲存為輸出資料 (ML00002) 之程式範例。

−1 × MW00000(1234) → ML00002 = −1234

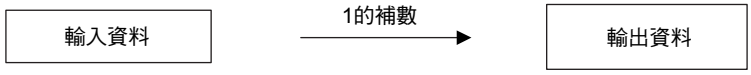


補充　為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果也不相同。

☞ 第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)

1 的補數 (COM)

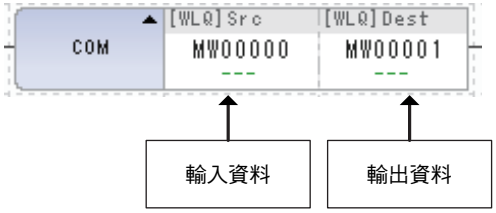
可將輸入資料 1 的補數儲存為輸出資料。



(註)可將 2 進位制格式輸入資料的 0 和 1 反轉，並將結果儲存於輸出資料。

格式

所採用之格式如下。



圖示 ：1 COM

按鍵輸入： COM

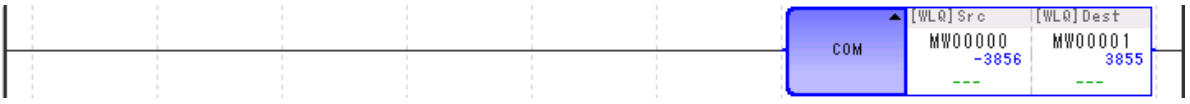
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	O	O	O	×	×	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	×	×	×	O	×

* C、# 暫存器除外

程式範例

以下為輸入資料 (MW00000) -3856 (HF0F0) 的 1 之補數被儲存為輸出資料 (MW00001) 之程式範例。

MW00000 = –3856(HF0F0) → MW00001 = 3855(H0F0F)

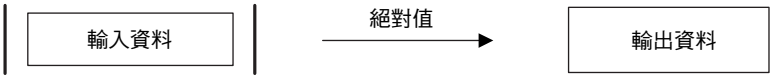


補充 為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ 第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)

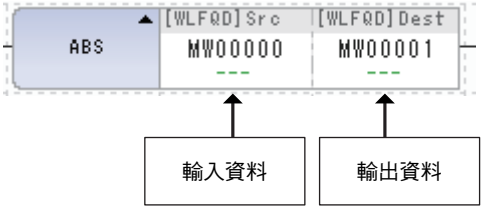
絕對值轉換 (ABS)

可將輸入資料的絕對值儲存為輸出資料。



格式

所採用之格式如下。



圖示 ： |n|

按鍵輸入： ABS

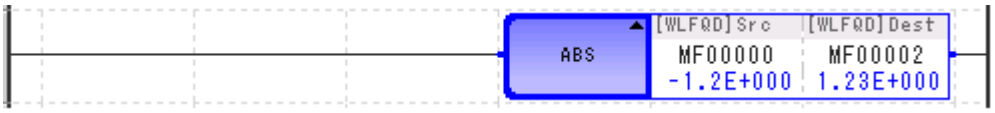
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	O	O	O	O	O	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	O*	O*	×	O	×

* C、# 暫存器除外

程式範例

以下為輸入資料 (MF00000)-1.23 的絕對值被儲存為輸出資料 (MF00002) 之程式範例。

| MF00000(−1.23) | → MF00002=1.23



補充 為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ 第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)

轉換為 2 進位制 (BIN)

可將 BCD 格式的輸入資料轉換為二進制資料 (BIN 轉換)，並將結果儲存為輸出資料。

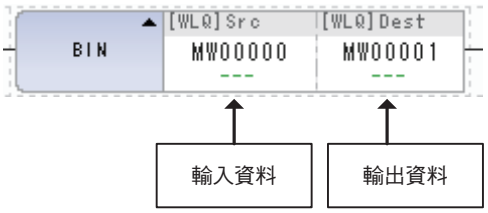
若輸入資料的格式並非 BCD (如 123FH 等)，那麼使用 BIN 轉換指令將無法產生正確的結果。

(註)若BCD格式的輸入資料為abcd時，則必須利用以下算式來運算輸出資料。
輸出資料 = (a x 1000) + (b x 100) + (c x 10) + d

補充　輸入資料及輸出資料通常會以 10 進位制顯示於畫面上。

格式

所採用之格式如下。



圖示　：　BIN / BCD
按鍵輸入：BIN

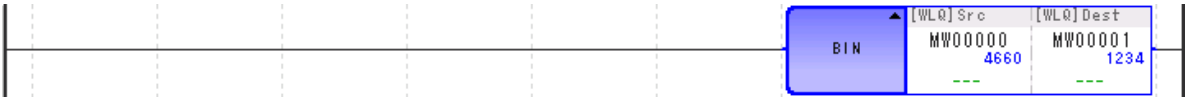
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	O	O	O	×	×	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	×	×	×	O	×

* C、# 暫存器除外

程式範例

以下為輸入資料 A (MW00000) 的 BCD (1234H (4660)) 被轉換為二進制資料 (轉換為 10 進位制的 1234)，且將結果儲存為輸出資料 (MW00001) 之程式範例。

MW00000 = 1234H ：(1 × 1000) + (2 × 100) + (3 × 10) + 4　→　MW00001 = 1234



補充　為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果也不相同。

☞　第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)

BCD 轉換 (BCD)

可將二進制資料的輸出資料轉換為 BCD，並將結果儲存為輸出資料。

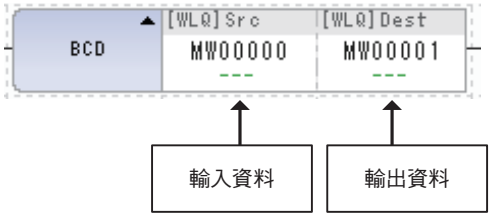
當輸入資料大於 9999 或是輸入資料為負數時，將無法產生正確的結果。

(註)若輸入資料以10進位制表示為abcd時，輸出資料必須利用下列算式來運算。
輸出資料 = (a x 4096) + (b x 256) + (c x 16) + d

補充　輸入資料及輸出資料通常會以 10 進位制顯示於畫面上。

格式

所採用之格式如下。



圖示　：BCD BIN
按鍵輸入：BCD

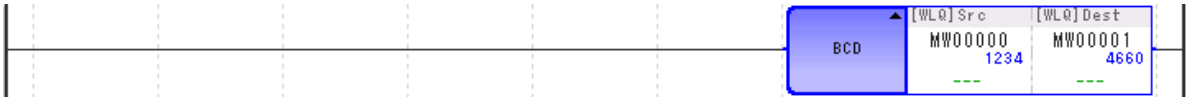
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	○	○	○	×	×	×	○	○
輸出資料 (Dest)	×	○*	○*	○*	×	×	×	○	×

* C、# 暫存器除外

程式範例

以下為輸入資料 A (MW00000) 的二進制資料 (10 進位制為 1234) 被轉換為 BCD (1234H (4660)) 且將結果儲存為輸出資料 (MW00001) 之程式範例。

MW00000 = 1234 ：(1 × 4096) + (2 × 256) + (3 × 16) + 4　→　MW00001 = 1234H(4660)

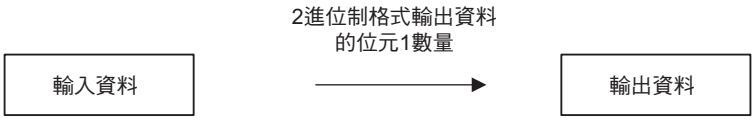


補充　為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

☞ 第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)

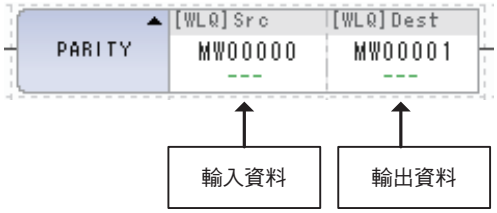
同位轉換 (PARITY)

可用來計算 2 進位制格式輸入資料的位元 1 數量，並將結果儲存為輸出資料。



格式

所採用之格式如下。



圖示 ： 0101 #?
按鍵輸入： PARITY

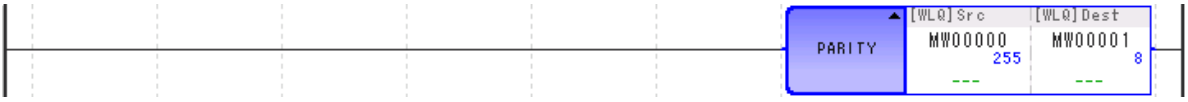
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	○	○	○	×	×	×	○	○
輸出資料 (Dest)	×	○*	○*	○*	×	×	×	○	×

* C、# 暫存器除外

程式範例

以下為輸入資料 A (MW00000) 的 255 (H00FF) 的位元 1 數量被儲存為輸出資料 (MW00001) 之程式範例。

MW00000(H0FF) 的位元 1 數量 = 8 → MW00001 = 8



補充 為不同資料類型進行運算時，輸出暫存器的資料類型不同，運算結果亦各異。

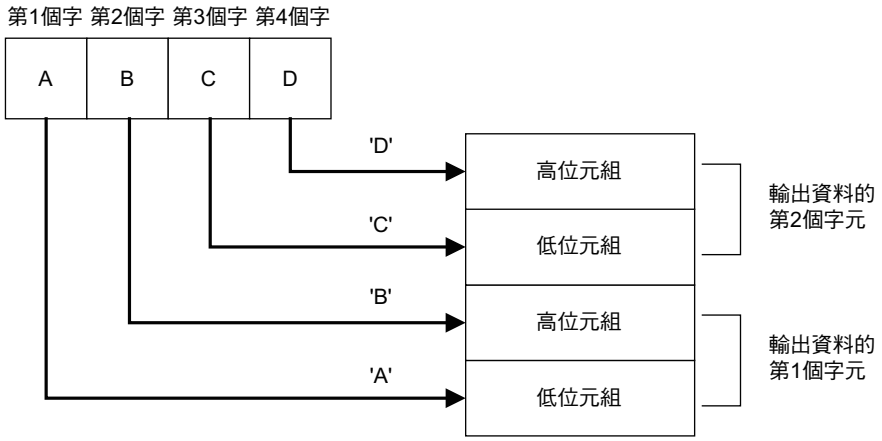
☞ 第 3 章 暫存器 – ◆ 不同資料類型之演算注意事項 (第 3-8 頁)

ASCII 轉換 1 (ASCII)

可將輸入字串轉換為 ASCII 碼，再將結果儲存為輸出資料。可區分大小寫。

最多可輸入 32 個字 (16 個字元)。

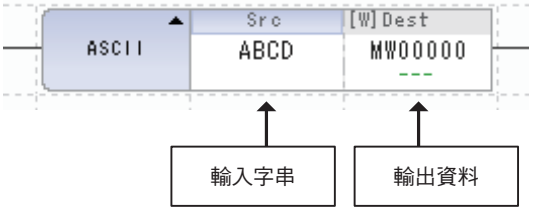
下圖為輸入字串 ASCII 碼的儲存位置。



(註)若字串為奇數時，最後一個字元的高位元組將變為 0。

格式

所採用之格式如下。



圖示 ： ASCII

按鍵輸入：ASCII

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入字串 (Src)					×*1				
輸出資料 (Dest)	×	○*2	×	×	×	×	×	×	×

*1. ASCII 字元

*2. C、# 暫存器除外

程式範例

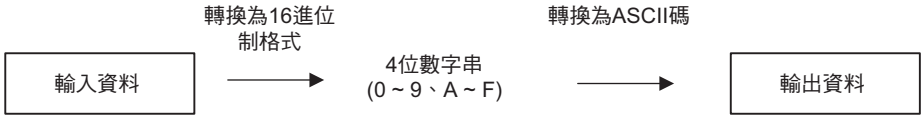
以下為輸入字串 Hello 被轉換為 ASCII 碼，然後再將結果儲存為輸出資料 (MW00000) 之程式範例。

下表為 ASCII 碼儲存後之結果。

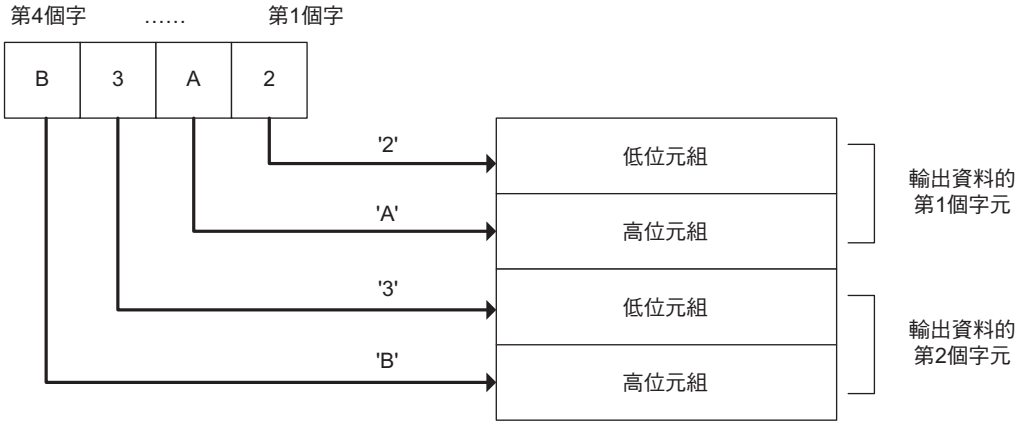
位址	ASCII 碼	字元
MW00000 (低位元組)	48 H	H
MW00000 (高位元組)	65 H	e
MW00001 (低位元組)	6 CH	l
MW00001 (高位元組)	6 CH	l
MW00002 (低位元組)	6 FH	o
MW00002 (高位元組)	0	–

ASCII 轉換 2 (BINASC)

可將被儲存為輸入資料 (1 字元) 的 16 位元二進制資料轉換為 16 進位制的 4 位數 ASCII 碼，並將結果儲存為輸出資料 (2 字元)。

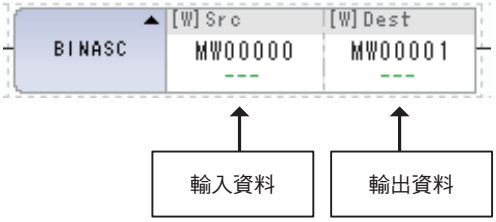


下圖為假設輸入資料為 10811 (2A3BH) 時 ASCII 碼之儲存位置。



格式

所採用之格式如下。



圖示 ： BIN ASC

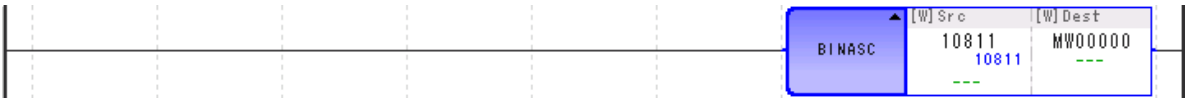
按鍵輸入： BINASC

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	O	×	×	×	×	×	×	O
輸出資料 (Dest)	×	O*	×	×	×	×	×	×	×

* C、# 暫存器除外

程式範例

以下為輸入資料 10811 (2A3BH) 被轉換為 ASCII 碼，然後再將結果儲存為輸出資料 (MW00000) 之程式範例。

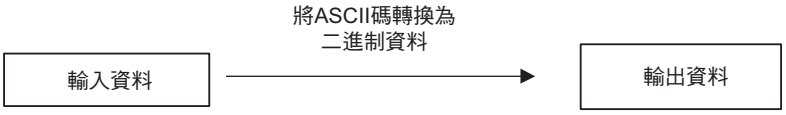


下表為 ASCII 碼儲存後之結果。

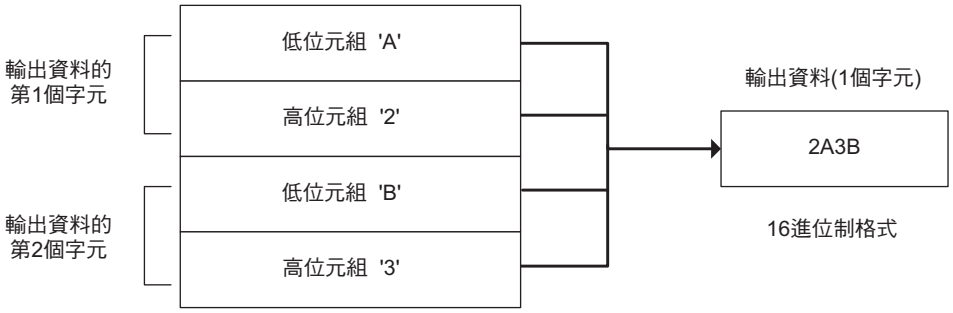
位址	ASCII 碼	字元
MW00000 (低位元組)	32 H	2
MW00000 (高位元組)	41 H	A
MW00001 (低位元組)	33 H	3
MW00001 (高位元組)	42 H	B

ASCII 轉換 3 (ASCBIN)

可將被儲存為輸入資料 (2 個字元) 的 16 進位制 4 位數 ASCII 碼所代表的數值轉換為 16 位元的二進制資料，並將結果儲存為輸出資料 (1 個字元)。

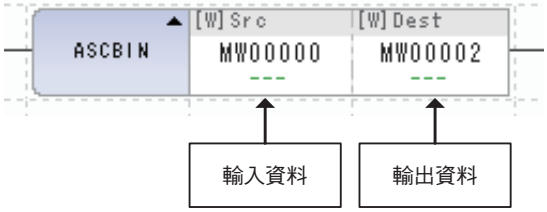


下圖為假設輸入資料的第 1 個字元為 4132H ('2' 'A')、第 2 個字元為 4232H ('3' 'B') 時之輸出資料。



格式

所採用之格式如下。



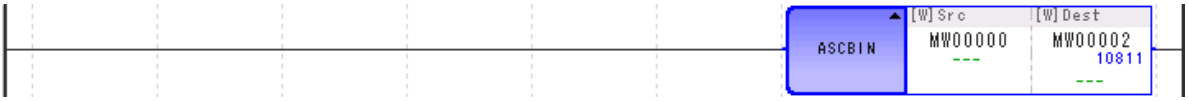
圖示 ： ASC BIN
按鍵輸入：ASCBIN

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	○	×	×	×	×	×	×	×
輸出資料 (Dest)	×	○*	×	×	×	×	×	×	×

* C、# 暫存器除外

程式範例

以下為利用 ASCBIN 指令將輸入資料 (MW00000) 儲存為輸出資料 (MW00002) 之程式範例。



下表為 ASCII 碼儲存後的結果。

位址	ASCII 碼	字元
MW00000 (低位元組)	32 H	2
MW00000 (高位元組)	41 H	A
MW00001 (低位元組)	33 H	B
MW00001 (高位元組)	42 H	3

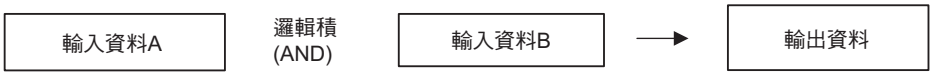
輸出資料 (MW00000) 為 10811(2A3BH)。

4.4 邏輯運算 / 比較指令

邏輯積 (AND)

可用來運算輸入資料 A 和輸入資料 B 的邏輯積，並將結果儲存為輸出資料。

僅適用整數型及長整數型資料。

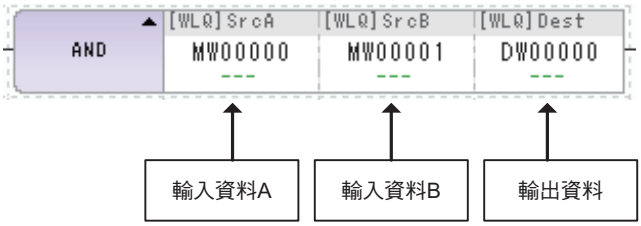


請利用以下的真值表，運算輸入資料的每個位元。

輸入資料 A	輸入資料 B	輸出資料
0	0	0
0	1	0
1	0	0
1	1	1

格式

所採用之格式如下。



圖示 ： ^

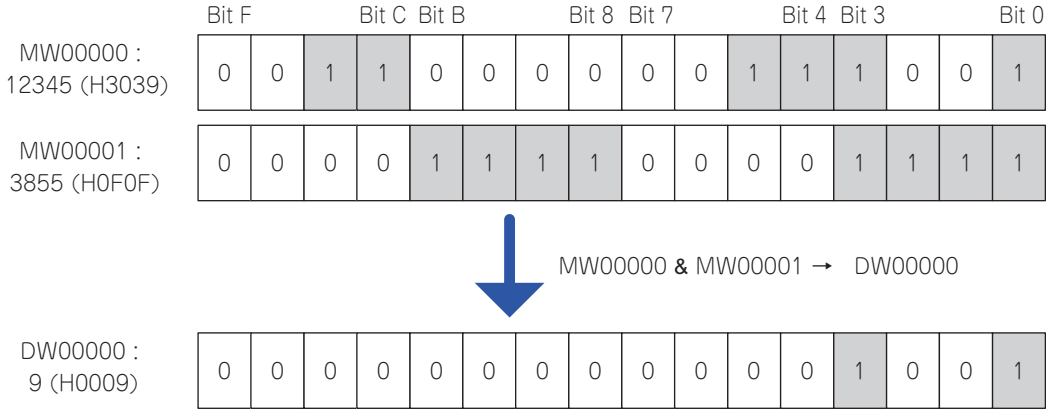
按鍵輸入： &

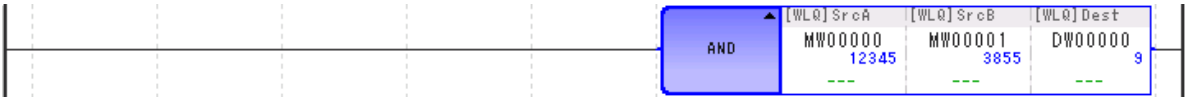
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	O	O	O	×	×	×	O	O
輸入資料 B (SrcB)	×	O	O	O	×	×	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	×	×	×	O	×

* C、# 暫存器除外

程式範例

以下為運算輸入資料 A (MW00000):12345 (H3039)、輸入資料 B (MW00001): 3855 (H0F0F) 的邏輯積，並將結果儲存為輸出資料 (DW00000) 之程式範例。

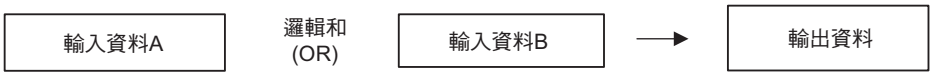




邏輯和 (OR)

可用來運算輸入資料 A 和輸入資料 B 的邏輯和，並將結果儲存為輸出資料。

僅適用整數型及長整數型資料。

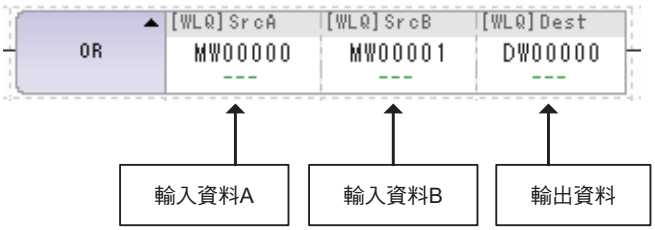


請利用以下的真值表，運算輸入資料的每個位元。

輸入資料 **A**	輸入資料 **B**	輸出資料
0	0	0
0	1	1
1	0	1
1	1	1

格式

所採用之格式如下。



圖示　：

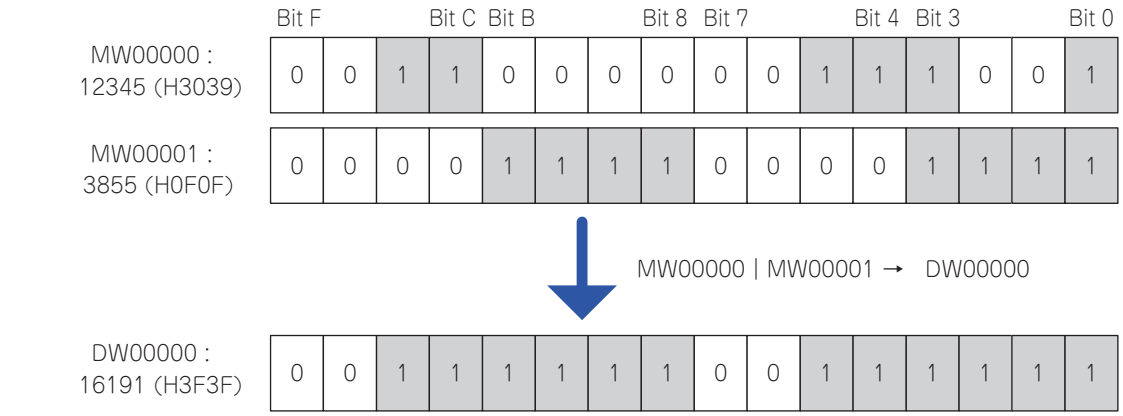
按鍵輸入：

輸出輸入項目	適用之資料類型								
	B	**W**	**L**	**Q**	**F**	**D**	**A**	**索引**	**常數**
輸入資料 A (SrcA)	×	O	O	O	×	×	×	O	O
輸入資料 B (SrcB)	×	O	O	O	×	×	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	×	×	×	O	×

* C、# 暫存器除外

程式範例

以下為運算輸入資料 A (MW00000):12345 (H3039)、輸入資料 B (MW00001): 3855 (H0F0F) 的邏輯和，並將結果儲存為輸出資料 (DW00000) 之程式範例。



互斥或 (XOR)

可用來演算輸入資料 A 和輸入資料 B 的互斥或，並將結果儲存為輸出資料。

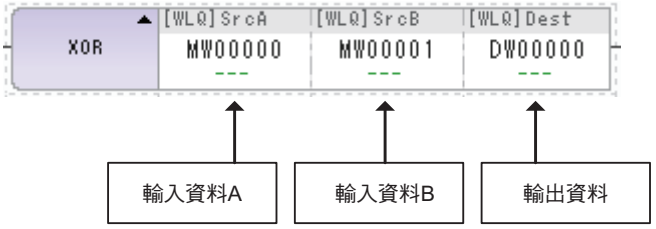
僅適用整數型及長整數型資料。

請利用以下的真值表，運算輸入資料的每個位元。

輸入資料 A	輸入資料 B	輸出資料
0	0	0
0	1	1
1	0	1
1	1	0

格式

所採用之格式如下。



圖示 ：⊕

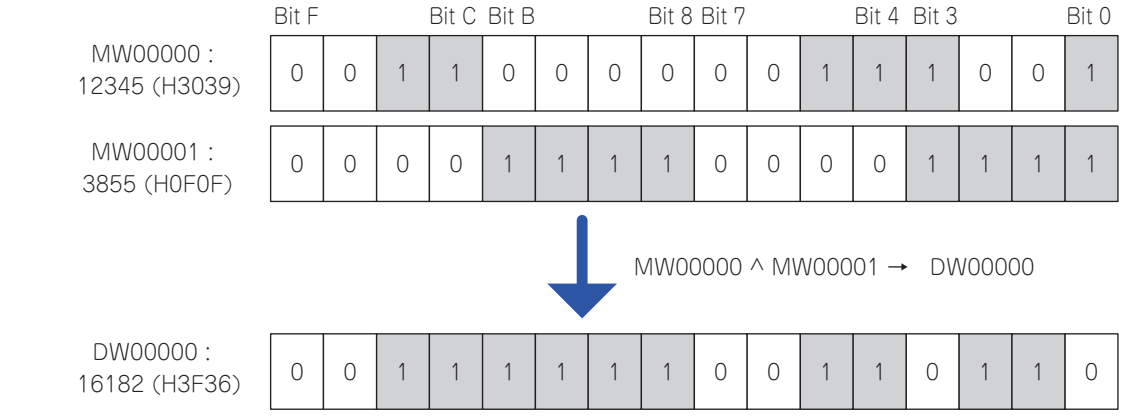
按鍵輸入：^

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	O	O	O	×	×	×	O	O
輸入資料 B (SrcB)	×	O	O	O	×	×	×	O	O
輸出資料 (Dest)	×	O*	O*	O*	×	×	×	O	×

* C、# 暫存器除外

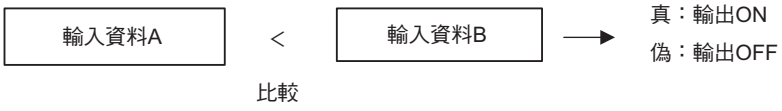
程式範例

以下為演算輸入資料 A (MW00000): 12345 (H3039)、輸入資料 B (MW00001): 3855 (H0F0F) 的互斥或，並將結果儲存為輸出資料 (DW00000) 之程式範例。



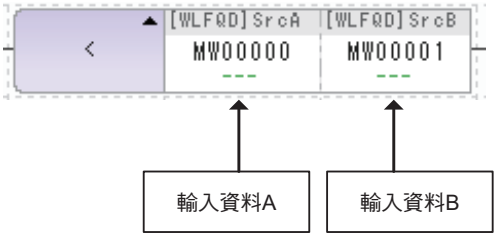
比較 (<)

將輸入資料 A 和輸入資料 B 互相比較，並將結果輸出為位元。



格式

所採用之格式如下。



圖示 ：<

按鍵輸入：<

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○

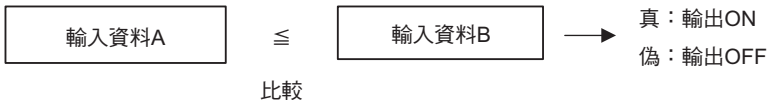
程式範例

當輸入資料 A (MW00000) 為 90、輸入資料 B 為 100 且為常數時，由於輸入資料 A 小於輸入資料 B，因此比較結果為真，這時候將輸出 ON，並執行右方的 INC 指令。

補充 若輸入資料為實數型資料時，顯示於 MPE720 上的數值將出現些微的精確度誤差，因此有可能發生與比較指令的執行結果不一致的情形。

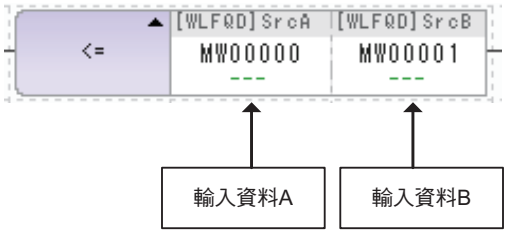
比較 (≦)

將輸入資料 A 和輸入資料 B 互相比較，並將結果輸出為位元。



格式

所採用之格式如下。



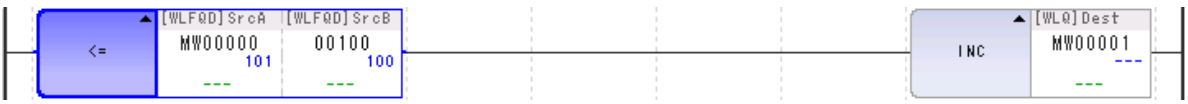
圖示 ：≦

按鍵輸入：<=

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○

程式範例

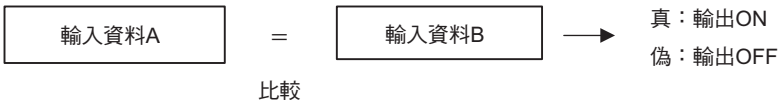
若輸入資料 A (MW00000) 為 101、輸入資料 B 為 100 且為常數時，由於輸入資料 A 大於輸入資料 B，因此比較結果為偽，這時候將輸出 OFF，且不執行右方的 INC 指令。



補充 若輸入資料為實數型資料時，顯示於 MPE720 上的數值將出現些微的精確度誤差，因此有可能發生與比較指令的執行結果不一致的情形。

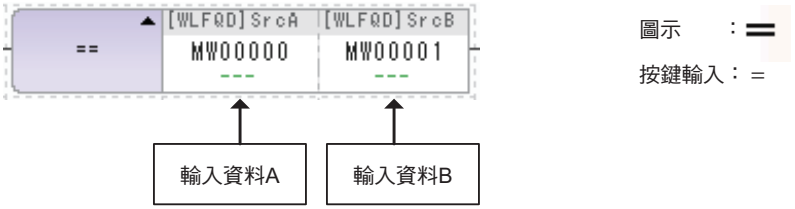
比較 (=)

將輸入資料 A 和輸入資料 B 互相比較，並將結果輸出為位元。



格式

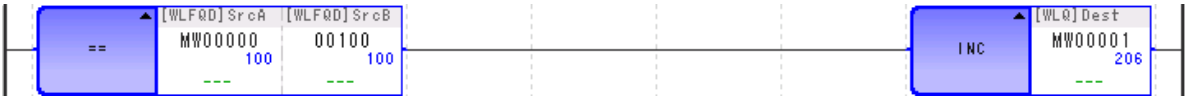
所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○

程式範例

若輸入資料 A (MW00000) 為 100、輸入資料 B 為 100 且為常數時，由於輸入資料 A 等於輸入資料 B，因此比較結果為真，這時候將輸出 ON，並執行右方的 INC 指令。



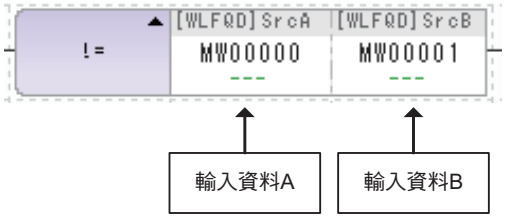
補充 若輸入資料為實數型資料時，顯示於 MPE720 上的數值將出現些微的精確度誤差，因此有可能發生與比較指令的執行結果不一致的情形。

比較 (≠)

將輸入資料 A 和輸入資料 B 互相比較，並將結果輸出為位元。

格式

所採用之格式如下。

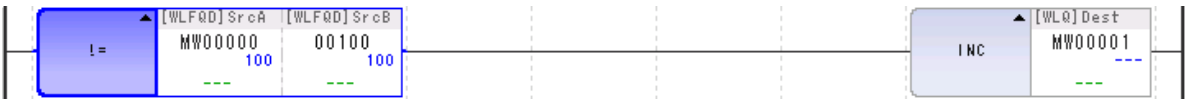


圖示　：≠
按鍵輸入：<>

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○

程式範例

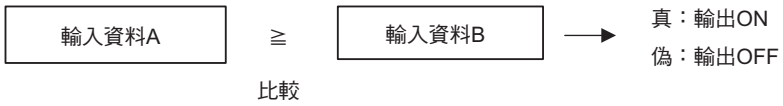
若輸入資料 A (MW00000) 為 100、輸入資料 B 為 100 且為常數時，由於輸入資料 A 和輸入資料 B 相同，因此比較結果為偽，這時候將輸出 OFF，且不執行右方的 INC 指令。



補充　若輸入資料為實數型資料時，顯示於 MPE720 上的數值將出現些微的精確度誤差，因此有可能發生與比較指令的執行結果不一致的情形。

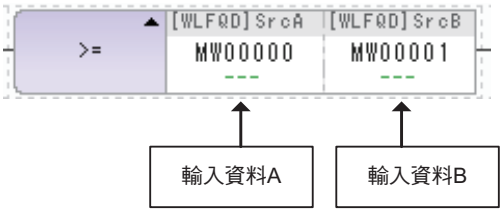
比較 (≧)

將輸入資料 A 和輸入資料 B 互相比較，並將結果輸出為位元。



格式

所採用之格式如下。

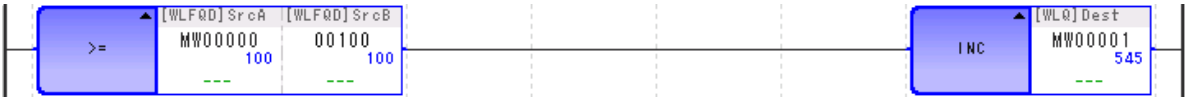


圖示　：≧
按鍵輸入：>=

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○

程式範例

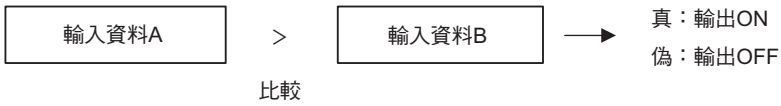
若輸入資料 A (MW00000) 為 100、輸入資料 B 為 100 且為常數時，由於輸入資料 A 大於輸入資料 B，因此比較結果為真，這時候將輸出 ON，並執行右方的 INC 指令。



補充　若輸入資料為實數型資料時，顯示於 MPE720 上的數值將出現些微的精確度誤差，因此有可能發生與比較指令的執行結果不一致的情形。

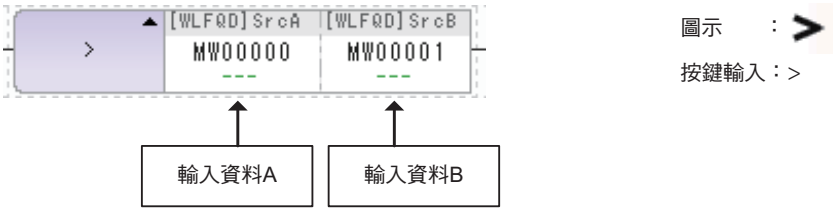
比較 (>)

將輸入資料 A 和輸入資料 B 互相比較，並將結果輸出為位元。



格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 A (SrcA)	×	○	○	○	○	○	×	○	○
輸入資料 B (SrcB)	×	○	○	○	○	○	×	○	○

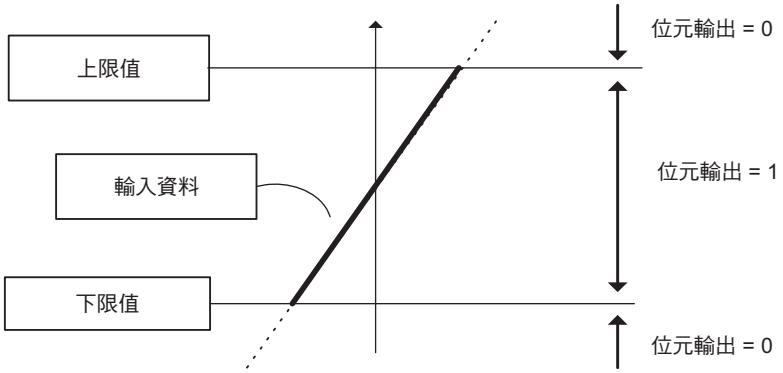
程式範例

若輸入資料 A (MW00000) 為 100、輸入資料 B 為 100 且為常數時，由於輸入資料 A 並未大於輸入資料 B，因此比較結果為偽，這時候將輸出 OFF，並執行右方的 INC 指令。

補充 若輸入資料為實數型資料時，顯示於 MPE720 上的數值將出現些微的精確度誤差，因此有可能發生與比較指令的執行結果不一致的情形。

檢查範圍 (RCHK)

可用來檢查輸入資料的數值是否在上限值與下限值之間的範圍內，並將結果輸出為位元。



· **位元輸出 = 1**

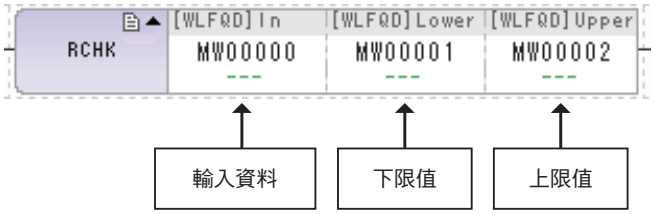
輸入資料的數值在上限值與下限值之間的範圍內時，將會輸出位元 1。

· **位元輸出 = 0**

輸入資料的數值不在上限值與下限值之間的範圍內時，將會輸出位元 0。

格式

所採用之格式如下。



圖示 ： R CHK

按鍵輸入： RCHK

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (In)	×	O	O	O	O	O	×	O	O
下限值 (Lower)	×	O	O	O	O	O	×	O	O
上限值 (Upper)	×	O	O	O	O	O	×	O	O

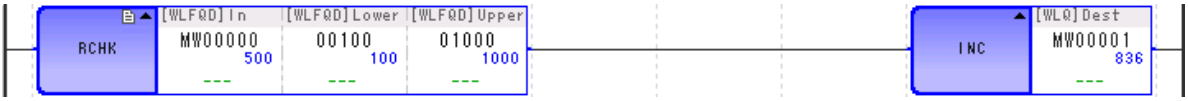
補充　請設定為下限值 ≦ 上限值。若下限值 > 上限值時，將無法保證能穩定動作。

程式範例

以下為執行 RCHK 指令之程式範例。

· **若輸入資料 (MW00000) = 80、下限值 = 100、上限值 = 1000 時**
由於輸入資料的數值小於下限值，將輸出位元 0，且不執行右方的 INC 指令。

· **若輸入資料 (MW00000) = 500、下限值 = 100、上限值 = 1000 時**
由於輸入資料的數值在上限值與下限值之間的範圍內，因此將輸出位元 1，且執行右方的 INC 指令。



· **若輸入資料 (MW00000) = 1000、下限值 = 100、上限值 = 1000 時**
由於輸入資料的數值符合上限值與下限值之間的範圍，因此將輸出位元 1，且執行右方的 INC 指令。

4.5 程式控制指令

叫出圖面 (SEE)

可用來由母圖面參照子圖面，或是由子圖面參照孫圖面。

格式

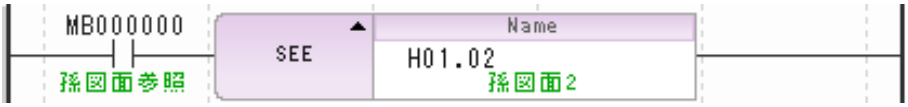
所採用之格式如下。

圖示　：SEE
按鍵輸入：SEE

輸出輸入項目	適用之資料類型
程式編號 (Name)	無法利用暫存器指令，必須直接指定程式編號。 您所指定的程式名稱將顯示於程式編號的下方。

程式範例

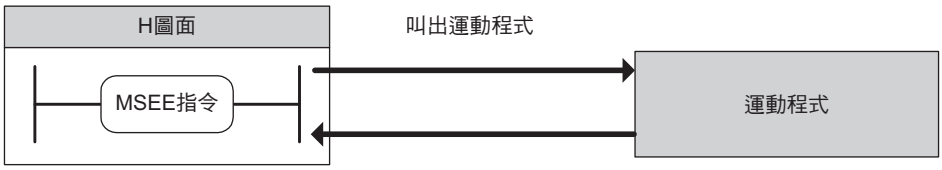
若繼電器 (MB000000) 為 ON 時，利用 SEE 指令來叫出 H01.02。接著，執行處理作業，並利用 SEE 指令執行以下步驟，若繼電器 (MB000000) 為 OFF 時，則不叫出 H01.02。



叫出運動程式 (MSEE)

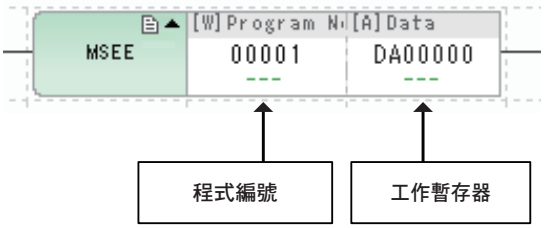
可用來參照您所指定的運動程式。

運動程式只能由 H 圖面參照。



格式

所採用之格式如下。



圖示 ： M See

按鍵輸入：MSEE

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
程式編號 (Program No.)	×	○*	×	×	×	×	×	×	○
工作暫存器 (Data)	×	×	×	×	×	×	○*	×	×

* 僅適用 M、D 暫存器

下表為工作暫存器之架構。

位址	資料類型	名稱	內容	輸出輸入
0	W	狀態旗標	運動程式狀態旗標	OUT
1	W	控制訊號	運動程式控制訊號	IN
2	W	插補用覆寫	執行插補指令時之覆寫值 範圍：0 ~ 32767 單位：1 = 0.01%	IN
3	W	系統工作編號	用來叫出運動程式之系統工作編號	IN

補充 程式編號的設定範圍為 1 ~ 512。

如欲進一步瞭解運動程式，請參閱下述手冊。

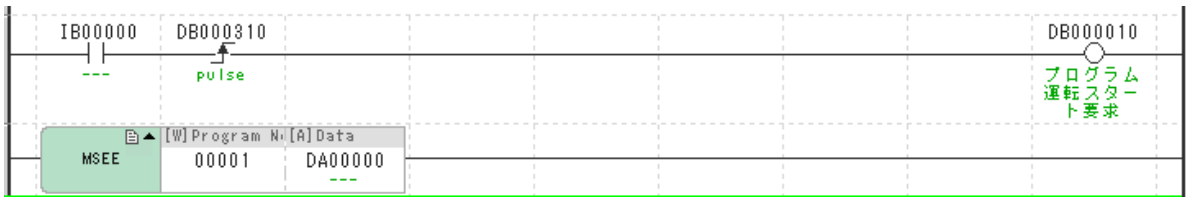
📖 **MP3000 系列　運動程式　程式編寫手冊 (資料編號：SIJP C880725 04)**

程式範例

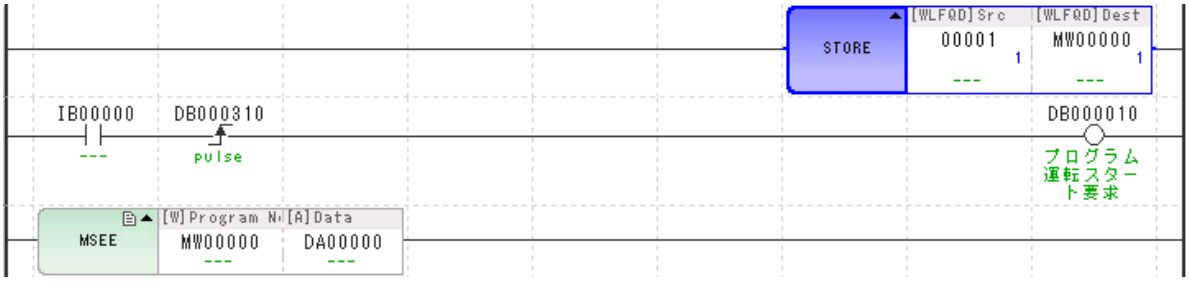
以下為執行程式編號 1 的運動程式 (MPM001) 之程式範例。

當繼電器 (IB00000) ON 時，要求程式開始運轉的控制訊號 (DB000010) 就會 ON，並且執行運動程式 (MPM001)。

- **直接指定**
  直接將程式編號設定為 「1」。



- **間接指定**
  將程式編號設定為 「MW00000」。



重要

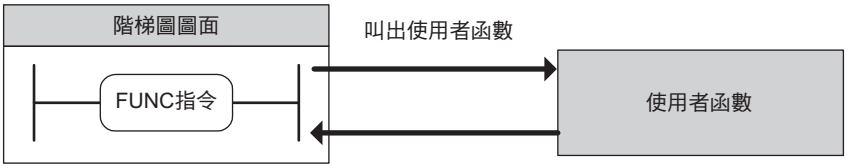
MSEE 指令將持續執行，直到運動程式執行完成。
使用間接指定方式時，到運動程式執行完畢為止，請勿變更暫存器的值。

叫出使用者函數 (FUNC)

可用來參照使用者函數。參照使用者函數前，必須先為函數進行定義。

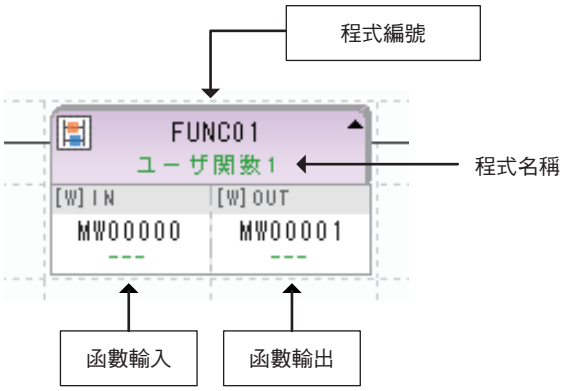
如欲進一步瞭解使用者函數，請參閱以下章節。

☞ 1.3 概要 – 使用者函數 (第 1-13 頁)



格式

所採用之格式如下。



圖示　：FUNC

按鍵輸入：FUNC

輸出輸入項目	適用之資料類型
程式編號 (Name)	無法利用暫存器指定，必須直接指定程式編號。 指定的程式名稱將顯示於指令的上方。
函數輸入	適用於已利用函數輸入定義設定完成的暫存器。
函數輸出	適用於已利用函數輸出定義設定完成的暫存器。

程式範例

如欲瞭解使用者函數之相關程式範例，請參閱以下章節。

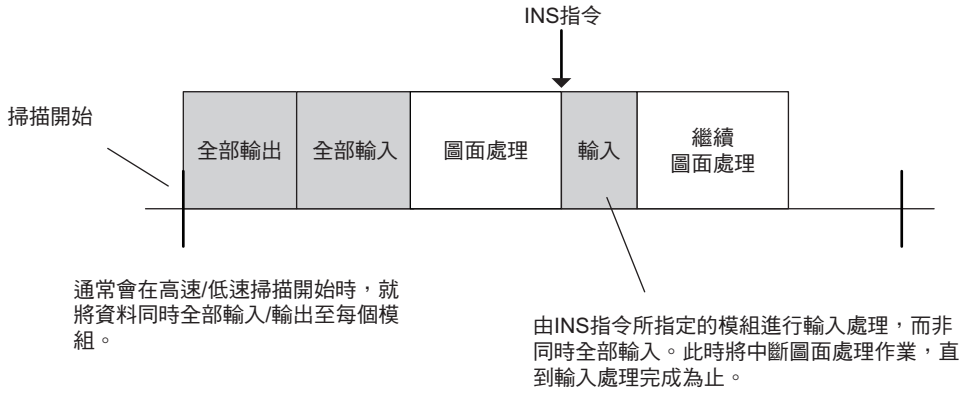
☞ 1.3 概要 - 使用者函數 (第 1-13 頁)

連續執行型直接輸入 (INS)

利用使用者程式來執行 INS 指令，而非一開始進行高速掃描 / 低速掃描，就由系統將資料同時全部輸出。執行 INS 指令時，將依照您所設定的參數資料表的內容，並利用您所指定的模組進行輸入。輸入動作完成前，將不會執行下一個指令。

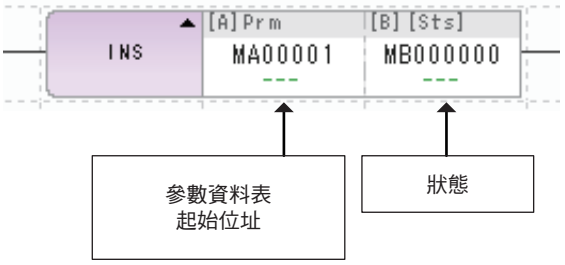
可指定之模組如下。

- LIO-01/02 模組 (LIO)
- LIO-04/05 模組 (LIO32)
- LIO-06 模組 (MIXIO)
- AI-01 模組 (AI)



格式

所採用之格式如下。



圖示 ： IN S

按鍵輸入：INS

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*1	×	×
狀態 (Sts)*2	○*1	×	×	×	×	×	×	×	×

*1. C、# 暫存器除外
*2. 可省略

下表為參數資料表的架構。

位址	資料類型	符號	名稱	規格	輸出輸入
0	W	RSSEL	指定元件 1	指定您所要輸入的模組。	IN
1	W	MDSEL	指定元件 2		IN
2	W	STS	狀態	依位元別輸出每個字元的輸入狀態。 0：正常 1：異常	OUT
3	W	N	字元數	指定連續輸入的字元數。	IN
4	W	ID1	輸入資料 1	輸出先前已輸入的資料。 發生錯誤時，將儲存為 0。	OUT
:	:	:	:		:
N + 3	W	IDN	輸入資料 N		OUT

下表為 MP3000 系列適用之所有參數說明。

參數	模組名稱			
	LIO-01/02 (LIO)	LIO-04/05 (LIO32)	LIO-06 (MIXIO)	AI-01 (AI)
RSSEL	可用來指定模組裝置槽 / 元件 /SLOT/ 附屬 SLOT 編號等。 16 進位制格式：zxuyH x ：裝置槽編號 (1 ~ 7) u ：元件編號 (0 ~ 4)*1 y ：SLOT 編號 (0 ~ 9) z ：附屬 SLOT 編號 (1~ 最大值取決於模組實際規格)			
MDSEL	0 (未使用)	偏移值： 0 ~ 1	頻道編號 –1：0 ~ 1	頻道編號 –1：0 ~ 7
STS	固定為 0	固定為 0	固定為 0	*2
N	1	1 ~ 2 –MDSEL	1 ~ 2 –MDSEL	1 ~ 8 –MDSEL

*1. 若指定元件編號為 0，將以元件編號 =1 的方式來進行處理。

*2. 利用 INS 指令，指定 AI 模組詳細定義中被刪除配置的頻道時，則該頻道編號將會被輸出為位元。原因在於被刪除配置的頻道資料無法被讀取。以下為位元和頻道之間的關係。
位元 0：頻道 1
位元 1：頻道 2
位元 2：頻道 3
位元 3：頻道 4
位元 4：頻道 5
位元 5：頻道 6
位元 6：頻道 7
位元 7：頻道 8

程式範例

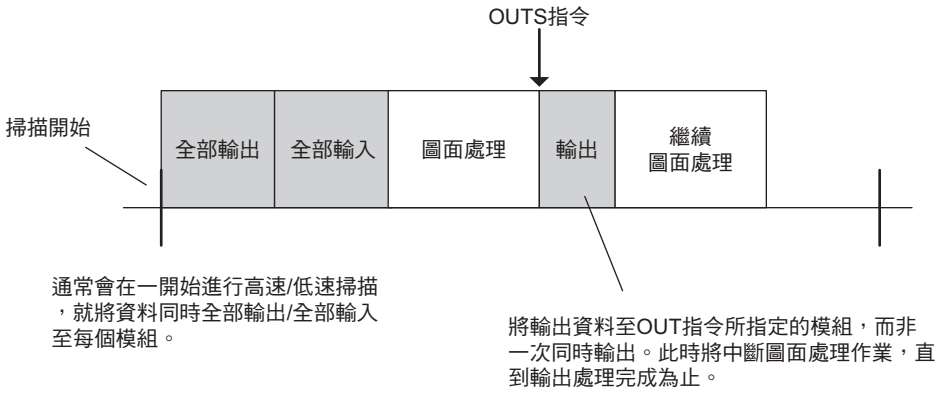
當 LIO-01 模組 LIO (附屬 SLOT 編號：1) 被連接到裝置槽 1、元件 1 的 SLOT2 時，這時候只要該模組輸入 1 個字元，LIO 的輸入資料就會被儲存為狀態 (MW00014)。

連續執行型直接輸出 (OUTS)

利用使用者程式來執行 OUTS 指令，而非一開始進行高速掃描 / 低速掃描，就由系統將資料同時全部輸出。執行 OUTS 指令時，將依照您所設定的參數資料表內容，直接將資料輸出至您所指定的模組。

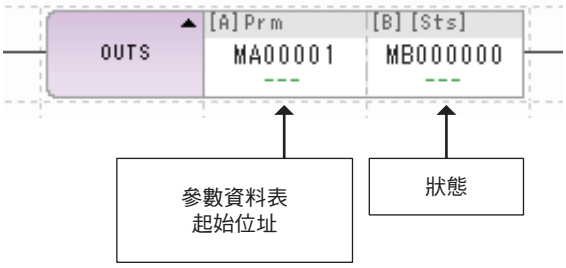
可指定之模組如下。

- LIO-01/02 模組 (LIO)
- LIO-04/05 模組 (LIO32)
- LIO-06 模組 (MIXIO)
- DO-01 模組 (DO)
- AO-01 模組 (AO)



格式

所採用之格式如下。



圖示 ： OUT S

按鍵輸入：OUTS

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*1	×	×
狀態 (Sts)*2	○*1	×	×	×	×	×	×	×	×

*1. C、# 暫存器除外

*2. 可省略

下表為參數資料表的架構。

位址	資料類型	符號	名稱	規格	輸出輸入
0	W	RSSEL	指定元件 1	指定您所要輸出的模組。	IN
1	W	MDSEL	指定元件 2		IN
2	W	STS	狀態	依位元別輸出每個字元的輸入狀態。 0：正常　1：異常	OUT
3	W	N	字元數	指定輸出字元數 (固定為 1)。	IN
4	W	OD1	輸出資料 1	設定您所要輸出的資料。	OUT
:	:	:	:		:
N + 3	W	ODN	輸出資料 N		OUT

下表為 MP3000 系列適用之所有參數說明。

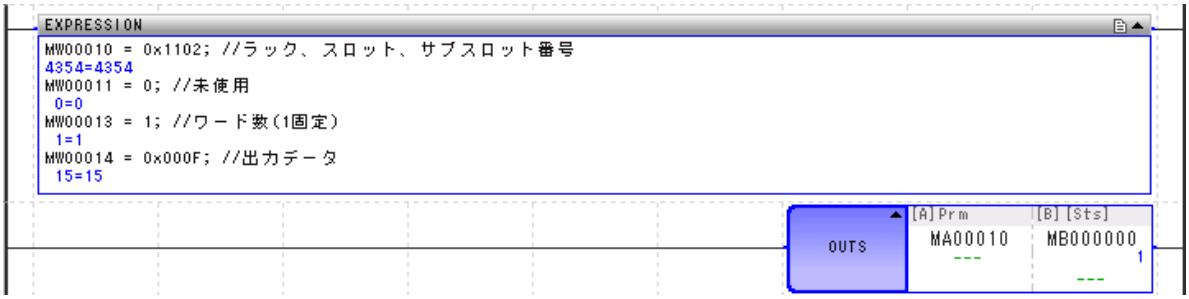
參數	模組名稱				
	LIO-01/02 (LIO)	LIO-04/05 (LIO32)	LIO-06 (MIXIO)	DO-01 (DO)	AO-01 (AO)
RSSEL	可用來指定模組裝置槽 / 元件 /SLOT/ 附屬 SLOT 編號等。 16 進位制格式：zxuyH x ：裝置槽編號 (1 ~ 7) u ：元件編號 (0 ~ 4)*1 y ：SLOT 編號 (0 ~ 9) z ：附屬 SLOT 編號 (1~ 最大值取決於模組實際規格)				
MDSEL	0 (未使用)	偏移值 : 0 ~ 1	偏移值 : 0 ~ 1	偏移值 : 0 ~ 3	頻道編號 –1：0 ~ 3
STS	固定為 0	固定為 0	固定為 0	固定為 0	*2
N	1	1 ~ 2 – MDSEL	1 ~ 2 – MDSEL	1 ~ 4 – MDSEL	1 ~ 4 – MDSEL

*1. 若指定元件編號為 0，將以元件編號 =1 的方式來進行處理。

*2. 利用 OUTS 指令，指定 AO 模組內容定義中被刪除配置的頻道時，則該頻道編號將會以位元方式被輸出。原因在於被刪除配置的頻道資料無法被讀取。以下為位元和頻道之間的關係。
位元 0：頻道 1
位元 1：頻道 2
位元 2：頻道 3
位元 3：頻道 4

程式範例

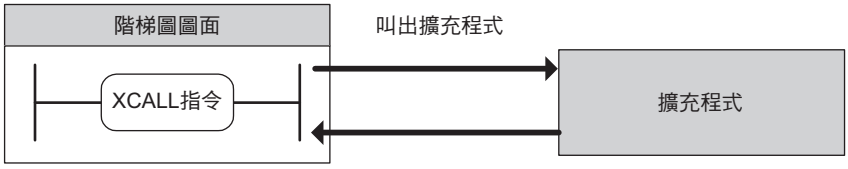
當 LIO-01 模組 LIO (附屬 SLOT 編號：1) 被連接到裝置槽 1、元件 1 的 SLOT2 時，這時候只要該模組輸出 1 個字元，狀態資料 (MW00014) 就會被輸出至 LIO。



執行擴充程式 (XCALL)

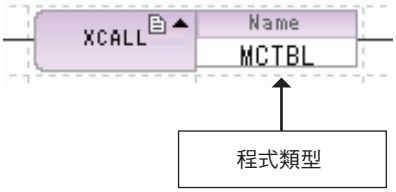
可用來執行擴充程式 (表格式程式 : 常數表)。MPE720 可將擴充程式轉換為階梯圖程式。利用 XCALL 指令即可執行轉換後的階梯圖程式。

同一個圖面雖然可以使用多個 XCALL 指令，但是卻無法多次叫出同一個擴充程式。



格式

所採用之格式如下。



圖示 ： X CALL

按鍵輸入： XCALL

輸出輸入項目	適用之資料類型
程式類型 (Name)	無法利用暫存器指定，設定類型如下。 MCTBL：常數表

程式範例

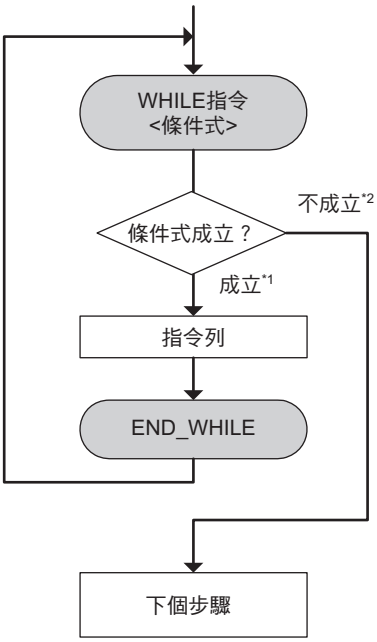
以下為叫出 MCTBL (常數表) 之程式範例。

補充 MP3000 系列不適用輸出輸入轉換表、互鎖表、零件組合表。

WHILE 陳述式 (WHILE、END_WHILE)

當 WHILE 指令的條件式成立時，就會執行位於 WHILE 指令和 END_WHILE 指令之間的程式，然後再回到 WHILE 指令。只要條件式成立，就會反覆執行動作。

若條件式不成立，END_WHILE 指令就會進入下一個步驟。WHILE 指令將無法再執行 END_WHILE 指令之間的指令列。



*1. 先執行指令列，再次回到 WHILE 指令。
*2. 進入下一個步驟，且不執行指令列。

格式

所採用之格式如下。

圖示　：WHILE　END_WHILE

按鍵輸入：WHILE, WEND

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
條件式	○*	○*	○*	○*	○*	○*	×	○*	○*

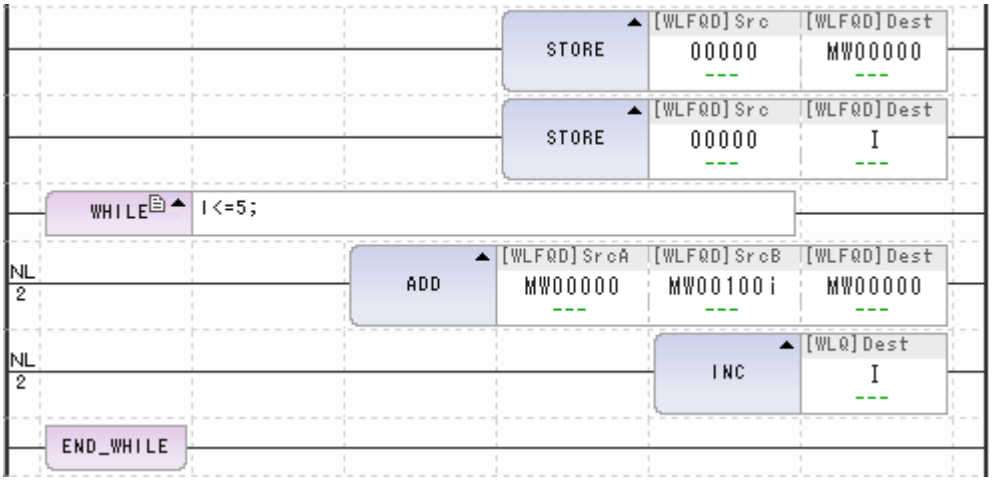
* 利用 EXPRESSION 來編寫。
請參閱以下章節為適合用來編寫的格式。
☞ 附錄 C EXPRESSION 指令格式

程式範例

以下係將 MW00100 ~ MW00105 相加，然後再將結果儲存為 MW00000 之程式範例。

由於條件式為 I ≦ 5，I 就會執行 0~5 的 ADD(+) 指令。

當 I = 6 時，由於條件式不成立，因此進入 END_WHILE 指令的下一個步驟。



重要

只要 WHILE 指令的條件式成立，就會反覆執行指令列。

條件式成立，或是花費較長時間判定條件式不成立時，恐將使得運動控制器的系統當機，此點需特別注意。

在上述範例中，一旦指令列內持續不遞增 I 的狀態，就會變成無限迴圈。

補充事項

◆ 適用之條件式

在 WHILE 指令的條件式中，只可編寫如以下輸出 bool 型 (真或偽) 結果的 EXPRESSION 算式。因此，若您所編寫的算式包含了指定運算子，將無法判讀。

算式範例	編寫	備註
MB000001 == true	OK	真：ON
MB000001 != false	OK	偽：OFF
MW00002 < 100	OK	–
MF00002 < sin(60.0)	OK	–
MW00001 == 0x00FF OK	OK	使用 16 進位制格式為 0x
MB000001 = true	NG	–
MW00001 = MW00002	NG	–

(註)請參閱以下章節為適用之指令、運算順序、表示方法等相關內容。

☞ 附錄 C EXPRESSION 指令格式

◆ 敘述句的層數 (巢套 Nesting)

FOR 敘述句、WHILE 敘述句、IF 敘述句當中可包含其他類型的敘述句，這就稱之為「巢套 (Nesting)」使用 FOR 敘述句、WHILE 敘述句、IF 敘述句時，使用的巢套 (Nesting) 最多 8 層 (8 Nest)。

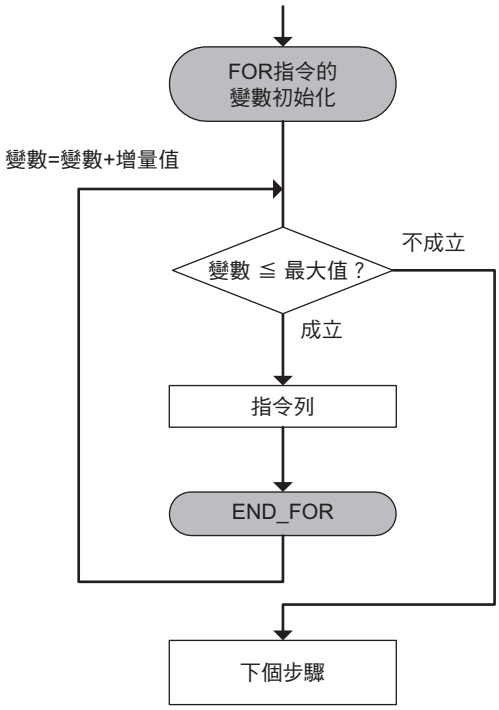
編寫接點後方的指令時，其處理方式與 IF 敘述句相同，皆包含在 Nest 的階層中。

FOR 陳述式 (FOR、END_FOR)

將會反覆執行 FOR 指令和 END_FOR 指令間的指令列。

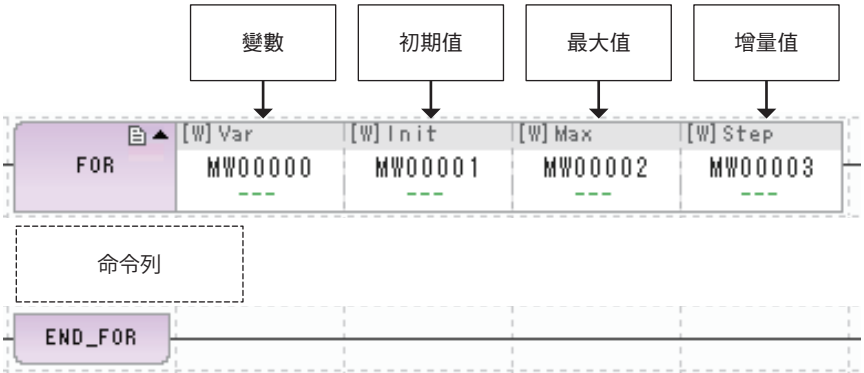
若暫存器被指定為變數時，就會從初始值開始，僅加上反覆執行時的增量值。

當變數的數值超過最大值時，FOR 指令的條件式將不成立，並進入下一個步驟。



格式

所採用之格式如下。



圖示　：FOR , END_FOR

按鍵輸入：FOR, FEND

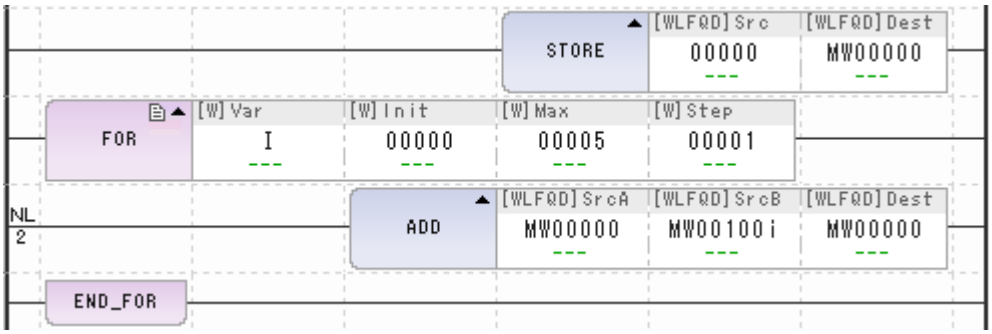
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
變數 (Var)	×	○*	×	×	×	×	×	○	×
初始值 (Init)	×	○	×	×	×	×	×	○	○
最大值 (Max)	×	○	×	×	×	×	×	○	○
增量值 (Step)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下係將 MW00100 ~ MW00105 相加，再將結果儲存為 MW00000 之程式範例。

將變數 1 初始化 (本範例為儲存 0)。接著，執行 ADD (+) 指令，直到變數超 (I) 過最大值 (5) 為止。當 I 為 6 時，條件式將不成立，END_FOR 指令就會進入下一個步驟。



補充事項

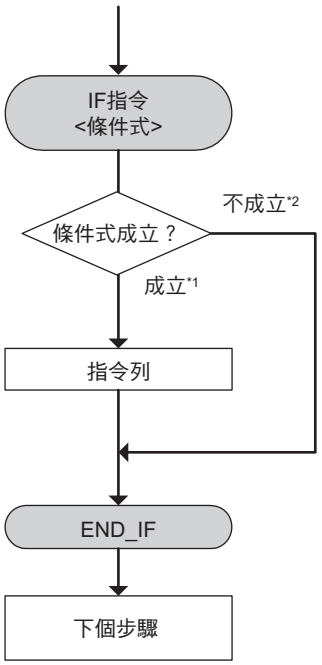
FOR 敘述句、WHILE 敘述句、IF 敘述句當中可包含其他類型的敘述句，這就稱之為「巢套 (Nesting)」使用 FOR 敘述句、WHILE 敘述句、IF 敘述句時，使用的巢套 (Nesting) 最多 8 層 (8 Nest)。

編寫接點後方的指令時，其處理方式與 IF 敘述句相同，皆包含在 Nest 的階層中。

IF 陳述式 (IF、END_IF)

當 IF 指令的條件式成立時，就會執行 IF 指令和 END_IF 指令間的指令列。

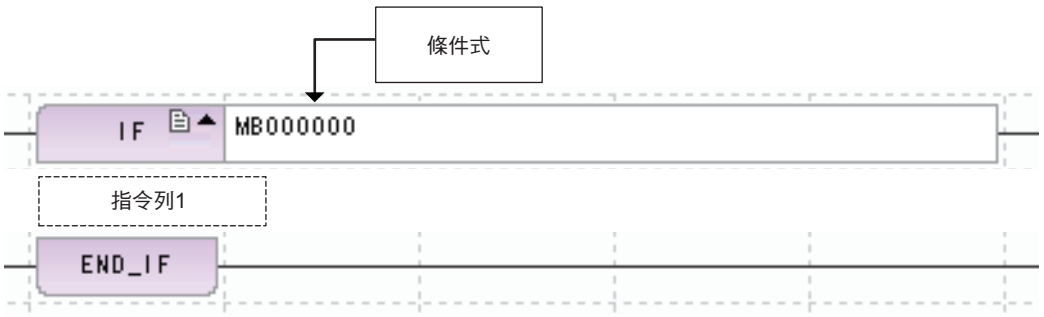
若條件式不成立，將不會執行指令列。



*1. 執行指令列後，進入下一個步驟。
*2. 不執行指令列，直接進入下一個步驟。

格式

所採用之格式如下。



圖示　：IF，END_IF

按鍵輸入：IF, IEND

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
條件式	○*	○*	○*	○*	○*	○*	×	○*	○*

* 利用 EXPRESSION 來編寫。
請參閱以下章節為適合用來編寫的格式。
☞ **附錄 C EXPRESSION 指令格式**

程式範例

當 IF 指令的條件式 (MB000001) ON 時，MW00010 的數值會被設定為 MW01000，然後再計算 MW00011。

補充事項

◆ 適用之條件式

在 WHILE 指令的條件式中，只可編寫如以下輸出 bool 型 (真或偽) 結果的 EXPRESSION 算式。因此，若您所編寫的算式包含了指定運算子，將無法判讀。

算式範例	編寫	備註
MB000001 == true	OK	true (真)：ON
MB000001 != false	OK	false (偽)：OFF
MW00002 < 100	OK	–
MF00002 < sin(60.0)	OK	–
MW00001 == 0x00FF OK	OK	使用 16 進位格式為「0x」。
MB000001 = true	NG	–
MW00001 = MW00002	NG	–

(註)請參閱以下章節為適用之指令、運算順序、表示方法等相關內容。

☞ 附錄 **C EXPRESSION** 指令格式

◆ 敘述句的層數 (巢套 Nesting)

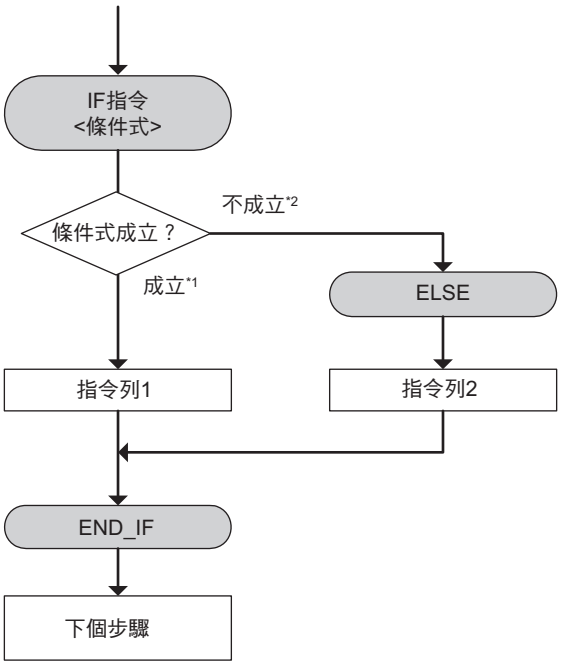
FOR 敘述句、WHILE 敘述句、IF 敘述句當中可包含其他類型的敘述句，這就稱之為「巢套 (Nesting)」使用 FOR 敘述句、WHILE 敘述句、IF 敘述句時，使用的巢套 (Nesting) 最多 8 層 (8 Nest)。

編寫接點後方的指令時，其處理方式與 IF 敘述句相同，皆包含在 Nest 的階層中。

IF-ELSE 陳述式 (IF、ELSE、END_IF)

若 IF 指令的條件式成立，只會執行指令列 1，並不會執行指令列 2。

若條件式不成立，只會執行指令列 2，並不會執行指令列 1。



*1. 執行指令列 1 後，進入下一個步驟。
*2. 執行指令列 2 後，進入下一個步驟。

格式

所採用之格式如下。

圖示　：IF, ELSE, END_IF

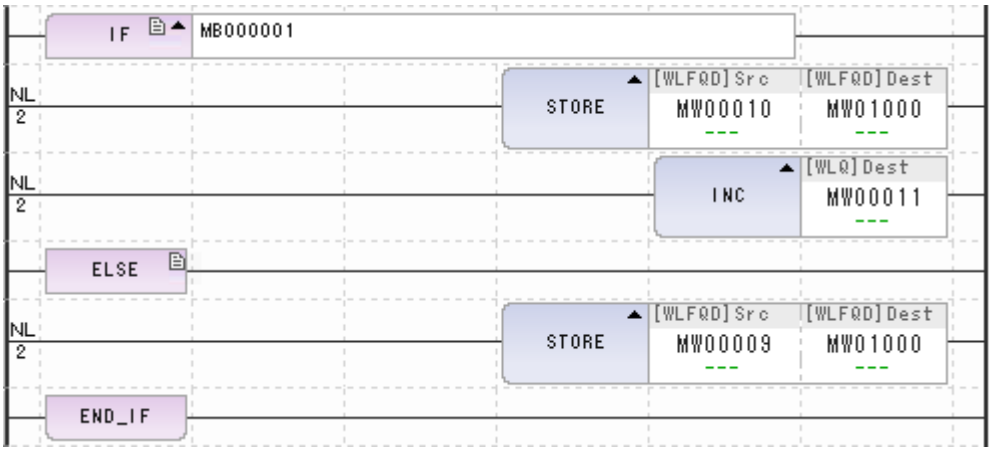
按鍵輸入：IF, ELSE, IEND

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
條件式	○*	○*	○*	○*	○*	○*	×	○*	○*

* 利用 EXPRESSION 來編寫。
請參閱以下章節為適合用來編寫的格式。
☞ **附錄 C EXPRESSION 指令格式**

程式範例

當 IF 指令的條件式 (MB000001) ON 時，MW00010 的數值會被設定為 MW01000，然後再計算 MW00011。當 IF 指令的條件式 (MB000001) OFF 後，MW00009 的數值將被設定為 MW01000。



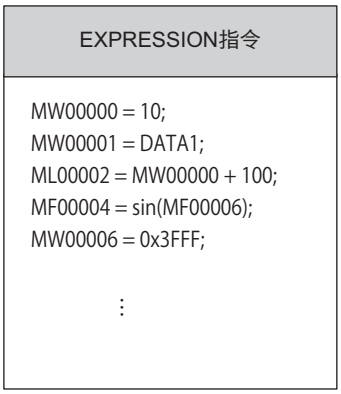
補充事項

適用之條件式、敘述句層數與 IF 敘述句相同。

算式編寫 (EXPRESSION)

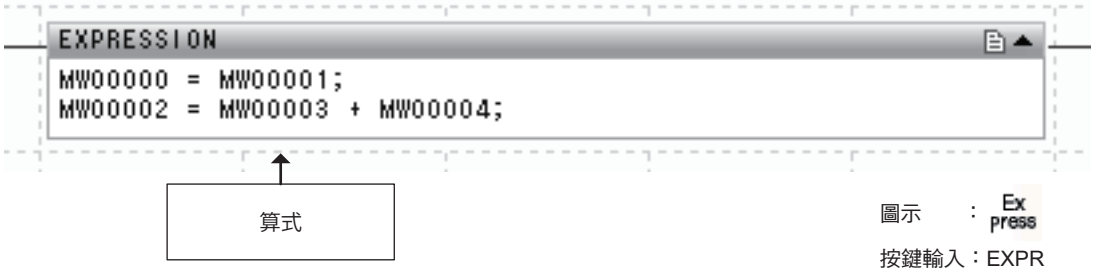
算式可用來編寫以下內容。

- 和 C 語言一樣，用來編寫非暫存器之變數名稱或結構體
- sin 函數或 cos 函數等基本函數
- 算數運算子、邏輯運算子、比較運算子、指定運算子
- 陣列



格式

所採用之格式如下。

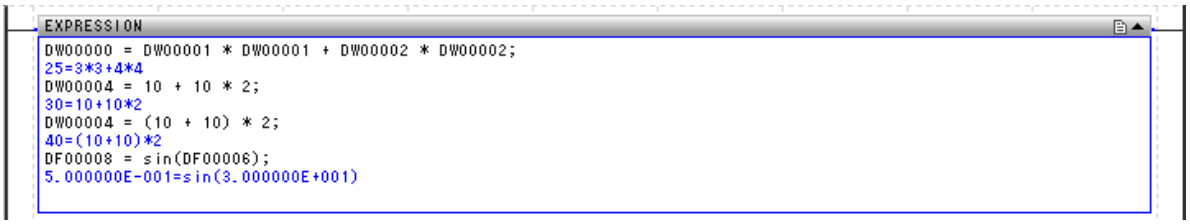


輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
算式	○*	○*	○*	○*	○*	○*	×	○*	○*

* 利用 EXPRESSION 來編寫。
請參閱以下章節為適合用來編寫的格式。
☞ 附錄 C EXPRESSION 指令格式

程式範例

以下為利用 EXPRESSION 指令來編寫多組運算式之程式範例。



補充事項

您也可以依照下表所示，來編寫用來輸出 bool 型 (真或偽) 結果的 EXPRESSION 算式。

算式範例	編寫	備註
MB000000 = true;	OK	true (真)：ON
MW00000 = MW00001+10	OK	–
MW00000 = 0x00FF;	OK	使用 16 進位格式為 「0x」。
MB000000 == true;	NG	–
MW00001 > MW00000;	NG	–

(註)請參閱以下章節為適用之指令、運算順序、表示方法等相關內容。

☞ 附錄 C EXPRESSION 指令格式

4.6 基本函數指令

平方根 (SQRT)

可用來運算整數型或實數型輸入資料的平方根，並將結果儲存為輸出資料。

不適用於長整數型資料。

補充　若輸出資料小於 0，就會利用輸入資料的絕對值，來輸出運算結果。

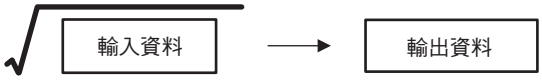
◆ 整數型 SQRT：輸入資料及輸出資料皆為整數型時

補充　整數型 SQRT 指令不同於數學所使用的平方根，係利用以下算式來進行運算。

sign (輸入資料) × $\sqrt{|\text{輸入資料}|\times 32768}$

也就是將數學所計算的平方根結果乘以$\sqrt{32768}$後所得到之數值。此外，若輸入資料為負數，將會運算絕對值的平方根，然後再將負數當作運算結果。運算的最大誤差值為 ± 2。

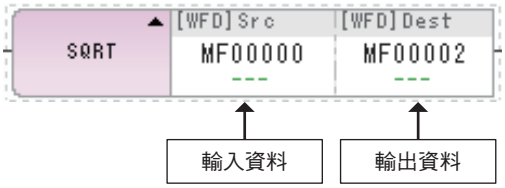
◆ 實數型 SQRT：上述以外的情況



補充　將先前的運算結果 (實數型資料) 當作輸入資料，然後再將其平方根儲存為實數型資料。

格式

所採用之格式如下。



圖示　：√

按鍵輸入： SQRT

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	○	×	×	○	○	×	○	○
輸出資料 (Dest)	×	○*	×	×	○*	○*	×	○	×

* C、# 暫存器除外

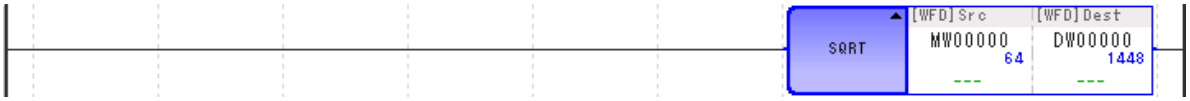
程式範例

以下為整數型及實數型輸入資料使用 SQRT 指令之程式範例。

· **整數型 SQRT 時**

將整數型輸入資料 (MW00000)：64 的平方根乘以 $128\sqrt{2}$，然後再將結果儲存為輸出資料 (DW00000)。

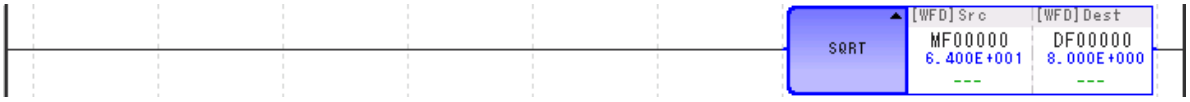
$\sqrt{64} \times 128\sqrt{2} \rightarrow$ DW00000 = 1448



· **實數型 SQRT 時**

將實數型輸入資料 (MF00000)：64.0 的平方根加以運算，然後再將結果儲存為輸出資料 (DF00000)。

$\sqrt{64.0} \rightarrow$ DF00000 = 8.0



正弦 (SIN)

可用來運算整數型或實數型輸入資料的正弦，並將結果儲存為輸出資料。

不適用於長整數型資料。

◆ 若輸入資料及輸出資料為整數型時

SIN （ 輸入資料 ） × 10000 ⟶ 輸出資料

(註)1. 輸入資料的單位為「度」(1 = 0.01)。
2. 將輸出 10000 倍數值的輸出資料作為運算結果。

◆ 若輸入資料及輸出資料為實數型時

SIN （ 輸入資料 ） ⟶ 輸出資料

(註)輸入資料的單位為「度」。

格式

所採用之格式如下。

圖示 ：sin

按鍵輸入：SIN

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	○	×	×	○	○	×	○	○
輸出資料 (Dest)	×	○*	×	×	○*	○*	×	○	×

* C、# 暫存器除外

◆ 整數型資料的情形下

輸入資料的單位為「度」(1 = 0.01)。

因此 SIN 函數的運算範圍為 -327.78 度 ~ 327.67 度。

係將 Sin 函數的輸出值乘以 10000，因此資料輸出範圍將變為 -10000 ~ 10000。

◆ 實數型資料的情形下

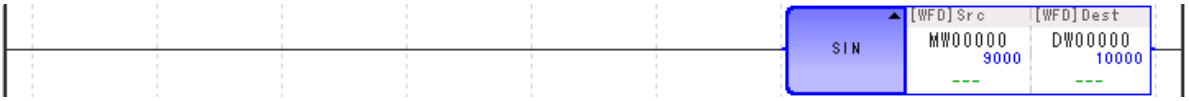
輸入資料的單位為「度」。

程式範例

以下為整數型及實數型輸入資料使用 Sin 指令之程式範例。

- **整數型 SIN 資料的情形下**
  將整數型輸入資料 (MW00000)：9000 的正弦加以運算，然後再將結果儲存為輸出資料 (DW00000)。

  SIN(90.00 度) × 10000 → DW00000 = 10000



- **實數型 SIN 資料的情形下**
  將實數型輸入資料 (MF00000)：90.0 的正弦加以運算，然後再將結果儲存為輸出資料 (DF00000)。

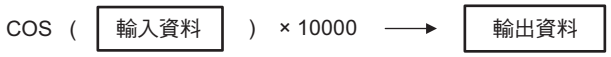
  Sin(90.0 度) → DF00000 = 1.0

餘弦 (COS)

可用來運算整數型或實數型輸入資料的餘弦，並將結果儲存為輸出資料。

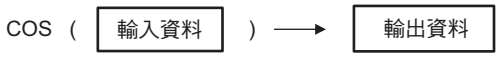
不適用於長整數型資料。

◆ 若輸入資料及輸出資料為整數型時



(註)1. 輸入資料的單位為「度」(1 = 0.01)。
2. 會將 10000 倍的數值儲存為輸出資料，以做為運算結果。
3. 輸入資料不得超出 -327.68 ~ 32.767 度的範圍。否則，將無法取得正確的結果。

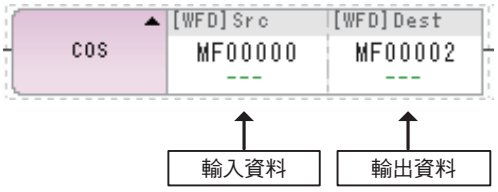
◆ 若輸入資料及輸出資料為實數型時



(註)輸入資料的單位為「度」。

格式

所採用之格式如下。



圖示 ： COS

按鍵輸入： COS

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	○	×	×	○	○	×	○	○
輸出資料 (Dest)	×	○*	×	×	○*	○*	×	○	×

* C、# 暫存器除外

◆ 整數型資料的情形下

輸入資料的單位為「度」(1 = 0.01)。

因此 COS 函數的運算範圍為 -327.78 度 ~ 327.67 度。

係將 COS 函數的輸出乘以 10000，因此資料輸出範圍將變為 -10000 ~ 10000。

◆ 實數型資料的情形下

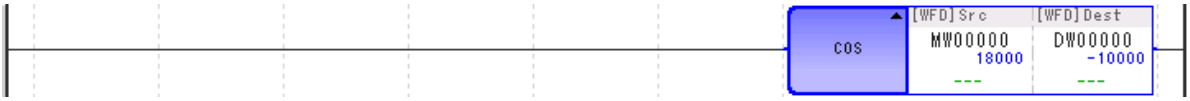
輸入資料的單位為「度」。

程式範例

以下為整數型及實數型輸入資料使用 COS 指令之程式範例。

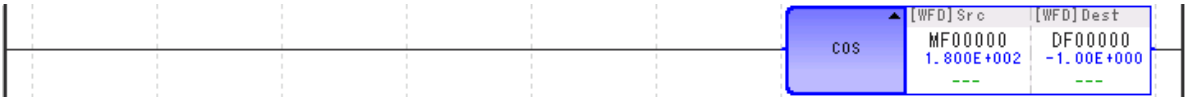
· **整數型 COS 資料的情形下**
將整數型輸入資料 (MW00000)：18000 的餘弦加以運算，然後再將結果儲存為輸出資料 (DW00000)。

COS(180.00 度) × 10000 → DW00000 = −10000



· **實數型 COS 資料的情形下**
將實數型輸入資料 (MF00000)：180.0 的餘弦加以運算，然後再將結果儲存為輸出資料 (DF00000)。

COS(180.0 度) → DF00000 = −1.0



正切 (TAN)

可用來運算實數型輸入資料的正切，並將結果儲存為輸出資料。

TAN （ 輸入資料 ） ⟶ 輸出資料

(註)輸入資料的單位為「度」。

格式

所採用之格式如下。

圖示 ：tan

按鍵輸入： TAN

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	×	×	×	○	○	×	×	○
輸出資料 (Dest)	×	×	×	×	○*	○*	×	×	×

* C、# 暫存器除外

程式範例

以下為運算輸入資料 (MW00000)：45 的正切，並將結果儲存為輸出資料 (DF00000) 之程式範例。

TAN(45.0 度) → DF00000 = 1.0

反正弦 (ASIN)

可用來運算實數型輸入資料的反正弦，並將結果儲存為輸出資料。

SIN （ 輸入資料 ）$^{-1}$ ⟶ 輸出資料

(註)輸出資料的單位為「度」。

格式

所採用之格式如下。

圖示 ： sin^{-1}

按鍵輸入：ASIN

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	×	×	×	○	○	×	×	○
輸出資料 (Dest)	×	×	×	×	○*	○*	×	×	×

* C、# 暫存器除外

設定輸入資料時不得超過 -1.0 ~ 1.0 的範圍。超出範圍時，輸出為 0。

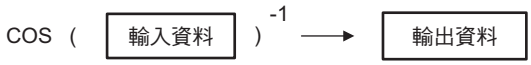
程式範例

以下為運算輸入資料 (MF00000)：1.0 的反正弦，並將結果儲存為輸出資料 (DF00000) 之程式範例。

SIN(1.0) $^{-1}$ → DF00000 = 90.0(度)

反餘弦 (ACOS)

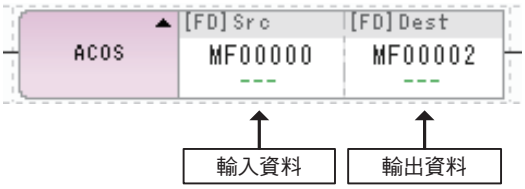
運算實數型輸入資料的反餘弦，並將結果儲存為輸出資料。



(註)輸出資料的單位為「度」。

格式

所採用之格式如下。



圖示 ： -1 COS

按鍵輸入：ACOS

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	×	×	×	○	○	×	×	○
輸出資料 (Dest)	×	×	×	×	○*	○*	×	×	×

* C、# 暫存器除外

設定輸入資料時不得超過 -1.0 ~ 1.0 的範圍。超出範圍時，輸出為 0。

程式範例

以下為運算輸入資料 (MF00000)：0.5 的反正弦，並將結果儲存為輸出資料 (DF00000) 之程式範例。

COS(0.5) $^{-1}$ → DF00000 = 60.0(度)

反正切 (ATAN)

可用來運算實數型輸入資料的反正切，並將結果儲存為輸出資料。

TAN (輸入資料) $^{-1}$ ⟶ 輸出資料

(註)輸出資料的單位為「度」。

格式

所採用之格式如下。

圖示 ： tan^{-1}

按鍵輸入：ATAN

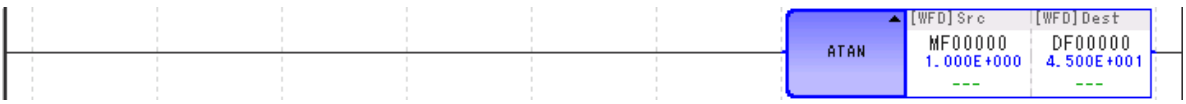
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	O	×	×	O	O	×	×	O
輸出資料 (Dest)	×	O	×	×	O*	O*	×	×	×

* C、# 暫存器除外

程式範例

以下為運算輸入資料 (MF00000)：1.0 的反正切，並將結果儲存為輸出資料 (DF00000) 之程式範例。

TAN(1.0) $^{-1}$ → DF00000 = 45.0(度)



指數 (EXP)

運算時只會將自然對數的底數 (e) 乘以實數型輸入資料，並將數值結果儲存為輸出資料。

(註)e 為自然對數的底數

格式

所採用之格式如下。

圖示 ： X^y

按鍵輸入：EXP

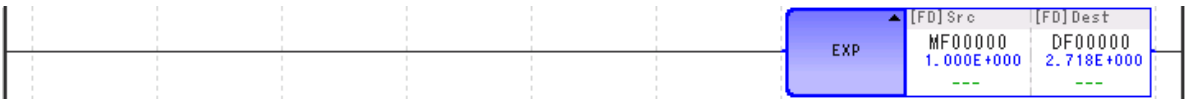
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	×	×	×	○	○	×	×	○
輸出資料 (Dest)	×	×	×	×	○*	○*	×	×	×

* C、# 暫存器除外

程式範例

以下係假設自然對數的底數 (e) 的輸入資料 (MF00000) 為 1.0 並進行運算時，再將結果儲存為輸出資料 (DF00000) 之程式範例。

$e^{1.0} \rightarrow$ DF00000 = 2.718282



補充　若運算結果溢位，將輸出最大值 3.402E+38，此時並不會發生運算錯誤。

自然對數 (LN)

假設實數型輸入資料為 X，計算其自然對數 ($\log_e X$)，並儲存為輸出資料。

格式

所採用之格式如下。

圖示　：ln

按鍵輸入：LN

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	×	×	×	○	○	×	×	○
輸出資料 (Dest)	×	×	×	×	○*	○*	×	×	×

* C、# 暫存器除外

若輸入資料 < 0 時，將使用輸入資料的絕對值，並輸出運算結果。

若輸入資料 = 0，將輸出為 - ∞。

程式範例

以下係假設 (MF00000) 為 2.718282 (≒ e) 時，並計算其自然對數，再將運算結果儲存為輸出資料 (DF00000) 之程式範例。

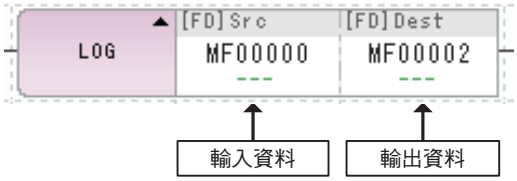
$Log_e 2.718282 \fallingdotseq log_e e \rightarrow DF00000 = 1.0$

常用對數 (LOG)

假設實數型輸入資料為 X，計算其常用對數 ($\log_{10}$ X)，並儲存為輸出資料。

格式

所採用之格式如下。



圖示 ： log

按鍵輸入： LOG

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入資料 (Src)	×	×	×	×	○	○	×	×	○
輸出資料 (Dest)	×	×	×	×	○*	○*	×	×	×

* C、# 暫存器除外

若輸入資料 < 0 時，將使用輸入資料的絕對值，並輸出運算結果。

若輸入資料 = 0，將輸出為 -∞。

程式範例

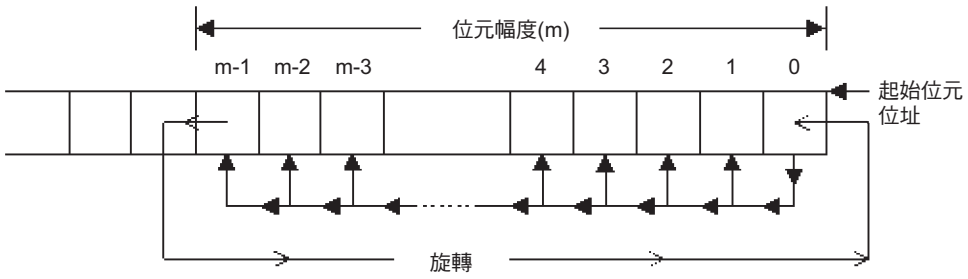
以下係假設輸入資料 (MF00000) 為 10.0 時，並計算其常用對數，再將運算結果儲存為輸出資料 (DF00000) 之程式範例。

$\log_{10}$ 10.0 → DF00000 = 1.0

4.7 資料移動指令

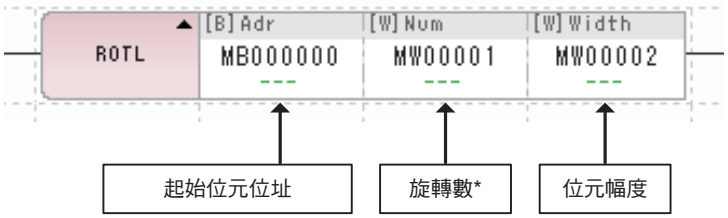
位元向左旋轉 (ROTL)

可僅用所指定的位元值，將已指定起始位元位址及位元幅度的資料向左旋轉。



格式

所採用之格式如下。



圖示 ： ROT L

按鍵輸入：ROTL

* 所要旋轉的位元值

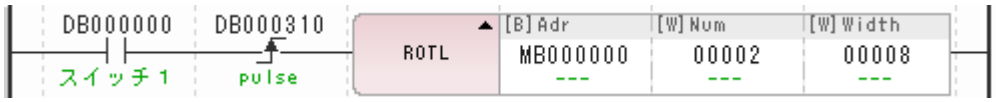
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
起始位元位址 (Adr)	O*	×	×	×	×	×	×	×	×
旋轉數 (Num)	×	O	×	×	×	×	×	O	O
位元幅度 (Width)	×	O	×	×	×	×	×	O	O

* C、# 暫存器除外

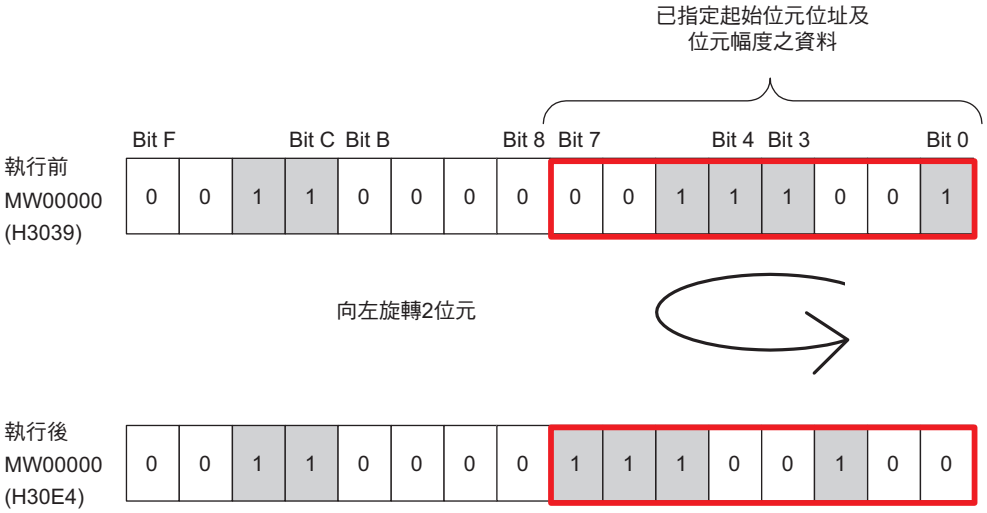
程式範例

以下係將起始位址 (MB000000) 和位元幅度 (8) 所指定的資料向左旋轉 2 位元之程式範例。

當開關 1 (DB000000) 開啟，便會執行 ROTL 指令。

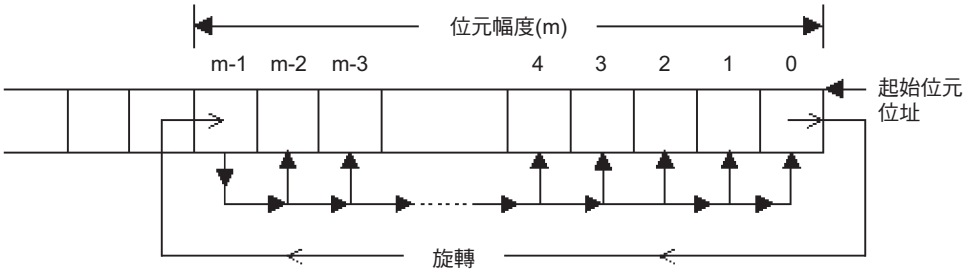


下圖為假設 MW00000 為 12345 (H3039) 時之動作。



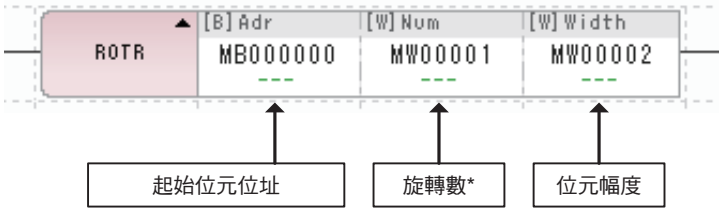
位元向右旋轉 (ROTR)

可僅用所指定的位元值，將已指定起始位元位址及位元幅度的資料向右旋轉。



格式

所採用之格式如下。



圖示　：

按鍵輸入：ROTR

* 所要旋轉的位元值

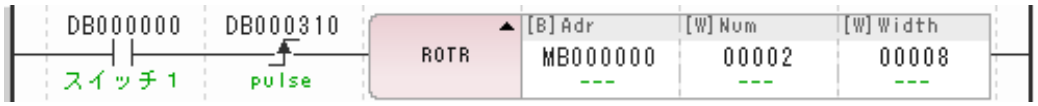
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
起始位元位址 (Adr)	○*	×	×	×	×	×	×	×	×
旋轉數 (Num)	×	○	×	×	×	×	×	○	○
位元幅度 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

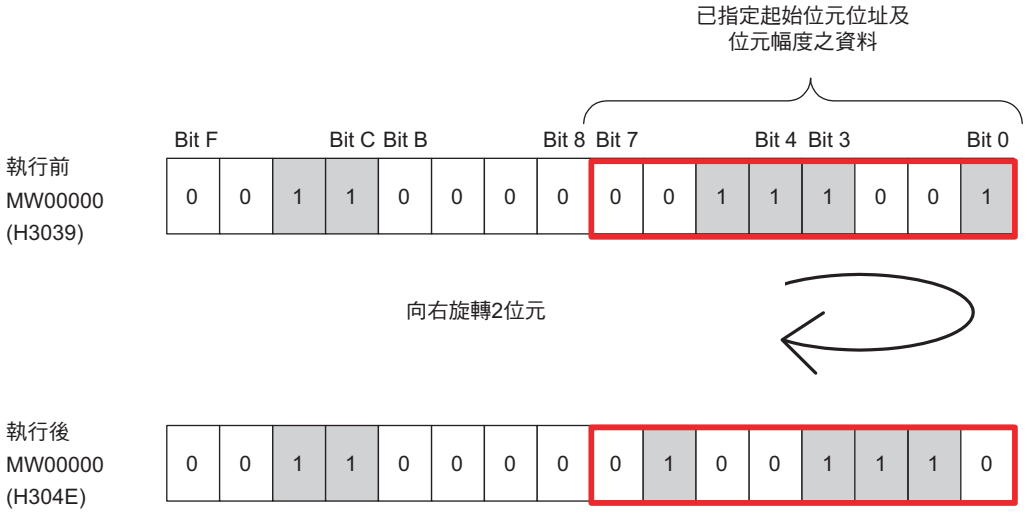
程式範例

以下係將起始位址 (MB000000) 和位元幅度 (8) 所指定的資料向右旋轉 2 位元之程式範例。

當開關 1 (DB000000) 開啟，便會執行 ROTR 指令。

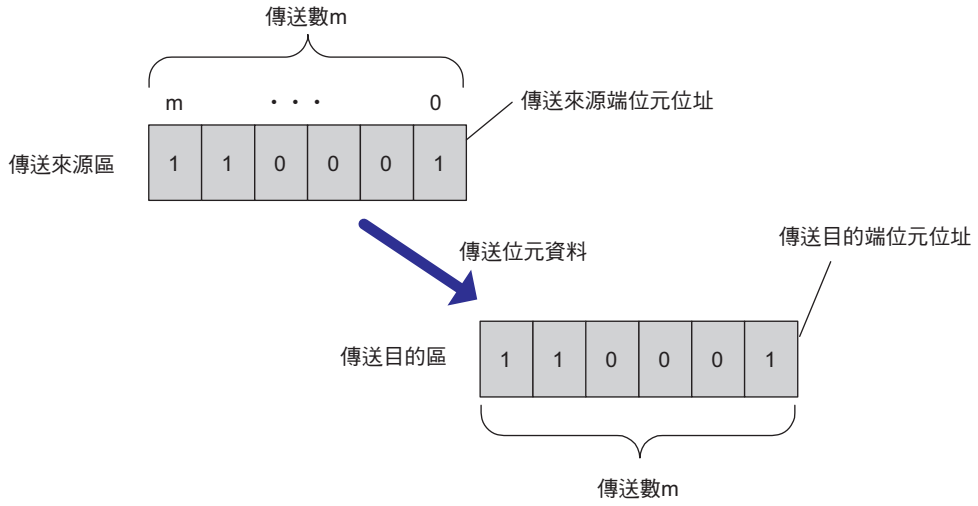


下圖為假設 MW00000 為 12345 (H3039) 時之動作。



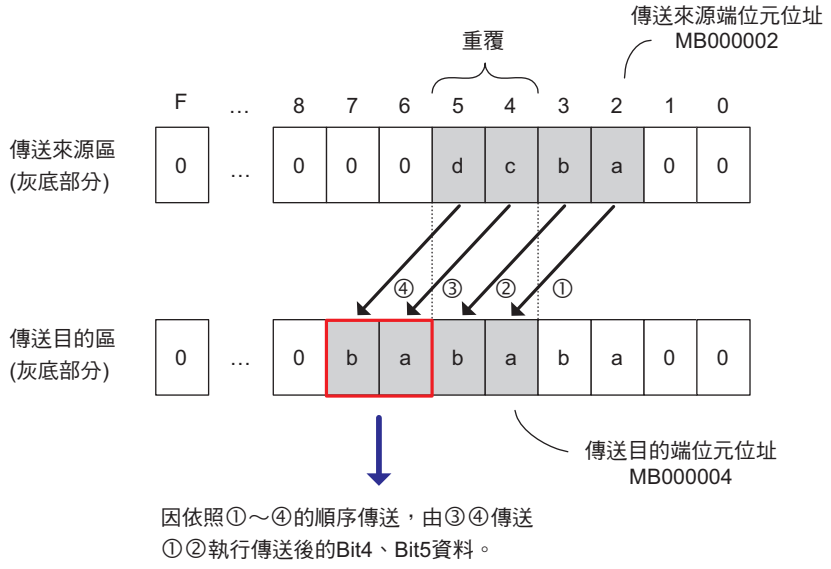
位元傳送 (MOVB)

可將傳送數的區域的位元資料從傳送來源端的位元位址傳送到目的端位元位址。



(註)根據繼電器編號由小而大，對每個位元分別進行傳送處理。
若傳送來源端和傳送目的端重覆時，有可能會在執行指令時，將不同於傳送來源區的資料傳送到傳送目的區。

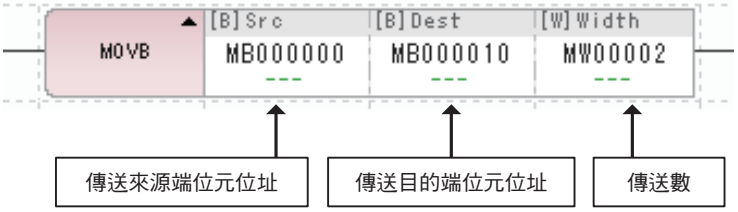
下圖所示為傳送來源區與傳送目的區重覆之範例。



因依照①～④的順序傳送，由③④傳送
①②執行傳送後的Bit4、Bit5資料。

格式

所採用之格式如下。



圖示　：
按鍵輸入：MOVB

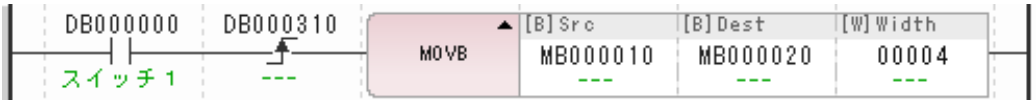
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送來源端位元位址 (Src)	○	×	×	×	×	×	×	×	×
傳送目的端位元位址 (Dest)	○*	×	×	×	×	×	×	×	×
傳送數	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下為傳送來源端位元位址 (MB000010) 的 4 位元資料被傳送到傳送目的端位元位址 (MB000020) 之程式範例。

當開關 1 (DB000000) 開啟，就會執行 MOVE 指令。

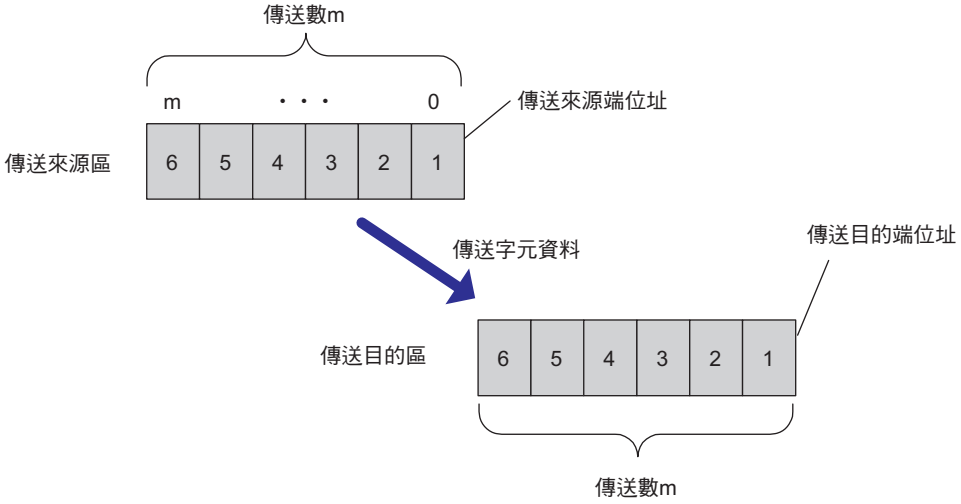


傳送來源區的資料將如下表所示被傳送到傳送目的區。

傳送來源區			傳送目的區		
暫存器	資料		暫存器	執行指令前的資料	執行指令後的資料
MB000010	0	⇒	MB000020	0	0
MB000011	1		MB000021	0	1
MB000012	1		MB000022	0	1
MB000013	1		MB000023	0	1

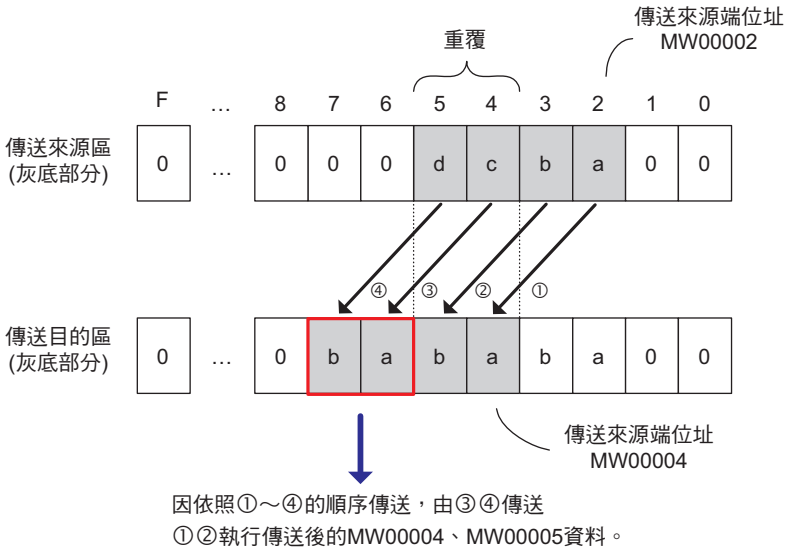
字元傳送 (MOVW)

可將傳送數的區域的字元資料從傳送來源端的位元位址傳送到目的端位元位址。



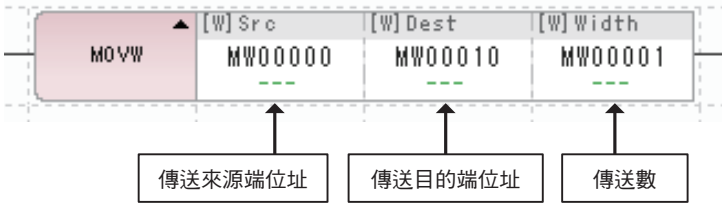
(註)根據暫存器編號由小而大，對每個字元分別進行傳送處理。
若傳送來源端和傳送目的端重覆時，可能會在執行指令時，將不同於傳送來源區的資料傳送到傳送目的區。

下圖所示為傳送來源區與傳送目的區重覆之範例。



格式

所採用之格式如下。



圖示　：

按鍵輸入：MOVW

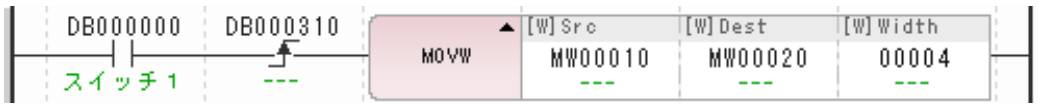
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送來源端位址 (Src)	×	○	×	×	×	×	×	○	×
傳送目的端位址 (Dest)	×	○*	×	×	×	×	×	○	×
傳送數 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下為傳送來源端位址 (MW00010) 的 4 個字元資料被傳送到傳送目的端位址 (MW00020) 之階梯圖程式範例。

當開關 1 (DB000000) 開啟，就會執行 MOVW 指令。



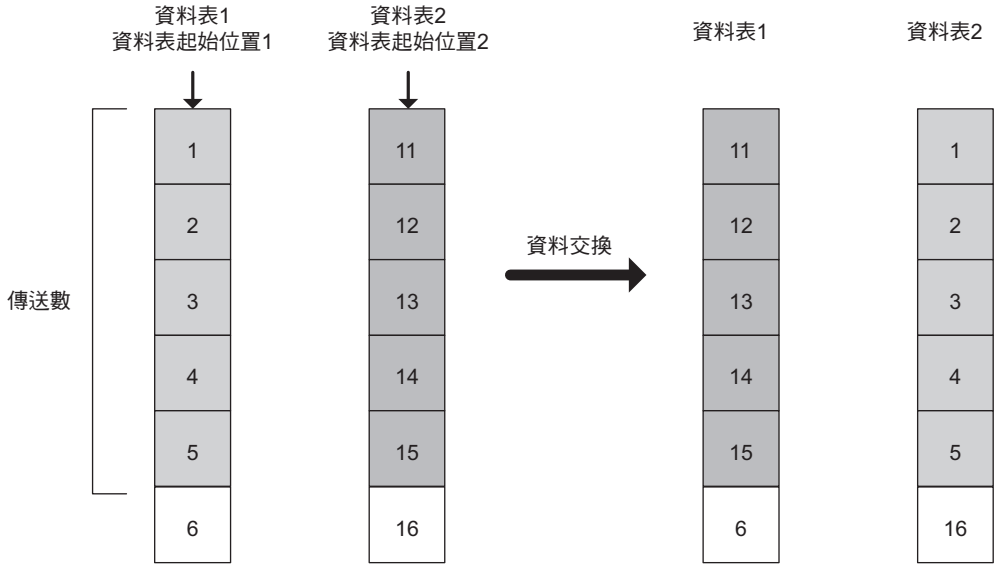
傳送來源區的資料將如下表所示被傳送到傳送目的區。

傳送來源區			傳送目的區		
暫存器	資料		暫存器	執行指令前的資料	執行指令後的資料
MW00010	10	⇒	MW00020	0	10
MW00011	20		MW00021	0	20
MW00012	30		MW00022	0	30
MW00013	40		MW00023	0	40

置換傳送 (XCHG)

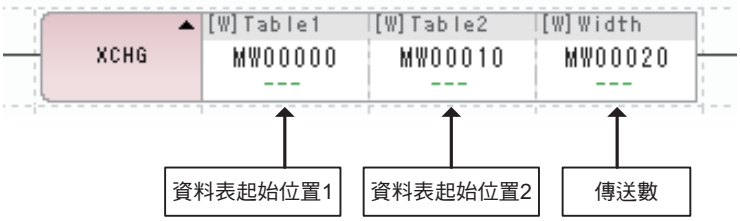
可僅互換資料表 1 和資料表 2 的資料的傳送數。

可互換用資料表起始位置 1、資料表起始位置 2、傳送數 (字元數) 指定之資料表 1 和資料表 2 的資料內容。



格式

所採用之格式如下。



圖示　：XCHG

按鍵輸入：XCHG

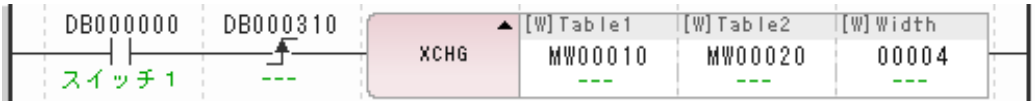
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
資料表起始位置 1 (Table1)	×	○*	×	×	×	×	×	×	×
資料表起始位置 2 (Table2)	×	○*	×	×	×	×	×	×	×
傳送數 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下為資料表 1 (MW00010) 和資料表 2 (MW00020) 僅互換 4 個字元資料之程式範例。

當開關 1 (DB000000) 開啟，就會執行 XCHG 指令。



資料表 1 和資料表 2 的資料將如下表所示進行互換。

資料表 1

暫存器	執行指令前的資料	執行指令後的資料
MW00010	10	123
MW00011	20	234
MW00012	30	345
MW00013	40	456

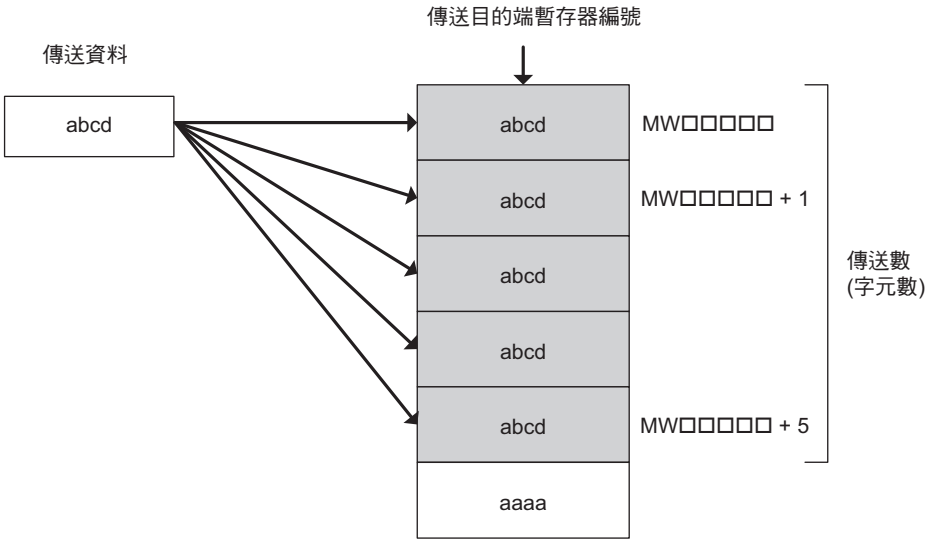
⇔

資料表 2

暫存器	執行指令前的資料	執行指令後的資料
MW00020	123	10
MW00021	234	20
MW00022	345	30
MW00023	456	40

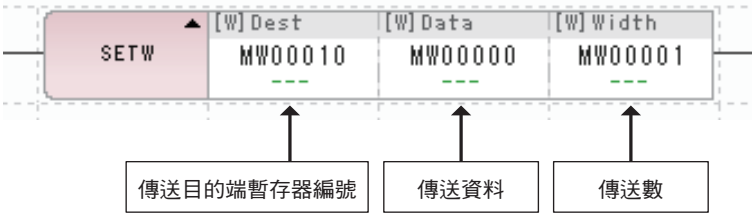
資料表初始化 (SETW)

由傳送目的端暫存器編號，將傳送資料所指定的資料儲存至傳送數的區域的所有暫存器。根據暫存器編號由小到大，依每個字元分別儲存。



格式

所採用之格式如下。



圖示　：

按鍵輸入：SETW

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送目的端暫存器編號 (Dest)	×	○*	×	×	×	×	×	×	×
傳送資料 (Data)	×	○	×	×	×	×	×	○	○
傳送數 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

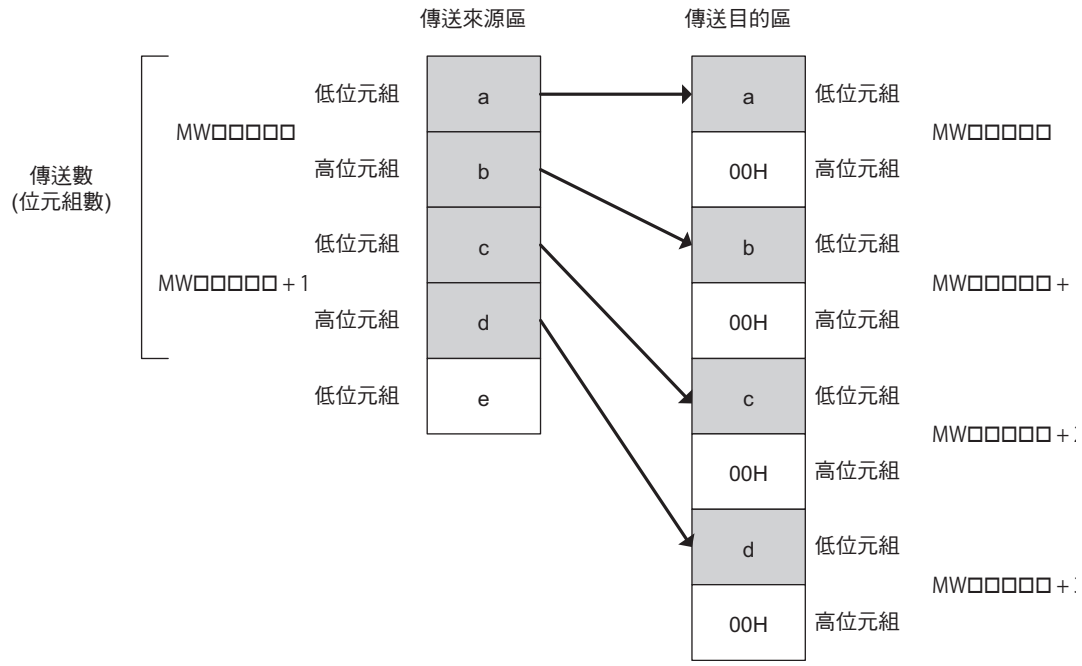
以下為開啟電源並進行第一次高速掃描時，從 MW00000 開始將 1000 字元區用傳送資料 (0) 初始化之程式範例。

開啟電源並進行第一次高速掃描後，所有的資料將如下表所示被初始化為 0。

暫存器	資料
MW00000	0
MW00001	0
:	:
MW00998	0
MW00999	0

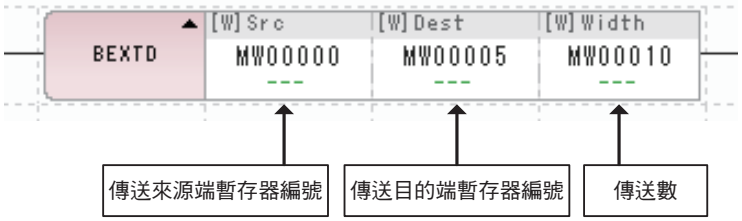
位元組 → 字元展開 (BEXTD)

將從傳送來源端暫存器編號開始至傳送數的區域的位元組資料，依各別位元組展開為 1 字元資料，並傳送至從傳送目的端暫存器編號到傳送數的區域。展開 1 字元時，高位元組將變為 0。



格式

所採用之格式如下。



圖示 ： B EXTD

按鍵輸入： BEXTD

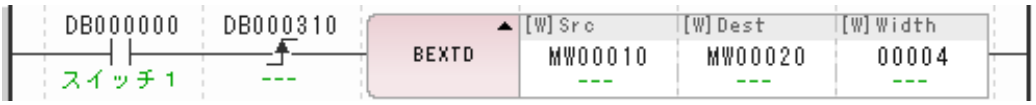
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送來源端暫存器編號 (Src)	×	○	×	×	×	×	×	×	×
傳送目的端暫存器編號 (Dest)	×	○*	×	×	×	×	×	×	×
傳送數 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下程式範例係以將從傳送來源端暫存器編號 (MW00010) 開始至 4 位元組的區域的資料，傳送到從傳送目的端暫存器編號 (MW00020) 至 4 位元組的區域。

當開關 1 (DB000000) 開啟，就會執行 BEXTD 指令。



傳送來源區的位元組資料將依下表所示被展開為字元資料，然後再傳送到傳送目的區。

傳送來源區		
暫存器		資料
MW00010	低位元組	10 H
	高位元組	20 H
MW00011	低位元組	30 H
	高位元組	40 H

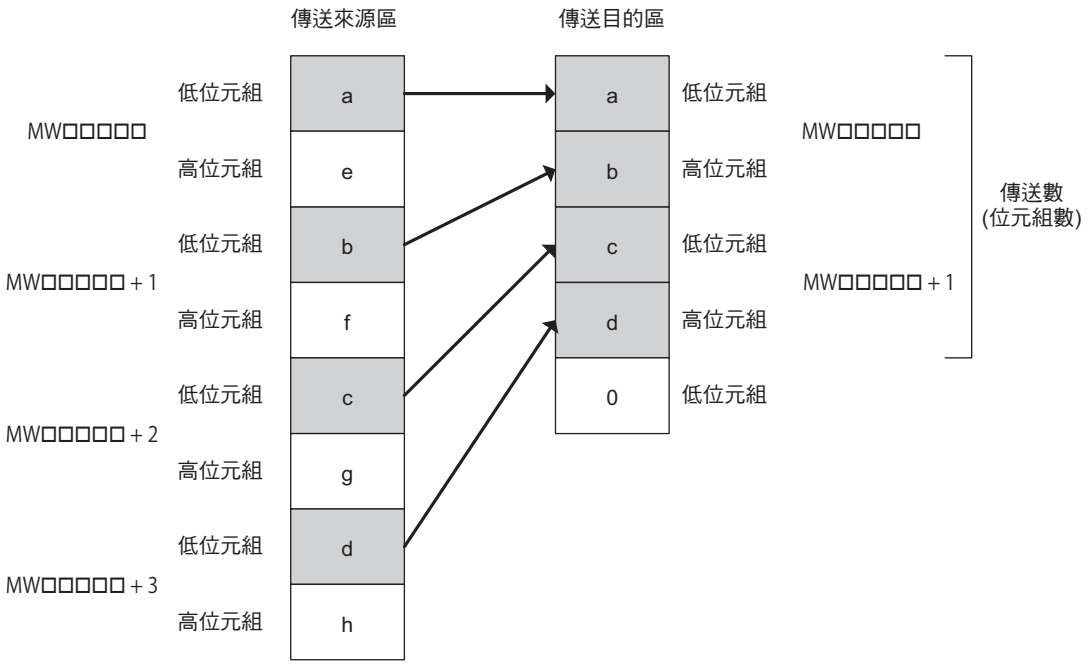
⇒

傳送目的區		
暫存器		資料
MW00020	低位元組	10 H
	高位元組	00 H
MW00021	低位元組	20 H
	高位元組	00 H
MW00022	低位元組	30 H
	高位元組	00 H
MW00023	低位元組	40 H
	高位元組	00 H

字元→位元組壓縮 (BPRESS)

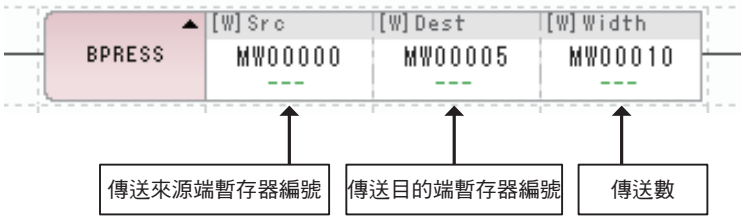
將從傳送來源端暫存器編號開始至儲存於傳送數區域的字元資料的低位元組，依位元組別分別儲存於從傳送目的端暫存器編號至傳送數的區域。所執行的動作和 BEXTED 指令相反。

高位元組將會被捨棄。



格式

所採用之格式如下。



圖示 ： BPRESS

按鍵輸入： BPRESS

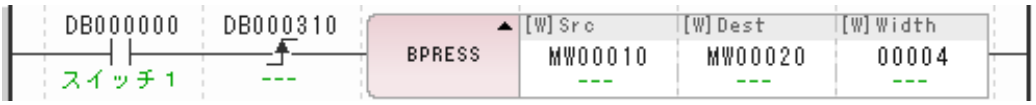
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送來源端暫存器編號 (Src)	×	○	×	×	×	×	×	×	×
傳送目的端暫存器編號 (Dest)	×	○*	×	×	×	×	×	×	×
傳送數 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下為將從傳送來源端暫存器編號 (MW00010) 開始至 4 字元區域的資料的低位元組，儲存到從傳送目的端暫存器編號 (MW00020) 至 4 位元組的區域之程式範例。

當開關 1 (DB000000) 開啟，就會執行 BPRESS 指令。



傳送來源區的字元資料將如下表所示，先被壓縮為位元組資料，然後再傳送到傳送目的區。

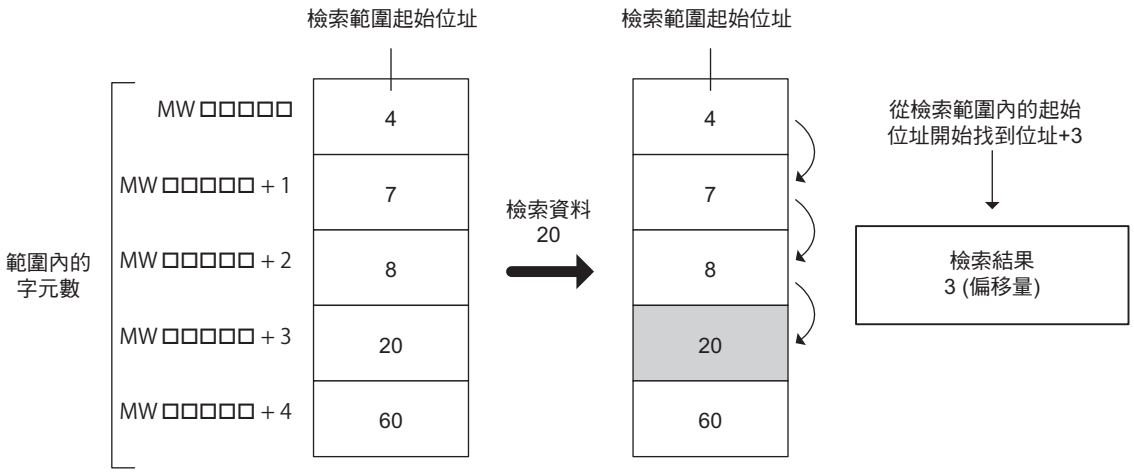
傳送來源區		
暫存器		資料
MW00010	低位元組	12 H
	高位元組	23 H
MW00011	低位元組	34 H
	高位元組	45 H
MW00012	低位元組	56 H
	高位元組	67 H
MW00013	低位元組	78 H
	高位元組	89 H

⇒

傳送目的區		
暫存器		資料
MW00020	低位元組	12 H
	高位元組	34 H
MW00021	低位元組	56 H
	高位元組	78 H

資料檢索 (BSRCH)

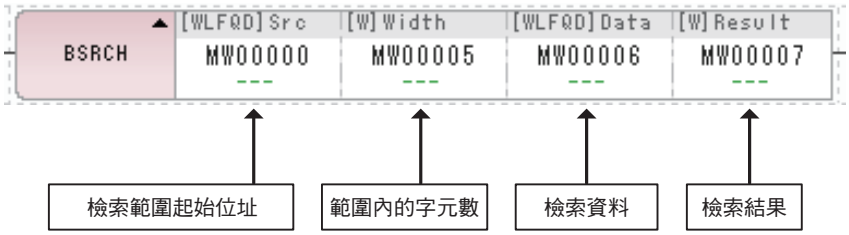
利用二元搜尋法搜尋檢索資料，從檢索範圍起始位址開始到範圍字元數的區域。若檢索開始暫存器資料與檢索資料一致，則偏移位元數就會被當作檢索結果輸出。檢索範圍必須先依升冪方式進行排序。



(註)1. 執行 BSRCH 指令前，請先依升冪方式為檢索區域排序。
2. 上圖為整數型資料之示意圖。長整數型及實數型亦同。
3. 若未檢索到任何資料，將儲存 -1 作為檢索結果。

格式

所採用之格式如下。



圖示 ： B SRCH

按鍵輸入： BSRCH

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
檢索範圍起始位址 (Src)	×	O	O	O	O	O	×	×	×
範圍內的字元數 (Width)	×	O	×	×	×	×	×	O	O
檢索資料 (Data)	×	O	O	O	O	O	×	O	O
檢索結果 (Result)	×	O*	×	×	×	×	×	×	×

* C、# 暫存器除外

專有名詞解說

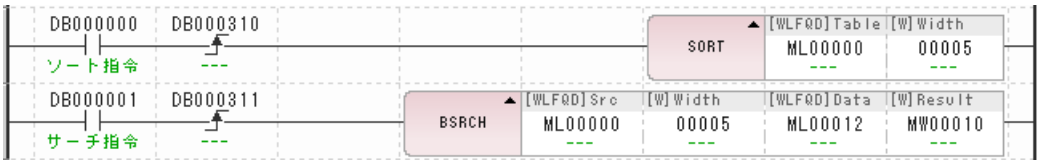
二元搜尋法

資料檢索演算法的一種，此種方法係針對完成排序的區域，高速進行資料檢索。

將位於中間區域的數值與檢索資料互相比較，檢索資料愈大，就從中間偏右區域開始檢索，反之，若檢索資料愈小，則由中間偏左區域開始檢索。採用此種步驟來檢索時，必須先以升冪方式為資料排序。

程式範例

當排序指令 (DB00000) 一開啟，就對 ML00000 ~ ML00008 進行排序。接著，當搜尋指令 (DB000001) 開啟後，就會從排序完成的資料區開始檢索所要檢索的資料 (ML00012)。



下表為 ML00000 ~ ML00008 (ML00012) 的資料，若您所要檢索的資料為 70，只要一執行第 1 行，就會依下表所示進行排序。執行第 2 行後，就會檢索到 70，此結果 4 就會被當作搜尋結果 (MW00010) 儲存起來。

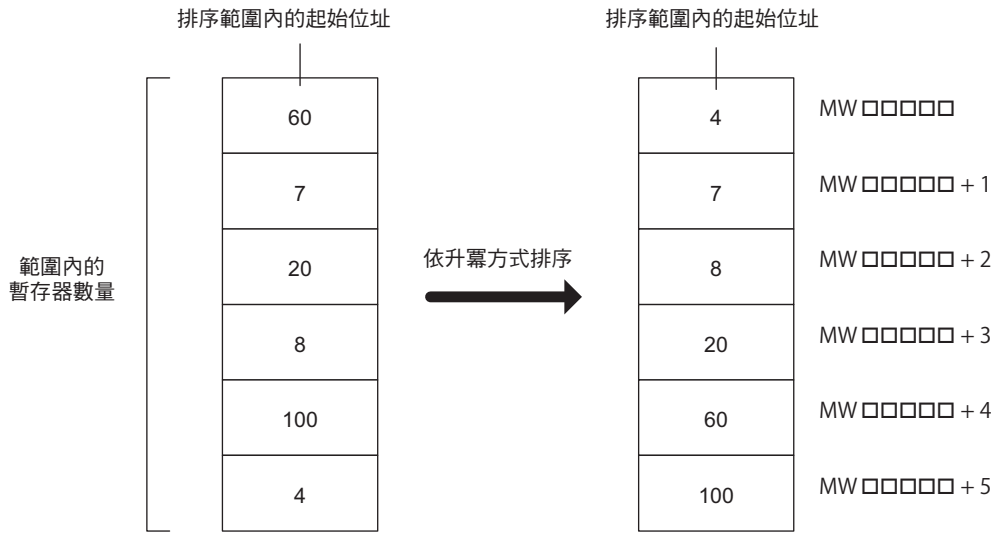
暫存器	執行第 1 行前的資料	執行第 1 行後的資料	第 2 行的執行結果
ML00000	100	15	ML00004 = 70，由此可得到 MW00010 = 4
ML00002	30	30	
ML00004	90	70	
ML00006	15	90	
ML00008	70	100	

排序 (SORT)

利用升冪方式從排序範圍起始位址開始到範圍內的暫存器數量的資料進行排序。

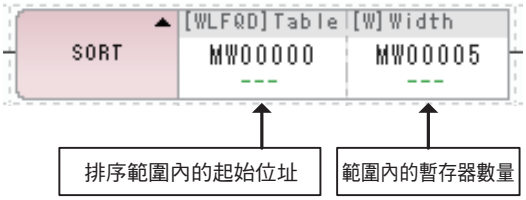
下圖將利用整數型資料來進行說明。長整數型、實數型資料的排序方法亦同。

可排序的資料數最多為 128 個。



格式

所採用之格式如下。



圖示　：SORT

按鍵輸入：SORT

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
排序範圍內的起始位址 (Table)	×	○*	○*	○*	○*	○*	×	×	×
範圍內的暫存器數量 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下為當排序指令 (DB000000) 一開啟，就以升冪方式為 ML00000 ~ ML00008 的資料進行排序之程式範例。

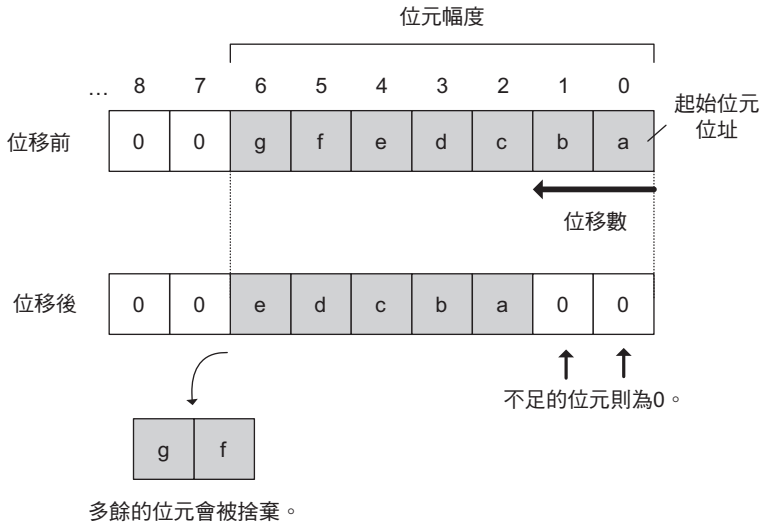
若 ML00000 ~ ML00008 為下表所示之資料，只要一執行 SORT 指令，就會依下表所示進行排序。

暫存器	執行指令前的資料	執行指令後的資料
ML00000	100	15
ML00002	30	30
ML00004	90	70
ML00006	15	90
ML00008	70	100

位元左移 (SHFTL)

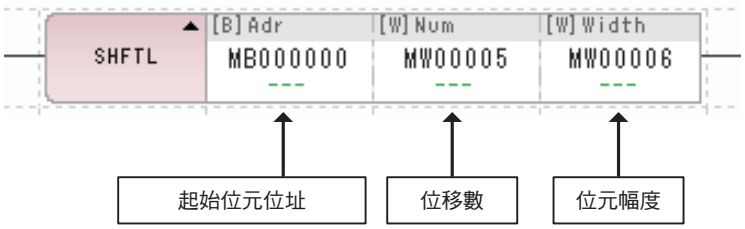
根據所指定的起始位元位址和位元幅度的位元列，僅依指定的位元數向左位移。

超出位元幅度的資料會被丟棄，不足的位元則會變為 0。



格式

所採用之格式如下。



圖示　：

按鍵輸入：SHFTL

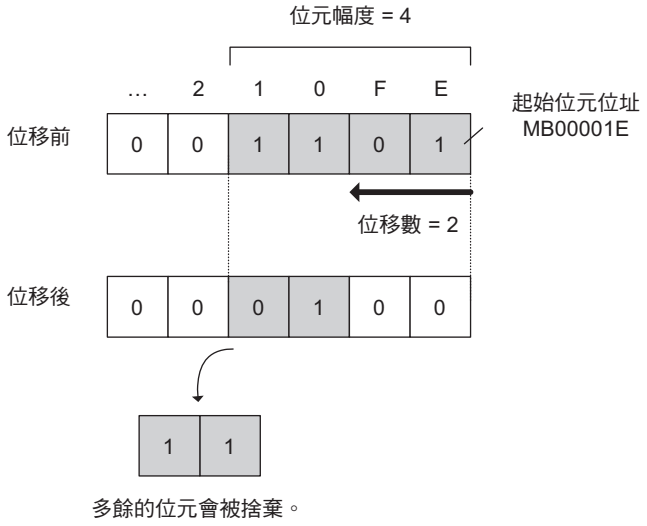
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
起始位元位址 (Adr)	O*	×	×	×	×	×	×	×	×
位移數 (Num)	×	O	×	×	×	×	×	O	O
位元幅度 (Width)	×	O	×	×	×	×	×	O	O

* C、# 暫存器除外

程式範例

以下為假設開關 1 (DB000000) 開啟時，然後再將起始位元位址 (MB00001E) 的 4 個位元向左位移 2 位元之程式範例。

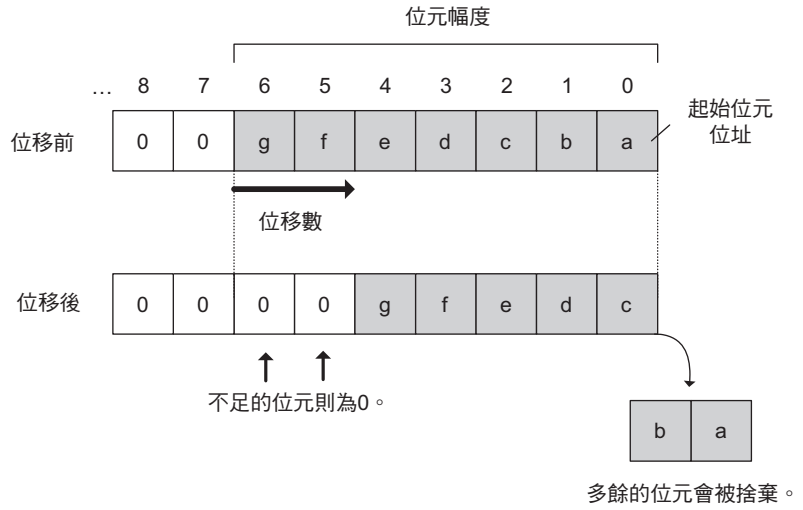
下圖為執行上述程式後所產生的結果。



位元右移 (SHFTR)

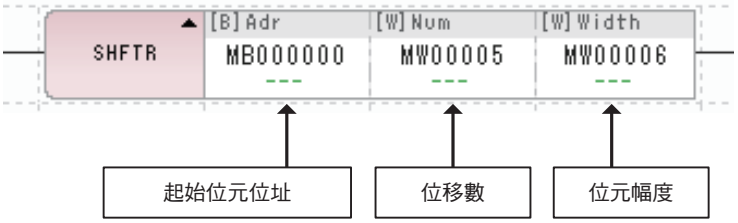
根據所指定的起始位元位址和位元幅度的位元列，僅依指定的位元數向右位移。

超出位元幅度的資料會被丟棄，不足的位元則會變為 0。



格式

所採用之格式如下。



圖示　：SHFT R

按鍵輸入：SHFTR

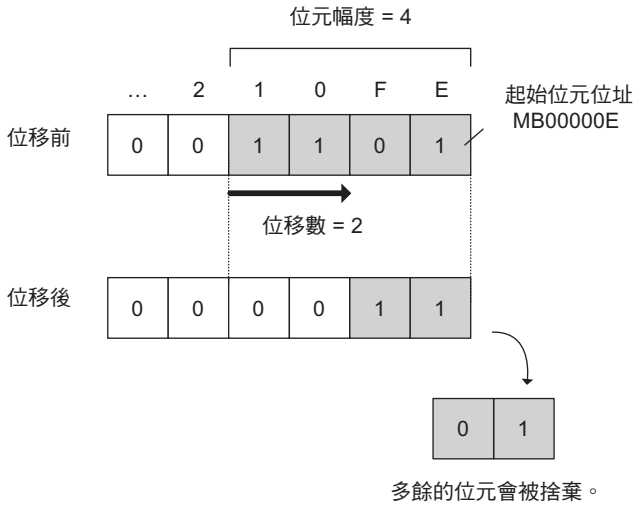
輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
起始位元位址 (Adr)	O*	×	×	×	×	×	×	×	×
位移數 (Num)	×	O	×	×	×	×	×	O	O
位元幅度 (Width)	×	O	×	×	×	×	×	O	O

* C、# 暫存器除外

程式範例

以下為假設開關 1 (DB000000) 開啟時，然後再將起始位元位址 (MB00001E) 的 4 個位元向右位移 2 位元之程式範例。

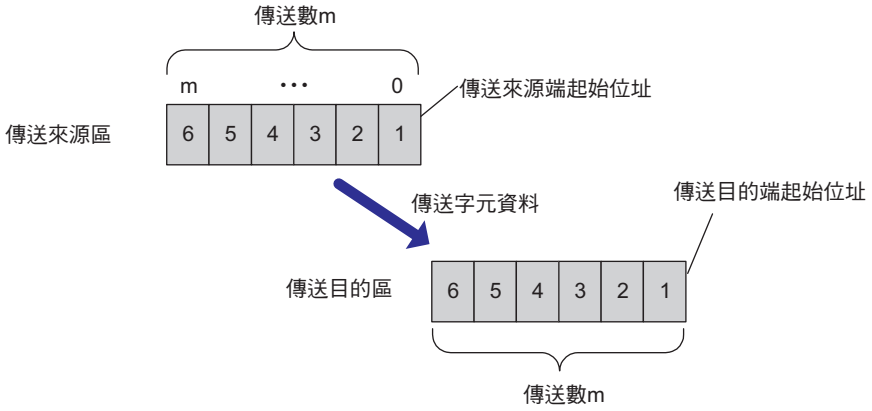
下圖為執行上述程式後所產生的結果。



複製字元 (COPYW)

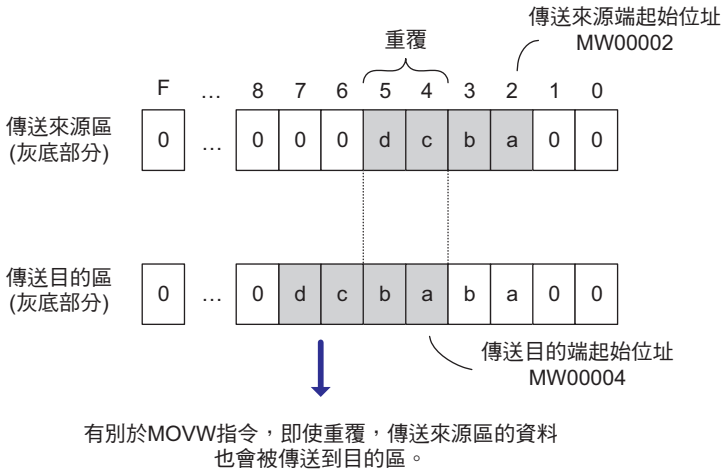
將傳送數的區域的字元資料，從傳送來源端位址傳送到傳送目的端位址。

會依資料的區塊別，複製傳送從傳送來源端到傳送目的端。因此，有別於字元傳送指令 (MOVW)，即使傳送來源端和傳送目的端重覆，所要傳送的資料區塊也會直接被複製到傳送目的端。



(註)當傳送來源端和傳送目的端資料區重疊時，所執行的動作將和 MOVW 指令不同。

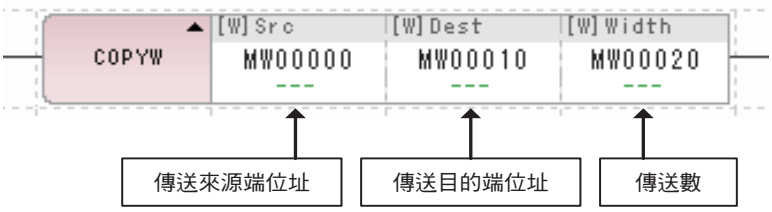
下圖所示為傳送來源區與傳送目的區重覆之範例。



有別於MOVW指令，即使重覆，傳送來源區的資料
也會被傳送到目的區。

格式

所採用之格式如下。



圖示　：

按鍵輸入： COPYW

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送來源端位址 (Src)	×	○	×	×	×	×	×	○	×
傳送目的端位址 (Dest)	×	○*	×	×	×	×	×	○	×
傳送數 (Width)	×	○	×	×	×	×	×	○	○

* C、# 暫存器除外

程式範例

以下程式範例係假設開關 1 (DB000000) 開啟時，將從傳送來源端位址 (MW00000) 開始至 5 字元的資料，複製到從傳送來源端位址 (MW00100) 開始至 5 字元資料區。

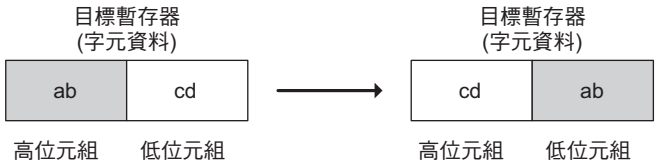
下圖為執行上述程式後所產生的結果。

暫存器	資料
MW00000	1
MW00001	2
MW00002	3
MW00003	4
MW00004	5

暫存器	執行指令前的資料	執行指令後的資料
MW00100	123	1
MW00101	234	2
MW00102	345	3
MW00103	456	4
MW00104	567	5

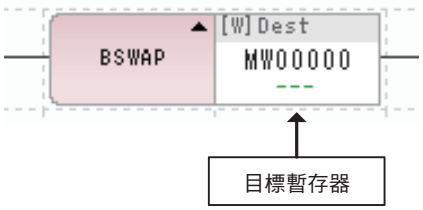
位元組調換 (BSWAP)

將目標暫存器之高位元組和低位元組互換。



格式

所採用之格式如下。



圖示 ： B SWAP

按鍵輸入：BSWAP

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
目標暫存器 (Dest)	×	○*	×	×	×	×	×	×	×

* C、# 暫存器除外

程式範例

以下係開關 1 (DB000000) 開啟後，將目標暫存器 (MW00000) 的高位元組和低位元組互換之程式範例。

若 MW00000 為 00FFH，只要執行 BSWAP 指令，MW00000 就會變為 FF00H。

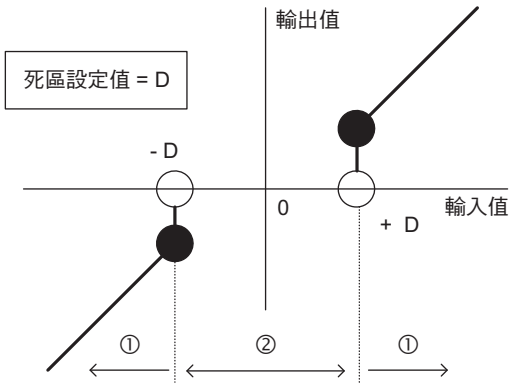
4.8 DDC 指令

死區 A (DZA)

可依照輸入值來運算對應所設定死區的輸出值。

下圖所示為假設｜輸入值｜ ≧ ｜ D ｜，由於超出死區範圍，會直接輸出您所輸入的數值。

若｜輸入值｜ ＜｜ D ｜，此時數值位於死區範圍內，因此將輸出 0。

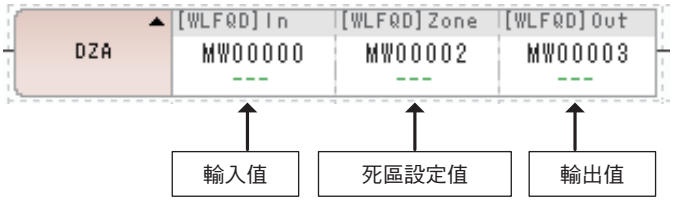


①當|輸入值| ≧ |D|時，
輸出值 = 輸入值

②當|輸入值| ＜ |D|時，
輸出值 = 0

格式

所採用之格式如下。



圖示 ：

按鍵輸入： DZA

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	○	○	○	○	○	×	○	○
死區設定值 (Zone)	×	○	○	○	○	○	×	○	○
輸出值 (Out)	×	○*	○*	○*	○*	○*	×	○	×

* C、# 暫存器除外

程式範例

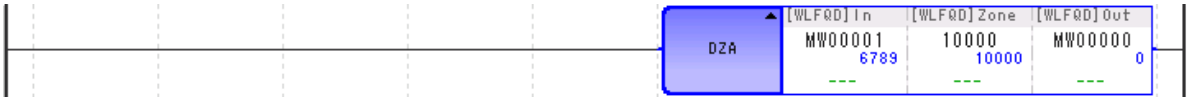
以下係假設死區設定值為 10000，再將運算結果儲存為輸出值 (MW00000) 之程式範例。

相對於輸入值 (MW00001 ~ MW00003)，輸出值如下圖所示運算運算。

· **超出死區範圍時**
由於｜ MW00001(12345) ｜≧｜ 10000 ｜，因此 MW00000 將變為 12345。

由於｜ MW00002(−12345) ｜≧｜ 10000 ｜，因此 MW00000 將變為 -12345。

· **未超出死區範圍時**
由於｜ MW00003(6789) ｜＜｜ 10000 ｜，因此 MW00000 將變為 0。

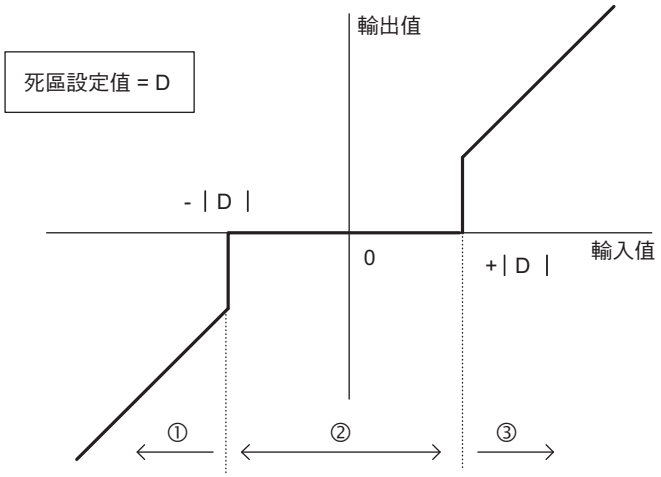


死區 B (DZB)

可依照輸入值來運算對應所設定死區的輸出值。

次如下圖所示，當｜輸入值｜ < ｜ D ｜時，數值位於死區範圍內，將輸出 0。

有別於 DZA 指令，當輸入值超出死區範圍時，依照輸入值的 ±，輸出 ± ｜ D ｜值至輸入值。



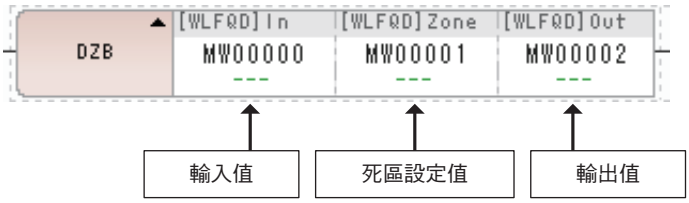
①當輸入值 < 0且｜輸入值｜≧｜D｜
輸出值 = 輸入值 －｜D｜

②當｜輸入值｜<｜D｜
輸出值 = 0

③當輸入值 > 0且｜輸入值｜≧｜D｜
輸出值 = 輸入值 ＋｜D｜

格式

所採用之格式如下。



圖示 ：

按鍵輸入： DZB

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	○	○	○	○	○	×	○	○
死區設定值 (Zone)	×	○	○	○	○	○	×	○	○
輸出值 (Out)	×	○*	○*	○*	○*	○*	×	○	×

* C、# 暫存器除外

程式範例

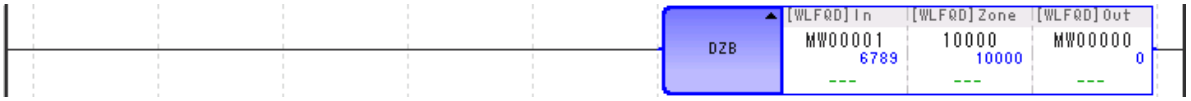
以下係假設死區設定值為 10000，再將運算結果儲存為輸出值 (MW00000) 之程式範例。

相對於輸入值 (MW00001)，輸出值如下圖所示運算。

- **超出死區範圍時**
  由於 MW00001(12345) ＞ 0 且｜ MW00001(12345) ｜≧｜ 10000 ｜，因此 MW00000 = 12345 – ｜ 10000 ｜ = 2345。

由於 MW00001(–12345) ＜ 0 且｜ MW00001(–12345) ｜≧｜ 10000 ｜，因此 MW00000 = –12345 + ｜ 10000 ｜ = –2345。

- **未超出死區範圍時**
  由於｜ MW00001(6789) ｜＜｜ 10000 ｜，因此 MW00000 將變為 0。

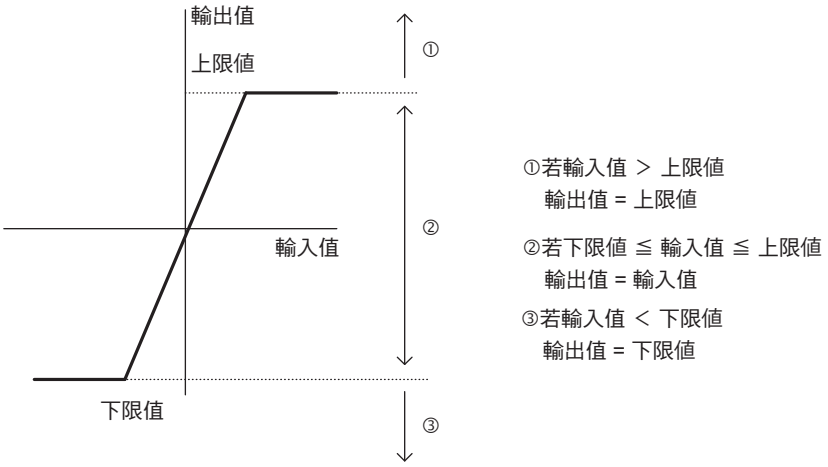


上下限值 (LIMIT)

可依照輸入值來控制輸出值避免數值超過所指定的上下限值。

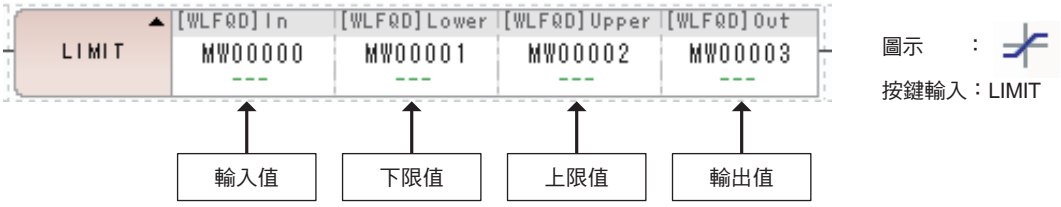
若輸入值在上下限限制範圍內時，將依下圖所示，直接將輸入值輸出。

若輸入值＞上限值，將輸出上限值，反之，若輸入值＜下限值，則輸出下限值。



格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	○	○	○	○	○	×	○	○
下限值 (Lower)	×	○	○	○	○	○	×	○	○
上限值 (Upper)	×	○	○	○	○	○	×	○	○
輸出值 (Out)	×	○*	○*	○*	○*	○*	×	○	×

* C、# 暫存器除外

補充　設定時，必須遵守下限值 ≦ 上限值的原則。

程式範例

以下係假設 LIMIT 指令的下限值為 -100、上限值為 10000，再將運算結果儲存為輸出值 (MW00000) 之程式範例。

相對於輸入值 (MW00001)，輸出值依照下圖所示運算。

· **若輸入值超出上下限值的範圍**
由於 MW00001 (12345) ＞上限值 (10000)，MW00000 將變為上限值 (10000)。

由於 MW00001 (-12345) ＜下限值 (-100)，MW00000 將變為下限值 (-100)。

· **若輸入值在上下限值的範圍內**
由於下限值 (-100) ＜ MW00001 (6789) ＜上限值 (10000)，MW00000 將變為 6789。

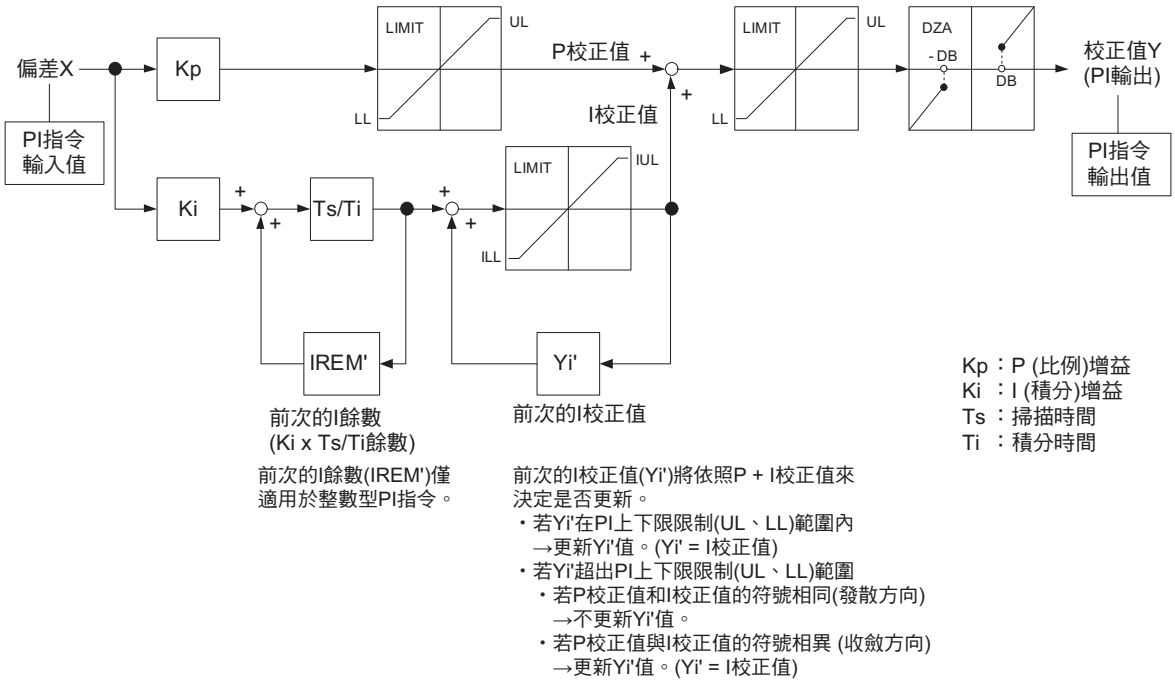
PI 控制 (PI)

只要輸入偏差 X，就會根據您所設定的參數資料表內容，進行 PI 運算及範圍運算，然後再輸出校正值 Y。

若參數資料表內的積分重置 ON，就會以 I 校正值為 0，並執行 PI 校正值運算。

可使用 PI 指令的輸入值為整數型及實數型。不適用於長整數型。

此外，整數型和實數型的參數資料表架構各不相同。

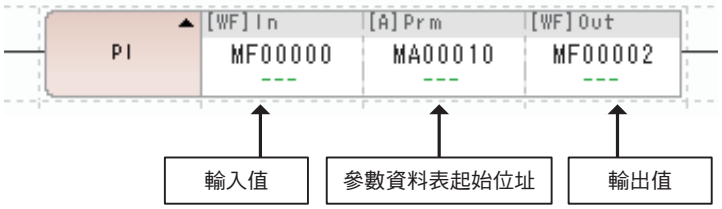


PI 指令在執行運算時，係利用以下公式來計算輸入值 X(s)、輸出值 Y(s)。

$$\frac{Y(s)}{X(s)} = Kp + Ki \times \frac{1}{Ti \times s}$$

格式

所採用之格式如下。



圖示　：PI
按鍵輸入： PI

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	O	×	×	O	×	×	O	O
參數資料表起始位址 (Prm)	×	×	×	×	×	×	O*	O	O
輸出值 (Out)	×	O*	×	×	O*	×	×	O	×

* C、# 暫存器除外

◆ 整數型 PI 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	Kp	P 增益	P 校正的增益 (增益 1 倍為 100)	IN
2	W	Ki	積分調整增益	積分電路輸入的增益 (增益 1 倍為 100)	IN
3	W	Ti	積分時間	積分時間 (ms)	IN
4	W	IUL	積分上限 LIMIT	I 校正值所對應之上限限制值	IN
5	W	ILL	積分下限 LIMIT	I 校正值所對應之下限限制值	IN
6	W	UL	PI 上限 LIMIT	P + I 校正值所對應之上限限制值	IN
7	W	LL	PI 下限 LIMIT	P + I 校正值所對應之下限限制值	IN
8	W	DB	PI 輸出死區	P + I 校正值所對應之死區幅度	IN
9	W	Y	PI 輸出	輸出 PI 校正值 (輸出至 Out 端)	OUT
10	W	Yi	I 校正值	儲存 I 校正值	OUT
11	W	IREM	I 餘數	儲存 I 餘數	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	IRST	積分重置	積分重置時輸入 ON	IN
1 ~ 7	–	(備用)	輸入用備用繼電器	IN
8 ~ F	–	(備用)	輸出用備用繼電器	OUT

◆ 實數型 PI 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	–	(備用)	備用暫存器	–
2	F	Kp	P 增益	P 校正的增益	IN
4	F	Ki	積分調整增益	積分電路輸入的增益	IN
6	F	Ti	積分時間	積分時間 (s)	IN
8	F	IUL	積分上限 LIMIT	I 校正值所對應之上限限制值	IN
10	F	ILL	積分下限 LIMIT	I 校正值所對應之下限限制值	IN
12	F	UL	PI 上限 LIMIT	P + I 校正值所對應之上限限制值	IN
14	F	LL	PI 下限 LIMIT	P + I 校正值所對應之下限限制值	IN
16	F	DB	PI 輸出死區	P + I 校正值所對應之死區幅度	IN
18	F	Y	PI 輸出	輸出 PI 校正值 (輸出至 Out)	OUT
20	F	Yi	I 校正值	儲存 I 校正值	OUT

* 繼電器輸出入的配置方式與整數型相同。

◆ 指令內部運算

依照以下運算所輸入的偏差 X，計算輸出值 (PI 校正值)。

以下計算公式的 Yi' 代表 Yi 前次的 I 校正值，而 Ts 則為掃瞄時間設定值。

補充　積分重置 (IRST) 功能開啟時，以 I 校正值為 0，並進行 PI 校正值的運算。

P 校正值 = (Kp x X) 的上下限限制 (UL、LL)

Yi (I 校正值) = {(Ki x X + IREM)/ $\frac{Ti}{Ts}$ + Yi'} 的上下限限制 (IUL、ILL)

Y (PI 校正值) = P 校正值 + I 校正值的上下限限制、(UL、LL)、死區 A (幅度為 DB)

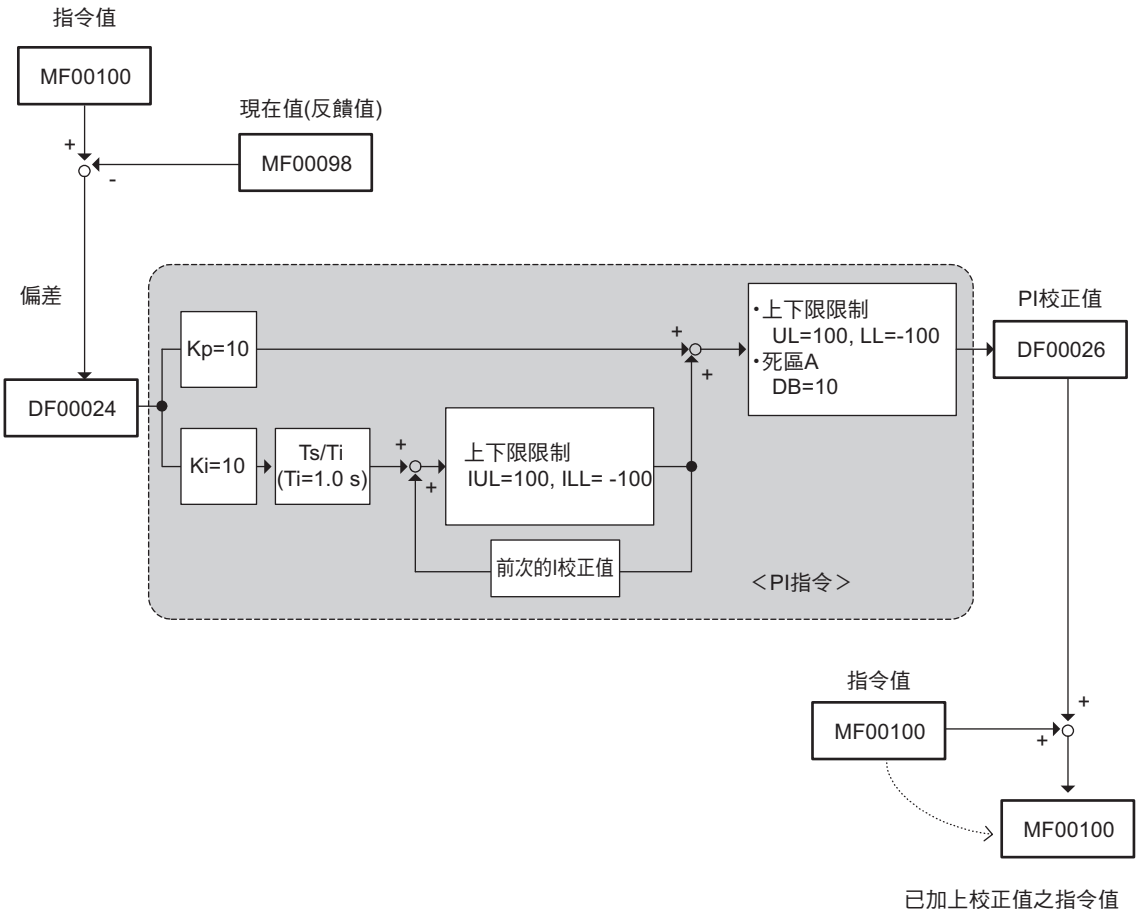
程式範例

以下為計算指令值 (MF00100) 時，將 PI 校正值納入考量之程式範例。

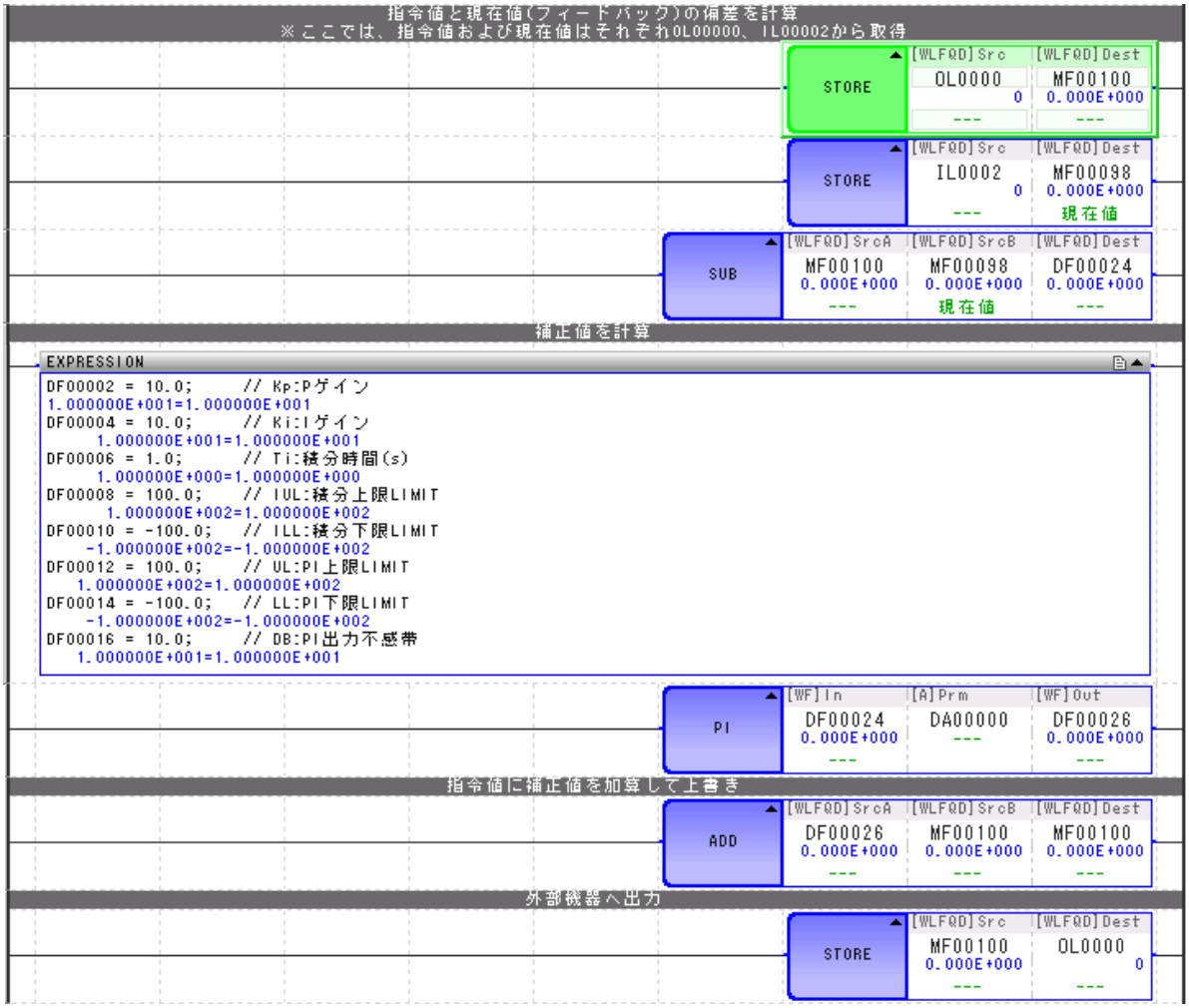
先計算指令值 (MF00100) 和現在值 (MF00098) 之偏差值 (DF00024)，然後輸入 PI 指令。

接著將輸出的 PI 校正值 (DF00026) 加上原來的指令值 (MF00100)，以求出目前所輸出的指令值。

以下為程式方塊圖。



以下為程式範例。



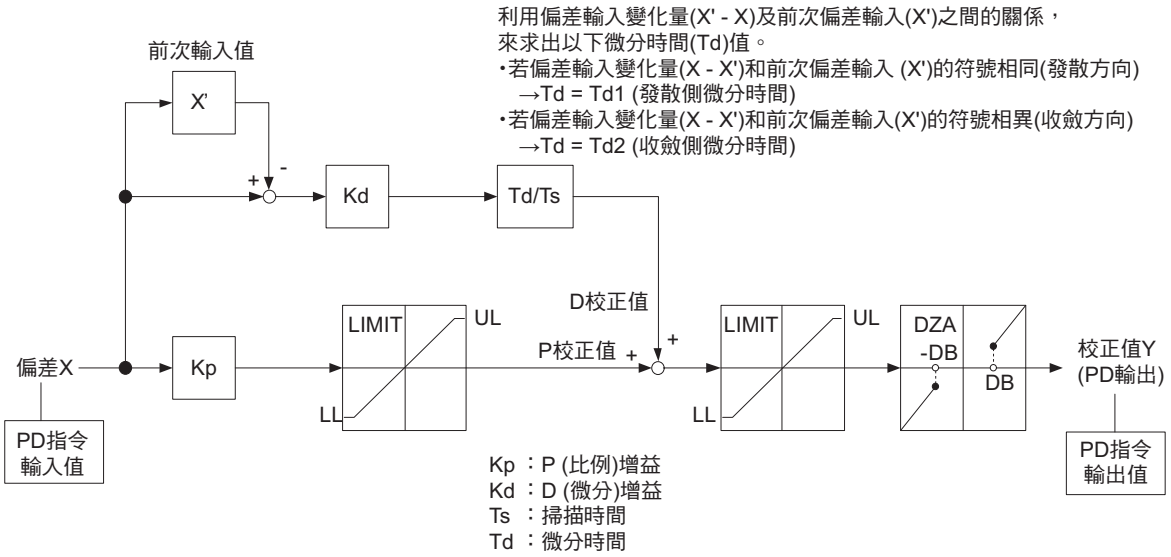
(註)OL00000 (指令值) 及 IL00002 (反饋值) 為被配置到外部裝置的暫存器。

PD 控制 (PD)

輸入偏差 X 後，根據您所設定的參數資料表內容，進行 PD 運算及執行範圍內的運算，最後再輸出校正值 Y。

可使用 PD 指令的輸入值為整數型及實數型。不適用於長整數型。

此外，整數型和實數型的參數資料表架構各不相同。

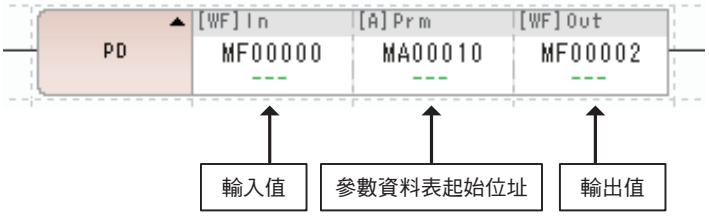


PD 指令在執行運算時，係利用以下公式來計算輸入值 X(s)、輸出值 Y(s)。

$$\frac{Y(s)}{X(s)} = Kp + Kd \times Td \times S$$

格式

所採用之格式如下。



圖示　：PD

按鍵輸入：PD

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	○	×	×	○	×	×	○	○
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*	○	○
輸出值 (Out)	×	○*	×	×	○*	×	×	○	×

* C、# 暫存器除外

◆ 整數型 PD 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	Kp	P 增益	P 校正的增益 (增益 1 倍為 100)	IN
2	W	Kd	D 增益	微分電路輸入增益 (增益 1 倍為 100)	IN
3	W	Td1	發散側微分時間	當輸入為發散方向時之微分時間 (ms)	IN
4	W	Td2	收斂側微分時間	當輸入為收斂方向時之微分時間 (ms)	IN
5	W	UL	PD 上限 LIMIT	P + D 校正值所對應之上限限制值	IN
6	W	LL	PD 下限 LIMIT	P + D 校正值所對應之下限限制值	IN
7	W	DB	PD 輸出死區	P + D 校正值所對應之死區幅度	IN
8	W	Y	PD 輸出	輸出 PI 校正值 (輸出至 Out)	OUT
9	W	X	儲存輸入值	儲存本次偏差輸入值	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0 ~ 7	–	(備用)	輸入用備用繼電器	IN
8 ~ F	–	(備用)	輸出用備用繼電器	OUT

◆ 實數型 PD 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	–	(備用)	備用暫存器	–
2	F	Kp	P 增益	P 校正的增益	IN
4	F	Kd	D 增益	微分電路輸入增益	IN
6	F	Td1	發散側微分時間	當輸入為發散方向時之微分時間 (s)	IN
8	F	Td2	收斂側微分時間	當輸入為收斂方向時之微分時間 (s)	IN
10	F	UL	PD 上限 LIMIT	P + D 校正值所對應之上限限制值	IN
12	F	LL	PD 下限 LIMIT	P + D 校正值所對應之下限限制值	IN
14	F	DB	PD 輸出死區	P + D 校正值所對應之死區幅度	IN
16	F	Y	PD 輸出	輸出 PD 校正值 (輸出至 Out)	OUT
18	F	X	儲存輸入值	儲存本次偏差輸入值	OUT

* 繼電器輸出入配置方式與整數型相同。

◆ 指令內部運算

依照以下運算所輸入的偏差 X，計算輸出值 (PD 校正值)。

在下述計算公式中，X' 代表 X 的前次輸入值、Ts 為掃瞄時間設定值、Td 則為微分時間。

若微分時間 (Td) 的 (X - X') 與 X' 符號相同時，就會變為 Td1，(X - X') 與 X' 符號相異時，則為 Td2。

P 校正值 = (Kp x X) 的上下限限制 (UL、LL)

D 校正值 = $Kd \times (X - X') \times \frac{Td}{Ts}$ 的上下限限制 (IUL、ILL)

PD 校正值 = (P 校正值 + D 校正值) 的上下限限制 (UL、LL)、死區 A (死區幅度 = DB)

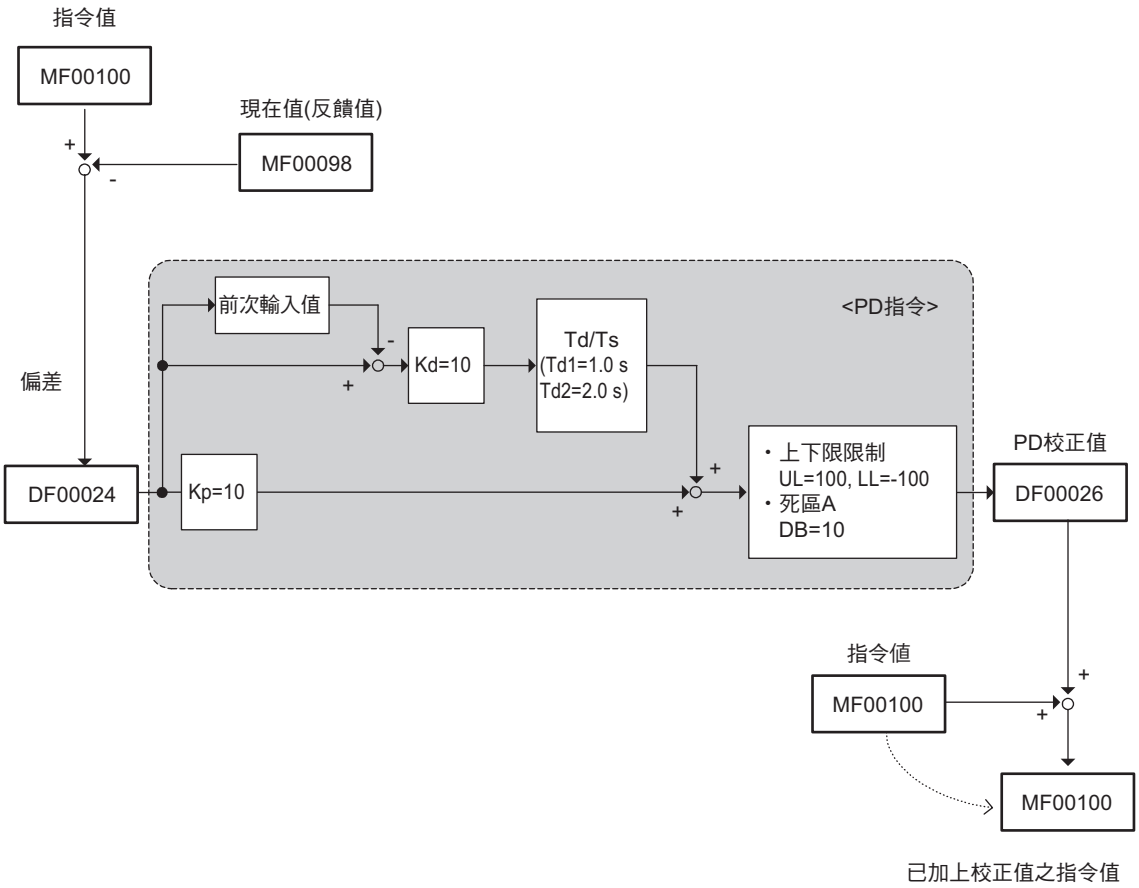
程式範例

以下為計算指令值 (MF00100) 時，將 PD 校正值納入考量之程式範例。

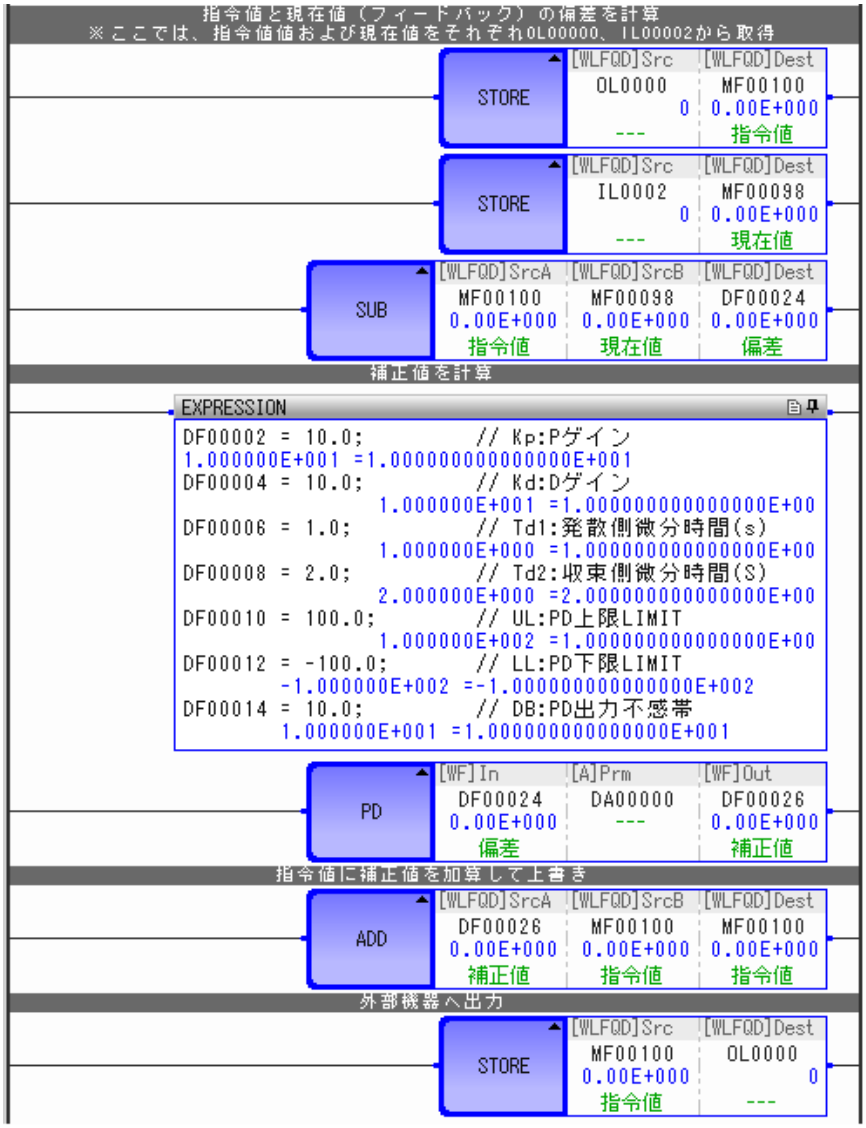
先計算指令值 (MF00100) 和現在值 (MF00098) 之偏差 (DF00024)，然後輸入 PD 指令。

將輸出的 PD 校正值 (DF000026) 加上原來的指令值 (MF00100)，以求出目前所輸出的指令值。

以下為程式方塊圖。



以下為程式範例。



(註)OL00000 (指令值) 及 IL00002 (反饋值) 為配置到外部裝置的暫存器。

補充事項

◆ 傳送函數

以下公式係用來表示 PD 運算的傳送函數。

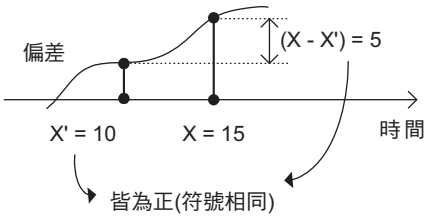
X(s) 為輸入值，Y(s) 為輸出值。

$$\frac{Y(s)}{X(s)} = Kp + Kd \times Td \times S$$

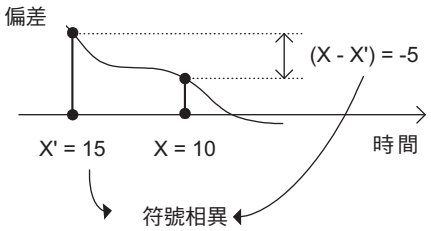
◆ 本次偏差 X 及前次偏差 X' 於發散側 / 收斂側之關係

下圖為本次偏差 X 和前次偏差 X' 於發散側 / 收斂側之關係。

＜以發散側 (偏差呈發散狀態) 為例＞



＜以收斂側 (偏差呈收斂狀態) 為例＞



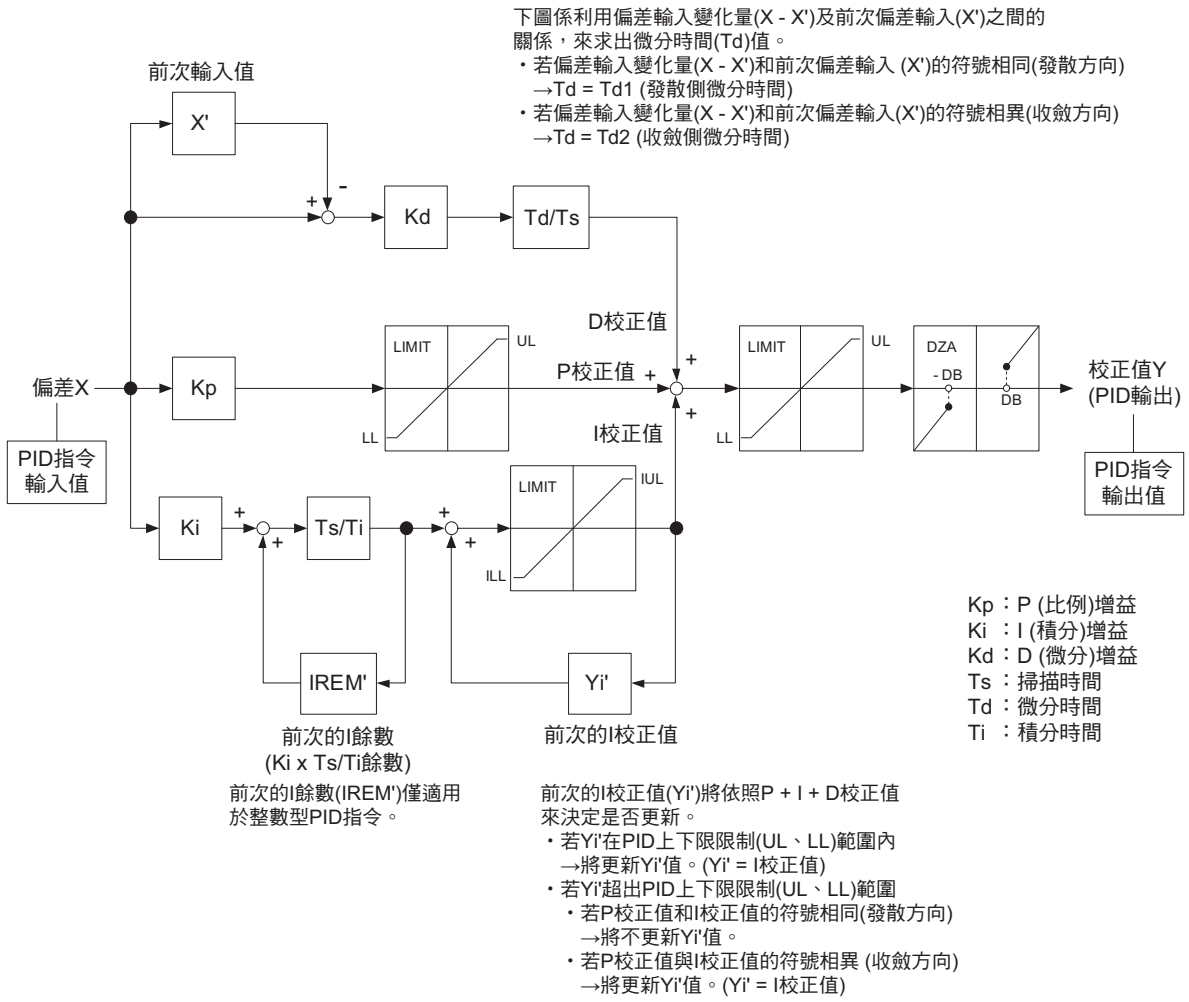
PID 控制 (PID)

在輸入偏差 X 後，根據您所設定的參數資料表內容，進行 PID 運算及執行範圍內的運算，最後再輸出校正值 Y。

將參數資料表內的積分重置開啟後，就會以 I 校正值 =0，進行 PI 校正值運算。

可使用 PID 指令的輸入值為整數型及實數型。不適用於長整數型。

此外，整數型和實數型的參數資料表架構各不相同。

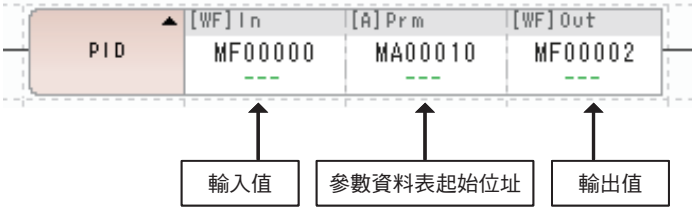


以下公式係利用輸入值 X(s)、輸出值 Y(s)，來進行 PID 指令運算。

$$\frac{Y(s)}{X(s)} = Kp + Ki \times \frac{1}{Ti \times s} + Kd \times Td \times S$$

格式

所採用之格式如下。



圖示　：PID
按鍵輸入：PID

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	O	×	×	O	×	×	O	O
參數資料表起始位址 (Prm)	×	×	×	×	×	×	O*	O	O
輸出值 (Out)	×	O*	×	×	O*	×	×	O	×

* C、# 暫存器除外

◆ 整數型 PID 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	Kp	P 增益	P 校正的增益 (增益 1 倍為 100)	IN
2	W	Ki	I 增益	積分電路輸入的增益 (增益 1 倍為 100)	IN
3	W	Kd	D 增益	微分電路輸入的增益 (增益 1 倍為 100)	IN
4	W	Ti	積分時間	積分時間 (ms)	IN
5	W	Td1	發散側微分時間	當輸入為發散方向時之微分時間 (ms)	IN
6	W	Td2	收斂側微分時間	當輸入為收斂方向時之微分時間 (ms)	IN
7	W	IUL	積分上限 LIMIT	I 校正值所對應之上限限制值	IN
8	W	ILL	積分下限 LIMIT	I 校正值所對應之下限限制值	IN
9	W	UL	PID 上限 LIMIT	P + I 校正值所對應之上限限制值	IN
10	W	LL	PID 下限 LIMIT	P + I 校正值所對應之下限限制值	IN
11	W	DB	PID 輸出死區	P + I 校正值所對應之死區幅度	IN
12	W	Y	PID 輸出	輸出 PI 校正值 (輸出至 Out)	OUT
13	W	Yi	I 校正值	儲存 I 校正值	OUT
14	W	IREM	I 的餘數	儲存 I 餘數	OUT
15	W	X	儲存輸入值	儲存本次偏差輸入值	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	IRST	積分重置	積分重置時輸入 ON	IN
1 ~ 7	–	(備用)	輸入用備用繼電器	IN
8 ~ F	–	(備用)	輸出用備用繼電器	OUT

◆ 實數型 PID 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	–	(備用)	備用暫存器	IN
2	F	Kp	P 增益	P 校正的增益	IN
4	F	Ki	I 增益	積分電路輸入的增益	IN
6	F	Kd	D 增益	微分電路輸入的增益	IN
8	F	Ti	積分時間	積分時間 (s)	IN
10	F	Td1	發散側微分時間	當輸入為發散方向時之微分時間 (s)	IN
12	F	Td2	收斂側微分時間	當輸入為收斂方向時之微分時間 (s)	IN
14	F	IUL	積分上限 LIMIT	I 校正值所對應之上限限制值	IN
16	F	ILL	積分下限 LIMIT	I 校正值所對應之下限限制值	IN
18	F	UL	PID 上限 LIMIT	P + I + D 校正值所對應之上限限制值	IN
20	F	LL	PID 下限 LIMIT	P + I + D 校正值所對應之下限限制值	IN
22	F	DB	PID 輸出死區	P + I + D 校正值所對應之死區幅度	IN
24	F	Y	PID 輸出	輸出 PID 校正值 (輸出至 Out)	OUT
26	F	Yi	I 校正值	儲存 I 校正值	OUT
28	F	X	儲存輸入值	儲存本次偏差輸入值	OUT

* 繼電器輸出入端的配置方法與整數型相同。

◆ 指令內部運算

依照以下運算所輸入的偏差 X，計算輸出值 (PID 校正值)。

在以下的計算公式中，X' 代表 X 的前次輸入值、Y' 為 Y 的前次 I 校正值、Ts 為掃描時間設定值、Td 則為微分時間。

若微分時間 (Td) 的 (X - X') 與 X' 符號相同時，就會變為 Td1，(X - X') 與 X' 符號相異時，則為 Td2。

補充　將積分重置 (IRST) 開啟，並以 I 校正值 = 0，然後再進行 PID 校正值運算。

P 校正值 = (Kp x X) 的上下限限制 (UL、LL)

Yi (I 校正值) = $\{(Ki \times X + IREM)/\frac{Ti}{Ts} + Yi'\}$ 的上下限限制 (IUL、ILL)

D 校正值 = Kd x (X - X') x $\frac{Td}{Ts}$ 的上下限限制 (IUL、ILL)

Y (PID 校正值) = (P + I + DI 校正值) 的上下限限制 (UL、LL)、死區 A (幅度 DB)

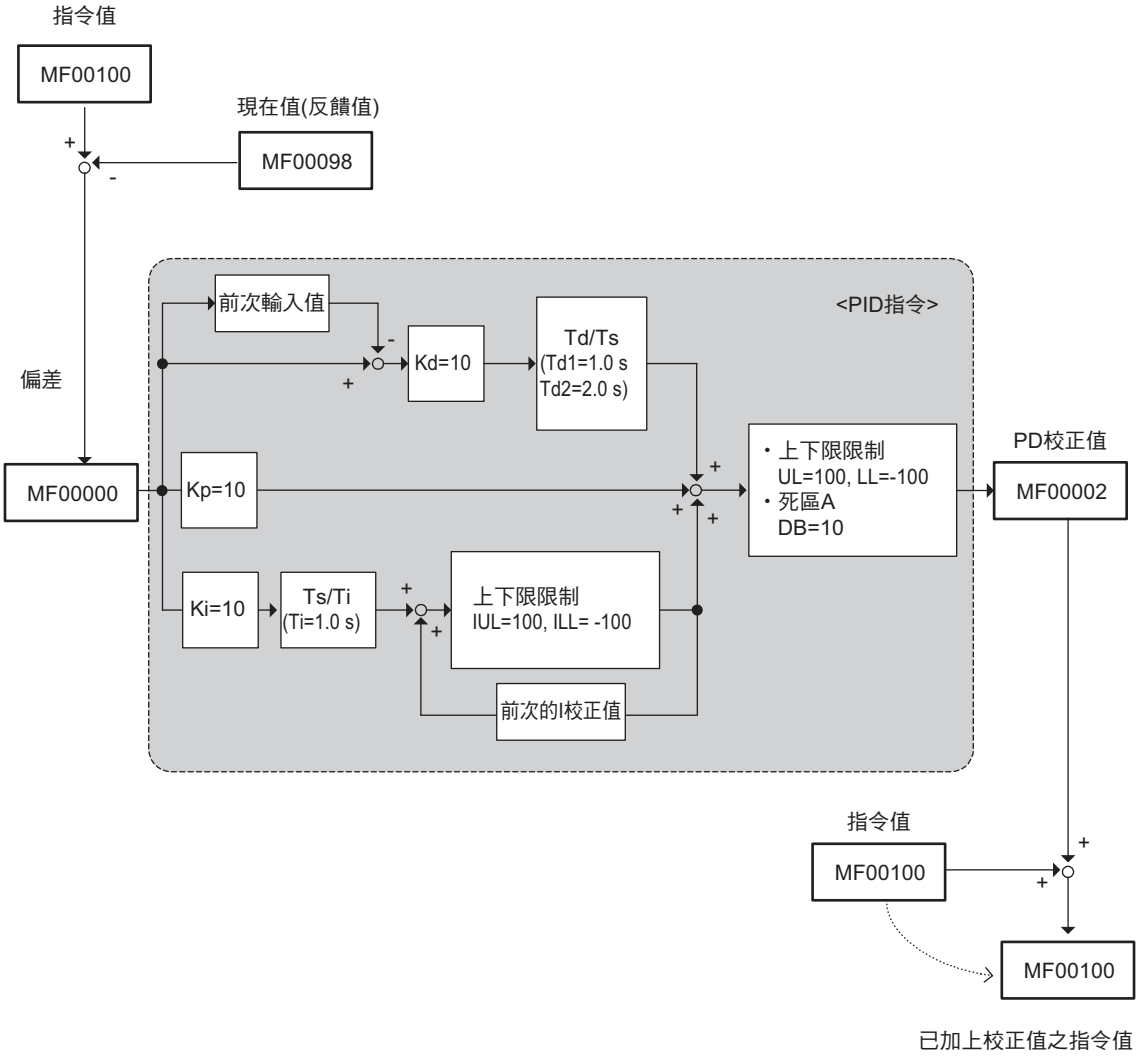
程式範例

以下為計算指令值 (MF00100) 時，將 PID 校正值納入考量之程式範例。

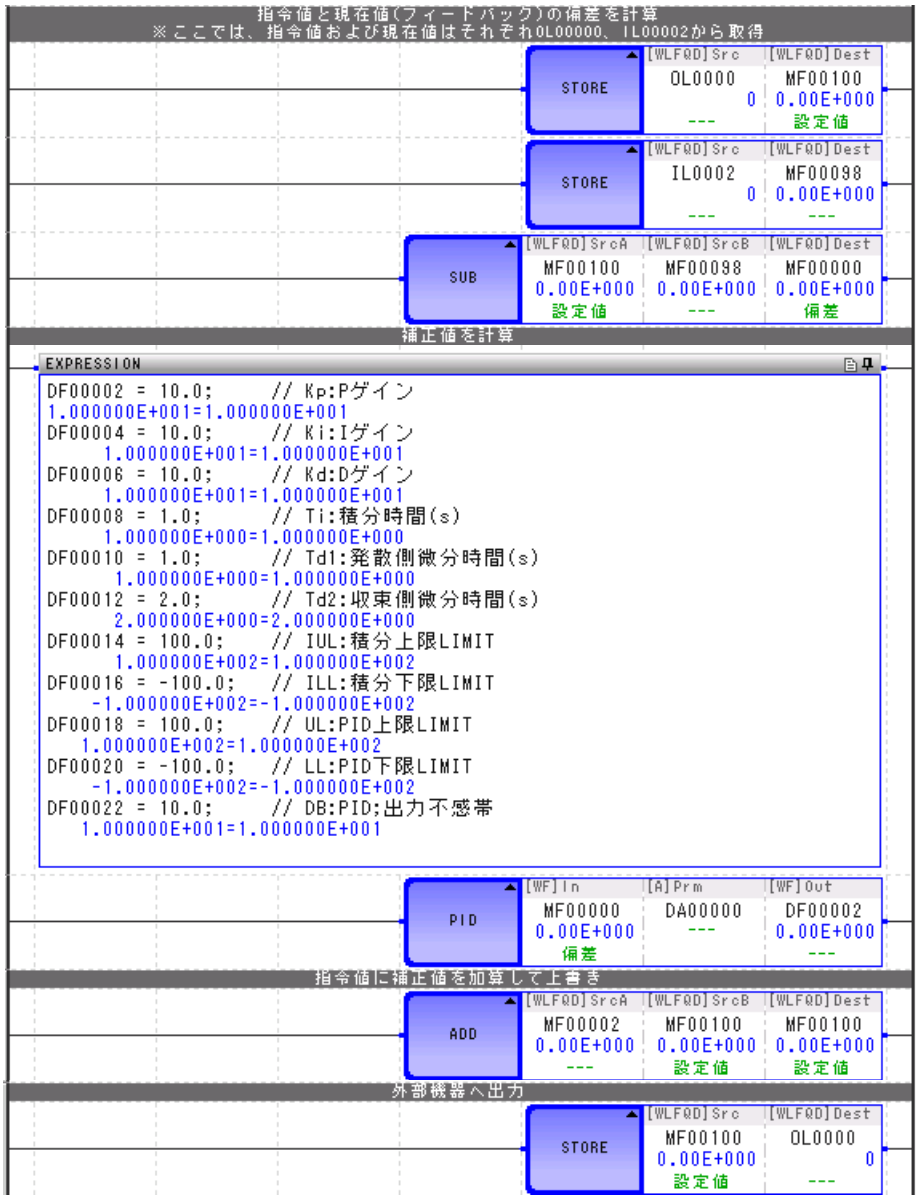
先計算指令值 (MF00100) 和現在值 (MF00098) 之偏差 (MF00000))，然後輸入 PID 指令。

再將輸出的 PID 校正值 (MF00002) 加上原來的指令值 (MF00100)，以求出目前所輸出的指令值。

以下為程式方塊圖。



以下為程式範例。



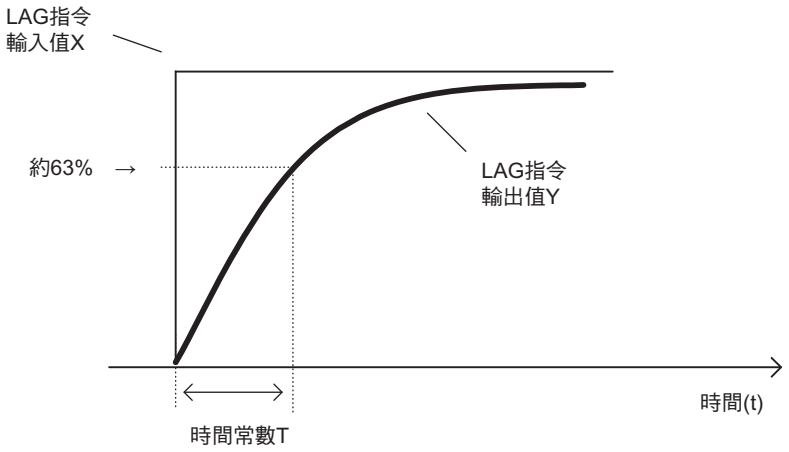
(註)OL00000 (指令值) 及 IL00002 (反饋值) 為配置到外部裝置的暫存器值。

1 次延遲 (LAG)

可根據事先設定好的參數資料表內容來運算 1 次延遲。

整數型或實數型皆可使用為 LAG 指令輸入值。不適用於長整數型。

整數型和實數型的參數資料表架構各不相同。



上圖為 LAG 運算圖，計算公式如下。

$$\frac{Y(s)}{X(s)} = \frac{1}{1+T\times s}$$

亦即

$$T\times \frac{dY}{dt} + Y = X$$

假設 LAG 指令內部為 dt = Ts、dY = Y - Y'，並進行以下運算。

在下述公式中，Y' 代表前次輸出值、Ts 為掃描時間設定值、REM 則為餘數。

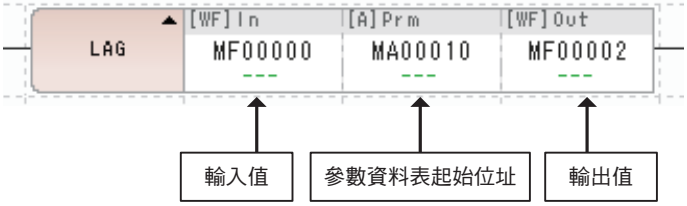
Ts 所使用的單位與 T 相同。

$$Y = \frac{T\times Y' + Ts\times X + REM}{T + Ts}$$

補充　當 LAG 重置 (IRST) 為開啟時，就會輸出 Y = 0、REM = 0 等數值。

格式

所採用之格式如下。



圖示　：LAG

按鍵輸入：LAG

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	○	×	×	○	×	×	○	○
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*	○	○
輸出值 (Out)	×	○*	×	×	○*	×	×	○	×

* C、# 暫存器除外

◆ 整數型 LAG 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	T	1 次延遲時間常數	1 次延遲時間常數 (ms)	IN
2	W	Y	LAG 輸出	LAG 輸出 (輸出至輸出值)	OUT
3	W	REM	餘數	儲存餘數	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	IRST	LAG 重置	LAG 重置時輸入 ON	IN
1 ~ 7	–	(備用)	輸入用備用繼電器	IN
8 ~ F	–	(備用)	輸出用備用繼電器	OUT

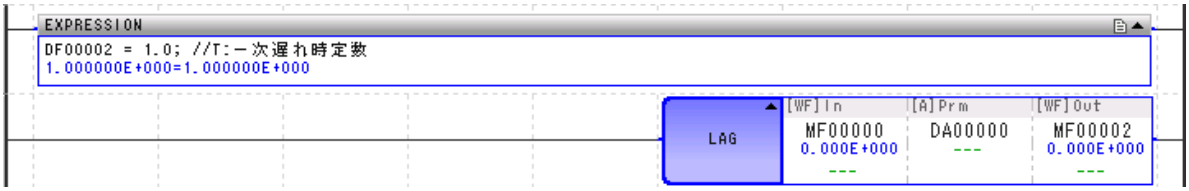
◆ 實數型 LAG 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	–	(備用)	備用暫存器	–
2	F	T	1 次延遲時間常數	1 次延遲時間常數 (s)	IN
4	F	Y	LAG 輸出	LAG 輸出 (輸出至輸出值)	OUT

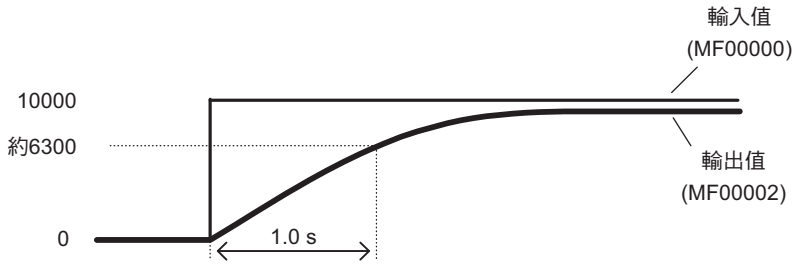
* 繼電器輸出入端的配置方法與整數型相同。

程式範例

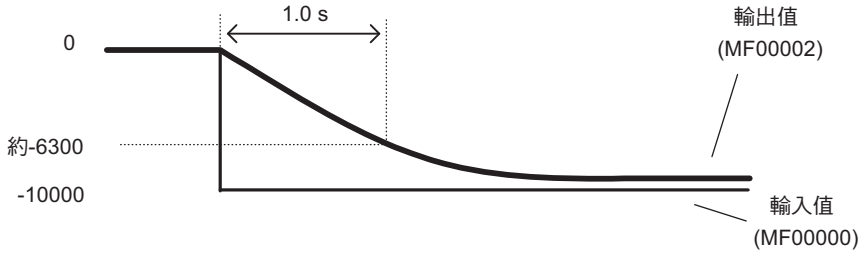
以下為假設 1 次延遲時間常數為 1.0，參數資料表的輸入值為 MF00000、輸出值為 MF00002 時，執行 LAG 指令之程式範例。



下圖為假設 MF00000 從 0 變為 10000 時，MF00002 之變化狀態。



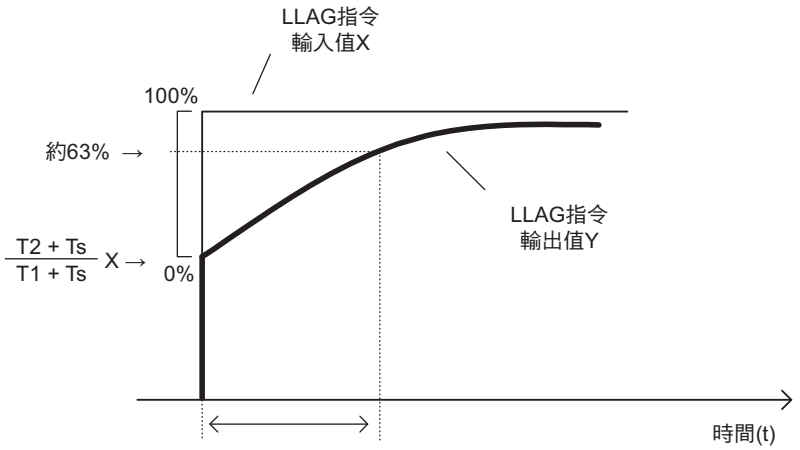
下圖為假設 MF00000 從 0 變為 -10000 時，MF00002 之變化狀態。



相位前進延遲 (LLAG)

可根據事先設定好的參數資料表內容來運算相位前進延遲。整數型及實數型資料皆適合當作 LLAG 運算之輸入值。不適用於長整數型資料。

整數型和實數型的參數資料表架構各不相同。



相位延遲時間常數T1

上圖為 LLAG 運算圖，計算公式如下。

$$\frac{Y(s)}{X(s)} = \frac{1+T2 \times s}{1+T1 \times s}$$

亦即

$$T1 \times \frac{dY}{dt} + Y = T2 \times \frac{dX}{dt} + X$$

假設 LLAG 指令內部為 dt = Ts、dY = Y - Y'、dX = X - X'，並進行以下運算。

在以下計算公式中，Y' 代表前次輸出值、X' 為前次輸入值、Ts 為掃描時間設定值、REM 則為餘數。

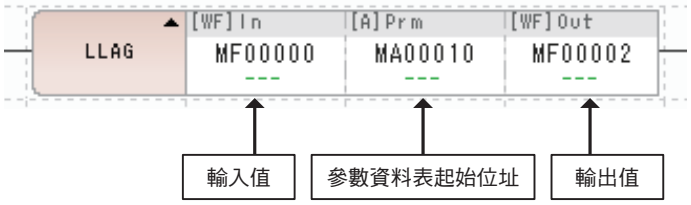
Ts 所使用的單位與 T1 相同。

$$Y = \frac{T1 \times Y' + (T2+Ts) \times X - T2 \times X' + REM}{T1 + Ts}$$

補充　當 LLAG 重置 (IRST) 為 ON 時，就會輸出 Y = 0、REM = 0、X = 0。

格式

所採用之格式如下。



圖示　：LLAG
按鍵輸入：LLAG

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	○	×	×	○	×	×	○	○
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*	○	○
輸出值 (Out)	×	○*	×	×	○*	×	×	○	×

* C、# 暫存器除外

◆ 整數型 LLAG 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	T2	相位前進之時間常數	相位前進之時間常數 (ms)	IN
2	W	T1	相位延遲之時間常數	相位延遲之時間常數 (ms)	IN
3	W	Y	LLAG 輸出	LLAG 輸出 (輸出至輸出值)	OUT
4	W	REM	餘數	儲存餘數	OUT
5	W	X	儲存輸入值	儲存輸入值	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	IRST	LLAG 重置	LLAG 重置時輸入 ON	IN
1 ~ 7	–	(備用)	輸入用備用繼電器	IN
8 ~ F	–	(備用)	輸出用備用繼電器	OUT

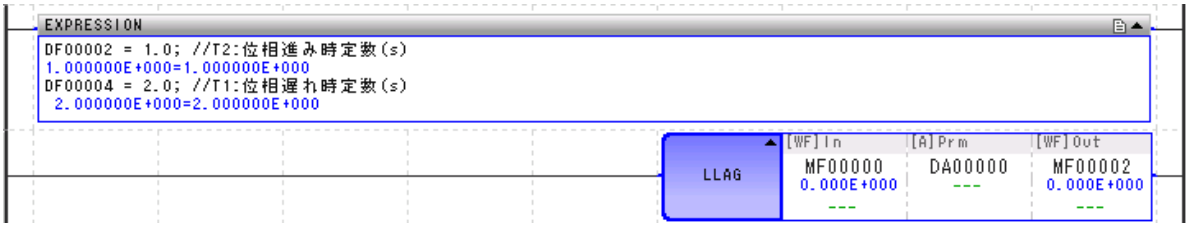
◆ 實數型 LAG 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	–	(備用)	備用暫存器	–
2	F	T2	相位前進之時間常數	相位前進之時間常數 (s)	IN
4	F	T1	相位延遲之時間常數	相位延遲之時間常數 (s)	IN
6	F	Y	LLAG 輸出	LLAG 輸出 (輸出至輸出值)	OUT
8	F	X	儲存輸入值	儲存輸入值	OUT

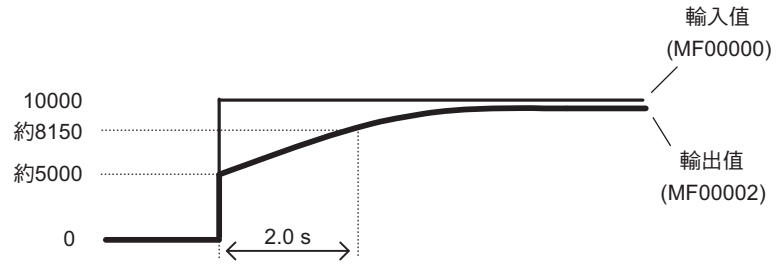
* 繼電器輸出入端的配置方法與整數型相同。

程式範例

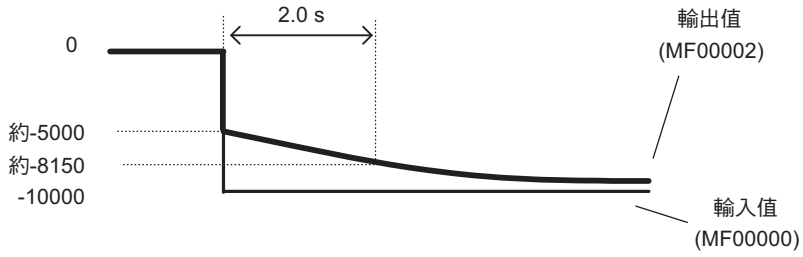
以下係假設相位前進時間常數為 1.0 秒、相位延遲時間常數為 2.0 秒時，以 MF00000 為參數資料表輸入值，MF00002 為輸出值，並執行 LLAG 指令之程式範例。



下圖為假設 MF00000 從 0 變為 10000 時，MF00002 之變化狀態。



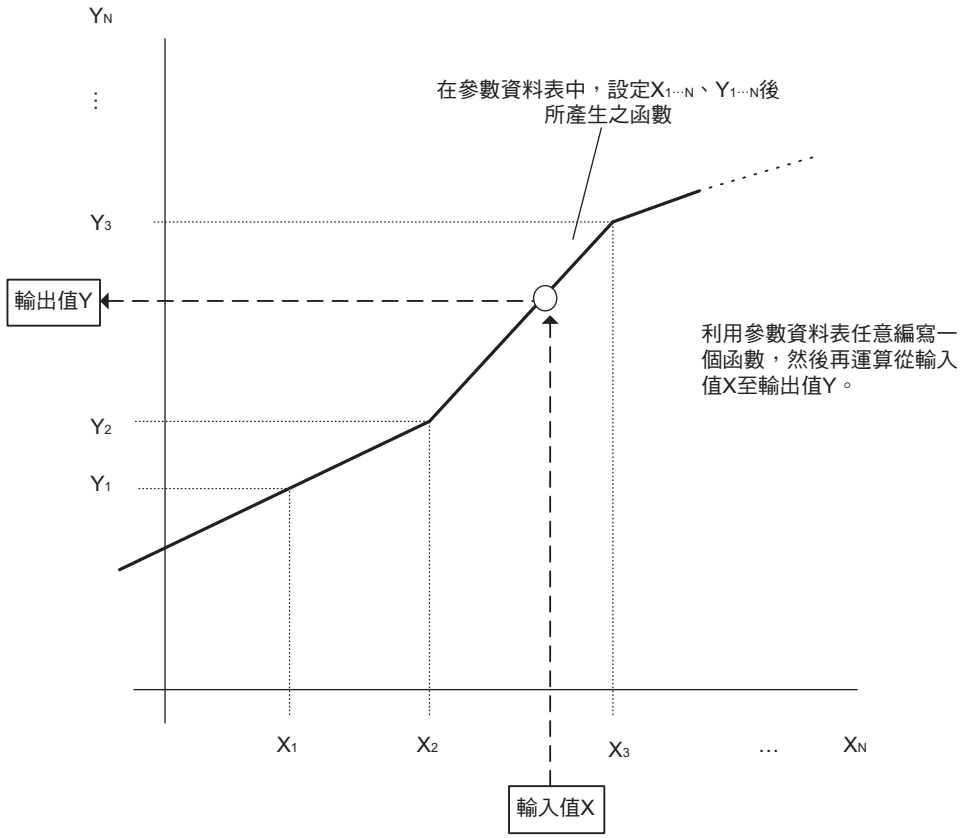
下圖為假設 MF00000 從 0 變為 -10000 時，MF00002 之變化狀態。



函數產生器 (FGN)

可根據您所設定的參數資料表來產生函數，然後再針對該函數，運算當輸入值為 X 時之輸出值 Y。

根據輸入值 X 的資料類型，為整數型、長整數型、實數型、4 長整數型及倍精度實數型的 FGN 指令，因而參數資料表結構皆不同。

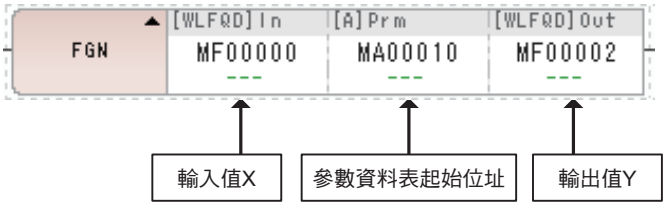


註記

編寫參數資料表時，請遵守 $X_1 < X_2 < \ldots < X_N$ 的原則。

格式

所採用之格式如下。



圖示 ： FGN

按鍵輸入： FGN

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 X(In)	×	O	O	O	O	O	×	O	O
參數資料表起始位址 (Prm)	×	×	×	×	×	×	O	×	×
輸出值 Y(Out)	×	O*	O*	O*	O*	O*	×	O	×

* C、# 暫存器除外

◆ 整數型 FGN 指令參數資料表

若輸入值 X 為整數型資料，請使用整數型 FGN 指令。

以下為參數資料表的編寫方法。

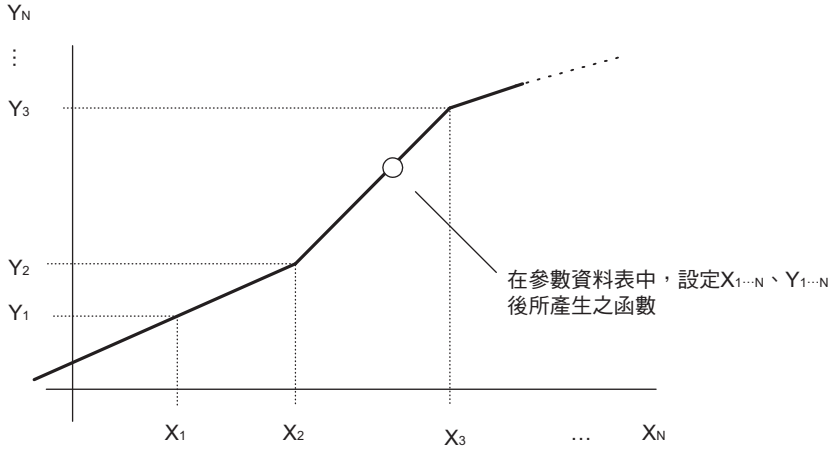
位址	資料類型	符號	名稱
0	W	N	X、Y 的組數
1	W	X_1	資料 X_1
2	W	Y_1	資料 Y_1
3	W	X_2	資料 X_2
4	W	Y_2	資料 Y_2
:	:	:	:
2N−1	W	X_N	資料 X_N
2N	W	Y_N	資料 Y_N

◆ 長整數型 / 實數型 FGN 指令之參數資料表

若輸入值 X 為長整數型，為長整數型 FGN 指令，若輸入值 X 為實數型，則為實數型 FGN 指令。

以下為參數資料表的編寫方法。

位址	資料類型	符號	名稱
0	W	N	X、Y 的組數
1	W	–	予備
2	L/F	X_1	資料 X_1
4	L/F	Y_1	資料 Y_1
6	L/F	X_2	資料 X_2
8	L/F	Y_2	資料 Y_2
:	:	:	:
4N–2	L/F	X_N	資料 X_N
4N	L/F	Y_N	資料 Y_N



◆ 4 長整數型 / 倍精度實數型 FGN 指令之參數資料表

若輸入值 X 為 4 長整數型，為 4 長整數型 FGN 指令，若輸入值 X 為倍精度實數型，則為倍精度實數型 FGN 指令。

以下為參數資料表的編寫方法。

位址	資料類型	符號	名稱
0	W	N	X、Y 的組數
1	W	–	備用
2	L	–	備用
4	Q/D	X_1	資料 X_1
8	Q/D	Y_1	資料 Y_1
12	Q/D	X_2	資料 X_2
16	Q/D	Y_2	資料 Y_2
:	:	:	:
8N–4	Q/D	X_N	資料 X_N
8N	Q/D	Y_N	資料 Y_N

註記

無論參數資料表屬於整數型、長整數型、實數型、4 長整數型及倍精度實數型等其中一種類型，設定時皆必須遵守 $X_1 < X_2 < \ldots < X_N$ 的原則。

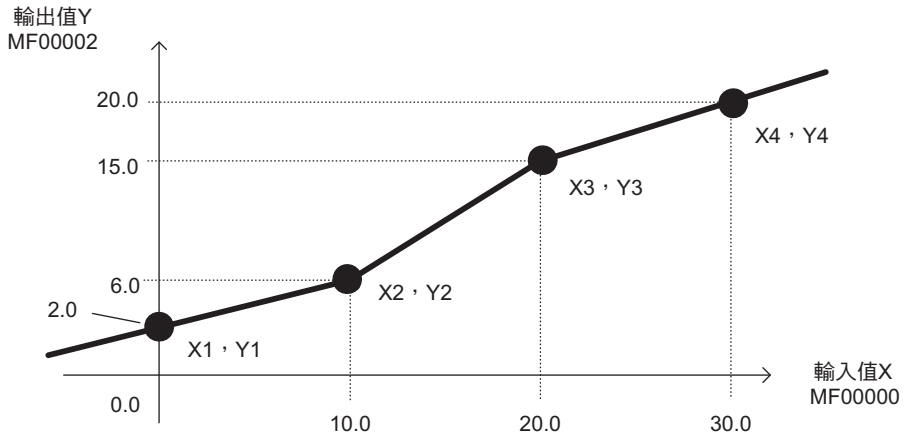
程式範例

以下為參數資料表使用實數型 FGN 指令來產生函數之程式範例。

組數	4
X1，Y1	0.0，2.0
X2，Y2	10.0，6.0
X3，X4	20.0，15.0
X4，Y4	30.0，20.0

```
IF  HighScan.FirstScan == true
NL
2   EXPRESSION
    DW00000 = 4;     // 組数(X,Yの組の数)
    DF00002 = 0.0;   DF00004 = 2.0;   // X1, Y1
    DF00006 = 10.0;  DF00008 = 6.0;   // X2, Y2
    DF00010 = 20.0;  DF00012 = 15.0;  // X3, Y3
    DF00014 = 30.0;  DF00016 = 20.0;  // X4, Y4
END_IF
FGN   [WLFQD]In   [A]Prm   [WLFQD]Out
      MF00000     DA00000  MF00002
      -1.2E+000   ---      1.52E+000
      ---                  ---
```

下圖為輸入值 X (MF00000) 和輸出值 Y (MF00002) 之間的關係。



補充事項

輸出值 Y 的運算方式如下。

- **X_n ≦ 輸入 X ≦ X_{n+1} 時，X_n、Y_n 為同一組**

$$輸出值Y = Y_n + \frac{Y_{n+1} - Y_n}{X_{n+1} - X_n} \times (輸入值X - X_n)\ (1 \leqq n \leqq N - 1)$$

- **Xn ≦ 輸入值 X ≦ X_{n+1} 時，X_n、Y_n 為不同組**

若輸入值 X ＜ X_1

$$輸出值Y = Y_1 + \frac{Y_2 - Y_1}{X_2 - X_1} \times (輸入值X - X_1)$$

若輸入值 X ＞ X_N

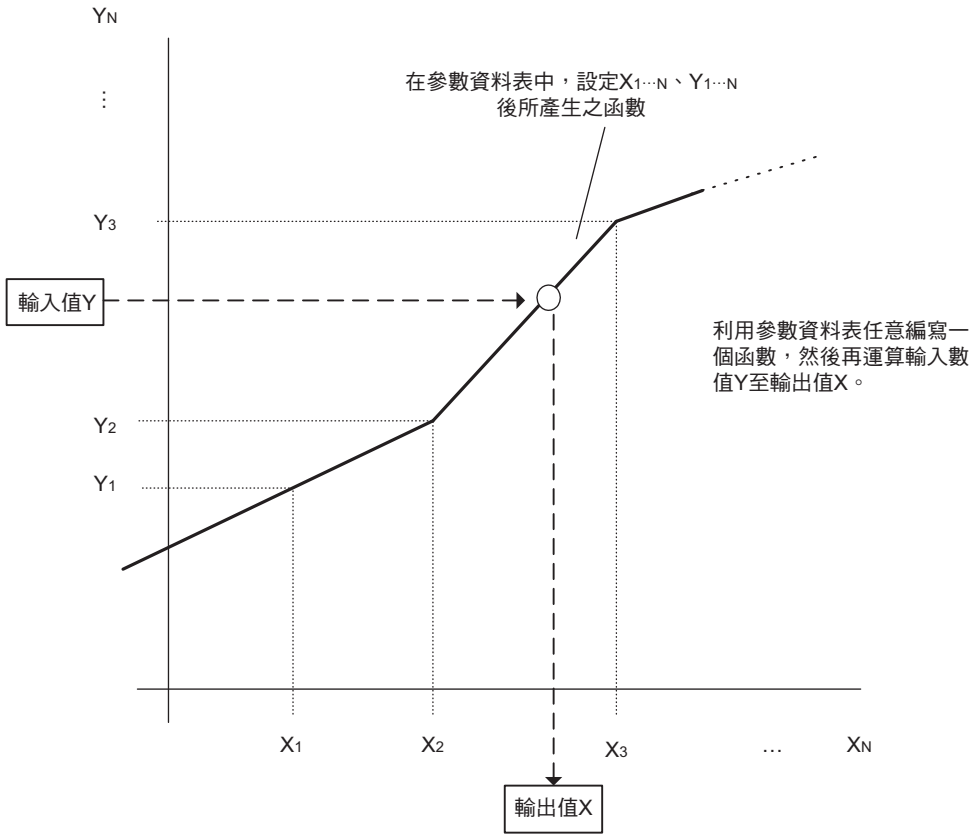
$$輸出值Y = Y_N + \frac{Y_N - Y_{N-1}}{X_N - X_{N-1}} \times (輸入值X - X_N)$$

反函數產生器 (IFGN)

可根據您所設定的參數資料表來產生函數，並針對函數運算輸入值 X。

可依照輸入值 X 的資料類型，選擇適合整數型、長整數型、實數型、4 長整數型及倍精度實數型的 IFGN 指令。

參數資料表的結構與 FGN 指令相同。

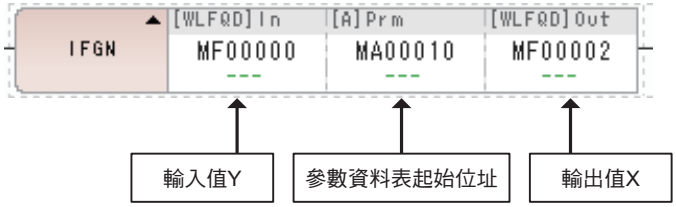


註記

設定參數資料表時，需遵守 $Y_1 < Y_2 < \ldots < Y_N$ 的原則。

格式

所採用之格式如下。



圖示 ： 1/FGN

按鍵輸入： IFGN

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 Y(In)	×	O	O	O	O	O	×	O	O
參數資料表起始位址 (Prm)	×	×	×	×	×	×	O	×	×
輸出值 X(Out)	×	O*	O*	O*	O*	O*	×	O	×

* C、# 暫存器除外

◆ 整數型 IFGN 指令參數資料表

若輸入值 X 為整數型時，為整數型 IFGN 指令。

以下為參數資料表的編寫方法。

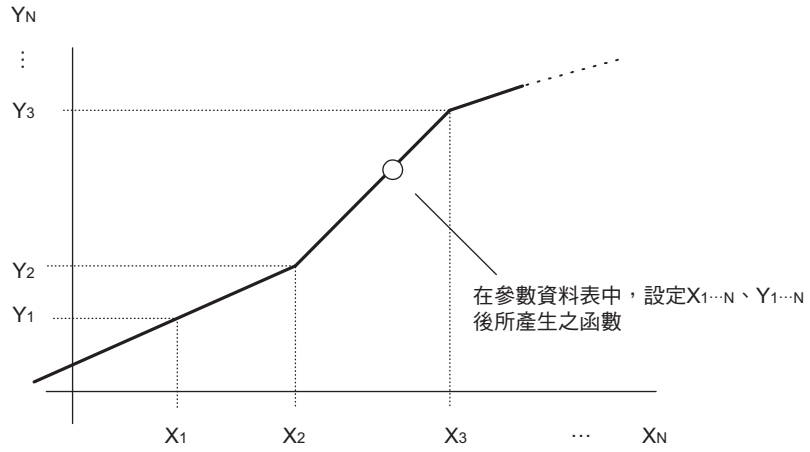
位址	資料類型	符號	名稱
0	W	N	X、Y 的組數
1	W	X_1	資料 X_1
2	W	Y_1	資料 Y_1
3	W	X_2	資料 X_2
4	W	Y_2	資料 Y_2
:	:	:	:
2N–1	W	X_N	資料 X_N
2N	W	Y_N	資料 Y_N

◆ 長整數型 / 實數型 IFGN 指令之參數資料表

若輸入值 Y 為長整數型，為整數型 IFGN 指令，若輸入值 Y 為實數型，則為實數型 IFGN 指令。

以下為參數資料表的編寫方法。

位址	資料類型	符號	名稱
0	W	N	X、Y 的組數
1	W	–	予備
2	L/F	X_1	資料 X_1
4	L/F	Y_1	資料 Y_1
6	L/F	X_2	資料 X_2
8	L/F	Y_2	資料 Y_2
:	:	:	:
4N–2	L/F	X_N	資料 X_N
4N	L/F	Y_N	資料 Y_N



◆ 4 長整數型 / 倍精度實數型 FGN 指令之參數資料表

若輸入值 X 為 4 長整數型，為 4 長整數型 FGN 指令，若輸入值 X 為倍精度實數型，則為倍精度實數型 FGN 指令。

以下為參數資料表的編寫方法。

位址	資料類型	符號	名稱
0	W	N	X、Y 的組數
1	W	–	予備
2	L	–	予備
4	Q/D	X_1	資料 X_1
8	Q/D	Y_1	資料 Y_1
12	Q/D	X_2	資料 X_2
16	Q/D	Y_2	資料 Y_2
:	:	:	:
8N–4	Q/D	X_N	資料 X_N
8N	Q/D	Y_N	資料 Y_N

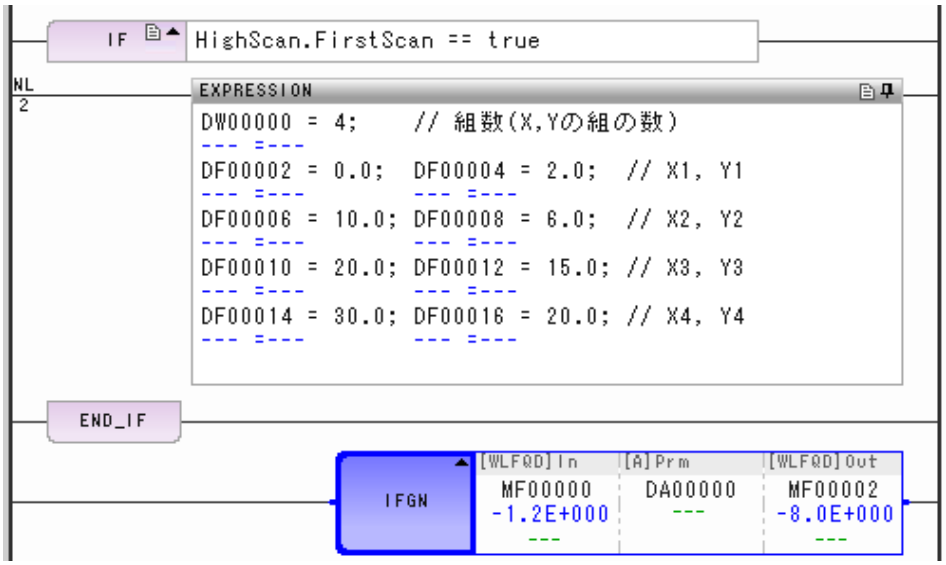

註記

無論參數資料表屬於整數型、長整數型、實數型、4 長整數型及倍精度實數型等其中一種類型，設定時皆必須遵守 $Y_1 < Y_2 < \ldots < Y_N$ 的原則。

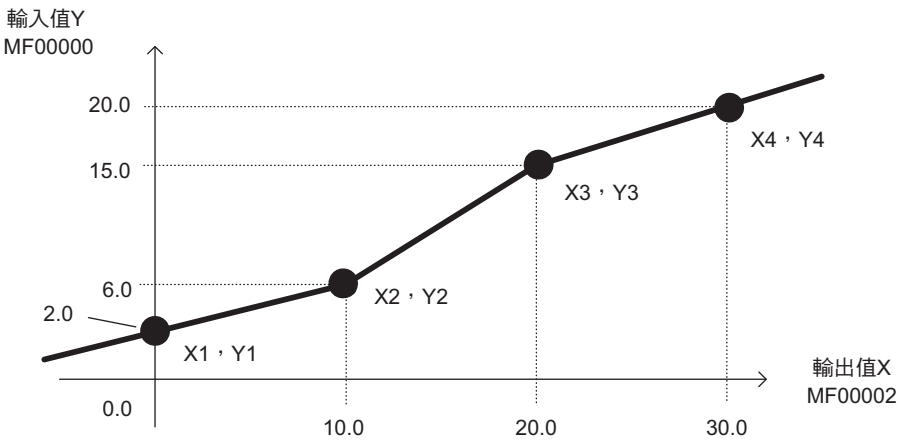
程式範例

以下為在參數資料表使用實數型 IFGN 指令，並使用函數之程式範例。

組數	4
X1，Y1	0.0，2.0
X2，Y2	10.0，6.0
X3，X4	20.0，15.0
X4，Y4	30.0，20.0



下圖為輸入值 Y (MF00000) 和輸出值 X (MF00002) 之間的關係。



補充事項

輸出值 X 的運算方式如下所示。

· $Y_n \leqq$ **輸入值** $Y \leqq Y_{n+1}$ **時，** X_n、Y_n **為同一組**

$$\text{輸出值}X = X_n + \frac{X_{n+1}-X_n}{Y_{n+1}-Y_n} \times (\text{輸入值}Y-Y_n)\ (1 \leqq n \leqq N-1)$$

· $Y_n \leqq$ **輸入值** $Y \leqq Y_{n+1}$ **時，** X_n，Y_n **為不同組**

若輸入值 $Y < Y_1$

$$\text{輸出值}X = X_1 + \frac{X_2-X_1}{Y_2-Y_1} \times (\text{輸入值}Y-Y_1)$$

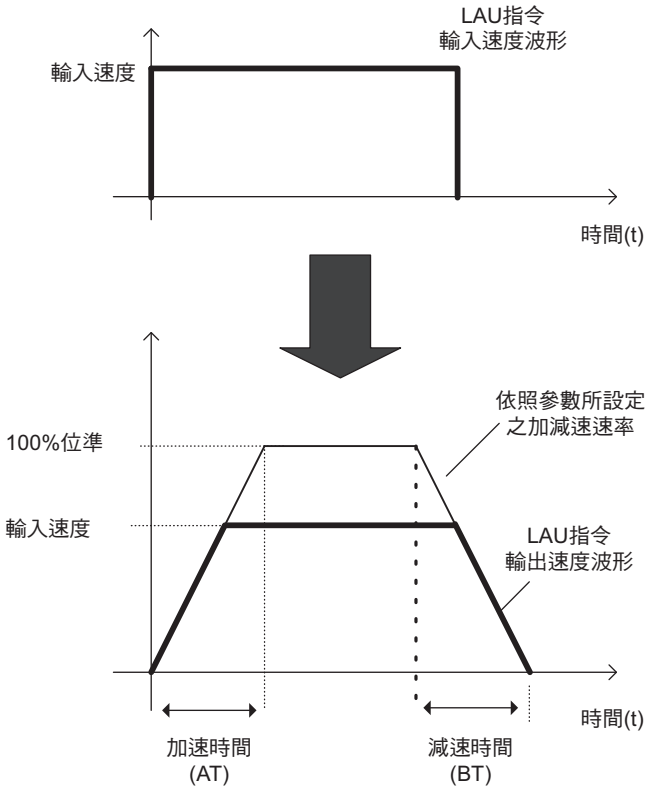
若輸入值 $Y > Y_N$

$$\text{輸出值}X = X_N + \frac{X_N-X_{N-1}}{Y_N-Y_{N-1}} \times (\text{輸入值}Y-Y_N)$$

直線加減速器 1 (LAU)

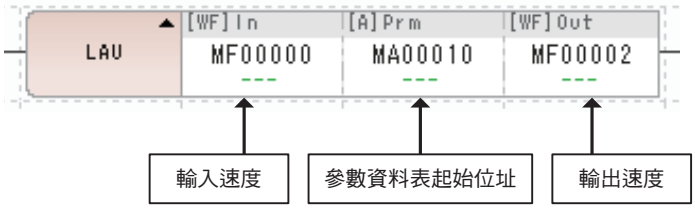
可依照輸入速度，輸出用固定的加減速速率，加速 / 減速後的結果。依照您所設定的參數資料表內容來進行加減速。整數型或實數型資料皆適合當作 LAU 運算輸入值。長整數型、4 長整數型及倍精度實數型資料皆不適用。

參數資料表的結構依整數型或實數型等資料類型而異。



格式

所採用之格式如下。



圖示 ：

按鍵輸入： LAU

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入速度 (In)	×	O	×	×	O	×	×	O	O
參數資料表起始位址 (Prm)	×	×	×	×	×	×	O	×	×
輸出速度 (Out)	×	O*	×	×	O*	×	×	O	×

* C、# 暫存器除外

◆ 整數型 LAU 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	LV	輸入 100% 位準	輸入 100% 的比例	IN
2	W	AT	加速時間	0% ~ 100% 所需之加速時間 (0.1 s)	IN
3	W	BT	減速時間	100% ~ 0% 所需之減速時間 (0.1 s)	IN
4	W	QT	緊急停止時間	100% ~ 0% 所需之緊急停止時間 (0.1 s)	IN
5	W	V	目前速度	LAU 輸出 (輸出為輸出速度)	OUT
6	W	DVDT	目前加減速度	假設一般加速速率為 5000 之比例	OUT
7	W	–	(備用)	備用暫存器	–
8	W	VIM	前次速度指令	用來儲存所輸入的速度指令前次值	OUT
9	W	DVDTK	DVDT 係數	目前加速度 (DVDT) 比例係數	IN
10	L	REM	餘數	加減速速率之餘數	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	RN	運轉中	運轉中輸入 ON	IN
1	QS	緊急停止	緊急停止時輸入 OFF	IN
2	DVDTF	不執行 DVDT 運算	不執行 DVDT 運算且輸入 ON	IN
3	DVDTS	選擇 DVDT 運算	選擇 DVDT 運算方式	IN
4 ~ 7	–	(備用)	輸入用備用繼電器	IN
8	ARY	加速中	加速中輸出 ON	OUT
9	BRY	減速中	減速中輸出 ON	OUT
A	LSP	零速	零速時輸出 ON	OUT
B	EQU	一致	當輸入速度 = 輸出速度時，輸出 ON	OUT
C ~ F	–	(備用)	輸出用備用繼電器	OUT

(註)緊急停止 (QS) 關閉時，使用緊急停止時間做為加減速時間 (QT)。

◆ 實數型 LAU 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *2	IN/OUT
1	W	–	(備用)	備用暫存器	–
2	F	LV*1	輸入 100% 位準	輸入 100% 的比例	IN
4	F	AT	加速時間	0% ~ 100% 所需之加速時間 (s)	IN
6	F	BT	減速時間	100% ~ 0% ま所需之減速時間 (s)	IN
8	F	QT	緊急停止時間	100% ~ 0% 所需之緊急停止時間 (s)	IN
10	F	V	目前速度	LAU 輸出 (輸出為輸出速度)	OUT
12	F	DVDT	目前加減速度	輸出目前的加減速度	OUT

*1. 實際的加減速時間取決於輸入 100% 位準的設定值 (LV) 和輸入速度的比率。
如欲進一步瞭解 LAU 指令內部的詳細運算方式，請參閱以下章節。

☞ **補充事項 (第 4-193 頁)**

*2. 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	RN	運轉中	運轉中輸入 ON	IN
1	QS	緊急停止	緊急停止時輸入 OFF	IN
2 ~ 7	–	(備用)	輸入用備用繼電器	IN
8	ARY	加速中	加速中輸出 ON	OUT
9	BRY	減速中	減速中輸出 ON	OUT
A	LSP	零速	零速時輸出 ON	OUT
B	EQU	一致	當輸入速度 = 輸出速度時，輸出 ON	OUT
C ~ F	–	(備用)	輸出用備用繼電器	OUT

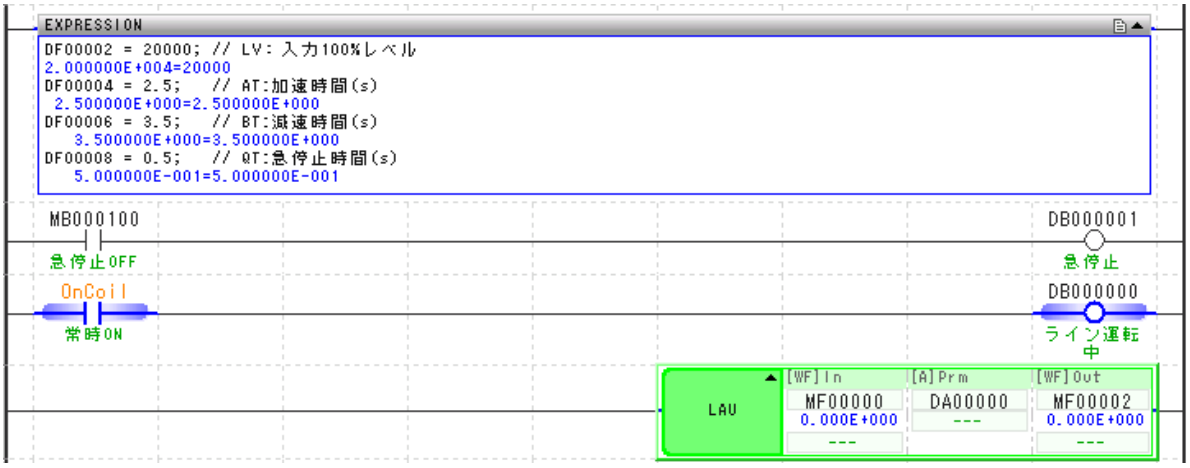
補充　當緊急停止 (QS) 關閉時，加減速時間即為緊急停止時間 (QT)。
若不需要緊急停止，請將緊急停止 (QS) 功能 OFF，同時將輸入速度設定為 0。

程式範例

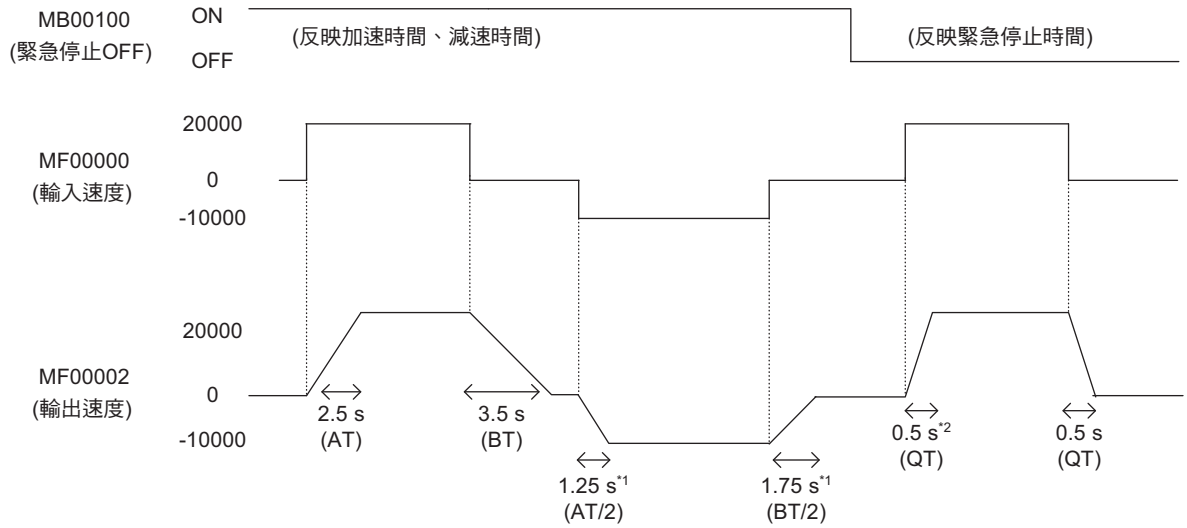
以下係依照您所編寫的加減速速率，並以 MF00000 為輸入速度、MF00002 為輸出速度，執行實數型 LAU 指令之程式範例。

以下參數係用來編寫加減速速率。

- 加減速速率的輸入 100% 位準 = 20000
- 加速時間 = 2.5 s
- 減速時間 = 3.5 s
- 緊急停止時間 = 0.5 s



以下為各暫存器所執行之動作。



*1. 距離 0 較遠的方向為加速時間，反之則為減速時間。
*2. 緊急停止時間亦可用來反映加速時間。

補充事項

◆ 可用來計算速度輸出值及目前加減速度之運算公式

說明計算速度輸出值 (加速、減速、緊急停止時) 及目前加減速度之運算公式。

■ 整數型 LAU 指令

整數型 LAU 指令係根據您所設定的參數，並利用下列運算公式來求出速度輸出值 (加速、減速或緊急停止時) 及目前加減速度。

運算公式中的 V 代表速度輸出值，V' 為前次速度輸出值，VI 為速度指令的輸入值，Ts 則為掃描時間設定值。

· 加速時之速度輸出值

以下運算公式可用來計算加速時之速度輸出值。

當VI > V' (V' ≧ 0)時，V = V' + ADV。
當VI < V' (V' ≦ 0)時，V = V' - ADV。

$$\text{加速速率(ADV)} = \frac{LV \times Ts(0.1ms) + REM}{AT(0.1s) \times 1000}$$

· 減速時之速度輸出值

以下運算公式可用來計算減速時之速度輸出值。

當VI > V' (V' < 0)時，V = V' + BDV。
當VI < V' (V' > 0)時，V = V' - BDV。

$$\text{減速速率(BDV)} = \frac{LV \times Ts\ (0.1\ ms) + REM}{BT\ (0.1\ s) \times 1000}$$

· 緊急停止時之速度輸出值

以下運算公式可用來計算緊急停止時之速度輸出值。

QS = OFF (VI > V'、V' < 0)時、V = V' + QDV。
QS = OFF (VI < V'、V' > 0)時、V = V' - QDV。

$$\text{緊急停止速率(QDV)} = \frac{LV \times Ts\ (0.1\ ms) + REM}{QT\ (0.1\ s) \times 1000}$$

· 目前加減速度

DVDT 運算不執行 (DVDTF) 開啟時，會根據 DVDT 運算選擇 (DVDTS)，以及以下任一個運算公式來求出目前的加減速度 (DVDT)。當 DVDTF 被設定為 OFF 時，DVDT 將變為 0。

$$\text{若DVDTS = ON，則DVDT} = \frac{(V - V') \times 5000}{ADV}$$

$$\text{若DVDTS = ON，則DVDT} = (V - V') \times DVDTK$$

補充

1. 以下為加速中 (ARY) 如何讓繼電器 ON 之條件。
   V' ≧ 0 且 ADV > 0 或是 V' ≦ 0 且 ADV < 0 時
2. 以下為減速中 (BRY) 如何讓繼電器 ON 之條件。
   · V' < 0 且 BDV > 0 或是 V' > 0 且 BDV < 0 時
   · V' < 0 且 QDV > 0 或是 V' > 0 且 QDV < 0 時
3. 零速 (LSP) 的繼電器啟動的條件為 V = 0。
4. 一致 (EQU) 的繼電器啟動的條件為 VI = V。
5. 當 RUN 運轉中 (RN) 被設定為關閉時，將輸出 V = 0、DVDT = 0、REM = 0。

■ 實數型 LAU 指令

實數型 LAU 指令可根據您所設定的參數，從以下運算公式，來求出速度輸出值 (加速、減速、緊急停止時) 及目前加減速度。

運算公式中的 V 代表速度輸出值，V' 為前次速度輸出值，VI 為速度指令的輸入值，Ts 則為掃描時間設定值。

· **加速時之速度輸出值**
以下運算公式可用來計算加速時之速度輸出值。

當VI > V' (V' ≧ 0)時，V = V' + ADV。
當VI < V' (V' ≦ 0)時，V = V' - ADV。

$$\text{加速速率(ADV)} = \frac{\text{LV}\times\text{Ts (0.1 ms)}}{\text{AT (s)}\times 10000}$$

· **減速時之速度輸出值**
以下運算公式可用來計算減速時之速度輸出值。

當VI < V' (V' > 0)時，V = V' + BDV。
當VI > V' (V' < 0)時，V = V' - BDV。

$$\text{減速速率(BDV)} = \frac{-\text{LV}\times\text{Ts (0.1 ms)}}{\text{BT(s)}\times 10000}$$

· **緊急停止時之速度輸出值**
以下運算公式可用來計算緊急停止時之速度輸出值。

QS = OFF (VI< V'、V' > 0)時、V = V' + QDV。
QS = OFF (VI > V'、V' < 0)時、V = V' - QDV。

$$\text{緊急停止速率(QDV)} = \frac{-\text{LV}\times\text{Ts (0.1 ms)}}{\text{QT (s)}\times 10000}$$

· **目前加減速度**
會先運算出速度輸出值 (V)，然後再利用以下運算公式來求出目前加減速度 (DVDT)。
DVDT = V – V'

補充

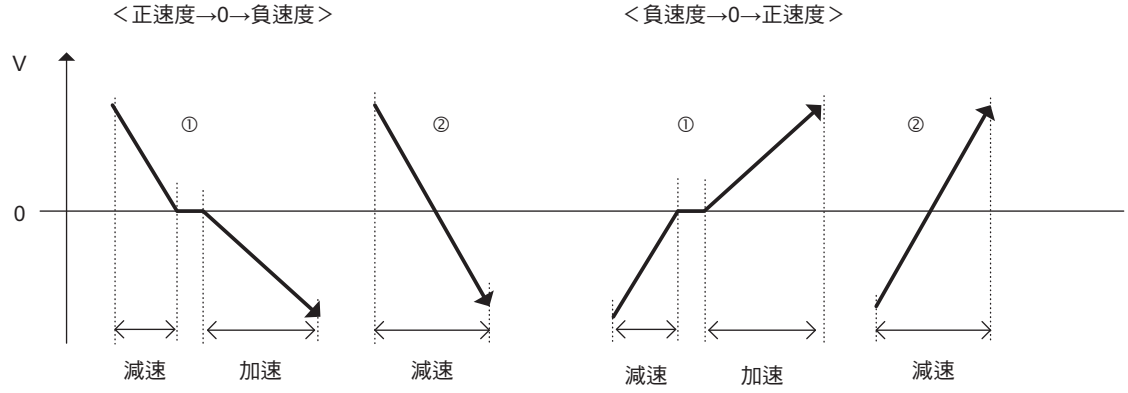
1. 以下為加速中 (ARY) 如何讓繼電器啟動之條件。
   V' ≧ 0 且 ADV > 0 或是 V' ≦ 0 且 ADV < 0 時
2. 以下為減速中 (BRY) 如何讓繼電器 ON 之條件。
   · V' < 0 且 BDV > 0 或是 V' > 0 且 BDV < 0 時
   · V' < 0 且 QDV > 0 或是 V' > 0 且 QDV < 0 時
3. 零速 (LSP) 的繼電器啟動的條件為 V = 0。
4. 一致 (EQR) 的繼電器啟動的條件為 VI = V。
5. 以下為加速中 (ARY) 如何讓繼電器 ON 之條件。
   V ≠ V' 且 DVDT 和 V 的符號相同
6. 以下為減速中 (BRY) 如何讓繼電器 ON 之條件。
   V ≠ V' 且 DVDT 和 V 的符號不同
7. 當 RUN 運轉中 (RN) 被設定為 OFF，將輸出 V = 0、DVDT = 0。

◆ 變更輸入速度時的加減速

說明變更輸入速度時的加減速。

當速度 = 0 且暫停動作時，會根據您所設定的減速時間及加速時間來進行加減速。(**參閱** ①)

當指令跨過速度 = 0 時，將依照減速時間來進行加減速，以避免造成速度改變。(**參閱** ②)

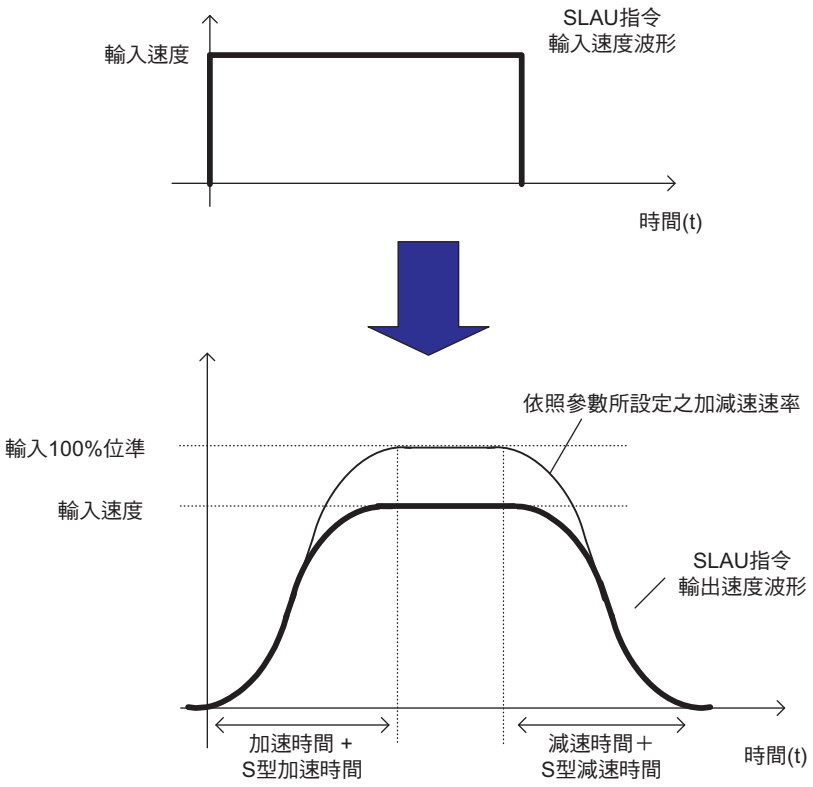


直線加減速器 2 (SLAU)

可依照輸入速度，輸出依照可變的加減速速率，加速 / 減速後之結果。加減速時係依照您所設定的參數資料表內容，並採用 S 型曲線方式。

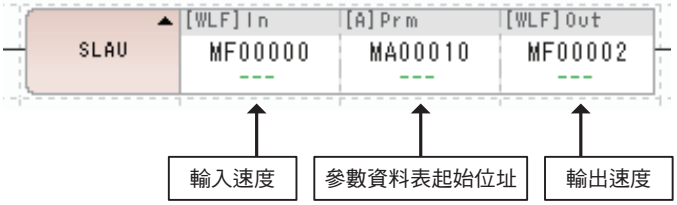
整數型、長整數型或實數型資料皆適合當作 SLAU 運算之輸入值。4 長整數型及倍精度實數型資料不適用。

參數資料表的結構依整數型或實數型等資料類型而異。



格式

所採用之格式如下。



圖示 ：

按鍵輸入： SLAU

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入速度 (In)	×	○	○	×	○	×	×	○	○
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*	○*	×
輸出速度 (Out)	×	○*	○*	×	○*	×	×	○	×

* C、# 暫存器除外

◆ 整數型 SLAU 指令參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	LV	輸入 100% 位準	輸入 100% 的比例	IN
2	W	AT	加速時間	0% ~ 100% 所需之加速時間 (0.1 s)	IN
3	W	BT	減速時間	100% ~ 0% 所需之減速時間 (0.1 s)	IN
4	W	QT	緊急停止時間	100% ~ 0% 所需之緊急停止時間 (0.1 s)	IN
5	W	AAT	加速 S 型時間	加速時之 S 型區域時間 (0.01 ~ 32.00 s)	IN
6	W	BBT	減速 S 型時間	減速時之 S 型區域時間 (0.01 ~ 32.00 s)	IN
7	W	V	目前速度	SLAU 輸出 (輸出為輸出速度)	OUT
8	W	DVDT1	目前加減速度 1	假設一般加速速率為 5000 之比例	OUT
9	W	–	(備用)	備用暫存器	–
10	W	ABMD	維持狀態下速度上升	維持指令穩定前之速度變化量	OUT
11	W	REM1	餘數	加減速速率之餘數	OUT
12	W	–	(備用)	備用暫存器	–
13	W	VIM	前次速度指令	儲存所輸入的速度指令前次值	OUT
14	L	DVDT2	目前加減速度 2	將實際的加減速度乘以 1000 倍	OUT
16	L	DVDT3	目前加減速度 3	目前加減速度 (= DVDT2/1000)	OUT
18	L	REM2	餘數	S 型區間加減速速率之餘數	OUT
20	W	REM3	餘數	目前速度之餘數	OUT
21	W	DVDTK	DVDT1 係數	目前加速度 1 (DVDT) 之比例係數 (-32768 ~ 32767)	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	RN	運轉中	運轉中輸入 ON	IN
1	QS	緊急停止	緊急停止時輸入 OFF	IN
2	DVDTF	不執行 DVDT1 運算	不執行 DVDT 運算且輸入 ON	IN
3	DVDTS	DVDT1 運算選擇	選擇 DVDT 運算方式	IN
4 ~ 7	–	(備用)	輸入用備用繼電器	IN
8	ARY	加速中	加速中輸出 ON	OUT
9	BRY	減速中	減速中輸出 ON	OUT
A	LSP	零速	零速時輸出 ON	OUT
B	EQU	一致	當輸入速度 = 輸出速度時，輸出 ON	OUT
C	–	(備用)	輸出用備用繼電器	OUT
D	CCF	工作繼電器	系統內部工作繼電器	OUT
E	BBF	工作繼電器	系統內部工作繼電器	OUT
F	AAF	工作繼電器	系統內部工作繼電器	OUT

(註)緊急停止 (QS) OFF 時，會使用緊急停止時間做為減速時間 (QT)。

◆ 長整數型 SLAU 指令之參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *2	IN/OUT
1	W	–	(備用)	–	–
2	L	LV	輸入 100% 位準	輸入值 100% 之比例	IN
4	L	AT	加速時間	0% ~ 100% 所需之加速時間 (0.1 s)	IN
6	L	BT	減速時間	100% ~ 0% 所需之減速時間 (0.1 s)	IN
8	L	QT	緊急停止時間	100% ~ 0% 所需之緊急停止時間 (0.1 s)	IN
10	L	AAT	加速 S 型時間	加速時的 S 型區域時間 (0.01 s)	IN
12	L	BBT	減速 S 型時間	減速時的 S 型區域時間 (0.01 s)	IN
14	L	V	目前速度	SLAU 輸出 (亦會輸出至 A 暫存器)	OUT
16	L	DVDT	目前加減速度	輸出目前的加減速度 (小數點以下無條件捨去)	OUT
18	L	ABMD	維持狀態下速度上升	維持指令穩定前之速度變化量	OUT
20	D*1	V_D	目前速度	系統專用 SLAU 輸出 (倍精度實數)	IN/OUT
24	D*1	DVDT_D	目前加減速度	系統專用目前加減速度 (倍精度實數)	IN/OUT

*1. D 為倍精度實數，包含 4 個字元。MPE720 無法用來表示實數值。
*2. 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	RN	運轉中	運轉中輸入 ON	IN
1	QS	緊急停止	緊急停止時輸入 OFF	IN
2	DVDTF	加減速旗標	在 ON 輸入狀態下，利用 1000 倍的數值來輸出目前加減速度 (DVDT)	IN
3 ~ 7	–	(備用)	輸入用備用繼電器	IN
8	ARY	加速中	加速中輸出 ON	OUT
9	BRY	減速中	減速中輸出 ON	OUT
A	LSP	零速	零速時輸出 ON	OUT
B	EQU	一致	當輸入值 = 輸出值時，輸出 ON	OUT
C ~ F	–	工作繼電器	系統內部工作繼電器	IN/OUT

(註)緊急停止 (QS) 關閉時，會使用緊急停止時間做為減速時間 (QT)。

◆ 實數型 SLAU 指令參數資料表

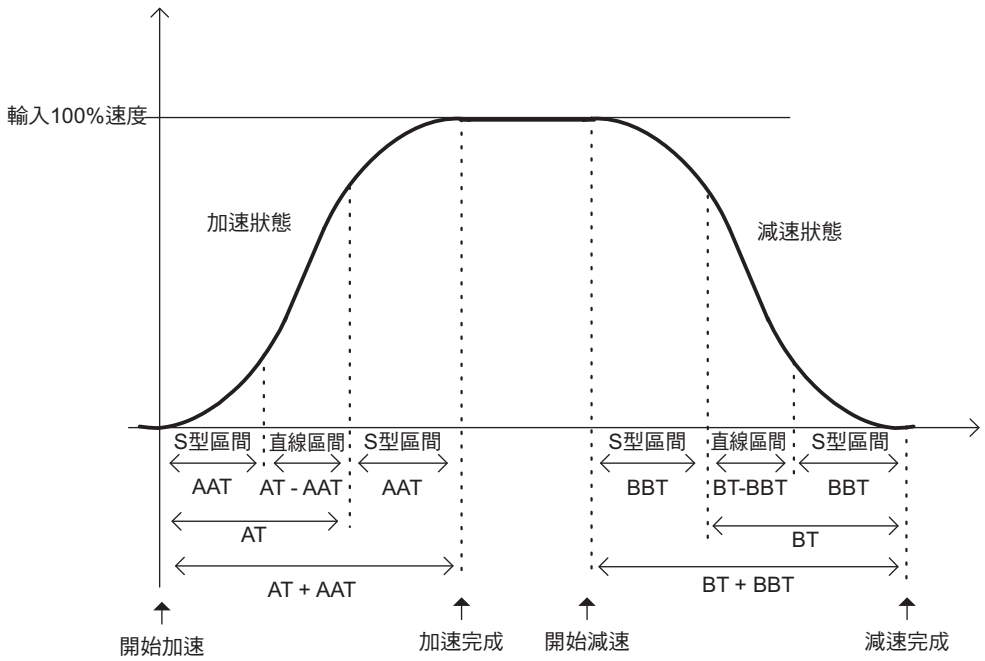
位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *	IN/OUT
1	W	–	(備用)	備用暫存器	–
2	F	LV	輸入 100% 位準	輸入 100% 的比例	IN
4	F	AT	加速時間	0% ~ 100% 所需之加速時間 (s)	IN
6	F	BT	減速時間	100% ~ 0% 所需之減速時間 (s)	IN
8	F	QT	緊急停止時間	100% ~ 0% 所需之緊急停止時間 (s)	IN
10	F	AAT	加速 S 型時間	加速時的 S 型區域時間 (s)	IN
12	F	BBT	減速 S 型時間	減速時的 S 型區域時間 (s)	IN
14	F	V	目前速度	SLAU 輸出 (輸出為輸出速度)	OUT
16	F	DVDT1	現在加減速度 1	輸出實際的加減速度	OUT
18	F	ABMD	維持狀態下速度上升	維持指令穩定前之速度變化量	OUT

* 以下為繼電器的輸出入配置表。

Bit	符號	名稱	規格	輸出入
0	RN	運轉中	運轉中輸入 ON	IN
1	QS	緊急停止	緊急停止時輸入 OFF	IN
2 ~ 7	–	(備用)	輸入用備用繼電器	IN
8	ARY	加速中	加速中輸出 ON	OUT
9	BRY	減速中	減速中輸出 ON	OUT
A	LSP	零速	零速時輸出 ON	OUT
B	EQU	一致	當輸入速度 = 輸出速度時，輸出 ON	OUT
C ~ F	–	(備用)	輸出用備用繼電器	OUT

(註)緊急停止 (QS) 關閉時，會使用緊急停止時間做為減速時間 (QT)。

範例 下圖所示為參數被運用在實際指令上的方式。



(註)如欲進一步瞭解 SLAU 指令內部的詳細運算方式，請參閱以下章節。

☞ **補充事項 (第 4-201 頁)**

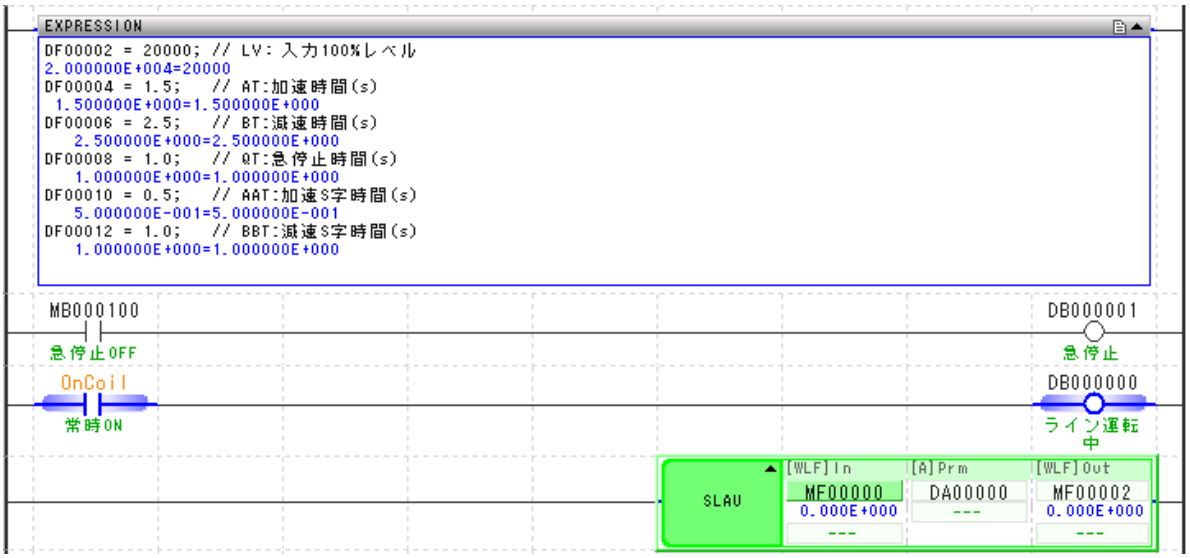
補充 當緊急停止 (QS) 被設定為 OFF 時，就會在緊急停止時間 (QT) 減速，此時輸出速度為 0。不需要先將輸入速度設定為 0。
此外，緊急停止時，將採直線方式減速，而非 S 型動作。設定時，請遵守直線加減速時間 (AT/BT) ≧ 加減速 S 型時間 (AAT/BBT) 的原則。

程式範例

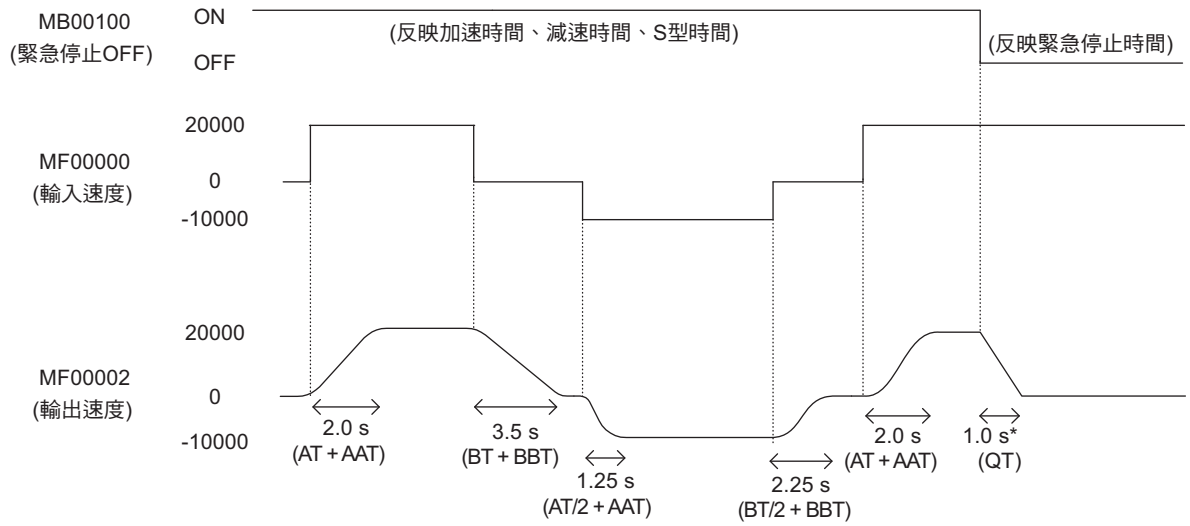
以下係依照您所編寫的加減速速率，並以 MF00000 為輸入速度、MF00002 為輸出速度，執行實數型 SLAU 指令之程式範例。

以下參數係用來編寫加減速速率。

- 加減速速率的輸入 100% 位準之速度 = 20000
- 加速時間 = 1.5 s
- 減速時間 = 2.5 s
- 緊急停止時間 = 0.5 s
- 加速 S 型時間 = 0.5 s
- 減速 S 型時間 = 1.0 s



以下為各暫存器所執行之動作。



* 當緊急停止訊號輸入時，無論 S 型時間、輸入速度為何，皆會依照緊急停止時間減速停止動作。

補充事項

◆ 計算速度輸出值及目前加減速度之運算公式

說明用來求出速度輸出值 (加速、減速、緊急停止、S 型區間加速、S 型區間減速時) 及目前加減速度之運算公式。

■ 整數型 SLAU 指令運算

整數型 SLAU 指令會根據您所設定的參數表，從以下運算公式，來求出速度輸出值 (加速、減速、緊急停止、S 型區間加速、S 型區間減速時) 及目前加減速度。

運算公式中的 V 代表速度輸出值，V' 為前次速度輸出值，VI 為速度指令的輸入值，Ts 則為掃描時間設定值。

· **加速時之速度輸出值**
以下運算公式可用來計算加速時之速度輸出值。

S型區間外(ADVS > ADV)且VI > V' (V' ≧ 0)時，即為V = V' + ADV。
S型區間外(ADVS > ADV)且VI < V' (V' ≦ 0)時，即為V = V' - ADV。

$$\text{加速速率(ADV)} = \frac{\text{LV}\times\text{Ts (0.1 ms)} + \text{REM1}}{\text{AT (0.1 s)}\times 1000}$$

· **減速時之速度輸出值**
以下運算公式可用來計算減速時之速度輸出值。

S型區間外(BDVS > BDV)且VI > V' (V' < 0)時，即為V = V' + BDV。
S型區間外(BDVS > BDV)且VI < V' (V' > 0)時，即為V = V' - BDV。

$$\text{減速速率(BDV)} = \frac{\text{LV}\times\text{Ts (0.1ms)} + \text{REM1}}{\text{BT (0.1 s)}\times 1000}$$

· **緊急停止時之速度輸出值**
以下運算公式可用來計算緊急停止時之速度輸出值。

QS = OFF (VI > V'、V' < 0)時、V = V' + QDV。
QS = OFF (VI < V'、V' > 0)時、V = V' - QDV。

$$\text{緊急停止速率(QDV)} = \frac{\text{LV}\times\text{Ts (0.1ms)} + \text{REM1}}{\text{QT (0.1 s)}\times 1000}$$

補充　緊急停止時，將採直線動作方式減速，而非 S 型動作。

· **S 型區間加速時之速度輸出值**
以下運算公式係用來計算 S 型區間加速時之速度輸出值。

S型區間內(ADVS < ADV)且VI > V' (V' ≧ 0)時，即為V = V' + ADVS。
S型區間內(ADVS < ADV)且VI < V' (V' ≦ 0)時，即為V = V' - ADVS。

S型區間加速速率(ADVS) = ADVS'± AADVS

$$\text{AADVS} = \frac{\text{ADV}\times\text{Ts (0.1 ms)} + \text{REM2}}{\text{AAT (0.01 s)}\times 100}$$

· **S 型區間減速時之速度輸出值**

以下運算公式係用來計算 S 型區間減速時之速度輸出值。

S型區間內(BDVS < BDV)且VI > V' (V' < 0)時，即為V = V' + BDVS。
S型區間內(BDVS < BDV)且VI < V' (V' > 0)時，即為V = V' - BDVS。

S型區間減速速率(BDVS) = BDVS'± BBDVS

$$BBDVS = \frac{BDV \times Ts\ (0.1ms) + REM2}{BBT\ (0.01\ s) \times 100}$$

· **目前加減速度**

當 DVDT1 運算不執行功能 (DVDTF) 開啟時，會根據 DVDT1 運算選擇 (DVDTS) 和以下任一個運算公式，來求出目前加減速度 1 (DVDT1)。當 DVDTF 被設定為關閉時，DVDT1 將變為為 0。

若DVDTS = ON，則$DVDT1 = \frac{(V - V') \times 5000}{ADV}$

若DVDTS = OFF，則DVDT1 = (V – V')×DVDTK

以下為目前加減速度 2 (DVDT2)。

加速中：S 型區間內：DVDT2 = ±ADVS
　　S 型區間外：DVDT2 = ±ADV
減速中：S 型區間內：DVDT2 = ±BDVS
　　S 型區間外：DVDT2 = ±BDV
緊急停止狀態：DVDT = ± QDV

會在完成以下運算後，輸出維持狀態下速度上升值 (ABMD)。

$$ABMD = \frac{DVDT2' \times DVDT2'}{2 \times AADVS\ (BBDVS)}$$

DVDT2' :目前加減速度2 (DVDT2)之前次值

補充

1. 以下為加速中 (ARY) 如何讓繼電器 ON 之條件。
   - V' >= 0 且 ADV > 0 或是 V' ≦ 0 且 ADV < 0 時
   - S 型區間且 V' ≧ 0、ADVS > 0，或 S 型區間且 V' ≦ 0、ADVS < 0 時
2. 以下為減速中 (BRY) 如何讓繼電器開啟之條件。
   - V' < 0 且 BDV > 0 或是 V' > 0 且 BDV < 0 時
   - V' < 0 且 QDV > 0 或是 V' > 0 且 QDV < 0 時
   - S 型區間 V' < 0 且 BDVS > 0，S 型區間 V' > 0 且 BDVS < 0 時
3. 零速 (LSP) 的繼電器啟動的條件為 V = 0。
4. 一致 (ECQ) 的繼電器啟動的條件為 VI = V。
5. 當 RUN 運轉中 (RN) 被設定為關閉，將輸出 V = 0、DVDT1、DVDT2、DVDT3 = 0、REM1、REM2、REM3 = 0。

■ 長整數型 / 實數型 SLAU 指令運算

長整數型 / 實數型 SLAU 指令會根據以下運算公式，來求出速度輸出值 (加速、減速、緊急停止、S 型區間加速、S 型區間減速時) 及目前加減速度。

運算公式中的 V 代表速度輸出值，V' 為前次速度輸出值，VI 為四度指令輸入值，Ts 為掃瞄時間設定值，ADVS' 為 ADVS 前次值，BDVS' 則為 BDVS 前次值。

· 加速時之速度輸出值

以下運算公式可用來計算加速時之速度輸出值。

S型區間外(ADVS > ADV)且VI > V' (V' ≧ 0)時，即為V = V' + ADV。
S型區間外(ADVS > ADV)且VI < V' (V' ≦ 0)時，即為V = V' - ADV。

$$\text{加速速率(ADV)} = \frac{\text{LV}\times\text{Ts (0.1ms)}}{\text{AT (s)}\times 10000}$$

· 減速時之速度輸出值

以下運算公式可用來計算減速時之速度輸出值。

S型區間外(BDVS > BDV)且VI > V' (V' < 0)時，即為V = V' + BDV。
S型區間外(BDVS > BDV)且VI < V'(V' > 0)時，即為V = V' - BDV。

$$\text{減速速率(BDV)} = \frac{-\text{LV}\times\text{Ts (0.1ms)}}{\text{BT(s)}\times 10000}$$

· 緊急停止時之速度輸出值

以下運算公式可用來計算緊急停止時之速度輸出值。

QS = OFF (VI > V'、V' < 0)時、V = V' + QDV。
QS = OFF (VI < V'、V' > 0)時、V = V' - QDV。

$$\text{緊急停止速率(QDV)} = \frac{-\text{LV}\times\text{Ts (0.1ms)}}{\text{QT (s)}\times 10000}$$

補充　緊急停止時，將採直線動作方式減速，而非 S 型動作。

· S 型區間加速時之速度輸出值

以下運算公式係用來計算 S 型區間加速時之速度輸出值。

S型區間內(ADVS < ADV)且VI > V' (V' ≧ 0)時，即為V = V' + ADVS。
S型區間內(ADVS < ADV)且VI < V' (V' ≦ 0)時，即為V = V' - ADVS。

S型區間加速速率(ADVS) = ADVS' ± AADVS

$$\text{AADVS} = \frac{\text{ADV}\times\text{Ts (0.1ms)}}{\text{AAT (s)}\times 10000}$$

· S 型區間減速時之速度輸出值

以下運算公式係用來計算 S 型區間減速時之速度輸出值。

S型區間內(BDVS < BDV)且VI > V' (V' < 0)時，即為V = V' + BDVS。
S型區間內(BDVS < BDV)且VI < V' (V' > 0)時，即為V = V' - BDVS。

S型區間減速速率(BDVS) = BDVS'± BBDVS

$$\text{BBDVS} = \frac{\text{BDV}\times\text{Ts (0.1ms)}}{\text{BBT (s)}\times 10000}$$

· **目前加減速度**
會先進行以下運算後，再輸出目前加減速度 1 (DVDT1)。

加速中：S 型區間內：DVDT = ADVS
　　S 型區間外：DVDT = ADV
減速中：S 型區間內：DVDT = BDVS
　　S 型區間外：DVDT = BDV
緊急停止狀態：DVDT = QDV

會先進行以下運算後，再輸出維持狀態下速度上升值 (ABMD)。

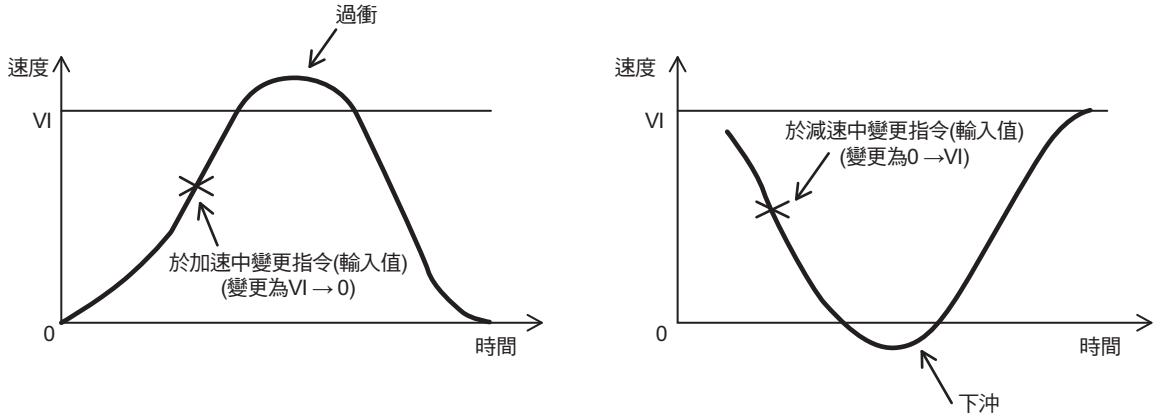
$$ABMD = \frac{DVDT \times DVDT}{2 \times AADVS(BBDVS)}$$

補充
1. V = 0 為零速 (LSP) 的繼電器 ON 之條件。
2. VI = V 為一致 (EQU) 的繼電器 ON 之條件。
3. 當 RUN 運轉狀態 (RN) 被設定為 OFF，將輸出 V = 0、DVDT = 0、AVMD = 0。

◆ 使用 SLAU 指令時之注意事項

■ 如欲在加減速中 [到達輸入速度 (VI) 前] 變更輸入值

若要在加速減速中變更輸入值 (VI)，請勿使用整數型 SLAU 指令。否則有可能會發生下圖所示的過衝及下沖等現象。

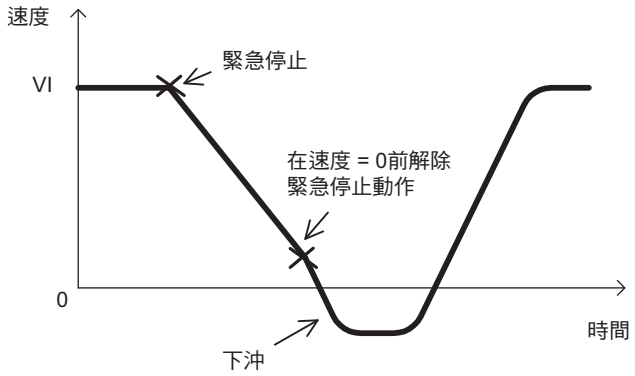


如欲在加減速中變更輸入值 (VI)，請使用應用程式，並執行以下任一種對策。

· 使用實數型 SLAU 指令。
· 同時使用整數型 SLAU 指令及 LIMIT 指令。也就是，以 LIMIT 指令的輸入值作為整數型 SLAU 指令的輸出值，以達到限制過衝 / 下沖現象之效果。

■ 想要在緊急停止及減速中，解除緊急停止動作時

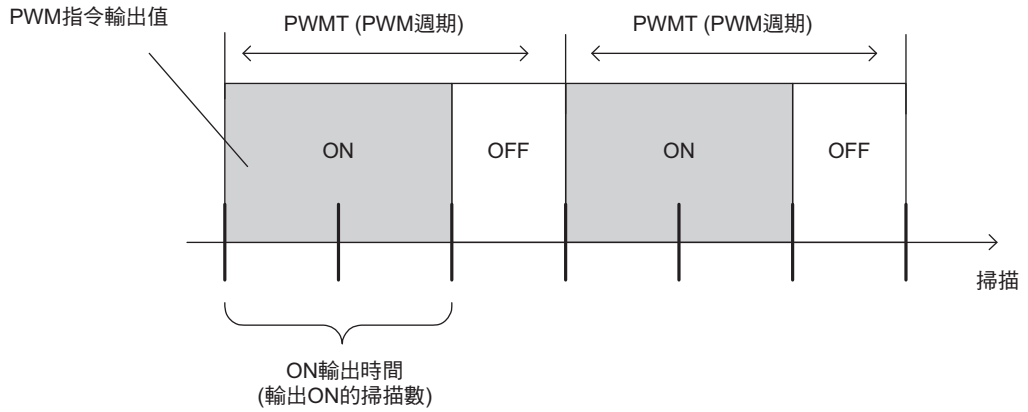
在緊急停止及減速中，請避免在輸出速度變為 0 之前，解除緊急停止動作。將有可能在趨近輸入速度時，導致下沖發生。



若不得以必須在輸出速度到達 0 之前解除緊急停止動作，必須對輸出速度使用 LIMIT 指令，以避免發生下沖現象。

脈衝幅度調變 (PWM)

將輸入值 (-100.00 ~ 100.00%) 轉換為 PMW，並以該結果輸出至輸出值和參數資料表中。輸入值僅適用整數型，而輸出值僅限使用位元型。長整數型及實數型不適用。



以下公式可用來求出 PWM 指令的 ON 輸出時間及 ON 輸出掃描數。

X 代表輸入值，PWMT 為 PWM 週期 (ms)，Ts 則為掃描時間設定值 (ms)。

$$\text{ON輸出時間} = \frac{\text{PWMT}(X+10000)}{20000}$$

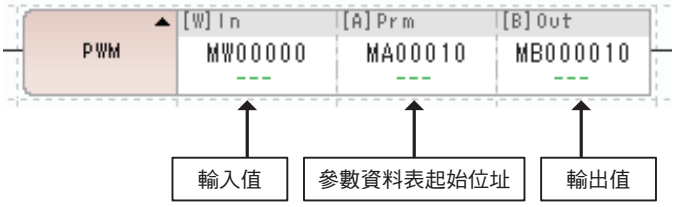
$$\text{ON輸出掃描數} = \frac{\text{PWMT}(X+10000)}{Ts \times 20000}$$

註記

1. 以下為輸入值與 PWM 輸出 ON 比率之間的關係。
   - 輸入值 100.00% → 100% ON (ON 輸出時間 = PWMT)
   - 輸入值 0.00% → 50% ON (ON 輸出時間 = PWMT/2)
   - 輸入值 -100.00% → 0% ON (ON 輸出時間 = 0)
2. 輸入電源後，請先將 PWM 重置 (PWMRST) 功能 ON，並清除內部運算資料後，再使用 PWM 指令。將以 PWM 重置時間點為起始點，進行 PMW 運算。

格式

所採用之格式如下。



圖示　：PWM
按鍵輸入：PWM

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸入值 (In)	×	○	×	×	×	×	×	○	○
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*	○*	×
輸出值 (Out)	○*	×	×	×	×	×	×	○	×

* C、# 暫存器除外

◆ 輸入值 / 輸出值範圍

輸入值範圍為 -10000 ~ 10000 (單位：0.01%)。

若超出範圍，將以 1000 為上限值，-1000 為下限值處理。

當 PWM 輸出被設定為 ON 時，輸出值為 1，反之若被設定為 OFF 時則輸出 0。

◆ 參數資料表

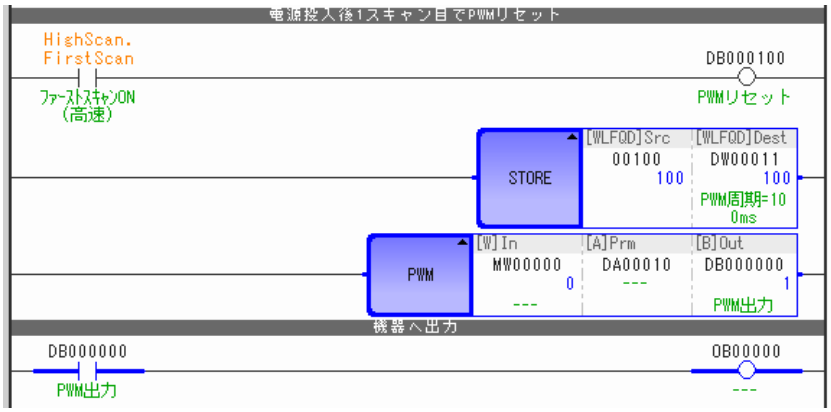
位址	資料類型	符號	名稱	規格	輸出入
0	W	RLY	繼電器輸出入	繼電器輸入、繼電器輸出 *1	IN/OUT
1	W	RWMT	PWM 週期	PWM 週期 (1 ms) 範圍：1 ~ 32767 ms	IN
2	W	ONCNT	ON 輸出設定計時器	ON 輸出的設定計時器 (1 ms)	OUT
3	W	CVON	ON 輸出計數計時器	ON 輸出的計數計時器 (1 ms)	OUT
4	W	CVONREM	ON 輸出計數計時器之餘數	ON 輸出的計數計時器之餘數 (0.1 ms)	OUT
5	W	OFFCNT	OFF 輸出設定計時器	OFF 輸出的設定計時器 (1 ms)	OUT
6	W	CVOFF	OFF 輸出計數計時器	OFF 輸出的計數計時器 (1 ms)	OUT
7	W	CVOFFREM	OFF 輸出計數計數器之餘數	OFF 輸出的計數計時器之餘數 (0.1 ms)	OUT

* 以下為繼電器的輸出入配置表。

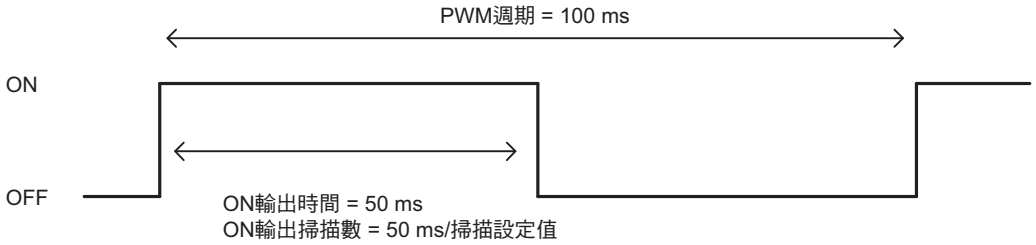
Bit	符號	名稱	規格	輸出入
0	PWMRST	PWM 重置	PWM 重置時輸入 ON	IN
2 ~ 7	–	(備用)	輸入用備用繼電器	IN
8	PWMOUT	PWM 輸出	PWM 輸出 (若輸出被設定為 ON 時，輸出值為 1，OFF 時則為 0)	OUT
9 ~ F	–	(備用)	輸出用備用繼電器	OUT

程式範例

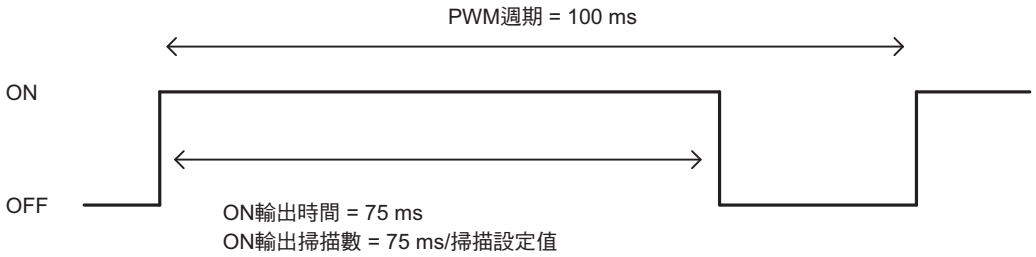
以下為 PWM 週期為 100 ms 時，輸入值 (MW00000) 所對應之 PWM 輸出被儲存於 OB000000 之程式範例。



假設 MW00000 為 0 (0%：ON 輸出時間 = PWM 週期 /2) 時的 OB000000 將如下圖所示。



假設 MW00000 為 5000 (50%：ON 輸出時間 = PWM 週期 x 3/4) 時的 OB000000 將如下圖所示。



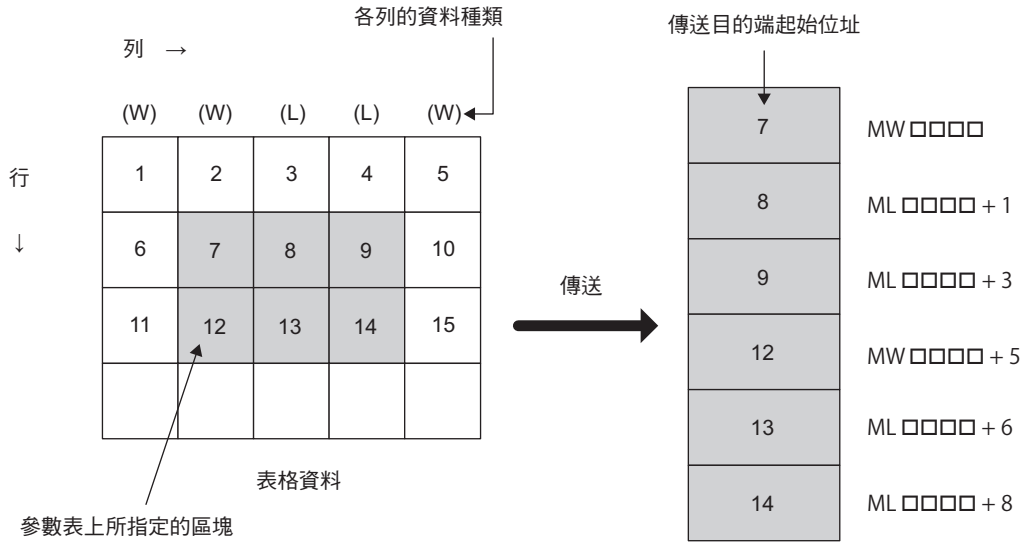
4.9 資料表操控指令

讀取區塊 (TBLBR)

將表格名稱、行編號、列編號所指定的表格資料區塊傳送至從傳送目的端起始位址開始的連續區域。依照所讀取的元素資料種類，儲存在傳送目的端的區域。

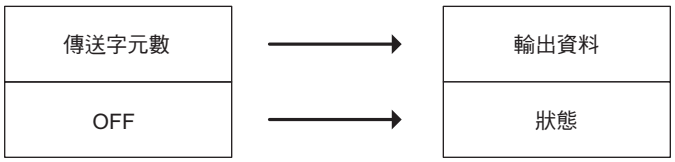
參照表格時，若有範圍以外、傳送目的端資料長度不足等不正常的情況，將回報錯誤，資料不會被讀取，因此仍保持傳送目的端區域的內容。

正常結束時會輸出傳送字元數，並且將狀態關閉。錯誤發生時會輸出錯誤碼，並且將狀態啟動。

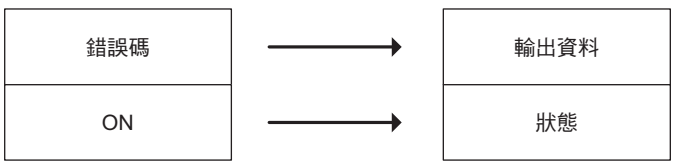


配合表格資料的資料種類來儲存。

· 傳送成功時



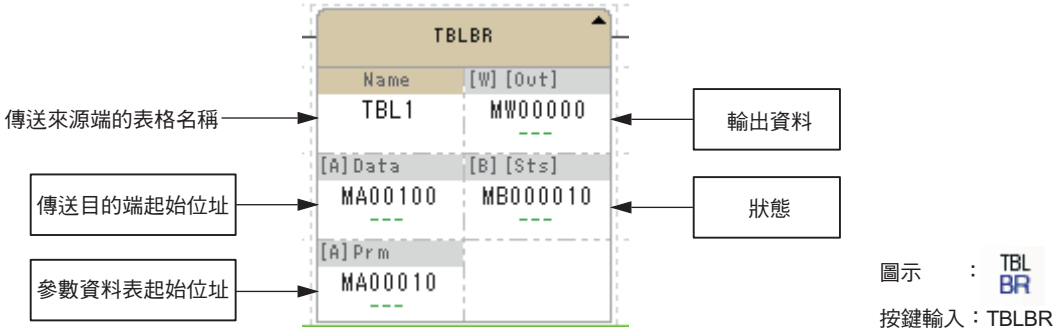
· 傳送失敗時



補充　傳送失敗時，傳送目的端區域會保持執行指令前的狀態。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送目的端起始位址 (Data)	×	×	×	×	×	×	○*2	×	×
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○	×	×
輸出資料 (Out) *1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts) *1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW1	表格元素的起始行編號	作為目標的表元素之起始行編號 (1 ~ 65535)	IN
2	L	COL1	表格元素的起始列編號	作為目標的表元素之起始列編號 (1 ~ 32767)	IN
4	W	RLEN	行元素的數量	行元素的數量 (1 ~ 32767)	IN
5	W	CLEN	列元素的數量	列元素的數量 (1 ~ 32767)	IN

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

程式範例

以下所示為啟動開關 1 (DB00000) 後，將記錄型表格資料 TBL1 的指定區塊傳送至從參數表起始位址 (MW00100) 開始的連續區域之程式範例。

參數表的設定如以下所示。

暫存器	資料	備註
DL00000	2	表格元素的起始行編號
DL00002	2	表格元素的起始列編號
DW00004	3	行元素的數量
DW00005	3	列元素的數量

表格資料 TBL1 的內容如下所示。

行	列				
	1 (W)*	2 (W)*	3 (L)*	4 (L)*	5 (F)*
1	1000	1001	10000	10001	1.1
2	2000	2002	20000	20002	1.2
3	3000	3003	30000	30003	1.3
4	4000	4004	40000	40004	1.4
5	5000	5005	50000	50005	1.5

傳送區域

* 顯示資料類型。

執行指令後，如以下所示資料將被傳送至 MW00100 開始的連續區域。

此外，輸出資料 (MW00000) 為 15 (字元數)，狀態 (MB000010) 為 0 (傳送成功)。

暫存器	資料	暫存器	資料	暫存器	資料
MW00100	2002	ML00101	20000	ML00103	20002
MW00105	3003	ML00106	30000	ML00108	30003
MW00110	4004	ML00111	40000	ML00113	40004

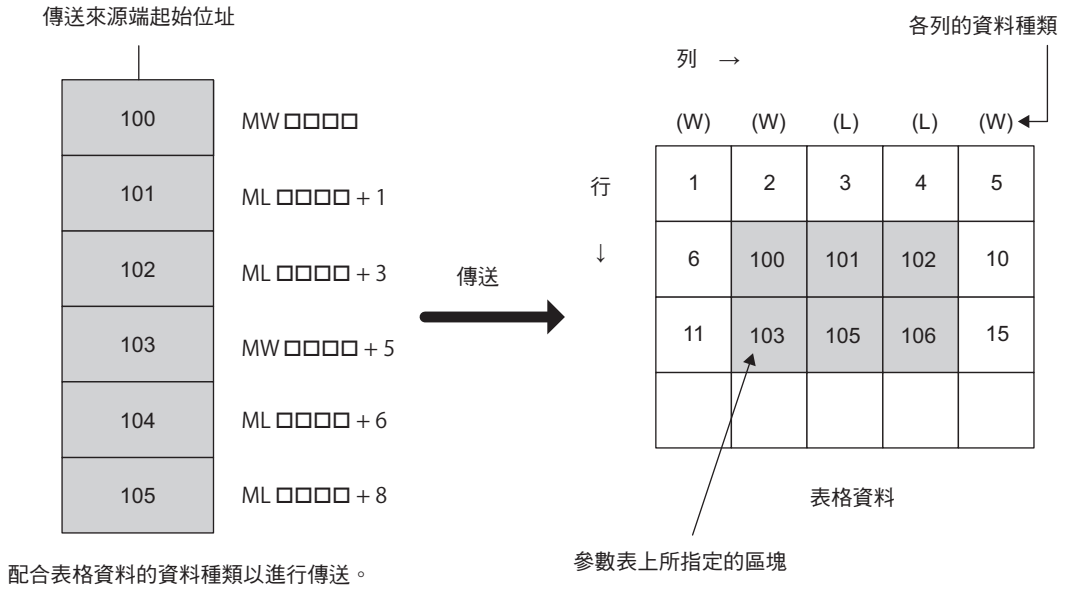
(註)暫存器的配置同上表。

寫入區塊 (TBLBW)

將從傳送來源端起始位址開始的連續區域的資料傳送至表格名稱、行編號、列編號所指定的表格資料區塊。傳送來源區的資料類型與表格資料的各元素資料類型將被視為一致並進行儲存。

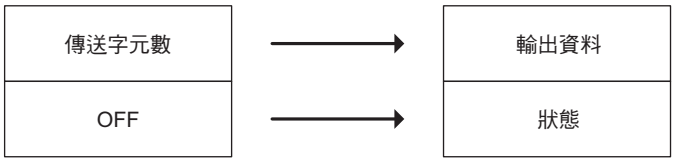
參照表格時，若有範圍以外、傳送來源端資料長度不足等不正常的情況時，將會回報錯誤，資料不會被寫入，傳送目的端表格資料的內容將會保持不變。

正常結束時會輸出傳送字元數，並且將狀態關閉。錯誤發生時會輸出錯誤碼，並且將狀態啟動。

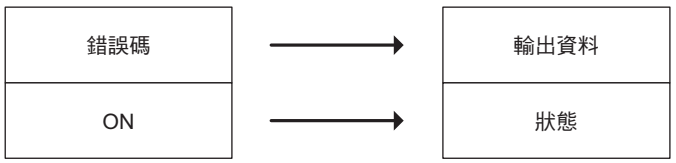


配合表格資料的資料種類以進行傳送。

· 傳送成功時



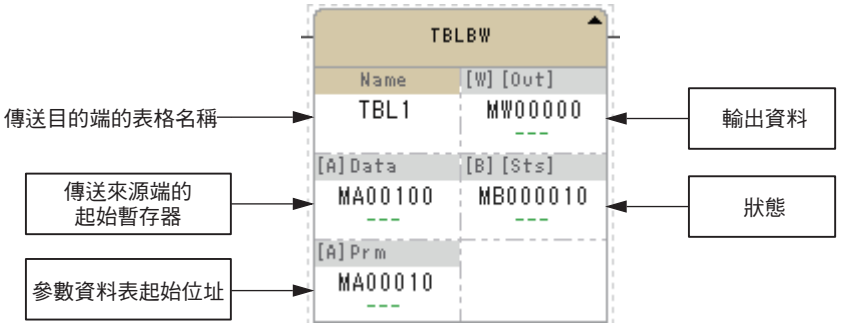
· 傳送失敗時



補充　傳送失敗時，傳送目的端的表格資料將保持執行指令前的狀態。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送來源端起始位址 (Data)	×	×	×	×	×	×	○*2	×	×
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○	×	×
輸出資料 (Out) *1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts) *1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW1	表格元素的起始行編號	作為目標的表元素之起始行編號 (1 ~ 65535)	IN
2	L	COL1	表格元素的起始列編號	作為目標的表元素之起始列編號 (1 ~ 32767)	IN
4	W	RLEN	行元素的數量	行元素的數量 (1 ~ 32767)	IN
5	W	CLEN	列元素的數量	列元素的數量 (1 ~ 32767)	IN

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

程式範例

以下所示為啟動開關 1 (DB00000) 後，將從參數表起始位址 (MW00100) 開始的連續區域的資料，傳送至記錄型表格資料 TBL1 的指定區塊之程式範例。

參數表的設定如以下所示。

暫存器	資料	備註
DL00000	2	表格元素的起始行編號
DL00002	2	表格元素的起始列編號
DW00004	3	行元素的數量
DW00005	3	列元素的數量

傳送的資料如以下所示。

暫存器	資料	暫存器	資料	暫存器	資料
MW00100	1	ML00101	2	ML00103	3
MW00105	4	ML00106	5	ML00108	6
MW00110	7	ML00111	8	ML00113	9

執行指令後表格資料 TBL1 的內容如下。

此外，輸出資料 (MW00000) 為 15 (字元數)，狀態 (MB000010) 為 0 (傳送成功)。

行	列				
	1 (W)*	2 (W)*	3 (L)*	4 (L)*	5 (F)*
1					
2		1	2	3	
3		4	5	6	
4		7	8	9	
5					

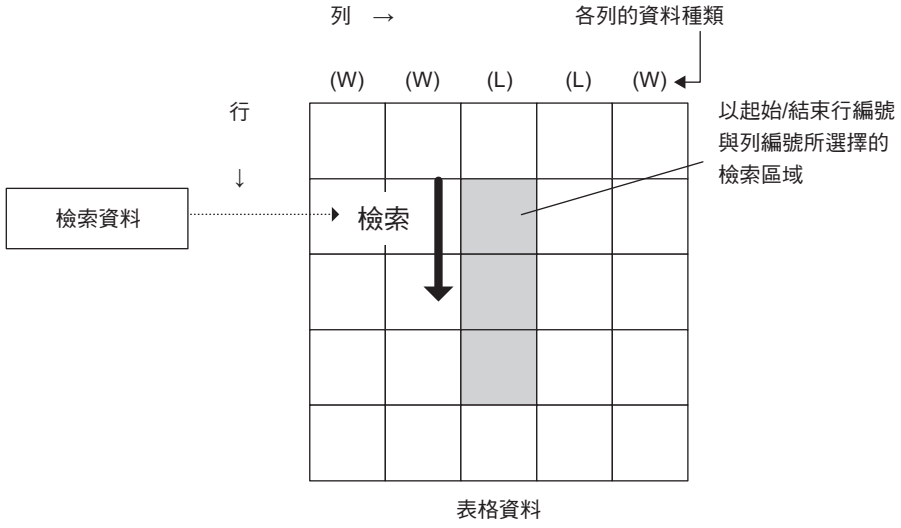
所被傳送的區域

* 顯示資料類型。

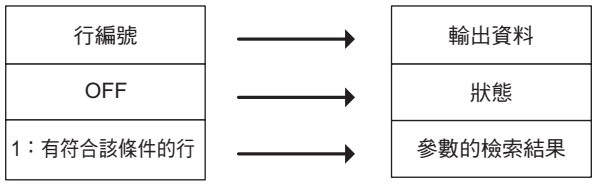
行搜尋 (垂直方向) (TBLSRL)

檢索表格名稱、行編號、列編號所指定的表格資料的列元素，若有和檢索資料一致的資料，其行編號將回報至輸出資料。檢索目標的資料類型將依照所指定的列元素之資料類型做自動判別。

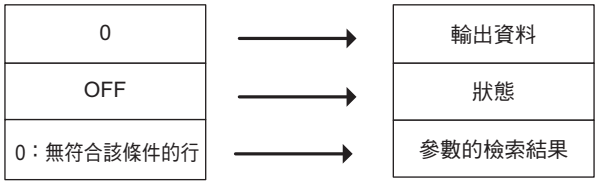
正常結束時，若有發現一致的資料，將輸入的參數表檢索結果設定為 1，並於輸出資料上設定行編號，再將狀態關閉。若沒有發現一致的資料，於檢索結果與輸出資料上設定為 0。發生錯誤時，請於輸出資料上設定錯誤碼，並將狀態切換為 ON。



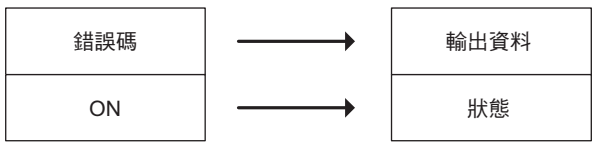
· 有一致的資料時



· 無一致的資料時

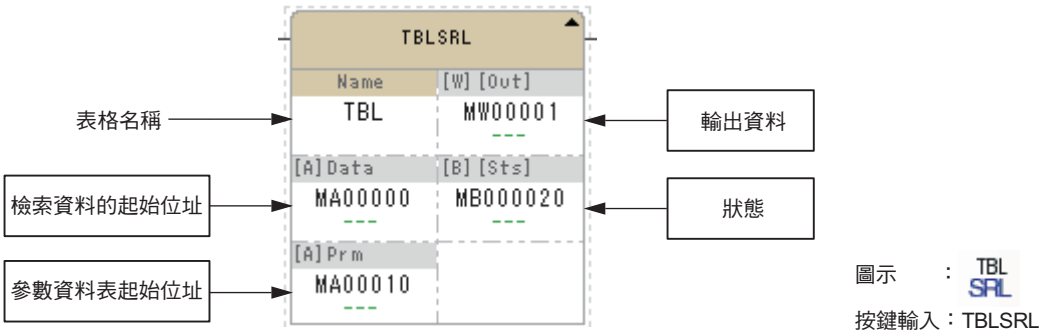


· 發生錯誤時



格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
檢索資料的起始位址 (Data)	×	×	×	×	×	×	○	×	×
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○	×	×
輸出資料 (Out) *1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts) (W) *1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW1	表格元素的起始行編號	作為目標之表格元素的起始行編號 (1 ~ 65535)	IN
2	L	ROW2	表格元素的最終行編號	作為目標的表元素之最終行編號 (1 ~ 65535)	IN
4	L	COLUMN	表格元素的列編號	作為目標的表元素之列編號 (1 ~ 32767)	IN
6	W	FIND	檢索結果	檢索結果 0：無符合該條件的行 1：有符合該條件的行	OUT

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

程式範例

以下所示為從陣列型的表格資料 TBL1 開始，於表格的直向來檢索檢索資料 (MW00000)32 的資料時的程式範例。

參數表的設定如以下所示。

暫存器	資料	備註
DL00010	2	表格元素的起始行編號
DL00012	5	表格元素的最終行編號
DL00014	2	表格元素的列編號

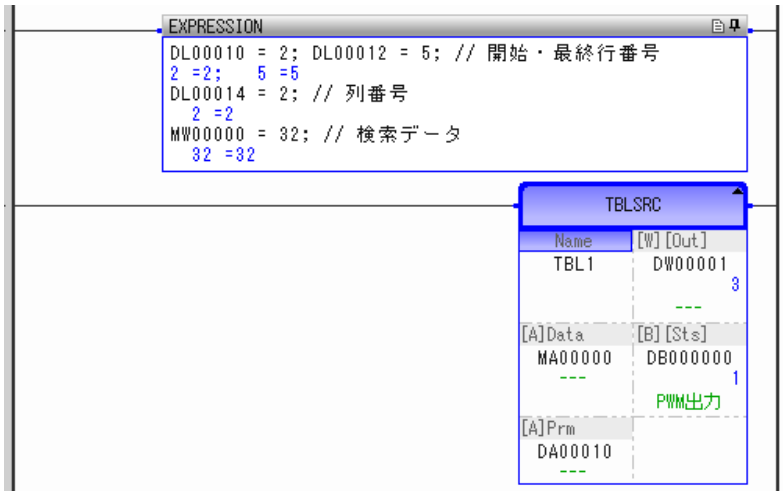
表格資料 TBL1 的內容如下所示。(表元素為整數型)

行	列				
	1 (W)*	2 (W)*	3 (W)*	4 (W)*	5 (W)*
1	11	12	13	14	15
2	21	22	23	24	25
3	31	32	33	34	35
4	41	42	43	44	45
5	51	52	53	54	55

* 顯示資料類型。

檢索區域

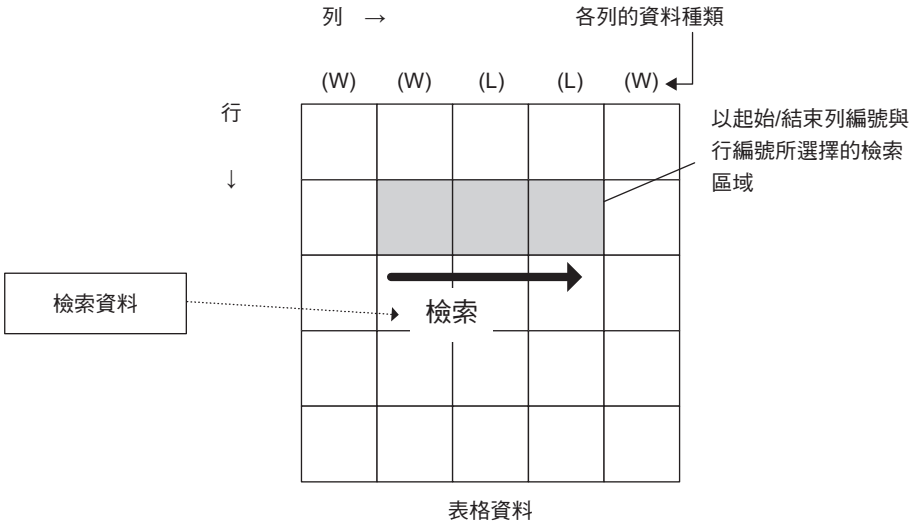
由於 32 和檢索區域的行編號 3 一致，因此 3 會被輸出至輸出資料 (DW00001)。



列搜尋 (水平方向) (TBLSRC)

檢索表格名稱、行編號、列編號所指定的表格資料的行元素，若有任何資料和檢索資料一致，該列編號將回報給輸出資料。檢索目標的資料類型將依照所指定之行元素的資料類型做自動判別。

正常結束時，若有發現一致的資料，將輸入的參數表檢索結果設定為 1，並於輸出資料上設定列編號，再將狀態關閉。沒有一致的資料時，請於檢索結果與輸出資料上設定為 0。發生錯誤時，請於輸出資料上設定錯誤碼，並將狀態切換為 ON。



· 有一致的資料時

列編號	→	輸出資料
OFF	→	狀態
1：有符合該條件的列	→	參數的檢索結果

· 無一致的資料時

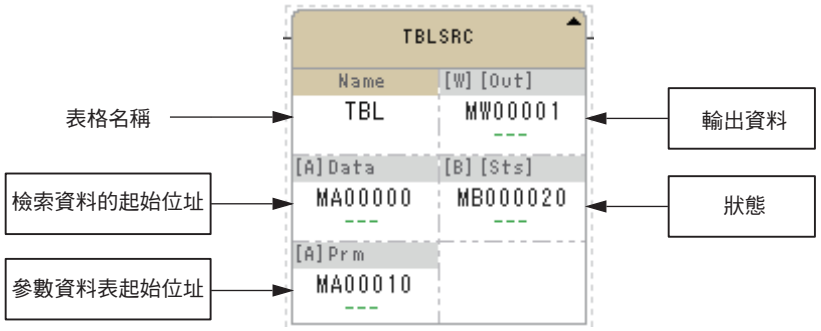
0	→	輸出資料
OFF	→	狀態
0：無符合該條件的列	→	參數的檢索結果

· 發生錯誤時

錯誤碼	→	輸出資料
ON	→	狀態

格式

所採用之格式如下。



圖示 ： TBL SRC

按鍵輸入： TBLSRC

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
檢索資料的起始位址 (Data)	×	×	×	×	×	×	○	×	×
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○	×	×
輸出資料 (Out) *1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts) *1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW1	格元素的行編號	作為目標的表元素之行編號 (1 ~ 65535)	IN
2	L	COLUMN1	表格元素的起始列編號	作為目標之表格元素的起始列編號 (1 ~ 32767)	IN
4	L	COLUMN2	表格元素的最終列編號	作為目標的表元素之最終列編號 (1 ~ 32767)	IN
6	W	FIND	檢索結果	表檢索結果 0：無符合該條件的列 1：有符合該條件的列	OUT

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	指定元素的資料類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

程式範例

以下所示為從陣列型的表格資料 TBL1 開始，以表格橫向來檢索檢索資料 (MW00000)34 的資料時的程式範例。

參數表的設定如以下所示。

暫存器	資料	備註
DL00010	3	表格元素的行編號
DL00012	2	表格元素的起始列編號
DL00014	5	表格元素的最終列編號

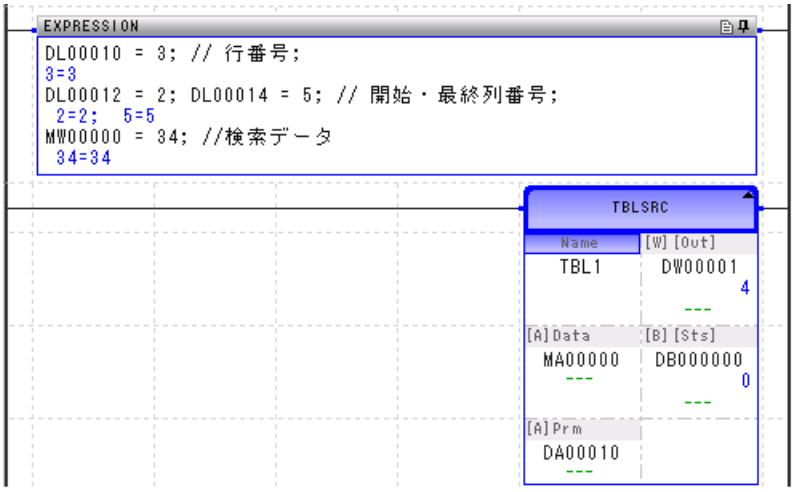
表格資料 TBL1 的內容如下所示。(表元素為整數型)

行	列				
	1 (W)*	2 (W)*	3 (W)*	4 (W)*	5 (W)*
1	11	12	13	14	15
2	21	22	23	24	25
3	31	32	33	34	35
4	41	42	43	44	45
5	51	52	53	54	55

檢索區域

* 顯示資料類型。

由於 34 和檢索區域的列編號 4 一致，因此 4 會被輸出至輸出資料 (DW00001)。



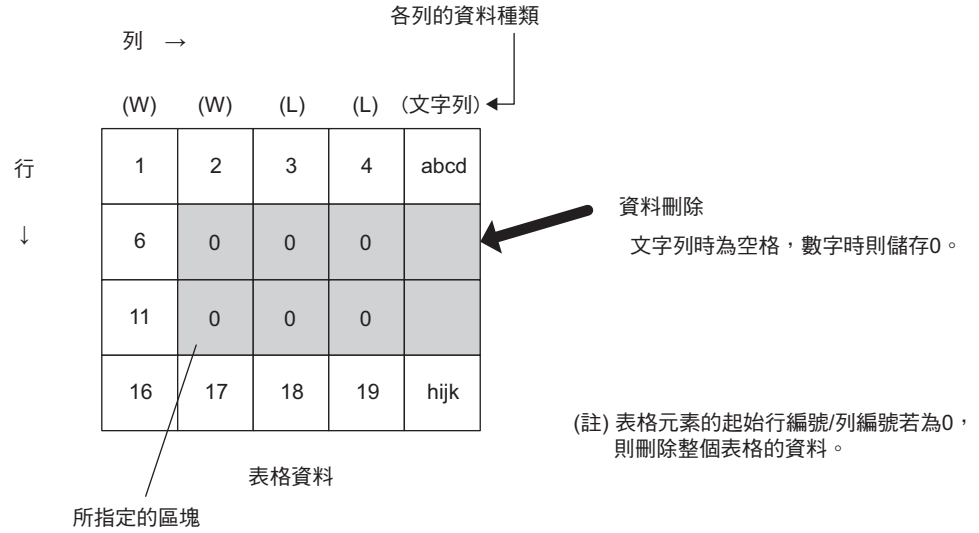
刪除區塊 (TBLCL)

將表格名稱、行編號、列編號所指定之表格資料的區塊資料刪除。表格元素的資料類型若為文字列則寫入空格，若為數字則寫入 0。

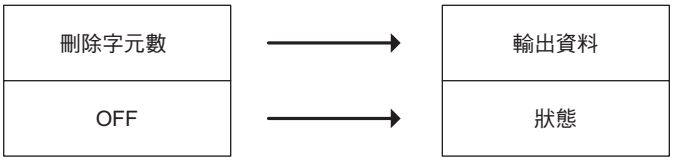
表格元素起始行編號與表元素起始列編號若均為 0，整個表格將被刪除。

表格參照上若有範圍以外、傳送目的端資料長度不足等不正常的情況，將回報錯誤，資料則不會被寫入。

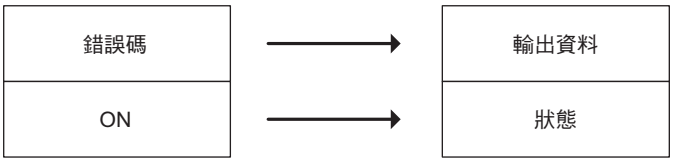
正常結束時，於輸出資料上設定刪除字元數，並將狀態切換為關閉。發生錯誤時，請於輸出資料上設定錯誤碼，並將狀態切換為開啟。



· 刪除成功時



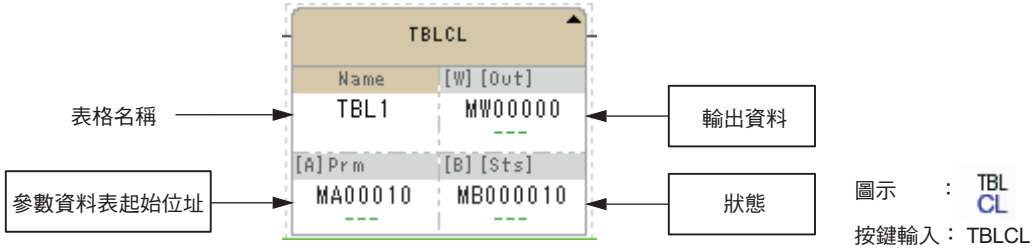
· 刪除失敗時



補充　刪除失敗時，資料將保持在執行指令前的狀態。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○	×	×
輸出資料 (Out)*1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts)*1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW	表格元素的起始行編號	作為目標的表元素之起始行編號 (1 ~ 65535)	IN
2	L	COL	表格元素的起始列編號	作為目標的表元素之起始列編號 (1 ~ 32767)	IN
4	W	RLEN	行元素的數量	行元素的數量 (1 ~ 32767)	IN
5	W	CLEN	列元素的數量	列元素的數量 (1 ~ 32767)	IN

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

程式範例

以下所示為啟動開關 1(DB000100) 後，從記錄型表格資料 TBL1 刪除指定區塊時的程式範例。

參數表的設定如以下所示。

暫存器	資料	備註
DL00000	2	表格元素的起始行編號
DL00002	2	表格元素的起始列編號
DW00004	3	行元素的數量
DW00005	3	列元素的數量

表格資料 TBL1 的內容如下所示。

行	列				
	1 (W)*	2 (W)*	3 (L)*	4 (文字列)*	5 (F)*
1	1000	1001	10000	ABCD	1.1
2	2000	2002	20000	BCDE	1.2
3	3000	3003	30000	CDEF	1.3
4	4000	4004	40000	DEFG	1.4
5	5000	5005	50000	EFGH	1.5

刪除的區域

* 顯示資料類型。

執行指令後，如以下所示資料將被刪除。

行	列				
	1 (W)*	2 (W)*	3 (L)*	4 (文字列)*	5 (F)*
1	1000	1001	10000	ABCD	1.1
2	2000	0	0		1.2
3	3000	0	0		1.3
4	4000	0	0		1.4
5	5000	5005	50000	EFGH	1.5

已刪除的區域

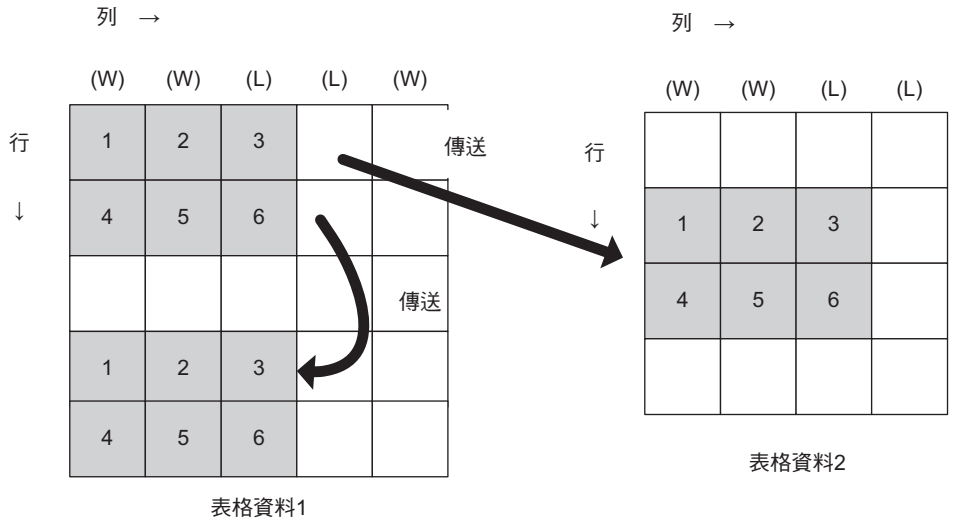
* 顯示資料類型。

傳送表格間區塊 (TBLMV)

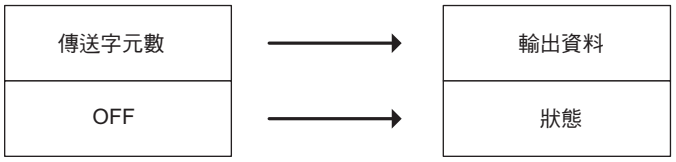
將表格名稱、行編號、列編號所指定之表格資料的區塊資料傳送至表格的其他區塊。可做不同表格之間的傳送以及同一個表格內的傳送。

傳送來源端與傳送目的端上的列元素資料類型不同時，將會回報錯誤，但不會傳送資料。

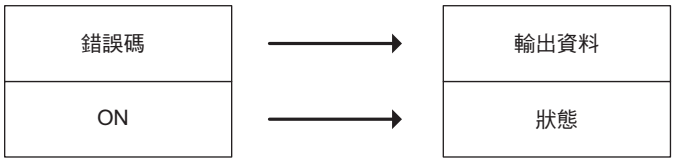
正常結束時會輸出傳送字元數，並且將狀態關閉。錯誤發生時會輸出錯誤碼，並且將狀態啟動。



· 傳送成功時



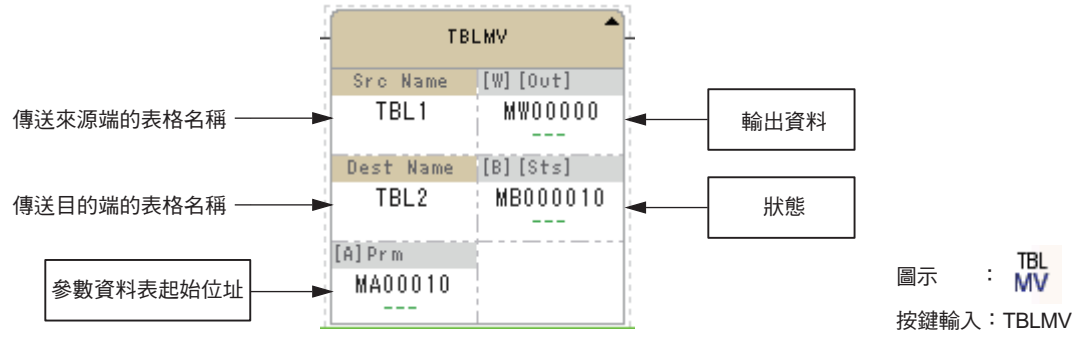
· 傳送失敗時



補充　傳送失敗時，資料將保持在執行指令前的狀態。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○	×	×
輸出資料 (Out)*1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts)*1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW1	表格元素的起始行編號	傳送來源端的表元素之起始行編號 (1 ~ 65535)	IN
2	L	COLUMN1	表格元素的起始列編號	傳送來源端的表元素之起始列編號 (1 ~ 32767)	IN
4	W	RLEN	行元素的數量	行元素的數量 (1 ~ 32767)	IN
5	W	CLEN	列元素的數量	列元素的數量 (1 ~ 32767)	IN
6	L	ROW2	表格元素的起始行編號	傳送目的端的表元素之起始行編號 (1 ~ 65535)	IN
8	L	COLUMN2	表格元素的起始列編號	傳送目的端的表元素之起始列編號 (1 ~ 32767)	IN

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

程式範例

以下所示為啟動開關 1 (DB000100) 後，從記錄型表格資料 TBL1 的指定區塊，傳送至表格資料 TBL2 的指定區塊時的程式範例。

表格資料 TBL1 的內容如下所示。

行	列		
	1 (W)*	2 (W)*	3 (L)*
1	1000	1001	10000
2	2000	2002	20000
3	3000	3003	30000
4	4000	4004	40000
5	5000	5005	50000

傳送區域

* 顯示資料類型。

參數表的設定如以下所示。

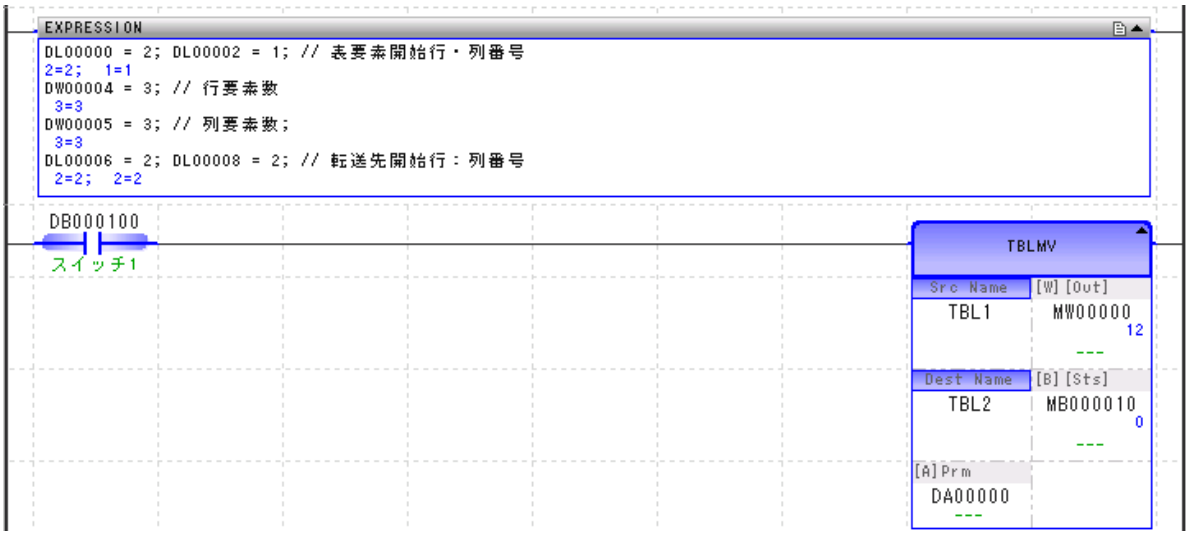
暫存器	資料	備註
DL00000	2	傳送來源端的起始行編號
DL00002	1	傳送來源端的起始列編號
DW00004	3	行元素的個數
DW00005	3	列元素的個數
DL00006	2	傳送目的端的起始行編號
DL00008	2	傳送目的端的起始列編號

表格資料 TBL2 的內容如下所示。

行	列				
	1 (W)*	2 (W)*	3 (W)*	4 (W)*	5 (W)*
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0

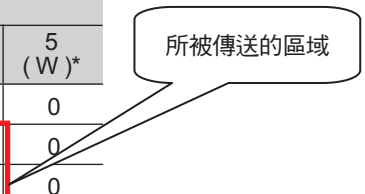
將被傳送的區域

* 顯示資料類型。



執行指令後表格資料 TBL2 的內容如下。

行	列				
	1 (W)*	2 (W)*	3 (W)*	4 (W)*	5 (W)*
1	0	0	0	0	0
2	0	2000	2002	20000	0
3	0	3000	3003	30000	0
4	0	4000	4004	40000	0
5	0	0	0	0	0



* 顯示資料類型。

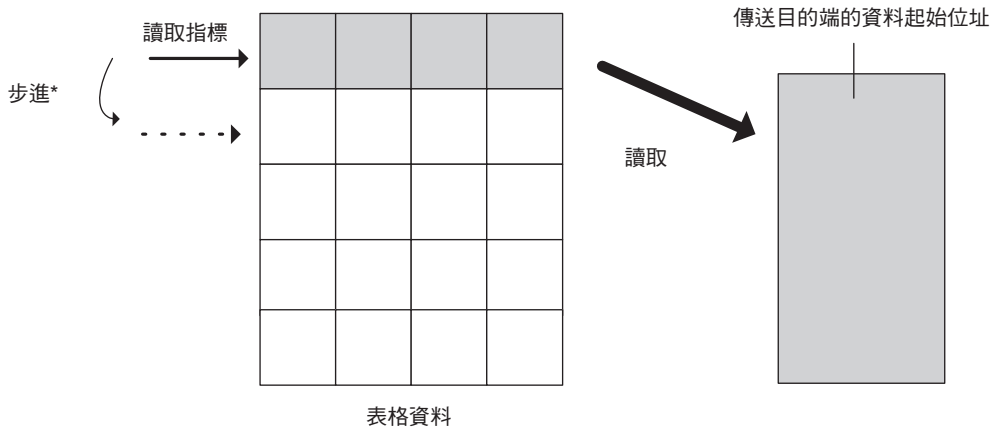
讀取佇列表 (QTBLR、QTBLRI)

連續讀取表格名稱、行編號、列編號所指定的表格資料的列元素，並儲存至從指定的暫存器開始的連續區域。讀取元素的資料類型將依照指定的表格自動做判別。儲存處的暫存器的資料類型將被忽略，所讀取的數值的資料類型也不會被轉換，而是依照表格的資料類型來儲存。

若為 QTBLR 指令，佇列表的讀取指標不會有所變化。若為 QTBLRI 指令，佇列表的讀取指標將步進 1 行。

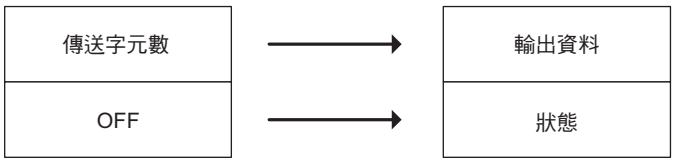
參照表格時，若有表格名稱錯誤、行編號超過範圍、佇列緩衝區空著等不正常的情況，將回報錯誤，資料不會被讀取，指標也不會步進。儲存處的暫存器內容也會被保持下來。

正常結束時會輸出傳送字元數，並且將狀態關閉。錯誤發生時會輸出錯誤碼，並且將狀態啟動。

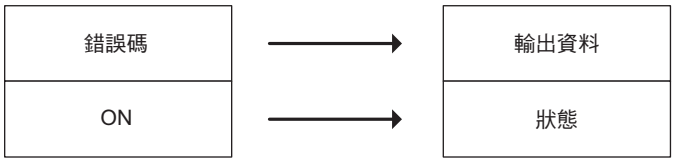


* 執行 QTBLR 指令後指標不會步進。
執行 QTBLRI 指令後指標會步進。

· 讀取成功時



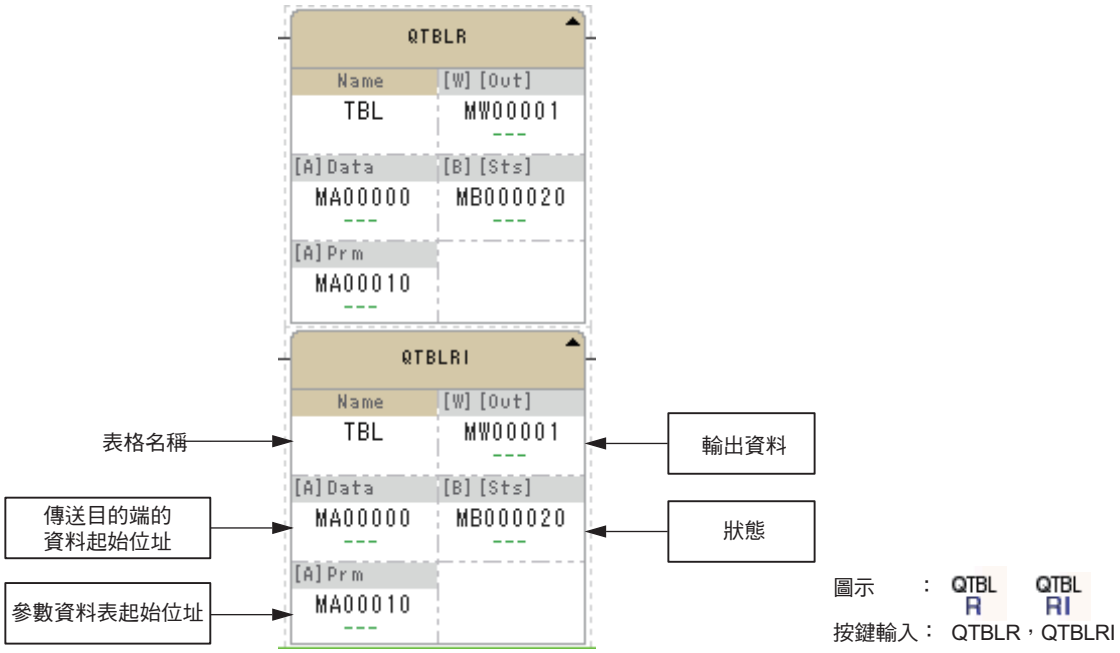
· 讀取失敗時



補充　讀取失敗時，傳送目的端資料將保持為執行指令前的狀態。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送目的端的資料起始位址 (Data)	×	×	×	×	×	×	○	×	×
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*2	×	×
輸出資料 (Out) *1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts) *1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW	表格元素相對的行編號	傳送來源端的表格元素之相對行編號 (1 ~ 65535)	IN
2	L	COLUMN	表格元素的起始列編號	傳送來源端的表格元素之起始列編號 (1 ~ 32767)	IN
4	W	CLEN	列元素的數量	傳送的列元素數量 (1 ~ 32767)	IN
5	W	備用			
6	L	RPTR	讀取指標	執行後佇列的讀取指標	OUT
8	L	WPTR	寫入指標	執行後佇列的寫入指標	OUT

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

◆ 設定表格元素的相對行編號

相對行編號	讀取行	備註
0	讀取指標行	僅 QTBLRI 指令時指標步進
1	讀取指標行	無指標步進
2	讀取指標行 -1	無指標步進
3	讀取指標行 -2	無指標步進
:	:	:
n	讀取指標行－ (n - 1)	無指標步進

程式範例

以下所示為啟動開關 2 (DB000002) 後，將陣列型表格資料 TBL1 的指定列元素從 DW00010 讀取至 DW00012 區域的程式範例。

啟動開關 2 之前，從 DW00000 變更 DW00002 的數值的同時，將開關 1 啟動 3 次，並將表格資料的內容如以下做設定。如欲進一步瞭解，請參閱以下項目之說明。

☞ **寫入佇列表 (QTBLW、QTBLWI) - 程式範例 (第 4-234 頁)**

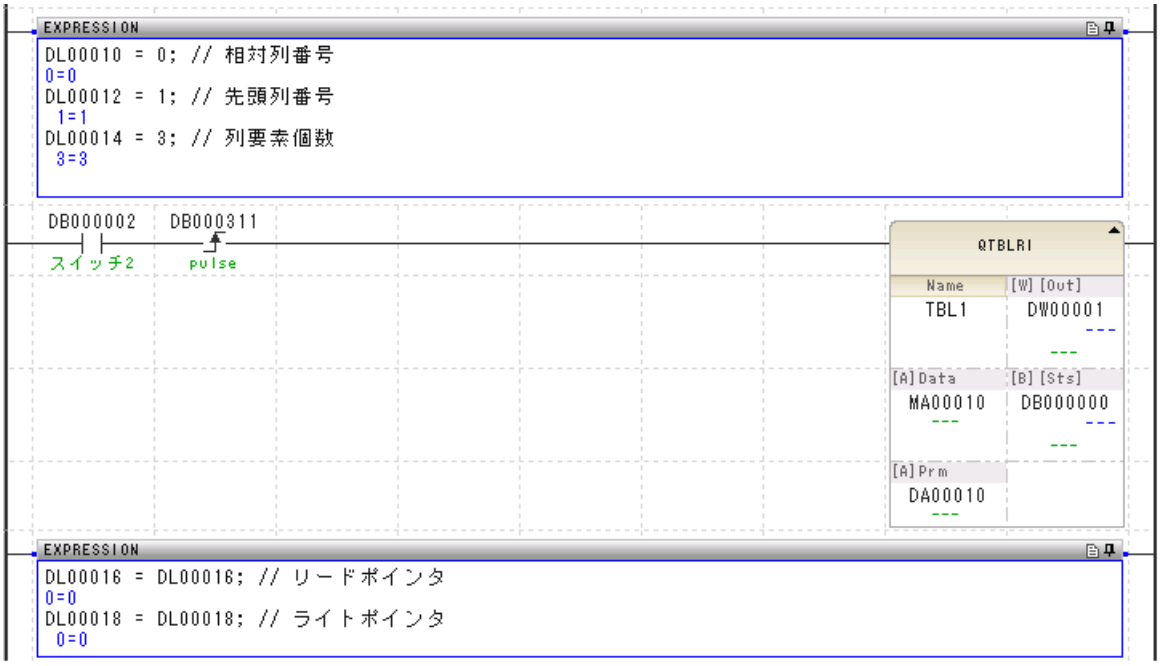
表格資料 TBL1 的內容如下所示。

行	列		
	1 (W)*	**2 (W)***	**3 (W)***
1	11	12	13
2	21	22	23
3	31	32	33

* 顯示資料類型。

參數表的設定如以下所示。

暫存器	資料	備註
DL00010	0	相對行編號
DL00012	1	起始列編號
DW00014	3	行元素的個數



在此將開關 2 (DB000002) 啟動 3 次，從第 1 次到第 3 次所讀取的資料變化如以下所示。

暫存器	第 1 次的資料	第 2 次的資料	第 3 次的資料
DW00010	11	21	31
DW00011	12	22	32
DW00012	13	23	33

第 1 次時讀取指標於第 1 行、第 2 次讀取指標於第 2 行 …，如此每執行一次讀取指標就會步進，因此得到以上的結果。

註記

由於電源輸入時的讀取指標與寫入指標不固定，因此在使用 QTBLR、QTBLRI、QTBLW 或 QTBLWI 指令前請先執行 QTBLCL 指令。
若沒有執行 QTBLCL 指令之下就執行 QTBLR、QTBLRI、QTBLW 或 QTBLWI 指令的話，有可能會導致計算錯誤。

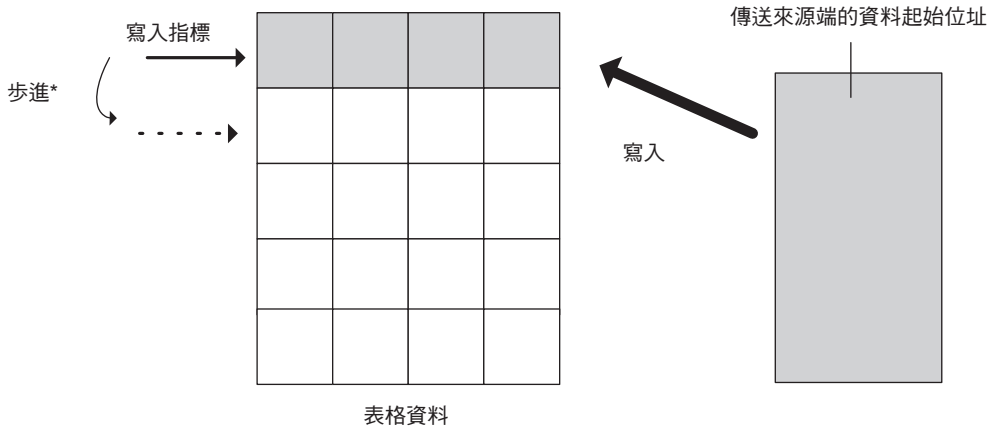
寫入佇列表 (QTBLW、QTBLWI)

將從指定暫存器開始的連續區域的資料，連續寫入至所指定的表格資料列。儲存來源端與儲存目的端的資料類型將被視為一致來進行處理。

若為 QTBLW 指令，佇列表的寫入指標將不會有所變化。若為 QTBLWI 指令，佇列寫入指標將步進 1 行。

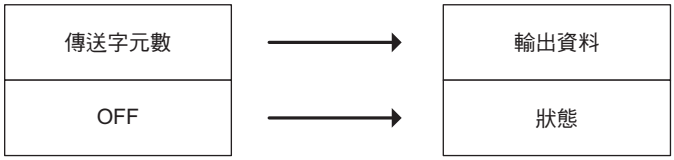
參照表格時，若有表格名稱錯誤、行編號超過範圍、佇列緩衝區已滿等不正常的情況，將回報錯誤，資料不會被寫入，因此指標也不會步進。儲存處的暫存器內容也會被保持下來。

正常結束時會輸出傳送字元數，並且將狀態關閉。錯誤發生時會輸出錯誤碼，並且將狀態啟動。

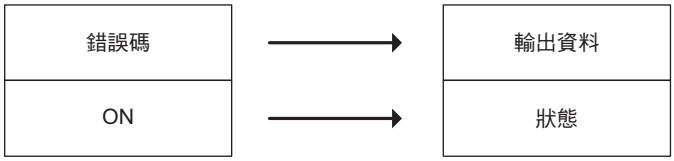


* 執行 QTBLR 指令後指標不會步進。
執行 QTBLRI 指令後指標會步進。

· 寫入成功時



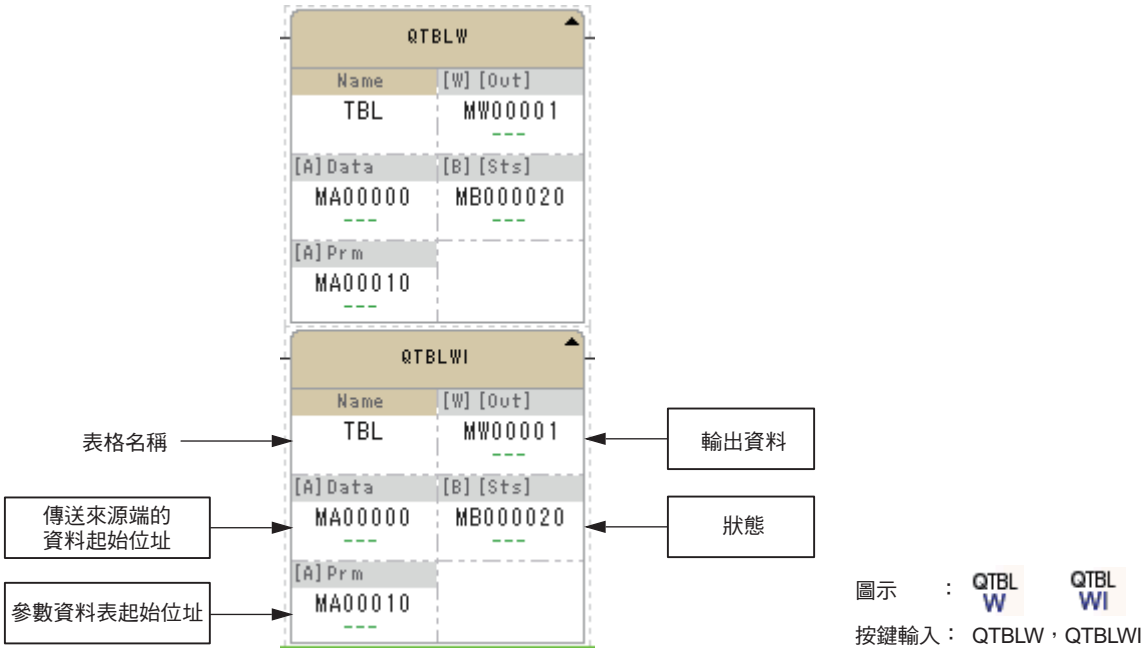
· 寫入失敗時



補充　寫入失敗時，表格資料將會保持在執行指令前的狀態。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
傳送來源端的資料起始位址 (Data)	×	×	×	×	×	×	○	×	×
參數資料表起始位址 (Prm)	×	×	×	×	×	×	○*2	×	×
輸出資料 (Out)*1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts)*1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

◆ 參數資料表

位址	資料類型	符號	名稱	規格	輸出入
0	L	ROW	表格元素相對的行編號	傳送目的端表格元素之相對行編號 (1 ~ 65535)	IN
2	L	COLUMN	表格元素的起始列編號	傳送目的端表格元素之起始列編號 (1 ~ 32767)	IN
4	W	CLEN	列元素的數量	傳送的列元素數量 (1 ~ 32767)	IN
5	W	備用			
6	L	RPTR	讀取指標	執行後佇列的讀取指標	OUT
8	L	WPTR	寫入指標	執行後佇列的寫入指標	OUT

◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

◆ 設定表格元素的相對行編號

相對行編號	讀取行	備註
0	寫入指標行	僅 QTBLWI 指令時指標步進
1	寫入指標行	無指標步進
2	寫入指標行－ 1	無指標步進
3	寫入指標行－ 2	無指標步進
:	:	:
n	寫入指標行－ (n - 1)	無指標步進

程式範例

以下所示為啟動開關 1 (DB000001) 後，從 MW00000 將 MW00002 的資料寫入至陣列型表格資料 TBL1 的指定列之程式範例。

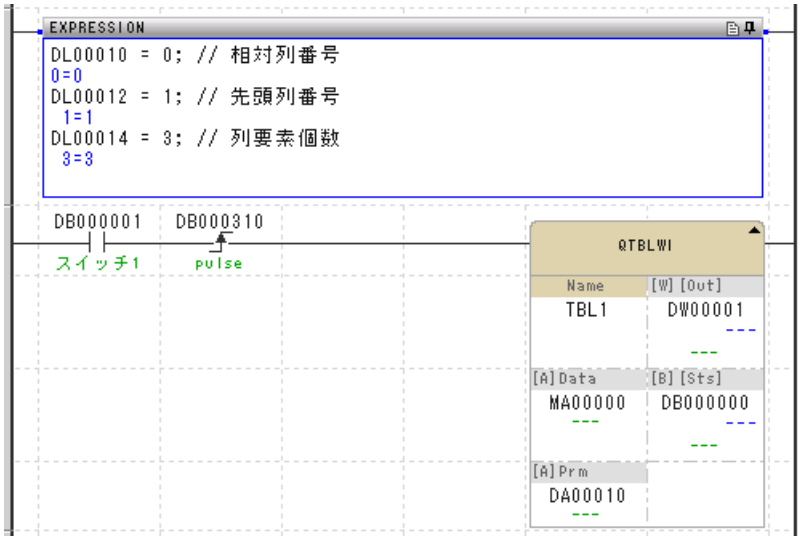
將執行前的表格資料 TBL1 進行初始化。

行	列		
	1 (W)*	**2 (W)***	**3 (W)***
1	0	0	0
2	0	0	0
3	0	0	0

* 顯示資料類型。

參數表的設定如以下所示。

暫存器	資料	備註
DL00010	0	相對行編號
DL00012	1	起始列編號
DW00014	3	行元素的個數



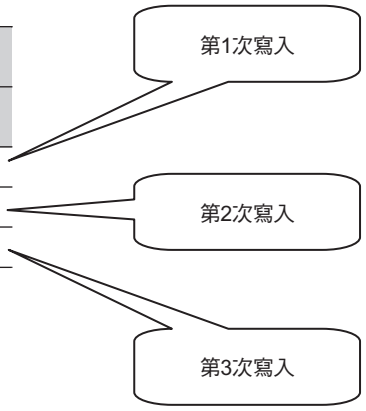
在此，如以下所示，從 MW00000 變更 MW00002 的數值，同時將開關 1 (DB000001) 啟動 3 次。

暫存器	第 1 次的資料	第 2 次的資料	第 3 次的資料
MW00000	11	21	31
MW00001	12	22	32
MW00002	13	23	33

第 1 次時寫入指標於第 1 行、第 2 次寫入指標於第 2 行 …，如此每執行一次寫入指標就會步進，執行 3 次後 TBL1 的資料便如以下所示。

行	列		
	1 (W)*	2 (W)*	3 (W)*
1	11	12	13
2	21	22	23
3	31	32	33

* 顯示資料類型。



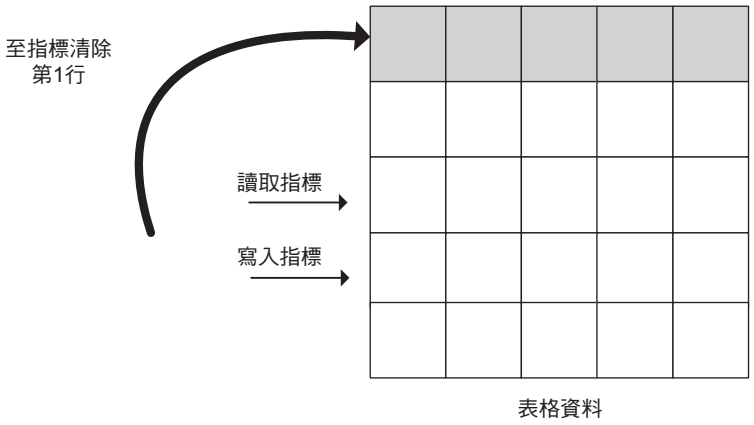
註記

由於電源輸入時的讀取指標與寫入指標不固定，因此在使用 QTBLR、QTBLRI、QTBLW 或 QTBLWI 指令前請先執行 QTBLCL 指令。
若沒有執行 QTBLCL 指令之下就執行 QTBLR、QTBLRI、QTBLW 或 QTBLWI 指令的話，有可能會導致計算錯誤。

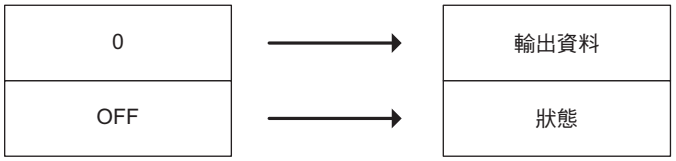
清除佇列指標 (QTBLCL)

將表格名稱所指定之表格資料的佇列讀取指標以及佇列寫入指標恢復至原始狀態 (第 1 行)。

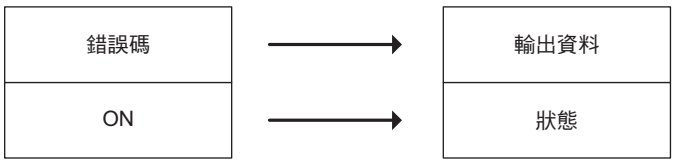
正常結束時，於輸出資料上設定為 0，並關閉狀態。發生錯誤時，請於輸出資料上設定錯誤碼，並將狀態切換為開啟。



· **成功清除佇列時**



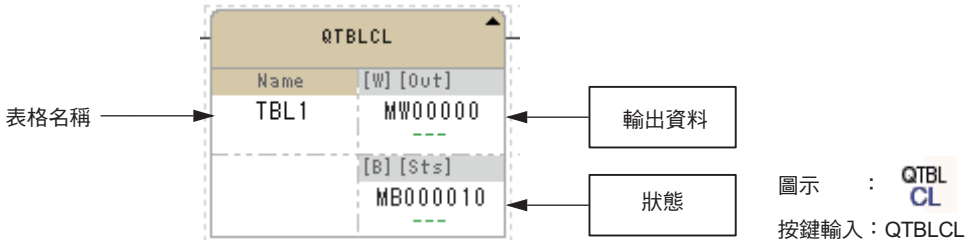
· **清除佇列失敗時**



補充　清除失敗時，佇列將保持執行指令前的狀態。

格式

所採用之格式如下。



輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
輸出資料 (Out)*1	×	○*2	×	×	×	×	×	○	×
狀態 (Sts)*1	○*2	×	×	×	×	×	×	×	×

*1. 可省略

*2. C、# 暫存器除外

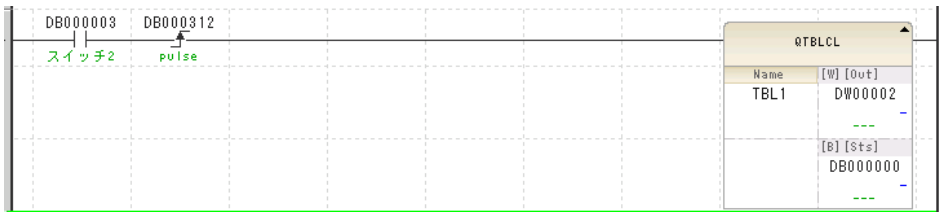
◆ 錯誤碼一覽表

錯誤碼	錯誤名稱	內容
0001 H	參照表未定義	尚未定義做為目標的表格。
0002 H	行編號於範圍外	表格元素的行編號未在做為目標的表格範圍內。
0003 H	列編號於範圍外	表格元素的列編號未在做為目標的表格範圍內。
0004 H	元素的數量不正確	作為目標的元素數量不恰當。
0005 H	儲存目的端容量不足	可儲存的容量不足。
0006 H	元素型不足	所指定的元素類型異常。
0007 H	佇列緩衝錯誤	佇列緩衝區空著時會讀取，佇列緩衝區已滿時則以指標步進的方式寫入。
0008 H	佇列表錯誤	所指定的表格不是佇列類型的表格。
0009 H	系統錯誤	執行指令時於系統內部偵測出無法預期的錯誤。

(註)錯誤碼在所有表格的操作指令上都相同。

程式範例

以下所示為啟動開關 2 (DB000003) 後，將指定的佇列表的佇列指標做初始化的程式範例。



註記

由於電源輸入時的讀取指標與寫入指標不固定，因此在使用 QTBLR、QTBLRI、QTBLW 或 QTBLWI 指令前請先執行 QTBLCL 指令。
若沒有執行 QTBLCL 指令之下就執行 QTBLR、QTBLRI、QTBLW 或 QTBLWI 指令的話，有可能會導致計算錯誤。

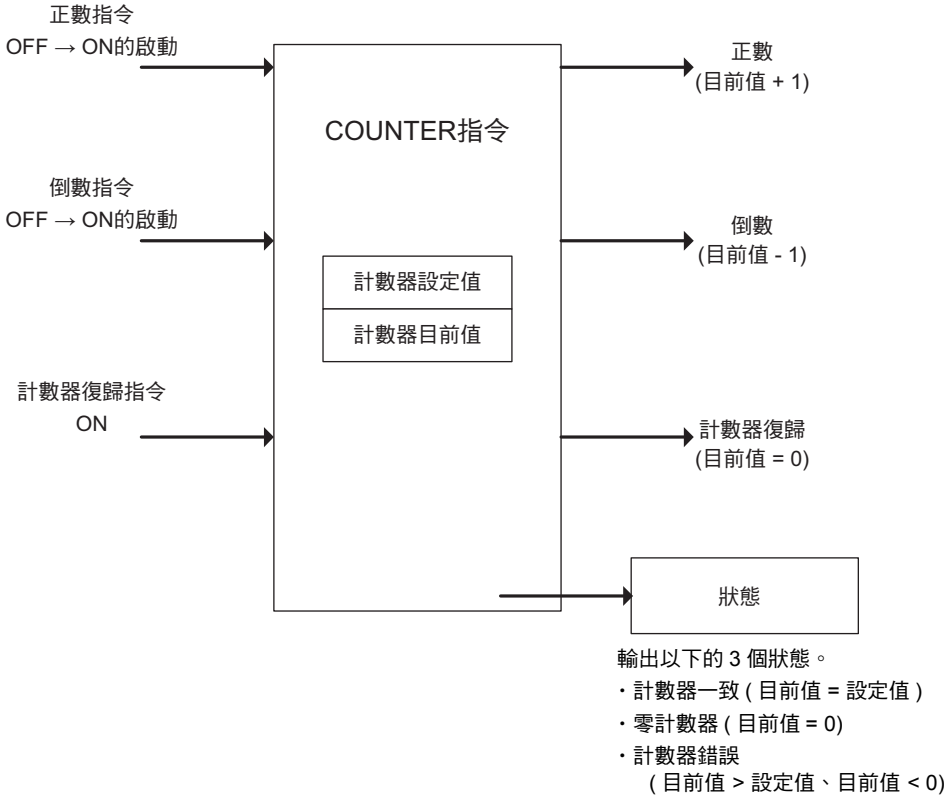
4.10 系統函數指令

計數器 (COUNTER)

正數 / 倒數指令從關閉變為啟動後，可上 / 下調整目前值。

計數器復歸指令啟動後，將計數器目前值設為 0。此外，比較計數器目前值與設定值後輸出結果。

計數器錯誤 (目前值 > 設定值) 時，不上 / 下調整目前值。

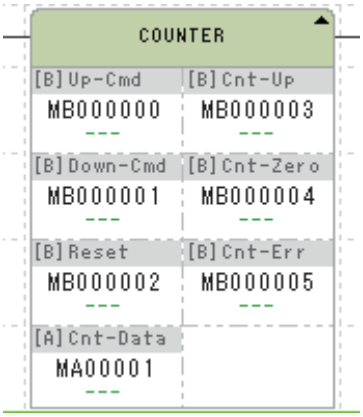


輸出以下的 3 個狀態。

- 計數器一致 (目前值 = 設定值)
- 零計數器 (目前值 = 0)
- 計數器錯誤
  (目前值 > 設定值、目前值 < 0)

格式

所採用之格式如下。



圖示　：COUN TER

按鍵輸入：COUNTER

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
正數指令 (Up-Cmd)	○	×	×	×	×	×	×	×	×
倒數指令 (Down-Cmd)	○	×	×	×	×	×	×	×	×
計數器復歸指令 (Reset)	○	×	×	×	×	×	×	×	×
計數器處理用的資料區域的起始位址 (Cnt-Data)	×	×	×	×	×	×	○*1	×	×
正數 (Cnt-Up)	○*2	×	×	×	×	×	×	×	×
零計數 (Cnt-Zero)	○*2	×	×	×	×	×	×	×	×
計數錯誤 (Cnt-Err)	○*2	×	×	×	×	×	×	×	×

*1. 僅 M、D 暫存器
*2. C、# 暫存器除外

各輸出入項目的內容說明如下。

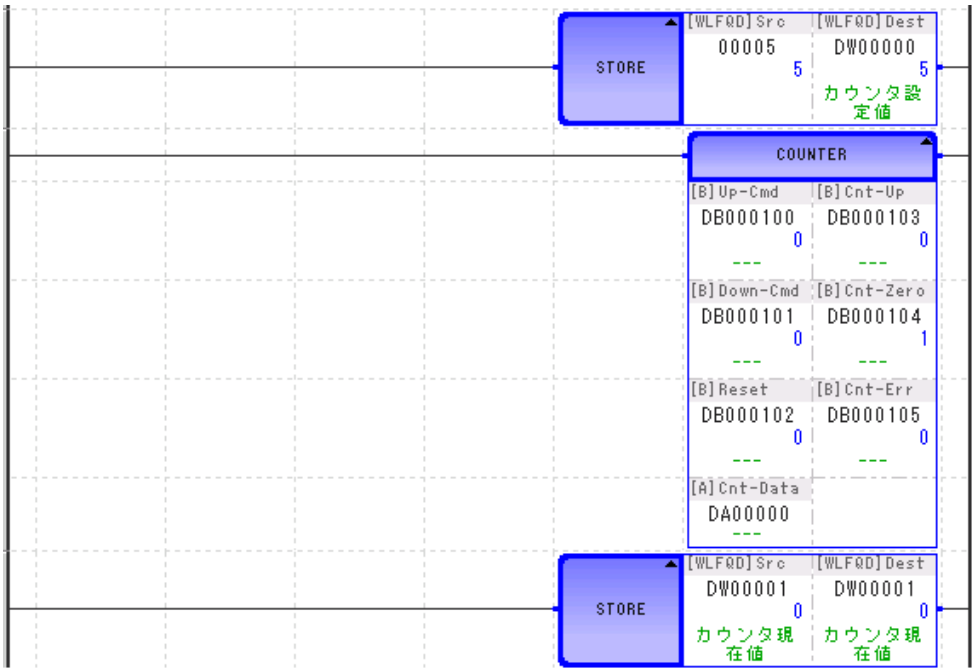
輸出輸入項目	內容	輸出入
正數指令 (Up-Cmd)	在 OFF → ON 的狀態下正數 * 目前值。	IN
倒數指令 (Down-Cmd)	在 OFF → ON 的狀態下倒數 * 目前值。	IN
計數器復歸指令 (Reset)	啟動後將目前值重置為 0。	IN
計數器處理用的資料區域的起始位址 (Cnt-Data)	+0 字元：設定值	IN
	+1 字元：目前值	OUT
	+2 字元：工作旗標	OUT
正數 (Cnt-Up)	目前值 = 以設定值啟動。	OUT
零計數 (Cnt-Zero)	目前值 = 以 0 啟動。	OUT
計數錯誤 (Cnt-Err)	目前值 > 以設定值啟動。 或目前值 < 0 啟動。	OUT

* 正數指令與倒數指令同時從關閉變為啟動時，目前值將不會有所變化。

程式範例

以下程式範例的第 1 行於計數器設定值上設定為 5，第 3 行則是執行計數器目前值 (DW00001) 的監控。

於此正數指令 (DB000100) 從關閉變為啟動後，將進行 DW00001 的正數，倒數指令 (DB000101) 從關閉變為啟動後，將進行 DW00001 的倒數。

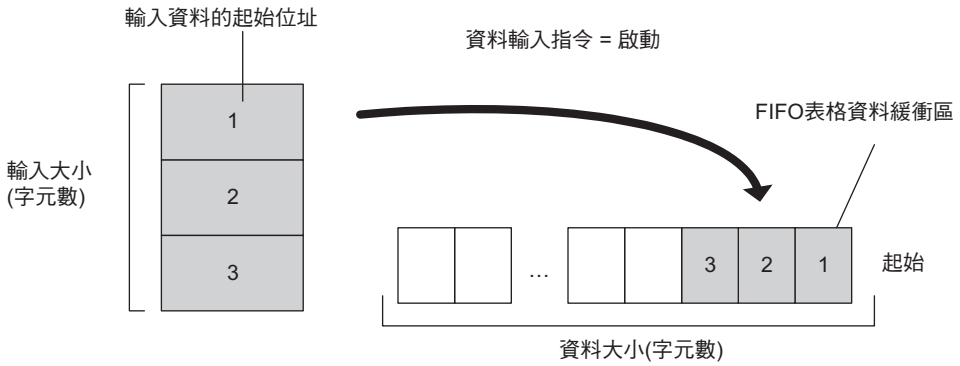


先進 / 先出 (FINFOUT)

此為先進 / 先出型區塊資料的傳送函數。FIFO 表格乃是由 4 個字元的標頭部分與資料緩衝區所組成。叫出本函數前，請務必先設定標頭部分的 3 個字元 (資料大小、輸入大小、輸出大小)。

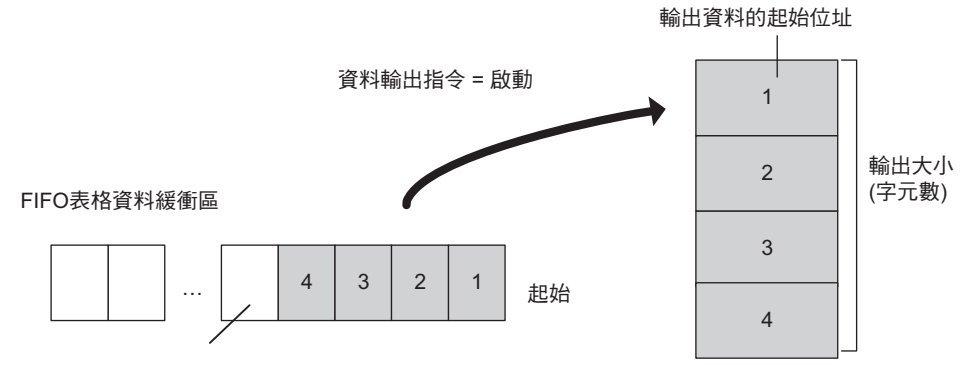
■ 資料輸入指令 (In-Cmd) = 啟動時

資料輸入指令啟動時，從指定的輸入資料區域將指定數量的資料依序儲存至 FIFO 表格的資料區域。



■ 資料輸出指令 (Out-Cmd) = 啟動時

資料輸出指令啟動時，指定數量的資料將從 FIFO 表格的資料區域起始傳送至指定的輸出資料區域。



■ 重置指令 (Reset) = 啟動時

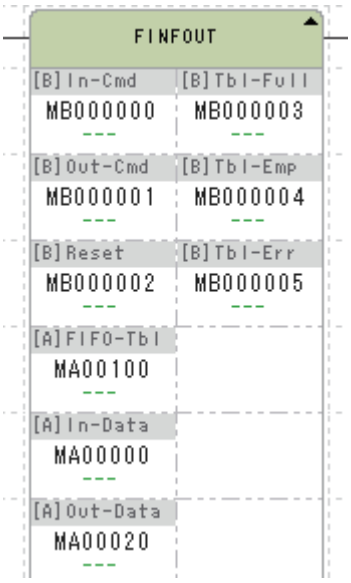
FIFO 表格的儲存數為 0，FIFO 表格空白 (Tbl-Emp) 將啟動。

(註)表格緩衝區的內容將保持原狀，不清除 0。

補充　資料空間大小 < 輸入大小或資料大小 < 輸出大小時，FIFO 表格錯誤 (Tbl-Err) 將啟動。

格式

所採用之格式如下。



圖示　：

按鍵輸入： FINFOUT

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
資料輸入指令 (In-Cmd)	○	×	×	×	×	×	×	×	×
資料輸出指令 (Out-Cmd)	○	×	×	×	×	×	×	×	×
重置指令 (Reset)	○	×	×	×	×	×	×	×	×
FIFO 表格的起始位址 (FIFO-Tbl)	×	×	×	×	×	×	○*1	×	×
輸入資料的起始位址 (in-Data)	×	×	×	×	×	×	○*1	×	×
輸出資料的起始位址 (Out-Data)	×	×	×	×	×	×	○*1	×	×
FIFO 表格已填滿 (Tbl-Full)	○*2	×	×	×	×	×	×	×	×
FIFO 表格為空白 (Tbl-Emp)	○*2	×	×	×	×	×	×	×	×
FIFO 表格錯誤 (Tbl-Err)	○*2	×	×	×	×	×	×	×	×

*1. 僅 M、D 暫存器

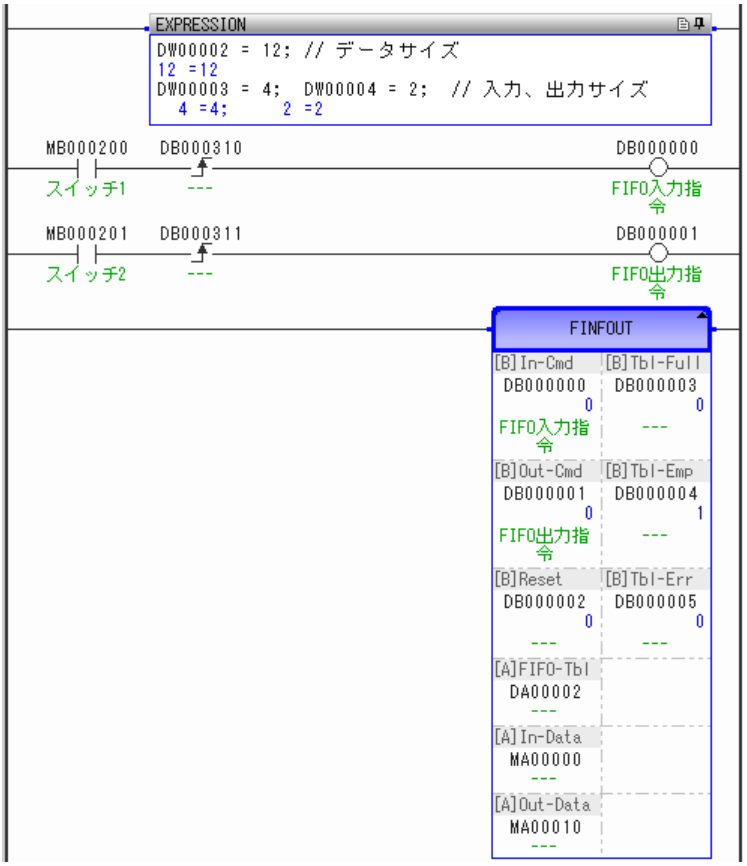
*2. C、# 暫存器除外

各輸出入項目的內容說明如下。

輸出輸入項目	內容	輸出入
資料輸入指令 (In-Cmd)	啟動後儲存至 FIFO 表格。	IN
資料輸出指令 (Out-Cmd)	啟動後從 FIFO 表格傳送。	IN
重置指令 (Reset)	啟動後儲存數變為 0。	IN
FIFO 表格的起始位址 (FIFO-Tbl)	+0 字元：資料大小	IN
	+1 字元：輸入大小	IN
	+2 字元：輸出大小	IN
	+3 字元：資料儲存數	OUT
	+4 字元 ~：資料	OUT
輸入資料的起始位址 (in-Data)	輸入資料的起始位址	IN
輸出資料的起始位址 (Out-Data)	輸出資料的起始位址	IN
FIFO 表格已填滿 (Tbl-Full)	FIFO 表格已填滿時啟動。	OUT
FIFO 表格為空白 (Tbl-Emp)	FIFO 表格為空白時啟動。	OUT
FIFO 表格錯誤 (Tbl-Err)	FIFO 表格發生錯誤時啟動。	OUT

程式範例

以下所示為製作資料大小為 12 字元、輸入大小為 4 字元、輸出大小為 2 字元的 FIFO 表格並執行 FINFOUT 指令的程式範例。



啟動開關 1 (MB000200) 後，從 MW00000 到 MW00003 的資料將儲存至 FIFO 表格緩衝區。

此外，資料儲存數 (DW00005) 變為 4。

暫存器	資料
MW00000	123
MW00001	234
MW00002	345
MW00003	456

FIFO表格緩衝區	資料
DW00006	123
DW00007	234
DW00008	345
DW00009	456
DW00010	0
:	:
DW00017	0

所儲存的區域

而後啟動開關 2(MB000201) 後，從 FIFO 表格緩衝區的起始處 2 個字元的資料，將從 MW0010 被輸出至 MW0011 的區域。此外，資料儲存數 (DW00005) 變為 2。

FIFO表格緩衝區	資料
DW00006	123 →345
DW00007	234 →456

暫存器	資料
MW00010	123
MW00011	234
MW00012	0

所儲存的區域

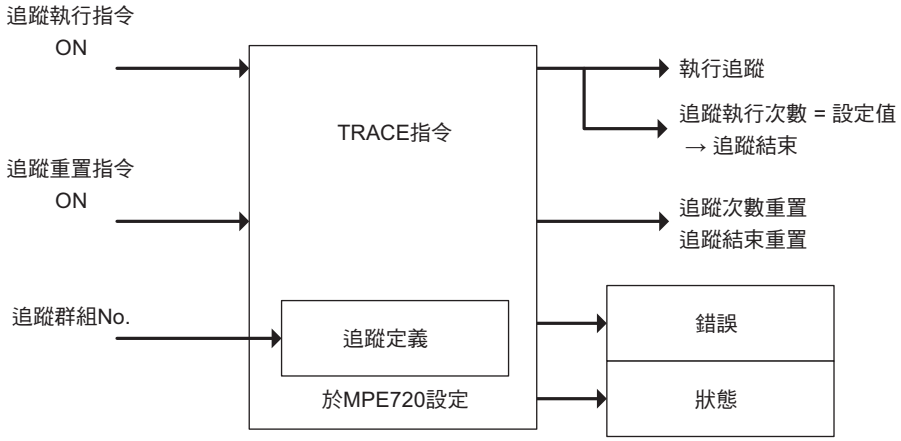
輸出的部分將消失，其後的區域會往前移動。

追蹤 (TRACE)

執行追蹤群組 No. (1 ~ 4) 所指定之追蹤資料的追蹤執行控制。

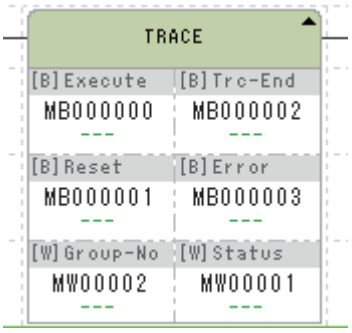
補充　追蹤定義的設定在 MPE720 的「資料追蹤的定義」執行。詳情請參閱以下使用手冊之說明。
🕮 運動控制器 MP2000/3000 系列　運動控制器系統　設定手冊 (資料編號：YTWMNCO-15003A)

- 追蹤執行指令 (Excute) 啟動的狀態下執行追蹤。
- 追蹤重置指令 (Reset) 啟動後，追蹤次數的計數器將被重置。此時追蹤結束 (Trc-End) 也會被重置。
- 追蹤執行所設定的次數後，追蹤結束 (Trc-End) 將變為啟動。



格式

所採用之格式如下。



圖示　：　TRACE

按鍵輸入：TRACE

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
追蹤執行指令 (Execute)	○	×	×	×	×	×	×	×	×
追蹤重置指令 (Reset)	○	×	×	×	×	×	×	×	×
追蹤群組 No. (Group-No)	×	○	×	×	×	×	×	○	○
追蹤結束 (Trc-End)	○*	×	×	×	×	×	×	×	×
錯誤 (Error)	○*	×	×	×	×	×	×	×	×
狀態 (Status)	×	○*	×	×	×	×	×	○	×

* C、# 暫存器除外

各輸出入項目的內容說明如下。

輸出輸入項目	內容	輸出入
追蹤執行指令 (Execute)	啟動後開始執行追蹤。	IN
追蹤重置指令 (Reset)	啟動後重置追蹤。	IN
追蹤群組 No. (Group-No)	追蹤群組 No. (1 ~ 4)	IN
追蹤結束 (Trc-End)	於追蹤結束時啟動。	OUT
錯誤 (Error)	於發生錯誤時啟動。	OUT
狀態 (Status)	追蹤執行狀態	OUT

狀態的結構如下。

Bit	名稱	備註
0	追蹤資料已滿	將指定群組的資料追蹤記憶體內轉一圈即可變為啟動。
1 ~ 7	系統預約	–
8	無追蹤定義	函數未執行。
9	指定群組 No. 錯誤	函數未執行。
A ~ C	系統預約	–
D	執行的時間點錯誤	函數未執行。
E	系統預約	–
F	系統預約	–

程式範例

以下所示為使用追蹤群組 No. 1 的定義並執行追蹤的程式範例。

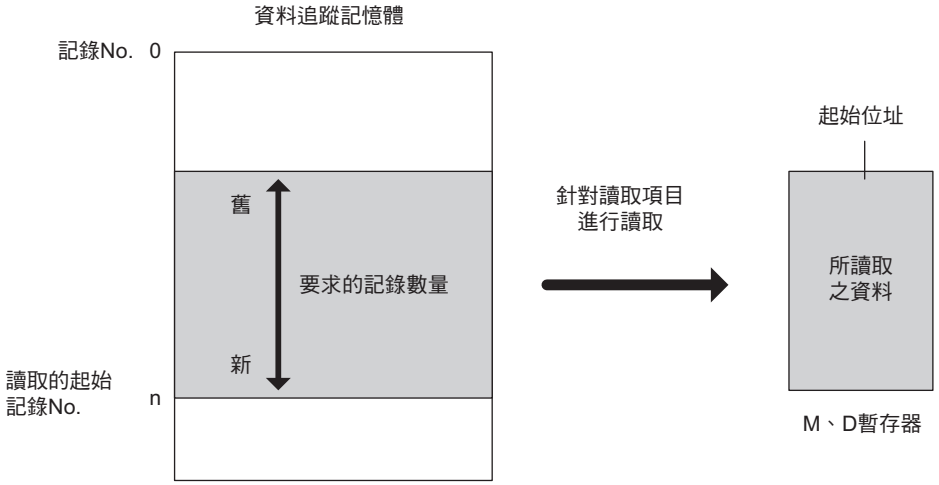
追蹤執行指令 (DB000000) 啟動後開始追蹤。


註記

請於 MPE720 設定追蹤群組 No. 1 的資料追蹤定義。此時取樣條件請設定為 「程式指定」。

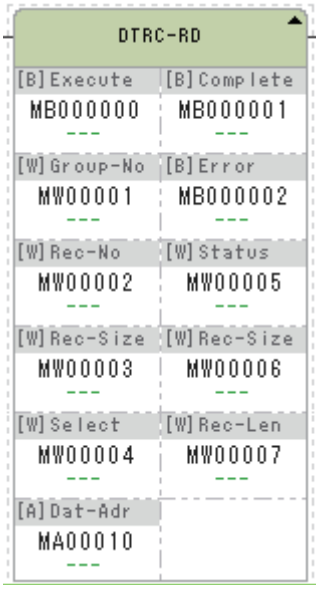
讀取資料追蹤 (DTRC-RD)

讀取運動控制器本體的追蹤資料，並儲存至暫存器。指定記錄 No. 與記錄數量後，便可讀取追蹤記憶體內的資料。只須指定記錄內的必要項目便可讀取。



格式

所採用之格式如下。



圖示 ： DTRC -RD

按鍵輸入：DTRCRD

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
讀取追蹤的執行指令 (Execute)	○	×	×	×	×	×	×	×	×
追蹤群組 No. (Group-No)	×	○	×	×	×	×	×	×	×
記錄 No. (Rec-No)	×	○	×	×	×	×	×	×	×
記錄數量 (Rec-Size)	×	○	×	×	×	×	×	×	×
項目選擇 (Select)	×	○	×	×	×	×	×	×	×
起始位址 (Dat-Adr)	×	×	×	×	×	×	○*1	×	×
追蹤結束 (Complete)	○*2	×	×	×	×	×	×	×	×
錯誤 (Error)	○*2	×	×	×	×	×	×	×	×
狀態 (Status)	×	○*2	×	×	×	×	×	×	×
讀取的記錄數量 (Rec-Size)	×	○*2	×	×	×	×	×	×	×
讀取 1 記錄數量 (Rec-Len)	×	○*2	×	×	×	×	×	×	×

*1. 僅 M、D 暫存器

*2. C、# 暫存器除外

各輸出入項目的內容說明如下。

輸出輸入項目	內容	輸出入
讀取追蹤的執行指令 (Execute)	資料追蹤的讀取執行指定	IN
追蹤群組 No. (Group-No)	資料追蹤群組 No. (1 ~ 4)	IN
記錄 No. (Rec-No)	讀取的起始記錄 No. (0 ~ 最多記錄數量 - 1)	IN
記錄數量 (Rec-Size)	讀取的所須記錄數量 (0 ~ 最多記錄數量 - 1)	IN
項目選擇 (Select)	讀取項目 (0001H ~ FFFFH) 位元 0 ~ F 可支援追蹤定義的資料指定 1 ~ 16。	IN
起始位址 (Dat-Adr)	讀取起始暫存器 No. (MA、DA)	IN
追蹤結束 (Complete)	追蹤讀取結束後啟動。	OUT
錯誤 (Error)	於發生錯誤時啟動。	OUT
狀態 (Status)	資料追蹤的讀取執行狀態	OUT
讀取的記錄數量 (Rec-Size)	讀取的記錄數量	OUT
讀取 1 記錄數量 (Rec-Len)	讀取 1 記錄數量 (字元數)	OUT

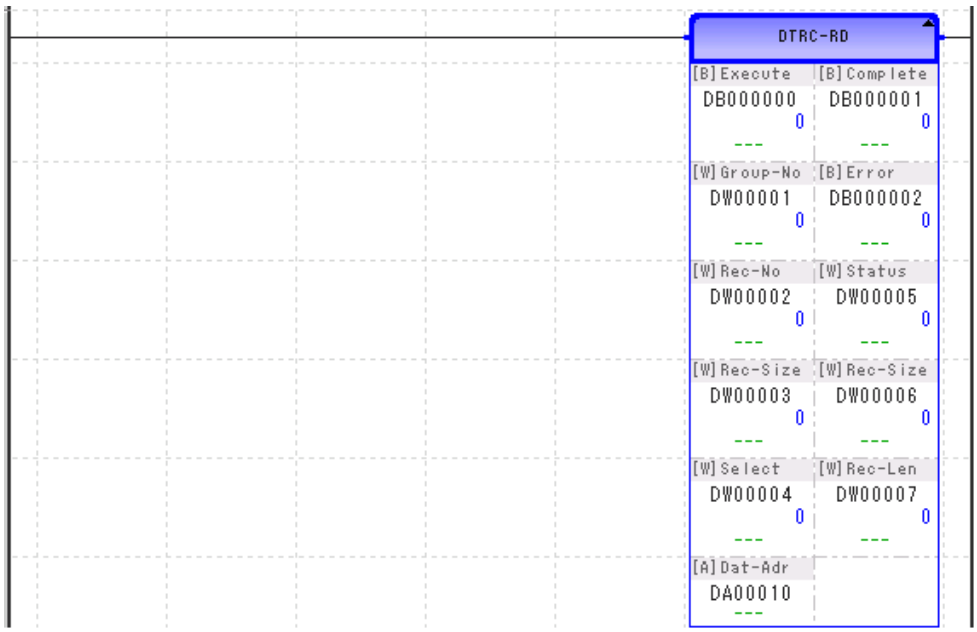
狀態的結構如下。

Bit	名稱	備註
0 ~ 7	系統預約	–
8	無追蹤定義	函數未執行。
9	群組 No. 錯誤	函數未執行。
A	指定記錄 No. 錯誤	函數未執行。
B	指定記錄數量錯誤	函數未執行。
C	資料儲存錯誤	函數未執行。
D	系統預約	–
E	系統預約	–
F	位址輸入錯誤	函數未執行。

程式範例

以下所示為讀取群組定義 No. 1 的資料追蹤之程式範例。

追蹤讀取執行指令 (DB000000) 啟動後，將執行追蹤資料的讀取。

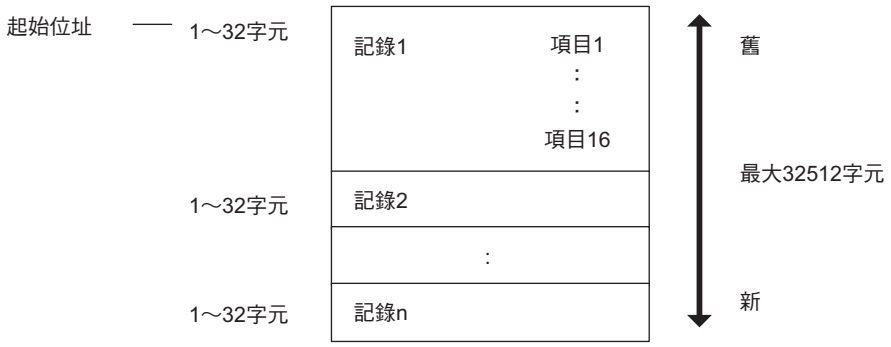


補充事項

◆ 讀取資料的結構

記錄長度由所選擇的項目資料所組成，1 個記錄大小為 1 ~ 32 個字元。

最多記錄數量會隨著記錄長度變動為 1015 ~ 32511。



◆ 記錄長度

記錄由所選擇的項目資料所組成。

如以下所示，記錄長度 (1 個記錄的字元數) 將依據所選擇的暫存器或點數等而有所不同。

· 1 個記錄的字元數 = Bn x 1 字元 + Wn x 1 字元 + Ln x 2 字元 + Fn x 2 字元
  Bn ：位元型暫存器的選擇點數
  Wn：整數型暫存器的選擇點數
  Ln ：長整數型暫存器的選擇點數
  Fn ：實數型暫存器的選擇點數
  合計最多為 16 點。
· 最大記錄長度 = 32 字元 (長整數型或實數型暫存器為 16 點時)
· 最小記錄長度 = 1 字元 (位元型或整數型暫存器為 1 點時)

◆ 記錄數量

如下所示，可指定的記錄數量將根據記錄長度而有所變化。

· 最大記錄長度時的記錄數量：0 ~ 1015
· 最小記錄長度時的記錄數量：0 ~ 32511　　(上限：32521/ 記錄長度 - 1)

◆ 最新記錄編號

如下表所示，各追蹤群組的最新記錄編號將儲存於系統暫存器。

系統暫存器編號	內容
SW00100	群組 1　最新記錄編號
SW00101	群組 2　最新記錄編號
SW00102	群組 3　最新記錄編號
SW00103	群組 4　最新記錄編號
SW00104	–
SW00105	–
SW00106	–
SW00107	–

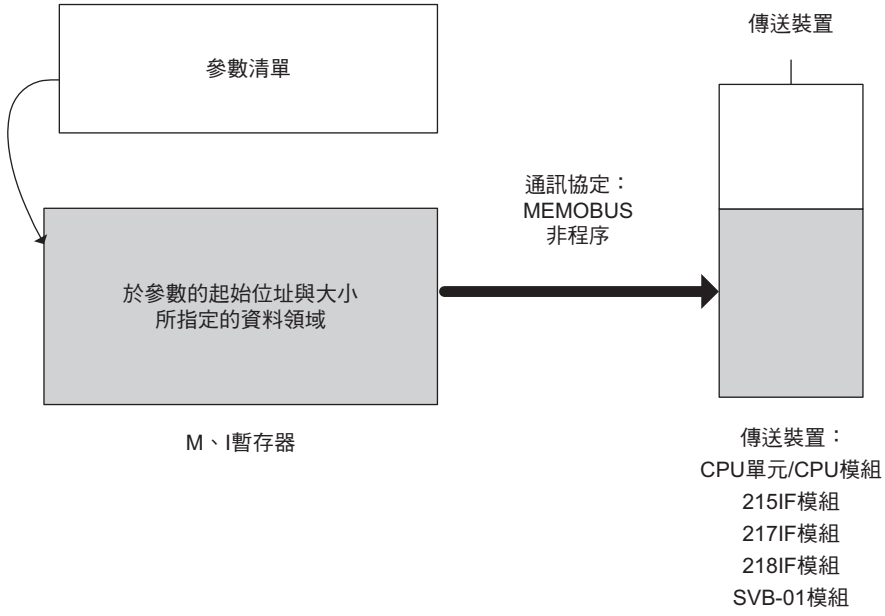
傳送訊息 (MSG-SND)

可針對傳送裝置類型中所指定的線路上的目的端傳送訊息。

支援以下的傳送裝置與協定。

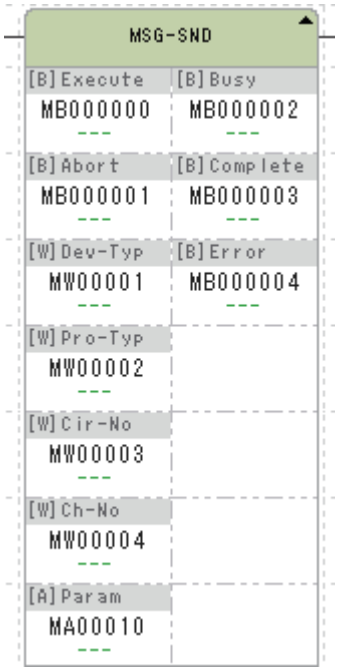
傳送裝置：CPU 單元 /CPU 模組、215IF 模組、217IF 模組、218IF 模組、SVB-01 模組

通訊協定：MEMOBUS 通訊，非程序



格式

所採用之格式如下。



圖示　：　MSG -SND

按鍵輸入：　MSG-SND

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
執行傳送指令 (Execute)	○	×	×	×	×	×	×	×	×
強制中斷傳送指令 (Abort)	○	×	×	×	×	×	×	×	×
傳送裝置類型 (Dev-Typ)	×	○	×	×	×	×	×	○	○
傳送通訊協定 (Pro-Typ)	×	○	×	×	×	×	×	○	○
線路編號 (Cir-No)	×	○	×	×	×	×	×	○	○
傳送緩衝頻道編號 (Ch-No)	×	○	×	×	×	×	×	○	○
參數清單起始位址 (Param)	×	×	×	×	×	×	○*1	×	×
處理中 (Busy)	○*2	×	×	×	×	×	×	×	×
處理完成 (Complete)	○*2	×	×	×	×	×	×	×	×
發生錯誤 (Error)	○*2	×	×	×	×	×	×	×	×

*1. 僅 M、G、D 暫存器
*2. C、# 暫存器除外

關於輸出入項目、詳細的參數內容、程式範例，請參照以下的手冊。

📖 **運動控制器 MP2000 系列　使用者手冊　階梯圖程式篇 (資料編號：SI-C887-1.2)**

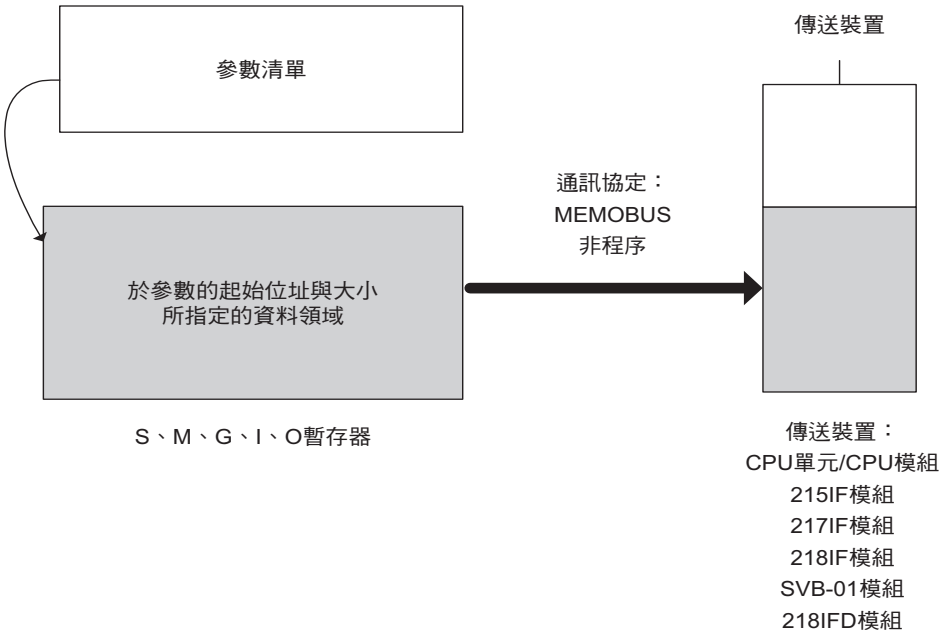
傳送訊息 (擴充) (MSG-SNDE)

可針對傳送裝置類型中所指定的線路上的目的端傳送訊息。

支援以下的傳送裝置與協定。

傳送裝置：CPU 單元 /CPU 模組、215IF 模組、217IF 模組、218IF 模組、SVB-01 模組、218IFD 模組

通訊協定：MEMOBUS 通訊，非程序



雖然基本動作與 MSG-SND 指令相同，但一般情況下請使用支援 MP3000 規格的 MSG-SNDE 指令。

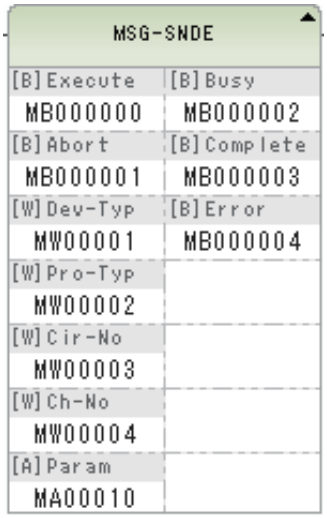
MSG-SND 指令是 MP2000 互換規格的指令。以下的暫存器存取上有所不同。

暫存器名稱	MSG-SNDE 的存取範圍		MSG-SND 的存取範圍	
系統暫存器	SW00000 ~ 65534	RW	–	–
保持暫存器	MW0000000 ~ 1048575	RW	MW0000000 ~ 0065534	RW
資料暫存器	GW0000000 ~ 2097151	RW	–	–
輸入暫存器	IW00000 ~ 17FFF	R	IW00000 ~ 0FFFF	R
輸出暫存器	OW00000 ~ 17FFF	RW	–	–

(註)R：只可讀取、RW：可讀取 / 寫入

格式

所採用之格式如下。



圖示　：MSG-SNDE

按鍵輸入：MSG-SNDE

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
執行傳送指令 (Execute)	○	×	×	×	×	×	×	×	×
強制中斷傳送指令 (Abort)	○	×	×	×	×	×	×	×	×
傳送裝置類型 (Dev-Typ)	×	○	×	×	×	×	×	○	○
傳送通訊協定 (Pro-Typ)	×	○	×	×	×	×	×	○	○
線路編號 (Cir-No)	×	○	×	×	×	×	×	○	○
傳送緩衝頻道編號 (Ch-No)	×	○	×	×	×	×	×	○	○
參數清單起始位址 (Param)	×	×	×	×	×	×	○*1	×	×
處理中 (Busy)	○*2	×	×	×	×	×	×	×	×
處理完成 (Complete)	○*2	×	×	×	×	×	×	×	×
發生錯誤 (Error)	○*2	×	×	×	×	×	×	×	×

*1. 僅 M、G、D 暫存器
*2. C、# 暫存器除外

關於輸出入項目、詳細的參數內容、程式範例，請參照以下的手冊。

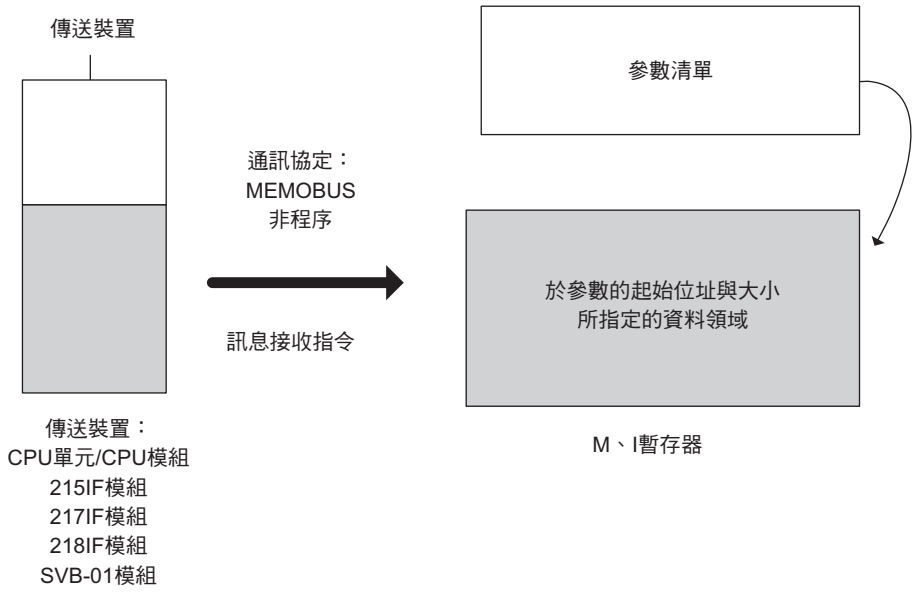
MP3000 系列　通訊功能　使用手冊 (資料編號：YTWMNCO-15003A)

接收訊息 (MSG-RCV)

以傳送裝置類型所指定之線路上的目的端來接收訊息。請在接收完成前 (訊息接收結束啟動後) 將訊息接收指令保持在啟動的狀態。支援以下的傳送裝置與通訊協定。

傳送裝置：CPU 單元 /CPU 模組、215IF 模組、217IF 模組、218IF 模組、SVB-01 模組

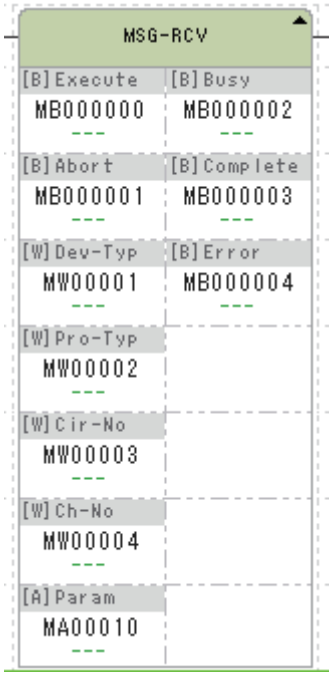
通訊協定：MEMOBUS 通訊，非程序



(註)傳送結束後，訊息接收結束便會啟動。
在此之前請將訊息接收指令保持在啟動的狀態。

格式

所採用之格式如下。



圖示　：　MSG-RCV

按鍵輸入：　MSG-RCV

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
執行接收指令 (Execute)	○	×	×	×	×	×	×	×	×
強制中斷接收指令 (Abort)	○	×	×	×	×	×	×	×	×
傳送裝置類型 (Dev-Typ)	×	○	×	×	×	×	×	○	○
傳送通訊協定 (Pro-Typ)	×	○	×	×	×	×	×	○	○
線路編號 (Cir-No)	×	○	×	×	×	×	×	○	○
傳送緩衝頻道編號 (Ch-No)	×	○	×	×	×	×	×	○	○
參數清單起始位址 (Param)	×	×	×	×	×	×	○*1	×	×
處理中 (Busy)	○*2	×	×	×	×	×	×	×	×
處理完成 (Complete)	○*2	×	×	×	×	×	×	×	×
發生錯誤 (Error)	○*2	×	×	×	×	×	×	×	×

*1. 僅 M、G、D 暫存器
*2. C、# 暫存器除外

關於輸出入項目、詳細的參數內容、程式範例，請參照以下的手冊。

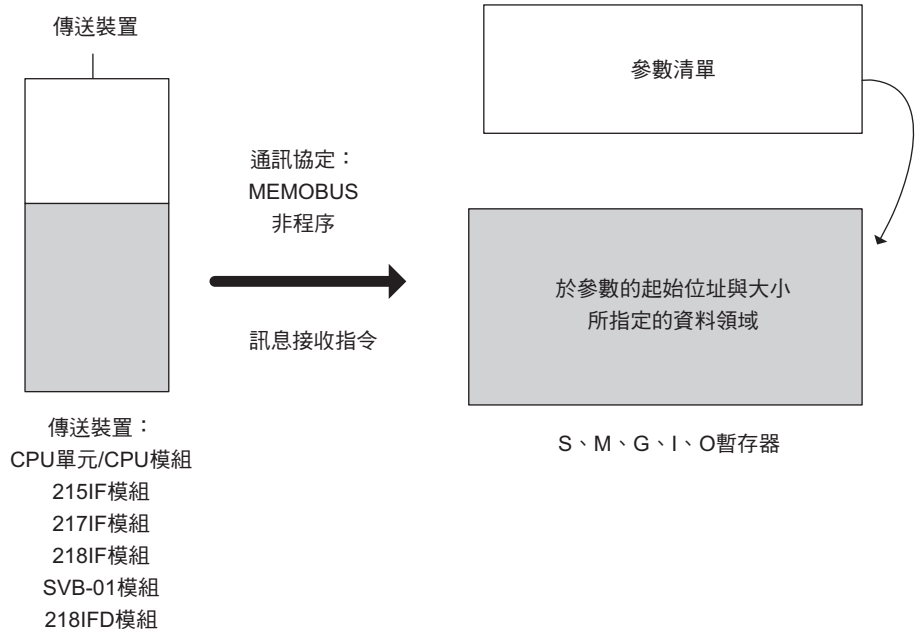
MP2000 系列　使用者手冊　階梯圖程式篇 (資料編號：SI-C887-1.2)

接收訊息 (擴充) (MSG-RCVE)

以傳送裝置類型所指定的線路上的目的端來接收訊息。請在接收完成前 (訊息接收結束啟動後) 將訊息接收指令保持在啟動的狀態。支援以下的傳送裝置與通訊協定。

傳送裝置：CPU 單元 /CPU 模組、215IF 模組、217IF 模組、218IF 模組、SVB-01 模組、218IFD 模組

通訊協定：MEMOBUS 通訊，非程序



(註)傳送結束後，訊息接收結束便會啟動。
在此之前請將訊息接收指令保持在啟動的狀態。

雖然基本動作與 MSG-RCV 指令相同，但一般情況下請使用支援 MP3000 規格的 MSG-RCVE 指令。

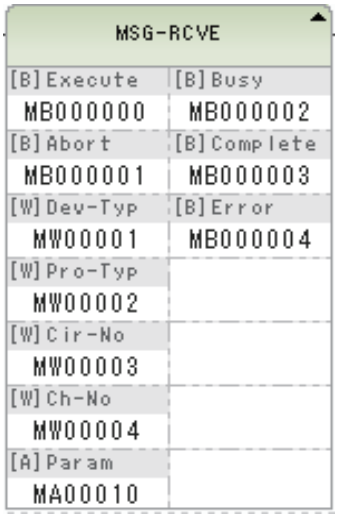
MSG-RCV 指令是 MP2000 互換規格的指令。與以下的暫存器存取並不相同。

暫存器名稱	MSG-RCVE 的存取範圍		MSG-RCV 的存取範圍	
系統暫存器	SW00000 ~ 65534	RW	–	–
保持暫存器	MW0000000 ~ 1048575	RW	MW0000000 ~ 0065534	RW
資料暫存器	GW0000000 ~ 2097151	RW	–	–
輸入暫存器	IW00000 ~ 17FFF	R	IW00000 ~ 0FFFF	R
輸出暫存器	OW00000 ~ 17FFF	RW	–	–

(註)R：只可讀取、RW：可讀取 / 寫入

格式

所採用之格式如下。



圖示　：　MSG -RCVE

按鍵輸入：　MSG-RCVE

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
執行接收指令 (Execute)	○	×	×	×	×	×	×	×	×
強制中斷接收指令 (Abort)	○	×	×	×	×	×	×	×	×
傳送裝置類型 (Dev-Typ)	×	○	×	×	×	×	×	○	○
傳送通訊協定 (Pro-Typ)	×	○	×	×	×	×	×	○	○
線路編號 (Cir-No)	×	○	×	×	×	×	×	○	○
傳送緩衝頻道編號 (Ch-No)	×	○	×	×	×	×	×	○	○
參數清單起始位址 (Param)	×	×	×	×	×	×	○*1	×	×
處理中 (Busy)	○*2	×	×	×	×	×	×	×	×
處理完成 (Complete)	○*2	×	×	×	×	×	×	×	×
發生錯誤 (Error)	○*2	×	×	×	×	×	×	×	×

*1. 僅 M、G、D 暫存器
*2. C、# 暫存器除外

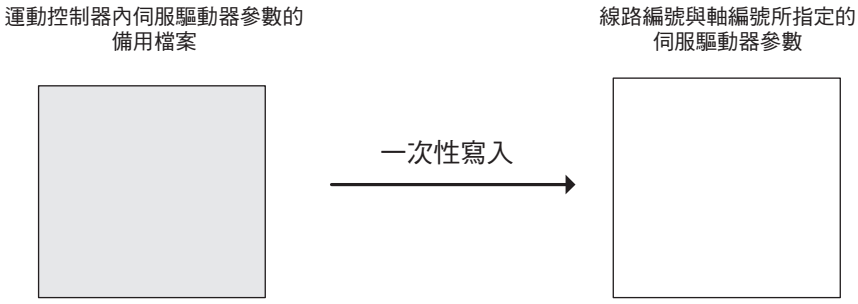
關於輸出入項目、詳細的參數內容、程式範例，請參照以下的手冊。

MP3000 系列　通訊功能　使用手冊 (資料編號：YTWMNCO-15003A)

將參數寫入伺服驅動器 (MLNK-SVW)

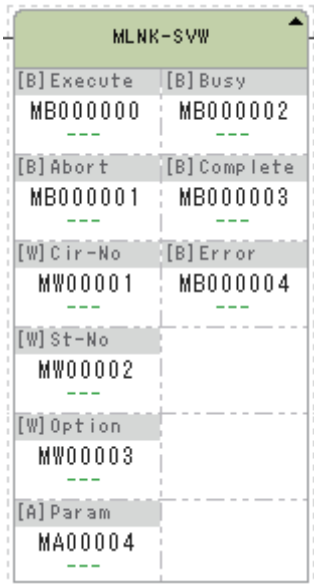
將作為伺服驅動器參數的備用檔案，保存於運動控制器的參數一次性寫入至所指定的線路編號與軸編號的伺服驅動器內。

透過使用本函數，可在進行伺服驅動器交換時，就算沒有 MPE720 等的工具也能夠只利用階梯圖處理來寫入伺服驅動器的參數。



格式

所採用之格式如下。



圖示　：　MLNK -SVW

按鍵輸入：MLNK-SVW

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
寫入指令 (Execute)	○	×	×	×	×	×	×	×	×
寫入處理強制中斷指令 (Abort)	○	×	×	×	×	×	×	×	×
線路編號 (Cir-No)	×	○	×	×	×	×	×	×	×
軸編號 (St-No)	×	○	×	×	×	×	×	×	×
選配設定 (Optsion)	×	○	×	×	×	×	×	×	×
參數資料表起始位址 (Param)	×	×	×	×	×	×	○*1	×	×
寫入中 (Busy)	○*2	×	×	×	×	×	×	×	×
寫入結束 (Complete)	○*2	×	×	×	×	×	×	×	×
錯誤 (Error)	○*2	×	×	×	×	×	×	×	×

*1. 僅 M、D 暫存器
*2. C、# 暫存器除外

各輸出入項目的內容說明如下。

輸出輸入項目	內容	輸出入
寫入指令 (Execute)	啟動後開始寫入伺服驅動器參數。	IN
寫入處理強制中斷指令 (Abort)	啟動後強制中斷寫入處理。	IN
線路編號 (Cir-No)	目標線路編號 (1 ~ 16)	IN
軸編號 (St-No)	目標軸編號 (1 ~ 16)	IN
選配設定 (Optsion)	指令選配 Bit 設定 Bit E：ID 確認啟用 / 不啟用設定　0 (啟用)、1 (不啟用) Bit F：版本確認啟用 / 不啟用設定　0 (啟用)、1 (不啟用) 未使用其他位元。即使設定也將被忽略。	IN
參數資料表起始位址 (Param)	函數使用工作的起始位址	IN
寫入中 (Busy)	伺服驅動器參數寫入時啟動。	OUT
寫入完成 (Complete)	伺服驅動器參數寫入完成的情形下僅掃描 1 次即啟動。	OUT
錯誤 (Error)	在錯誤發生時僅掃描 1 次即啟動。 (錯誤內容將被輸出至 PARAM00、PARAM01。)	OUT

選配設定的內容如下。

Bit	內容
0 ~ D	未使用 (設定將被忽略。)
E	ID 確認啟用 / 不啟用設定 (0：啟用 /1：不啟用) 原檔案的 ID 資訊與寫入目標的 ID 資訊不同時，將發生錯誤 (ID 資訊不一致)。 若選擇「1 (不啟用)」，該錯誤將不會被偵測出，因此可執行寫入處理。 使用於針對步進馬達驅動器進行寫入時。 若選擇「1 (不啟用)」，由於不會進行機種資訊的確認，因此在寫入不同機種的參數時可能會發生問題，請小心注意。 使用離線狀態下編輯或保存的伺服驅動器參數檔案來執行函數時，也同樣會發生錯誤 (ID 資訊不一致)。此時若經確認沒有問題，請選擇「1 (不啟用)」。
F	版本確認啟用 / 不啟用設定 (0：啟用 /1：不啟用) 原伺服驅動器 (通訊 I/F) 的版本與寫入目標的版本不同時，將會發生錯誤 (版本不一致)。 若版本不同，伺服驅動器參數的項目與設定範圍等有可能會不同。確認後若沒有問題，請選擇「1 (不啟用)」。即可執行寫入處理。 使用離線狀態下編輯或保存的伺服驅動器參數檔案來執行函數時，也同樣會發生錯誤 (版本資訊不一致)。此時若經確認沒有問題，請選擇「1 (不啟用)」。

◆ 使用函數工作的詳細內容

說明使用函數工作的詳細內容。參數 No. 係為從起始位址開始的字元偏移數。

範例　起始位址若為 MA00100 時，設定 PARAM 05 時請於 MW00105 設定數值。

參數 No.	IN/OUT	內容
PARAM 00	OUT	處理結果
PARAM 01	OUT	錯誤碼
PARAM 02	OUT	Cir-No 複製
PARAM 03	OUT	ST-No 複製
PARAM 04	SYS	系統用 #1
PARAM 05	SYS	系統用 #2
PARAM 06	SYS	系統用 #3

■ 處理結果 (PARAM 00)

在伺服驅動器上輸出處理結果。

- 0000H：處理中 (Busy)
- 1000H：處理完成 (Complete)
- 8□□□H：發生錯誤 (Error)

  錯誤的分類如下。

錯誤碼	內容
8100H	預約
8200H	位址設定錯誤 (資料位址設定於範圍以外。)
8300H	預約
8400H	線路編號設定錯誤 (線路編號設定於範圍以外。)
8500H	預約
8600H	軸編號設定錯誤 (軸編號設定於範圍以外。)
8700H	預約
8800H	通訊 I/F 任務錯誤 (從通訊 I/F 任務得到錯誤回應。)
8900H	預約
8A00H	函數執行重複錯誤 (同時有多個 MLNK-SVW 函數執行中。)

■ 錯誤碼 (PARAM 01)

輸出通訊 I/F 任務的錯誤碼。僅在處理結果 (PARAM00) 為 8800H 時啟用。

錯誤碼	內容
0000H	預約
0001H	無伺服驅動器參數的備份檔案
0002H	備份檔案不正確
0003H	ID 資訊不一致
0004H	版本資訊不一致
0005H	模組異常
0006H	伺服驅動器控制器部分的指令重複錯誤
0007H	通訊異常
0008H	定義以外的指令
0009H	不正確的參數
000AH	系統內部錯誤

■ Cir-No 複製 (PARAM 02)

輸入資料 Cir-No 的複製。

■ St-No 複製 (PARAM 03)

輸入資料 St-No 的複製。

■ 系統用 #1 (PARAM 04)

於系統中使用。於使用者程式啟動電源後，在最初的掃描設定為 「0000H」。除此以外的時間點請不要進行變更。

■ 系統用 #2 (PARAM 05)

於系統中使用。於使用者程式啟動電源後，在最初的掃描設定為 「0000H」。除此以外的時間點請不要進行變更。

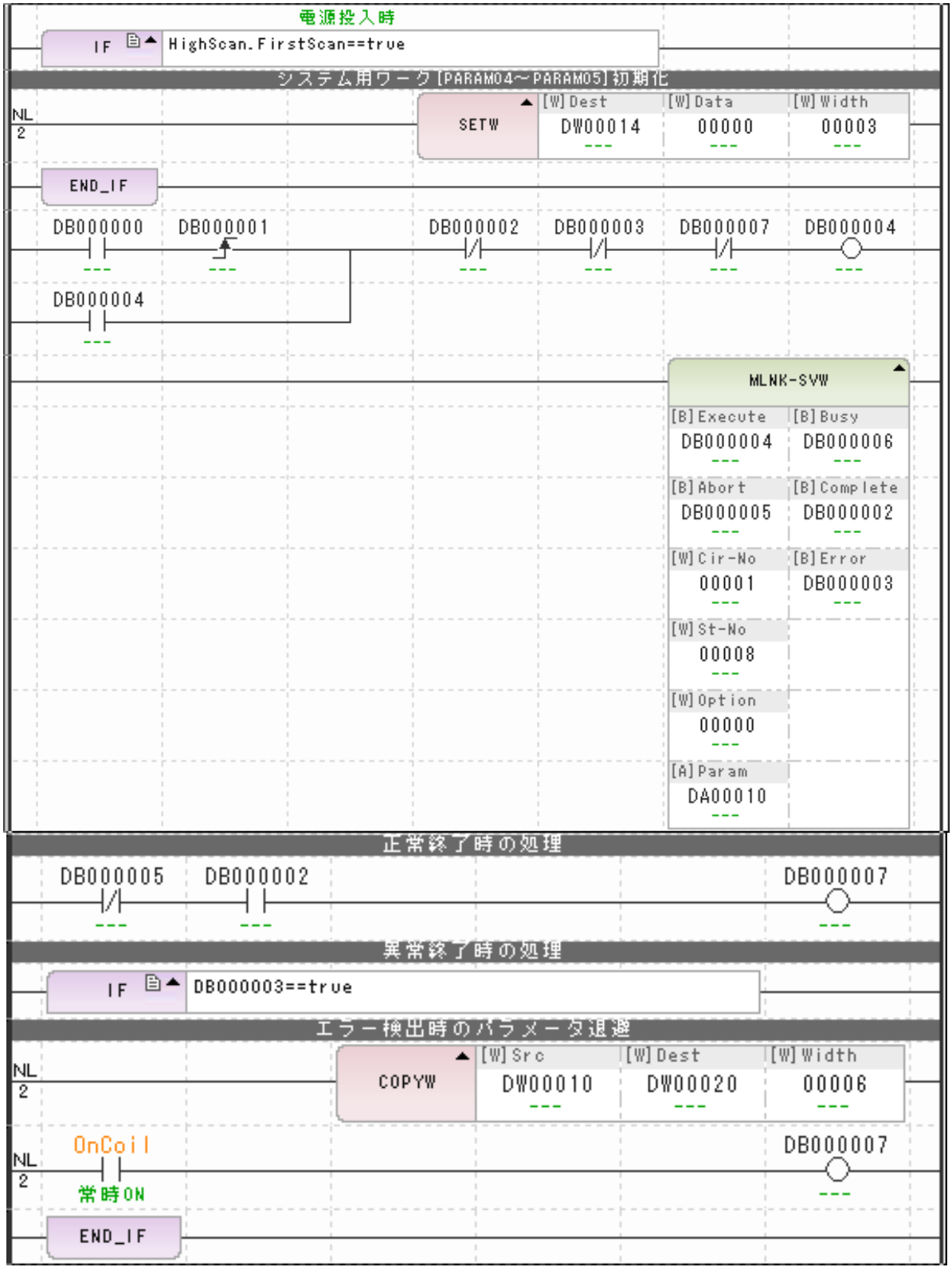
■ 系統用 #3 (PARAM 06)

於系統中使用。於使用者程式啟動電源後，在最初的掃描設定為 「0000H」。除此以外的時間點請不要進行變更。

程式範例

以下所示為將參數寫入伺服驅動器時的程式範例。

若運動控制器上存在有伺服驅動器參數的備份檔案，可透過啟動 DB000000，在模組結構定義下 MECHATROLINK 的線路編號為 1，並且可針對在 MECHATROLINK 詳細定義下的 ST#8 伺服驅動器參數，執行單次寫入。



資料寫入運轉暫存器 (MOTREG-W)

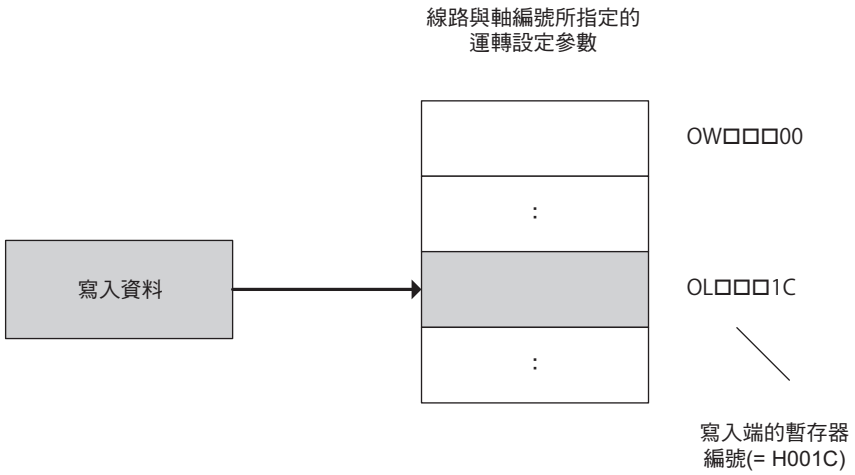
本系統函數可用來存取所指定的動轉暫存器。

指定線路編號、軸編號以及暫存器編號，並於運轉暫存器寫入數值。

適用對象為運轉設定的參數。

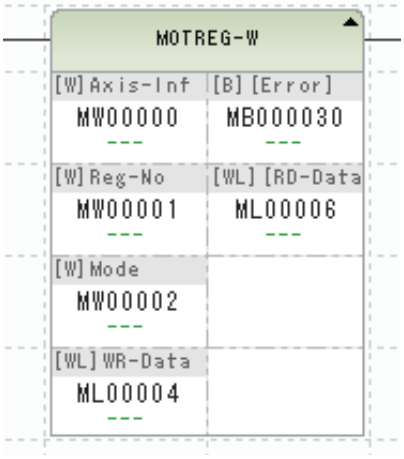
註記

將相同的運轉設定參數儲存至線路編號與軸編號不同的多個軸時啟用。使用 STORE 指令或 EXPRESSION 指令寫入時，需依照線路編號與軸編號等來考量偏移。



格式

所採用之格式如下。



圖示　：　MOT REG-W

按鍵輸入：MOTREG-W

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
軸資訊 (Axis-Inf)	×	○	×	×	×	×	×	×	×
暫存器編號 (Reg-No)	×	○	×	×	×	×	×	×	×
模式 (Mode)	×	○	×	×	×	×	×	×	×
寫入資料 (WR-Data)	×	○	○	×	×	×	×	×	×
錯誤 (Error)	○*	×	×	×	×	×	×	×	×
所讀取之資料 (RD-Data)	×	○*	○*	×	×	×	×	×	×

* C、# 暫存器除外，可省略

各輸出入項目的內容說明如下。

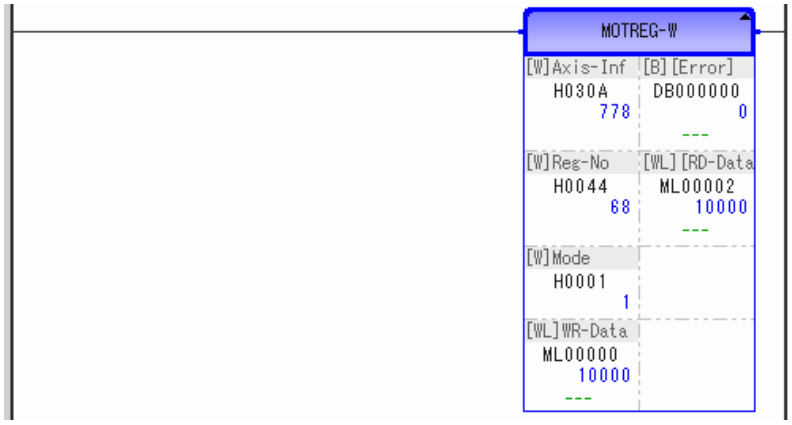
輸出輸入項目	內容	輸出入
軸資訊 (Axis-Inf)	線路編號、軸編號 高位元組：線路編號 (01 ~ 10) (hex) 低位元組：軸編號 (01 ~ 10) (hex)	IN
暫存器編號 (Reg-No)	指定整數型時：0000 ~ 007F (hex) 指定長整數型時：0000 ~ 007E (hex)	IN
模式 (Mode)	存取類別、存取大小 ・高位元組：存取類別 0：於指定暫存器寫入 WR-Data 1：將指定暫存器與 WR-Data 的邏輯和寫入指定暫存器 2：將指定暫存器與 WR-Data 的邏輯積寫入指定暫存器 上述以外：將 WR-Data 寫入指定暫存器。 ・低位元組：存取大小 0：整數型 1：長整數型 上述以外：整數型	IN
寫入資料 (WR-Data)	若模式的存取大小為整數型，以及輸入資料類型為長整數型時，則僅可使用低階字元。	IN
錯誤 (Error)	錯誤發生 (錯誤發生時啟動) 原因 由於線路編號、軸編號、暫存器編號超過範圍，或沒有模組，因此無法進行暫存器的寫入 / 讀取。 錯誤發生時，於 RD-Data 輸出 0。	OUT
所讀取之資料 (RD-Data)	寫入後變為讀取資料。 指定整數型時，輸出符號擴充的資料。	OUT

程式範例

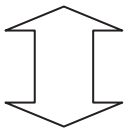
以下所示為將寫入資料 (ML00000) 的數值寫入至線路 3 上的軸 10 的 STEP 移動量 (OL□□□44) 時的程式範例。

請進行以下設定。

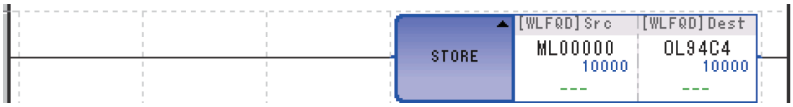
- 軸資訊 = H030A (線路 3、軸 10)
- 暫存器編號 = H0044
- 模式 = H0001 (長整數型)



即使直接指定暫存器編號並以 STORE 指令進行存取，也會得到相同的結果。



等價



讀取運轉暫存器資料 (MOTREG-R)

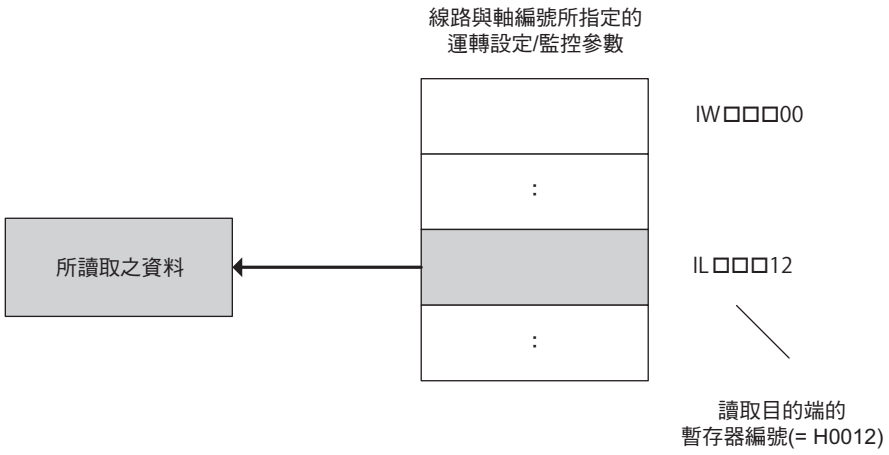
本系統函數可用來存取所指定的運轉暫存器。

指定線路編號、軸編號與暫存器編號，並讀取運轉暫存器的數值。

適用對象為運轉設定參數與運轉監控參數。

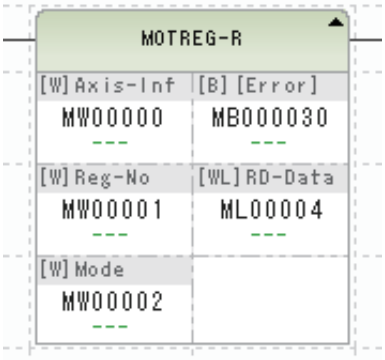
註記

從線路編號與軸編號不同的多個軸來讀取相同的運轉設定參數時啟用。使用 STORE 指令或 EXPRESSION 指令等讀取時，需依照線路編號與軸編號等來考量偏移。



格式

所採用之格式如下。



圖示　：MOT REG-R

按鍵輸入：MOTREG-R

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
軸資訊 (Axis-Inf)	×	○	×	×	×	×	×	×	×
暫存器編號 (Reg-No)	×	○	×	×	×	×	×	×	×
模式 (Mode)	×	○	×	×	×	×	×	×	×
錯誤 (Error)	○*	×	×	×	×	×	×	×	×
所讀取之資料 (RD-Data)	×	○*	○*	×	×	×	×	×	×

* C、# 暫存器除外，可省略

各輸出入項目的內容說明如下。

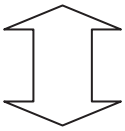
輸出輸入項目	內容	輸出入
軸資訊 (Axis-Inf)	線路編號、軸編號 高位元組：線路編號 (01 ~ 10) (hex) 低位元組：軸編號 (01 ~ 10) (hex)	IN
暫存器編號 (Reg-No)	指定整數型時：0000 ~ 007F (hex) 指定長整數型時：0000 ~ 007E (hex)	IN
模式 (Mode)	暫存器類型、存取大小 ・高位元組：暫存器類型 0：I 暫存器 (運轉監控參數) 1：O 暫存器 (運轉設定參數) 上述以外：I 暫存器 ・低位元組：存取大小 0：整數型 1：長整數型 上述以外：整數型	IN
錯誤 (Error)	錯誤發生 (錯誤發生時啟動) 原因 由於線路編號、軸編號、暫存器編號超過範圍，或沒有模組，因此無法進行暫存器的寫入 / 讀取。 錯誤發生時，於 RD-Data 輸出 0。	OUT
所讀取之資料 (RD-Data)	指定整數型時，輸出符號擴充的資料。	OUT

程式範例

以下所示為讀取線路 1 上的軸 2 機械座標系反饋位置 (IL8096) 時的程式範例。

請進行以下設定。

- 軸資訊 =H0102 (線路 1、軸 2)
- 暫存器編號 = H0016
- 模式 = H0001 (運轉監控參數、長整數型)



等價

即使直接指定暫存器編號並以 STORE 指令存取至 DL00002，也會得到相同的結果。

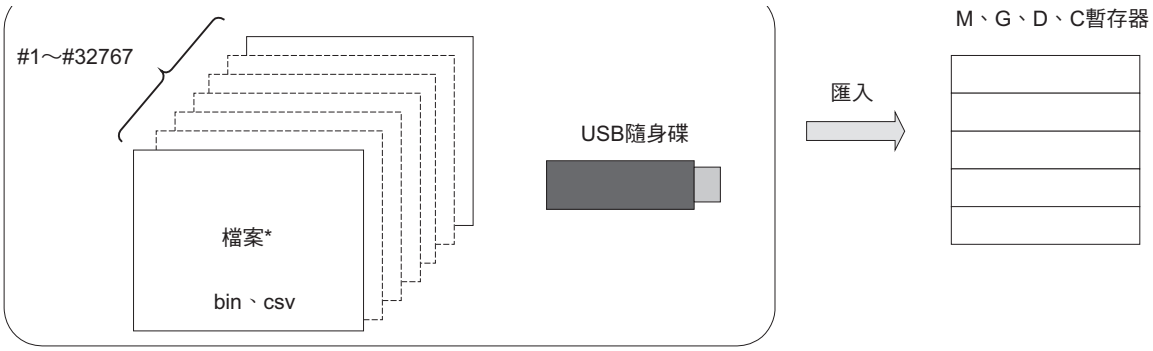
匯入 (IMPORT/IMPORTL)

從 USB 隨身碟匯入暫存器的資料，並於暫存器中展開。

匯入檔案的格式可從二進制資料 (bin) 與 CSV 資料 (csv) 中做選擇。

匯入端的暫存器上可指定 M 暫存器、G 暫存器、D 暫存器或 C 暫存器。

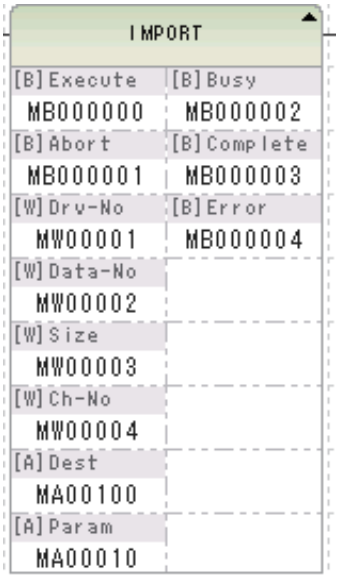
IMPORT/IMPORTL 指令可同時並列執行 2 個指令。



* 從 USB 隨身碟中的以下檔案匯入。
￥MP_DATA￥DAT00001.BIN (CSV)
:
￥MP_DATA￥DAT32767.BIN (CSV)

格式

所採用之格式如下。



圖示　：IM PORT インポート　IM PORTL インポート(拡張)

按鍵輸入：IMPORT/IMPORTL

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
匯入指令 (Execute)	○	×	×	×	×	×	×	×	×
匯入強制終止指令 (Abort)	○	×	×	×	×	×	×	×	×
驅動器編號 (Drv-No)	×	○	×	×	×	×	×	○	○
資料編號 (Data-No)	×	○	×	×	×	×	×	○	○
傳送字元數 (Size)	×	○*1	○*2	×	×	×	×	○	○
並列執行的頻道編號 (Ch-No)	×	○	×	×	×	×	×	○	○
傳送目的端暫存器編號 (Dest)	×	×	×	×	×	×	○	×	×
參數清單起始位址 (Param)	×	×	×	×	×	×	○	×	×
匯入執行中 (Busy)	○	×	×	×	×	×	×	×	×
匯入執行結束 (Complete)	○	×	×	×	×	×	×	×	×
錯誤 (Error)	○	×	×	×	×	×	×	×	×

*1. IMPORT 指令時
*2. IMPORTL 指令時

各輸出入項目的內容說明如下。

輸出輸入項目	內容	輸出入
匯入指令 (Execute)	啟動後開始進行匯入指令。 執行指令時需為啟動狀態。	IN
匯入強制終止指令 (Abort)	啟動後強制終止匯入處理。	IN
驅動器編號 (Drv-No)	驅動器編號 (1：USB 隨身碟)	IN
資料編號 (Data-No)	資料編號 (1 ~ 32767)	IN
傳送字元數 (Size)	傳送字元數 (IMPORT:1 ~ 32767，IMPORTL:1 ~ 2147483647)	IN
並列執行的頻道編號 (Ch-No)	並列執行的頻道編號 (1、2)	IN
傳送目的端暫存器編號 (Dest)	傳送目的端暫存器編號 (MA、GA、DA、CA)	IN
參數清單起始位址 (Param)	參數清單起始位址 (MA、GA、DA)	IN
匯入執行中 (Busy)	匯入執行時啟動。	OUT
匯入執行結束 (Complete)	匯入完成後啟動。	OUT
錯誤 (Error)	於發生錯誤時啟動。	OUT

◆ 參數的詳細內容

說明參數的詳細內容。

<IMPORT>

位址	資料類型	參數 No.	IN/OUT	內容
0	W	PARAM00	OUT	處理結果
1	W	PARAM01	IN	格式類別
2	W	PARAM02	IN	CSV 檔案時檔案內的偏移行數
3	W	PARAM03	IN	檔案內資料的字元偏移
4	W	PARAM04	OUT	系統預約
5	W	PARAM05	OUT	系統預約

<IMPORTL>

位址	資料類型	參數 No.	IN/OUT	內容
0	W	PARAM00	OUT	處理結果
1	W	PARAM01	IN	格式類別
2	L	PARAM02	IN	CSV 檔案時檔案內的偏移行數
4	L	PARAM03	IN	檔案內資料的字元偏移
6	W	PARAM04	OUT	系統預約
7	W	PARAM05	OUT	系統預約

■ 處理結果 (PARAM00)

回報 IMPORT/IMPORTL 指令的處理結果。

· 00□□H：處理中 (Busy)

· 10□□H：處理完成 (Complete)

· 8□□□H：發生錯誤 (Error)

錯誤的分類如下。

錯誤碼	內容
8101H	範圍外錯誤 (驅動器編號)
8102H	範圍外錯誤 (資料編號)
8103H	範圍外錯誤 (傳送數)
8104H	範圍外錯誤 (並列執行的頻道編號)
8105H	範圍外錯誤 (傳送目的端 / 來源端暫存器編號)
8106H	範圍外錯誤 (格式類別)
8107H	範圍外錯誤 (開啟類別)
8108H	範圍外錯誤 (檔案內資料的字元偏移)
8109H	範圍外錯誤 (參數清單起始位址)
810AH	範圍外錯誤 (檔案內的偏移行數)
8201H	未裝上 USB 隨身碟
8202H	檔案開啟錯誤
8203H	檔案搜尋錯誤
8204H	檔案寫入錯誤
8205H	檔案讀取錯誤
8206H	檔案關閉錯誤
8301H	超過檔案可處理數量，因此無法處理
8302H	檔案輸出入逾時

■ 格式類別 (PARAM01)

設定匯入檔案的格式。

匯入 MPE720 的暫存器表單資料時，請設定 2。

1：從二進制資料 (DAT□□□□□.BIN) 匯入。

□□□□□ 所填入的是以資料編號 (Data-No) 所設定的數值。

2：從 CSV 檔案 (DAT□□□□□.CSV) 匯入。

□□□□□ 所填入的是以資料編號 (Data-No) 所設定的數值。

■ CSV 檔案時檔案內的偏移行數 (PARAM02)

若為 CSV 檔案，則設定偏移的行數。

匯入 MPE720 的暫存器表單資料時，請指定 2。

若為二進制檔案，該參數將會被忽略。

■ 檔案內資料的字元偏移 (PARAM03)

設定偏移的字元數。

■ 系統預約 (PARAM04)

系統所使用的工作領域。

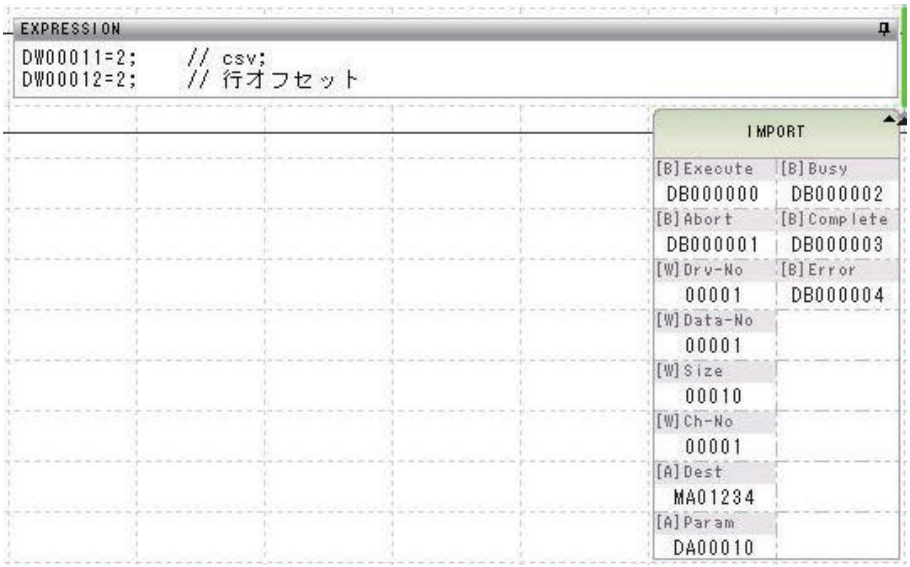
■ 系統預約 (PARAM05)

系統所使用的工作領域。

程式範例

以下所示為將 MPE720 的暫存器表單資料匯入至 MW01234 ~ MW01243 的程式範例。關於 MPE720 的操作步驟請參照以下項目。

☞ 補充事項 (第 4-274 頁)



執行指令前

暫存器	數值
MW01233	0
MW01234	0
MW01235	0
MW01236	0
MW01237	0
MW01238	0
MW01239	0
MW01240	0
MW01241	0
MW01242	0
MW01243	0
MW01244	0

執行指令後

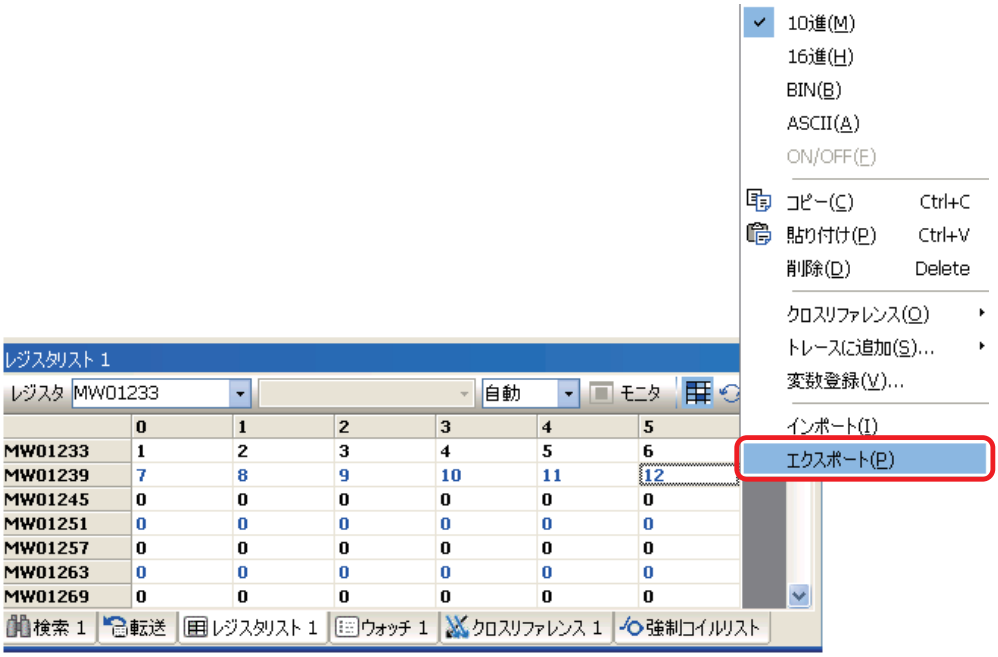
暫存器	數值
MW01233	0
MW01234	2
MW01235	3
MW01236	4
MW01237	5
MW01238	6
MW01239	7
MW01240	8
MW01241	9
MW01242	10
MW01243	11
MW01244	0

補充事項

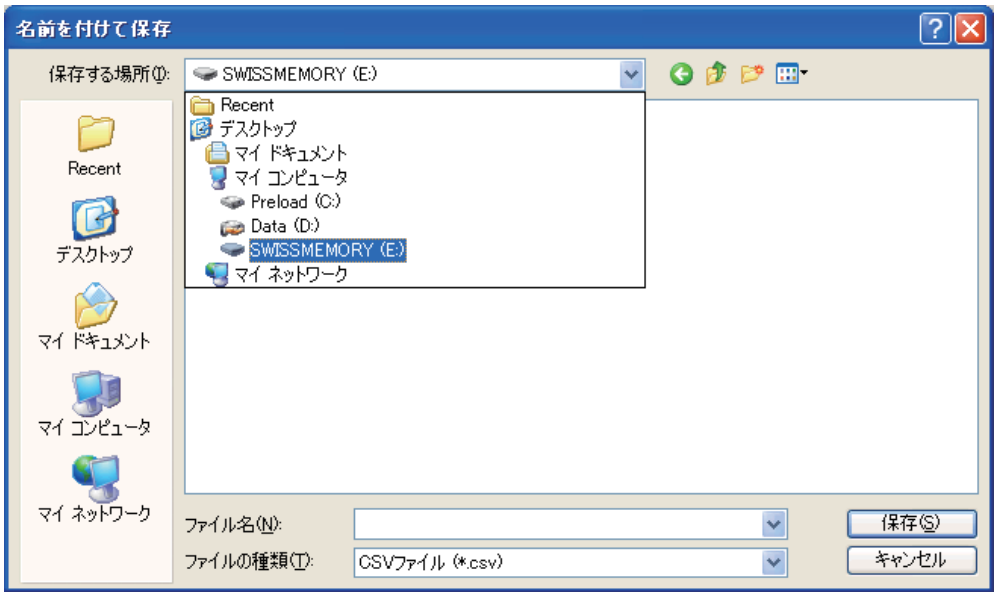
請依照以下的步驟匯出 MPE720 的暫存器表單資料。

此外，本處記載了依照上述程式範例內容的步驟。

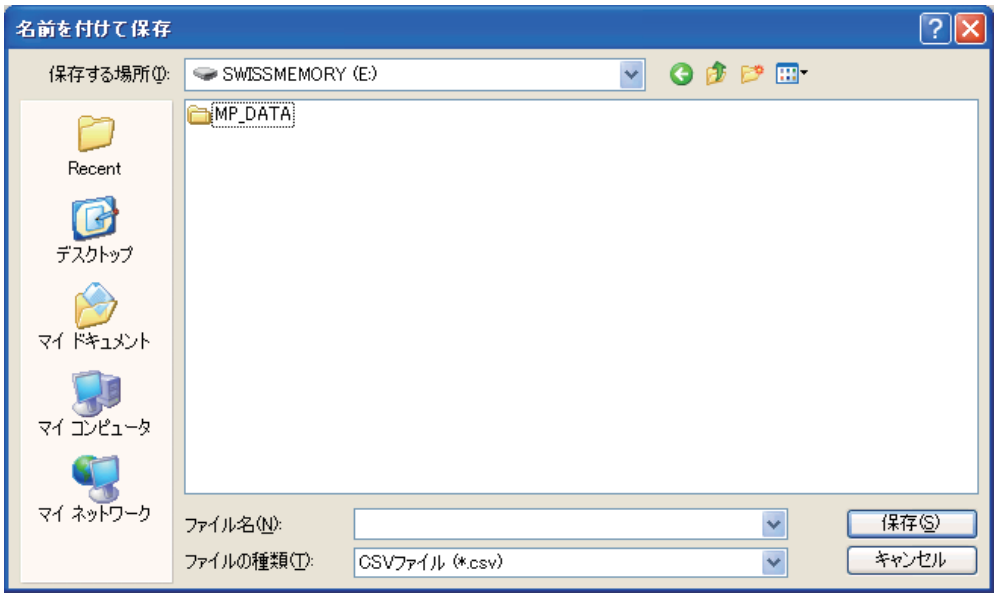
1. **將 USB 隨身碟安裝於 PC 中。**
2. **於 MPE720 顯示暫存器表單。**
3. **於暫存器表單上按右鍵，從顯示的表單中選擇 [匯出]。**



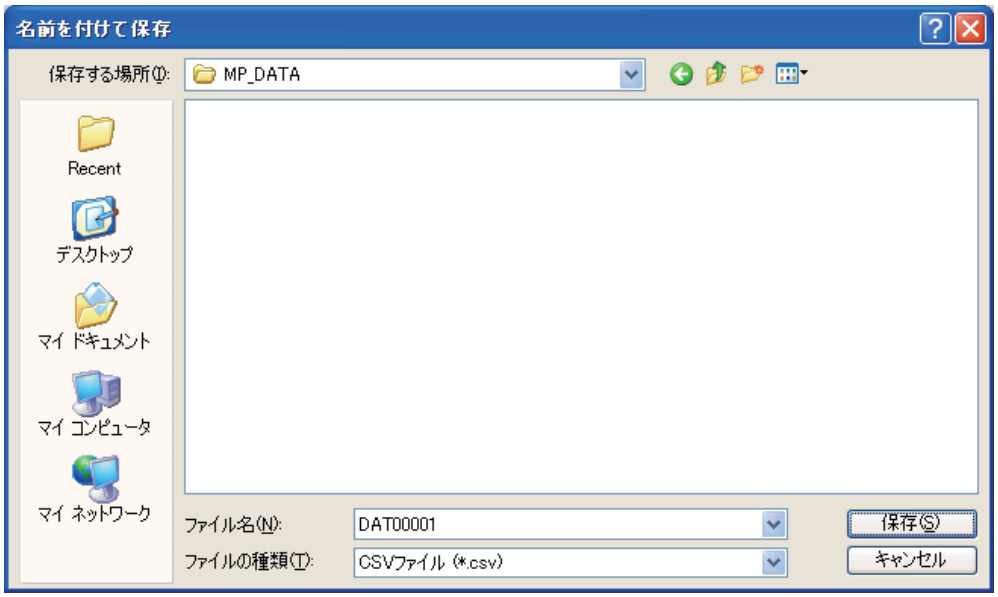
4. **選擇 USB 隨身碟的驅動器代號。**



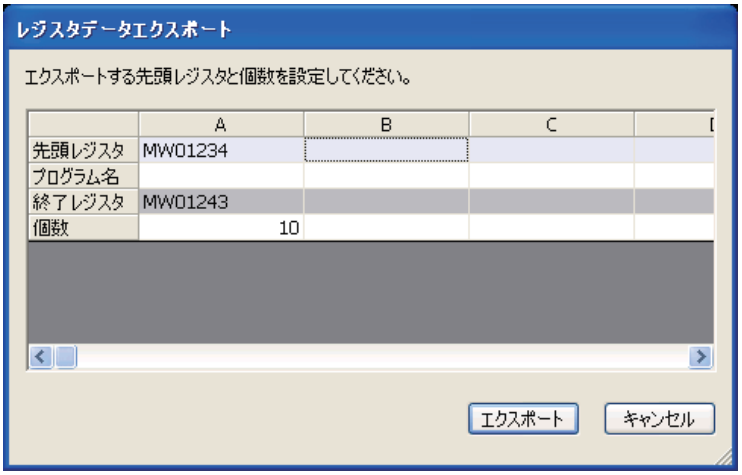
5. **點選 [MP_DATA] 資料夾。**
若 [MP_DATA] 資料夾不存在，請製作資料夾。



6. **輸入「DAT00001」於 [檔案名] 窗格，並按下 [儲存] 鍵。**



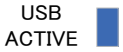
7. 輸入「**MW01234**」至 **[** 起始暫存器 **]** ，再輸入「**10**」至 **[** 數量 **]**，並按下 **[** 匯出 **]** 鍵。



8. 將 **USB** 隨身碟從 **PC** 取下。

9. 將 **USB** 隨身碟安裝至運動控制器。

10. 待 **USB ACCESS LED** 亮燈。



11. 製作上述程式範例的程式。

12. 執行 **IMPORT/IMPORTL** 指令。

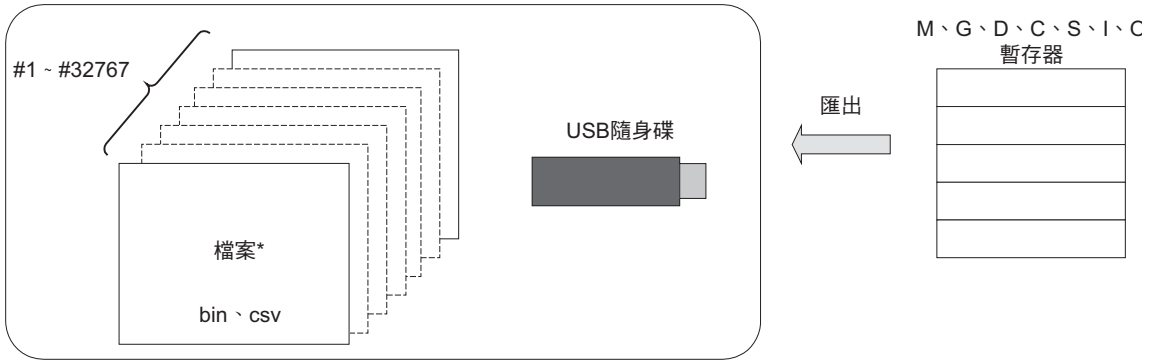
匯出 (EXPORT/EXPORTL)

將暫存器資料匯出至 USB 隨身碟。

匯出檔案的格式可從二進制資料 (bin) 與 CSV 資料 (csv) 中做選擇。

匯出來源端的暫存器上可指定 M 暫存器、G 暫存器、D 暫存器、C 暫存器、S 暫存器、I 暫存器或 O 暫存器。

EXPORT/EXPORTL 指令可同時並列執行 2 個指令。



* 使用以下的檔名匯出至 USB 隨身碟。
￥MP_DATA￥DAT00001.BIN (CSV)
:
￥MP_DATA￥DAT32767.BIN (CSV)

格式

所採用之格式如下。

EXPORT	
[B]Execute	[B]Busy
MB000000	MB000002
[B]Abort	[B]Complete
MB000001	MB000003
[W]Drv-No	[B]Error
MW00001	MB000004
[W]Data-No	
MW00002	
[W]Size	
MW00003	
[W]Ch-No	
MW00004	
[A]Src	
MA00100	
[A]Str	
MA00200	
[A]Param	
MA00010	

圖示 ： EXPORT エクスポート　EXPORTL エクスポート(拡張)

按鍵輸入： EXPORT/EXPORTL

輸出輸入項目	適用之資料類型								
	B	W	L	Q	F	D	A	索引	常數
匯出指令 (Execute)	○	×	×	×	×	×	×	×	×
匯出強制終止指令 (Abort)	○	×	×	×	×	×	×	×	×
驅動器編號 (Drv-No)	×	○	×	×	×	×	×	○	○
資料編號 (Data-No)	×	○	×	×	×	×	×	○	○
傳送字元數 (Size)	×	○*1	○*2	×	×	×	×	○	○
並列執行的頻道編號 (Ch-No)	×	○	×	×	×	×	×	○	○
傳送來源端暫存器編號 (Src)	×	×	×	×	×	×	○	×	×
文字列輸出用的暫存器編號 (Str)	×	×	×	×	×	×	○	×	×
參數清單起始位址 (Param)	×	×	×	×	×	×	○	×	×
匯出執行中 (Busy)	○	×	×	×	×	×	×	×	×
匯出執行結束 (Complete)	○	×	×	×	×	×	×	×	×
錯誤 (Error)	○	×	×	×	×	×	×	×	×

*1. EXPORT 指令時
*2. EXPORTL 指令時

各輸出入項目的內容說明如下。

輸出輸入項目	內容	輸出入
匯出指令 (Execute)	啟動後開始進行匯出指令。 執行指令時需為啟動狀態。	IN
匯出強制終止指令 (Abort)	啟動後強制終止匯出處理。	IN
驅動器編號 (Drv-No)	驅動器編號 (1：USB 隨身碟)	IN
資料編號 (Data-No)	資料編號 (1 ~ 32767)	IN
傳送字元數 (Size)	傳送字元數 (EXPORT:1 ~ 32767，EXPORTL:1 ~ 2147483647)	IN
並列執行的頻道編號 (Ch-No)	並列執行的頻道編號 (1、2)	IN
傳送來源端暫存器編號 (Src)	傳送來源端暫存器編號 (MA、GA、DA、CA、SA、IA、OA)	IN
文字列輸出用的暫存器編號 (Str)	文字列輸出用的暫存器編號 *1 *2 (MA、GA、DA、CA)	IN
參數清單起始位址 (Param)	參數清單起始位址 (MA、GA、DA)	IN
匯出執行中 (Busy)	匯出執行中啟動。	OUT
匯出執行結束 (Complete)	匯出完成後啟動。	OUT
錯誤 (Error)	於發生錯誤時啟動。	OUT

*1. CSV 檔案時啟用。若為二進制檔案，該項目將被忽略。
*2. 請務必設定顯示文字列終端的 0 (NULL 文字)。

◆ 參數的詳細內容

說明參數的詳細內容。

<EXPORT>

位址	資料類型	參數 No.	IN/OUT	內容
0	W	PARAM00	OUT	處理結果
1	W	PARAM01	IN	格式類別
2	W	PARAM02	IN	檔案開啟類別
3	W	PARAM03	IN	檔案內資料的字元偏移
4	W	PARAM04	OUT	系統預約
5	W	PARAM05	OUT	系統預約

<EXPORTL>

位址	資料類型	參數 No.	IN/OUT	內容
0	W	PARAM00	OUT	處理結果
1	W	PARAM01	IN	格式類別
2	L	PARAM02	IN	檔案開啟類別
4	L	PARAM03	IN	檔案內資料的字元偏移
6	W	PARAM04	OUT	系統預約
7	W	PARAM05	OUT	系統預約

■ 處理結果 (PARAM00)

回報 EXPORT/EXPORTL 指令的處理結果。

- 00□□H：處理中 (Busy)
- 10□□H：處理完成 (Complete)
- 8□□□H：發生錯誤 (Error)

  錯誤的分類如下。

錯誤碼	內容
8101H	範圍外錯誤 (驅動器編號)
8102H	範圍外錯誤 (資料編號)
8103H	範圍外錯誤 (傳送數)
8104H	範圍外錯誤 (並列執行的頻道編號)
8105H	範圍外錯誤 (傳送目的端 / 來源端暫存器編號)
8106H	範圍外錯誤 (格式類別)
8107H	範圍外錯誤 (開啟類別)
8108H	範圍外錯誤 (檔案內資料的字元偏移)
8109H	範圍外錯誤 (參數清單起始位址)
810BH	文字列以上 (未偵測出 NULL 文字)
8201H	未裝上 USB 隨身碟
8202H	檔案開啟錯誤
8203H	檔案搜尋錯誤
8204H	檔案寫入錯誤
8205H	檔案讀取錯誤
8206H	檔案關閉錯誤
8301H	超過檔案可處理數量，因此無法處理
8302H	檔案輸出入逾時

■ 格式類別 (PARAM01)

設定匯出檔案的格式。

匯出 MPE720 的暫存器表單資料時，請設定 2。

1：匯出至二進制檔案 (DAT□□□□□.BIN)。

□□□□□ 所填入的是以資料編號 (Data-No) 所設定的數值。

2：匯出至 CSV 檔案 (DAT□□□□□.CSV)。

□□□□□ 所填入的是以資料編號 (Data-No) 所設定的數值。

■ 檔案開啟類別 (PARAM02)

若為二進制檔案，則設定檔案開啟類別。

1：製作新的檔案並匯出。

2：匯出至既有的檔案。

選擇此類別便可變更資料的其中一部分。

若為 CSV 檔案，該參數將被忽略。

■ 檔案內資料的字元偏移 (PARAM03)

若為二進制檔案，則設定偏移的字元數。

若為 CSV 檔案，該參數將被忽略。

■ 系統預約 (PARAM04)

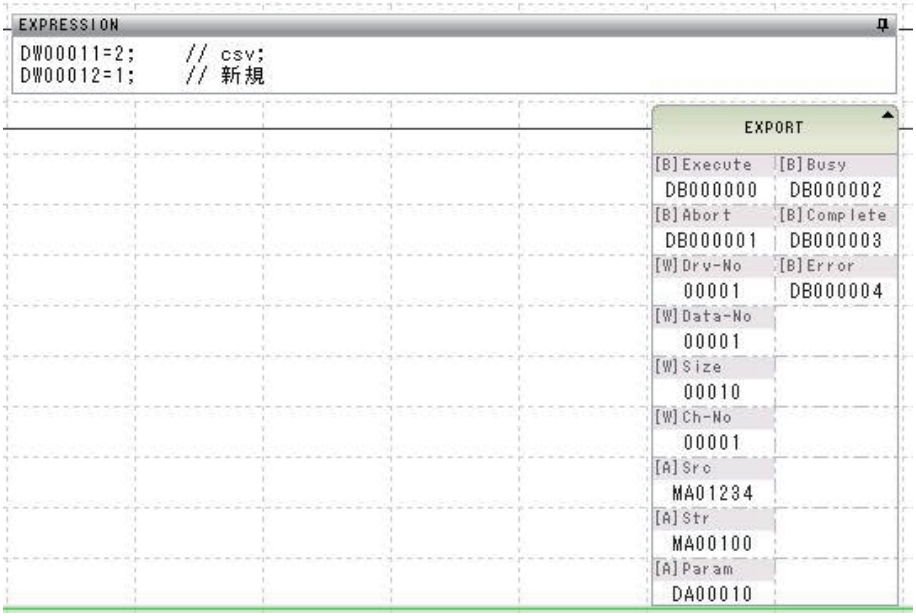
系統所使用的工作領域。

■ 系統預約 (PARAM05)

系統所使用的工作領域。

程式範例

以下所示為將 MW01234 ~ MW01243 的資料匯出至 CSV 檔案的程式範例。



暫存器資料

暫存器	數值
MW01233	1
MW01234	2
MW01235	3
MW01236	4
MW01237	5
MW01238	6
MW01239	7
MW01240	8
MW01241	9
MW01242	10
MW01243	11
MW01244	12

USB 隨身碟的檔案 (DAT00001.CSV)

文字檔的內容
MW1234↵
↵
00002↵
00003↵
00004↵
00005↵
00006↵
00007↵
00008↵
00009↵
00010↵
00011↵

開發工具 (MPE720) 功能介紹

5

本章將介紹階梯圖程式的開發工具 MPE720 的所有功能。

本章將介紹支援階梯圖程式編寫及除錯的開發工具 MPE720 Ver. 7 的以下功能。

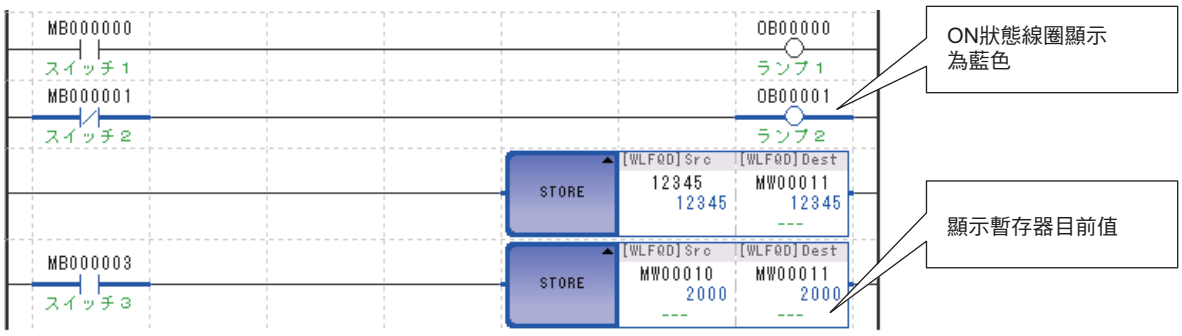
- 階梯圖程式執行中監控功能
- 檢索 / 取代
- 交互參照
- 雙重線圈
- 線圈強制開啟 / 關閉
- 參照所要叫出的程式
- 暫存器清單
- 調整面板
- 開啟 / 關閉階梯圖程式
- 檢視
- 安全功能
- 追蹤功能
- 使用運動程式

5.1 階梯圖程式執行中監控功能

此功能可用來監控所有指令的執行狀態。只要連接運動控制器，即可使用。

當繼電器輸出 ON 的指令，就會顯示為藍色。

而且，可用來顯示執行中指令參數的暫存器目前值。



5.2 檢索 / 取代

程式內的檢索 / 取代

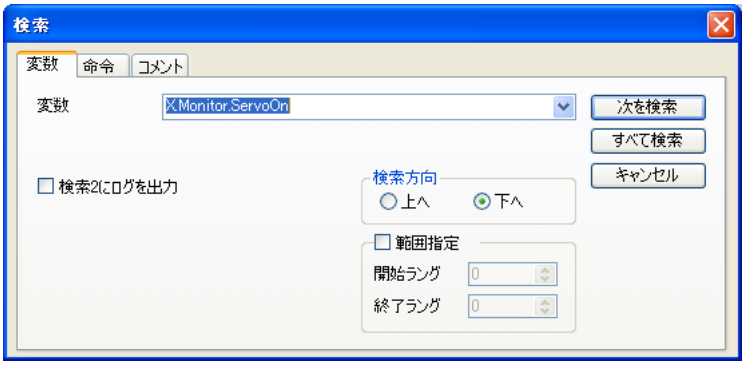
指定程式後，即可對程式內部所使用的變數、指令及註解進行檢索。亦可對暫存器及註解進行檢索 / 取代。

接下來將介紹檢索及取代之操作方法。

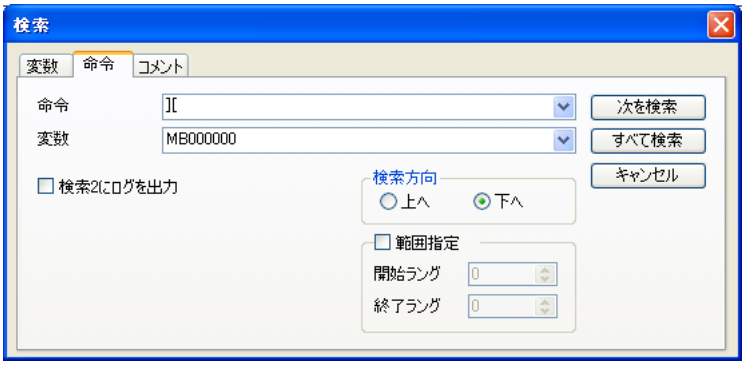
程式內的檢索

1. **讓您所要檢索的程式顯示在階梯圖編輯畫面的最上層，接著再從主選單上依序選擇 [編輯] － [檢索]。**
   畫面上將出現 [檢索] 對話框。

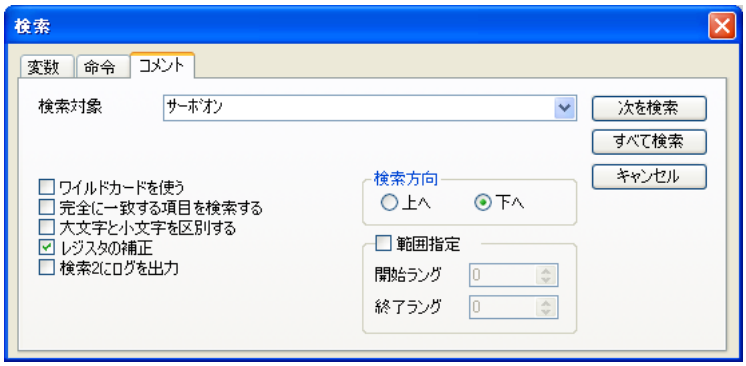
2. **選擇 [變數]/[指令]/[註解] 其中一個索引標籤，並設定您所要檢索的內容。**



[變數] 索引標籤：適用於變數及暫存器。亦可從變數子視窗當中複製及剪貼，以輸入變數。



[指令] 索引標籤：[指令] 欄可用來輸入指令名稱或指令鍵配置。
在 [指令] 欄中輸入指令後，即可叫出 [變數] 欄。此外，當您在 [指令] 欄中輸入 SEE 指令後，[變數] 欄位中就會顯示 [程式名稱]。
亦可從變數子視窗當中複製及剪貼，以輸入變數。



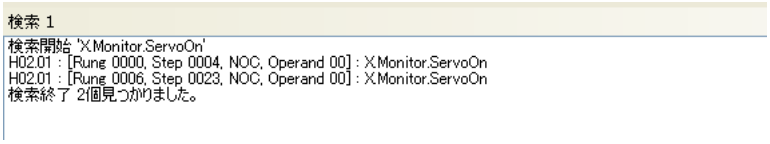
[註解] 索引標籤：適用於物件註解、指令集註解、程式註解及算式中的註解。

- 使用通用字元 (註解)：可使用通用字元 「*」與 「?」作為檢索字元。
- 檢索完全相符的項目 (註解)：檢索與註解欄字串完全相符的檢索字串。是否區分大小寫字母，請利用 [區分大小寫] 功能來進行設定。
- 區分大小寫 (註解)：設定是否區分半形英文字母的大小寫。
- 校正暫存器 (註解)：若您在檢索目標中所輸入的字串被辨識為暫存器時，必須設定是否將字串轉換為正確的暫存器標示。
- 將日誌輸出至檢索 2：只要勾選核取方塊，檢索結果就會被輸出到檢索 2 子視窗上，此時檢索 1 子視窗的輸出內容變更。若未勾選核取方塊，會輸出至檢索 1 子視窗。
- 指定範圍：只要勾選核取方塊，即可指定開始指令集及結束指令集，以設定檢索範圍。

3. 選擇 [檢索下一筆] 鍵或 [全部檢索] 鍵，就會開始進行檢索。

點擊 [檢索下一筆] 鍵，畫面上就會出現條件符合的指令物件選擇狀態。

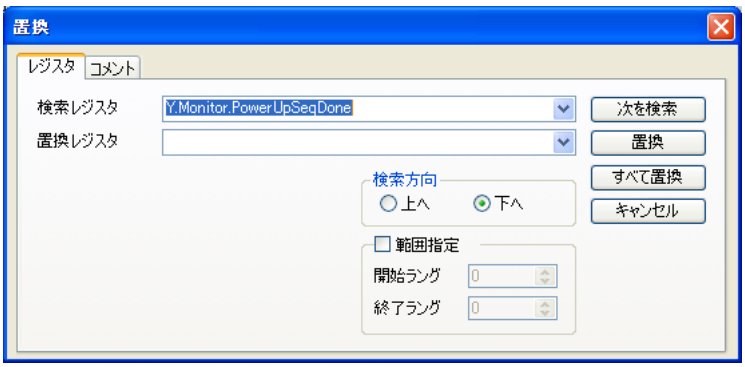
點擊 [全部檢索] 鍵後，檢索結果便會顯示在檢索 1 或檢索 2 子視窗上。



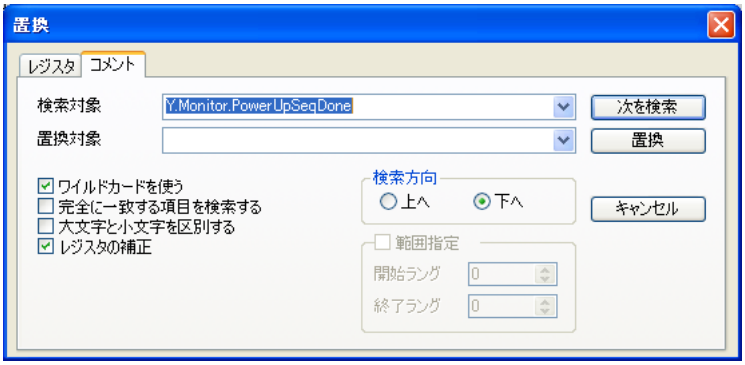
程式內的取代

1. **讓您所要檢索 / 取代的程式顯示在階梯圖編輯畫面的最上層，接著再從主選單上依序選擇 [編輯] － [取代]。**
   畫面上將出現 [取代] 對話框。

2. **請先選擇 [暫存器]、[註解] 其中任一個索引標籤，然後再設定您所要檢索及取代的內容。**



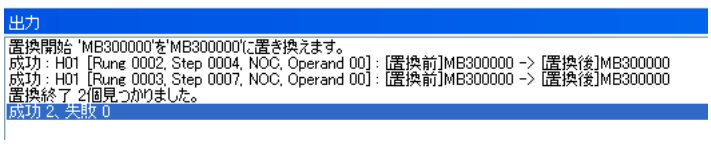
[暫存器] 索引標籤：適用於暫存器。



[註解] 索引標籤：適用於物件註解、指令集註解、程式註解及算式中的註解。

- **使用通用字元 (註解)：可使用通用字元「*」與「?」作為檢索字元。**
  (註)若在 [取代暫存器]、[取代目標] 中輸入「*」和「?」，則被取代的對象將為文字，而非通用字元。
- **指定範圍：只要勾選核取方塊，即可指定開始指令集及結束指令集，以設定檢索範圍。**
  但選擇 [註解] 索引標籤時，將無法使用。

3. **開始進行檢索及取代。**
   點擊 [檢索下一筆] 鍵後，畫面上就會顯示條件符合的指令物件選擇狀態，此時只要按一下 [取代] 鍵，即可取代為 [取代暫存器] 或 [取代目標] 的文字。
   在 [暫存器] 索引標籤畫面中點擊 [全部取代] 鍵，條件符合的暫存器就會被取代，取代結果則會顯示在輸出子視窗中。



檢索 / 取代專案檔案內的資料

可針對專案檔內的所有階梯圖程式及運動程式 (或所指定的程式) 檢索變數。或是用來檢索 / 取代暫存器及位址。

註記

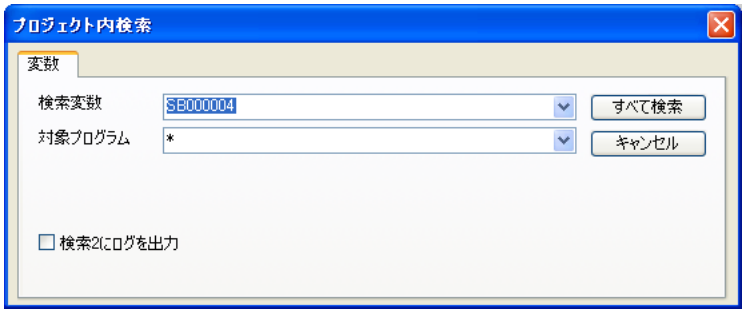
本功能僅適用於運動控制器連線中斷狀態 (離線狀態)。

接下來將介紹檢索或取代專案檔內資料的操作方法。

檢索專案檔案內的資料

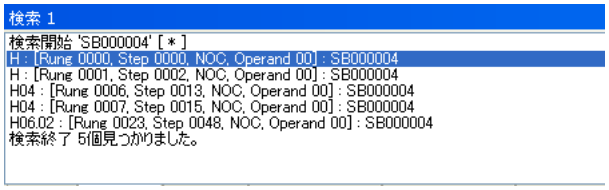
1. **讓您所要檢索的程式顯示在階梯圖編輯畫面的最上層，接著從主選單上依序選擇 [編輯] － [在專案內檢索]。**
   畫面上將出現 [在專案內檢索] 對話框。

2. **設定您所要檢索的變數 (位址) 及檢索目標的程式名稱。**



(註)1. 亦可從變數子視窗當中複製及剪貼，以輸入變數。
2. 加上「，」(逗號) 或空格後，即可在 [目標程式] 中指定多個程式。或者，也可以在 [目標程式] 中輸入如以下的「*」(通用字元)。
   * ，H* ，L* ，I* ，A* ，F* (所有函數) ，MPM* ，MPS*
   僅適用上述型態的通用字元。其他 (「H01.*」等) 類型皆不適用。
3. 勾選 [輸出日誌至檢索 2] 的核取方塊，即可將檢索結果輸出至檢索 2 子視窗，這時候檢索 1 子視窗的輸出內容將完全不變。若未勾選核取方塊，會將輸出至檢索 1 子視窗。

3. **開始進行檢索。**
   點擊 [全部檢索] 鍵，畫面上將顯示狀態列以告知目前的檢索狀態，並將檢索結果顯示於檢索子視窗上。



取代專案檔案內的資料

註記

- 當您對專案檔內的資料執行取代動作後，就會開始進行編譯及儲存，此時將無法再回復到取代前的狀態。重要的檔案必須在執行取代動作前，完成備份。
- 在執行取代動作前，若利用 MPE720 的 Engineering Manager 開啟運動程式，則程式將不會自動執行更新。因此，請先關閉運動程式再執行取代動作。

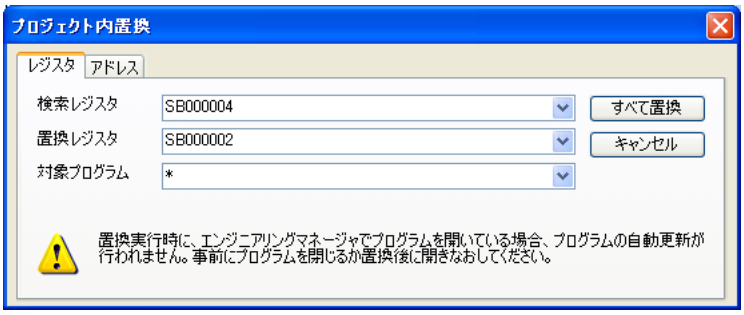
1. 讓您所要檢索的程式顯示在階梯圖編輯畫面的最上層，接著從主選單上依序選擇 [編輯] － [從專案檔裡取代]。

畫面上將出現 [從專案檔裡取代] 對話框。

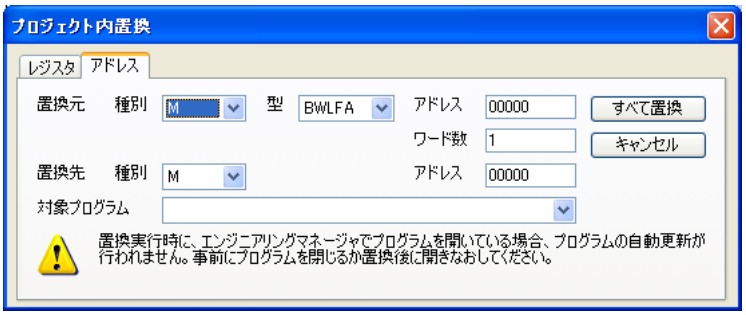
2. 設定您所要檢索的變數 (位址) 及檢索目標的程式名稱。

(註)1. 亦可從變數子視窗當中複製及剪貼，以輸入變數。
2. 加上「，」(逗號) 或空格後，即可在 [目標程式] 中指定多個程式。或者，也可以在 [目標程式] 中輸入如以下的「*」(通用字元)。
＊，H*，L*，I*，A*，F*(所有函數)，MPM*，MPS*
僅適用上述型態的通用字元。其他 (「H01.*」等) 類型皆不適用。

3. 請先選擇 [暫存器]、[位址] 其中任一個索引標籤，然後再設定您所要檢索及取代的內容。



[暫存器] 索引標籤：單一暫存器為取代目標。

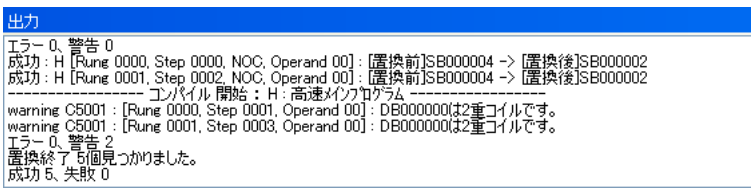


[位址] 索引標籤：符合條件的暫存器為取代目標。

(註)可以在 [目標程式] 中輸入如以下的「*」(通用字元)。
＊，H*，L*，I*，A*，F*，MPM*，MPS*

4. 開始進行檢索及取代。

點擊 [全部取代] 鍵，取代結果就會顯示在輸出子視窗上。

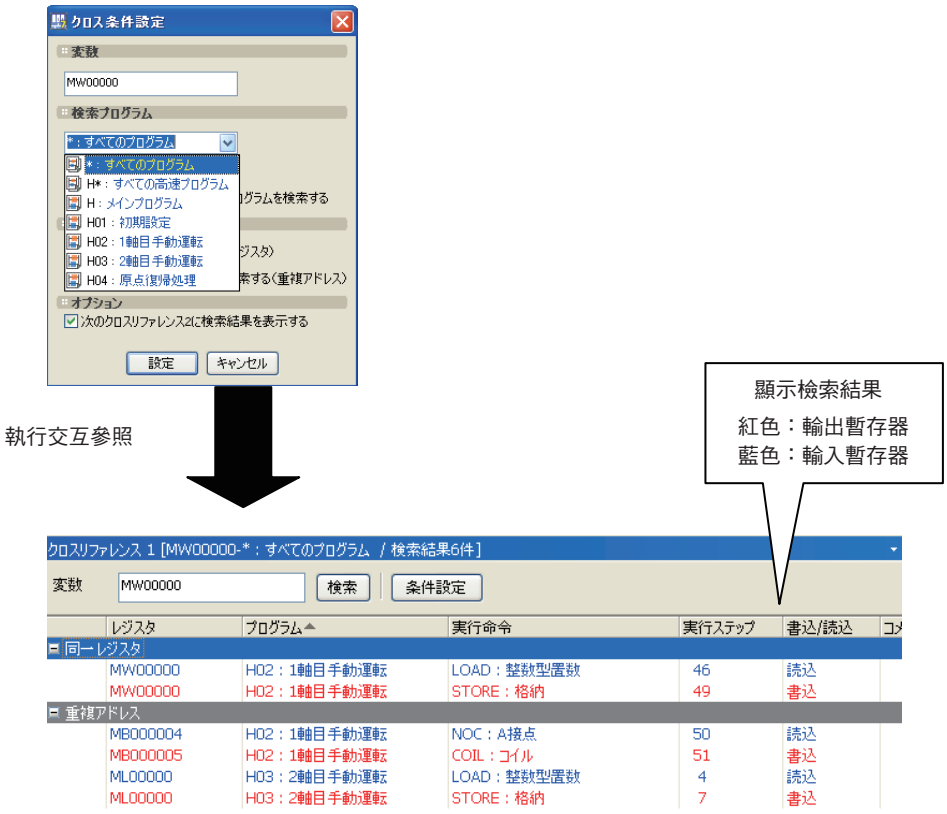


(註)編譯過程中一旦程式發生錯誤，取代動作將無法正確完成。
當完成取代動作後，已取代的暫存器變數或位址將會顯示在畫面上。

5.3 交互參照

所謂交互參照就是可用來檢索暫存器是否為某個程式所使用，若是在使用中，檢索是在何處被使用之功能。

顯示檢索結果時，若是被當作輸出暫存器使用的位置將顯示為紅色，若為輸入暫存器使用的位置則顯示為藍色。



若數值與某個暫存器所設定的數值不同，則數值可能會被覆寫在程式裡的其他位置。此時，只要利用「交互參照」來檢索暫存器，並確認顯示紅色的部分，即可找出哪些程式已經被覆寫了。

範例

使用陣列時，應利用下列方式來檢索。

1. 暫存器 [Register] 時

→ 此時檢索目標為MW00000和MW00001。

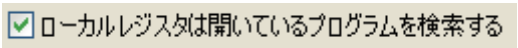
2. 暫存器 [常數] 時

→ 此時檢索目標為MW00000和MW00005。

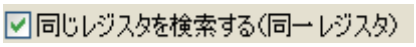
3. 暫存器 [常數] 時 (LONG 型)

→ 此時檢索目標為ML00000和ML00010。

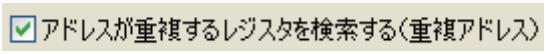
交互參照的設定條件如下，下表將分別說明各項功能。



核取方塊	檢索方法
開啟	可針對 MPE720 畫面上某個有效的畫面，檢索局部暫存器 (D 暫存器)。
關閉	可針對您所指定的圖面，檢索局部暫存器 (D 暫存器)。

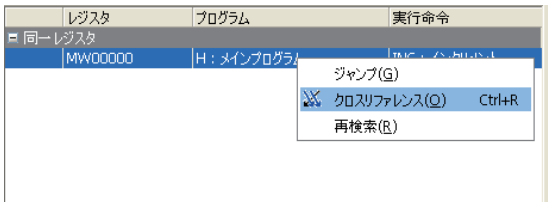
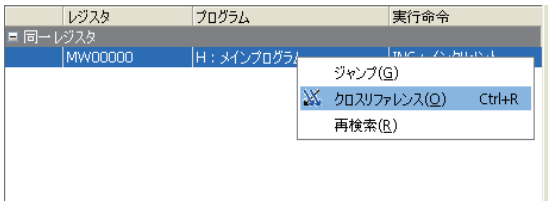
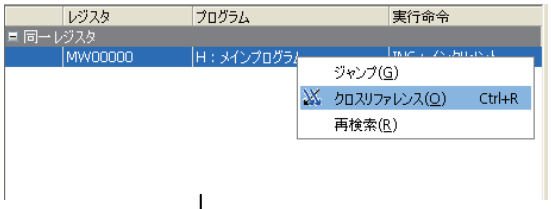
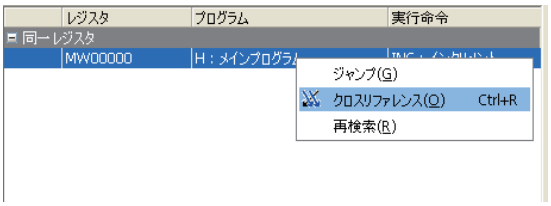


核取方塊	檢索方法
開啟	檢索與已檢索過暫存器的同一個暫存器。 利用「MW00000」來檢索程式中的變數時，若您希望畫面上能顯示清單，請勾選此核取方塊。 
關閉	未找到和過去檢索過的暫存器資料類型相同的暫存器。 利用「MW00000」來檢索程式中的變數時，若您不希望畫面上顯示清單，則不需要勾選此核取方塊。 



核取方塊	檢索方法
開啟	檢索重複的位址。 利用「ML00000」等其他的資料類型來檢索程式中變數時，若您希望在畫面上顯示清單，請勾選此核取方塊。 
關閉	不檢索重複的位址。 利用「ML00000」等其他的資料類型來檢索程式中變數時，若您不希望畫面上顯示清單，則不需要勾選此核取方塊。 

☑ 次のクロスリファレンス2に検索結果を表示する

核取方塊	檢索方法
開啟	利用交互參照視窗來執行交互參照功能時，檢索結果將會顯示在其他視窗中。 交互參照的結果最多可顯示在 3 個視窗上。 交互參照1  切換視窗 交互參照2  切換視窗 交互參照3  切換視窗
關閉	利用交互參照視窗來執行交互參照功能時，會更新在同一個視窗中並顯示檢索結果。 交互參照1  更新視窗

5.4 檢查重複線圈

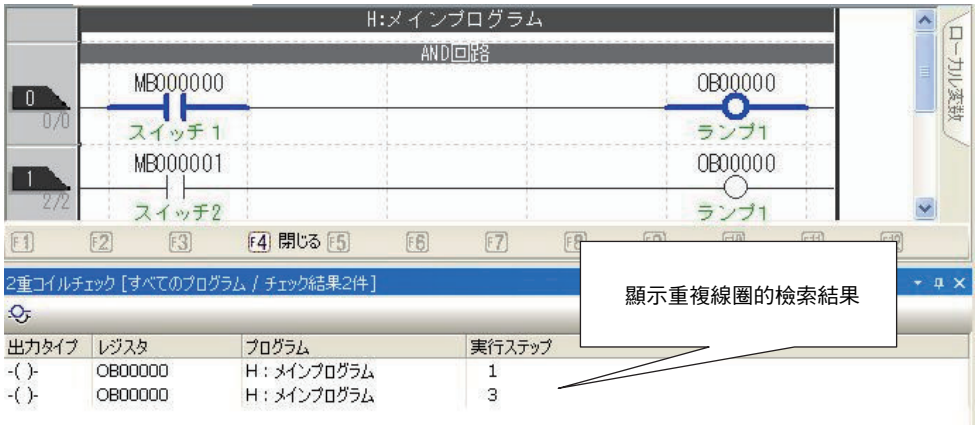
本功能係以所有階梯圖程式為對象，檢索程式中的重複線圈 (使用同一暫存器線圈)，並將檢索結果顯示在畫面上。

註記

和專案連線後，由於會使用專案檔的資料，因此畫面上有可能會顯示與專案連線中的運動控制器資料不一致的結果。
若要在專案連線狀態下檢索重複線圈，請先利用 [從運動控制器讀取]，將資料傳送到專案檔中。

從主選單上依序選擇 [除錯] － [檢查重複線圈]。

檢索重複線圈，並將結果輸出至重複線圈檢查子視窗中。



註記

當編譯選項中的 [勾選重複線圈核取方塊] 被設定為 ON 後，即使正在執行編譯，仍會繼續檢索重複線圈，並將結果以警告方式顯示在輸出子視窗中。

5.5 線圈強制開啟 / 關閉

本功能可用來將階梯圖編寫器所指定的線圈強制開啟 / 關閉。

無論線圈左側輸出為何，皆可利用本功能來輸出線圈。

例如編寫以下階梯圖程式時，即使實際上並沒有開關 (IB00000)，只要將繼電器 (DB000001) 強制設定為開啟，即可模擬開關啟動時的動作。

利用階梯圖程式設定強制開啟 / 關閉

進入階梯圖程式編輯畫面，再將您所指定的線圈物件強制設定為開啟 / 關閉，即可用來監控程式。

1. **進入強制開啟 / 關閉的線圈選擇狀態。**

2. **進入主選單，並依序選擇 [除錯] － [強制開啟] 或 [強制關閉]。**
   即可強制連接或中斷您所選擇的線圈。

<線圈強制開啟>

補充　如欲解除 [強制開啟] (或 [強制關閉])，請從主選單上依序 [除錯] － [強制取消]。

利用強制線圈清單子視窗變更為強制開啟 / 關閉

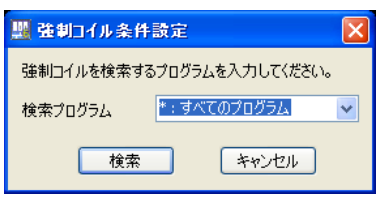
強制線圈清單子視窗係利用清單方式來顯示階梯圖程式中的強制線圈開啟 / 關閉狀態。

或是以所有程式為對象，變更或解除強制線圈開啟 / 關閉 / 取消狀態等。

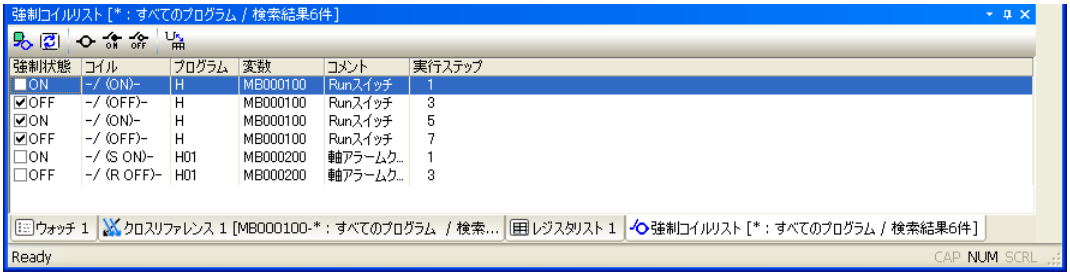
利用強制線圈清單子視窗來檢索強制線圈

1. **顯示強制線圈清單子視窗。**
   (註)請從主選單中依序選擇 [顯示] － [其他視窗] － [強制線圈清單]，即可切換為顯示 / 不顯示強制線圈清單子視窗。

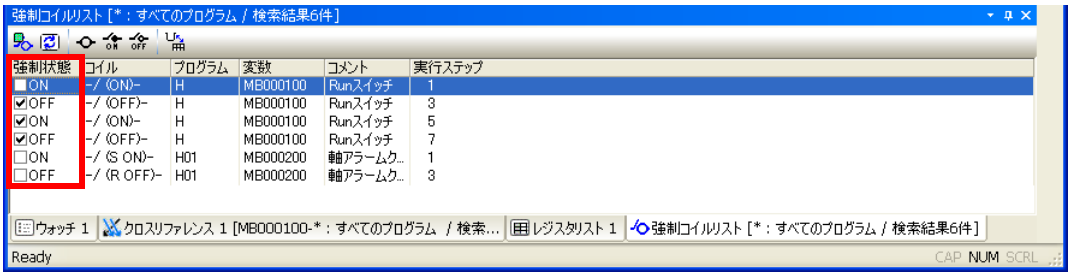
2. **從主選單上依序選擇 [除錯] － [強制線圈清單]。**
   (註)上述的情況係以所有的程式為對象，檢索強制線圈。若要指定所要檢索的目標程式時，請按下強制線圈條件設定鍵 ( )，接著畫面上就會顯示強制線圈條件設定視窗。



檢索結果將會顯示在強制線圈清單子視窗中。



3. 在要強制開啟 / 關閉的線圈上勾選核取方塊。

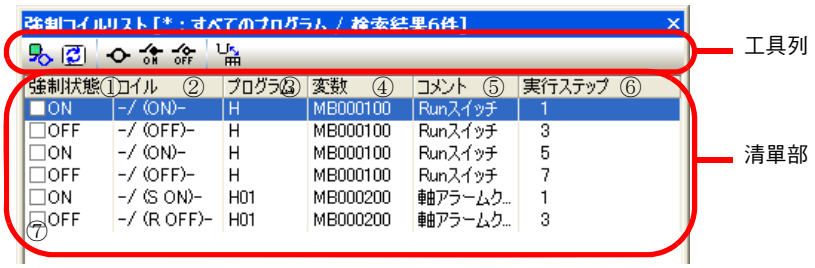


註記

1. 在強制線圈清單子視窗的清單上按一下右鍵，畫面上將出現彈出式選單，若選擇 [全部勾選] 或 [全部不勾選]，可將線圈強制狀態的核取方塊全部勾選或是全部不勾選。
2. 進入強制線圈清單子視窗的檢索結果畫面，並選擇其中任一行或是雙擊該畫面，就會跳到符合的階梯圖程式裡的線圈。或者，您也可以在強制線圈清單子畫面的清單上按一下右鍵，即使從彈出式選單選擇 [跳至]，也可以執行同樣的動作。若程式尚未開啟，將自動開啟程式，並跳至符合的程式裡的線圈。
3. 在強制線圈清單子畫面的清單上按一下右鍵，從出彈出式選單選擇 [交互參照]，或是從主選單上依序選擇 [除錯] – [交互參照]，以交互參照並檢索被設定為線圈的暫存器，檢索結果將被顯示在交互參照子視窗中。
4. 若要在檢索顯示中編輯階梯圖程式，則編輯完成的程式線圈將會顯示為灰色。

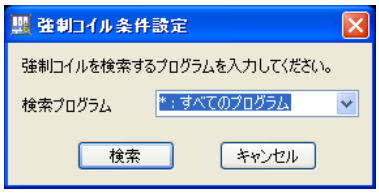
強制線圈清單子畫面各部分名稱及概述

強制線圈清單子畫面係由為了操作用來顯示強制線圈的「清單部」，以及強制線圈檢索、重新檢索或是變更強制狀態的「工具列」所組成。



◆ 工具列

· **強制線圈條件設定鍵 (** **)**
可用來叫出強制線圈條件設定對話框。指定用來檢索強制線圈的程式後，就會檢索強制線圈。



· **重新檢索鍵 (** **)**
從強制線圈條件設定對話框所指定的程式再次檢索強制線圈。

· **強制取消鍵 (** **)**
取消已勾選核取方塊的線圈強制狀態。

· **強制開啟鍵 (** **)**
強制開啟已勾選核取方塊的線圈。

· **強制關閉鍵 (** **)**
強制關閉已勾選核取方塊的線圈。

· **變數顯示切換鍵 (** **)**
切換使用中線圈的暫存器顯示方法為暫存器或變數。

◆ 清單部

① 強制狀態
將您所檢索到的線圈強制狀態顯示為 「ON」或 「OFF」。

② 線圈
顯示您所檢索到的線圈。
總共有以下 6 種類型的線圈。

線圈類型	線圈記號	
	ON	OFF
線圈	–/ (ON)–	–/ (OFF)–
設定線圈	–/ (S ON)–	–/ (S OFF)–
重置線圈	–/ (R ON)–	–/ (R OFF)–

③ 程式
顯示已檢索到線圈的程式名稱。

④ 變數
顯示已檢索到的線圈其所設定的變數或暫存器。

⑤ 註解
顯示變數註解。

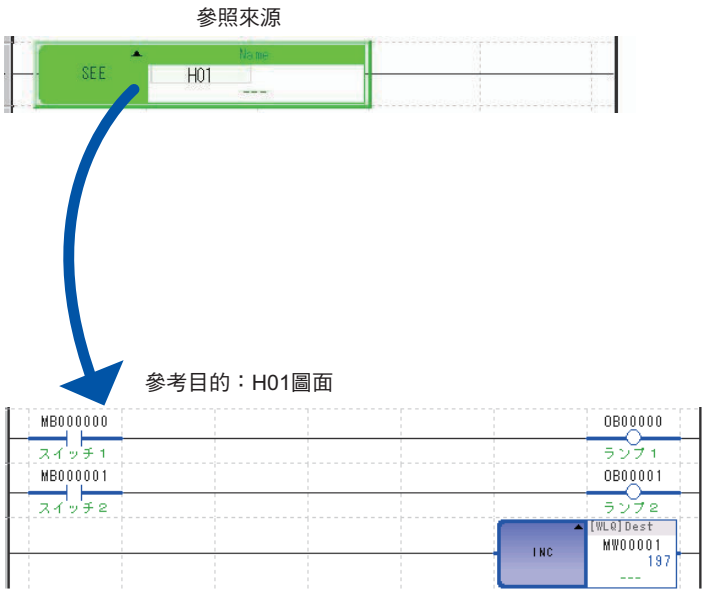
⑥ 執行步驟
顯示所檢索到的線圈執行步驟編號。

⑦ 核取方塊
勾選核取方塊後，就會開始強制操作 (開啟 / 關閉 / 取消)。皆可從工具列上的按鍵或是彈出式視窗，將線圈設定為強制開啟 / 關閉 / 取消狀態。

5.6 參照所要叫出的程式

可用來開啟 SEE 指令及 FUNC 指令所參照的圖面。

選擇您要確認程式的 SEE 指令物件或 FUNC 指令物件，接著再由主選單上依序選擇 [除錯] － [從現在位置叫出參照 (R)]。



5.7 暫存器清單

「暫存器清單」功能係利用暫存器清單子畫面 (1、2、3) 來監控連續區內暫存器的目前值 (暫存器圖表)。連接運動控制器後，即可即時進行監控。亦可編輯數值。

註記

- 暫存器圖表亦可在直接連線時，顯示專案檔的資料。和專案連線時，由於使用運動控制器的資料，因此顯示暫存器圖表時，有可能會發生和目前連線中的專案檔不一致的結果。
  若要在專案連線狀態下叫出暫存器圖表，請先利用 [由運動控制器讀取]，將資料傳送到專案檔中。
- 暫存器清單適用於 S・I・O・M・C・D・G 暫存器。但 C 暫存器為讀取專用，僅具參照功能。

顯示暫存器圖表

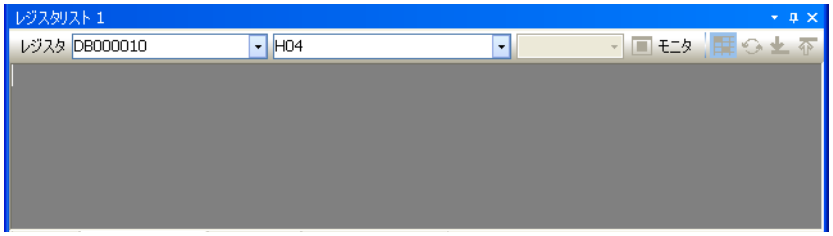
暫存器圖表背景色所代表的意義如下。

綠色	該暫存器目前為階梯圖所使用
紅色	重複暫存器 (該暫存器目前為多種資料類型所使用)

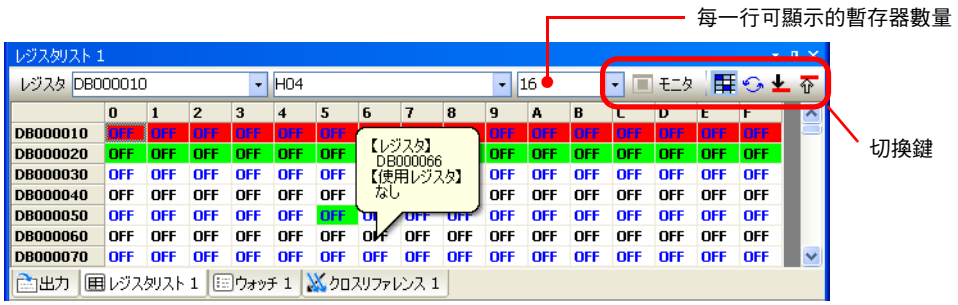
暫存器圖表的顯示步驟如下。

1. **選擇任一個暫存器清單子視窗 (1、2、3) 的索引標籤。**
   利用快速啟動程式，依序選擇 [監控] － [暫存器清單]，畫面上就會自動顯示暫存器清單 1 的子視窗。
   (註)從主選單上依序選擇 [顯示] － [暫存器清單] － [暫存器清單 1/ 暫存器清單 2/ 暫存器清單 3] 後，即可將暫存器子視窗 (1、2、3) 的顯示畫面切換為開啟 / 關閉。

2. **請在 [暫存器] 欄中，輸入您要顯示暫存器圖表的暫存器位址。若要叫出 D 暫存器清單，亦可如下圖所示，輸入程式編號。**



3. **按下 Enter 鍵。**
   讓您所指定的暫存器顯示在暫存器圖表的第一行。



D 暫存器之暫存器圖表範例及圖說文字 (Balloon) 顯示範例

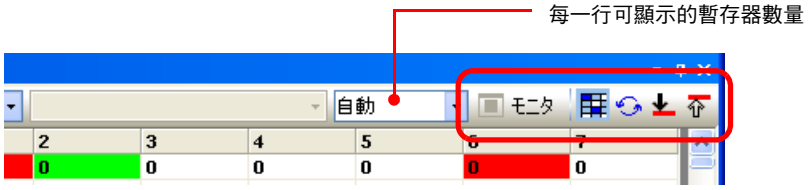
M 暫存器的暫存器圖表範例

註記

- 將游標移動到暫存器圖表上，畫面上就會出現該游標所在位置的暫存器及暫存器使用狀況的圖說文字 (Balloon)。
- 可變更每一行所顯示的暫存器數量。或是按一下清單右上方的 5 個鍵，即可切換顯示內容。
- 在暫存器清單上按一下右鍵，接著再從彈出式選單中選擇 [10 進制 /16 進制 /BIN/ASCII]，即可變更 [數值] 欄的資料類型。但「B」和「F」資料類型無法變更。
- 藍色和黑色將間隔 1 行交互顯示。
- [監控] 圖示只會在連接運動控制器時 (連線狀態) 動作。

切換暫存器圖表畫面

可用來變更每 1 行所顯示的暫存器數量。或是按一下顯示在右上方的 5 個鍵，即可切換暫存器圖表的顯示內容。



◆ 每一行可顯示的暫存器數量

直接輸入數值，或是利用從清單中選擇，即可在 1~16 的範圍內，變更每一行所顯示的暫存器數量 (Bit 型固定為「16」)。若選擇 [自動]，根據暫存器清單子視窗的顯示大小，自動設定所要顯示的暫存器數量。

◆ 監控器開啟 (モニタ)/ 關閉 (モニタ) 鍵

僅適用於連線狀態。每點擊一次即可切換開啟 / 關閉。

當監控狀態開啟時，畫面上將持續顯示暫存器更新狀態，監控關閉時則不會更新。

◆ 暫存器圖表顯示 ( )/ 不顯示 ( ) 鍵

每點擊一次即可切換顯示 / 不顯示。

選擇 [顯示]：目前被階梯圖程式使用中的暫存器背景將顯示為綠色，若該暫存器被多個資料類型所使用，則背景色將顯示為「紅色」。

選擇 [不顯示]：所有的暫存器背景將全部顯示為「白色」。

◆ 暫存器圖表重新顯示鍵 ( )

點擊後，就會開始更新暫存器圖表畫面。

註記

上述的 [暫存器圖表顯示 / 不顯示] 鍵無法在不顯示 ( ) 狀態下動作。

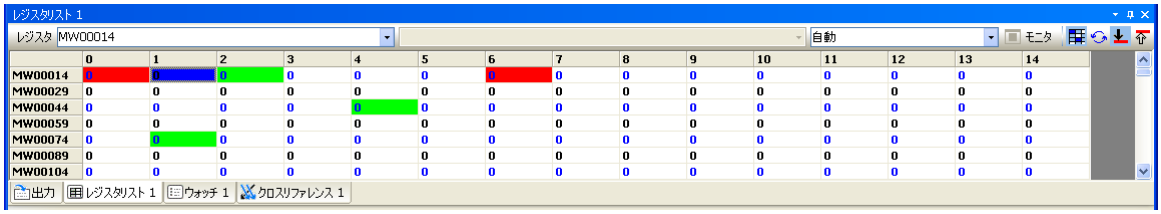
◆ 重複暫存器檢索鍵 (/)

本按鍵可以用來檢索並顯示重複的暫存器。按下 [↑] 鍵及 [↓] 鍵，即可分別往上或往下，檢索重複的暫存器。

結果將顯示於 [暫存器清單] 子畫面中，如有重複的暫存器，背景色將顯示為藍色。

註記

[暫存器圖表顯示 / 不顯示] 鍵無法在不顯示 ( ) 狀態下動作。



資料編輯

在暫存器圖表中，雙擊資料格，或是按下 F2 鍵，可顯示本文游標，即可進行下一項編輯動作。

- 直接輸入資料
- 刪除 (設定為 0)
- 複製、貼上

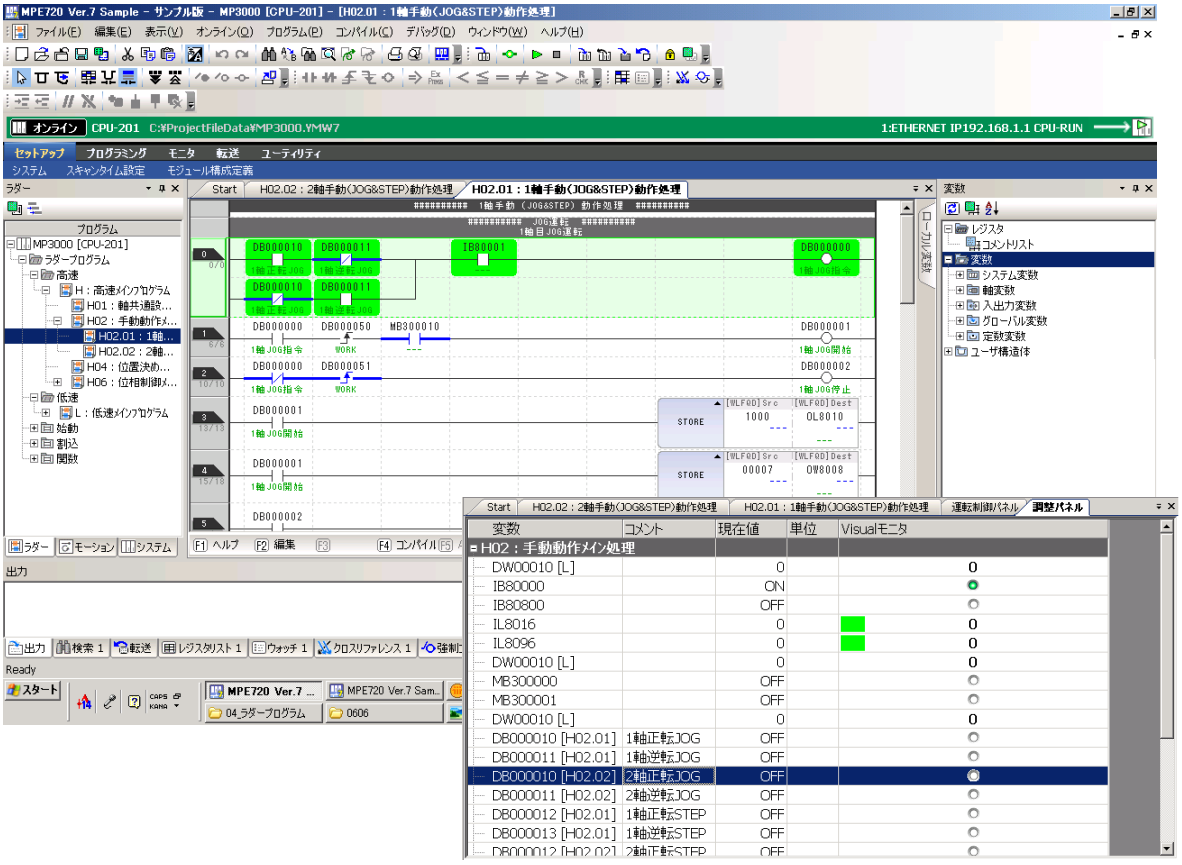
按下 Enter 鍵，以確定編輯內容。連線狀態下，變更後的資料將直接被反映在運動控制器的動作上。

5.8 調整面板

調整面板可用來登錄任一個暫存器，以及顯示、編輯目前值資料。除了數值外，亦可用來顯示註解標示或顯示燈等。

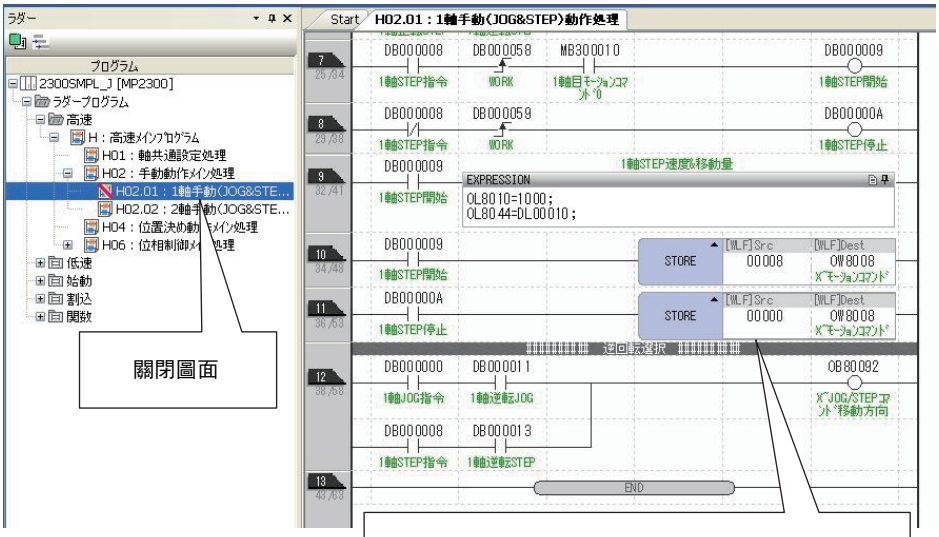
可進行所編寫的應用程式的運轉操作或動作確認。

只要調整 Visual 監控功能，就會依照狀況來顯示資料。



5.9 開啟 / 關閉階梯圖程式

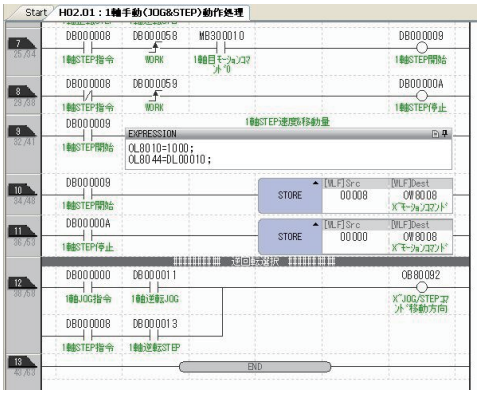
階梯圖程式開啟 / 關閉功能係以圖面為單位，將階梯圖程式處理作業切換為開啟 / 關閉。

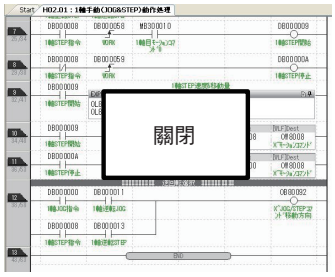


・被設定為關閉的階梯圖圖面將不執行動作。

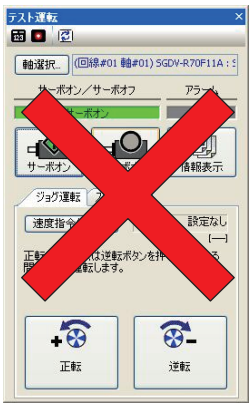
・無法用來控制線圈的開啟/關閉或執行運算。

只要將伺服馬達的伺服器設定為開啟或是將組成 JOG 運轉等序列的階梯圖圖面暫存關閉，就能利用 MPE720 的測試運轉及模組架構定義等，確認監控功能的操作是否正確。





階梯圖圖面關閉



因階梯圖圖面在移動，無法利用MPE720隨心所欲地操控監控功能。

可利用MPE720隨心所欲地操控監控功能。

5.10 檢視

「檢視」功能係利用檢視子視窗 (1、2、3) 來監控您所指定的 S、I、O、M、C、D 暫存器數值及註解。連接運動控制器後，即可及時進行監控。亦可編輯數值。


註記

進行專案連結時，已登錄在檢視功能中的資料僅會被儲存在運動控制器。若要讓檢視資料反映在專案檔上，請利用運動控制器將資料全部傳送。

顯示檢視資料

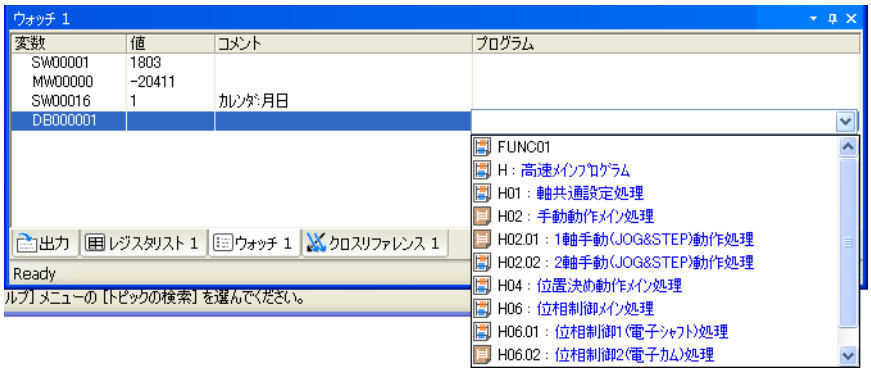
1. 選擇檢視子視窗 (1、2、3) 其中任一個索引標籤。

從快速啟動程式中依序選擇 [監控] ─ [檢視] 後，畫面上就會自動顯示檢視 1 子視窗。

(註)從檢視子視窗 **(1、2、3)** 的主選單上依序選擇 [顯示] ─ [檢視] ─ [檢視 1/ 檢視 2/ 檢視 3]，即可切換畫面顯示為開啟 / 關閉。

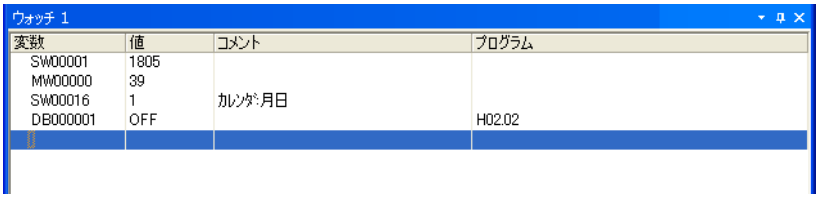
2. 雙擊 [變數] 欄，或是按下 F2 鍵，即可叫出本文游標，接著就能輸入您所要檢視的暫存器或變數暫存器。

(註)1. 亦可利用階梯圖程式及變數子視窗中的拖曳、複製或貼上等功能來輸入資料。
2. 如欲檢視 D 暫存器，也可以依照下圖所示，輸入程式編號。



3. 按下 Enter 鍵。

可叫出您所指定的暫存器內容。



在行上按一下右鍵，當彈跳式選單出現後，請選擇 [10 進制 /16 進制 /BIN/ASCII]，即可變更 [數值] 欄的資料類型。

編輯 [數值] 欄

雙擊 [數值] 欄，或按下 F2 鍵，當 [數值] 欄出現本文游標後，即可直接輸入資料，或是執行複製、貼上等動作。


註記

無法用來編輯註解。

資料輸入完成後，請按下 Enter 鍵，以確定編輯內容。


註記

連線狀態下，變更後的資料將直接被反映在運動控制器的動作上。

5.11 安全功能

MPE720 Ver. 7 包含以下安全性功能。利用這些安全性功能，以專案或是程式 (圖面) 為單位，來設定存取權限，以達到保護資料的目的。

- **使用者管理 (使用者名稱 / 密碼設定)**
  可用來註冊或變更具有開啟專案權限的使用者。
  只要連接運動控制器的狀態下設定，即可設定運動控制器的存取權。
- **設定專案密碼**
  可設定用來開啟專案檔的密碼。
- **設定程式密碼**
  可設定用來開啟階梯圖程式及運動程式的密碼。密碼可依程式別分別設定。
- **設定連線安全性**
  可設定用來讀取運動控制上的資料的安全性 (密碼) 或限制權限。限制具有某種權限等級以上的使用者才能利用運動控制器讀取程式資料，或是開啟程式。

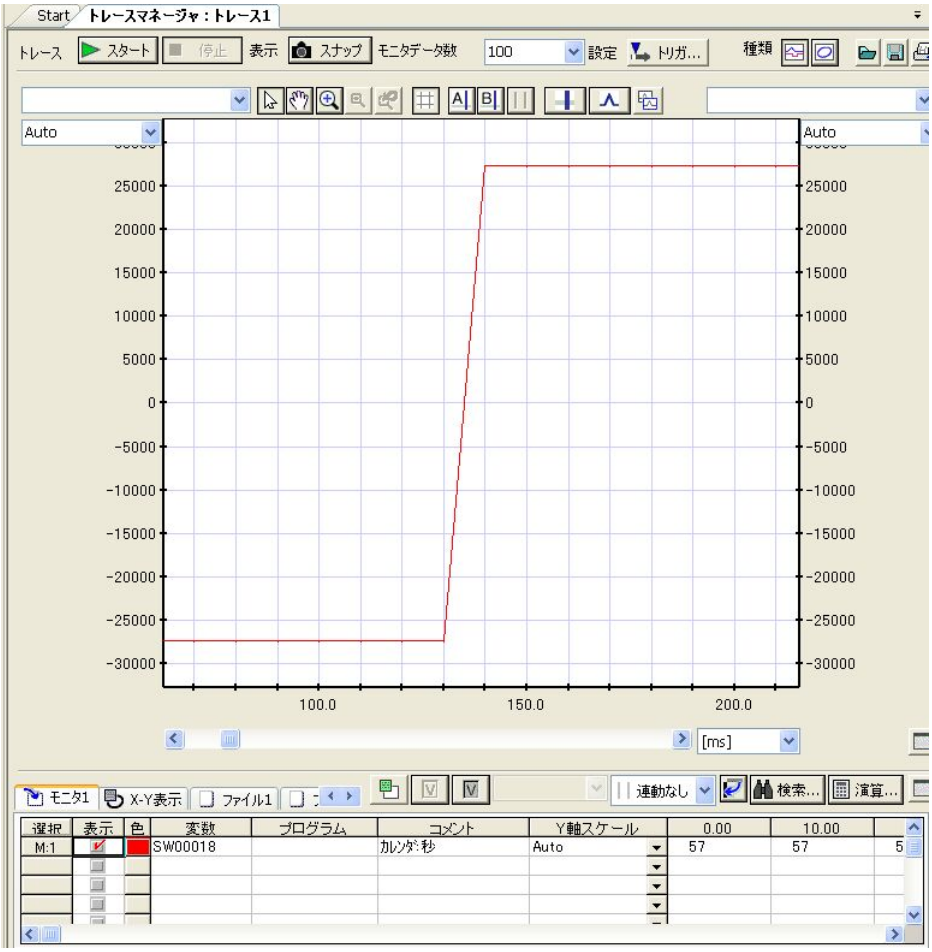
5.12 追蹤功能

MPE720 Ver. 7 包含 3 種追蹤功能。

- **即時追蹤**
  利用圖表方式即時監控您所指定的暫存器。
- **資料追蹤**
  依照您所指定的時間，利用運動控制器搜尋您所指定暫存器，以進行資料運算，並繪製成為圖表。
  可依照您所指定的時間進行暫存器資料分析，以作為階梯圖程式的除錯器之用。
- **XY 追蹤**
  可在每次掃描時取得 X 軸、Y 軸的位置資料，並顯示為 2D 平面圖。

以上 3 種功能，皆可將追蹤資料輸出為 CSV 檔。
適合做為階梯圖程式、運動程式的動作確認及除錯用途。

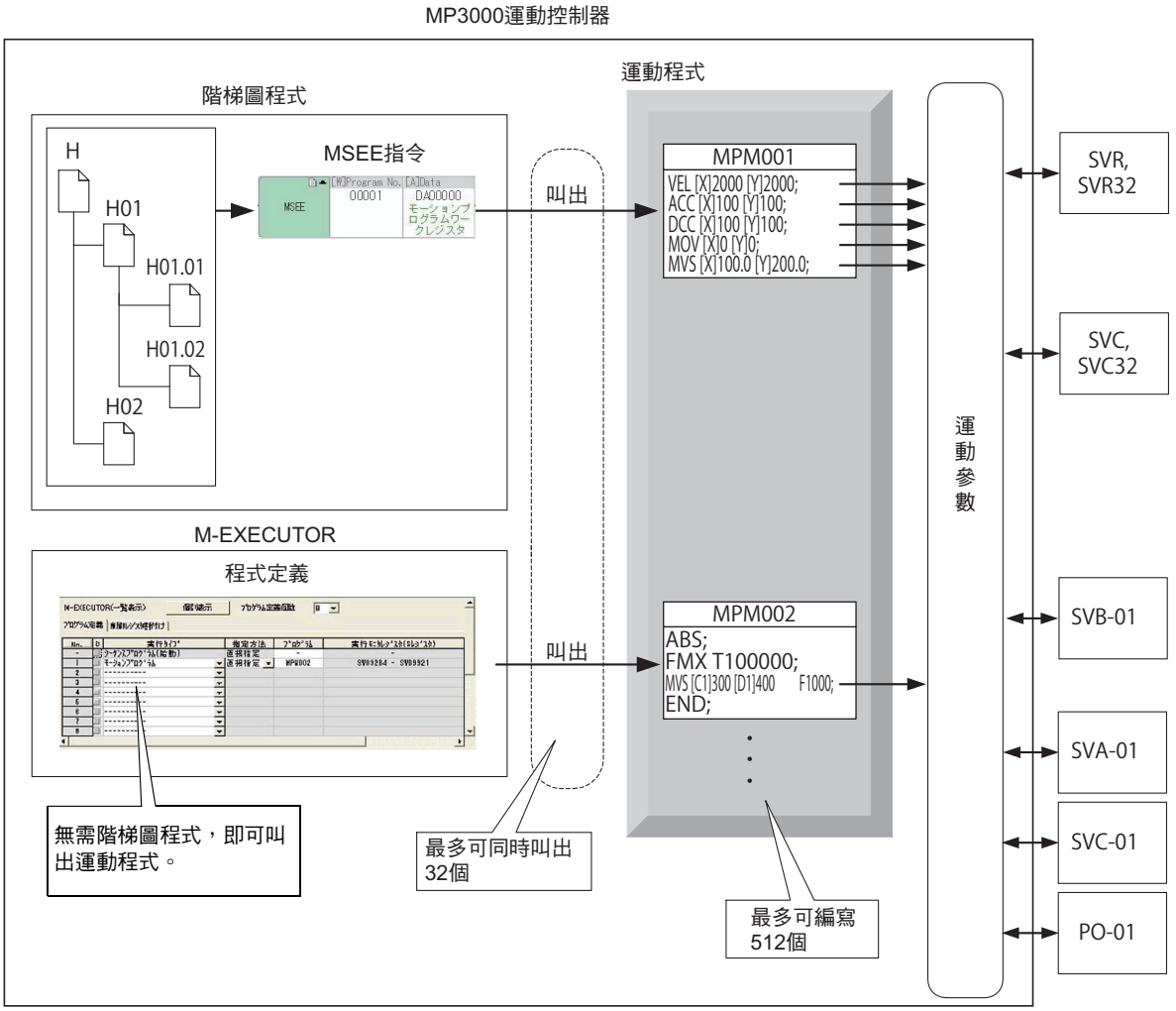
以下為資料追蹤視窗之範例。



5.13 高階使用方法

運動程式

運動程式是一種可用來編寫運動語言等文字形式的語言的程式。除了基本的動作控制及運算外，只要使用運動程式，就能輕鬆編寫像是線性內插及循環內插等複雜的動作。
利用階梯圖程式的 DWG.H (高速掃描處理畫面) 來編寫 MSEE 指令或是在 M-EXECUTOR 模組的執行登錄視窗內完成登錄，即可開始執行運動程式。



如欲進一步瞭解運動程式，請參閱下述手冊。

MP3000 系列　運動程式　程式編寫手冊 (資料編號：YTWMNCO-15005A)

系統服務暫存器一覽表

附錄 A

本章將針對運動控制器系統內置的系統暫存器當中，說明系統服務暫存器。

系統暫存器內置於運動控制器的系統中，可用來讀取系統的錯誤資訊及運轉狀態。

	內容
SW000000	系統服務暫存器
SW000030	系統狀態
SW000050	系統錯誤狀態
SW000080	使用者運算錯誤狀態 (概述)
SW000090	系統服務執行狀態
SW000110	使用者運算錯誤狀態 (內容)
SW000190	警報計數器及警報清除
SW000200	系統輸出入錯誤狀態
SW000504	系統預約
SW000652	CF 卡相關系統暫存器 (僅限 MP2200/CPU-02、CPU-03)
SW000698	配置狀態
SW000800	模組資訊
SW001312	系統預約
SW001411	MPU-01 模組系統狀態
SW002048	系統預約
SW003200	運動程式資訊
SW005200 ～ SW008191	系統預約

系統服務暫存器可分為以下 5 類。

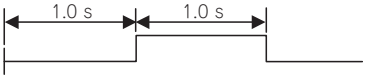
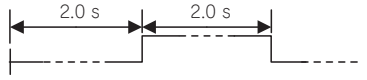
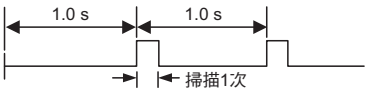
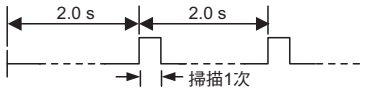
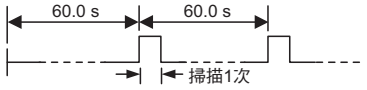
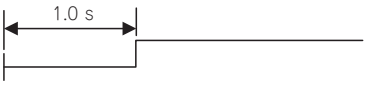
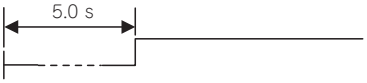
- 適用所有圖面
- DWG.H (高速掃描處理畫面) 專用
- DWG.L (低速掃描處理畫面) 專用
- 掃描執行狀態及行事曆
- 系統程式軟體編號及程式記憶體可用空間

適用所有圖面

名稱	暫存器編號	備註
系統預約	SB000000	(未使用)
H (高速) 掃描	SB000001	開始高速掃描後，只一次掃描時開啟
L (低速) 掃描	SB000003	開始低速掃描後，只一次掃描時開啟
常時 (開啟)	SB000004	常時開啟 (= 1)
系統預約	SB000005， SB000006	(未使用)
H (高速) 掃描執行中	SB000007	執行高速掃描時開啟 (= 1)
系統預約	SB000008 ～ SB00000F	(未使用)

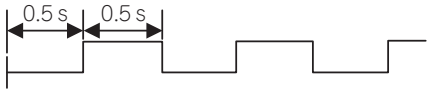
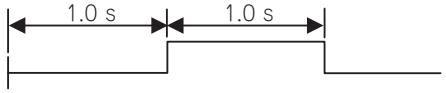
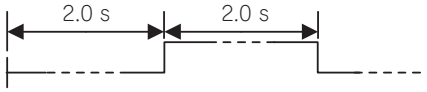
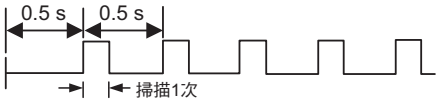
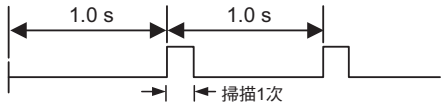
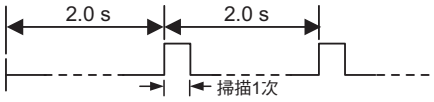
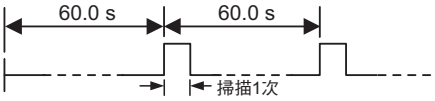
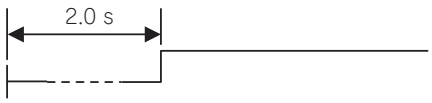
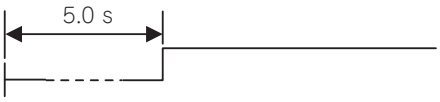
DWG.H (高速掃描處理畫面) 專用

開始 H (高速) 掃描後，即執行動作。

名稱	暫存器編號	備註
1 次掃描閃爍電驛	SB000010	
0.5 s 閃爍電驛	SB000011	
1.0 s 閃爍電驛	SB000012	
2.0 s 閃爍電驛	SB000013	
0.5 s 取樣繼電器	SB000014	
1.0 s 取樣繼電器	SB000015	
2.0 s 取樣繼電器	SB000016	
60.0 s 取樣繼電器	SB000017	
開始掃描處理 1.0 s 後繼電器	SB000018	
開始掃描處理 2.0 s 後繼電器	SB000019	
開始掃描處理 5.0 s 後繼電器	SB00001A	

DWG.L (低速掃描處理畫面) 專用

開始 L (低速) 掃描後，即執行動作。

名稱	暫存器編號	備註
1 次掃描閃爍電驛	SB000030	
0.5 s 閃爍電驛	SB000031	
1.0 s 閃爍電驛	SB000032	
2.0 s 閃爍電驛	SB000033	
0.5 s 取樣繼電器	SB000034	
1.0 s 取樣繼電器	SB000035	
2.0 s 取樣繼電器	SB000036	
60.0 s 取樣繼電器	SB000037	
開始掃描處理 1.0 s 後繼電器	SB000038	
開始掃描處理 2.0 s 後繼電器	SB000039	
開始掃描處理 5.0 s 後繼電器	SB00003A	

掃描執行狀態及行事曆

名稱	暫存器編號	備註
高速掃描設定值	SW00004	高速掃描設定值 (0.1 ms)
高速掃描目前值	SW00005	高速掃描目前值 (0.1 ms)
高速掃描最大值	SW00006	高速掃描最大值 (0.1 ms)
高速掃描設定值 2	SW00007	高速掃描設定值 (1 μs)
高速掃描目前值 2	SW00008	高速掃描目前值 (1 μs)
高速掃描最大值 2	SW00009	高速掃描最大值 (1 μs)
低速掃描設定值	SW00010	低速掃描設定值 (0.1 ms)
低速掃描目前值	SW00011	低速掃描目前值 (0.1 ms)
低速掃描最大值	SW00012	低速掃描最大值 (0.1 ms)
系統預約	SW00013	(未使用)
掃描執行時之目前值	SW00014	目前執行中的掃描目前值 (0.1 ms)
行事曆：年	SW00015	西元 1999 年：0099 (BCD) (僅最後 2 位數)
行事曆：月日	SW00016	12 月 31 日：1231 (BCD)
行事曆：時分	SW00017	23 時 59 分：2359 (BCD)
行事曆：秒	SW00018	59 秒：59 (BCD)
行事曆：週	SW00019	0：日、1：一、2：二、3：三、4：四、5：五、6：六

系統程式軟體編號及程式記憶體可用空間

名稱	暫存器編號	備註
系統程式軟體編號	SW00020	Sxxxx (以 BCD 值來儲存 xxxx)
系統編號	SW00021 ～ SW00025	(未使用)
程式記憶體可用空間	SL00026	以位元組為單位
記憶體總容量	SL00028	以位元組為單位

範例程式

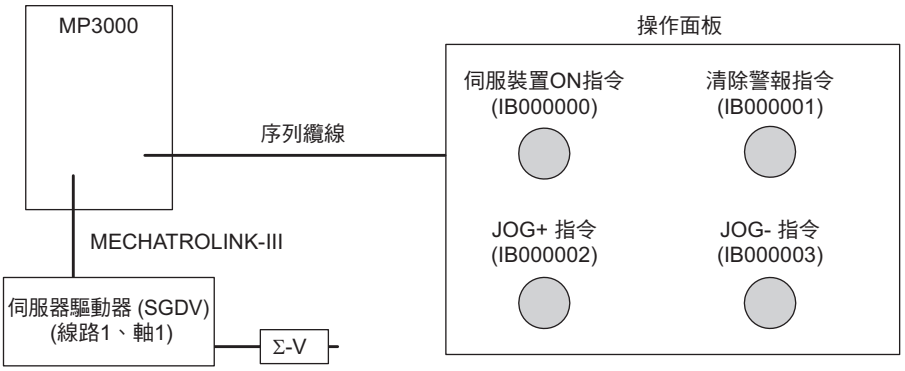
附錄 B

本章將以執行試運轉時所使用的階梯圖程式為例進行說明。

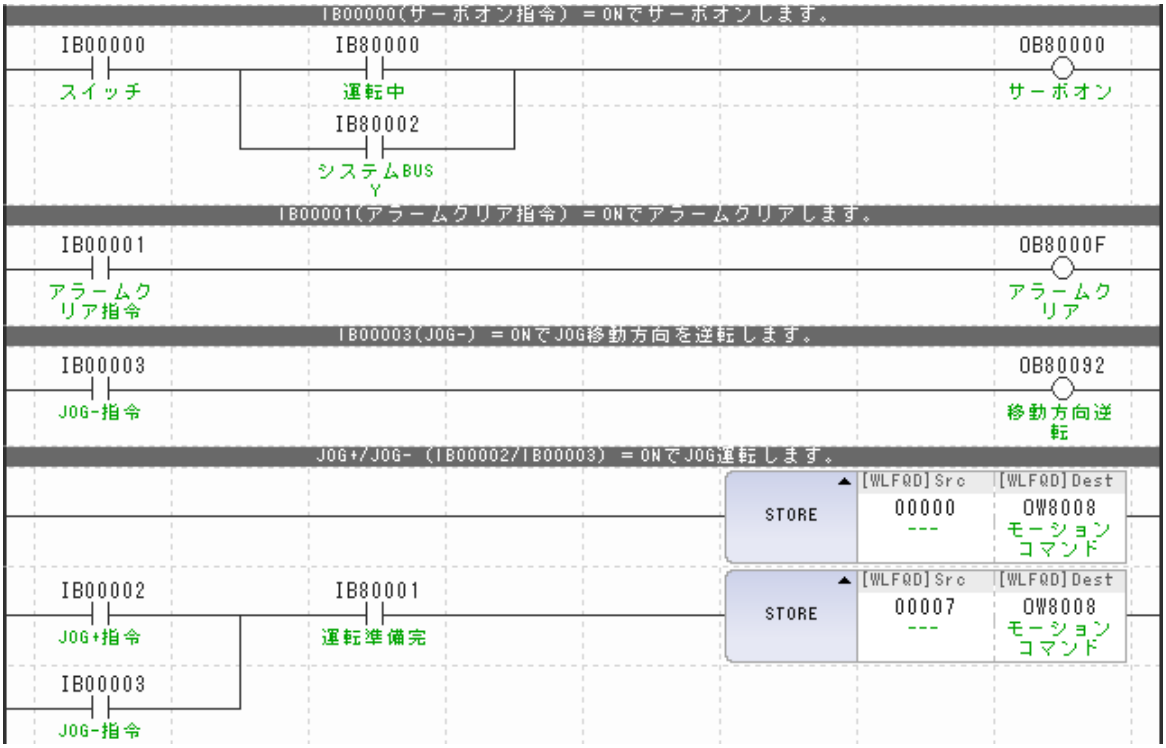
B.1 利用操作面板執行 JOG 運轉

以下所示係操作面板及馬達被連接至運動控制器時，使用操作面板上的開關來啟動馬達之架構範例及階梯圖程式範例。

■ 架構範例

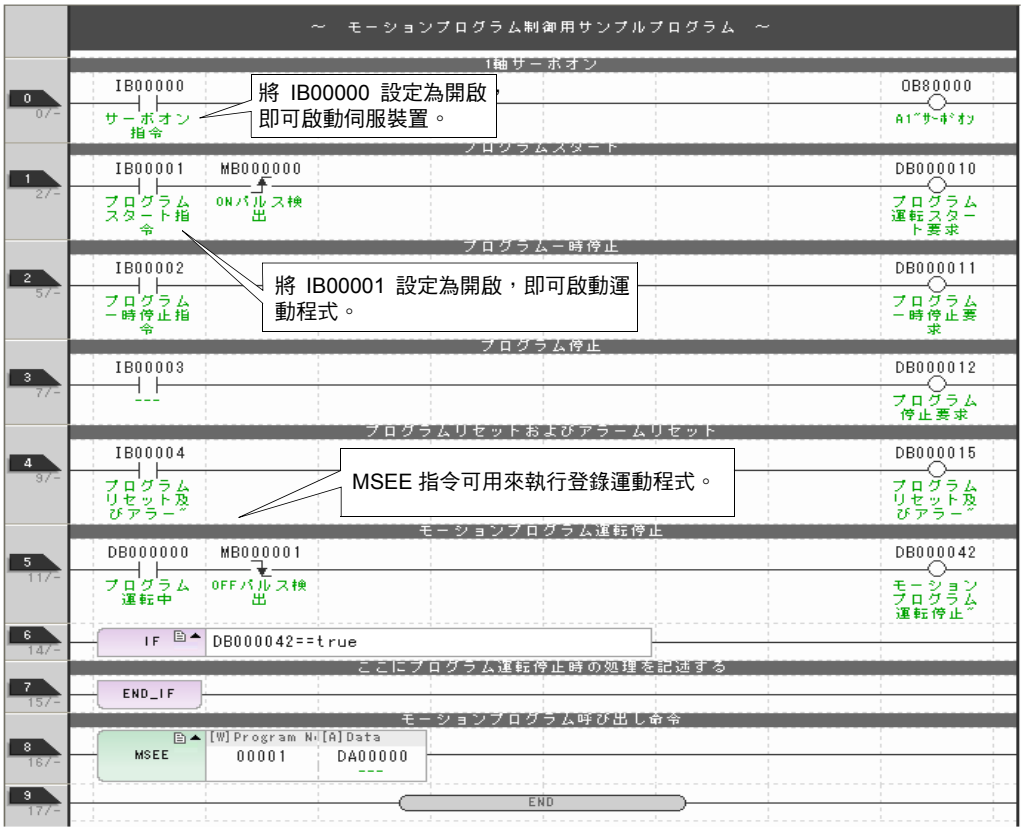


■ 階梯圖程式範例



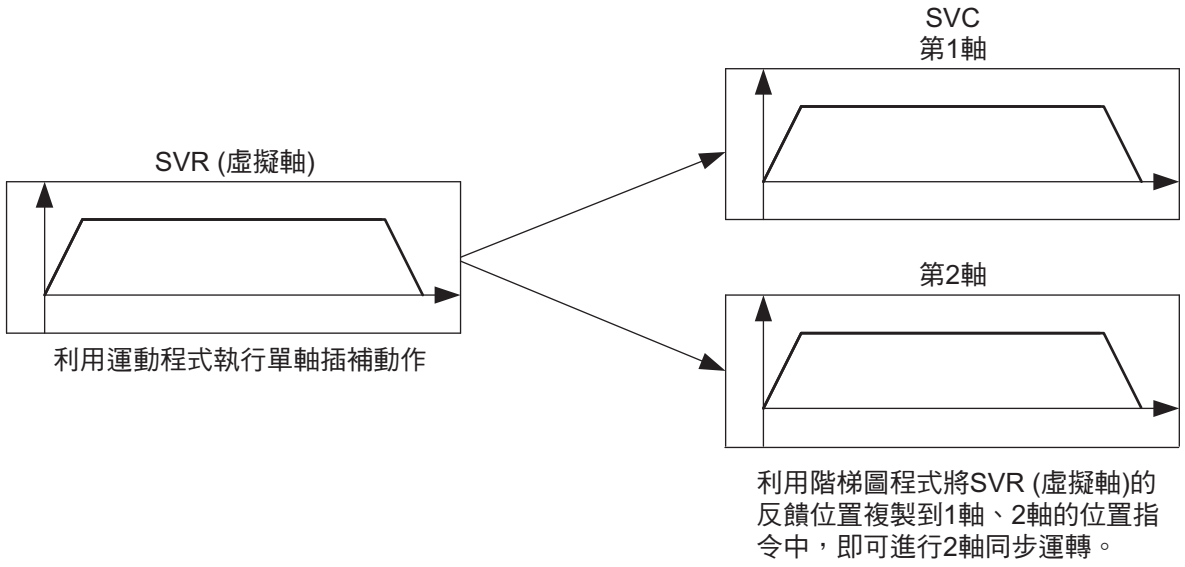
B.2 運動程式控制

以下係用來執行運動程式之階梯圖程式範例。



B.3 使用假想軸的簡單 2 軸同步運轉

利用運動程式來移動 SVR (虛擬軸)，接著再利用階梯圖程式將 SVR (虛擬軸) 的反饋位置分配至 2 個實軸，即可執行 2 軸同步運轉。

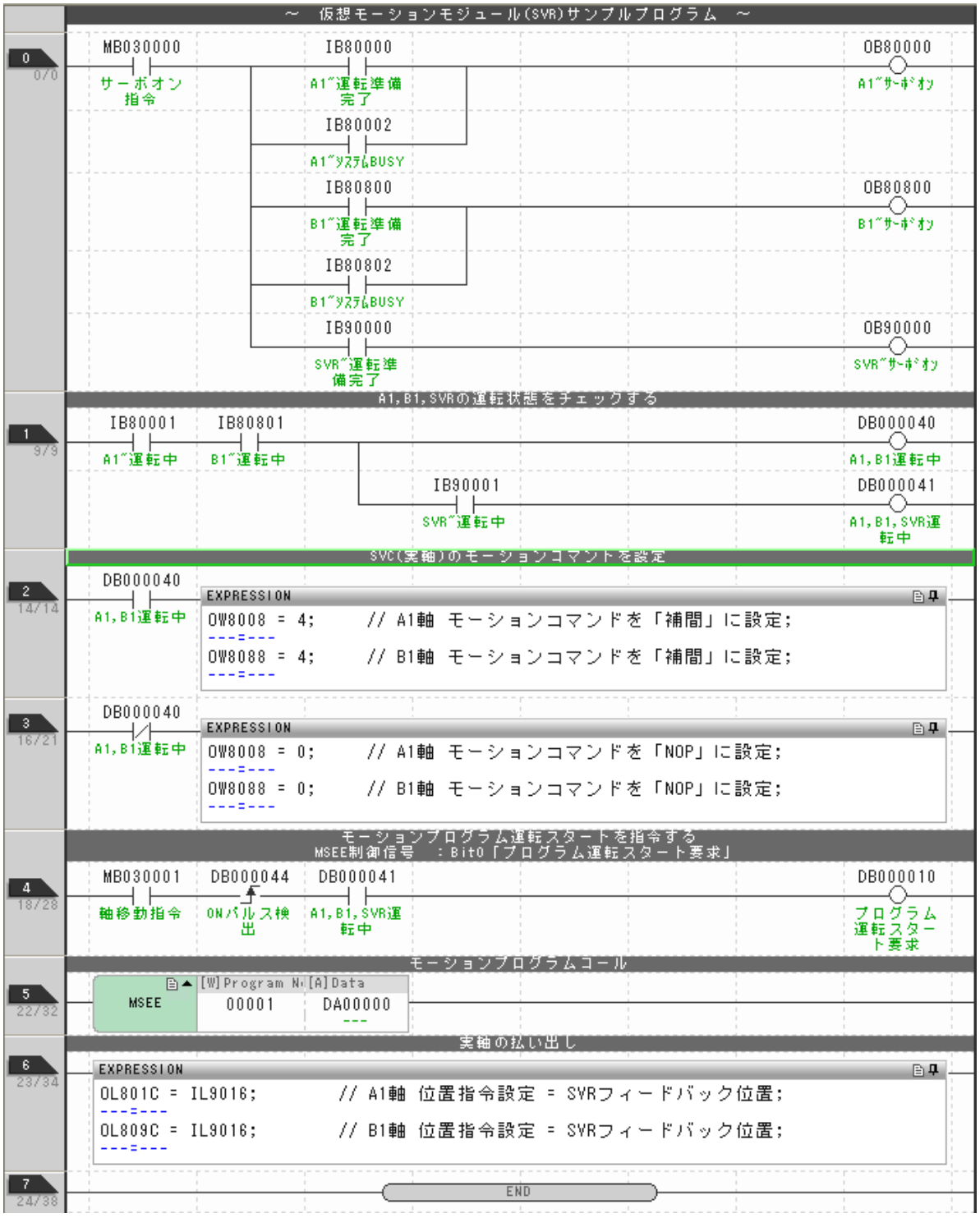


以下為上述之運動程式及階梯圖程式範例。

■ **運動程式範例**

```
FMX T10000K;                  " 設定最高插補速度 K = 1000
INC;                          " 增量模式
IAC T500;                     " 插補加速時間 = 500 ms
IDC T500;                     " 插補減速時間 = 500 ms
MVS [SVR] 1000K F10000K;   " 移動距離 1000000 時之插補動作
END;
```

■ 階梯圖程式範例



重要

本程式省略了軸異常時之復歸方法。使用本程式前，應先預設可能會發生的異常狀況，並新增程式以達到安全運轉的目的。

EXPRESSION 指令格式

附錄 C

本章將說明 EXPRESSION 指令的格式。

C.1 適用之算式組成要素

適用於 EXPRESSION 指令的算式係由運算子、運算元 (常數及變數) 以及函數等組成。

接下來將說明算式的各種組成要素。

運算子

運算子的類型及適用之運算子

下表為運算子的類型及適用之運算子。

類型	適用之運算子	
算術運算子	+	加法
	−	減法
	*	乘法
	/	除法
	%	餘數
	&	各位元之積
	¦	各位元之和
邏輯運算子 (僅適用於位元型)	&&	邏輯積
	¦¦	邏輯和
	!	邏輯非
比較運算子	==	等於右值
	!=	不等於右值
	>	大於右值
	>=	大於或等於右值
	<	小於右值
	<=	小於或等於右值
指定運算子	=	將右值代入左值
保留字	true	邏輯式的數值判斷 (真)
	false	邏輯式的數值判斷 (偽)
控制指令	IF，ELSE，IEND	ELSE 可省略

優先順序及組合規則

運算子有其優先順序，下表將介紹其組合規則 (運算元評估順序)。

優先度	運算子	說明	組合規則
高	[] ()	算式	由左而右
↑	− !	單項	由右而左
	* / %	乘法、除法、餘數	由左而右
	+ −	加法、減法	
	< > <= >=	關係	
	== !=	關係 (等價)	
	&	各位元之積	
	¦	各位元之和	
↓	&&	邏輯積	
低	¦¦	邏輯和	

(註)若同一行的運算子具有相同的優先順序，則必須根據組合規則來進行評估。

運算元

常數

無論是整數或實數皆適合當作常數。

・以 64 位元整數值 (4 長整數) 來表示的數值範圍皆適合當作整數。

(-9,223,372,036,854,775,808 ～ 9,223,372,036,854,775,807)

・以 64 位元 double 型 (倍精度實數) 來表示的數值範圍皆適用於實數。

± (2.225E-308 ～ 1.798E+308)

補充

以 16 進制方式來表示 EXPRESSION 指令、IF 指令、WHILE 指令時，必須以 0x□□□□ 來表示。
若標示為 H□□□□，將造成錯誤發生。
範例)　H012F　…　NG　　0x012F　…　OK
其他指令 (STORE 等) 需標示為 H□□□□。

變數

EXPRESSION 指令中，將可使用 C 語言的任意變數名稱適用至運動控制器的暫存器後才可編寫。

而且 C 語言雖然沒有 bool 型的變數，但運動控制器的位元型暫存器仍會被當作 bool 型來處理。bool 型的變數只有 true 或 false 的 2 個值，而且僅適用於邏輯式。

■ 變數名稱的限制

適用之變數名稱需符合以下限制條件。

・必須以數值以外的字元做為起始。

・ASCII 字元當中，只有英文字母、底線「_」及數字適用。

・請勿使用與以下函數名稱相同的變數名稱。

☞ **EXPRESSION 指令所適用之指令 (第 C-4 頁)**

範例

Abc	OK
Get_input()	OK
1ab	NG
Sin	NG

EXPRESSION 指令所適用之指令

以下為 EXPRESSION 指令所適用之指令。

指令	內容	範例	保留字
+	加法	MW00001 = MW00002 + MW00003	○
−	減法	MW00001 = MW00002 − MW00003	○
*	乘法	MW00001 = MW00002 × MW00003	○
/	除法	MW00001 = MW00002 / MW00003	○
%	餘數	MW00001 = MW00002 % MW00003	○
&	各位元之積	MW00001 = MW00002 & 4096	○
｜	各位元之和	MW00001 = MW00002 ｜ 4096	○
&&	邏輯積	MB000010 = MB000011 && MB000012	○
‖	邏輯和	MB000010 = MB000011 || MB000012	○
!	邏輯非	MB000010 = !MB000011	○
==	等於右值	MB000010 = MB000011 == true	○
>=	右值大於或等於左值	MB000010 = MW00020 >= MW00021	○
>	右值大於左值	MB000010 = MW00020 > MW00021	○
<	右值小於左值	MB000010 = MW00020 < MW00021	○
<=	右值小於或等於左值	MB000010 = MW00020 <= MW00021	○
=	將右值代入左值	MW00001 = MW00002	○
true	真	MB000010 = MB000011 == true	○
false	偽	MB000010 = MB000011 == false	○
sin()	SIN	MW00001 = sin(MW00002)	○
cos()	COS	MF00002 = cos(MF00004)	○
atan()	ARCTAN	MF00002 = atan(MF00004)	○
tan()	TAN	MF00002 = tan(MF00004)	○
()	括弧	MW00001 = (MW00002 + MW00003) / MW00004	○
asin()	ARCSIN	MF00002 = asin(MF00004)	○
acos()	ARCCOS	MF00002 = acos(MF00004)	○
sqrt()	SQRT	MW00001 = sqrt(MW00002)	○
abs()	ABS	MW00001 = abs(MW00002)	○
exp()	EXP	MF00002 = exp(MF00004)	○
log()	LOG 自然對數	MF00002 = log(MF00004)	○
log10()	LOG10 常用對數	MF00002 = log10(MF00004)	○

C.2 編寫方式之限制

算式係由運算元及運算子互相組合進行編寫，在編寫方式上具有幾項限制條件。
若不符合限制條件，則無法被判讀為算式。

接下來，將說明各種限制條件。

算術運算子

此種運算子適用於整數型及實數型運算元。單項負號僅能使用一次。位元運算僅能使用整數型。位元型運算元不適用於算式運算。而且當計算值超過暫存器範圍時，也不會自動轉換資料類型，因此 使用者必須先為暫存器配置適合的資料類型。

範例

MW00001 = MW00002 + MW00003	OK
MW00001 = MW00002 / 345	OK
MF00002 = (MW00004 + MF00002) / (ML00018 + MW00008)	OK
MW00001 = MW00002 & 4096	OK
MB000010 = MB000011 - MB000012	NG
MW00001 = MB000011 * MW00001	NG

比較運算子

此種運算子適用於整數型及實數型運算元。左邊必須編寫位元型暫存器。若要利用 「==」或「!=」來比較整數位元型運算元，則比較對象必須使用 true/false。

範例

MB000010 = MW00002 != MW00003	OK
MB000010 = MF00002 < 99.99	OK
MB000010 = MW00002 >= MW00003	OK
MB000010 = MB000011 == true	OK
MB000010 = MB000011 != 0	NG
MB000010 = MB000011 == 1	NG

邏輯運算子

此類運算子僅適用於位元型運算元。

範例

MB000010 = MB000011 && MB000012	OK
MB000010 = !MB000011	OK
MB000010 = (MW000020 >= 50) && MB000011	OK
MB000010 = MW00001 ¦¦ MW00002	NG
MB000010 = !MW00001	NG

指定運算子

只要資料為實數型或整數型，即使資料類型不同也能代入實數型及整數型暫存器中。但若將實數型資料代入整數型暫存器，有可能會造成捨入誤差。

只有邏輯值 (位元型暫存器或 true/false) 能被代入位元型暫存器。若將邏輯值以外的數值代入位元型暫存器，會與 0 或 0.0 互相比較，再將其真偽轉換為被代入的編碼。

位元型不得代入位元型以外的暫存器。

範例

```
MW00001 = MW00002;              OK
MF00000 = MW00002 / 345;        OK
MB000010 = MB000010;            OK
MW00010 = MB0000101;            NG
MW00001 = true;                 NG
```

函數

函數所對應的引數及回傳值取決於運動控制器的函數規格。

換句話說，就是當您輸入整數及整數型暫存器到 sin()、cos() 和 atan() 後，將回傳一個整數的輸出值，若是輸入實數及實數型暫存器，則會回傳一個實數的輸出值。

tan() 的引數為實數，若是輸入整數型暫存器，將被當作實數型資料來處理。

範例

```
MW00001 = sin(MW00002);          OK
MF00002 = cos(MF00000×3.14);     OK
MW00001 = -atan(MF00002);        OK
```

括弧

■ 組合

利用 「(」與 「)」即可組合多個式子。

範例

```
MW00001 = -(MW00002 + 10) / (MW00003 - MW00005); OK
```

■ 陣列

和 C 語言一樣，可利用 「[」與 「]」來指定陣列。

範例

```
MW00001 = MW00002[100];          OK
MW00001 = MW00002[MW00003];      OK
MB000010 = MB000020[0]：         OK
```

運動參數相關注意事項

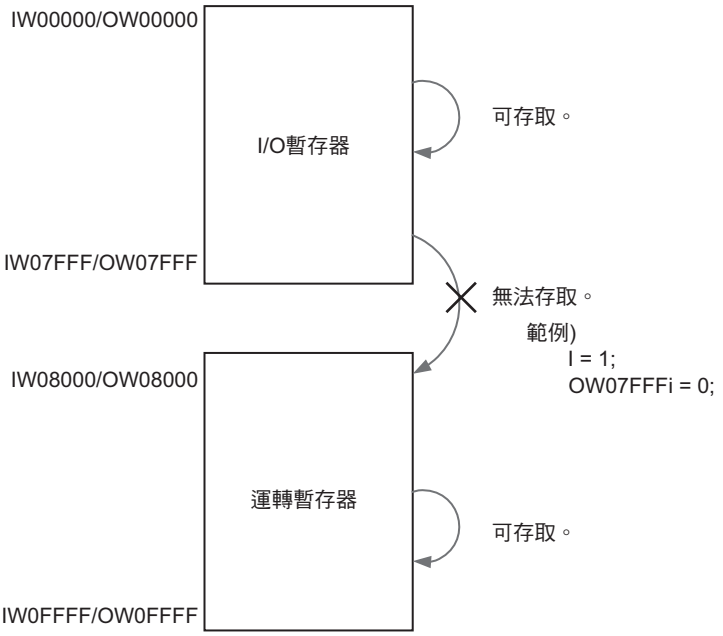
附錄 D

本章將針對運動參數相關注意事項加以說明。

接下來將說明運動參數相關注意事項。

■ 使用索引時，請勿再從 I/O 暫存器參照運轉暫存器。

I/O 暫存器和運轉暫存器無法被配置到連續的記憶體中。
使用索引時，必須能夠存取 I/O 暫存器或運轉暫存器所在範圍內的所有暫存器。



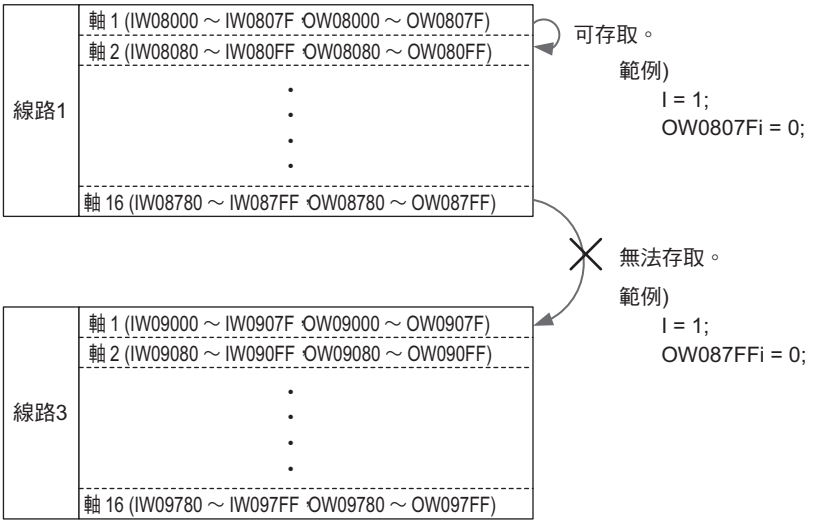
■ 使用索引時，請勿參照其他線路的運轉暫存器。

不同線路的運轉暫存器無法被配置到連續的記憶體中。

使用索引時，必須能夠存取每個線路的運轉暫存器所在範圍內的所有暫存器。

只要線路相同，軸能對不同的運轉暫存器進行存取。

線路編號	1 軸	2 軸	• • •	16 軸
1	OW08000 ～ OW0807F	OW08080 ～ OW080FF	• • •	OW08780 ～ OW087FF
3	OW09000 ～ OW0907F	OW09080 ～ OW090FF	• • •	OW09780 ～ OW097FF
5	OW0A000 ～ OW0A07F	OW0A080 ～ OW0A0FF	• • •	OW0A780 ～ OW0A7FF
7	OW0B000 ～ OW0B07F	OW0B080 ～ OW0B0FF	• • •	OW0B780 ～ OW0B7FF
9	OW0C000 ～ OW0C07F	OW0C080 ～ OW0C0FF	• • •	OW0C780 ～ OW0C7FF
11	OW0D000 ～ OW0D07F	OW0D080 ～ OW0D0FF	• • •	OW0D780 ～ OW0D7FF
13	OW0E000 ～ OW0E07F	OW0E080 ～ OW0E0FF	• • •	OW0E780 ～ OW0E7FF
15	OW0F000 ～ OW0F07F	OW0F080 ～ OW0F0FF	• • •	OW0F780 ～ OW0F7FF
17	OW18000 ～ OW1807F	OW18080 ～ OW180FF	• • •	OW18780 ～ OW187FF
19	OW19000 ～ OW1907F	OW19080 ～ OW190FF	• • •	OW19780 ～ OW197FF
21	OW1A000 ～ OW1A07F	OW1A080 ～ OW1A0FF	• • •	OW1A780 ～ OW1A7FF
23	OW1B000 ～ OW1B07F	OW1B080 ～ OW1B0FF	• • •	OW1B780 ～ OW1B7FF
25	OW1C000 ～ OW1C07F	OW1C080 ～ OW1C0FF	• • •	OW1C780 ～ OW1C7FF
27	OW1D000 ～ OW1D07F	OW1D080 ～ OW1D0FF	• • •	OW1D780 ～ OW1D7FF
29	OW1E000 ～ OW1E07F	OW1E080 ～ OW1E0FF	• • •	OW1E780 ～ OW1E7FF
31	OW1F000 ～ OW1F07F	OW1F080 ～ OW1F0FF	• • •	OW1F780 ～ OW1F7FF


線路1
軸 1 (IW08000 ～ IW0807F、OW08000 ～ OW0807F)
軸 2 (IW08080 ～ IW080FF、OW08080 ～ OW080FF)
軸 16 (IW08780 ～ IW087FF、OW08780 ～ OW087FF)
可存取。
範例)
I = 1;
OW0807Fi = 0;
無法存取。
範例)
I = 1;
OW087FFi = 0;
線路3
軸 1 (IW09000 ～ IW0907F、OW09000 ～ OW0907F)
軸 2 (IW09080 ～ IW090FF、OW09080 ～ OW090FF)
軸 16 (IW09780 ～ IW097FF、OW09780 ～ OW097FF)

運動控制器規格一覽表

附錄 E

本章將說明適用於運動控制器之程式規格。

下表為運動控制器程式之相關規格。

規格		CPU 單元 /CPU 模組	備註
運動程式	程式數	最多 512 個	最多可編寫共 512 個運動程式 / 序列程式。
	群組數	16 群組	–
	任務數	最多 32 個任務 (可同時執行的運動程式數)	–
	並列處理數 (每個任務)	最多 8 個並列 (有以下 4 種模式可供選擇) ・主程式 4 並列 x 副程式 2 並列 ・主程式 8 並列 x 副程式 1 並列 ・主程式 2 並列 x 副程式 4 並列 ・主程式 1 並列 x 副程式 8 並列	使用 MPE720，即可切換模式。
	執行登錄	・利用階梯圖程式來執行 MSEE 指令 ・使用 M-EXECUTOR	–
	啟動方法	因檢測到控制訊號 Bit 0 (要求啟動程式) 啟動而啟動程式	–
	對定位速度進行覆寫	設定範圍為 0.01% ~ 327.67%	–
	動作模式	ABS/INC 模式	使用專用指令 (ABS/INC)，即可切換模式。
	指令單位	・SVC/SVC32/SVC-01/SVB-01/SVR/SVR32 pulse，mm，deg，inch，μm ・SVA-01/PO-01 pulse，mm，deg，inch	–
	最小指令單位	・pulse 1 ・mm，deg，inch，μm 1，0.1，0.01，0.001，0.0001，0.00001	–
	指令範圍	-2147483648 ~ +2147483647 (32 位元附加符號)	–
	同步控制軸數 (每個任務)	・定位、線性內插、原點復歸、附略過 (Skip) 功能線性內插、指定時間定位 最多 32 軸 ・循環內插 2 軸 ・螺旋內插 3 軸 ・外部定位 1 軸	–
	同步控制攝影機數	最多 32 個攝影機	–
	同步控制機構數	最多 4 個機構	–
序列程式	程式數	最多 512 個 (利用啟動處理、高速掃描處理、低速掃描處理來選擇執行時間點)	最多可編寫共 512 個運動程式 / 序列程式。
	任務數	最多 32 任務 (可同時執行的序列程式數)	–
	並列處理數 (每個任務)	無法使用並列處理 (PFORK 指令)。	–
	執行登錄	使用 M-EXECUTOR	–
	啟動方法	利用系統啟動	將程式登錄至 M-EXECUTOR，即可利用系統啟動。

(續下頁)

(續上頁)

規格		CPU 單元 /CPU 模組	備註
可存取之暫存器	M 暫存器	1048576 字元	電池組備用記憶體
	S 暫存器	65535 字元	電池組備用記憶體
	G 暫存器	2097152 字元	無法利用所有程式共用的電池來進行備份的暫存器。
	I 暫存器	65536 字元 + 設定參數 + CPUIF 專用	–
	O 暫存器	65536 字元 + 監控參數 + CPUIF 專用	–
	C 暫存器	16384 字元	–
	D 暫存器	可指定 0 ~ 16384 字元	各 DWG 內固有的內部暫存器。僅適用於相對應之 DWG。

索引

7 劃

8 劃

9 劃

10 劃

11 劃

修訂記錄

修訂相關資訊及資料編號等刊載於本書封底右下方。

資料編號　YTWMNCO-15004A

Published in Taiwan 2012年 6月 11-9 ◇1-1

- 2012年 6月：發行年月日
- 11-9：初版日期
- ◇1：改版編號
- -1：改版流水號

發行年 / 月	改版編號	改版序號	項目編號	變更項目
2014 年 4 月	◇3	0	所有章節	新增：MP3300 相關編寫
			4.10	新增：IMPORTL 指令和 EXPORTL 指令相關編寫
			封底	變更：位址
2013 年 8 月	◇2	0	前言	變更：PL 記載內容
			4.5	變更：JNS 指令及 OUTS 指令的參數 RSSEL 裝置槽編號 1 ~ 4 → 1 ~ 7 SLOT 編號 0 ~ 8 → 0 ~ 9
			封底	變更：位址
2012 年 6 月	◇1	1	附錄 C.1	新增：「控制指令」運算子
2012 年 5 月		–	所有章節	全部修正
			4.10	新增：將參數寫入伺服驅動器 (MLNK-SVW)
			封底	變更：位址
2011 年 9 月	–	–	–	初版發行

運動控制器　MP3000系列

階梯圖程式
程式編寫手冊

台灣安川電機股份有限公司

事務所/技術服務中心
地址：23143新北市新店區北新路3段207號12樓
TEL: (02)8913-1333　FAX: (02)8913-1513/1519

台南服務中心
地址：74144台南市新市區創業路18號2樓
TEL: (06)505-1432　FAX: (06)505-6405

代理商 / 經銷商

YASKAWA

資料編號　YTWMNCO-15004A
Published in Taiwan　2014年12月　11-9 ㊍-0
13-12-9