Instrucciones RLL del PLC DL06

En este capítulo

Introducción

Los PLCs DL06 ofrecen una amplia variedad de instrucciones para realizar diversos tipos de operaciones. Este capítulo le muestra cómo utilizar cada instrucción normal de lógica ladder de relevadores (RLL). Además de estas instrucciones, usted puede también necesitar referirse a las instrucciones de tambor (DRUM) en el capítulo 6, o a las instrucciones de programación por etapas en el capítulo 7.

Hay dos formas de encontrar rápidamente la instrucción que usted necesita.

- Si sabe la categoría de la instrucción (booleana, comparativos booleanos, etc.) use el título en la parte superior de las páginas para encontrar las páginas que discuten las instrucciones en esa categoría.
- Si usted sabe el nombre individual de la instrucción, utilice el índice siguiente para encontrar la página que discute la instrucción.

5

Usando Instrucciones booleanas

¿Ud. se ha preguntado porqué muchos fabricantes de PLC siempre citan el tiempo de barrido para un programa booleano de 1K al usar las instrucciones booleanas? Simple. La mayoría de los programas utilizan muchas instrucciones booleanas. El PLC trabaja con estas instrucciones que son simples, diseñadas para unir contactos de entradas y salidas en serie o en paralelo, en varias combinaciones. Ya que el programa *DirectSOFT* le permite usar símbolos gráficos para construir el programa, usted no tiene que saber la abreviatura o el mnemotécnico de las instrucciones. Sin embargo, pueden ser útiles cuando vea el listado mnemotécnico de un programa. Estos mnemotécnicos también se usan como una variante con el programador portátil.

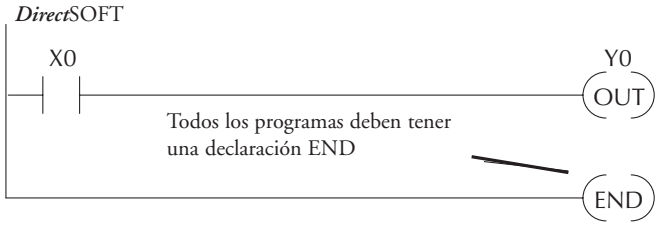
DS5	Implied
HPP	Usado

Muchas de las instrucciónes en este capítulo no son instrucciónes usadas in ***Direct***SOFT, pero son implicadas. Esto quiere decir que no son comandos desde el teclado. Sin embargo, pueden ser vistas en Mneumonic View (nemotécnicos) del programa cuando un programa en ***Direct***SOFT ha sido desarrollado y a sido aceptado (compilado). Cada instrucción listada en este capítulo tendrá una pequeña tabla como en la figura adyacente para indicar como se usa la instrucción con ***Direct***SOFT y el programador HPP.

Los siguientes párrafos muestran como estas instrucciónes son usadas para construir programas ladder simples.

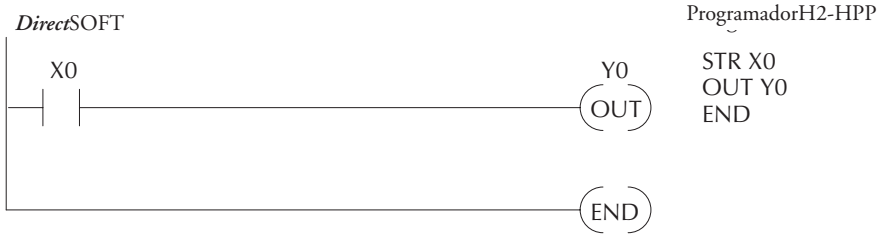
Instrucción END

Todos los programas DL06 deben tener una declaración END como instrucción final. Esto le dice a la CPU que éste es el final del programa. Normalmente, cualquiera instrucción colocada después de la instrucción END no es ejecutada. Hay excepciones a esto tal como rutinas de interrupción, etc.



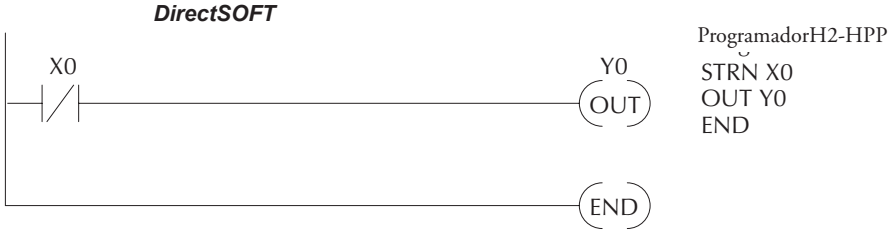
Renglones simples

Siempre se debe usar un contacto para iniciar un renglón (**rung** en inglés) que contiene contactos y bobinas (con algunas excepciones). La instrucción booleana que hace esto se llama STORE o instrucción STR y el símbolo es un contacto normalmente abierto. La salida es representada por la instrucción OUT cuyo símbolo es una bobina. El ejemplo siguiente muestra cómo entrar un solo contacto y una sola bobina de salida en un renglón.



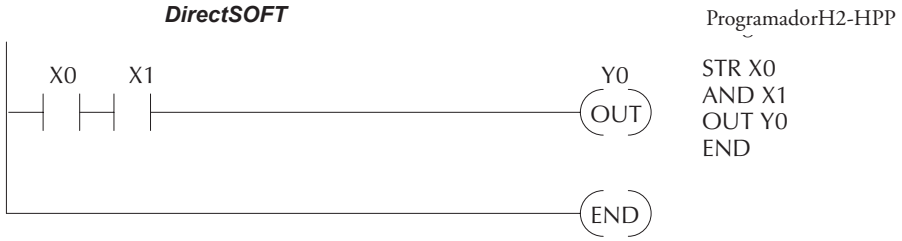
Contactos Normalmente Cerrados

Los contactos normalmente cerrados son muy comunes. estos se hacen con las instrucciones **Store, Not,** o **STRN**. El siguiente ejemplo muestra un simple renglón con un contacto normalmente cerrado.



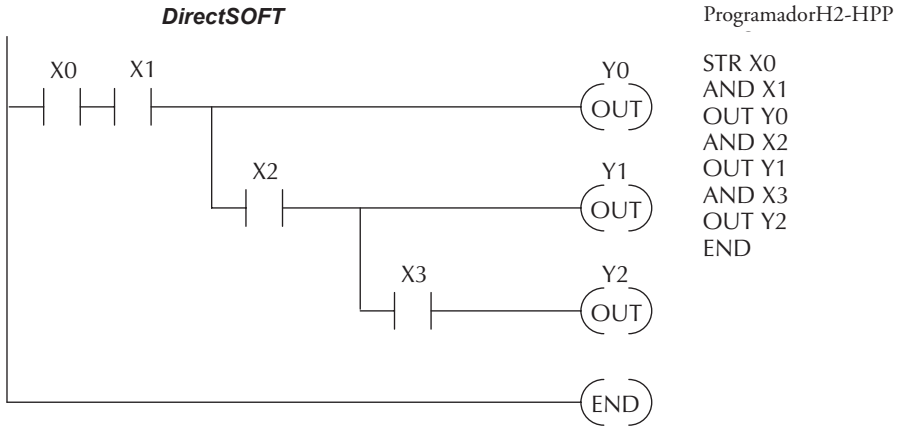
Contactos en serie

Use la instrucción AND para unir dos o más contactos en serie. El ejemplo siguente muestra dos contactos en serie y una salida en una bobina. Las instrucciones usadas serían el STR X0, AND X1 seguidos por OUT Y0.



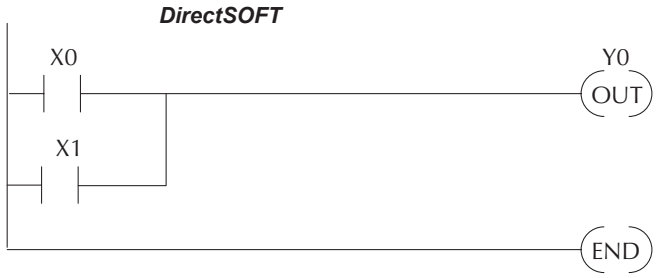
Salidas en el medio del renglón

A veces es necesario usar salidas en el medio del renglón para obtener salidas adicionales que son condiciones a otros contactos. No se deben colocaar más instrucciones en un renglón después de una rama que conecta a una salida. El ejemplo siguiente muestra cómo se puede utilizar la instrucción AND para continuar un renglón con más salidas condicionales.



Elementos en paralelo

Usted puede también tener que unir contactos en paralelo. La instrucción OR permite hacer esto. El ejemplo siguiente muestra dos contactos en paralelo y una sola salida. Las instrucciones serían el STR X0, OR X1, seguidos por OUT Y0.

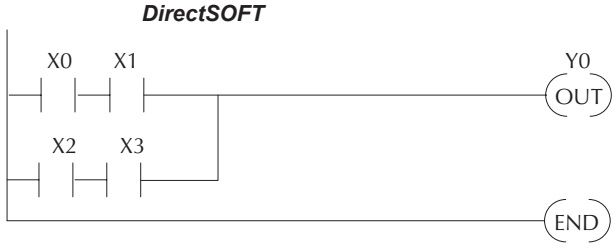


ProgramadorH2-HPP

```
STR X0
OR X1
OUT Y0
END
```


Uniendo ramas conectadas en serie y en paralelo

A menudo es necesario unir varios grupos de elementos en serie en paralelo. La instrucción OR STORE (ORSTR) permite esta operación. El ejemplo siguiente muestra un circuito con elementos en serie unidos en paralelo.

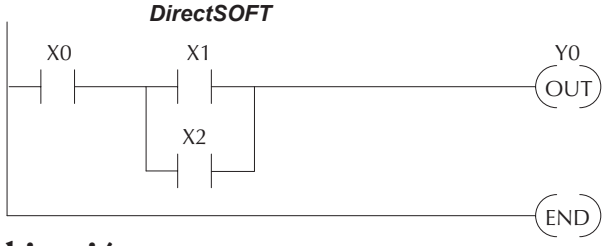


ProgramadorH2-HPP

```
STR X0
AND X1
STR X2
AND X3
ORSTR
OUT Y0
END
```

Ramas en paralelo que se unen en serie

Usted puede también unir una o más ramas paralelas en serie. La instrucción AND STORE (ANDSTR) permite esta operación. El ejemplo siguiente muestra un circuito simple con ramas de contactos en serie con contactos en paralelo.

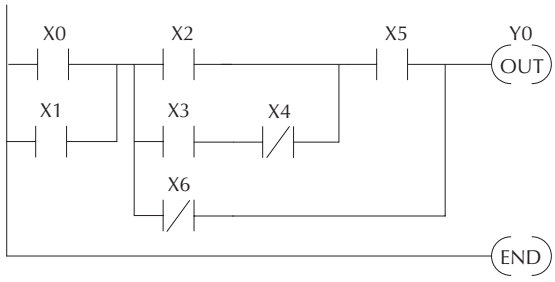


ProgramadorH2-HPP

```
STR X0
STR X1
OR X2
ANDSTR
OUT Y0
END
```

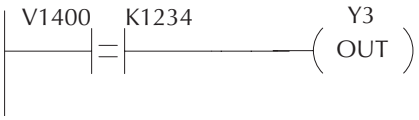
Circuitos combinación

Usted puede combinar varios tipos de ramas en serie y paralelas para solucionar la mayoría de problemas de lógica. El ejemplo siguiente muestra un circuito simple de combinación.



Comparación booleana

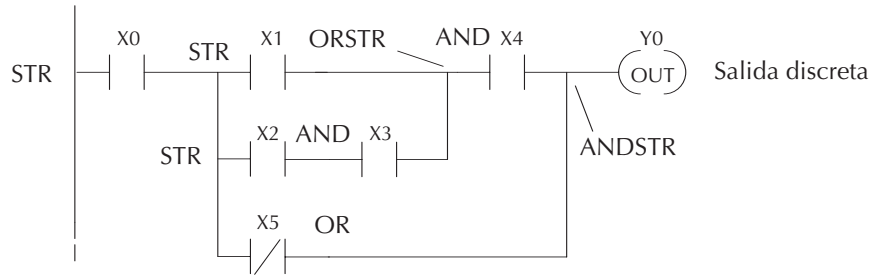
Algunos fabricantes de PLCs hacen realmente difícil el hacer una comparación simple de dos números. Algunos requieren mover los datos a varios lugares antes de que se pueda realizar realmente la comparación. Los PLCs DL06 tienen instrucciones boleanas comparativas que le permiten solucionar rápida y fácilmente este problema. La comparación booleana permite evaluación de dos valores de 4 dígitos usando contactos boleanos (los valores deben ser del mismo tipo, tal como, BCD, decimal. etc). Las evaluaciones válidas son: igual a, no igual a, igual a o mayor que, y menor que.En el ejemplo siguiente cuando el valor en la dirección de memoria V1400 es igual al valor constantee 1234, Y3 se energizará.



Stack booleano

Hay límites de cuántos elementos usted puede incluir en un renglón. Esto es porque el PLC DL06 usa una memoria o stack booleano de 8 niveles para evaluar los varios elementos de lógica. El stack booleano es un área de almacenamiento temporal que soluciona la lógica en el renglón. Cada vez que el programa encuentra una instrucción STR, la instrucción se pone en el nivel superior del stack. Cualquiera otras instrucciones STR ya en el stack booleano se van hacia abajo un nivel. Las instrucciones ANDSTR y ORSTR combinan niveles del stack booleano cuando se encuentran en la lógica. Ocurrirá un error durante la compilación del programa si la CPU encuentra un renglón que use más que los 8 niveles del stack.

El ejemplo siguiente muestra cómo se usa el stack para solucionar lógica booleana.



STR X0

1	STR X0
2	
3	
4	
5	
6	
7	
8	

STR X1

1	STR X1
2	STR X0
3	
4	
5	
6	
7	
8	

STR X2

1	STR X2
2	STR X1
3	STR X0
4	
5	
6	
7	
8	

AND X3

1	X2 AND X3
2	STR X1
3	STR X0
4	
5	
6	
7	
8	

ORSTR

1	X1 o (X2 AND X3)
2	STR X0
3	
.	.
8	

AND X4

1	X4 AND {X1 o (X2 AND X3)}
2	STR X0
3	
.	.
8	

ORNOT X5

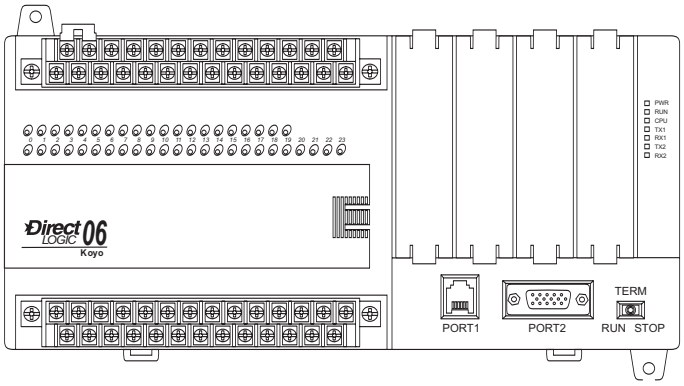
1	NOT X5 OR X4 AND {X1 OR (X2 AND X3)}
2	STR X0
3	
.	.
8	

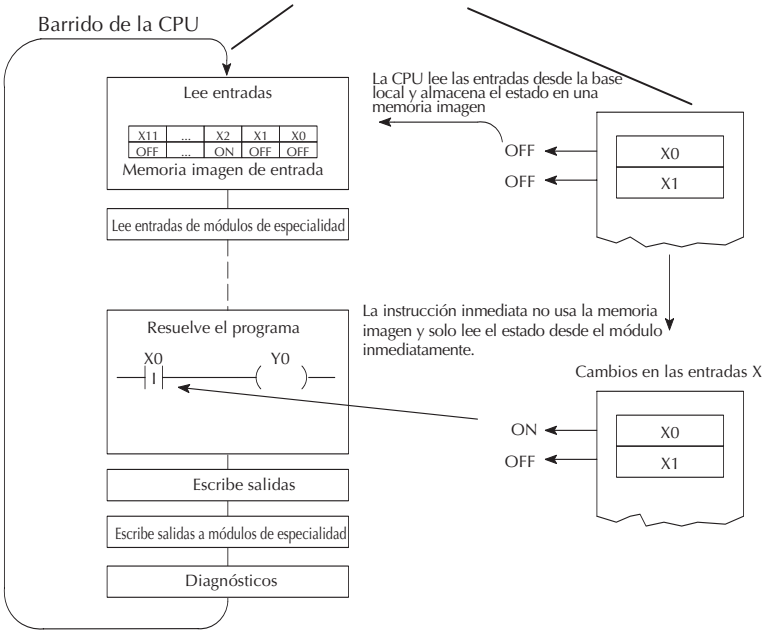
ANDSTR

Instrucciones booleanas inmediatas

El PLC DL06 puede terminar generalmente un ciclo de operación en una cuestión de milisegundos. Sin embargo, en algunos casos no se puede esperar algunos milisegundos hasta que ocurra la actualización siguiente de E/S. EL PLC DL06 tiene funciones de entradas y salidas inmediatas, que son instrucciones booleanas especiales que permiten leer directamente a las entradas y escribir directamente a las salidas durante la porción de la ejecución del programa del ciclo de la CPU. Recuerde que esto se hace normalmente durante la porción de la actualización de las entradas o de las salidas del ciclo de la CPU. Las instrucciones inmediatas demoran más para ejecutarse porque se interrumpe la ejecución del programa mientras la CPU lee o escribe las E/S. Esta función normalmente no se hace hasta que las entradas sean leídas o las salidas sean escritas en la porción del ciclo de la CPU.

NOTA: Aunque la instrucción inmediata de entrada lee el estado más corriente del punto de entrada, solamente usa los resultados para solucionar esa instrucción. No usa el nuevo estado para actualizar la memoria imagen. Por lo tanto, cualquier instrucción regular que siga usará los valores de la memoria imagen. Cualquier instrucción inmediata que siga accesará a las E/S otra vez para actualizar el estado.La instrucción



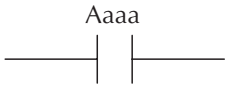


Instrucciones booleanas

La instrucción Store(STR)

DS5	Usado
HPP	Usado

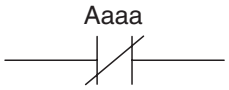
Comienza un nuevo renglón o una rama adicional en un renglón con un contacto normalmente abierto. El estado del contacto será el mismo estado como el punto de la memoria imagen asociada o localización de memoria.



La instrucción Store Not (STRN)

DS5	Usado
HPP	Usado

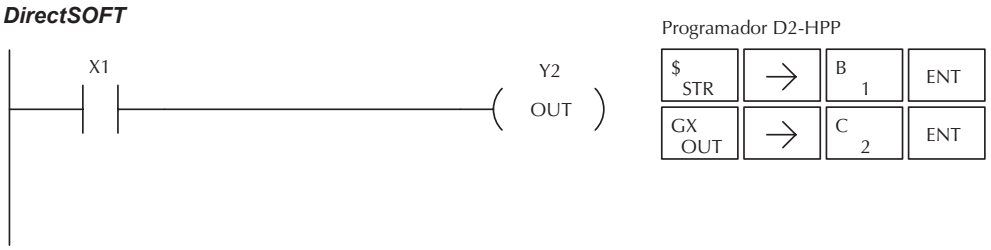
Comienza un nuevo renglón o una rama adicional en un renglón con un contacto normalmente cerrado. El estado del contacto será opuesto al estado como la memoria imagen asociada o localización de memoria.



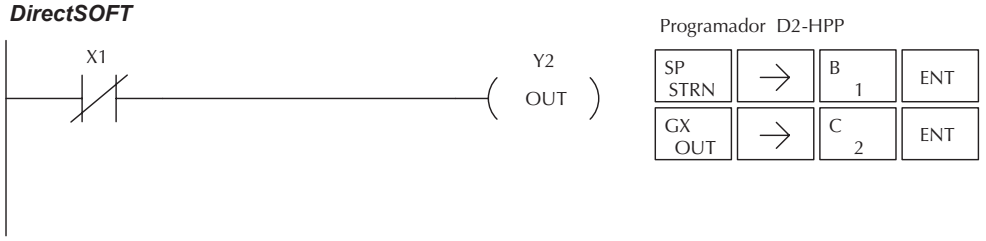
5

Tipo de operando de datos	Rango del DL06
..... A	aaa
Entradas X	0–777
Salidas ..... Y	0–777
Relevadores de control C	0–1777
Etapas ..... S	0–1777
Temporizador ..... T	0–377
Contador C ..... CT	0–177
Relevadores especialesl ..... SP	0–777

En el ejemplo siguiente, cuándo la entrada X1 está ON, se activará la salida Y2.



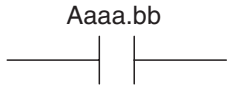
En el siguiente ejemplo, cuándo la entrada X1 está OFF, se activará la salida Y2.



Instrucción Store Bit-of-Word (STRB)

DS5	Usado
HPP	Usado

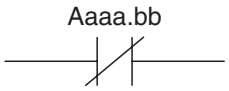
La instrucción STRB comienza un nuevo renglón o una rama adicional en un renglón con un contacto normalmente abierto. El estado del contacto será el mismo estado como el bit referenciado en la localización asociada de la memoria.



Instrucción Store Not Bit-of-Word (STRNB)

DS5	Usado
HPP	Usado

Comienza un nuevo renglón o una rama adicional en un renglón con un contacto normalmente cerrado. El estado del contacto será opuesto al estado del bit referenciado en la localización asociada de la memoria.



Tipo de operando de datos		Rango del DL06	
.....	A	aaa	bb
Memoria V	B	Vea el mapa de memoria	0 a 15
Puntero	PB	Vea el mapa de memoria	0 a 15

En el ejemplo siguiente de la instrucción STRB, cuando el bit 12 de la memoria V1400 está ON, la salida Y2 se activará. Note que en *Direct*SOFT se usa "**B**"1400.12.

En el ejemplo de STRNB, cuando el bit 12 de la memoria V1400 está apagado, se activará la salida Y2.

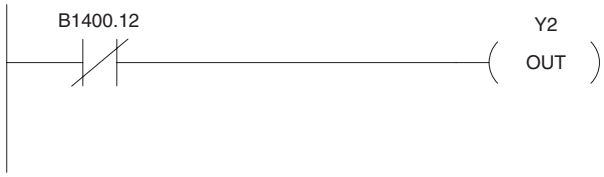
DirectSOFT

Programador D2-HPP

STR	SHFT	B	→	V	1	4	0	0
→	K	1	2	ENT				
OUT	→	2	ENT					

En el ejemplo de STRNB, cuando el bit 12 de la memoria V1400 está apagado, se activará la salida Y2.

DirectSOFT



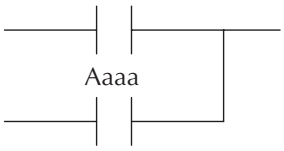
Programador D2-HPP

STRN	SHFT	B	→	V	1	4	0	0
→	K	1	2	ENT				
OUT	→	2	ENT					

La instrucción OR lógica (OR)

DS5	Implied
HPP	Usado

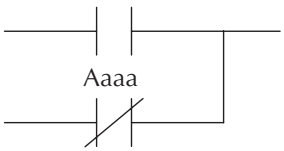
La instrucción OR hace un OR lógico con un contacto normalmente abierto en paralelo con otro contacto en un renglón. El estado del contacto será el mismo estado como el punto asociado de memoria imagen. Vea el ejemplo abajo para entender el significado .



La instrucción ORN lógica (ORN)

DS5	Implied
HPP	Usado

La instrucción lógica ORN hace un OR lógico con un contacto normalmente cerrado en paralelo con otro contacto en un renglón. El estado del contacto será opuesto al estado del punto asociado de memoria imagen.



5

Tipo de operando de datos	Rango del DL06
..... A	aaa
Entradas X	0-777
Salidas ..... Y	0-777
Relevadores de control C	0–1777
Etapas ..... S	0–1777
Temporizador ..... T	0–377
Contador ..... CT	0–177
Relevadores especialesl ..... SP	0-777

En el siguiente ejemplo, cuándo la entrada X1 o X2 está ON, se activará la salida Y5.

DirectSOFT

Programador D2-HPP

$ STR	→	B 1	ENT
Q OR	→	C 2	ENT
GX OUT	→	F 5	ENT

En el siguiente ejemplo, cuándo la entrada X1 está ON o X2 está apagada, se activará la salida Y5.

DirectSOFT

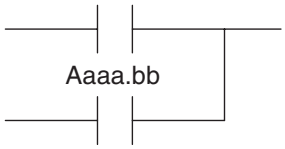
Programador D2-HPP

$ STR	→	B 1	ENT
R ORN	→	C 2	ENT
GX OUT	→	F 5	ENT

La instrucción Or Bit-of-Word (OR)

DS5	Implied
HPP	Usado

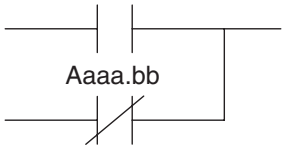
La instrucción OR hace un OR lógico de un contacto normalmente abierto en paralelo a otro contacto en un renglón. El estado del contacto será el mismo estado que el bit referido en la dirección de memoria asociada.



La instrucción Or Not Bit-of-Word (ORN)

DS5	Implied
HPP	Usado

La instrucción ORN hace un OR lógico de un contacto normalmente cerrado en paralelo a otro contacto en un renglón. El estado del contacto será el estado opuesto al bit referido en la dirección de memoria asociada.

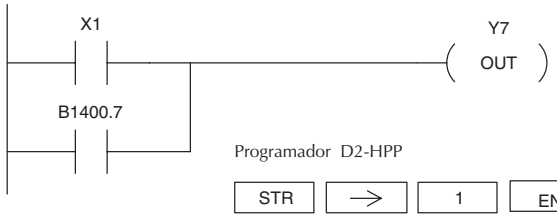


5

Tipo de operando de datos		Rango del DL06	
. . . . .	A	aaa	bb
Memoria	B	Vea el mapa de memoria	0 a 15
Puntero	PB	Vea el mapa de memoria	0 a 15

En el ejemplo siguiente de la instrucción OR, cuando la entrada X1 o el bit 7 de V1400 está activado, se energizará la salida Y5. Note que en *Direct*SOFT se usa "**B**"1400.7.

DirectSOFT

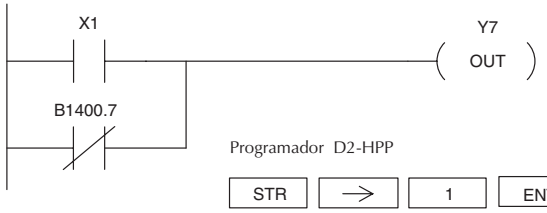


Programador D2-HPP

STR	→	1	ENT					
OR	SHFT	B	→	V	1	4	0	0
→	K	7	ENT					
OUT	→	7	ENT					

En el ejemplo siguiente de la instrucción OR, cuando la entrada X1 está activada o el bit 7 de V1400 no está activado, se energizará la salida Y5.

DirectSOFT



Programador D2-HPP

STR	→	1	ENT					
ORN	SHFT	B	→	V	1	4	0	0
→	K	7	ENT					
OUT	→	7	ENT					

La instrucción AND lógica (AND)

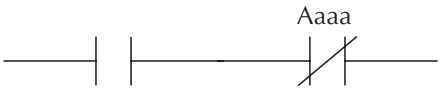
DS5	Implied
HPP	Usado

La instrucción AND lógica hace la función AND lógica en un contacto normalmente abierto en serie con otro contacto en un renglón. El estado del contacto será el mismo estado qu el de la entrada física asociada de memoria imagen.

La instrucción ANDN lógica (ANDN)

DS5	Implied
HPP	Usado

La instrucción ANDN lógica hace la función AND lógica en un contacto normalmente cerrado en serie con otro contacto en un renglón. El estado del contacto será opuesto al estado de la entrada física asociada de memoria imagen.

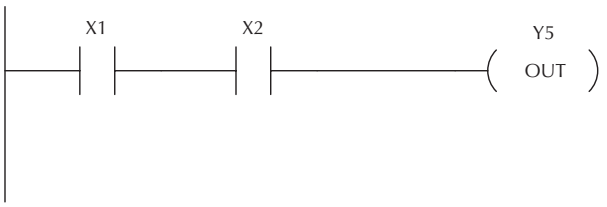


5

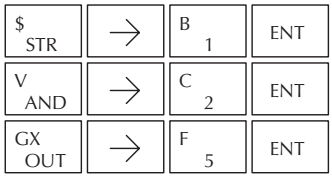
Tipo de operando de datos	Rango del DL06
..... A	aaa
Entradas X	0–777
Salidas ..... Y	0–777
Relevadores de control C	0–1777
Etapas ..... S	0–1777
Temporizador ..... T	0–377
Contador ..... CT	0–177
Relevadores especialesl ..... SP	0–777

En el siguiente ejemplo de AND, cuándo las entradas X1 y X2 están ON, se activará la salida Y5.

DirectSOFT

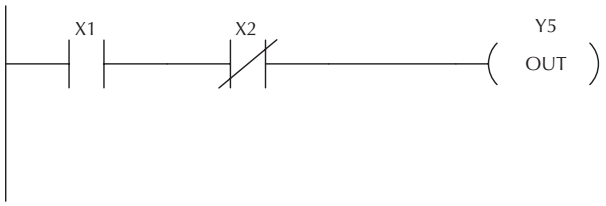


Programador D2-HPP

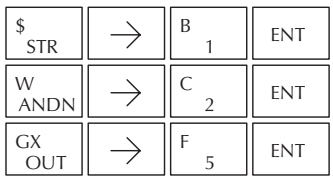


En el siguiente ejemplo de ANDN, cuándo la entrada X1 está ON y X2 está apagada, se activará la salida Y5.

DirectSOFT



Programador D2-HPP



La instrucción AND Bit-of-Word (AND).

DS5	Implied
HPP	Usado

(Bit of Word significa bit de palabra) La instrucción AND hace un AND lógico de un contacto normalmente abierto en serie con otro contacto en un renglón. El estado del contacto será el mismo estado que el bit referido en la dirección de memoria asociada.

La instrucción And Not Bit-of-Word (ANDN)

DS5	Implied
HPP	Usado

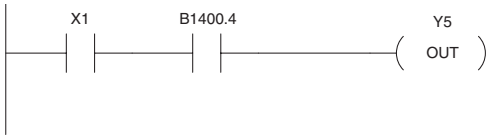
La instrucción ANDN hace un AND lógico de un contacto normalmente cerrado en serie con otro contacto en un renglón. El estado del contacto será opuesto del estado del bit referido en la dirección de memoria asociada.

5

Tipo de operando de datos		Rango del DL06	
.....	A	aaa	bb
Memoria	B	Vea el mapa de memoria	0 a 15
Puntero	PB	Vea el mapa de memoria	0 a 15

En el ejemplo siguiente de Bit of Word, cuando la entrada X1 y el bit 4 de V1400 están ON, se energizará la salida Y5. Note que en *Direct*SOFT se usa "**B**"1400.4.

DirectSOFT

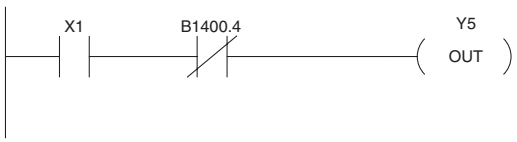


Programador D2-HPP

STR	→	1	ENT					
AND	SHFT	B	→	V	1	4	0	0
→	K	4	ENT					
OUT	→	5	ENT					

En el siguiente ejemplo de And Not Bit-of-Word, cuando la entrada X1 está ON y el bit 4 de V1400 está OFF, se energizará la salida Y5.

DirectSOFT



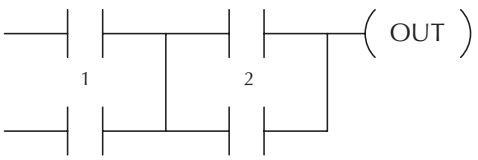
Programador D2-HPP

STR	→	1	ENT					
ANDN	SHFT	B	→	V	1	4	0	0
→	K	4	ENT					
OUT	→	5	ENT					

La instrucción AND Store (AND STR)

DS5	Implied
HPP	Usado

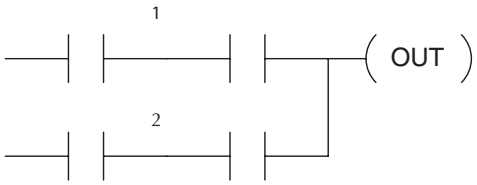
La instrucción ANDSTR hace una función AND lógica con dos ramas de un renglón en serie. Ambas ramas deben comenzar con la instrucción STR.



La instrucción OR Store (OR STR)

DS5	Implied
HPP	Usado

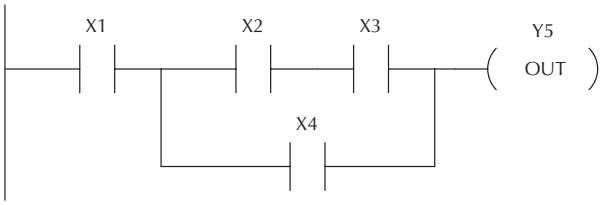
La instrucción ORSTR hace una función OR lógica con dos ramas de un renglón en paralelo. Ambas ramas deben comenzar con la instrucción STR.



5

En el siguiente ejemplo, la rama compuesta de los contactos X2, X3, y X4 se ha operado AND con la rama compuesta del contacto X1.

DirectSOFT

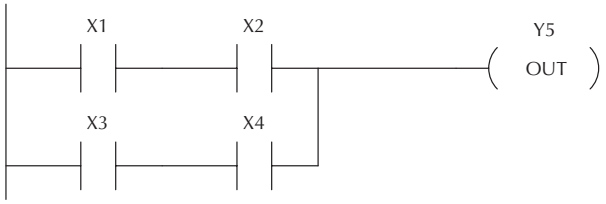


Programador D2-HPP

$ STR	→	B 1	ENT
$ STR	→	C 2	ENT
V AND	→	D 3	ENT
Q OR	→	E 4	ENT
L ANDST	ENT		
GX OUT	→	F 5	ENT

En el siguiente ejemplo OR , la rama compuesta de los contactos X1 y X2 se han operado OR con la rama compuesta de los contactos X3 y X4.

DirectSOFT



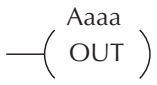
Programador D2-HPP

$ STR	→	B 1	ENT
V AND	→	C 2	ENT
$ STR	→	D 3	ENT
V AND	→	E 4	ENT
M ORST	ENT		
GX OUT	→	F 5	ENT

La instrucción OUT (OUT)

DS5	Usado
HPP	Usado

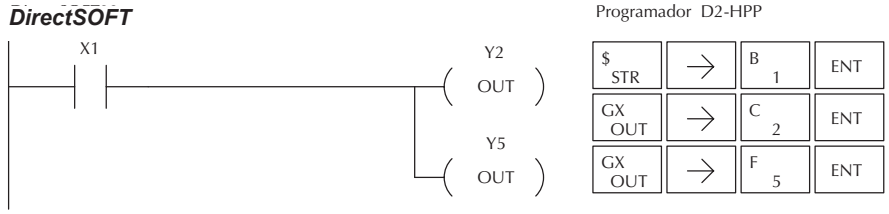
La instrucción OUT contiene el estado del renglón (ON/OFF) y deja salir el estado discreto (ON/OFF) al punto especificado de la memoria imagen.



No debe usarse más de una instrucción OUT que referencie la misma localización discreta ya que sólo la última instrucción OUT en el programa controlará el punto físico de salida. En vez de eso, use la instrucción OROUT.

Tipo de operando de datos	Rango del DL06
..... A	aaa
Entradas X	0–777
Salidas ..... Y	0–777
Relevadores de control C	0–1777

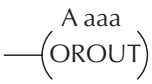
En el siguiente ejemplo Out, cuándo la entrada X1 está ON, se activarán las salidas Y2 y Y5.



La instrucción Or Out (OROUT)

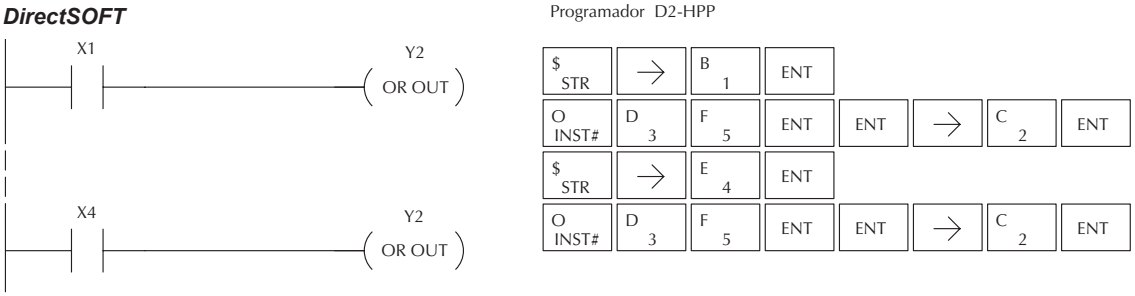
DS5	Usied
HPP	Usado

La instrucción OROUT permite que más de un renglón de lógica discreta controle una sola salida. Pueden ser usadas múltiples instrucciones OROUT que referencian la misma bobina de salida, ya que todos los contactos que controlan la salida son operados con la función OR. Si el estado de cualquier renglón está ON, la salida estará también ON.



Tipo de operando de datos	Rango del DL06
A	aaa
Entradas X	0–777
Salidas ..... Y	0-777
Relevadores de control C	0–1777

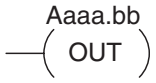
En el ejemplo siguiente, cuando una o las dos salidas X1 o X4 están ON, en cualquier parte del programa, se energizará la salida Y2 .



La instrucción Out Bit-of-Word (OUT)

DS5	Usado
HPP	Usado

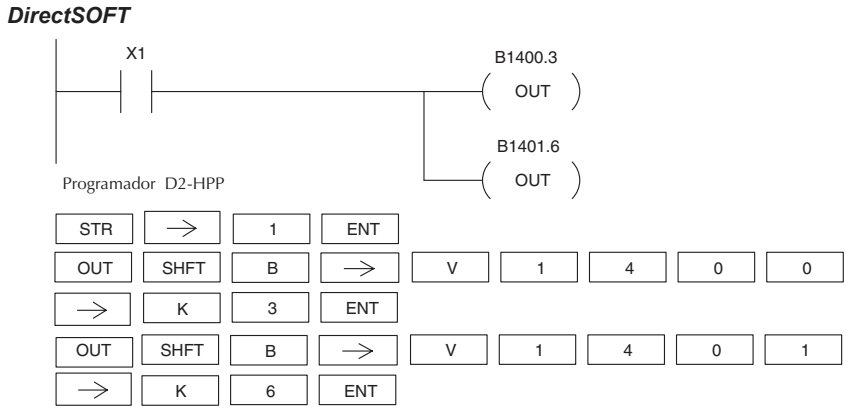
La instrucción OUT tiene el estado del renglón (ON/OFF) y produce el estado discreto (ON/OFF) del bit especificado en la dirección de memoria referida. Generalmente no deben ser usadas múltiples instrucciones OUT que se refieren al mismo bit de la misma palabra puesto que solamente la última instrucción en el programa controlará el estado del bit.



Tipo de operando de datos		Rango del DL06	
.....	A	aaa	bb
Memoria	B	Vea el mapa de memoria	0 a 15
Puntero	PB	Vea el mapa de memoria	0 a 15

5

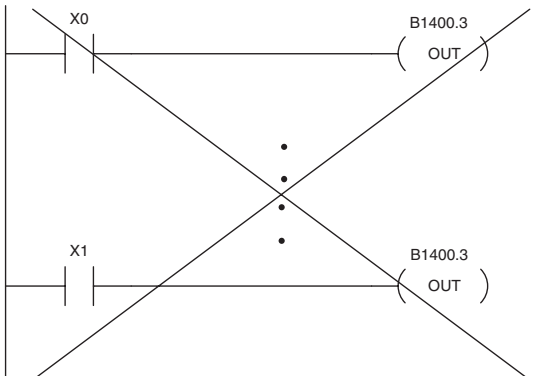
NOTA: Si la palabra Bit-of-Word se entra como V1400.3 en DirectSOFT, ser'a convertida a B1400.3. :a



palabra Bit-of-Word puede ser también ingresada como B1400.3.

En el ejemplo siguiente de la instrucción OUT, cuando la entrada X1 está encendida, el bit 3 de V1400 y el bit 6 de V1401 se activarán.

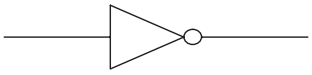
El ejemplo siguiente de Out Bit-of-Word contiene dos instrucciones Out Bit-of-Word usando el mismo bit en la misma palabra de memoria. El estado final del bit 3 de V1400 es controlado en última instancia por el último renglón de lógica en el que es referido, es decir, X1 va a forzar el estado lógico controlado por X0. ***Para evitar esta situación, no deben ser usadas múltiples instrucciones Out Bit-of-Word que usan la misma dirección en la programación.***



La instrucción Not (NOT)

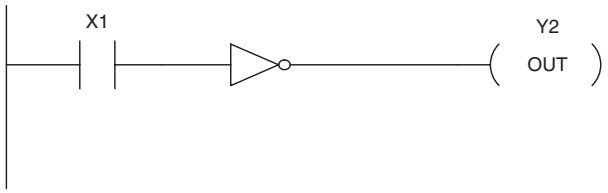
DS5	Usado
HPP	Usado

La instrucción NOT invierte el estado del renglón en el punto de la instrucción.

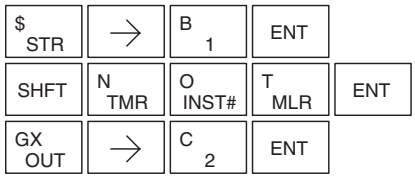


En el ejemplo siguiente cuando X1 está apagado, Y2 se activará. Esto es porque la instrucción NOT invierte el estado del renglón.

DirectSOFT



Programador D2-HPP

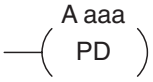


NOTE: DirectSOFT Release 1.1i and later supports the use of the NOT instrucción. The above example renglón is merely intended to show the visual representation of the NOT instrucción. The NOT instrucción can only be selected in DirectSOFT from the Instrucción Browser. *The renglón cannot be created or displayed in DirectSOFT versions earlier than 1.1i.*

La instrucción Positive Differential (PD)

DS5	Usado
HPP	Usado

La instrucción PD se conoce típicamente como "one shot". Cuándo la lógica de entrada produce una transición de OFF para ON, la salida se



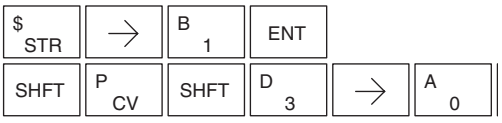
Tipo de operando de datos	Rango del DL06
.....A	aaa
Entradas X	0–777
Salidas ..... Y	0–777
Relevadores de control C	0–1777

activará por un barrido de la CPU.

En el ejemplo siguiente, cada vez que X1 hace una transición de OFF para ON, C0 se activará

DirectSOFT

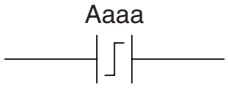
Programador D2-HPP



por un barrido.

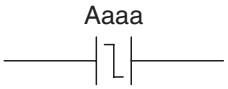
La instrucción Store Positive Differential (STRPD)

DS5	Usado
HPP	Usado



La instrucción STRPD comienza un nuevo renglón o una rama adicional en un renglón con un contacto. El contacto se cierra en un barrido de la CPU cuando el estado del punto asociado de memoria imagen hace una transición de OFF para ON. Después, el contacto permanece abierto hasta que haya otra transición de OFF para ON (el símbolo dentro del contacto representa la transición). Esta función se llama a veces "one shot".

La instrucción Store Negative Differential (STRND)



La instrucción STRND comienza un nuevo renglón o una rama adicional en un renglón con un contacto. El contacto se cierra en un barrido de la CPU cuando el estado del punto asociado de memoria imagen hace una transición de ON para OFF. Luego el contacto permanece abierto hasta que haya otra transición de ON para OFF (el símbolo dentro del contacto representa la transición).

DS5	Usado
HPP	Usado

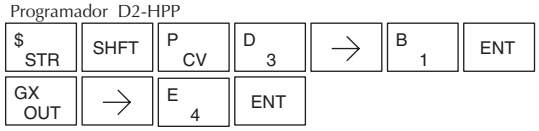
NOTE: When using DirectSOFT, these instrucciónes can only be entered from the Instrucción Browser.

Tipo de operando de datos	Rango del DL06
..... A	aaa
Entradas X	0–777
Salidas ..... Y	0–777
Relevadores de control C	0–1777
Etapas ..... S	0–1777
Temporizador ..... T	0–377
Contador ..... CT	0–177

En el ejemplo siguiente, cada vez que X1 hace la transición de OFF para ON, Y4 se activará por un barrido.

DirectSOFT

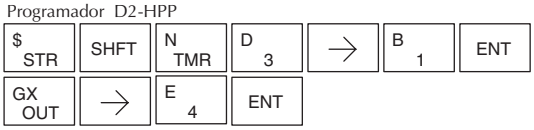
Programador D2-HPP



En el ejemplo siguiente, cada vez que X1 hace la transición de ON para OFF, la salida Y4 se activará por un barrido.

DirectSOFT

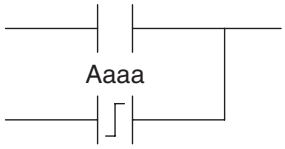
Programador D2-HPP



La instrucción Or Positive Differential (ORPD)

DS5	Implied
HPP	Usado

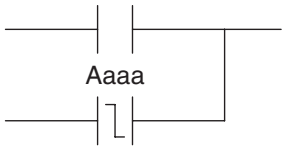
La instrucción ORPD hace on OR lógico de un contacto en paralelo a otro contacto en un renglón. El estado del contacto estará abierto hasta que el punto asociado de la memoria imagen hace una transición de OFF para ON, cerrándose en un barrido de la CPU. Después de eso, sigue abierto hasta otra transición.



La instrucción Or Negative Differential (ORND)

DS5	Implied
HPP	Usado

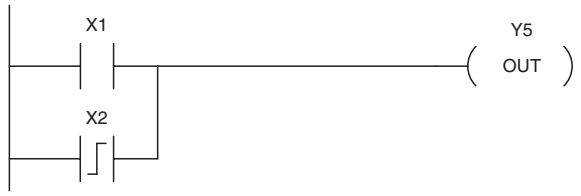
La instrucción ORPND hace on OR lógico de un contacto en paralelo a otro contacto en un renglón. El estado del contacto estará abierto hasta que el punto asociado de la memoria imagen hace una transición de ON para OFF, cerrándose en un barrido de la CPU. Después de eso, sigue abierto hasta otra transición.



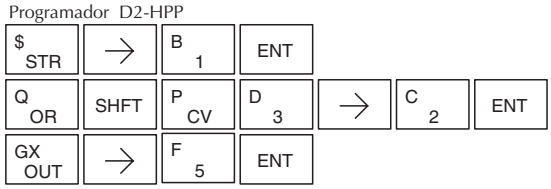
Tipo de operando de datos	Rango del DL06
..... A	aaa
Entradas X	0–777
Salidas ..... Y	0–777
Relevadores de control C	0–1777
Etapas ..... S	0–1777
Temporizador ..... T	0–377
Contador ..... CT	0–177

En el ejemplo siguiente, se activará la salida Y5 cuando X1 está ON o por un barrido de la CPU cuando haya una transición en X2 desde OFF a ON

DirectSOFT



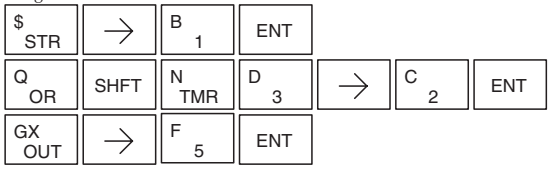
Programador D2-HPP



En el ejemplo siguiente, se activará la salida Y5 cuando X1 está ON o también por un barrido de la CPU cuando haya una transición en X2 desde ON a OFF.

DirectSOFT

Programador D2-HPP



La instrucción And Positive Differential (ANDPD)

DS5	Implied
HPP	Usado

La instrucción ANDPD hace la función AND lógica entre un contacto normalmente abierto en serie con otro contacto en un renglón. El estado del contacto estará abierto hasta que el punto asociado de la memoria imagen haga una transición de OFF para ON, cerrándolo por un barido de la CPU. Después de eso, sigue abierto hasta otra transición de OFF para ON.

La instrucción And Negative Differential (ANDND)

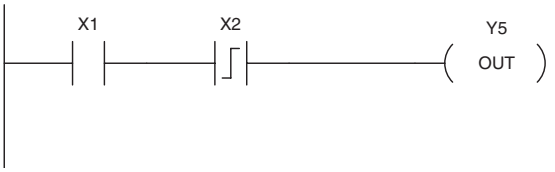
DS5	Implied
HPP	Usado

La instrucción ANDND hace la función AND lógica entre un contacto normalmente abierto en serie con otro contacto en un renglón. El estado del contacto estará abierto hasta que el punto asociado de la memoria imagen haga una transición de ON para OFF, cerrándolo por un barrido de la CPU. Después de eso, sigue abierto hasta otra transición de ON para OFF.

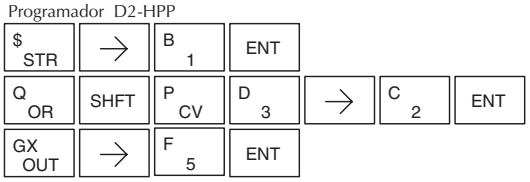
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Entradas X	0–777
Salidas . . . . . Y	0–777
Relevadores de control C	0–1777
Etapas . . . . . S	0–1777
Temporizador . . . . . T	0–377
Contador . . . . . CT	0–177

En el ejemplo siguiente, se activará Y5 cuando X1 está ON y al mismo tiempo en un barrido de la CPU cuando haya una transición en X2 desde OFF para ON.

DirectSOFT

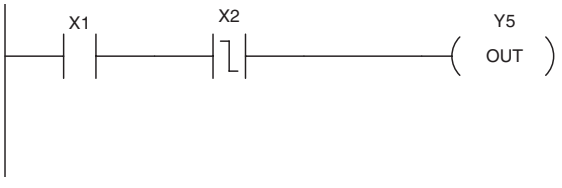


Programador D2-HPP

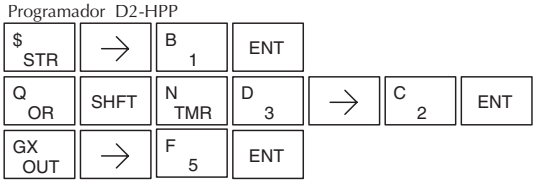


En el ejemplo siguiente, se activará Y5 cuando X1 está ON y al mismo tiempo en un barrido de la CPU cuando haya una transición en X2 desde ON a OFF.

DirectSOFT



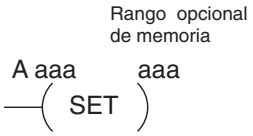
Programador D2-HPP



La instrucción Set (SET)

DS5	Usado
HPP	Usado

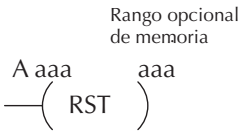
La instrucción SET coloca ON o prende un punto de memoria imagen o un rango consecutivo de memorias imagen. Una vez que la memoria se hace ON permanecerá así hasta que sea vuelta a OFF por la instrucción RESET. No es necesario que el renglón que controle la instrucción SET permanezca ON.



La instrucción Reset (RST)

DS5	Usado
HPP	Usado

Esta instrucción vuelve a 0, a OFF o apaga un punto de memoria imagen o un rango consecutivo de memorias imagen. Una vez que la localización de memoria es OFF no es necesario que el renglón permanezca ON.



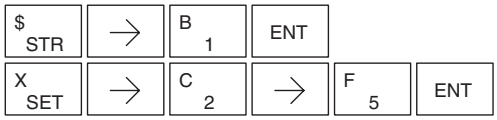
5

Tipo de operando de datos	Rango del DL06
.....A	aaa
Entradas X	0–777
Salidas .....Y	0–777
Relevadores de control C	0–1777
Etapas .....S	0–1777
Temporizador .....T	0–377
Contador .....CT	0–177

En el ejemplo siguiente cuando X1 está ON, Y2 hasta Y5 se activarán o se harán ON y permanecerán energizadas.

DirectSOFT

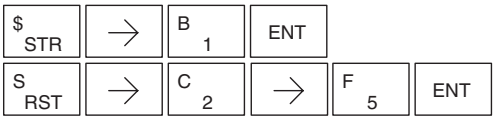
Programador D2-HPP



En el ejemplo siguiente cuando X1 está ON, las salidas Y2 hasta Y5 será vueltas a OFF y permanecerán desenergizadas.

DirectSOFT

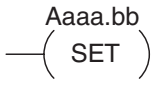
Programador D2-HPP



La instrucción Set Bit-of-Word (SET)

DS5	Usado
HPP	Usado

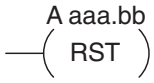
La instrucción SET activa un bit en una dirección de memoria V. Una vez que el bit se haga ON seguirá ON hasta que se repone a OFF usando la instrucción RST. No es necesario que el renglón que controla la instrucción SET permanezca activado.



La instrucción Reset Bit-of-Word (RST)

DS5	Usado
HPP	Usado

La instrucción RST repone a OFF un bit en una dirección de memoria V. Una vez que el bit se haga OFF no es necesario que el renglón que controla la instrucción RST permanezca activado.



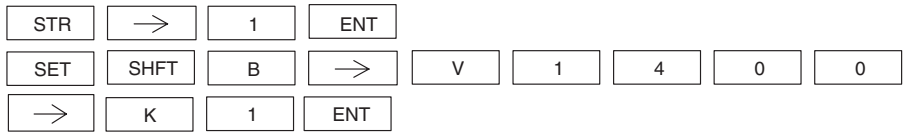
5

Tipo de operando de datos	Rango del DL06	
. . . . . A	aaa	bb
Memoria V . . . . . B	Vea el mapa de memoria	0 a 15
Puntero. . . . . PB	Vea el mapa de memoria	0 a 15

En el ejemplo siguiente cuando X1 se activa ON, el bit 1 en V1400 se va al estado ON.

DirectSOFT

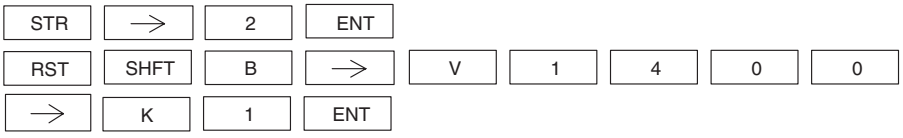
Programador D2-HPP



En el ejemplo siguiente cuando X2 se activa ON, el bit 1 en V1400 se va al estado OFF.

DirectSOFT

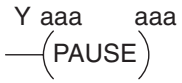
Programador D2-HPP



La instrucción Pause (PAUSE)

DS5	Usado
HPP	Usado

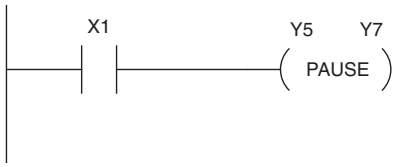
La instrucción Pause incapacita la actualización de salidas en un rango de salidas. El programa ladder continuará funcionando y actualizando la memoria imagen. Sin embargo, las salidas en el rango especificado en la instrucción Pause serán apagadas en los puntos de salidas (Colocadas OFF).



Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Salidas . . . . . Y	0–777

En el ejemplo siguiente, cuándo X1 está ON, se apagarán las salidas Y5 hasta Y7. La ejecución del programa ladder no se afectará.

DirectSOFT



Ya que el programador D2-HPP no tiene una tecla específica de Pause, usted puede utilizar el número correspondiente de la instrucción para la entrada (# 960), o puede teclear cada letra del comando.

Programador D2-HPP

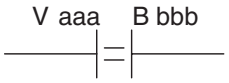
En algunos casos, usted puede querer que ciertos puntos de salida en el rango especificado en la instrucción Pause funcionen normalmente. En ese caso, use AUX 58 para cancelar la instrucción Pause.

Instrucciones de comparación booleanas

La instrucción Store If Equal (STRE)

DS5	Implied
HPP	Usado

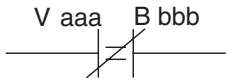
La instrucción STRE comienza una rama nueva o adicional en un renglón con un contacto de comparación normalmente abierto. El contacto estará ON cuándo el valor contenido en Vaaa es igual al valor contenido en Bbbb.



La instrucción Store If Not Equal (STRNE)

DS5	Implied
HPP	Usado

La instrucción STRE comienza una rama nueva o adicional en un renglón con un contacto de comparación normalmente abierto. El contacto estará ON cuándo el valor de Vaaa no es igual a Bbbb.

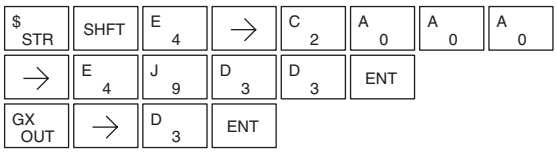


Tipo de operando de datos	Rango del DL06	
..... B	aaa	bbb
Memoria V V	Vea el mapa de memoria	Vea el mapa de memoria
Puntero ..... P	Vea el mapa de memoria	Vea el mapa de memoria
Constante ..... K	—	0–9999

En el ejemplo siguiente, cuando el valor BCD en la memoria V2000 es igual a 4933, se activará la salida Y3.

DirectSOFT

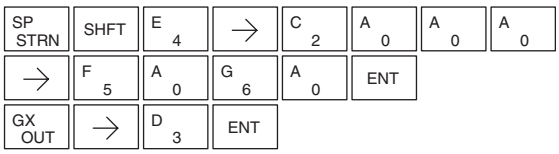
Programador D2-HPP



En el ejemplo siguiente, cuando el valor BCD en la memoria V2000 no sea igual a 5060, se activará la salida Y3.

DirectSOFT

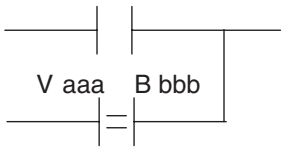
Programador D2-HPP



La instrucción Or If Equal (ORE)

DS5	Implied
HPP	Usado

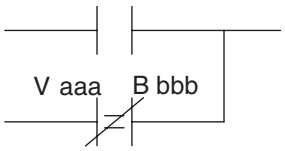
La instrucción ORE conecta un contacto comparativo normalmente abierto en paralelo con otro contacto. El contacto estará encendido cuando Vaaa = Bbbb.



La instrucción Or If Not Equal (ORNE)

DS5	Implied
HPP	Usado

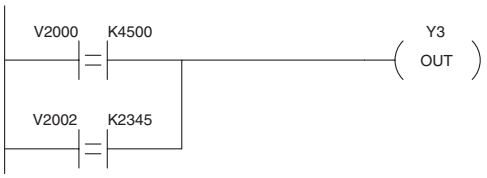
La instrucción ORNE conecta un contacto comparativo normalmente cerrado en paralelo con otro contacto. El contacto estará encendido cuando Vaaa no es igual a Bbbb.



Tipo de operando de datos	Rango del DL06	
. . . . . B	aaa	bbb
Memoria V V	Vea el mapa de memoria	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria	Vea el mapa de memoria
Constante . . . . . K	—	0–9999

En el ejemplo siguiente, cuando el valor en la dirección de memoria V2000 es igual a 4500 o V2002 es igual a 2500, se energizará la salida Y3.

DirectSOFT

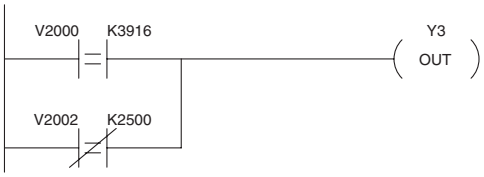


Programador D2-HPP

$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
E 4	F 5	A 0	A 0	ENT				
Q OR	SHFT	E 4	→	C 2	A 0	A 0	C 2	→
C 2	D 3	E 4	F 5	ENT				
GX OUT	→	D 3	ENT					

En el ejemplo siguiente, cuando el valor en la dirección de memoria V2000 es igual a 3916 o V2002 es diferente a 2500, se energizará la salida Y3.

DirectSOFT



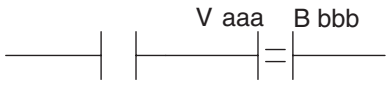
Programador D2-HPP

$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
D 3	J 9	B 1	G 6	ENT				
R ORN	SHFT	E 4	→	C 2	A 0	A 0	C 2	→
C 2	F 5	A 0	A 0	ENT				
GX OUT	→	D 3	ENT					

La instrucción And If Equal (ANDE)

DS5	Implied
HPP	Usado

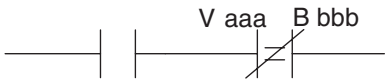
La instrucción ANDE conecta un contacto comparativo normalmente abierto en serie con otro contacto. El contacto estará encendido cuando Vaaa = Bbbb.



La instrucción And If Not Equal (ANDNE)

DS5	Implied
HPP	Usado

La instrucción ANDNE conecta un contacto comparativo normalmente cerrado en serie con otro contacto. El contacto estará encendido cuando Vaaa no es igual a Bbbb.

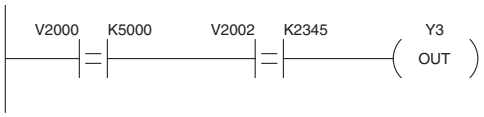


5

Tipo de operando de datos	Rango del DL06	
..... B	aaa	bbb
Memoria V V	Vea el mapa de memoria	Vea el mapa de memoria
Puntero ..... P	Vea el mapa de memoria	Vea el mapa de memoria
Constante ..... K	—	0–9999

En el ejemplo siguiente, cuando el valor BCD en la dirección de memoria V2000 es igual a 5000 o V2002 es igual a 2345, se energizará la salida Y3.

DirectSOFT

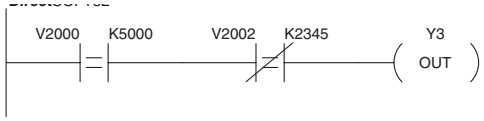


Programador D2-HPP

$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
F 5	A 0	A 0	A 0	ENT				
V AND	SHFT	E 4	→	C 2	A 0	A 0	C 2	→
C 2	D 3	E 4	F 5	ENT				
GX OUT	→	D 3	ENT					

En el ejemplo siguiente, cuando el valor BCD en la dirección de memoria V2000 es igual a 5000 o V2002 es diferente a 2345, se energizará la salida Y3.

DirectSOFT

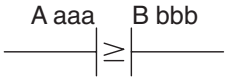


Programador D2-HPP

$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
F 5	A 0	A 0	A 0	ENT				
V AND	SHFT	E 4	→	C 2	A 0	A 0	C 2	→
C 2	D 3	E 4	F 5	ENT				
GX OUT	→	D 3	ENT					

La instrucción Comparative Store (STR)

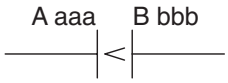
DS5	Implied
HPP	Usado



La instrucción de comparación STR comienza una rama nueva o adicional en un renglón con un contacto de comparación normalmente abierto. El contacto estará ON cuándo aaa es igual a o mayor que Bbbb.

La instrucción Store Not (STRN)

DS5	Implied
HPP	Usado



La instrucción de comparación STRN comienza una rama nueva o adicional en un renglón con un contacto de comparación normalmente cerrado. El contacto estará ON cuándo aaa sea menor que Bbbb.

Tipo de operando de datos	Rango del DL06	
A/B	aaa	bbb
Memoria V V	Vea el mapa de memoria	Vea el mapa de memoria
Puntero . . . . . p	Vea el mapa de memoria	Vea el mapa de memoria
Constante . . . . . K	—	0–9999
Timer . . . . . TA	0–377	
Contador . . . . . CTA	0–177	

En el ejemplo siguiente, cuando el valor en la dirección de memoria V2000 es mayor o igual a 1000, se energizará la salida Y3..

DirectSOFT

Programador D2-HPP

$ STR	→	SHFT	V AND	C 2	A 0	A 0	A 0
→	B 1	A 0	A 0	A 0	ENT		
GX OUT	→	D 3	ENT				

En el ejemplo siguiente, cuando el valor en la dirección de memoria V2000 es menor que 4050, se energizará la salida Y3.

DirectSOFT

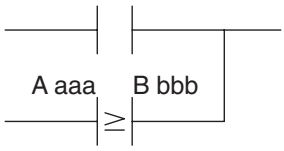
Programador D2-HPP

SP STRN	→	SHFT	V AND	C 2	A 0	A 0	A 0
→	E 4	A 0	F 5	A 0	ENT		
GX OUT	→	D 3	ENT				

La instrucciónOr comparativa(OR)

DS5	Implied
HPP	Usado

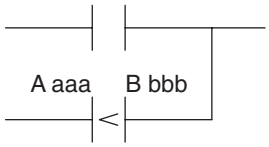
La instrucción OR comparativa conecta un contacto comparativo normalmente abierto en paralelo con otro contacto. El contacto será encendido cuando Aaaa es igual o mayor que Bbbb.



La instrucción Or Not compArativa(ORN)

DS5	Implied
HPP	Usado

La instrucción ORN comparativa conecta un contacto comparativo normalmente cerrado en paralelo con otro contacto. El contacto estará encendido cuando Aaaa es menor que Bbbb.

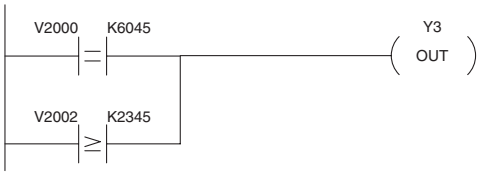


5

Tipo de operando de datos	Rango del DL06	
. . . . . A/B	aaa	bbb
Memoria V V	Vea el mapa de memoria	Vea el mapa de memoria
PUnterO . . . . . p	Vea el mapa de memoria	Vea el mapa de memoria
Constante . . . . . K	—	0–9999
Temporizador . . . . . TA	0–377	
Contador . . . . . CTA	0–177	

En el ejemplo siguiente, cuando el valor BCD en la dirección de memoria V2000 = 6045 o V2002 ≥ 2345, se energizará la salida Y3.

DirectSOFT

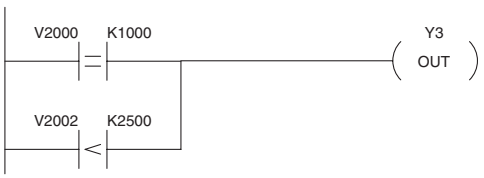


Programador D2-HPP

$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
G 6	A 0	E 4	F 5	ENT				
Q OR	→	SHFT	V AND	C 2	A 0	A 0	C 2	→
C 2	D 3	E 4	F 5	ENT				
GX OUT	→	D 3	ENT					

En el ejemplo siguiente, cuando el valor BCD en la dirección de memoria V2000 = 1000 o V2002 es menor que 2500, se energizará la salida Y3.

DirectSOFT



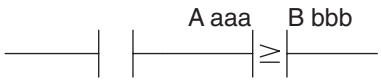
Programador D2-HPP

$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
B 1	A 0	A 0	A 0	ENT				
R ORN	→	SHFT	V AND	C 2	A 0	A 0	C 2	→
C 2	F 5	A 0	A 0	ENT				
GX OUT	→	D 3	ENT					

La instrucción And (AND)

DS5	Implied
HPP	Usado

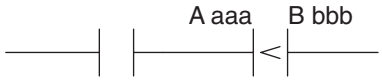
La instrucción de comparación AND conecta un contacto comparativo normalmente abierto en serie con otro contacto. El contacto estará activado a cuando Aaaa es igual o mayor que Bbbb.



La instrucción And Not (ANDN)

DS5	Implied
HPP	Usado

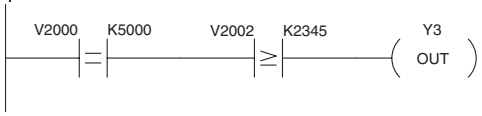
La instrucción de comparación ANDN conecta un contacto comparativo normalmente cerrado en serie con otro contacto. El contacto estará activado cuando Aaaa sea menor que Bbbb.



Tipo de operando de datos	Rango del DL06	
..... A/B	aaa	bbb
Memoria V V	Vea el mapa de memoria	Vea el mapa de memoria
Puntero ..... p	Vea el mapa de memoria	Vea el mapa de memoria
Constante ..... K	—	0–9999
Temporizador ..... TA	0–377	
Contador ..... CTA	0–177	

En el ejemplo siguiente, cuando el valor en la dirección de memoria V2000 es igual a 5000 y V2002 es mayor o igual a 2345, se energizará la salida Y3.

DirectSOFT

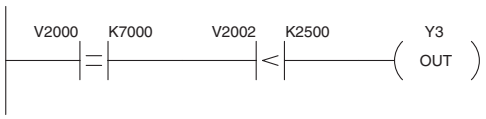


Programador D2-HPP

$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
F 5	A 0	A 0	A 0	ENT				
V AND	→	SHFT	V AND	C 2	A 0	A 0	C 2	→
C 2	D 3	E 4	F 5	ENT				
GX OUT	→	D 3	ENT					

En el ejemplo siguiente, cuando el valor en la dirección de memoria V2000 es igual a 7000 y V2002 es menor que 2500, se energizará la salida Y3.

DirectSOFT



Programador D2-HPP

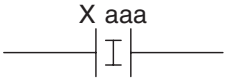
$ STR	SHFT	E 4	→	C 2	A 0	A 0	A 0	→
H 7	A 0	A 0	A 0	ENT				
W ANDN	→	SHFT	V AND	C 2	A 0	A 0	C 2	→
C 2	F 5	A 0	A 0	ENT				
GX OUT	→	SHFT	Y AND	D 3	ENT			

Instrucciones de acción inmediata

La instrucción Store Immediate (STRI)

DS5	Implied
HPP	Usado

La instrucción STRI comienza una rama nueva o adicional en un renglón. El estado del contacto será el mismo que el estado del punto asociado de la entrada *en el momento que la instrucción se ejecuta*. La memoria imagen no se actualiza.



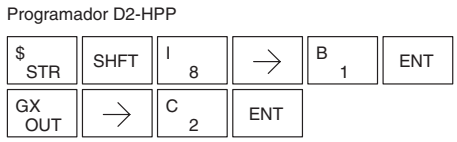
La instrucción Store Not Immediate (STRNI)

DS5	Implied
HPP	Usado

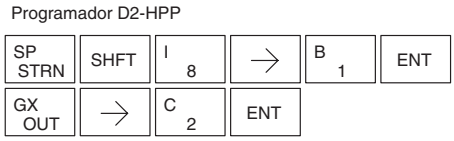
La instrucción STRNI comienza una rama nueva o adicional en un renglón. El estado del contacto será opuesto al estado del punto asociado de la entrada *en el momento que se ejecuta la instrucción*. La memoria imagen no se actualiza.

Tipo de operando de datos	Rango del DL06
	aaa
Entradas X	0–777

En el ejemplo siguiente, cuando X1 está ON, se activará la salida Y2.



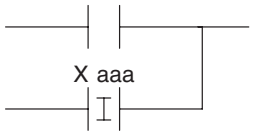
En el ejemplo siguiente, cuando X1 está OFF, se activará la salida Y2.



La instrucción Or Immediate (ORI)

DS5	Implied
HPP	Usado

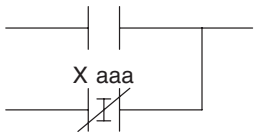
La instrucción ORI conecta dos contactos en paralelo. El estado del contacto será igual que el estado del punto asociado de la entrada *en el momento que se ejecuta la instrucción*. La memoria imagen no es actualizada.



La instrucción Or Not Immediate (ORNI)

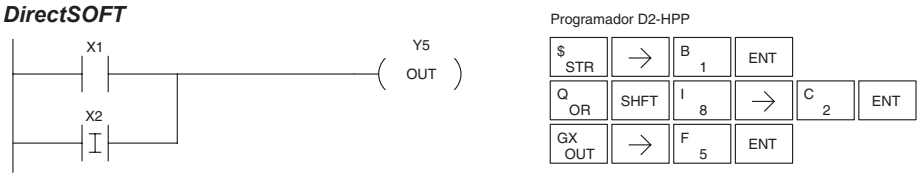
DS5	Implied
HPP	Usado

La instrucción ORNI conecta dos contactos en paralelo. El estado del contacto será opuesto al estado del punto asociado de la entrada *en el momento que se ejecuta la instrucción*. La memoria imagen no es actualizada.

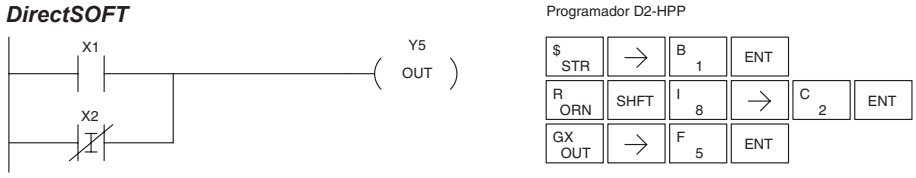


Tipo de operando de datos	Rango del DL06
	aaa
Entradas X	0–777

En el ejemplo siguiente, cuando X1 o X2 están encendidas, se energizará la salida Y5.



En el ejemplo siguiente, cuando X1 está encendida o X2 está apagada, se energizará Y5.



La instrucción And Immediate (ANDI)

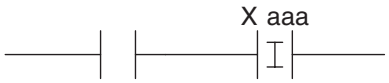
DS5	Implied
HPP	Usado

La instrucción ANDI conecta dos contactos en serie. El estado del contacto será igual que el estado del punto asociado de entrada *en el momento que se ejecuta la instrucción*. La memoria imagen no es actualizada.

La instrucción And Not Immediate (ANDNI)

DS5	Implied
HPP	Usado

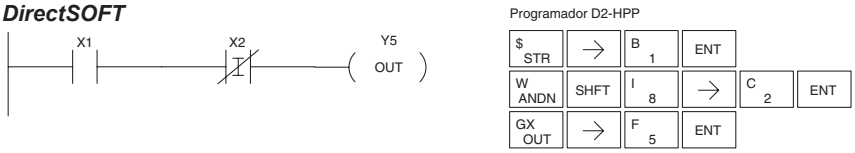
La instrucción ANDNI conecta dos contactos en serie. El estado del contacto será opuesto al estado del punto asociado de entrada *en el momento que se ejecuta la instrucción*. La memoria imagen no es actualizada.



Tipo de operando de datos	Rango del DL06
	aaa
Entradas X	0–777

En el ejemplo siguiente, cuando X1 y X2 están encendidas, se energizará Y5.

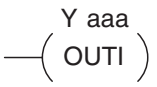
En el ejemplo siguiente, cuando X1 está encendida y X2 está apagada, se energizará Y5.



La instrucción Out Immediate (OUTI)

DS5	Usado
HPP	Usado

La instrucción Inmediata OUTI refleja el estado del renglón (ON/OFF) y las salidas del estado discreto (ON/OFF) en el punto especificado de la salida del módulo y la memoria imagen, *en el momento que se ejecuta la instrucción*.

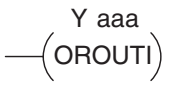


Si se usan múltiplas instrucciones OUTI que se refieren al mismo punto discreto es posible que el estado de la salida del módulo cambie múltiples veces en un barrido de la CPU. Vea OR OUT Inmediato.

La instrucción Or Out Immediate (OROUTI)

DS5	Usado
HPP	Usado

La instrucción OROUTI ha sido diseñada para usar más de un renglón de lógica discreta para controlar una sola salida. Se puede usar múltiples instrucciones OROUT con la misma bobina de salida, desde que todos los contactos de control de la salida se operan OR juntos.



Si el estado de cualquier renglón está ON *en el momento que se ejecuta la instrucción*, la salida estará también ON.

Tipo de operando de datos	Rango del DL06
	aaa
Salidas . . . . . Y	0–777

En el ejemplo siguiente, cuándo X1 está ON, prenderá el punto Y2 de la salida en el módulo de salida. Para entrar la instrucción en el programador D2-HPP, puede usar el número de la instrucción #350 como se muestra, o teclee cada letra del comando.

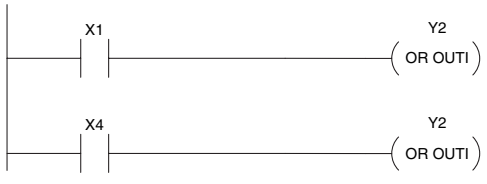
DirectSOFT

Programador D2-HPP

$ STR	→	B 1	ENT		
O INST#	D 3	F 5	A 0	ENT	ENT
→	C 2	ENT			

En el ejemplo siguiente, cuándo X1 o X4 están ON, se activará la salida Y2.

DirectSOFT



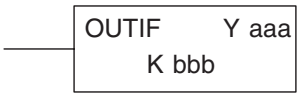
Programador D2-HPP

$ STR	→	B 1	ENT		
O INST#	D 3	F 5	A 0	ENT	ENT
→	C 2	ENT			
$ STR	→	E 4	ENT		
O INST#	D 3	F 5	A 0	ENT	ENT
→	C 2	ENT			

La instrucción Load Immediate Formatted (LDIF)

DS5	Usado
HPP	Usado

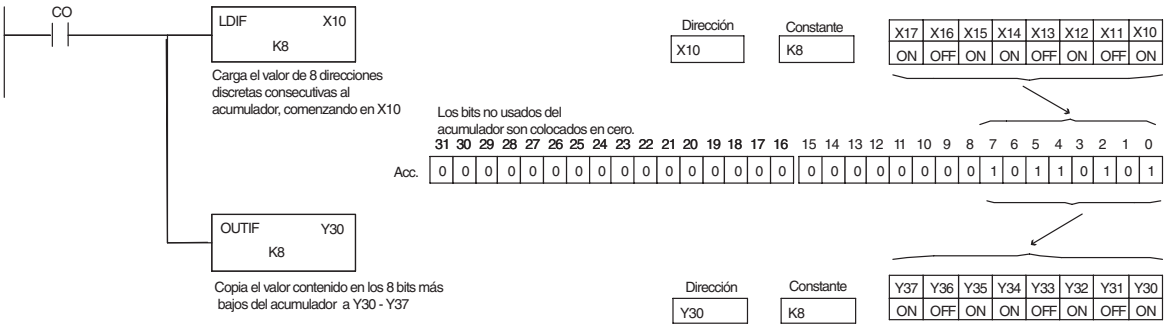
La instrucción LDFI carga un valor binario de 1 hasta 32 bits en el acumulador. El valor refleja el estado actual del módulo (s) de la entrada(s) *en el momento que la instrucción se ejecuta.* Los bits del acumulador que no son usados por la instrucción son colocados en OFF.



Tipo de operando de datos	Rango del DL06
	aaa
Salidas . . . . . Y	0-777
Constante . . . . . K	1-32

En el ejemplo siguiente, cuándo C0 está ON, el modelo binario de X10-X17 se carga en el acumulador usando la instrucción LDIF. La instrucción OUTIF se usa para copiar el número especificado de bits en el acumulador a las salidas especificadas en el módulo de salidas físicas, tales como Y30-Y37.

Esta técnica es útil para copiar rápidamente un conjunto de valores de entradas a salidas (sin esperar el barrido de la CPU).



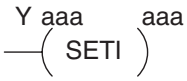
Programador D2-HPP

$ STR	→	NEXT	NEXT	NEXT	NEXT	A 0	ENT				
SHFT	L ANDST	D 3	I 8	F 5	→	B 1	A 0	→	I 8	ENT	
GX OUT	SHFT	I 8	F 5	→	D 3	A 0	→	I 8	ENT		

La instrucción Set Immediate (SETI)

DS5	Usado
HPP	Usado

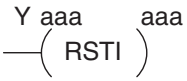
La instrucción SET Inmediate (SETI) coloca una salida física o un rango de salidas en la memoria imagen y el punto (s) correspondiente(s) de la salida *en el momento en que se ejecuta la instrucción*. Una vez que las salidas se configuran ON no es necesario que el renglón permanezca ON. La instrucción RSTI se puede usar para poner las salidas en OFF.



La instrucción Reset Immediate (RSTI)

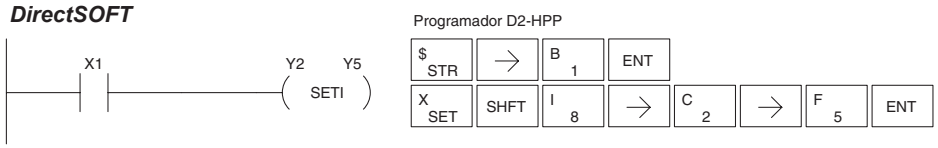
DS5	Usado
HPP	Usado

La instrucción RSTI vuelve a 0 u OFF inmediatamente o apaga una salida o un rango de salidas en la memoria imagen y el o los puntos de las salidas *en el momento en que se ejecuta la instrucción*. Una vez que las salidas son colocadas en OFF no es necesario que el renglón permanezca ON.

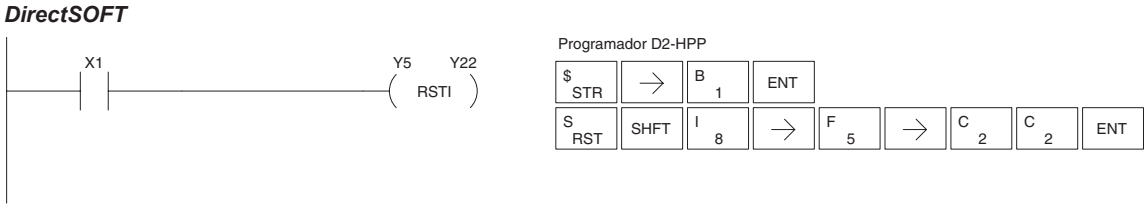


Tipo de operando de datos	Rango del DL06
	aaa
Salidas . . . . . Y	0–777

En el ejemplo siguiente, cuándo X1 está ON, se colocará ON Y2 hasta Y5 en la memoria imagen y en los puntos correspondientes de salidas físicas.



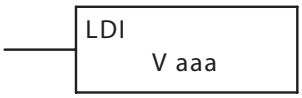
En el ejemplo siguiente, cuándo X1 está ON, Y5 hasta Y22 se colocará OFF en la memoria imagen y en el o los módulos correspondiente de salidas físicas.



La instrucción Load Immediate (LDI)

DS5	Usado
HPP	Usado

La instrucción LDI carga un valor de 16 bits de la memoria en el acumulador. El rango válido de direcciones incluye todos los puntos de entrada en la base local. El valor refleja el estado actual de los puntos de entrada *en el momento que se ejecuta la instrucción*. Esta instrucción se puede usar en vez de la instrucción de LDIF que requiere usted especificar el número de puntos de entrada.

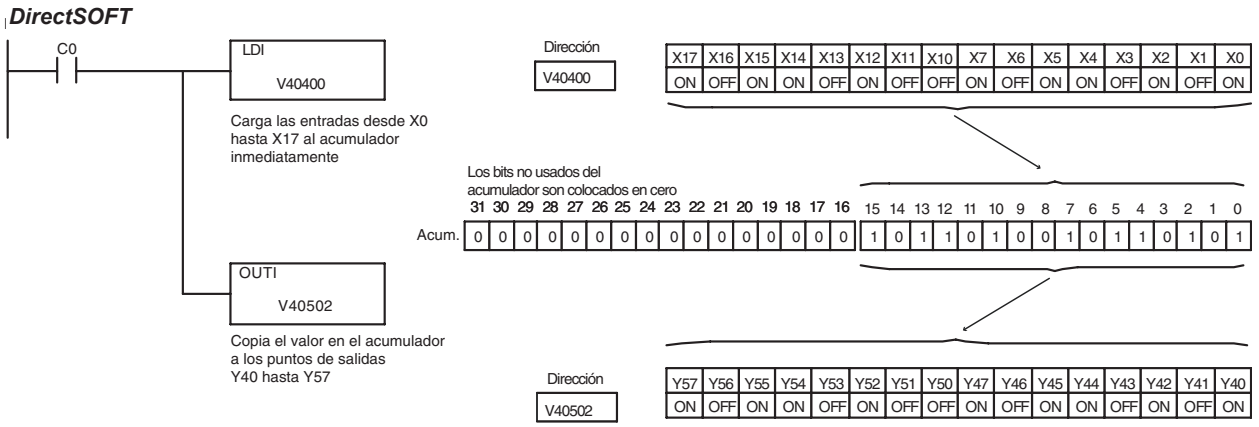


Tipo de operando de datos	Rango del DL06
	aaa
Entradas V	40400-40437

En el ejemplo siguiente, cuándo C0 está ON, se carga en el acumulador el modelo binario de X0-X17, usando la instrucción LDI.

La instrucción OUTI es usada para copiar los 16 bits en el acumulador a puntos de salidas, tales como Y40-Y57.

Esta técnica es útil para copiar rápidamente un valor de entradas a puntos de salida (sin esperar que ocurra un barrido de la CPU).



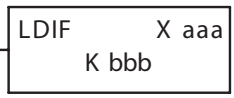
Programador D2-HPP

$ STR	→	NEXT	NEXT	NEXT	NEXT	A 0	ENT		
SHFT	L ANDST	D 3	I 8	→	E 4	A 0	E 4	A 0	A 0	ENT
GX OUT	SHFT	I 8	→	NEXT	E 4	A 0	F 5	A 0	C 2	ENT

La instrucción Load Immediate Formatted (LDIF)

DS5	Usado
HPP	Usado

La instrucción LDFI carga un valor binario de 1 hasta 32 bits en el acumulador. El valor refleja el estado actual del módulo(s) de la entrada(s) *en el momento que la instrucción se ejecuta.* Los bits del acumulador que no son usados por la instrucción son colocados en OFF.

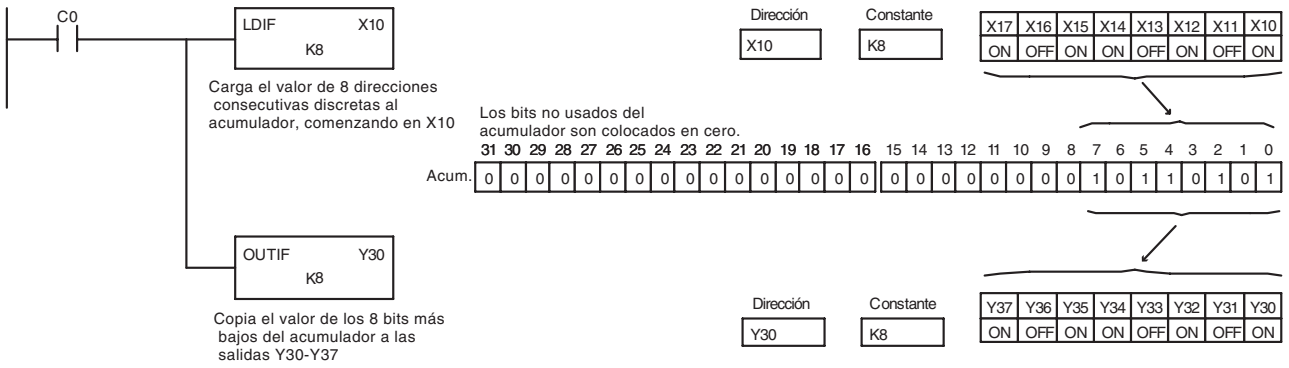


Tipo de operando de datos	Rango del DL06	
	aaa	bbb
Entradas X	0-777	- -
Constantee . . . . . K	- -	1-32

En el ejemplo siguiente, cuándo C0 está ON, el modelo binario de X10-X17 se carga en el acumulador usando la instrucción LDIF. La instrucción OUTIF se usa para copiar el número especificado de bits en el acumulador a las salidas especificadas en el módulo de salidas físicas, tales como Y30-Y37.

Esta técnica es útil para copiar rápidamente un conjunto de valores de entradas a salidas (sin esperar el barrido de la CPU).

DirectSOFT



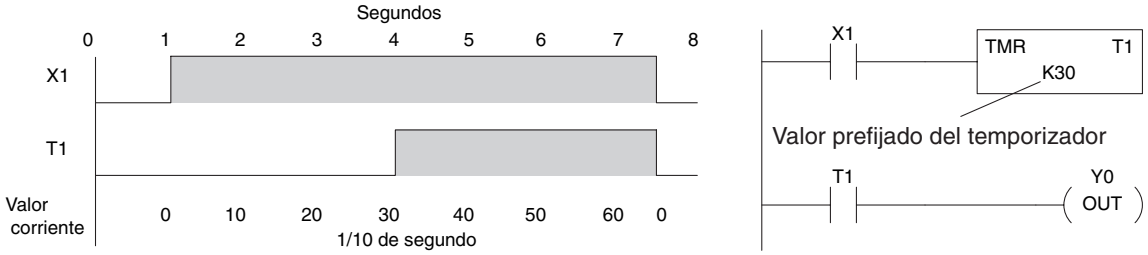
Programador D2-HPP

$ STR	→	NEXT	NEXT	NEXT	NEXT	A 0	ENT			
SHFT	L ANDST	D 3	I 8	F 5	→	B 1	A 0	→	I 8	ENT
GX OUT	SHFT	I 8	F 5	→	D 3	A 0	→	I 8	ENT	

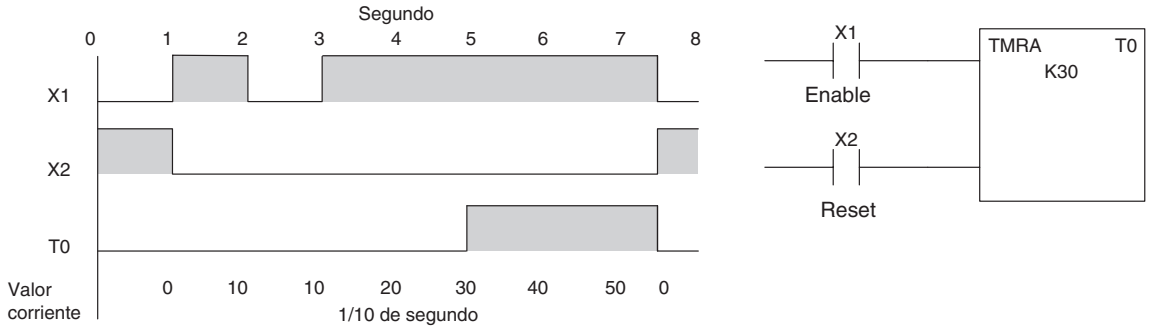
Instrucciones de temporizador, contadores y shift registers

Usando temporizadores o timers

Los temporizadores se usan para medir el tiempo de un evento por una cantidad de tiempo deseada. El temporizador de una entrada medirá el tiempo mientras la entrada está activada. Cuando la entrada cambia de activada a desactivada (ON a OFF) el valor corriente del temporizador se va a 0. Hay bases de tiempo de un décimo de segundo y un centésimo de segundo disponibles, con un tiempo máximo de 999,9 y 99,99 segundos respectivamente. Hay un bit discreto asociado a cada temporizador para indicar que el valor corriente es igual a o mayor que el valor prefijado. El diagrama que mide el tiempo abajo muestra la relación entre la entrada del temporizador, el bit discreto asociado, el valor actual, y el valor prefijado del temporizador.



Hay algunos usos que necesitan un temporizador acumulador, queriendo decir que tiene la capacidad de medir el tiempo, parar y después reanudar de donde paró. El temporizador acumulador trabaja en forma similar al temporizador regular, pero se requieren dos entradas. La entrada "enable" parte y para el temporizador. Cuando el temporizador para, se mantiene el tiempo transcurrido. Cuando el temporizador comienza otra vez, el conteo de tiempo continúa a partir del tiempo transcurrido. Cuando se activa la entrada "reset", el tiempo transcurrido es apagado y el temporizador comenzará en 0 cuando se parte nuevamente. Hay bases de tiempo de un décimo de segundo y un centésimo de segundo disponibles con un tiempo máximo de 9999999,9 y 999999,99 segundos respectivamente. El diagrama que mide el tiempo abajo muestra la relación entre la entrada del temporizador, reset del temporizador, bit discreto asociado, valor corriente y valor prefijado del temporizador.

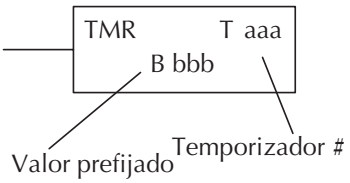


.NOTA: No se usa la coma decimal en este temprozador, pero hay una coma implicada. Los valores corriente y prefijado de todo los tipos de tempriozadores están en formato BCD.

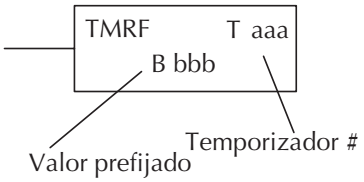
Las instrucciones temporizador (TMR) y temporizador rápido (TMRF)

DS5	Usado
HPP	Usado

La instrucción TMR es un temporizador de una entrada con base de tiempo de 0,1 segundo que cuenta tiempo hasta un máximo de 999,9 segundos. La instrucción TMRF es un temporizador de una entrada con base de tiempo de 0,01 segundo que cuenta tiempo hasta un máximo de 99,99 segundos. Estos temporizadores se activan si la lógica de entrada es verdadera (ON) y serán vueltos a 0 si la lógica de entrada es falsa (OFF).



Especificaciones de la instrucción



La Referencia del temporizador (**Taaa**): Especifica el número del temporizador.
Valor Prefijado (**Preset value**) (**Bbbb**): un valor Constantee (K) o una localización de memoria, **expresadas en BCD**.
Valor corriente: Se refiere al valor de conteo del tiempo en unidades de base de tiempo, y se puede ver en la dirección de memoria T* asociada, **valor expresado en BCD**. Por ejemplo, el valor corriente del temporizador para T3 se va a la memoria V3**.**
Bit de estado (**Status bit**): El bit de estado indica si el temporizador ya alcanzó el valor prefijado de tiempo. Se encuentra en la dirección asociada de memoria T. Estará ON si el valor corriente es igual a o mayor que el valor prefijado del temporizador específico. Por ejemplo, el bit de estado para el Temporizador 2 es T2.

NOTA: La constantee de valor prefijado (K) del temporizador puede ser cambiada usando un Programador Portátil, aún cuando la CPU está en el modo RUN. Por lo tanto, una memoria en el valor prefijado es requerida solamente si el programa ladder debe cambiar el valor prefijado.

Tipo de operando de datos	Rango del DL06	
. . . . . A/B	aaa	bbb
Temporizadores . . . . . T	0–777	—
Memoria V para valores prefijado V	—	400-677 1200–7377 7400–7577 10000-17777
Punteros (solo valor prefijado) . . . . . P	—	400-677 1200–7377 7400–7577 10000-17777
Constantes (solo valor prefijado)) . . . . . K	—	0–9999
Bits de estado de temporizadores . . . . . T/V	0–377 o V41100–41117	
Valores corrientes de temporizadores . . . . V /T*	0–377	

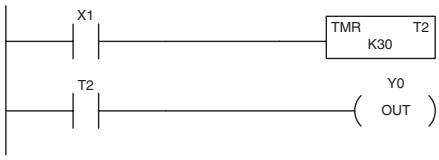
*NOTA: * Con el programador D2-HPP, los bits discretos de estado y el valor corriente del temporizador se obtienen con la misma referencia. DirectSOFT usa referencias separadas, tal como "T2" para el bit de estado y "TA2" para el valor corriente del temporizador T2.*

Usted puede realizar funciones cuando el temporizador alcanza el valor prefijado especificado usando el bit de estado. O, usando contactos de comparación para realizar funciones en intervalos diferentes de tiempo, basado en un temporizador. Los ejemplos siguientes muestran

Ejemplo de uso de temporizador con los bits de estado

En el ejemplo siguiente, se usa un temporizador con un valor prefijado de 3 segundos. El bit de estado del temporizador (T2) prenderá cuando el temporizador ha cronometrado por 3 segundos. El temporizador es vuelto a 0 cuándo X1 se apaga, haciendo OFF el bit de estado y coloca en 0 el valor corriente del temporizador.

DirectSOFT



Programador D2-HPP

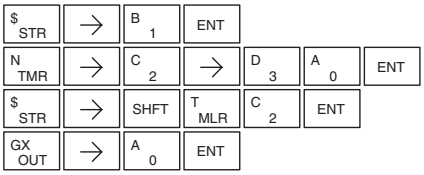
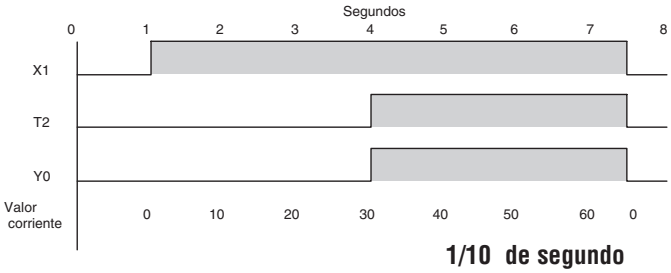


Diagrama de tiempos



Ejemplo de temporizador con contactos de comparación

En el ejemplo siguiente, se usa un temporizador con un valor prefijado de 4,5 segundos. Los contactos de comparación se usan para activar Y3, Y4, y Y5 en un intervalo de un segundo respectivamente. Cuándo X1 se apaga, el temporizador vuelve a 0 y los contactos de comparación se abren con lo cual Y3, Y4 y Y5 se apagarán.

DirectSOFT

Direct SOFT32

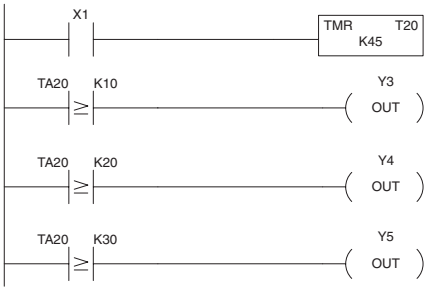
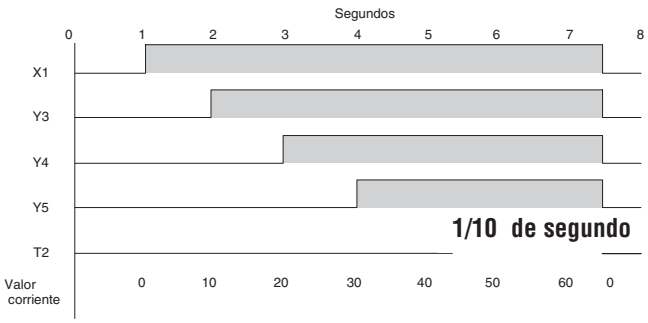
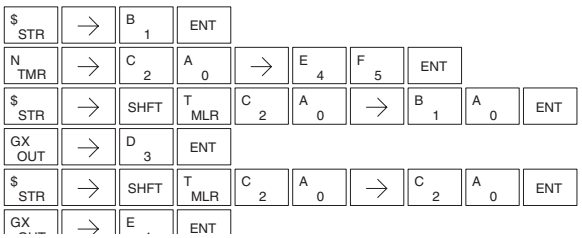


Diagrama de tiempos



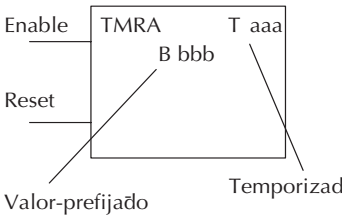
Programador D2-HPP



La Instrucción temporizador acumulador (TMRA)

DS5	Usado
HPP	Usado

La instrucción TMRA es un temporizador de base de tiempo 0,1 segundo con dos entradas,que cuenta hasta a un máximo de 9999999,9 segundos.

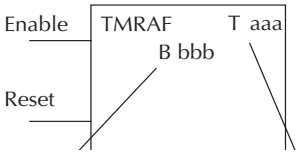


Temporizador acumulador rápido (TMRAF)

DS5	Usado
HPP	Usado

La instrucción TMRAF es un temporizador de base de tiempo de 0,01 segundo con dos entradas que cuenta hasta un máximo de 999999,99 segundos.

Cada uno de estos temporizadores usa 2 palabras de memoria. Estos temporizadores tienen dos entradas, un Enable y un RESET. El temporizador comienza a contar el tiempo cuando la entrada Enable está ON y para el conteo cuando está OFF (Sin volver a cero el valor corriente). La entrada RESET coloca en 0 el valor corriente del temporizador.



La referencia del temporizador (Taaa): Especifica el número del temporizador.
Valor prefijado (Preset value) (Bbbb):): un valor constantee (K) o una memoria V, **en BCD**.
Valor corriente: se refiere al valor de conteo del tiempo, y se puede ver en la memoria T* asociada. Por ejemplo, el valor corriente para T3 se va a la memoria V3, **y está en BCD.**
Bit de estado discreto: El bit de estado indica si el temporizador ya alcanzó el valor prefijado de tiempo. Se encuentra en la dirección asociada de memoria T. Estará ON si el valor corriente es igual a o mayor que el valor prefijado del temporizador específico. Por ejemplo, el bit de estado para el temporizador 2 es T2.

NOTA: El TMRA usa dos direcciones consecutivas de memoria para el valor de 8 dígitos y por lo tanto dos direcciones consecutivas de temporizador. Por ejemplo, si es usado TMRA 1, el próximo número disponible del temporizador es TMRA 3.

Tipo de operando de datos	Rango del DL06	
. . . . . A/B	aaa	bbb
Timers . . . . . T	0–777	—
Memoria V para valores prefijados V	—	400-677 1200–7377 7400–7577 10000-17777
Punteros (solamente valores prefijados). . . . . P	—	400-677 1200–7377 7400–7577 10000-17777
Constantes (solamente valores prefijados) . . . K	—	0–99999999
Bits de estado del temporizador . . . . . T/V	0–377 or V41100–41117	
Valores corrientes del temporizador . . . . . V /T*	0–377	

*NOTA: * Con el programador D2-HPP, los bits de estado y el valor corriente del temporizador se obtienen con la misma referencia. DirectSOFT separa las referencias, tal como "T2" para el bit de estado y "TA2" para el valor corriente del temporizador T2.*

Los ejemplos siguientes muestran dos métodos de programar los temporizadores. Uno ejecuta la función cuando el temporizador alcanza el valor prefijado usando de valor del bit de estado y el otro

Ejemplo de temporizador acumulador con bits de estado

En el ejemplo siguiente, un temporizador acumulador es usado con un valor prefijado de 3 segundos. El bit de estado temporizador (T6) prenderá cuando el temporizador ha medido un tiempo en total por 3 segundos (30 x0,1 segundo) y activará Y7.

Note en este ejemplo que el temporizador cuenta el tiempo por 1 segundo, para por 1 segundo y luego reanuda el conteo del tiempo. El temporizador volverá a 0 cuándo C10 prende, haciendo OFF el bit de estado y coloca en 0 el valor corriente del temporizador.

DirectSOFT

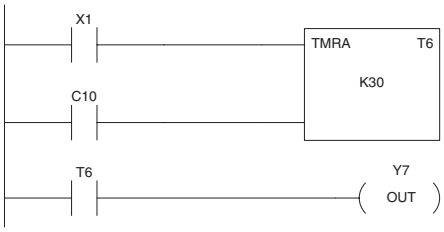
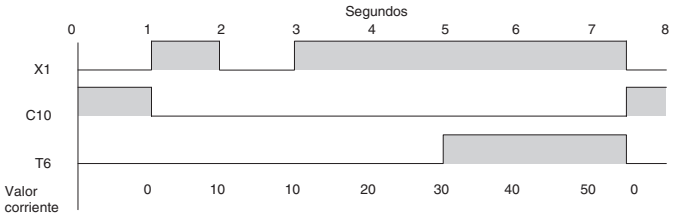
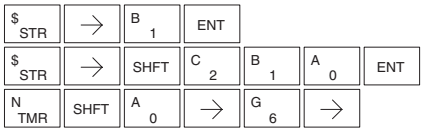


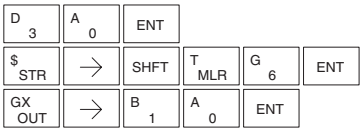
Diagrama de tiempos



Programador D2-HPP



Programador D2-HPP (continuación)



Ejemplo de temporizador acumulador usando contactos de comparación

En el ejemplo siguiente, un temporizador se usa con un valor prefijado de 4,5 segundos. Los contactos de comparación se usan para activar las salidas Y3, Y4 y Y5 en intervalos de un segundo respectivamente. Los contactos de comparación se apagarán cuando el valor corriente del temporizador vuelve a 0.

Contactos

Direct SOFT32

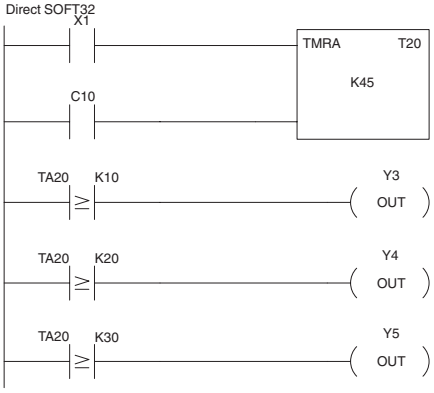
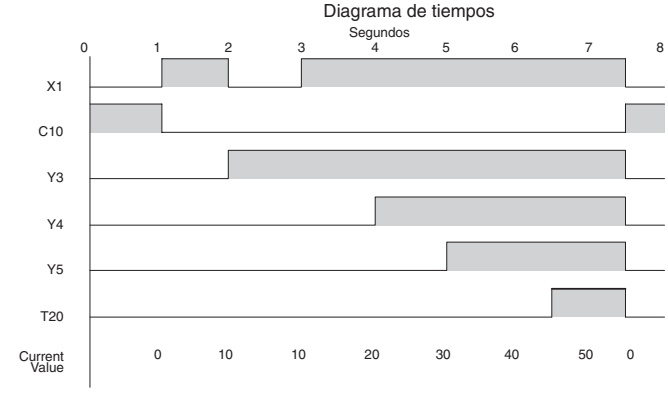
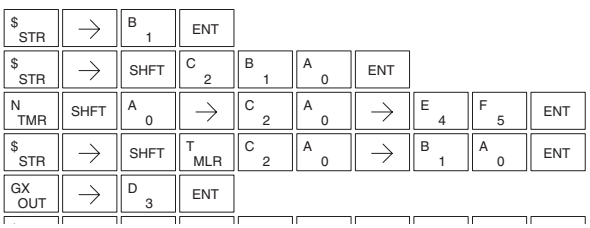


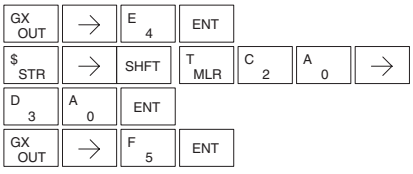
Diagrama de tiempos



Programador D2-HPP



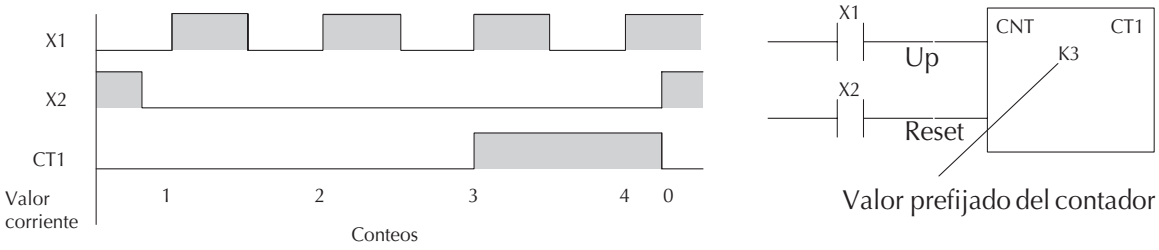
Programador D2-HPP (continuación)



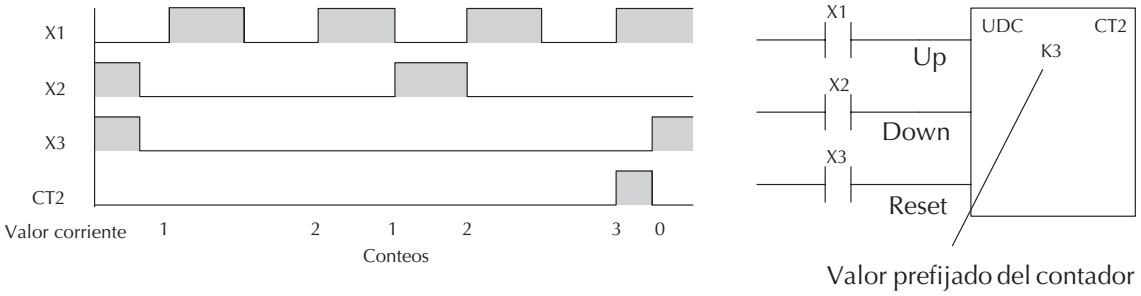
Usando Contadores

Los contadores se utilizan para contar eventos. Los contadores disponibles son contadores ascendentes, contadores incrementales/decrementales y contadores de etapas (usados con programas RLL*PLUS*).

El contador ascendente (CNT) tiene dos entradas, una entrada de conteo (UP) y una entrada RESET. El valor de conteo máximo es 9999. El diagrama de tiempos abajo muestra la relación entre la entrada , el reset, el bit de estado asociado, el valor corriente y el valor prefijado del contador.

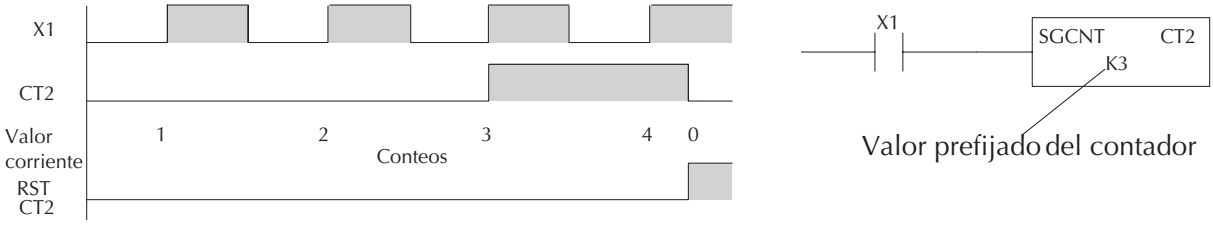


El contador incremental/decremental (UDC) tiene tres entradas, una entrada para contar ascendiendo (UP), otra para contar descendiendo (Down) y el reset. El valor de conteo máximo es 99999999. El diagrama de tiempos abajo muestra la relación entre las entradas, reset, bit de estado asociado, valor corriente y valor prefijado del contador.



Nota: El contador UDC usa dos memorias consecutivas para el valor de 8 dígitos, y por lo tanto, 2 contadores. Por ejemplo si se usa UDC CT1, el próximo contador disponible será CT3.

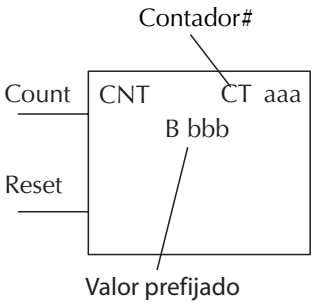
El contador de etapas (SGCNT) tiene una entrada de conteo y es vuelto a cero por la instrucción RST. Esta instrucción es útil cuando la programación se usa la programación estructurada RLL*PLUS*. El valor de cuenta máximo es 9999. El diagrama de tiempos abajo muestra la relación entre la entrada, el bit de estado asociado, el valor corriente, el valor prefijado



La instrucción Contador (CNT)

DS5	Usado
HPP	Usado

El Contador es una instrucción de dos entradas que incrementa el valor corriente cuando hay una transición lógica de la entrada COUNT de OFF para ON. Cuándo la entrada RESET del contador está ON el contador vuelve a 0. Cuando el valor corriente es igual al valor prefijado, el bit de estado del contador se hace ON y el contador continúa contando hasta un conteo máximo de 9999. El valor máximo se mantendrá hasta que el contador sea vuelto a 0.



Especificaciones de la instrucción

Referencia del contador (CTaaa): Especifica el número del contador.

Valor prefijado (Bbbb): Una constantee (K) o una dirección de memoria, **expresado en BCD**.

Valor corriente: Los valores corrientes del contador son obtenidos en el contenido de la memoria de CT* asociada, **expresado en BCD**. La localización de memoria es el número del contador + 1000. Por ejemplo, el valor contador corriente para CT3 está en la dirección de memoria V1003.

El bit de estado: El bit de estado es accesado referenciandose a la dirección asociada de memoria de CT. Estará ON si el valor es igual o mayor que el valor prefijado. Por ejemplo el bit de estado discreto para el contador 2 es CT2.

NOTE: A Memoria preset is required if the ladder program or OIP must change the preset.

Tipo de operando de datos	Rango del DL06	
. . . . . A/B	aaa	bbb
Contadores . . . . . CT	0–177	—
Memoria V (*solamente valor prefijado*) V	—	400-677 1200–7377 7400–7577 10000-17777
Punteros (*solamente valor prefijado*) . . . . . P	—	400-677 1200–7377 7400–7577 10000-17777
Constantees (*solamente valor prefijado*) . . . . K	—	0–9999
Bits de estado del contador . . . . . CT/V	0–177 o V41140–41147	
Valores corrientes del contador . . . . . V /CT*	1000-1177	

*NOTA: * Con el programador D2-HPP, los bits de estado y el valor corriente del contador se obtienen con la misma referencia. DirectSOFT separa las referencias, tal como "CT2" para el bit de estado y "CTA2" para el valor corriente del contador CT2.*

Ejemplo de contador usando el bit de estado

En el ejemplo siguiente, cuando X1 hace una transición de OFF para ON, el valor corriente del contador CT2 se incrementará en uno. Cuando el valor corriente llega al valor prefijado de 3, el bit de estado del contador CT2 prenderá y se activará Y7. Cuándo la entrada RESET C10 prende, el bit de estado del contador se apagará y el valor corriente será 0. El valor corriente para el contador CT2 se tendrá en la memoria V1002.

DirectSOFT

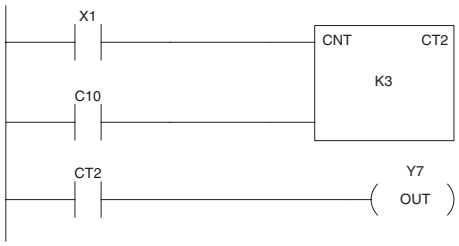
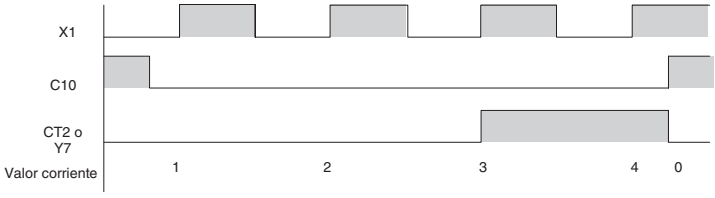


Diagrama del contador



Programador D2-HPP

$ STR	→	B 1	ENT					
$ STR	→	SHFT	C 2	B 1	A 0	ENT		
GY CNT	→	C 2	→	D 3	ENT			

Programador D2-HPP (cont.)

$ STR	→	SHFT	C 2	SHFT	T MLR	C 2	ENT	
GX OUT	→	B 1	A 0	ENT				

Ejemplo de contador usando contactos de comparación

En el ejemplo siguiente, cuándo X1 hace una transición de OFF para ON, el el valor corriente del contador CT2 se incrementará en 1. Los contactos de comparación se usan para activar las salidas Y3, Y4, y Y5 en conteos diferentes. Cuándo el contacto de entrada RESET C10 se cierra, el bit de estado se apagará y el valor corriente del contador volverá a 0 y los contactos de comparación se apagarán.

DirectSOFT

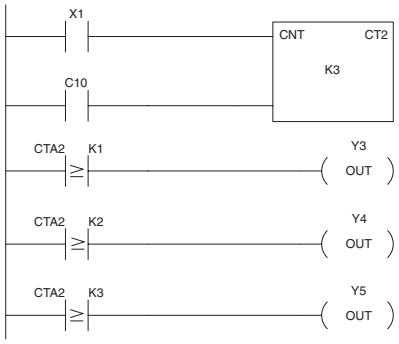
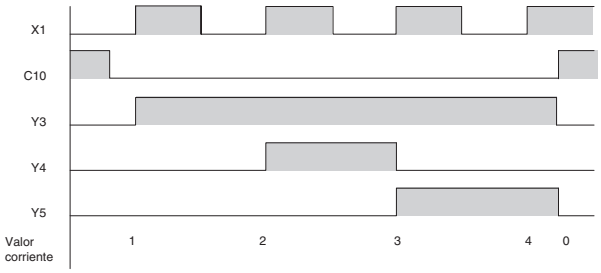


Diagrama del contador



Programador D2-HPP

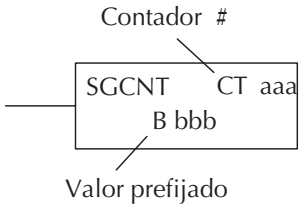
$ STR	→	B 1	ENT					
$ STR	→	SHFT	C 2	B 1	A 0	ENT		
GY CNT	→	C 2	→	D 3	ENT			
$ STR	→	SHFT	C 2	SHFT	T MLR	C 2		
→	B 1	ENT						
GX OUT	→	D 3	ENT					

Programador D2-HPP(cont)

$ STR	→	SHFT	C 2	SHFT	T MLR	C 2		
→	C 2	ENT						
GX OUT	→	E 4	ENT					
$ STR	→	SHFT	C 2	SHFT	T MLR	C 2		
→	D 3	ENT						
GX OUT	→	F 5	ENT					

La instrucción Contador de Etapas (SGCNT)

DS5	Usado
HPP	Usado



El contador de etapas es un contador de una entrada que incrementa cuando hay una transición lógica de la entrada de OFF para ON. Este contador difiere de otros contadores ya que tendrá su valor actual hasta que se use la instrucción RST (es decir, no tiene una entrada de reset, como los contadores CNT o UDC). El contador de etapas está diseñado para uso en programas RLL*PLUS,* pero puede ser usado en programas de lógica ladder de relevador. Cuándo el valor actual es igual al valor prefijado, el bit contador de estado prende y el contador continúa contando hasta un conteo máximo de 9999. El valor máximo se mantendrá hasta que el contador será vuelto a 0.

Especificaciones de la instrucción

Referencia del contador (CTaaa): Especifica el número del contador.

Valor prefijado (Bbbb): Una constantee (K) o una dirección de memoria, **en BCD**.

Valor corriente: Los valores corrientes del contador son obtenidos en el contenido de la memoria del CT* asociada, **en BCD**. La dirección de memoria es el número del contador + 1000. Por ejemplo, el valor corriente del contador CT3 se va a la dirección de memoria V1003.

El Bit de estado: El bit de estado es accesado referenciandose a la localización asociada de memoria de CT. Estará ON si el valor es igual o mayor que el valor prefijado. Por ejemplo el bit de estado discreto para el contador 2 es CT2.

NOTA: Al usar un contador dentro de etapas, las etapas deben estar activas por un barrido antes de que la entrada al contador haga una transición de 0-1. Si no es así, no hay transición verdadera y el contador no contará.

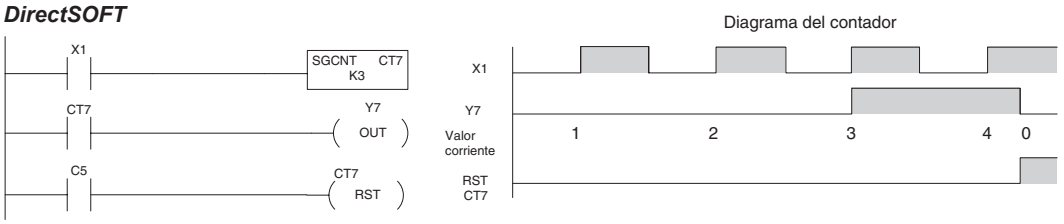
NOTA: Solamente se requiere un memoria de valor predefinido si el programa ladder o una interface de operador debe cambiar el valor.

Tipo de operando de datos	Rango del DL06	
. . . . . A/B	aaa	bbb
Contadores . . . . . CT	0–177	—
Memoria V (*solamente valor prefijado*) V	—	400-677 1200–7377 7400–7577 10000-17777
Punteros (*solamente valor prefijado*) . . . . . P	—	400-677 1200–7377 7400–7577 10000-17777
Constantes (*solamente valor prefijado*) . . . . . K	—	0–9999
Bits de estado del contador . . . . . CT*V*	0–177 o V41140–41147	
Valores corrientes del contador . . . . . V /CT*	1000-1177	

*NOTA: * Con el programador D2-HPP, los bits de estado y el valor corriente del contador se obtienen con la misma referencia. DirectSOFT separa las referencias, tal como "CT2" para el bit de estado y "CTA2" para el valor corriente del contador CT2.*

Ejemplo del contador de etapas usando el bit de estado

En el ejemplo siguiente, cuándo X1 hace una transición de OFF para ON, el valor corriente del contador de etapas CT7 incrementará en 1. Cuándo el valor corriente alcanza 3, el bit de estado del contador CT7 prenderá y se activará Y7. El bit de estado del contador CT7 permanecerá ON hasta que el contador sea vuelto a 0 usando la instrucción RST. Cuándo el contador es vuelto a 0, el bit de estado del contador se apagará y el valor corriente será 0. El valor corriente para el contador CT7 se obtendrá en la memoria V1007.



Programador D2-HPP

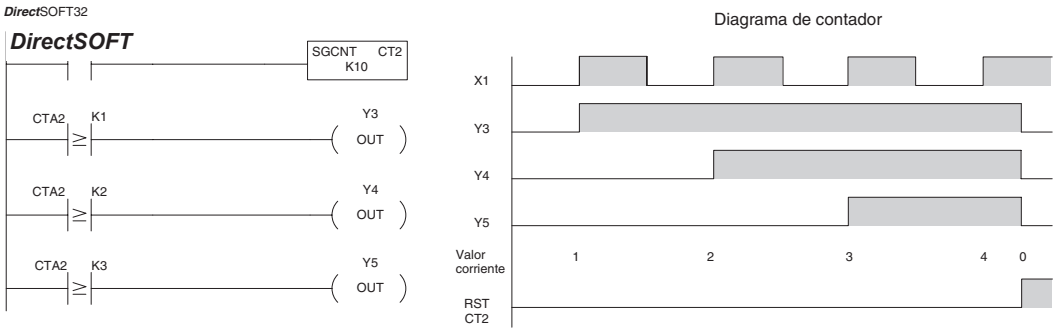
$ STR	→	B 1	ENT					
SHFT	S RST	SHFT	G 6	SHFT	GY CNT	→		
H 7	→	D 3	ENT					
$ STR	→	SHFT	C 2	SHFT	T MLR	H 7	ENT	

Programador D2-HPP (cont.)

GX OUT	→	B 1	A 0	ENT				
$ STR	→	SHFT	C 2	F 5	ENT			
S RST	→	SHFT	C 2	SHFT	T MLR	H 7	ENT	

Ejemplo de contador de etapas usando contactos de comparación

En el ejemplo siguiente, cuándo X1 hace una transición de OFF para ON, el valor corriente del contador CT2 incrementará en 1. Los contactos de comparación se usan para activar Y3, Y4 y Y5 en conteos diferentes. Aunque esto no se muestre en el ejemplo, cuando el contador usa la instrucción RST, el bit de estado del contador se apagará y el valor corriente será 0. El valor corriente para el contador CT2 se mantendrá en la memoria V1002 (o CTA2).



Programador D2-HPP

$ STR	→	B 1	ENT				
SHFT	S RST	G 6	SHFT	GY CNT	→		
C 2	→	B 1	A 0	ENT			
$ STR	→	SHFT	C 2	SHFT	T MLR	C 2	
→	B 1	ENT					
GX OUT	→	D 3	ENT				

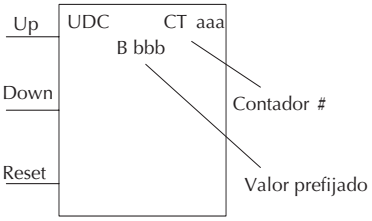
Programador D2-HPP (cont)

$ STR	→	SHFT	C 2	SHFT	T MLR	C 2
→	C 2	ENT				
GX OUT	→	E 4	ENT			
$ STR	→	SHFT	C 2	SHFT	T MLR	C 2
→	D 3	ENT				
GX OUT	→	F 5	ENT			

La instrucción Up Down Counter (UDC)

DS5	Usado
HPP	Usado

El contador UDC cuenta subiendo el conteo en la transición de falso para verdadero (OFF a ON) en la entrada UP y cuenta hacia abajo en cada transición de OFF para ON en la entrada Down. El contador vuelve a 0 cuando la entrada RESET está ON. El rango de conteo es 0-99999999. La entrada de conteo que no se usa debe estar apagada para que la entrada activa de conteo pueda funcionar.



Atención: El contador UDC usa dos direcciones de memoria para el valor corriente de 8 dígitos. Esto es, el contador UDC usa dos direcciones de memorias consecutivas. Si se usa el contador UDC CT1 en un programa, el próximo contador disponible en ese programa es CT3.

Especificación de la instrucción

Referencia del contador (CTaaa): Especifica el número del contador.

Valor prefijado(Bbbb): Valor constantee (K) o dos direcciones consecutivas de memoria V, **en BCD**.

Valor corriente: El valor corriente de conteo es un valor de palabra doble que se puede acceder referenciando las direcciones de memoria de CT* asociadas, **en BCD**. La dirección de memoria V es el número del contador + 1000. Por ejemplo, el valor corriente para el contador CT5 está en las memorias V1005 y V1006.

El bit de estado de contador: El bit de estado es accesado al referenciar la dirección asociada de memoria del contador CT. Opera estando ON si el valor es igual a o mayor que el valor prefijado. Por ejemplo el bit de estado discreto para el contador 12 sería CT12.

NOTA: UDC usa dos memorias consecutivas para el valor de 8 dígitos, por lo tanto dos localizaciones consecutivas de temporizador. Por ejemplo, si se usa UDC CT1, el número disponible siguiente es CT3.

NOTA: Solamente se requiere un memoria de valor predefinido si el programa ladder o una interface de operador debe cambiar el valor.

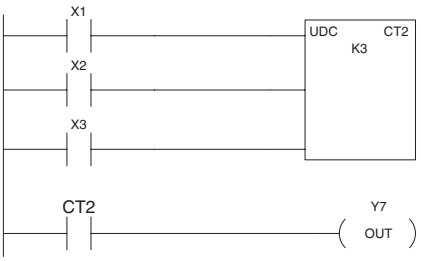
Tipo de operando de datos	Rango del DL06	
. . . . . A/B	aaa	bbb
Contadores . . . . . CT	0–177	—
Memoria V (*solamente valor prefijado*) V	—	400-677 1200–7377 7400–7577 10000-17777
Punteros (*solamente valor prefijado*) . . . . . P	—	400-677 1200–7377 7400–7577 10000-17777
Constantes (*solamente valor prefijado*) . . . . . K	—	0–9999
Bits de estado del contador . . . . . CT/V	0–177 o V41140–41147	
Valores corrientes del contador . . . . . V /CT*	1000-1177	

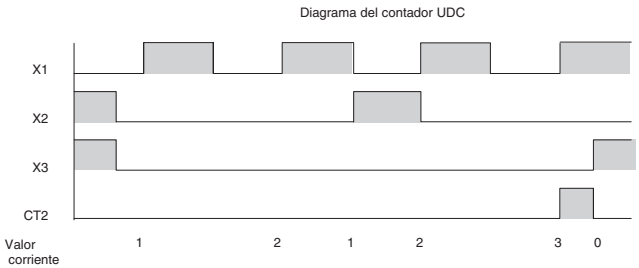
*NOTA: * * Con el programador D2-HPP, los bits de estado y el valor corriente del contador se obtienen con la misma referencia. DirectSOFT usa referencias diferentes, tal como "CT2" para el bit de*

Ejemplo de contador incremental/decremental usando el bit de estado

En el ejemplo siguiente, si X2 y X3 están apagados, cuándo X1 pase de OFF para ON el valor corriente del contador incrementará en 1. Si X1 y X3 están apagados el valor corriente del contador decrece en 1 cuándo X2 pasa de OFF para ON. Cuándo el valor de conteo alcanza el valor prefijado de 3, el bit de estado del contador prenderá. Cuándo X3 prende, el bit de estado del contador se apagará y el valor actual se hará 0.

DirectSOFT





Programador D2-HPP

$ STR	→	B 1	ENT					
$ STR	→	C 2	ENT					
$ STR	→	D 3	ENT					
SHFT	U ISG	D 3	C 2	→	C 2			

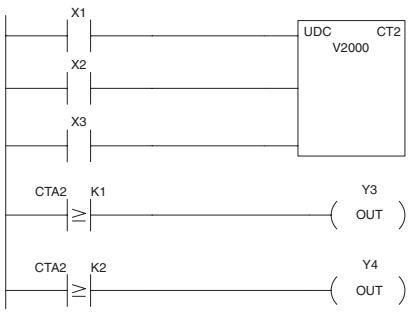
Programador D2-HPP (cont)

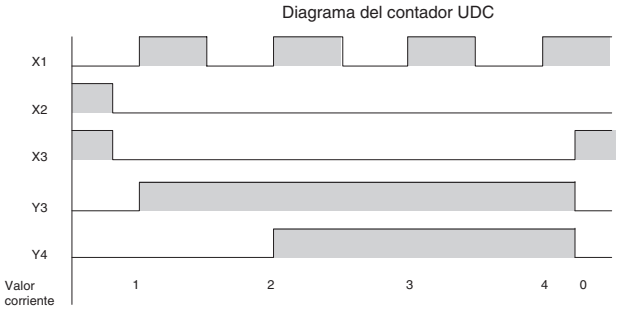
→	D 3	ENT						
$ STR	→	SHFT	C 2	SHFT	T MLR	C 2	ENT	
GX OUT	→	B 1	A 0	ENT				

Ejemplo de contador UDC con contactos de comparación

En el ejemplo siguiente, si X2 y X3 están apagados, cuándo X1 pase de OFF para ON el valor corriente del contador incrementará en 1. Si X1 y X3 están apagados el valor corriente del contador decrece en 1 cuándo X2 pasa de OFF para ON. Cuándo el valor de conteo alcanza el valor prefijado de 3, el bit de estado del contador prenderá. Cuándo X3 prende, el bit de estado del contador se apagará y el valor actual se hará 0.

DirectSOFT





Programador D2-HPP

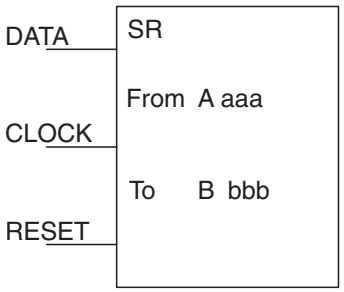
$ STR	→	B 1	ENT				
$ STR	→	C 2	ENT				
$ STR	→	D 3	ENT				
SHFT	U ISG	D 3	C 2	→	C 2	→	
SHFT	V AND	C 2	A 0	A 0	A 0	ENT	
$ STR	→	SHFT	C 2	SHFT	T MLR	C 2	

Programador D2-HPP (cont)

→	B 1	ENT				
GX OUT	→	D 3	ENT			
$ STR	→	SHFT	C 2	SHFT	T MLR	C 2
→	C 2	ENT				
GX OUT	→	E 4	ENT			

La instrucción Shift Register (SR)

DS5	Usado
HPP	Usado



La instrucción SR mueve un número predefinido bits de relevadores de control C. Los rangos de control en el bloque de bits deben comenzar al inicio de una frontera de 8 bits en bloques de 8 bits.

La instrucción SR tiene tres contactos.

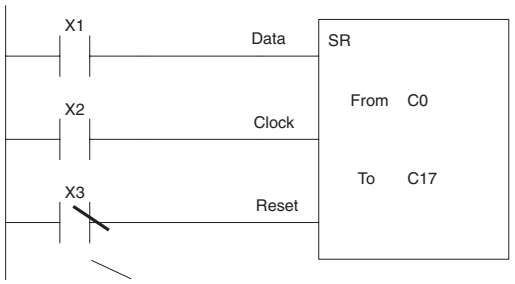
- Data — Determina si el bit a ser colocado en la primera ubicación del bit es 1 o 0.
- Clock — Mueve los bits una posición en cada transición de OFF para ON.
- Reset — Vuelve a 0 (OFF) todos los bits.

Con cada transición de OFF para ON del clock, los bits que componen el bloque son movidos una posición de bit y el estado de la entrada de datos es colocado en el estado del bit que inicia el bloque. La dirección del movimiento depende de lo que sea colocado en los campos FROM y TO. De C0 a C17 definiría un bloque de dieciséis bits para ser cambiado de la izquierda a la derecha. Con la información en FROM (de) C17 a C0 definiría un bloque de dieciséis bits, para ser movido de la derecha a la izquierda. (Vea el ejemplo abajo).

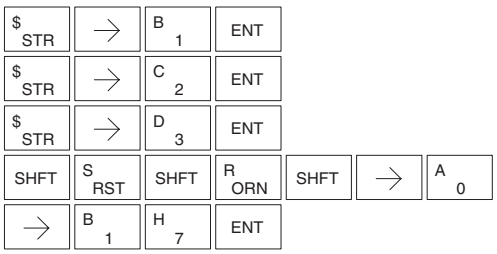
El tamaño máximo del bloque SR depende del número de relevadores disponibles de control. El tamaño mínimo del bloque es 8 relevadores de control.

Tipo de operando de datos	Rango del DL06	
. . . . . A/B	aaa	bbb
Control Relay C	0–1777	0–1777

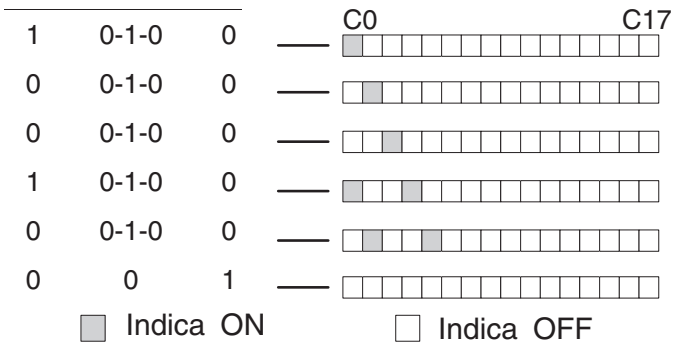
DirectSOFT



Programador D2-HPP



Entradas en barridos sucesivos Bits de shift register



Operaciones de carga y copia del acumulador y stack

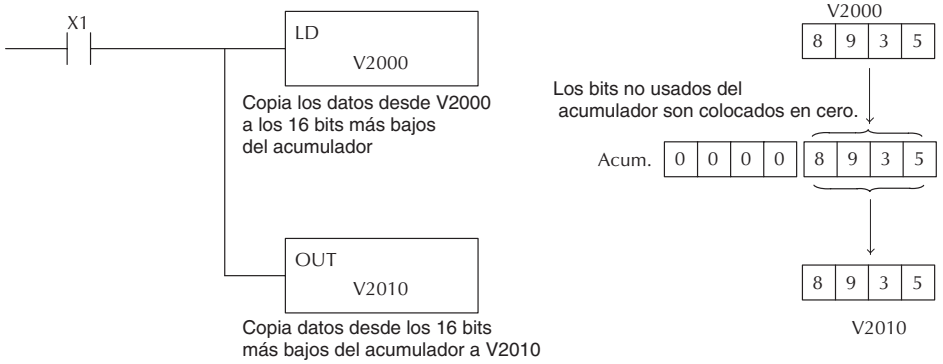
Usando el acumulador

El acumulador en la unidad de procesamiento central (CPU) del PLC DL06 es una memoria intermediaria (RAM) de 32 bits que se usa como una localización de almacenamiento temporaria para datos que se copian o son manipulados de alguna manera. Por ejemplo, usted tiene que usar el acumulador para realizar operaciones aritméticas tales como sumar, restar, multiplicar, etc. Ya que hay 32 bits, usted puede operar con un número de 8 dígitos BCD o datos ASCII sobre cualquier tipo de datos . *El acumulador es vuelto a 0 al fin de cada barrido de la CPU, es decir, el acumulador no retiene información*.

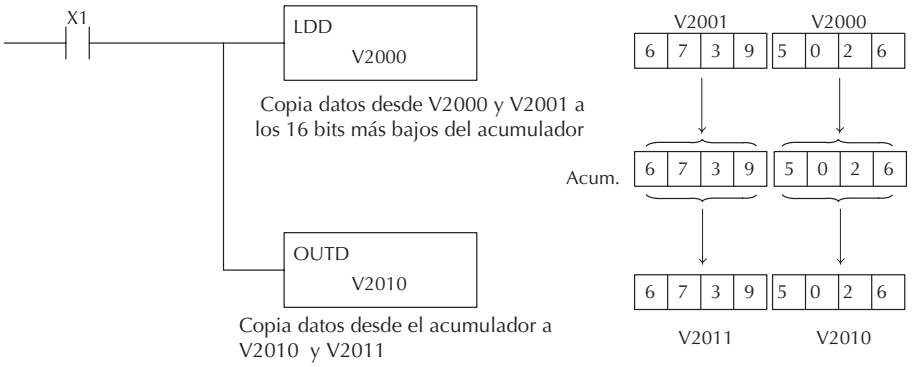
5

Copiando datos al acumulador

Las instrucciones LD y OUT y sus variaciones se usan para copiar datos de una dirección de memoria V al acumulador o para copiar los datos del acumulador a una memoria V. El ejemplo siguiente copia los datos de la memoria V2000 a la memoria V2010.



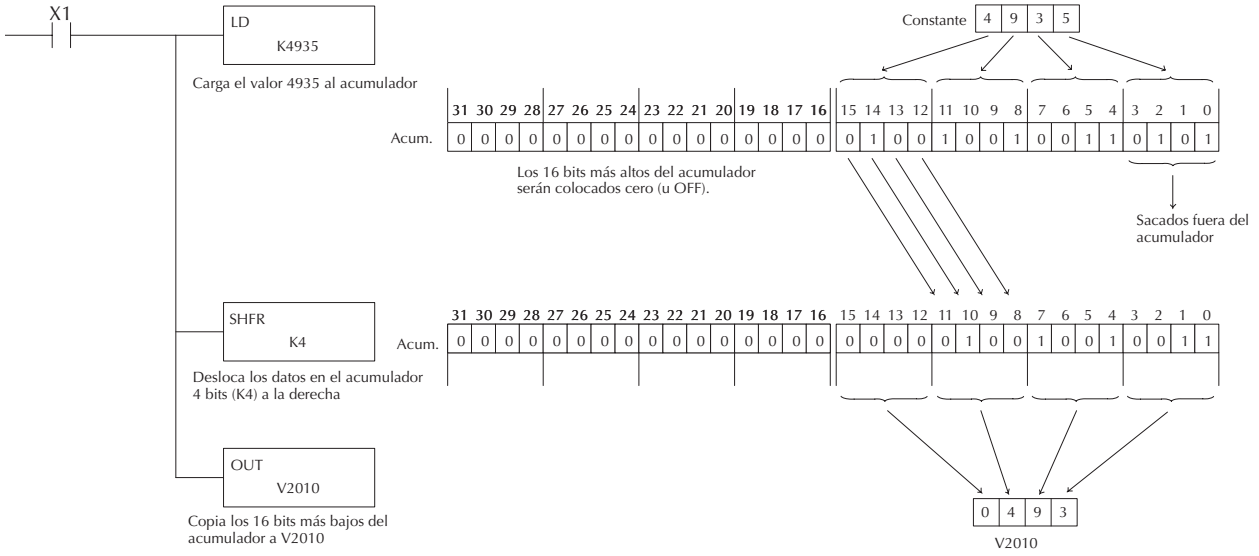
Ya que el acumulador es de 32 bits y las direcciones de memoria V son de 16 bits, las instrucciones LDD y OUTD (o las variaciones de las mismas) usan dos direcciones consecutivas de memoria V o una constantee de 8 dígitos BCD para copiar los datos al o desde una dirección de memoria V al acumulador. Por ejemplo si usted quiere copiar los datos de V2000 y V2001 para V2010 y V2011 la manera más eficiente de realizar esta función sería como sigue:



Cambiando los datos del acumulador

Las instrucciones que manipulan datos también usan el acumulador. El resultado de los datos manipulados se queda en el acumulador. Los datos que tenía el acumulador antes de hacer la operación correspondiente se pierden en el acumulador.

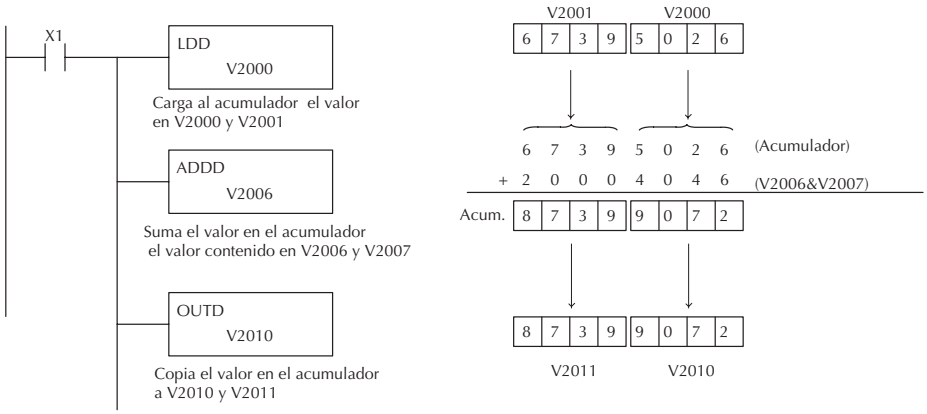
El ejemplo siguiente copia la constantee 4935 en el acumulador, disloca a la derecha los datos en 4 bits y copia el resultado a V2010.



Algunas instrucciones de manipulación de datos usan 32 bits. Usan dos direcciones consecutivas de memoria V o una constantee de 8 dígitos BCD para manipular los datos en el acumulador.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V2000 y V2001 en el acumulador usando la instrucción LDD.

El valor en el acumulador se suma al valor en V2006 y V2007 usando la instrucción ADDD. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.



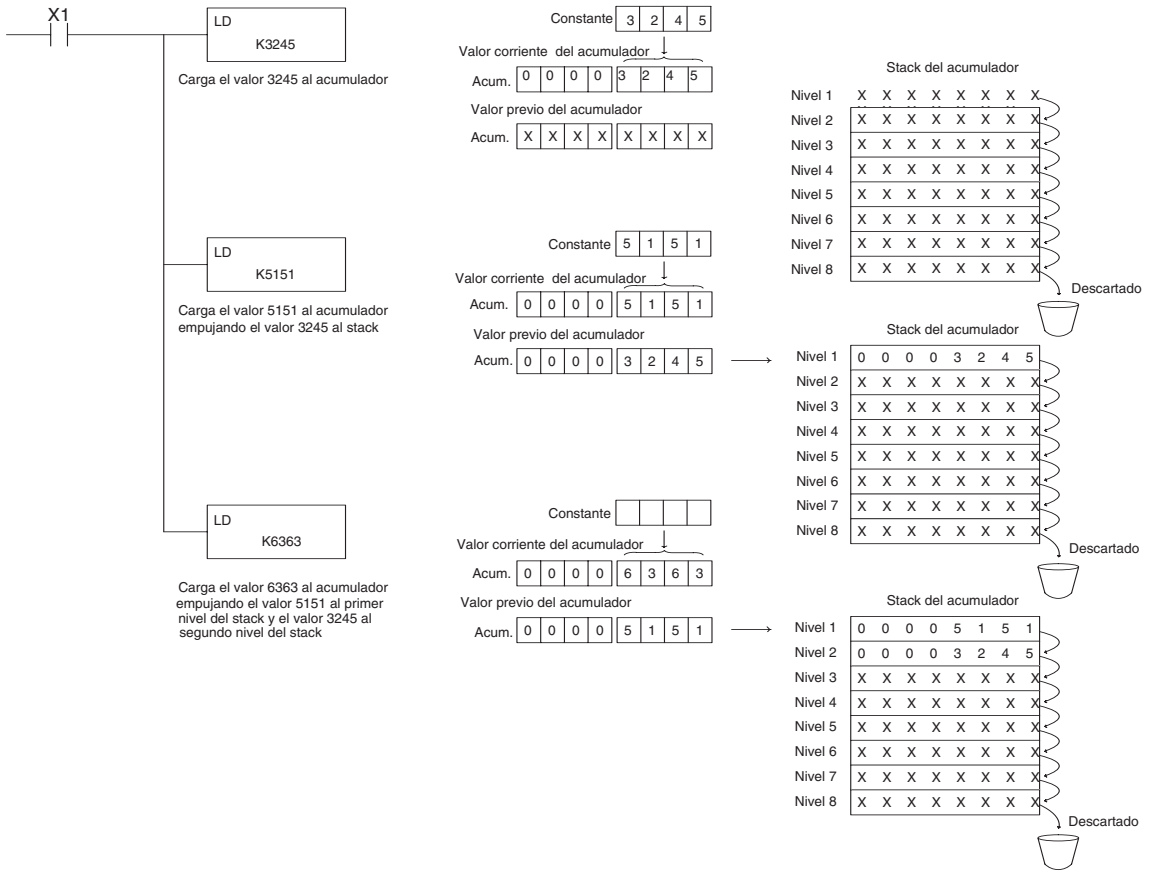
Usando el Stack del acumulador

El stack del acumulador (Una pila de memorias) es usado por instrucciones que requieren más de un parámetro para ejecutar una función o para una función definida por el usuario. El Stack del acumulador se usa cuando se ejecuta más de una instrucción LD sin el uso de una instrucción OUT. El contenido del stack vuelve a 0 al fin de cada barrido.

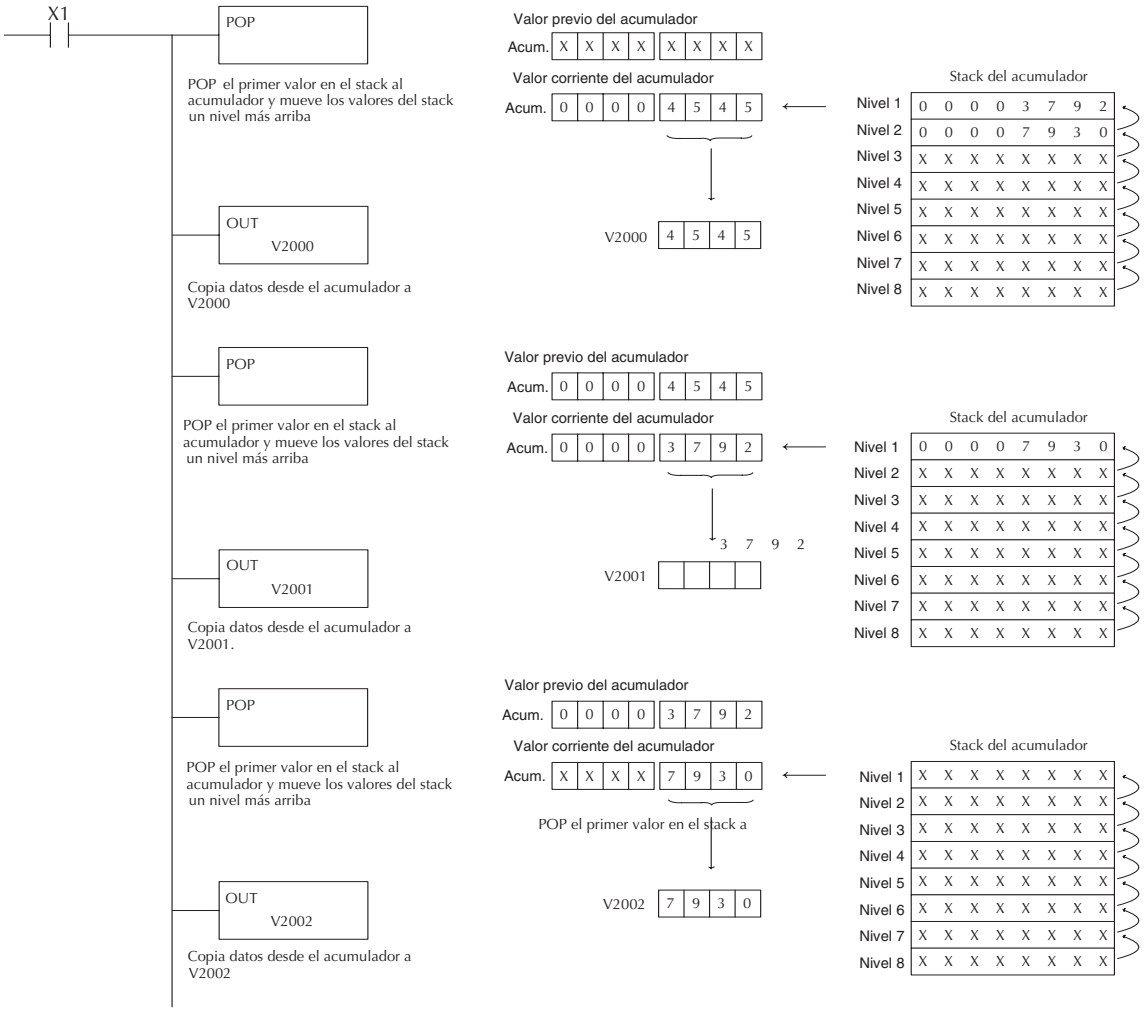
La primera instrucción LD en el barrido coloca un valor en el acumulador. Cada instrucción LD después, sin el uso de la instrucción OUT, coloca un valor en el acumulador y el valor que estaba en el acumulador se coloca en el Stack del acumulador.

La instrucción OUT anula la instrucción previa de LD y no coloca el valor que estaba en el acumulador en el Stack del acumulador cuando se ejecuta la próxima instrucción LD. Cada vez que un valor se coloca en el acumulador amontona los otros valores en el Stack y se empujan hacia abajo una dirección de memoria.

El acumulador tiene ocho niveles de profundidad (ocho registros de 32 bits). Si hay un valor en la octava localización cuando un valor nuevo se coloca en el Stack, el valor en la octava localización sale fuera del Stack y no se puede recuperar, es decir, se pierde



La instrucción POP rota los valores hacia arriba por el Stack al acumulador. Cuándo se ejecuta la instrucción POP el valor que estaba en el acumulador se limpia y el valor que estaba encima del Stack pasa al acumulador. Los valores en el Stack se dislocan una posición hacia arriba en el Stack .

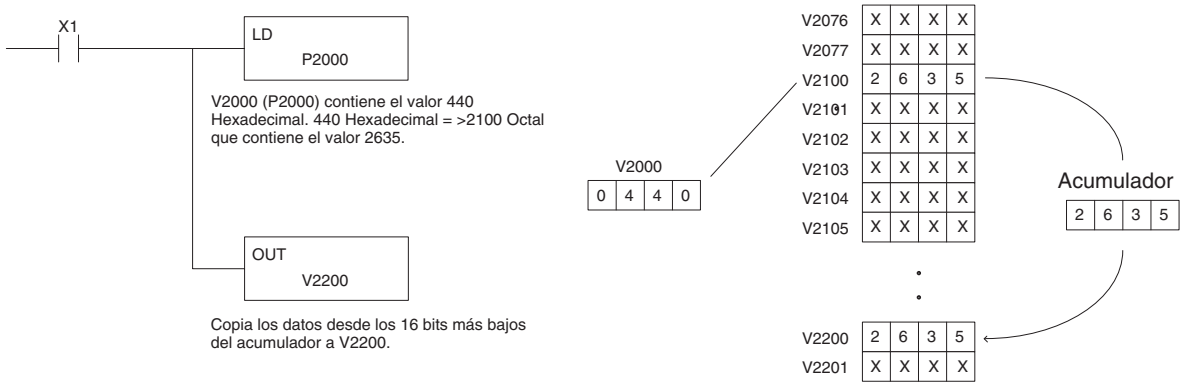


Usando punteros

Muchas de las instrucciones del PLC de la serie DL06 permitirán usar los punteros de la memoria V como un operando (comúnmente conocido como direccionamiento indirecto). Los punteros permiten que las instrucciones obtengan los datos de direcciones de memoria V indicadas por el valor del puntero.

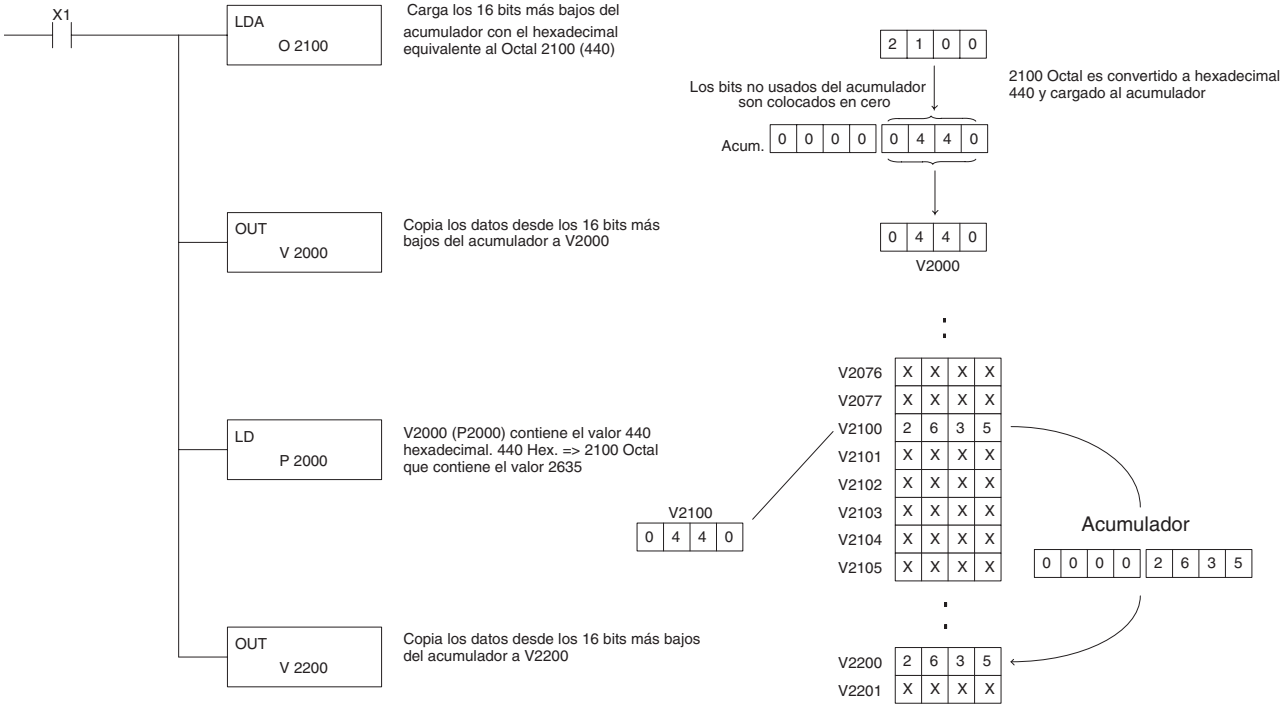
NOTA: La dirección de memoria DL06 V está en octal. Sin embargo, el puntero se refiere a una dirección de memoria V con valores hexadecimales. Use la instrucción LDA para transformar una dirección a la dirección de puntero. Esta instrucción realiza la conversión Octal a Hexadecimal automáticamente.

En el ejemplo siguiente usaremos un operando del puntero en una instrucción LD. La direción de memoria V2000 es usada como localización del puntero. V2000 contiene el valor 440 que la CPU ve como el equivalente hexadecimal de la memoria octal V2100. La CPU copiará los datos de V2100 que en este ejemplo contiene el valor [2635] en la palabra más baja del acumulador.



5

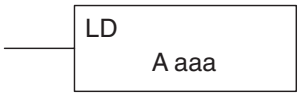
El ejemplo siguiente es idéntico al de arriba con una excepción. La instrucción LDA convierte automáticamente la dirección octal a hexadecimal.



La instrucción Load (LD)

DS5	Usado
HPP	Usado

La instrucción Load (LD) es una instrucción de 16 bits que carga o copia el valor (Aaaa), que es una dirección de memoria V o una constantee de 4 dígitos BCD/Hexadecimal, en los 16 bits más bajos del acumulador. Los 16 bits más altos del acumulador son forzados a 0.



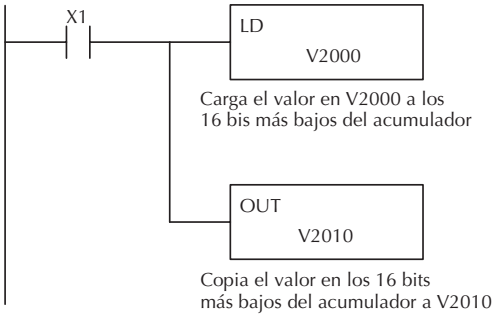
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–FFFF

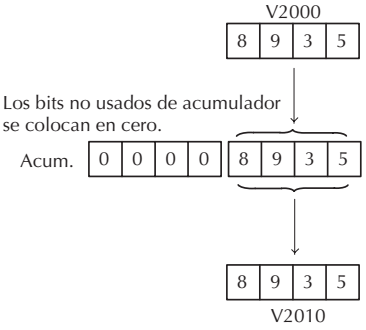
Indicadores	Descripción
SP53	Está ON cuando el puntero está fuera del rango disponible.
SP70	ON si el valor en el acumulador por cualquier instrucción es negativo.
SP76	ON cuando cualquier instrucción carga un valor 0 al acumulador.

NOTA: Dos instrucciones consecutivas LD colocarán el valor de la primera instrucción LD en el Stack del acumulador.

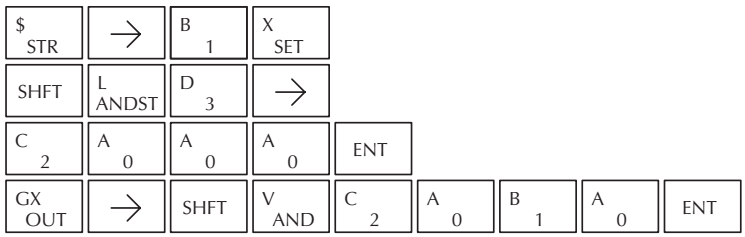
En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V2000 al acumulador y luego se copia a V2010.

DirectSOFT





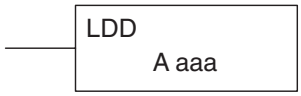
Programador D2-HPP



La instrucción Load Double (LDD)

DS5	Usado
HPP	Usado

La instrucción LDD es una instrucción de 32 bits que carga o copia el valor (Aaaa), que es: o dos direcciones consecutivas de memoria V o una constantee de 8 dígitos BCD/Hexadecimal, en el acumulador.



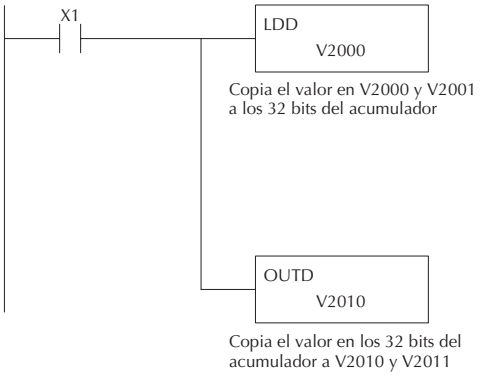
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–FFFF

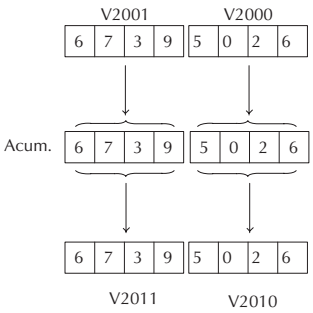
Indicadores	Descripción
SP53	Está ON cuando el puntero está fuera del rango disponible.
SP70	ON si el valor en el acumulador por cualquier instrucción es negativo.
SP76	ON cuando cualquier instrucción carga un valor 0 al acumulador.

NOTA: Dos instrucciones LDD consecutivas colocarán el valor de la primera instrucción LDD en el Stack del acumulador.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor de 32 bits en V2000 y V2001 en el acumulador y es copiado a V2010 y V2011.

DirectSOFT





Programador D2-HPP

$ STR	→	B 1	ENT	
SHFT	L ANDST	D 3	D 3	→
C 2	A 0	A 0	A 0	ENT
GX OUT	SHFT	D 3	→	
C 2	A 0	B 1	A 0	ENT

La instrucción Load Formatted (LDF)

DS5	Usado
HPP	Usado

La instrucción LDF carga o copia un conjunto de 1 a 32 bits consecutivos de direcciones discretas de memoria en el acumulador. La instrucción requiere una dirección (Aaaa) de inicio y el número de bits (Kbbb) a ser cargado. Los bits no usados del acumulador se colocan en 0.

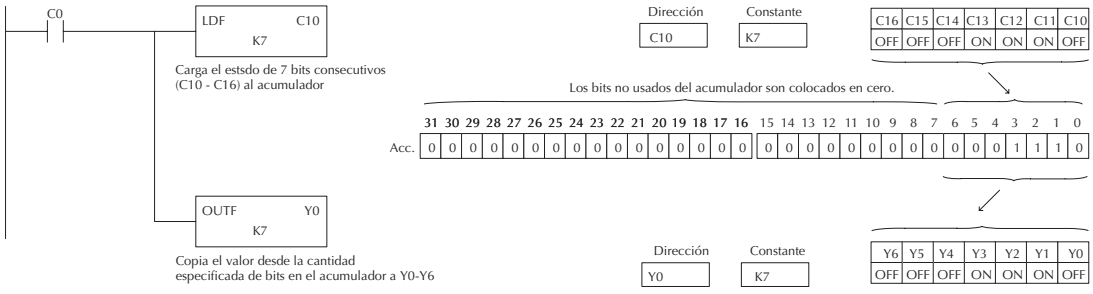
Tipo de operando de datos	Rango del DL06	
..... A	aaa	bbb
Entradas X	0–777	—
Salidas ..... Y	0–777	—
Relevadores de control C	0–1777	—
Bits de Etapas ..... S	0–1777	—
Bits de temporizadores ..... T	0–377	—
Bits de contadores ..... CT	0–177	—
Relevadores especiales ..... SP	0–777	—
Constante ..... K	—	1–32

Indicadores	Descripción
SP70	On anytime the value in the accumulator is negative.
SP76	On when any instrucción loads a value of zero into the accumulator.

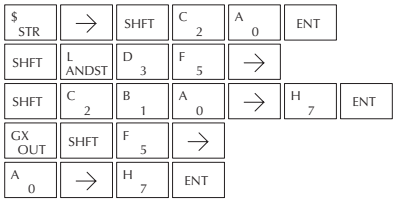
NOTA: Dos instrucciones consecutivas LDF colocarán el valor de la primera instrucción en el Stack del acumulador.

En el ejemplo siguiente, cuándo C0 está ON, el conjunto de bits de C10-C16 (7 bits) será copiado al acumulador usando la instrucción LDF. Los 7 bits más bajos del acumulador son

DirectSOFT



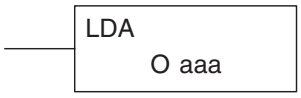
Programador D2-HPP



La instrucción Load Address (LDA)

DS5	Usado
HPP	Usado

La instrucción LDA es una instrucción de 16 bits. Convierte cualquier valor octal (o dirección) al valor del equivalente hexadecimal y lo carga (o copia) al acumulador. Esta instrucción es útil cuando se requiere un parámetro de dirección ya que todas las direcciones para el sistema DL06 están en octal.



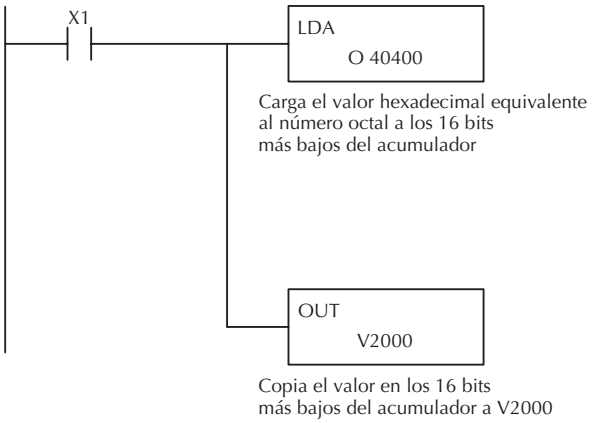
Tipo de operando de datos	Rango del DL06
	aaa
Octal Address . . . . . O	Vea el mapa de memoria

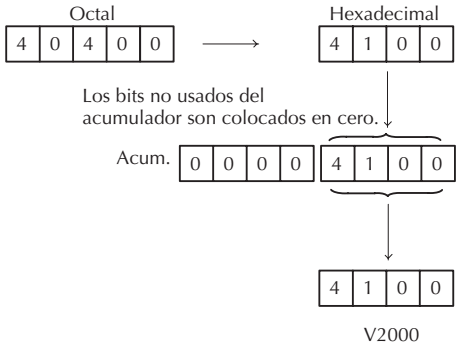
Indicadores	Descripción
SP70	On anytime the value in the accumulator is negative.
SP76	On when any instrucción loads a value of zero into the accumulator.

NOTA: Dos instrucciones consecutivas LDA colocan el contenido de la primera instrucción en el stack del acumulador.

En el ejemplo siguiente cuando X1 está ON, el número octal 40400 será convertido a un 4100 hexadecimal y cargado en el acumulador usando la instrucción LDA. El valor en los 16 bits más bajos del acumulador es copiado a V2000 usando la instrucción OUT.

DirectSOFT





Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	A 0	→				
E 4	A 0	E 4	A 0	A 0	ENT			
GX OUT	→	SHFT	V AND	C 2	A 0	A 0	A 0	ENT

La instrucción Load Accumulator Indexed (LDX)

DS5	Usado
HPP	Usado

Esta instrucción de 16 bits especifica una dirección de la fuente (la memoria V) que será dislocada por el valor en la primera dirección del Stack . Esta instrucción LDX interpreta el valor en la primera dirección del Stack como hexadecimal. El valor en la dirección ya dislocada (la dirección de la fuente + el desvío) es cargado en los 16 bits más bajos del acumulador. Los 16 bits más altos del acumulador son forzados a 0.

LDX
A aaa

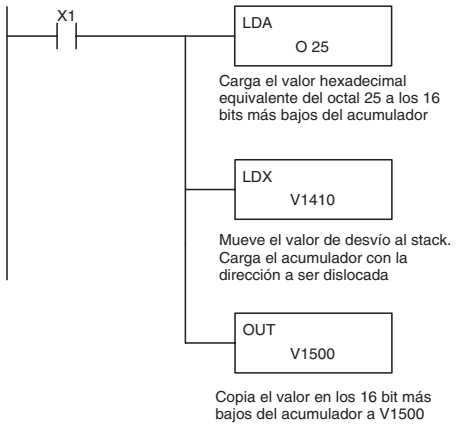
Sugerencia: — La instrucción LDA se puede usar para convertir una dirección de octal a una dirección hexadecimal y cargar el valor en el acumulador

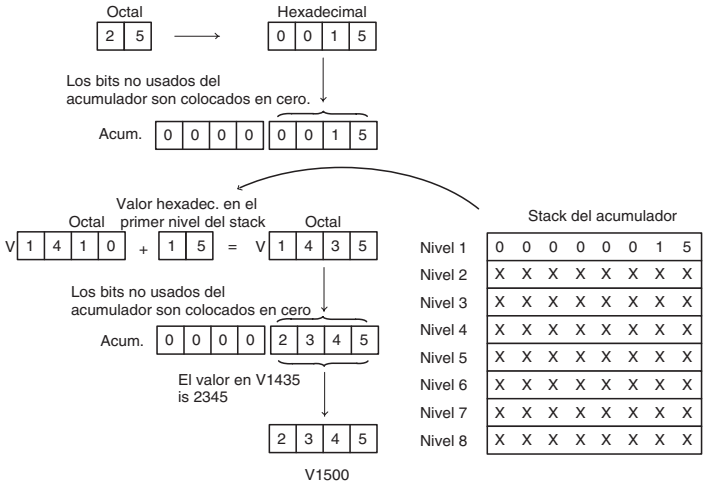
Tipo de operando de datos	Rango del DL06	
A	aaa	aaa
Memoria V	Vea el mapa de memoria	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria	Vea el mapa de memoria

Indicadores	Descripción
SP53	Está ON cuando el puntero está fuera del rango disponible.
SP70	ON si el valor en el acumulador por cualquier instrucción es negativo.
SP76	ON cuando cualquier instrucción carga un valor 0 al acumulador.

NOTA: Dos instrucciones consecutivas de la instrucción LDX colocarán el valor de la primera instrucción en el Stack del acumulador.

En el ejemplo siguiente cuando X1 está ON, el equivalente hexadecimal del octal 25 será cargado al acumulador (este valor se colocará en el Stack cuando se ejecuta la instrucción LDX). La dirección de memoria V1410 se suma al valor en el primer el nivel del Stack y el valor de esta dirección es cargado en los 16 bits más bajos del acumulador usando la instrucción LDX. El valor en los 16 bits más bajos del acumulador es copiado a V1500 usando la instrucción OUT.





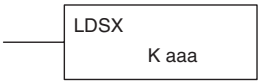
Programador D2-HPP

$ STR	→	B 1	ENT								
SHFT	L ANDST	D 3	A 0	→	C 2	F 5	ENT				
SHFT	L ANDST	D 3	X SET	→	B 1	E 4	B 1	A 0	ENT		

La instrucción Load Accumulator Indexed from Data Constantes (LDSX)

DS5	Usado
HPP	Usado

La instrucción LDSX es una instrucción de 16 bits. La instrucción especifica un Area de Data Label (DLBL) (de Etiqueta de Datos) donde se almacenan constantees numéricas o ASCII. Este valor se carga en los 16 bits más bajos del acumulador.



La instrucción LDSX usa el valor en el primer nivel del Stack del acumulador como un "desvío" para determinar cuál constantee numérica o ASCII dentro del Area DLBL se carga en el acumulador. La instrucción de LDSX interpreta el valor en el primer nivel del Stack del acumulador como un valor hexadecimal.

Sugerencia: — La instrucción LDA se puede usar para convertir octal a hexadecimal y cargar el valor en el acumulador.

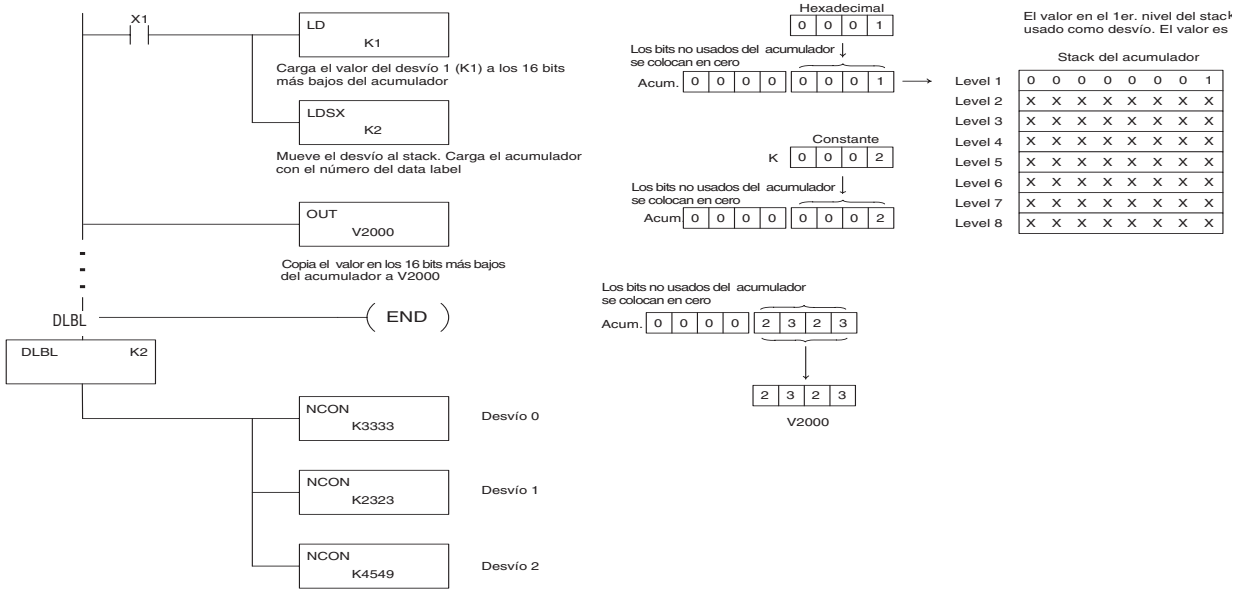
5

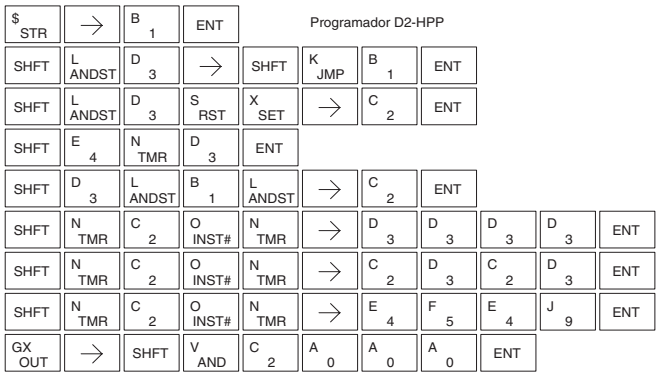
Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . K	1-FFFF

Indicadores	Descripción
SP53	Está ON cuando el puntero está fuera del rango disponible.
SP70	ON si el valor en el acumulador por cualquier instrucción es negativo.
SP76	ON cuando cualquier instrucción carga un valor 0 al acumulador.

NOTA: Dos instrucciones consecutivas LDSX colocan el contenido de la primera instrucción en el stack del acumulador.

En el ejemplo siguiente cuando X1 está ON, se carga un desvío de 1 en el acumulador. Este valor se colocará en el primer nivel del Stack del acumulador cuando se ejecuta la instrucción LDSX. La instrucción LDSX especifica el área DLBL K2 donde se encuentran las constantees numéricas en el programa y carga el valor constantee, indicado por el valor de desvío en el Stack, en los 16 bits más bajos del acumulador.





La instrucción Load Real Number (LDR)

DS5	Usado
HPP	N/A

La instrucción LDR carga un número real contenido en dos direcciones consecutivas de la memoria V o en una constantee de 8 dígitos en el acumulador.

```
LDR
    A aaa
```

Tipo de operando de datos	Rango del DL06
. . . . . A	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero . . . . . P	Vea el mapa de memoria
Real Constante . . . . . R	-3.402823E^{+38} to + -3.402823E^{+38}

Indicadores	Descripción
SP70	On anytime the value in the accumulator is negative.
SP76	On when any instrucción loads a value of zero into the accumulator.

*Direct*SOFT le permite que entre los números reales directamente, usando una "R" como prefijo que indica un ingreso de número real. Usted puede entrar una constantee tal como Pi(π), mostrado en el ejemplo a la derecha.

```
LDR
    R3,14159
```

Para entrar números negativos, use un signo menos (-) después de la "R".

Para números muy grandes o números muy pequeños, se puede usar la notación exponencial. El número a la derecha es 5,3 millones. La instrucción OUTD lo copia a V1400 y V1401.

```
LDR
    R5,3E6
OUTD
    V1400
```

¡Estos números reales están en el formato de punto flotante IEEE de 32 bits, de modo que ocupan dos direcciones de memoria V, a pesar de que el número puede ser muy grande o pequeño! Si usted ve un número real almacenado en hexadecimal, binario o aún BCD, el número mostrado será muy difícil de descifrar. Así como todos los otros tipos de números, usted debe seguir las direcciones del número real en la memoria, de modo que puedan ser leídos en otra parte con las instrucciones apropiadas

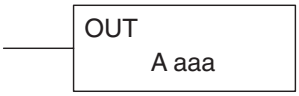
El ejemplo previo encima almacenó un número real en V1400 y V1401. Suponga que ahora queremos recuperar ese número. Use solamente LDR con el tipo de datos V, como se muestra a la derecha. Luego podríamos realizar las operaciones matemáticas reales o convertirlo a un número binario.

```
LDR
    V1400
```

La instrucción Out de bloque (OUT)

DS5	Usado
HPP	Usado

Es una instrucción de 16 bits que copia el valor en los 16 bits más bajos contenido en el acumulador a una localización especificada de memoria V (Aaaa).



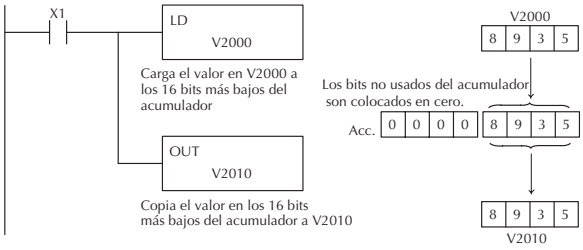
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP53	ON si la CPU no puede resolver la lógica

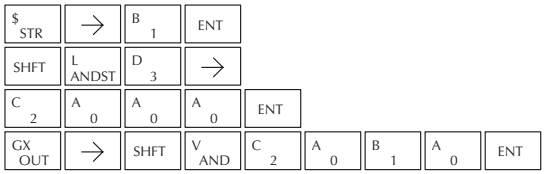
5

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 se carga en los 16 bits más bajos del acumulador usando la instrucción LD. Luego se copia el valor en los 16 bits más bajos del acumulador a V2010 con la instrucción OUT de bloque.

DirectSOFT



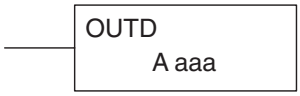
Programador D2-HPP



La instrucción Out Double (OUTD)

DS5	Usado
HPP	Usado

La instrucción OUT Doble es una instrucción de 32 bits que copia el valor en el acumulador a dos direcciones consecutivas de la memoria V en una localización (Aaaa) especificada.

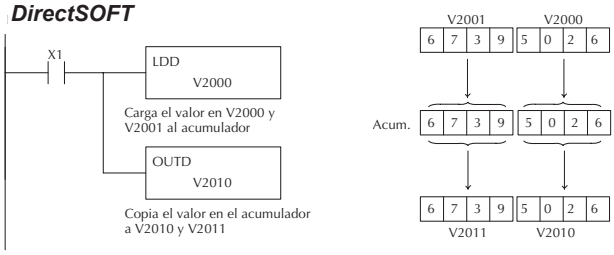


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

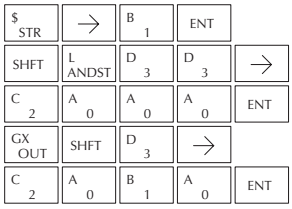
Indicadores	Descripción
SP53	On if CPU cannot solve the logic.

En el ejemplo siguiente, cuándo X1 está ON, el valor de 32 bits en V2000 y V2001 se carga en el acumulador usando la instrucción LDD. El valor en el acumulador es colocado en V2010 y V2011 usando la instrucción OUTD.

DirectSOFT



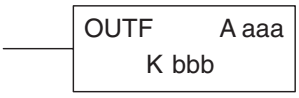
Programador D2-HPP



La instrucción Out Formatted (OUTF)

DS5	Usado
HPP	Usado

La instrucción OUTF carga 1-32 bits del acumulador a las direcciones discretas especificadas de memoria V. La instrucción requiere una dirección (Aaaa) de inicio y el número de bits (Kbbb) a ser transportados. Los bits no usados son colocados en 0.

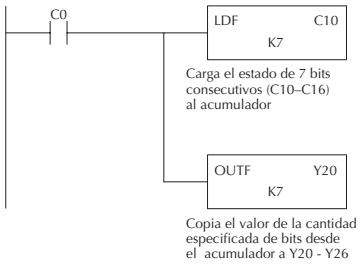


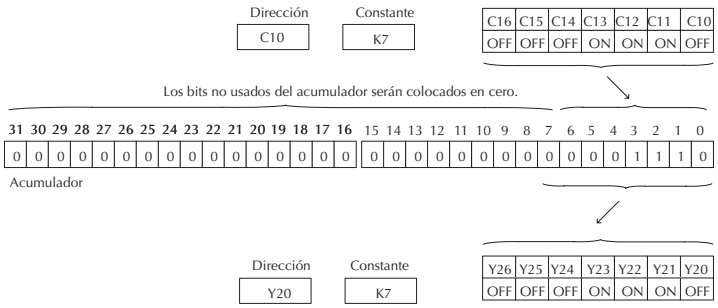
Tipo de operando de datos	Rango del DL06	
..... A	aaa	bbb
Entradas X	0–777	—
Salidas ..... Y	0–777	—
Relevadores de control C	0–1777	—
Constantee ..... K	—	1–32

5

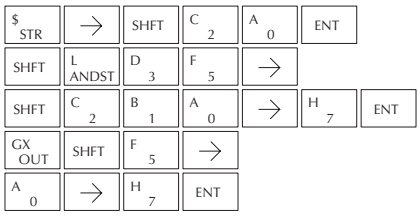
En el siguiente ejemplo, cuando C0 está ON, serán cargados los bits C10 a C16 (7 bits) al acumulador usando la instrucción LDF. Los 7 bits más bajos del acumulador son copiados a Y0 a Y6 usando la instrucción OUTF.

DirectSOFT





Programador D2-HPP



La instrucción Pop (POP)

DS5	Usado
HPP	Usado

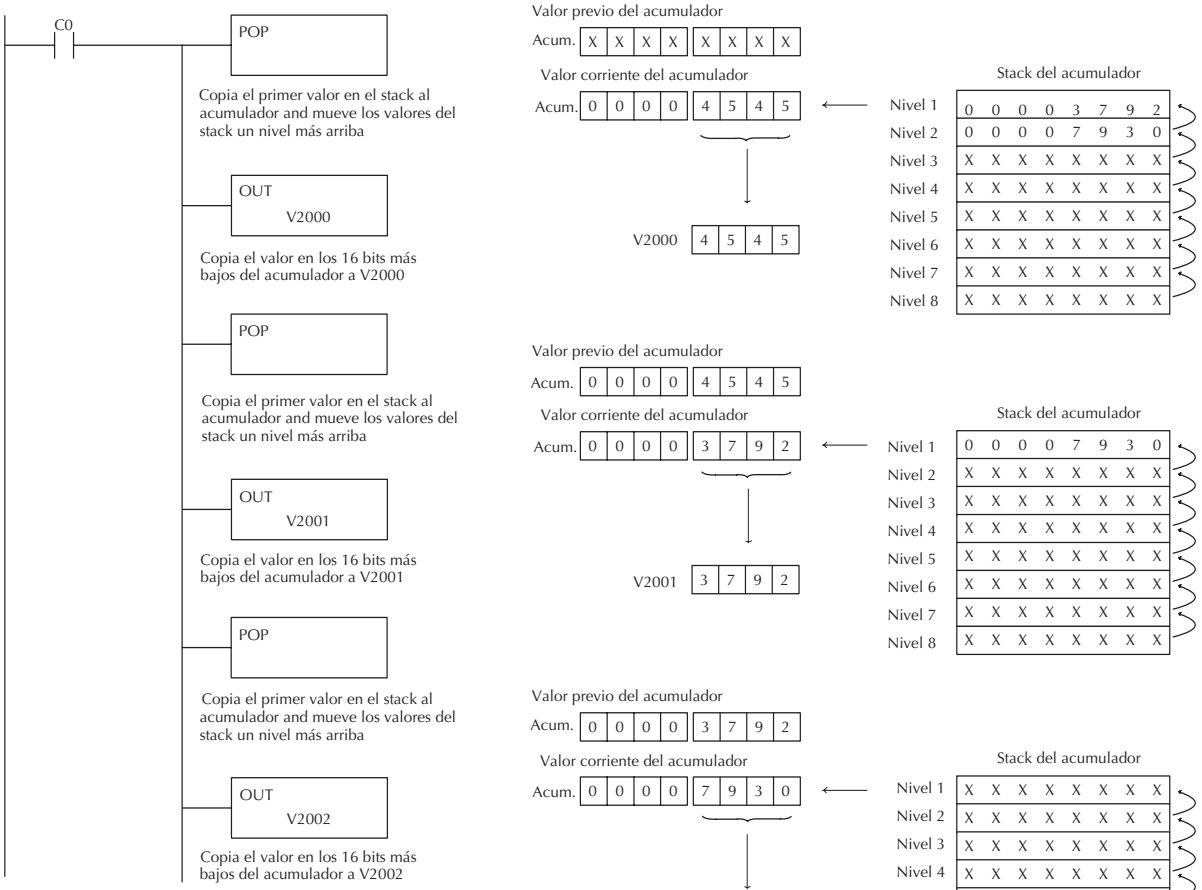
La instrucción POP mueve el valor del primer nivel del Stack del acumulador (32 bit) al acumulador y mueve cada valor en el Stack un nivel más arriba de lo que estaba.

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea cero.

La instrucción Pop continuada

En el ejemplo siguiente, cuándo C0 está ON, el valor 4545 que estaba encima del Stack se mueve al acumulador usando la instrucción POP. El valor es copiado a V2000 usando la instrucción OUT. El próximo POP mueve el valor 3792 al acumulador y copia el valor a V2001. El último POP mueve el valor 7930 al acumulador y copia el valor a V2002 con la instrucción OUT. Note que si el valor en el Stack usa más de 16 bits (4 dígitos) debe usarse la instrucción OUTD y deben ser asignadas 2 direcciones de memoria V para cada OUTD.

DirectSOFT



Programador D2-HPP

$ STR	→	SHFT	C 2	A 0	ENT			
SHFT	P CV	SHFT	O INST#	P CV	ENT			
GX OUT	→	SHFT	V AND	C 2	A 0	A 0	A 0	ENT
SHFT	P CV	SHFT	O INST#	P CV	ENT			
GX OUT	→	SHFT	V AND	C 2	A 0	A 0	B 1	ENT
SHFT	P CV	SHFT	O INST#	P CV	ENT			
GX OUT	→	SHFT	V AND	C 2	A 0	A 0	C 2	ENT

La instrucción Out Indexed (OUTX)

DS5	Usado
HPP	Usado

La instrucción OUTX es una instrucción de 16 bits. Copia un valor de 16 bits o de 4 dígitos desde el primer nivel del Stack del acumulador hasta una dirección cambiada por un número de desvío que es el valor en el acumulador (la memoria V + el desvío). Esta instrucción interpreta el valor del desvío como un número hexadecimal. Los 16 bits más altos del acumulador son forzados a 0.

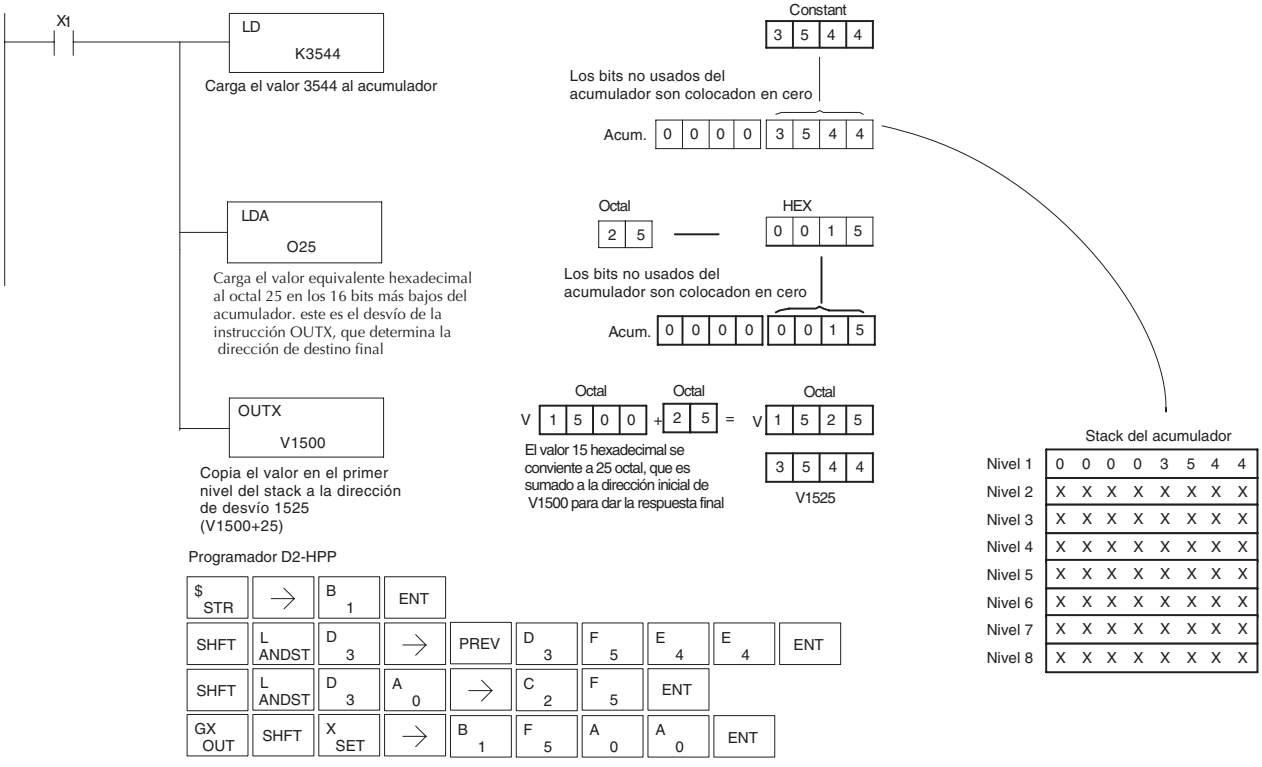
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP53	ON si la CPU no puede resolver la lógica

En el ejemplo siguiente, cuándo X1 está ON, la constantee 3544 es cargada al acumulador. Este es el valor que será copiado a la memoria V de destino con desvío (V1525). El valor 3544 será colocado en el Stack cuando se ejecuta la instrucción LDA. Recuerde, dos instrucciones consecutivas LDA colocan el valor de la primera instrucción LD en el Stack. La instrucción LDA convierte el valor 25 octal a 15 hexadecimal y coloca el valor en el acumulador.

La instrucción OUTX copia el valor 3544 que está en el primer nivel del Stack del acumulador a V1525.

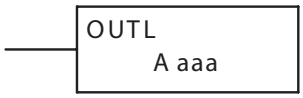
DirectSOFT



La instrucción Out Least (OUTL)

DS5	Usado
HPP	Usado

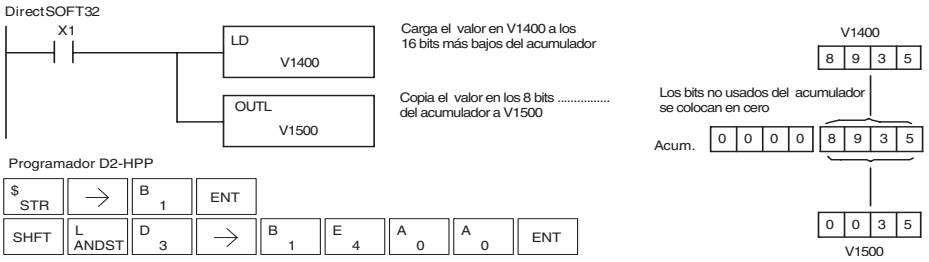
La instrucción OUTL copia el valor en los 8 bits más bajos del acumulador a los 8 bits más bajos de la memoria especificada (en otras palabras, copia el byte más bajo de la palabra más baja del acumulador).



Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 se carga en los 16 bits más bajos del acumulador usando la instrucción LD. El valor en los 8 bits más bajos del acumulador es copiado a V1500 usando la instrucción OUTL.

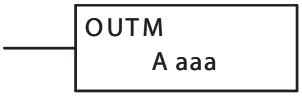
DirectSOFT



La instrucción Out Most (OUTM)

DS5	Usado
HPP	Usado

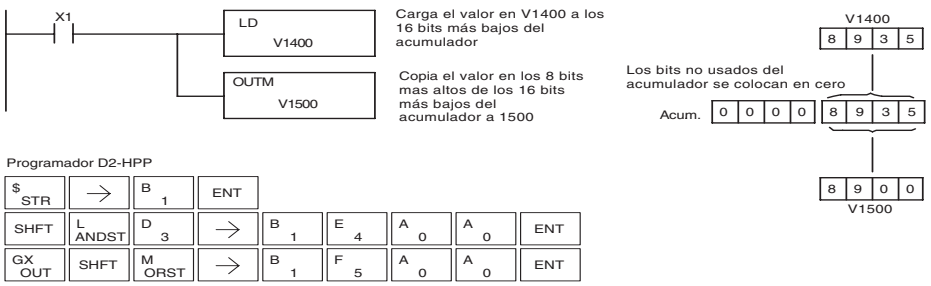
La instrucción OUTM copia el valor en los 8 bits más altos de la palabra más baja del acumulador a los 8 bits más altos de la memoria especificada (en otras palabras, copia el byte más alto de la palabra más baja del acumulador).



Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 se carga en los 16 bits más bajos del acumulador usando la instrucción LD. El valor en los 8 bits más altos de los 16 bits más bajos del acumulador es copiado a V1500 usando la instrucción OUTM.

DirectSOFT



Las instrucciones lógicas con el acumulador

La instrucción And de bloque (AND)

DS5	Usado
HPP	Usado

La instrucción AND es una instrucción de 16 bits lógica que hace la función AND del valor en los 16 bits más bajos del acumulador con una localización especificada de memoria V (Aaaa). El resultado se va al acumulador. Una indicación discreta del estado con un relevador especial SP indica si el resultado es cero.

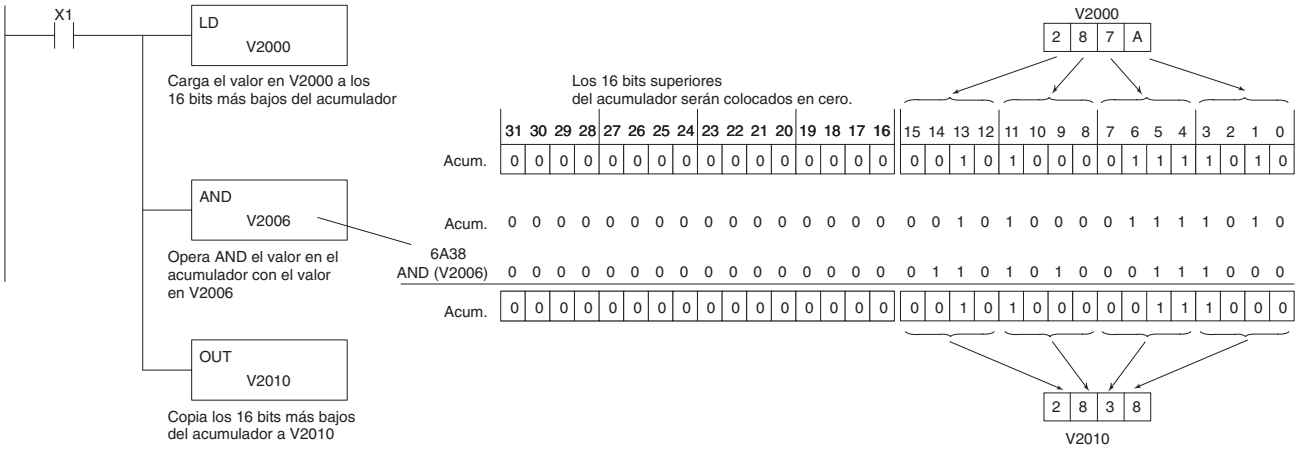
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	Está ON si el resultado de la instrucción en el acumulador es 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 se carga en el acumulador usando la instrucción LD. El valor en el acumulador es operado AND con el valor en V2006 usando la instrucción AND. El valor en los 16 bits más bajos del acumulador es copiado a V2010 usando la instrucción OUT.

DirectSOFT



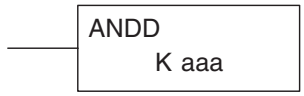
Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	C 2	A 0	A 0	A 0	ENT
V AND	→	SHFT	V AND	C 2	A 0	A 0	G 6	ENT
GX OUT	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT

La instrucción And Double (ANDD)

DS5	Usado
HPP	Usado

ANDD es una instrucción de 32 bits que hace la función lógica AND del valor en el acumulador con dos direcciones consecutivas de memoria V o un valor (Aaaa) constantee de 8 dígitos (máximo). El resultado se va al acumulador. Las indicaciones de estado discretas con SP indican si el resultado de la instrucción ANDD es cero o un número negativo (el bit más significativo está ON).



Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–FFFFFFFF

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON si el resultado en el acumulador es negativo

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga en el acumulador usando la instrucción LDD. El valor en el acumulador es operado como AND con 36476A38 usando la instrucción ANDD. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

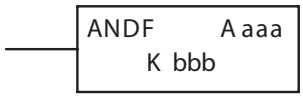
Programador D2-HPP

$ STR	→	B 1	ENT																				
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT														
V AND	SHFT	D 3	→	SHFT	K JMP	D 3	G 6	E 4	H 7	G 6	SHFT	A 0	SHFT	D 3	I 8	ENT							
GX OUT	SHFT	D 3	→	C 2	A 0	B 1	A 0	ENT															

La instrucción And Formatted (ANDF)

DS5	Usado
HPP	Usado

La instrucción ANDF hace la función lógica AND entre el valor binario en el acumulador y un rango especificado de bits en la memoria (1-32) . La instrucción requiere una localización (Aaaa) de inicio y el número de bits (Kbbb) a ser operados AND. Las indicaciones de estado discretas con SP indican si el resultado es cero o un número negativo (el bit más significativo = 1).



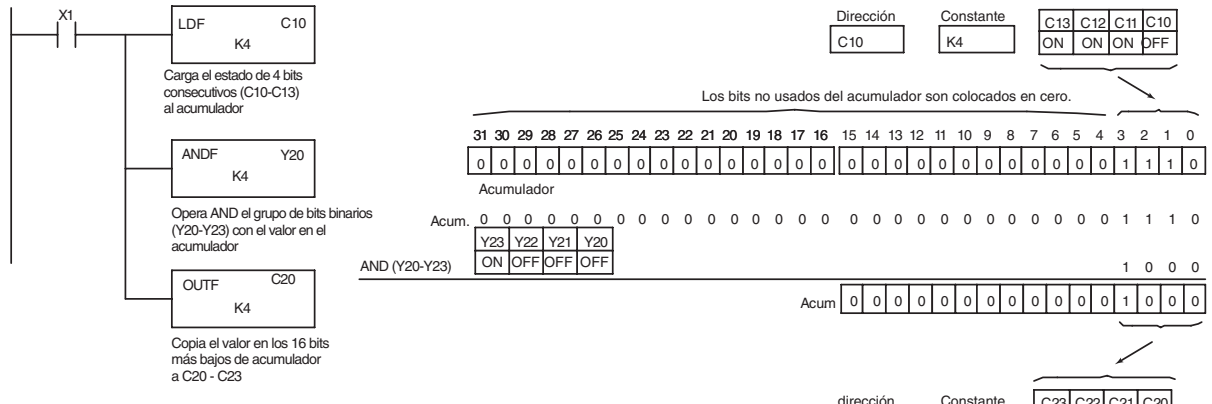
Tipo de operando de datos	Rango del DL06	
..... B	aaa	bbb
Entradas X	0-777	-
Salidas ..... Y	0-777	-
Relevadores de control C	0-1777	-
Bits de etapas ..... S	0-1777	-
Bits de temporizadores ..... T	0-377	-
Bits de contadores ..... CT	177	-
Relevadores especiales ..... SP	0-777	-
Constante ..... K	-	1-32

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON si el resultado en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON la instrucción LDF copia C10-C13 (4 bits binarios) en el acumulador. El contenido del acumulador es operado AND lógico con el modelo de bits Y20-Y23 usando la instrucción ANDF. La instrucción OUTF copia los 4 bits más bajos del acumulador a C20-C23.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT											
SHFT	L ANDST	D 3	F 5	→	NEXT	NEXT	NEXT	NEXT	B 1	A 0	→	E 4	ENT	
V AND	SHFT	F 5	→	NEXT	C 2	A 0	→	E 4	ENT					
GX OUT	SHFT	F 5	→	PREV	PREV	C 2	A 0	→	E 4	ENT				

La instrucción And with Stack (ANDS)

DS5	Usado
HPP	Usado

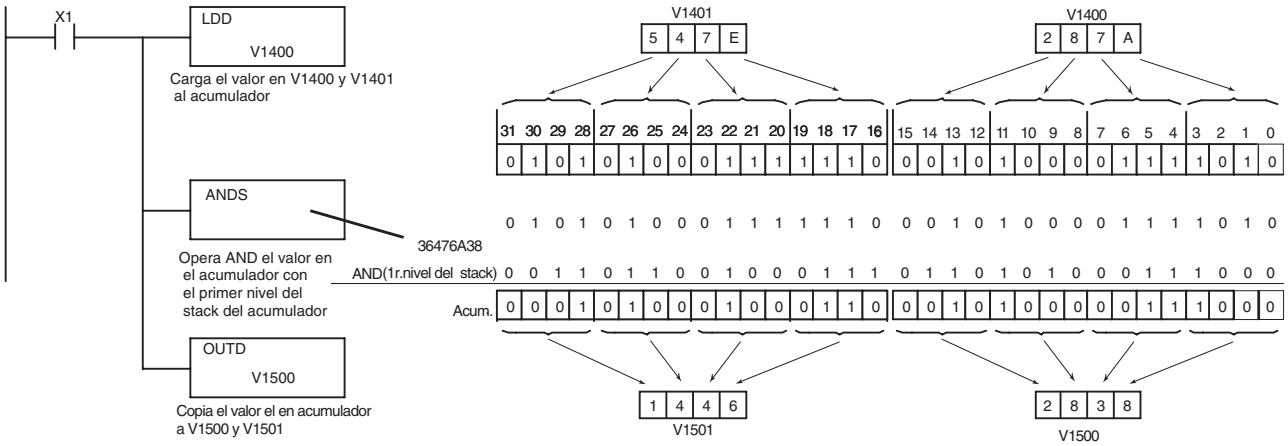
La instrucción ANDS es una instrucción de 32 bits que hace la función lógica AND entre el valor en el acumulador con el valor del primer nivel del Stack del acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador se remueve del Stack y todos los valores son movidos para arriba un nivel. Indicaciones de estado discretas SP indican si el resultado del ANDS es cero o un número negativo (el bit más significativo está ON).

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON si el resultado en el acumulador es negativo

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente cuando X1 está ON, el valor binario en el acumulador hace la función AND con el valor binario en el primer nivel del Stack del acumulador. El resultado se va al acumulador. El valor de 32 bits luego es copiado a V1500 y V1501.

DirectSOFT



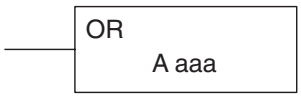
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
V AND	SHFT	S RST	ENT						
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT	

La instrucción de bloque Or (OR)

DS5	Usado
HPP	Usado

La instrucción OR es una instrucción de 16 bits que hace la función lógica OR entre el valor en los 16 bits más bajos del acumulador con una localización especificada de memoria V (Aaaa). El resultado se va al acumulador. La indicación de estado discreta SP indica si el resultado de la función OR es cero.



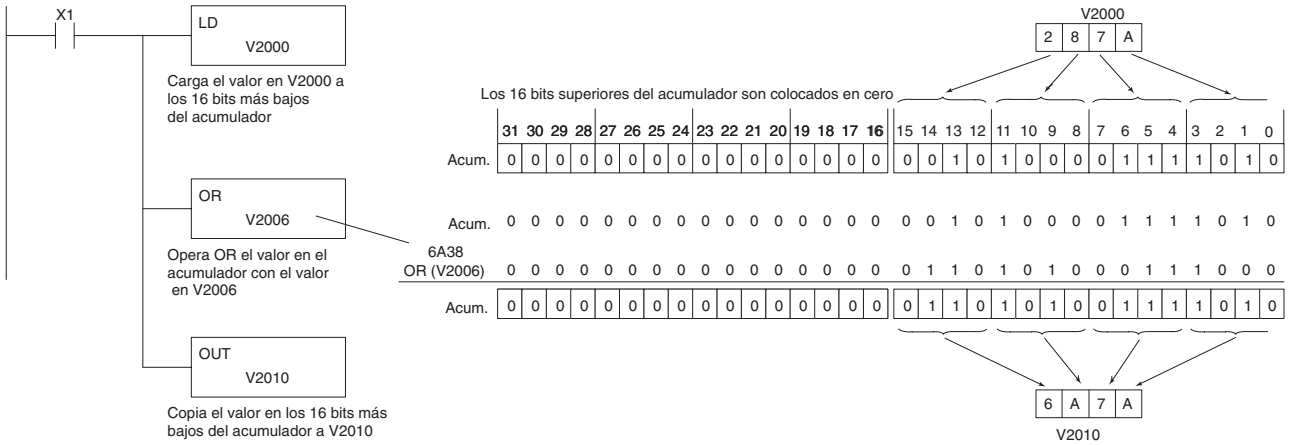
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V. V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 se carga en el acumulador usando la instrucción LD. El valor en el acumulador es operado con V2006 usando la instrucción OR. El valor en los 16 bits más bajos del acumulador es copiado a V2010 usando la instrucción OUT.

DirectSOFT



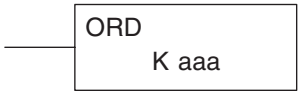
Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	C 2	A 0	A 0	A 0	ENT
Q OR	→	SHFT	V AND	C 2	A 0	A 0	G 6	ENT
GX OUT	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT

La instrucción Or Double (ORD)

DS5	Usado
HPP	Usado

ORD es una instrucción de 32 bits que hace la función OR entre el valor en el acumulador con el valor (Aaaa), que es dos direcciones consecutivas de memoria V o una constantee de 8 dígitos (max.). El resultado se va al acumulador. Indicaciones de estado discretas SP indican si el resultado de la operación ORD es cero o un número negativo (el bit más significativo está ON).



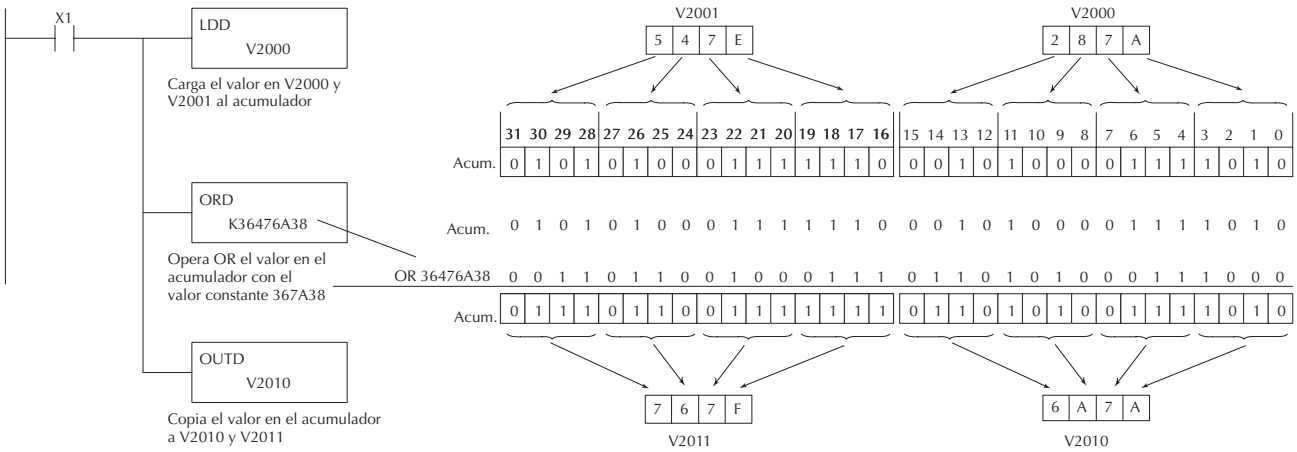
Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–FFFFFFFF

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON si el resultado en el acumulador es negativo.

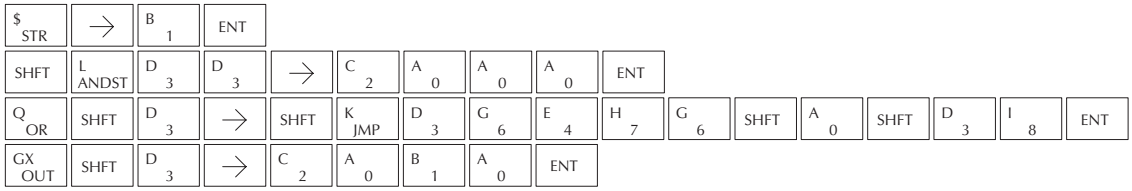
NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga en el acumulador usando la instrucción LDD. El valor en el acumulador es operado OR con 36476A38 usando la instrucción ORD. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

DirectSOFT



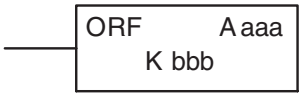
Programador D2-HPP



La instrucción Or Formatted (ORF)

DS5	Usado
HPP	Usado

La instrucción ORF hace la función OR lógica entre el valor binario en el acumulador y un rango especificado de bits (1-32). La instrucción requiere una localización (Aaaa) de inicio y el número de bits (Kbbb) a ser operados OR. Indicaciones de estado discretas SP indican si el resultado es cero o negativo (el bit más significativo = 1).



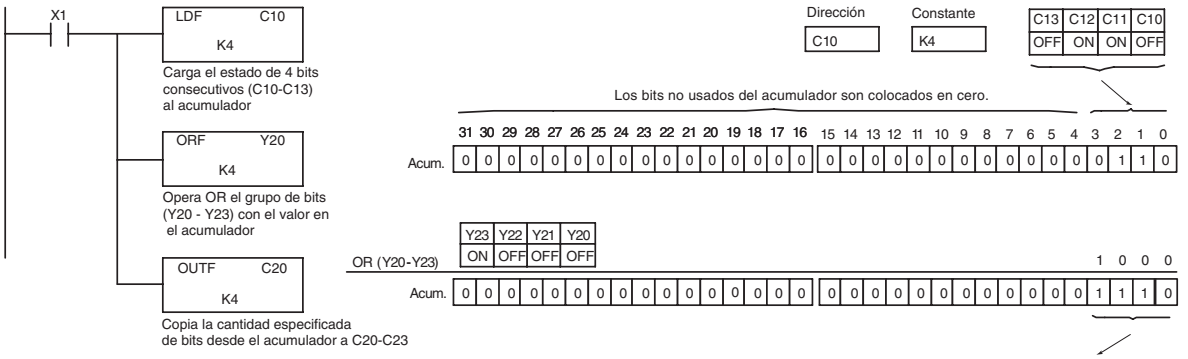
Tipo de operando de datos	Rango del DL06	
A/B	aaa	bbb
Entradas X	0-777	- -
Salidas . . . . . Y	0-777	- -
Relevadores de control C	0-1777	- -
Bits de etapas . . . . . S	0-1777	- -
Bits de temporizadores . . . . . T	0-377	- -
Bits de contadores . . . . . CT	0-177	- -
Relevadores especiales . . . . . SP	0-777	- -
Constante . . . . . K	-	1-32

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON la instrucción ORF carga C10-C13 (4 bits) en el acumulador. La instrucción opera OR el contenido del acumulador con los bits Y20-Y23. La instrucción OUTF copia los 4 bits más bajos del acumulador a C20-C23.

DirectSOFT



Handheld Programmer Keystrokes

$ STR	→	B 1	ENT											
SHFT	L ANDST	D 3	F 5	→	NEXT	NEXT	NEXT	NEXT	B 1	A 0	→	E 4	ENT	
Q OR	SHFT	F 5	→	NEXT	C 2	A 0	→	E 4	ENT					
GX OUT	SHFT	F 5	→	PREV	PREV	C 2	A 0	→	E 4	ENT				

La instrucción Or with Stack (ORS)

DS5	Usado
HPP	Usado

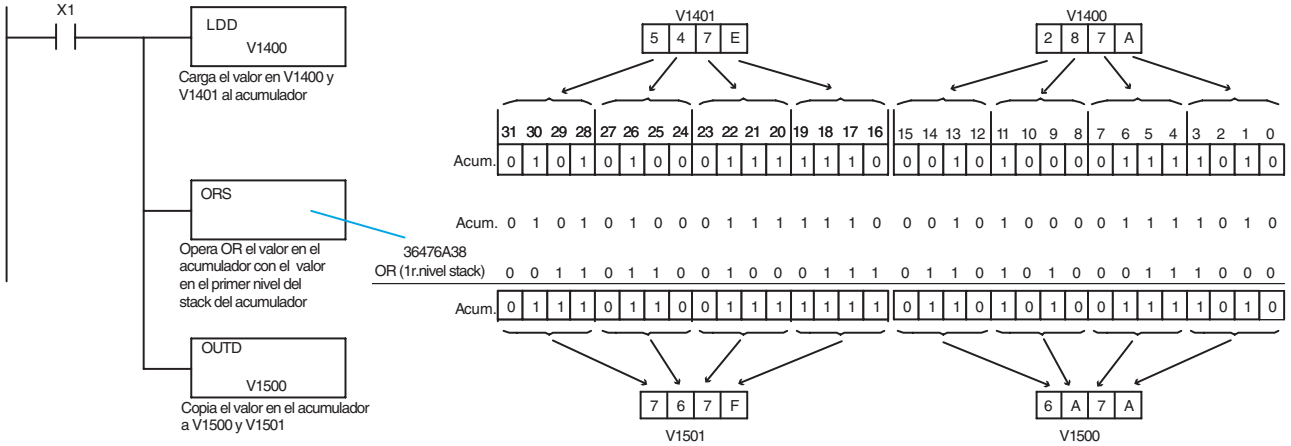
La instrucción ORS es una instrucción de 32 bits que opera OR lógicamente el valor en el acumulador con el primer nivel del Stack del acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador se quita del Stack y todos los valores son movidos un nivel para arriba. Indicaciones de estado discretas SP indican si el resultado del OR con el Stack es cero o un número negativo (el bit más significativo está ON).

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON cuando el valor en el acumulador es negativo.

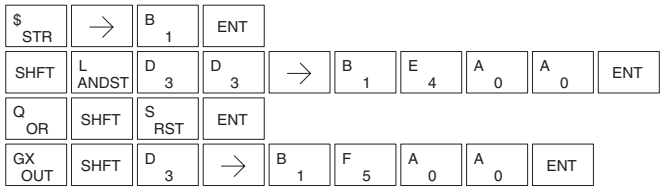
NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente cuando X1 está ON, el valor binario en el acumulador será operado OR con el valor binario en el primer nivel del Stack. El resultado se va al acumulador.

DirectSOFT



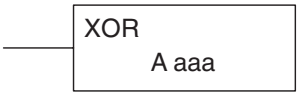
Programador D2-HPP



La instrucción Exclusive Or (XOR)

DS5	Usado
HPP	Usado

La instrucción XOR es una instrucción de 16 bits que realiza un OR exclusivo entre el valor en los 16 bits más bajos del acumulador y una localización especificada de memoria V (Aaaa). El resultado se va al acumulador. La indicación de estado discreta SP indica si el resultado del XOR es cero.



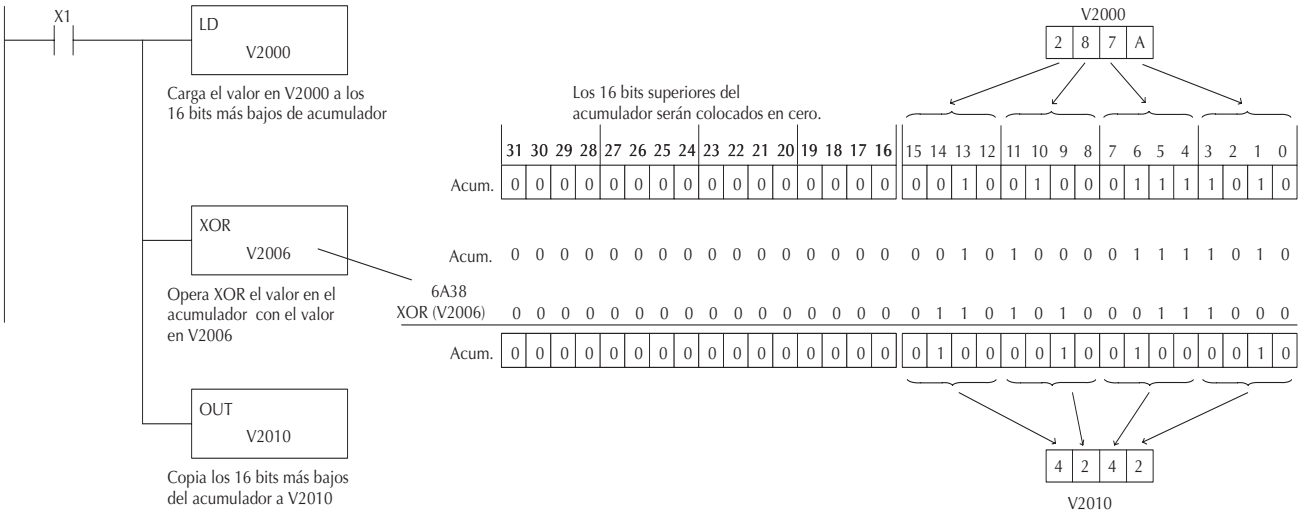
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 se carga en el acumulador usando la instrucción LD. El valor en el acumulador es operado con V2006 usando la instrucción XOR. El valor en los 16 bits más bajos del acumulador es copiado a V2010 usando la instrucción OUT.

DirectSOFT



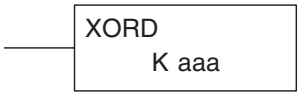
Programador D2-HPP

$ STR	→	SHFT	X SET	B 1	ENT					
SHFT	L ANDST	D 3	→	SHFT	V AND	C 2	A 0	A 0	A 0	ENT
SHFT	X SET	SHFT	Q OR	→	SHFT	V AND	C 2	A 0	A 0	G 6	ENT
GX OUT	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT		

La instrucción Exclusive Or Double (XORD)

DS5	Usado
HPP	Usado

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 se carga en el acumulador usando la instrucción LD. El valor en el acumulador es operado con V2006 usando la instrucción XOR, es decir, hace un OR exclusivo entre el acumulador y V2006. El valor en los 16 bits más bajos del acumulador es copiado a V2010 usando la instrucción OUT).



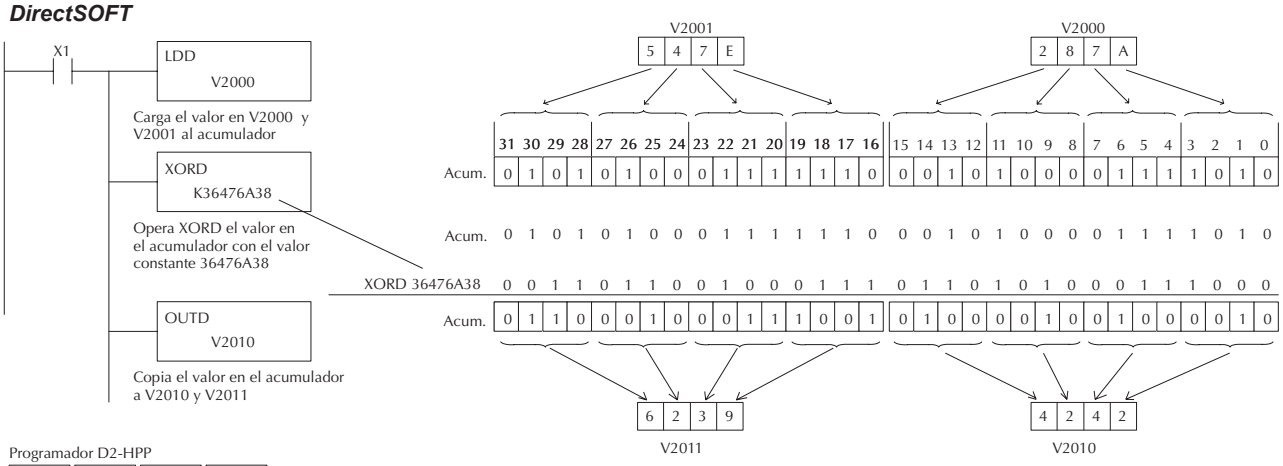
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–FFFFFFFF

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga en el acumulador usando la instrucción LDD. El valor en el acumulador es operado con un OR exclusivo con 36476A38 usando la instrucción XORD. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

DirectSOFT



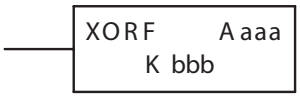
Programador D2-HPP

$ STR	→	B 1	ENT							
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT	
SHFT	X SET	Q OR	SHFT	D 3	→	SHFT	K JMP			
D 3	G 6	E 4	H 7	G 6	SHFT	A 0	SHFT	D 3	I 8	ENT
GX OUT	SHFT	D 3	→	C 2	A 0	B 1	A 0	ENT		

La instrucción Exclusive Or Formatted (XORF)

DS5	Usado
HPP	Usado

La instrucción XORF realiza un XORF un OR exclusivo del valor binario en el acumulador y un rango especificado de bits (1-32).



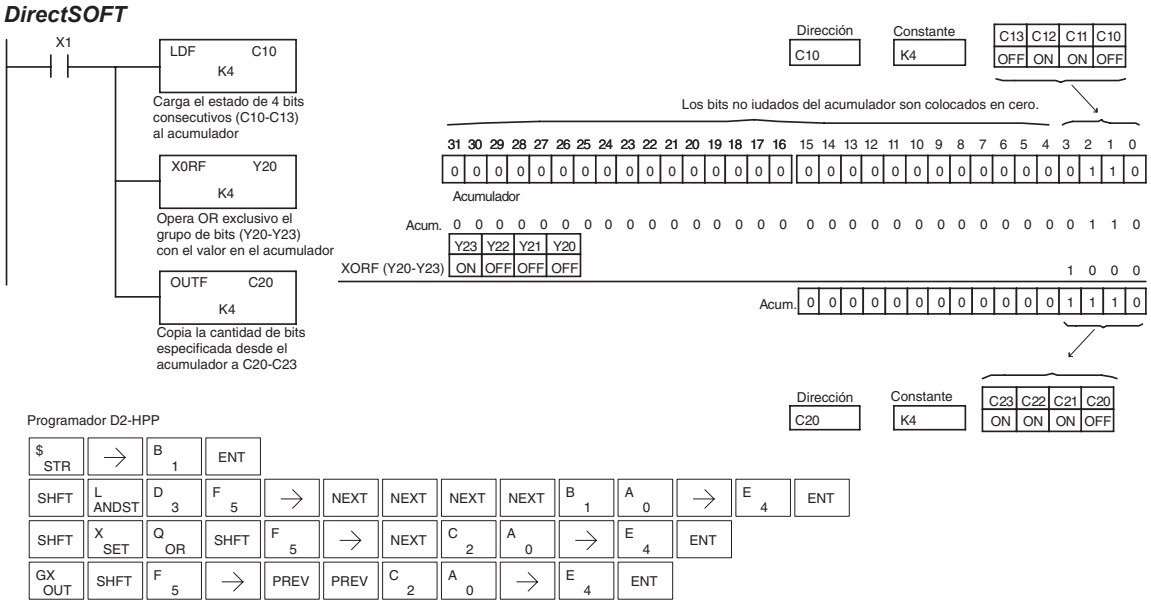
La instrucción requiere una dirección (Aaaa) de inicio y el número de bits (bbbb) a ser operados. Las indicaciones de estado discretas SP indican si el resultado de la operación es cero o negativo (el bit más significativo = 1).

Tipo de operando de datos	Rango del DL06	
A/B	aaa	bbb
Entradas X	0-777	-
Salidas . . . . . Y	0-777	-
Relevadores de control C	0-1777	-
Bits de etapas . . . . . S	0-1777	-
Bits de temporizadores . . . . . T	0-377	-
Bits de contadores . . . . . CT	177	-
Relevadores especiales . . . . . SP	0-777	-
Constante . . . . . K	-	1-32

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el modelo binario de C10-C13 (4 bits) es cargado en el acumulador usando la instrucción LDF. El valor en el acumulador será operado XROF con los bits Y20-Y23 usando la instrucción XORF. El valor en los 4 bits más bajos del acumulador



La instrucción Exclusive Or with Stack (XORS)

DS5	Usado
HPP	Usado

La instrucción XORS es una instrucción de 32 bits que realiza un OR exclusivo del valor en el acumulador con el primer nivel del Stack del acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador se quita del Stack y todos los valores son movidos un nivel para arriba. Indicaciones de estado discretas SP indican si el resultado de la instrucción XORS es cero o un número negativo (el bit más significativo está ON). Recuerde que el stack se hace cero al fin de cada barrido.

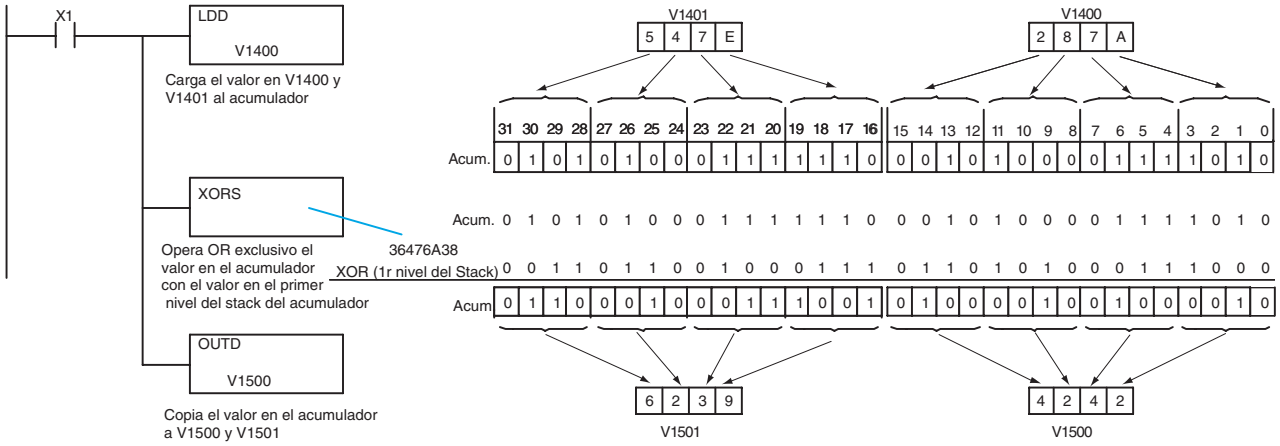
NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

5

Indicadores	Descripción
SP63	ON si el resultado en el acumulador es cero.
SP70	ON si el resultado en el acumulador es negativo

En el ejemplo siguiente cuando X1 está ON, el valor binario en el acumulador será operado OR exclusivo con el valor binario en el primer nivel del Stack del acumulador. El resultado residirá en el acumulador. La instrucción OUTD copia el valor en el acumulador a V1500.

DirectSOFT



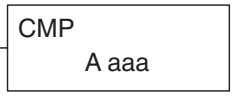
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	X SET	Q OR	SHFT	S RST	ENT				
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT	

La instrucción Compare (CMP)

DS5	Usado
HPP	Usado

La instrucción CMP es una instrucción de 16 bits que compara el valor en los 16 bits más bajos del acumulador con el valor en una localización especificada de memoria V (Aaaa). La indicación SP correspondiente del estado será prendida indicando el resultado de la comparación.

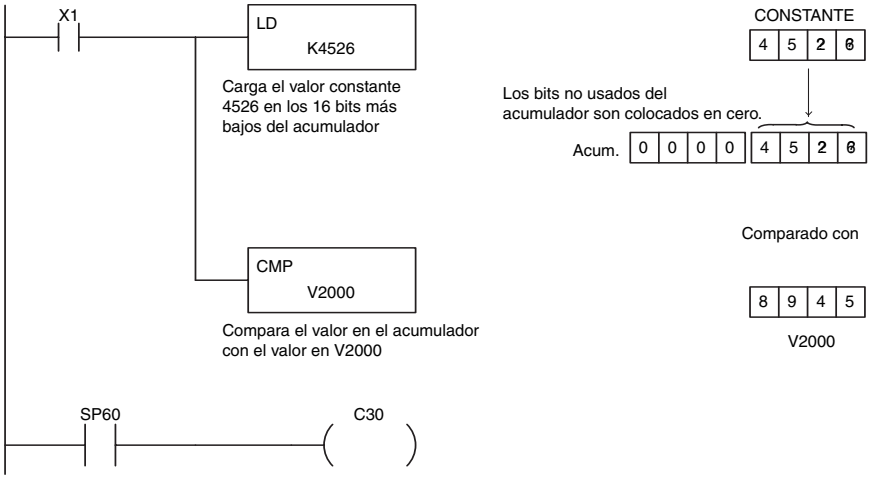


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP60	ON si el resultado en el acumulador es menor que el valor de la instrucción.
SP61	ON si el resultado en el acumulador es igual al valor de la instrucción.
SP62	ON si el resultado en el acumulador es mayor que el valor de la instrucción.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente cuando X1 está ON, la constantee 4526 es cargada en los 16 bits más bajos del acumulador usando la instrucción LD. El valor en el acumulador es comparado con el valor BCD en V2000 usando la instrucción CMP. La indicación SP correspondiente del estado será prendida indicando el resultado de la comparación. En este ejemplo, si el valor en el acumulador es menor que el valor especificado en la instrucción CMP, SP60 prenderá activando C30.



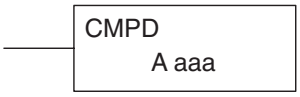
Programador D2-HPP

$ STR	→	B 1	ENT							
SHFT	L ANDST	D 3	→	SHFT	K JMP	E 4	F 5	C 2	G 6	ENT
SHFT	C 2	SHFT	M ORST	P CV	→	C 2	A 0	A 0	A 0	ENT
$ STR	→	SHFT	SP STRN	G 6	A 0	ENT				
GX OUT	→	SHFT	C 2	D 3	A 0	ENT				

La instrucción Compare Double (CMPD)

DS5	Usado
HPP	Usado

La instrucción CMPD es una instrucción de 32 bits que compara el valor en el acumulador con el valor (Aaaa), que es dos direcciones consecutivas de memoria V o una constantee de 8 dígitos (máximo). La indicación SP correspondiente del estado será activada indicando el resultado de la comparación.



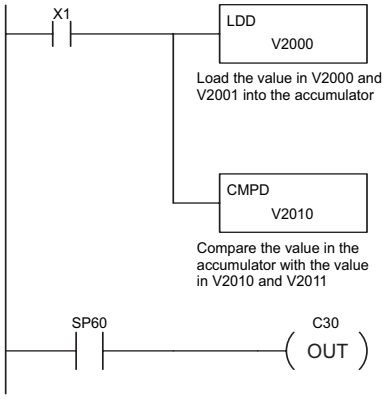
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–FFFFFFFF

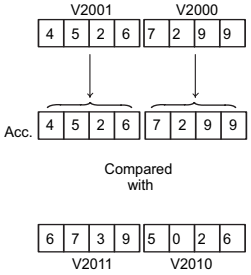
Indicadores	Descripción
SP60	ON si el resultado en el acumulador es menor que el valor de la instrucción.
SP61	ON si el resultado en el acumulador es igual al valor de la instrucción.
SP62	ON si el resultado en el acumulador es mayor que el valor de la instrucción.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente cuando X1 está ON, el valor en V2000 y V2001 se carga al acumulador usando la instrucción LDD. El valor en el acumulador es comparado con el valor en V2010 y V2011 usando la instrucción CMPD. La indicación SP correspondiente del estado será prendida indicando el resultado de la comparación.

En este ejemplo, si el valor en el acumulador es menor que el valor especificado en la instrucción, SP60 prenderá activando C30.





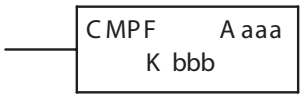
Handheld Programmer Keystrokes

$ STR	→	B 1	ENT								
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT		
SHFT	C 2	SHFT	M ORST	P CV	D 3	→	C 2	A 0	B 1	A 0	ENT
$ STR	→	SHFT	SP STRN	G 6	A 0	ENT					
GX OUT	→	SHFT	C 2	D 3	A 0	ENT					

La instrucción Compare Formatted (CMPF)

DS5	Usado
HPP	Usado

La instrucción CMPF compara el valor en el acumulador con un número especificado de bits consecutivos (1-32). La instrucción requiere una localización (Aaaa) de inicio y el número de bits (Kbbb) a ser comparado. La indicación correspondiente del estado SP será prendida indicando el resultado de la comparación.

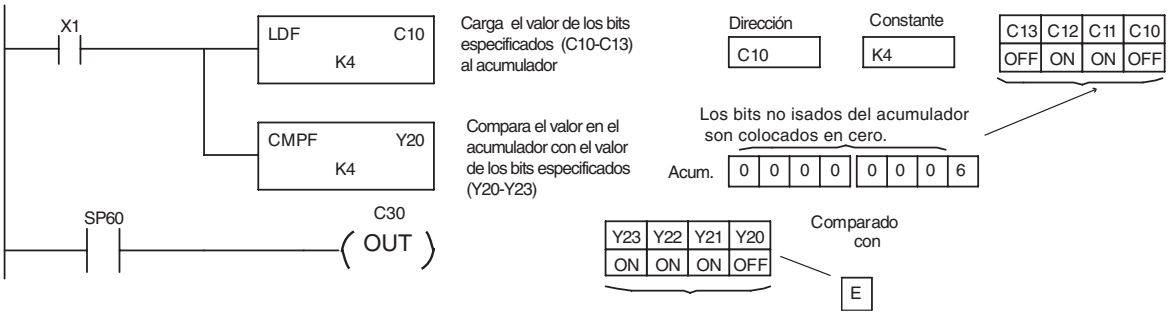


Tipo de operando de datos	Rango del DL06	
..... A/B	aaa	bbb
Entradas X	0-777	-
Salidas ..... Y	0-777	-
Relevadores de control C	0-1777	-
Bits de etapas ..... S	0-1777	-
Bits de temporizadores ..... T	0-377	-
Bits de contadores ..... CT	0-177	-
Relevadores especiales ..... SP	0-777	-
Constante ..... K	-	1-32

Indicadores	Descripción
SP60	ON si el resultado en el acumulador es menor que el valor de la instrucción..
SP61	ON si el resultado en el acumulador es igual al valor de la instrucción.
SP62	ON si el resultado en el acumulador es mayor que el valor de la instrucción.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON la instrucción LDF carga el valor binario de C10-C13 en el acumulador. La instrucción CMPF compara el valor en el acumulador al valor en Y20-Y23 (hexadecimal E). La indicación SP correspondiente del estado será prendida indicando el resultado de la comparación. . En este ejemplo, si el valor en el acumulador es menor que el valor especificado en la instrucción, SP60 prenderá activando C30.



La instrucción Compare with Stack (CMPS)

DS5	Usado
HPP	Usado

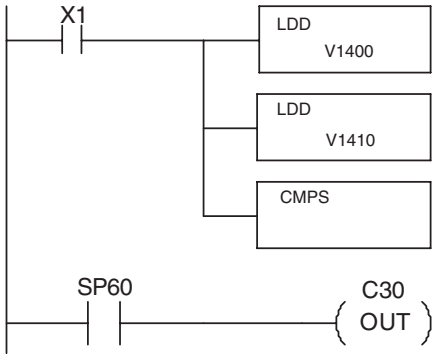
La instrucción CMPS es una instrucción de 32 bits que compara el valor en el acumulador con el valor en el primer nivel del Stack del acumulador. La indicación correspondiente del estado SP será prendida indicando el resultado de la comparación. Esto no afecta el valor en el acumulador. Recuerde que el stack se hace 0 al fin de cada barrido.

Indicadores	Descripción
SP60	ON si el resultado en el acumulador es menor que el valor de la instrucción.
SP61	ON si el resultado en el acumulador es igual que el valor de la instrucción.
SP62	ON si el resultado en el acumulador es mayor que el valor de la instrucción.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente cuando X1 está ON, el valor en V1400 y V1401 se carga en el acumulador usando la instrucción LDD. El valor en V1410 y V1411 se carga en el acumulador usando la instrucción LDD. El valor que se cargó en el acumulador desde V1400 y V1401 se coloca en el primer nivel del Stack cuando la segunda instrucción LDD es ejecutada. El valor en el acumulador es comparado con el valor en el primer nivel del Stack del acumulador usando la instrucción CMPS. La indicación SP correspondiente del estado será prendida indicando el resultado de la comparación. En este ejemplo, si el valor en el acumulador es menor que el valor en el Stack , SP60 prenderá, activando C30.

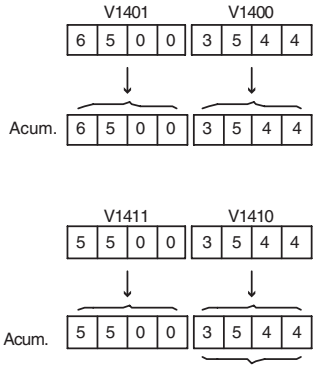
DirectSOFT



Carga el valor en V1400 y V1401 al acumulador

Carga el valor en V1410 y V1411 al acumulador

Compara el valor en el acumulador con el valor en el primer nivel del stack del acumulador



Comparado con el primer nivel del stack

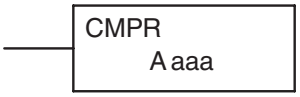
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	B 1	A 0	ENT
SHFT	C 2	SHFT	M ORST	P CV	S RST	ENT			
$ STR	PREV	G 6	A 0	ENT					
GX OUT	→	NEXT	NEXT	NEXT	SHFT	C 2	D 3	A 0	ENT

La instrucción Compare Real Number (CMPR)

DS5	Usado
HPP	Usado

La instrucción CMPR compara un valor del número real en el acumulador con dos direcciones consecutivas de memoria V que contienen un número real. La indicación correspondiente del estado SP será prendida indicando el resultado de la comparación. Ambos números a ser comparados tienen 32 bits.



Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . R	-3.402823E+ 038 hasta + -3.402823E+ 038

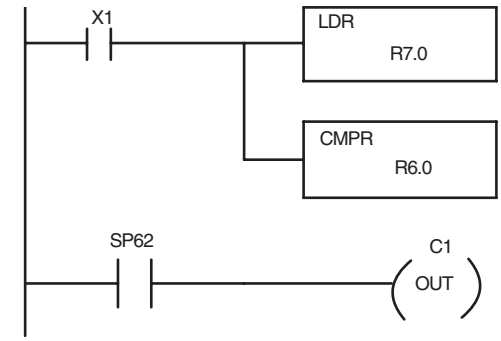
Indicadores	Descripción
SP60	ON si el resultado en el acumulador es menor que el valor de la instrucción..
SP61	ON si el resultado en el acumulador es igual que el valor de la instrucción.
SP62	ON cuando el valor en el acumulador es mayor que el valor de la instrucción.
SP71	ON en cualquier momento que la memoria V especificada por un puntero (P) no es válida

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

NOTA: El número real no es absolutamente preciso; permite un rango desde negativo hasta positivo, pero no es muy preciso ya que solo representa 23 bits de resolución.

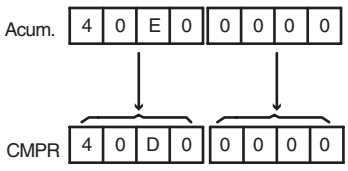
En el ejemplo siguiente cuando X1 está ON, la instrucción LDR carga la representación real del número 7,0 en el acumulador. La instrucción CMPR compara el contenido del acumulador con la representación real del número 6,0. Ya que 7 > 6, la indicación discreta SP correspondiente del estado es activada (el relevador especial SP62) activando el relevador de control C1.

DirectSOFT



Carga la representación del número real del decimal 7 al acumulador

Compara el valor con la representación del número real del decimal 6

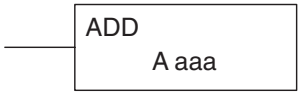


Instrucciones aritméticas

La instrucción Add (ADD)

DS5	Usado
HPP	Usado

ADD es una instrucción de 16 bits que suma un valor BCD en el acumulador con un valor BCD en una direccción de memoria V (Aaaa). **No se puede usar una constantee K como parámetro en la instrucción.** El resultado se va al acumulador.

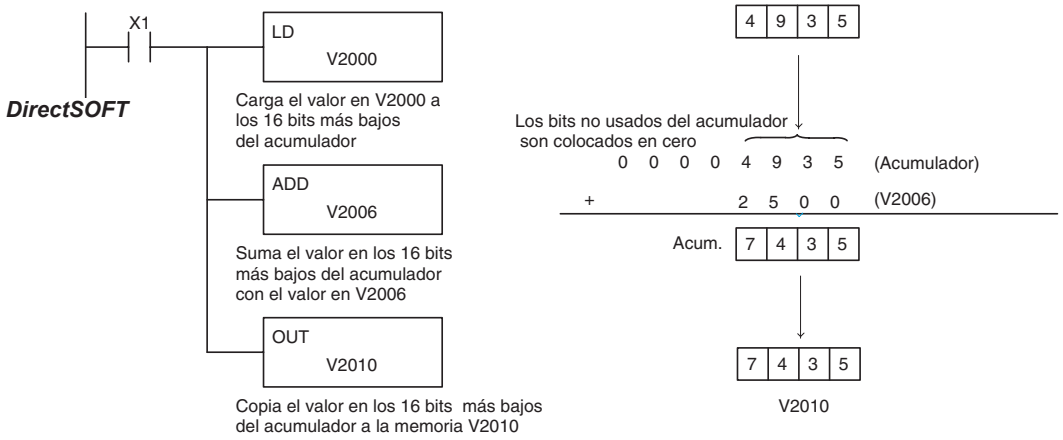


Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON cuando el resultado de la operación causa que el valor en el acumulador sea cero.
SP66	ON cuando el resultado de la operación de 16 bits resulta en un "pasa para" .
SP67	ON cuando el resultado de la operación de 32 bits resulta en un "pasa para" .
SP70	ON en cualquier momento que el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra uno de tipo diferente.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V2000 en el acumulador usando la instrucción LD. El valor en los 16 bits más bajos del acumulador es sumado al valor en V2006 usando la instrucción ADD. El valor en el acumulador es copiado a V2010 usando la instrucción OUT.



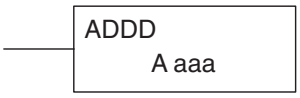
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	→	C 2	A 0	A 0	A 0	ENT	
SHFT	A 0	D 3	D 3	→	C 2	A 0	A 0	G 6	ENT
GX OUT	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT	

La instrucción Add Double (ADDD)

DS5	Usado
HPP	Usado

ADDD es una instrucción de 32 bits que suma el valor BCD en el acumulador con un valor BCD (Aaaa), que son 2 direcciones consecutivas de memoria V o una constantee de 8 dígitos (max) BCD. El resultado se va al acumulador.

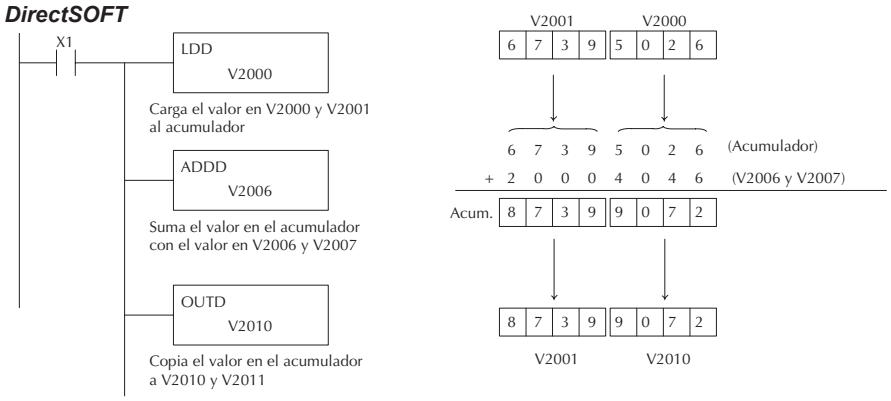


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–99999999

Indicadores	Descripción
SP63	ON cuando el resultado de la operación causa que el valor en el acumulador sea cero.
SP66	ON cuando el resultado de la operación de 16 bits resulta en un "pasa para" .
SP67	ON cuando el resultado de la operación de 32 bits resulta en un "pasa para".
SP70	ON en cualquier momento que el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra uno de tipo diferente.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga en el acumulador usando la instrucción LDD. El valor en el acumulador se suma con el valor en V2006 y V2007 usando la instrucción ADDD. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.



Programador D2-HPP

$ STR	→	B 1	ENT							
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT	
SHFT	A 0	D 3	D 3	D 3	→	C 2	A 0	A 0	G 6	ENT
GX OUT	SHFT	D 3	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT

La instrucción Add Real (ADDR)

DS5	Usado
HPP	Usado

ADDR
A aaa

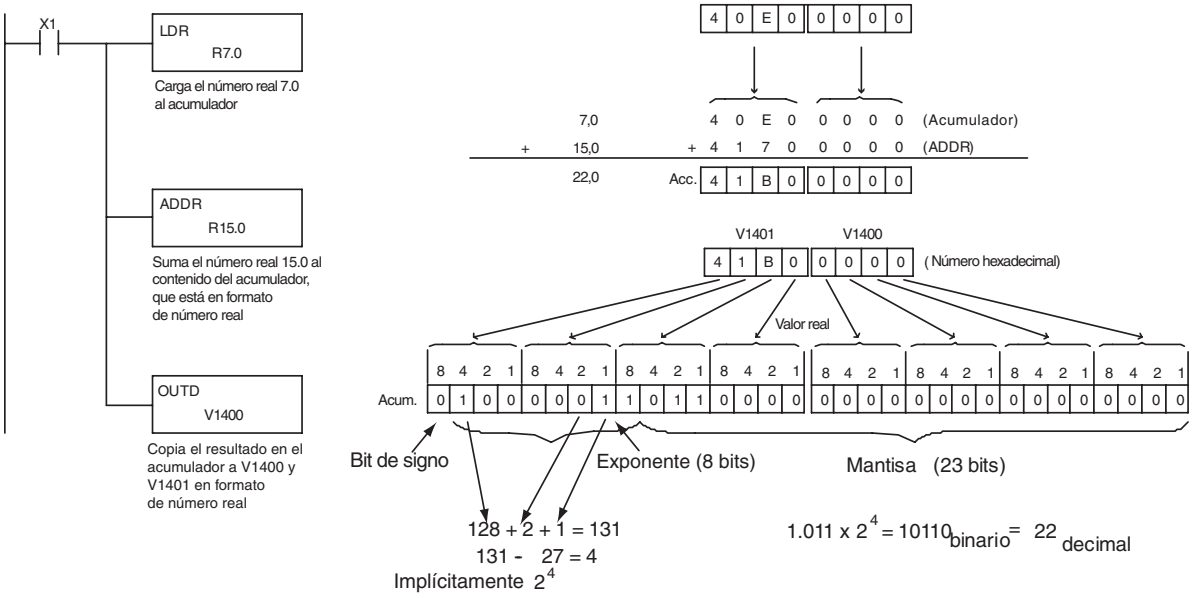
La instrucción ADDR suma un número real en el acumulador con una constantee real o un número real que ocupa dos direcciones consecutivas de memoria V. El resultado se va al acumulador. Ambos números deben estar de acuerdo al formato de punto flotante IEEE de 32 bits.

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . R	-3.402823E+ 38 to + -3.402823E+ 38

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP71	ON en cualquier momento que la memoria V especificada por un puntero (P) no es válida.
SP72	ON cuando el valor en el acumulador es un número de punto flotante inválido.
SP73	ON cuando una suma o sustracción con signo da como resultado un bit de signo incorrecto.
SP74	ON cuando una operación de punto flotante resulta en un error underflow.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

Este ejemplo muestra la convención de punto flotante IEEE de 32 bits



NOTA: El programador D2-HPP no permite entrar números reales con conversión automática al formato IEEE de 32 bits. Ud debe utilizar DirectSOFT en este caso, para usar esta función.

La instrucción Subtract (SUB)

DS5	Usado
HPP	Usado

SUB (Resta) es una instrucción de 16 bits que resta el valor BCD (Aaaa) en una dirección de memoria V del valor BCD en los 16 bits más bajos del acumulador. El resultado se va al acumulador.

```
SUB
  A aaa
```

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP64	ON cuando la instrucción de resta de 16 bits pide un “préstamo”.
SP65	ON cuando la instrucción de resta de 32 bits pide un “préstamo”.
SP70	ON cuando el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

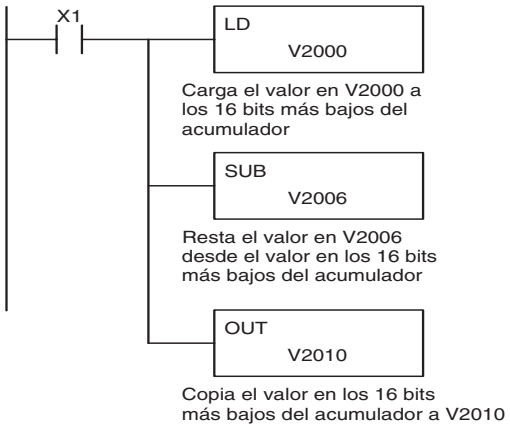
5

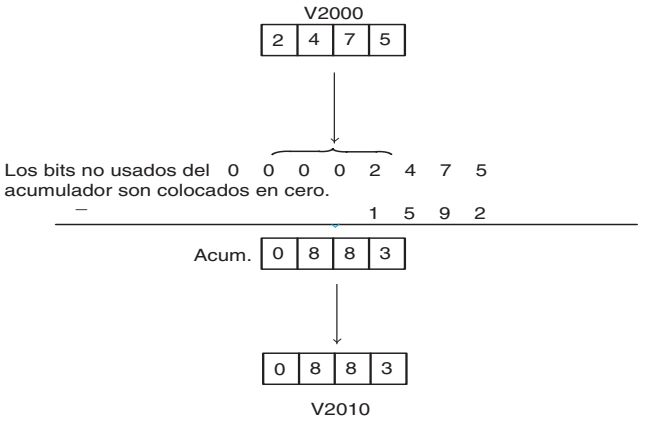
NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 se carga en el acumulador usando la instrucción LD.

El valor en V2006 se resta del valor en el acumulador usando la instrucción SUB. El valor en el acumulador es copiado a V2010 usando la instrucción OUT.

DirectSOFT





Programador D2-HPP

$ STR	→	B 1	ENT							
SHFT	L ANDST	D 3	→	C 2	A 0	A 0	A 0	ENT		
SHFT	S RST	U ISG	B 1	→	SHFT	V AND	C 2	A 0	A 0	G 6	ENT
GX OUT	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT		

La instrucción Subtract Double (SUBD)

DS5	Usado
HPP	Usado

SUBD
A aaa

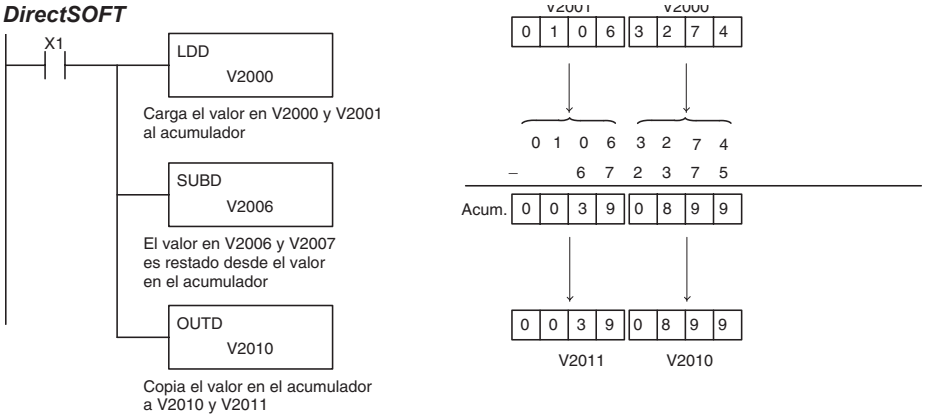
Resta Doble SUBD es una instrucción de 32 bits que resta el valor BCD (Aaaa), que puede ser 2 direcciones consecutivas de memoria V o una constantee de 8 dígitos (máximo), desde el valor BCD en el acumulador.

Tipo de operando de datos	Rango del DL06
. . . . . A	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–99999999

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP64	ON cuando la instrucción de resta de 16 bits pide un "préstamo".
SP65	ON cuando la instrucción de resta de 32 bits pide un "préstamo".
SP70	ON cuando el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V2000 y V2001 en el acumulador usando la instrucción LDD. El valor en V2006 y V2007 se resta del valor en el acumulador. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.



Programador D2-HPP

$ STR	→	B 1	ENT								
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT		
SHFT	S RST	SHFT	U ISG	B 1	D 3	→	C 2	A 0	A 0	G 6	ENT
GX OUT	SHFT	D 3	→	C 2	A 0	B 1	A 0	ENT			

La instrucción Subtract Real (SUBR)

DS5	Usado
HPP	N/A

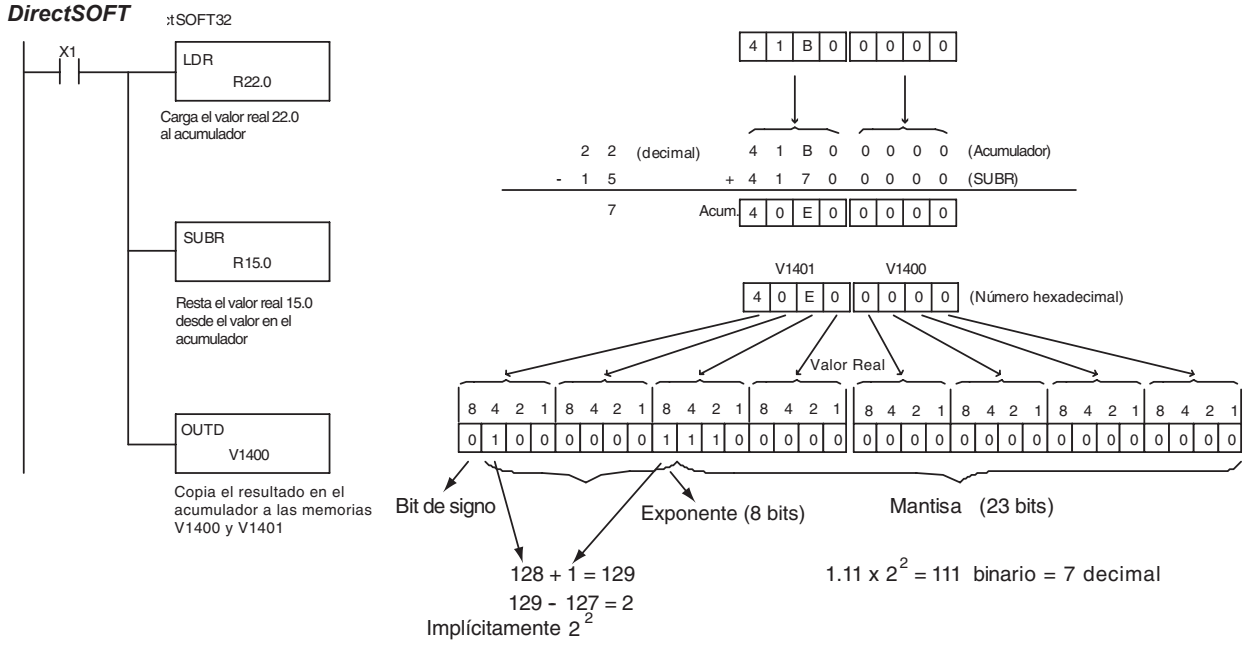
La instrucción SUBR resta un número real en el acumulador de una constantee real o un número real que ocupa 2 direcciones consecutivas de memoria V. El resultado se va al acumulador. Ambos números deben seguir el formato de punto flotante IEEE de 32 bits.

```
SUBR
   A aaa
```

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . R	-3.402823E + 38 hasta +-3.402823E + 38

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP71	ON en cualquier momento que la memoria V especificada por un puntero (P) no es válida.
SP72	ON cuando el valor en el acumulador es un número de punto flotante inválido.
SP73	ON cuando una suma o sustracción con signo da como resultado un bit de signo incorrecto.
SP74	On cuando una operación de punto flotante resulta en un error de underflow.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

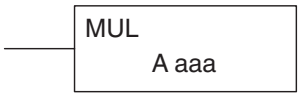


NOTA: El programador D2-HPP no permite entrar números reales con conversión automática al formato IEEE de 32 bits. Ud debe usar DirectSOFT en este caso, para usar esta función.

La instrucción Multiply (MUL)

DS5	Usado
HPP	Usado

MUL es una instrucción de 16 bits que multiplica el valor BCD (Aaaa), que es una dirección de memoria V o una constantee de 4 dígitos (max.) por el valor BCD en los 16 bits más bajos del acumulador. El resultado puede ser de hasta 8 dígitos y se va al acumulador.



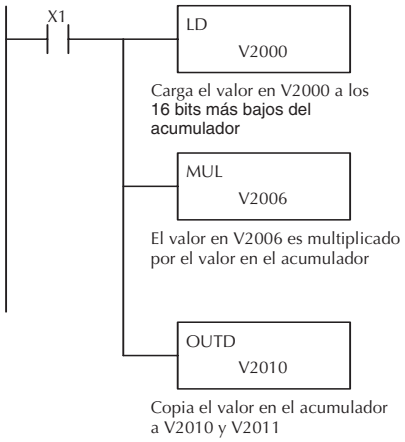
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–9999

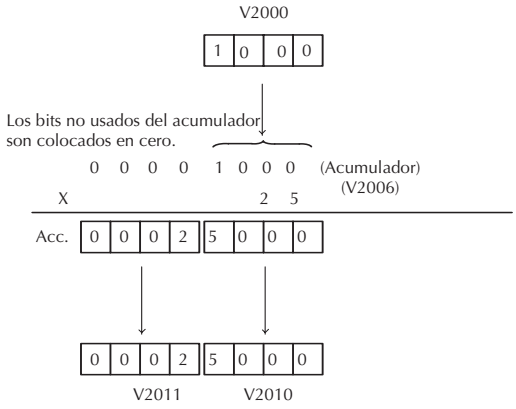
Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

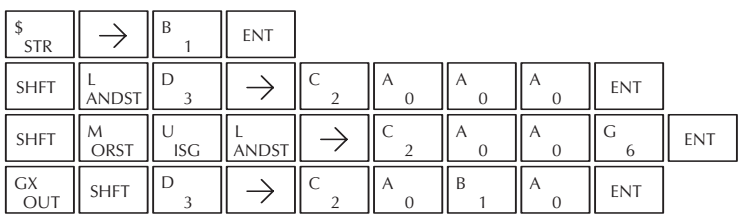
En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V2000 al acumulador usando la instrucción LD. El valor en V2006 es multiplicado por el valor en el acumulador. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

DirectSOFT





Programador D2-HPP



La instrucción Multiply Double (MULD)

DS5	Usado
HPP	Usado

MULD es una instrucción de 32 bits que multiplica el valor de 8 dígitos BCD en el acumulador por el valor de 8 dígitos BCD en 2 direcciones consecutivas de memoria V especificadas en la instrucción. Los 8 dígitos más bajos del resultado se van al acumulador. Los dígitos superiores del resultado se van al Stack del acumulador.

```
MULD
    A aaa
```

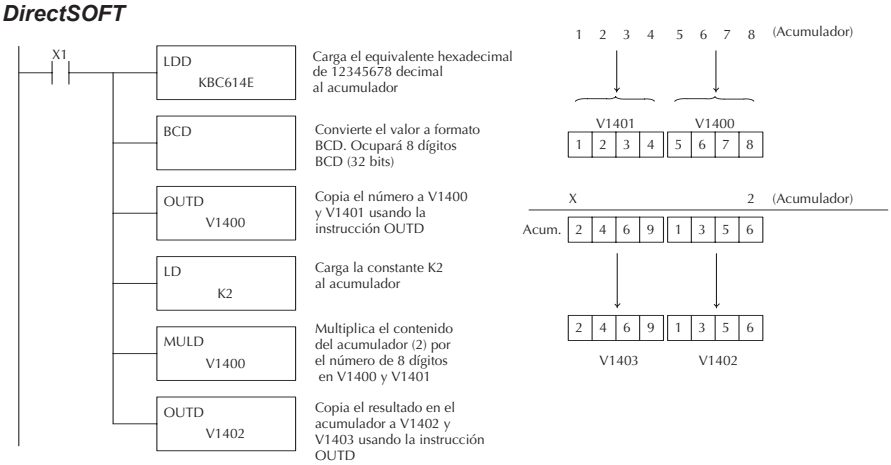
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, la constantee hexadecimal Kbc614e se carga en el acumulador. Cuándo es convertido a BCD el número es "12345678". Esos números se almacenan en V1400 y V1401. Después de cargar la constantee K2 en el acumulador, se multiplica por 12345678, que es 24691356.

DirectSOFT



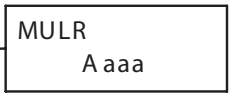
Programador D2-HPP

$ STR	→	B 1	ENT																				
SHFT	L ANDST	D 3	D 3	→	PREV	SHFT	B 1	C 2	SHFT	G 6	B 1	E 4	SHFT	E 4	ENT								
SHFT	B 1	C 2	D 3	ENT																			
GX OUT	SHFT	D 3	→	B 1	E 4	A 0	A 0	ENT															
SHFT	L ANDST	D 3	→	PREV	C 2	ENT																	
SHFT	M ORST	U ISG	L ANDST	D 3	→	B 1	E 4	A 0	A 0	ENT													
GX OUT	SHFT	D 3	→	B 1	E 4	A 0	C 2	ENT															

La instrucción Multiply Real (MULR)

DS5	Usado
HPP	Usado

La instrucción MULR multiplica un número real en el acumulador con una constantee real o un número real que ocupa dos direcciones consecutivas de memoria V. El resultado se va al acumulador. Ambos números deben estar de acuerdo al formato de punto flotante IEEE.

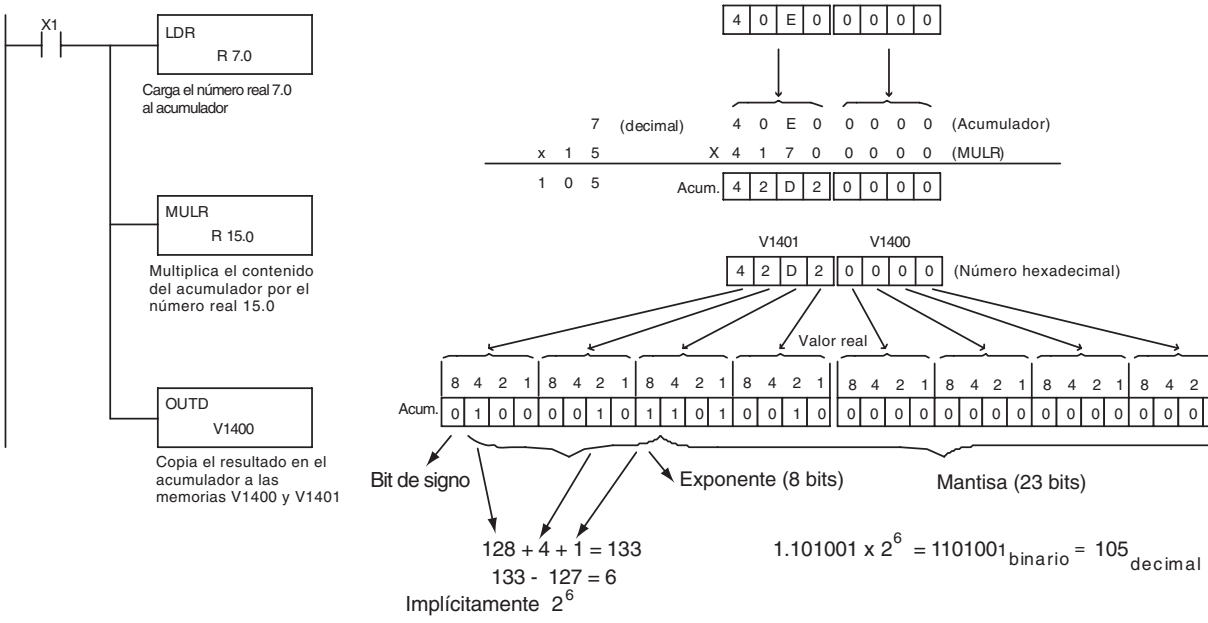


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante real . . . . . R	-3.402823E +38 to + -3.402823E +38

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP71	ON en cualquier momento que la memoria V especificada por un puntero (P) no es válida.
SP72	ON cuando el valor en el acumulador es un número de punto flotante inválido.
SP73	ON cuando una suma o sustracción con signo da como resultado un bit de signo incorrecto.
SP74	On cuando una operación de punto flotante resulta en un error de underflow.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT

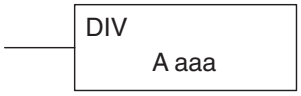


NOTE: The current HPP does not support real number entry with automatic conversion to the 32-bit IEEE format. You must use DirectSOFT for this feature.

La instrucción Divide (DIV)

DS5	Usado
HPP	Usado

DIV es una instrucción de 16 bits que divide el valor BCD en el acumulador por un valor BCD (Aaaa), que es una localización de memoria V o una constantee de 4 dígitos (max.) La primera parte del cociente se va al acumulador y el resto se va al primer nivel del Stack.



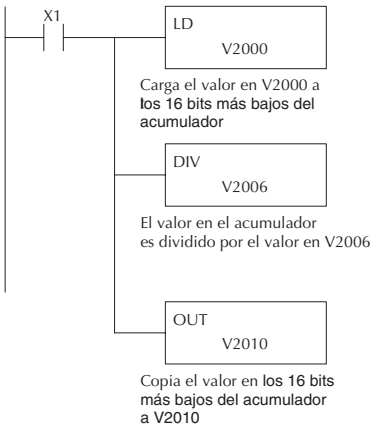
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–9999

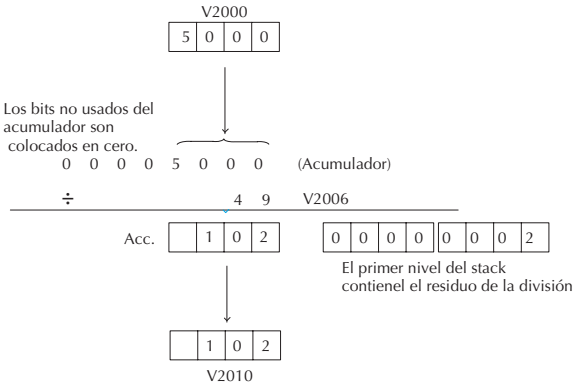
Indicadores	Descripción
SP53	On cuando el valor del operando es más grande de lo que puede aceptar el acumulador.
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V2000 al acumulador usando la instrucción LD. El valor en el acumulador será dividido por el valor en V2006 usando la instrucción DIV. El valor en el acumulador es copiado a V2010 usando la instrucción OUT.

DirectSOFT





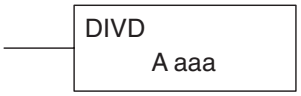
Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	C 2	A 0	A 0	A 0	ENT
SHFT	D 3	I 8	V AND	→	C 2	A 0	A 0	G 6	ENT
GX OUT	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT

La instrucción Divide Double (DIVD)

DS5	Usado
HPP	Usado

DIVD es una instrucción de 32 bits que divide el valor BCD en el acumulador por un valor BCD (Aaaa), que se debe obtener de 2 direcciones consecutivas de memoria V. (No se puede usar una constantee como el parámetro de la instrucción) La primera parte del cociente se va al acumulador y el resto se va al primer nivel del Stack.

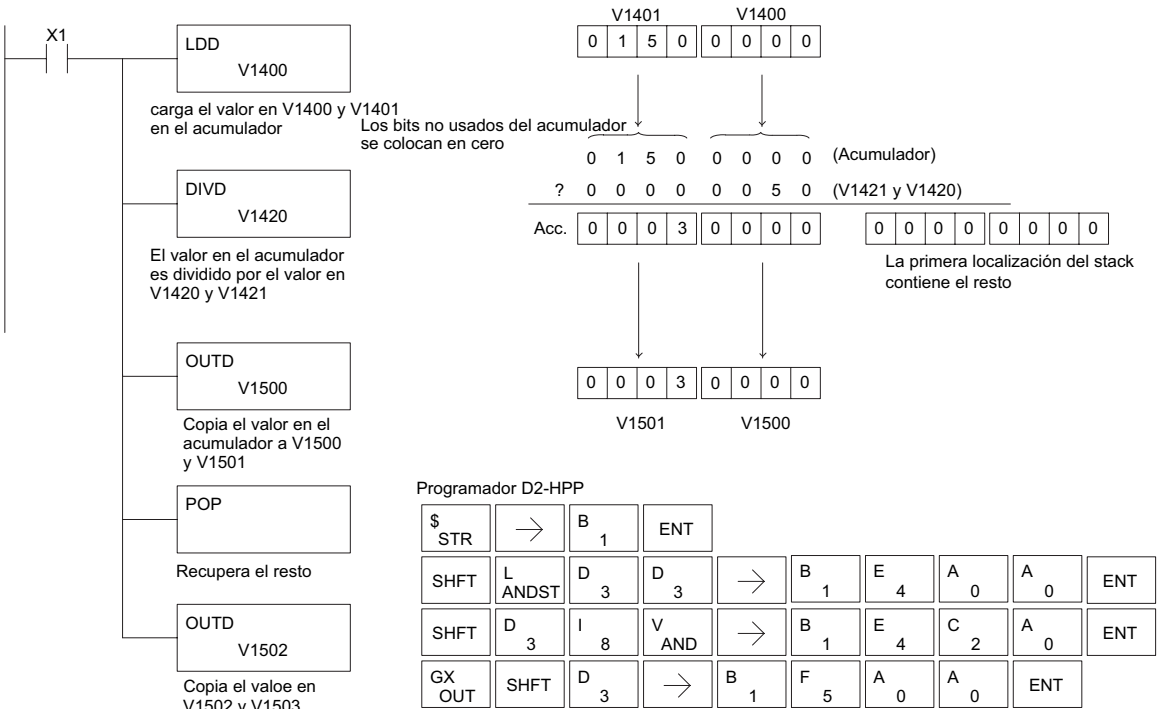


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

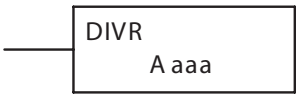
En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El valor en el acumulador es dividido por el valor en V1420 y V1421 usando la instrucción DIVD. La primera parte del cociente se va al acumulador y el resto se va al primer nivel del Stack . El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.



La instrucción Divide Real (DIVR)

DS5	Usado
HPP	N/A

La instrucción DIVR divide un número real en el acumulador por una constantee real o por un número real que ocupa 2 direcciones consecutivas de memoria V. El resultado se va al acumulador. Ambos números deben estar de acuerdo al formato punto flotante IEEE.

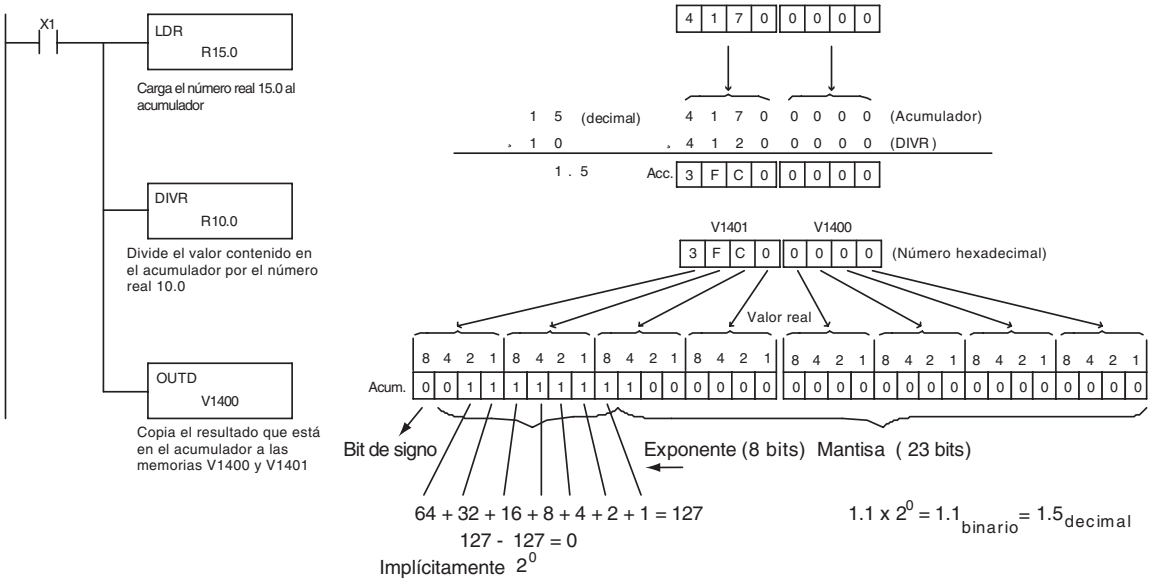


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante Real . . . . . R	-3.402823E + 38 hasta + -3.402823E + 38

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP71	ON en cualquier momento que la memoria V especificada por un puntero (P) no es válida.
SP72	ON cuando el valor en el acumulador es un número de punto flotante inválido.
SP74	ON cuando una operación de punto flotante resulta en un error underflow.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT

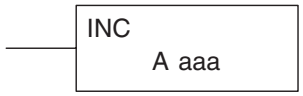


NOTA: El programador D2-HPP no permite entrar números reales con conversión automática al formato IEEE de 32 bits. Ud debe usar DirectSOFT en este caso, para usar esta función.

La instrucción Increment (INC)

DS5	Usado
HPP	Usado

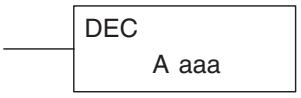
La instrucción INC incrementa un valor BCD en "1"en una dirección especificada de memoria V cada vez que se ejecuta la instrucción.



La instrucción Decrement (DEC)

DS5	Usado
HPP	Usado

La instrucción DEC decrementa en "1" un valor BCD en una dirección especificada de memoria V cada vez que se ejecuta la instrucción.

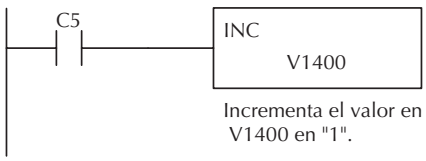


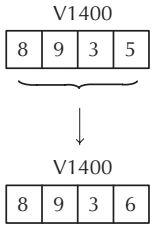
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

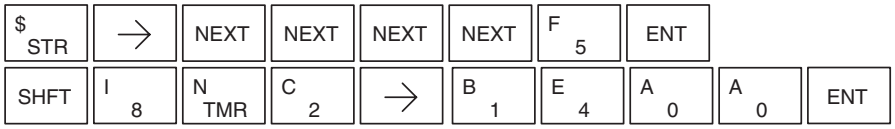
NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT



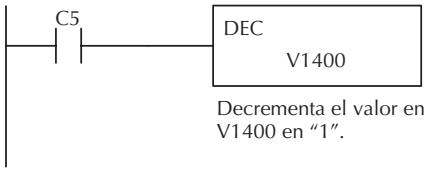


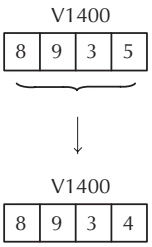
Programador D2-HPP



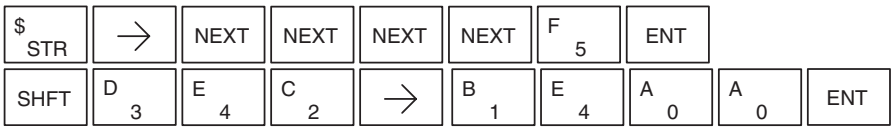
En el ejemplo siguiente, cuándo C5 está ON, el valor contenido en V1400 aumenta en 1.

DirectSOFT





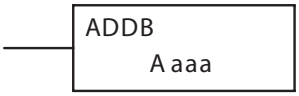
Programador D2-HPP



La instrucción Add Binary (ADDB)

DS5	Usado
HPP	Usado

ADDB es una instrucción de 16 bits que suma el valor binario en los 16 bits más bajos del acumulador con el valor (Aaaa) binario que es una localización de memoria V o una constantee de 16 bits. El resultado puede ser de hasta de 32 bits y se va al acumulador. Note que se puede usar el complemento de 2 para expresar números negativos. Vea el apéndice J para más explicaciones.

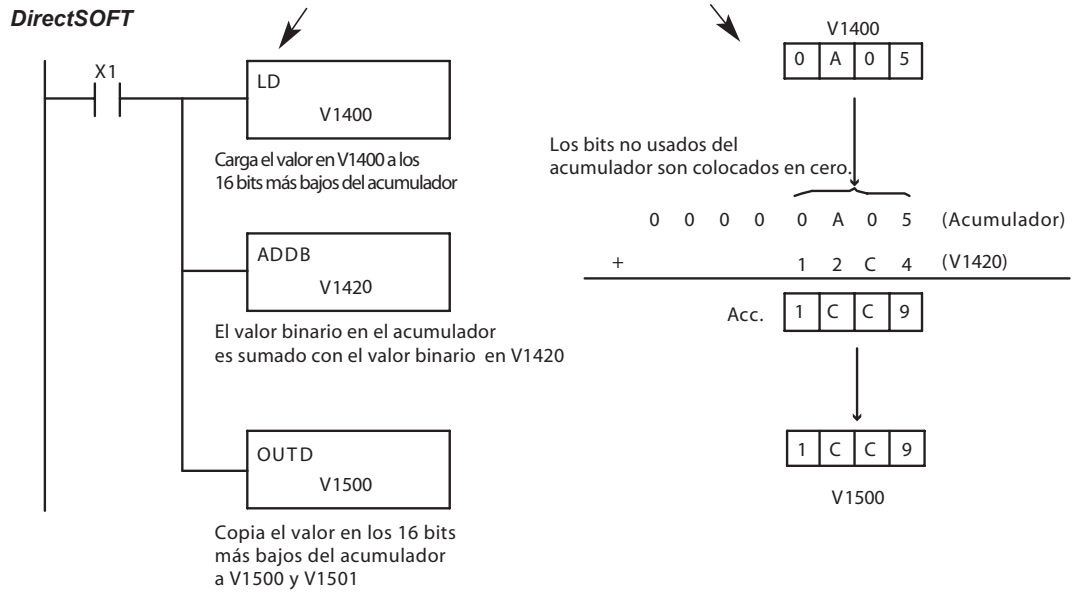


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0-FFFF, h=65636

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP66	ON cuando la instrucción de 16 bits de suma resulta en un "pasa para".
SP67	ON cuando la instrucción de 32 bits de suma resulta en un "pasa para".
SP70	ON cuando el valor en el acumulador es negativo.
SP73	ON si una suma o resta con signo resulta con el bit de un signo incorrecto.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el siguiente ejemplo, cuando X1 está ON, el valor en V1400 se carga en el acumulador usando la instrucción LD. El valor binario en el acumulador es sumado al valor binario en

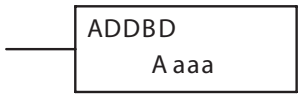


V1420 usando la instrucción ADDB. El valor en el acumulador es copiado a V1500 y V1501

La instrucción Add Binary Double (ADDBD)

DS5	Usado
HPP	Usado

ADDBD es una instrucción de 32 bits que suma el valor binario en el acumulador con el valor (Aaaa), que corresponde a dos localizaciones consecutivas de memoria V o una constantee binaria de 32 bits. El resultado reside en el acumulador. Note que se puede usar el complemento de 2 para expresar números negativos. Vea el apéndice J para más explicaciones.

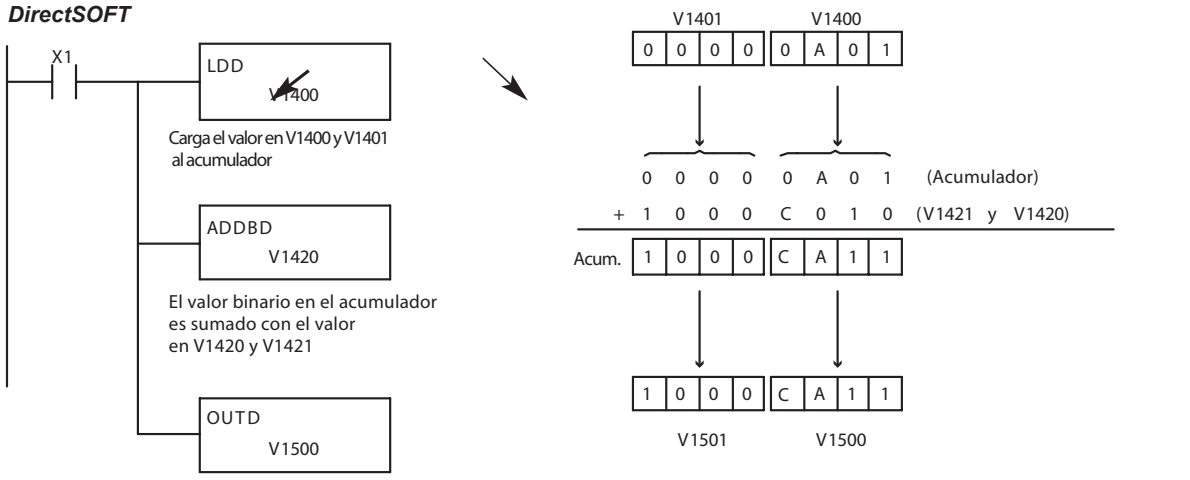


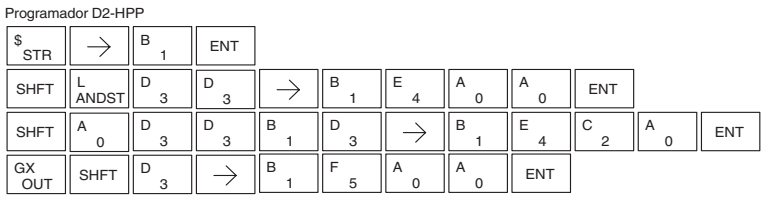
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0-FFFF FFFF

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP66	ON cuando la instrucción de 16 bits de suma resulta en un "pasa para".
SP67	ON cuando la instrucción de 32 bits de suma resulta en un "pasa para".
SP70	ON cuando el valor en el acumulador es negativo.
SP73	ON si una suma o resta con signo resulta con el bit de un signo incorrecto.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El valor binario en el acumulador se suma con el valor binario en V1420 y V1421 usando la instrucción ADDBD. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.

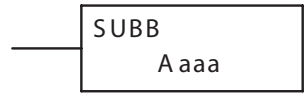




La instrucción Subtract Binary (SUBB)

DS5	Usado
HPP	Usado

SUBB es una instrucción de 16 bits que resta el valor (Aaaa) binario que es una dirección de memoria V o una constantee del valor binario en el acumulador. El resultado se va al acumulador. Note que se puede usar el complemento de 2 para expresar números negativos. Vea el apéndice I para más explicaciones.

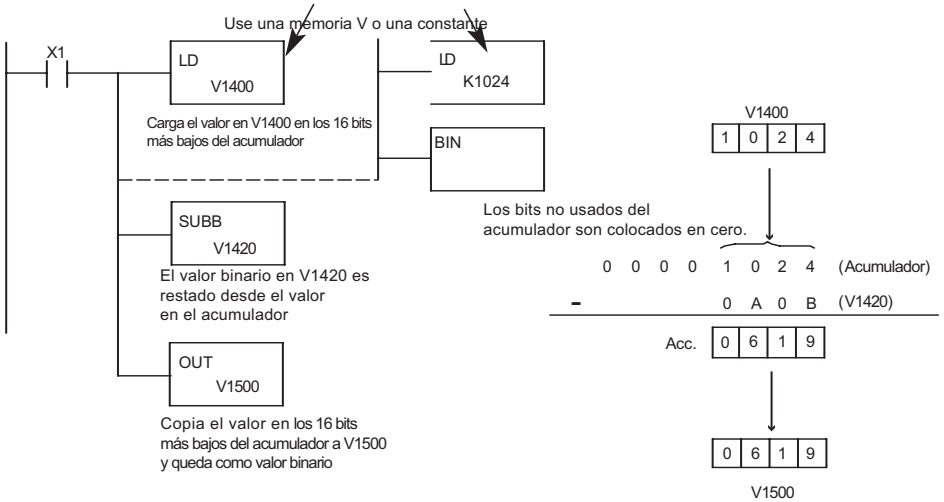


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0-FFFF, h=65636

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP64	ON cuando la instrucción de 16 bits de resta resulta en un "préstamo".
SP65	ON cuando la instrucción de 32 bits de resta resulta en un "préstamo".
SP70	ON cuando el valor en el acumulador es negativo.
SP73	ON si una suma o resta con signo resulta con el bit de un signo incorrecto.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 se cargará al acumulador usando la instrucción LD. El valor binario en V1420 es restado del valor binario en el acumulador usa la instrucción SUBB. El valor en el acumulador es copiado a V1500 usando la instrucción OUT.



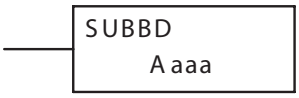
Programador D2-HPP

STR	→	X(IN)	1	ENT				
SHFT	L	D	V	1	4	0	0	ENT
SHFT	S	SHFT	U	B	B	→		
V	1	4	2	0	ENT			
OUT	SHFT	D	→	V	1	5	0	0	ENT

La instrucción Subtract Binary Double (SUBBD)

DS5	Usado
HPP	Usado

SUBBD es una instrucción de 32 bits que resta el valor (Aaaa) binario que son 2 direcciones consecutivas de memoria V o una constantee binaria de 32 bits, del valor binario en el acumulador. El resultado se va al acumulador. Note que el complemento de 2 se puede usar para expresar números negativos.

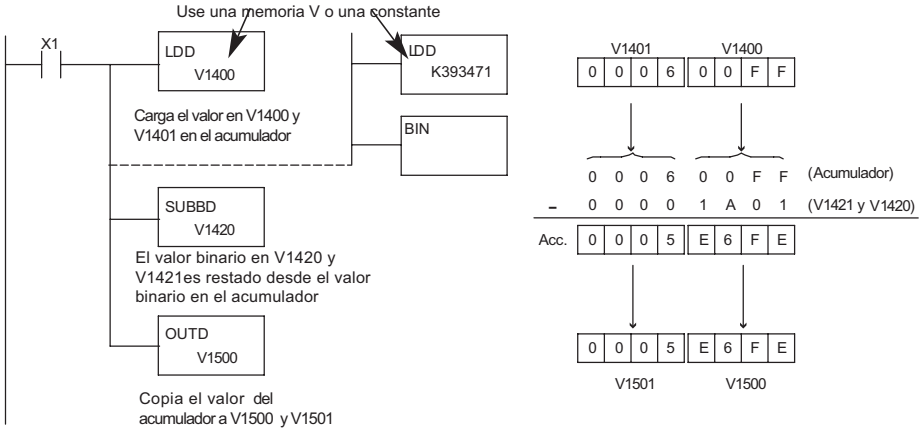


Tipo de operando de datos	Rango del DL06
..... A	aaa
Memoria V	Vea el mapa de memoria
Puntero..... P	Vea el mapa de memoria
Constante ..... K	0-FFFF FFFF

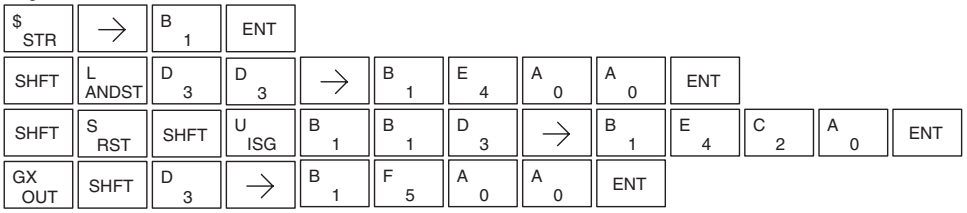
Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP64	ON cuando la instrucción de 16 bits de resta resulta en un "préstamo".
SP65	ON cuando la instrucción de 32 bits de resta resulta en un "préstamo"
SP70	ON cuando el valor en el acumulador es negativo.
SP73	ON si una suma o resta con signo resulta con el bit de un signo incorrecto.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se cargará en el acumulador usando la instrucción LDD. El valor binario en V1420 y V1421 es restado del valor binario en el acumulador usando la instrucción SUBB. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.



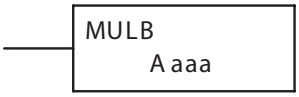
Programador D2-HPP



La instrucción Multiply Binary (MULB)

DS5	Usado
HPP	Usado

MULB es una instrucción de 16 bits que multiplica el valor (Aaaa) binario, que es una dirección de memoria V o una constantee binaria de 16 bits, por el valor binario en el acumulador. El resultado puede llegar a ser de hasta 32 bits y se va al acumulador. Note que se puede usar el complemento de 2 para expresar números negativos. Vea el apéndice J para más explicaciones.

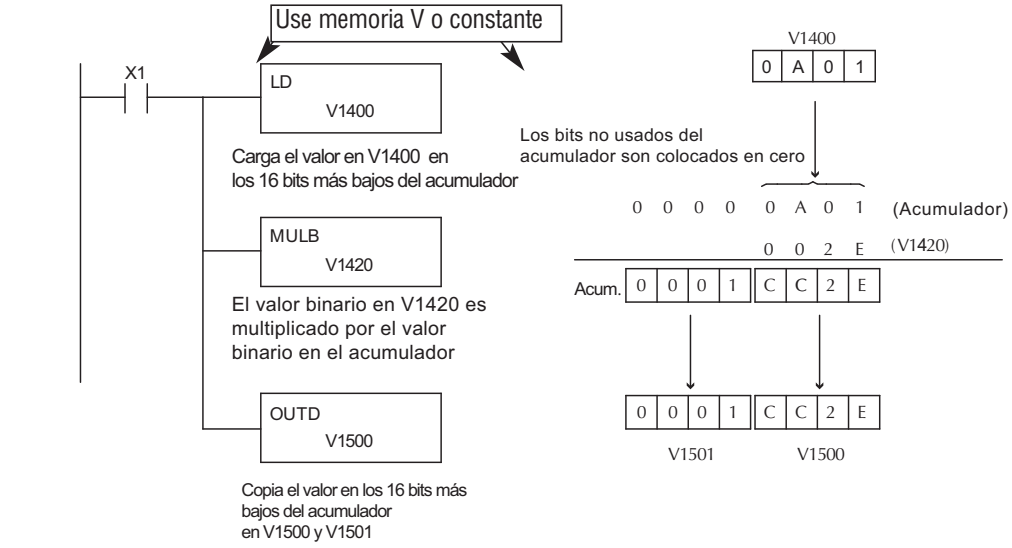


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0-FFFF

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuando X1 está ON, el valor en V1400 se carga al acumulador usando la instrucción LD. Luego el valor binario en V1420 es multiplicado por el valor binario en el acumulador usando la instrucción MULB. El valor en el acumulador es copiado a V1500 usando la instrucción OUT.



Programador D2-HPP

STR	→	X	1	ENT							
SHFT	L	D	V	1	4	0	0	ENT			
SHFT	M	U	L	B	→	V	1	4	2	0	ENT
OUT	SHFT	D	→	V	1	5	0	0	ENT		

La instrucción Divide Binary (DIVB)

DS5	Usado
HPP	Usado

DIVB es una instrucción de 16 bits que divide el valor binario en el acumulador por un valor (Aaaa) binario, que es una dirección de memoria V o una constantee binaria de 16 bits. La primera parte del cuociente se va al acumulador y el residuo se va al primer nivel del stack. Note que se puede usar el complemento de 2 para expresar números negativos. Vea el apéndice J para más explicaciones.

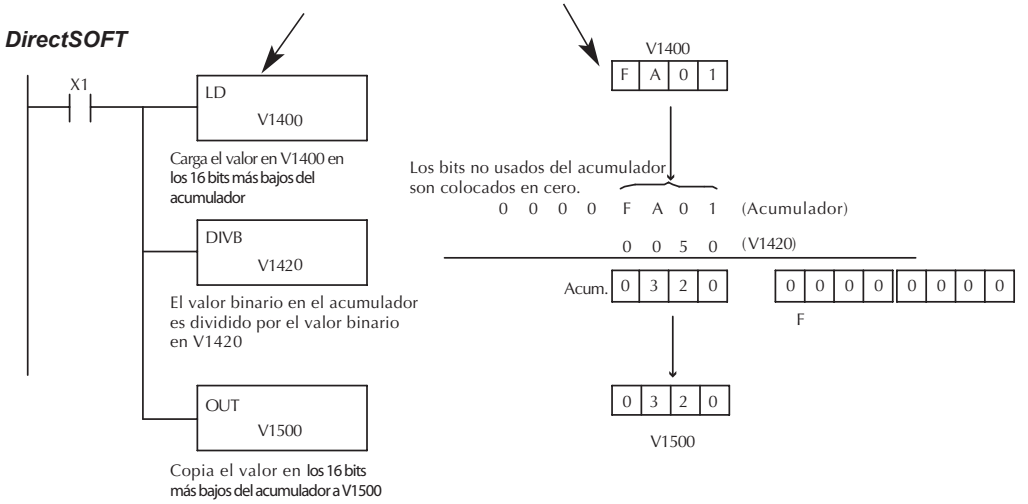
DIVB
A aaa

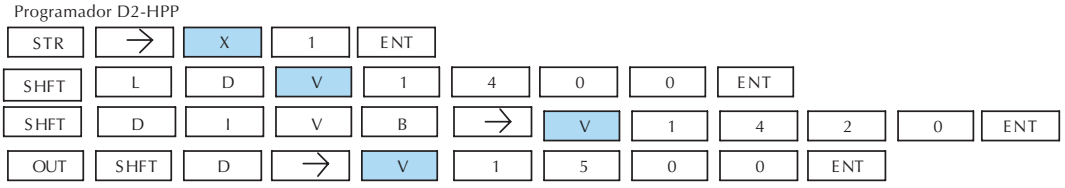
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0-FFFF

Indicadores	Descripción
SP53	ON cuando el valor del operando es mayor que lo que puede trabajar el acumulador.
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V1400 al acumulador usando la instrucción LD. El valor binario en el acumulador es dividido por el valor binario en V1420 usando la instrucción DIVB. El valor en el acumulador es copiado a V1500 usando la instrucción OUT.

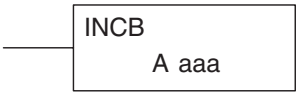




La instrucción Increment Binary (INCB)

DS5	Usado
HPP	Usado

La instrucción INCB incrementa un valor binario en "1" en una dirección especificada de memoria V cada vez que se ejecuta la instrucción.

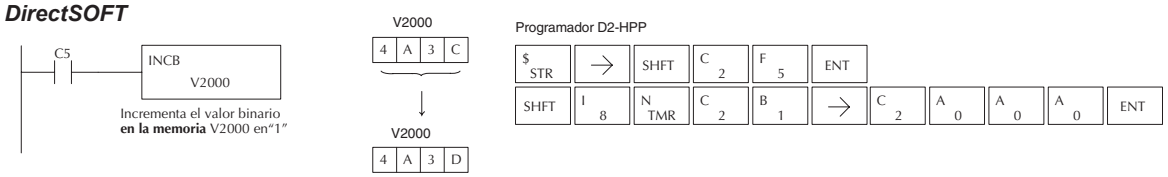


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.

5

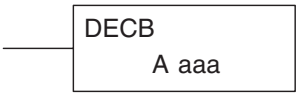
En el ejemplo siguiente cuando C5 está ON, el valor binario en V2000 es aumentado en 1.



La instrucción Decrement Binary (DECB)

DS5	Usado
HPP	Usado

La instrucción DECB decrementa en "1"un valor binario en una dirección especificada de la memoria V, cada vez que la instrucción se ejecuta

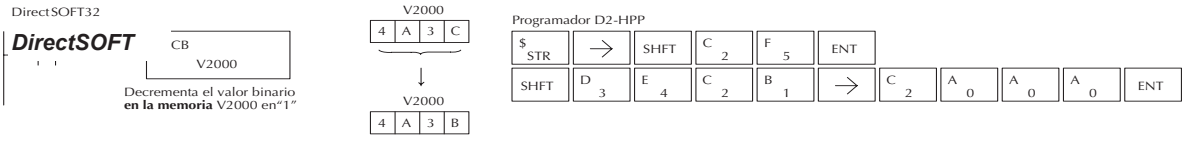


Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP63	On cuando el resultado de la instrucción causa que el valor en el acumulador sea cero.

NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP.

En el ejemplo siguiente cuando C5 está ON, el valor binario en V2000 es disminuido en 1.



La instrucción Add Formatted (ADDF)

DS5	Usado
HPP	Usado

ADDF es una instrucción de 32 bits que suma el valor BCD en el acumulador con el valor BCD (Aaaa), que es un rango de bits discretos. El rango (Kbbb) especificado puede ser 1 a 32 bits consecutivos. El resultado se va al acumulador.

```
ADDF    A aaa
        K bbb
```

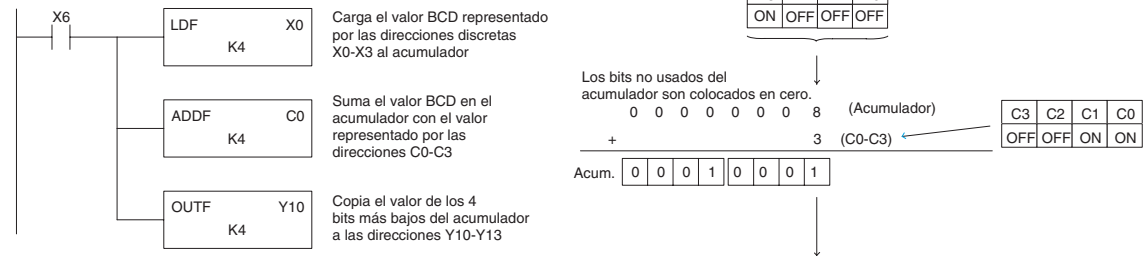
Tipo de operando de datos	Rango del DL06	
..... A	aaa	bbb
Entradas X	0–777	—
Salidas ..... Y	0–777	—
Relevadores de control C	0–1777	—
Bits de etapas ..... S	0–1777	—
Bits de temporizadores ..... T	0–377	—
Bits de contadores ..... CT	0–177	—
Relevadores especiales ..... SP	0-137 320-717	—
Global I/O ..... GX	0-3777	—
Constante ..... K	—	1–32

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP66	ON cuando la instrucción de suma de 16 bits resulta en un "pasa para".
SP67	ON cuando la instrucción de suma de 32 bits resulta en un "pasa para".
SP70	ON cuando el valor en el acumulador es negativo.
SP73	ON si se espera un número BCD y se encuentra un número diferente de BCD.
SP75	ON si se ejecuta una instrucción BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X6 está ON, el valor BCD formado por las direcciones discretas X0-X3 se carga en el acumulador usando la instrucción LDF. El valor BCD formado por las direcciones discretas C0-C3 se suma al valor en el acumulador usando la instrucción ADDF. El valor en los 4 bits más bajos del acumulador es copiado a Y10-Y13 usando la instrucción OUTF.

DirectSOFT



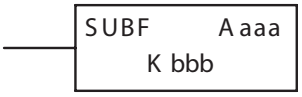
Programador D2-HPP

$ STR	→	G 6	ENT						
SHFT	L ANDST	D 3	F 5	→	A 0	→	E 4	ENT	
SHFT	A 0	D 3	D 3	F 5	→	NEXT	NEXT	NEXT	NEXT
A 0	→	E 4	ENT						
GX OUT	SHFT	F 5	→	B 1	A 0	→	E 4	ENT	

La instrucción Subtract Formatted (SUBF)

DS5	Usado
HPP	Usado

SUBF es una instrucción de 32 bits que resta el valor BCD (Aaaa), que es un rango de bits distintos del valor BCD en el acumulador. El rango (Kbbb) especificado puede ser 1 a 32 bits consecutivos. El resultado se va al acumulador.



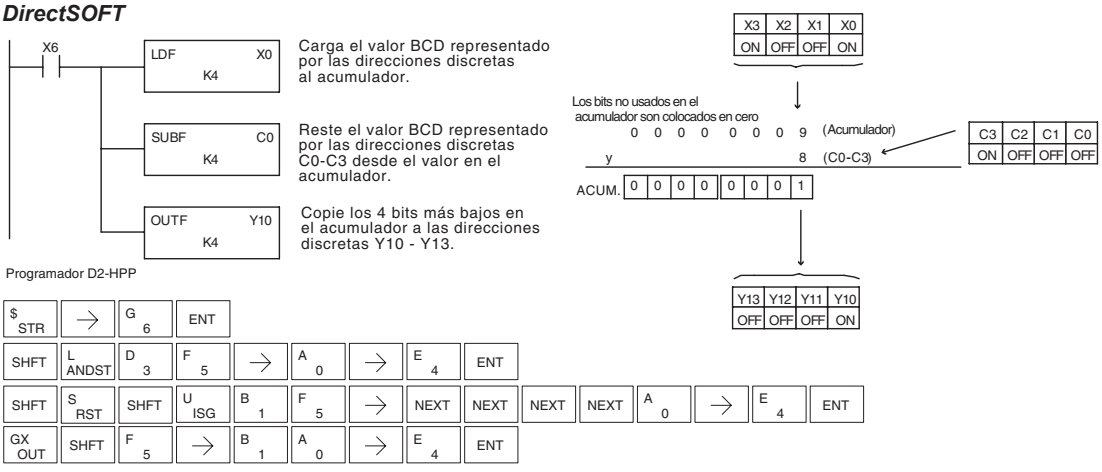
Tipo de operando de datos	Rango del DL06	
..... A	aaa	bbb
Entradas X	0–777	—
Salidas ..... Y	0–777	—
Relevadores de control C	0–1777	—
Bits de etapas ..... S	0–1777	—
Bits de temporizadores ..... T	0–377	—
Bits de contadores ..... CT	0–177	—
Relevadores especiales ..... SP	0-137 320-717	—
Global I/O ..... GX	0-3777	—
Constante ..... K	—	1–32

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP64	ON cuando la instrucción de resta de 16 bits resulta en un "préstamo".
SP65	ON cuando la instrucción de resta de 32 bits resulta en un "préstamo".
SP70	ON cuando el valor en el acumulador es un número negativo.
SP73	On cuando hay una instrucción de suma o resta que resulta en un bit de signo incorrecto.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X6 está ON, el valor BCD formado por las direcciones discretas X0-X3 se carga al acumulador usando la instrucción LDF. El valor BCD formado por las direcciones discretas C0-C3 se resta del valor en el acumulador usando la instrucción SUBF. El valor en los 4 bits más bajos del acumulador es copiado a Y10-Y13 usando la instrucción OUTF.

DirectSOFT

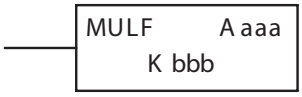


Programador D2-HPP

La instrucción Multiply Formatted (MULF)

DS5	Usado
HPP	Usado

MULF es una instrucción de 16 bits que multiplica el valor BCD en el acumulador por el valor BCD (Aaaa) que es un rango de bits discretos. El rango (Kbbb) especificado puede ser 1 a 16 bits consecutivos. El resultado se va al acumulador.



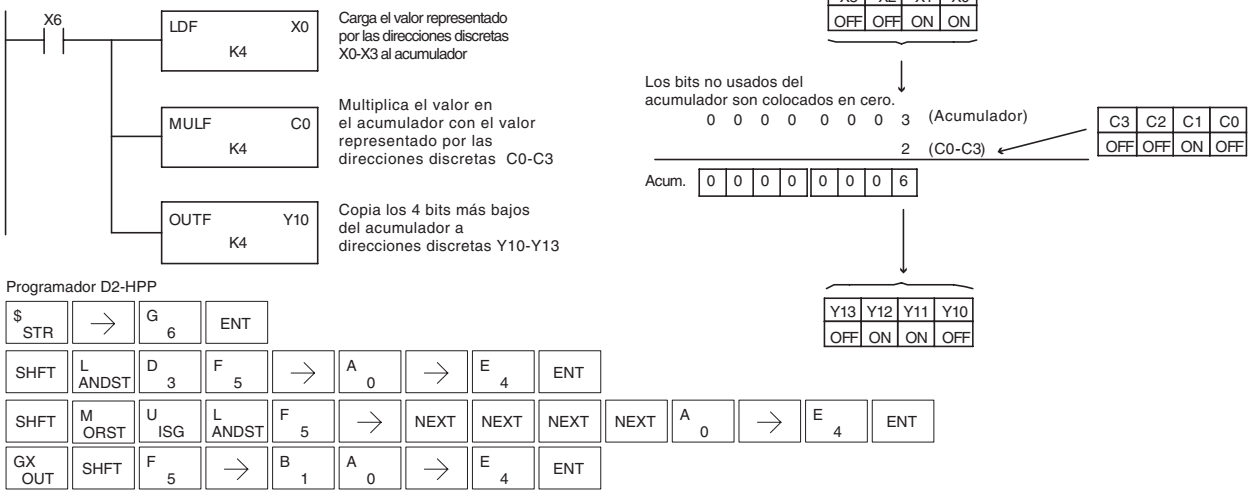
Tipo de operando de datos	Rango del DL06	
..... A	aaa	bbb
Entradas X	0–777	—
Salidas ..... Y	0–777	—
Relevadores de control C	0–1777	—
Bits de etapas ..... S	0–1777	—
Bits de temporizadores ..... T	0–377	—
Bits de contadores ..... CT	0–177	—
Relevadores especiales ..... SP	0-137 320-717	—
Global I/O ..... GX	0-3777	—
Constante ..... K	—	1–16

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es un número negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X6 está ON, el valor formado por las direcciones discretas X0-X3 se carga al acumulador usando la instrucción LDF. El valor formado por las direcciones discretas C0-C3 es multiplicado por el valor en el acumulador usando la instrucción MULF. El valor en los 4 bits más bajos del acumulador es copiado a Y10-Y13 usando la instrucción OUTF.

DirectSOFT



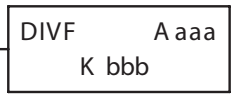
Programador D2-HPP

$ STR	→	G 6	ENT											
SHFT	L ANDST	D 3	F 5	→	A 0	→	E 4	ENT						
SHFT	M ORST	U ISG	L ANDST	F 5	→	NEXT	NEXT	NEXT	NEXT	A 0	→	E 4	ENT	
GX OUT	SHFT	F 5	→	B 1	A 0	→	E 4	ENT						

La instrucción Divide Formatted (DIVF)

DS5	Usado
HPP	Usado

DIVF es una instrucción de 16 bits que divide el valor BCD en el acumulador por el valor BCD (Aaaa), que es un rango de bits discretos. El rango (Kbbb) especificado puede ser 1 a 16 bits consecutivos. La primera parte del cociente se va al acumulador y el residuo se va al primer nivel del Stack.



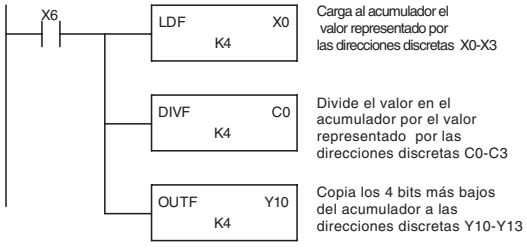
Tipo de operando de datos	Rango del DL06	
. . . . . A	aaa	bbb
Entradas X	0–777	—
Salidas . . . . . Y	0–777	—
Relevadores de control C	0–1777	—
Bits de etapas . . . . . S	0–1777	—
Bits de temporizadores . . . . . T	0–377	—
Bits de contadores . . . . . CT	0–177	—
Relevadores especiales . . . . . SP	0-137 320-717	—
Global I/O. . . . . GX	0-3777	—
Constante . . . . . K	—	1–16

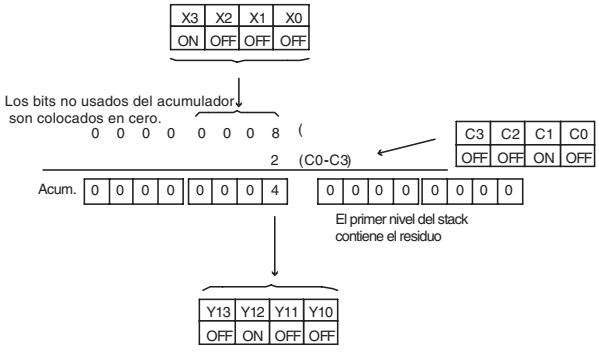
Indicadores	Descripción
SP53	ON cuando el valor del operando en más grande de lo que puede trabajar el acumulador.
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es un número negativo (MSB es 1).
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

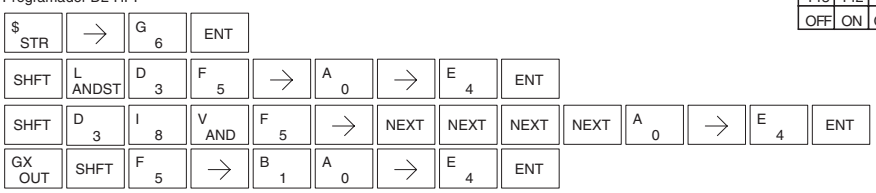
En el ejemplo siguiente, cuándo X6 está ON, se carga el valor formado por las direcciones discretas X0-X3 al acumulador usando la instrucción LDF. El valor en el acumulador es dividido por el valor formado por las direcciones discretas C0-C3 usando la instrucción DIVF. El valor en los 4 bits más bajos del acumulador es copiado a Y10-Y13 usando la instrucción OUTF.

DirectSOFT





Programador D2-HPP



La instrucción Add Top of Stack (ADDS)

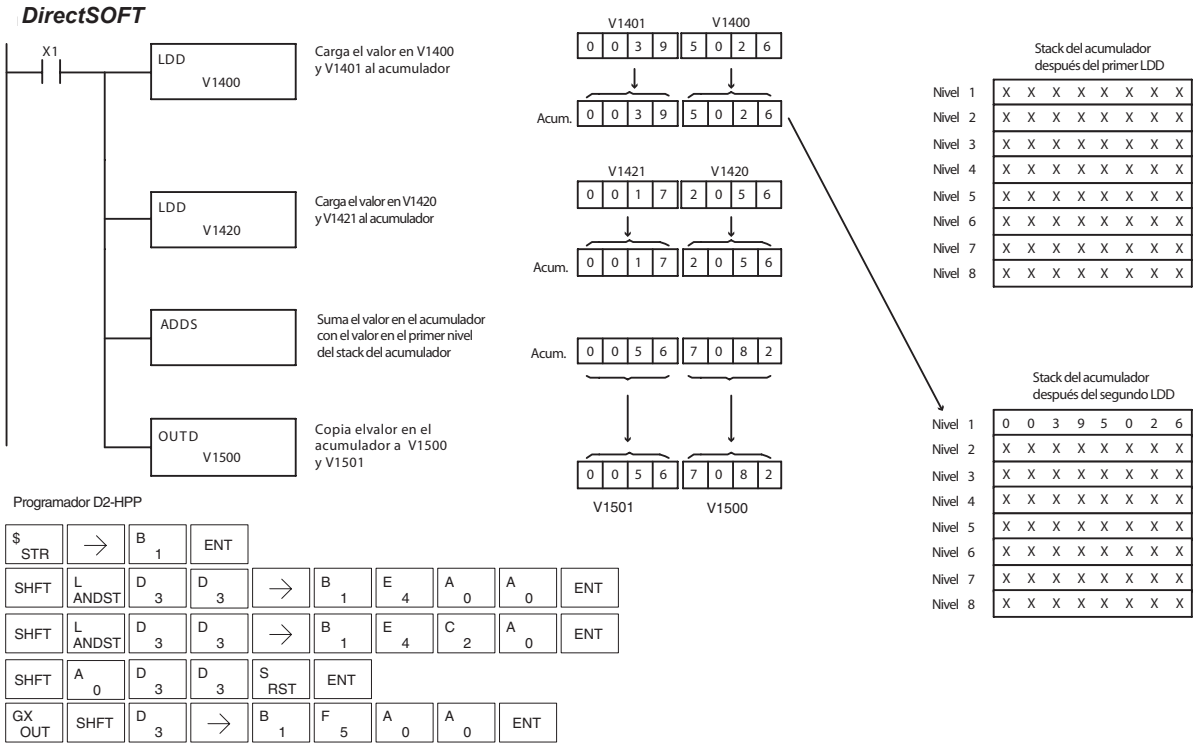
DS5	Usado
HPP	Usado

ADDS es una instrucción de 32 bits que suma el valor BCD en el acumulador con el valor BCD en el primer nivel del Stack del acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador es removido y todos los valores del Stack se mueven un nivel para arriba.

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP66	ON cuando la instrucción de suma de 16 bits da un resultado con “pasa para”.
SP67	ON cuando la instrucción de suma de 32 bits da un resultado con “pasa para”.
SP70	ON cuando el valor en el acumulador es negativo.
SP73	ON cuando una suma o resta con signo resulta con un bit de signo incorrecto.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

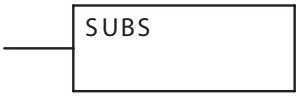
En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El valor en V1420 y V1421 se carga al acumulador usando la instrucción LDD, empujando el valor previamente cargado al acumulador en el Stack del acumulador. El valor en el primer nivel del Stack se suma con el valor en el acumulador usando la instrucción ADDS. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.



La instrucción Subtract Top of Stack (SUBS)

DS5	Usado
HPP	Usado

SUBS es una instrucción de 32 bits que resta el valor BCD en el primer nivel del Stack del acumulador del valor BCD en el acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador es removido y todos los valores del Stack se mueven un nivel hacia arriba.

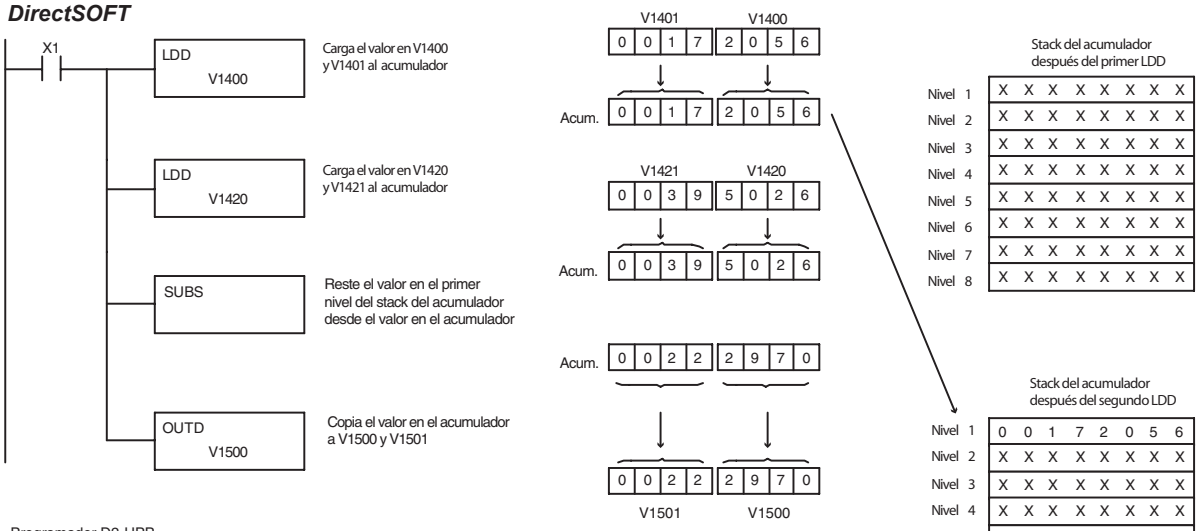


Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP64	ON cuando la instrucción de resta de 16 bits resulta en un "préstamo".
SP65	ON cuando la instrucción de resta de 32 bits resulta en un "préstamo".
SP70	ON cuando el valor en el acumulador es un número negativo.
SP73	ON cuando una suma o resta con signo resulta con un bit de signo incorrecto.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El valor en V1420 y V1421 se carga al acumulador usando la instrucción LDD, empujando el valor previamente cargado en el acumulador en el Stack delacumulador. El valor BCD en el primer nivel del Stack del acumulador se resta del valor BCD en el acumulador usando instrucción SUBS. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.

DirectSOFT



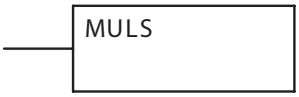
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	C 2	A 0	ENT
SHFT	S RST	SHFT	U ISG	B 1	S RST	ENT			
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT	

La instrucción Multiply Top of Stack (MULS)

DS5	Usado
HPP	Usado

MULS es una instrucción de 16 bits que multiplica un valor de 4 dígitos BCD en el primer nivel del Stack del acumulador por un valor de 4 dígitos BCD en el acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador es removido y todos valores del Stack se mueven un nivel hacia arriba.

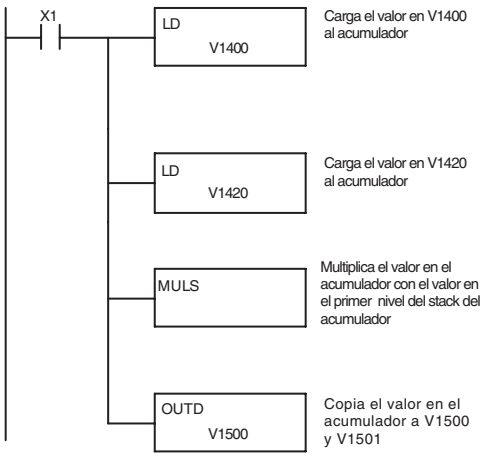


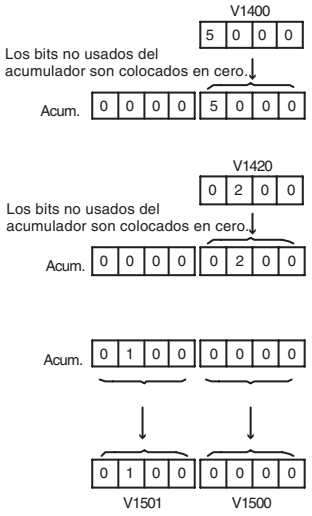
Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es un número negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

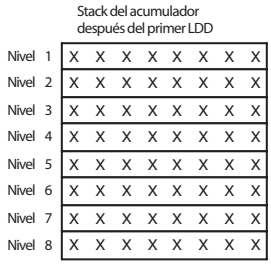
NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

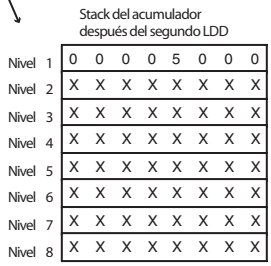
En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V1400 al acumulador usando la instrucción LD. El valor en V1420 se carga al acumulador usando la instrucción LD, empujando el valor previamente cargado en el acumulador al Stack del acumulador. El valor BCD en el primer nivel del Stack del acumulador es multiplicado por el valor BCD en el acumulador usando la instrucción MULS. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.

DirectSOFT









Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	→	B 1	E 4	C 2	A 0	ENT
SHFT	M ORST	U ISG	L ANDST	S RST	ENT			
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT

La instrucción Divide by Top of Stack (DIVS)

DS5	Usado
HPP	Usado

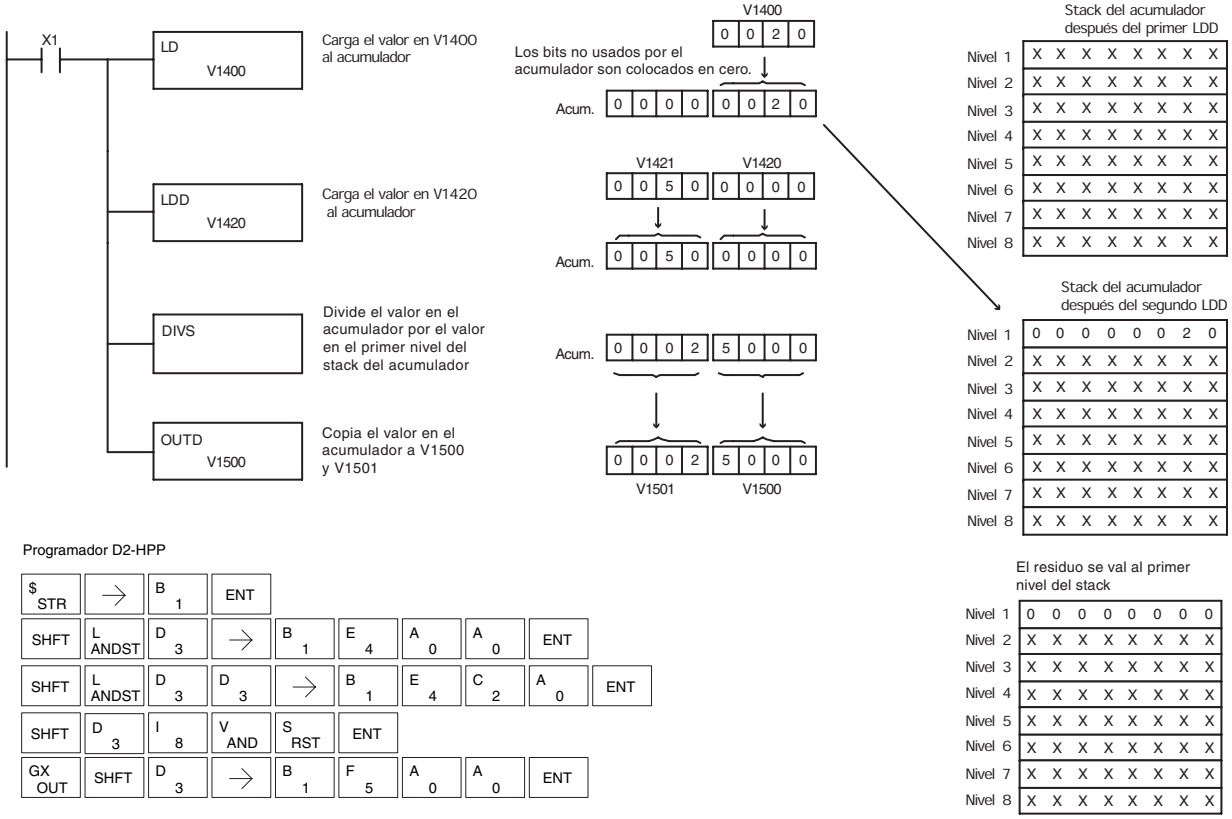
DIVS es una instrucción de 32 bits que divide el valor de 8 dígitos BCD en el acumulador por un valor de 4 dígitos BCD en el primer nivel del Stack del acumulador. El resultado se va al acumulador y el residuo se va al primer nivel del Stack del acumulador.

Indicadores	Descripción
SP53	ON cuando el valor en el operando es más grande de lo que el acumulador puede trabajar.
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es un número negativo.
SP75	ON si se espera un número BCD y se encuentra un número diferente de BCD.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

5

En el ejemplo siguiente, cuándo X1 está ON, la instrucción LD carga el valor que está en V1400 al acumulador. El valor en V1420 se carga al acumulador usando la instrucción LDD, empujando el valor previamente cargado en el acumulador al Stack del acumulador. El valor BCD en el acumulador es dividido por el valor BCD en el primer nivel del Stack del acumulador usando la instrucción DIVS. Luego se copia el valor en el acumulador a V1500 y V1501usando la instrucción OUTD.



Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	C 2	A 0	ENT
SHFT	D 3	I 8	V AND	S RST	ENT			
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT

La instrucción Add Binary Top of Stack (ADDBS)

DS5	Usado
HPP	Usado

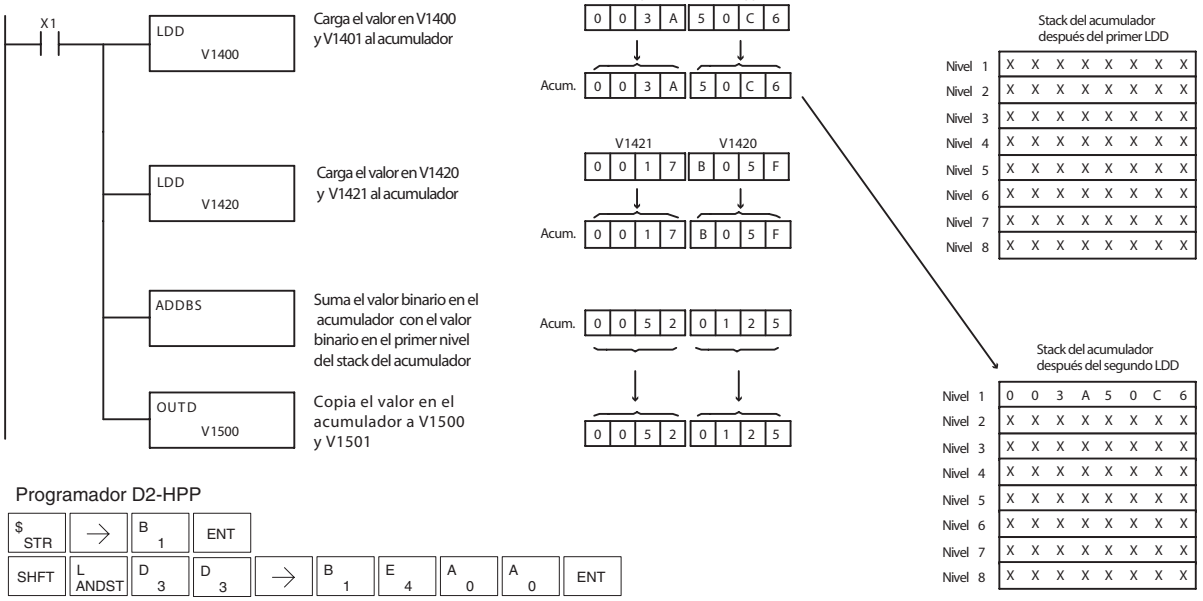
La instrucción ADDBS es una instrucción de 32 bits que suma el valor binario en el acumulador con el valor binario en el primer nivel del Stack del acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador se elimina y todos valores del Stack se mueven un nivel hacia arriba.

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP66	ON cuando la instrucción de suma de 16 bits da un resultado con “pasa para”.
SP67	ON cuando la instrucción de suma de 32 bits da un resultado con “pasa para”.
SP70	ON cuando el valor en el acumulador es negativo.
SP73	ON cuando una suma o resta con signo resulta con un bit de signo incorrecto.

NOTA: Las indicaciones de estado discretas SP son sólo válidas hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El valor en V1420 y V1421 se carga al acumulador usando la instrucción LDD, empujando el valor previamente cargado en el acumulador al Stack del acumulador. El valor binario en el primer nivel del Stack del acumulador se suma con el valor binario en el acumulador usando la instrucción ADDBS. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.Double instrucción.

DirectSOFT



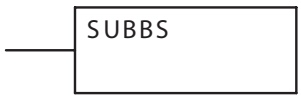
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	C 2	A 0	ENT
SHFT	A 0	D 3	D 3	B 1	S RST	ENT			
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT	

La instrucción Subtract Binary Top of Stack (SUBBS)

DS5	Usado
HPP	Usado

SUBBS es una instrucción de 32 bits que resta el valor binario en el primer nivel del Stack del acumulador del valor binario en el acumulador. El resultado se va al acumulador. El valor en el primer nivel del Stack del acumulador se pierde y todas direcciones del Stack se mueven un nivel hacia arriba.

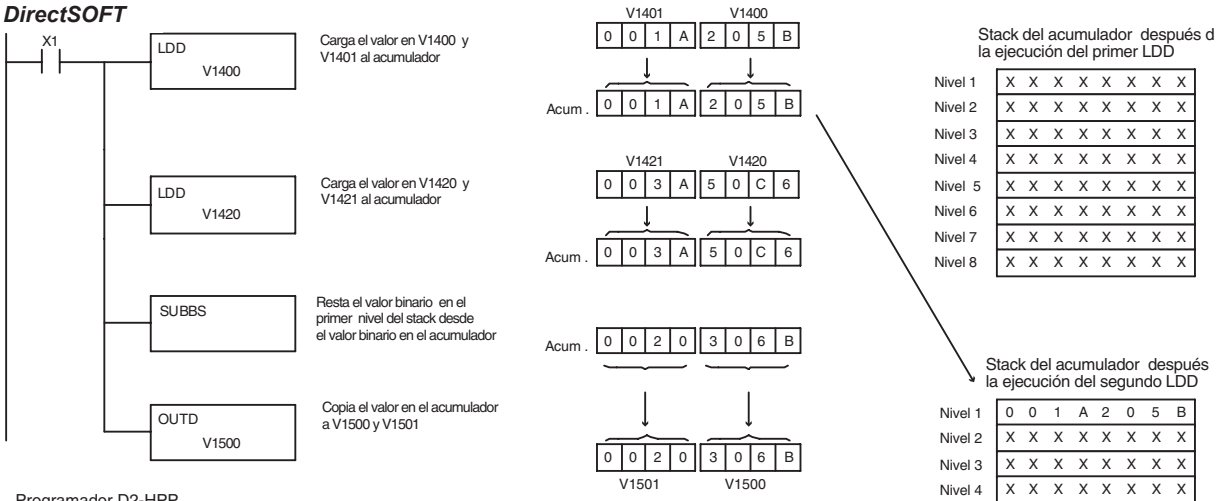


Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP64	ON cuando la instrucción de resta de 16 bits resulta en un "préstamo".
SP65	ON cuando la instrucción de resta de 32 bits resulta en un "préstamo".
SP70	ON en cualquier momento que el valor en el acumulador es negativo.
SP73	ON cuando una suma o resta con signo resulta con un bit de signo incorrecto.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El valor en V1420 y V1421 se carga al acumulador usando la instrucción LDD, empujando el valor previamente cargado en el acumulador en el Stack del acumulador.

El valor binario en el primer nivel del Stack del acumulador se resta del valor binario en el acumulador que usa la instrucción SUBBS. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.



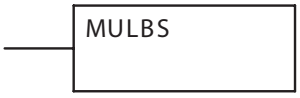
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	C 2	A 0	ENT
SHFT	S RST	SHFT	U ISG	B 1	B 1	S RST	ENT		
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT	

La instrucción Multiply Binary Top of Stack (MULBS)

DS5	Usado
HPP	Usado

MULBS es una instrucción de 16 bits que multiplica el valor binario de 16 bits en el primer nivel del Stack del acumulador por el valor binario de 16 bits en el acumulador. El resultado se va al acumulador y puede ser de 32 bits (8 dígitos máximos.) El valor en el primer nivel del Stack del acumulador se pierde y todas direcciones del Stack se mueven un nivel hacia arriba.



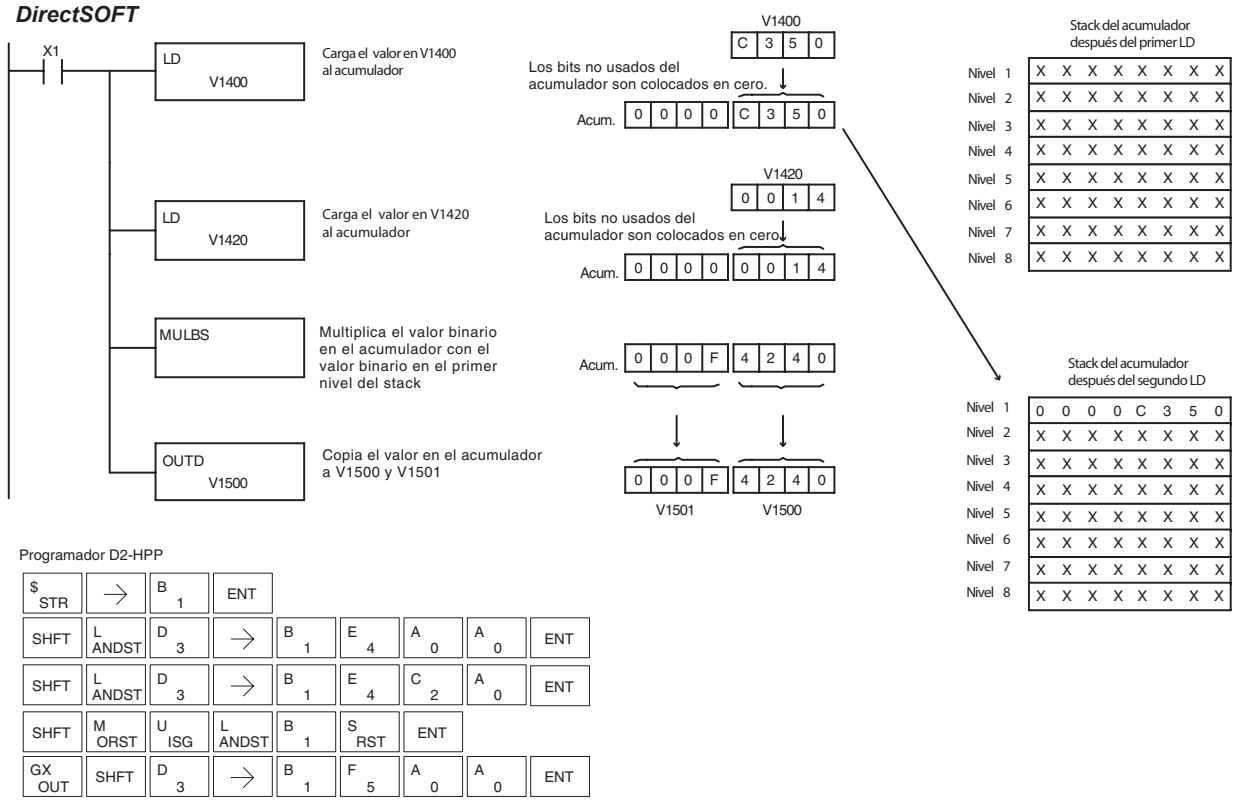
Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	On cuando el valor en el acumulador es negativo.

5

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, la instrucción LD mueve el valor en V1400 al acumulador. El valor en V1420 se carga al acumulador usando la instrucción LD, empujando el valor previamente Cargado en el acumulador al Stack. El valor binario en el primer nivel del Stack es multiplicado por el valor binario en el acumulador usando la instrucción MULBS.

La instrucción OUTD copia el valor en el acumulador a V1500 y V1501.



Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	→	B 1	E 4	C 2	A 0	ENT
SHFT	M ORST	U ISG	L ANDST	B 1	S RST	ENT		
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT

La instrucción Divide Binary by Top OF Stack (DIVBS)

DS5	Usado
HPP	Usado

Esta es una instrucción de 32 bits que divide el valor binario de 32 bits en el acumulador por el valor binario de 16 bits en el primer nivel del stack del acumulador.

El resultado reside en el acumulador y el resto reside en el primer nivel del stack del acumulador.

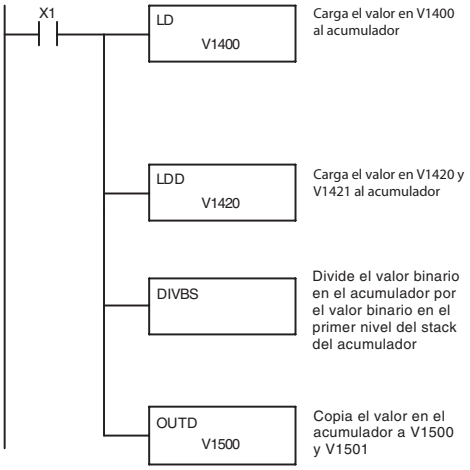
Indicadores	Descripción
SP53	On cuando el valor del operando es más grande que lo que puede aceptar el acumulador
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	On cuando el valor en el acumulador es negativo.

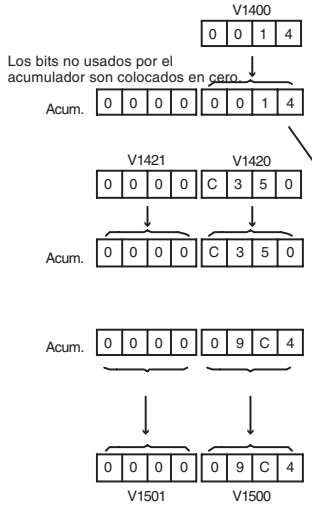
NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

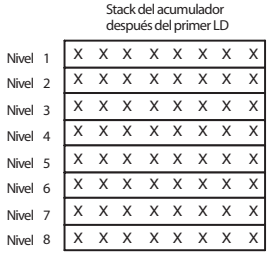
En el ejemplo siguiente, cuando X1 está ON, se carga el valor en V1400 en el acumulador usando la instrucción LD. El valor en V1420 y V1421también se carga en el acumulador usando la instrucción LDD, empujando el valor cargado previamente en el acumulador sobre el stack del acumulador. El valor binario en el acumulador es dividido por el valor binario en el primer nivel del stack del acumulador usando la instrucción DIVBS.

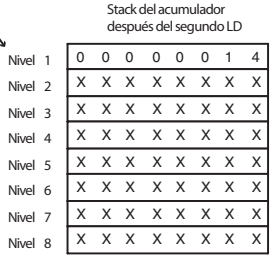
El valor en el acumulador se copia a V1500 y a V1501 usando la instrucción OUTD.

DirectSOFT



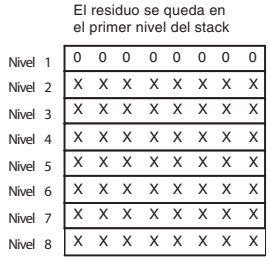






Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	C 2	A 0	ENT
SHFT	D 3	I 8	V AND	B 1	S RST	ENT		
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT



Funciones transcendentales

El PLC DL06 permite ejecutar funciones numéricas especiales para complementar su capacidad de procesar números reales. Las funciones transcendentales incluyen el seno, coseno, y tangente trigonométricos y también sus inversos (arcoseno, arcocoseno y arcotangente). La función de raíz cuadrada también se agrupa con estas otras funciones.

Las instrucciones transcendentales funcionan en un número real localizado en el acumulador (no puede ser BCD o binario). El resultado de la operación reside en el acumulador. La función de raíz cuadrada funciona en el rango completo de números reales positivos. Las funciones de seno, coseno y tangente requieren números expresados en radianes. Usted puede trabajar con ángulos expresados en grados primero convirtiéndolos a radianes con la instrucción radián (RADR) y luego ejecutando la función trigonométrica. Todas las funciones transcendentales utilizan los bits de indicación siguientes:

Indicadores	Descripción
SP53	ON cuando el valor del operando es más grande que de lo que el acumulador puede aceptar.
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP72	ON cuando el valor en el acumulador es un número de punto flotante inválido.
SP73	ON cuando el valor en el acumulador es negativo.
SP75	ON cuando se ejecuta una instrucción de número real y fue encontrado un número que no es real.

La instrucción Sine Real (SINR)

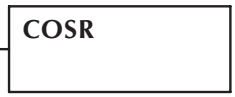
DS5	Usado
HPP	N/A

La instrucción SINR calcula el seno del número real almacenado en el acumulador. El resultado se va al acumulador. El número original y el resultado deben estar en el formato de 32 bits IEEE.

La instrucción Cosine Real (COSR)

DS5	Usado
HPP	N/A

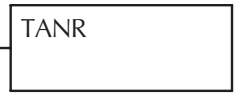
La instrucción COSR calcula el coseno del número real almacenado en el acumulador. El resultado se va al acumulador. El número original y el resultado deben estar en el formato de 32 bits IEEE.



La instrucción Tangent Real (TANR)

DS5	Usado
HPP	N/A

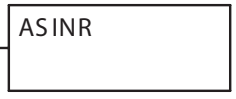
La instrucción TANR calcula la tangente del número real almacenado en el acumulador. El resultado se va al acumulador. El número original y el resultado deben estar en el formato de 32 bits IEEE.



La instrucción Arc Sine Real (ASINR)

DS5	Usado
HPP	N/A

La instrucción ASINR calcula el arcoseno del número real almacenado en el acumulador. El resultado se va al acumulador. El número original y el resultado deben estar en el formato de 32 bits IEEE.



La instrucción Arc Cosine Real (ACOSR)

DS5	Usado
HPP	N/A

La instrucción ACOSR calcula el arcocoseno del número real almacenado en el acumulador. El resultado se va al acumulador. El número original y el resultado deben estar en el formato de 32 bits IEEE.

La instrucción Arc Tangent Real (ATANR)

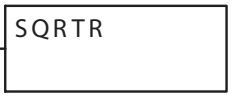
DS5	Usado
HPP	N/A

La instrucción ATANR calcula el arcotangente del número real almacenado en el acumulador. El resultado se va al acumulador. El número original y el resultado deben estar en el formato de 32 bits IEEE.

La instrucción Square Root Real (SQRTR)

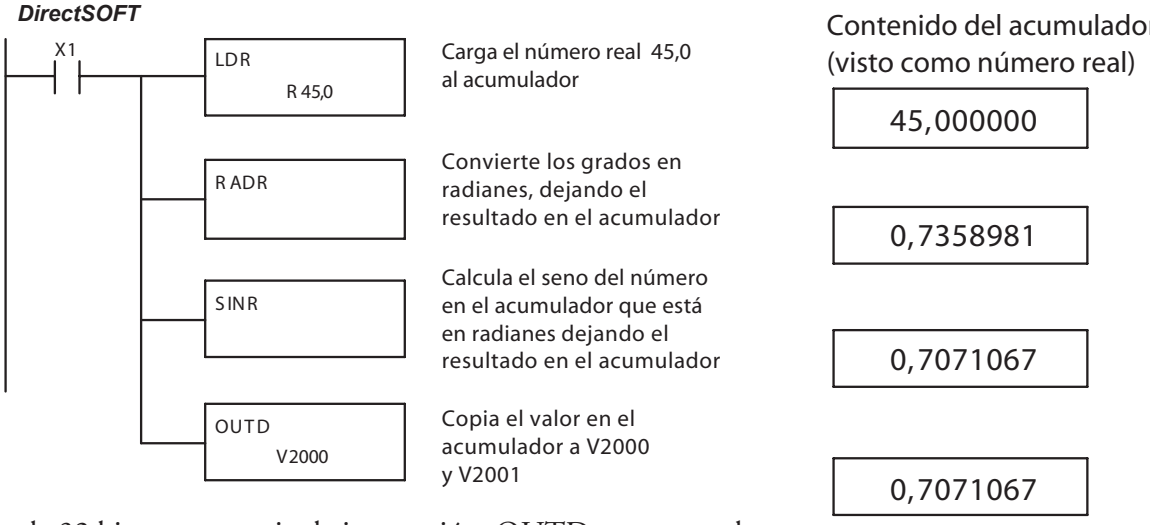
DS5	Usado
HPP	N/A

La instrucción SQRTR calcula la raíz cuadrada del número real almacenado en el acumulador. El resultado se va al acumulador. El número original y el resultado deben estar en el formato de 32 bits IEEE.



NOTA: La función raíz cuadrada puede ser útil en varias situaciones. Sin embargo, si se trata de hacer la función de extracción de raíz para un instrumento medidor de flujo del tipo de placa orificio como PV para un lazo de PID, note que el lazo de PID ya tiene una función de extracción de raíz cuadrada incluida.

El ejemplo siguiente toma el **seno** de 45 grados. Ya que estas funciones trascendentales operan sólo con números reales, hacemos una instrucción LDR (Carga real) con el operando 45,0. Las funciones trigonométricas operan sólo con radianes, así que se debe convertir los grados a radianes usando la instrucción RADR. Después de usar la instrucción SINR (Seno Real), se usa la instrucción OUTD para mover el resultado del acumulador a la memoria V. El resultado es de 32 bits, y se necesita la instrucción OUTD para moverlo.



NOTA: El D2-HPP no permite el uso de números reales con la conversión automática al formato de 32 bit de IEEE. Usted debe utilizar DirectSOFT para entrar números reales, usando la instrucción LDR.

Instrucciones de operación con bits

La instrucción Sum (SUM)

DS5	Usado
HPP	Usado

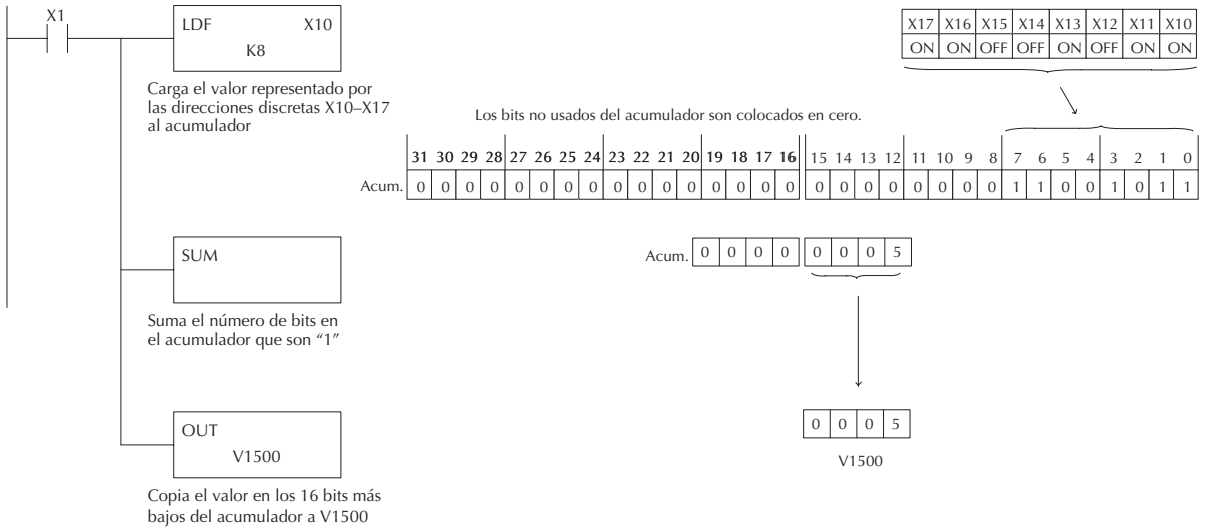
La instrucción SUM cuenta el número de bits que son "1" en el acumulador. El resultado en hexadecimal se va al acumulador.

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor formado por las direcciones discretas X10-X17 al acumulador usando la instrucción LDF. Luego es contado el número de bits del acumulador que son "1s", usando la instrucción SUM. El valor en el acumulador es copiado a V1500 usando la instrucción OUT.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	F 5	→	B 1	A 0	→	I 8	ENT
SHFT	S RST	SHFT	U ISG	M ORST	→	ENT			
GX OUT	→	PREV	PREV	PREV	B 1	F 5	A 0	A 0	ENT

La instrucción Shift Left (SHFL)

DS5	Usado
HPP	Usado

SHFL es una instrucción de 32 bits que desplaza los bits en el acumulador un número especificado de lugares (Aaaa) a la izquierda, es decir, en la dirección desde el bit menos significativo al más significativo. Las posiciones vacías se llenan con ceros y los bits que son desplazados fuera del acumulador se pierden.

```
SHFL
   A aaa
```

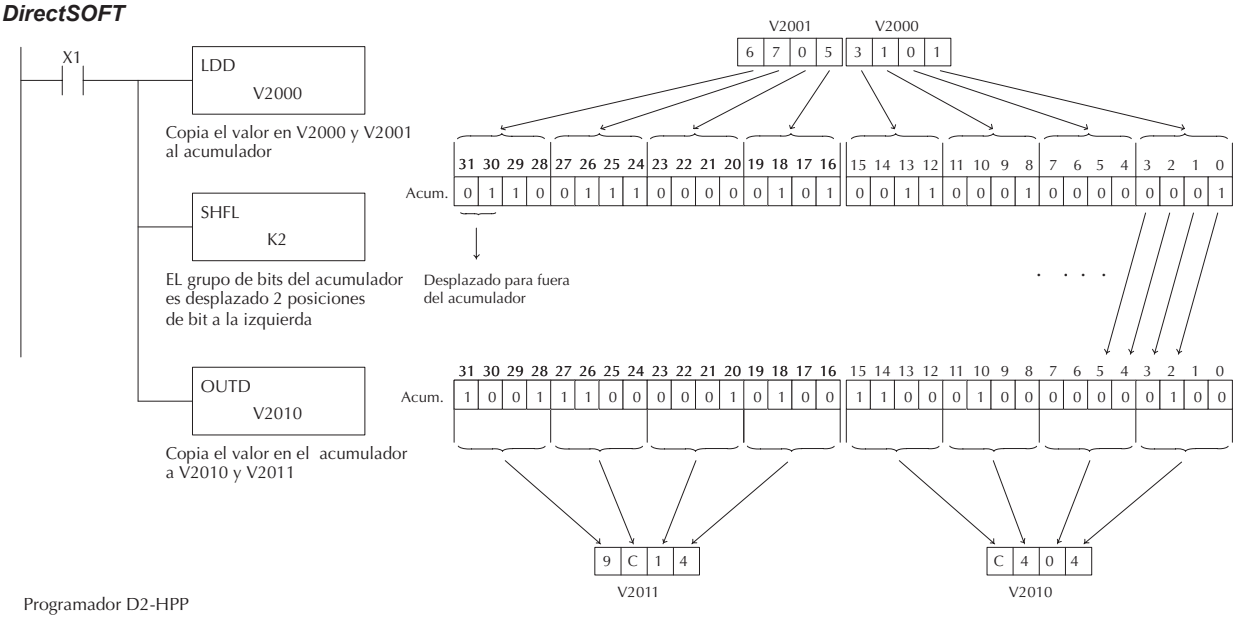
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Constante . . . . . K	1-32

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga al acumulador usando la instrucción LDD. El conjunto de bits en el acumulador se desplaza 2 bits a la izquierda usando la instrucción SHFL. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT



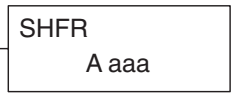
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT
SHFT	S RST	SHFT	H 7	F 5	L ANDST	→	C 2	ENT	
GX OUT	SHFT	D 3	→	C 2	A 0	B 1	A 0	ENT	

La instrucción Shift Right (SHFR)

DS5	Usado
HPP	Usado

SHFR es una instrucción de 32 bits que desplaza los bits en el acumulador un número especificado de lugares (Aaaa) a la derecha, es decir, en la dirección desde el bit más significativo al menos significativo. Las posiciones vacías se llenan con ceros y los bits que son desplazados fuera del acumulador se pierden.



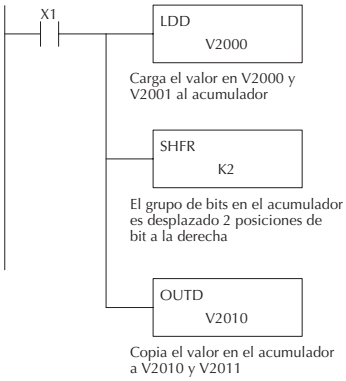
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Constante . . . . . K	1-32

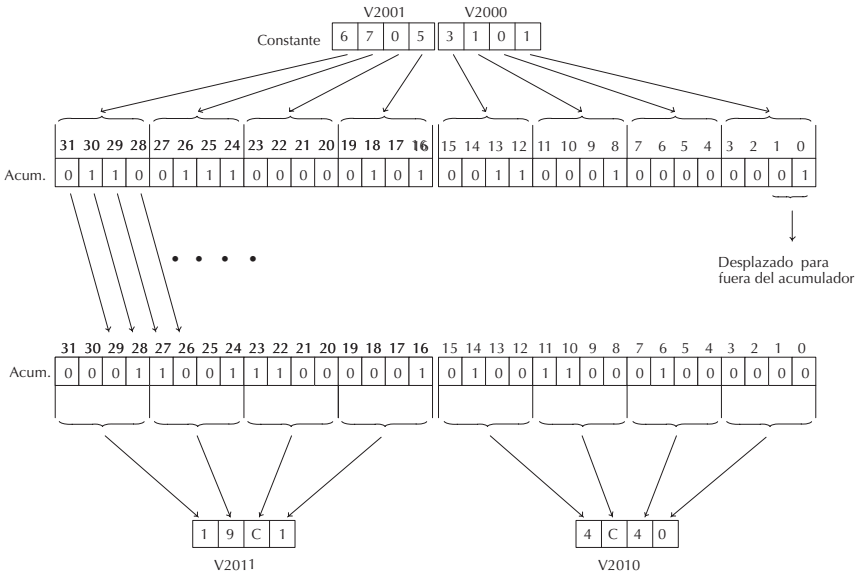
Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga al acumulador usando la instrucción LDD. El conjunto de bits en el acumulador se cambia de 2 bits a la derecha usando la instrucción SHFR. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT





Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT
SHFT	S RST	SHFT	H 7	F 5	R ORN	→	C 2	ENT	
GX OUT	SHFT	D 3	→	C 2	A 0	B 1	A 0	ENT	

DS5	Usado
HPP	Usado

ROTL es una instrucción de 32 bits que desplaza los bits en el acumulador un número (Aaaa) especificado de lugares a la izquierda y los que se perderían se van al extremo derecho,es decir, los bits se desplazan en la dirección desde el bit menos significativo al más signficativo.

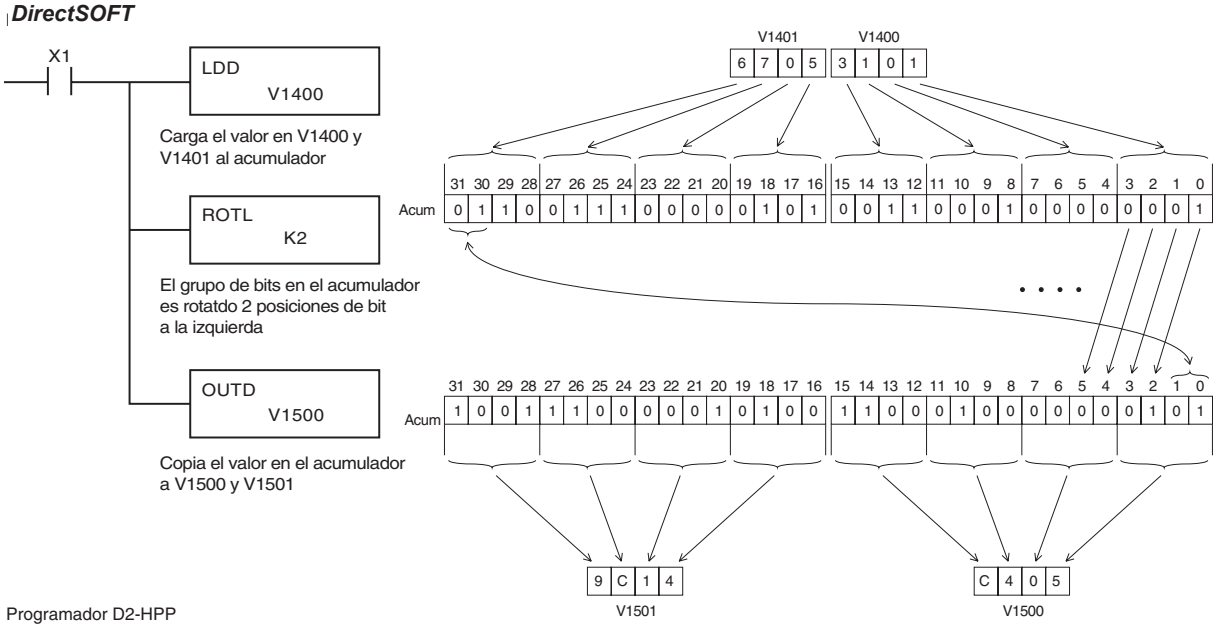
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Constante . . . . . K	1-32

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El conjunto de bits en el acumulador se mueve 2 bits a la izquierda usando la instrucción ROTL. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT



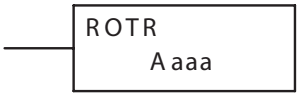
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	R ORN	O INST#	T MLR	L ANDST	→	C 2	ENT		
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT	

La instrucción Rotate Right (ROTR)

DS5	Usado
HPP	Usado

ROTR es una instrucción de 32 bits que desplaza los bits en el acumulador un número (Aaaa) especificado de lugares a la derechaes decir, los bits de desplazan en la dirección desde el bit más significativo al menos significativo.



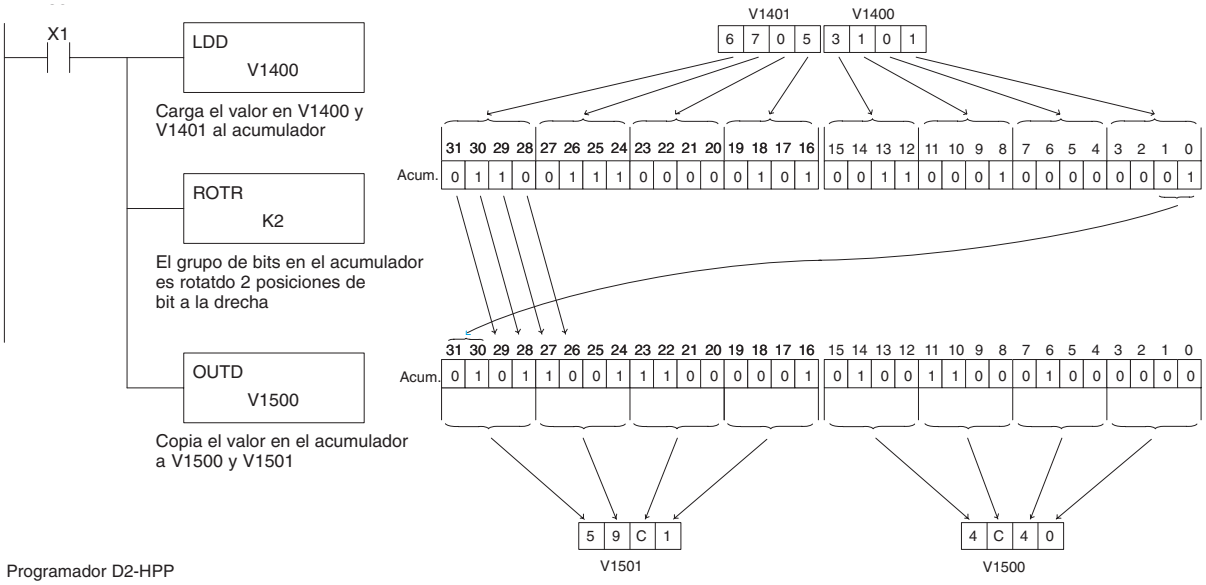
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Constante . . . . . K	1-32

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

5

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. El conjunto de bits en el acumulador desplaza 2 bits a la derecha usando la instrucción ROTR. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	R ORN	O INST#	T MLR	R ORN	→	C 2	ENT		
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT	

La instrucción Encode (ENCO)

DS5	Usado
HPP	Usado

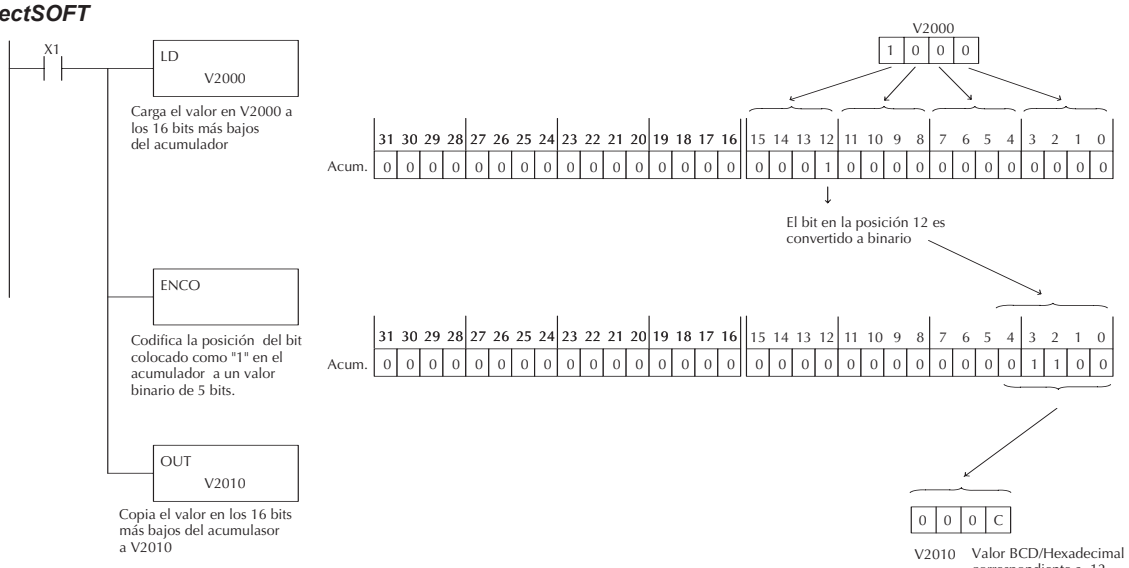
La instrucción ENCO es una instrucción de 16 bits que codifica la posición del bit en el acumulador que tiene un valor de 1 y retorna la representación binaria apropiada de 5 bits. Si el bit más significativo está en 1 (Bit 31), la instrucción ENCO colocaría el valor 1F hexadecimal (decimal 31) en el acumulador. Si el valor a ser codificado es 0000 o 0001, la instrucción colocará un cero en el acumulador. Si el valor a ser codificado tiene más de un conjunto de posiciones de bit en "1", el bit menos significativo con un "1" será codificado y SP53 se hará ON.

Indicadores	Descripción
SP53	ON cuando el valor del operando es más grande de lo que puede procesar el acumulador.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor en V2000 al acumulador usando la instrucción LD. La posición del bit que está en "1" (posición 12) en el acumulador es codificada como el valor binario correspondiente usando la instrucción ENCO. El valor en los 16 bits más bajos del acumulador es copiado a V2010 usando la instrucción OUT.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	C 2	A 0	A 0	A 0	ENT
SHFT	E 4	N TMR	C 2	O INST#	ENT			
GX OUT	→	SHFT	V AND	C 2	A 0	B 1	A 0	ENT

La instrucción Decode (DECO)

DS5	Usado
HPP	Usado

La instrucción DECO decodifica un valor binario de 5 bits en el rango de 0-31 (0-1F hexadecimal) en el acumulador poniendo la posición apropiada del bit en "1".

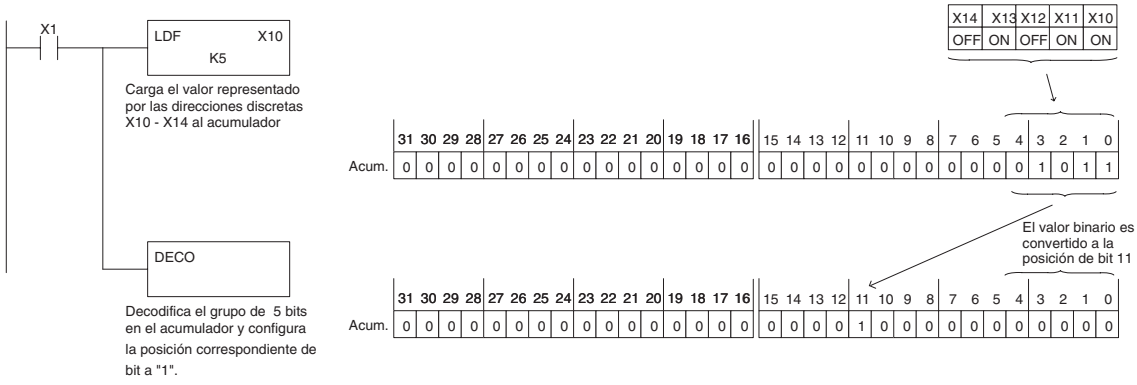
Si el acumulador contiene el valor F (hexadecimal), el bit 15 será colocado como "1" en el acumulador. Los demás bits serán 0.

Si el valor para ser decodificado es más que 31, el número es dividido por 32 hasta que el valor sea menor que 32 y entonces el valor se decodifica.

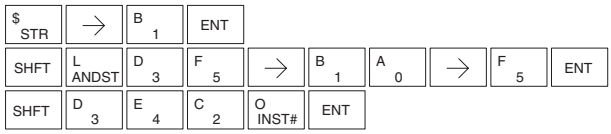
En el ejemplo siguiente cuando X1 está ON, el valor formado por las direcciones discretas X10-X14 se carga al acumulador usando la instrucción LDF.

Los cinco bits en el acumulador son decodificados poniendo la posición correspondiente de bit en "1" usando la instrucción DECO.

DirectSOFT



Programador D2-HPP



Instrucciones de conversión de formatos numéricos

La instrucción Binary (BIN)

DS5	Usado
HPP	Usado

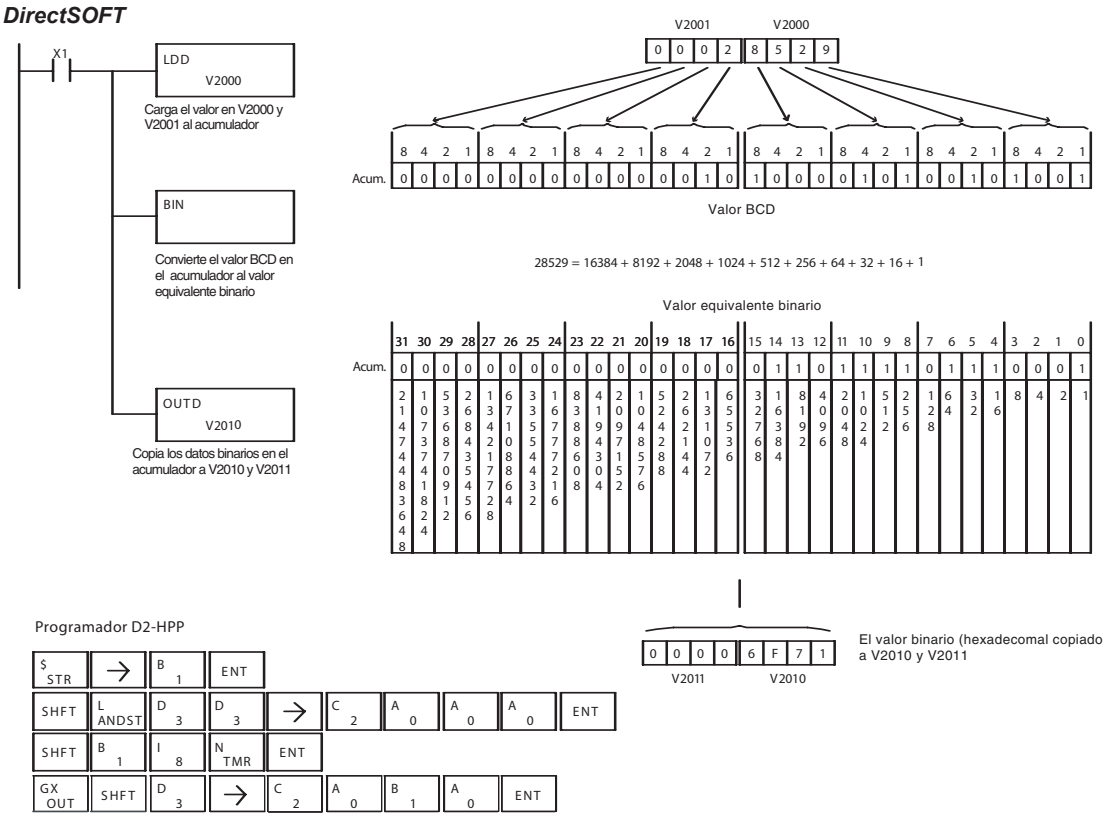
La instrucción BIN convierte un valor BCD en el acumulador al valor binario equivalente(o decimal, como se llama en *Direct*SOFT). El resultado se va al acumulador.

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP75	On cuando se ejecuta una instrucción BCD y se encuentra un número que NO es BCD.

5

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga al acumulador usando la instrucción LDD. El valor BCD en el acumulador es convertido a binario (hexadecimal) usando la instrucción BIN. El valor binario en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

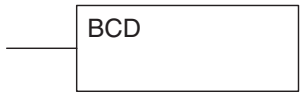
(El programador D2-HPP mostrará el valor binario en V2010 y V2011 como un valor hexadecimal)



La instrucción Binary Coded Decimal (BCD)

DS5	Usado
HPP	Usado

La instrucción BCD convierte un valor binario en el acumulador al valor equivalente BCD. El resultado se va al acumulador.

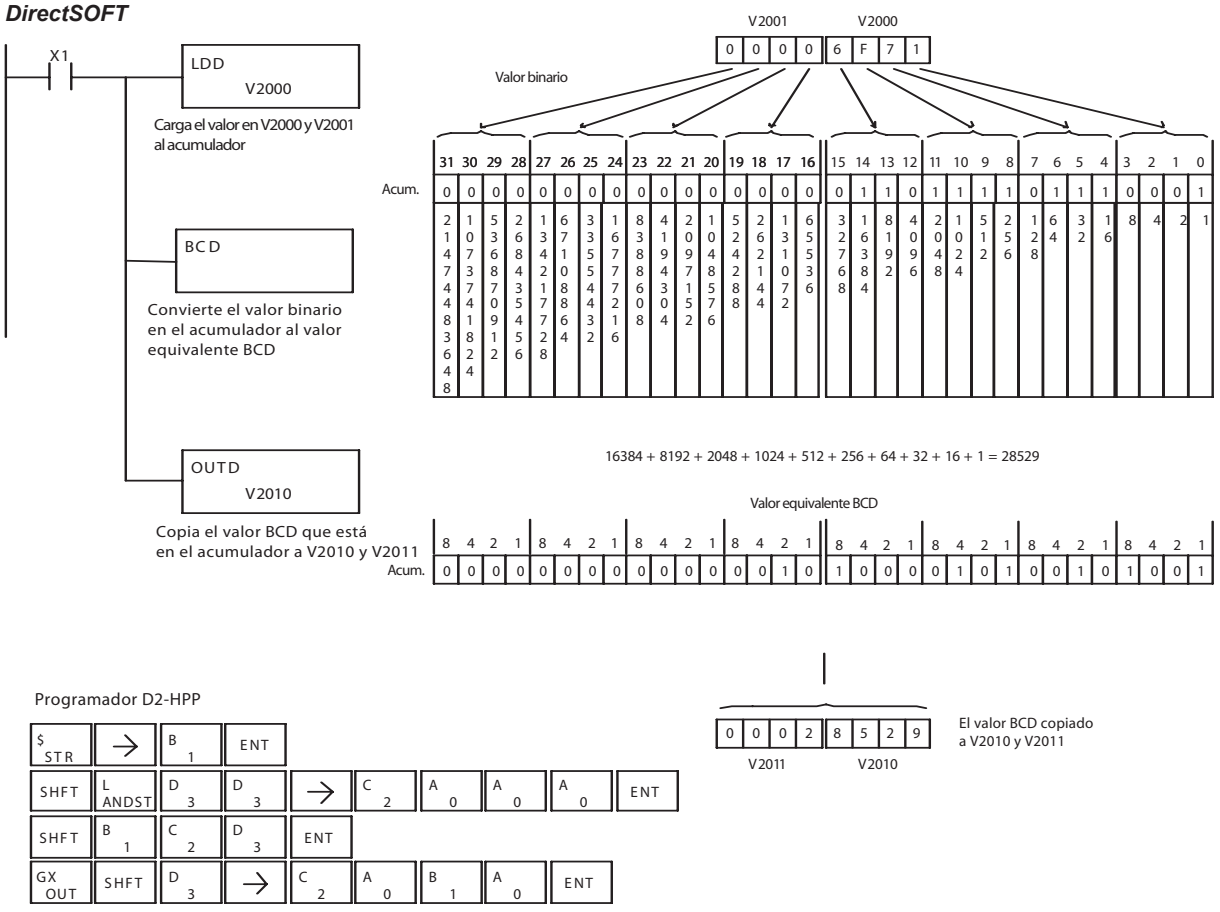


Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

En el ejemplo siguiente, cuándo X1 está ON, el valor binario (hexadecimal) en V2000 y V2001 se carga al acumulador usando la instrucción LDD. El valor binario en el acumulador es convertido al valor equivalente BCD usando la instrucción BCD. El valor BCD en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

5

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT
SHFT	B 1	C 2	D 3	ENT					
GX OUT	SHFT	D 3	→	C 2	A 0	B 1	A 0	ENT	

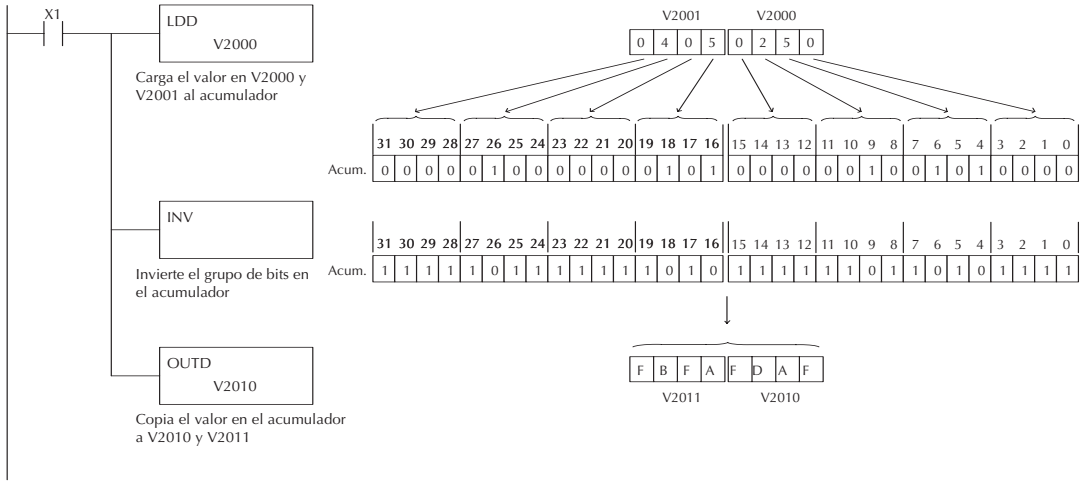
La instrucción Invert (INV)

DS5	Usado
HPP	Usado

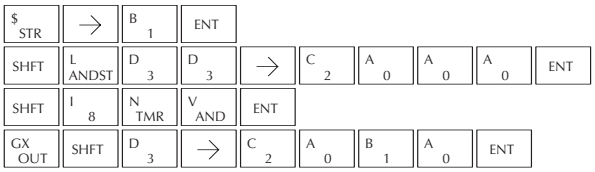
La instrucción INV invierte o toma el complemento de uno del valor de 32 bits en el acumulador. El resultado se va al acumulador. Esto es, cada bit que es cero pasa a ser uno y cada bit que es uno pasa a ser cero, en la misma posición de la palabra.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V2000 y V2001 se carga al acumulador usando la instrucción LDD. El valor en el acumulador se invierte usando la instrucción INV. El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

DirectSOFT



Programador D2-HPP



La instrucción Ten's Complement (BCDCPL)

DS5	Usado
HPP	Usado

La instrucción BCDCPL toma el complemento de 10's (BCD) del acumulador con 8 dígitos. El resultado se va al acumulador. El cálculo para esta instrucción es:

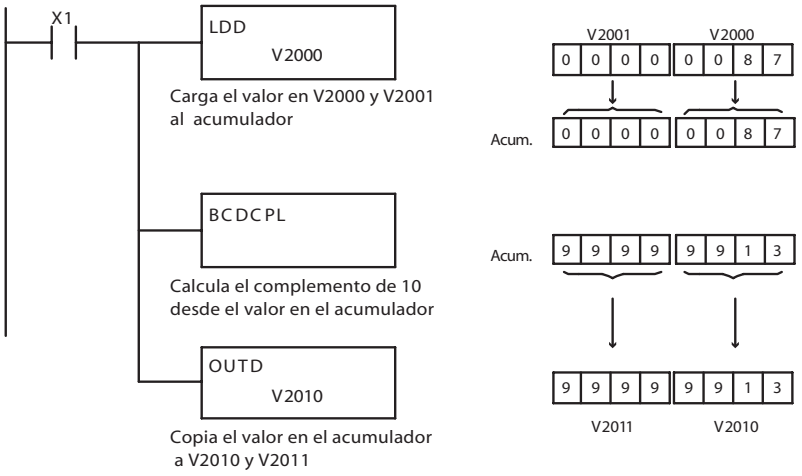
100000000
−acumulador
valor del complemento de 10

En el ejemplo siguiente cuando X1 está ON, el valor en V2000 y V2001 se carga al acumulador.

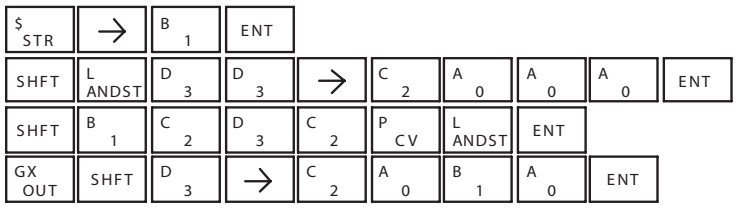
Se calcula entonces el complemento de 10 del acumulador con los 8 dígitos usando la instrucción BCDCPL.

El valor en el acumulador es copiado a V2010 y V2011 usando la instrucción OUTD.

DirectSOFT



Programador D2-HPP



La instrucción Binary to Real Conversion (BTOR)

DS5	Usado
HPP	Usado

La instrucción BTOR convierte un valor binario en el acumulador al formato de número real equivalente (punto flotante). El resultado se va al acumulador. El número binario y el número real pueden usar los 32 bits del acumulador.

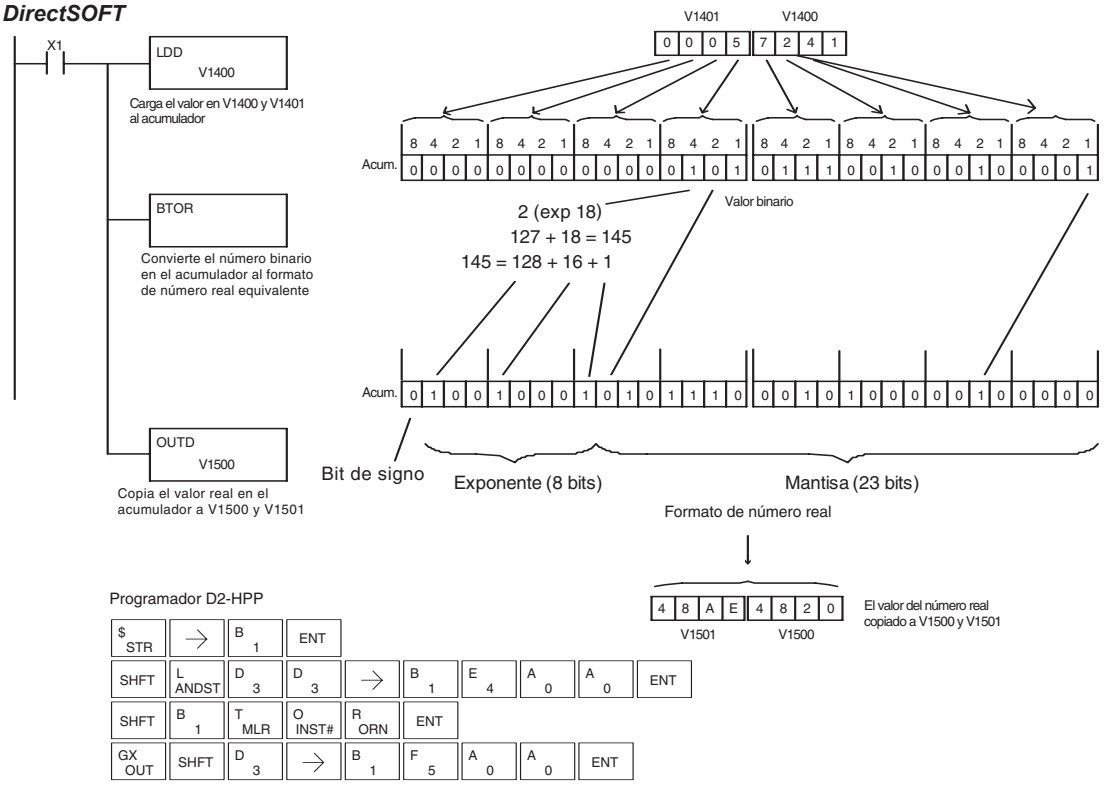
*NOTA: Esta instrucción sólo trabaja con valores **binarios**. No trabajará con valores decimales con signo.*

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. La instrucción BTOR convierte el valor binario en el acumulador al formato real equivalente del número. El peso binario del MSB (El bit más significativo) es convertido al exponente real del número sumándolo a 127 (decimal). Luego los bits restantes son copiados a la mantisa como es mostrado en el diagrama. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD.

El programador D2-HPP mostraría el valor binario en V1500 y V1501 como un valor hexadecimal.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	B 1	T MLR	O INST#	R ORN	ENT			
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT

La instrucción Real to Binary Conversion (RTOB)

DS5	Usado
HPP	Usado

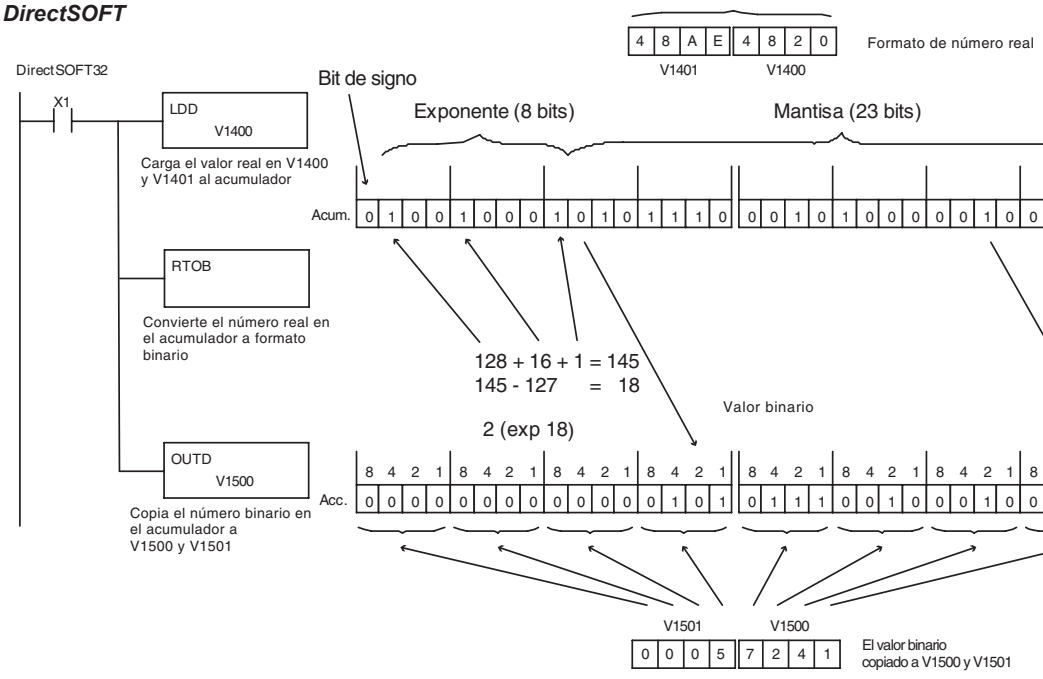
La instrucción RTOB convierte un número real en el acumulador a un valor binario. El resultado se va al acumulador. El número binario y el número real pueden usar los 32 bits del acumulador. El valor real es truncado a un número entero.

NOTA$_1$: La porción decimal del resultado será truncada.
NOTA$_2$: si el número real es negativo, se torna en un valor decimal con signo.

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP72	ON cuando el valor en el acumulador es un número de punto flotante inválido.
SP73	ON cuando una suma o resta con signo resulta en un bit de signo incorrecto.
SP75	ON cuando un número no puede ser convertido a binario.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 y V1401 se carga al acumulador usando la instrucción LDD. La instrucción de RTOB convierte el valor real en el acumulador al formato equivalente de numeración binaria. El valor en el acumulador es copiado a V1500 y V1501 usando la instrucción OUTD. El programador D2-HPP mostraría el valor binario en V1500 y V1501 como un valor hexadecimal.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	D 3	→	B 1	E 4	A 0	A 0	ENT
SHFT	R ORN	T MLR	O INST#	B 1	ENT			
GX OUT	SHFT	D 3	→	B 1	F 5	A 0	A 0	ENT

La instrucción Radian Real Conversion (RADR)

DS5	Usado
HPP	N/A

RADR convierte el valor real del grado almacenado en el acumulador al número real equivalente en radianes. El resultado se va al acumulador.

La instrucción Degree Real Conversion (DEGR)

DS32	Usado
HPP	N/A

La instrucción DEGR convierte el valor real de radián almacenado en el acumulador al número real equivalente en grados. El resultado se va al acumulador.

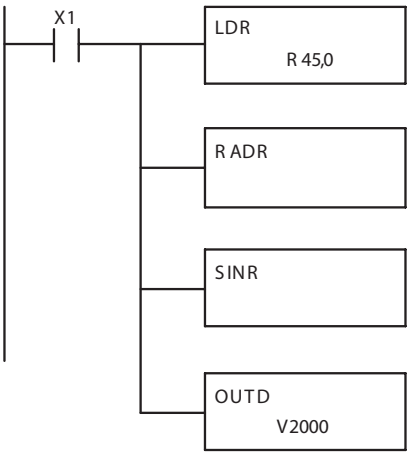
Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.
SP72	ON cuando el valor en el acumulador es un número de punto flotante inválido.
SP73	ON cuando una suma o resta con signo resulta en un bit de signo incorrecto.
SP75	ON cuando un número no puede ser convertido a formato binario (antes era SP74)

Las dos instrucciones descritas encima convierten números reales en el acumulador del formato de grado al formato de radián, y vice-versa. En el formato de grado, un círculo contiene 360 grados. En el formato de radián, un círculo contiene aproximadamente 6.28 radianes (2xπ). Estos convierten números positivos y negativos reales y ángulos de más de un círculo completo. Estas funciones son muy útiles cuando son combinadas con las funciones trigonométricas trascendentales (vea la sección en instrucciones aritméticas).

NOTA: El programador D2-HPP no permite entrar números reales con conversión automática al formato de 32 bits IEEE. Usted debe usar DirectSOFT para entrar números reales, usando la instrucción LDR.

El ejemplo siguiente calcula el seno de 45 grados. Ya que las funciones trascendentales operan sólo con números reales, se hace un LDR (Cargue real) 45,0. Las funciones trigonometricas operan sólo en radianes, así que debemos convertir los grados a radianes usando la instrucción RADR. Después de usar la instrucción SINR (Seno Real), se usa la instrucción OUTD para copiar el resultado del acumulador a la memoria V. El resultado es de 32 bits de ancho, requiriendo el OUTD para moverlo.

DirectSOFT



Carga el número real 45,0 al acumulador

Convierte los grados en radianes, dejando el resultado en el acumulador

Calcula el seno del número en el acumulador que está en radianes dejando el resultado en el acumulador

Copia el valor en el acumulador a V2000 y V2001

Contenido del acumulador (visto como número real)

45,000000

0,7358981

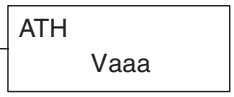
0,7071067

0,7071067

La instrucción ASCII a HEX (ATH)

DS5	Usado
HPP	N/A

La instrucción ATH convierte una tabla de valores de ASCII a una tabla de valores hexadecimales. Los valores de ASCII son dos dígitos y sus equivalentes hexadecimales solamente son un dígito. Esto significa que una tabla ASCII de cuatro direcciones de memoria V sólo requiere dos direcciones de memoria V para la tabla equivalente hexadecimal.



Los parámetros de la función son cargados en el Stack del acumulador y en el acumulador por dos instrucciones adicionales.

Abajo están listados los pasos necesarios de programar una función de tabla de ASCII a hexadecimal.

El ejemplo en la página siguiente muestra un programa para la función de conversión ASCII a hexadecimal.

Paso 1: — Cargue el número de direcciones de memoria V para la tabla ASCII en el primer nivel del Stack del acumulador.

Paso 2: — Cargue la dirección de memoria V de inicio para la tabla ASCII en el acumulador. Este parámetro debe ser un valor en hexadecimal.

Paso 3: — Especifique la dirección de memoria V (Vaaa) de inicio para la tabla hexadecimal en la instrucción de ATH.

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria

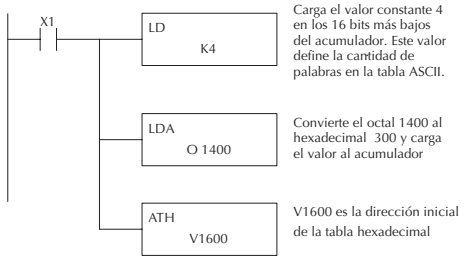
Indicadores	Descripción
SP53	ON cuando el valor del operando es más grande de lo que puede procesar el acumulador.

Sugerencia: — Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

En el ejemplo en la página siguiente, cuándo X1 está ON la constantee (K4) es cargada en el acumulador usando la instrucción LD y se colocará en el primer nivel del Stack del acumulador cuando se ejecuta la próxima instrucción LD. La localización de inicio para la tabla ASCII (V1400) es cargada en el acumulador usando la instrucción LDA. La dirección inicial para la tabla hexadecimal (V1600) es especificada en la instrucción ATH. La tabla de abajo lista valores válidos ASCII para la conversión ATH.

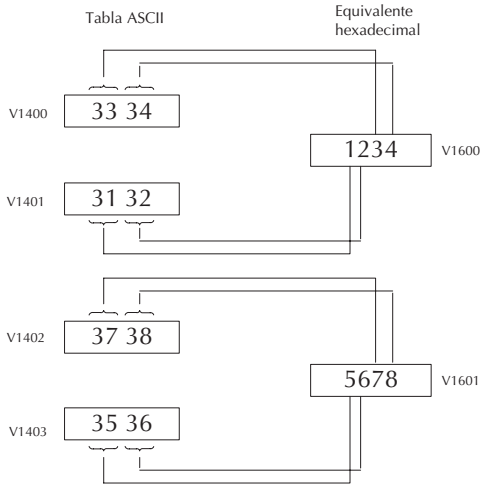
Valores ASCII válidos para la conversión ATH			
Valor ASCII	Valor hexadecimal	Valor ASCII	Valor hexadecimal
30	0	38	8
31	1	39	9
32	2	41	A
33	3	42	B
34	4	43	C
35	5	44	D
36	6	45	E
37	7	46	F

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	→	PREV	E 4	ENT			
SHFT	L ANDST	D 3	A 0	→	B 1	E 4	A 0	A 0	ENT
SHFT	A 0	T MLR	H 7	→	B 1	G 6	A 0	A 0	ENT

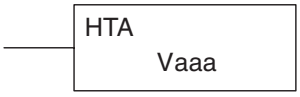


La instrucción HEX a ASCII (HTA)

DS5	Usado
HPP	N/A

La instrucción HTA convierte una tabla de valores hexadecimales a una tabla especificada de valores ASCII. Los valores hexadecimales son de un dígito y sus equivalentes ASCII son de dos dígitos.

Esto significa que una tabla hexadecimal de dos direcciones de memoria V requeriría cuatro direcciones de memoria V para la tabla equivalente de ASCII. Los parámetros de la función son cargados en el Stack del acumulador y el acumulador por dos instrucciones adicionales. Abajo están listados los pasos necesarios para programar la función de transformación de la tabla hexadecimal a ASCII. El ejemplo en la página siguiente muestra un programa para la función de conversión hexadecimal a ASCII.



Paso 1: Cargue el número de direcciones de memoria V en la tabla hexadecimal al primer nivel del Stack del acumulador.

Paso 2: Cargue la localización de la memoria V de inicio para la tabla hexadecimal al acumulador. Este parámetro debe ser un valor hexadecimal.

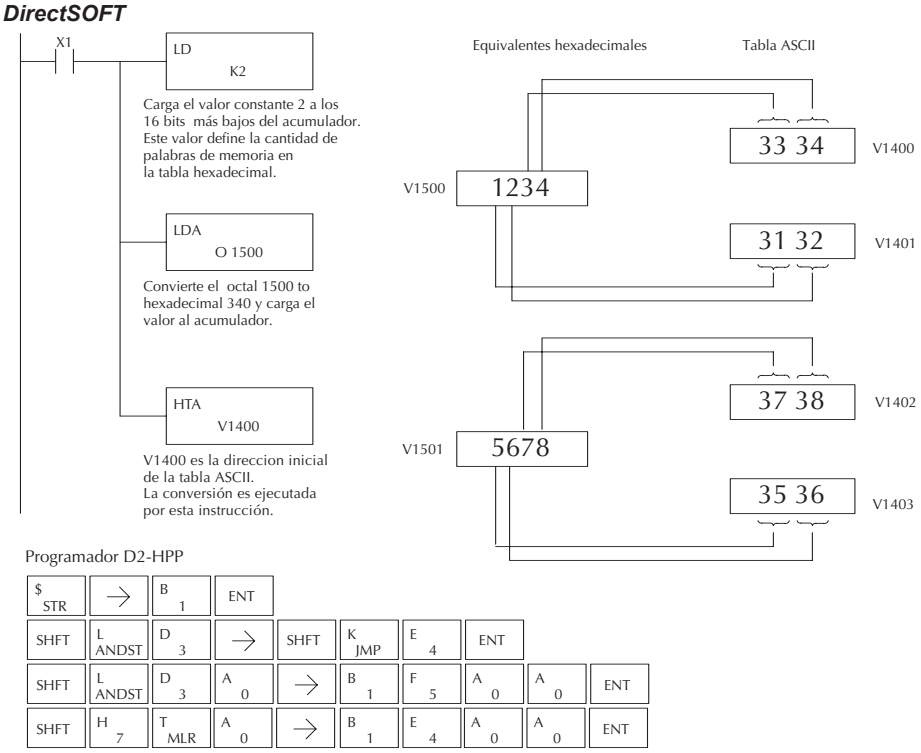
Paso 3: Especifique la localización de memoria V (Vaaa) de inicio para la tabla ASCII en la instrucción HTA.

Sugerencia: — Se puede usar la instrucción LDA para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria, para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Tipo de datos del operando	Rango en el DL06
	aaa
Memoria V V	Vea el mapa de memorias

Indicadores	Descripción
SP53	ON cuando el valor del operando es más grande de lo que puede procesar el acumulador.

En el ejemplo siguiente, cuándo X1 está ON, se carga la constantee (K2) al acumulador usando la instrucción LD. La dirección inicial para la tabla hexadecimal (V1500) es cargada al acumulador usando la instrucción LDA. La dirección inicial para la tabla ASCII (V1400) es especificada en la instrucción HTA.



La tabla de abajo lista los valores válidos de ASCII para la conversión HTA.

Valores ASCII válidos para conversión HTA			
Valor hexadecimal	**Valor ASCII**	**Valor hexadecimal**	**Valor ASCII**
0	30	8	38
1	31	9	39
2	32	A	41
3	33	B	42
4	34	C	43
5	35	D	44
6	36	E	45
7	37	F	46

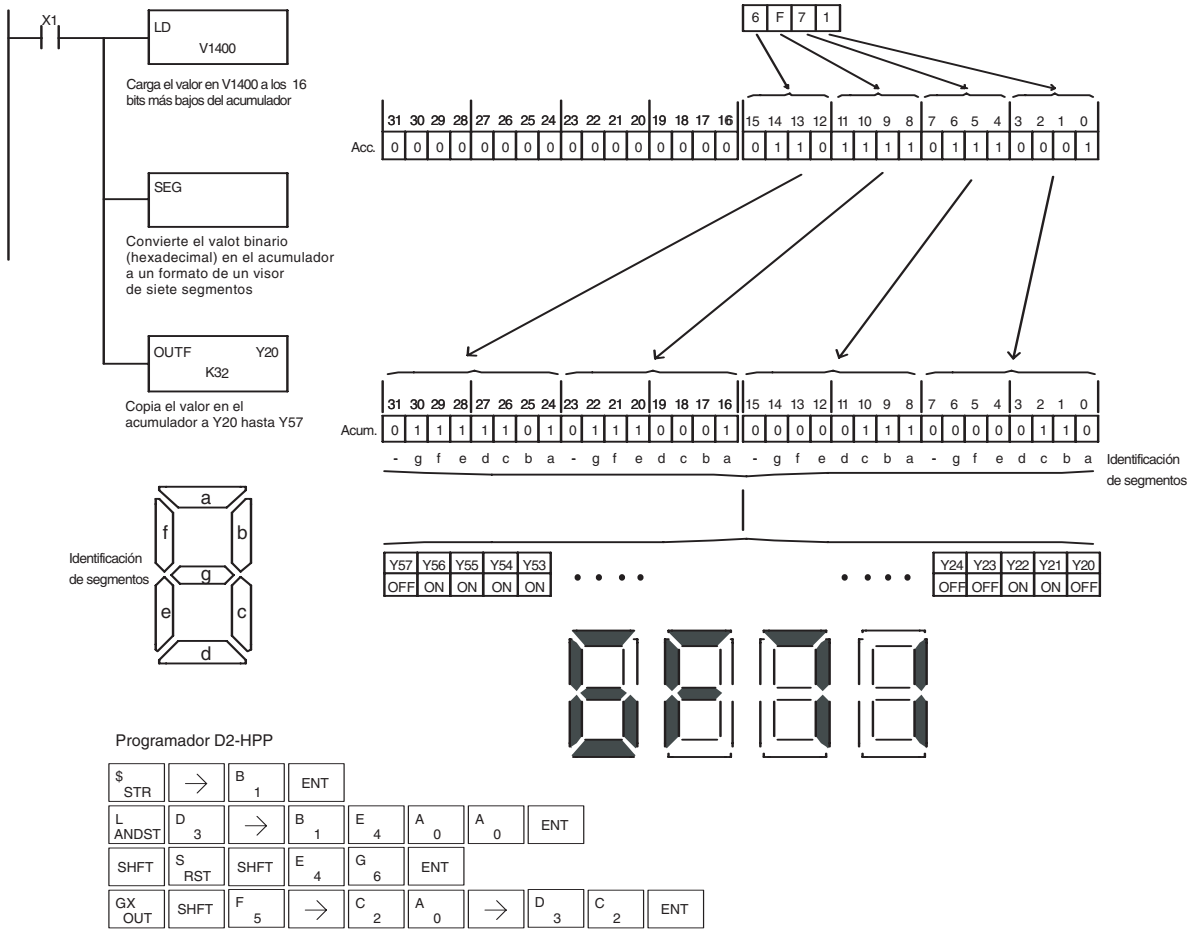
La instrucción Segment (SEG)

DS5	Usado
HPP	Usado

La instrucción SEGMENT convierte un valor hexadecimal de 4 dígitos en el acumulador a un formato de visor de 7 segmentos. El resultado se va al acumulador.

En el ejemplo siguiente, cuándo X1 está ON, el valor en V1400 se carga en los 16 bits más bajos del acumulador usando la instrucción LD. El valor binario (hexadecimal) en el acumulador es convertido a un formato de siete segmentos usando la instrucción SEGMENT. El modelo de bits en el acumulador es copiado a Y20-Y57 usando la instrucción OUTF.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT						
L ANDST	D 3	→	B 1	E 4	A 0	A 0	ENT		
SHFT	S RST	SHFT	E 4	G 6	ENT				
GX OUT	SHFT	F 5	→	C 2	A 0	→	D 3	C 2	ENT

La instrucción Gray Code (GRAY)

DS5	Usado
HPP	Usado

La instrucción GRAY convierte un valor de código Gray de 16 bits a un valor BCD. La conversión BCD requiere 10 bits del acumulador. Los 22 bits superiores son colocados en "0". Esta instrucción está diseñada para uso con aparatos (típicamente encoders) que usan el código Gray, como los encoders absolutos.

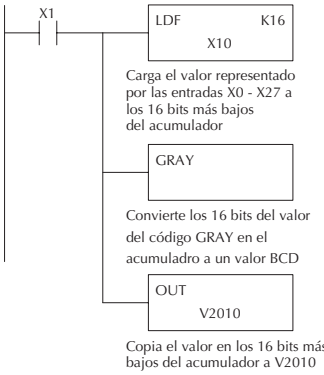
La instrucción GRAY convertirá directamente un número de código GRAY a un número BCD para aparatos que tienen una resolución de 512 o 1024 conteos por revolución. Si un aparato que tiene una resolución de 360 conteos por revolución lo deberá ser usada debe restar un valor BCD de 76 del valor convertido para obtener el resultado apropiado. Para un aparato que tiene una resolución de 720 conteos por revolución usted debe restar un valor BCD de 152.

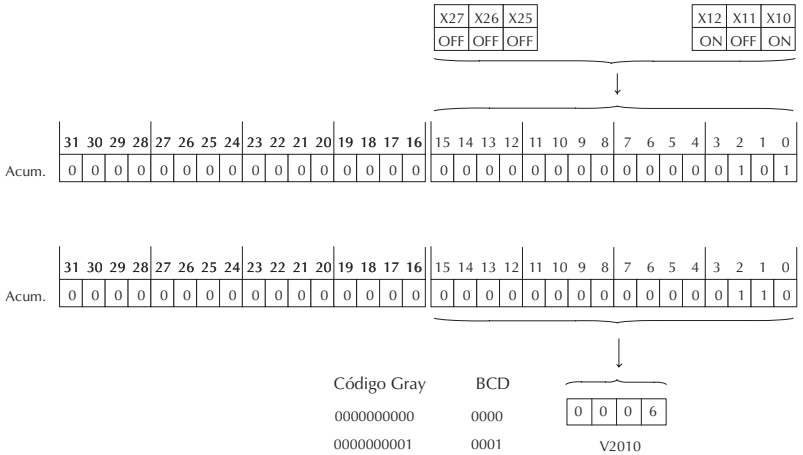
Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

En el ejemplo siguiente, cuándo X1 está ON el valor binario representado por X10-X27 es cargado al acumulador usando la instrucción LDF. El valor del código GRAY en el acumulador es convertido a BCD usando la instrucción GRAY. El valor en los 16 bits más bajos del acumulador es copiado a V2010.

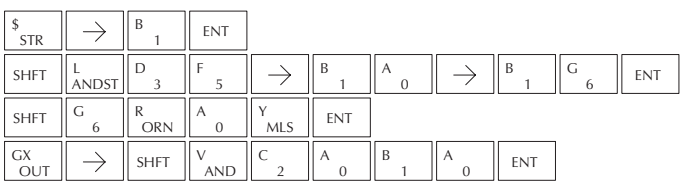
NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

DirectSOFT





Programador D2-HPP



Código Gray	BCD
0000000000	0000
0000000001	0001
0000000011	0002
0000000010	0003
0000000110	0004
0000000111	0005
0000000101	0006
0000000100	0007
1000000001	1022
1000000000	1023

La instrucción Shuffle Digits (SFLDGT)

DS5	Usado
HPP	Usado

La instrucción SFLDGT baraja un máximo de 8 dígitos re-arreglándolos en una orden especificada. Esta función requiere que los parámetros sean cargados al primer nivel del Stack del acumulador y al acumulador con dos instrucciones adicionales. Abajo están listados los pasos necesarios para usar la función SFLDGT.

SFLDGT

Paso 1: Cargue el valor (dígitos) para ser barajados en el primer nivel del Stack del acumulador.

Paso 2: Cargue la orden en que los dígitos serán barajados en el acumulador.

Paso 3: Use la instrucción SFLDGT.

NOTA: Si el número especificado para especificar el orden contiene un 0 o 9 hasta F la posición correspondiente será colocada como 0.

Indicadores	Descripción
SP63	ON cuando el resultado de la instrucción hace que el valor en el acumulador sea 0.
SP70	ON cuando el valor en el acumulador es negativo.

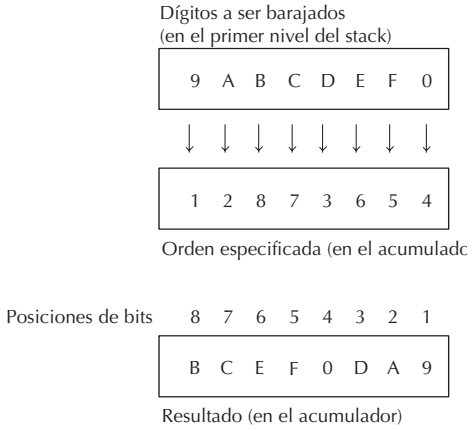
Diagrama de bloque de barajada de dígitos

Hay un máximo de 8 dígitos que se pueden barajar.

Las posiciones de los bits en el primer nivel del Stack del acumulador definen los dígitos a ser barajados.

Ellos corresponden a las posiciones de bits en el acumulador que define la orden que los dígitos se barajarán.

Los dígitos se barajan y el resultado se va al acumulador.



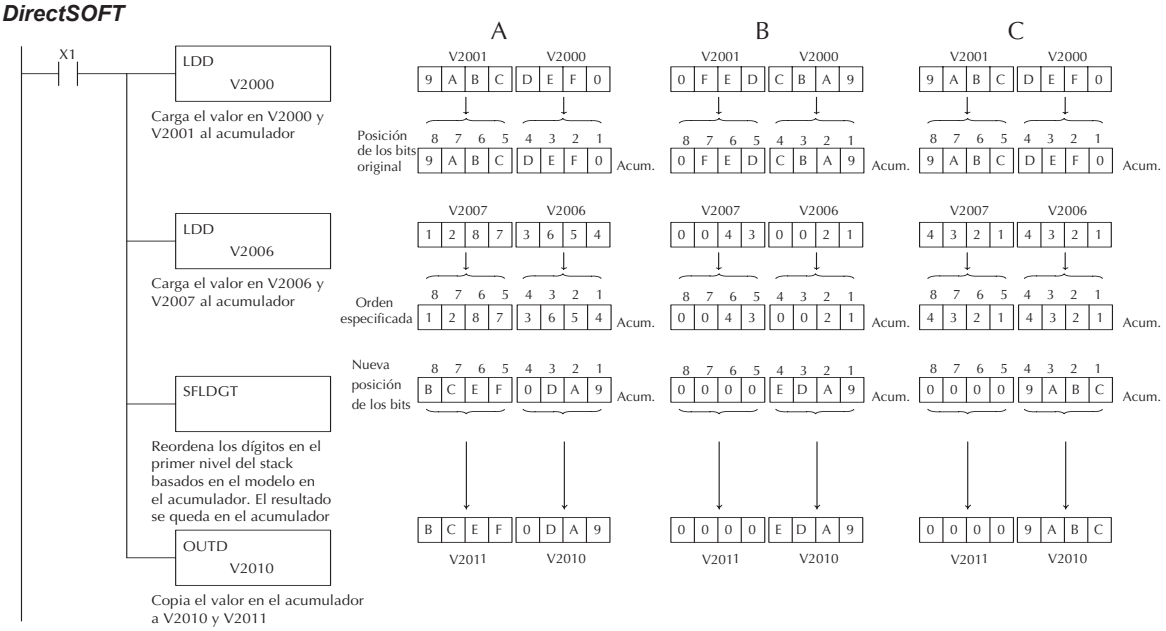
En el ejemplo siguiente cuando X1 está ON, el valor en el primer nivel del Stack del acumulador será reorganizado en la orden especificada por el valor en el acumulador.

El ejemplo A muestra cómo los dígitos siendo barajados trabajan, cuándo 0 o 9 hasta F no se usan, cuándo se especifica la orden que los dígitos deben ser barajados. También, no hay números duplicados en la orden especificada.

El ejemplo B muestra cómo los dígitos siendo barajados trabajan, cuando se usa un 0 o de 9 hasta F, cuándo se especifica la orden que los dígitos deberán ser barajados. Note que cuando se ejecuta la instrucción SFLDGT, las posiciones de bits en la primera dirección del Stack que tenía un 0 o de 9 hasta F correspondiente en el acumulador (orden especificado) son puestos a "0".

El ejemplo C muestra cómo los dígitos siendo barajados trabajan, cuando se usan números duplicados, especificando la orden en que los dígitos deberán ser barajados. Note que cuando se ejecuta la instrucción SFLDGT, se usa el número duplicado más significativo en la orden especificada en el resultado.

DirectSOFT



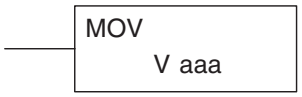
Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	C 2	A 0	A 0	G 6	ENT
SHFT	S RST	SHFT	F 5	L ANDST	D 3	G 6	T MLR	ENT	
GX OUT	SHFT	D 3	→	C 2	A 0	B 1	A 0	ENT	

Instrucciones de tablas (Tablas son simplemente memorias consecutivas)

La instrucción Move (MOV)

DS5	Usado
HPP	Usado



La instrucción MOV copia los valores de una tabla de memoria V a otra tabla de memoria V de una misma longitud. La tabla original queda intacta. Los parámetros de función se cargan en el primer nivel del Stack del acumulador y en el acumulador con dos instrucciones adicionales. Abajo están listados los pasos necesarios para programar la función MOV.

- Paso 1 Cargue el número de direcciones de memoria V a ser copiados al primer nivel del Stack del acumulador. Este parámetro es un valor hexadecimal (máx. kFFF, 4096 decimall)
- Paso 2 Cargue la dirección inicial de memoria V de las direcciones a ser copiadas al acumulador. Este parámetro es un valor hexadecimal.
- Paso 3 Use la instrucción MOV que especifica donde está la dirección inicial de memoria V (Vaaa) en la tabla de destino.

Sugerencia: — Para parámetros que necesitan valores hexadecimales cuando se refieren a direcciones de memoria, se puede usar la instrucción LDA para convertir una dirección de octal al equivalente hexadecimal y cargar el valor en el acumulador.

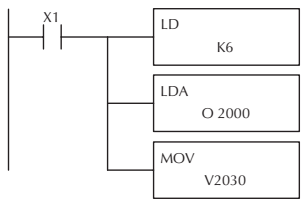
Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP53	ON cuando el valor del operando es más grande que lo que el acumulador puede usar

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor constantee (K6) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en el primer nivel del Stack después que se ejecuta la instrucción LDA. La dirección octal 2000 (V2000), que es la dirección inicial para la tabla fuente, se carga al acumulador. La dirección de la tabla de destino (V2030) es especificada en la instrucción MOV.

Esta es la única instrucción que permite escribir datos a memoria no-volátil.

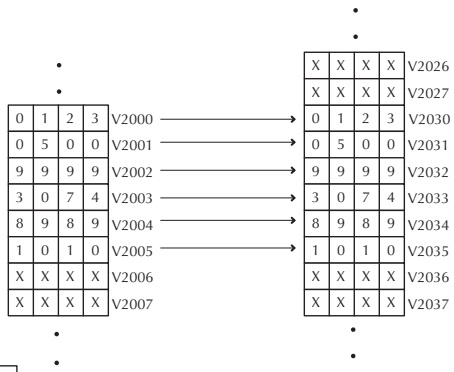
DirectSOFT



Carga el valor constante 6 hexadecimal en los 16 bits más bajos del acumulador

Convierte el octal 2000 al hexadecimal 400 y carga el valor al acumulador

Copia la tabla especificada a una tabla que comienza en la dirección V2030



Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	→	SHFT	K JMP	G 6	ENT		
SHFT	L ANDST	D 3	A 0	→	C 2	A 0	A 0	A 0	ENT
SHFT	M ORST	O INST#	V AND	→	C 2	A 0	D 3	A 0	ENT

La instrucción Move Memory Cartridge (MOVMC)

La instrucción Load Label (LDLBL)

DS5	Usado
HPP	Usado

Las instrucciones MOVMC y LDLBL son usadas para copiar los datos en la memoria ladder de un programa a memoria V. La instrucción LDLBL se usa con la instrucción MOVMC cuando se copian los datos de memoria ladder en un programa a la memoria V.

MOVMC
V aaa

Para copiar los datos de la memoria ladder a la memoria V, se cargan los parámetros de la función a los primeros dos niveles del Stack del acumulador y a acumulador con dos instrucciones adicionales.

LDLBL
K aaa

Esta instrucción está relacionada con las instrucciones DLBL, ACON y NCON. Le recomendamos que vea las definiciones de estas instrucciones.

5

Abajo están listados los pasos necesarios para programar las funciones MOVMC y LDLBL.

- Paso 1: Cargue el número de palabras a ser copiado en el segundo nivel del Stack del acumulador.
- Paso 2: Cargue el desplazamiento del área de data label en la memoria ladder y el comienzo del bloque de memoria V en el primer nivel del Stack .
- Paso 3:Cargue la etiqueta de datos de fuente (LDLBL Kaaa) al acumulador cuando se copian los datos de memoria ladder a la memoria V. Esto es la fuente de la localización del valor.
- Paso 4: la instrucción MOVMC que especifica el destino en la memoria V (Vaaa). Esto es el destino de copia.

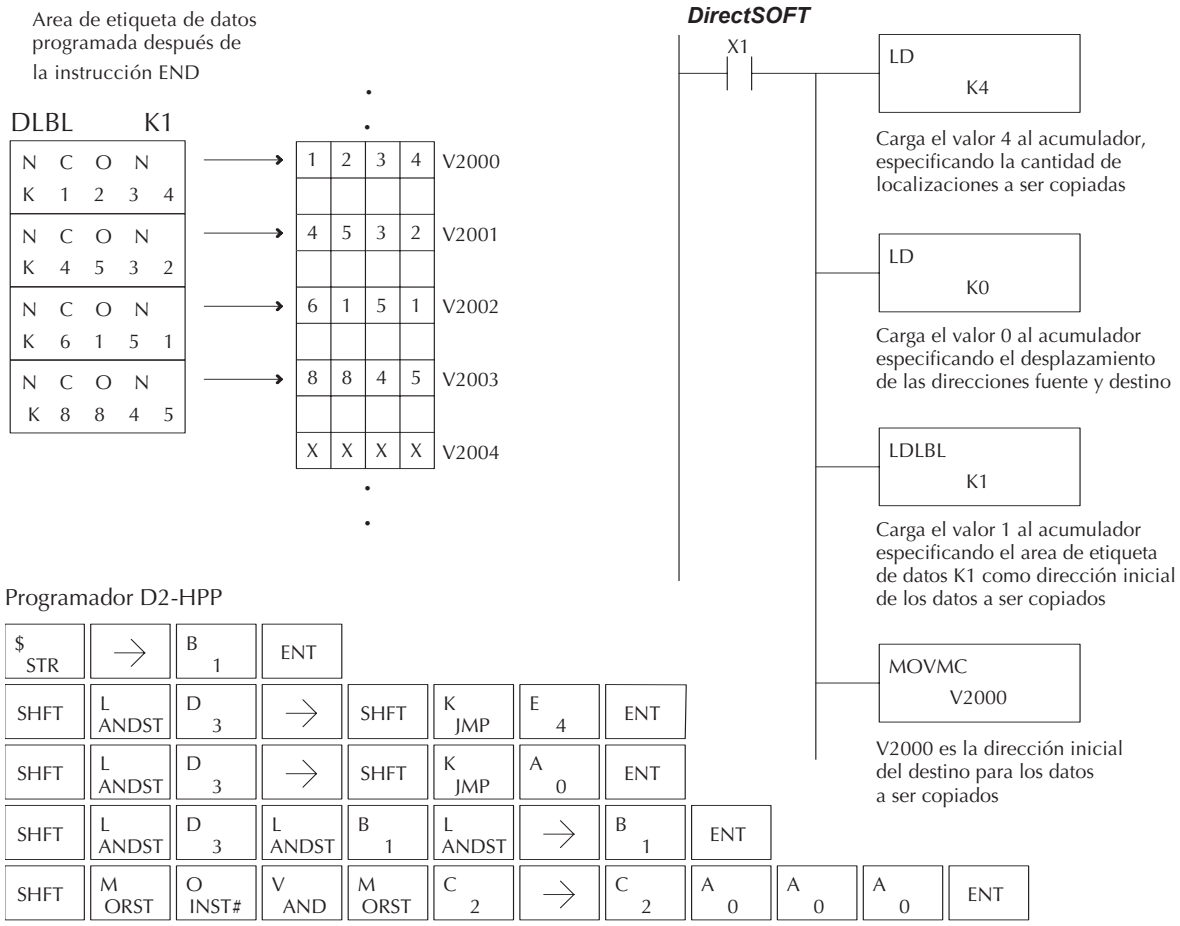
Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria

NOTA: Para más información sobre LDLBL, vea la página 5-187.

ADVERTENCIA: El desplazamiento para este uso de la instrucción comienza en 0, pero puede ser cualquier número que no dé lugar a datos fuera del área de datos de la fuente que es copiada en la tabla de destino. Cuando un desplazamiento está fuera de los límites de la información de la fuente, entonces serán transferidos valores desconocidos de datos en la tabla de destino.

Copie datos de un area de etiqueta de datos a la memoria V (Data Label Area)

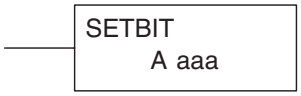
En el ejemplo de abajo, se copian los datos de un área de etiqueta de datos a la memoria V. Cuándo X1 está ON, se carga el valor constantee (K4) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en la segunda localización del Stack después que se ejecutan las próximas instrucciones LD y LDLBL. El valor constantee (K0) es cargado al acumulador, especificando el desplazamiento para los datos fuente y destino. Se coloca en la primera localización del Stack, después que se ejecuta la instrucción de LDLBL. La dirección fuente de donde se copian los datos es cargada al acumulador usando la instrucción LDLBL. La instrucción MOVMC especifica la dirección inicial de la tabla de destino y ejecuta la copia de datos del área de etiqueta de datos a la memoria V.



La instrucción SETBIT

DS5	Usado
HPP	Usado

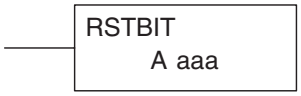
La instrucción SETBIT pone un solo bit en "1" dentro de un rango de direcciones de memoria V.



La instrucción RSTBIT

DS5	Usado
HPP	Usado

La instrucción RSTBIT coloca un solo bit en "0" en un rango de localizaciones de memoria V.



La descripción siguiente se aplica a las instrucciones SETBIT y RSTBIT.

Paso 1: Cargue la longitud de la tabla (el número de direcciones de memoria V) al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 hasta FF, que es 255 decimal.

Paso 2: Cargue la dirección inicial de memoria V de la tabla al acumulador. Este parámetro debe ser un valor hexadecimal. Usted puede usar la instrucción LDA para convertir una dirección octal a hexadecimal.

Paso 3: Coloque la instrucción SETBIT o RSTBIT. Esto especifica la referencia del número del bit que usted quiere hacer "1" o "0". El número del bit está en octal, y el primer bit en la tabla es el número "0".

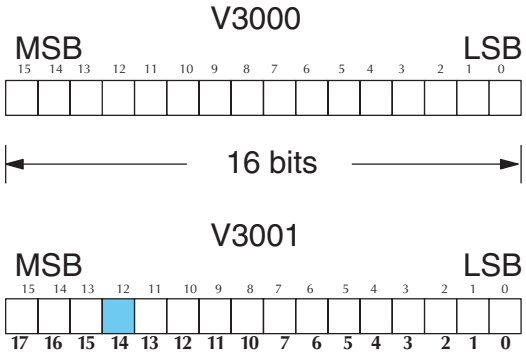
Sugerencia: — Recuerde que cada dirección de memoria V contiene 16 bits. Así, los bits de la primera palabra de la tabla se numeran de 0 a 17 octal. Por ejemplo, si la longitud de tabla es de seis palabras, entonces 6 palabras = (6x16 bits) = 96 bits (decimal) o 140 octal. El rango permisible de números de referencia de bits sería 0 a 137 octal. SP 53 se hará ON si el bit especificado está fuera del rango de la tabla.

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria

Indicadores	Descripción
SP53	ON cuando el número de bit referenciado en el Set Bit o en el Reset Bit excede el rango de la tabla.

NOTA: Indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP.

Por ejemplo, suponga que tenemos una tabla que comienza en V3000 que tiene dos palabras, como mostrado a la derecha. Cada palabra en la tabla contiene 16 bits, o 0 a 17 en octal. Para poner el bit 12 en la segunda palabra, usamos su referencia octal (bit 14). Entonces calculamos la dirección octal de bit desde el comienzo de la tabla, de modo que 17 + 14 = 34 octal. El programa siguiente muestra cómo poner el bit (como mostrado) en "1".



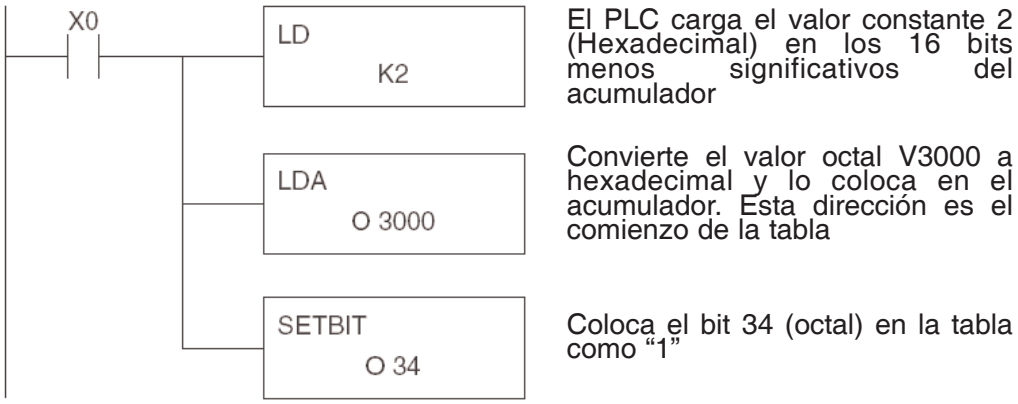
En este ejemplo usaremos la entrada X0 para disparar o activar la operación SETBIT.

Primero, cargamos la longitud de la tabla (2 palabras) al Stack del acumulador.

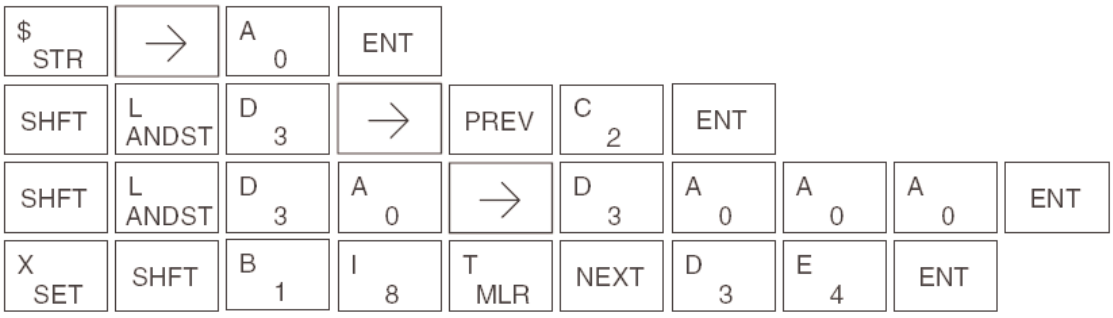
Luego, cargamos la dirección inicial en el acumulador. Ya que V3000 es un número octal lo tenemos que convertir a hexadecimal usando la instrucción LDA.

Finalmente, usamos la instrucción SETBIT (o RSTBIT) y especificamos la dirección octal del bit (bit 34), referenciada de la tabla.

DirectSOFT



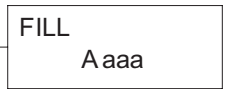
Programador D2-HPP



La instrucción Fill (FILL)

DS5	Usado
HPP	Usado

La instrucción FILL llena una tabla de hasta 255 direcciones de memoria V con un valor (Aaaa), que es una dirección de memoria V o una constantee de 4 dígitos. Los parámetros de la función son cargados al primer nivel del Stack del acumulador y al acumulador con dos instrucciones adicionales. Abajo están listados los pasos necesarios para programar la función FILL.



Paso 1:— Cargue la cantidad de direcciones de memoria V a ser llenadas al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, de 0 a FF, que es 255 decimal.

Paso 2:— Cargue la dirección inicial de memoria V de la tabla en el acumulador. Este parámetro debe ser un valor hexadecimal.

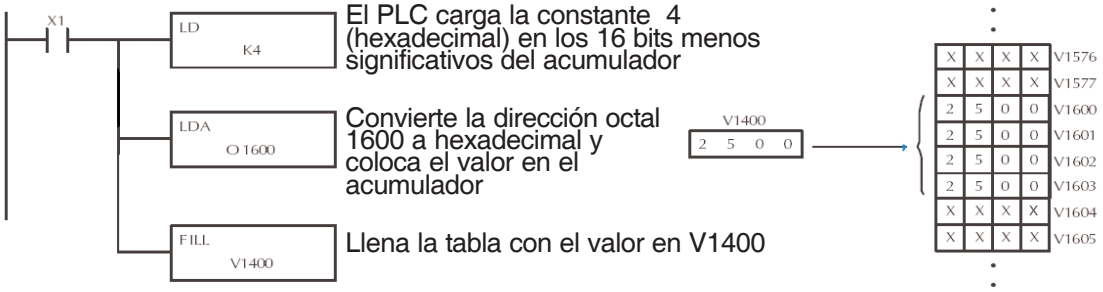
Paso 3:— Coloque la instrucción FILL que especifica el valor para llenar la tabla.

Sugerencia: — Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria
Constante . . . . . K	0–FF

Indicadores	Descripción
SP53	On si la dirección de memoria está fuera de rango

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor (K4 constante) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en el primer nivel del Stack del acumulador cuando se ejecuta la instrucción LDA. La dirección octal 1600 (V1600) es la dirección inicial de la tabla y se carga en el acumulador usando la instrucción LDA. El valor para llenar la tabla con (V1400) es especificado en la instrucción FILL.

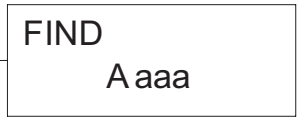


Programador D2-HPP

La instrucción Find (FIND)

DS5	Usado
HPP	Usado

La instrucción Find se usa para buscar un valor especificado en una tabla de memoria V de hasta 255 direcciones. Los parámetros de la función FIND son cargados en el primer y segundo nivel del Stack del acumulador y del acumulador con tres instrucciones adicionales. Abajo están listados los pasos necesarios para programar la función Find.



Paso 1: Cargue la longitud de la tabla (el número de direcciones de memoria V) en el segundo nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, de 0 hasta FF, que es 255 decimal.

Paso 2: Cargue la dirección de la memoria V de inicio de la tabla en el primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal.

Paso 3: Cargue la cantidad de memorias a ser saltadas de la dirección inicial para comenzar la búsqueda. Este parámetro debe ser un valor hexadecimal.

Paso 4: Coloque la instrucción Find especificando el primer valor a ser encontrado en la tabla.

Resultado: — El resultado es expresado en hexadecimal y muestra cuantas memorias V hay entre el inicio de la tabla y la memoria encontrada. El desvío de la dirección inicial a la primera dirección de la memoria V que contiene el valor de búsqueda es vuelto al acumulador. SP53 se pondrá ON si se especifica una dirección fuera de la tabla en el desvío, o el valor no es encontrado. Si el valor no se encuentra serán colocados 0s en el acumulador.

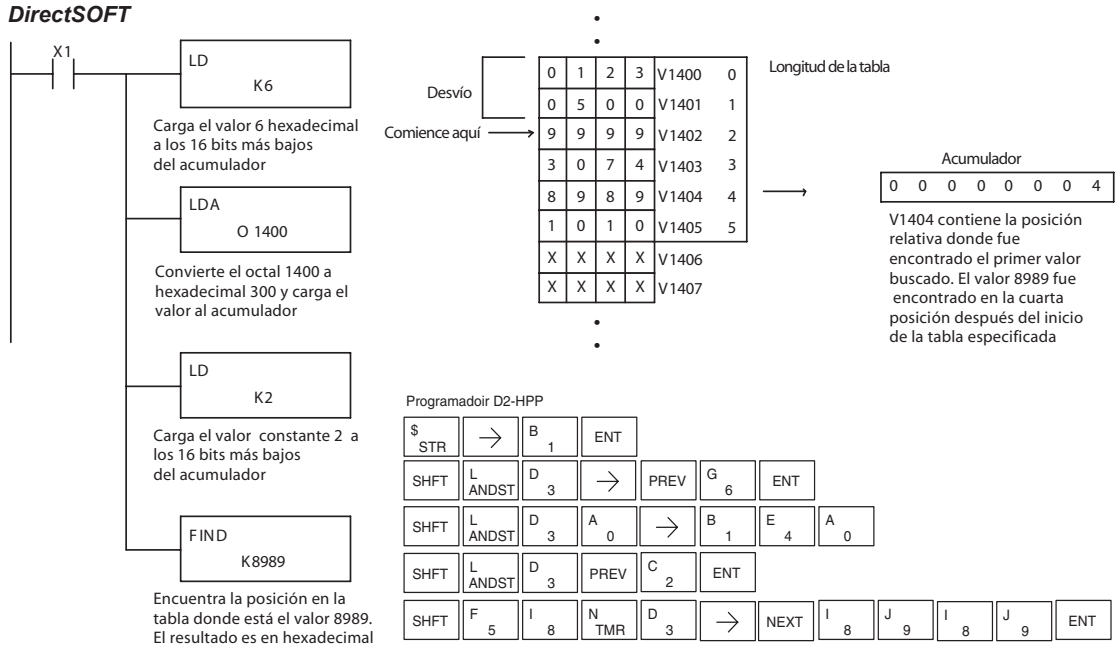
Sugerencia: — Para parámetros que requieran valores hexadecimales cuando se refieran a direcciones de memoria se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Tipo de operando de datos	Rango del DL06
..... **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Constante ..... K	0–FF

Indicadores	Descripción
SP53	ON si no hay un valor en la tabla que sea igual a valor de búsqueda.

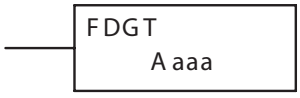
NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP.

En el ejemplo siguiente, cuando X1 está encendido, el valor de la constante K6 se carga en el acumulador usando el instrucción..Este valor especifica la longitud de la tabla y se coloca en la segunda localización del stack cuando se ejecuta la instrucción siguiente. La dirección octal 1400 (V1400) es la localización inicial de la tabla y se carga en el acumulador. Este valor se pone en el primer nivel del stack del acumulador cuando se ejecuta el instrucción siguienteLD. El desplazamiento (offset K2) se carga en los 16 bits más bajos del acumulador usando la instrucción LD. El valor que se encontrará en la tabla se especifica en el instrucción FIND. Si se encuentra un valor igual al valor de la búsqueda, el desplazamiento (desde el inicio de la tabla) donde se encuentra el valor residirá en el acumulador.



La instrucción Find Greater Than (FDGT)

La instrucción FDGT se usa para buscar la primera ocurrencia de un valor en una tabla de memoria V que es más grande que el valor (Aaaa) especificado, que puede ser una dirección de memoria V o una constantee de 4 dígitos. Los parámetros de la función son cargados en el primer nivel del Stack del acumulador y el acumulador por dos instrucciones adicionales. Abajo están listados los pasos necesarios para programar la instrucción FDGT.



Paso 1: Cargue la longitud de la tabla (hasta 255 direcciones) al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, de 0 hasta FF.

Paso 2: Cargue la direción inicial de la tabla en el acumulador. Este parámetro debe ser un valor hexadecimal.

Paso 3: Coloque la instrucción FDGT que especifica el valor prefijado de búsqueda. El resultado es expresado en hexadecimal.

Resultado:— El desvío de la dirección inicial a la primera dirección de memoria V que contiene el valor más grande de búsqueda es vuelto al acumulador. SP53 se pondrá ON si el valor no se encuentra y se colocarán "0s" en el acumulador.

Sugerencia: — Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria la instrucción LDA se puede usar para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

NOTA: Esta instrucción no tiene una cantidad de memorias a ser saltadas tal como el que se usa para la instrucción FIND.

Tipo de operando de datos	Rango del DL06
. . . . . A	aaa
Memoria V V	Vea el mapa de memoria
Constante . . . . . K	0–FF

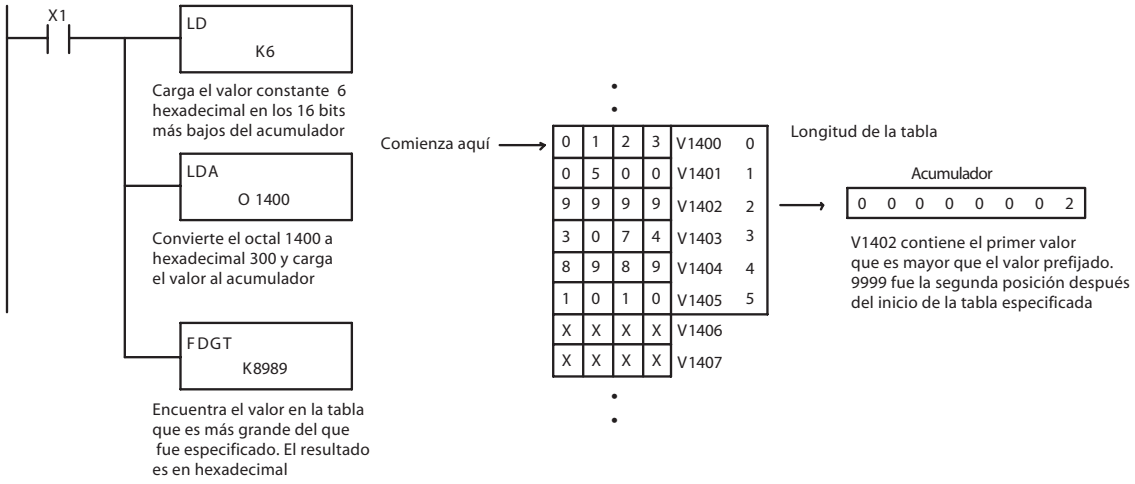
Indicadores	Descripción
SP53	ON si no hay un valor en la tabla que sea mayor que el valor de búsqueda

NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP. El puntero para esta instrucción comienza en 0 y se va al acumulador.

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor constantee (K6) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en la primera localización del Stack después que se ejecuta la instrucción LDA. La dirección octal 1400 (V1400) es la dirección inicial de la tabla y se carga al acumulador. El valor prefijado de búsqueda se especifica en la instrucción Find. Si se encuentra un valor mayor que el valor prefijado de búsqueda, el desvío (de la dirección inicial de la tabla) donde el valor se localiza residirá en el acumulador.

Si no hay un valor en la tabla que sea mayor que el valor buscado, se almacena un cero en el acumulador y SP53 se hará ON.

DirectSOFT



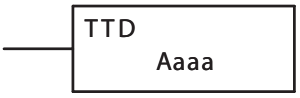
Programador D2-HPP

$ STR	→	B 1	ENT							
SHFT	L ANDST	D 3	→	PREV	G 6	ENT				
SHFT	L ANDST	D 3	A 0	→	B 1	E 4	A 0	A 0	ENT	
SHFT	F 5	D 3	G 6	T MLR	→	NEXT	I 8	J 9	I 8	J 9	ENT

La instrucción Table to Destination (TTD)

DS5	Usado
HPP	Usado

La instrucción TTD copia un valor de una tabla de memoria V a una dirección de memoria V e incrementa el puntero de la tabla en 1. La primera dirección de memoria V en la tabla contiene el puntero de la tabla que indica la próxima dirección a ser copiada en la tabla. La instrucción será ejecutada una vez por barrido si el renglón es verdadero u ON. El puntero de la tabla vuelve a 1 cuando el valor se hace igual a la última dirección en la tabla. Los parámetros de la función son cargados al primer nivel del Stack del acumulador y al acumulador con dos instrucciones adicionales.



Abajo están listados los pasos necesarios para programar la instrucción TTD.

Paso 1: Cargue la longitud de la tabla de datos (el número de direcciones de memoria V) al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 hasta FF, que es 255 decimal..

Paso 2: Cargue la dirección de memoria V inicial para la tabla en el acumulador. (Recuerde, para esta instrucción la dirección inicial de la tabla se usa como el puntero de la tabla.) Este parámetro debe ser un valor hexadecimal.

Paso 3: Coloque la instrucción TTD especificando el destino de la memoria V (Vaaa)

Sugerencia: — Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Sugerencia:— La instrucción se ejecutará cada barrido si la lógica de la entrada está ON. Si usted no quiere que la instrucción ejecute en más que un barrido, se debe usar una instrucción one shot (PD) en la lógica de entrada.

Sugerencia: — La localización del puntero debe ser puesta al valor donde comenzará la operación de tabla. Se debe usar el relevador especial SP0 o una instrucción one shot (PD) de modo que el valor sólo sea puesto en un barrido y no afecte la operación de la instrucción.

Tipo de operando de datos	Rango del DL06
..... **A**	**aaa**
Memoria V V	Vea el mapa de memoria

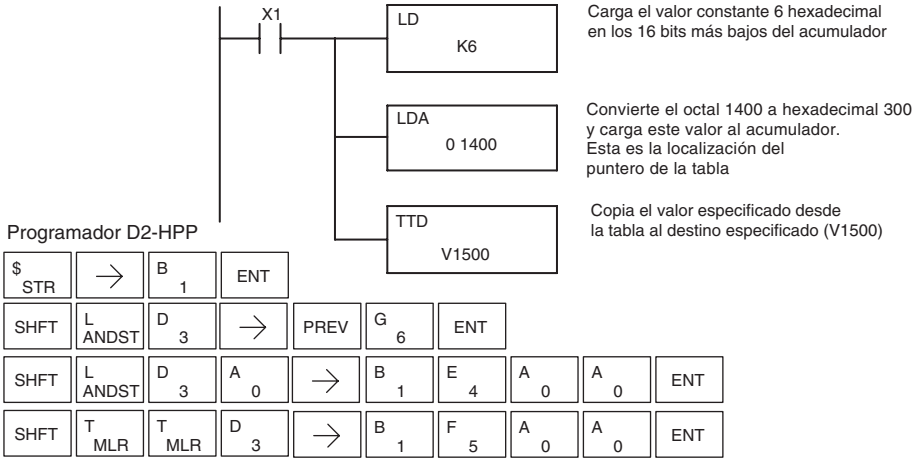
Indicadores	Descripción
SP56	ON cuando el puntero de la tabla llega a la longitud de la tabla.

NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP o en el fin del barrido. El puntero para esta instrucción comienza en 0 y se hace 1 cuando se llega a la longitud de la tabla. Note que el puntero se vuelve a "1" en esta ocasión y no a 0.

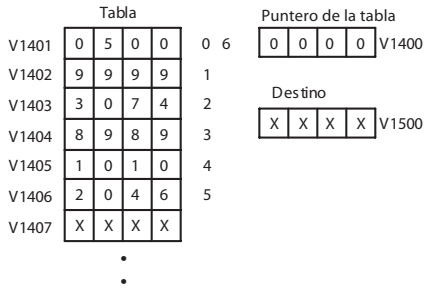
En el ejemplo siguiente, cuándo X1 está ON, se carga el valor constantee (K6) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en la primera localización del Stack después que se ejecuta la instrucción LDA. La dirección octal 1400 (V1400) es la dirección inicial de la tabla fuente y se carga en el acumulador.

Recuerde, V1400 se usa como la localización de un puntero, y no es realmente parte de la tabla fuente de datos. La localización de destino (V1500) es especificada en la instrucción TTD. El puntero de la tabla (V1400 en este caso) será aumentado en "1" después de cada ejecución de la instrucción TTD.

DirectSOFT



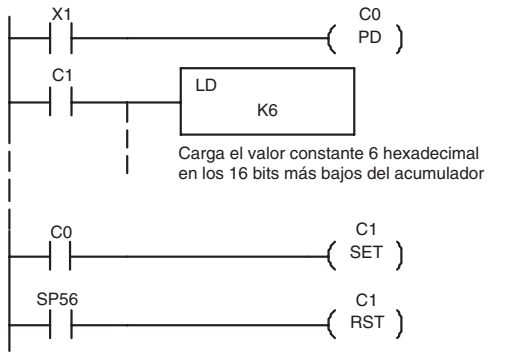
Es importante entender cómo se numeran las direcciones de la tabla. Si examina la tabla del ejemplo, usted notará que la primera dirección de datos, V1401, se usará cuando el puntero es igual a 0, y nuevamente cuando el puntero es igual a seis. ¿Por qué? Porque el puntero es sólo igual a 0 antes de la primera ejecución. De allí en adelante, incrementa de uno a seis y luego vuelve a 1.



También, nuestro ejemplo usa un contacto normal de entrada (X1) para controlar la ejecución. Ya que el barrido de la CPU es extremadamente rápido y el puntero aumenta en 1 automáticamente, la tabla hace el ciclo por todas las direcciones muy rápidamente.

Si esto es un problema, se tiene la opción de usar SP56 en unión con una instrucción one shot (PD) y un enclavamiento (C1 por ejemplo) para permitir a la tabla pasar por todas direcciones una vez y luego para. La lógica mostrada aquí no es necesaria, es solamente un método opcional.

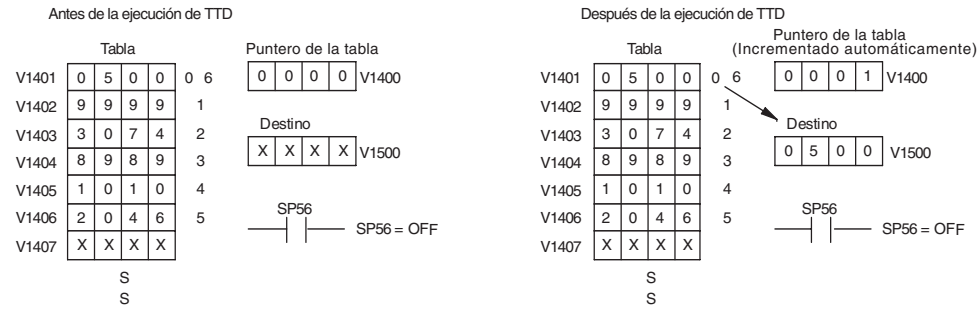
DirectSOFT (Ejemplo de enclavamiento usando SP56)



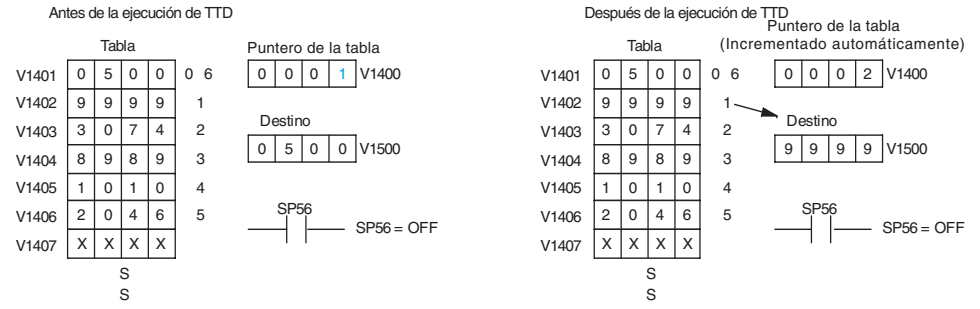
Ya que los relevadores especiales vuelven a 0 al fin del barrido, este enclavamiento debe estar justamente después de la instrucción TTD en el programa

El diagrama en esta página muestra los resultados barrido por barrido de la ejecución del programa del ejemplo.Note como el puntero automáticamente salta de 0 a 6 y luego comienza en 1 en vez de 0. También, note como SP56 es ON solamente hasta el fin del barrido

Barrido N

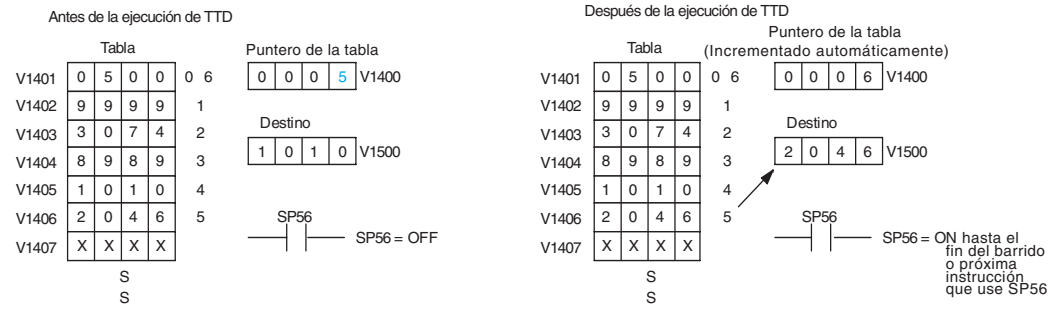


Barrido N+1

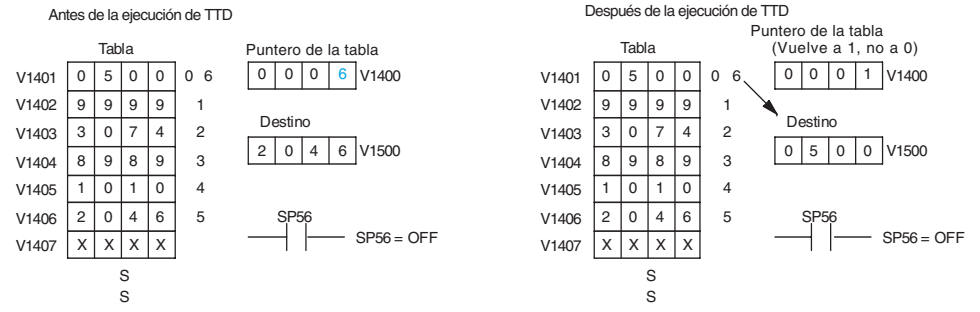


S
S
S

Barrido N+5



Barrido N+6



La instrucción Remove from Bottom (RFB)

DS5	Usado
HPP	Usado

La instrucción RFB copia un valor del fondo de una tabla de memoria V a una dirección de memoria V y decrementa un puntero de la tabla en "1". La primera localización de memoria V en la tabla contiene el puntero de la tabla que indica la próxima localización en la tabla a ser copiada. La instrucción se ejecutará una vez por barrido si el renglón es verdadero. La instrucción parará la operación cuando el puntero es igual a 0. Los parámetros de la función son cargados al primer nivel del Stack del acumulador y al acumulador con dos instrucciones adicionales. Abajo están listados los pasos necesarios para programar la instrucción RFB.

Paso 1:— Cargue la longitud de la tabla (la cantidad de direcciones de memoria V) en el primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 hasta FF.

Paso 2:— Cargue la dirección de la memoria V inicial de la tabla al acumulador. (Recuerde, para esta instrucción, la primera dirección de la tabla se usa como el puntero de la tabla). Este parámetro debe ser un valor hexadecimal.

Paso 3:— Coloque la instrucción RFB que especifica la dirección (Vaaa) de la memoria V de destino.

Sugerencia: — Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Sugerencia:— La instrucción se ejecutará cada barrido si la lógica de entrada está ON. Si usted no quiere que la instrucción se ejecute más que un barrido, se debe usar una instrucción one shot (PD) en la lógica de entrada.

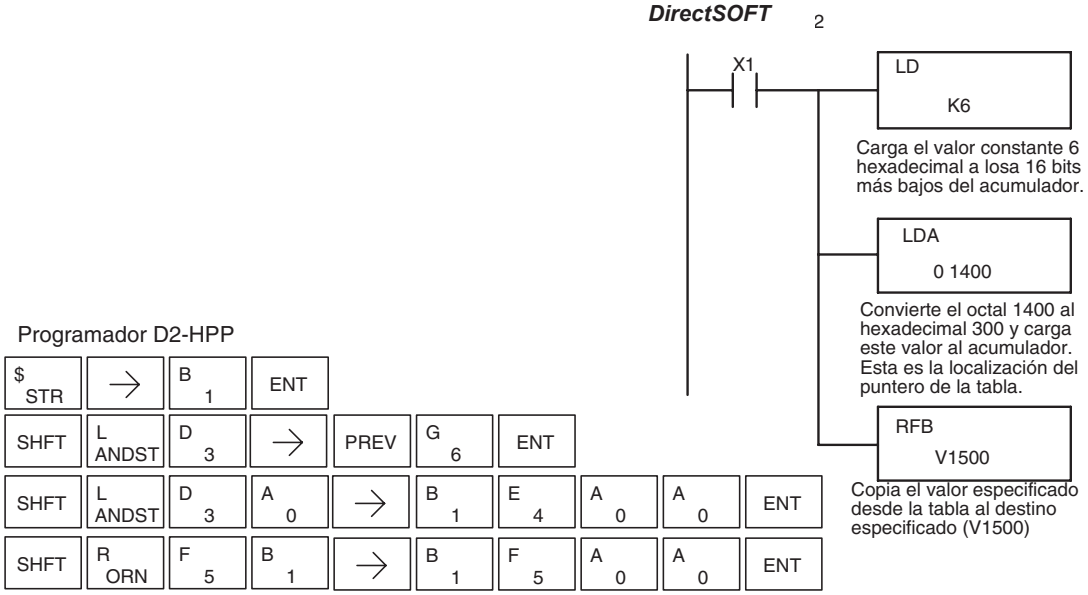
Sugerencia: — La localización del puntero debe ser puesta al valor donde comenzará la operación de tabla. Se debe usar el relevador SP0 especial o la instrucción one shot (PD) de modo que el valor sólo sea puesto en un barrido y no afecte la operación de la instrucción.

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria

Indicadores	Descripción
SP56	ON cuando el valor del puntero de la tabla es igual a 0

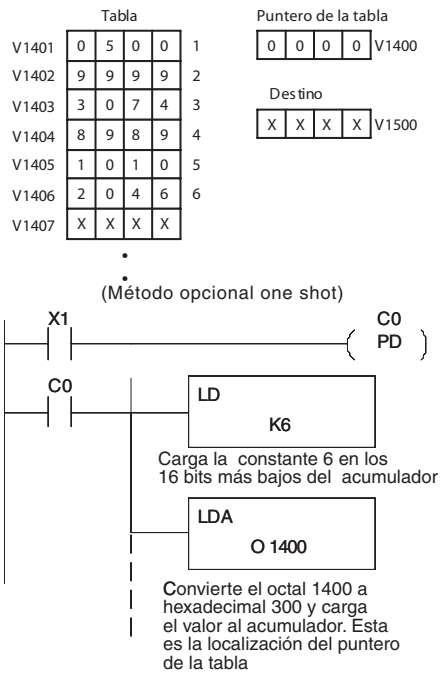
NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP o en el fin del barrido. El puntero para esta instrucción puede ser colocado para iniciar en cualquier parte de la tabla. ***No es colocado automáticamente****. Ud. debe colocar un valor en el puntero en algún lugar del programa ladder.*

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor constantee (K6) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en el primer nivel del Stack después que se ejecuta la instrucción LDA. La dirección octal 1400 (V1400) es la dirección inicial de la tabla fuente y se carga en el acumulador. Recuerde, V1400 se usa como la localización de un puntero y no es realmente parte de la fuente de datos de la tabla. La dirección del destino (V1500) es especificada en la instrucción TFB. El puntero de la tabla (V1400 en este caso) será decrementado en "1" después de cada ejecución de la instrucción RFB.



Es importante entender cómo se numeran las direcciones de la tabla. Si usted examina la tabla del ejemplo, usted verá que la primera localización de datos, V1401, se usará cuando el puntero es igual a uno. La segunda localización de datos, V1402, se usará cuando el puntero es igual a dos, etc.

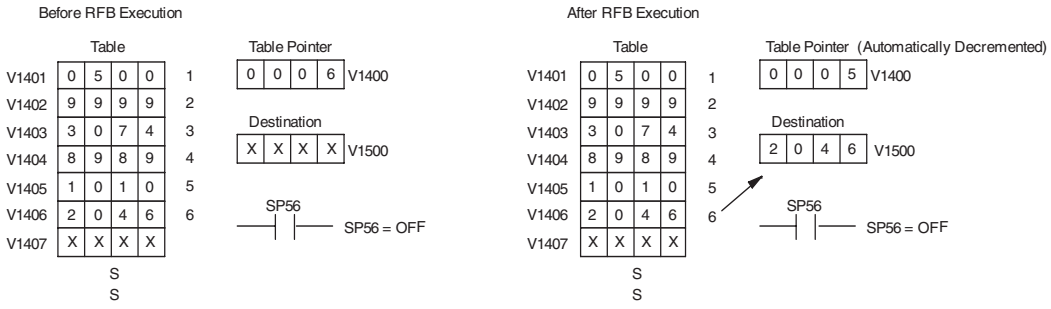
También, nuestro ejemplo usa un contacto normal de entrada (X1) para controlar la ejecución. Ya que el barrido de la CPU es extremadamente rápido y el puntero decrementa el valor automáticamente, la tabla recorre las direcciones en un ciclo muy rápido. Si esto es un problema para su aplicación, usted tiene la opción de usar una instrucción one shot (PD) para quitar un valor cada vez que el contacto de entrada hace la transición de OFF para ON.



El esquema siguiente muestra los resultados de la ejecución barrido por barrido para nuestro programa del ejemplo. Advierta cómo el puntero automáticamente decrece de 6 a 0. También, note cómo SP56 es sólo ON hasta el fin del barrido.

Example of Execution

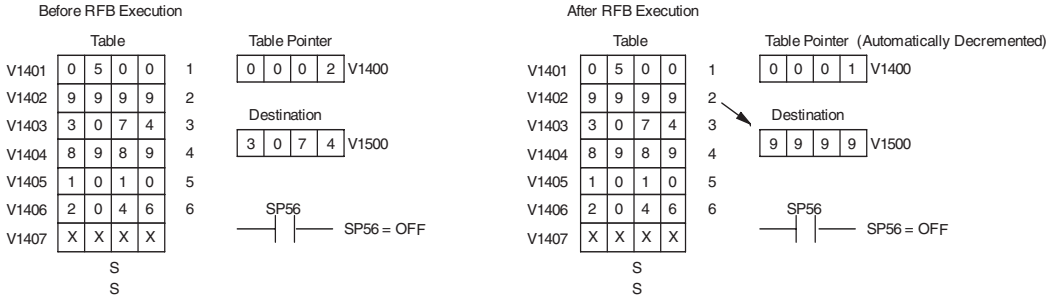
Scan N



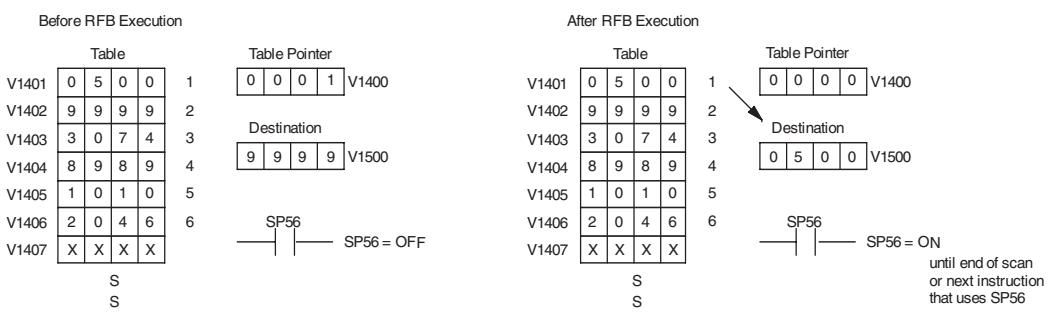
Scan N+1

S
S
S

Scan N+4



Scan N+5



La instrucción Source a Table (STT)

DS5	Usado
HPP	Usado

La instrucción SST copia un valor de memoria V a una tabla de memoria V e incrementa el puntero de la tabla en 1. Cuándo el puntero de la tabla alcanza el fin de la tabla, vuelve a 1. La primera dirección de memoria V en la tabla contiene el puntero de la tabla que indica la próxima dirección en la tabla que almacenará un valor. La instrucción se ejecutará una vez por barrido si el renglón es verdadero.

Los parámetros de la función se cargan al primer nivel del Stack del acumulador y al acumulador con dos instrucciones adicionales. Abajo están listados los pasos necesarios de programar la instrucción STT.

5

Paso 1: Cargue la longitud de la tabla (la cantidad de direcciones de memoria V) al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 a FF.

Paso 2: Cargue la dirección inicial de la memoria V en la tabla al acumulador. (Recuerde, la dirección inicial de la tabla se usa como el puntero de la tabla.) Este parámetro debe ser un valor hexadecimal.

Paso 3: Coloque la instrucción STT especificando la dirección (Vaaa) de la memoria V fuente. Aquí es de donde se moverá el valor.

Sugerencia: — Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria, se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Sugerencia:— La instrucción se ejecutará cada barrido si la lógica de entrada está ON. Si usted no quiere que la instrucción se ejecute por más que un barrido, se debe usar instrucción one shot (PD) en la lógica de entrada.

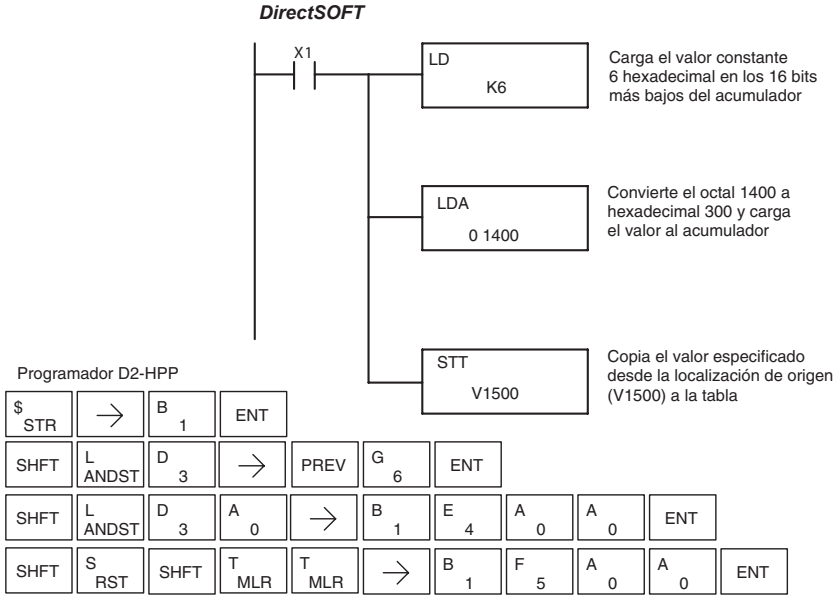
Sugerencia: — El valor contador de la tabla debe ser configurado para indicar el punto de partida de la operación. También, debe ser configurado a un valor que esté dentro de la longitud de la tabla. Por ejemplo, si la tabla es de 6 palabras, entonces el rango admisible de los valores que podrían estar en el puntero debe estar entre 0 y 6. Si el valor está fuera de este rango, los datos no se moverán. También, se debe usar una instrucción one shot (PD) de modo que el valor sólo sea puesto en un barrido y no afecte la operación de la instrucción.

Tipo de operando de datos	Rango del DL06
..... **A**	**aaa**
Memoria V V	Vea el mapa de memoria

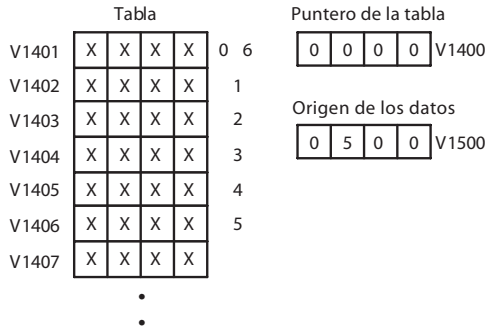
Indicadores	Descripción
SP56	On cuando el puntero de la tabla es igual a la longitud de la tabla.

NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP o en el fin del barrido. El puntero para esta instrucción comienza en 0 y vuelve a 1 automáticamente cuando se alcanza la longitud de

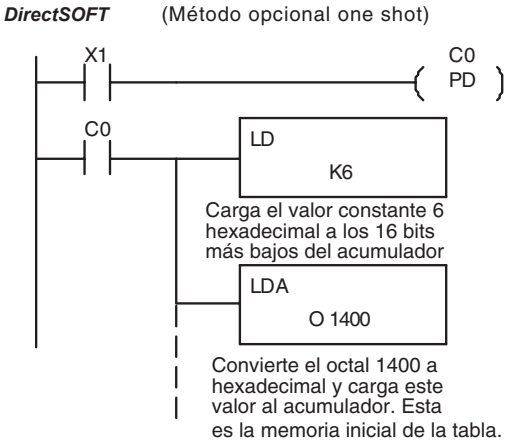
En el ejemplo siguiente, cuándo X1 está ON, se carga el valor constantee (K6) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en la primera localización del Stack después que se ejecuta la instrucción LDA. La dirección octal 1400 (V1400), que es la de la tabla de destino y el puntero de la tabla, se carga al acumulador. La dirección de la fuente de datos (V1500) es especificada en la instrucción STT. El puntero de la tabla será aumentado en "1" cada vez que se ejecuta la instrucción.



Es importante entender cómo se numeran las direcciones de tabla. Si usted examina la tabla del ejemplo, usted notará que la primera dirección de almacenamiento de datos, V1401, se usará cuando el puntero es igual a 0 y otra vez cuando el puntero es igual a seis. ¿Por qué? Porque el puntero es sólo igual a 0 antes de la primera ejecución. De allí en adelante incrementa de 1 a 6 y entonces vuelve a 1.

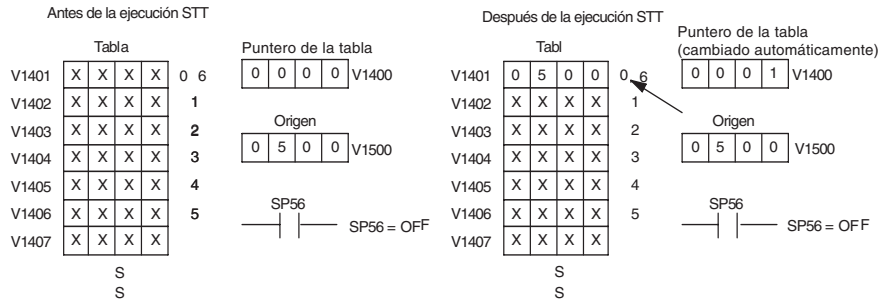


También, nuestro ejemplo usa un contacto normal de entrada (X1) para controlar la ejecución. Ya que el barrido de la CPU es extremadamente rápido y el puntero se incrementa automáticamente, los datos de la fuente se cambiarían a todas las direcciones de tabla muy rápidamente. Si esto es un problema para su aplicación, usted tiene la opción de usar una instrucción one shot (PD) para mover un valor cada vez que el contacto de entrada hace la transición de OFF para ON.

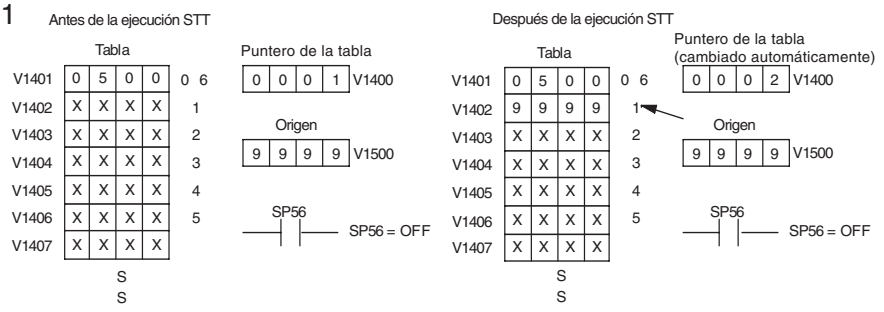


El esquema siguiente muestra barrido por barrido los resultados de la ejecución para el programa ejemplo. Note cómo el puntero automáticamente hace un ciclo de 0 - 6, y luego comienza de nuevo en 1 en vez de 0. También, note cómo es afectado SP56 por la ejecución. Aunque el ejemplo no lo muestre, asumimos que hay otra parte del programa que cambia el valor en V1500 (la fuente de datos) antes de la ejecución de la instrucción STT. Esto no es necesario pero hace más fácil de ver cómo la fuente de datos se copia en la tabla.

Barrido N

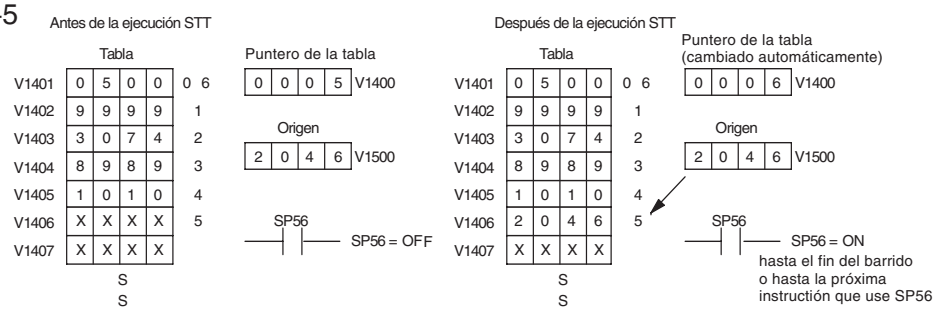


Barrido N+1

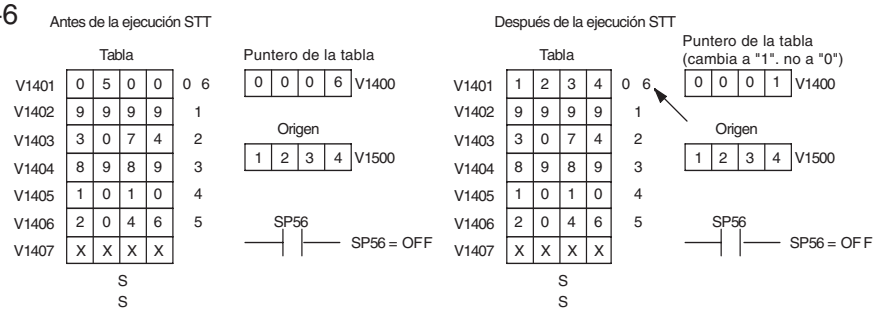


S
S
S

Barrido N+5



Barrido N+6



La instrucción Remove from Table (RFT)

DS5	Usado
HPP	Usado

La instrucción RFT remueve un valor de una tabla y lo almacena en una dirección de memoria V. Cuándo un valor se remueve de la tabla todos los otros valores se mueven para arriba 1 nivel. La primera dirección de memoria V en la tabla contiene el contador de la longitud de la tabla. El valor corriente del contador de la tabla disminuye en 1 cada vez que se ejecuta la instrucción. Si el contador de la longitud es cero o mayor que la longitud máxima de la tabla (especificada en el primer nivel del Stack del acumulador) la instrucción no se ejecutará y SP56 se hará ON.

La instrucción se ejecutará una vez por barrido si que el renglón fuera verdadero. Los parámetros de la instrucción son cargados al primer nivel del Stack del acumulador y al acumulador con dos instrucciones adicionales. Abajo están listados los pasos necesarios de programar la instrucción RFT.

5

Paso 1: Cargue la longitud de la tabla (el número de direcciones de memoria V) al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 a FF.

Paso 2: Cargue la localización de la memoria V de inicio a la tabla en el acumulador. (Recuerde, la localización de inicio de la tabla se usa como el contador de longitud de tabla). Este parámetro debe ser un valor hexadecimal.

Paso 3: Coloque la instrucción RFT que especifica la dirección de la memoria de destino (Vaaa). Esto es, donde el valor en la tabla será movido.

Sugerencia: — Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria, se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Sugerencia:— La instrucción se ejecutará cada barrido si el renglón está ON. Si usted no quiere que la instrucción ejecute más que un barrido, se debe usar la instrucción PD (one shot) en la lógica del renglón.

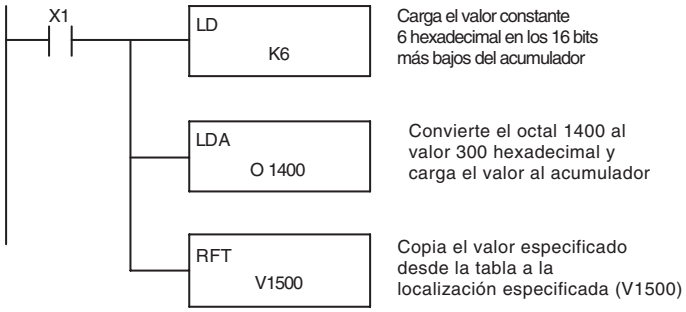
Sugerencia: — El valor del contador de la tabla se debe definir para indicar el punto de partida de la operación. También, debe ser colocado a un valor que esté dentro de la longitud de la tabla. Por ejemplo, si la tabla es de 6 palabras de longitud, entonces el rango admisible de los valores que podrían estar en el contador de la tabla debe estar entre 1 y 6. Si el valor está fuera de este rango o es 0, los datos no se moverán de la tabla. También, se debe usar una instrucción one shot (PD) de modo que el valor sólo sea colocado en un barrido y no afecte la operación de la instrucción.

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria

Indicadores	Descripción
SP56	ON cuando el valor corriente del contador de la tabla es igual a 0

NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP o en el fin del barrido. El puntero para esta instrucción puede comenzar en cualquier lugar en la tabla. ***No es colocado automáticamente****. Usted tiene que cargar un valor en el puntero en algún lugar en su programa.*

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor constantee (K6) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en la primera localización del Stack después que se ejecuta la instrucción LDA. La dirección octal 1400 (V1400) es la dirección inicial de la tabla fuente y se carga en el acumulador. La localización (V1500 del destino) es especificado en la instrucción RFT. El contador de la tabla será disminuido en "1" después que se ejecuta la instrucción.



Programador D2-HPP

$ STR	→	B 1	ENT						
SHFT	L ANDST	D 3	→	PREV	G 6	ENT			
SHFT	L ANDST	D 3	A 0	→	B 1	E 4	A 0	A 0	ENT
SHFT	R ORN	F 5	T MLR	→	B 1	F 5	A 0	A 0	ENT

Ya que el contador de la tabla especifica el rango de los datos que se sacarán de la tabla, es importante entender cómo se numeran las direcciones de la tabla. Si usted examina la tabla del ejemplo, usted notará que las direcciones de datos se numeran desde encima de la tabla. Por ejemplo, si el contador de la tabla comenzó en 6, entonces todas las seis direcciones se afectarían durante la ejecución de la instrucción.

Tabla

V1401	0	5	0	0	1
V1402	9	9	9	9	2
V1403	3	0	7	4	3
V1404	8	9	8	9	4
V1405	1	0	1	0	5
V1406	2	0	4	6	6
V1407	X	X	X	X	

Contador de tabla

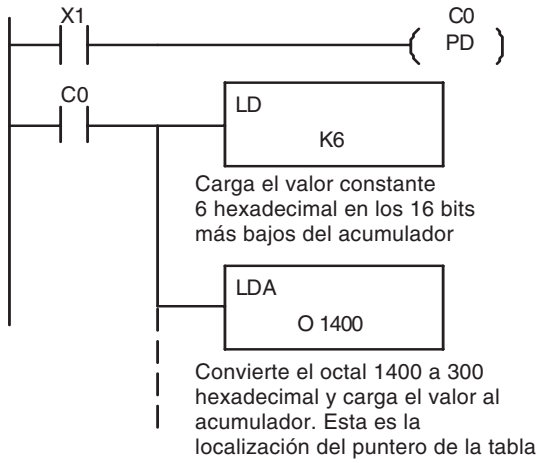
0	0	0	6	V1400

Destino

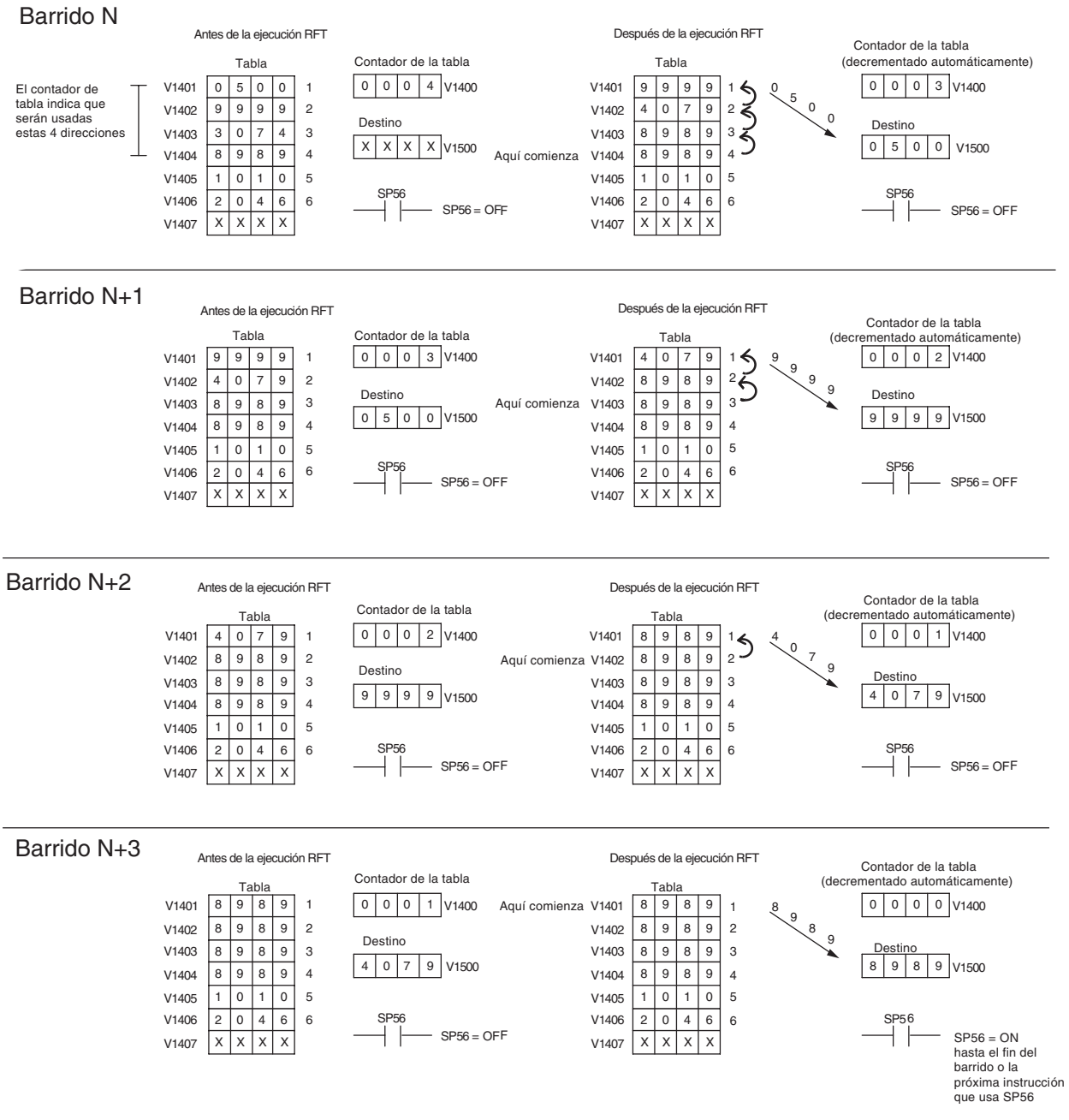
X	X	X	X	V1500

También, nuestro ejemplo usa un contacto normal de entrada (X1) para controlar la ejecución. Ya que el barrido es extremadamente rápido, y el puntero se decrementa automáticamente, los datos se sacarían de la tabla muy rápidamente. Si esto es un problema para su aplicación, usted tiene una opción de usar una instrucción one shot (PD) para sacar un valor cada vez en la transición del contacto de entrada de OFF para ON.

DirectSOFT (Método opcional "one-shot")



El esquema siguiente muestra barrido por barrido los resultados de la ejecución para el programa del ejemplo. En el ejemplo mostramos el valor corriente del contador de la tabla en 4 inicialmente. (Recuerde, usted puede poner el valor corriente del contador de la tabla a cualquier valor que este dentro del rango de la tabla). El contador de la tabla automáticamente decrece de 4 hasta 0 cuando se ejecuta la instrucción. Note cómo las últimas dos posiciones de la tabla, 5 y 6, no son movidas para arriba por la tabla. También, note cómo SP56, que se hace ON cuando el contador de la tabla es cero, es ON sólo hasta que el fin del barrido.



La instrucción Add a Top (ATT)

DS5	Usado
HPP	Usado

La instrucción ATT lleva un valor a la memoria inicial de una tabla de memoria desde una dirección de memoria V. Cuándo el valor es agregado a la tabla todos los otros valores se corren hacia abajo 1 localización.

La instrucción se ejecutará una vez por barrido si el renglón fuera ON o verdadero. Los parámetros de la función son cargados al primer nivel del Stack del acumulador y al acumulador con dos instrucciones adicionales. Abajo están listados los pasos necesarios para programar la instrucción ATT.

Paso 1: Cargue la longitud de la tabla (cantidad de direcciones de memoria V) al primer nivel del Stack. Este parámetro debe ser un valor hexadecimal, 0 a FF.

Paso 2: Cargue la dirección de memoria V inicial de la tabla al acumulador. (Recuerde, para esta instrucción, la dirección inicial de la tabla se usa como el contador de longitud de la tabla). Este parámetro debe ser un valor hexadecimal.

Paso 3: Coloque la instrucción ATT que especifica la dirección de la memoria origen (Vaaa). Esto es, desde donde se moverá el valor.

Sugerencia:- La instrucción se ejecutará en cada barrido si el renglón fuera verdadero. Si usted no quiere que la instrucción se ejecute en más que un barrido, se debe usar una instrucción one shot (PD) en la lógica de activación.

Sugerencia: - Para parámetros que requieran valores en hexadecimal cuando se refieran a direcciones de memoria, se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

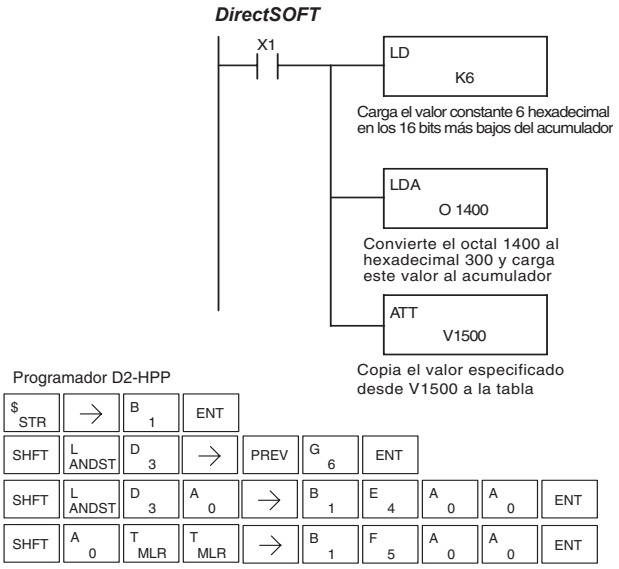
Sugerencia: - Se debe definir el valor del contador de la tabla para indicar el punto de partida de la operación. También, debe ser puesto a un valor que esté dentro de la longitud de la tabla. Por ejemplo, si la tabla es de longitud de 6 palabras, entonces el rango admisible de los valores que podrían estar en el contador de la tabla debe estar entre 1 y 6. Si el valor está fuera de este rango o es 0, los datos no se irán a la tabla. También, se debe usar una instrucción one shot (PD) de modo que el valor sólo sea colocado en un barrido y no afecte la operación de la instrucción.

Tipo de operando de datos	Rango del DL06
..... **A**	**aaa**
Memoria V V	Vea el mapa de memoria

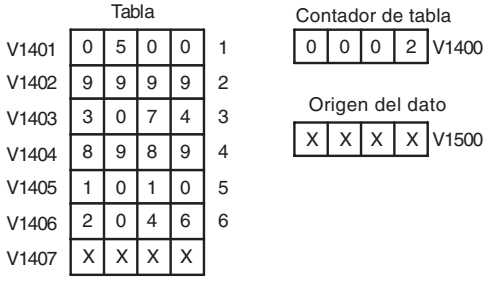
Indicadores	Descripción
SP56	ON cuando el valor corriente del contador de la tabla es igual a 0

NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP o en el fin del barrido. El puntero para esta instrucción puede comenzar en cualquier lugar en la tabla. ***No es colocado automáticamente****. Usted tiene que cargar un valor en el puntero en algún lugar en su programa.*

En el ejemplo siguiente, cuándo X1 está ON, se carga el valor constantee (K6) al acumulador usando la instrucción LD. Este valor especifica la longitud de la tabla y se coloca en la primera localización del Stack después que se ejecuta la instrucción LDA. La dirección octal 1400 (V1400), que es la dirección inicial de la tabla de destino y contador de tabla, se carga en el acumulador. La dirección de la tabla origen (V1500) es especificada en la instrucción ATT. El contador de la tabla será aumentado en "1" después que se ejecuta la instrucción.



En la instrucción ATT, el contador de la tabla determina el número de las adiciones de valores que se pueden hacer antes la instrucción pare de ejecutarse. Es útil entender cómo el sistema usa este contador para controlar la ejecución. Por ejemplo, si el contador de la tabla está colocado en 2 y la longitud de la tabla es 6 palabras, entonces allí pueden haber solamente 4 adiciones de datos antes de que la ejecución se pare. Esto puede ser calculado fácilmente por:

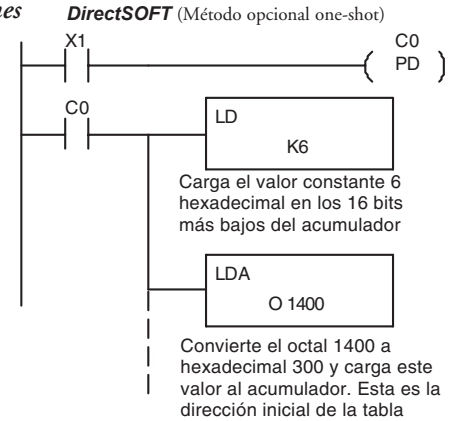


(Ejemplo: 6 - 2 = 4)

Longitud de tabla - contador de tabla = número de ejecuciones

También, el ejemplo usa un contacto normal de entrada (X1) para controlar la ejecución. Ya que el barrido es extremadamente rápido y el contador de la tabla se incrementa automáticamente, los datos pasan a la tabla muy rápidamente.

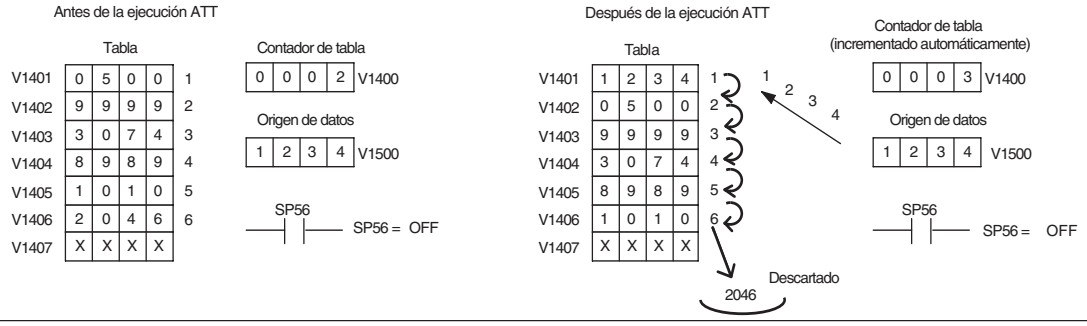
Si esto es un problema para su aplicación, tiene la opción de usar una instrucción one shot (PD) para agregar solamente un valor cada vez que el contacto hace la transición de OFF para ON.



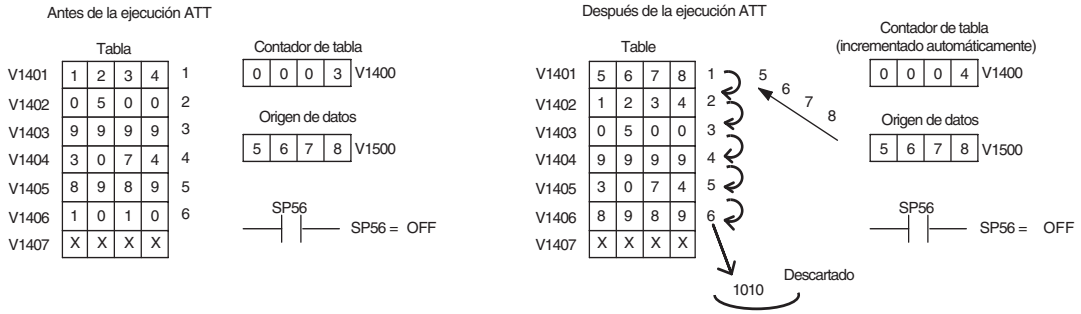
El esquema siguiente muestra barrido por barrido los resultados de la ejecución para el programa del ejemplo. El contador de la tabla es configurado como 2 inicialmente, e incrementará automáticamente de 2 hasta 6 cuando se ejecuta la instrucción. Note cómo SP56 se hace ON cuando el contador de la tabla es 6, que es igual a la longitud de la tabla. Además, aunque el ejemplo no lo muestre, asumimos que hay otra parte del programa que cambia el valor en V1500 (el origen de datos) antes de la ejecución de la instrucción de ATT.

Ejemplo de ejecución

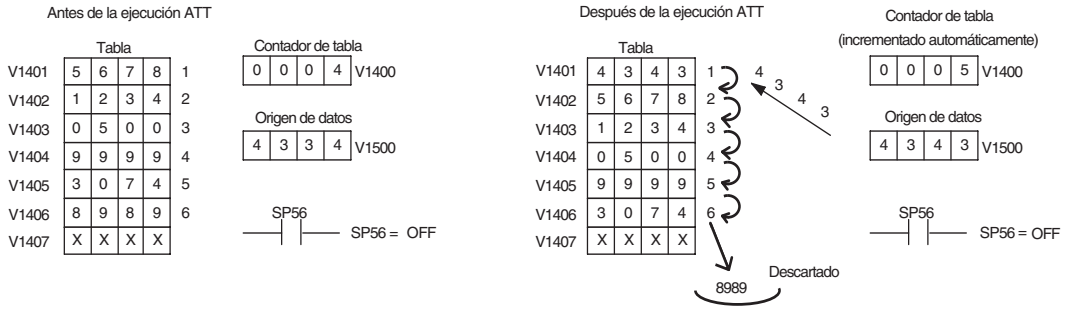
Barrido N



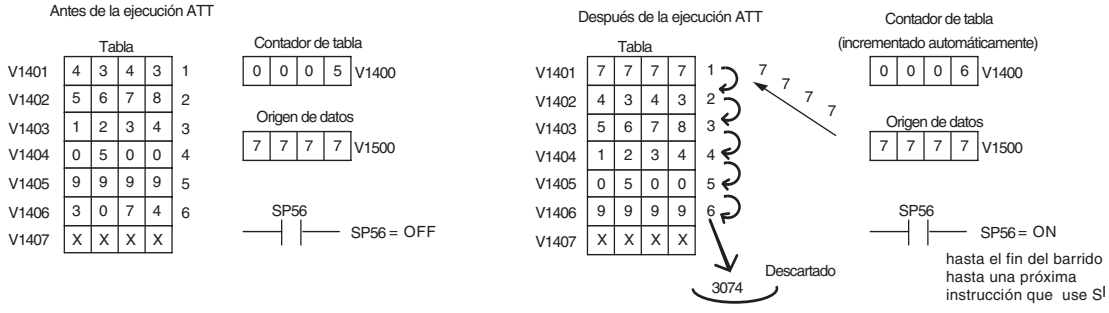
Barrido N+1



Barrido N+2



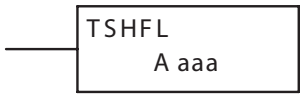
Barrido N+3



La instrucción Table Shift Left (TSHFL)

DS5	Usado
HPP	Usado

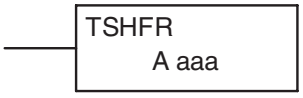
La instrucción TSHFL mueve todos los bits en una tabla de memoria a la izquierda el número especificado de posiciones de bit, esto es, desde el bit menos al más significativo.



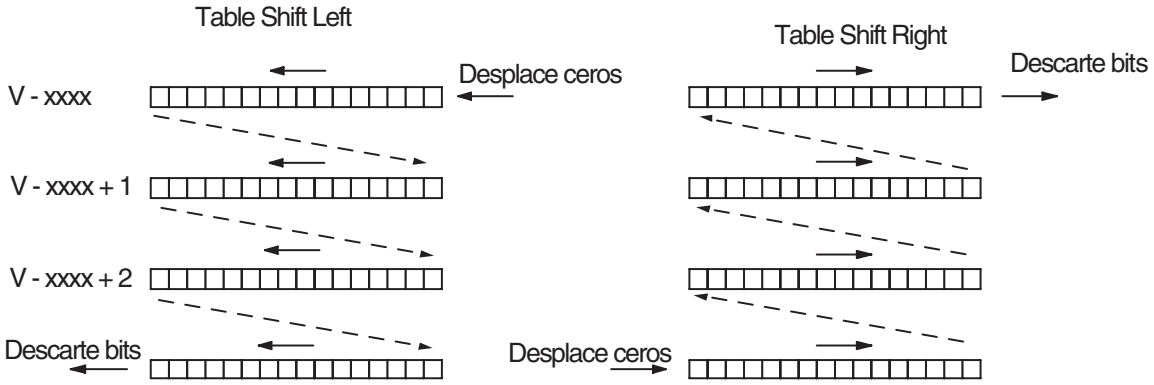
La instrucciónTable Shift Right (TSHFR)

DS5	Usado
HPP	Usado

La instrucción TSHFR mueve todos los bits en una tabla de memoria V a la derecha, un número especificado de posiciones de bit, esto es, desde el bit más al menos significativo.



La descripción siguiente se aplica a ambas instrucciones. Una tabla es solamente un rango de direcciones de memoria V. Las instrucciones TSHFL y TSHFR mueven los bits serialmente a lo largo de la tabla entera. Los bits se mueven saliendo del fin de una palabra y hasta el fin opuesto de una palabra adyacente. Al final de la tabla los bits son sacados o se desplazan ceros a la tabla. Las tablas del ejemplo debajo son arbitrariamente de cuatro palabras de largo.



Paso 1: Cargue la longitud de la tabla (cantidad de direcciones de memoria V) en el primer nivel del stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 hasta FF

Paso 2: Cargue la localización de la memoria V de inicio de la tabla al acumulador. Este parámetro debe ser un valor hexadecimal. Usted puede usar la instrucción LDA para convertir una dirección de octal a hexadecimal.

Paso 3: Coloque la instrucción que desee. Esta especifica el número de posiciones de bits que desea desplazar la tabla entera. El número de posiciones de bits debe estar en octal.

Sugerencia: — Recuerde que cada dirección de memoria V contiene 16 bits, de modo que los bits de la primera palabra de la tabla se numeran de 0 a 17 octal. Si usted quiere desplazar la tabla entera 20 bits, eso es 24 octal. SP 53 será ON si el número de bits a ser desplazado es más grande que los bits totales contenidos dentro de la tabla.

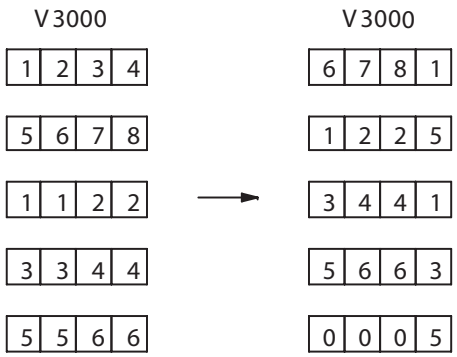
El relevador especial SP67 será colocado ON si el último bit desplazado (justamente antes de eliminarlo) es un "1.

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria

Indicadores	Descripción
SP53	ON cuando el número de bits a ser desplazados es más grande que el número de bits en la tabla
SP67	ON cuando el último bit que se desplazó es un "1" (antes de que sea eliminado)

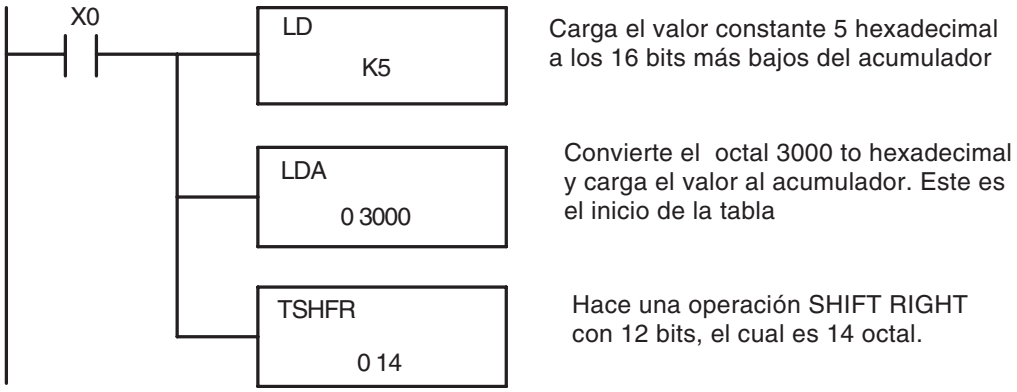
NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP. o hasta el fin del barrido.

La tabla del ejemplo contiene a la derecha los datos BCD como mostrado (para propósitos de demostración). Suponga que queremos hacer una movida de bits a la derecha de 3 dígitos BCD (12 bits). La conversión al octal de 12 bits es 14 octal. Usando instrucción TSHFR y especificando un desplazamiento a la derecha con el octal 14, tenemos la tabla resultante mostrada a la derecha. Note que se ha sacado la sucesión 2-3-4 de la secuencia y se ha desplazado la sucesión 0-0-0 en la parte inferior.



El ejemplo siguiente ladder asume que los datos en V3000 a V3004 ya existen, como mostrado arriba. Usaremos la entrada X0 para provocar la operación. Primero, cargaremos la longitud de tabla (5 palabras) al Stack del acumulador. Luego cargamos la dirección de inicio al acumulador. Ya que V3000 es un número octal lo tenemos que convertir a hexadecimal usando la instrucción LDA. Finalmente, usamos la instrucción TSHFR y especificamos el número de bits para ser desplazados (12 decimal), que es 14 octal.

DirectSOFT



Programador D2-HPP

$ STR	→	A 0	ENT								
SHFT	L ANDST	D 3	→	PREV	F 5	ENT					
SHFT	L ANDST	D 3	A 0	→	D 3	A 0	A 0	A 0	ENT		
SHFT	T MLR	SHFT	S RST	H 7	F 5	R ORN	→	NEXT	B 1	E 4	ENT

La instrucción AND Move (ANDMOV)

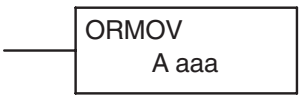
DS5	Usado
HPP	Usado

La instrucción ANDMOV copia los datos de una tabla a la dirección especificada de memoria, haciendo la operación AND de cada palabra con los datos de acumulador cuando se procesa.

La instrucción OR Move (ORMOV)

DS5	Usado
HPP	Usado

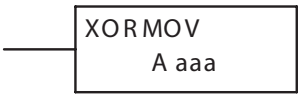
La instrucción OR MOVE copia los datos de una tabla a la dirección especificada de memoria, haciendo la operación OR de cada palabra con el valor contenido en el acumulador cuando se procesa.



La instrucción Exclusive OR Move (XORMOV)

DS5	Usado
HPP	Usado

La instrucción Exclusive OR Move copia los datos de una tabla a la dirección especificada de memoria, haciendo una operación OR exclusiva de cada palabra con el valor del acumulador cuando se procesa.



5

La descripción siguiente se aplica a las instrucciones ANDMOV, ORMOV y XORMOV.

Estas instrucciones copian los datos de una tabla a otra tabla en la localización especificada, ejecutan una operación lógica en cada palabra con el contenido del acumulador y crean entonces la otra tabla.

Paso 1: Cargue la longitud de la tabla (el número de direcciones de memoria V) al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 hasta FF.

Paso 2: Cargue la dirección inicial de la memoria V de la tabla en el acumulador. Este parámetro debe ser un valor hexadecimal. Usted puede usar la instrucción LDA para convertir una dirección octal a uno hexadecimal.

Paso 3: Cargue el valor BCD/hexadecimal que expresa el conjunto de bits a ser el operando AND en el acumulador que será combinado lógicamente con el contenido de la tabla durante la operación.

Paso 4: Coloque una de las instrucciones ANDMOV, ORMOV o XORMOV. La que sea escogida especifica la dirección inicial de la copia de la tabla original. Esta tabla nueva será automáticamente de la misma longitud que la tabla original.

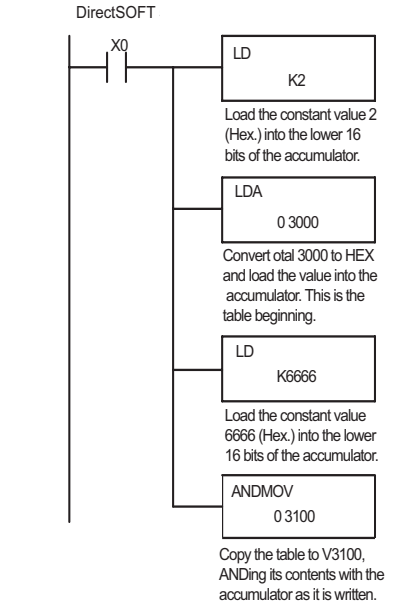
Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria

La tabla del ejemplo contiene a la derecha los datos BCD como mostrado (para propósitos de demostración). Suponga que queremos mover una tabla de dos palabras localizada en V3000 y hacer la operación AND con K6666. La copia de la tabla en V3100 muestra el resultado de la operación AND con cada palabra.

El programa en esta página realiza el ejemplo de la operación de ANDMOV anterior. Asume que los datos en la tabla en V3000 - V3001 ya existen. Primero cargamos la longitud de la tabla (dos palabras) en el acumulador. Luego cargamos la dirección de inicio de la tabla origen, usando la instrucción LDA. Luego cargamos los datos en el acumulador para ser operados AND con la tabla. En la instrucción ANDMOV se especifica el destino de la tabla, V3100.

Programador D2-HPP

$ STR	→	A 0	ENT						
SHFT	L ANDST	D 3	→	PREV	C 2	ENT			
SHFT	L ANDST	D 3	A 0	→	D 3	A 0	A 0	A 0	ENT
SHFT	L ANDST	D 3	→	PREV	G 6	G 6	G 6	G 6	ENT
V AND	SHFT	M ORST	O INST#	V AND	→	D 3	B 1	A 0	A 0	ENT

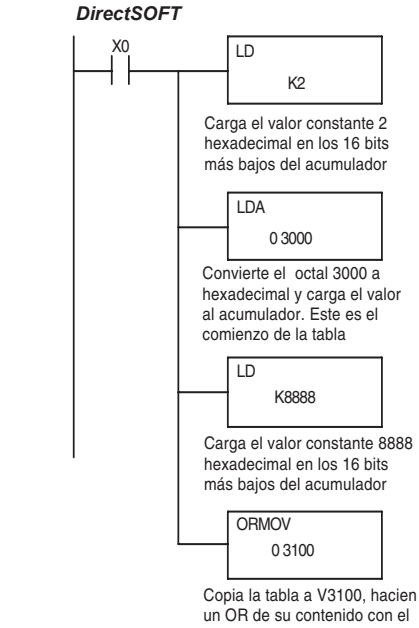


El ejemplo de la derecha muestra una tabla de dos palabras en V3000 y lógica OR con K8888. La copia de la tabla en V3100 muestra el resultado de la operación OR con cada palabra.

El programa de abajo realiza el ejemplo de ORMOV arriba. Asume que los datos en la tabla en V3000 - V3001 ya existen. Primero cargamos la longitud de la tabla (dos palabras) en el acumulador. Luego cargamos la dirección de inicio de la tabla fuente, usando la instrucción LDA. Luego cargamos los datos en el acumulador para ser operados OR con la tabla. En la instrucción ORMOV se especifica el destino de la tabla, V3100.

Programador D2-HPP

$ STR	→	A 0	ENT						
SHFT	L ANDST	D 3	→	PREV	C 2	ENT			
SHFT	L ANDST	D 3	A 0	→	D 3	A 0	A 0	A 0	ENT
SHFT	L ANDST	D 3	→	PREV	I 8	I 8	I 8	I 8	ENT
Q OR	SHFT	M ORST	O INST#	V AND	→	D 3	B 1	A 0	A 0	ENT



El ejemplo a la derecha muestra una tabla de dos palabras en V3000 y hace una operación XOR lógico con K3333. La copia de la tabla en V3100 muestra el resultado de la operación XOR para cada palabra.

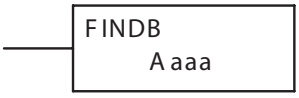
El ejemplo del programa ladder para el XORMOV es similar al de arriba para el ORMOV. Use sin emabargo la instrucción XORMOV.

En el programador usted debe usar la tecla SHFT y deletrear "XORMOV" explícitamente..

La instrucción Find Block (FINDB)

DS5	Usado
HPP	N/A

La instrucción FINFB busca una ocurrencia de un bloque especificado de valores en una tabla de memoria V. Los parámetros de la instrucción son cargados al primer y segundo nivel del Stack del acumulador y el acumulador por tres instrucciones adicionales. Si el bloque se encuentra, su dirección inicial se almacenará en el acumulador. Si el bloque no se encuentra, el relevador especial SP53 se hará ON.



Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V. V	Vea el mapa de memoria
Puntero. . . . . P	Vea el mapa de memoria

Indicadores	Descripción
SP56	ON cuando la instrucción FINDB fue ejecutada pero no encontró el bloque de datos

NOTA: Las indicaciones de estado discretas SP son válidas solamente hasta que se ejecute otra instrucción que use los mismos relevadores especiales SP.

Los pasos necesarios para programar la instrucción **FINDB** están listados abajo.

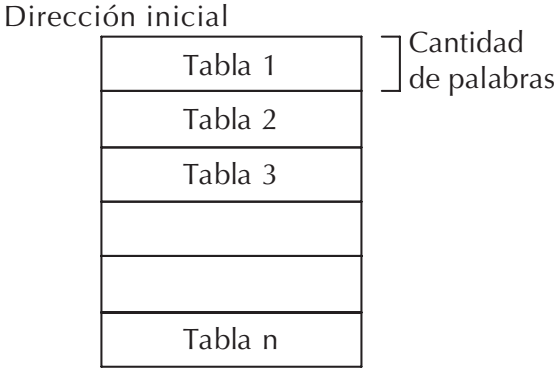
Paso 1: Cargue el número de byte en el bloque a ser localizado. Este parámetro debe ser un valor hexadecimal, 0 a FF, que es 255 decimal..

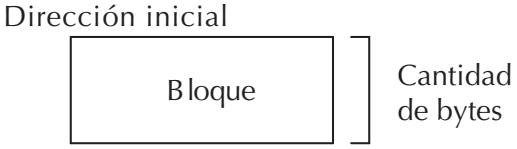
Paso 2: Cargue la longitud de una tabla (el número de palabras) a ser buscada. FINDB buscará múltiples tablas que están adyacentes en la memoria V. Este parámetro debe ser un valor hexadecimal, 0 hasta FF.

Paso 3: Cargue la localización final para todas las tablas en el acumulador. Este parámetro debe ser un valor hexadecimal. Usted puede usar la instrucción LDA para convertir una dirección octal a hexadecimal.

Paso 4: Cargue la dirección inicial de la tabla para todas las tablas en el acumulador. Este parámetro debe ser un valor hexadecimal.

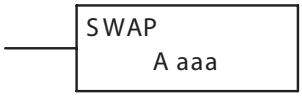
Paso 5: Coloque la instrucción FINDB. Esta especifica la dirección inicial del bloque de los datos que usted trata de localizar.





La instrucción Swap (SWAP)

DS5	Usado
HPP	Usado



Esta instrucción SWAP intercambia datos en dos tablas de igual longitud.

Paso 1: Cargue la longitud de las tablas (la cantidad de direcciones de memoria V) al primer nivel del Stack del acumulador. Este parámetro debe ser un valor hexadecimal, 0 hasta FF. Recuerde que las tablas deben ser de igual longitud.

Paso 2: Cargue la dirección de la memoria V de inicio de la primera tabla al acumulador. Este parámetro debe ser un valor hexadecimal. Usted puede usar la instrucción de LDA para convertir una dirección octal a hexadecimal.

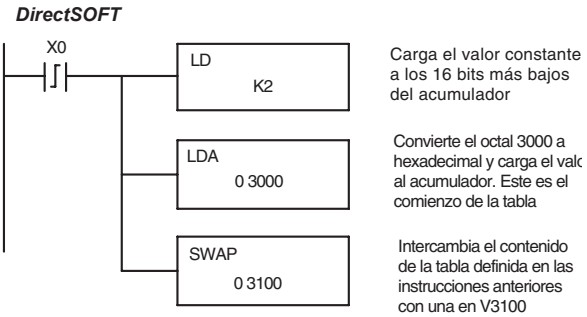
Paso 3: Coloque la instrucción SWAP. Esta especifica la dirección inicial de la segunda tabla. Este parámetro debe ser un valor hexadecimal.

Sugerencia: —- El intercambio de datos ocurre dentro de un solo barrido. Si la instrucción ejecuta en múltiples barridos, será difícil de saber el contenido real de cualquier tabla en algún tiempo particular. De modo que solo ejecuta esta instrucción en un solo barrido.

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria

El ejemplo a la derecha muestra una tabla de 2 palabras comenzando en V3000. Haremos la función SWAP con otra tabla de 2 palabras comenzando en V3100. El programa ladder para esto es mostrado abajo.

El programa del ejemplo adyacente usa un contacto PD (dispara por un barrido en la transición de OFF para ON de X0). Primero, cargamos la longitud de las tablas (dos palabras) al acumulador. Luego cargamos la dirección de la primera tabla (V3000) en el acumulador usando la instrucción LDA, convirtiendo la dirección de octal a hexadecimal. **Note que no importa cuál tabla declaramos "primero", porque los resultados de intercambio serán los mismos.**



Programador D2-HPP

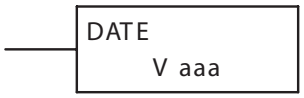
$ STR	SHFT	P CV	D 3	→	A 0	ENT							
SHFT	L ANDST	D 3	→	PREV	C 2	ENT							
SHFT	L ANDST	D 3	A 0	→	D 3	A 0	A 0	A 0	ENT				
SHFT	S RST	SHFT	W ANDN	A 0	P CV	→	D 3	B 1	A 0	A 0	ENT		

Instrucciones de fecha y hora

La instrucción Date (DATE)

DS5	Usado
HPP	Usado

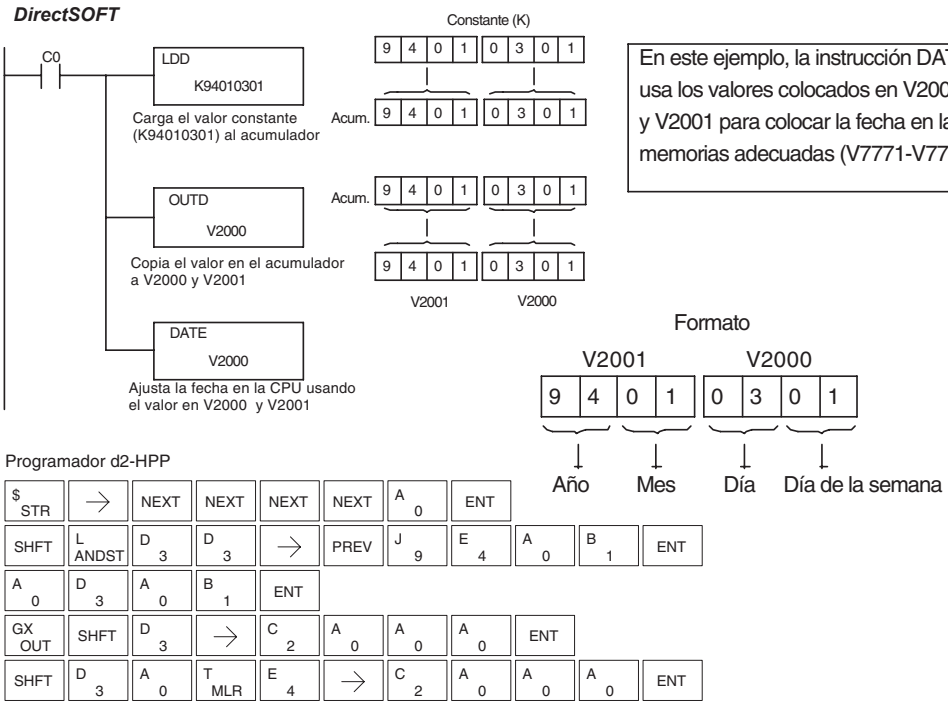
La instrucción DATE puede ser usada para poner la fecha en la CPU. La instrucción requiere dos direcciones consecutivas de memoria V (Vaaa) para ajustar la fecha. Si los valores especificados en las direcciones no son válidos, la fecha no se ajustará en la CPU. La fecha actual se puede leer desde 4 memorias V consecutivas V (V7771 hasta V7774).



En el ejemplo siguiente, cuándo C0 está ON, el valor constantee (K94010301) es cargado en el acumulador usando la instrucción LDD (C0 debe ser un contacto de una instrucción One Shot (PD)). El valor en el acumulador es copiado a V2000 usando la instrucción OUTD. La instrucción DATE usa el valor en V2000 para ajustar la fecha en la CPU.

Fecha	Rango	Memoria V(BCD) (Sólo para lectura)
Añ	0-99	V7774
Mes	1-12	V7773
Día	1-31	V7772
Día de la semana	0-06	V7771
Los valores a entrar como día de la semana son: 0=Domingo, 1=Lunes, 2=Martes, 3=Miércoles, 4=Jueves, 5=Viernes, 6=Sábado		

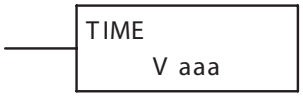
Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria



La instrucción Time (TIME)

DS5	Usado
HPP	Usado

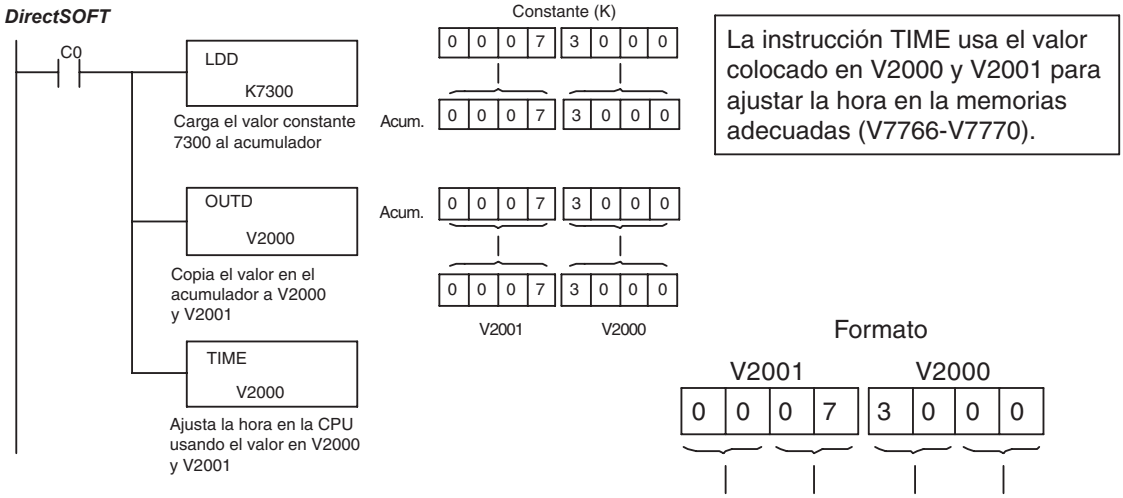
La instrucción TIME se puede usar para ajustar la hora (24 horas) en la CPU. La instrucción requiere dos direcciones consecutivas de memoria V (Vaaa) que se usan para ajustar la hora, minutos y segundos. Si los valores en las direcciones especificadas no son válidos, el tiempo no se ajustará. El tiempo actual se puede leer en las direcciones de memoria V7747 y V7766-V7770.



Hora	Rango	Dirección de Memoria V (BCD) (Sólo para lectura)
1/100 segundos (10ms)	0-99	V7747
Segundos	0-59	V7766
Minutos	0-59	V7767
Hora	0-23	V7770

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria

En el ejemplo siguiente, cuándo C0 está ON, se carga el valor constantee (K73000) al acumulador usando la instrucción LDD. (C0 debe ser un contacto de una instrucción one shot (PD)). El valor en el acumulador es copiado a V2000 usando la instrucción OUTD. La instrucción TIME usa el valor en V2000 para ajustar la hora en la CPU.



Programador D2-HPP

$ STR	→	NEXT	NEXT	NEXT	NEXT	A 0	ENT
SHFT	L ANDST	D 3	D 3	→	PREV	H 7	D 3	A 0	A 0	A 0	ENT
A 0	D 3	A 0	B 1	ENT							
GX OUT	SHFT	D 3	→	C 2	A 0	A 0	A 0	ENT			
SHFT	T MLR	SHFT	I 8	M ORST	E 4	→	C 2	A 0	A 0	A 0	ENT

Instrucciones de control de la CPU

La instrucción No Operation (NOP)

DS5	Usado
HPP	Usado

La instrucción NOP es una dirección de memoria vacía (no programada) .

DirectSOFT

Programador D2-HPP

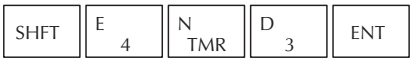
La instrucción End (END)

DS5	Usado
HPP	Usado

La instrucción END marca el punto de terminación del barrido del programa normal. ES NECESARIO COLOCAR una instrucción END al fin del cuerpo principal del programa. Si se omite la instrucción END ocurrirá un error y la CPU no entrará en Modo Run. Las etiquetas de datos, los programas de subrutina s e interrupción se colocan después la instrucción END. La instrucción END no es condicional; por lo tanto, no se coloca ningún contacto de entrada.

DirectSOFT

Porgramador D2-HPP



La instrucción Stop (STOP)

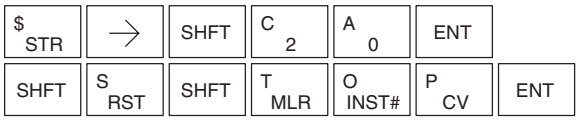
DS5	Usado
HPP	Usado

La instrucción STOP cambia el modo operacional de la CPU, del modo RUN a Program (STOP). Esta instrucción se usa típicamente para parar la operación del PLC en una condición de error.

En el ejemplo siguiente, cuándo C0 prende, la CPU detiene la operación y cambia el modo a Program.

DirectSOFT

Programador D2-HPP



Indicadores	Descripción
SP16	On cuando el PLC DL06 pasa al modo TERM_PRG
SP53	On cuando la instrucción STOP es ejecutada..

La instrucción Reset Watch Dog Timer (RSTWT)

DS5	Usado
HPP	Usado

La instrucción RSTWT coloca el temporizador de barrido de la CPU a 0. El ajuste original del temporizador de watchdog es 200 milisegundos.

—(RSTWT)

Watch dog timer es un temporizador que supervisa que el barrido no exceda el valor prefijado. El barrido de la CPU muy rara vez excede 200 ms, pero es posible que suceda.

Los lazos For/Next, subrutinas, rutinas de interrupción e instrucciones de tablas se pueden programar de tal forma que puede ser que el barrido llegue a ser más largo que 200 ms.

Cuándo se usa una o más instrucciones de una forma que podría exceder el watch dog timer, esta instrucción se puede usar para reponer este temporizador.

Un error (E003 de tiempo muerto de software) ocurrirá y la CPU entrará el modo de programa si el tiempo de barrido excede el valor prefijado en este temporizador . Es muy importante la colocación de la instrucción RSTWT en el programa.

La instrucción se tiene que ejecutar antes que el tiempo de barrido exceda el ajuste del watch dog timer.

Si el tiempo de barrido es continuamente más largo que el temporizador watchdog, el valor de tiempo muerto se puede aumentar permanentemente del valor normal de 200 ms con la función auxiliar apropiada en su paquete de programación. Esto elimina la necesidad de la instrucción RSTWT.

En el ejemplo siguiente el temporizador watchdog de la CPU será repuesto a 0 cuando la instrucción de RSTWT se ejecuta. Vea la instrucción For/Next para un ejemplo detallado.

DirectSOFT

Programador D2-HPP

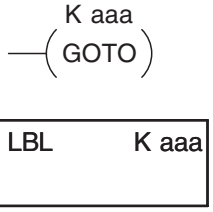
SHFT	R ORN	S RST	T MLR	W ANDN	T MLR	ENT

Instrucciones de control de programa

La instrucción Goto Label (GOTO) (LBL)

DS5	Usado
HPP	Usado

Estas instrucciones se saltan todas instrucciones entre el Goto y la instrucción correspondiente de LBL. El valor del operando para el Goto y la instrucción correspondiente de LBL es el mismo. La lógica entre Goto y la instrucción de LBL no se ejecuta cuando la instrucción de Goto se habilita. Pueden ser usadas hasta 256 instrucciones de Goto y 256 instrucciones de LBL en el programa.

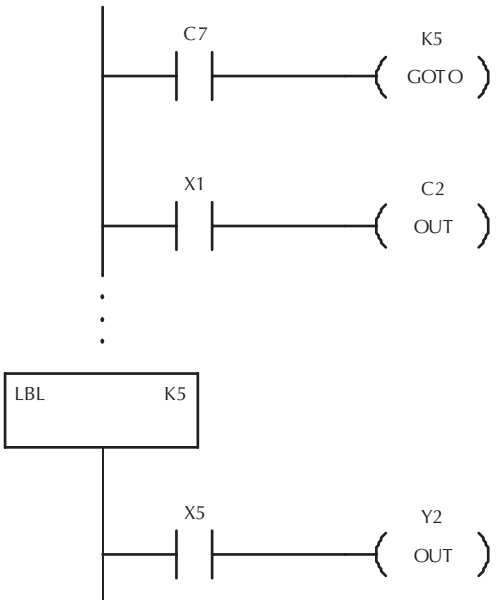


Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . K	1-FFFF

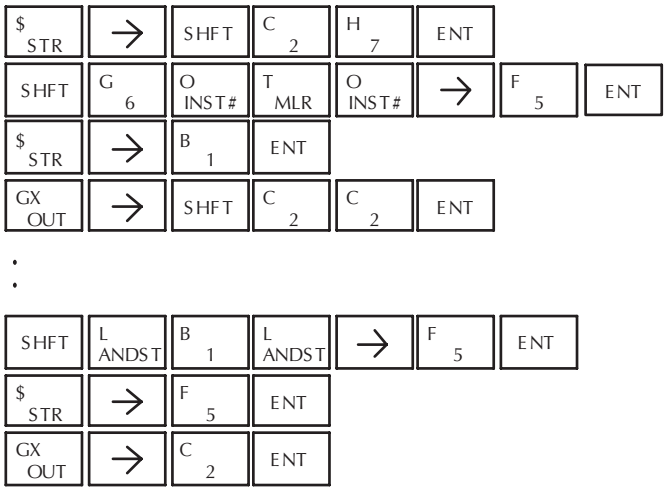
5

En el ejemplo siguiente, cuándo C7 está ON, se saltará toda la lógica del programa entre el GOTO y la instrucción correspondiente de LBL (designado con el mismo valor constantee de Kaaa). Las instrucciones a ser saltadas no serán ejecutadas por la CPU.

DirectSOFT



Programador D2-HPP

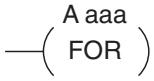


La instrucción For / Next (FOR) (NEXT)

DS5	Usado
HPP	Usado

Las instrucciones FOR y NEXT se usan para ejecutar una sección de la lógica ladder entre la instrucción FOR y NEXT un número de veces especificado.

Cuándo la instrucción FOR es activada, el programa se ejecutará el número de veces especificado en esa sección del programa. Si la instrucción FOR no es energizada no se ejecutan las instrucciones en la sección de la lógica ladder entre el FOR y NEXT.



Las instrucciones FOR/ NEXT no se pueden anidar, es decir, no se puede usar una instrucción dentro de otra. La actualización normal de entradas y salidas y el trabajo de la CPU se suspende al ejecutar el lazo FOR/NEXT.

El barrido del programa puede aumentar significativamente, dependiendo de la cantidad de tiempo que tome para ejecutarse la lógica entre las instrucción FOR/NEXT.

Con la excepción de instrucciones inmediatas de entradas y salidas, las entradas y salidas no se actualizarán hasta que la ejecución del programa se complete para ese barrido.

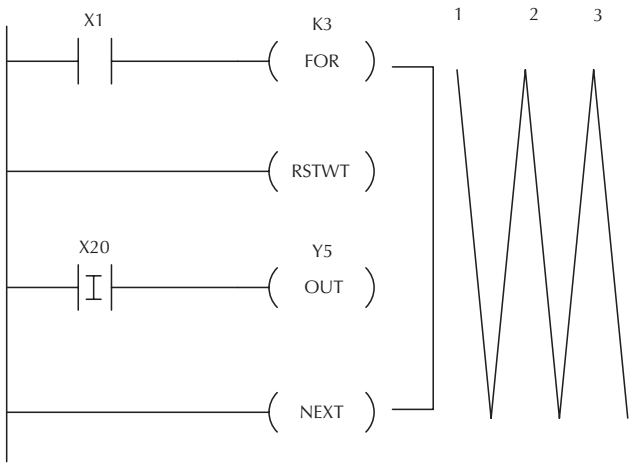
Dependiendo del plazo de tiempo requerido para completar la ejecución del programa, puede ser necesario usar la instrucción de RSTWT dentro del lazo FOR/NEXT.

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V. V	Vea el mapa de memoria
Constante . . . . . K	1-9999

En el ejemplo siguiente, cuándo X1 está ON, el programa de aplicación dentro del lazo FOR/NEXT se ejecutará tres veces. Si X1 está apagado el programa dentro del lazo no se ejecutará. Las instrucciones inmediatas pueden o no pueden ser necesarias dependiendo de su aplicación. También, la instrucción RSTWT no es necesaria si el lazo FOR/NEXT no extiende el tiempo de barrido más de lo que esté ajustado el Watch dog timer.

Para más información del Watch dog timer, vea la instrucción RSTWT.

DirectSOFT



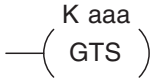
Programador D2-HPP

$ STR	→	B 1	ENT			
SHFT	F 5	O INST#	R ORN	→	D 3	ENT
SHFT	R ORN	S RST	T MLR	W ANDN	T MLR	ENT
$ STR	SHFT	I 8	→	C 2	A 0	ENT
GX OUT	→	F 5	ENT			
SHFT	N TMR	E 4	X SET	T MLR	ENT	

La instrucción Goto Subroutine (GTS) (SBR)

DS5	Usado
HPP	Usado

La instrucción de GOTO Subrutine permite que una sección de la lógica ladder sea colocada fuera del cuerpo principal del programa y ejecutada sólo cuando sea necesario. Puede haber un máximo de 256 instrucciones de GTS y 256 instrucciones de SBR usados en un programa. Las instrucciones de GTS se pueden anidar hasta 8 niveles. Un error E412 ocurrirá si se exceden los límites máximos.



Típicamente esto se usará en una aplicación donde un bloque de lógica del programa puede ser lento de ejecutar y no es necesario ejecutar las instrucciones en cada barrido. El LABEL de la subrutina y toda la lógica asociada se colocan después la declaración END en el programa. Cuándo la subrutina es llamada desde el programa principal, la CPU ejecutará la subrutina (SBR) con el mismo número (K) constantee que la instrucción de GTS que llamó la subrutina.

El código en una subrutina es ejecutado solamente cuando sea necesario ya que va después de la instrucción END. El código que no es ejecutado no afecta el tiempo de barrido completo del programa.

Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . K	1-FFFF

La instrucción Subroutine Return (RT)

DS5	Usado
HPP	Usado

Cuándo se ejecuta una instrucción RT en la subrutina, la CPU volverá al punto en el cuerpo principal del programa de donde se llamó la subrutina. Esta instrucción se usa como terminación de la subrutina, que debe ser la última instrucción en la subrutina y es una instrucción incondicional (no hay ningún contacto de entrada en el renglón).

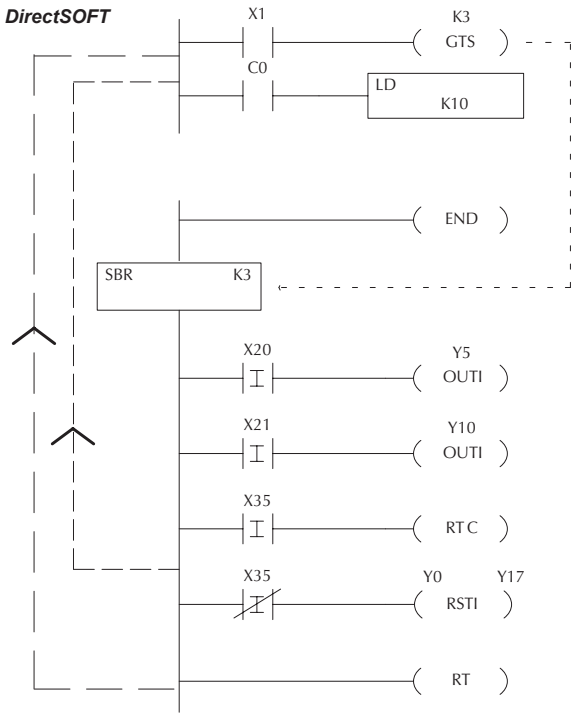
La instrucción Subroutine Return Conditional (RTC)

DS5	Usado
HPP	Usado

La instrucción RTC es una instrucción opcional usada con un contacto de entrada para implementar un regreso condicional de la subrutina. Se necesita aún la instrucción RT para terminación de la subrutina.

En el ejemplo siguiente, cuándo X1 está ON, se llamará la Subrutina K3. La CPU saltará al Label K3 de la Subrutina y se ejecutará la lógica ladder en la subrutina.

Si X35 está ON la CPU volverá al programa principal con la instrucción de RTC. Si X35 no está ON, Y0-Y17 será vuelto a OFF y luego la CPU volverá al cuerpo principal del programa.



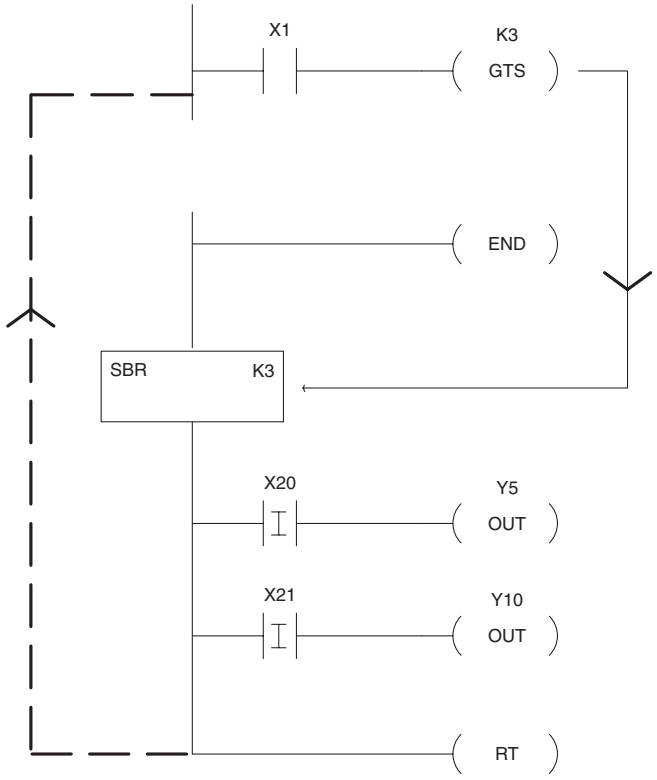
Programador D2-HPP

$ STR	→	B 1	ENT				
SHFT	G 6	T MLR	S RST	→	D 3	ENT	

SHFT	E 4	N TMR	D 3	ENT			
SHFT	S RST	SHFT	B 1	R ORN	→	D 3	ENT
$ STR	SHFT	I 8	→	C 2	A 0	ENT	
GX OUT	SHFT	I 8	→	F 5	ENT		
$ STR	SHFT	I 8	→	C 2	B 1	ENT	
GX OUT	SHFT	I 8	→	B 1	A 0	ENT	
$ STR	SHFT	I 8	→	D 3	F 5	ENT	
SHFT	R ORN	T MLR	C 2	ENT			
SP STRN	SHFT	I 8	→	D 3	F 5	ENT	
S RST	SHFT	I 8	→	A 0	→	B 1	H 7	ENT
SHFT	R ORN	T MLR	ENT				

En el ejemplo siguiente, cuándo X1 está ON, se llamará la Subrutina K3. La CPU saltará al Label K3 de la subrutina y se ejecutará la lógica ladder en la subrutina. La CPU volverá al cuerpo principal del programa después que se ejecuta la instrucción RT.

DirectSOFT



Programador D2-HPP

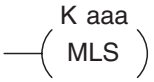
$ STR	→	B 1	ENT				
SHFT	G 6	T MLR	S RST	→	D 3	ENT	

SHFT	E 4	N TMR	D 3	ENT			
SHFT	S RST	SHFT	B 1	R ORN	→	D 3	ENT
$ STR	SHFT	I 8	→	C 2	A 0	ENT	
GX OUT	→	F 5	ENT				
$ STR	SHFT	I 8	→	C 2	B 1	ENT	
GX OUT	→	B 1	A 0	ENT			
SHFT	R ORN	T MLR	ENT				

La instrucción Master Line Set (MLS)

DS5	Usado
HPP	Usado

La instrucción MLS permite que el programa controle las secciones de lógica ladder formando un nuevo riel de energía controlado por el riel principal izquierdo de energía. El riel principal izquierdo es siempre la línea maestra 0. Cuándo se usa una instrucción de MLS K1, se crea un riel nuevo de energía en el nivel 1. Las instrucciones MLS y MLR pueden ser usadas para anidar rieles de energía de hasta siete niveles de profundidad.



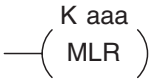
Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . K	1-FFFF

5

La instrucción Master Line Reset (MLR)

DS5	Usado
HPP	Usado

Las instrucción MLR marca en final del control de la instrucción correspondiente MLS. La referencia MLR es una menos que el de la instrucción correspondiente MLS.



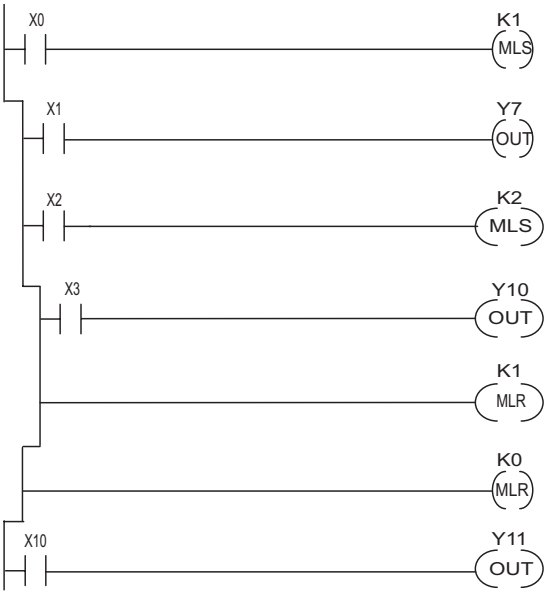
Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . K	1-FFFF

Entendiendo relevadores de control maestros (Master Line)

Las instrucciones MLS y MLR permiten activar o desactivar rápidamente secciones de un programa ladder. Esta característica le proporciona flexibilidad del control al programa.

El ejemplo siguiente muestra cómo las instrucciones MLS y MLR operan creando un riel de energía secundario en la lógica de control.

DirectSOFT



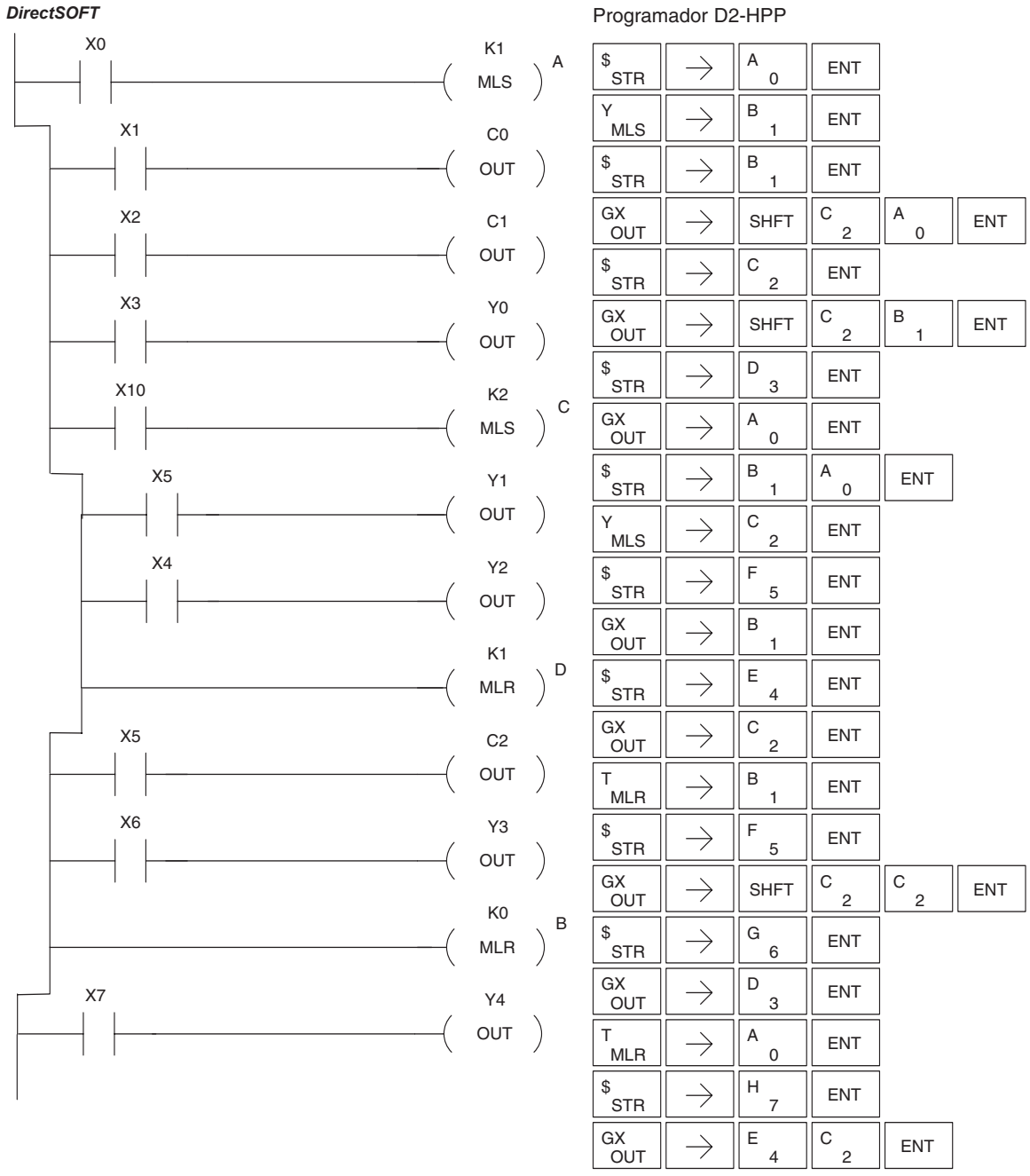
Cuando el contacto X0 está ON, será ejecutada la lógica debajo de la primera instrucción MLS.

Cuando los contactos Xo y X2 están ON, será ejecutada la lógica debajo de la segunda instrucción MLS.

Las instrucciones MLR indica el final del área de control maestra.

Ejemplo de MLS/MLR

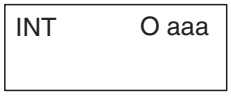
En el programa del ejemplo siguiente con MLS/MLR, la lógica funcionará entre el primer MLS K1 (A) y MLR K0 (B) sólo si la entrada X0 está ON. La lógica entre el MLS K2 (C) y MLR K1 (D) funcionará sólo si la entrada X10 y X0 están ON . El último renglón no es controlado por ninguna de las bobinas MLS.



Instrucciones de acción de interrupción

La instrucción Interrupt (INT)

DS5	Usado
HPP	Usado



La instrucción INT permite que sea colocada una sección de lógica ladder debajo del cuerpo principal del programa y ejecutada sólo cuando sea necesario. Los modos HSIO de alta velocidad de entradas y salidas 10, 20, y 40 pueden engendrar una interrupción. Con el modo 40, usted puede escoger una interrupción externa (la entrada X0) o una interrupción basada en tiempo (entre 3-999 ms).

Típicamente, las interrupciones se usan en una aplicación cuando se necesita una respuesta rápida a una entrada o cuando se debe ejecutar una sección de programa más rápido que el barrido normal de la CPU. La instrucción etiqueta de interrupción (Interrupt label) y toda la lógica asociada se deben colocar después de la declaración END en el programa. Cuándo ocurre una interrupción, la CPU completará la ejecución de la instrucción que se está procesando en la lógica ladder y luego ejecuta la rutina de interrupción. Después de la ejecución de la rutina de interrupción el programa ladder reanuda del punto en que se interrumpió.

Vea la sección de operación del modo 40 (interrupción) para más detalles en la configuración de interrupción. En el DL06, sólo hay disponible una interrupción de software. La interrupción de software usa el interrupt # 00 (INT 0), que significa que el hardware interrupt #0 y el software interrupt no se pueden usar juntas. Las interrupciones de hardware se marcan en octal para corresponder con la señal de la entrada de hardware (Por ejemplo, X1 iniciará INT 1).

Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . O	1-FFFF

La instrucción Interrupt Return (IRT)

DS5	Usado
HPP	Usado

La instrucción IRT se ejecuta normalmente como la última instrucción en la rutina de interrupción. Vuelve la CPU al punto en el programa principal de donde se llamó. IRT es una instrucción incondicional (no se necesita contacto de entrada en el renglón).

La instrucción Interrupt Return Conditional (IRTC)

DS5	Usado
HPP	Usado

IRTC es una instrucción opcional usada con un contacto de entrada para causar un regreso condicional de la rutina de interrupción. En todo caso se requiere IRT para terminar la rutina de interrupción.

La instrucción Enable Interrupts (ENI)

DS5	Usado
HPP	Usado

La instrucción ENI se coloca en el programa principal ladder (antes de la instrucción END) para posibilitar la interrupción. La interrupción permanece habilitada hasta que el programa ejecute una instrucción DISI.

La instrucción Disable Interrupts (DISI)

DS5	Usado
HPP	Usado

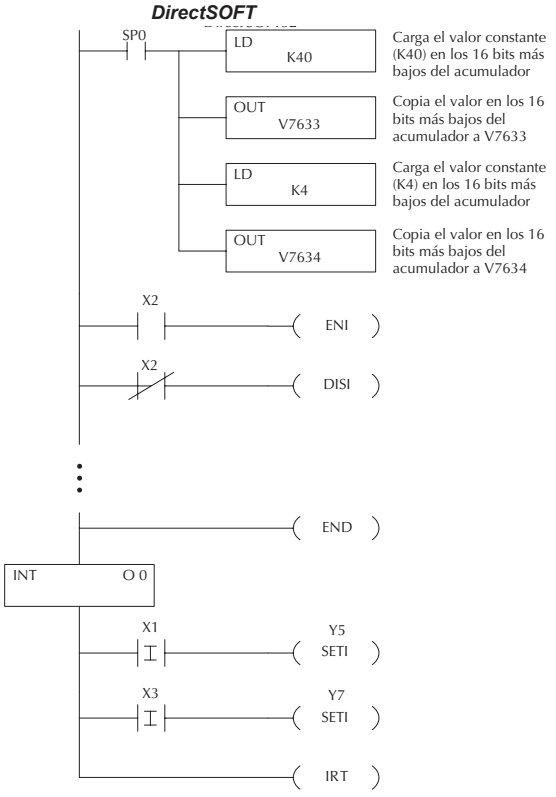
Una instrucción DISI en el cuerpo principal del programa de aplicación (antes la instrucción END) incapacitará la interrupción (ya sea externa o por tiempo). La interrupción permanece incapacitada hasta que el programa ejecute una instrucción ENI.

Ejemplo de programa de interrupción externa

En el ejemplo siguiente, se hace una inicialización en el primer barrido usando el contacto de primer barrido SP0. La característica de interrupción es el modo HSIO 40. Luego se configura X0 como interrupción externa escribiendo al registro de configuración correspondiente, V7634. Vea la operación del modo 40 en el capítulo 3 para más detalles.

Durante la ejecución del programa, se activa la interrupción cuándo X2 está ON. Cuándo X2 está apagado se incapacitará la interrupción. Cuándo ocurre una señal de interrupción (X0) la CPU saltará a la marca de interrupción INT O0.

Se realizará entonces la lógica de aplicación en la rutina de interrupción. La CPU volverá al cuerpo principal del programa después que se ejecuta la instrucción IRT.



Programador D2-HPP

$ STR	→	SHFT	SP STRN	A 0	ENT				
SHFT	L ANDST	D 3	→	SHFT	K JMP	E 4	A 0	ENT	
GX OUT	→	SHFT	V AND	H 7	G 6	D 3	D 3	ENT	
SHFT	L ANDST	D 3	→	SHFT	K JMP	E 4	ENT		
GX OUT	→	SHFT	V AND	H 7	G 6	D 3	E 4	ENT	
$ STR	→	C 2	ENT						
SHFT	E 4	N TMR	I 8	ENT					
SP STRN	→	C 2	ENT						
SHFT	D 3	I 8	S RST	I 8	ENT				

SHFT	E 4	N TMR	D 3	ENT		
SHFT	I 8	N TMR	T MLR	→	A 0	ENT
$ STR	SHFT	I 8	→	B 1	ENT	
X SET	SHFT	I 8	→	F 5	ENT	
$ STR	SHFT	I 8	→	D 3	ENT	
X SET	SHFT	I 8	→	H 7	ENT	
SHFT	I 8	R ORN	T MLR	ENT		

Ejemplo de programa de interrupción por tiempo

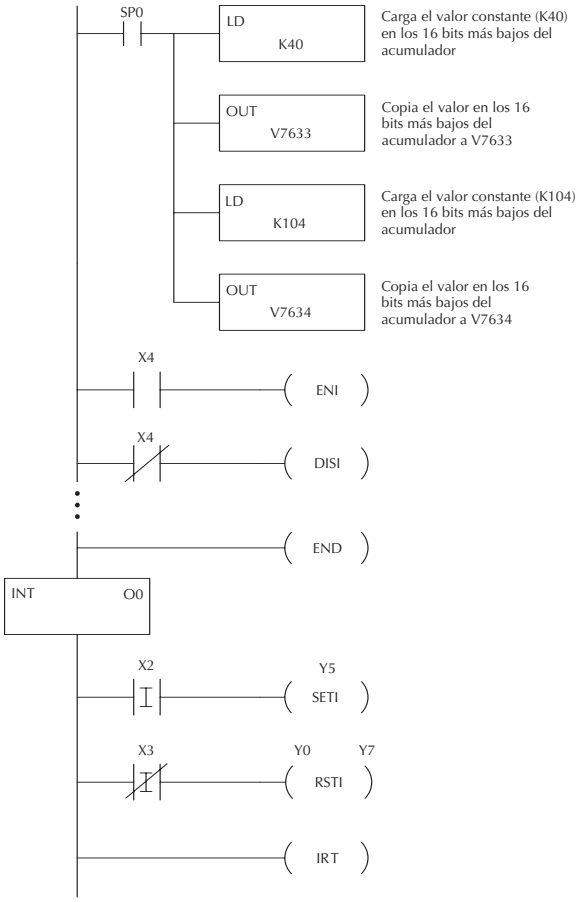
En el ejemplo siguiente, se hace una inicialización en el primer barrido, usando el contacto SP0 de primer barrido. La característica de interrupción es el modo HSIO 40. Luego se configura el temporizador de HSIO como una interrupción de 10 ms escribiendo K104 al registro de configuración para X0 (V7634).

Vea la operación del modo 40 en el capítulo 3 para más detalles. Cuándo X4 prende, la interrupción se habilitará.

Cuándo X4 se apaga, la interrupción se incapacitará. Cada 10 ms la CPU saltará a la marca de interrupción INT O0. Se ejecutará la lógica de aplicación en la rutina de interrupción.

Si X3 no está ON Y0-Y7 será colocado OFF y luego la CPU volverá al cuerpo principal del programa.

DirectSOFT



Programador D2-HPP

$ STR	→	B 1	ENT					
SHFT	L ANDST	D 3	→	SHFT	K JMP	E 4	A 0	ENT
GX OUT	→	SHFT	V AND	H 7	G 6	D 3	D 3	ENT
SHFT	L ANDST	D 3	→	SHFT	K JMP	B 1	A 0	E 4	ENT
GX OUT	→	SHFT	V AND	H 7	G 6	D 3	E 4	ENT
$ STR	→	E 4	ENT					
SHFT	E 4	N TMR	I 8	ENT				
SP STRN	→	E 4	ENT					
SHFT	D 3	I 8	S RST	I 8	ENT			

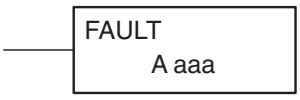
SHFT	E 4	N TMR	D 3	ENT			
SHFT	I 8	N TMR	T MLR	→	A 0	ENT	
$ STR	SHFT	I 8	→	C 2	ENT		
X SET	SHFT	I 8	→	F 5	ENT		
SP STRN	SHFT	I 8	→	D 3	ENT		
X SET	SHFT	I 8	→	A 0	→	H 7	ENT
SHFT	I 8	R ORN	T MLR	ENT			

Instrucciones de mensajes

La instrucción Fault (FAULT)

DS5	Usado
HPP	Usado

La instrucción Fault se usa para mostrar un mensaje de FALLA en un programador portátil, o el visor opcional LCD o en el menú PLC>Diagnostics> messages> Fault messages en *Direct*SOFT. El mensaje tiene un máximo de 23 caracteres y puede ser datos de memoria V, datos o constantes numéricas o texto ASCII.



Para mostrar un valor en una dirección de memoria V, especifique la dirección de memoria V en la instrucción. Para mostrar los datos en las instrucciones ACON (constante ASCII) o NCON (constante numérica), especifique el valor de la constante (K) para el área correspondiente de etiqueta de datos (Data Label).

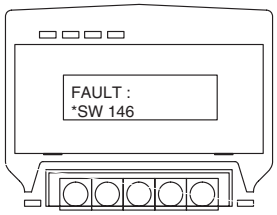
5

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V. V	Vea el mapa de memoria
Constante . . . . . K	1-FFFF

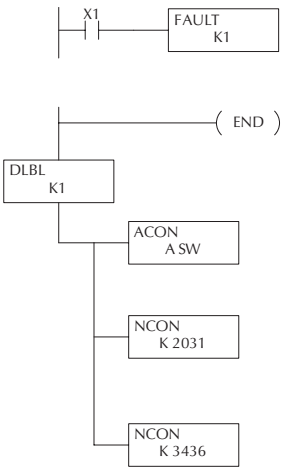
Indicadores	Descripción
SP50	ON cuando la instrucción FAULT es ejecutada

Ejemplo de instrucción Fault

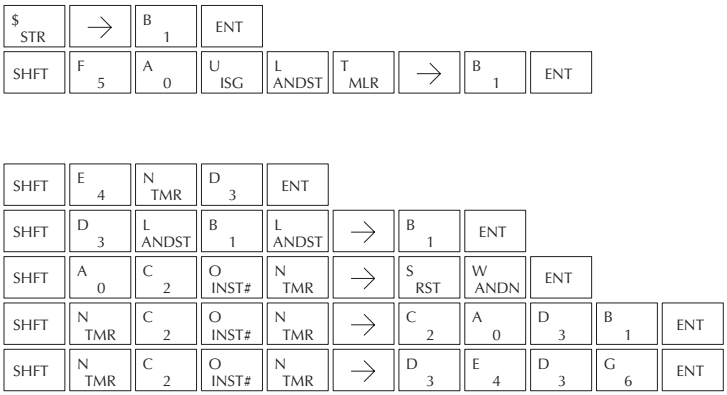
En el ejemplo siguiente cuando X1 está ON, se mostrará el mensaje “Baja presion 5” en el Programador portátil o en el visor LCD. El NCON usa el equivalente hexadecimal ASCII del texto a ser mostrado.



DirectSOFT



Programador D2-HPP



La instrucción Data Label (DLBL)

DS5	Usado
HPP	Usado

La instrucción DLBL marca el comienzo de un área ASCII/numérica de datos. DLBLs se programa después la declaración END.

Se puede usar un máximo de 64 instrucciones de DLBL en un programa. Se pueden usar múltiples NCONs y ACONs en un área de DLBL.

```
DLBL      K aaa
```

Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . K	1-FFFF

La instrucción ASCII Constant (ACON)

DS5	Usado
HPP	Usado

La instrucción ACON se usa con la instrucción DLBL para almacenar texto ASCII para uso con otras instrucciones. Se pueden almacenar 2 caracteres ASCII en una instrucción ACON.

Si se almacena solamente un carácter en un ACON será insertado un espacio delantero.

```
ACON
      A aaa
```

Tipo de operando de datos	Rango del DL06
	aaa
ASCII . . . . . A	0-9 A-Z

La instrucción Numerical Constant (NCON)

DS5	Usado
HPP	Usado

La instrucción NCON se usa con la instrucción DLBL para almacenar el equivalente hexadecimal ASCII de datos numéricos para el uso con otras instrucciones.

Se pueden almacenar 2 dígitos en una instrucción de NCON.

```
NCON
      K aaa
```

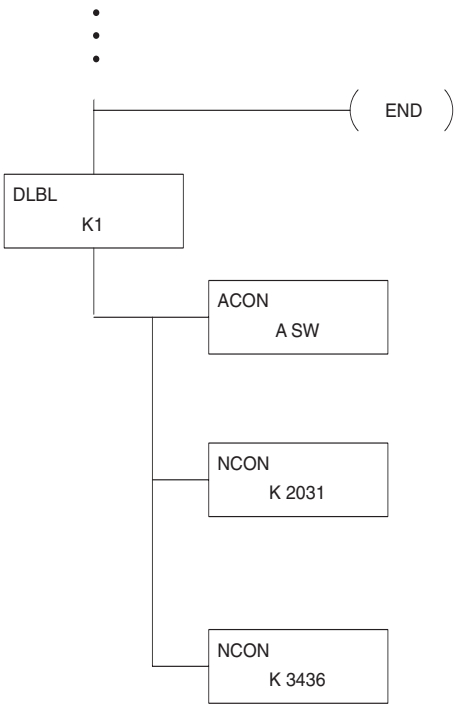
Tipo de operando de datos	Rango del DL06
	aaa
Constante . . . . . K	1-FFFF

Ejemplo de Data Label

En el ejemplo siguiente, se usa un ACON y 2 instrucciones de NCON dentro de una instrucción DLBL para construir un mensaje de texto.

Vea la instrucción FAULT para información de cómo exhibir los mensajes. El Manual del visor DV-1000 tiene también información de los mensajes a ser exhibidos.

DirectSOFT

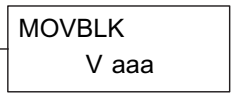


Programador D2-HPP

SHFT	E 4	N TMR	D 3	ENT						
SHFT	D 3	L ANDST	B 1	L ANDST	→	B 1	ENT			
SHFT	A 0	C 2	O INST#	N TMR	→	S RST	W ANDN	ENT		
SHFT	N TMR	C 2	O INST#	N TMR	→	C 2	A 0	D 3	B 1	ENT
SHFT	N TMR	C 2	O INST#	N TMR	→	D 3	E 4	D 3	G 6	ENT

La instrucción Move Block (MOVBLK)

DS5	Usado
HPP	Usado

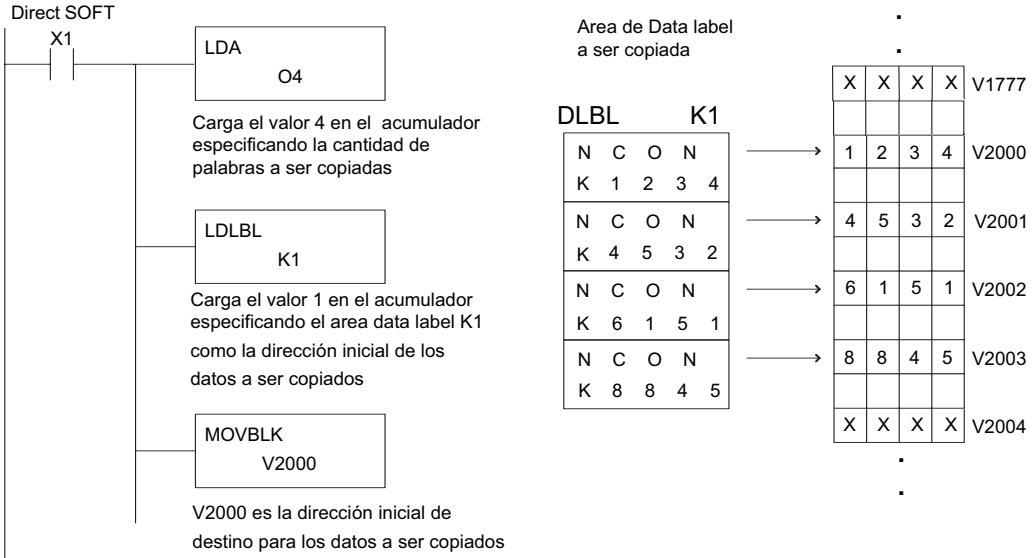


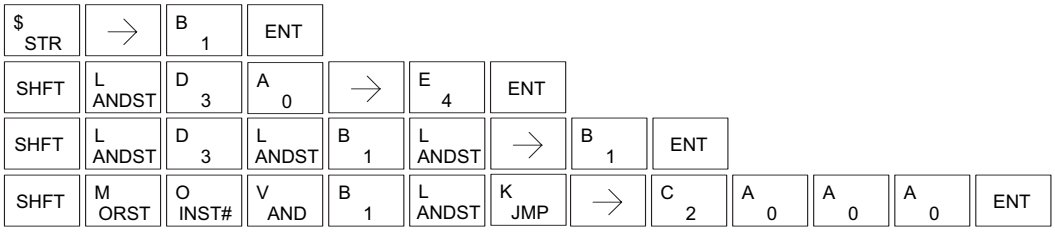
Esta instrucción copia un número especificado de palabras de un área de etiqueta de datos (Data Label) de la memoria del programa (ACON, NCON) a la localización especificada de Memoria. Se describen a continuación los pasos para usar esta instrucción:

- Paso 1:Cargue la cantidad de palabras (octal) que se copiarán al primer nivel del stack del acumulador
- Paso 2: Cargue la etiqueta de datos origen (LDLBL Kaaa) en el acumulador. De aquí es de donde serán copiados los datos.
- Paso 3: Insiera la instrucción MOVBLK que especifica la memoria de destino. Aquí es donde serán copiados los datos.

La instrucción Copy Data From a Data Label Area to Memory

Cuando X1 está encendido, el valor octal (O4) se copia al primer nivel del stack del acumulador usando el instrucción LDA. Este valor especifica la cantidad de palabras que se copiarán. La instrucción LDLBL cargará la dirección de los datos origen (K1) en el acumulador. Aquí es de donde serán copiados los datos. La instrucción MOVBLK especifica la localización inicial de destino y ejecuta el copiado de datos del área de laetiqueta de datos a Memoria.

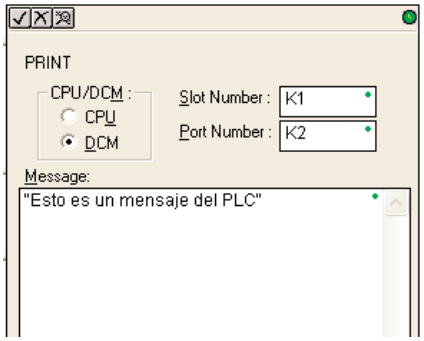




La instrucción Print Message (PRINT)

DS5	Usado
HPP	N/A

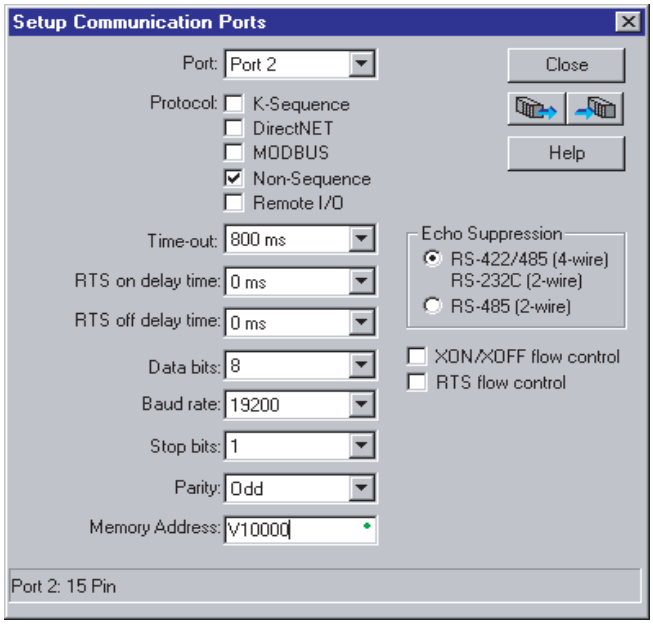
La instrucción PRINT imprime un mensaje con texto o con texto y variable o datos empotrados al puerto 2 en la CPU DL06 o el módulo D0-DCM, el cual debe estar configurado adecuadamente con el protocolo Non-sequence.



Tipo de operando de datos		Rango del DL06
		aaa
Constante	K	2 para puerto; 1-4 para ranura

Usted puede recordar de las especificaciones del PLC DL06 que los puertos son capaces de procesar varios protocolos. El puerto 1 no puede ser configurado como protocolo "Non-sequence".Para configurar el puerto 2 en *Direct*SOFT, escoja el menú " **PLC**", luego **SETUP**, luego "**Setup Sec. Comm Port**". Aparece un cuadro de diálogo como la figura de abajo :

- **Port**: En el campo de la lista de puertos disponibles en la parte superior, escoja "Port 2"
- **Protocol**: Haga clic en el cuadro de verificación a la izquierda de "Non- sequence"
- **Time-out**: El período que el puerto esperará después que envíe un mensaje para obtener una respuesta antes de detectar un error.
- **RTS On Delay Time**: tiempo que espera el PLC para mandar datos después que la señal TRS se ha hecho ON.
- **RTS Off Delay Time**: tiempo que espera el PLC DL06 después de mandar datos para hacer OFF la señal TRS.
- **Data Bits**: Seleccione 7 o 8 bits y hágalo igual a los bits de datos especificados para los aparatos conectados.
- **Baud Rate**: Las tasas disponibles de baud incluyen 300, 600, 1200, 2400, 4800, 9600, 19200, y 38400 Baud. Escoja una tasa más alta de baud inicialmente, y baje el valor si experimenta errores de datos o problemas de ruido en la red.



- **Stop Bits**: Escoja 1 o 2 bits de parada que debe ser los mismos que los de la impresora conectada.
- **Parity**: Escoja paridad none, even, o odd para verificación de error. Asegúrese de hacer igual la paridad especificada en la impresora a ser conectada.
- **Echo Suppression**: Seleccione el botón de radio adecuado basado en la configuración usada en el; puerto 2 (RS-232C, RS-422 o RS-485).
- **Xon/Xoff Flow control**: Escoja esta seleción si Ud. tiene el puerto 2 conectado para el control de flujo con hardware (Xon/Xoff) con las señales RTS y CTS conectada entre los dispositivos.
- **RTS Flow control**: Escoja esta selección is Ud. tiene la señal RTS del puerto 2 cableada a la impresora.
- **Memory address**: Escoja una dirección de memoria para usar como buffer para almacenamiento de datos ASCII.

Luego haga clic en el botón para enviar la configuración del puerto a la CPU y haga clic en CLOSE.

El puerto 2 en el DL06 tiene niveles de voltaje RS232 normales y debe trabajar con la mayoría de las conexiones seriales de las impresoras. .

Elemento de texto – esto se usa para imprimir conjuntos de caracteres. Los conjuntos de caracteres se definen como los caracteres (fuera de 0) entre comillas. Dos números hexadecimales precedidos por el signo de dólar significa un código de 8 bits de caracteres ASCII. También, dos caracteres precedidos por el signo de dólar se interpretan según la tabla siguiente:

#	Código de caracteres	Descripción
1	$$	Signo dólar ($)
2	$”	Comillas (”)
3	$L o $l	Line feed (LF)
4	$N o $n	Carriage return line feed (CRLF)
5	$P o $p	Form feed
6	$R o $r	Carriage return (CR)
7	$T o $t	Tab

Los ejemplos siguientes muestran varias convenciones de sintaxis y la longitud de las señales de salida a la impresora.

Ejemplo:

” ” Longitud 0 sin el carácter

”A” Longitud 1 con el carácter A

” ” Longitud 1 con espacio en blanco

” $” ” Longitud 1 con comillas

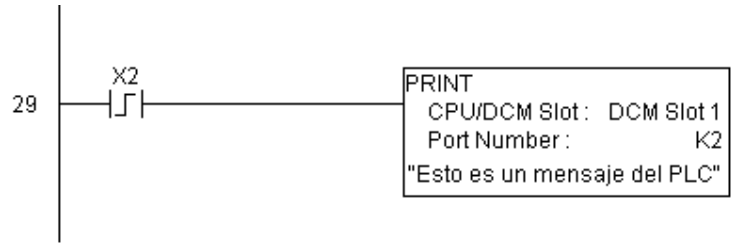
” $ R $ L ” Longitud 2 con un CR y un LF

” $ 0 D $ 0 A ” " Longitud 2 con un CR y un LF

” $ $ ” Longitud 1 con la marca $

Al imprimir una línea ordinaria de texto, usted necesitará incluir "**comillas**" antes y después del conjunto de texto. El código de error 499 ocurrirá en la CPU cuando la instrucción PRINT contiene texto inválido o está sin comillas.Es importante probar los datos de la instrucción PRINT durante el desarrollo de la aplicación.

El ejemplo siguiente imprime el mensaje al puerto 2. Usamos un contacto PD, que causa que la instrucción de mensaje sea activa por un barrido solamente. Note el $N al fin del mensaje, que produce un carriage return/line feed en la impresora. Esto prepara la impresora para imprimir la próxima línea y comenzar desde el margen izquierdo.



Envía el mensaje al puerto 2 del módulo D0-DCM cuando X2 hace una transición de OFF para ON

Elemento de memoria V - esto se usa para imprimir el contenido de memorias V en el formato entero o real. Use el número de memoria V o el número de memoria V con ":" y el tipo de datos. Los tipos de datos se muestran en la tabla abajo. El código del carácter debe ser con letras mayúsculas.

NOTA: Debe colocarse un espacio antes y después de la dirección de memoria V para separarla de la cadena de texto. Si no se hace esto aparecerá el error 499.

#	Código de caracteres	Descripción
1	ninguno	Binario de 16 bits (decimal)
2	: B	4 dígitos BCD
3	: D	Binario de 32 bits (decimal)
4	: D B	8 dígitos BCD

Ejemplos:

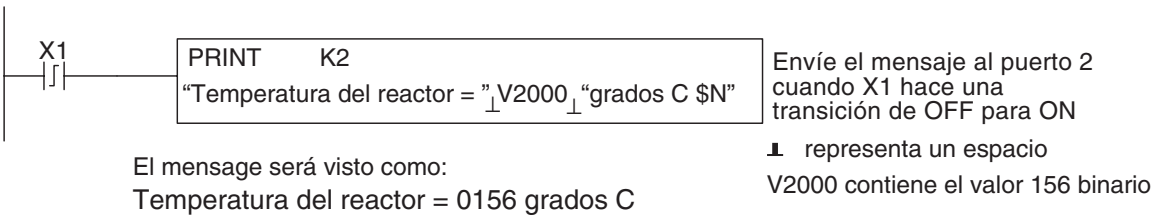
V2000 Imprime datos binarios en V2000 como número decimal

V2000 : B Imprime datos BCD en V2000

V2000 : D Imprime un número binario en V2000 y V2001 para un número decimal

V2000 : D B Imprime datos BCD en V2000 y V2001

Ejemplo: El ejemplo siguiente imprime un texto que contiene un texto y una variable. "Temperatura del reactor" marca los datos, que están en V2000 (como binario). Usted puede usar el calificativo :B después de V2000 si los datos están en el formato BCD, por ejemplo. El texto final agrega las unidades de grados a la línea de texto y el $N agrega un Carriage return/Line feed (CRLF), un comando de la impresora.



Elemento Texto de memoria V - Esto se usa para imprimir texto almacenado en memoria V. Use el signo % seguido por el número de caracteres después del número de memoria V para representar el texto.

Si usted asigna "0" como el número de caracteres, la función de impresión leerá al conteo desde el carácter de la primera localización. Luego comenzará en la próxima dirección de memoria V y leerá ese número de códigos ASCII para el texto desde la memoria.

Ejemplo:

V2000 % 16 Se imprimen 16 caracteres en V2000 hasta V2007.

V2000 % 0 Serán impresos los caracteres en V2001 a Vxxxx (determinado por el número en V2000).

Elemento de bit

Esto se usa para imprimir el estado del bit designado en la memoria V o un bit de relevador C. El elemento bit puede ser asignado por un punto (.) y el número de bit precedido por el número de memoria V o el número de relevador C. El tipo de salida se describe como mostrado en la tabla de abajo.

#	Formato de datos	Descripción
1	ninguno	Imprime 1 cuando el estado es ON y 0 cuando el estado es OFF
2	:BOOL	Imprime "TRUE" cuando el estado es ON y "FALSE" cuando el estado es OFF
3	:ONOFF	Imprime "ON" cuando el estado es ON y "OFF"cuando el estado es OFF

Ejemplo:

V2000.15 Imprime el estado del bit 15 en V2000, en formato 1/0.

C100 Imprime el estado de C100 en el formato 1/0.

C100 : BOOL Imprime el estado de C100 en formato TRUE/FALSE

C100 : ON/OFF Imprime el estado de C100 en formato ON/OFF

V2000.15 : BOOL Imprime el estado del bit 15 en V2000 en formato TRUE/FALSE

El máximo número de caracteres que puede imprimir es 128. El número de caracteres para cada elemento se lista en la tabla de abajo:

Tipo de elemento	Cantidad máxima de caracteres
Texto, 1 carácter	1
16 bit binarios	6
32 bits binarios	11
4 dígitos BCD	4
8 dígitos BCD	8
Número real (punto flotante)	12
Real con exponente	12
Texto en una memoria V	2
Bit (formato 0/1)	1
Bit (formato TRUE/FALSE)	5
Bit (formato ON/OFF)	3

El nemotécnico del programador D2-HPP es "PRINT" seguido del campo DEF.

Los relevadores especiales SP116 y SP117 indican el estado de los puertos del PLC DL06 **busy** (Ocupado), o **communications error** (error de comunicación)).

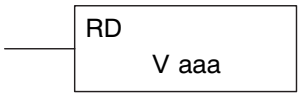
Vea el apéndice D relativo a relevadores especiales para una mejor descripción.

NOTA: Usted debe usar el relevador especial apropiado con la instrucción PRINT para asegurarse que el programa ladder no trate de IMPRIMIR a un puerto que está todavía ocupado por una instrucción PRINT previa o una instrucción WX o RX. .

Instrucciones de módulos inteligentes

La instrucción Read from Intelligent Module (RD)

DS32	Usado
HPP	Usado



Esta instrucción lee un bloque de datos (máximo cantidad de 128 bytes) de un módulo inteligente de E/S a la memoria de la CPU. Cargue los parámetros de la función en el primer y segundo nivel del stack del acumulador y al acumulador por tres instrucciónes adicionales

Se enumeran abajo los pasos para programar esta instrucción.

- Paso 1: Cargue el número de la base en el primer byte y el númerode la ranura (1 a 4) o al segundo byte del segundo nivel del Stack del acumulador.
- Paso 2: Cargue le número de bytes a ser transferidos al primer nivel del Stack del acumulador. Pueden ser transferidos hasta 128 bytes (o 64 palabras de 16 bits) en cada transacción, ya que hay 2 bytes por cada palabra.
- Paso 3: Cargue la dirección de los datos desde donde van a ser leídos. Este parámetro requiere un valor hexadecimal.
- Paso 4: Coloque la instrucción RD especificando la dirección inicial de memoria V (Aaaa) de donde se leerán los datos.

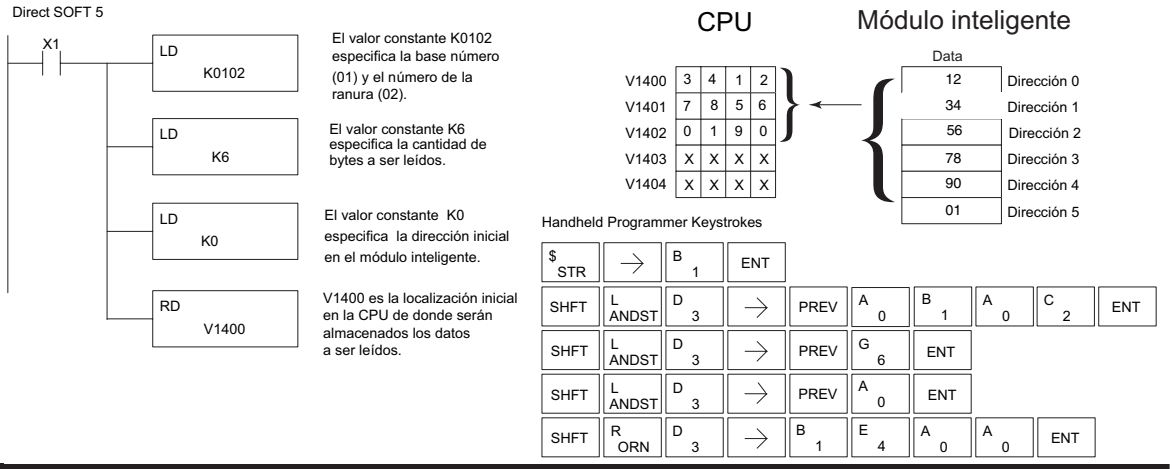
Sugerencia: — Para parámetros que requieren valores hexadecimales, se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor en el acumulador.

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria

Indicadores	Descripción
SP54	On cuando las instrucciones RX, WX RD, WT son ejecutada con parámetros errados.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

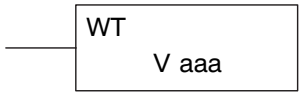
En el ejemplo siguiente cuando X1 está ENCENDIDO, la instrucción RD leerá seis bytes de datos de un módulo inteligente en la base 1, ranura 2 que comienzan en la dirección 0 en el módulo inteligente y copiará la información en las localizaciones de memoria V1400 hasta V1402



La instrucción Write to Intelligent Module (WT)

DS32	Usado
HPP	Usado

Esta instrucción escribe un bloque de datos (máximo de 128 bytes) a un módulo inteligente de E/S desde un bloque de Memoria en la CPU. Los parámetros de la instrucción son cargados en el primer y segundo nivel del stack del acumulador y el acumulador por tres instrucciones adicionales.



Se enumeran abajo los pasos para programar esta instrucción.

- Paso 1: Cargue el número de la base en el primer byte y el númerode la ranura (1 a 4) o al segundo byte del segundo nivel del Stack del acumulador.
- Paso 2: Cargue le número de bytes a ser transferidos al primer nivel del Stack del acumulador. Pueden ser transferidos hasta 128 bytes (o 64 palabras de 16 bits) en cada transacción, ya que hay 2 bytes por cada palabra.
- Paso 3: Cargue la dirección de los datos del módulo inteligente donde van a ser recibidos los datos. Este parámetro requiere un valor hexadecimal.
- Paso 4: Coloque la instrucción WT que especifica la dirección inicial de memoria V (Aaaa) en la CPU desde donde se leerán los datos.

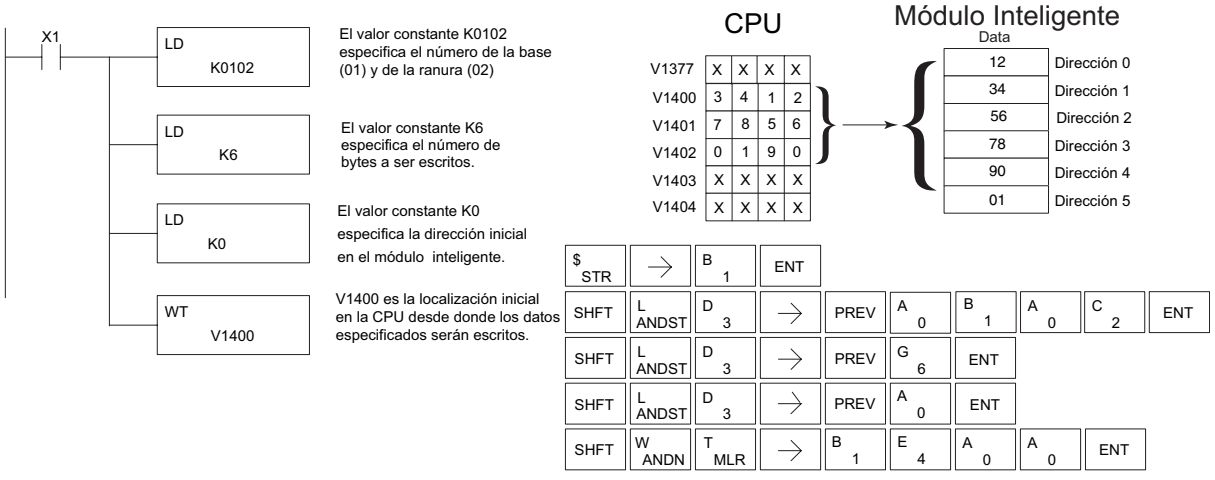
Sugerencia: — Para parámetros que requieren valores hexadecimales, se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor en el acumulador.

Tipo de operando de datos	Rango del DL06
	aaa
Memoria V V	Vea el mapa de memoria

Indicadores	Descripción
SP54	On cuando se ejecutan las instrucciones RX, WX RD, WT con parámetros errados.

NOTA: Las indicaciones de estado discretas SP son válidas sólo hasta que se ejecute otra instrucción que use el mismo relevador especial SP.

En el ejemplo siguiente, cuando X1 está encendido, el instrucción WT escribe seis bytes de datos a un módulo inteligente en la base 1, ranura 2 que comienzan en la dirección 0 en el módulo inteligente y copia los datos de las localizaciones de MemoriaV1400 hasta V1402

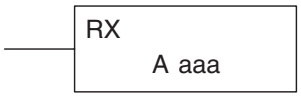


Instrucciones de comunicación en una red

La instrucción Read from Network (RX)

DS32	Usado
HPP	Usado

La instrucción RX es usada por el aparato maestro en una red para leer un bloque de datos de un aparato esclavo en la misma red. Los parámetros de la función son cargados al primer y segundo nivel del Stack del acumulador y al acumulador con tres instrucciones adicionales. Abajo están listados los pasos necesarios para programar la instrucción RX.



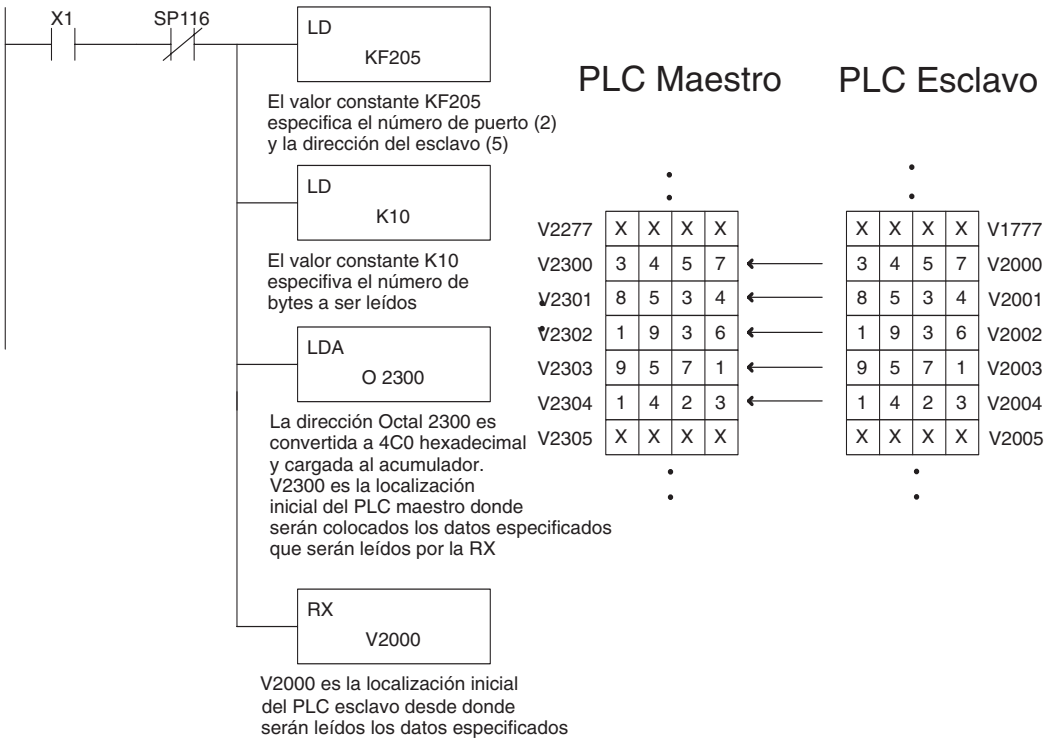
- Paso 1: Cargue la dirección del esclavo (0- 90 BCD) en el primer byte y el puerto del PLC (KF2) o ECOM maestro (0- 7) al segundo byte del segundo nivel del Stack del acumulador.
- Paso 2: Cargue el número de bytes a ser transferidos al primer nivel del Stack del acumulador. Pueden ser transferidos hasta 128 bytes (o 64 palabras de 16 bits) en cada transacción, ya que hay 2 bytes por cada palabra.
- Paso 3: Cargue la dirección de los datos a ser leída en el aparato esclavo al acumulador. Este parámetro requiere un valor hexadecimal.
- Paso 4: Coloque la instrucción RX especificando la dirección inicial de memoria V (Aaaa) en el aparato esclavo de donde se leerán los datos.

Sugerencia: — Para parámetros que requieren valores hexadecimales, se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor en el acumulador.

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero . . . . . P	Vea el mapa de memoria
Entradas X	0–777
Salidas . . . . . Y	0–777
Relevadores de control C	0–1777
Bits de etapas . . . . . S	0–1777
Bits de temporizadores . . . . . T	0–377
Bits de contadores . . . . . CT	0–177
Relevadores especiales . . . . . SP	0–777
Memoria de programa . . . . . $	0–7680 (2K de memoria de programa)

En el ejemplo siguiente, cuándo X1 está ON y el relevador SP116 del puerto "busy" (ocupado) (vea relevadores especiales) no está ON, la instrucción RX tendrá acceso al puerto 2, que opera como maestro. Serán leídos diez bytes consecutivos o 5 palabras de datos (V2000 - V2004) de un aparato esclavo en el nodo de dirección 5 y copiado a las memorias V2300-V2304 en la CPU con el puerto maestro.

DirectSOFT



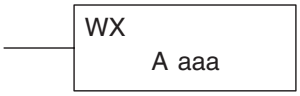
Programador D2-HPP

$ STR	→	B 1	ENT									
W ANDN	→	SHFT	SP STRN	B 1	B 1	G 6	ENT					
SHFT	L ANDST	D 3	→	SHFT	K JMP	SHFT	F 5	SHFT	C 2	A 0	F 5	ENT
SHFT	L ANDST	D 3	→	SHFT	K JMP	B 1	A 0	ENT				
SHFT	L ANDST	D 3	A 0	→	C 2	D 3	A 0	A 0	ENT			
SHFT	R ORN	X SET	→	C 2	A 0	A 0	A 0	ENT				

La instrucción Write a Network (WX)

DS5	Usado
HPP	Usado

La instrucción WX se usa para escribir un bloque de datos desde el aparato maestro a un aparato esclavo en la misma red. Los parámetros de la instrucción WX son cargados al acumulador y al primer y segundo nivel del Stack. Abajo están listados los pasos necesarios para ejecutar un programa para escribir datos en la red.



Paso 1: Cargue la dirección de esclavo (0-90 BCD) en el byte bajo y "F2" o ECOM maestro (0-7) en el byte alto del acumulador (las próximas dos instrucciones empujan esta palabra hacia abajo al segundo nivel del Stack).

Paso 2: Cargue el número de bytes a ser transferido al acumulador (la próxima instrucción empuja esta palabra al primer nivel del Stack). Pueden ser transferidos hasta 128 bytes (o 64 palabras de 16 bits) en cada transacción.

Paso 3: Cargue la dirección inicial de la CPU maestra al acumulador. Esta es la dirección de memoria de donde se escribirán los datos. Este parámetro requiere un valor hexadecimal.

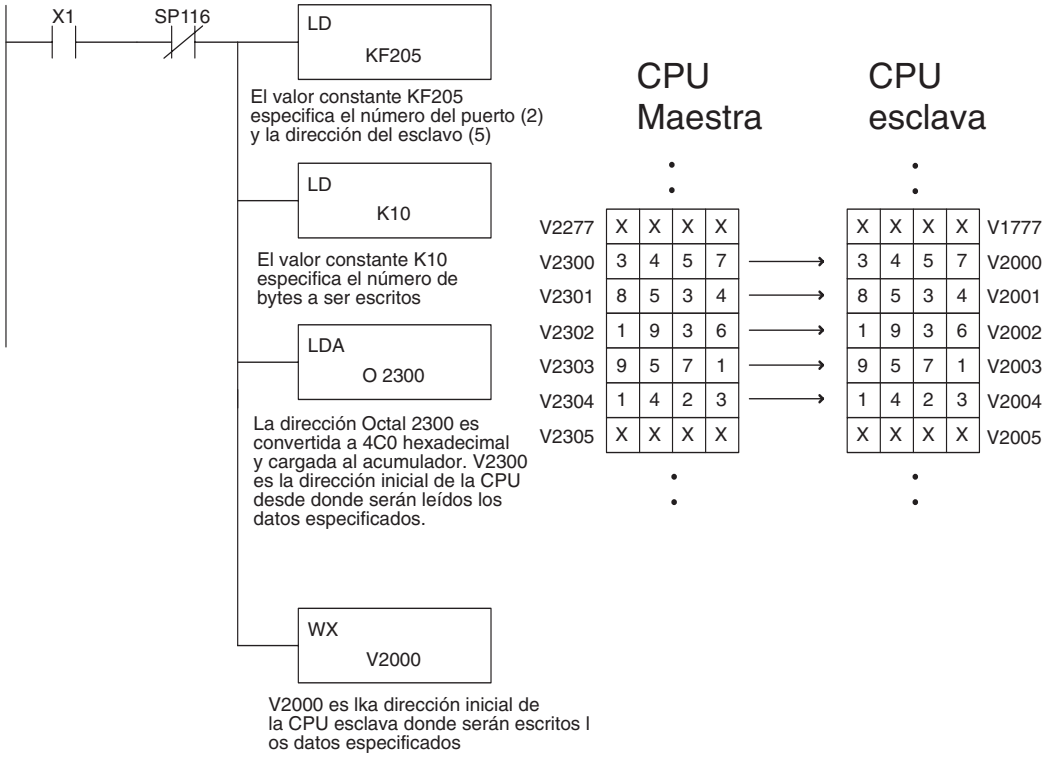
Paso 4: Coloque la instrucción WX especificando la dirección de memoria V (Aaaa) donde los datos serán escritos al esclavo.

Sugerencia: — Para parámetros que requieren valores hexadecimales, se puede usar la instrucción LDA para convertir una dirección octal al equivalente hexadecimal y cargar el valor al acumulador.

Tipo de operando de datos	Rango del DL06
. . . . . **A**	**aaa**
Memoria V V	Vea el mapa de memoria
Puntero . . . . . P	Vea el mapa de memoria
Entradas X	0–777
Salidas . . . . . Y	0–777
Relevadores de control C	0–1777
Etapas . . . . . S	0–1777
Bits de estado de temporizadores . . . . . T	0–377
Bits de estado de contadores . . . . . CT	0–177
Relevadores especiales . . . . . SP	0–777
Memoria de programa . . . . . $	0–7680 (2K de memoria de programa)

En el ejemplo siguiente cuando X1 está ON y el relevador SP116 "busy" (ocupado) (vea los relevadores especiales) no está ON, la instrucción WX tendrá acceso al puerto 2 que opera como maestro. Se leen diez bytes o 5 palabras consecutivas de datos de la CPU maestra y son copiados a las memorias V2000-V2004 en el aparato esclavo en la dirección del nodo 5.

DirectSOFT



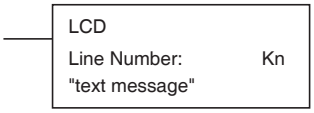
Programador D2-HPP

$ STR	→	B 1	ENT									
W ANDN	→	SHFT	SP STRN	B 1	C 1	E 6	ENT					
SHFT	L ANDST	D 3	→	SHFT	K JMP	SHFT	F 5	SHFT	C 2	A 0	F 5	ENT
SHFT	L ANDST	D 3	→	SHFT	K JMP	B 1	A 0	ENT				
SHFT	L ANDST	D 3	A 0	→	C 2	D 3	A 0	A 0	ENT			
SHFT	W ANDN	X SET	→	C 2	A 0	A 0	A 0	ENT				

La instrucción LCD

DS5	Usado
HPP	N/A

La instrucción LCD causa que un mensaje definido de texto de usuario sea mostrado en el panel del visor LCD. El visor es 16 caracteres de ancho y 2 filas de alto de modo que puede ser mostrado un "mensaje de texto" de 32 caracteres. Cada fila se define separadamente; el número máximo de caracteres que cada instrucción aceptará es 16.



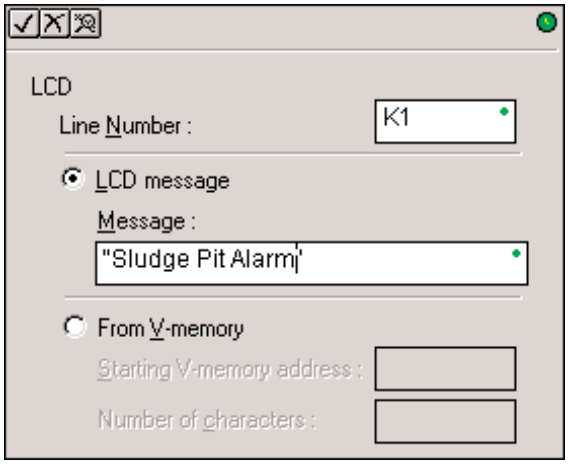
El mensaje de texto se puede entrar directamente en el campo de mensaje del diálogo de la instrucción o se puede localizar dondequiera en la memoria V. Si el texto se localiza en la memoria V, se usa la instrucción LCD para señalar el inicio de la localización de la memoria donde está el texto deseado. También se debe colocar la longitud del texto.

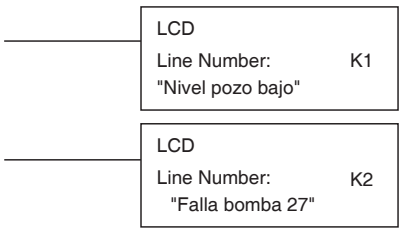
De la carpeta del proyecto de *Direct*SOFT32, utilice el navegador de instrucciones para localizar la instrucción LCD o use F7. Cuando usted selecciona la instrucción LCD y hace clic en OK, aparecerá el diálogo del LCD, según lo mostrado en los ejemplos. La instrucción LCD se inserta en el programa ladder a través de esta caja de diálogo de configuración.

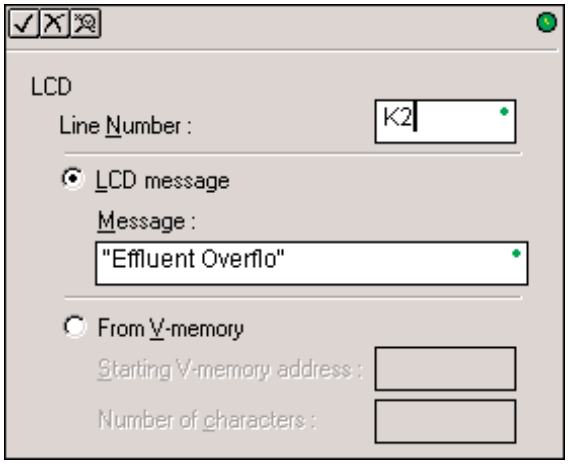
Los textos del visor pueden tener variables incluidas. Se puede incluir la fecha y la hora y valores de memoria V en el texto mostrado. Vea los ejemplos de cada caso.

Colocación directa de texto

Los dos diálogos muestran a la derecha las selecciones necesarias para crear las dos instrucciones ladder de la figura abajo. Deben colocarse comillas en el texto definido. En el primer diálogo, el texto "Nivel pozo bajo" usa quince espacios de caracteres y aparecerá en la línea 1 cuando la instrucción se activa. Note que el número de la línea es K1. Haciendo clic en el botón superior a la izquierda hace que la instrucción sea colocada en el programa.





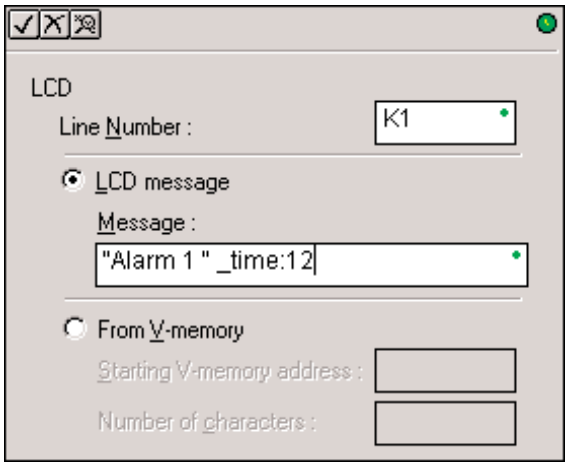


Identificando el segundo número de la línea como K2, el texto "Falla bomba 27" aparecerá en la segunda línea del visor cuando se activa la segunda instrucción.

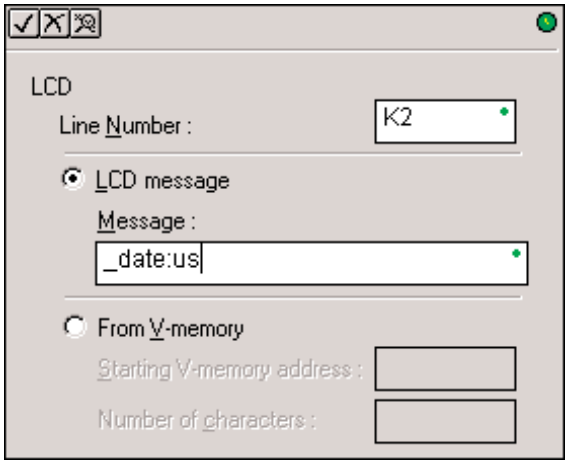
N	i	v	e	l		p	o	z	o		b	a	j	o	
F	a	l	l	a		b	o	m	b	a		2	7		

Mostrando fecha y/o hora

Se puede incluir la fecha y/o la hora en el texto desplegado usando las variables listadas en la tabla abajo. Estas variables se pueden incluir en el campo "Message" del diálogo LCD. En el ejemplo, la variable de tiempo (formato de 12 horas) es incluida colocando _time:12. Este formato de tiempo usa un máximo de siete espacios de caracteres. El segundo diálogo crea una instrucción que imprime la fecha en la segunda línea del visor, cuando está activado.



Variables y formatos de fecha y hora		
_date:us	Formato de E.E.U.U	MM/DD/AA
_date:e	Formato deEuropa	DD/MM/AA
_date:a	Formato de Asia	YY/MM/DD
_time:12	Formato de12 horas	HH:MMAM/PM
_time:24	Formato de 24 horas	HH:MM:SS




```
LCD
Line Number:          K1
"Alarma 1 " _time:12
```

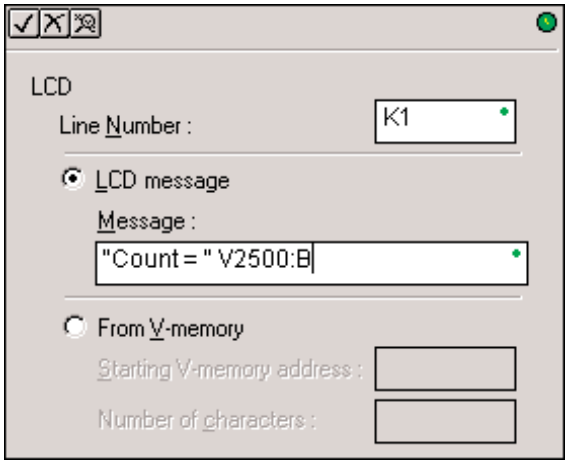
```
LCD
Line Number:          K2
  _date:us
```

A	l	a	r	m	a		1		1	1	:	2	1	P	M
0	5	-	0	8	-	0	2								

Colocando datos de memorias V

Cualquier dato en memoria puede ser mostrado en cualquiera de seis formatos disponibles de datos. Hay un ejemplo a la derecha.

Hay una lista de formatos de datos y sufijos en la próxima página. Note que formatos diferentes de datos necesitan de una cantidad diferente de caracteres en el visor.




```
LCD
Line Number:          K2
 "Piezas = " V2500:B
```

P	i	e	z	a	s	=		0	4	1	2				

Sufijos de formatos de datos para datos incluídos de memoria V

Varios formatos de datos están disponibles para desplegar datos de memoria V en el LCD. Las opciones se muestran en la tabla a continuación. Se usan dos puntos para separar la localización incluída de memoria V del sufijo de formato de datos y calificativo. Vea el ejemplo anterior.

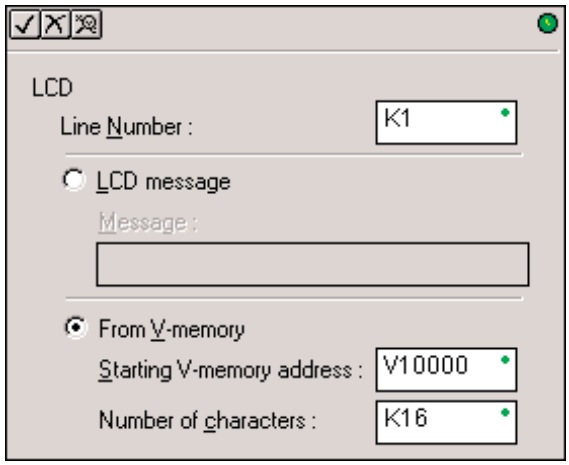
Formato de datos	Sufijo	Ejemplo	Caracteres mostrados												
ningún sufijo (Formato de 16-bits)		V2000 = 0000 0000 0001 0010	**1**	**2**	**3**	**4**									
		V2000			1	8									
	[:S]	V2000:S	1	8											
	[:C0]	V2000:C0	0	0	1	8									
	[:0]	V2000:0			1	8									
:B (4 dígitos BCD)		V2000 = 0000 0000 0001 0010	**1**	**2**	**3**	**4**									
	[:B]	V2000:B	0	0	1	2									
	[:BS]	V2000:BS	1	2											
	[:BC0]	V2000:BC0	0	0	1	2									
	[:B0]	V2000:B0			1	2									
:D (32 bits decimales)		V2000 = 0000 0000 0000 0000	**Palabra doble**												
		V2001 = 0000 0000 0000 0001	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**		
	[:D]	V2000:D							6	5	5	3	6		
	[:DS]	V2000:DS	6	5	5	3	6								
	[:DC0]	V2000:DC0	0	0	0	0	0	0	6	5	5	3	6		
	[:D0]	V2000:D0							6	5	5	3	6		
:DB (8 dígitos BCD)		V2000 = 0000 0000 0000 0000	**Palabra doble**												
		V2001 = 0000 0000 0000 0011	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**					
	[:DB]	V2000:DB	0	0	0	3	0	0	0	0					
	[:DBS]	V2000:DBS	3	0	0	0	0								
	[:DBC0]	V2000:DBC0	0	0	0	3	0	0	0	0					
	[:DB0]	V2000:DB0				3	0	0	0	0					
:R (Número real- Palabra doble)		V2001/V2000 = 222.11111 (real number)	**Palabra doble**												
			1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
	[:R]	V2000:R				f	2	2	2	.	1	1	1	1	1
	[:RS]	V2000:RS	f	2	2	2	.	1	1	1	1	1			
	[:RC0]	V2000:RC0	f	0	0	0	2	2	2	.	1	1	1	1	1
	[:R0]	V2000:R0				f	2	2	2	.	1	1	1	1	1
:E (Número real- Palabra doble con exponente)		V2001/V2000 = 222.1 (real number)	**Palabra doble**												
			1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
	[:E]	V2000:E		f	2	.	2	2	1	0	0	E	+	0	2
	[:ES]	V2000:ES	f	2	.	2	2	1	0	0	E	+	0	2	
	[:EC0]	V2000:EC0	f	2	.	2	2	1	0	0	E	+	0	2	
	[:E0]	V2000:E0	f	2	.	2	2	1	0	0	E	+	0	2	
f = indicación más/ menos (más = sin símbolo, menos = -)															

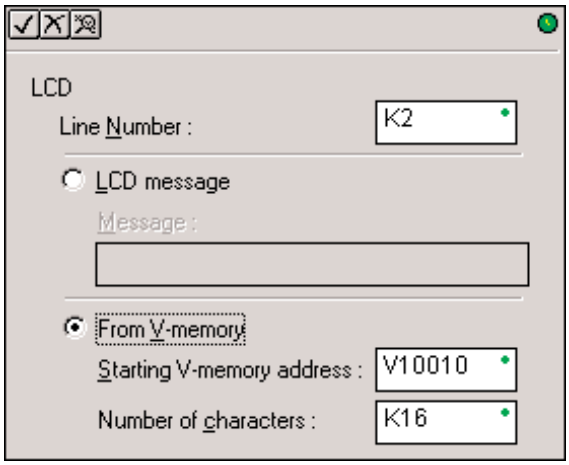
Los calificativos “S”,” C0”, y “0” alteran la presentación de ceros y espacios a la izquierda. “S” saca los espacios y justifica el resultado a la izquierda. “C0” reemplaza los espacios delanteros con ceros. “0” es una modificación de “C”0. “0” elimina cualquier cero delantero en la versión de formato “C0” y los convierte a espacios.

Colocación de texto desde la memoria V

Alternativamente, un texto que ya está en la memoria V se puede mostrar en el visor LCD siguiendo el ejemplo en esta página. El diálogo LCD se usa dos veces, una vez por cada línea en el visor.

El diálogo necesita la dirección del primer carácter y el número de caracteres a ser exhibido. Por ejemplo, los dos diálogos mostrados en la próxima página crearían las dos instrucciones LCD de abajo. Cuándo activadas, estas instrucciones harían que sean exhibidos los caracteres ASCII en V10000 -V10017. Los caracteres ASCII y sus direcciones correspondiente de memoria se muestran en la tabla de abajo.





LCD
Line Number: K1
Starting V Memory Address: V10000
Number of Characters: K16

LCD
Line Number: K2
Starting V Memory Address: V10010
Number of Characters: K16

V10000	a	T
V10001	q	n
V10002	e	u
V10003	h	
V10004	r	o
V10005	o	n
V10006	C	
V10007		
V10010	l	A
V10011	a	t
V10012	t	
V10013	m	e
V10014	e	p
V10015	a	r
V10016	u	t
V10017	a	r

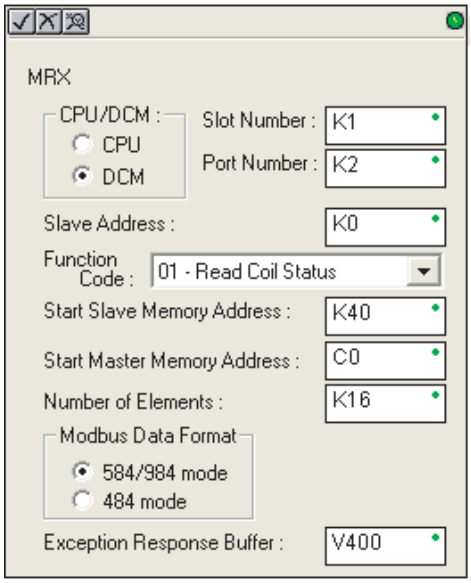
T	**a**	**n**	**q**	**u**	**e**		**h**	**o**	**r**	**n**	**o**		**C**		
A	**l**	**t**	**a**		**t**	**e**	**m**	**p**	**e**	**r**	**a**	**t**	**u**	**r**	**a**

Instrucciones MODBUS RTU

La instrucción MODBUS Read from Network (MRX)

DS5	Usado
HPP	N/A

La instrucción MRX es usada por el maestro (master) de la red DL06 para leer un bloque de datos de un aparato conectado como esclavo y para escribir los datos en direcciones de memoria V dentro del maestro. La instrucción permite al usuario especificar si el puerto corresponde a la CPU o al módulo D0-DCM, el código de la Función de MODBUS, la dirección de la estación del esclavo, las direcciones de memoria de inicio del maestro y de esclavo, el número de elementos para transferir, formato de datos de MODBUS y la memoria intermedia (buffer) de Respuesta de Excepción.



- **CPU/DCM** : especifica si el maestro lee datos desde el puerto 2 o desde el módulo D0-DCM.
- **Slot Number** (Número de la ranura): debe ser la ranura del PLC donde está instalado el D0-DCM).
- **Port Number** (Número del puerto): debe ser el Puerto DL06 2 (K2).
- **Slave Address** (Dirección de esclavo): especifica una dirección del esclavo (0-247).
- **Function Code** (Código de Función): Los siguientes códigos de función MODBUS pueden ser manejados por la instrucción MRX:

  01 – Lea un grupo de bobinas

  02 – Lea un grupo de entradas

  03 – Lea registros

  04 – Lea registros de entradas

  07 – Lea el estado de Excepción

- **Start Slave Memory Address** (Dirección inicial de memoria de esclavo) : especifica la dirección de memoria de esclavo de inicio de los datos a ser leídos. Vea la tabla en la página siguiente.
- **Start Master Memory Address** (Dirección de memoria inicial del maestro) : especifica la dirección de memoria de inicio en el maestro, donde los datos serán colocados. Vea la página siguiente.
- **Number of Elements** (Número de Elementos) : especifica cuántas bobinas, entradas, registros o registro de entradas se leerán. Vea la tabla en la página siguiente.
- **MODBUS Data Format** (Formatos de Datos de MODBUS): especifica el formato de datos MODBUS 584/984 o 484 a ser usado.
- **Exception Response Buffer** (Memoria Intermedia de Respuesta por Excepción): especifica la dirección de memoria del PLC maestro donde se colocará la Respuesta por Excepción. Vea la tabla en la página siguiente.

Rangos de direcciones de esclavo MRX

Código de función	Formato de datos MODBUS	Rango de direcciones de esclavo
01 – Lea bobina	Modo 484	1–999
01 – Lea bobina	Modo 584/984	1–65535
02 – Lea estado de entradas	Modo 484	1001–1999
02 – Lea estado de entradas	Modo 584/984	10001–19999 (5 dígitos) o 100001–165535 (6 dígitos)
03 – Lea Holding register	Modo 484	4001–4999
03 – Lea Holding register	Modo 584/984	40001–49999 (5 dígitos) o 4000001–465535 (6 dígitos)
04 – Lea Input register	Modo 484	3001–3999
04 – Lea Input register	Modo 584/984	30001–39999 (5 dígitos) o 3000001–365535 (6 dígitos)
07 – Lea estado	Modo 484 y 584/984	n/a

Rango de direcciones de memoria del maestro MRX	
Tipo de operando de datos	**Rango del DL06**
Entradas X	0–1777
Salidas. . . . . Y	0–1777
Relevadores de control. C	0–3777
Bits de etapas . . . . . S	0–1777
Bits de temporizadores . . . . . T	0–377
Bits de contadores . . . . . CT	0–377
Relevadores especiales . . . . . SP	0–777
Memoria V. V	all
Entradas globales . . . . . GX	0–3777
Salidas globales . . . . . GY	0–3777

Número de elementos	
Tipo del operando de datos	**Rango en el DL06**
Memoria V V	Todas
Constante . . . . . K	Bits: 1–2000 Registros: 1–125

Buffer de Exception Response	
Tipo de operando de datos	**Rango del DL06**
Memoria V V	Todas

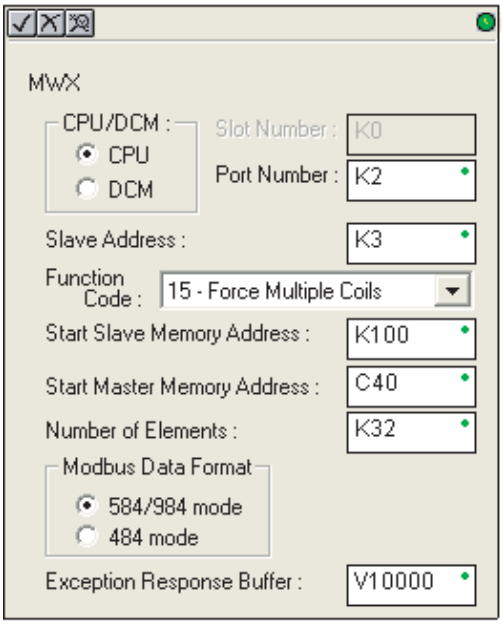
Ejemplo de MRX

See page 5-208

La instrucción MODBUS Write a Network (MWX)

DS5	Usado
HPP	N/A

La instrucción MWX usa para escribir un bloque de datos de la memoria del maestro de la red del PLC DL06 a las direcciones de memoria de MODBUS dentro de un aparato esclavo en la red. La instrucción permite al usuario especificar si el puerto corresponde a la CPU o al módulo D0-DCM, el código de la función de MODBUS, la dirección de la estación esclavo, las direcciones de memoria inicial del maestro y esclavo, el número de elementos a transferir, el formato de datos de MODBUS y la Memoria Intermediaria (buffer) de Respuesta por Excepción.



- **CPU/DCM** : especifica si el maestro lee datos desde el puerto 2 o desde el módulo D0-DCM.
- **Slot Number** (Número de la ranura): debe ser la ranura del PLC donde está instalado el módulo D0-DCM).
- **Port number** (número del puerto): Debe ser el Puerto 2 (K2) del PLC DL06 o del D0-DCM
- **Slave Address** (Dirección de esclavo): especifica una dirección de la estación esclavo (0-247).
- **Function Code** (Código de función): Los siguientes códigos de función MODBUS pueden ser manejados por la instrucción MRX:

  05 – Fuerce una bobina

  06 – Prefije un registro único

  15 – Fuerce bobinas múltiplas

  16 – Prefije registros múltiplos

- **Start Slave Memory Address** (Dirección de memoria inicial del esclavo) : Especifica la dirección de memoria de esclavo de inicio de los datos a ser escritos. Vea la tabla en la página siguiente.
- **Start Master Memory Address** (Dirección de memoria inicial del maestro) : especifica la dirección de memoria de inicio en el maestro, de donde los datos serán escritos. Vea la tabla en la página siguiente.
- **Number of elements** (Número de elementos) : Especifica cuántas bobinas, entradas, registros o registro de entradas se escribirán. Vea la tabla en la página siguiente.
- **MODBUS Data Format** (Formatos de datos de MODBUS): especifica el formato de datos MODBUS 584/984 o 484 a ser usado.
- **Exception Response Buffer** (Memoria Intermediaria de Respuesta por Excepción): especifica la dirección maestro de memoria donde se colocará la Respuesta por Excepción. Vea la tabla en la página siguiente.

Rangos de direcciones de esclavo MWX

Rangos de direcciones de memoria del maestro MWX	
Tipo de operando de datos	**Rango del DL06**
Entradas X	0–1777
Salidas . . . . . Y	0–1777
Relevadores de control C	0–3777
Bits de etapas . . . . . S	0–1777
Bits de temporizadores . . . . . T	0–377
Bits de contadores . . . . . CT	0–377
Relevadores especiales. . . . . SP	0–777
Memoria V V	Todas
Entradas globales . . . . . GX	0–3777
Salidas globales . . . . . GY	0–3777

Rangos de direcciones de memoria del maestro MWX

Cantidad de elementos	
Tipo de operando de datos	**Rango del DL06**
Memoria V V	all
Constante . . . . . K	Bits: 1–2000 Registros: 1–125

Número de elementos MWX

Número de elementos	
Tipo del operando de datos	**Rango en el DL06**
Memoria V V	Todas
Constante . . . . . K	Bits: 1–2000 Registros: 1–125

Buffer de exception response MWX

Cantidad de elementos	
Tipo de operando de datos	**Rango del DL06**
Memoria V V	Todas

Ejemplo de MRX y de MWX

El puerto 2 del DL06 tiene dos contactos de relevadores especiales asociados con éste. Uno indica "Puerto ocupado" (SP116), y el otro indica "Error de comunicación de puerto" (SP117).

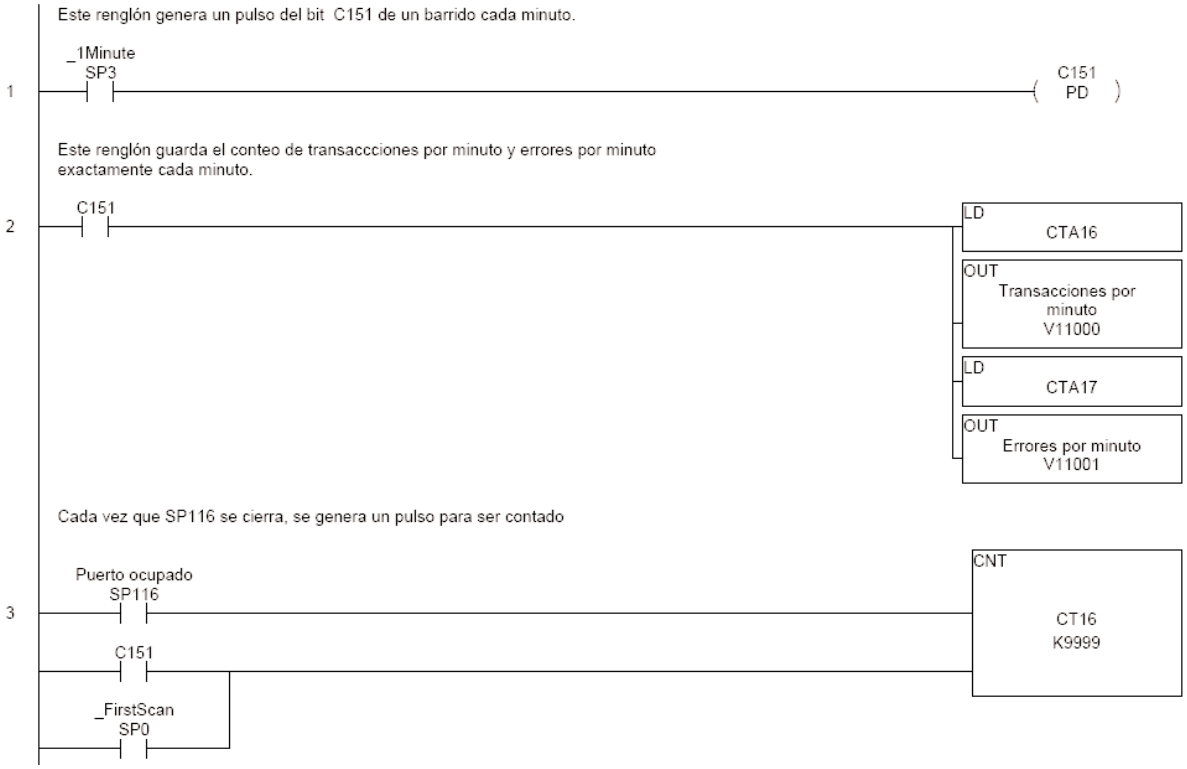
El bit de "puerto ocupado" está ON mientras el PLC se comunica con un esclavo. Cuándo el bit está apagado el programa puede iniciar el próximo pedido a la red.

El bit de "Error de comunicación de puerto" se activa cuando el PLC ha detectado un error. El uso de este bit es opcional. Cuando se usa, debe estar adelante de cualquier instrucción de red ya que el bit de error es vuelto a 0 cuando se ejecuta una instrucción MRX o MWX.

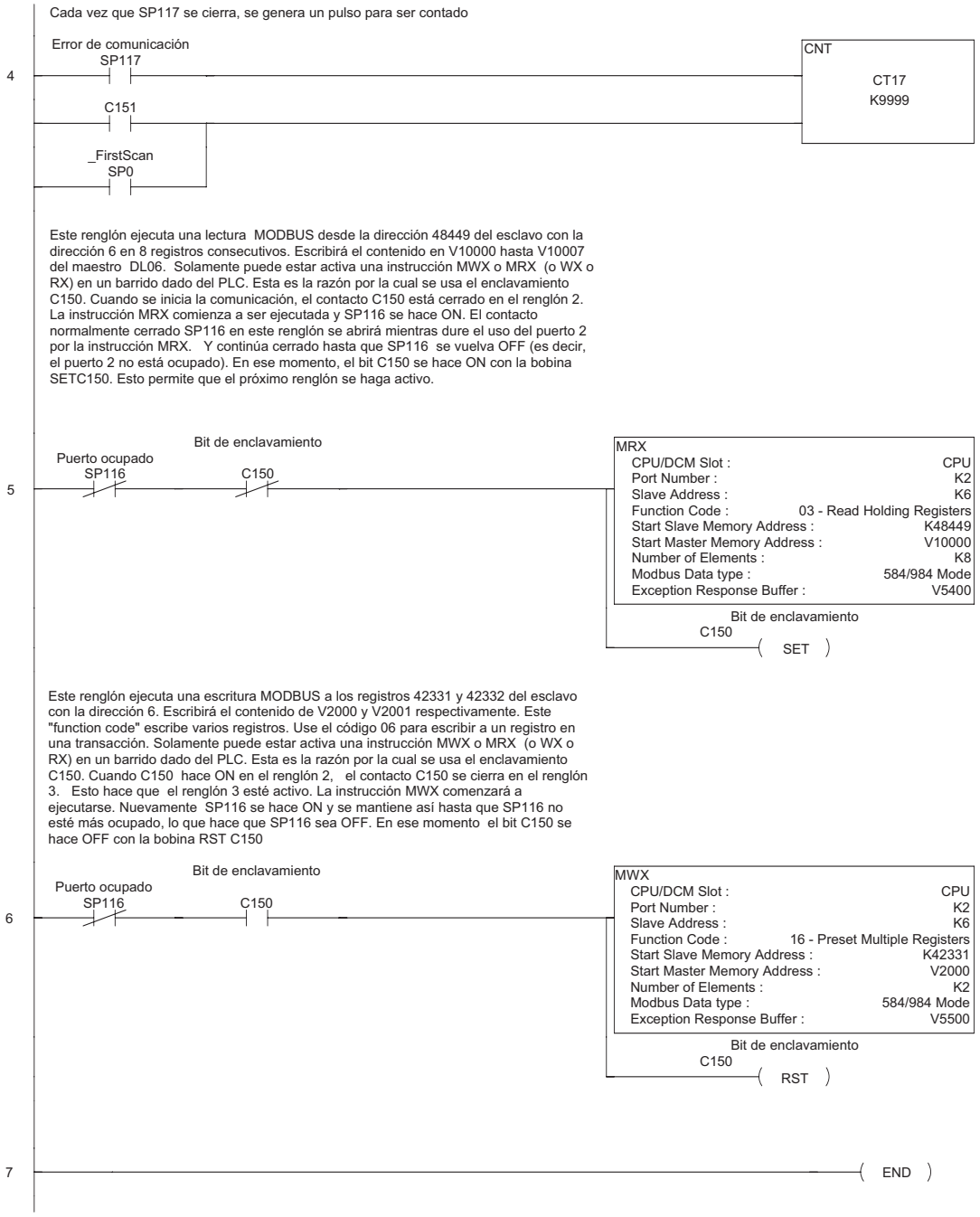
Típicamente las comunicaciones de red durarán más de 1 barrido. El programa debe esperar terminar la comunicación para comenzar la próxima transacción. Es por eso que se coloca un enclavamiento entre la lectura y la escritura con el relevador C150, en el ejemplo de la página 5-209.

El relevador interno C150 es usado para enclavar la lectura y la escritura. Vea por favor el ejemplo mostrado abajo para entender como funciona el enclavamiento.

En algunas aplicaciones hay ruido electromagnético que puede bajar la cantidad de transacciones y aparecen errores de comunicación. es perfectasmente aceptable tener un 5% de errores en algunas aplicaciones. Los renglones 1 a 4 supervisan la cantidad de transacciones y errores por minuto. El renglón 2 guarda el conteo de cada minuto, lo que permite calcular el tiempo promedio que demora cada transacción.



Continúa en la próxima página....



En este ejemplo tratamos con una tasa de 9600 kbps. Hubo 1350 transacciones por minito leídas en V11000; Cuando se aumentó a 38400 kbps, fueron 3535 transacciones por minuto.

Por lo tanto, para una velocidad transmisión de 9600 kbps, cada transacción ocurre cada 44.4 milisegundos. Para 38400 kbps, cada, 19.98 ms.

Instrucciones ASCII

El PLC DL06 utiliza varias instrucciones y métodos que permiten leer y escribir texto ASCII a través del puerto 2 de comunicación o el módulo D0-DCM o aún desde el módulo coprocesador F0-CP128. El puerto 2 del DL06 puede ser usado para leer o escribir formatos ASCII pero no pueden ser usados ambos métodos al mismo tiempo en el mismo PLC, es decir, es necesario hacer una lógica para que una operación sea ejecutada en un período y luego la otra ejecutada en el próximo. Vea el apéndice G.

Vea una lista de transformación del código ASCII a decimal y a hexadecimal en el apéndice G.

El PLC DL06 también puede descifrar caracteres ASCII embutidos en uno de los protocolos aceptados (K-sequence, *Direct*Net, Modbus) en el puerto del PLC.

Leyendo Texto ASCII

Hay varios métodos que el PLC DL06 puede usar para leer ASCII.

1) Instrucción de lectura ASCII IN (AIN) – Esta instrucción se usa para recibir texto ASCII puro con parámetros tales como texto ASCII de longitud fija o variable, caracteres de terminación, opción de intercambio de bytes e instrucciones de control. Puede usar lectores de código de barras, balanzas, etc. para escribir texto ASCII al puerto 2 basados en los parámetros de la instrucción AIN.
2) Escriba texto ASCII embutido directamente a la memoria V desde una interfase hombre máquina o un aparato maestro similar usando un protocolo de los aceptados con el puerto 1 o 2 o el módulo D0-DCM. La instrucción AIN no se usa en este caso.
3) Si un PLC DL06 es maestro en una red, la instrucción RX (Network Read) puede ser usada para leer datos ASCII embutidos, desde un esclavo, usando protocolos aceptados con el puerto 2. La instrucción RX coloca los datos directamente en la memoria V.

Escribiendo Texto ASCII

Las siguientes instrucciones pueden ser usadas para escribir texto ASCII:

1) **Print from V–memory** (PRINTV) – Use esta instrucción para escribir texto ASCII puro, almacenado en la memoria del PLC, por el puerto 2 a un panel o a una impresora serial. La instrucción acepta la dirección inicial de memoria V, la longitud de la cadena, opción de cambio de bytes, etc. Cuando el bit de permiso está activado, la cadena de texto es escrita en el puerto 2.
2) **Print a V–memory** (VPRINT) – Use esta instrucción para crear cadenas ASCII en el PLC (Por ejemplo mensajes de alarma). Cuando el bit de permiso de la instrucción está activado, el mensaje es escrito en una localización de memoria pre-definida. Luego la instrucción PRINTV puede ser usada para escribir esta cadena ya definida saliendo por el puerto 2. Pueden ser usadas fechas de tipo americano, europeo o asiátic.

Adicionalmente, si un PLC DL06 es maestro en una red, la instrucción WR (Network write) puede ser usada para escribir datos ASCII embutidos a una interfase hombre máquina o a un esclavo directamente desde la memoria V usando uno de los protocolos usados por el PLC con el puerto 2.

Administrando texto ASCII

Las siguientes instrucciones pueden ser útiles para programar las cadenas de texto ASCII en la memoria del PLC DL06:

- **ASCII Find** (AFIND) –Encuentra en que localización de memoria está contenida una porción especifica de la cadena ASCII. Se permiten búsquedas para adelante y para atrás.
- **ASCII Extract** (AEX) – Extrae una porción especifica de la cadena ASCII de una serie de memorias V y lo coloca en otra serie de memorias.(Típicamente algún valor).
- **Compare V–memory** (CMPV) – Esta instrucción es usada para comparar 2 bloques de memoria y es usada típicamente para detectar un cambio en una cadena ASCII. El tipo de formato debe ser igual en ambos datos (por ejemplo, BCD, ASCII, etc.)
- **Swap Bytes** (SWAPB) – Típicamente usada para intercambiar bytes de una memoria en datos ASCII que han sido escritos directamente a la memoria desde una interface hombre maquina o equipo maestro similar con un protocolo de comunicación. Las instrucciones AIN y AEX tienen la función de intercambio de bytes integrada.
- **ASCII to HEX** (ATH) –Convierte una tabla de valores ASCII a una tabla de valores hexadecimales.
- **HEX to ASCII** (HTA) – Convierte una tabla de valores hexadecimales a una tabla de valores ASCII.

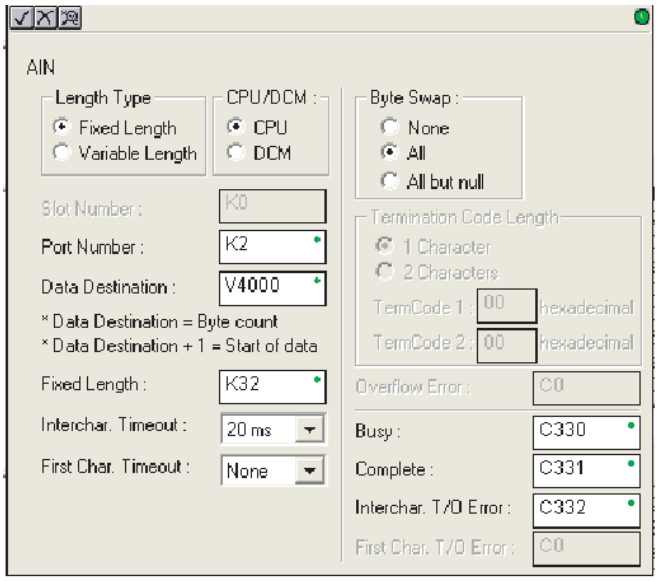
Para poder usar las funciones de lectura y escritura ASCII, debe configurarse el puerto 2 con el protocolo Non-Sequence. Vea más detalles en el apéndice K de este manual.

La instrucción ASCII Input (AIN)

DS5	Usado
HPP	N/A

La instrucción AIN permite recibir cadenas ASCII a través del puerto 2 y coloca la cadena en una serie de memorias (Una tabla). Los datos ASCII pueden ser recibidos como un número fijo de bytes o como una cadena variable con un o más caracteres de terminación especificados. Otras características incluyen, preferencia de intercambio de bytes, tiempo de llegada excedido,(Timeout) y bits de alarma libremente definidos para Ocupado (Busy), Completo (Complete) y error de tiempo (Timeout error).

Configuración AIN de longitud fija



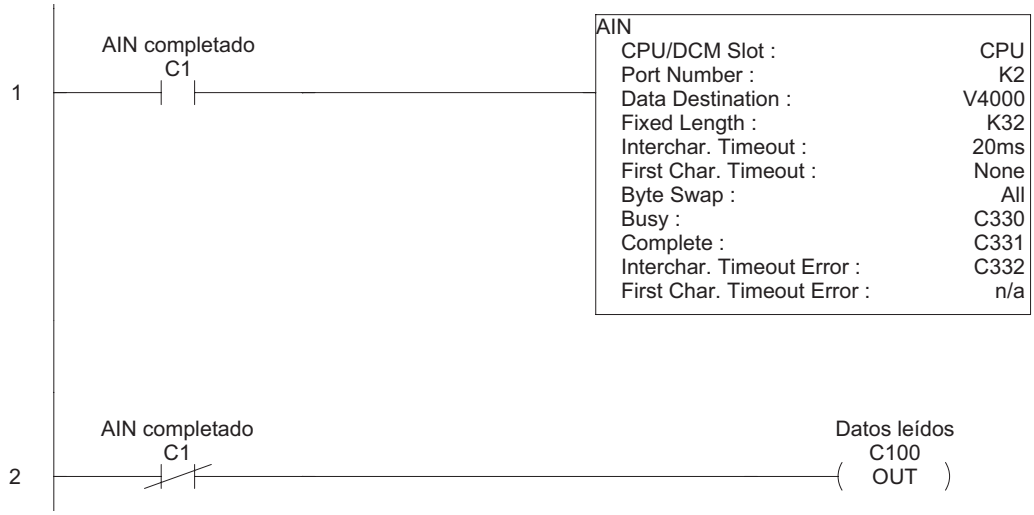
- **Length Type**: Seleccione Fixed Length de acuerdo a la longitud de la cadena ASCII que será enviada al puerto del PLC.
- **CPU/DCM** : especifica si el maestro lee datos desde el puerto 2 o desde el módulo D0-DCM.
- **Port Number**: Siempre puerto No. 2 (K2)
- **Data Destination**: Especifique donde en la memoria será colocada la cadena ASCII.
- **Fixed Length**: Especifique la longitud, en bytes, de la cadena ASCII que será recibida por el puerto.
- **Inter–character Timeout**: si el periodo de los caracteres ASCII entrantes excede el tiempo ajustado, será activado. el error de timeout. No será almacenada ninguna información en la localización de memoria de destino. El bit será desactivado cuando el renglón donde está la instrucción está desactivado. Un valor de 0 ms elimina esta función.
- **First Character Timeout**: Si el período transcurrido entre cuando AIN está activado hasta el tiempo que el primer carácter es recibido es mayor que el valor prefijado, será activado el bit de First Carácter Timeout. Este bit será desactivado cuando la instrucción AIN sea desactivada. Un valor de 0 ms elimina esta función.
- **Byte Swap**: Intercambia el byte más alto con el más bajo en cada memoria V de la cadena ASCII de longitud fija. Vea la instrucción SWAPB para más detalles.
- **Busy Bit**: Este está ON cuando la instrucción AIN está recibiendo datos ASCII.
- **Complete Bit**: Se coloca ON cuando los datos ASCII han sido recibidos con la longitud especificada y OFF cuando los bits que hacen la instrucción AIN activa están desactivados.
- **Inter–character Timeout Error bit**: Se coloca ON cuando el Character Timeout ha sido sobrepasado. Vea la explicación de Character Timeout arriba.
- **First Character Timeout Error bit**: Se coloca ON cuando el Character Timeout ha sido sobrepasado. Vea la explicación de Character Timeout arriba.

Antes de cada lectura, debe hacerse un reset de la instrucción AIN. Puede hacerse esta acción desactivando y activando el renglón donde está AIN o también con la instrucción ACBR (vea la página 5-228),

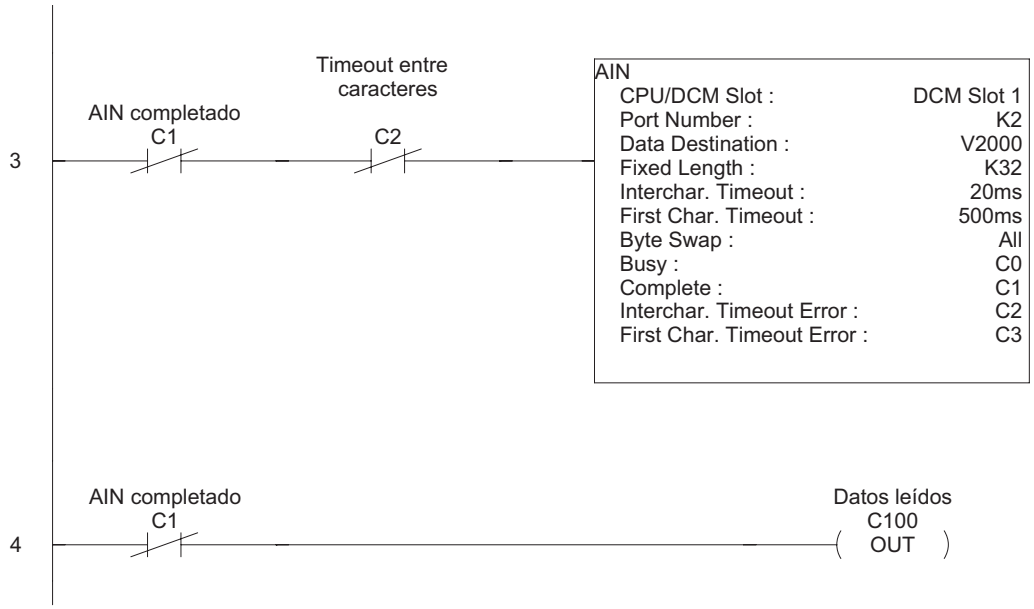
Parámetro	
Destino de los datos	Todas las memorias V
Longitud fija	K1–128
Bits: Busy, Complete, Timeout Error, Overflow	C0–3777

Ejemplos de longitud fija de AIN

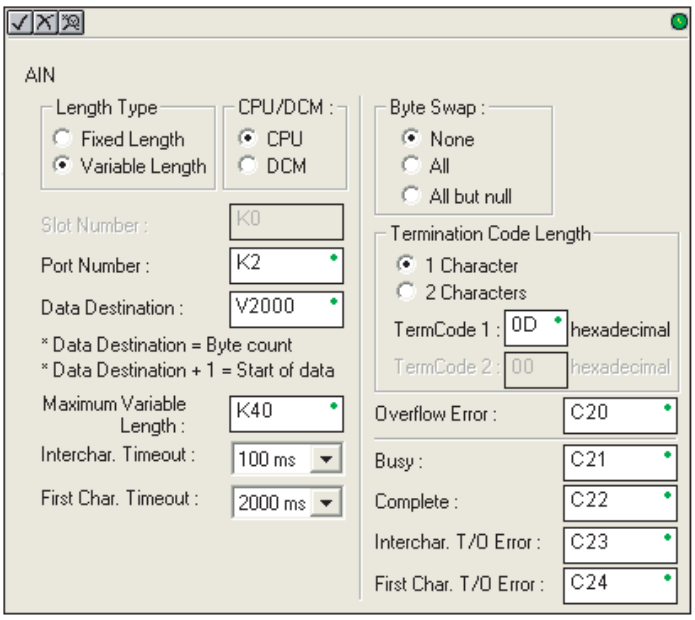
Ejemplo de longitud fija cuando el PLC está leyendo el puerto continuamente y el tiempo de operación no es crítico.



Ejemplo de longitud fija cuando el tiempo de transmisión carácter a carácter es crítico.



Configuración AIN de longitud variable:



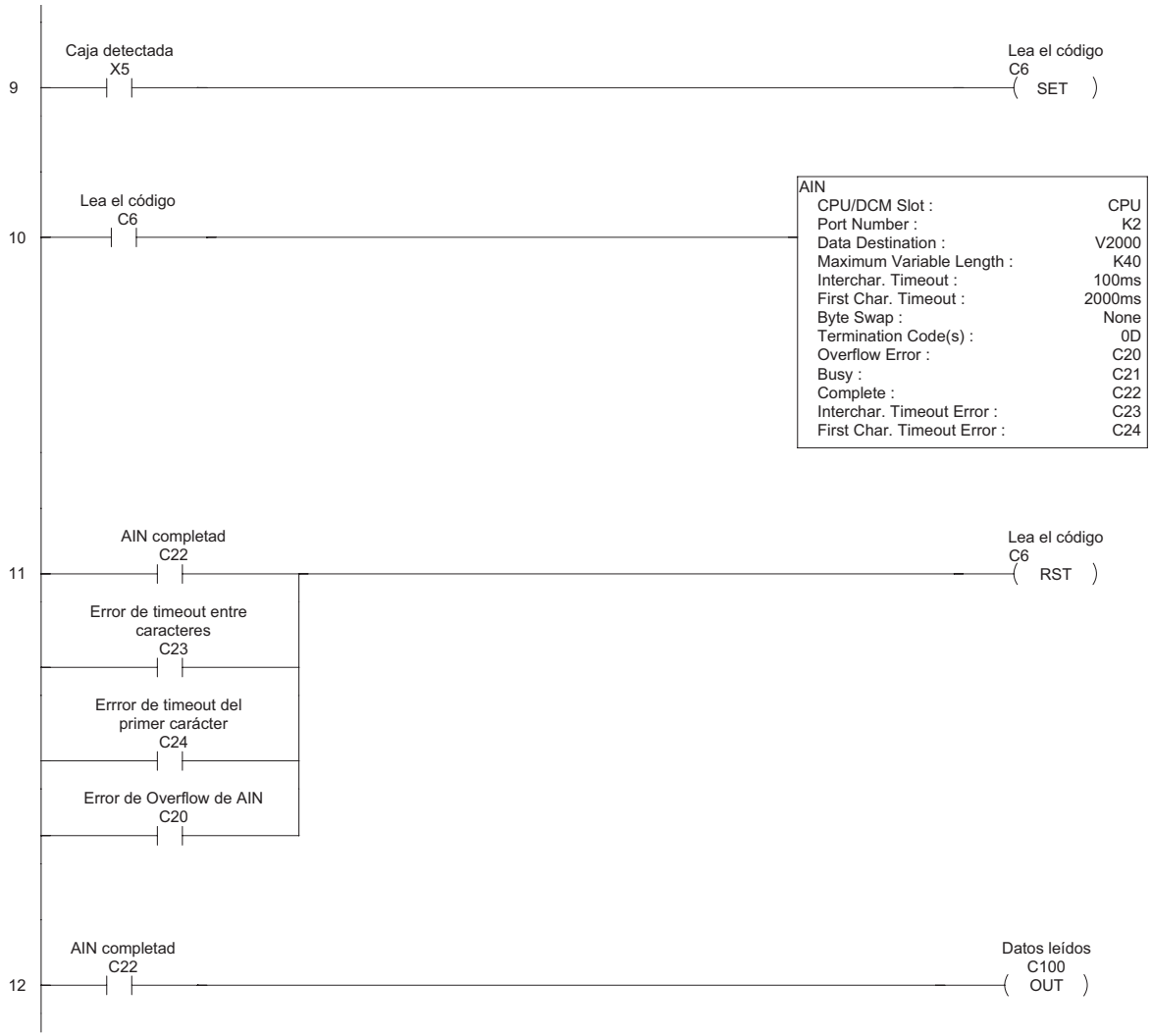
- **Length Type**: Seleccione Variable Length de acuerdo a la longitud de la cadena ASCII que será enviada al puerto del PLC.
- **CPU/DCM** : especifica si el maestro lee datos desde el puerto 2 o desde el módulo D0-DCM.
- **Port Number**: Siempre use el puerto No. 2 (K2)
- **Data Destination**: Especifique donde será colocada la cadena ASCII en la memoria.
- **Maximum Variable Length**: Especifique, en bytes, la máxima longitud de la cadena ASCII que será recibida por el puerto.
- **Inter–character Timeout**: si el período de los caracteres ASCII entrantes excede el tiempo ajustado, será activado el error de Timeout. No será almacenada ninguna información en la localización de memoria de destino. El bit será desactivado cuando el renglón donde está la instrucción está desactivado. Un valor de 0 ms elimina esta función.
- **First Character Timeout**: Si el período transcurrido entre cuando AIN está activado hasta el tiempo que el primer carácter es recibido es mayor que el valor prefijado, el bit de First Carácter Timeout será activado. Este bit será desactivado cuando la instrucción AIN sea desactivada. Un valor de 0 ms elimina esta función.
- **Byte Swap**: Intercambia el byte más alto con el más bajo en cada memoria de la cadena ASCII de longitud variable. Vea la instrucción SWAPB para las detalles.
- **Termination Code Length**: consiste de uno o dos caracteres como valor hexadecimal. Vea la tabla ASCII en el apéndice G.
- **Overflow Error Bit**: ES colocado ON cuando los datos ASCII recibidos exceden la longitud variable máxima especificada.
- **Busy Bit**: Este bit está ON cuando la instrucción AIN está recibiendo datos ASCII.
- **Complete Bit**: Se coloca ON cuando los datos ASCII han sido recibidos hasta que se termina de recibir los caracteres de terminación y OFF cuando los bits que hacen real la instrucción AIN están desactivados.
- **Inter–character Timeout Error Bit**: Se coloca ON cuando el Character Timeout ha sido sobrepasado. Vea la explicación de Character Timeout arriba.
- **First Character Timeout Error Bit**: Se coloca ON cuando el Character Timeout ha sido sobrepasado. Vea la explicación de Character Timeout arriba.

Parámetro	
Destino de los datos	Todas las memorias V
Longitud variable	K1–128
Bits: Busy, Complete, Timeout Error, Overflow	C0–3777

Ejemplo de longitud variable con AIN

Ejemplo de AIN con longitud variable usado para leer códigos de barras en cajas

En este ejemplo se tiene un lector de código de barras que lee el código de una caja de cartón sobre una correa transportadora detectado por un sensor photoelectrico conectado a la entrada X5. Los datos son colocados en formato ASCII en el conjunto de 40 registros consecutivos que comienzan en V2000.



La instrucción ASCII Find (AFIND)

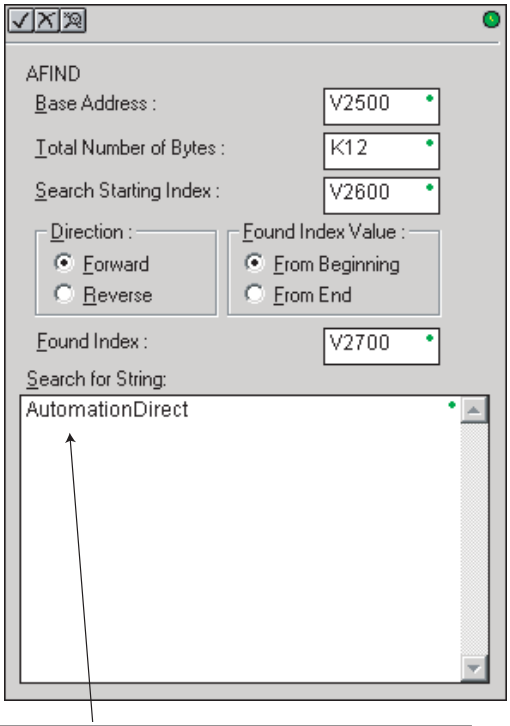
DS5	Usado
HPP	N/A

La instrucción AFIND localiza una cadena ASCII específica o una porción de ella en un rango de registros y coloca un número en una memoria especificada, número que corresponde al número del byte donde se encontró la cadena deseada. Este número se llama Found Index.

Otras características incluyen número Search Starting Index (Índice de inicio de búsqueda) para saltar los bytes que no son necesarios antes de iniciar la operación de búsqueda, búsqueda en la dirección Forward o Reverse, es decir, para adelante o para atrás y por último la selección de cual es el lado de donde se comienza a contar el valor Found Index. (From Beginning o From End)

Aquí está la definición de cada uno de los parámetros:

- **Base Address** (Dirección base): Define el comienzo de una tabla de memorias donde está almacenada la cadena ASCII.
- **Total Number of Bytes** (Cantidad de bytes): Define el número total de bytes que serán vistos para encontrar el texto ASCII deseado.El número máximo es 128 bytes.
- **Search Starting Index** (Indice de búsqueda): Define hasta que byte se va a saltar (con respecto a la dirección base) antes de iniciar la búsqueda.
- **Direction - Forward**: es la dirección de inicio para buscar la cadena desde memorias de localización baja para memorias de localización alta. **Reverse** es la dirección de inicio para buscar la cadena desde memorias de localización alta para memorias de localización baja.
- **Found Index Value**(# de índice encontrado): Define si el byte inicial o final de la cadena ASCII será cargado en la memoria Found Index.
- **Found Index**: Define la dirección de memoria donde será almacenado el Found Index number. Si el texto deseado no es encontrado el contenido de esta memoria será FFFF.
- **Search for String**: La cadena a ser buscada, con hasta 128 caracteres.

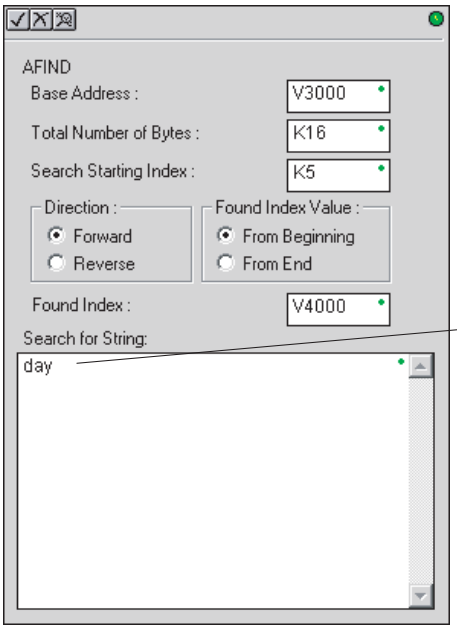


Parámetro	Rango del DL06
Dirección base	Toda la memoria V
Número total de bytes	Toda la memoria V o K1–128
Search Starting Index	Toda la memoria V o K0–127
Found Index	Toda la memoria V

NOTA: No se necesitan comillas en el elemento Search for String Las comillas son caracteres válidos que AFIND puede buscar.

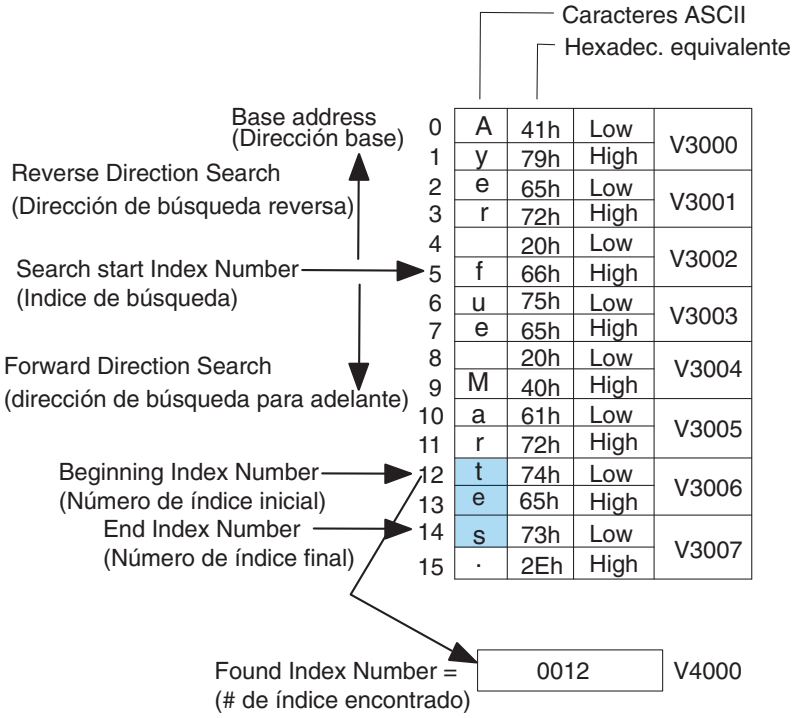
Ejemplo de búsqueda con AFIND

En el ejemplo siguiente, se usa la instrucción AFIND para buscar la porción "tes" en la palabra "Martes" en el texto ASCII "Ayer fue Martes", que ha sido colocado en una tabla de memorias. Note que el valor Search Starting Index (K)3 combinado con un Forward Direction Search es usado para prevenir que se encuentre esta secuencia de texto antes del tercer caracter. El Found Index number será colocado en V4000.



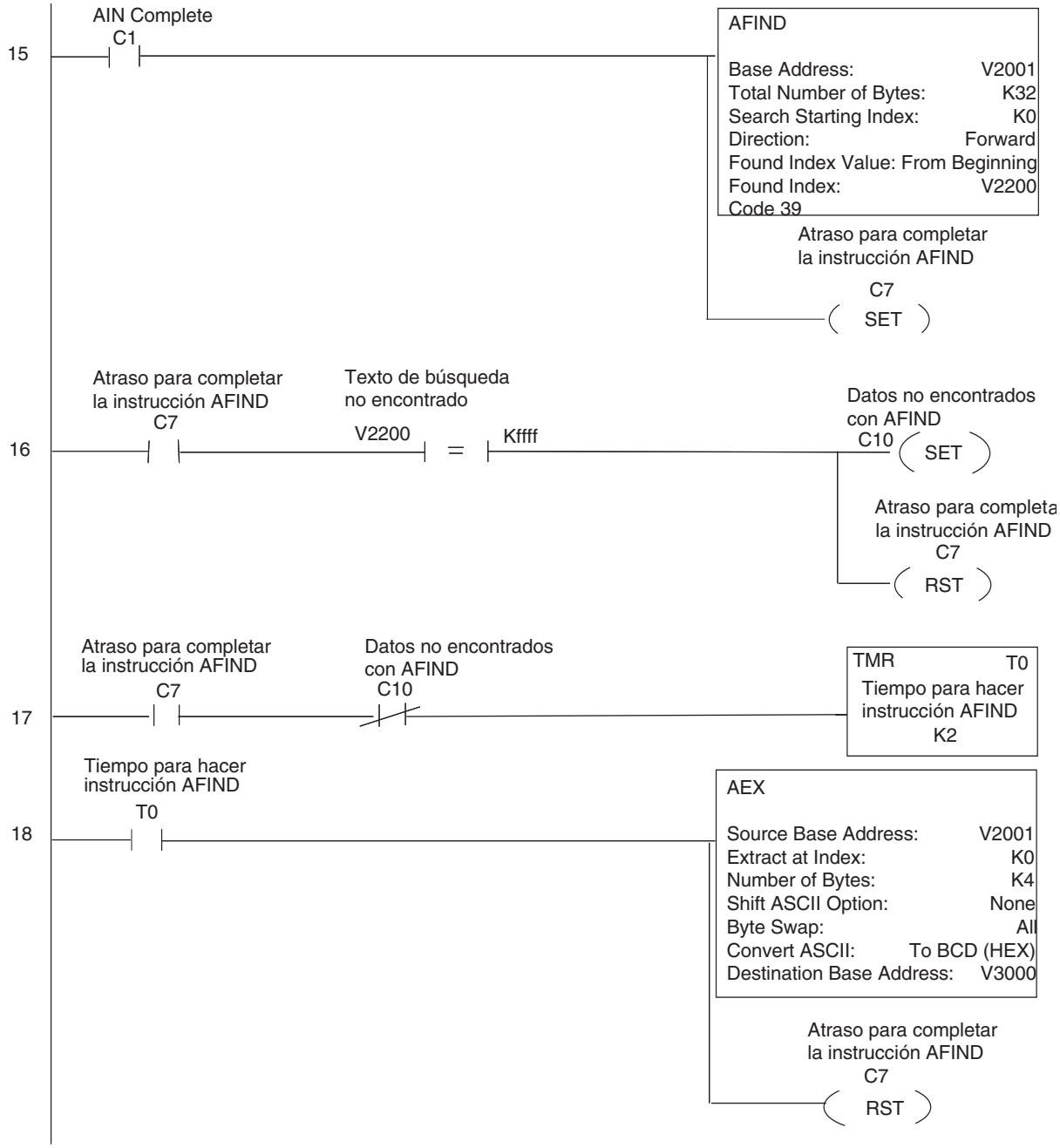
Note que no se colocan comillas alrededor del texto de búsqueda (Search String). Use comillas solamente si son realmente parte del texto de búsqueda (Search for String)

Quizás necesite hacer un "byte swap" en la cadena ASCII buscada, dependiendo de como fueron colocados los datos ASCII.



Ejemplo de instrucción AFIND combinado con instrucción AEX

Se puede usar el bit Complete de una instrucción AIN para activar una instrucción AFIND para encontrar un conjunto de texto ASCII. Cuando éste sea encontrado, la instrucción AEX puede usarse para extraer el texto localizado. Vea el ejemplo a continuación.



La instrucción ASCII Extract (AEX)

DS5	Usado
HPP	N/A

La instrucción ASCII Extract (AEX) extrae un número especificado de bytes de datos ASCII de una tabla de memoria y la coloca en otra tabla.

Otras características incluyen

Extract at Index que es iniciar la extracción a un número prefijado de bytes para saltar bytes no necesarios antes de comenzar la operación de extracción.

Shift ASCII option, que permite desplazar los datos extraídos para conveniencia

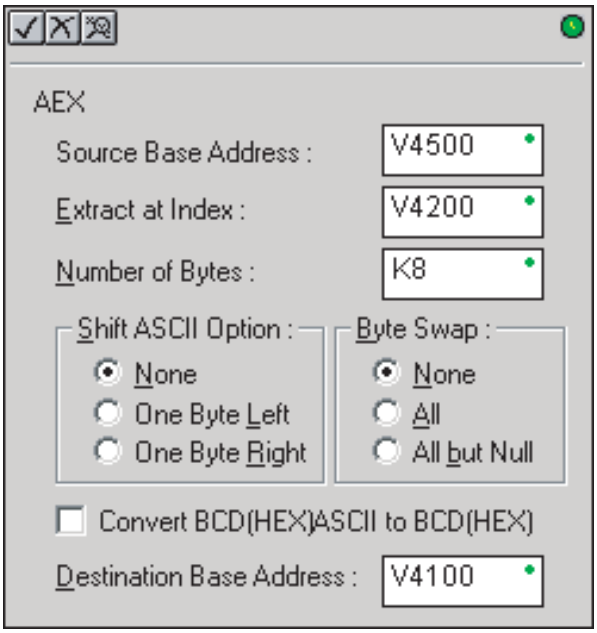
Byte Swap que es intercambio de bytes en una palabra

Convertir datos a un número BCD.

Aquí está la definición de cada uno de los parámetros:

- **Source Base Address** (Dirección de la tabla fuente): Define el comienzo de la tabla de memorias donde la cadena ASCII está almacenada.
- **Extract at Index**: Define a que byte va a saltar (siendo el inicio el Source base Address) antes de extraer los datos.
- **Number of Bytes**: define el número de bytes a ser extraído.
- **Shift ASCII Option**: : Desplaza todos los datos extraídos un byte a la izquierda o a la derecha para remover caracteres no deseados, si fuera necesario.
- **Byte Swap**: Intercambia el byte más alto con el más bajo en cada palabra de memoria de los datos extraídos. Vea la instruccion SWAPB para más detalles.
- **Convert BCD**(Hex) ASCII a BCD (Hex): Esta selección permite convertir datos numéricos ASCII a números Hexadecimal.
- **Destination Base Address**: Define la dirección de memoria donde serán almacenados los datos extraídos.

Vea un ejemplo en la página anterior.

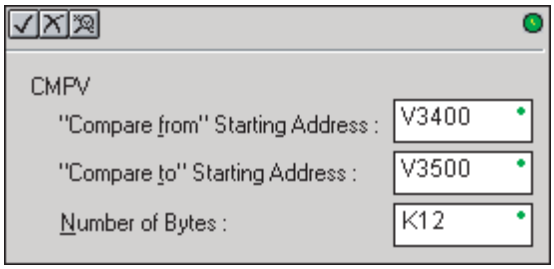


Parámetro	Rango del DL06	
Dirección fuente	**Dirección fuente**	
Extraiga al "Index"	**Extraiga al "Index"**	
Número de bytes Cuando "convert BCD(Hex) ASCII" no esté marcado.	Rango constante: K1–128	Memoria V que contiene el valor BCD: 1–128
Number of Bytes Cuando "convert BCD(Hex) ASCII" está marcado	Rango constante: K1–4	Memoria V que contiene el valor BCD: 1–4
Dirección de destino	Toda la memoria V	

La instrucción ASCII Compare (CMPV)

DS5	Usado
HPP	N/A

La instrucción CMPV compara dos tablas de memoria. Esta instrucción compara cualquier tipo de datos (ASCII a ASCII< BCD a BCD, etc.) de una tabla de memorias a otra tabla de memorias por una longitud de bytes definida. Es resultado de la comparación es dado por el relevador especial SP61.



Si SP61 es 1, el resultado de la comparación es igual.

Si SP61 es 0, el resultado de la comparación no es igual.

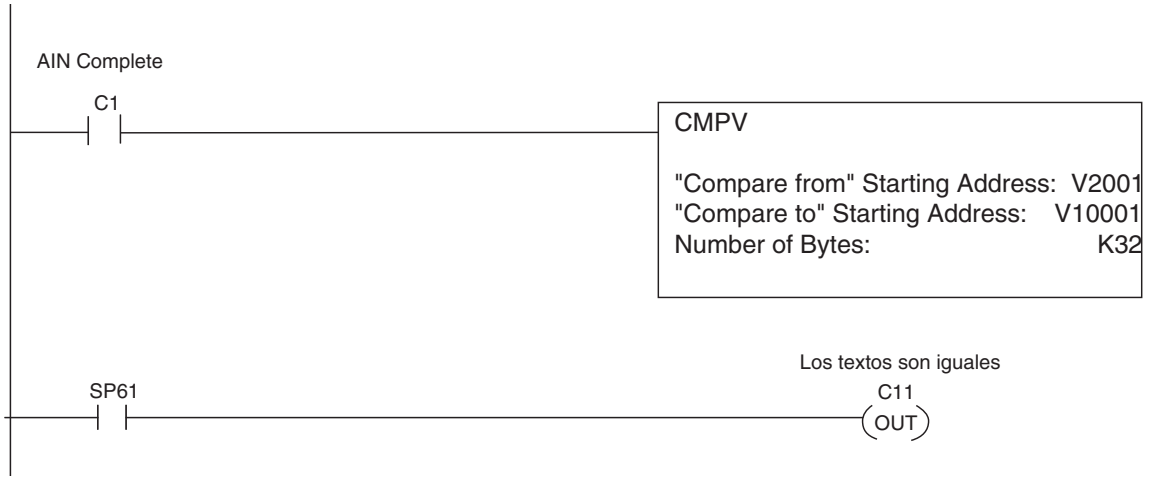
Aquí está la definición de cada uno de los parámetros:

- "**Compare from" Starting Address**: Esta dirección define el comienzo de la tabla de memorias con un texto ASCII a la cual será comparada la segunda tabla de memorias.
- "**Compare to" Starting Address**: Esta dirección define el comienzo de la segunda tabla de memorias donde está la cadena ASCII.
- **Number of Bytes**: Número define la longitud de cada tabla de memoria a ser comparada.

Parámetro	Rango del DL06
Compare desde la dirección inicial	Toda la memoria V
Compare a la dirección inicial	Toda la memoria V
Número de bytes	K0–127

Ejemplo de CMPV

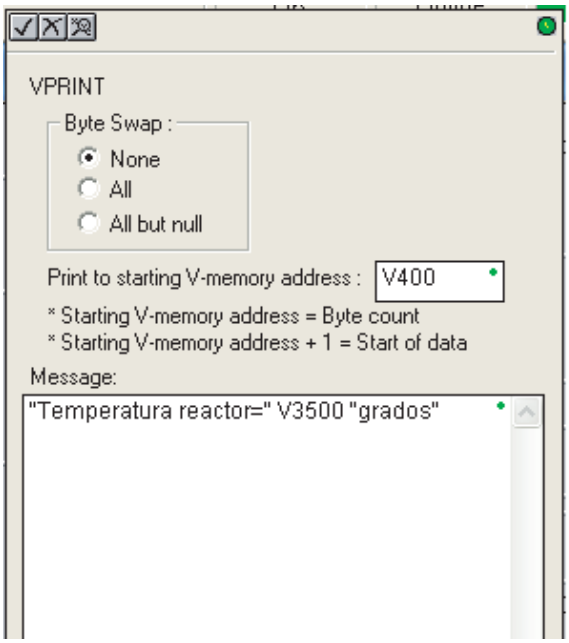
La instrucción CMPV se ejecuta cuando la instrucción AIN ha sido completada. Si el resultado de la comparación es igual, SP61 se hará ON y C11 será activado.



La instrucción ASCII Print a V–memory (VPRINT)

DS5	Usado
HPP	N/A

La instrucción VPRINT escribirá una cadena ASCII definida en uno de los campos de la instrucción en una tabla de memorias.



Esta instrucción puede intercambiar bytes, puede suprimir ceros a la izquierda, convertir espacios a ceros, usar fecha en formato EE.UU., europeo o asiático y horas en formato de 12 horas o de 24 horas.

Aquí está la definición de cada uno de los parámetros:

- **Byte Swap**: Este parámetro intercambia el byte más alto con el más bajo en cada palabra de la memoria donde la cadena ASCII es escrita, si es seleccionado ALL; para no hacer este intercambio seleccione None.
- **Print a Starting V–memory Address**: Este parámetro define el comienzo de la tabla de memorias donde será colocada la cadena ASCII por la instrucción VPRINT. La primera memoria de la tabla contendrá la longitud en bytes de la cadena ASCII. La segunda memoria y las que siguen contendrán los datos ASCII de la cadena impresa en la tabla.
- **Message**: Aquí en "mensaje" se escribe el texto a ser almacenado en la tabla de memorias, entre comillas (""); Acepta contenidos numéricos de memorias o fechas y hora.

NOTA: *Starting V-memory Address (Dirección de memoria inicial) es la primera memoria V de la serie de memorias especificadas que contendrá la longitud de la secuencia de texto ASCII en bytes. La memoria V +1 y las memorias subsecuentes contendrán la secuencia de texto ASCII que se imprime a la memoria V.*

Parámetro	Rango del DL06
Imprima a la dirección inicial de memoria V	Todo el rango de memoria V

Colocación de fecha y hora con VPRINT– Pueden ser usados los códigos mostrados en la tabla de abajo en la secuencia de texto con VPRINT ASCII para "imprimir a la memoria" la fecha y hora corriente.

#	Character code	Date / Time Stamp Options
1	_date:us	Norma americana (Mes/día/año con 2 dígitos)
2	_date:e	Norma europea (día/mes/año con 2 dígitos)
3	_date:a	Norma asiática (año con 2 dígitos/mes. Día)
4	_time:12	Norma de 12 horas (0-12:minutos AM/PM)
5	_time:24	Norma de 24 horas (0-24:minutos)

Modificadores de números contenidos en memoria – Los siguientes modificadores de números pueden ser usados en un mensaje VPRINT para almacenar el número en formato entero o real. Puede usar el número contenido en una memoria V sin modificador o con el modificador de tipo de datos después de ":". Los tipos de datos son mostrados en la tabla abajo: El código debe ser escrito con mayúsculas.

NOTA: Debe colocar un espacio antes y después de la dirección de memoria V para separarla de la cadena de texto. Si no hace ésto aparece el error 499.

#	Character code	Descripción
1	none	Binario de 16 bits (Número decimal)
2	: B	BCD de 4 dígitos
3	: D	Binario de 32 bits (Número decimal)
4	: D B	BCD de 8 dígitos
5	: R	Número de punto flotante (Número real)
6	: E	Número de punto flotante (Número real con exponente)

Ejemplos:

V2000 imprime datos binarios en V2000 como decimal

V2000 : B imprime datos BCD en V2000

V2000 : D imprime datos binarios en V2000 y V2001 como decimal

V2000 : D B imprime datos en V2000 y V2001

V2000 : R imprime datos de punto flotante en V2000 y V2001como número real

V2000 : E imprime datos de punto flotante en V2000/V2001como número real con exponente

Los siguientes modificadores pueden ser agregados a los anteriores para suprimir o convertir ceros a la izquierda o espacios. El código debe ser escrito con mayúsculas.

#	Código de caracteres	Descripción
1	S	Elimina espacios a la izquierda
2	C0	Convierte espacios a la izquierda a ceros
3	0	Elimina ceros a la izquierda

Ejemplo con V2000 = 0018 (Formato binario)

Memoria con modificador	Número de caracteres			
	1	2	3	4
V2000	0	0	1	8
V2000:B	0	0	1	2
V2000:B0	1	2		

Ejemplo con V2000 = sp sp 0018 (Formato binario) donde "sp" significa espacio

Memoria con modificador	Número de caracteres			
	1	2	3	4
V2000	sp	sp	1	8
V2000:B	sp	sp	1	2
V2000:BS	1	2		
V2000:BC0	0	0	1	2

Modificadores de texto contenido en memoria – El siguiente modificador de longitud de texto puede ser usado en un mensaje VPRINT para almacenar el texto a partir de la primera o consecutivas direcciones de memoria. Use el signo "%" seguido del número de caracteres en la cadena que Ud. desea imprimir. Si usa "0" como número de caracteres, la instrucción leerá la cantidad de caracteres desde la primera dirección. Luego comenzará en la próxima dirección y leerá la cantidad leída de caracteres ASCII indicada allí desde la memoria indicada.

Ejemplo:

V2000 % 16 Se transfieren 16 caracteres en V2000 hasta V2007

V2000 % 0 Se transfieren XX caracteres a partir de V2001 (XX es determinado por el número almacenado en V2000).

Modificadores de bits contenidos en memoria– Los siguientes modificadores de estados de bits en una memoria o en un relevador de control pueden ser usados en un mensaje VPRINT para almacenar un cierto formato. Puede usarse la dirección del relevador o la memoria seguida

#	Formato de datos	Descripción
1	Sin modificador	Imprime un "1" para el estado ON; "0" para OFF
2	: BOOL	Imprime "TRUE" para el estado ON; "FALSE" para OFF
3	: ONOFF	Imprime "ON" para el estado ON; "OFF" para OFF

de "." Y el número de bit con el modificador de tipo de datos después de ":". Los tipos de datos son mostrados en la tabla abajo: El código debe ser escrito con mayúsculas.

Ejemplos:

V2000 . 15 imprime el estado del bit 15 en V2000 en el formato 1 o 0.

C100 imprime el estado de C100 en el formato 1 o 0.

C100 : BOOL imprime el estado de C100 en el formato TRUE o FALSE

C100 : ON/OFF imprime el estado de C100 en el formato ON u OFF

V2000.15 : imprime el estado del bit 15 en V2000 en el formato TRUE o FALSE.

El máximo número de caracteres que se puede operar con VPRINT es 128. En la lista a continuación es mostrado el número requerido por cada elemento, sin importar si se usa o no los modificadores :S, :C0 o :0.

Tipo de elemento	Cantidad máxima de caracteres
Texto, 1 carácter	1
Binario de 16 bit	6
Binario de 32 bit	11
BCD de 4 dígitos	4
BCD de 8 dígitos	8
Punto flotante (Número real)	3
Punto flotante (real con exponente)	13
Texto en memoria V	2
Bit (formato 1/0)	1
Bit (formato TRUE/FALSE)	5
Bit (formato ON/OFF)	3

Modificadores de caracteres especiales – Los siguientes modificadores pueden ser usados en un mensaje VPRINT para almacenar caracteres especiales. Caracteres en una cadena ASCII son definidos como los caracteres contenidos entre comillas (") en el campo de mensaje en VPRINT. Dos números hexadecimales que sean precedidos por el signo $ significa un código de caracteres ASCII de 8 bits. También, dos caracteres precedidos por el signo $ es interpretado de acuerdo a la siguiente tabla:

#	Código de carácter	Descripción
1	$$	Signo dólar ($)
2	$”	Comillas (")
3	$L o $l	Line feed (LF) usado por ejemplo con impresoras
4	$N o $n	Carriage return line feed (CRLF)
5	$P o $p	Form feed
6	$R o $r	Carriage return (CR)
7	$T o $t	Tab

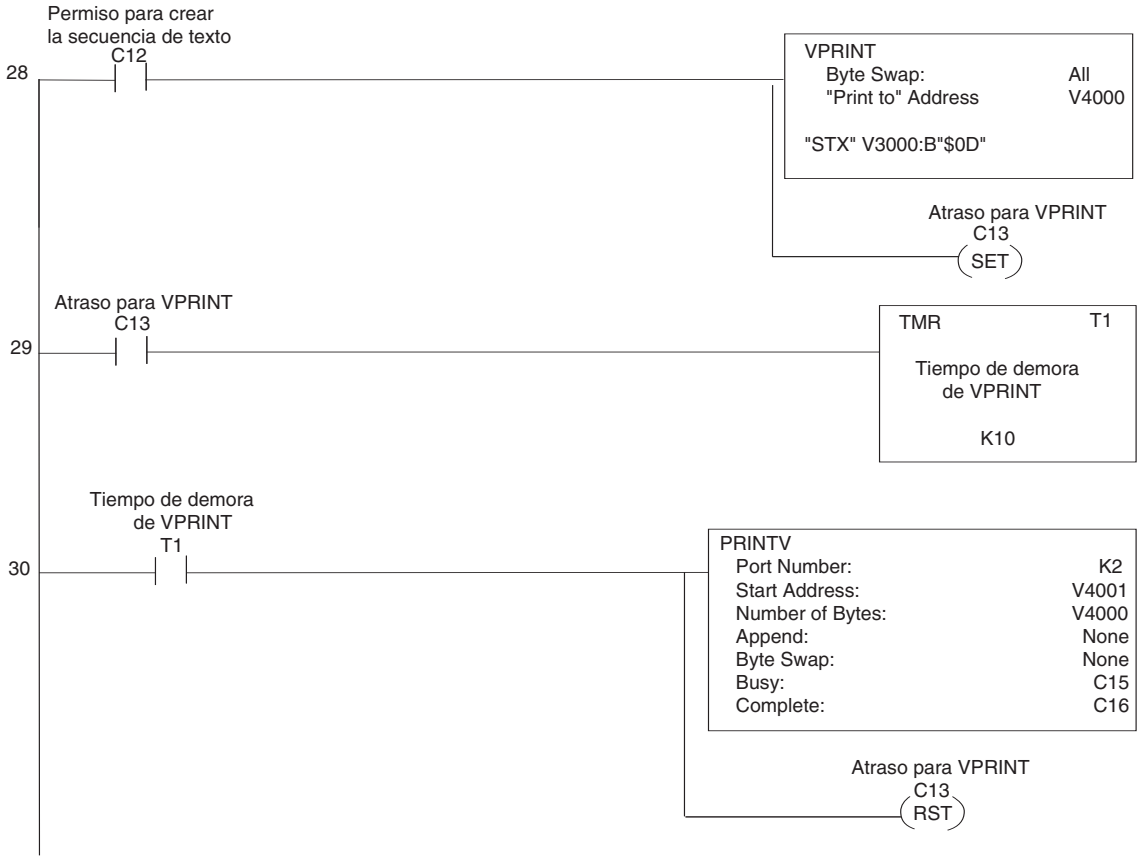
Los siguientes ejemplos muestran varias convenciones de sintaxis y la longitud de la salida a la impresora.

” ”	Longitud 0 sin ningún carácter
”A”	Longitud 1 con carácter A
” ”	Longitud 1 con espacio en blanco
” $” ”	Longitud 1 con comillas
” $ R $ L ”	Longitud 2 con un CR y un LF
” $ 0 D $ 0 A ”	Longitud 2 con un CR y un LF
” $ $ ”	Longitud 1 con un carácter $

Al imprimir una línea de texto, Ud. deberá incluir comillas antes y después de la cadena de texto. Aparecerá el error 499 en la CPU si la instrucción contiene texto inválido o no contiene comillas. Es importante probar los datos de la instrucción VPRINT durante el desarrollo del programa.

Ejemplo de VPRINT combinado con la instrucción PRINT V

Se usa aquí la instrucción VPRINT para crear una cadena de caracteres en la memoria Vque se inicia en V4000. Luego se usa la instrucción PRINTV para generar una salida de caracteres ASCII por el puerto 2 del PLC.

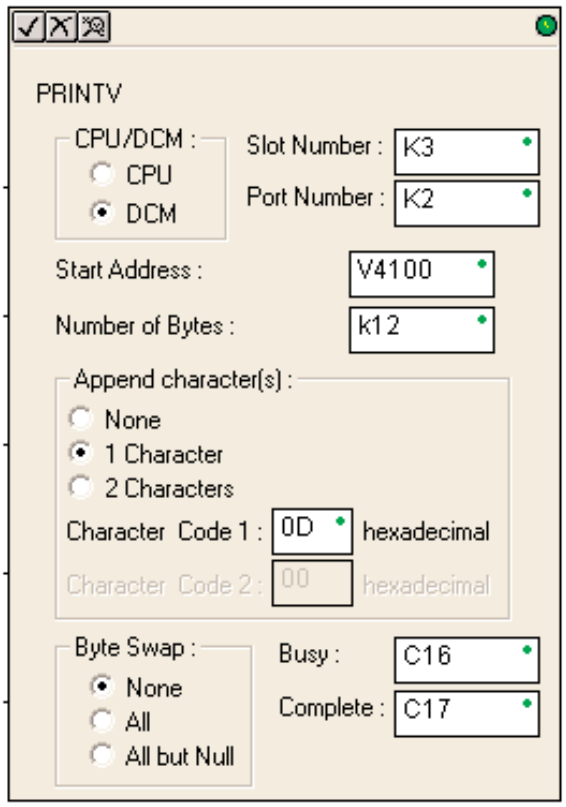


La instrucción ASCII Print from V–memory (PRINTV)

DS5	Usado
HPP	N/A

La instrucción PRINTV enviará un texto ASCII de longitud determinada definida en una tabla de memorias saliendo por el puerto 2 de la CPU o del módulo D0-DCM.

Esta instrucción puede agregar caracteres definidos por el usuario después de un texto de datos para aparatos que requieren caracteres específicos de terminación, puede intercambiar bytes y usar indicaciones definidas por el usuario para los estados Busy y Complete.



Aquí está la definición de cada uno de los parámetros:

- **CPU/DCM** : especifica si el maestro lee datos desde el puerto 2 o desde el módulo D0-DCM.
- **Port Number**: Para el PLC DL06 debe ser siempre puerto 2 (K2)
- **Start Address** (Dirección inicial) : define el comienzo de una tabla que contiene la cadena ASCII a ser transferida.
- **Number of Bytes** (cantidad de bytes): define la longitud de la cadena a ser transferida.
- **Append Characters**: Define los caracteres ASCII a ser agregados al final de la cadena para aparatos que necesiten caracteres de terminación. Debe ser escrito en hexadecimal.
- **Byte Swap**: Este parámetro intercambia el byte más alto con el más bajo en cada palabra de la memoria donde la cadena ASCII mientras imprime. Vea la instrucción SWAPB para más detalles.
- **Busy Bit**: Este bit, a ser definido por el usuario, cambia a ON mientras la instrucción está imprimiendo datos ASCII. Muestra que la CPU está ocupada.
- **Complete Bit**: Este bit, a ser definido por el usuario, cambia a ON cuando la instrucción terminó de imprimir y es OFF cuando los bits de permiso de ejecución de la instrucción PRINTV están desactivados (Es decir, el renglón es falso).

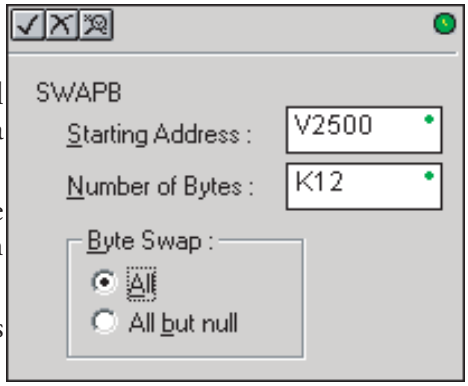
Vea el ejemplo de la página anterior para ver el uso de esta instrucción.

Parámetro	Rango en el DL06
CPU/DCM	Ranura 1 a 4
Port number (número de puerto)	Puerto 2 (K2)
Start Address (Dirección inicial)	Todas las memorias V
Number of bytes (Cantidad de Bytes)	Toda la memoria o k1–128
Bits: Busy, Complete	C0–3777

La instrucción ASCII Swap Bytes (SWAPB)

DS5	Usado
HPP	N/A

La instrucción SWAPB intercambia posiciones de bytes (del byte más alto al más bajo y viceversa) en cada memoria de la tabla que contiene una cadena ASCII (o un conjunto de datos no necesariamente ASCII)



Aquí está la definición de cada uno de los parámetros:

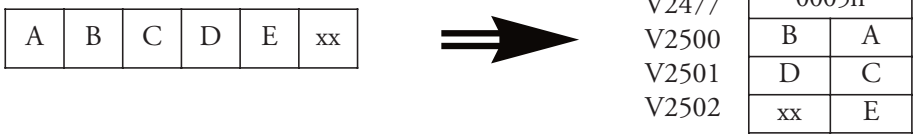
- **Starting Address**: Este campo define la dirección del comienzo de la tabla de memorias que usará la instrucción para intercambiar bytes.
- **Number of Bytes**: Este campo define el número de bytes a ser intercambiados a partir de la dirección Starting Address.
- **Byte Swap**: Define si será intercambiados todos los bytes o todos menos el primero.

Parámetro	Rango del DL06
Starting Address	Cualquier memoria V
Number of Bytes	Todas las memorias V o K1–128

Indicadores	Descripción
SP53	On si la CPU no puede ejecutar la instruccción.
SP71	On cuando un valor usado por la instrucción es inválido.

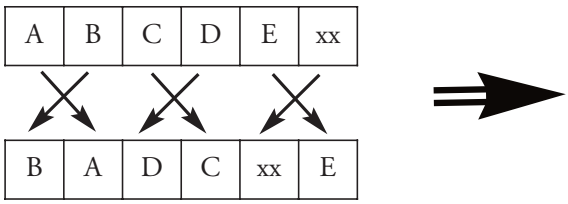
Preferencias de Byte Swap

No Byte Swapping (sin intrercambio de bits)
(AIN, AEX, PRINTV, VPRINT)



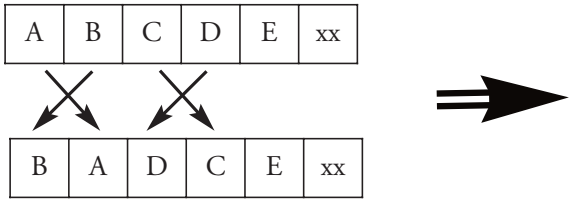
	Byte Alto	Byte Bajo
V2477	0005h	
V2500	B	A
V2501	D	C
V2502	xx	E

Byte Swap All (Intrecambie todo)



	Byte Alto	Byte Bajo
V2477	0005h	
V2500	A	B
V2501	C	D
V2502	E	xx

Byte Swap All but Null (Intercambie todo excepto el carácter NULL)



	Byte High	Byte Low
V2477	0005h	
V2500	B	A
V2501	D	C
V2502	xx	E

Ejemplo de SWAPB

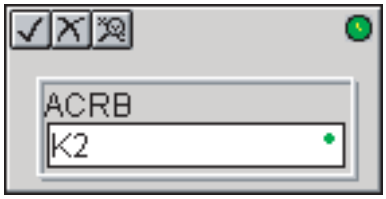
El bit Complete de AIN se usa para activar la instrucción SWAPB. Use la instrucción STRPD para que la instrucción SWAPB sea ejecutada en un barrido solamente.

La instrucción ASCII Clear Buffer (ACRB)

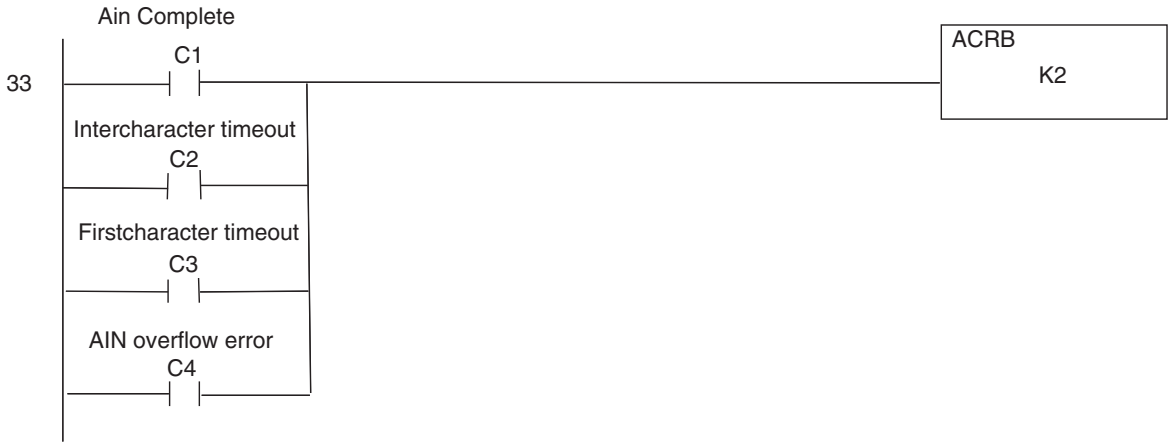
DS5	Usado
HPP	N/A

Esta instrucción limpia el buffer (memoria internediaria de almacenaje temporario) de recibimiento del puerto 2 del PLC DL06 de los caracteres recibidos.

Ejemplo de ACRB



El bit AIN Complete o los bits de diagnóstico de AIN se utilizan para limpiar el buffer o la memoria de almacenaje intermediaria de caracteres ASCII.



Esta página ha sido dejada en blanco intencionalmente.

Instrucciones (IBox) o Cuadros Inteligentes

Las instrucciones designadas comúnmente Iboxes, enumeradas en esta sección son nuevas instrucciones disponibles al usar ***DirectSOFT5*** para programar su PLC DL06 (el PLC DL06 requiere la versión v2.10 de firmware o mas nuevo para usar las nuevas funciones de ***DirectSOFT5***). Para más información sobre ***DirectSOFT5*** y para bajar una versión gratuita, visite por favor nuestro sitio de Internet en: www.automationdirect.com.

IBoxes de ayuda de señales analógicas		
Instrucción	**Ibox #**	**Página**
Analog Input / Output Combo Module PunteroSetup (ANLGCMB)	IB-462	5-232
Analog Input Module PunteroSetup (ANLGIN)	IB-460	5-234
Analog Output Module PunteroSetup (ANLGOUT)	IB-461	5-236
Analog Scale 12 Bit BCD to BCD (ANSCL)	IB-423	5-238
Analog Scale 12 Bit Binary to Binary (ANSCLB)	IB-403	5-239
Filter Over Time - BCD (FILTER)	IB-422	5-240
Filter Over Time - Binary (FILTERB)	IB-402	5-242
Hi/Low Alarm - BCD (HILOAL)	IB-421	5-244
Hi/Low Alarm - Binary (HILOALB)	IB-401	5-246

IBoxes de ayuda de señales discretas		
Instrucción	**Ibox #**	**Página**
Off Delay Timer (OFFDTMR)	IB-302	5-248
On Delay Timer (ONDTMR)	IB-301	5-250
One Shot (ONESHOT)	IB-303	5-252
Push On / Push Off Circuit (PONOFF)	IB-300	5-253

IBoxes de memorias		
Instrucción	**Ibox #**	**Página**
Move Single Word (MOVEW)	IB-200	5-254
Move Double Word (MOVED)	IB-201	5-255

IBoxes de aritmética		
Instrucción	**Ibox #**	**Página**
BCD to Real with Implied Decimal Point (BCDTOR)	IB-560	5-256
Double BCD to Real with Implied Decimal Point (BCDTORD)	IB-562	5-257
Math - BCD (MATHBCD)	IB-521	5-258
Math - Binary (MATHBIN)	IB-501	5-260
Math - Real (MATHR)	IB-541	5-262
Real to BCD with Implied Decimal Point and Rounding (RTOBCD)	IB-561	5-263
Real to Double BCD with Implied Decimal Point and Rounding (RTOBCDD)	IB-563	5-264
Square BCD (SQUARE)	IB-523	5-265
Square Binary (SQUAREB)	IB-503	5-266
Square Real(SQUARER)	IB-543	5-267
Sum BCD Numbers (SUMBCD)	IB-522	5-268
Sum Binary Numbers (SUMBIN)	IB-502	5-269
Sum Real Numbers (SUMR)	IB-542	5-270

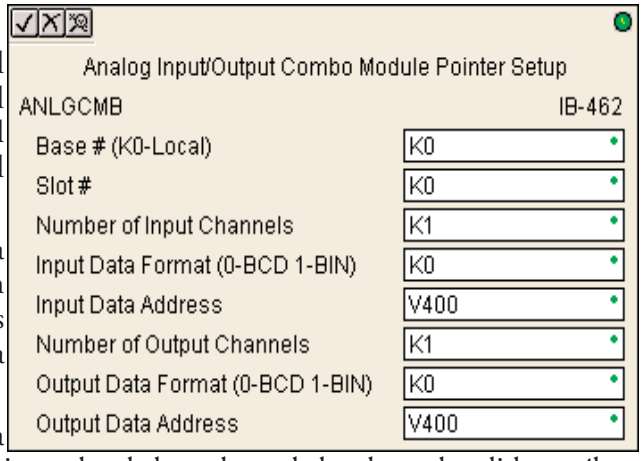
IBoxes de Comunicación		
Instrucción	**Ibox #**	**Página**
ECOM100 Configuration (ECOM100)	IB-710	5-272
ECOM100 Disable DHCP (ECDHCPD)	IB-736	5-274
ECOM100 Enable DHCP (ECDHCPE)	IB-735	5-276
ECOM100 Query DHCP Setting (ECDHCPQ)	IB-734	5-278
ECOM100 Send E-mail (ECEMAIL)	IB-711	5-280
ECOM100 Restore Default E-mail Setup (ECEMRDS)	IB-713	5-281
ECOM100 E-mail Setup (ECEMSUP)	IB-712	5-286
ECOM100 IP Setup (ECIPSUP)	IB-717	5-290
ECOM100 Read Descripción (ECRDDES)	IB-726	5-292
ECOM100 Read Gateway Address (ECRDGWA)	IB-730	5-294
ECOM100 Read IP Address (ECRDIP)	IB-722	5-296
ECOM100 Read Module ID (ECRDMID)	IB-720	5-298
ECOM100 Read Module Name (ECRDNAM)	IB-724	5-300
ECOM100 Read Subnet Mask (ECRDSNM)	IB-732	5-302
ECOM100 Write Descripción (ECWRDES)	IB-727	5-304
ECOM100 Write Gateway Address (ECWRGWA)	IB-731	5-302
ECOM100 Write IP Address (ECWRIP)	IB-723	5-304
ECOM100 Write Module ID (ECWRMID)	IB-721	5-310
ECOM100 Write Name (ECWRNAM)	IB-725	5-312
ECOM100 Write Subnet Mask (ECWRSNM)	IB-733	5-314
ECOM100 RX Network Read (ECRX)	IB-740	5-316
ECOM100 WX Network Write(ECWX)	IB-741	5-319
NETCFG Network Configuration (NETCFG)	IB-700	5-322
Network RX Read (NETRX)	IB-701	5-324
Network WX Write (NETWX)	IB-702	5-327

IBoxes de CTRIO		
Instrucción	**Ibox #**	**Página**
CTRIO Configuration (CTRIO)	IB-1000	5-330
CTRIO Add Entry to End of Preset Table (CTRADPT)	IB-1005	5-332
CTRIO Clear Preset Table (CTRCLRT)	IB-1007	5-335
CTRIO Edit Preset Table Entry (CTREDPT)	IB-1003	5-338
CTRIO Edit Preset Table Entry and Reload (CTREDRL)	IB-1002	5-342
CTRIO Initialize Preset Table (CTRINPT)	IB-1004	5-346
CTRIO Initialize Preset Table (CTRINTR)	IB-1010	5-350
CTRIO Load Profile (CTRLDPR)	IB-1001	5-354
CTRIO Read Error (CTRRDER)	IB-1014	5-357
CTRIO Run to Limit Mode (CTRRTLM)	IB-1011	5-359
CTRIO Run to Position Mode (CTRRTPM)	IB-1012	5-362
CTRIO Velocity Mode (CTRVELO)	IB-1013	5-365
CTRIO Write File to ROM (CTRWFTR)	IB-1006	5-368

Configurador del módulo de entradas y salidas análogas (ANLGCMB) (IB-462)

DS5	Usado
HPP	N/A

Esta instrucción genera la lógica para configurar el método del puntero para un módulo análogo combinación de entradas y salidas en el primer barrido del PLC después de una transición de modo program a RUN.



La instrucción ANLGCMB determina el formato de datos y las direcciones del puntero basadas en el tipo de CPU, el número de la base y de la ranura del módulo.

La dirección de datos de entrada es la localización inicial de memoria V en donde serán almacenados los valores de los datos de entrada análoga y crea una localización para cada canal de entrada.

La dirección de datos de salidas es la localización inicial de memoria V de usuario en donde los valores de los datos de salidas análoga serán puestos por código ladder o un dispositivo externo, creando una localización para cada canal de salida.

Puesto que la lógica de IBox se ejecuta solamente en el primer barrido, la instrucción no puede tener ninguna lógica de entrada.

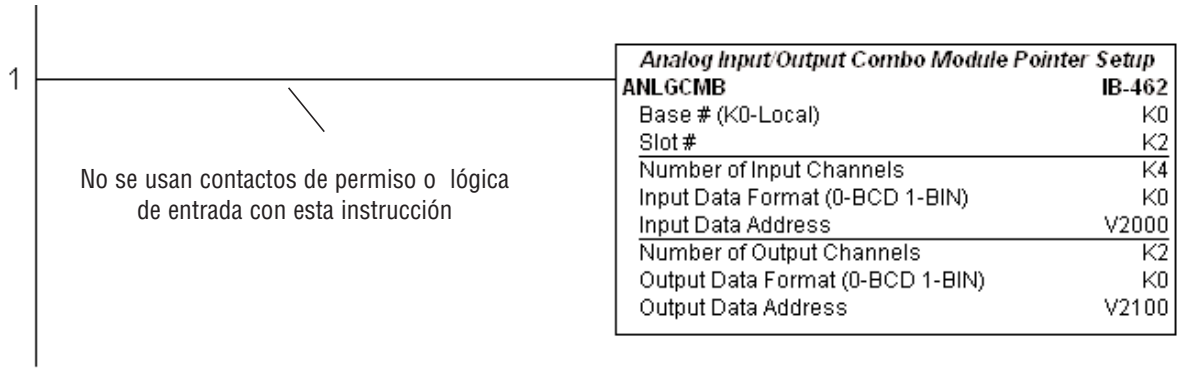
Parámetros ANLGCMB

- **Base #** (K0-Local): Debe ser 0 para PLC DL06.
- **Slot #** (Ranura): Especifica qué ranura de opción del PLC es ocupada por el módulo análogo (1-4)
- **Number of Input Channels**: Especifica el número de canales de entradas análogas a ser explorados.
- **Input Data Format** (0-BCD 1-BIN): Especifica el formato de datos de entradas análogas (BCD o binario) - el formato binario puede ser usado para exhibir datos en paneles de interface de operador.
- **Input Data Address**: Especifica la localización de memoria V inicial que será utilizada para almacenar los datos de entradas análogas.
- **Number of Output Channels**: Especifica el número de canales de salidas análogas que serán usados.
- **Output Data Format** (0-BCD 1-BIN): Especifica el formato de los datos de salidas análogas (BCD o binario).
- **Output Data Address**: Especifica la localización de memoria V inicial que será usada como origen de los datos de salidas análogas.

Parámetro	Rango del DL06
Base # (K0-Local) . . . . . K	K0 (Solamente base local)
Slot # . . . . . K	K1-4
Number of Input Channels . . . . . K	K1-8
Input Data Format (0-BCD 1-BIN) . . . . . K	BCD: K0; Binario: K1
Input Data Address V	Vea el mapa de memoria V del DL06 - Data Words
Number of Output Channels . . . . . K	K1-8
Output Data Format (0-BCD 1-BIN) . . . . . K	BCD: K0; Binario: K1
Output Data Address V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ANLGCMB

En el ejemplo siguiente, se usa la instrucción ANLGCMB para configurar el método del puntero para un módulo de combinación de E/S análogas que esté instalado en la ranura de opción 2. Se activan cuatro canales de entradas y los datos de salidas análogas serán escritos a V2000 - V2003 en formato BCD. Se activan dos canales de salidas y los valores análogos serán leídos en V2100 - V2101 en formato BCD.

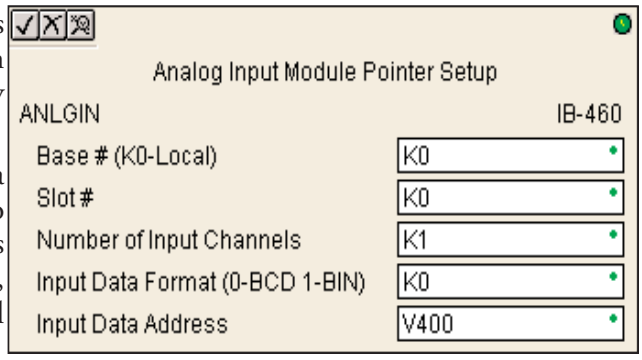


Configurador del módulo de entrada análoga (ANLGIN) (IB-460)

DS5	Usado
HPP	N/A

La configuración del módulo de entradas análogas genera la lógica para configurar el método del puntero para un módulo de entradas análogas en el primer barrido del PLC después de una transición de modo program a RUN.

Este IBox determina el formato de datos y las direcciones del puntero basadas en el tipo de CPU, el número de la base y de la ranura.



La dirección de los datos de entrada es la localización de memoria V de usuario inicial en donde serán almacenados los valores de datos de entradas análogas, creando una localización para cada canal de entrada.

Puesto que esta lógica se ejecuta solamente en el primer barrido, este IBox no puede tener ninguna lógica de entrada.

5

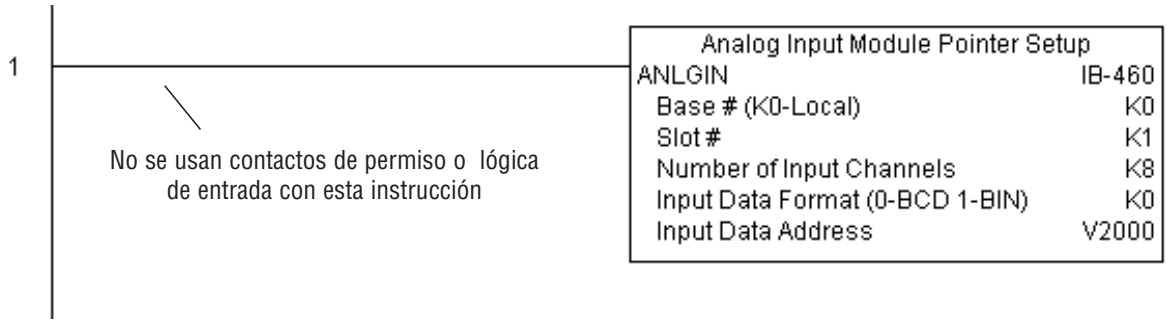
Parámetros ANLGIN

- **Base #** (K0-Local): Debe ser 0 para PLC DL06.
- **Slot #**: Especifica qué ranura de opción del PLC es ocupada por el módulo análogo (1-4)
- **Number of Input Channels**: Especifica el número de canales de entradas análogas a ser explorados.
- **Input Data Format** (0-BCD 1-BIN): Especifica el formato de datos de entradas análogas (BCD o binario) - el formato binario puede ser usado para exhibir datos en paneles de interface de operador.
- **Input Data Address**: Especifica la localización de memoria V inicial que será utilizada para almacenar los datos de entradas análogas.

Parámetro	Rango del DL06
Base # (K0-Local) . . . . . K	K0 (Solamente base local)
Slot # . . . . . K	K1-4
Number of Input Channels . . . . . K	K1-8
Input Data Format (0-BCD 1-BIN) . . . . . K	BCD: K0; Binario: K1
Input Data Address V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ANLGIN

En el ejemplo siguiente, se usa la instrucción ANLGIN para configurar el método del puntero para un módulo de entradas análogas que esté instalado en la ranura de opción 1. Se activan ocho canales de entradas y los datos análogos serán escritos a V2000 - V2007 en formato BCD.

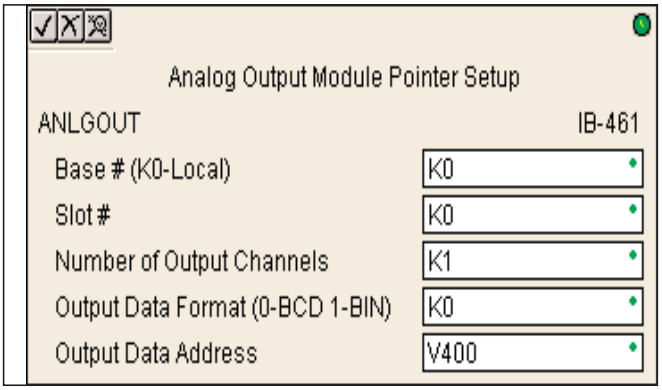


Configurador del módulo de salidas análogas (ANLGOUT) (IB-461)

DS5	Usado
HPP	N/A

La instrucción configurador del módulo de salidas análogas genera la lógica para configurar el método del puntero para un módulo de salidas análogas en el primer barrido del PLC después de una transición de modo program a RUN.

Este IBox determina el formato de datos y las direcciones del indicador basadas en el tipo de CPU, el número de la base y de la ranura.



La dirección de los datos de salidas es la localización inicial de memoria V de usuario en donde los valores de los datos de salidas análogas serán puestos por código ladder o un dispositivo externo, siendo una localización para cada canal de salida.

Puesto que esta lógica se ejecuta solamente en el primer barrido, este IBox no puede tener ninguna lógica de entrada.

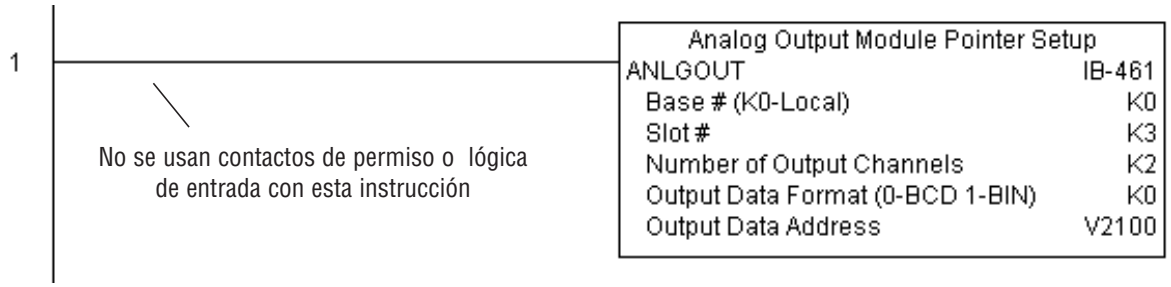
Parámetros ANLGOUT

- **Base #** (K0-Local): Debe ser 0 para PLC DL06.
- **Slot #**: Especifica qué ranura de opción del PLC es ocupada por el módulo análogo (1-4)
- **Number of Output Channels**: Especifica el número de canales de salidas análogas que serán usados.
- **Output Data For:** Especifica el formato de los datos de salidas análogas (BCD o binario).
- **Output Data Address**: Especifica la localización de memoria V inicial que será usada como origen de los datos de salidas análogas.

Parámetro	Rango del DL06
Base # (K0-Local) . . . . . K	K0 (local base only)
Slot # . . . . . K	K1-4
Number of Output Channels . . . . . K	K1-8
Output Data Format (0-BCD 1-BIN). . . . . K	BCD: K0; Binary: K1
Output Data Address V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ANLGOUT

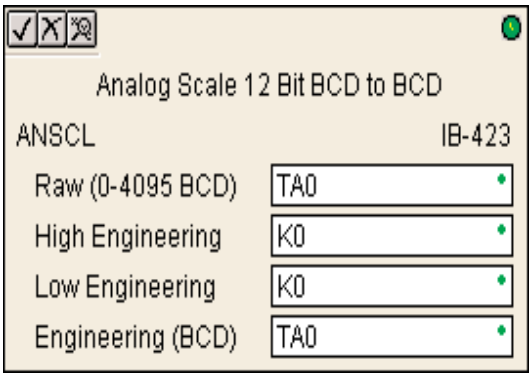
En el ejemplo siguiente, se utiliza la instrucción ANLGOUT para configurar el método del puntero para un módulo de salidas análogas que esté instalado en la ranura de opción 3. Son activados dos canales de salidas y los datos análogos serán leídos en V2100 - V2101 en formato BCD.



Escala de un valor análogo de 12 bits BCD a BCD (ANSCL) (IB-423)

DS5	Usado
HPP	N/A

Esta instrucción escala un valor análogo BCD de 12 bits (BCD 0-4095) en unidades de ingeniería BCD. Usted especifica el valor de la unidad de ingeniería más alto (cuando el valor sin escala es 4095), y el valor de ingeniería más bajo (cuando el valor sin escala es 0), y la dirección de memoria V de salida que usted desea poner el valor de unidad que dirige escalado. Las unidades de ingeniería se generan como BCD y pueden estar en el el rango completo de 0 a 9999 (Vea la instrucción ANSCLB - si sus unidades sin escala están en formato binario).



Observe que esta instrucción IBox trabaja solamente con valores unipolares sin escala positivos. No trabaja con valores bipolares ni con valores crudos de magnitud más signo.

Parámetros ANSCL

- **Raw** (0-4095 BCD): Especifica la localización de la memoria V donde está el valor unipolar sin escala de rango 0-4095.
- **High Engineering:** Especifica el alto valor de ingeniería cuando la entrada es 4095.
- **Low Engineering:** Especifica el alto bajo de ingeniería cuando la entrada es 0.
- **Engineering** (BCD): Especifica la memoria V en donde será colocado el valor a escala BCD.

Parámetro	Rango del DL06
Raw (0-4095 BCD) . . . . . V,P	Vea el mapa de memoria V del DL06 - Data Words
High Engineering . . . . . K	K0-9999
Low Engineering . . . . . K	K0-9999
Engineering (BCD) . . . . . V,P	Vea el mapa de memoria V del DL06 - Data Words

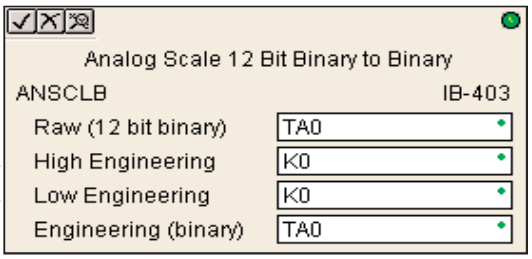
Ejemplo de ANSCL

En el ejemplo siguiente, se utiliza la instrucción ANSCL para colocar a escala un valor crudo (BCD 0-4095) que esté en V2000. El rango de la escala de ingeniería se define como 0-100 (valor bajo de ingeniería - alto valor de ingeniería). El valor a escala será colocado en V2100 en formato BCD.

Escala de un valor análogo de 12 bits binario a binario (ANSCLB) (IB-403)

DS5	Usado
HPP	N/A

Esta instrucción escala un valor análogo binario de 12 bits (0-4095 decimal) en unidades de ingeniería binarias. Usted especifica el valor de la unidad de ingeniería más alto (cuando el valor sin escala es 4095), y el valor de ingeniería más bajo (cuando el valor sin escala es 0), y la dirección de memoria V de salida que usted desea poner el valor de unidad que dirige escalado Las unidades de ingeniería se generan como binarias y pueden estar en el el rango completo de 0 a 9999 (Vea la instrucciónANSCL - si sus unidades sin escala están en formato BCD).



Observe que esta instrucción IBox trabaja solamente con valores unipolares sin escala positivos. No trabaja con valores bipolares ni con valores crudos de magnitud más signo.

Parámetros ANSCLB

- **Raw** (12 bit binary): Especifica la localización de la memoria V donde está el valor unipolar sin escala de rango (12 bit binario = 0-4095 decimal)
- **High Engineering**: Especifica el alto valor de ingeniería cuando la entrada es 4095.
- **Low Engineering**: Especifica el alto bajo de ingeniería cuando la entrada es 0.
- **Engineering** (binary): Especifica la memoria V en donde será colocado el valor a escala binario o decimal.

Parámetro	Rango del DL06
Raw (12 bit binary) . . . . . V,P	Vea el mapa de memoria V del DL06 - Data Words
High Engineering . . . . . K	K0-65535
Low Engineering . . . . . K	K0-65535
Engineering (binary) . . . . . V,P	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ANSCLB

En el ejemplo siguiente, la instrucción ANSCLB es usada para colocar a escala un valor crudo (0-4095 binario) que esté en V2000. El rango del escalamiento de ingeniería se define como 0-1000 (valor bajo de ingeniería - alto valor de ingeniería). El valor escalado será colocado en V2100 en formato binario.

Filtro - BCD (FILTER) (IB-422)

DS5	Usado
HPP	N/A

La instrucción FILTER realizará un filtro de primer orden en los datos en bruto sobre un intervalo definido de tiempo.La ecuación es:

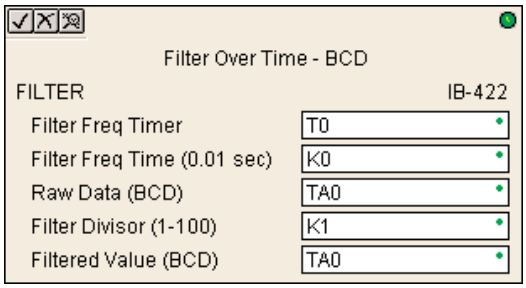
Nuevo valor = Valor antiguo+ [(Valor en bruto- Valor antiguo) / FDC]
donde,

Nuevo valor: Nuevo valor filtrado

Valor antiguo: Valor filtrado antiguo

FDC: Constantee divisor del filtro

Valor en bruto: Datos en bruto



La Constantee divisor del filtro FDC es un número entero en el rango K1 a K100, tal que crea un amortiguamiento sobre el valor en bruto y si es igual a K1 entonces entonces no sería hecho ningun filtrado.

La frecuencia en la cual se realiza el cálculo se especifica por tiempo en centésimos de un segundo (0,01 segundo) como el parámetro del constantee tiempo del filtro. Observe que hay una instrucción de temporizador embutida en el IBox y no debe ser usado en cualquier otro lugar en su programa. El control del renglón determina si el cálculo será ejecutado. Si es falso, el valor del filtro no es actualizado. En el primer barrido deonde pasa de modo program al modo RUN, el valor del filtro se inicializa a 0 para dar al cálculo un punto de partida consistente.

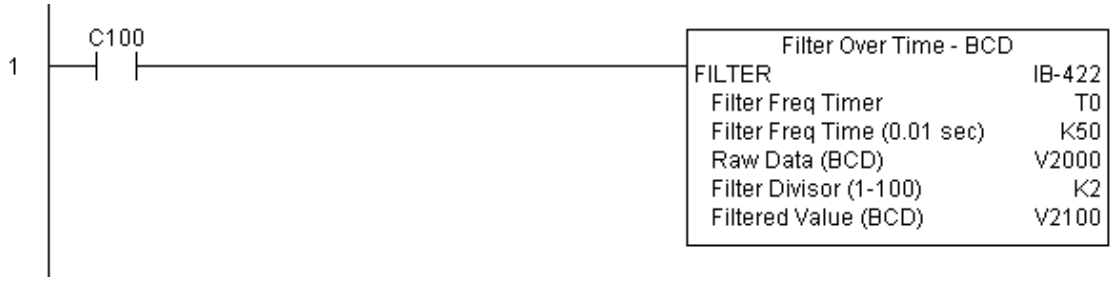
Parámetros de la instrucción FILTER

- **Filter Frequency Timer**: Especifica el número del temporizador (T) que es usado por la instrucción Filter
- **Filter Frequency Time** (0.01sec): Especifica la frecuencia en la cual se realiza el cálculo
- **Raw Data (BCD)**: Especifica la localización de memoria V del valor sin filtro en bruto BCD
- **Filter Divisor** (1-100): Esta contante es usada para controlar el efecto de filtrado. Un valor más grande aumentará el efecto que alisamiento del filtro.Un valor de 1 resulta sin filtrado.
- **Filtered Value** (BCD): Especifica la localización de memoria C en donde será colocado el valor filtrado en BCD

Parámetro	Rango del DL06
Filter Frequency Timer . . . . . T	T0-377
Filter Frequency Time (0.01 sec) . . . . . K	K0-9999
Raw Data (BCD) V	Vea el mapa de memoria V del DL06 - Data Words
Filter Divisor (1-100) . . . . . K	K1-100
Filtered Value (BCD) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de FILTER

En el ejemplo siguiente, es usada la instrucción FILTER para filtrar un valor en BCD que esté en V2000. El temporizador (T0) se coloca a 0.5 s, la frecuencia en la cual el cálculo del filtro será realizada. La constantee del filtro se coloca en 2. Un valor más grande aumentará el efecto de alisamiento del filtro. Un valor de 1 resulta sin filtrado. El valor filtrado será colocado en V2100.



Filtro Binario (FILTERB) (IB-402)

DS5	Usado
HPP	N/A

La instruccion filtro binario (decimal) realizará un filtro de primer orden en los datos en bruto sobre un intervalo definido de tiempo. La ecuación es

Nuevo valor = Valor antiguo+ [(Valor en bruto- Valor antiguo) / FDC]

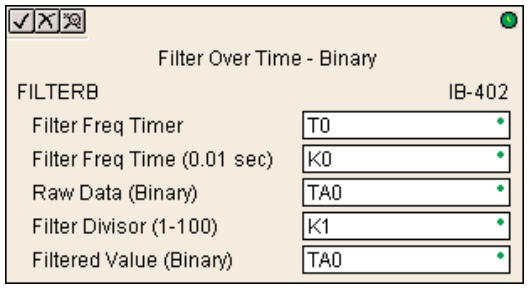
donde,

Nuevo valor: Nuevo valor filtrado

Valor antiguo: Valor filtrado antiguo

FDC: Constantee divisor del filtro

Valor en bruto Datos en bruto



La Constantee divisor del filtro FDC es un número entero en el rango K1 a K100, tal que crea un amortiguamiento sobre el valor en bruto y si es igual a K1 entonces no sería hecho ningun filtrado.

La frecuencia en la cual se realiza el cálculo se especifica por tiempo en centésimos de un segundo (0,01 segundo) como el parámetro del constante tiempo del filtro. Observe que hay una instrucción de temporizador embutida en el IBox y no debe ser usado en cualquier otro lugar en su programa. El control del renglón determina si el cálculo será ejecutado. Si es falso, el valor del filtro no es actualizado. En el primer barrido deonde pasa de modo program al modo RUN, el valor del filtro se inicializa a 0 para dar al cálculo un punto de partida consistente.

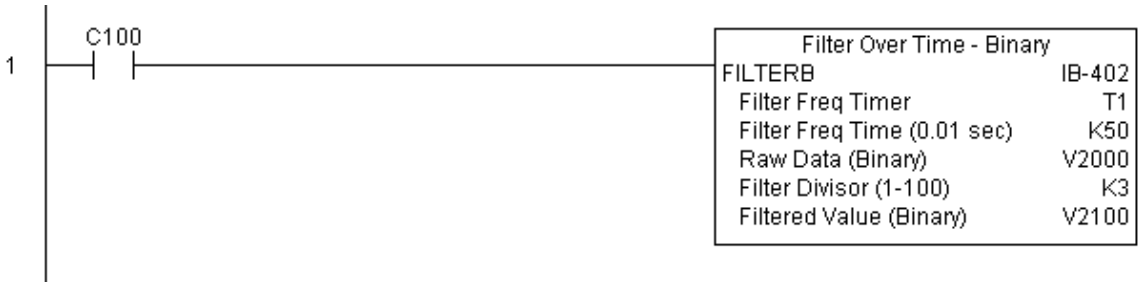
Parámetros de la instrucción FILTERB

- **Filter Frequency Timer**: Especifica el número del temporizador (T) que es usado por la instrucción FilterB
- **Filter Frequency Time** (0.01sec): Especifica la frecuencia en la cual se realiza el cálculo
- **Raw Data (Binario)**: Especifica la localización de memoria V del valor sin filtro en bruto binario (decimal)
- **Filter Divisor** (1-100): Esta contante es usada para controlar el efecto de filtrado. Un valor más grande aumentará el efecto que alisamiento del filtro.Un valor de 1 resulta sin filtrado.
- **Filtered Value** (Binario): Especifica la localización de memoria C en donde será colocado el valor filtrado en binario

Parámetro	Rango del DL06
Filter Frequency Timer . . . . . T	T0-377
Filter Frequency Time (0.01 sec) . . . . . K	K0-9999
Raw Data (Binary) V	Vea el mapa de memoria V del DL06 - Data Words
Filter Divisor (1-100) . . . . . K	K1-100
Filtered Value (Binary) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de FILTERB

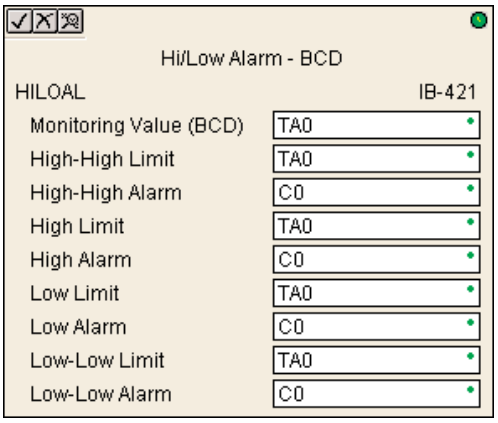
En el ejemplo siguiente, es usada la instrucción FILTERB para filtrar un valor en binario que está en V2000. El temporizador (T1) se coloca a 0,5 s, la frecuencia en la cual el cálculo del filtro será realizada. La constantee del filtro se coloca en 3.0. Un valor más grande aumentará el efecto de alisamiento del filtro. Un valor de 1 resulta sin filtrado. El valor filtrado será colocado en V2100.



Alarma Hi/Low - BCD (HILOAL) (IB-421)

DS5	Usado
HPP	N/A

Esta instrucción supervisa el valor BCD de una posición de memoria V y configura cuatro estados posibles de alarmas, Alta-Alta, Alta, Baja, y Baja-Baja siempre que la instrucción IBox sea verdadera. Usted define los niveles de alarmas como valores constantees BCD (K0-k9999) o como valor de memoria V en BCD.



Usted debe asegurarse de que los niveles de alarma sean válidos, esto es, HH > = H > L > = LL. Note que cuando la condición de alarma Alto-Alta o Baja-Baja es verdadera, la alarma alta y baja también estará activada, respectivamente. Esto significa que usted puede usar el mismo nivel y la misma alarma para las alarmas Alta-Alta y Alta en caso de que usted necesite solamente una "alarma Alta".También observe que las condiciones de límite son inclusivas. Es decir, si el límite bajo es K50, y el límite Baja-Baja es K10, y si el valor de supervisión iguala 10, después la alarma Baja y la alarma Baja-Baja quiere ambas estén ENCENDIDAS. Si no hay flujo de energía al IBox, entonces todos los bits de alarmas serán desactivados sin importar el valor del parámetro de supervisión.

Parámetros de HILOAL

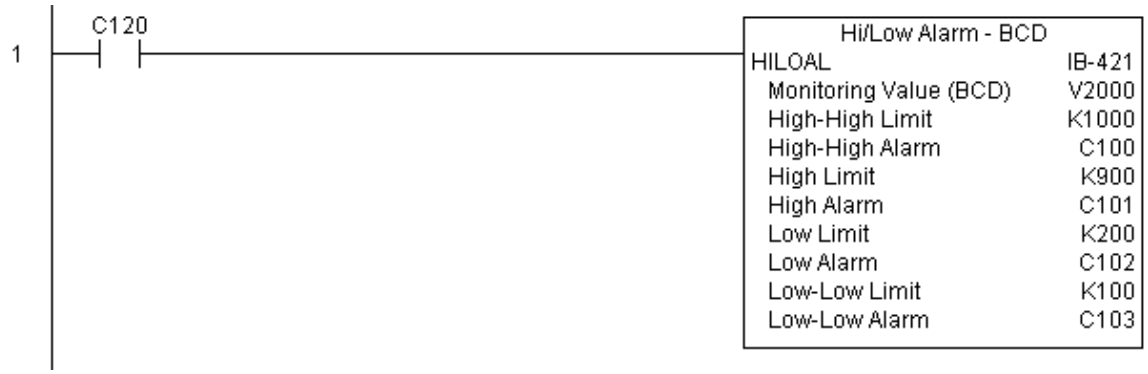
- **Monitoring Value** (BCD): Especifica la memoria V del valor BCD que se supervisará.
- **High-High Limit:** Constantee o memoria V que especifica el nivel de alarma Alta-Alta.
- **High-High Alarm**: Bit de alarma Alta Alta activada cuando se alcanza el nivel High-High limit
- **High Limit**: Memoria V o constantee que especifica el nivel de alarma Alta
- **High Alarm**: Bit de alarma Alta activada cuando se alcanza el nivel High limit
- **Low Limit**: Memoria V o constantee que especifica el nivel de alarma Baja
- **Low Alarm**: Bit de alarma Baja activada cuando se alcanza el nivel Low limit
- **Low-Low Limit**: Memoria V o constantee que especifica el nivel de alarma Low Low limit
- **Low-Low Alarm**: Bit de alarma Baja activada cuando se alcanza el nivel

Parámetro	Rango del DL06
Monitoring Value (BCD) V	Vea el mapa de memoria V del DL06 - Data Words
High-High Limit . . . . . V, K	K0-9999; or Vea el mapa de memoria V del DL06 - Data Words
High-High Alarm . . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06
High Limit . . . . . V, K	K0-9999; or Vea el mapa de memoria V del DL06 - Data Words
High Alarm . . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06
Low Limit . . . . . V, K	K0-9999; or Vea el mapa de memoria V del DL06 - Data Words
Low Alarm . . . . . X, Y, C, GX,GY,B	Vea el mapa de memoria DL06
Low-Low Limit . . . . . V, K	K0-9999; or Vea el mapa de memoria V del DL06 - Data Words
Low-Low Alarm . . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06

Ejemplo de HILOAL

En el ejemplo siguiente, la instrucción de HILOAL es usada para supervisar un valor BCD que está en V2000. Si el valor en V2000 exceede el valor de K900, se activará C101. Si el valor continúa aumentando hasta el nivel High-high, se activrá el bit C100. Ambos bits estarían encendido en este caso. Los niveles y alarmaa alta y alta-alta se pueden definir al mismo valor si se desea usar una alarma "Alta".

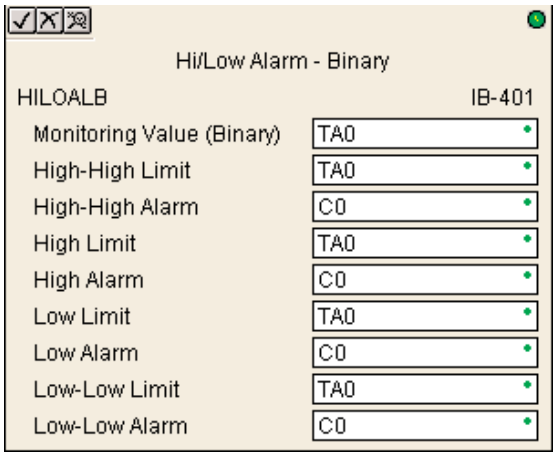
Si el valor en V2000 es igual o cae debajo del nivel K200, se desactivará C102. Si el valor continúa disminuyendo debajo del límite Bajo-Bajo K100, se desactivará C103 .Ambos bits estarían encendido en este caso. Las alarmas baja y baja-baja se pueden definir al mismo valor si se desa usar una alarma "Baja".



Alarm Hi/Low- Binaria (HILOALB) (IB-401)

DS5	Usado
HPP	N/A

Esta instrucción supervisa el valor binario de una posición de memoria V y configura cuatro estados posibles de alarmas, Alta-Alta, Alta, Baja, y Baja-Baja siempre que la instrucción IBox sea verdadera. Usted define los niveles de alarmas como valores constantees binarios (K0-K65535) o como valor de memoria V en binario.



Usted debe asegurarse de que los niveles de alarma sean válidos, esto es, HH > = H > L > = LL. Note que cuando la condición de alarma Alto-Alta o Baja-Baja es verdadera, la alarma alta y baja también estará activada, respectivamente. Esto significa que usted puede usar el mismo nivel y la misma alarma para las alarmas Alta-Alta y Alta en caso de que usted necesite solamente una "alarma Alta".También observe que las condiciones de límite son inclusivas. Es decir, si el límite bajo es K50, y el límite Baja-Baja es K10, y si el valor de supervisión iguala 10, después la alarma Baja y la alarma Baja-Baja quiere ambas estén ENCENDIDAS. Si no hay flujo de energía al IBox, entonces todos los bits de alarmas serán desactivados sin importar el valor del parámetro de supervisión.

Parámetros de HILOALB

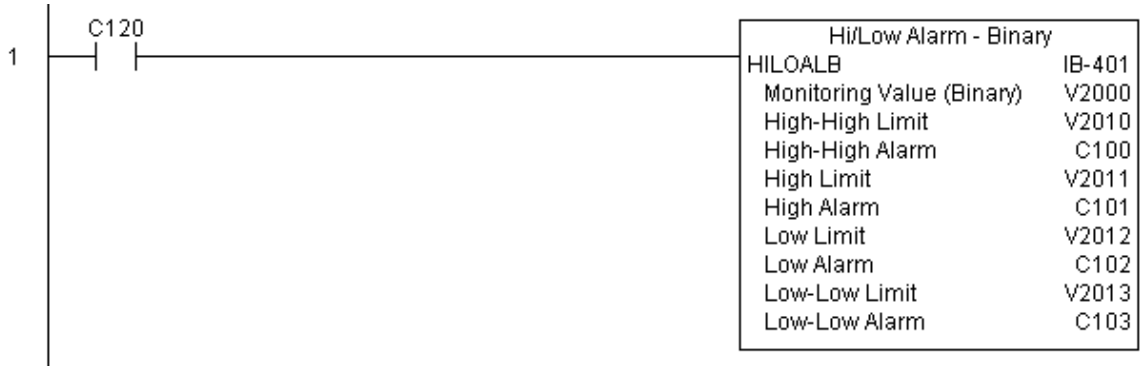
- **Monitoring Value** (Binario): Especifica la memoria V del valor binario que se supervisará.
- **High-High Limit:** Constantee o memoria V que especifica el nivel de alarma Alta-Alta.
- **High-High Alarm**: Bit de alarma Alta Alta activada cuando se alcanza el nivel High-High limit
- **High Limit**: Memoria V o constantee que especifica el nivel de alarma Alta
- **High Alarm**: Bit de alarma Alta activada cuando se alcanza el nivel High limit
- **Low Limit**: Memoria V o constantee que especifica el nivel de alarma Baja
- **Low Alarm**: Bit de alarma Baja activada cuando se alcanza el nivel Low limit
- **Low-Low Limit**: Memoria V o constantee que especifica el nivel de alarma Low Low limit
- **Low-Low Alarm**: Bit de alarma Baja activada cuando se alcanza el nivel

Parámetro	Rango del DL06
Monitoring Value (Binary) V	Vea el mapa de memoria V del DL06 - Data Words
High-High Limit . . . . . V, K	K0-65535; o Vea el mapa de memoria V del DL06 - Data Words
High-High Alarm . . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06
High Limit . . . . . V, K	K0-65535; o Vea el mapa de memoria V del DL06 - Data Words
High Alarm. . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06
Low Limit . . . . . V, K	K0-65535; o Vea el mapa de memoria V del DL06 - Data Words
Low Alarm . . . . . X, Y, C, GX,GY,B	Vea el mapa de memoria DL06
Low-Low Limit . . . . . V, K	K0-65535; o Vea el mapa de memoria V del DL06 - Data Words
Low-Low Alarm . . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06

Ejemplo de HILOALB

En el ejemplo siguiente, la instrucción HILOALB es usada para supervisar un valor binario que esté en V2000. Si el valor en V2000 es igual o mayor que el límite alto del valor binario en V2011, el bit C101 se activa. Si el valor continúa aumentando hasta ser igual o ser mas grande que el valor límite Alto-Alto en V2010, se activará el bit C100. Ambos bits estarían encendidos en este caso. Los límites y las alarmas Alta y Alta-Alta se pueden configurar al mismo valor o memoria V si se desea usar un límite o alarma "Alta".

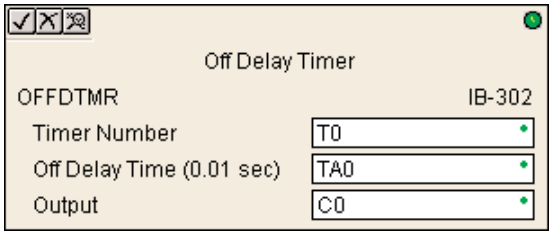
Si el valor en V2000 satisface o baja debajo del límite bajo del valor binario en V2012, se activará el bit C102. Si el valor continúa disminuyendo debajo del límite Bajo-Bajo en V2013, se activará el bit C103. Ambos bits estarían encendidos en este caso. Los límites y las alarmas Baja y Baja-Baja se pueden definir con la misma memoriaV o el mismo valor si se desea usar un límite o alarma Baja.



Temporizador Off Delay (OFFDTMR) (IB-302)

DS5	Usado
HPP	N/A

El temporizador Off delay retrasa "el apagado" del parámetro de salida (Output) especificado en la instrucción (en centésimo de segundo) basado en el flujo de energía en el IBox. Una vez que el IBox reciba energía, el bit de salida se encenderá inmediatamente. Cuando el flujo de energía al IBox hace falso, la salida segurá ENCENDIDA por la cantidad de tiempo especificada (en centésimo de segundo). Una vez que el tiempo ha expirado, la salida se apagará. Si el flujo de energía al IBox se hace verdadero ANTES DE QUE el tiempo de retardo se haya cumplido, el temporizador se rearma y la salida seguirá encendida - así que usted no debe continuamente tener NINGÚN flujo de energía al IBox POR LO MENOS el tiempo de retardo especificado antes de que la salida se apague.



Este IBox utiliza un temporizador (TMRF), que no debe ser usado en cualquier otro lugar en el programa ladder.

Parámetros de OFFDTMR

- **Timer Number**: Especifica el número del temporizador (TMRF) que es usado por la instrucción OFFDTMR
- **Off Delay Time** (0,01sec): Especifica cuánto tiempo la salida seguirá encendida si el flujo de energía al Ibox se torna falso
- **Output**: Especifica la salida que será apagada con un retardo de tiempo.

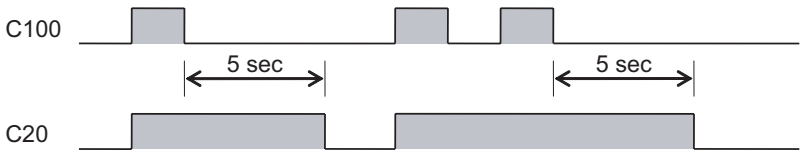
Parámetro	Rango del DL06
Timer Number . . . . . T	T0-377
Off Delay Time . . . . . K,V	K0-9999; Vea el mapa de memoria V del DL06 - Data Words
Output . . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06

Ejemplo de OFFDTMR

En el ejemplo siguiente, se usa la instrucción OFFDTMR para retrasar la salida C20. El temporizador 2 (t2) define el retardo en 5 segundos.

Cuando se cierra el contacto C100, C20 se activa y permanecerá encendido mientras C100 está encendido. Cuando se abre el contacto C100, C20 permanecerá activado por el tiempo especificado (5s), y luego se apaga.

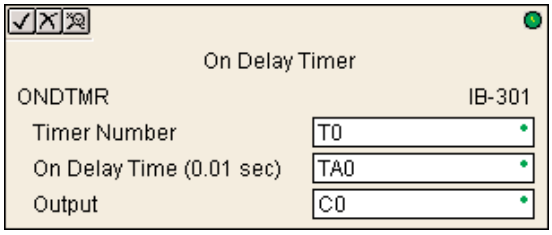
Ejemplo de diagrama de tiempos



Temporizador On Delay (ONDTMR) (IB-301)

DS5	Usado
HPP	N/A

El temporizador On delay retrasa el tiempo en que el parámetro de salida se activa por la cantidad de tiempo especificada (en centésimo de segundo) basada en el flujo de energía en el IBox. Una vez que el IBox pierde la energía, la salida se desactiva inmediatamente.Si el flujo de energía se apaga ANTES DE QUE transcurra el tiempo de retardo, entonces el contador de tiempo SE REARMA y la salida se apaga, así que usted debe tener flujo de energía continuo al IBox por lo menos el tiem[po de retardo especificado antes de que la salida se active.



Este IBox utiliza un temporizador (TMRF), que no puede ser usado en cualquier otro lugar en el programa.

Parámetros de ONDTMR

- **Timer Number**: Especifica el número del temporizador (TMRF) que es usado por la instrucción ONDTMR
- **On Delay Time** (0,01sec): Especifica cuánto tiempo la salida se encenderá si el flujo de energía al Ibox es verdadero.
- **Output**: Especifica la salida que será encendida con un retardo de tiempo.

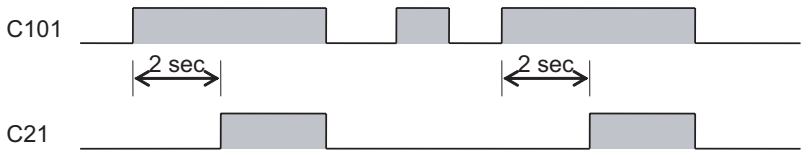
Parámetro	Rango del DL06
Timer Number . . . . . T	T0-377
On Delay Time . . . . . K,V	K0-9999; Vea el mapa de memoria V del DL06 - Data Words
Output . . . . . X, Y, C, GX,GY, B	Vea el mapa de memoria DL06

Ejemplo de ONDTMR

En el ejemplo siguiente, la instrucción ONDTMR es usada para retardar el "encendido" de la salida C21. El temporizador 1 (T1) define como de 2 segundos el período de "atraso".

Cuando se cierrael contacto C101, se cierra el contacto C21 con un atraso de 2 segundos. Cuando se abre el contacto C101, el contacto C21 se abre inmediatamente.

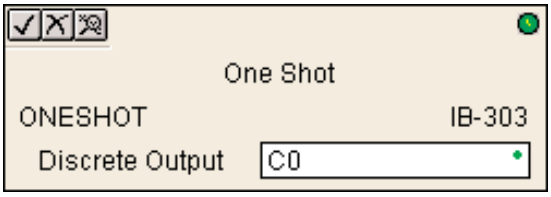
Ejemplo de diagrama de tiempos



One Shot (ONESHOT) (IB-303)

DS5	Usado
HPP	N/A

La instrución One Shot encenderá el bit de salida definido en el parámetro durante un barrido en la transición desde apagado a encendido del flujo de energía en el IBox. Este IBox es simplemente un nombre diferente para la bobina PD (diferencial positivo).



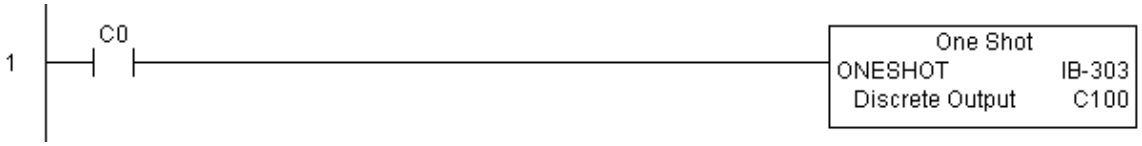
Parámetros de ONESHOT

- **Discrete Output**: Especifica la salida que estará encendida por un barrido

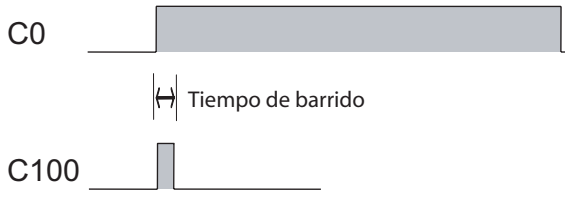
Parámetro	Rango del DL06
Discrete Output . . . . . X, Y, C	Vea el mapa de memoria DL06

Ejemplo de ONESHOT

En el ejemplo siguiente, la instrucción ONESHOT es usada para activar C100 por un barrido del PLC cuando el contacto C0 haga la transición desde abierto para cerrado. La lógica de entrada debe producir una transición desde abierto para cerrado para ejecutar la instrucción One shot.



Ejemplo de diagrama de tiempos



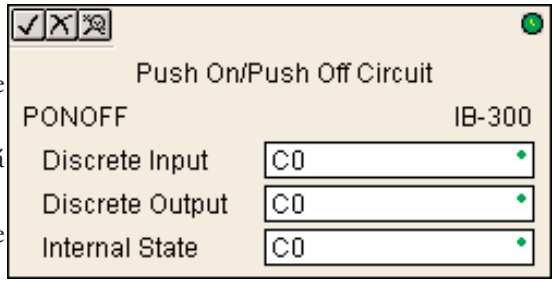
Circuito Push On / Push Off (PONOFF) (IB-300)

DS5	Usado
HPP	N/A

Esta instrucción conmuta un estado de la salida siempre que el flujo de la energía de la entrada haga una transición de apagado a encendido. Requiere un parámetro adicional de bit para trabajar con la información del estado. Este bit adicional no debe ser usado en cualquier otro lugar en el programa. Esto también se conoce como "circuito de flip-flop".

Parámetros de PONOFF

- **Discrete Input:** Especifica la entrada que conmutará la salida especificada
- **Discrete Output:** Especifica la salida que será "conmutada."
- **Internal State:** Especifica un bit de trabajo que será usado por la instrucción



Parámetro	Rango del DL06
Discrete Input . . . . X,Y,C,S,T,CT,GX,GY,SP,B,PB	Vea el mapa de memoria DL06
Discrete Output . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Internal State . . . . . X, Y, C	Vea el mapa de memoria DL06

Ejemplo de PONOFF

En el ejemplo siguiente, la instrucción PONOFF es usada para controlar los estados de la salida C20 con una sola entrada C10. Cuando el contacto C10 se cierra una vez, el bit C20 se activa. Cuando el contacto C10 se cierra otra vez, el bit C20 se apaga. El bit C100 es un usado internamente por la instrucción.

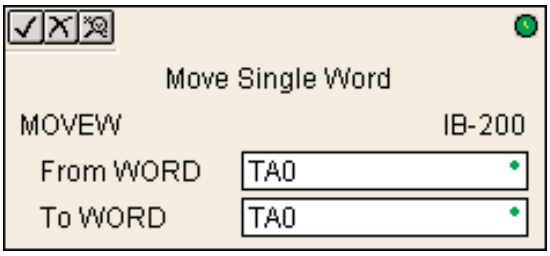
Mover una palabra (MOVEW) (IB-200)

DS5	Usado
HPP	N/A

Esta instrucción copia el contenido de una palabra a otra posición de memoria directamente o indirectamente con un puntero, ya sea como constantee HEXADECIMAL, desde una posición de memoria, o indirectamente a través de un puntero.

Parámetros de MOVEW

- **From WORD**: Especifica la palabra cuyo contenido será copiado a otra localización
- **To WORD**: Especifica la localización adonde será copiado el contenido de "From WORD"



Parámetro	Rango del DL06
From WORD . . . . . V,P,K	K0-FFFF; Vea el mapa de memoria V del DL06 - Data Words
To WORD. . . . . V,P	Vea el mapa de memoria V del DL06 - Data Words

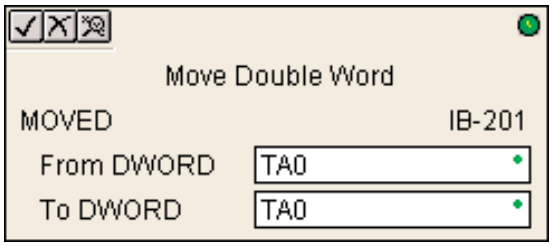
Ejemplo de MOVEW

En el ejemplo siguiente, la instrucción MOVEW es usada para copiar los 16 bits de datos desde V2000 a V3000 cuando C100 se cierra.

Mover una palabra doble (MOVED) (IB-201)

DS5	Usado
HPP	N/A

Esta instrucción copia el contenido de una palabra doble a a dos posiciones de memoria consecutivas directamente o indirectamente con un puntero, ya sea como constantee de palabra doble HEXADECIMAL, desde una posición de memoria doble , o indirectamente a través de un puntero.



Parámetros de MOVED

- **From WORD**: Especifica la palabra doble cuyo contenido será copiado a otra localización
- **To WORD**: Especifica la localización adonde será copiado el contenido de "From WORD"

Parámetro	Rango del DL06
From DWORD . . . . . V,P,K	K0-FFFFFFFF; Vea el mapa de memoria V del DL06 - Data Words
To DWORD . . . . . V,P	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de MOVED

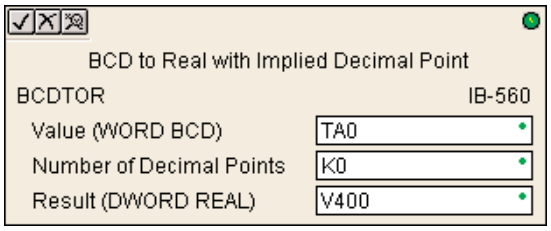
En el ejemplo siguiente, la instrucción MOVED instrucción es usada para copiar los 32 bits de datos desde V2000 y V2001 a V3000 y V3001 cuando C100 se cierra.

BCD a Real (BCDTOR) (IB-560)

DS5	Usado
HPP	N/A

Esta instrucción convierte un valor dado en una palabra BCD de 4 dígitos a un número Real, con una coma definida por la cantidad de decimales(K0-K4).

Por ejemplo, BCDTOR K1234 con un número implicado de coma igual a K1, resultaría R123.4



Parámetros de BCDTOR

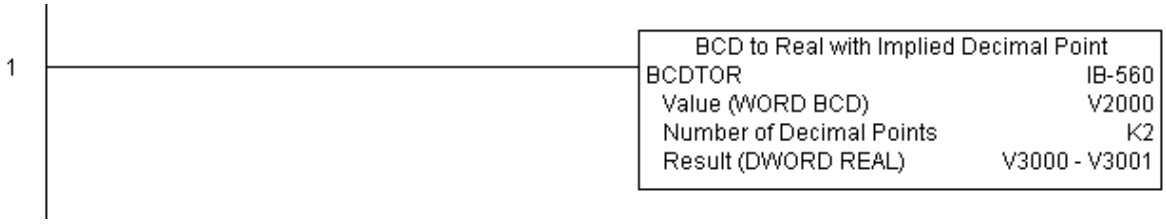
- **Value (WORD BCD)**: Especifica la palabra o la constantee que será convertida a un número real
- **Number of Decimal Points:** Especifica la cantidad de decimales en el resultado real
- **Result (DWORD REAL)**: Especifica la localización en donde será colocado el número real

Parámetro	Rango del DL06
Value (WORD BCD) . . . . . V,P,K	K0-9999; Vea el mapa de memoria V del DL06 - Data Words
Number of Decimal Points . . . . . K	K0-4
Result (DWORD REAL) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de BCDTOR

En el ejemplo siguiente, la instrucción BCDTOR es usada para convertir los datos de 16 bits en V2000 desde un formato de datos de 4 dígitos BCD a un formato de datos real de 32bits (de coma flotante) y almacenado en V3000 y V3001.

K2 en la cantidad de decimales indica que tendrá dos dígitos a la derecha de la coma.

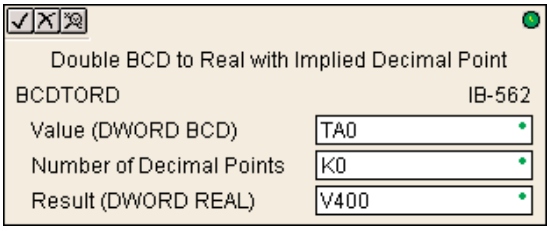


BCD doble a Real (BCDTORD) (IB-562)

DS5	Usado
HPP	N/A

Esta instrucción convierte un valor dado en una palabra doble BCD de 8 dígitos a un número Real, con una coma definida por la cantidad de decimales(K0-K8).

Por ejemplo, BCDTOR K12345678 con un número implicado de coma igual a K5, resultaría R123.45678



Parámetros de BCDTOR

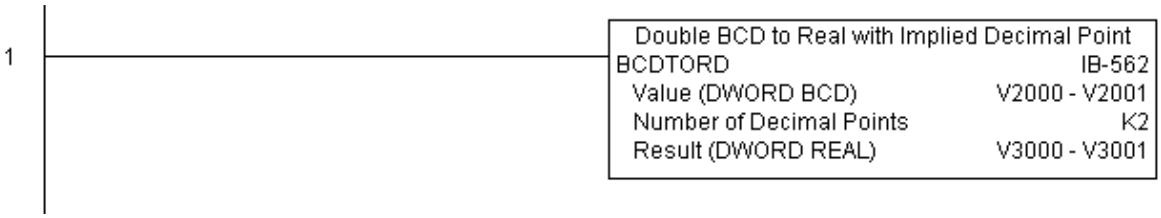
- **Value (WORD BCD)**: Especifica la palabra doble o la constantee que será convertida a un número real
- **Number of Decimal Points:** Especifica la cantidad de decimales en el resultado real
- **Result (DWORD REAL)**: Especifica la localización en donde será colocado el número real

Parámetro	Rango del DL06
Value (DWORD BCD) . . . . . V,P,K	K0-99999999; Vea el mapa de memoria V del DL06 - Data Words
Number of Decimal Points . . . . . K	K0-8
Result (DWORD REAL) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de BCDTORD

En el ejemplo siguiente, la instrucción BCDTORD es usada para convertir los datos de 32 bits en V2000 desde un formato de datos de 8 dígitos BCD a un formato de datos real de 32bits (de coma flotante) y almacenado en V3000 y V3001.

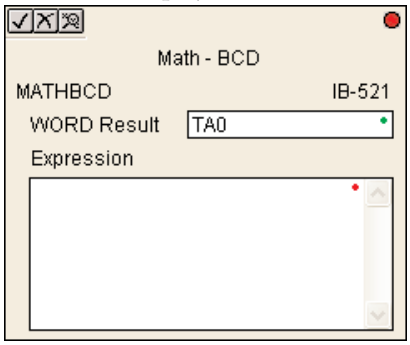
K2 en la cantidad de decimales indica que tendrá dos dígitos a la derecha de la coma.



Math - BCD (MATHBCD) (IB-521)

DS5	Usado
HPP	N/A

La instrucción Math-BCD le permite crear expresiones matemáticas complejas tal como usted haría en los programa Visual Basic, EXCEL o C++ para hacer cálculos complejos, con paréntesis de hasta 4 niveles de profundidad.



Además de + - * /, usted puede hacer el modulo (% o resto), AND de bits(&),OR (|), XOR (^), y algunas funciones conBCD - convertir a BCD (BCD), a convertir a binario (BIN), complemento del BCD (BCDCPL), convertir desde código Gray (GRAY), invertir bits (INV) y BCD/HEX a exhibición de siete segmentos (SEG).

Ejemplo: **((V2000 + V2001) / (V2003 - K100)) * GRAY(V3000 & K001F)**

Cada referencia de memoria V DEBE se usada como valor de formato BCD de una palabra. Los resultados intermedios pueden ir hasta valores de 32 bits, pero el resultado debe corresponder a una palabra BCD de 16 bits, para que el cálculo sea válido. Un ejemplo típico de ésto es hacer escala usando multiplicación y luego división, (V2000 * K1000)/K4095. El resultado de la multiplicación excederá probablemente 9999 pero estará dentro de 32 bits. La operación de dividir dividirá 4095 en el acumulador de 32 bits, generando un resultado que quepa siempre en 16 bits.

Usted puede referirse a valores binarios de memoria V usando la función de conversión BCD en una posición de memoriaV pero NO una expresión. Ésto es, BCD(V2000) es correcto y convertirá V2000 de binario a BCD, pero la operación BCD(V2000 + V3000) sumará V2000 como BCD a V3000 como BCD, y luego interpretará el resultado como binario y lo convierte a BCD - NO SERÁ CORRECTO.

También, el resultado final es un número BCD de 16 bits y de modo que usted podría hacer la instrucción BIN en la operación completa para almacenar el resultado como binario.

Parámetros de MATHBCD

- **WORD Result**: Especifica la localización en donde el resultado en BCD de la expresión matemática será colocado (el resultado debe caber en una localización de memoria V de 16 bits)
- **Expression:** Especifica la expresión matemática que se ejecutará y el resultado se almacena en la memoria especificada WORD Result. Cada localización de memoria V usada en la expresión debe estar en formato BCD.

Parámetro	Rango del DL06
WORD Result V	Vea el mapa de memoria V del DL06 - Data Words
Expression	Text

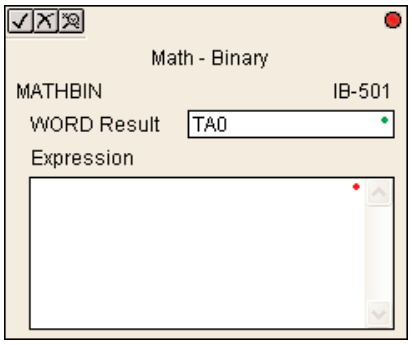
Ejemplo de MATHBCD

En el ejemplo siguiente, se usa la instrucción MATHBCD para calcular la expresión que multiplica el valor BCD en V1200 por 1000, después se divide por 4095 y carga el valor que resulta en V2000.

Math - Binaria (MATHBIN) (IB-501)

DS5	Usado
HPP	N/A

La instrucción Math-Binaria le permite crear expresiones matemáticas complejas como usted haría en los programa Visual Basic, EXCEL o C++ para hacer cálculos complejos, con paréntesis de hasta 4 niveles de profundidad. Además de + - * /, usted puede hacer el modulo (% o resto), AND de bits(&),OR (|), XOR (^), y algunas funciones conBCD - convertir a BCD (BCD), convertir a binario (BIN), decodificar bits(DECO), codificar bits (ENCO), invertir bits (INV) y HEX a exhibición de siete segmentos (SEG) y sumar bits (SUM).



Ejemplo: **((V2000 + V2001) / (V2003 - K100)) * SUM(V3000 & K001F)**

Cada referencia de memoria V DEBE se usada como valor de formato binario de una palabra. Los resultados intermedios pueden ir hasta valores de 32 bits, pero el resultado debe corresponder a una palabra binaria de 16 bits, para que el cálculo sea válido. Un ejemplo típico de ésto es hacer escala usando multiplicación y luego división, (V2000 * K1000)/K4095. El resultado de la multiplicación excederá probablemente 65535 pero estará dentro de 32 bits. La operación de dividir dividirá 4095 en el acumulador de 32 bits, generando un resultado que quepa siempre en 16 bits.

Usted puede referirse a valores BCD de memoria V usando la función de conversión BIN en una posición de memoriaV pero NO una expresión. Ésto es, BIN(V2000) es correcto y convertirá V2000 de BCD a binario, pero la operación BIN(V2000 + V3000) sumará V2000 como binario a V3000 como binario y luego interpretará el resultado como BCD y lo convierte a binario - NO SERÁ CORRECTO.

También, el resultado final es un número binario de 16 bits y de modo que usted podría aplicar la instrucción BCD en la operación completa para almacenar el resultado como BCD.

Parámetros de MATHBCD

- **WORD Result**: Especifica la localización en donde el resultado en binario de la expresión matemática será colocado (el resultado debe caber en una localización de memoria V de 16 bits)
- **Expression:** Especifica la expresión matemática que se ejecutará y el resultado se almacena en la memoria especificada WORD Result. Cada localización de memoria V usada en la expresión debe estar en formato binario.

Parámetro	Rango del DL06
WORD Result V	Vea el mapa de memoria V del DL06 - Data Words
Expression	Text

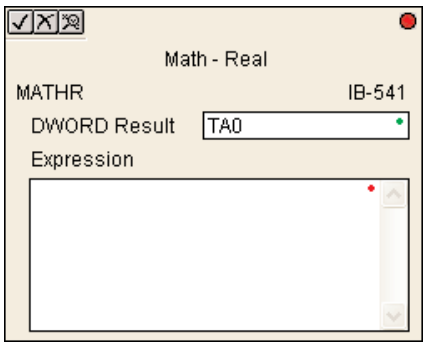
Ejemplo de MATHBIN

En el ejemplo siguiente, se usa la instrucción MATHBIN para calcular la expresión que multiplica el valor binario en V1200 por 1000, después se divide por 4095 y carga el valor que resulta en V2000.

Math - Real (MATHR) (IB-541)

DS5	Usado
HPP	N/A

Esta instrucción le permite crear expresiones matemáticas complejas como usted haría en los programa Visual Basic, EXCEL o C++ para hacer cálculos complejos, con paréntesis de hasta 4 niveles de profundidad. Además de + - * /, usted puede hacer una instrucción And (&) Or (|) Xor (^) y muchas funciones Reales - Arco Coseno (ACOSR), ArcoSeno (ASINR), Arco Tangente (ATANR), Coseno (COSR), Convertir Radianes a Grados (DEGR), Invertir Bits (INV), Convertir grrados a Radianes (RADR), HEX a Seven Segment Display (SEG), Seno (SINR), Raíz cuadrada (SQRTR), Tangente (TANR).



Example: ((V2000 + V2002) / (V2004 - R2.5)) * SINR(RADR(V3000 / R10.0))

Cada referencia de memoria V DEBE caber en un valor ajustado a formato REAL de palabra doble.

5

Parámetros de MATHR

- **DWORD Result**: Especifica la localización en donde será colocado el resultado Real de la expresión matemática (el resultado debe caber en una localización ajustada a formato real de palabra doble)
- **Expression**: Especifica la expresión matemática que se ejecutará y el resultado se almacena en la localización especificada DWORD Result. Cada localización de memoria V en la expresión debe estar en formato Real.

Parámetro	Rango del DL06
DWORD Result V	Vea el mapa de memoria V del DL06 - Data Words
Expression	Text

Ejemplo de MATHR

En el ejemplo siguiente, se usa la instrucción MATHR para calcular la expresión que multiplica el valor REAL (de coma flotante) en V1200 por 10,5 y después se divide por 2,7 y se coloca el valor resultante de 32 bits en V2000 y V2001.

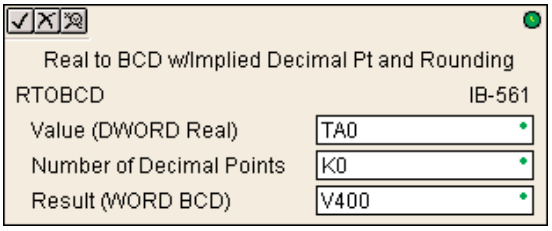
Real a BCD con redondeo (RTOBCD) (IB-561)

DS5	Usado
HPP	N/A

Esta instruccción convierte el valor absoluto de un número Real dado a un número en BCD de 4 dígitos, con la cantidad de decimales definidas en la instrucción (K0-K4) y además realiza el redondeo.

Por ejemplo, RTOBCD R56.74 con una cantidad de decimales igual a K1, resultaría BCD 567. Si el número implicado de comas fuera 0, entonces la función resultaría BCD 57 (note que redondeó para arriba).

Si el número Real es negativo, el resultado será igual al valor positivo absoluto.



5

Parámetros de RTOBCD

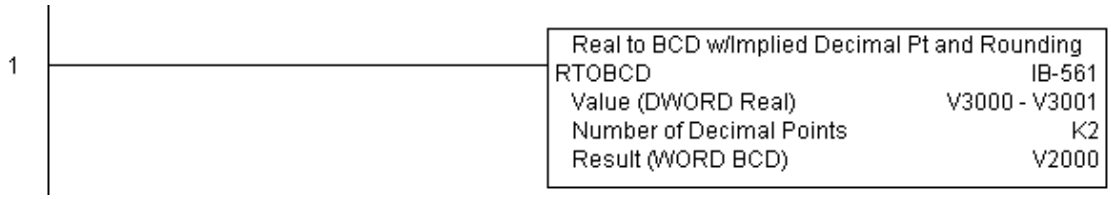
- **Value (DWORD Real):** Especifica la localización o el número Real que será convertido y redondeado a un número BCD con decimales implicados.
- **Number of Decimal Points:** Especifica el número de decimales implicados en el resultado Result WORD
- **Result (WORD BCD):** Especifica la localización en donde será almacenado el valor BCD que ha sido redondeado y transformado

Parámetro	Rango del DL06
Value (DWORD Real) . . . . . V,P,R	R ; Vea el mapa de memoria V del DL06 - Data Words
Number of Decimal Points . . . . . K	K0-4
Result (WORD BCD) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de RTOBCD

En el ejemplo siguiente, se usa la instrucción RTOBCD para convertir el formato de datos Real de 32 bits (de coma flotante) en V3000 y V3001 al formato de datos de 4 dígitos en BCD y almacenados en V2000.

K2 en el parámetro **Number of Decimal Points** (cantidad de decimales implicados) en los datos tendrá dos decimales implicados.



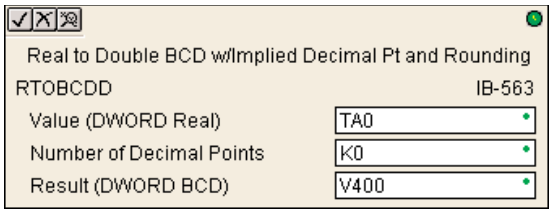
Real a BCD doble con redondeo (RTOBCDD) (IB-563)

Esta instruccción convierte el valor absoluto de un número Real dado a un número en BCD de 8 dígitos, con la cantidad de decimales definidas en la instrucción (K0-K8) y además realiza el redondeo.

DS5	Usado
HPP	N/A

Por ejemplo, RTOBCDD R38156.74 con una cantidad de decimales igual a K1, resultaría BCD 381567. Si el número implicado de comas fuera 0, entonces la función resultaría BCD 38157 (nota que redondeó para arriba).

Si el número Real es negativo, el resultado será igual al valor positivo absoluto.



5

Parámetros de RTOBCDD

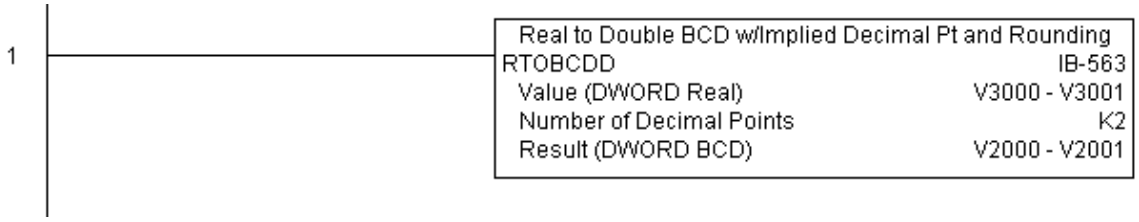
- **Value (DWORD Real):** Especifica la localización o el número Real que será convertido y redondeado a un número BCD con decimales implicados.
- **Number of Decimal Points:** Especifica el número de decimales implicados en el resultado Result WORD
- **Result (WORD BCD):** Especifica la localización en donde será almacenado el valor BCD que ha sido redondeado y transformado

Parámetro	Rango del DL06
Value (DWORD Real) . . . . . V,P,R	R ; Vea el mapa de memoria V del DL06 - Data Words
Number of Decimal Points . . . . . K	K0-8
Result (DWORD BCD) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de RTOBCDD

En el ejemplo siguiente, se usa la instrucción RTOBCDD para convertir el formato de datos Real de 32 bits (de coma flotante) en V3000 y V3001 al formato de datos de 8 dígitos en BCD y almacenados en V2000 y V2001.

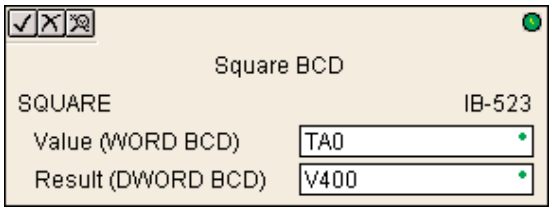
K2 en el Number of Decimal Points (cantidad de decimales implicados) en los datos tendrá dos decimales implicados.



BCD al cuadrado (SQUARE) (IB-523)

DS5	Usado
HPP	N/A

Esta instrucción eleva al cuadrado el número de 4 dígitos dado en formato BCD y lo escribe como un resultado de 8 dígitos en formato BCD (palabra doble).



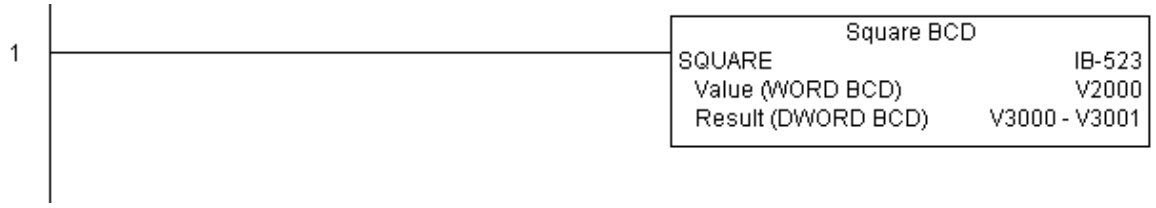
Parámetros de SQUARE

- **Value (WORD BCD)**: Especifica la palabra o la constantee en BCD que será elevada al cuadrado
- **Result (DWORD BCD)**: Especifica la localización en donde será almacenado el valor al cuadrado de DWORD BCD

Parámetro	Rango del DL06
Value (WORD BCD) . . . . . V,P,K	K0-9999 ; Vea el mapa de memoria V del DL06 - Data Words
Result (DWORD BCD) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de SQUARE

En el ejemplo siguiente, se usa la instrucción SQUARE para elevar al cuadrado el valor BCD de 4 dígitos en V2000 y para almacenar el resultado de 8 dígitos en V3000 y V3001



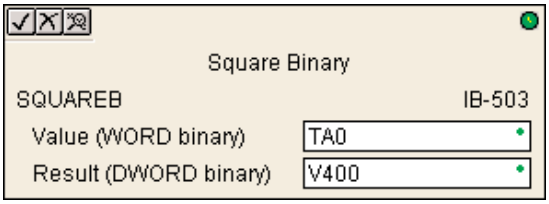
Binario al cuadrado (SQUAREB) (IB-503)

DS5	Usado
HPP	N/A

Esta instrucción eleva al cuadrado el número de 4 dígitos dado en formato binario y lo escribe como un resultado de 8 dígitos en formato binario.

Parámetros de SQUAREB

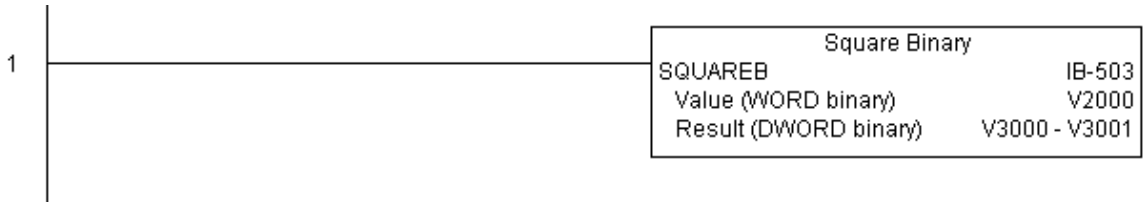
- **Value (WORD binary)**: Especifica la palabra o la constantee en FORMATO BINARIO que será elevada al cuadrado
- **Result (DWORD binary)**: Especifica la localización en donde será almacenado el valor al cuadrado de la palabra doble en formato binario.



Parámetro	Rango del DL06
Value (WORD Binary) . . . . . V,P,K	K0-65535; Vea el mapa de memoria V del DL06 - Data Words
Result (DWORD Binary) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de SQUAREB

En el ejemplo siguiente, se usa la instrucción SQUAREB para elevar al cuadrado el valor de formato binario de 4 dígitos en V2000 y para almacenar el resultado de 8 dígitos en V3000 y V3001



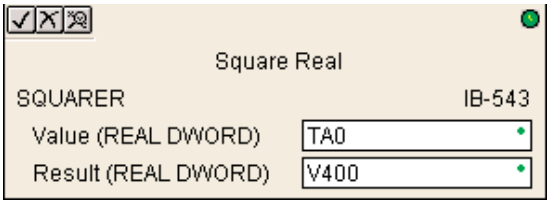
Real al cuadrado (SQUARER) (IB-543)

DS5	Usado
HPP	N/A

Esta instrucción eleva al cuadrado un número dado en formato Real y lo escribe como un resultado de 8 dígitos en formato Real.

Parámetros de SQUARER

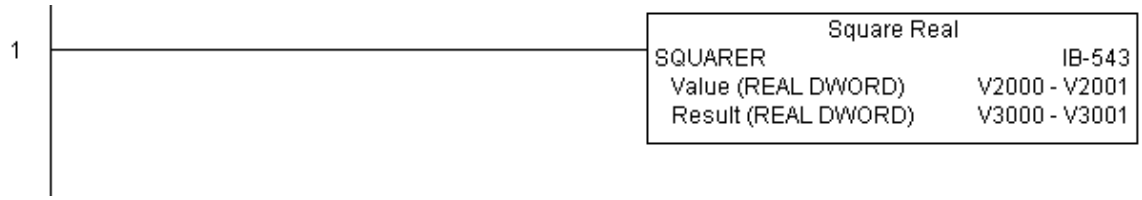
- **Value (REAL DWORD):** Especifica la palabra o la constantee en formato Real que será elevada al cuadrado
- **Result (REAL DWORD):** Especifica la localización en donde será almacenado el valor al cuadrado.



Parámetro	Rango del DL06
Value (REAL DWORD) . . . . . V,P,R	R ; Vea el mapa de memoria V del DL06 - Data Words
Result (REAL DWORD) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de SQUARER

En el ejemplo siguiente, se usa la instrucción SQUARER para elevar al cuadrado el valor de coma flotante o valor REAL en V2000 y V2001 y para almacenar el resultado REAL en V3000 y V3001.

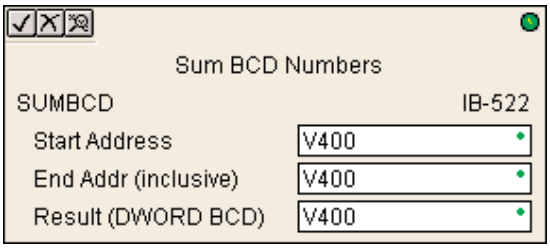


Suma de números BCD (SUMBCD) (IB-522)

DS5	Usado
HPP	N/A

Esta instruccción suma una lista de números de 4 dígitos consecutivos en formato BCD en un resultado de 8 dígitos BCD en una palabra doble.

Usted debe especificar las direcciones iniciales y finales del grupo de memoriaV (incluyendo estas direcciones). Cuando está activada, esta instrucción sumará todos los números en el grupo (de modo que usted puede desear usar un contacto diferencial positivo como condición de la instrucción).



SUMBCD podría ser usado como la primera parte para calcular un promedio.

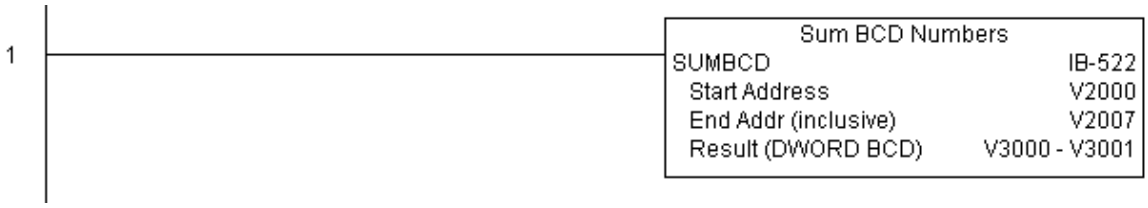
Parámetros de SUMBCD

- **Start Address:** Especifica la dirección inicial de un conjunto de valores consecutivos de localización de memoria V que se sumarán (BCD)
- **End Addr (inclusive):** Especifica la dirección final de un conjunto de valores consecutivos de localización de memoria V que se sumarán (BCD)
- **Result (DWORD BCD):** Especifica la localización de memoria V en donde será colocado el resultado de la suma del conjunto de valores consecutivos BCD

Parámetro	Rango del DL06
Start Address V	Vea el mapa de memoria V del DL06 - Data Words
End Address (inclusive) V	Vea el mapa de memoria V del DL06 - Data Words
Result (DWORD BCD) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de SUMBCD

En el ejemplo siguiente, se usa la instrucción de SUMBCD para sumar todos los valores BCD en las palabras V2000 hasta V2007 y para almacenar el resultado, que es un valor de 8 dígitos BCD, en V3000 y V3001.

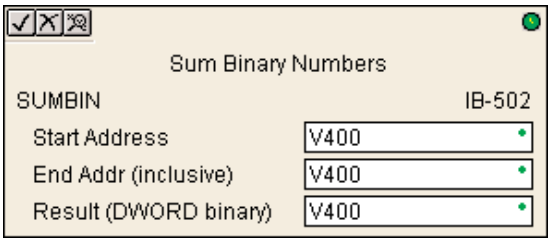


Suma de números Binarios (SUMBIN) (IB-502)

DS5	Usado
HPP	N/A

Esta instruccción suma una lista de números de 4 dígitos consecutivos en formato binario en un resultado de 8 dígitos binario, en una palabra doble.

Usted debe especificar las direcciones iniciales y finales del grupo de memoriaV (incluyendo estas direcciones). Cuando está activada, esta instrucción sumará todos los números en el grupo (de modo que usted puede desear usar un contacto diferencial positivo como condición de la instrucción).



SUMBIN podría ser usado como la primera parte para calcular un promedio.

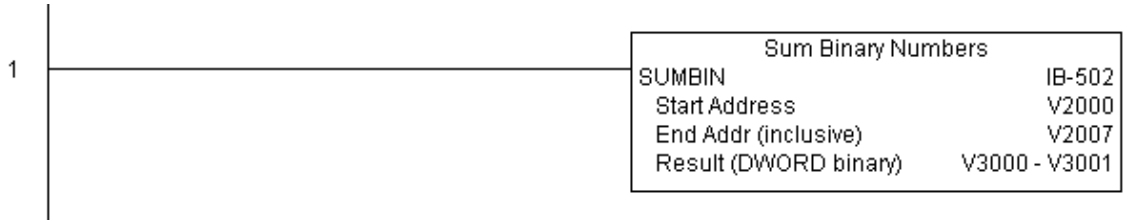
Parámetros de SUMBIN

- **Start Address:** Especifica la dirección inicial de un conjunto de valores consecutivos de localización de memoria V que se sumarán (Binarios)
- **End Addr (inclusive):** Especifica la dirección final de un conjunto de valores consecutivos de localización de memoria V que se sumarán (Binarios)
- **Result (DWORD Binary):** Especifica la localización de memoria V en donde será colocado el resultado de la suma del conjunto de valores consecutivos Binarios

Parámetro	Rango del DL06
Start Address V	Vea el mapa de memoria V del DL06 - Data Words
End Address (inclusive) V	Vea el mapa de memoria V del DL06 - Data Words
Result (DWORD Binary) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de SUMBIN

En el ejemplo siguiente, se usa la instrucción SUMBIN para sumar todos los valores binarios en las palabras V2000 hasta V2007 y para almacenar el resultado, que es un valor de 8 dígitos binarios, en V3000 y V3001.

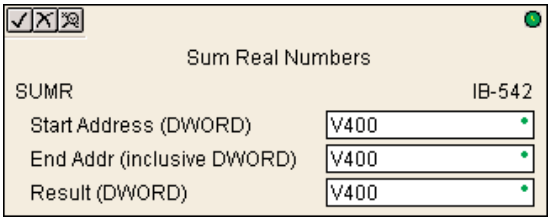


Suma de números Reales (SUMR) (IB-542)

DS5	Usado
HPP	N/A

Esta instruccción suma una lista de números consecutivos en formato REAL en un resultado en una palabra doble de formato real o coma flotante.

Usted debe especificar las direcciones iniciales y finales del grupo de memoriaV (incluyendo estas direcciones).



Recuerde que los números reales ocupan 2 palabras de memoria V cada uno, de modo que el número de valores reales sumados es igual a la mitad del número de las memorias.

Observe que la dirección final puede ser CUALQUIER palabra de la dirección final de 2 palabras, por ejemplo, si usted está sumabdo los 4 números reales almacenados en V2000 hasta V2007 (V2000, V2002, V2004, y V2006), usted puede especificar V2006 o V2007 como la dirección final y usted obtendrá el mismo resultado.

Cuando está activada, esta instrucción sumará todos los números en el grupo (de modo que usted puede desear usar un contacto diferencial positivo como condición de la instrucción).

SUMR cpodría ser usado como la primera parte para calcular un promedio.

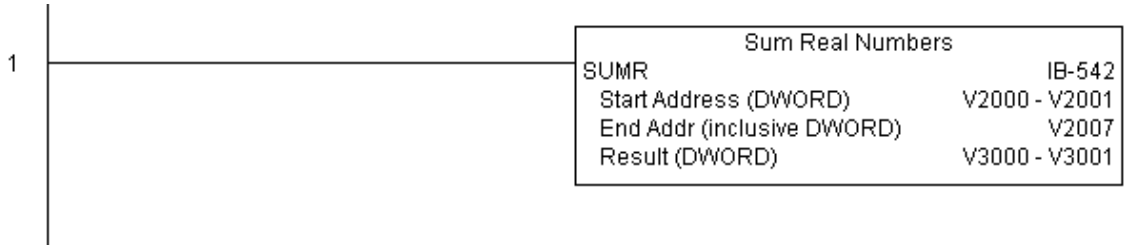
Parámetros de SUMR

- **Start Address(DWORD):** Especifica la dirección inicial de un conjunto de valores consecutivos de localización de memoria V que se sumarán (Real)
- **End Addr (inclusive DWORD):** Especifica la dirección final de un conjunto de valores consecutivos de localización de memoria V que se sumarán (Real)
- **Result (DWORD):** Especifica la localización de memoria V en donde será colocado el resultado de la suma del conjunto de valores consecutivos en formato de coma flotante.

Parámetro	Rango del DL06
Start Address (inclusive DWORD) V	Vea el mapa de memoria V del DL06 - Data Words
End Address (inclusive DWORD) V	Vea el mapa de memoria V del DL06 - Data Words
Result (DWORD) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de SUMR

En el ejemplo siguiente, se usa la instrucción SUMR para sumar todos los valores reales en las palabras V2000 hasta V2007 y para almacenar el resultado, que es un valor de de coma flotante, en V3000 y V3001.

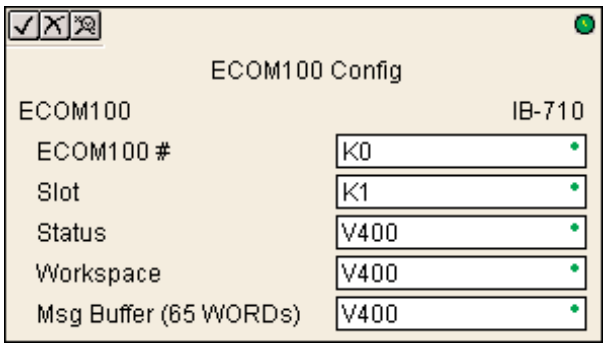


Configuración de ECOM100 (ECOM100) (IB-710)

DS5	Usado
HPP	N/A

La instrucción configuración ECOM100 define toda la información común para un módulo específico ECOM100 que sea utilizado por otro IBox ECOM100; por ejemplo, ECRX - leer la red con ECOM100, ECEMAIL - ECOM100 envíar e-mail, ECIPSUP - configuración del IP ECOM100, etc.

Usted DEBE tener el IBox de configuración ECOM100 en la parte superior de su programa ladder o de etapas con cualquier otra configuración IBox. El parámetro Message buffer (almacenador intermediario de mensajes) especifica la dirección inicial de un almacenador intermediario (Message buffer) de 65 Palabras. Ésto es, 101 direcciones octales (por ejemplo. V1400 hasta V1500).



Si usted tiene más de un ECOM100 en su PLC, usted debe tener una configuración IBox ECOM100 diferente para CADA módulo ECOM100 en su sistema que use una instrucción IBox ECOM .

Los parámetros de **espacio de trabajo** (**Workspace**) y de estado (**Status**) y el almacenador intermediario de mensaje (Message buffer) son registros internos, privados, usados por la instrucción configuración de ECOM100 y DEBEN SER ÚNICOS en esta instrucción y NO SE DEBEN utilizar en cualquier otro lugar en su programa.

Para que funcione LA MAYORÍA DE IBoxes ECOM100, usted debe accionar el DIP switch 7 ON en la placa de circuito ECOM100. Usted puede mantener el DIP switch 7 apagado si usted está usando SOLAMENTE IBoxes ECOM100 ECRX o ECWX.

Parámetros de ECOM100

- **ECOM100#**: Este es un número lógico de identificación asociado a este módulo específico ECOM100 en la ranura especificada. El resto de IBoxes que necesitan referirse a este módulo ECOM100 debe referirse a este número lógico
- **Slot**: Especifica la ranura opcional que ocupa el módulo
- **Status**: Especifica una localización de lmemoria V que será usada por la instrucción
- **Workspace**: Especifica una localización de memoria V que será usada por la instrucción
- **Msg Buffer**: Especifica la dirección inicial de un almacenador intermediario de 65 palabras que será usado por el módulo para configuración

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Slot . . . . . K	K1-4
Status V	Vea el mapa de memoria V del DL06 - Data Words
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Msg Buffer (65 words usado) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECOM100

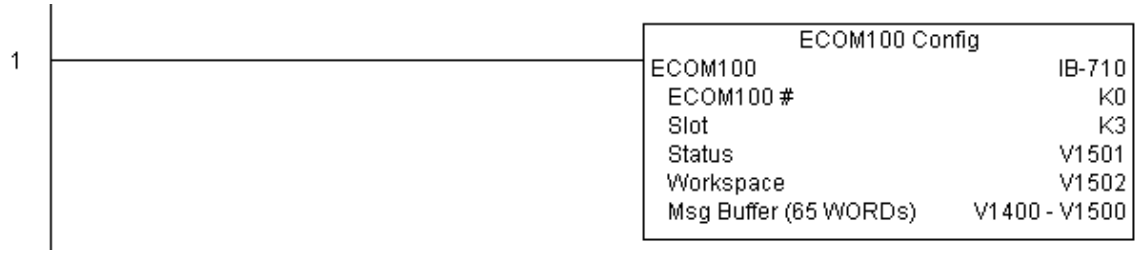
Esta instrucción coordina toda la interacción con otras instrucciones IBox relacionadas con ECOM100 (ECxxxx). Usted debe tener una instrucción de configuración IBox para cada módulo ECOM100 que esté instalado en su sistema. Estas instrucciones IBox de configuración de ECOM100 deben estar en la parte superior de su programa y se deben ejecutar en cada barrido.

Este IBox define que o módulo ECOM100# K0 está en la ranura 3. Cualquier instruccción IBox relacionada a ECOM100 que necesite referirse a este módulo específico (tal como ECEMAIL, ECRX…)debe tener la referencia K0 para el parámetro ECOM100 #.

El registro de estado (Status register) está diseñado para divulgar cualquier información de error o que ha sido completada la operación a otros IBoxes relacionados con ECOM100. Este registro de memoria V no debe ser usado en ningún otro lugar en el programa entero.

El registro del espacio de trabajo (Workspace register) es usado para mantener la información del estado sobre el ECOM100, junto con compartir adecuadamente y enclavarse con otros IBoxes realcionados con ECOM100 en el programa. Este registro de memoria V no debe ser usado en ningón otro lugar en el programa entero.

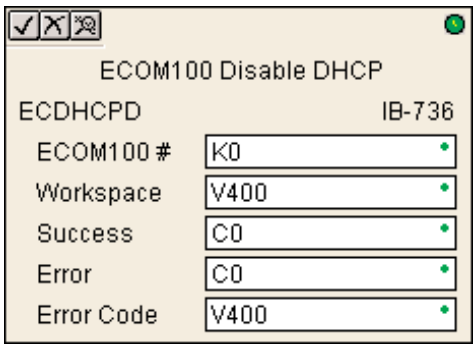
El almacenador intermediario de mensajes (Message Buffer) 65 palabras (130 bytes) es un grupo común de memoria que es usado por otros IBoxes relacionados con ECOM100 (tal como ECEMAIL). De esta manera, usted puede tener una cierta cantidad de IBoxes ECEMAIL, pero necesita solamente 1 almacenador intermediario común para generar y enviar cada email. Estos registro de memoria V no deben ser usados en ningón otro lugar en el programa entero.



Desabilitar DHCP en ECOM100 (ECDHCPD) (IB-736)

DS5	Usado
HPP	N/A

Esta instrucción configura el ECOM100 para utilizar las definiciones internas de TCP/IP en una transición de On para Off al control de este IBox. Para configurar las definiciones de TCP/IP del ECOM100 manualmente, puede usar la utilidad NetEdit3, o usted puede hacerla por el programa del PLC usando la instruccion Ibox de configuración de IP ECOM100 (ECIPSUP), o los IBoxes individuales: ECWRIP (Escribir dirección IP de ECOM), ECWRGWA (Escribibir la dirección de gateway) y ECWRSNM (Escribir Subnet Mask).



El parámetro del espacio de trabajo es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en su programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000).

La configuración se almacena en la memoria Flash-ROM en el ECOM100 y la ejecución de este IBox inhabilitará el módulo ECOM100 por lo menos un medio segundo hasta que escribe la memoria Flash-ROM. Por lo tanto, SE RECOMIENDA que usted ejecute solamente este IBox UNA VEZ, en el primer barrido. Puesto que requiere una transición de Off para On , use un SP0 NORMALMENTE CERRADO (NO se ejecuta en el primer barrido) para conducir el flujo de energía al IBox.

Para este ECOM100 IBox funcione, usted debe mover el DIP switch 7 en el circuito del módulo ECOM100.

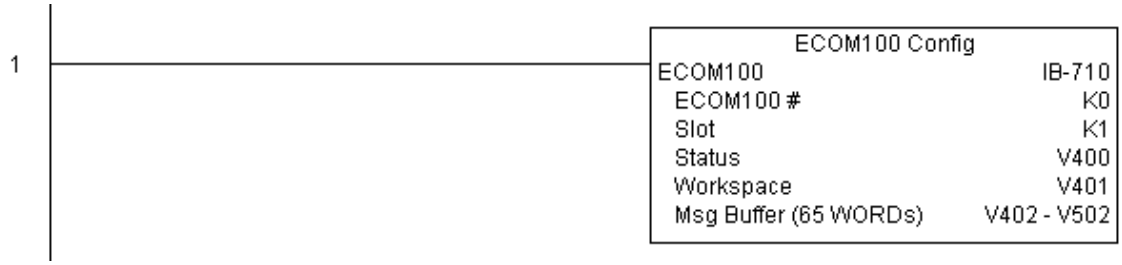
Parámetros de ECDHCPD

- **ECOM100#**: Éste es un número lógico asociado a este módulo específico en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que será usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la requisición se termine con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code:** Especifica la localización en donde será escrito el código de error

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECDHCPD

Renglón 1: La instrucción ECDHCPD es responsable por la coordinación y enclavamiento de todos los Iboxes relacionados con ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100 de número K0. El resto de los IBoxes ECxxxx se refieren a este módulo como K0. Si usted necesita mover el módulo en la base a una ranura diferente, solamente necesita cambiar la información en este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx usando este módulo específico ECOM100. V401 es usado para coordinación y enclavamiento de la lógica en todo los otros IBoxes ECxxxxusando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario(buffer) de 130 bytes disponible para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100



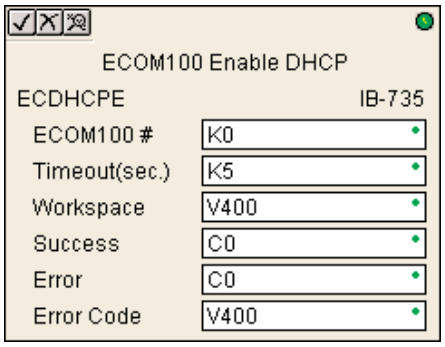
Renglón 2: En el segundo barrido, deshabilita DHCP en el ECOM100. DHCP es el mismo protocolo usado por PCs para usar un servidor de DHCP para asignar automáticamente la direción de IP (IP ADDRESS), la dirección de entrada y el subnet mask de los módulos ECOM100. Típicamente DHCP es deshabilitado asignando un IP ADDRESS HARD CODED con NetEdit3 o usando uno de lOS IBoxes de configuración de PI ECOM100, pero este IBox permite que usted deshabilite DHCP en el ECOM100 usando su programa ladder. El ECDHCPD es accionado en una transición de OFF para ON, no por una flujo de enrgía constantee (similar a una entrada de un contador). El comando de deshabilitar DHCP será enviado al ECOM100 siempre que el flujo de energía en el IBox vaya de OFF a ON. Si se completa esta acción con éxito, se activa el bit C100. Si hay una falla, se activa el bit C101. Si falla, usted puede observar V2000 para conocer el código de error específico.

Habilitar DHCP en ECOM100 (ECDHCPE) (IB-735)

DS5	Usado
HPP	N/A

Esta instrucción le dirá el ECOM100 que obtenga su configuración de TCP/IP desde un servidor de DHCP en una transición de OFG para ON en el flujo de energía del IBox.

El IBox será colocado en estado éxitoso una vez que el ECOM100 haya recibido las configuraciones de TCP/IP desde el servidor de DHCP. Puesto que es posible que el servidor de DHCP no sea asequible, se suministra un parámetro de atraso de tiempo (Timeout) de modo que el IBox pueda terminar, pero con un error (código de error = el decimal 1004).



Vea también el IBox 717 - configuración de IP del ECOM100 (ECIPSUP) - para configurar directamente TODOS LOS parámetros de TCP/IP en una sola instrucción - IP ADDRESS, subnet mask, y dirección de entrada.

The parámetro **Workspace** es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Uno de los parámetros de éxito(success) o de error (Error) se activará una vez que el comando se haya completado. Si hay un error, el parámetro del código de error divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000).

La configuración "Habilitar DHCP" se almacena en memoria Flash-ROM en el ECOM100 y la ejecución de este IBox deshabilitará el módulo ECOM100 por lo menos 0,5 segundo hasta que escribe la Flash-ROM. Por lo tanto, SE RECOMIENDA que se ejecute solamente este IBox UNA VEZ, en el primer barrido. Puesto que requiere una transición de OFF para ON para ejecutarse, use un SP0 NORMALMENTE CERRADO para controlar el flujo de energía al IBox.

Para que este IBox funcione, se debe colocar el DIP switch 7 a ON en el circuito del módulo ECOM100.

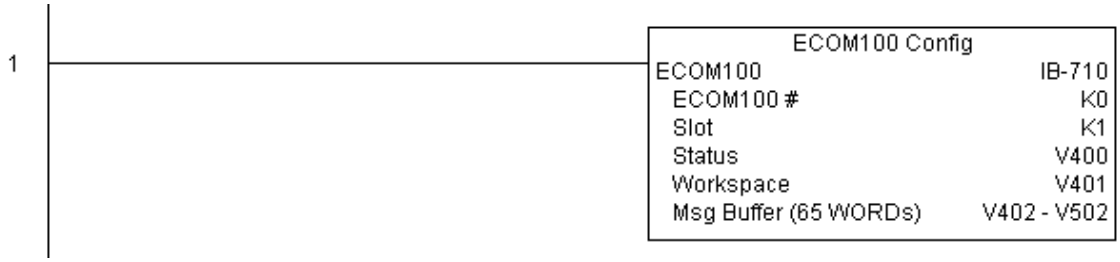
Parámetros de ECDHCPE

- **ECOM100#**: Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Timeout(sec)**: Especifica un período de atraso de modo que la instrucción se complete
- **Workspace**: Especifica una localización de memoria V que es usada por la instrucción
- **Success**: Especifica un bit que se activa cuando la petición se completa con éxito
- **Error**: Especifica un bit que se activa cuando la petición no se ha completado con éxito
- **Error Code**: Especifica la localización en donde será escrito el código de error

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Timeout (sec) . . . . . K	K5-127
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECDHCPE

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rórulot en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido, se habilita DHCP en el ECOM100. DHCP es el mismo protocolo usado en PCs para usar un servidor de DHCP para asignar automáticamente el IP ADDRESS , la dirección de la entrada, y el subnet mask del módulo ECOM100. Esto se hace típicamente usando NetEdit3, pero este IBox permite que usted habilite DHCP en el ECOM100 usando el programa ladder. La instrucción ECDHCPE es accionada por una transición de OFF para ON, y entonces no es controlado por flujo de energía (Por ejemplo, tal como una entrada de contador). Los comandos para habilitar DHCP serán enviados al ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO. La instrucción ECDHCPE no hace más que activar el bit para activar DHCP en el ECOM100, y luego interroga el ECOM100 una vez cada segundo para ver si el ECOM100 ha encontrado un servidor de DHCP y tiene un IP ADDRESS válido. Por lo tanto, es necesario un parámetro de **timeout** en caso de que el ECOM100 no pueda encontrar un servidor de DHCP. Si ocurre un **timeout**, es decir, no encuentra el servidor después de un tiempo, el bit del error se activará y el código de error será el decimal 1005. El bit de éxito(Success) se activará solamente si el ECOM100 encuentra un servidor de DHCP y le asigna un IP ADDRESS válido. Si se encuentra el servidor, el bit C100 se activa. Si hay una falla, se activa el bit C101. Si hay un error, usted puede inspeccionar V2000 para ver el código de error específico.

Configuración de ECOM100 Query DHCP (ECDHCPQ) (IB-734)

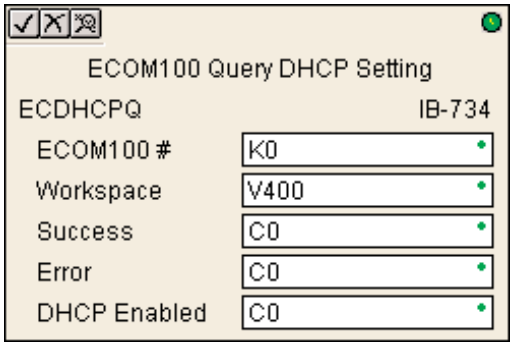
DS5	Usado
HPP	N/A

Esta instrucción determinará si DHCP está habilitado en el módulo ECOM100 en una transición desde OFF para ON a este IBox. El bit del parámetro DHCP Enabled estará ENCENDIDO si DHCP está habilitado o APAGADO si está deshabilitado.

El parámetro del espacio de trabajo (Workspace) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta instrucción y NO DEBE ser usado en ningún otro lugar en el programa.

El parámetro del bit de éxito(Success) o de error (Error) se activará cuando el comando se haya completado.

Para este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición en el circuito de ECOM100.



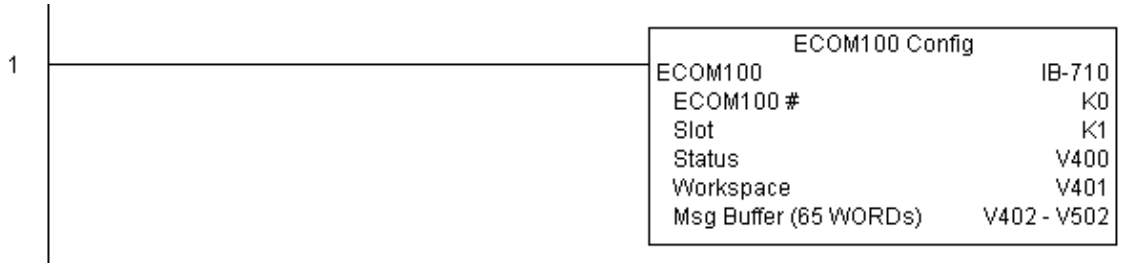
Parámetros de ECDHCPQ

- **ECOM100#:** Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** specifies a bit that will turn on once the instrucción is completed successfully
- **Error**: Especifica un bit que se activa cuando la requisición no se ha terminado con éxitoy completed
- **DHCP Enabled:** Especifica un bit que se activa si el DHCP de módulo ECOM100 se habilita o sigue estando apagado si está deshabilitado - después de que una petición a la instrucción, asegúrese verificar el estado del bit de Success o de Error junto con estado del bit DHCP Enabled para confirmar una petición acertada al módulo

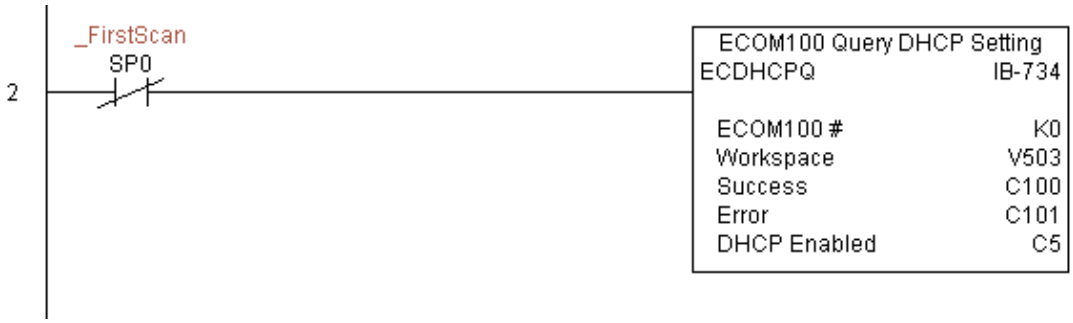
Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
DHCP Enabled . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de ECDHCPQ

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



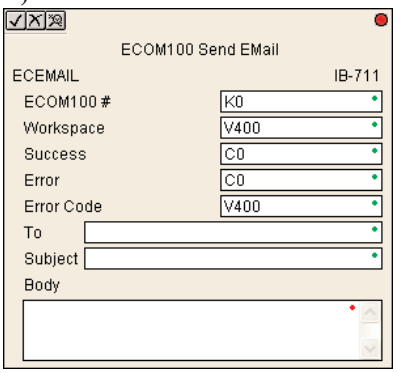
Renglón 2: En el segundo barrido del PLC, lee si DHCP está habilitado o deshabilitado en el ECOM100 y lo almacena en C5. DHCP es el mismo protocolo usado por PCs para usar un servidor de DHCP para asignar automáticamente el IP ADDRESS de módulos ECOM100, la dirección del Gateway, y el subnet mask. La instrucción ECDHCPQ es activada por una transición de OFF para ON, (similar a una entrada de un contador). El comando de leer (Query) si DHCP está habilitado o no será enviado al ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO. Si es hecho con éxito, el bit C100 se activa. Si hay una falla, se activa el bit C101.



Envíar E-mail con ECOM100 (ECEMAIL) (IB-711)

DS5	Usado
HPP	N/A

Esta instrucción, se comportará como cliente de correo electrónico (E-mail) y enviará una petición de SMPT a su servidor de SMTP para enviar un mensaje de e-mail a las direcciones de e-mail en el campo **To**: y también a ésos enumerados en cc:, lista definida explicitamente en la instrucción ECCEMAIL, cuando haya una transición de OFF para ON. Enviará la petición de SMTP basada en el número de ECOM100 especificado, que corresponde a una configuración única específica ECOM100 en la parte superior de su programa.



El campo Body (Contenido) contiene lo mismo que las instrucciones PRINT y VPRINT para texto y variables embutidas, permitiendo que usted embuta datos en tiempo real en su email (por ejemplo. "V2000 =" V2000:B).

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta instrucción y NO DEBE ser usado en cualquier otro lugar en su programa. O el parámetro de éxito o de error se activará una vez que la petición se haya completado. Si hay un error, el parámetro del código de error divulgará un código de error ECOM100 (menos de 100), un error del protocolo de SMPT (entre 100 y 999), o un error de lógica del PLC (mayor de 1000).

Puesto que el ECOM100 es solamente un cliente de e-mail y requiere el acceso a un servidor de SMTP, usted DEBE tener los parámetros de SMTP configurados correctamente en el ECOM100 vía el Home Page y/o la instrucción de configuración del e-mail (ECEMSUP) de ECOM100's. Para obtener al Home Page del módulo ECOM100, use su browser preferido de Internet y navegue hasta el IP ADDRESS del módulo ECOM100, por ejemplo. http://192.168.12.86

Hay una limitación de hasta aproximadamente 100 caracteres en los datos de mensaje para la instrucción entera, incluyendo los campos **Subject:** y **Body:**. Para ahorrar espacio, el módulo ECOM100 le permite tener una lista explicita de direcciones de e-mail en el campo de copia a carbón (cc:) de modo que usted pueda configurar ésos en el módulo ECOM100, y mantenga el campo **To:** pequeño (o aún vacío), para dejar mas espacio para los campos **Subject:** y **Body:**.

Para que este IBox de ECOM100 funcione, usted debe mover el DIP Switch 7 a ON en el circuito del módulo ECOM100.

Parámetros de ECEMAIL

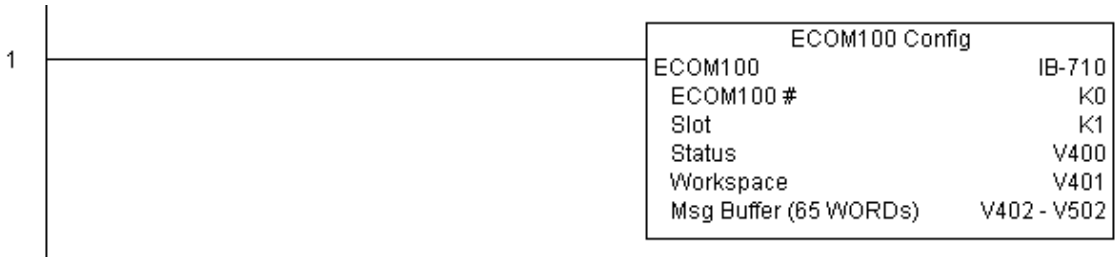
- **ECOM100#:** Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que deben referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace**: Especifica una localización de memoria V que es usada por la instrucción
- **Success**: Especifica un bit que se activa cuando la petición se completa con éxito
- **Error**: Especifica un bit que se activa cuando la petición no se ha completado con éxito

- **Error Code:** Especifica la localización en donde será escrito el código de error
- **To:** Especifica la dirección de E-mail donde será enviado el mensaje
- **Subject:** Asunto del mensaje de E-mail
- **Body:** Mensaje con datos que son iguales a los que pueden tenr las instrucciones PRINT y VPRINT para texto y variables embutidas, permitiendo que usted embutA datos en tiempo real en el mensaje del e-mail

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria DL06
To:.	Text
Subject:.	Text
Body:.	See PRINT and VPRINT instruccións

Ejemplo de ECEMAIL

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100 K0. El resto de los IBoxes ECxxxx se refieren a este módulo como K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



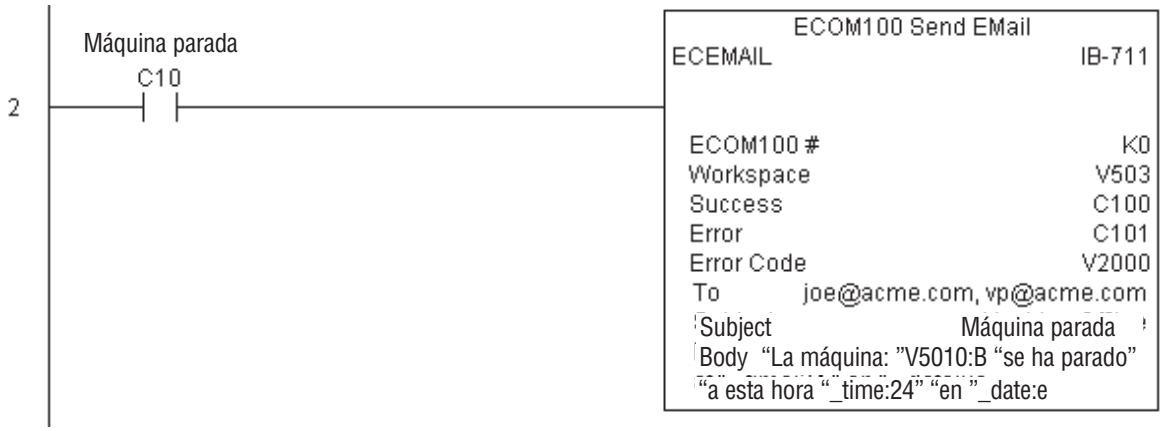
(Este ejemplo continúa en la próxima página)

Ejemplo de ECEMAIL (continuado)

Renglón 2: Cuando se para una máquina, envie un e-mail a Joe (joe@acme.com) en el departamento de mantención y al gerente de producción (vp@acme.com) informando qué máquina está parada junto con la fecha y hora cuando la máquina se paró.

El ECEMAIL es accionado con una trancisión de OFF para ON, no necesita de un flujo de energía constante (similar a una entrada de un contadir cuando cuenta una vez). Será enviado un email siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO. Esto ayuda a prevenir que se envíen correos electrónicos continuamente.

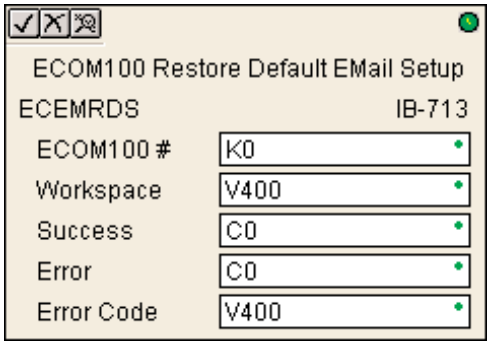
Si se envía el email, se activa el bit C100. Si hay una falla, se activa el bit C101. Si falla, usted puede observar V2000 para ver el código de error de SMTP u otros códigos de error posibles.



ECOM100 Restore Default E-mail Setup (ECEMRDS) (IB-713)

DS5	Usado
HPP	N/A

Esta instrucción de restaurar la configuración del e-mail por defecto ECOM100, en una transición de OFF para ON, restaurará los datos originales de configuración del e- mail almacenados en el ECOM100 de nuevo a la copia de trabajo basada en ECOM100 # especificado, que corresponde a una configuración única específica de ECOM100 (ECOM100) en la partre superior de su programa.



Cuando el ECOM100 es energizado, copia los datos de la configuración del e-mail almacenados en la ROM a la copia de trabajo en RAM. Ud. puede entonces modificar esta copia de trabajo del programa usando el IBox de configuración del e-mail ECOM100 (ECEMSUP). Después de modificar la copia de trabajo, usted puede restaurar más adelante los datos originales de la configuración con el programa usando este IBox.

El parámetro del espacio de trabajo es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta instrucción y NO DEBE ser usado en ningún otro lugar en su programa.¶

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000).

Para este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

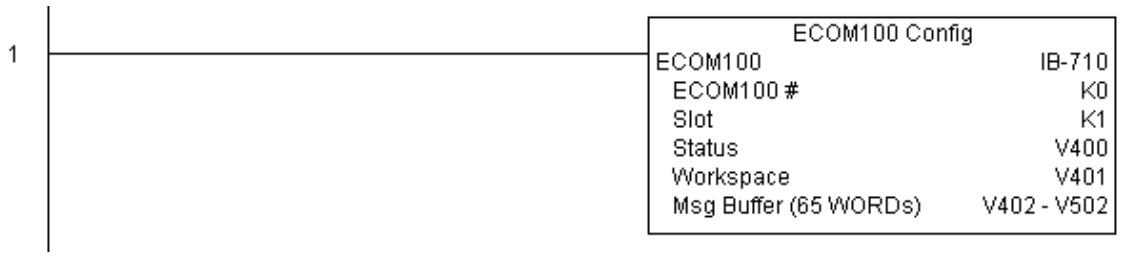
Parámetros de ECEMRDS

- **ECOM100#:** Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code: Especifica la localización en donde será escrito el código de error**

Parámetro	Rango del DL06
ECOM100# ..... K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words

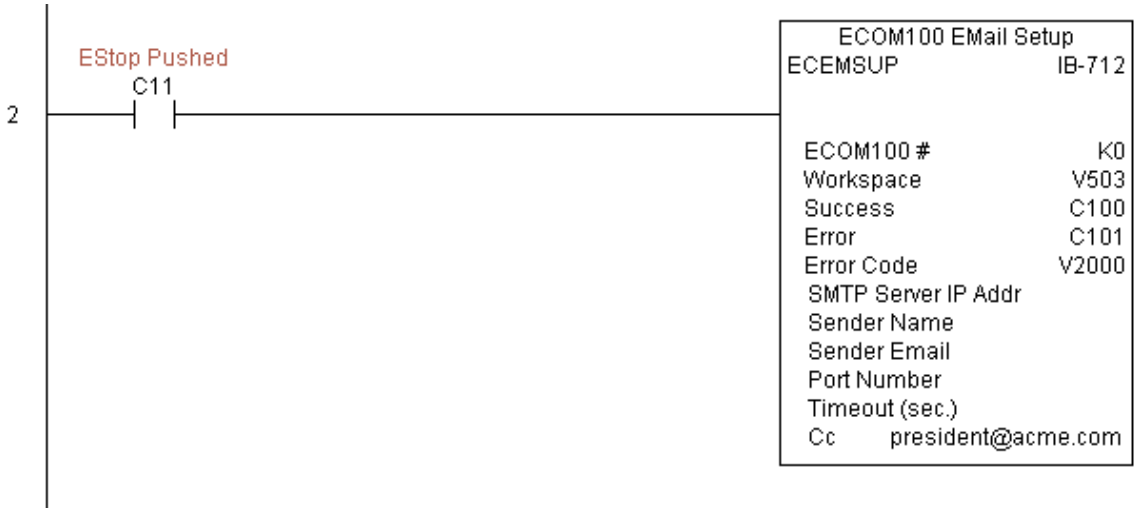
Ejemplo de ECEMRDS

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100 K0. El resto de los IBoxes ECxxxx se refieren a este módulo como K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: Siempre que se empuje una parada de emergencia, asegúrese que el presidente de la compania reciba copias de todos los e-mails que son enviados.

La instrucción IBox de configuración de una e-mail de ECOM100 le permite definir o cambiar los datos de configuración de SMTP para uso con e-mail almacenados en el módulo ECOM100.

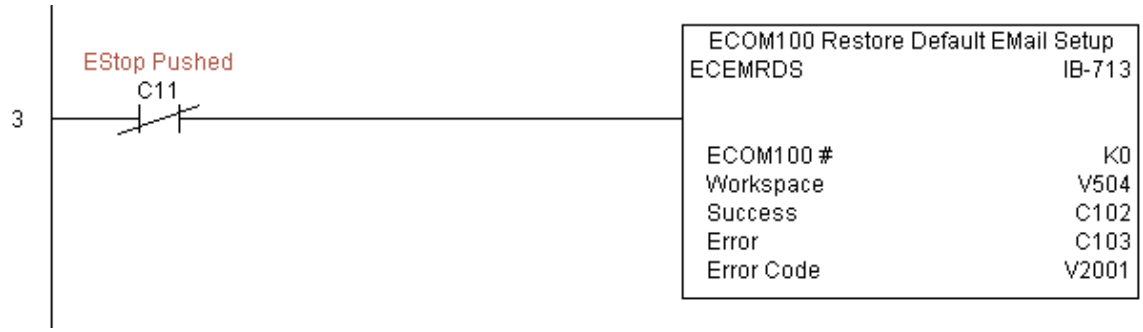


Ejemplo de ECEMRDS (continuado)

Renglón 3: Una vez que se rearme la parada de emergencia, retire al presidente de la lista cc: restaurando la configuración de email por defecto en el módulo ECOM100.

La instrucción ECEMRDS se acciona en la transición de OFF para ON , (similar a la entrada de conteo de un contador). La configuración del e- mail almacenada en la memoria ROM del módulo ECOM100 será copiada sobre la "copia de trabajo" siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO (la copia de trabajo puede ser cambiada usando la instrucción IBox ECEMSUP).

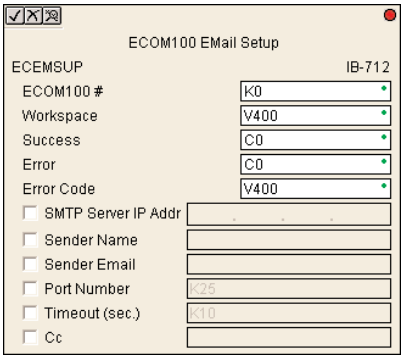
Si funciona correctamente, se activa el bit C102. Si hay una falla, se activa C103. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.



Configuración de E-mail con ECOM100 (ECEMSUP) (IB-712)

DS5	Usado
HPP	N/A

Esta instrucción, en una transición de OFF para ON, modificará la copia de trabajo de la configuración de e-mail actual en el ECOM100 basado en el número especificado de ECOM100, que corresponde a una configuración única específica ECOM100 (ECOM100) en la parte superior de su programa.



Usted puede escoger y elegir cualquiera o todos los campos que se modificarán usando esta instrucción. Observe que estos cambios son acumulativos: Si usted se ejecuta IBoxes múltiples de configuración de e-mail de ECOM100, después todos los cambios se realizan en la orden que se ejecutan. También observe que usted puede restaurar la configuración del e-mail original ECOM100 que es almacenado en el ECOM100 a la copia de trabajo usando el IBox de restaurar la configuración del email por defecto ECOM100 (ECEMRDS).

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

El parámetro del bit de éxito (Success) o de error se activará una vez que el comando se haya completado .Si hay un error, el parámetro del código de error divulgará un código de error ECOM100 (menos de 100), o un error de la lógica del PLC (mayor de 1000).

Hay un límite de aproximadamente 100 caracteres/bytes de datos de configuración para la instrucción entera. Si fuera necesario, usted podría dividir la configuración entera con IBoxes múltiples ECEMSUP sobre una base de campo por campo, por ejemplo, hace el campo copia a carbón (cc:) en un IBox ECEMSUP y los parámetros restantes de configuración en otro.

Para que este ECOM100 IBox funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

Parámetros de ECEMSUP

- **ECOM100#:** Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- Error Code: Especifica la localización en donde será escrito el código de error
- **SMTP Server IP Addr:** Parámetro opcional que especifica el IP ADDRESS del servidor del SMTP en la red de módulos ECOM100
- **Sender Name:** Parámetro opcional que especifica el nombre del remitente el cual aparecerá en el campo "From:"a los que reciben el E-mail
- **Sender EMail:** Parámetro opcional que especifica la dirección del e-mail del remitente de la cual aparecerá en el campo "From:"a los que reciben el E-mail

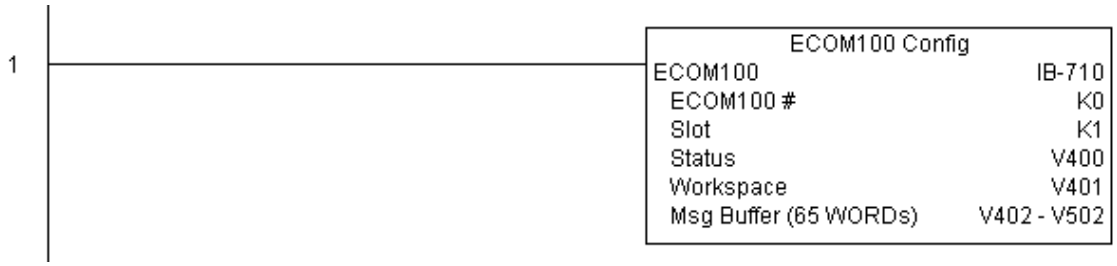
Parámetros de ECEMSUP (continuado)

- **Port Number:** Parámetro opcional que especifica the TCP/IP Port Number to send SMTP requests; usually this does not to be configured (see your network administrator for information on this setting)
- **Timeout (sec):** Parámetro opcional que especifica la cantidad de segundos que debe esperar para que el servidor SMTP envíe el e-mail a todos los que reciben éste.
- **Cc:** Parámetro opcional que especifica una lista de direcciones de e-mail "carbon copy" que será enviadas

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECEMSUP

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.

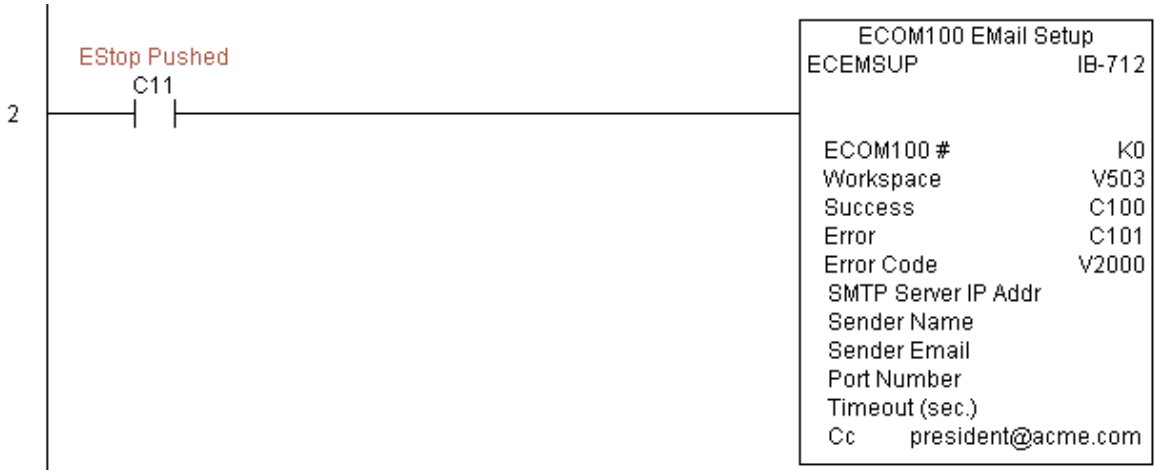


(Este ejemplo continúa en la próxima página)

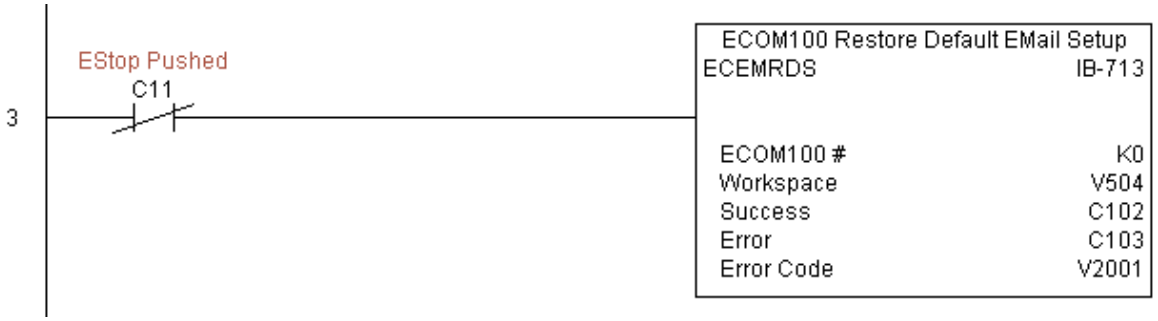
Ejemplo de ECEMSUP (continuado)

Renglón 2: Siempre que se empuje un botón de parada de emergencia, asegúrese de que el presidente de la compañía consiga las copias de todos los email. La instrucción IBox de configuración de email con ECOM100 le permite definir o cambiar los parámetrso de smtp de un e- mail almacenados en el ECOM100. La instrucción ECEMSUP es activada en la transición de OFF para ON (similar a una entrada de un contador). En la energización del PLC, la configuración del e- mail almacenada en la memoria ROM del ECOM100 se copia a una memoria RAM "copia de trabajo". Usted puede cambiar esta copia de trabajo usando el IBox ECEMSUP. Para restaurar la la configuración original en la memoria ROM, use la instrucción IBox ECEMRDS.

Si funciona correctamente, se activa el bit C102. Si hay una falla, se activa el bit C103. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.



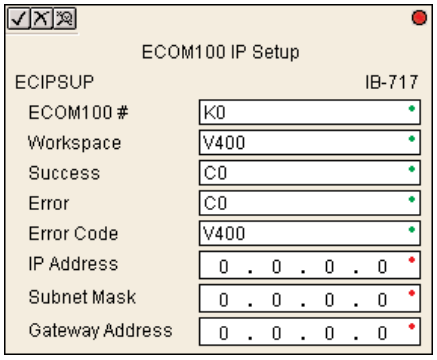
Renglón 3: Una vez que se remueva la condición de parada de emergencia , saque el presidente de la lista en cc: restaurando la configuración de e-mail por defecto en el módulo ECOM100.



Configuración de PI de ECOM100 (ECIPSUP) (IB-717)

DS5	Usado
HPP	N/A

Esta instrucción configurará los tres parámetros de TCP/IP en el ECOM100: IP ADDRESS, subnet mask, y dirección de Gateway, en una transición desde OFF para ON para activar el IBox. El ECOM100 es especificado por el número ECOM100 #, que corresponde a un IBox de configuración única (ECOM100) en la parte superior de su programa.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000).

Estos datos de configuración se almacenan en memoria Flash-ROM en el ECOM100 e deshabilitarán el módulo ECOM100 por lo menos un medio segundo hasta que se escribe la memoria Flash-ROM. Por lo tanto, SE RECOMIENDA ALTAMENTE que usted ejecute solamente este IBox UNA VEZ en el primer barrido. Ya que se requiere una transición de OFF para ON, use un SP0 NORMALMENTE CERRADO para ejecutar al IBox.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

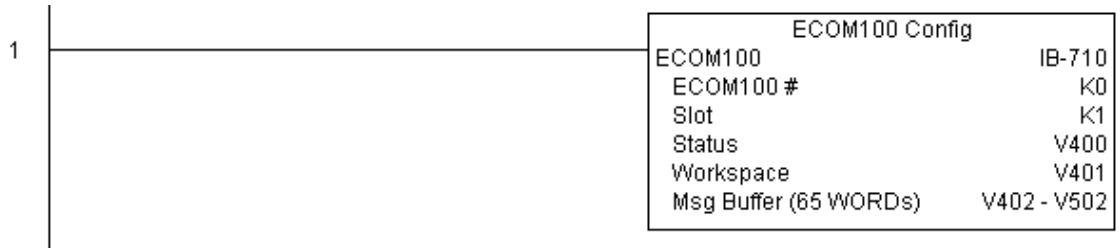
Parámetros de ECIPSUP

- **ECOM100#:** Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace**: Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxitoy completed
- **Error Code: Especifica la localización en donde será escrito el código de error**
- **IP Address:** Especifica la dirección de IP del módulo
- **Subnet Mask:** Especifica la Subnet Mask para el módulo en questión
- **Gateway Address:** Especifica la dirección de Gateway para el módulo en questión

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words
IP Address . . . . . IP Address	0.0.0.1. to 255.255.255.254
Subnet Mask Address . . . . . IP Address Mask	0.0.0.1. to 255.255.255.254
Gateway Address . . . . . IP Address	0.0.0.1. to 255.255.255.254

Ejemplo de ECIPSUP

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.

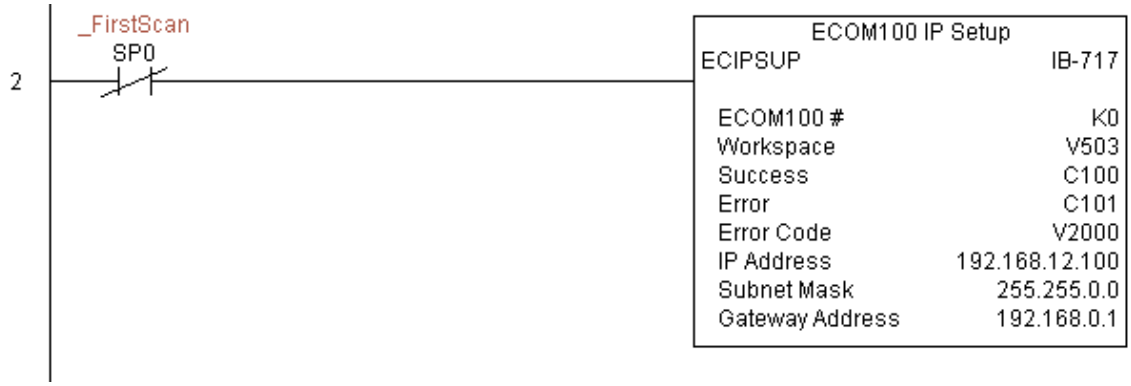


Renglón 2: En el segundo barrido, configure todos los parámetros de TCP/IP en el ECOM100:

IP Address:	192.168. 12.100
Subnet Mask:	255.255. 0. 0
Dirección de Gateway:	192.168. 0. 1

La instrucción ECIPSUP es accionada en una transición de OFF para ON, (similar a una entrada de un contador). El comando de escribir los parámetros de la configuración de TCP/IP será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

Si funciona correctamente, se activa el bit C102. Si hay una falla, se activa C103. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.

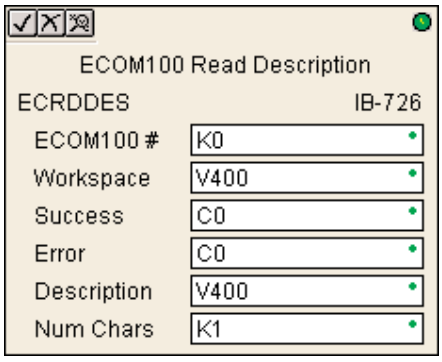


Lea la descripción del ECOM100 (ECRDDES) (IB-726)

DS5	Usado
HPP	N/A

Esta instrucción leerá el campo de descripción del módulo ECOM100 hasta el número de caracteres especificados en una transición de OFF para ON al IBox.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.MUST BE UNIQUE in this one instrucción and MUST NOT be usado anywhere else in your program.



Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000).ete.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición en el circuito de ECOM100. ECOM100 circuit board.

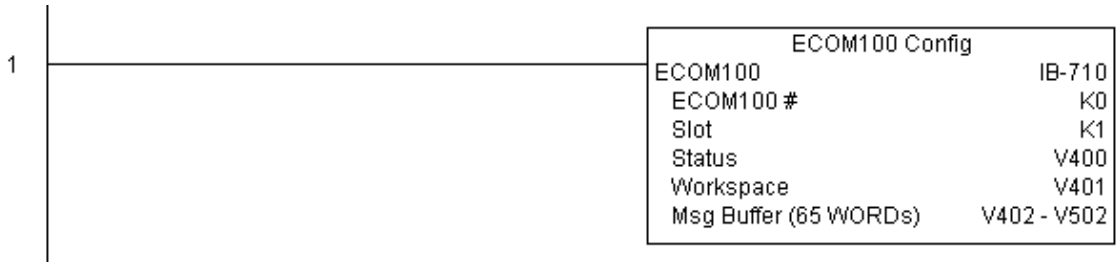
Parámetros de ECRDDES

- **E**COM100#: éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico cified slot. All other ECxxxx IBoxes that need to reference this ECOM100 module must reference this logical number
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Descripción:** Especifica la dirección de memoria V para el almacemiento donde el nombre del módulo será colocado
- **Num Char:** Especifica la cantidad de caracteres (bytes) a ser leídos desde el campo Descripción del módulo ECOM100

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Descripción V	Vea el mapa de memoria V del DL06 - Data Words
Num Chars . . . . . K	K1-128

Ejemplo de ECRDDES

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulo como K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido , lee la descripción del módulo del módulo ECOM100 y la almacena en las memorias V3000 hasta V3007 (16 caracteres). Este texto se puede exhibir en una interface de operador, por ejemplo.

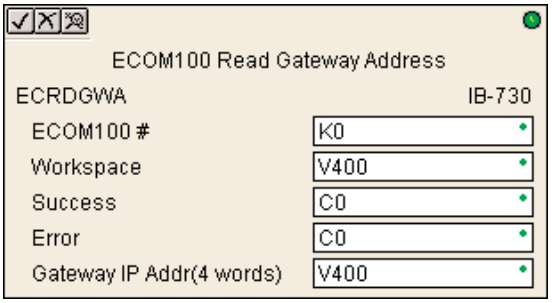
La instrucción ECRDDES es activada en una transición de OFF para ON, (similar a una entrada de un contador). El comando de leer la descripción del módulo será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa C101.

Lea la dirección Gateway del ECOM100 (ECRDGWA) (IB-730)

DS5	Usado
HPP	N/A

Esta instrucción leerá las 4 partes de la dirección IP del Gateway y las almacenará en 4 posiciones de memoria V consecutivas en formato decimal, en una transición desde OFF para ON para activar este IBox.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000).ete.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

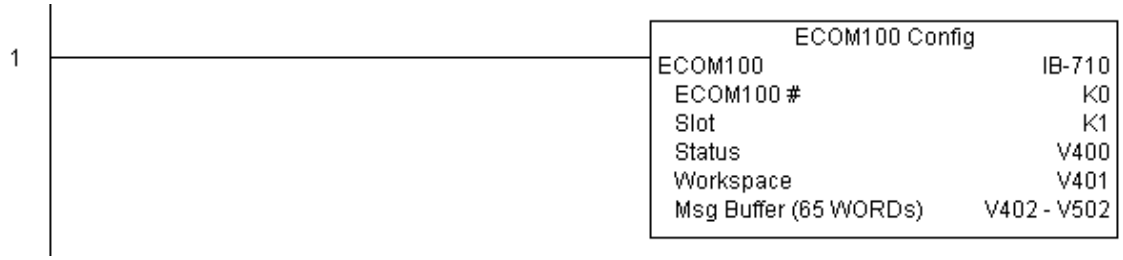
Parámetros de ECRDGWA

- **ECOM100#:** éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico cified slot. All other ECxxxx IBoxes that need to reference this ECOM100 module must reference this logical number
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- Gateway IP Addr: Especifica la dirección inicial donde la dirección del Gateway del módulo ECOM100 será colocada en 4 localizaciones consecutivas de memoria V

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Gateway IP Address (4 Words) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECRDGWA

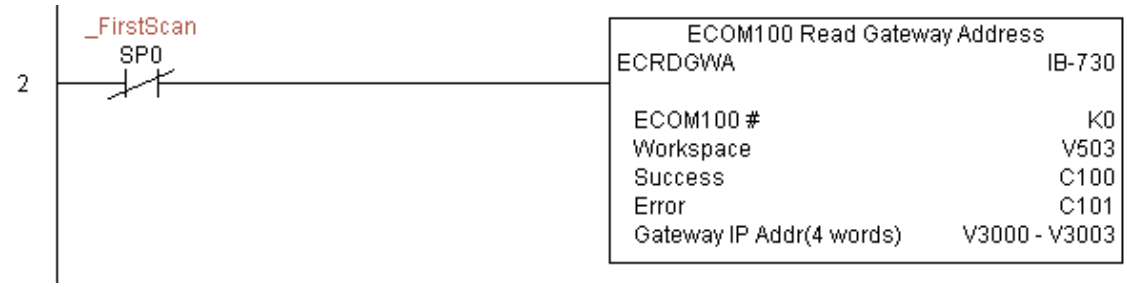
Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulo como K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido , lee la dirección Gateway del módulo ECOM100 y la almacena en V3000 hasta V3003 (4 númerosdecimales). Esta información se puede exhibir en una interface de operador, por ejemplo.

La instrucción ECRDGWA es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de leer la dirección de Gateway del módulo será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

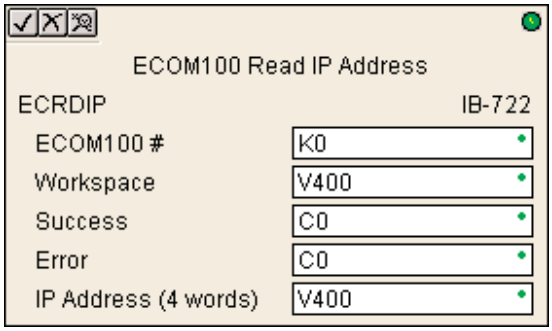
Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa C101.



ECOM100 Read IP Address (ECRDIP) (IB-722)

DS5	Usado
HPP	N/A

ECOM100 Read IP Address will read the 4 parts of the IP address and store them in 4 consecutive V Memory locations in decimal format, on a transition from OFF to ON to the IBox.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), etc.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición en el circuito de ECOM100. ECOM100 circuit board.

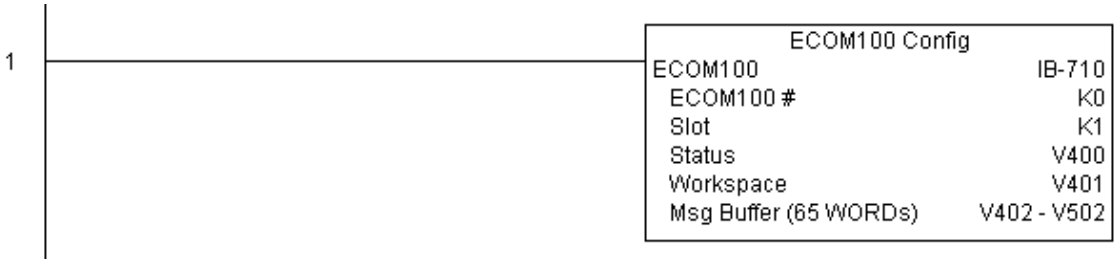
Parámetros de ECRDIP

- **ECOM100#**: éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico cified slot.
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **IP Address:** Especifica la dirección inicial donde será colocada la dirección de IP del módulo ECOM100, en 4 localizaciones consecutivas de memoria V

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
IP Address (4 Words) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECRDIP

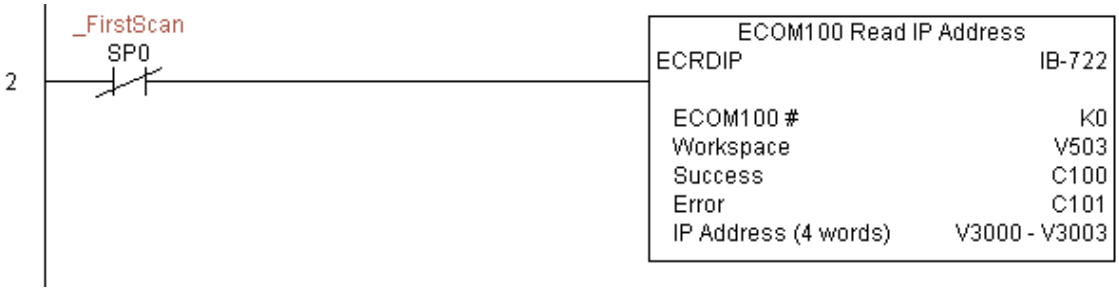
Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido, lee la dirección IP del módulo ECOM100 y la almacena en V3000 hasta V3003 (4 númerosdecimales). Esta información se puede exhibir en una interface de operador, por ejemplo.

La instrucción ECRDIP es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de leer la dirección IP del módulo será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa el bit C101.

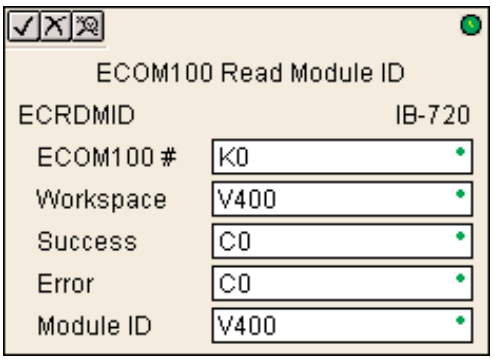


ECOM100 Read Module ID (ECRDMID) (IB-720)

DS5	Usado
HPP	N/A

ECOM100 Read Module ID will read the binary (decimal) WORD sized Module ID on a transition from OFF to ON to the IBox.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.



Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), etc.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

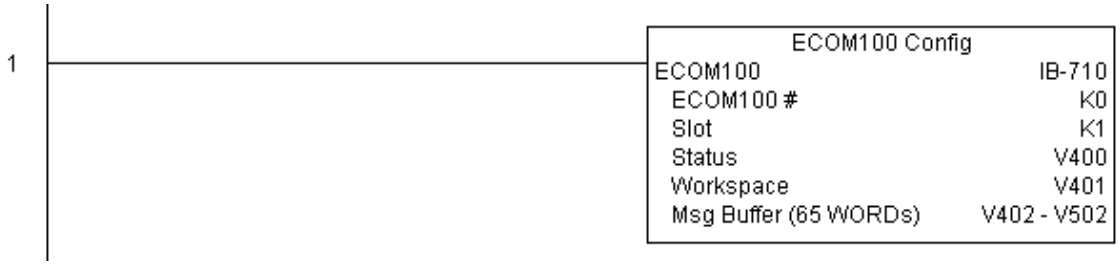
Parámetros de ECRDMID

- **ECOM100#**: Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Module ID**: Especifica la localización donde será colocada la identificacion **Module ID** (decimal) del módulo ECOM100

Parámetro	Rango del DL06
ECOM100# ..... K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Module ID. V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECRDMID

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todos los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido, lee la identificación ID del módulo ECOM100 y la almacena en V2000. Esta información se puede exhibir en una interface de operador, por ejemplo.

La instrucción ECRDMID es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de leer la identificación ID del módulo será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

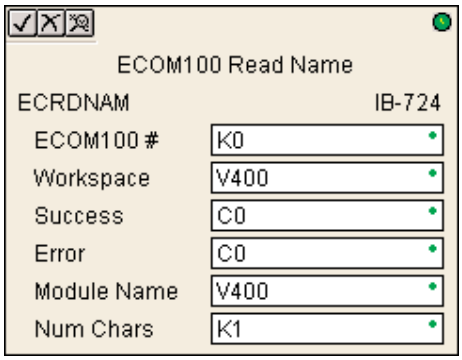
Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa el bir C101.

Leer el nombre del módulo ECOM100 (ECRDNAM) (IB-724)

DS5	Usado
HPP	N/A

Esta instrucción leerá el nombre del módulo hasta el número de caracteres especificados en una transición de APAGADO a ENCENDIDO al IBox.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.



Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), ete.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

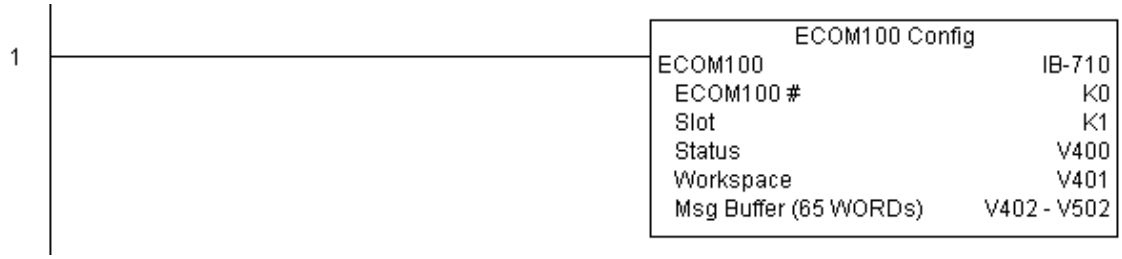
Parámetros de ECRDNAM

- **ECOM100#:** éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Module Name:** Especifica la localización inicial del almacenador intermediario en donde será almacenado el nombre del módulo ECOM100
- **Num Chars:** Especifica la cantidad de caracteres (bytes) a ser leídos desde el campo Module Name del módulo ECOM100

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Module Name V	Vea el mapa de memoria V del DL06 - Data Words
Num Chars . . . . . K	K1-128

Ejemplo de ECRDNAM

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todos los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido, lee el Module Name (Nombre del módulo) de ECOM100 y la almacena en V3000 hasta V3003 (8 caracteres). Esta información se puede exhibir en una interface de operador, por ejemplo.

La instrucción ECRDNAM es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de leer el nombre del módulo será enviado desde el módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

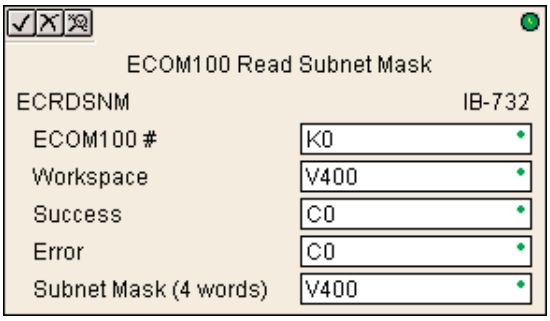
Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa el bit C101.

Lea Subnet Mask ECOM100 (ECRDSNM) (IB-732)

DS5	Usado
HPP	N/A

Esta instrucción permite leer las 4 partes del Subnet Mask y las almacena en 4 localizaciones de memria V consecutivas en formato decimal, en una transición desde OFF para ON al IBox.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.



Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), ete.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

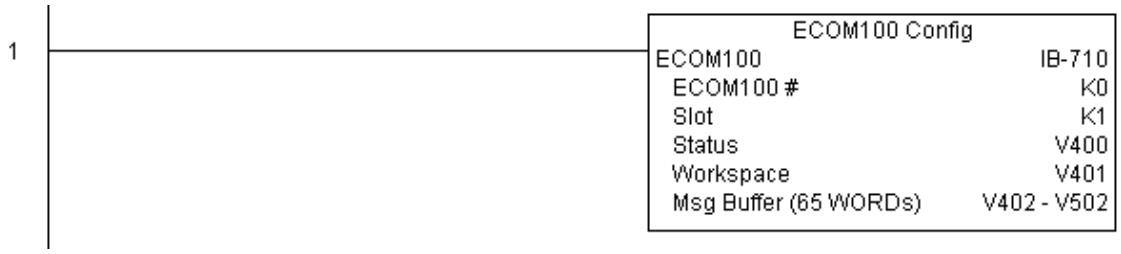
Parámetros de ECRDSNM

- **ECOM100#**: éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxitoy completed
- **Subnet Mask:** Especifica la dirección inicial donde será almacenada la Subnet Mask del módulo ECOM100 en 4 localizaciones de memoria V consecutivas

Parámetro	Rango del DL06
ECOM100# ..... K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Subnet Mask (4 Words) V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de ECRDSNM

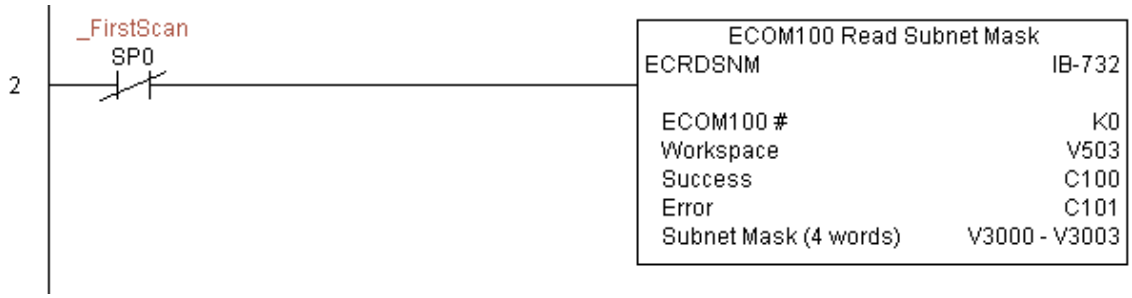
Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todos los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido, lee el Subnet Mask desde el módulo ECOM100 y la almacena en V3000 hasta V3003 (4 números decimales). Esta información se puede exhibir en una interface de operador, por ejemplo.

La instrucción ECRDSNM es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de leer el Subnet Mask será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

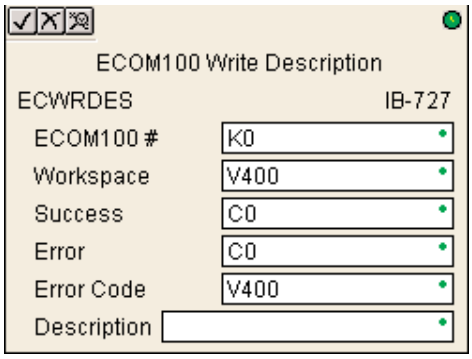
Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa C101.



ECOM100 Write Descripción (ECWRDES) (IB-727)

DS5	Usado
HPP	N/A

Esta instrucción permite escribir una descripción al módulo ECOM100 en una transición de APAGADO a ENCENDIDO al IBox.



Si usted utiliza un signo dólar ($) o una comilla ("), use la secuencia de escape de PRINT/VPRINT de **dos** signos dólar ($$) para un signo solamente o una comilla con un signo dólar ($") para un carácter de comilla.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), ete.

La descripción se almacena en la memoria Flash-ROM en el módulo ECOM100 y la ejecución de este IBox deshabilitará el módulo ECOM100 por lo menos un medio segundo hasta que escribe la memoria Flash-ROM. Por lo tanto, SE RECOMIENDA que usted ejecute solamente este IBox UNA VEZ en el primer barrido. Ya que se requiere la ejecución de una trancision de OFF para ON, use un contacto SP0 NORMALMENTE CERRADO para conducir el flujo de energía al IBox.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

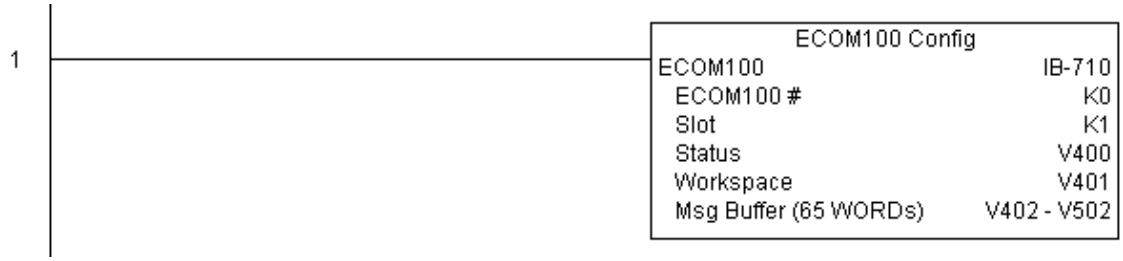
Parámetros de ECWRDES

- **ECOM100#**: éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico cified slot. All other ECxxxx IBoxes that need to reference this ECOM100 module must reference this logical number
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code**: Especifica la localización en donde será escrito el código de error
- **Descripción:** Especifica la descripción (Descripción) que será escrita al módulo

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words
Descripción	Text

Ejemplo de ECWRDES

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido del PLC, esta instrucción define la descripción del módulo (Module Descripción) ECOM100. Típicamete, this is done using NetEdit, but this IBox allows you to configure the module descripción in the ECOM100 using your ladder program.

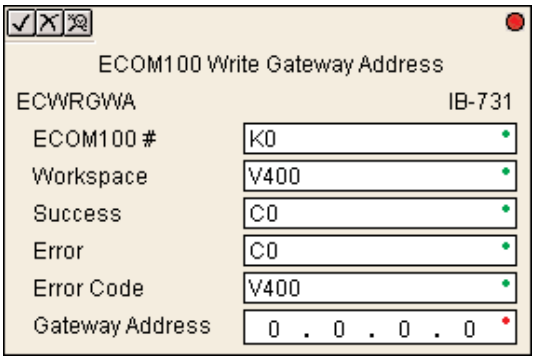
La instrucción EWRDES es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de escribir la descripción del módulo será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa el bit C101. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.

Escriba la dirección de Gateway del ECOM100 (ECWRGWA) (IB-731)

DS5	Usado
HPP	N/A

Esta instrucción permite escribir una dirección dada deGateway al módulo ECOM100 en una transición de APAGADO a ENCENDIDO al IBox. Vea también la instrucción IBox ECIPSUP IB-717 para configurar TODOS LOS parámetros de TCP/IP en una sola instrucción - IP ADDRESS, subnet mask, y dirección de Gateway.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), ete.

La dirección del Gateway se almacena en memria Flash-ROM en el ECOM100 y la ejecución de este IBox deshabilitará el módulo ECOM100 por lo menos un medio segundo hasta que escribe la Flash-ROM. Por lo tanto, SE RECOMIENDA que usted ejecute solamente este IBox UNA VEZ, en el primer barrido. Ya que se requiere la ejecución de una trancision de OFF para ON, use un contacto SP0 NORMALMENTE CERRADO para conducir el flujo de energía al IBox.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

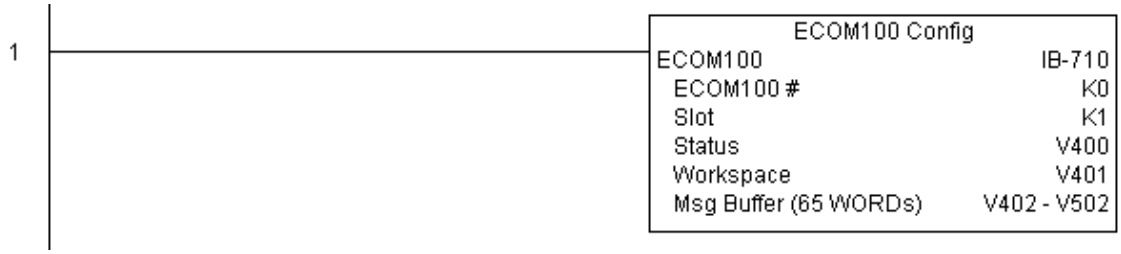
Parámetros de ECWRGWA

- **ECOM100#:** éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico cified slot. All other ECxxxx IBoxes that need to reference this ECOM100 module must reference this logical number
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code:** Especifica la localización en donde será escrito el código de error
- **Gateway Address:** Especifica la dirección de Gateway que será escrita el módulo

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words
Gateway Address	0.0.0.1. to 255.255.255.254

Ejemplo de ECWRGWA

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todos los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.

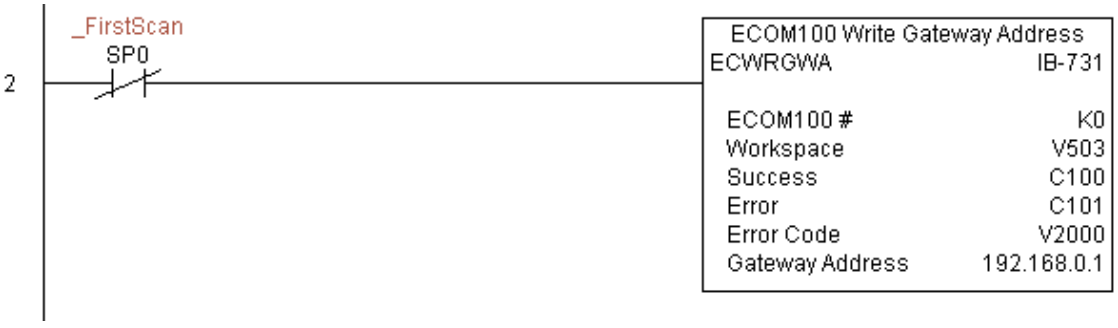


Renglón 2: En el segundo barrido del PLC, esta instrucción asigna la dirección de Gateway del módulo ECOM100 a 192.168.0.1.

La instrucción ECWRGWA es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de escribir la dirección de Gateway será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa el bit C101. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.

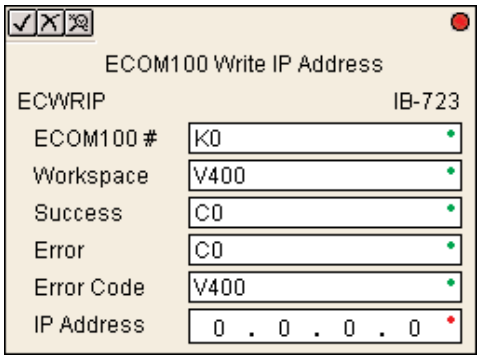
Para configurar todos los parámetros de TCP/IP al módulo ECOM100 en una instrucción, vea el IBox ECIPSUP.



ECOM100 Write IP Address (ECWRIP) (IB-723)

DS5	Usado
HPP	N/A

Esta instrucción escribe una direccion IP dada al módulo ECOM100 en una transición de APAGADO a ENCENDIDO al IBox. Vea también la instrucción IBox ECIPSUP IB-717 para configurar TODOS LOS parámetros de TCP/IP en una sola instrucción - IP ADDRESS, subnet mask, y dirección de Gateway.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), etc.

La dirección de IP se almacena en memoria Flash-ROM en el ECOM100 y la ejecución de este IBox deshabilitará el módulo ECOM100 por lo menos un medio segundo hasta que escribe la Flash-ROM. Por lo tanto, SE RECOMIENDA que usted ejecute solamente este IBox UNA VEZ, en el primer barrido. Ya que se requiere la ejecución de una trancision de OFF para ON, use un contacto SP0 NORMALMENTE CERRADO para conducir el flujo de energía al IBox.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

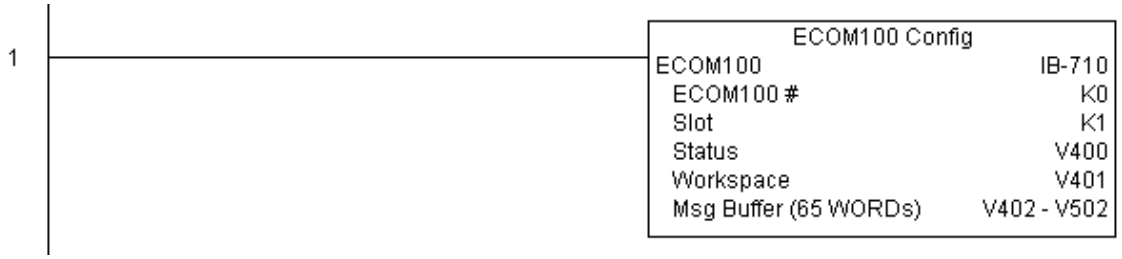
Parámetros de ECWRIP

- **ECOM100#:** éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code:** Especifica la localización en donde será escrito el código de error
- **IP Address:** Especifica la dirección IP que será escrita al módulo

Parámetro	Rango del DL06
ECOM100# ..... K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words
IP Address	0.0.0.1. to 255.255.255.254

Ejemplo de ECWRIP

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todos los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido del PLC, esta instrucción asigna la dirección IP Address del módulo ECOM100 a 192.168.12.100.

La instrucción ECWRIP es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de escribir la dirección de IP será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

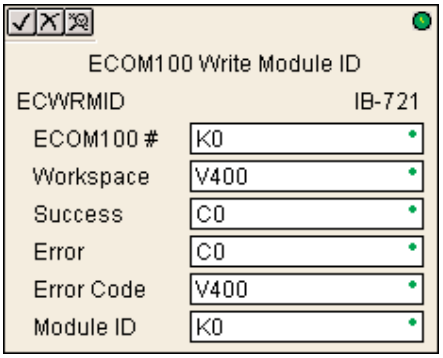
Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa el bit C101. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.

Para configurar todos los parámetros de TCP/IP de ECOM100 en una instrucción, vea el IBox ECIPSUP.

ECOM100 Write Module ID (ECWRMID) (IB-721)

DS5	Usado
HPP	N/A

Esta instrucción escribe una identificación dada (Module ID) al módulo en una transición de APAGADO a ENCENDIDO al IBox.



Si la identifiación del módulo (ID) es configurada en hardware usando los DIP switches, esta instrucción IBox will fail and return error code 1005 (decimal).

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), ete.

La información **Module ID** se almacena en memoria Flash-ROM en el ECOM100 y la ejecución de este IBox deshabilitará el módulo ECOM100 por lo menos un medio segundo hasta que escriba la memoria Flash-ROM. Por lo tanto, SE RECOMIENDA que usted ejecute solamente UNA VEZeste IBox, en el primer barrido. Ya que se requiere la ejecución de una trancision de OFF para ON, use un contacto SP0 NORMALMENTE CERRADO para conducir el flujo de energía al IBox.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

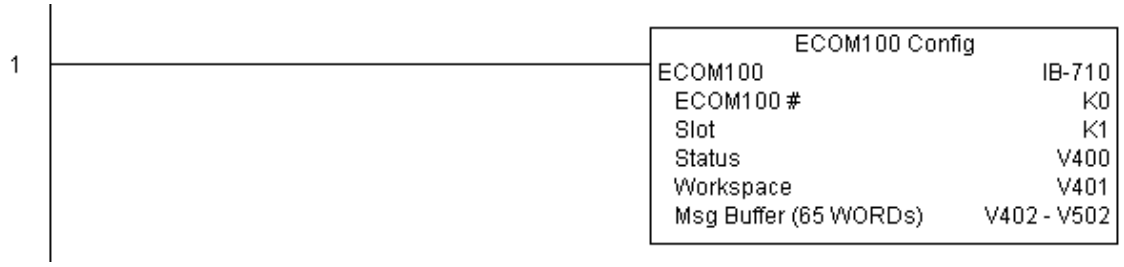
Parámetros de ECWRMID

- **ECOM100#:** éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico cified slot. All other ECxxxx IBoxes that need to reference this ECOM100 module must reference this logical number
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code:** Especifica la localización en donde será escrito el código de error
- **Module ID**: Especifica la identificación Module ID que será escrita al módulo

Parámetro	Rango del DL06
ECOM100# ..... K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words
Module ID	K0-65535

Ejemplo de ECWRMID

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todos los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido del PLC, esta instrucción asigna la identificación Module ID del módulo ECOM100. Típicamente esto es hecho usando NetEdit3, pero este IBox le permite configurar the module ID del ECOM100 usando el programa ladder.

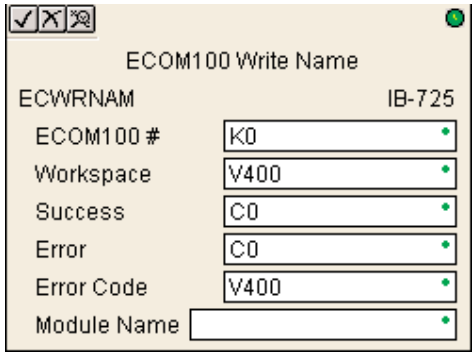
La instrucción EWRMID es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de escribir el module ID será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

Si funciona correctamente, se activa el bit C100. Si hay una falla, se activa el bit C101. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.

Escribir el nombre del ECOM100 (ECWRNAM) (IB-725)

DS5	Usado
HPP	N/A

Esta instrucción escribe un nombre dado al módulo ECOM100 en una transición de APAGADO a ENCENDIDO al IBox.



Si usted utiliza un signo dólar ($) o una comilla ("), use la secuencia de escape de PRINT/VPRINT de **dos** signos dólar ($$) para un signo solamente o una comilla con un signo dólar ($") para un carácter de comilla.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), ete.

El nombre se almacena en la memoria Flash-ROM en el ECOM100 y la ejecución de este IBox deshabilitará el módulo ECOM100 por lo menos un medio segundo hasta que escribe la Flash-ROM. Por lo tanto, SE RECOMIENDA que usted ejecute solamente este IBox UNA VEZ en el primer barrido. Ya que se requiere la ejecución de una trancision de OFF para ON, use un contacto SP0 NORMALMENTE CERRADO para conducir el flujo de energía al IBox.

Para que este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

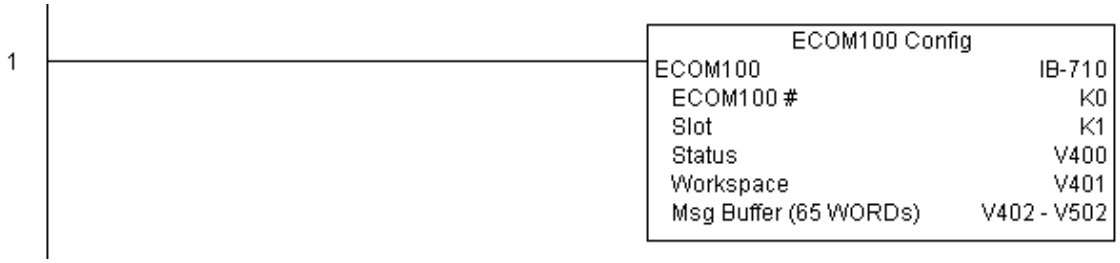
Parámetros de ECWRNAM

- **ECOM100#:** Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code:** Especifica la localización en donde será escrito el código de error
- **Module Name:** Especifica el nombre que será escrita al módulo

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words
Module Name	Text

Ejemplo de ECWRNAM

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido del PLC, esta instrucción asigna el nombre del módulo ECOM100. Típicamente esto es hecho usando NetEdit3, pero este IBox le permite configurar the nombre del ECOM100 usando el programa ladder.

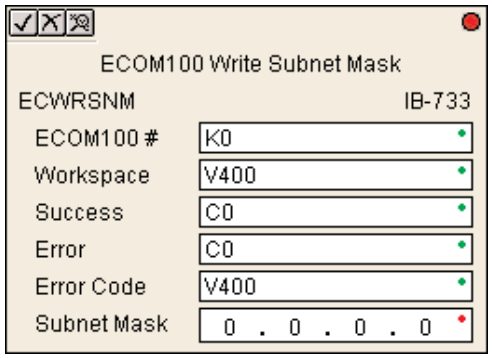
La instrucción EWRNAM es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de escribir el nombre será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

Si funciona correctamente, se activa el bit C100. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.

Escribir Subnet Mask a ECOM100 (ECWRSNM) (IB-733)

DS5	Usado
HPP	N/A

Esta instrucción escribe una Subnet Mask dada al módulo ECOM100 en una transición de APAGADO a ENCENDIDO al IBox. Vea también la instrucción IBox ECIPSUP IB-717 para configurar TODOS LOS parámetros de TCP/IP en una sola instrucción - IP ADDRESS, subnet mask, y dirección de Gateway.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Los bits de los parámetros Success o Error se activan una vez que el comando sea completado. Si hay un error, el parámetro Error Code (código de error) divulgará un código de error ECOM100 (menos de 100), o un error de lógica del PLC (mayor de 1000), ete.

La información de Subnet Mask se almacena en la memoria Flash-ROM en el ECOM100 y la ejecución de este IBox deshabilitará el módulo ECOM100 por lo menos un medio segundo hasta que escribe la Flash-ROM. Por lo tanto, SE RECOMIENDA que usted ejecute solamente este IBox UNA VEZ en el primer barrido. Ya que se requiere la ejecución de una trancision de OFF para ON, use un contacto SP0 NORMALMENTE CERRADO para conducir el flujo de energía al IBox.

Para este IBox ECOM100 funcione, usted debe mover el DIP switch 7 a la posición ON en el circuito de ECOM100.

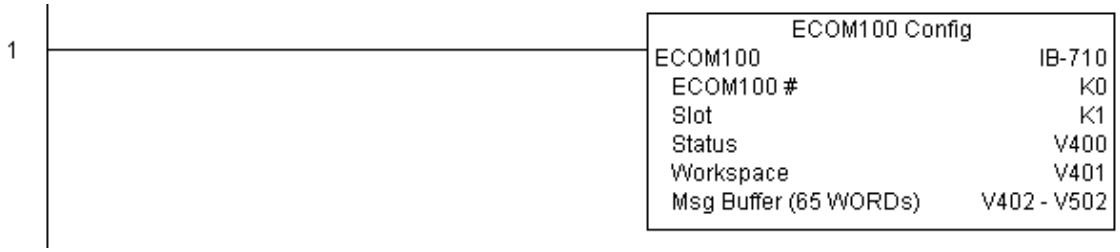
Parámetros de ECWRSNM

- **ECOM100#:** éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito
- **Error Code:** Especifica la localización en donde será escrito el código de error
- **Subnet Mask:** Especifica la Subnet Mask que será escrita al módulo

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error Code V	Vea el mapa de memoria V del DL06 - Data Words
Subnet Mask	Masked IP Address

Ejemplo de ECWRSNM

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinatar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



Renglón 2: En el segundo barrido del PLC, esta instrucción asigna el Subnet Mask del módulo ECOM100 a 255.255.0.0

La instrucción ECWRSNM es activada por una transición de OFF para ON, (similar a la entrada de un contador). El comando de escribir el Subnet Mask será enviado al módulo ECOM100 siempre que el flujo de energía en el IBox vaya de APAGADO a ENCENDIDO.

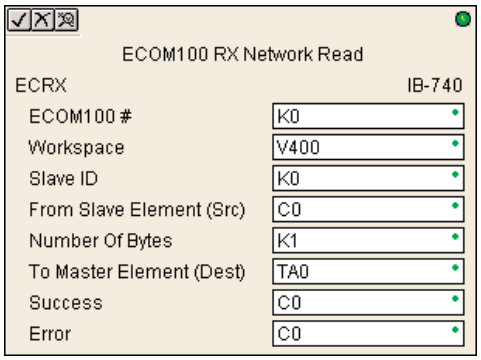
Si funciona correctamente, se activa el bit C100. Si hay una falla, usted puede mirar V2001 para ver el código de error específico.

Para configurar todos los parámetros de TCP/IP de ECOM100 en una instrucción, vea el IBox ECIPSUPP.

Lee datos RX con ECOM100 (ECRX) (IB-740)

DS5	Usado
HPP	N/A

Esta instrucción es la instrucción RX con enclavamiento incorporado con otros IBoxes de ECOM100 RX (ECRX) y ECOM100 WX (ECWX) en el programa para simplificar el establecimiento de una red de comunicaciones. Realizará la instrucción RX en la red especificada de módulos ECOM100, que corresponde a un IBox de configuración específica en la parte superior del programa.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Siempre que este IBox tenga energía, leerá datos de elementos del esclavo especificado en el almacenador intermediario dado de la memoria V de destino, dándole oportunidad de que otros IBoxes de ECOM100 RX y ECOM100 WX en ése ECOM100 # sean ejecutados.

Por ejemplo, si usted desea leer y escribir datos continuamente a partir de 5 esclavos diferentes, usted puede tener todas estas instrucciones de ECRX y de ECWX en UN RENGLON controlado con SP1 (siempre encendido).Se ejecutarán secuencialmente, automáticamente.

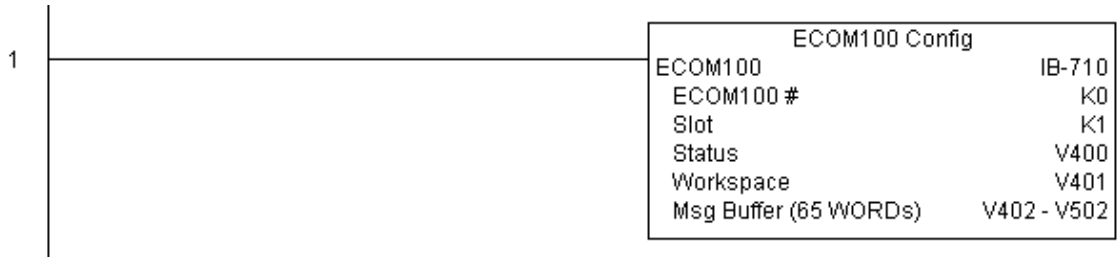
Parámetros de ECRX

- **ECOM100#**: Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Slave ID**: Especifica el PLC esclavo que será apuntado por la instrucción de ECRX
- **From Slave Element (Src)**: Especifica la dirección de esclavo de donde se leerán los datos
- **Number of Bytes**: Especifica el número de bytes a leer en el PLC esclavo
- **To Master Element (Dest)**: Especifica la localización en donde serán colocados los datos del esclavo en el PLC maestro con el ECOM100
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
ECOM100# ..... K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Slave ID ..... K	K0-90
From Slave Element (Src) X,Y,C,S,T,CT,GX,GY,V	Vea el mapa de memoria DL06
Number of Bytes ..... K	K1-128
To Master Element (Dest) V	Vea el mapa de memoria V del DL06 - Data Words
Success ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de ECRX

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todos los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



(Este ejemplo continúa en la próxima página)

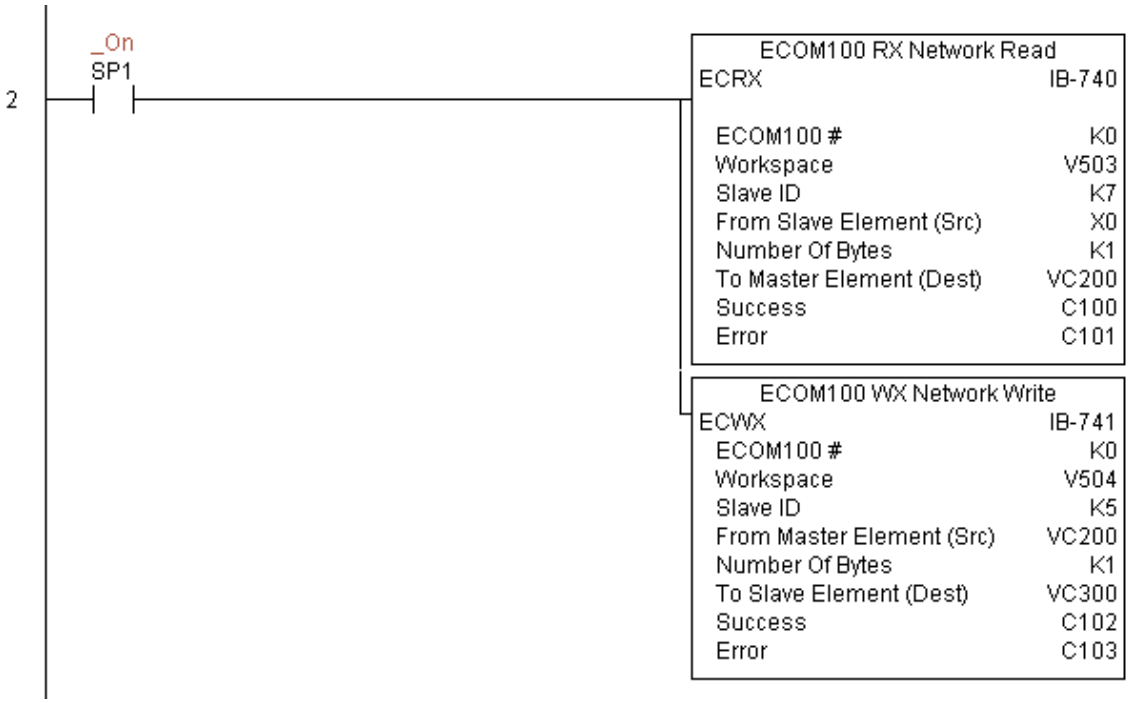
Ejemplo de ECRX (continuado)

Renglón 2: Usando ECOM100 # K0, lea X0-X7 del esclavo K7 y escribalo al esclavo K5 tan rápidamente como sea posible.Almacénelos en este PLC local en C200-C207, y escribalo a C300-C307 en el esclavo K5.

Las instrucciones ECRX y ECWX trabajan con los IBoxes para simplificar todo el establecimiento de una red haciendo los enclavamientos y distribuyendo los recursos apropiados. También suministran un reportaje de errores muy simplificada. Usted no tiene que preocuparse de ningun relevador especial SP116, por ejemplo, Busy u "ocupado" o los "bits de error", o en qué número de ranura está colocado un módulo, o tener contadores o shift registers o cualquiera otro enclavamiento para la administración de recursos.

En este ejemplo, SP1 (siempre ON) está conduciendo los IBoxes ECRX y ECWX en el mismo renglón. En el barrido que termina RX, la instrucción WX comenzará en ese mismo barrido. Tan pronto como la instrucción WX termine, cualquier operación pendiente debajo de ella en el programa conseguirá una chance the ejecutarse. Si no hay IBoxes de ECOM100 pendientes debajo del ECWX, el ECRX comenzaría su petición otra vezen el pxóximo barrido.

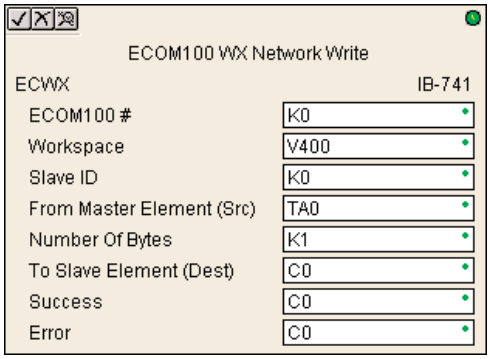
Usando el ECRX y el ECWX para todos las lecturas y escrituras en la red es el método más rápido que el PLC puede hacer para establecimiento de una red. Para los puertos seriales locales, los módulos de DCM, o los módulos originales de ECOM, use las instrucciones IBoxes NETCFG y NETRX/NETWX.



Escribe datos WX con ECOM100 (ECWX) (IB-741)

DS5	Usado
HPP	N/A

Esta instrucción es la instrucción WX con enclavamiento incorporado con otros IBoxes de ECOM100 RX (ECRX) y ECOM100 WX (ECWX) en el programa para simplificar el establecimiento de una red de comunicaciones. Realizará la instrucción WX en la red especificada de módulos ECOM100, que corresponde a un IBox de configuración específica en la parte superior del programa



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Siempre que este IBox tenga energía, escribirá datos de elementos desde el maestro en el almacenador intermediario dado de memoria V al esclavo especificado comenzando con el elemento dado del esclavo, dándole oportunidad de que otros IBoxes de ECOM100 RX y ECOM100 WX en ése ECOM100 # sean ejecutados.

Por ejemplo, si usted desea leer y escribir datos continuamente a partir de 5 esclavos diferentes, usted puede tener todas estas instrucciones de ECRX y de ECWX en UN RENGLON controlado con SP1 (siempre encendido).Se ejecutarán secuencialmente, automáticamente.

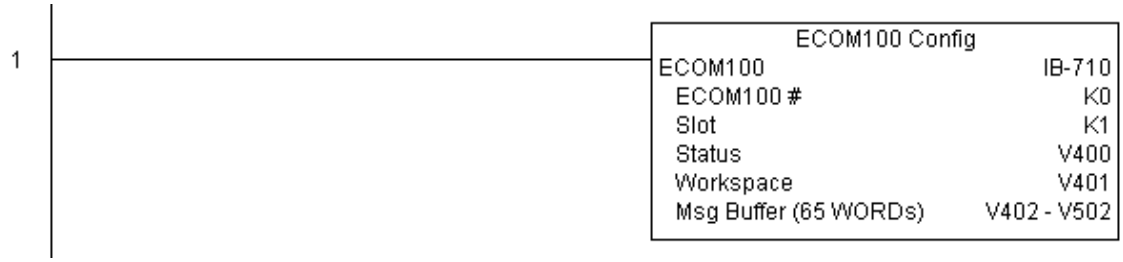
Parámetros de ECWX

- ECOM100#: Éste es un número lógico asociado a este módulo específico ECOM100 en la ranura especificada. El resto de los IBoxes ECxxxx que necesitan referirse a este módulo ECOM100 deben referirse a este número lógico
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Slave ID**: Especifica el PLC esclavo que será apuntado por la instrucción de ECWX
- **From Master Element (Src)**: Especifica una localización de memoria V en el PLC maestro con ECOM100 de donde será el origen de los datos
- **Number of Bytes:** Especifica la cantidad de bytes a serescritos al PLC esclavo con ECOM(100)
- **To Slave Element (Dest):** Especifica la dirección del esclavo donde serán escritos los datos
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
ECOM100# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Slave ID . . . . . K	K0-90
From Master Element (Src) V	Vea el mapa de memoria V del DL06 - Data Words
Number of Bytes . . . . . K	K1-128
To Slave Element (Dest) . . X,Y,C,S,T,CT,GX,GY,V	Vea el mapa de memoria DL06
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de ECWX

Renglón 1: Esta instrucción es responsable por la coordinación y enclavamiento de todos los tipos de IBoxes ECOM100 para un módulo específico ECOM100. Marque el ECOM100 con un rótulo en la ranura 1 como ECOM100K0. El resto de los IBoxes ECxxxx se refieren a este módulocomo K0. Si usted necesita cambiar el módulo en la base a una ranura diferente, se necesita solamente cambiar este IBox. V400 es usado como registro global de estado del resultado para otros IBoxes ECxxxx que usan este módulo específico ECOM100. V401 es usado para coordinar y enclavar la lógica en todo los otros IBoxes ECxxxx usando este módulo específico ECOM100. V402-V502 es un campo común almacenador intermediario de 130 bytes disponibles para uso por otros IBoxes ECxxxx usando este módulo específico ECOM100.



(Este ejemplo continúa en la próxima página)

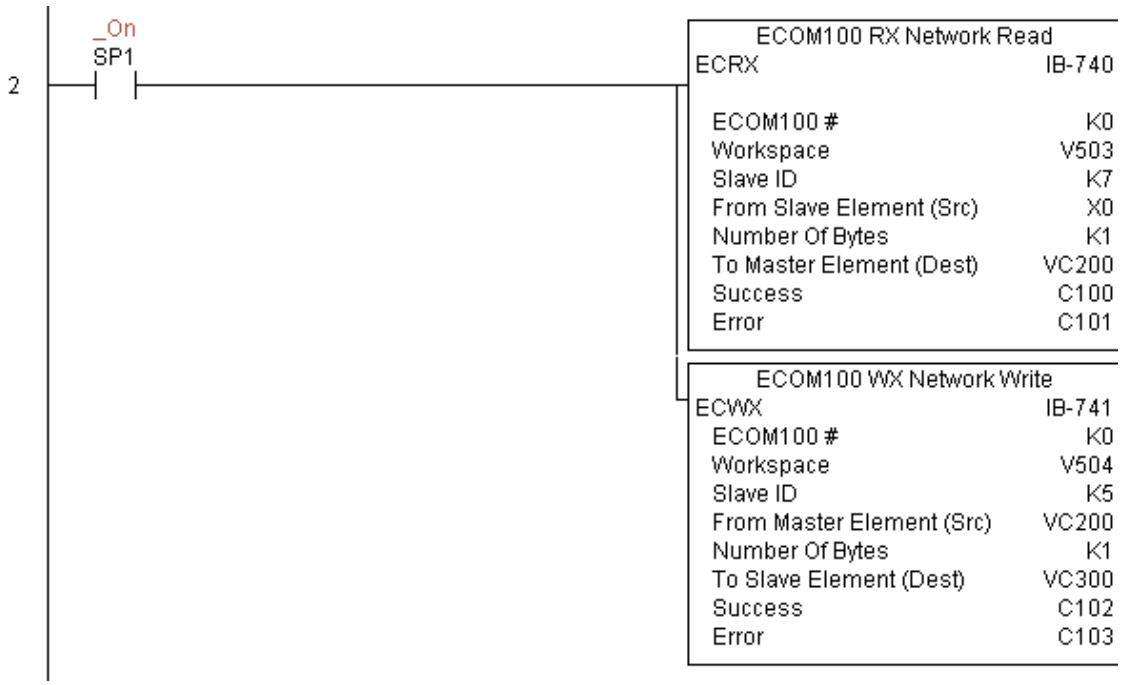
Ejemplo de ECWX (continuado)

Renglón 2: Usando ECOM100 # K0, lea X0-X7 del esclavo K7 y escribalo al esclavo K5 tan rápidamente como sea posible. Almacene los datos en este PLC local en C200-V207, y escribalos a C300-C307 en el esclavo K5.

Las instrucciones ECRX y ECWX trabajan con el IBox de configuración del ECOM100 para simplificar todo el establecimiento de una red administrando todo el enclavamiento y recursos apropiados. También suministran un reportaje de error muy simplificada. Usted no tiene que preocuparse de ningun SP "Busy" o "bits de error", o en qué número de ranura está un módulo, o tener algunos contadores o shift registers u otros enclavamientos para la administración de recursos.

En este ejemplo, el contacto SP1 (Siempre ON) está controlando las instruccciones IBoxes ECRX y ECWX en el mismo renglón. En el barrido cuando se termina la ejecución de la instruccion RX, la instrucción comenzará en el mismo barrido. Tan pronto como termine WX, cualquier operación pendiente debajo de ella en el programa tendrá la oprtunidad de poder ser ejecutada. Si no hay IBoxes ECOM100 pendientes debajo del ECWX, entonces en el próximo barrido la instrucción ECRX comenzaría su petición otra vez.

Usando el ECRX y el ECWX para toda sus lectturas y escrituras en la red de ECOM100 es lo más rápido que el PLC puede hacer para el establecimiento de una red. Para los puertos seriales locales, los módulos de DCM, o los módulos originales de ECOM, use los IBoxes NETCFG y NETRX/NETWX.

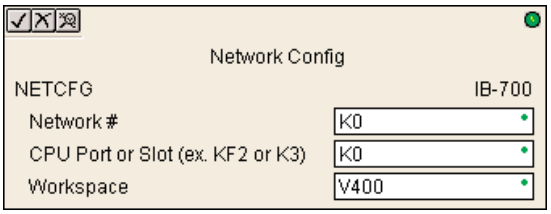


Configuración de una read NETCFG (NETCFG) (IB-700)

DS5	Usado
HPP	N/A

Esta instrucción define toda la información común necesaria para realizar establecimiento de una red con instrucciones RX/WX usando las instrucciones IBox NETRX y NETWX utilizando un puerto serial local de la CPU, el módulo D0-DCM o H0-ECOM.

Usted debe tener la instrucción de configuración de la red en la parte superior de su programa ladder o de etapas con cualquier otros IBoxes de configuración.



Si usted utiliza más que un puerto serial local, D0-DCM o H0-ECOM en su PLC para el establecimiento de una red RX/WX, usted debe tener una instrucción diferente de configuración de red para CADA red de RX/WX en su sistema que utilice alguna instrucción IBox del tipo NETRX/NETWX.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

El segundo parámetro "puerto o ranura de la CPU" es el mismo valor que en el byte mas significativo de la primera instrucción del LD si usted estuviera haciendo el programa de RX o de WX usted mismo. Este valor es específico de la CPU y del puerto, pero los valores posibles incluyen KF2 para el puerto serial local 2 de la CPU de 06, K3 para un DCM o un ECOM en la ranura 3 de un local 205 bajo, o de K37 para un DCM en una base 3, ranura 7 de 405 extensiones.

Parámetros de NETCFG

- **Network#:** Especifica un único número para cada red de ECOM(100) o de DCM a ser usado
- **CPU Port or Slot:** Especifica el número de acceso de la CPU o el número de la ranura DCM/ECOM(100) usado
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción

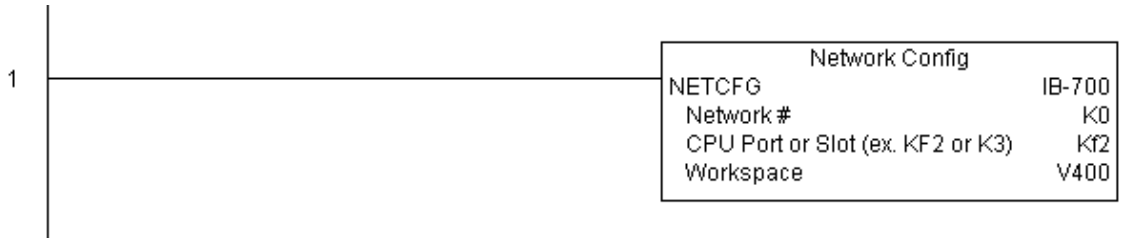
Parámetro	Rango del DL06
Network# . . . . . K	K0-255
CPU Port or Slot . . . . . K	K0-FF
Workspace V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de NETCFG

Esta instrucción coordina toda la interacción con otras instrucciones IBox (NETRX/NETWX) en al red. Usted debe tener un IBox de configuración de red para cada red de puerto serial, del módulo de DCM, o la red original del módulo de ECOM en su sistema. Las instrucciones IBox de configuración deben estar en la parte superior de su programa y debe ser ejecutadas en cada barrido.

Este IBox define la red # K0 estando ubicada en el puerto serial local #2 (KF2) de la CPU. Para los puertos seriales locales de la CPU o los módulos de DCM/ECOM, use el mismo valor que usted utilizaría en el byte más significativo de la primera instrucción del LD en un renglón normal de RX/WX de referirse al puerto o al módulo. Cualquiera de los IBoxes NETRX o NETWX que necesite referirse a esta red específica incorporaría K0 para el parámetro de número de red.

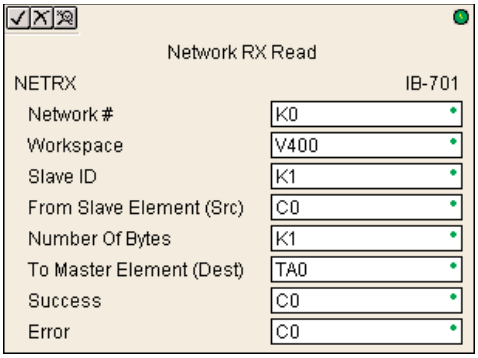
El registro del espacio de trabajo es usado para mantener la información del estado sobre el puerto o el módulo, junto con compartir apropiado y enclavamientos con el otros IBoxes del tipo NETRX y NETWX en el programa.Este registro de memoria de V no debe ser usado en cualquier otro lugar en el programa entero.



Leer la red RX (NETRX) (IB-701)

DS5	Usado
HPP	N/A

La instrucción RX lee datos en la red con enclavamiento incorporado con el resto de los IBoxes RX (NETRX) y WX (NETWX) en el programa ladder, para simplificar el establecimiento de una red de comunicación. Realizará el RX en el número de la red especificada, que corresponde a una configuración de red única específica (NETCFG) en la parte superior del programa ladder.



El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

Siempre que este IBox tenga energía, leerá datos del elemento del esclavo especificado en el almacenador intermediario dado de la memoria V de destino, dandole la oportunidad de ejecutar otros IBoxes RX y WX en ese número de red.

Por ejemplo, si usted desea leer y escribir datos continuamente a partir de 5 esclavos diferentes, usted puede tener todas estas instrucciones de NETRX y NETWX en UN RENGLON controlado con SP1 (siempre encendido).Se ejecutarán secuencialmente, automáticamente.

Parámetros de NETRX

- **Network#:** Especifica el número de red (puertos de la CPU, DCM, ECOM) definido por la instrucción NETCFG
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Slave ID:** Especifica el PLC esclavo que será interrogado por la instrucción
- **From Slave Element (Src):** Especifica la dirección del PLC esclavo de donde serán leídos los datos
- **Number of Bytes:** Especifica la cantidad de bytes a ser leídos desde el PLC esclavo
- **To Master Element (Dest):** Especifica la localización de memoria en el PLC maestro donde serán colocados los datos del esclavo
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

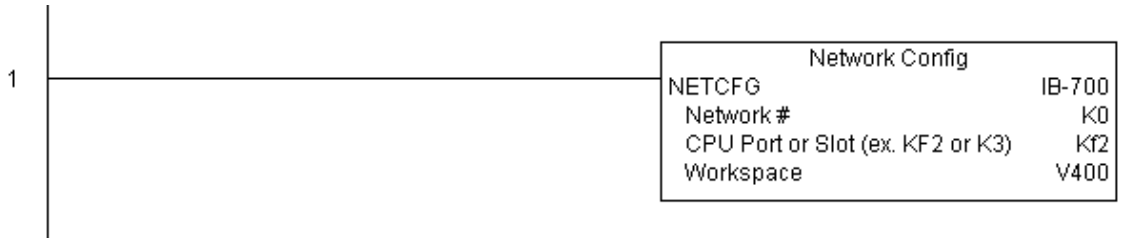
Parámetro	Rango del DL06
Network# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Slave ID . . . . . K	K0-90
From Slave Element (Src) X,Y,C,S,T,CT,GX,GY,V	Vea el mapa de memoria DL06
Number of Bytes . . . . . K	K1-128
To Master Element (Dest) V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de NETRX

Esta instrucción coordina toda la interacción con otras instrucciones IBox (NETRX/NETWX) en al red. Usted debe tener un IBox de configuración de red para cada red de puerto serial, del módulo de DCM, o la red original del módulo de ECOM en su sistema. Las instrucciones IBox de configuración deben estar en la parte superior de su programa y debe ser ejecutadas en cada barrido.

Este IBox define la red # K0 estando ubicada en el puerto serial local #2 (KF2) de la CPU. Para los puertos seriales locales de la CPU o los módulos de DCM/ECOM, use el mismo valor que usted utilizaría en el byte más significativo de la primera instrucción del LD en un renglón normal de RX/WX de referirse al puerto o al módulo. Cualquiera de los IBoxes NETRX o NETWX que necesite referirse a esta red específica incorporaría K0 para el parámetro de número de red.

El registro del espacio de trabajo es usado para mantener la información del estado sobre el puerto o el módulo, junto con compartir apropiado y enclavamientos con el otros IBoxes del tipo NETRX y NETWX en el programa.Este registro de memoria de V no debe ser usado en cualquier otro lugar en el programa entero.



(Este ejemplo continúa en la próxima página)

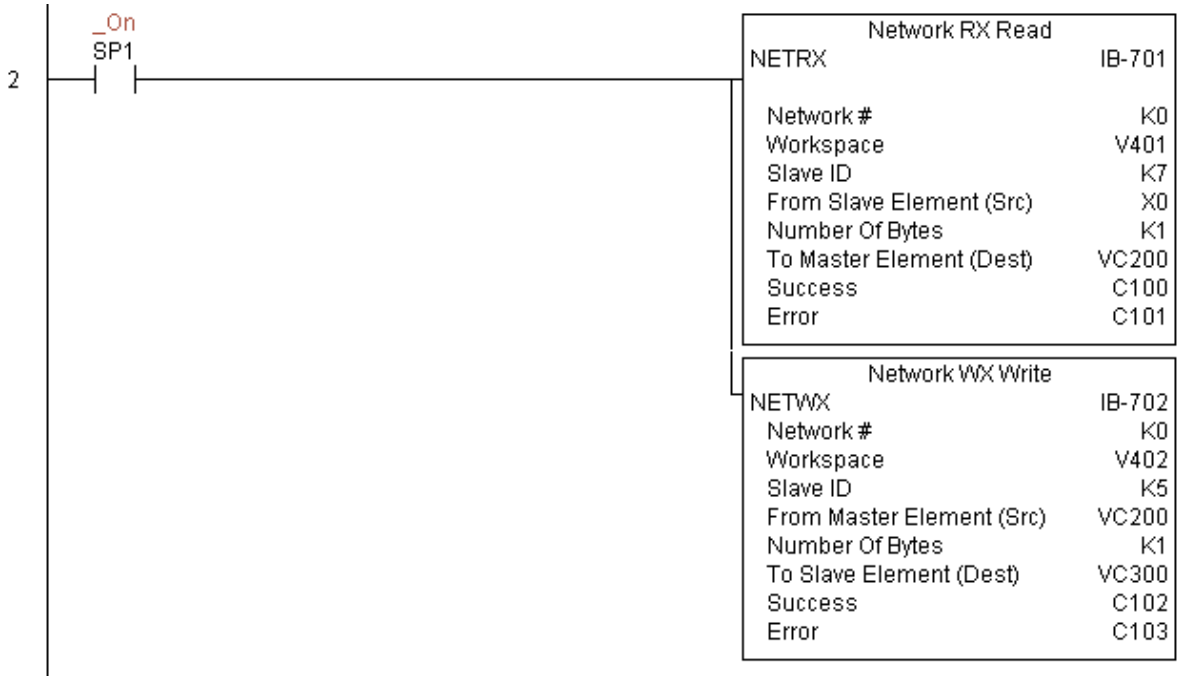
Ejemplo de NETRX (continuado)

Renglón 2: Using Network# K0, read X0-X7 from Slave K7 and write them to slave K5 as fast as possible. Store them in this local PLC in C200-C207, and write them to C300-C307 in slave K5.

Los IBoxes NETRX y NETWX trabajan con los IBox de configuración de la red para simplificar todo el establecimiento de una red administrando los enclavamientos y y recursos apropiados. También suministran un reportaje de error muy simplificado. Usted no necesita preocuparse de ningón SP "Busy bits" o "bits de error", o qué número de acceso o en que número de ranura está instalado un módulo, o tener contadores o shift register u otros enclavamientos para la admministración de recursos.

En este ejemplo, SP1 (siempre ON) está controlando los IBoxes NETRX y NETWX en el mismo renglón. En el barrido que la red leída termina, la red escribe comenzará que igual explora.¶Tan pronto como la se complete de ejecutar la instrucción RX, cualquier operación pendiente debajo de ella en el programa va a poder ser ejecutada. Si no hay IBoxes NETRX o NETWX pendientes debajo de este IBox, entonces en el próximo barrido el NETRX comenzaría su operación nuevamente.

Usando los IBoxes NETRX y el NETWX para todos los puertos seriales, para D0-DCM, o una red original de ECOM es la forma más rápida que el PLC puede hacer establecimiento de una red. Para los módulos ECOM100, use los IBoxes ECOM100 y ECRX/ECWX.

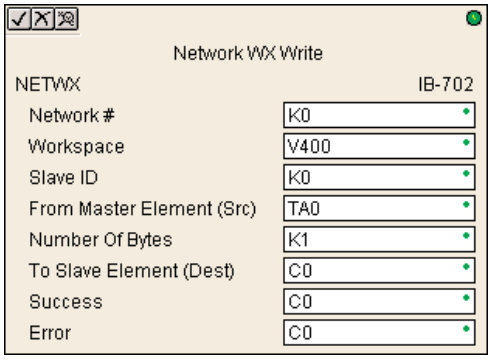


Escribir a la red WX (NETWX) (IB-702)

DS5	Usado
HPP	N/A

La instrucción WX escribe datos en la red con enclavamiento incorporado con el resto de los IBoxes RX (NETRX) y WX (NETWX) en el programa ladder, para simplificar el establecimiento de una red de comunicación. Realizará el RX en el número de la red especificada, que corresponde a una configuración de red única específica (NETCFG) en la parte superior del programa ladder.

El parámetro Workspace (espacio de trabajo) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.



Siempre que este IBox tenga energía, escribirá datos desde una memoria del PLC maestro especificado a la memoria V de destino en un esclavo dado, dándole la oportunidad de ejecutar otros IBoxes RX y WX en ese número de red.

Por ejemplo, si usted desea leer y escribir datos continuamente a partir de 5 esclavos diferentes, usted puede tener todas estas instrucciones de NETRX y NETWX en UN RENGLON controlado con SP1 (siempre encendido). Se ejecutarán secuencialmente, automáticamente.

Parámetros de NETWX

- **Network#:** Especifica el número de red (puertos de la CPU, DCM, ECOM) definido por la instrucción NETCFG
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Slave ID:** Especifica el PLC esclavo que será escrito por la instrucción
- **From Master Element (Src):** Especifica la localización del PLC maestro en donde los datos serán originados
- **Number of Bytes:** Especifica la cantidad de bytes a ser escritos al PLC esclavo
- **To Slave Element (Dest):** Especifica la dirección del esclavo en donde serán escritos los datos
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

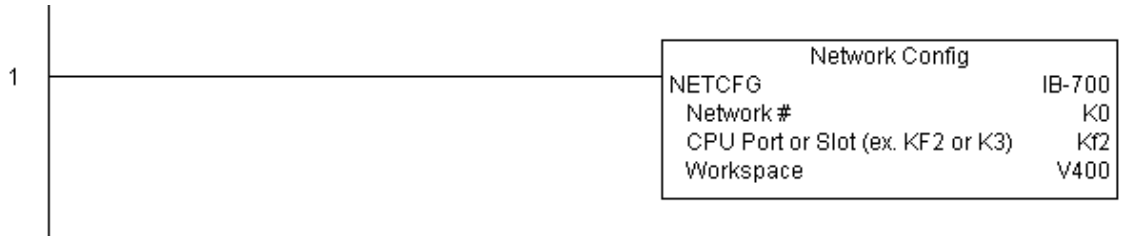
Parámetro	Rango del DL06
Network# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Slave ID . . . . . K	K0-90
From Master Element (Src) V	Vea el mapa de memoria V del DL06 - Data Words
Number of Bytes . . . . . K	K1-128
To Slave Element (Dest) . . X,Y,C,S,T,CT,GX,GY,V	Vea el mapa de memoria DL06
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de NETWX

Esta instrucción coordina toda la interacción con otras instrucciones IBox (NETRX/NETWX) en al red. Usted debe tener un IBox de configuración de red para cada red de puerto serial, del módulo de DCM, o la red original del módulo de ECOM en su sistema. Las instrucciones IBox de configuración deben estar en la parte superior de su programa y debe ser ejecutadas en cada barrido.

Este IBox define la red # K0 estando ubicada en el puerto serial local #2 (KF2) de la CPU. Para los puertos seriales locales de la CPU o los módulos de DCM/ECOM, use el mismo valor que usted utilizaría en el byte más significativo de la primera instrucción del LD en un renglón normal de RX/WX de referirse al puerto o al módulo. Cualquiera de los IBoxes NETRX o NETWX que necesite referirse a esta red específica incorporaría K0 para el parámetro de número de red.

El registro del espacio de trabajo es usado para mantener la información del estado sobre el puerto o el módulo, junto con compartir apropiado y enclavamientos con el otros IBoxes del tipo NETRX y NETWX en el programa.Este registro de memoria de V no debe ser usado en cualquier otro lugar en el programa entero.



(Este ejemplo continúa en la próxima página)

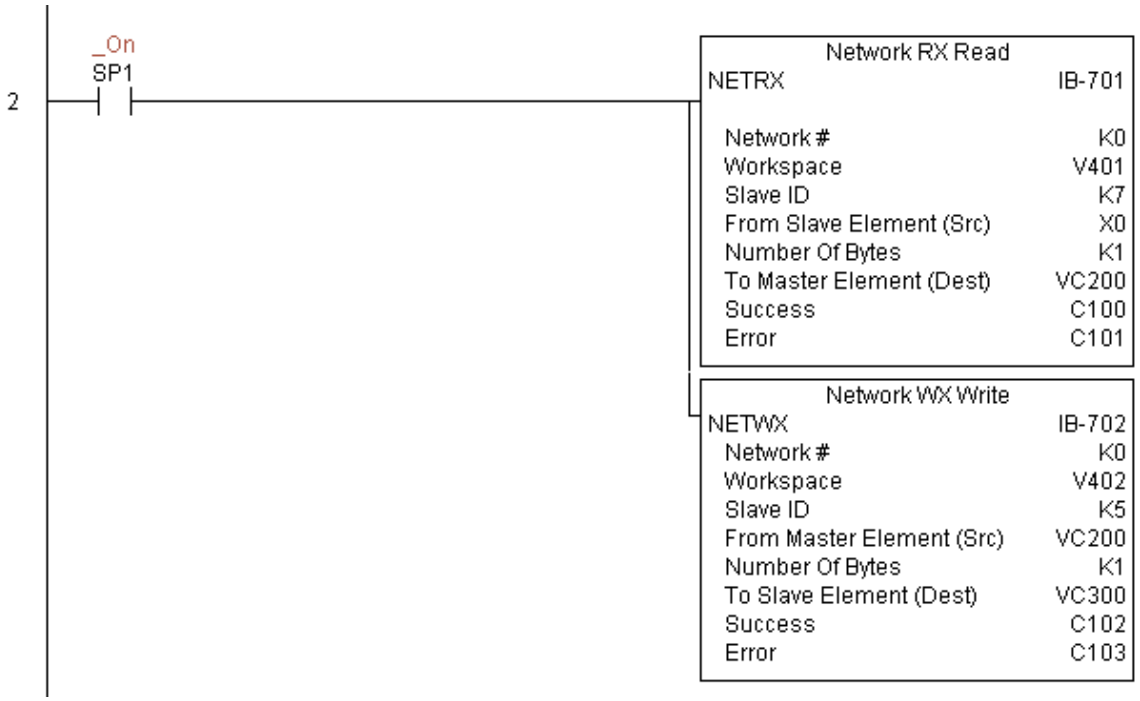
Ejemplo de NETWX (continuado)

Renglón 2: Con la red # K0, lea X0-X7 del esclavo K7 y escríbalo al esclavo K5 tan rápidamente como sea posible. Almacénelos en este PLC local en C200-C207, y escríbalos a C300-C307 en el esclavo K5.

Las instrucciones NETRX y NETWX trabajan con el IBox de configuración de la red para simplificar todo el establecimiento de una red controlados los enclavamientos y recursos apropiados. También suministran un reportaje de error muy simplificado.. Usted no necesita preocuparse de ningón SP "Busy bits" o "bits de error", o qué número de acceso o en que número de ranura está instalado un módulo, o tener contadores o shift register u otros enclavamientos para la admministración de recursos.

En este ejemplo, SP1 (siempre ON) está controlando los IBoxes NETRX y NETWX en el mismo renglón. En el mismo barrido en que termina la instrucción RX, comenzará la instrucción WX. Tan pronto como WX termine, cualquier operación pendiente debajo de ella en el programa podrá ser ejecutada. Si no hay IBoxes NETRX o NETWX pendientes debajo de este IBox, entonces la instrucción NETRX comenzaría su petición en el próximo barrido.

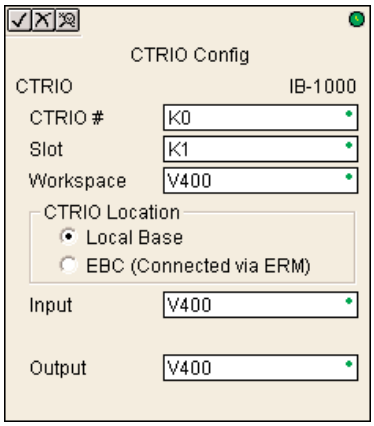
Usando los IBoxes NETRX y el NETWX para todos los puertos seriales, para D0-DCM, o una red original de ECOM es la forma más rápida que el PLC puede hacer establecimiento de una red. Para los módulos ECOM100, use los IBoxes ECOM100 y ECRX/ECWX.



Configuración de CTRIO (CTRIO) (IB-1000)

DS5	Usado
HPP	N/A

Esta instrucción define toda la información común para un módulo específico de CTRIO que sea usado por las otras instrucciones IBox de CTRIO (por ejemplo, CTRLDPR - cargar perfil deCTRIO, CTREDRL - CTRIO corregir y recargar una tabla de valores predefinidos, CTRRTLM - Modo Run to limit de CTRIO …).



Los parámetros de entradas-salidas para esta instrucción se pueden copiar directamente de la configuración del banco de trabajo de CTRIO para este módulo de CTRIO.

Usted debe tener los IBoxes de configuración de CTRIO en la parte superior de su programa ladder o de etapas junto con cualquier otros IBoxes de configuración.

Si usted tiene más de un CTRIO en su PLC, usted debe tener un IBox de CTRIO diferente para CADA módulo de CTRIO en su sistema que use alguna instrucción de IBox CTRIO . Cada IBox de configuración de CTRIO debe tener un ÚNICO valor de número de CTRIO. Ésto es cómo los IBoxes CTRIO se distinguen entre los diferente módulos de CTRIO en su sistema.

El parámetro del espacio de trabajo(Workspace) es un registro interno, privado usado por este IBox y DEBE SER ÚNICO en esta una instrucción y NO DEBE ser usado en cualquier otro lugar en el programa.

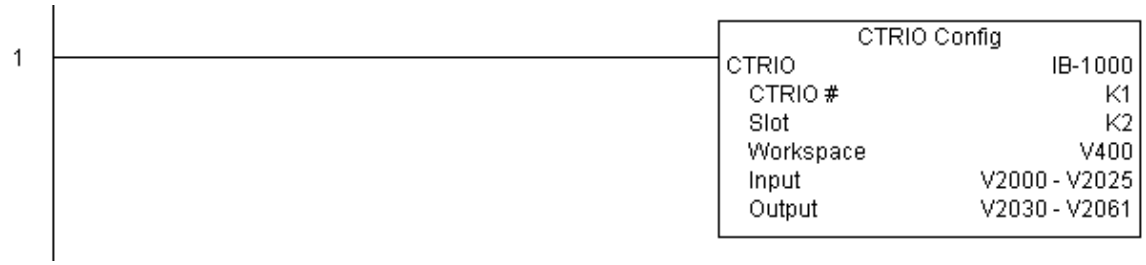
Parámetros de CTRIO

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario
- **Slot:** Especifica which PLC option slot the CTRIO module occupies
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **CTRIO Location:** Especifica donde se localiza el módulo (base local solamente para DL06)
- **Input:** Esto necesita ser configurado al mismo registro de memoria V que se especifica en **CTRIO Workbench** que dirección que comienza para las entradas, para este CTRIO único.
- **Output**: Esto necesita ser configurado al mismo registro de memoria V que se especifica en **CTRIO Workbench** como 'Starting V address for outputs' para este CTRIO único.

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Slot . . . . . K	K1-4
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Input V	Vea el mapa de memoria V del DL06 - Data Words
Output V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de CTRIO

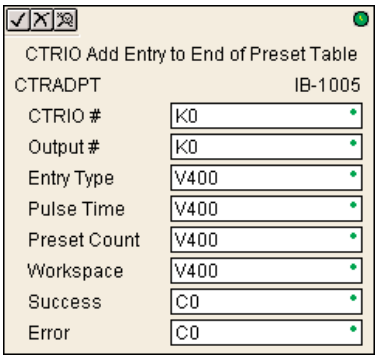
Renglón 1: Este ejemplo configura el módulo H0-CTRIO en la ranura 2 de la bnase del PLC DL06. Cada CTRIO en el sistema debe usar un I-box CTRIO diferente antes de que otros I Boxes CTRxxxx puedan ser usados. El módulo H0-CTRIO ha sido configurado para usa V2000 hasta V2025 en sus datos de entrada, y V2030 hasta V2061 for para sus datos de salidas.



Cree una tabla de valores predefinidos en CTRIO (CTRADPT) (IB-1005)

DS5	Usado
HPP	N/A

Esta instrucción añadirá una entrada al final de una tabla de valores predefinidos en un recurso específico de salida de CTRIO, en una transición de OFF para ON. Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede utilizar el IBox de CTRIO leer código de error (CTRRDER) para obtener para obtener información adicional del error.



Posibles tipos de entradas:

K0: Set

K1: Reset

K2: Pulse On (usa Pulse Time)

K3: Pulse Off (usa Pulse Time)

K4: Toggle

K5: Reset Count

Observe que el parámetro Pulse Time no tiene importancia para algunos tipos de entradas.

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

Parámetros de CTRAPT

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario (Vea CTRIO Config)
- **Output#**: Especifica una salidadel módulo H0-CTRIO a ser usada por la instrucción
- **Entry Type:** Especifica un tipo de entrada a ser agregado al finalde una tabla de valores predefinidos
- **Pulse Time:** Especifica un tiempo de un pulso para los tipos de entradas Pulse On yPulse Off
- **Preset Count:** Especifica un valor inicial de conteo en que comenzará después de un Reset
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Entry Type . . . . . V,K	K0-5; Vea el mapa de memoria V del DL06 - Data Words
Pulse Time . . . . . V,K	K0-65535; Vea el mapa de memoria V del DL06 - Data Words
Preset Count . . . . . V,K	K0-2147434528; Vea el mapa de memoria DL06
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

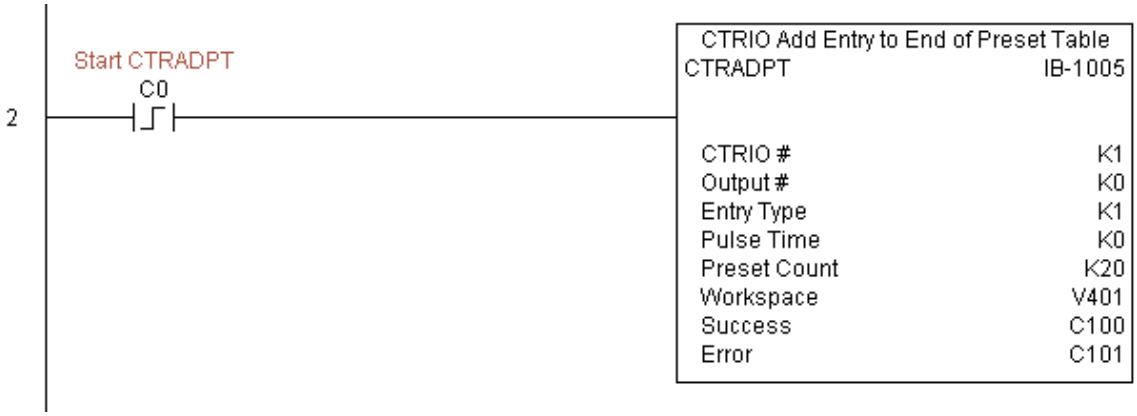
Ejemplo de CTRADPT

Renglón 1: Este ejemplo considera instalar el módulo H0-CTRIO en la ranura 2 del DL06. Cada H0-CTRIO en el sistema necesitará un IBox separada de CTRIO antes de que algún IBox de CTRxxxx pueda ser usado. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para los datos de entradas y V2030 hasta V2061 para sus datos de salidas.

Renglón 2: Este renglón es es un método de muestreo para permitir el comando de CTRADPT. Se usa un bit C para permitir que el programador controle un comando con Data View para propósitos de prueba.

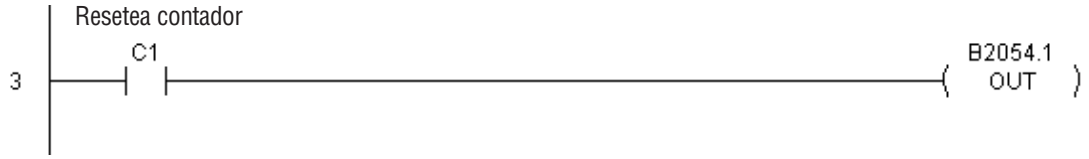
Al activar C0 causará que la instrucción CTRADPT agregue una nueva tabla de valores predefinidos en la salida número 0 en el H0-CTRIO en la ranura 2. El nuevo valor predefinido serán un comando de RESET (tipo de entrada K1=Reset), y el tiempo de pulso es dejado en cero pues el tipo del reset no utiliza esto, y la cuenta en la cual él se reseteará será 20.

El procedimiento de funcionamiento para este ejemplo es cargar el archivo de CTRADPT_ex1.cwb a su H0-CTRIO, luego introducir el código mostrado aquí, cambiar al modo RUN, permitir la salida número 0 activando el bit C2 en Data View, gire le encoder conectado al módulo H0-CTRIO a un valor superior a 10 conteos y la salida y el LED de salida 0 se encenderá y permanercerá así en todos los conteos que estén arriba de 10. Ahora resetee el contador con el bit C1, habilite C0 para executar la instrucción CTRADPT para dar un reset a la salida 0 cuando el conteo llegue a 20, encienda el bit C2 para activar la salida 0, y luego mueva el encoder encoder a un valor mas grande que 10+ (La salida 0 debe encenderse) y luego continúe hasta que pase de 20+ (la salida 0 debe apagarse).



Ejemplo de CTRADPT (continuado)

Renglón 3: Este renglón le permite al programador que resetee el contador desde *Direct*SOFT.

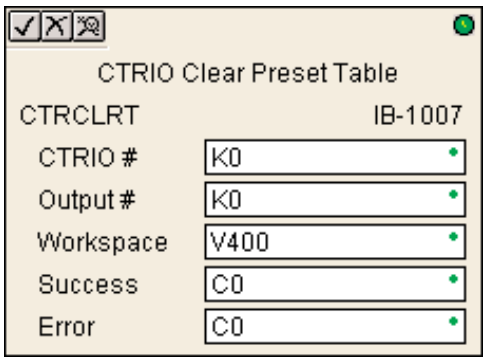


Renglón 4: Este renglón le permite al programador que habilite la salida 0 desde **Direct**SOFT.

Limpia la tabla de valores predefinidos de CTRIO (CTRCLRT) (IB-1007)

DS5	Usado
HPP	N/A

Esta instrucción coloca en cero una tabla de valores predefinidos en la memoria RAM del PLC, en una transición de OFF para ON. Este IBox tomará más de un barrido del PLC para ejecutarse.



El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede utilizar el código de error leído del IBox de CTRIO (CTRRDER) para obtener información adicional del error.

El registro Workspace (espacio de trabajo) es usado internamente y no debe ser usado en ningún otro lugar en el programa entero.

5

Parámetros de CTRCLRT

- CTRIO#: Especifica un módulo H0-CTRIO con un número definido por el usuario(Vea CTRIO Config)TRIO Config)
- **Output#:** Especifica una salida de un módulo H0-CTRIO a ser usado por la instrucción
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRCLRT

Renglón 1:Este ejemplo instala el módulo H0-CTRIO en la ranura 2 del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado algun otro IBox CTRxxxx. El módulo H0-CTRIO se ha configurado para utilizar V2000 hasta V2025 para sus datos de entradas y V2030 con V2061 para sus datos de salidas.

Renglón 2: Este renglón es un método para permitir el comando de CTRCLRT. Se usa un bit C para permitir que el programador controle el comando desde Data View para propósitos de prueba.

Activando C0 causará que la instrucción de CTRCLRT limpie la tabla de valores predefinidos en la salida 0 en el H0-CTRIO en la ranura 2 .

El procedimiento de funcionamiento para este ejemplo es cargar el archivo de CTRCLRT_ex1.cwb a su módulo H0-CTRIO, luego introducir el código mostrado aquí, cambio al modo RUN, habilitar la salida 0 activando el bit C2 en Data View, gire el encoder conectado al módulo H0-CTRIO para obtener un valor mas grade que 10 y el LED de la salida 0 se encenderá y permanecerá encendida hasta que el conteo llegue a sobre 20, y en ese momento se apagará. Luego resetee el contador con el bit C1, habilite el bit C0 para ejecutat la instrucción CTRCLRT para hacer cero la tabla de valores predefinidos, active el bit C2 para habilitar la salida 0, y luego gire el encoder a un valor de conteo sobre 10+ (La salida 0 NO DEBE encenderse).

Ejemplo de CTRCLRT (continuado)

Renglón 3: Este renglón le permite al programador que resetee el contador desde *Direct*SOFT.

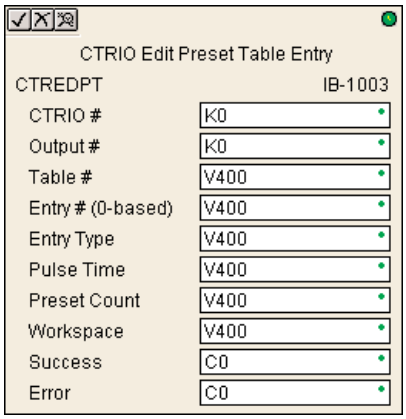
Renglón 4: Este renglón le permite al programador que habilite la salida 0 desde **Direct**SOFT.

Corregir una tabla de valores predefinidos de CTRIO (CTREDPT) (IB-1003)

DS5	Usado
HPP	N/A

Esta instrucción corrige, en una transición de APAGADO a ENCENDIDO a este IBox, una sola entrada de una tabla de valores predefinidos en un recurso específico de la salida de CTRIO. Este IBox es bueno si usted está corrigiendo más de una entrada en un archivo a la vez. Si usted desea hacer solamente una correción y después recargar la tabla inmediatamente, vea el IBox CTRIO CTREDRL. Este IBox tomará más de un barrido del PLC para ejecutarse.

El bit de éxito o de error se activará cuando el comando sea completado. Si el bit de error está encendido, usted puede usar el IBox código de error leído de CTRIO (CTRRDER) para conseguir una información más detallada del error.



Tipo de entrada:

K0: Set

K1: Reset

K2: Pulse On (usa Pulse Time)

K3: Pulse Off (usa Pulse Time)

K4: Toggle

K5: Reset Count

Observe que el parámetro Pulse Time es ignorado por algunos tipos de entradas.

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

Parámetros de CTREDPT

- **CTRIO#:** Especifica un módulo H0-CTRIO con un número definido por el usuario (Vea CTRIO Config)
- **Output#**: Especifica una salida del módulo H0-CTRIO a ser usado by the instrucción
- **Table#:** Especifica un número de tabla en la cual será modificada un dato
- **Entry#:** Especifica la localización del dato en la tabla de valores predefinidos a ser modificada
- **Entry Type**: Especifica un tipo de entrada a ser agregado durante la modificación
- **Pulse Time:** Especifica un pulse time para tipos de entradas Pulse On y Pulse Off
- **Preset Count:** Especifica un valor inicial de conteo para comenzar después de un Reset
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxitosuccessfully

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Table# . . . . . V,K	K0-255; Vea el mapa de memoria V del DL06 - Data Words
Entry# . . . . . V,K	K0-255; Vea el mapa de memoria V del DL06 - Data Words
Entry Type . . . . . V,K	K0-5; Vea el mapa de memoria V del DL06 - Data Words
Pulse Time . . . . . V,K	K0-65535; Vea el mapa de memoria V del DL06 - Data Words
Preset Count . . . . . V,K	K0-2147434528; Vea el mapa de memoria DL06
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTREDPT

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de quepueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas, y V2030 hasta V2061 para sus datos de salidas.

(Este ejemplo continúa en la próxima página)

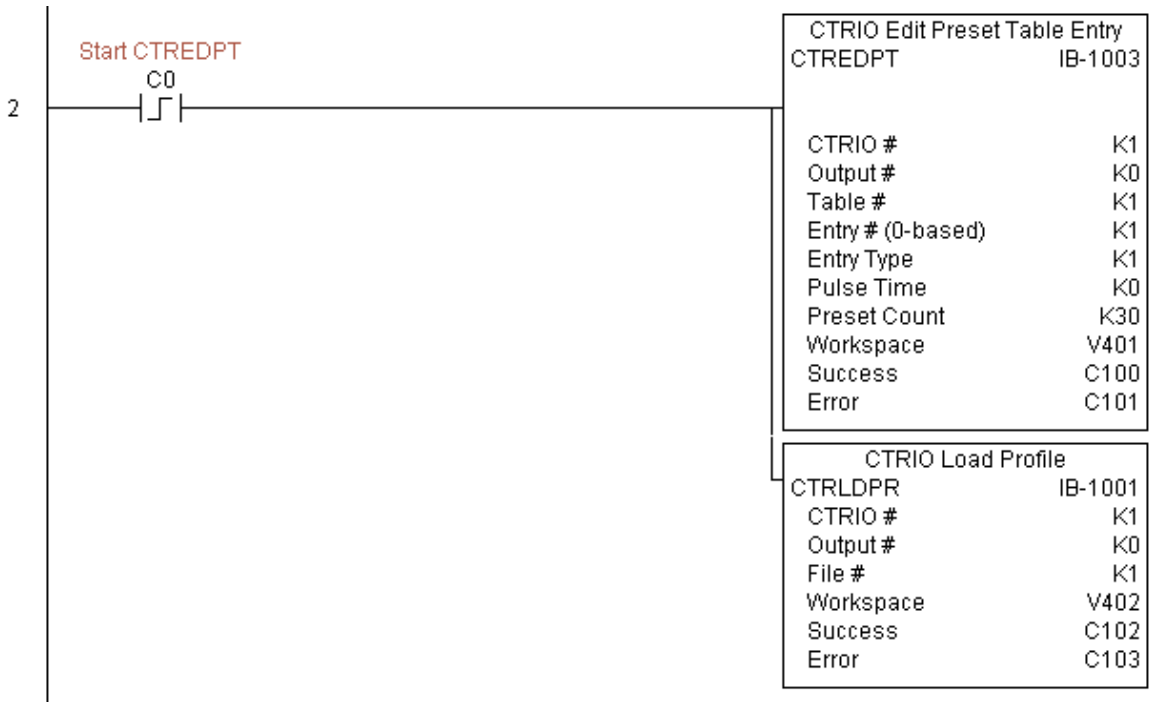
Ejemplo de CTREDPT (continuado)

Renglón 2: Este renglón es un método para permitir el comando de CTREDPT.¶ Se usa un bit Co para permitir que el programador controle el comando desde Data View para propósitos de prueba.

Al activar el bit C0 causará que la instrucción CTREDPT cambie el segundo valor predefinido a un reset de un conteo en 20 a un reset en una conteo en 30 para la salida 0 en el H0-CTRIO en la ranura 2.

El procedimiento de funcionamiento para este ejemplo es cargar el archivo de CTREDPT_ex1.cwb al H0-CTRIO, luego colocar el código mostrado aquí, cambiae el modo a RUN, permitir habilitar la salida 0 activando el bit C2 en Data View, luego gire el encoder conectado al módulo H0-CTRIO para obtener una valor sobre10 y el LED de la salida 0 se encenderá y permanecerá encendido hasta cuando se llegue a un valor sobre 20, cuando se apagará. Luego haga un reset del contador con el bit C1, habilite el bit C0 para ejecutar la instrucción CTREDPT command para cambiar el segundo valor predefinido, active el bit C2 para habiliar la salida 0, y luego gire el encoder a un valor sobre 10+ (La salida 0 debería activarse) y luego continúe mas arriba de un conteo de 30 (La salida 0 debería apagarse).

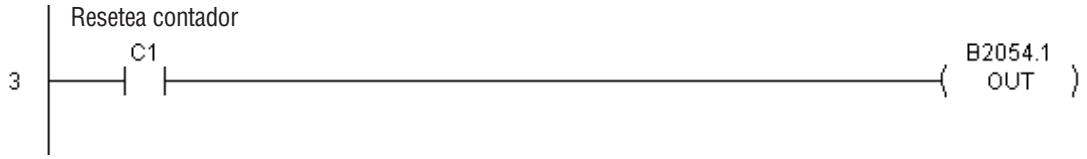
Observe que debemos también cargar el perfil después de cambiar el o los valores predefinidos, esta es la razón por la cual la instrucción CTRLDPR sigue la instrucción CTREDPT en este ejemplo.



(Este ejemplo continúa en la próxima página)

Ejemplo de CTREDPT(continuado)

Renglón 3: Este renglón le permite al programador que resetee el contador desde *Direct*SOFT.

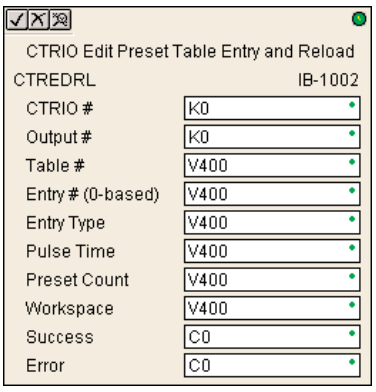


Renglón 4: Este renglón le permite al programador que habilite la salida 0 desde *Direct*SOFT.

Modificar una tabla de valores predefinidos de CTRIO (CTREDRL) (IB-1002)

DS5	Usado
HPP	N/A

Esta instrucción ejecutará la doble operación de modificar datos en una tabla de valores predefinidos y recargar auna salida de un módulo H0-CTRIO en una instrucción, en una transición de APAGADO a ENCENDIDO a este IBox. Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede utilizar el IBox de CTRIO leer código de error (CTRRDER) para obtener información adicional del error.



Tipo de entrada:

K0: Set

K1: Reset

K2: Pulse On (uses Pulse Time)

K3: Pulse Off (uses Pulse Time)

K4: Toggle

K5: Reset Count

Observe que el parámetro Pulse Time no tiene importancia para algunos tipos de entradas.

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

Parámetros de CTREDRL

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario (Vea CTRIO Config)
- **Output#**: Especifica una salida del módulo H0-CTRIO a ser usada por la instrucción
- **Table#:** Especifica un Table number of which an Entry is to be edited
- **Entry#**: Especifica un Entry location in the Preset Table to be edited
- **Entry Type**: Especifica el tipo de entrada a ser agregado durante la modificación
- **Pulse Time**: Especifica un tiempo del pulso para los tipos de entradas Pulse On y Pulse Off
- **Preset Count**: Especifica un valor inicial de conteo al comenzar después de un Reset
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Table# . . . . . V,K	K0-255; Vea el mapa de memoria V del DL06 - Data Words
Entry# . . . . . V,K	K0-255; Vea el mapa de memoria V del DL06 - Data Words
Entry Type . . . . . V,K	K0-5; Vea el mapa de memoria V del DL06 - Data Words
Pulse Time . . . . . V,K	K0-65535; Vea el mapa de memoria V del DL06 - Data Words
Preset Count . . . . . V,K	K0-2147434528; Vea el mapa de memoria DL06
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTREDRL

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.

(Este ejemplo continúa en la próxima página)

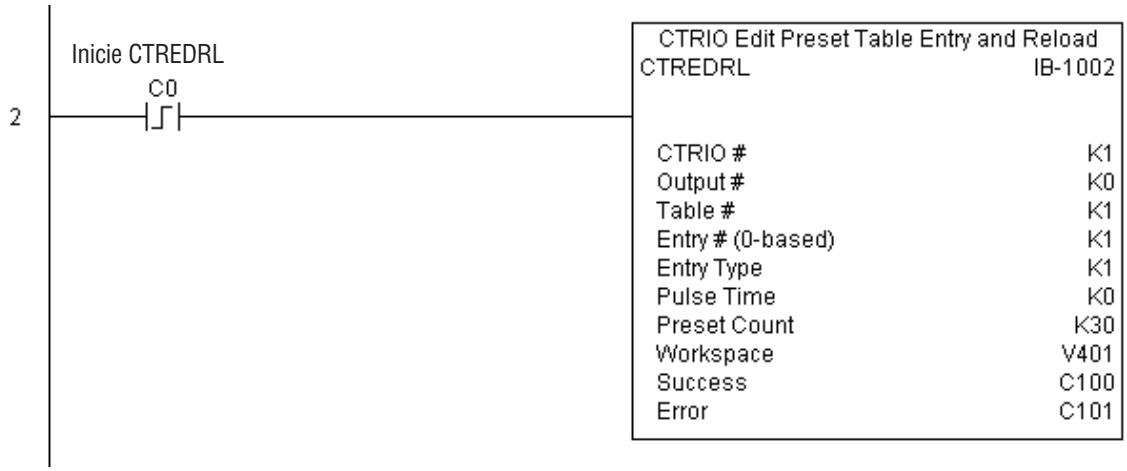
Ejemplo de CTREDRL (continuado)

Renglón 2: Este renglón es un método para permitir usar la instrucción CTREDRL. Se usa un bit C para permitir que el programador controle la instrucción desde Data View para propósitos de prueba.

Al activar el bit C0 causará que la instrucción CTREDRL cambie el segundo valor predefinido a un reset de un conteo en 20 a un reset en una conteo en 30 para la salida 0 en el H0-CTRIO en la ranura 2.

El procedimiento de funcionamiento para este ejemplo es cargar el archivo de CTREDRL_ex1.cwb al H0-CTRIO, luego colocar el código mostrado aquí, cambie el modo a RUN, luego habilite la salida 0 activando el bit C2 en Data View, luego gire el encoder conectado al módulo H0-CTRIO para obtener una valor sobre 10 y el LED de la salida 0 se encenderá y permanecerá encendido hasta cuando se llegue a un valor sobre 20, cuando se apagará. Luego haga un reset del contador con el bit C1, habilite el bit C0 para ejecutar la instrucción CTREDRL para cambiar el segundo valor predefinido a 30, y luego gire el encoder a un valor sobre 10+ (La salida 0 debería activarse) y luego continúe mas arriba de un conteo de 30 (La salida 0 debería apagarse).

Observe que no es necesario recargar el archivo separadamente. sin embargo, la instrucción puede cambiar un valor por vez.



(Este ejemplo continúa en la próxima página)

Ejemplo de CTREDRL (continuado)

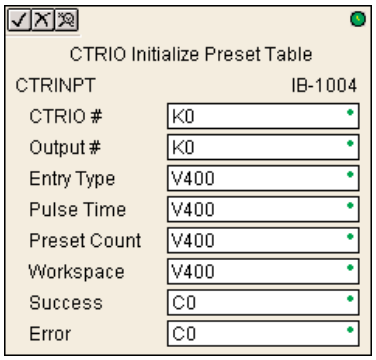
Renglón 3: Este renglón le permite al programador que resetee el contador desde *Direct*SOFT.

Renglón 4: Este renglón le permite al programador que habilite la salida 0 desde *Direct*SOFT.

Initializar una tabla de valores predefinidos de CTRIO (CTRINPT) (IB-1004)

DS5	Usado
HPP	N/A

Esta instrucción creará una Tabla de Valores Predefinidos en la memoria, pero como como archivo, en un recurso de salidas específica de CTRIO, en una transición de APAGADO a ENCENDIDO a este IBox.Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede utilizar el IBox de CTRIO leer código de error (CTRRDER) para obtener para obtener información adicional del error.



Tipo de entrada:

K0: Set

K1: Reset

K2: Pulse On (uses Pulse Time)

K3: Pulse Off (uses Pulse Time)

K4: Toggle

K5: Reset Count

Observe que el parámetro **Pulse Time** no tiene importancia para algunos tipos de entradas.

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

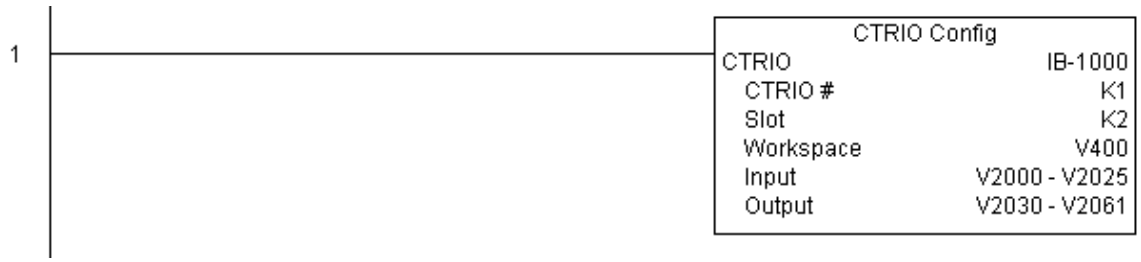
Parámetros de CTRINPT

- **CTRIO#**: Especifica un módulo CTRIO con un número definido por el usuario(Vea CTRIO Config)
- **Preset Count**: Especifica un valor inicial de conteo al comenzar después de un Reset
- **Output#**: Especifica una salida del módulo H0-CTRIO a ser usada por la instrucción
- **Entry Type**: Especifica el tipo de entrada a ser agregado durante la modificación
- **Pulse Time**: Especifica un tiempo del pulso para los tipos de entradas Pulse On y Pulse Off
- **Preset Count:** Especifica un valor inicial de conteo al comenzar después de un Reset
- **Workspace**: Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Entry Type . . . . . V,K	K0-5; Vea el mapa de memoria V del DL06 - Data Words
Pulse Time . . . . . V,K	K0-65535; Vea el mapa de memoria V del DL06 - Data Words
Preset Count . . . . . V,K	K0-2147434528; Vea el mapa de memoria DL06
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRINPT

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.



(Este ejemplo continúa en la próxima página)

Ejemplo de CTRINPT(continuado)

Renglón 2: Este renglón es un método para permitir usar la instrucción CTRINPT. Se usa un bit C para permitir que el programador controle la instrucción desde Data View para propósitos de prueba.

Al activar el bit C0 causará que la instrucción CTRINPT cree una tabla de valores predefinidos pero no como archivo y la usará con la salida 0. En este caso, el valor predefinido será un SET cuando el conteo llegue a 15 para la salida 0.

El procedimiento de funcionamiento para este ejemplo es cargar el archivo de CTRINPT_ex1.cwb al módulo H0-CTRIO, luego colocar el código mostrado aquí, cambie el modo a RUN, luego habilite la salida 0 activando el bit C2 en Data View, luego gire el encoder conectado al módulo H0-CTRIO para obtener una valor sobre 15 y el LED de la salida 0 no se encenderá. Luego haga un reset del contador con el bit C1, habilite el bit C0 para ejecutar la instrucción CTRINPT para crear un único valor predefinido para hacer un set an contar 15, y luego gire el encoder a un valor sobre 15+ (La salida 0 debería activarse).

Observe que no es necesario recargar el archivo separadamente. Sin embargo, la instrucción puede cambiar un valor por vez.

(Este ejemplo continúa en la próxima página)

Ejemplo de CTRINPT (continuado)

Renglón 3: Este renglón le permite al programador que resetee el contador desde *Direct*SOFT.

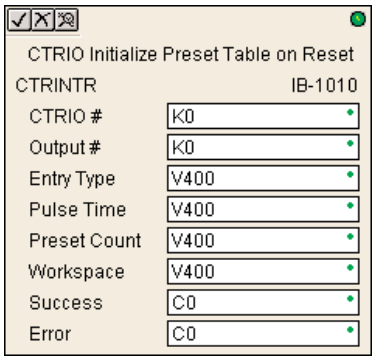
Renglón 4: Este renglón le permite al programador que habilite la salida 0 desde *Direct*SOFT.

Inicializar una tabla de valores predefinidos en CTRIO (CTRINTR) (IB-1010)

DS5	Usado
HPP	N/A

Esta instrucción creará una sola entrada en la tabla de valores predefinidos en memoria pero no como archivo, en una transición de APAGADO a ENCENDIDO a este IBox.

Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede utilizar el IBox de CTRIO leer código de error (CTRRDER) para obtener para obtener información adicional del error.



Entry Type:

K0: Set

K1: Reset

K2: Pulse On (uses Pulse Time)

K3: Pulse Off (uses Pulse Time)

K4: Toggle

K5: Reset Count

Observe que el parámetro **Pulse Time** no tiene importancia para algunos tipos de entradas..

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

Parámetros de CTRINTR

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario(Vea CTRIO Config)
- **Output#**: Especifica una salida del módulo H0-CTRIO a ser usada por la instrucción
- **Entry Type**: Especifica el tipo de entrada a ser agregado durante la modificación
- **Pulse Time:** Especifica un tiempo del pulso para los tipos de entradas Pulse On y Pulse Off
- **Preset Count:** Especifica un valor inicial de conteo al comenzar después de un Reset
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Entry Type . . . . . V,K	K0-5; Vea el mapa de memoria V del DL06 - Data Words
Pulse Time . . . . . V,K	K0-65535; Vea el mapa de memoria V del DL06 - Data Words
Preset Count . . . . . V,K	K0-2147434528; Vea el mapa de memoria DL06
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRINTR

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.

(Este ejemplo continúa en la próxima página)

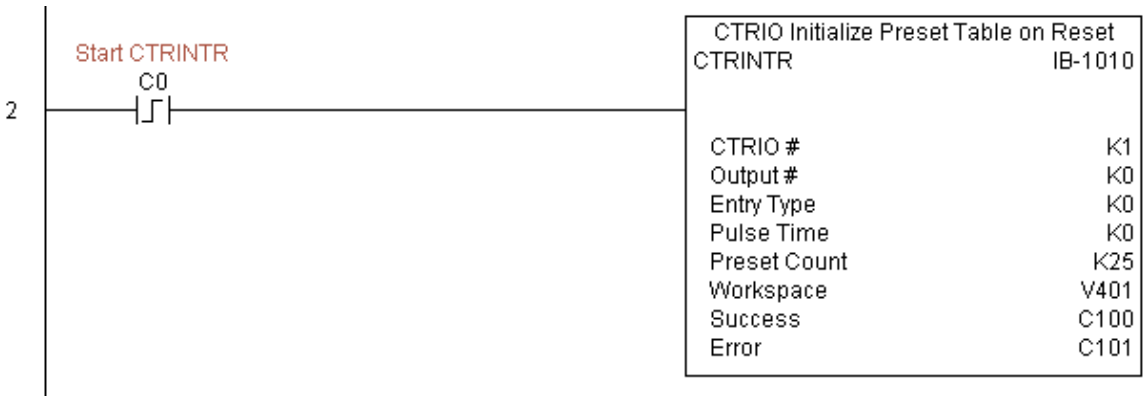
Ejemplo de CTRINTR(continuado)

Renglón 2: Este renglón es un método para permitir usar la instrucción CTRINTR. Se usa un bit C para permitir que el programador controle la instrucción desde Data View para propósitos de prueba.

Al activar el bit C0 causará que la instrucción CTRINTR cree una tabla única de valores predefinidos pero no como archivo y la usará con la salida 0. En este caso, el valor predefinido será un RESET cuando el conteo llegue a 25 para la salida 0.

El procedimiento de funcionamiento para este ejemplo es cargar el archivo de CTRINTR_ex1.cwb al módulo H0-CTRIO, luego coloque el código mostrado aquí, cambie el modo a RUN, luego habilite la salida 0 activando el bit C2 en Data View, luego gire el encoder conectado al módulo H0-CTRIO para obtener una valor sobre 1o y el LED de la salida 0 se encenderá. Luego active el bit C0 para ejecutar la instrucción CTRINTR y luego gire el encoder a un valor sobre 25+ (La salida 0 debería activarse).

Observe que no es necesario recargar el archivo separadamente. Sin embargo, la instrucción puede cambiar un valor por vez.



(Este ejemplo continúa en la próxima página)

Ejemplo de CTRINTR (continuado)

Renglón 3: Este renglón le permite al programador que resetee el contador desde *Direct*SOFT.

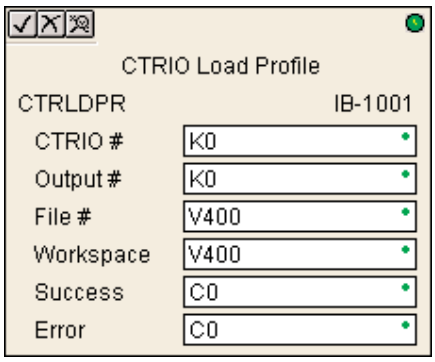
Renglón 4: Este renglón le permite al programador que habilite la salida 0 desde *Direct*SOFT.

Cargar un perfil en CTRIO (CTRLDPR) (IB-1001)

DS5	Usado
HPP	N/A

Esta instrucción carga un archivo de perfil de un módulo H0-CTRIO an un CTRIO Output resource en una transición desde OFF para ON a este IBox.

Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede utilizar el IBox de CTRIO leer código de error (CTRRDER) para obtener para obtener información adicional del error



El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

5

Parámetros de CTRLDPR

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario(Vea CTRIO Config)
- **Output#**: Especifica una salioda de CTRIO a ser usada por la instrucción
- **File#:** Especifica un número de archivo de CTRIO a ser cargado
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# ..... K	K0-255
Output# ..... K	K0-3
File# ..... V,K	K0-255; Vea el mapa de memoria V del DL06 - Data Words
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error ..... X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRLDPR

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.

Renglón 2: Este IBox de carga el perfil de CTRIO cargará el archivo 1 en la memoria de trabajo de la salida 0 en el módulo H0-CTRIO 1. Este programa de ejemplo requiere que Ud. cargue CTRLDPR_IBox.cwb en su módulo H0-CTRIO.

(Este ejemplo continúa en la próxima página)

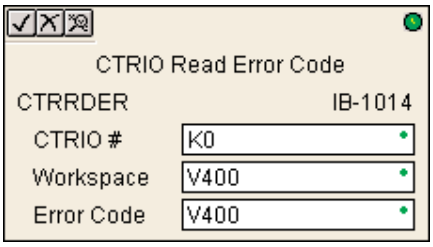
Ejemplo de CTRLDPR(continuado)

Renglón 3: Si el archivo es cargado con éxito, active el bit C1.

Lea error en CTRIO (CTRRDER) (IB-1014)

DS5	Usado
HPP	N/A

Esta instrucción obtendrá el valor de código decimal de error del módulo CTRIO (enumerado abajo) y lo pondrá en el registro dado del código de error, en una transición de APAGADO a ENCENDIDO al IBox.



Ya que el código de error en el CTRIO se mantiene solamente hasta que se da otro comando de CTRIO, usted debe utilizar esta instrucción inmediatamente después del IBox de CTRIO que entrega un error con el parámetro del bit de error.

El registro Workspace (espacio de trabajo) es para uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

Códigos de error:

0: No hay error

100: El código de comando especificado está desconocido o sin apoyo

101: Número de archivo no encontrado en el sistema de archivos

102: El tipo del archivo es incorrecto para la función de salida especificada

103: El tipo del perfil es desconocido

104: La entrada especificada no se configura como límite en esta salida

105: El borde especificado de la entrada del límite está fuera de rango

106: La función de entrada especificada no está configurada o es inválida

107: El número especificado de la función de entrada está fuera de rango

108: La función Especificada de valor predefinida es inválida

109: La tabla de valores prefedinidos está llena

110: La entrada especificada de la tabla está fuera de rango

111: El número especificado del registro está fuera de rango

112: El registro especificado es una entrada o salida que no está configurada

2001: Código de error de lectura de error - no puede tener acceso a CTRIO a través de ERM

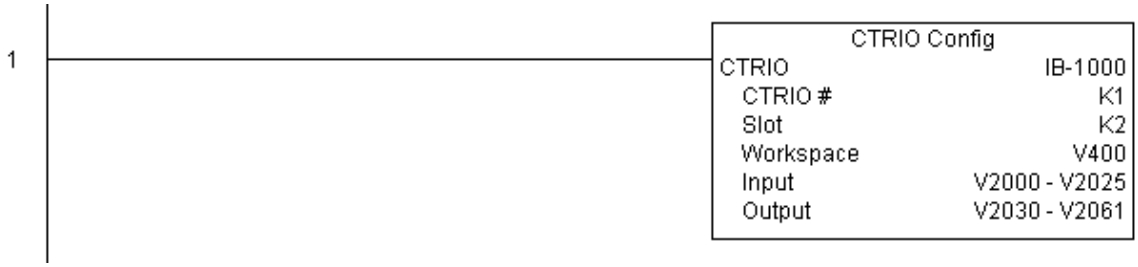
Parámetros de CTRRDER

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario(Vea CTRIO Config)
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Error Code:** Especifica la localización en donde será escrito el código de error

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Error Code V	Vea el mapa de memoria V del DL06 - Data Words

Ejemplo de CTRRDER

Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.

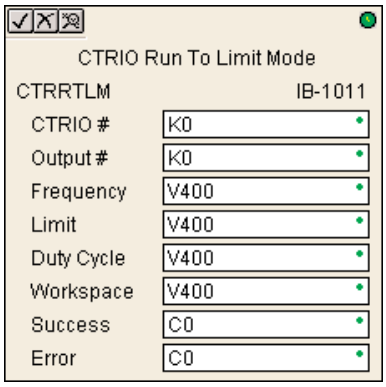


Renglón 2: Este IBox que lee el código de error desde CTRIO leerá información detallada del error del módulo CTRIO número 1. Este programa ejemplo requiere que usted cargue CTRRDER_IBox.cwb en el módulo H0-CTRIO.

Modo Run to Limit del CTRIO (CTRRTLM) (IB-1011)

DS5	Usado
HPP	N/A

Esta instrucción carga el comando de RUN to Limit y los parámetros dados en un recurso específico de la salida en una transición de APAGADO a ENCENDIDO al IBox. Las entradas de CTRIO se deben configurar como Limit(s) para que esta función trabaje.



Valores Límites Hexadecimales Válidos:

K00 - Borde de subida de Ch1/C

K10 - Borde de caída de Ch1/C

K20 - Ambos bordes de Ch1/C

K01 - Borde de subida de Ch1/D

K11 - Borde de caída de Ch1/D

K21 - Ambos bordes de Ch1/D

K02 - Borde de subida de Ch2/C

K12 - Borde de caída de Ch2/C

K22 - Ambos bordes de Ch2/C

K03 - Borde de subida de h2/D

K13 - Borde de caída de Ch2/D

K23 - Ambos bordes de Ch2/D

Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede usar el IBox de leer el código de error de CTRIO (CTRRDER) para obtener información ós detallada del error.

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa. be usado anywhere else in your program.

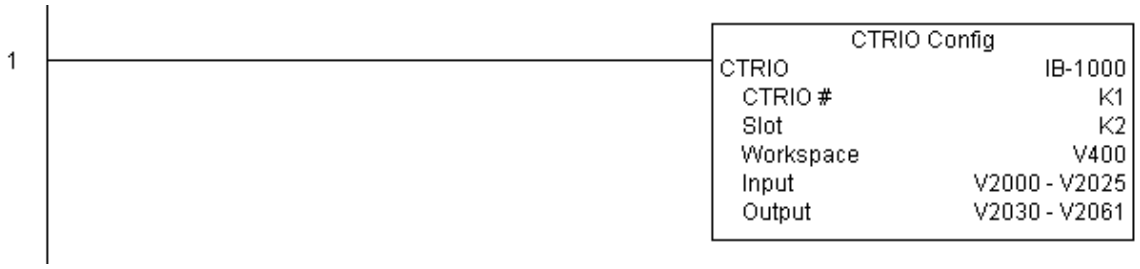
Parámetros de CTRRTLM

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario(Vea CTRIO Config)
- **Output#:** Especifica la salida del móduloH0-CTRIO a ser usada por la instrucción
- **Frequency**: Especifica la frecuencia de pulsos de la salida (20Hz - 20KHz)
- **Limit:** Las entradas del H0-CTRIO se deben configurar como Limites para que esta función funcione
- **Duty Cycle**: Especifica el % de tiempo On en el ciclo completo. Esto es un número hexadecimal. El valor por defecto de 0 es el 50%, también entrando 50 resultará en 50%. 50% se define como mitad del tiempo ON y mitad del tiempo OFF
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Frequency . . . . . V,K	K20-20000; Vea el mapa de memoria V del DL06 - Data Words
Limit . . . . . V,K	K0-FF; Vea el mapa de memoria V del DL06 - Data Words
Duty Cycle . . . . . V,K	K0-99; Vea el mapa de memoria V del DL06 - Data Words
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRRTLM

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.



Renglón 2: Este IBox de CTRIO instala en la salida 0 en el módulo H0-CTRIO número 1 el hacer salir pulsos en una frecuencia de 1000 Hertz hasta que se llegue al Límite 0. Este programa ejemplo requiere que usted cargue CTRRTLM_IBox.cwb en su módulo Ho-CTRIO.

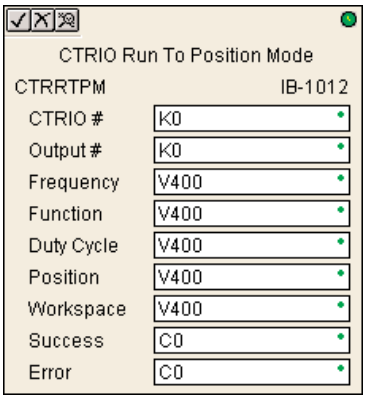
Ejemplo de CTRRTLM (continuado)

Renglón 3: Si los parámetros del modo Run to Limit es ACEPTABLE, active los bits de dirección y de habilitar la salida.

Modo Run to Position del CTRIO (CTRRTPM) (IB-1012)

DS5	Usado
HPP	N/A

Esta instrucción carga el comando Run to Position para colocar el comando y los parámetros dados en un recurso específico de la salida en una transición de APAGADO a ENCENDIDO a este IBox.



Los valores válidos de la función son:

00: Menor que Ch1/Fn1

10: Más grande que Ch1/Fn1

01: Menor que Ch1/Fn2

11: Más grande que Ch1/Fn2

02: Menor que Ch2/Fn1

12: Más grande que Ch2/Fn1

03: Menor que Ch2/Fn2

13: Más grande que Ch2/Fn2

Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando sa haya completado. Si el bit de error está encendido, usted puede usar el IBox de leer el código de error de CTRIO (CTRRDER) para obtener información más detallada del error.

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa. be usado anywhere else in your program.

Parámetros de CTRRTPM

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario(Vea CTRIO Config)TRIO Config Ibox)
- **Output#:** Especifica la salida del móduloH0-CTRIO a ser usada por la instrucción
- **Frequency**: Especifica la frecuencia de pulsos de la salida (20Hz - 20KHz)
- **Duty Cycle**: Especifica el % de tiempo On en el ciclo completo. Esto es un número hexadecimal. El valor por defecto de 0 es el 50%, también entrando 50 resultará en 50%. 50% se define como mitad del tiempo ON y mitad del tiempo OFF
- **Position:** Especifica el valor de conteo, según lo medido en la entrada del codificador, en el momento en el cual el tren de pulsos de salida será apagado
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Frequency . . . . . V,K	K20-20000; Vea el mapa de memoria V del DL06 - Data Words
Duty Cycle . . . . . V,K	K0-99; Vea el mapa de memoria DL06
Position . . . . . V,K	K0-2147434528; Vea el mapa de memoria DL06
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRRTPM

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.

Ejemplo de CTRRTPM (continuado)

Renglón 2: Este IBox instala la salida número 0 en el CTRIO 1 para hacer salir pulsos en una frecuencia de 1000 Hertz, usa la comparación ' mayor que Ch1/Fn1', hasta que se alcanza la posición 1500 en la entrada. Este programa ejemplo requiere que usted cargue CTRRTPM_IBox.cwb en su módulo H0-CTRIO.

Renglón 3: Si los parámetros del modo Run to Postion sons ACEPTABLES, activa los bits de dirección y de habilitar la salida.¶.

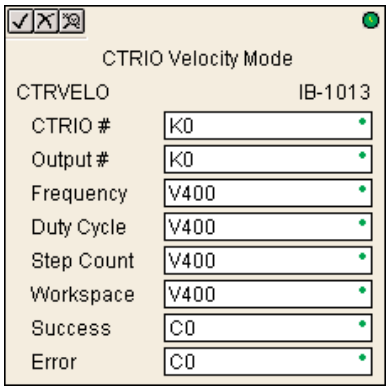
Modo de Velocidad de CTRIO (CTRVELO) (IB-1013)

DS5	Usado
HPP	N/A

Esta instrucción carga el comando de velocidad para colocar el comando y los parámetros dados en un recurso específico de la salida en una transición de APAGADO a ENCENDIDO a este IBox.

Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando el comando se haya completado. Si el bit de error está encendido, usted puede utilizar el IBox de CTRIO leer código de error (CTRRDER) para obtener para obtener información adicional del error.

El registro Workspace (espacio de trabajo) es para el uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.



5

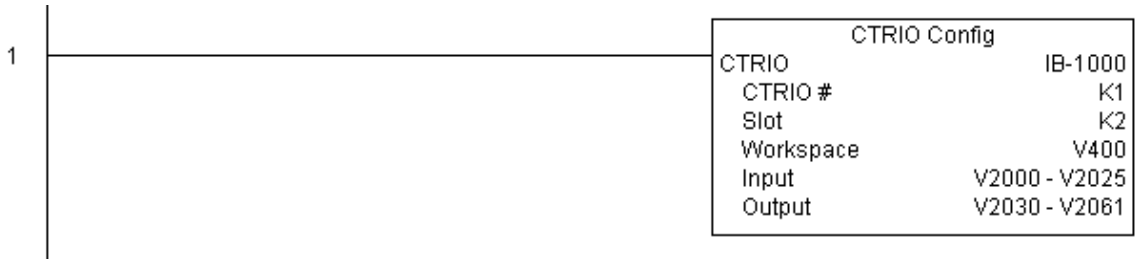
Parámetros de CTRVELO

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario(Vea CTRIO Config)TRIO Config Ibox)
- **Output#:** Especifica la salida del móduloH0-CTRIO a ser usada por la instrucción
- **Frequency**: Especifica la frecuencia de pulsos de la salida (20Hz - 20KHz)
- **Duty Cycle**: Especifica el % de tiempo On en el ciclo completo. Esto es un número hexadecimal. El valor por defecto de 0 es el 50%, también entrando 50 resultará en 50%. 50% se define como mitad del tiempo ON y mitad del tiempo OFF
- **Step Count:** Especifica la posición de la blanco como número 32-bit hexadecimal, un valor de Kffffffff causará que el perfil funcione continuamente mientras la salida sea habilitada
- **Workspace:** Especifica una localización de memoria V que es usada por la instrucción
- **Success:** Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Frequency . . . . . V,K	K20-20000; Vea el mapa de memoria V del DL06 - Data Words
Duty Cycle . . . . . V,K	K0-99; Vea el mapa de memoria DL06
Step Count . . . . . V,K	K0-2147434528; Vea el mapa de memoria DL06
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRVELO

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la base del PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que pueda ser usado cualquier IBox de CTRxxxx. El módulo H0-CTRIO se ha configurado para usar V2000 hasta V2025 para sus datos de entradas y V2030 hasta V2061 para sus datos de salidas.



Renglón 2: Este IBox de modo de velocidad CTRIO configura la salida 0 en el módulo H0-CTRIO número 1 para generar 10,000 pulsos a una frecuencia de 1000 Hz. Este ejemplo de programa requiere que Ud. cargue CTRVELO_IBox.cwb en el módulo H0-CTRIO.

(Este ejemplo continúa en la próxima página)

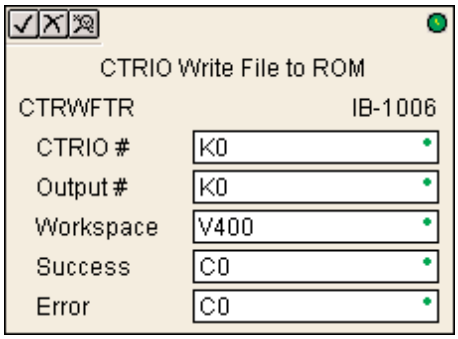
Ejemplo de CTRVELO (continuado)

Renglón 3: Si los parámetros del Modo Velocidad están correctos, active el bit de dirección y habilite la salida.

Escriba archivo a ROM en CTRIO (CTRWFTR) (IB-1006)

DS5	Usado
HPP	N/A

Esta instrucción escribe los cambios runtime realizados a una tabla de valores predefinidos de H0-CTRIO a una memroia Flash-ROM en una transición de APAGADO a ENCENDIDO a este IBox. Este IBox tomará más de un barrido del PLC para ejecutarse. El bit de éxito o de error se activará cuando la instrucción se haya completado.



Si el bit de error está encendido, usted puede utilizar el IBOx de lectura del código de error del módulo H0-CTRIO (CTRRDER) para conseguir una información más detallada del error.

El registroWorkspace (espacio de trabajo) es para uso interno por esta instrucción de IBox y NO DEBE ser usado en cualquier otro lugar en su programa.

5

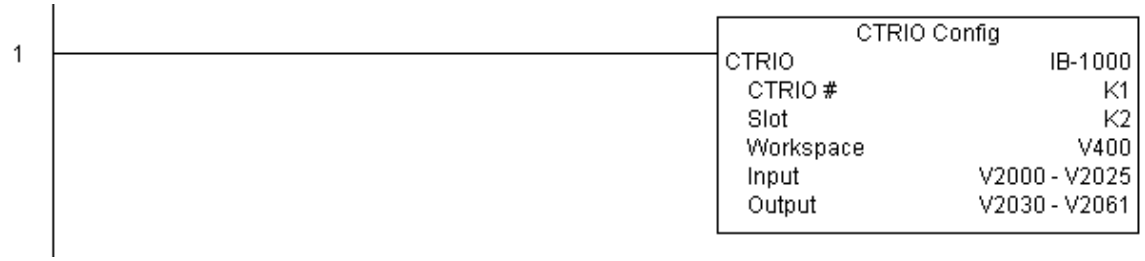
Parámetros de CTRWFTR

- **CTRIO#:** Especifica un módulo CTRIO con un número definido por el usuario (Vea CTRIO Config)
- **Output#**: Especifica una salida del módulo H0-CTRIO a ser usado por la instrucción
- **Workspace**: Especifica una localización de memoria V que es usada por la instrucción
- **Success**: Especifica un bit que se activa cuando la petición se completa con éxito
- **Error:** Especifica un bit que se activa cuando la requisición no se ha terminado con éxito

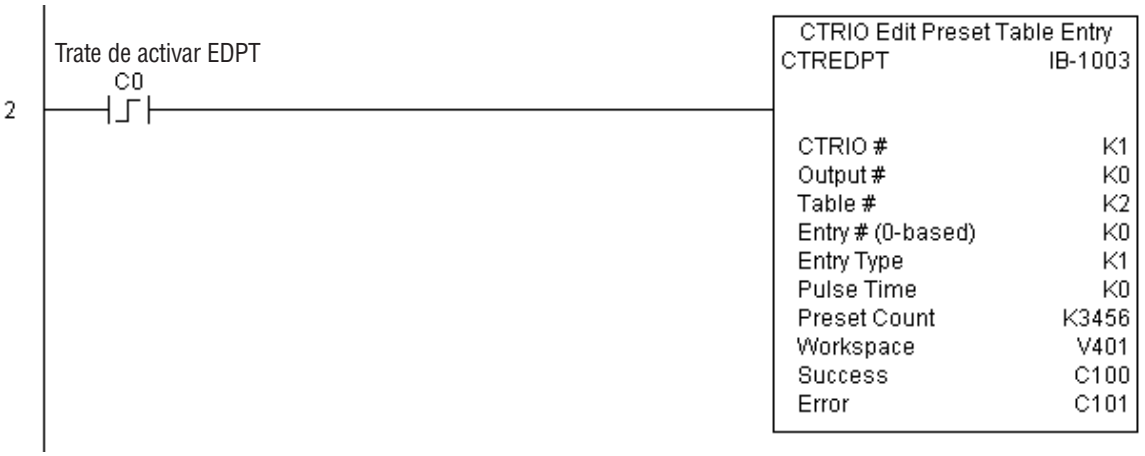
Parámetro	Rango del DL06
CTRIO# . . . . . K	K0-255
Output# . . . . . K	K0-3
Workspace V	Vea el mapa de memoria V del DL06 - Data Words
Success . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06
Error . . . . . X,Y,C,GX,GY,B	Vea el mapa de memoria DL06

Ejemplo de CTRWFTR

Renglón 1: Este ejemplo instala el módulo H0-CTRIO en la ranura 2 de la basedel PLC. Cada H0-CTRIO en el sistema necesitará un IBox de CTRIO separado antes de que cualquier IBox de CTRxxxx pueda ser usado para él. El H0-CTRIO se ha configurado para utilizar V2000 hasta V2025 para sus datos de entradas, y V2030 hasta V2061 para sus datos de salidas.



Renglón 2: Este IBox de CTRIO cambiará la entrada 0 en la tabla #2 para ser un RESET en el conteo 3456. Este programa de ejemplo requiere que usted cargue CTRWFTR_IBox.cwb en su módulo H0-CTRIO.



(Este ejemplo continúa en la próxima página)

Ejemplo de CTRWFTR (continuado)

Renglón 3: Si el archivo se modifica con éxito, use un IBox de escribir archivo a ROM para almacenar la tabla corregida de nuevo a la ROM del módulo H0-CTRIO, de tal modo de hacer los cambios retentivos.