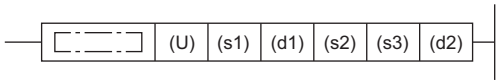
Writing data to the specified file

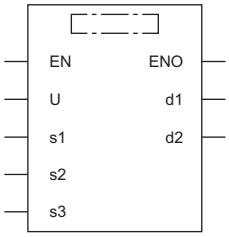
SP.FWRITE

FX5S FX5UJ FX5U FX5UC

This instruction writes device data to the specified file in an SD memory card.

Ladder	ST
	ENO:=SP_FWRITE(EN,U,s1,s2,s3,d1,d2);

FBD/LD



Setting data

■Descriptions, ranges, and data types

Operand	Description	Range	Data type	Data type (label)
(U)	Dummy	■FX5S CPU module U1 ■FX5UJ CPU module U1 to U8 ■FX5U/FX5UC CPU module U1 to U10	Device name*2	ANY16
(s1)	Drive specification	2 (fixed)*1	16-bit signed binary	ANY16
(d1)	Start device where the control data is stored	☞ Page 513 Control data (d1)	Word	ANY16_ARRAY (Number of elements: 8)
(s2)	Start device where the file name is stored	☞ Page 514 File name (s2)	Unicode string	ANYSTRING_DOUBLE
(s3)	Head device where the write-target data is stored	☞ Page 515 Write data (s3)	Word	ANY16*3
(d2)	Bit device that turns on upon completion of the processing	—	Bit	ANYBIT_ARRAY (Number of elements: 2)
EN	Execution condition	—	Bit	BOOL
ENO	Execution result	—	Bit	BOOL

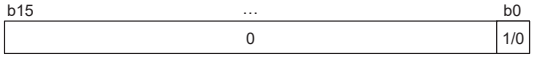
*1 Only drive 2 (for the SD memory card) can be set.
*2 Specification with a label is not allowed.
*3 When specifying setting data by using a label, define an array to secure enough operation area and specify an element of the array label.

■Applicable devices

Operand	Bit	Word			Double word		Indirect specification	Constant			Others (U)
	X, Y, M, L, SM, F, B, SB, S	T, ST, C, D, W, SD, SW, R	U□\G□	Z	LC	LZ		K, H	E	$	
(U)	—	—	—	—	—	—	—	—	—	—	○
(s1)	○	○	—	—	—	—	○	○	—	—	—
(d1)	—	○	—	—	—	—	○	—	—	—	—
(s2)	—	○	—	—	—	—	○	—	—	○	—
(s3)	○*1	○	—	—	—	—	○	—	—	—	—
(d2)	○*2	○*3	—	—	—	—	—	—	—	—	—

*1 If the bit device digit is specified in (s3), only multiples of 16 (0, 16, 32, 64…) can be specified as the device number. Only K4 can be specified as the number of digits.
*2 S cannot be used.
*3 T, ST, and C cannot be used.
Only bit specification of word device is applicable.

■Control data (d1)

Operand: (d1)				
Device	**Item**	**Description**	**Setting range**	**Set by**
+0	Execution/completion type	Specify the execution type. ■00**H: Writing binary data • 0000H: 16-bit binary data • 0001H: 32-bit binary data ■01**H: Writing data after converted to CSV format • 0100H: Decimal (signed 16-bit data) • 0101H: Decimal (unsigned 16-bit data) • 0110H: Decimal (signed 32-bit data) • 0111H: Decimal (unsigned 32-bit data) • 0120H: Hexadecimal (16-bit data) • 0121H: Hexadecimal (32-bit data) • 0130H: String (ASCII data) • 0140H: Floating point real number (single-precision real number)	0000H 0001H 0100H 0101H 0110H 0111H 0120H 0121H 0130H 0140H	User
+1	Completion status	The completion status is stored upon completion of the instruction. • 0000H: Completed successfully • Other than 0000H: Completed with an error (error code) (☞ Page 579 Error codes generated for file operation instructions)	—	System
+2	Number of data actually written	The number of actually written data for the data specified in (s3) is stored. The unit of the value conforms to the data type specification in (d1)+7.	—	System
+3	Application setting area	 b0: Write start position setting Specify the write start position.*1 • 0: Add to the end of the file • 1: Add converting the last line feed code of file to a comma (write continuing the last line)	—	—
+4 +5	File position	■When "Writing binary data" is specified by (d1)+0 • 00000000H: From the beginning of the file • 00000001H to FFFFFFFEH: From the specified position (The unit conforms to the data type specification in (d1)+7.) • FFFFFFFFH: Added to the end of the file. ■When "Converting and writing data in CSV format" is specified by (d1)+0 • 00000000H to FFFFFFFEH: From the beginning of the file • FFFFFFFFH: Added to the end of the file.	00000000H to FFFFFFFFH	User
+6	Number of columns	When "Writing binary data" is specified by (d1)+0, set 0. When "Converting and writing data in CSV format" is specified by (d1)+0, specify the number of columns to write. • 0: No specification of number of columns. Only one row is written. • Other than 0: Number of specified columns	0000H to FFFFH (0 to 65535)	User

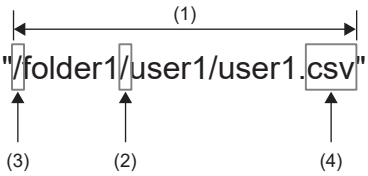
Operand: (d1)				
Device	**Item**	**Description**	**Setting range**	**Set by**
+7	Data type specification	• 0: Word • 1: Even number of bytes*2 • 2: Unit of the data type specified by (d1)+0 • 3: Odd number of bytes*2*3 "0: Word", "1: Even number of bytes", and "3: Odd number of bytes" can be specified only when "0000H: 16-bit binary data" or "0100H: Decimal (signed 16-bit data)" is specified by (d1)+0.	0, 1, 2, 3	User

*1 The setting is enabled only when “01**H: Writing data after converted to CSV format” has been selected for execution/completion type (d1)+0 and “FFFFFFFFH: Added to the end of the file” has been specified for file position (d1)+4 and (D1)+5. When the setting is disabled, specify “0.”

*2 When "1: Even number of bytes" or "3: Odd number of bytes" is specified by (d1)+7 (Data type specification), the setting range of (s3)+0 (Number of request write data) is 1 to 32767.

*3 Even when "3: Odd number of bytes" is specified by (d1)+7 (Data type specification), specify the number in units of words. Add one byte to the odd number of bytes to be written and set the number in (s3)+0 (Number of request write data).

■File name (s2)

Operand: (s2)				
Device	**Item**	**Description**	**Setting range**	**Set by**
+0 to +☐	File name character string	Specify the folder path where the files are stored and the file name. • The folder path and file name (including an extension) must be within 253 characters in total. • The folder path must be within 244 characters. (Delimiters are not included.) • The number of folder path hierarchies must be within 10 levels. • When omitting an extension in the file name, omit the ". (period)" as well. • The file name must be within 60 characters (a period and extension excluded). If 61 or more characters are used, the extension is ignored and replaced with ".BIN" or ".CSV". • Do not specify a half-width space at the end of the character string or just before each delimiter.  (1): Up to 253 characters (2): Use "/" or "\" as delimiters for the folder path and file. (3): Can be omitted. When it is omitted, (1) is up to 252 characters. (4): If the extension is omitted, “.BIN” or “.CSV” will be automatically added by the system.	Unicode string	User

■Write data (s3)

Operand: (s3)				
Device	**Item**	**Description**	**Setting range**	**Set by**
+0	Number of request write data	Specify the number of data to be requested to write. The processing unit and setting range vary depending on Execution/completion type set in (d1)+0 and data type specification in (d1)+7. ■When "Writing binary data" is specified by (d1)+0 • When 16-bit binary data is specified: In units of words (1 to 65535)*1*2 • When 32-bit binary data is specified: In units of double words (1 to 32767) ■When "Converting and writing data in CSV format" is specified by (d1)+0: Number of elements • When decimal (signed 16-bit data) is specified: In units of words (1 to 65535)*1*2 • When decimal (unsigned 16-bit data) is specified: In units of words (1 to 65535) • When decimal (signed 32-bit data) is specified: In units of double words (1 to 32767) • When decimal (unsigned 32-bit data) is specified: In units of double words (1 to 32767) • When hexadecimal (16-bit data) is specified: In units of words (1 to 65535) • When hexadecimal (32-bit data) is specified: In units of double words (1 to 32767) • When a string (ASCII data) is specified: Number of elements (1 to 1023) • When a floating point real number (single-precision real number) is specified: In units of double words (1 to 32767) ■When 0 is specified • If the specified file exists, the data will be written to 0 point, and the program will terminate normally. • If the specified file does not exist, only a folder will be created, and the program will terminate without creating a file.	1 to 65535, 0	User
+1 to +□	Write data	The number of data to be requested to write is stored.	0000H to FFFFH	User

*1 When "1: Even number of bytes" or "3: Odd number of bytes" is specified by (d1)+7 (Data type specification), the setting range of (s3)+0 (Number of request write data) is 1 to 32767.

*2 Even when "3: Odd number of bytes" is specified by (d1)+7 (Data type specification), specify the number in units of words. Add one byte to the odd number of bytes to be written and set the number in (s3)+0 (Number of request write data).

Processing details

- This instruction writes the specified number of data to the specified file. Set the execution/completion type in the control data to specify the file write-target format.
- The write target is the SD memory card only.
- The processing completion bit device (d2) automatically turns on at the execution of the END instruction in the scan in which the completion of processing of the SP.FWRITE instruction is detected. The bit device (d2) turns off at the execution of the END instruction in the next scan. If the processing completion bit device (d2) is ON, it will automatically turn OFF when the SP.FWRITE instruction is executed.
- If the SP.FWRITE instruction completes with an error, the error completion device (d2)+1 turns on or off in synchronization with (d2).
- SM753 (File being accessed) turns on while the SP.FWRITE instruction is being executed. While SM753 is on, the SP.FWRITE instruction cannot be executed. (If the instruction is executed, no processing is performed.)
- Even if an error is detected during the execution of the instruction, the bit devices (d2) and (d2)+1 and SM753 do not turn on.
- Specify data in (s3)+0, (d1)+4, (d1)+5, and (d1)+2 depending on the combination of (d1)+0 and (d1)+7.

Execution/completion type (d1)+0		**Data type specification (d1)+7**	**Processing unit and setting range**		
			Number of request write data (s3)+0	**File position (d1)+4, (d1)+5**	**Number of data actually written (d1)+2**
Writing binary data	0000H: 16-bit binary data	0: Word	Word (0[*1], 1 to 65535)	Word (00000000H to 7FFFFFFFH, FFFFFFFFH)	Word
		1: Even number of bytes	Word (0[*1], 1 to 32767)	Byte (00000000H to FFFFFFFFH)	Byte
		2: Unit of the data type specified by the execution/completion type	Word (0[*1], 1 to 65535)	Word (00000000H to 7FFFFFFFH, FFFFFFFFH)	Word
		3: Odd number of bytes	Word (0[*1], 1 to 32767)[*2]	Byte (00000000H to FFFFFFFFH)	Byte
	0001H: 32-bit binary data	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Double word (0[*1], 1 to 32767)	Double word (00000000H to 3FFFFFFFH, FFFFFFFFH)	Double word
		3: Odd number of bytes	(Cannot be specified)		

Execution/completion type (d1)+0		Data type specification (d1)+7	Processing unit and setting range		
			Number of request write data (s3)+0	File position (d1)+4, (d1)+5	Number of data actually written (d1)+2
Writing data after converted to CSV format	0100H: Decimal (signed 16-bit data)	0: Word	Word (0[*1], 1 to 65535)	Head/end[*3]	Word
		1: Even number of bytes	Word (0[*1], 1 to 32767)	Head/end[*3]	Byte
		2: Unit of the data type specified by the execution/completion type	Word (0[*1], 1 to 65535)	Head/end[*3]	Word
		3: Odd number of bytes	Word (0[*1], 1 to 32767)	Head/end[*3]	Byte
	0101H: Decimal (unsigned 16-bit data)	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Word (0[*1], 1 to 65535)	Head/end[*3]	Word
		3: Odd number of bytes	(Cannot be specified)		
	0110H: Decimal (signed 32-bit data)	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Double word (0[*1], 1 to 32767)	Head/end[*3]	Double word
		3: Odd number of bytes	(Cannot be specified)		
	0111H: Decimal (unsigned 32-bit data)	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Double word (0[*1], 1 to 32767)	Head/end[*3]	Double word
		3: Odd number of bytes	(Cannot be specified)		
	0120H: Hexadecimal (16-bit data)	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Word (0[*1], 1 to 65535)	Head/end[*3]	Word
		3: Odd number of bytes	(Cannot be specified)		
	0121H: Hexadecimal (32-bit data)	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Double word (0[*1], 1 to 32767)	Head/end[*3]	Double word
		3: Odd number of bytes	(Cannot be specified)		
	0130H: String (ASCII data)	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Number of elements (0[*1], 1 to 1023)	Head/end[*3]	Number of elements
		3: Odd number of bytes	(Cannot be specified)		
	0140H: Floating point real number (single-precision real number)	0: Word 1: Even number of bytes	(Cannot be specified)		
		2: Unit of the data type specified by the execution/completion type	Double word (0[*1], 1 to 32767)	Head/end[*3]	Double word
		3: Odd number of bytes	(Cannot be specified)		

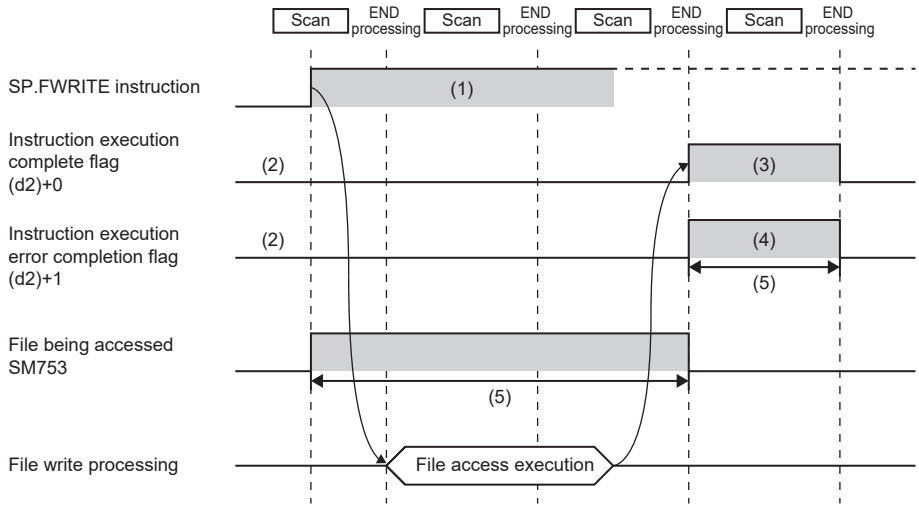
*1 If the specified file does not exist and the number of write target data is 0, a folder to up to the specified path will be created, and the program will terminate normally without creating a file.
If the specified file exists and the number of request write data is 0, the program will terminate normally with the number of actually written data to 0 point

*2 To specify the number of request write data, add 1 byte to the number of odd byte data to be written, and specify the sum in word units.

*3 The set value is from 00000000H to FFFFFFFEH (from the beginning of the file) or FFFFFFFFH (added to the end of the file).

■Timing chart

Below is shown the timing chart (flag updating timing) from the execution of the SP.FWRITE instruction to the completion.

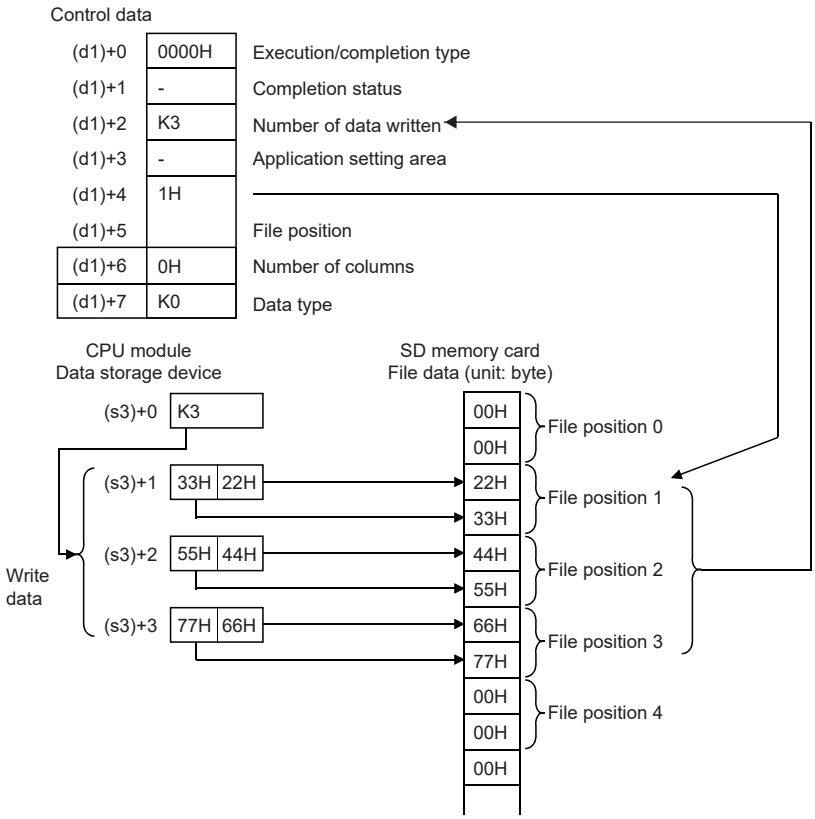


(1) No processing during scanning except when the instruction is turned OFF and ON
(2) Updating to OFF when the instruction is turned OFF and ON
(3) Turning ON upon normal/error completion*1
(4) Turning OFF upon normal completion and turning ON upon error completion*1
(5) Turning ON only during 1 scan after instruction execution
(6) Turning OFF when instruction execution complete flag is turned ON

*1 The complete flag is not turned ON when an error is detected during instruction execution.

■Writing binary data

- If the extension of the target file is omitted, the extension will be ".BIN".
- If a file that does not exist is specified, it will be created and data will be saved to the beginning of the file. The newly created file has the archive attribute.
- When an existing file is specified, data will be saved to the beginning of the file. If the size of data exceeds the size of the existing area in the file during writing, the excess data is additionally stored.
- If the specified position exceeds the existing file size, 0 point of data is written and the processing completes successfully.
- If the media runs out of free space during additional saving of data, an error occurs. In this case, the successfully written and additionally saved data is kept in the written state, and the program will terminate abnormally after additionally saving only the part that can be additionally saved. (8002H is stored in (d1)+1.)
- In 16-bit binary data write mode, when the number of request write data and file position are specified, data is written as follows.



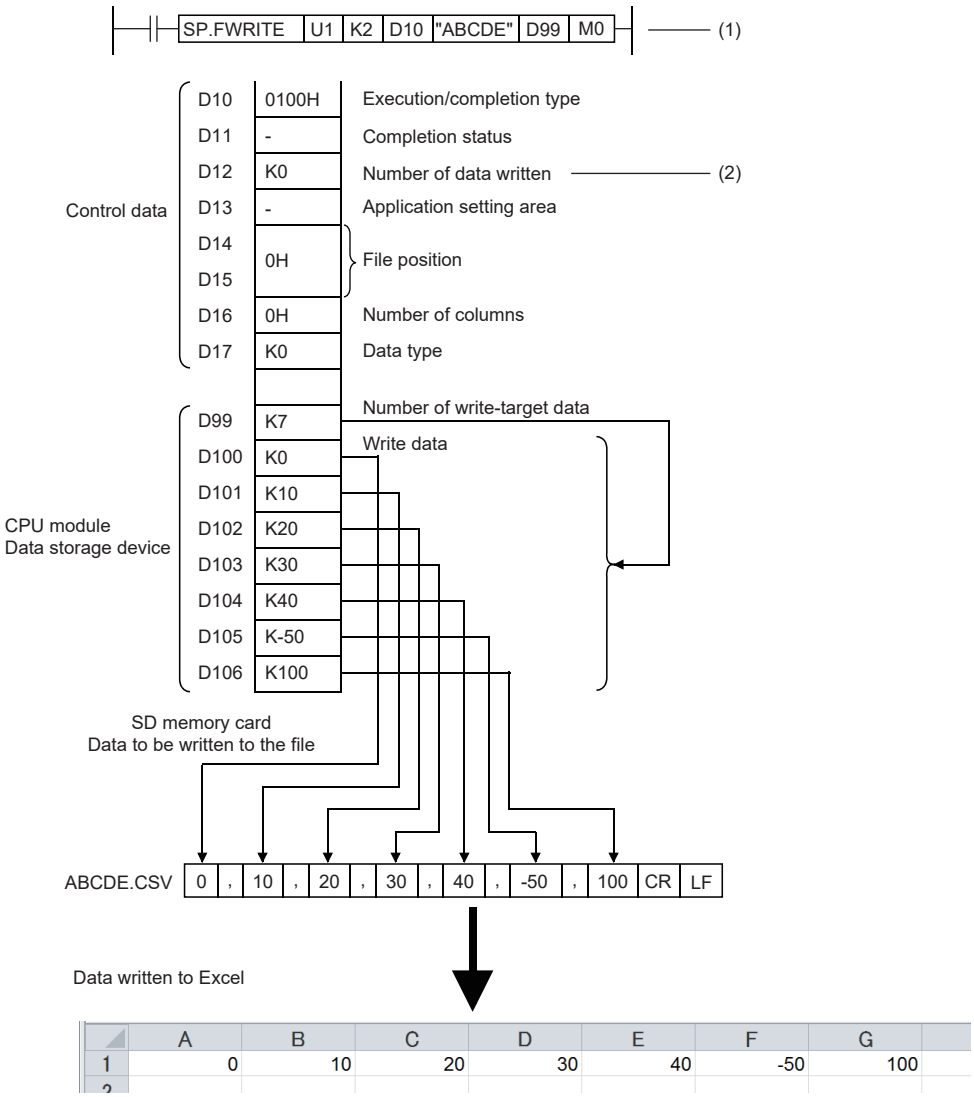
■Converting and writing data in CSV format

- If the extension is omitted, the extension will be ".CSV".
- When an existing file is specified, the following occurs.
  - When a value specified by (d1)+4 and (d1)+5 is other than FFFFFFFFH, data is saved to the file after deleting all the existing data in the file.
  - When a value specified by (d1)+4 and (d1)+5 is FFFFFFFFH, data is added and saved to the end of the file.*1
- When an existing file has been specified and “Add converting the last line feed code of file to a comma” has been specified in (d1)+3.b0, the data will be saved after the last line feed code of the file is converted to a comma. However, the last character of the file is not a line feed code, the character will not be converted to a comma, and the data will be saved to the end of the file.
- If a file that does not exist is specified, it will be created and data will be saved to the beginning of the file. The newly created file has the archive attribute.
- If the media runs out of free space during additional saving of data, an error occurs. In this case, the successfully saved data is kept in the written state, and the program will terminate abnormally after additionally saving only the part that can be additionally saved. (8002H is stored in (d1)+1.)
- When the number of columns is set to 0, data is read as a single-row data in a CSV format file.

*1 When “Added to the end of the file” has been specified in (d1)+3.b0, the data will be saved by staring a new line under the last line of the file.
When “Add converting the last line feed code of file to a comma (write continuing the last line)” has been specified, the data will be saved continuing the last line of the file.

Ex.

The number of columns is set to 0 when writing data after conversion to the CSV format.



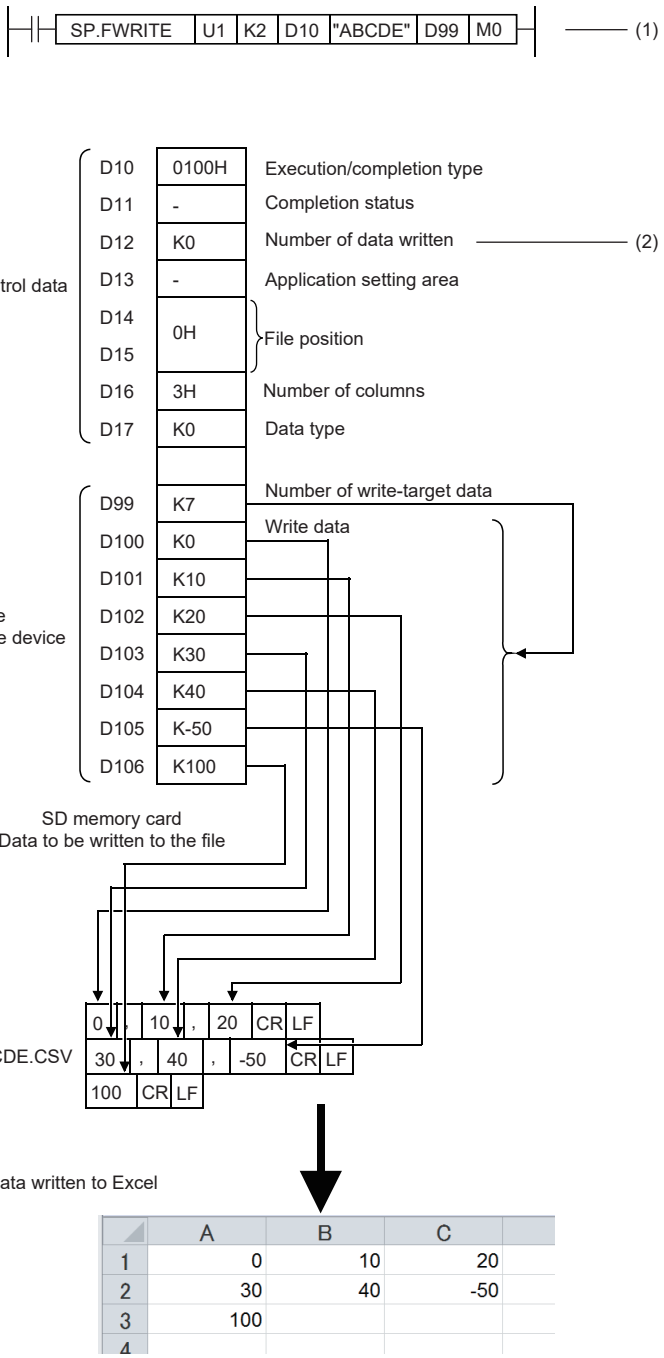
(1) Specified in units of words

(2) Same as the number of write data if case of normal completion

- When the specified number of columns is set to a value other than 0, a CSV format file is stored as the table with the specified number of columns.

Ex.

The number of columns is set to a value other than 0 when writing data after conversion to the CSV format.



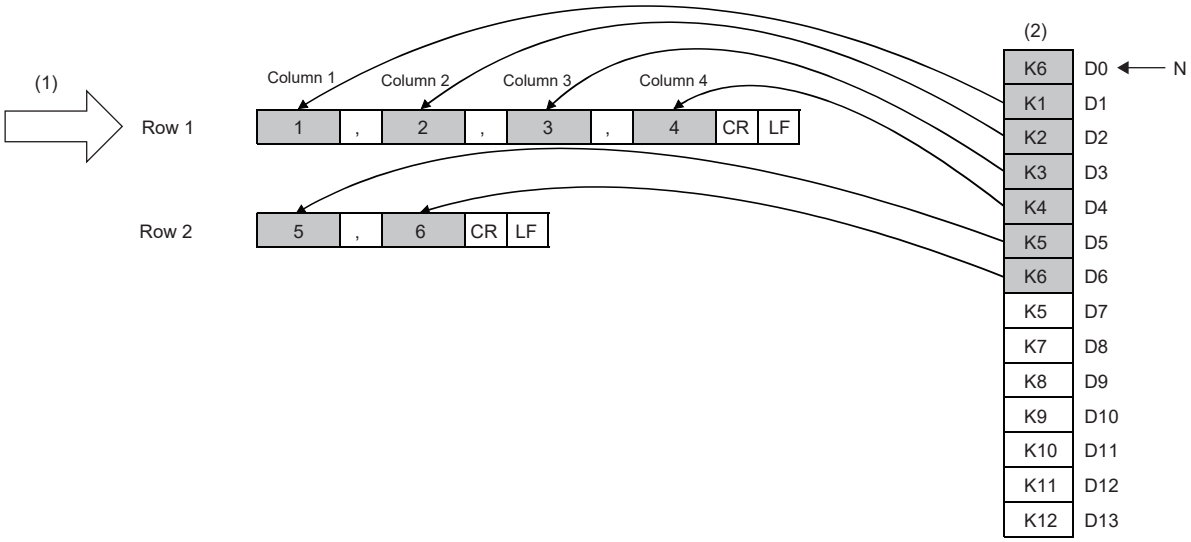
(1) Specified in units of words

(2) Same as the number of write data if case of normal completion

• The following figure shows how data is added.

[Specify the file to which data will be written.] (Even if the file exists, it is deleted and re-created.)

- Execution/completion type: Writing data after converted to CSV format (Decimal (signed 16-bit data))
- Number of request write data: 6H
- File position: 00000000H (from the beginning of the file)
- Number of columns specification: 4H
- Data type specification: Words
- Write start device: D0



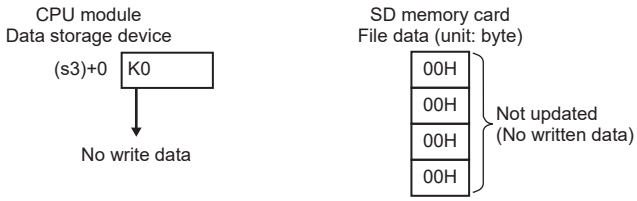
(1) Starting row

(2) Data in the device (write data)

N: Number of data

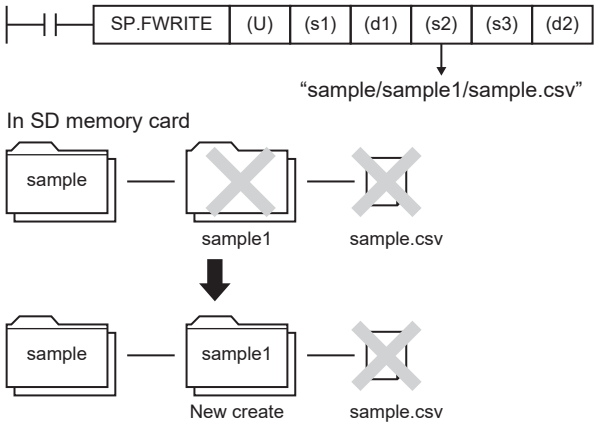
• When the number of request write data is 0, the processing varies depending on the existence of the specified file.

[When the specified file exists]



Since data is written to 0 point, the specified file will not be updated.
The program is finished normally.

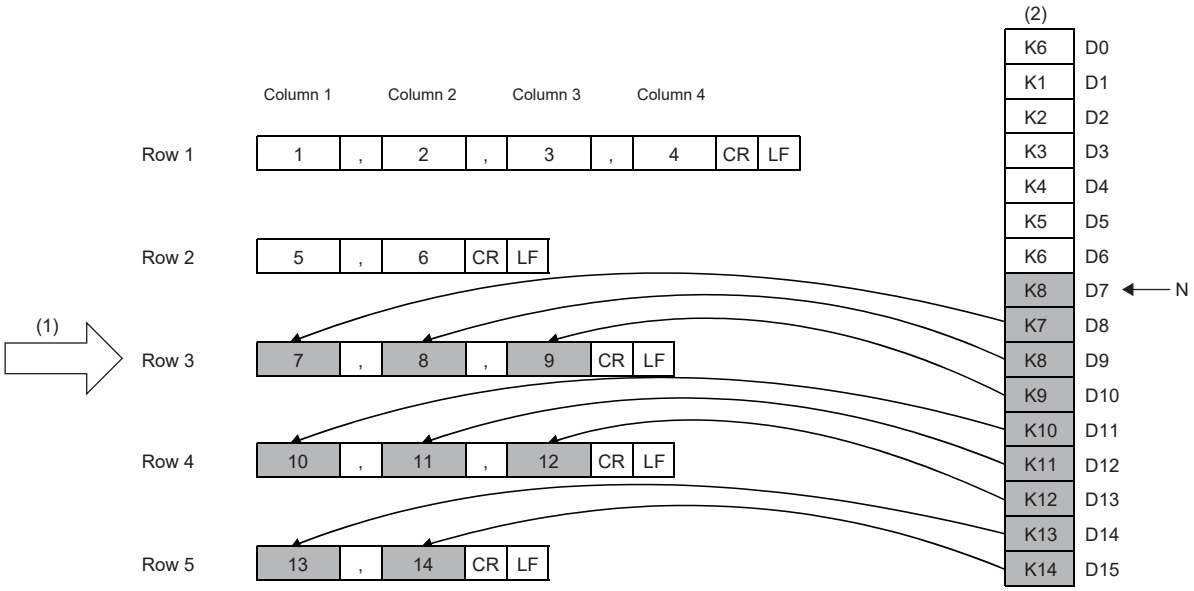
[When the specified file does not exist]



The specified folder path will be created, and the processing will be terminated. When the folder exists, no processing will be performed.
The specified file will not be created.

[Added to the end of the file]

- Execution/completion type: Writing data after converted to CSV format (Decimal (signed 16-bit data))
- Number of request write data: 8H
- File position: FFFFFFFFH (added to the end of the file)
- Number of columns specification: 3H
- Data type specification: Words
- Write start device: D7



(1) Starting position

(2) Data in the device (write data)

N: Number of data

Point

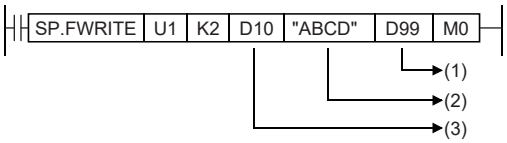
- To justify the written data columns, set “Number of request write data” to an integral multiple of “Specified number of columns.” If the setting is not an integral multiple, the lines will vary in the number of columns.
- When data is added to the end of a file, columns are shifted if "Number of columns" is changed from the previous writing.
- When “String (ASCII data)” has been specified for execution/completion type, set 00H (NULL) at the end of the string of one element. When the string has an even number of bytes, set 0000H (NULL for 2 bytes) for the next 1 word.
- When “String (ASCII data)” has been specified for execution/completion type, the maximum number of characters in 1 element is 1999. If the number of characters exceeds 1999 and 00H (NULL) is not stored, the 2000th and following characters will not be written, and the next element will be written.
- When “String (ASCII data)” has been specified for execution/completion type, up to 1023 elements can be written by executing the instruction once.

• The following figures show an example of specifying "String (ASCII data)" for the execution/completion type.

Ex.

Writing data after converted to CSV format (string (ASCII data))

[Data to be written to a file]



(1) Data to be written
(2) File name
(3) Control data

[Control data]

D10	H0130
D11	H0000
D12	-
D13	K0
D14	0H
D15	
D16	0H
D17	K2

D10: Execution/completion type: String (ASCII data)
D11: Completion status
D12: Number of data actually written
D13: Application setting area
D14, D15: File position
D16: Number of columns
D17: Data type specification

CPU module
Data storage device

D099	K3
D100	H4241
D101	H4443
D102	H0000
D103	H3231
D104	H0033
D105	H3534
D106	H0036

"ABCD"
"123"
"456"

D99: Number of request write data
D100 to D102: String to be written to the 1st column of the 1st line
D103, D104: String to be written to the 2nd column of the 1st line
D105, D106: String to be written to the 3rd column of the 1st line

SD memory card
Data to be written to the file

ABCD.CSV

ABCD	,	123	,	456	CR	LF

Data written to Excel

	A	B	C	D
1	ABCD	123	456	
2				

Point

- Set 00H (NULL) in the end of the string in an element. When the number of bytes of the string is even, set 0000H (two bytes of NULL) in the next one word.
- The maximum number of characters in one element is 1999. If this maximum number is exceeded and 00H (NULL) is not stored, characters of 2000th character and after are not written and the write processing shifts to the next element.
- A maximum of 1023 elements can be written in a single instruction execution.

• The following table lists values to set in (s3)+1 and later and data to be written in a CSV file when "0140H: Floating point real number (single-precision real number)" is set to (d1)+0.

Execution/completion type ((d1)+0)	Value to set in the write data ((s3)+1 and later)	Data to be written in a CSV file
0140H: Floating point real number (single-precision real number)	Values within the range of: $-2^{128} < \text{data} \leq -2^{-126}$, 0, $2^{-126} \leq \text{data} < 2^{128}$	A value (0 to 7 digits in the decimal part) as given on the left is written in exponential format.
	Values other than above	0 is written. (Data cannot be converted.)

• The following shows how the file size (total number of bytes) is calculated when a CSV format file is written to the SD memory card.

[Total number of bytes] = [Total number of bytes excluding the last row] + [Number of bytes of the last row]

([Number of bytes of each row] = [Number of columns*1] +1+ [total number of bytes of all data values per line]*2)

*2 The specified number of columns applies to rows other than the last row. The number of columns of the last row is calculated as shown below because it may differ from the specified number of rows depending on the number of write data.

· The number of rows excluding the last row is calculated. (Number of rows excluding the last row = number of requested write data÷number of columns (remainders rounded down)

· The number of columns of the last row is calculated. (Number of columns of the last row = number of requested write data - (number of rows excluding the last row × number of columns))

*3 The following shows how the number of bytes of each data value is calculated.

Sign of data value	Number of bytes of each data value	Range of bytes	Example
Positive	Number of digits	1 to 5 (word specification) 1 to 3 (byte specification)	• 12345: 5 bytes • 67: 2 bytes
Negative	Number of digits + 1	2 to 6 (word specification) 2 to 4 (byte specification)	• -12345: 6 bytes • -67: 3 bytes

Program example

An example of a program for writing binary data/CSV data to the SD memory card by using the SP.FWRITE instruction is shown below.

■Writing binary data

When X0 is turned ON, the data in D101 to D200 are written starting from the 16th word from the beginning of the "sample.bin" file stored in the SD memory card.

[Program operation]

1. Control data is created during RUN.
2. Set the number of request write data.*1
3. The drive contact of X0 is held in M0. When the drive contact is turned ON, the instruction execution complete flag and instruction error completion flag are initialized.
4. The SP.FWRITE instruction is executed*2
5. Since the instruction execution complete flag and instruction error completion flag are ON only during 1 scan, they are held in the M150 and M151 devices to identify the normal/abnormal completion.

*1 The data to be written must have been set although this step is not contained in the program example.

*2 The instruction is executed after confirming that the following special devices are OFF to prevent simultaneous execution of another file operation instruction.
- SM606 (Memory card disable request)
- SM753 (File being accessed)

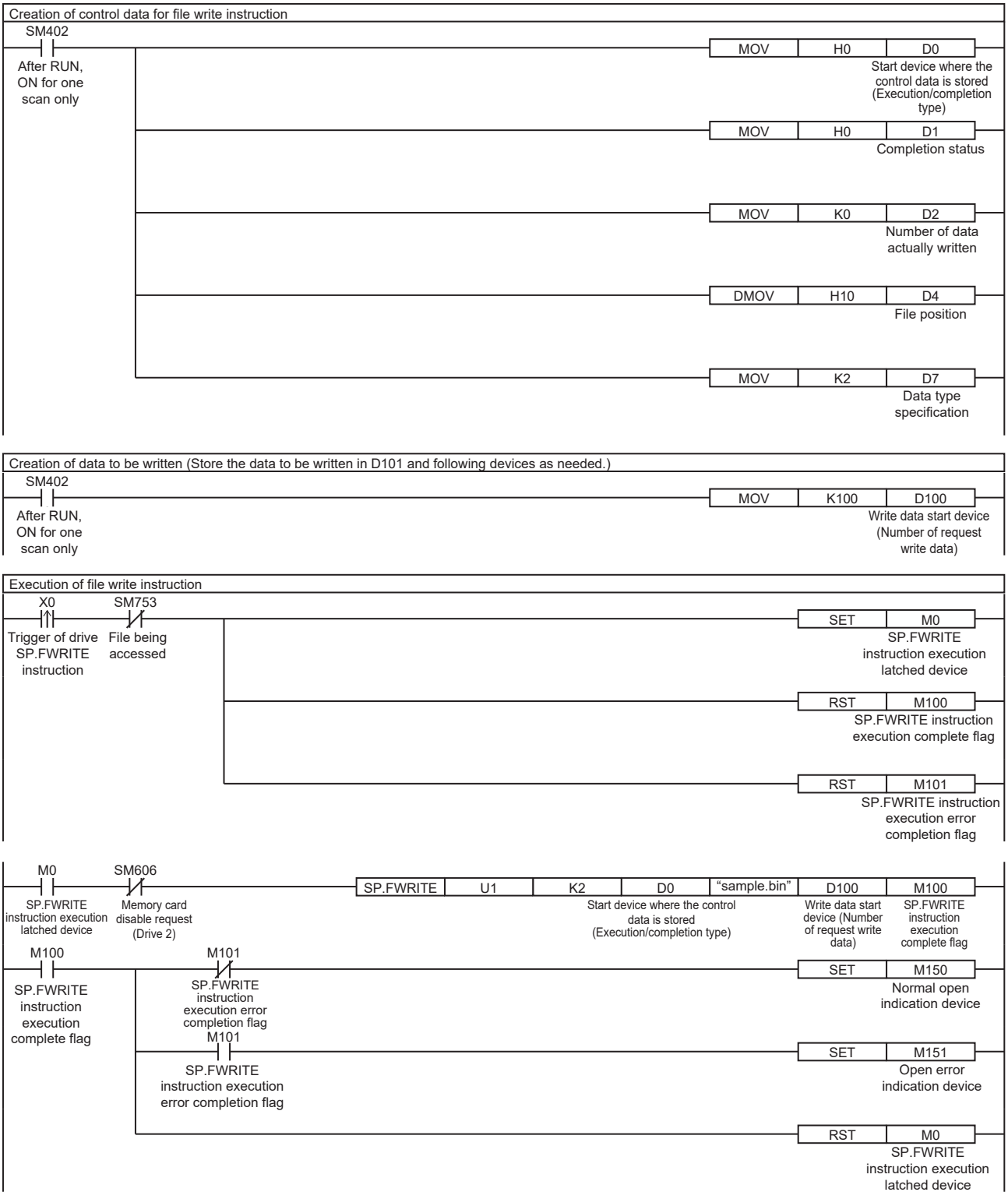
[Devices used]

Device	Description
X0	Trigger of drive SP.FWRITE instruction
D0	Start device where the control data is stored • D0: Execution/completion type • D1: Completion status • D2: Number of data actually written • D4,D5: File position • D7: Data type specification
D100	Head device where the write-target data is stored • D100: Number of request write data • D101 to D200: Write data
M0	SP.FWRITE instruction execution latched device
M100	SP.FWRITE instruction execution complete flag
M101	SP.FWRITE instruction execution error completion flag
M150	Normal open indication device
M151	Open error indication device

[SP.FWRITE instruction operand setting]

Operand	Description	Set value
(U)	Dummy	U1
(s1)	Drive specification	K2 (SD memory card)
(d1)	Start device where the control data is stored	D0: 0H (16-bit binary data) D1: 0H(Completed successfully) D2: K0 D4,D5: 10H(Writing starting from the 16th word from the beginning) D7: K2 (Unit of data type specified in D0)
(s2)	Start device where the file name is stored	"sample.bin"
(s3)	Head device where the write-target data is stored	D100: K100 (Writing 100 words) D101 to D200: Write data
(d2)	Bit device that turns on upon completion of the processing	M100: Execution complete flag M101: Execution error completion flag

[Ladder program]



[ST program]

```
//(1) Creation of control data for file write instruction
IF SM402 THEN
D0 := H0; //Execution/completion type (Control data start device)
D1 := H0; //Completion status
D2 := 0; //Number of data actually written
D4:UD := H10; //File position
D7 := 2; //Data type specification
END_IF;

//(2) Creation of data to be written
IF SM402 THEN
D100 := 100; //Number of request write data
//Store the data to be written in D101 and following devices as needed.
END_IF;

//(3) Processing to start up the drive contact (X0)
IF LDP(TRUE, X0) THEN
//Checking that the file being accessed flag is OFF
IF (SM753 <> TRUE) THEN
SET(TRUE, M0); //Holds drive contact
RST(TRUE, M100); //Initialize instruction execution complete flag
RST(TRUE, M101); //Initialize instruction execution error complete flag
END_IF;
END_IF;

//(4) Execution of file write instruction
IF M0 THEN
//Checking that the memory card disable request is OFF
IF (SM606 <> TRUE) THEN
//EN = TRUE (Enable Input, always execute)
//U = U1 (Dummy)
//S1 = 2 (Drive specification, 2 fixed)
//S2 = "sample.bin" (Start device where the file name is stored)
//S3 = D100(Head device where the write-target data is stored)
//D1 = D0 (Start device where the control data is stored)
//D2 = M100 (Bit device that turns on upon completion of the processing)
SP_FWRITE(TRUE, U1, 2, "sample.bin", D100, D0, M100);
END_IF;
END_IF;

//(5) Checking the instruction execution complete flag
IF M100 THEN
SET((M101 <> TRUE), M150); //Holds instruction execution complete flag
SET(M101, M151); //Holds instruction execution error complete flag
RST(TRUE, M0); //Releasing the drive contact
END_IF;
```

[Execution result]

When the target device data is written by using the program example, the results are as shown below.

[Write target]*1

Device Name	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Current Value	String
D99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	..
D100	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0064	d.
D101	0	0	0	0	0	0	1	1	1	1	1	0	1	0	0	1	03E9	)064
D102	0	0	0	0	0	0	1	1	1	1	1	0	1	0	1	0	03EA	
D103	0	0	0	0	0	0	1	1	1	1	1	0	1	0	1	1	03EB	
D104	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	0	03EC	
D105	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	1	03ED	
																		(1)
D196	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0048	H.
D197	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0049	I.
D198	0	0	0	0	0	1	0	0	0	1	0	0	1	0	1	0	004A	J.
D199	0	0	0	0	0	1	0	0	0	1	0	0	1	0	1	1	044B	K.
D200	0	0	0	0	0	1	0	0	0	1	0	0	1	1	0	0	044C	L.
D201	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	..
D202	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	..

[Write result (sample.bin)]

sample.bin

ADRESS	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
00000000	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	ZZZZZZZZZZZZZZZZ
00000010	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	5A	ZZZZZZZZZZZZZZZZ
00000020	E9	03													EF	03	F0	03	. . . . .
00000030	F1	03	F2	03	F3	03	F4	03	F5	03	F6	03	F7	03	F8	03	. . . . .
00000040	F9	03	FA	03	FB	03	FC	03	FD	03	FE	03	FF	03	00	04	. . . . .
00000050	01	04	02	04	03	04	04	04	05	04	06	04	07	04	08	04	. . . . .
00000060	09	04	0A	04	0B	04	0C	04	0D	04	0E	04	0F	04	10	04	. . . . .
00000070	11	04	12	04	13	04	14	04	15	04	16	04	17	04	18	04	. . . . .
00000080	19	04	1A	04	1B	04	1C	04	1D	04	1E	04	1F	04	20	04	. . . . .
00000090	21	04	22	04	23	04	24	04	25	04	26	04	27	04	28	04	! . " . # . $. % . & . ' . (.
000000A0	29	04	2A	04	2B	04	2C	04	2D	04	2E	04	2F	04	30	04	) . * . + . , . - . . . / . 0.
000000B0	31	04	32	04	33	04	34	04	35	04	36	04	37	04	38	04	1. 2. 3. 4. 5. 6. 7. 8.
000000C0	39	04	3A	04	3B	04	3C	04	3D	04	3E	04	3F	04	40	04	9. : . ; . <. =. >. ?. @.
000000D0	41	04	42	04	43	04	44	04	45	04	46	04	47	04	48	04	A. B. C. D. E. F. G. H.
000000E0	49	04	4A	04	4B	04	4C	04									I . J . K. L.

(2)

(1) In the program example, 100 words of data are written. Accordingly, the number of request write data is stored in D100, and the data to be written are stored in D101 to D200.

(2) 100 words from 03E9 to 044C (D101 to D200) are written. In the program example, the data is written starting from the 16th word, and therefore 03E9 (device value of D101) is stored in the 16th word.

*1 Device/Buffer Memory Batch Monitor (hexadecimal display) of engineering tool

■Converting and writing data in CSV format

When X0 is turned on, the data in D101 to D115 are written to the end of the “sample.csv” file stored in the SD memory card.

[Program operation]

1. Control data is created during RUN.
2. Set the number of request write data.*1
3. The drive contact of X0 is held in M0. When the drive contact is turned ON, the instruction execution complete flag and instruction error completion flag are initialized.
4. The SP.FWRITE instruction is executed.*2
5. Since the instruction execution complete flag and instruction error completion flag are ON only during 1 scan, they are held in the M150 and M151 devices to identify the normal/abnormal completion.

*1 The data to be written must have been set although this step is not contained in the program example.

*2 The instruction is executed after confirming that the following special devices are OFF to prevent simultaneous execution of another file operation instruction.

· SM606 (Memory card disable request)

· SM753 (File being accessed)

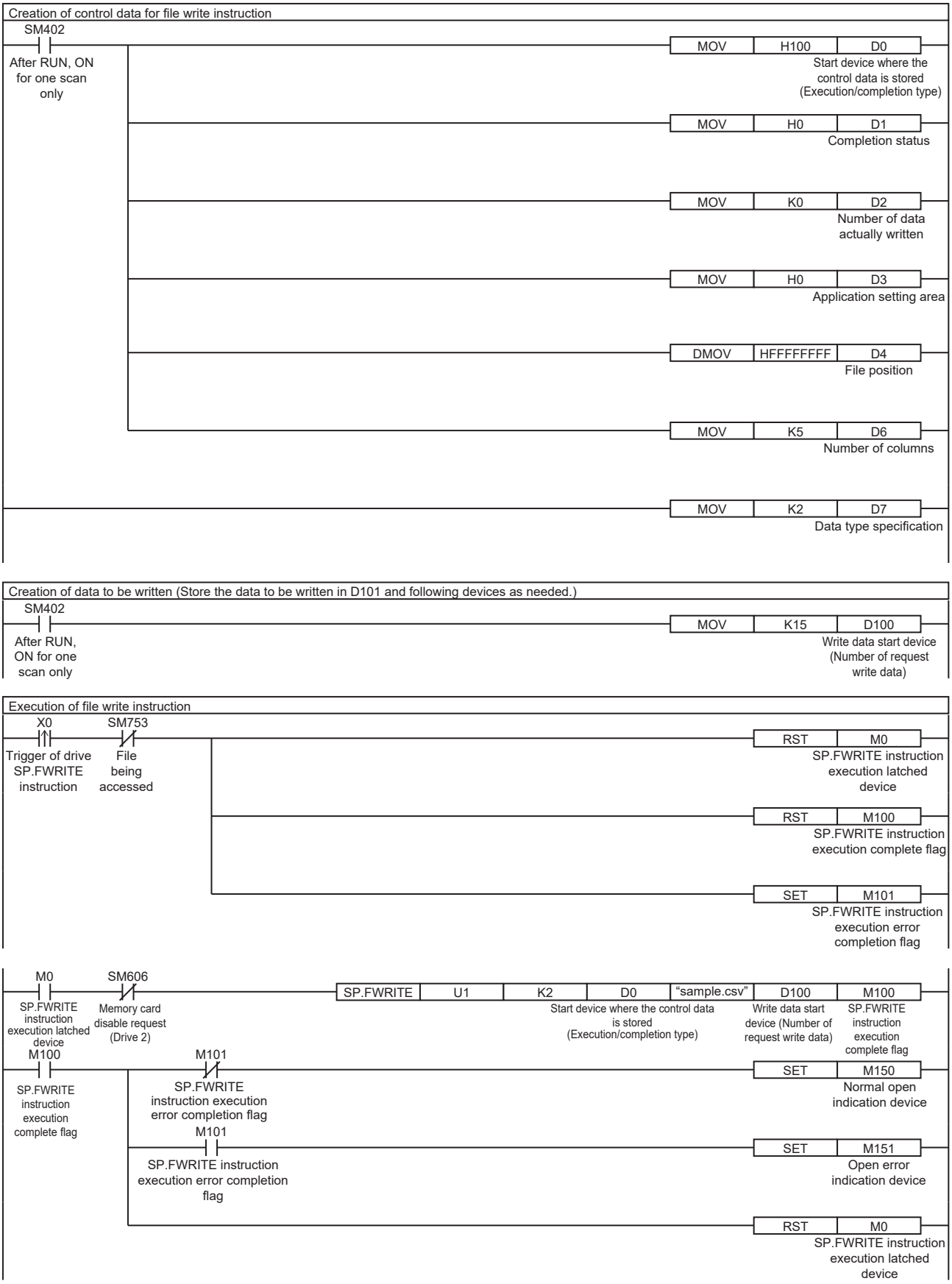
[Devices used]

Device	Description
X0	Trigger of drive SP.FWRITE instruction
D0	Start device where the control data is stored • D0: Execution/completion type • D1: Completion status • D2: Number of data actually written • D3: Application setting area • D4,D5: File position • D6: Number of columns • D7: Data type specification
D100	Head device where the write-target data is stored • D100: Number of request write data • D101 to D115: Write data
M0	SP.FWRITE instruction execution latched device
M100	SP.FWRITE instruction execution complete flag
M101	SP.FWRITE instruction execution error completion flag
M150	Normal open indication device
M151	Open error indication device

[SP.FWRITE instruction operand setting]

Operand	Description	Set value
(U)	Dummy	U1
(s1)	Drive specification	K2 (SD memory card)
(d1)	Start device where the control data is stored	D0: 100H (Decimal (signed 16-bit data)) D1: 0H (Completed successfully) D2: K0 D3: H0 (Add to the end of the file) D4,D5: FFFFFFFFH (Added to the end of the file.) D6: K5 (Writing by 5 columns) D7: K2 (Unit of data type specified in D0)
(s2)	Start device where the file name is stored	“sample.csv”
(s3)	Head device where the write-target data is stored	D100: K15 (Writing 15 words) D101 to D115: Write data
(d2)	Bit device that turns on upon completion of the processing	M100: Execution complete flag M101: Execution error completion flag

[Ladder program]



[ST program]

```
//(1) Creation of control data for file write instruction
IF SM402 THEN
D0 := H100; //Execution/completion type (Control data start device)
D1 := H0; //Completion status
D2 := 0; //Number of data actually written
D3 := H0; //Application setting area
D4:UD := HFFFFFFFF; //File position
D6 := 5; //Number of columns
D7 := 2; //Data type specification
END_IF;

//(2) Creation of data to be written
IF SM402 THEN
D100 := 15; //Number of request write data
//Store the data to be written in D101 and following devices as needed.
END_IF;

//(3) Processing to start up the drive contact (X0)
IF LDP(TRUE, X0) THEN
//Checking that the file being accessed flag is OFF
IF (SM753 <> TRUE) THEN
SET(TRUE, M0); //Holds drive contact
RST(TRUE, M100); //Initialize instruction execution complete flag
RST(TRUE, M101); //Initialize instruction execution error complete flag
END_IF;
END_IF;

//(4) Execution of file write instruction
IF M0 THEN
//Checking that the memory card disable request is OFF
IF (SM606 <> TRUE) THEN
//EN = TRUE (Enable Input, always execute)
//U = U1 (Dummy)
//S1 = 2 (Drive specification, 2 fixed)
//S2 = "sample.csv" (Start device where the file name is stored)
//S3 = D100 (Head device for storing the data to be written)
//D1 = D0 (Start device where the control data is stored)
//D2 = M100 (Bit device that turns on upon completion of the processing)
SP_FWRITE(TRUE, U1, 2, "sample.csv", D100, D0, M100);
END_IF;
END_IF;

//(5) Checking the instruction execution complete flag
IF M100 THEN
SET((M101 <> TRUE), M150); //Holds instruction execution complete flag
SET(M101, M151); //Holds instruction execution error complete flag
RST(TRUE, M0); //Releasing the drive contact
END_IF;
```

[Execution result]

When the target device data is written by using the program example, the results are as shown below.

[Write target]*1

Device Name	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Current Value	String
D100	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	15	
D101	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	
D102	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	15	
D103	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	16	
D104	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	255	].
D105	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	256	
D106	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	4095	].
D107	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	4096	
D108	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	32767	
D109	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-32768	
D110	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	-32767	
D111	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	]□
D112	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
D113	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	4369	
D114	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	21845	)U
D115	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	-21846	I

[Write result (sample.csv)] (1)

	A	B	C	D	E	F	
1	1	2	3	4	5	6	
2	1	2	3	4	5	6	
3	A	B		D	E	F	
4	1	15	16	255	256		
5	4095	4096	32767	-32768	-32767	(2)	
6	-1	0	4369	21845	-21846		
7							

(1) In the program example, 15 words of data are written. Accordingly, the number of request write data is stored in D100, and the data to be written are stored in D101 to D115.

(2) The 15 words from 1 to -21846 (D101 to D115) are written. In the program example, the data are written by 5 columns to the end of the file, and therefore the data are stored after the existing data (that exists before writing).

*1 Device/Buffer Memory Batch Monitor (decimal display) of engineering tool)

Precautions

- The SP.FWRITE instruction cannot be executed in user interrupt programs. An error (3582H) will occur.
- When “01**H” has been specified in Execution/completion type (d1)+0 and an option other than “FFFFFFFFH” has been specified in File position (d1)+4 and (d1)+5, if “Add converting the last line feed code of file to a comma (write continuing the last line)” is specified in (d1)+3.b0, an error will occur.
- The SP.FWRITE instruction cannot be executed while SM606 (Memory card disable request) is ON. When SM606 is turned ON during execution of the instruction, the program will terminate abnormally. (The data that has been written before SM606 is turned ON will be stored in the SD memory card.)
- The files in the system folder ($MELPRJ$) cannot be handled by the SP.FWRITE instruction. If an access to any file in the system folder ($MELPRJ$) is requested, a calculation error (3405H) will occur.
- The SP.FWRITE instruction cannot be executed simultaneously with the SP.DEVST instruction, the SP.FTPPUT instruction, and the SP.FTPGET.
- Do not disconnect the power or remove the SD memory card during execution of the SP.FWRITE instruction. (The file may be damaged, or an error may occur.)

Operation error

Error code (SD0/SD8067)	Description
2820H	The bit label digits specified in (s2) and (s3) are unacceptable settings (the number of digits is not K4).
	The storage device of the control data (d1) exceeds the end of the device range.
	The value in the device specified by (s3)+0 is out of the range (1 to 65535), or exceeds the setting area specified by (s3)+1 and later in the device/label memory.
3405H	The drive specified by (s1) is not the one for the SD memory card.
	Any value that is set in the device specified by (d1) and later as control data is out of the range.
	The file name string specified by (s2) cannot be read. • The number of characters of the string in the file name specified exceeds the range. • An inhibited value is set. • The specified file name string ends with a delimiter. • The system folder ($MELPRJ$) directly under the route folder is specified. • A half-width period is specified at the end of the specified file name string or immediately before each delimiter.
3427H	The combination of the execution/completion type specified in (d1)+0, the write start position set in (d1)+3 and the file position specified in (d1)+4 is not allowed.
	An invalid combination of (d1)+0 (Execution/completion type) and (d1)+7 (Data type specification) is specified.
3582H	The SP.FWRITE instruction is executed in an interrupt program.

If the SP.FWRITE instruction completes with an error, an error code is stored in the device specified by (d1)+1. (Note that an error code is not stored if the instruction results in an operation error.)

For the error code stored in (d1)+1, refer to the following.

☞ Page 579 Error codes generated for file operation instructions