IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

General Description

This document was produced to assist the PLC programmer in setting up their program to access Shared Data Variables within the Mettler Toledo IND780 Weigh Terminal. For a list of Shared Data Variables that can be accessed using this method, please refer to the 64059110 - IND780 Terminal Shared Data Reference.

The program documented in this document, and available for download IND780_ARIO_FLT_SDA_RC20_V01.ACD, was written in **ControlLogix5000, version 20**. It is highly recommended that the program be read from a ControlLogix Editor since the presentation in this document must be limited out of the necessity for brevity and simplicity.

The Shared Data mode PLC communications mode is not available with the Allen-Bradley Remote I/O option. Consequently, Block Transfers are used instead.

Commands from the PLC to the IND780 to read from or write to Shared Data Variables are handled by the Block Transfer Write. The PLC program must set up the Block Transfer Write buffer with the desired command to the IND780.

Responses from the IND780 Terminal are handled by the Block Transfer Read. The PLC program must read the data returned in the Block Transfer Read buffer to determine what the result of the command was.

We'll briefly describe the sequence here, and then cover it in more detail later in the document.

IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

Reading a Value from the IND780

Note that for all examples below, the 8-byte Shared Data variable field code is the Shared Data Variable name in ASCII with two spaces preceding it.

Below is the Block Transfer Write table definition as it appears in the IND780's PLC Manual.

Base #	0	1	2	3	4	5	6	7	8	9
N#:0	Display Mode*	16 Byte Display String: sent from PLC to terminal shared data if preceding word is non-zero value and discrete display bits are set to 7								8 Byte>> ASCII
N#:10	<<Floating Point Write Field Code: shows where next value will be loaded			Floating Point Write Value		8 Byte ASCII String Write Field Code: shows where the next value will be loaded				40 Byte>>
N#:20	<<40 Byte String Data. Note: if string is shorter than 40 bytes it must be left justified and null-terminated >>									
N#:30	<< 40 Byte String Data. Note: if string is shorter than 40 bytes it must be left-justified (and null-terminated)>>									8 Byte>> ASCII
N#:40	<<Floating Point Read Field Code: requests FP value for BTR			8 Byte (ASCII) String Read Field Code: requests string value for BTR			Reserved			
N#:50	Reserved									
N#:60	Reserved									

To tell the IND780 to return a floating point variable, such as AJ0101, the variable's name needs to be written into the table starting at word 39 as shown below.

Step 1: Send Shared Data Name to be read via a Block Transfer Write

8 Byte Shared Data Name →

0	1	2	3	4	5	6	7	8
9	10	11	12	13	14			
15	16	17	18	19	20			
21	22	23	24	25	26			
27	28	29	30	31	32			
33	34	35	36	37	38			
39	40	41	42	43	44			
45	46	47	48	49	50			
51	52	53	54	55	56			
57	58	59	60	61	62			
63								

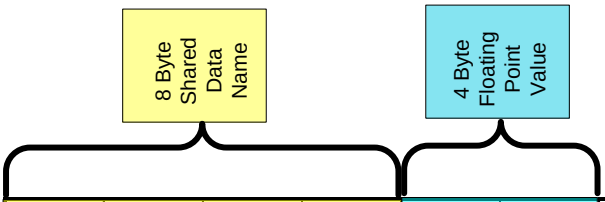
Write Block Transfer – Word Map

Below is the Block Transfer Read table as shown in the PLC manual. This is where the response from the IND780 will be returned to the PLC.

IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

Base #	0	1	2	3	4	5	6	7	8	9
N#:0	8 Byte (ASCII) Floating Point Read Field Code: name of value sent in next field				Floating Point Read Value		8 Byte (ASCII) String Read Field Code: name of string sent in next field			
N#:10	40 Byte Data String>>									
N#:20	<<40 Byte String Data. Note: if string is shorter than 40 bytes it must be left-justified (and null-terminated)>>									
N#:30	Reserved									
N#:40	Reserved									
N#:50	Reserved									
N#:60	Reserved									

The requested Shared Data Variable name will be returned in the first four words, and the 2 word (4 byte) floating point value will immediately follow it as shown below.



Step 2: Confirm that the Shared Data name matches the requested name, then read the data.

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	**Read Block Transfer – Word Map**					

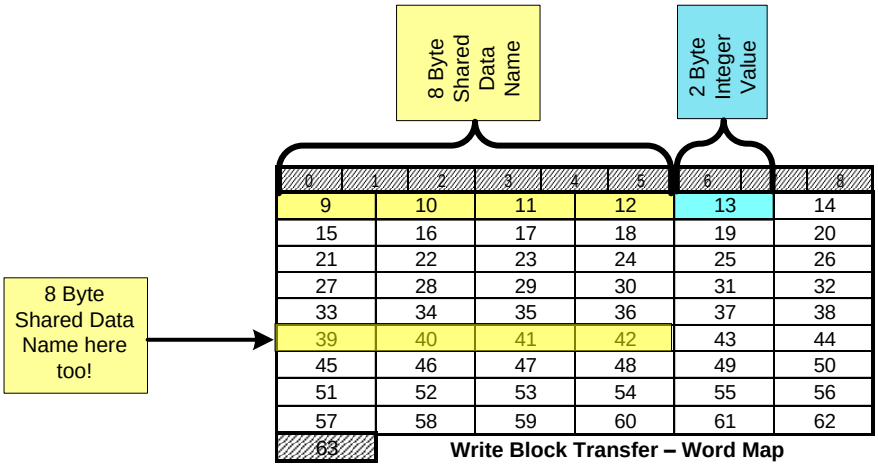
Writing a Value to the IND780

Writing a value is actually simpler than reading one, because all that has to be done is to setup the Block Transfer Write buffer. For instance, to write a 1 to the Integer Shared Data Variable AI0101 the PLC program would need to write the name of the Shared Data Variable (AI0101) to words 9 thru 12, followed by the desired Integer value in Word 13 as shown below.

IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

Base #	0	1	2	3	4	5	6	7	8	9
N#:0	Display Mode*	16 Byte Display String: sent from PLC to terminal shared data if preceding word is non-zero value and discrete display bits are set to 7								8 Byte>> ASCII
N#:10	<<Floating Point Write Field Code: shows where next value will be loaded			Floating Point Write Value		8 Byte ASCII String Write Field Code: shows where the next value will be loaded				40 Byte>>
N#:20	<<40 Byte String Data. Note: if string is shorter than 40 bytes it must be left justified and null-terminated >>									
N#:30	<< 40 Byte String Data. Note: if string is shorter than 40 bytes it must be left-justified (and null-terminated)>>									8 Byte>> ASCII
N#:40	<<Floating Point Read Field Code: requests FP value for BTR			8 Byte (ASCII) String Read Field Code: requests string value for BTR			Reserved			
N#:50	Reserved									
N#:60	Reserved									

The table below may be a little easier to understand.



Write Block Transfer – Word Map

Note that it is also necessary to repeat the name of the Shared Data variable in words 39 thru 42 so that the subsequent Block Transfer Read will get a response from the IND780 and not time-out.

String Reads and Writes work the same way, but must access the String fields in the Block Transfer buffers instead of the Floating Point fields.

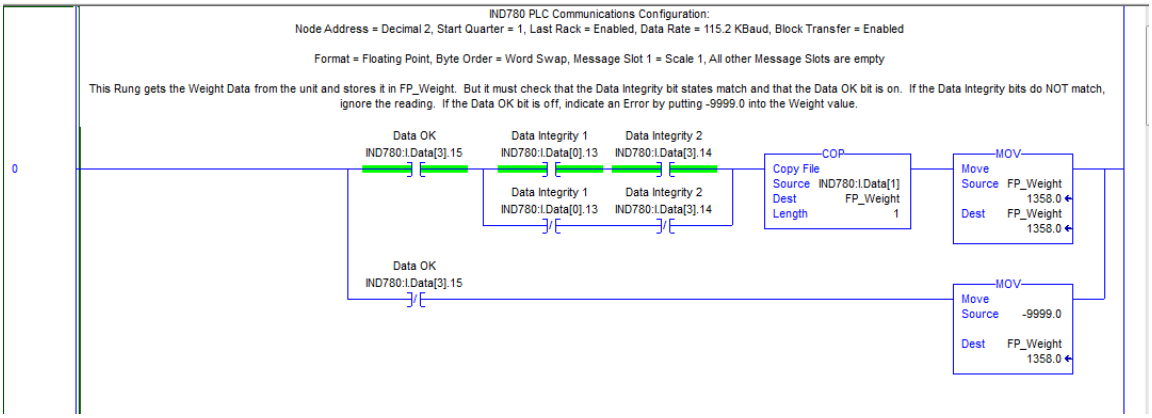
IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

The PLC Program Description

Main Routine

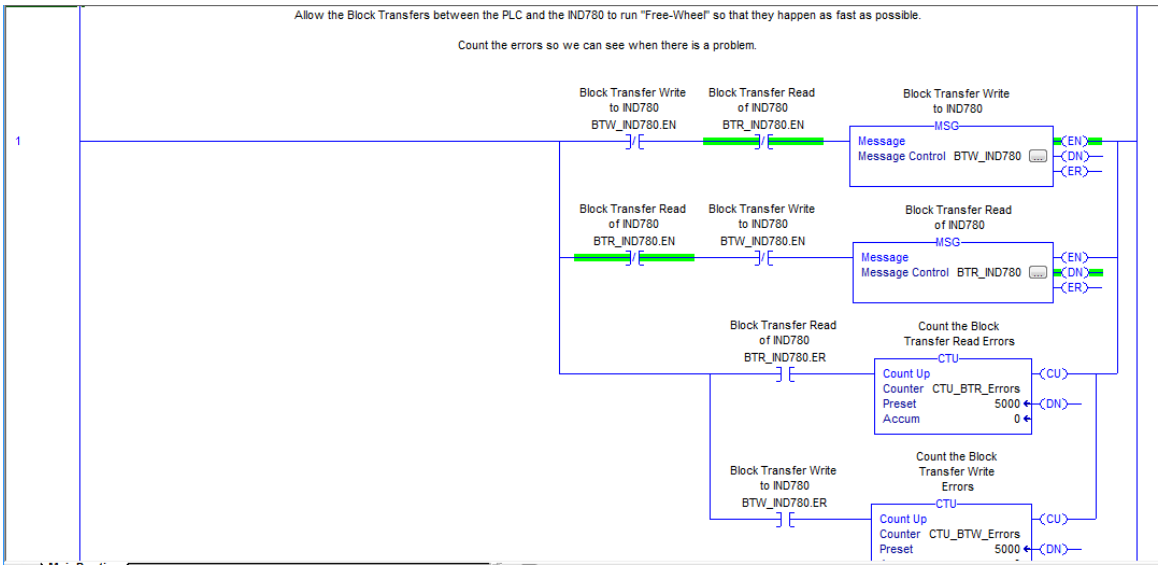
Read the Scale

Below is the logic used to Read the Main Scale when using Remote I/O. Note that it does not go through the Block Transfers used for the Shared Data Variable Access.



Run the Block Transfers

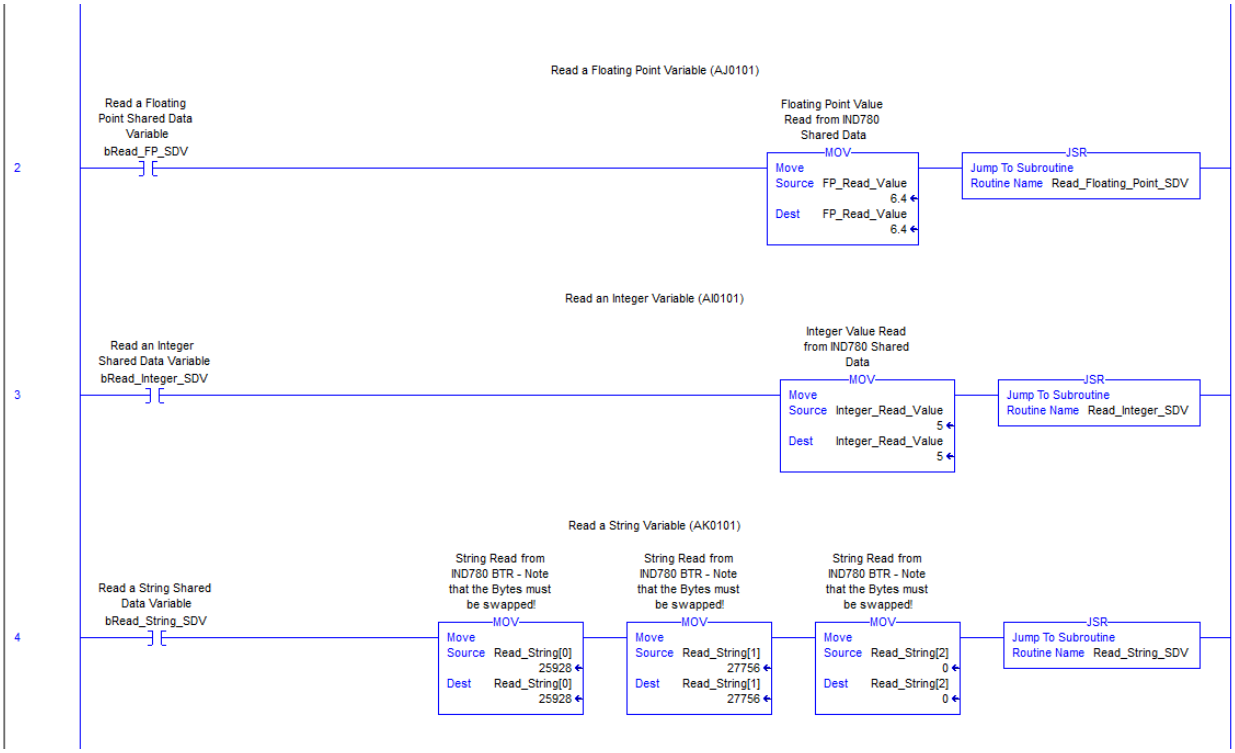
The block transfers must not run at the same time, hence the logic on the rung ahead of each one. In this program, we're letting the Block Transfers run as fast as possible.



IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

Shared Data Read Triggers

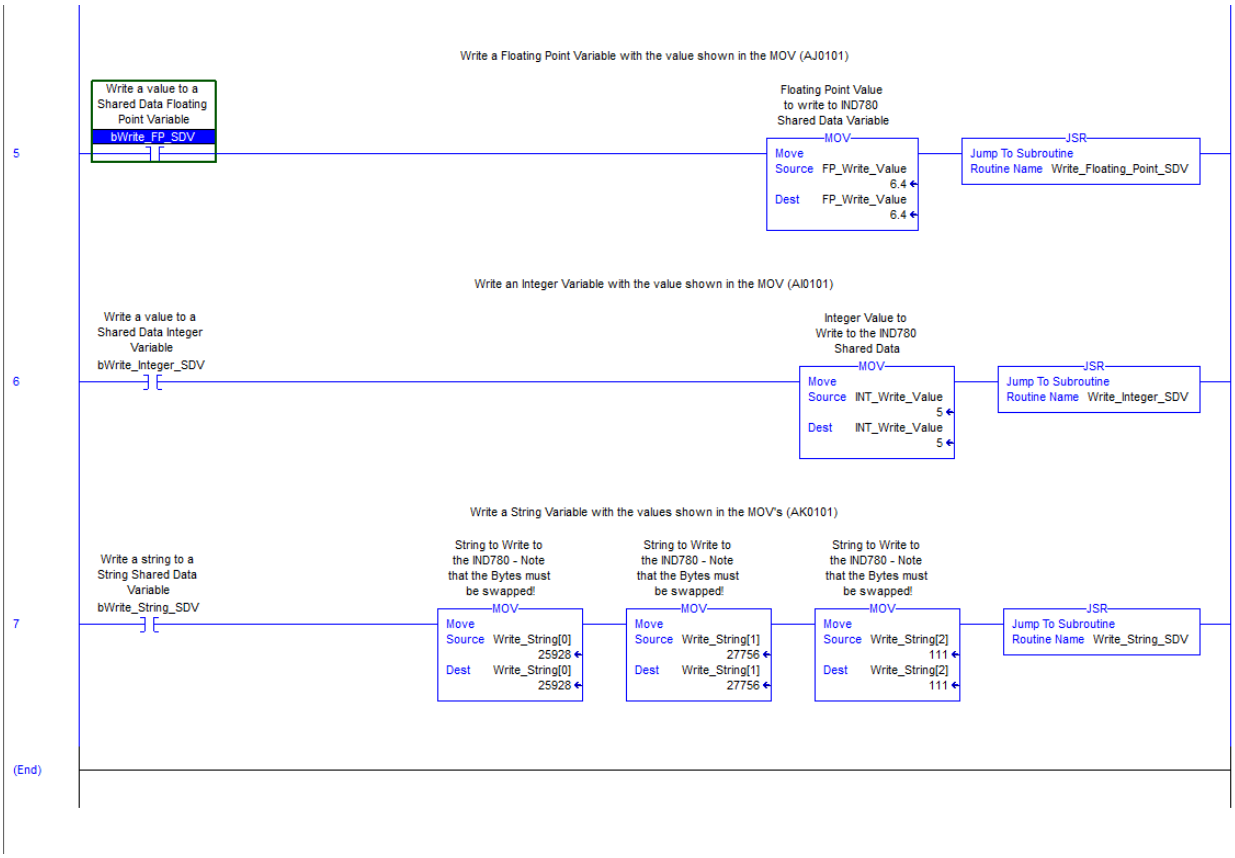
Each of the rungs below are used to trigger a different kind of read. Note that the MOV instructions in each rung are used to display the results of the Read action and have no effect on the program's logic. To trigger each of the actions, merely set the corresponding "bRead" tag to True.



Shared Data Write Triggers

Each of the rungs below are used to trigger a different kind of write. Again, note that the MOV instructions in each rung are used to display the results of the Write action and have no effect on the program's logic. To trigger each of the actions, merely set the corresponding "bWrite" tag to True.

IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O



Shared Data Variable Name Constants

It is useful to define constants containing the Shared Data Variable names that will be used throughout the program. Since the characters in the names need to be transposed before transmitting to the IND780, they can be transposed in the constants without requiring PLC logic. This can provide a significant simplification of the program.

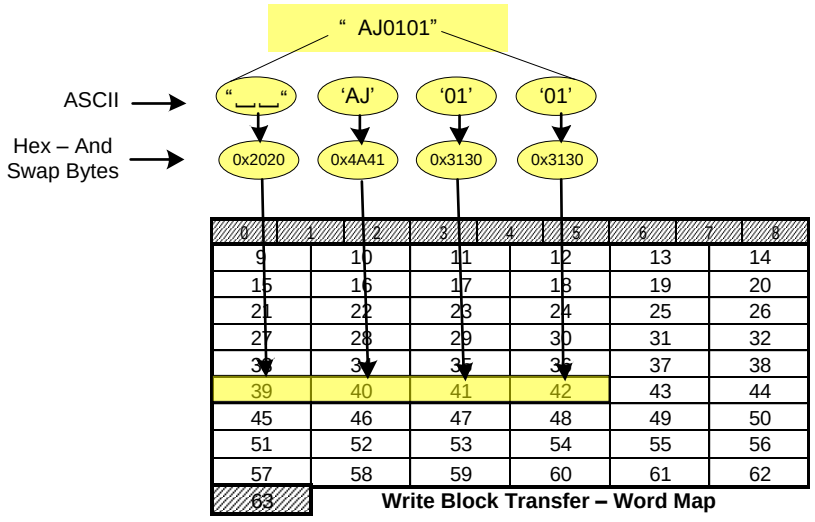
Below are the constants defined for this program. Note that the characters have been swapped.

AI0101	{...}	{...}	ASCII	INT[4]	Integer Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AI0101[0]	' '		ASCII	INT	Integer Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AI0101[1]	'IA'		ASCII	INT	Integer Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AI0101[2]	'10'		ASCII	INT	Integer Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AI0101[3]	'10'		ASCII	INT	Integer Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AJ0101	{...}	{...}	ASCII	INT[4]	Floating Point Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AJ0101[0]	' '		ASCII	INT	Floating Point Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AJ0101[1]	'JA'		ASCII	INT	Floating Point Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AJ0101[2]	'10'		ASCII	INT	Floating Point Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AJ0101[3]	'10'		ASCII	INT	Floating Point Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AK0101	{...}	{...}	ASCII	INT[4]	String Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AK0101[0]	' '		ASCII	INT	String Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AK0101[1]	'KA'		ASCII	INT	String Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AK0101[2]	'10'		ASCII	INT	String Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.
AK0101[3]	'10'		ASCII	INT	String Shared Data name String for AJ0101 - NOTE that Bytes are SWAPPED.

IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

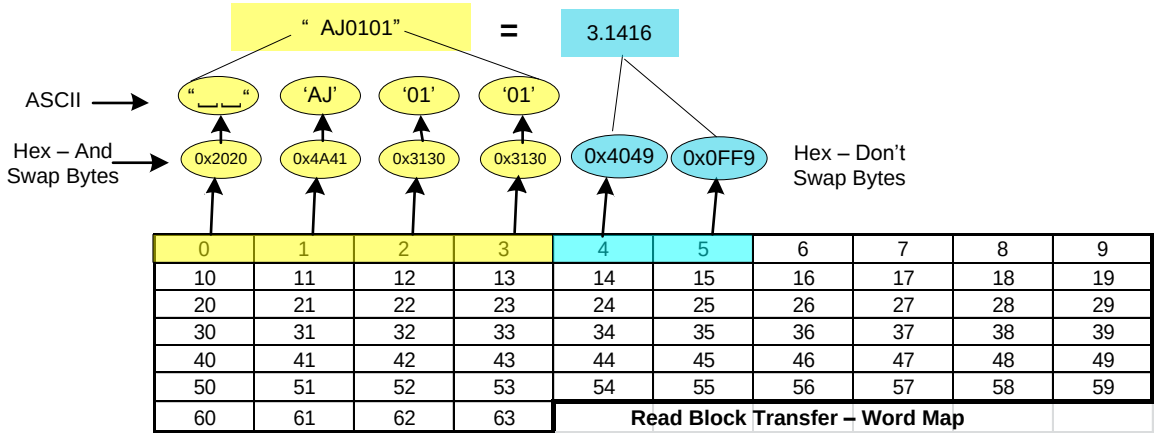
Read_Floating_Point_SDV Routine

To Read a Floating Point variable, the PLC must first populate the BTW buffer starting at word **39** with the desired Shared Data Variable name (note the byte order). Since the Block Transfers are freewheeling, that is running constantly, nothing else needs to happen except to wait for the result from the IND780.



0	1	2	3	4	5	6	7	8
9	10	11	12	13	14			
15	16	17	18	19	20			
21	22	23	24	25	26			
27	28	29	30	31	32			
33	34	35	36	37	38			
39	40	41	42	43	44			
45	46	47	48	49	50			
51	52	53	54	55	56			
57	58	59	60	61	62			
63	**Write Block Transfer – Word Map**							

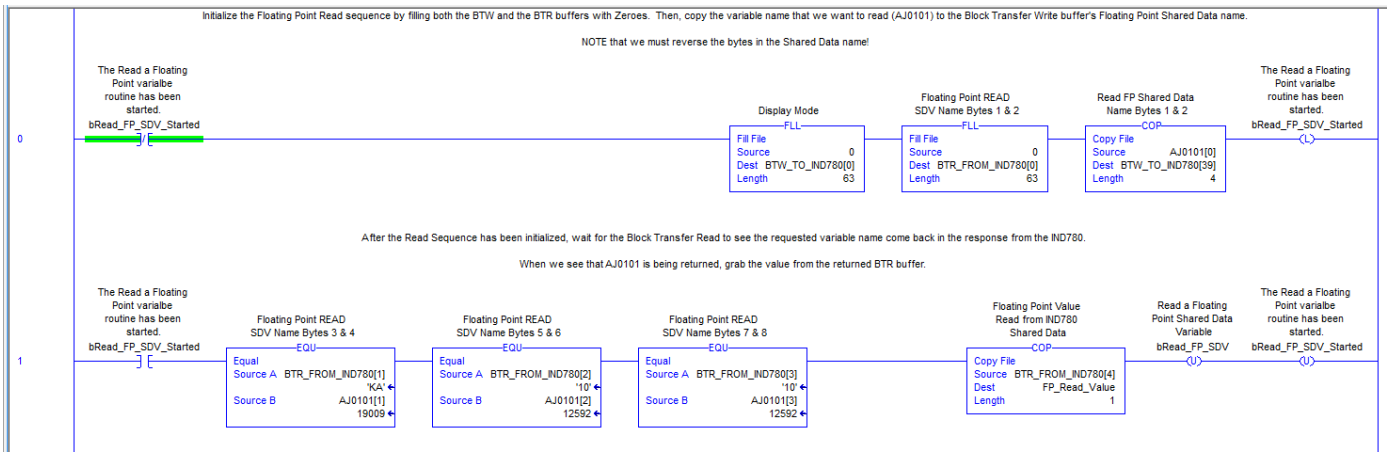
When the PLC reads that the requested Shared Data Variable name (AJ0101) is present in the Block Transfer Read buffer, it can now go get the resulting value.



0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	**Read Block Transfer – Word Map**					

Below is the corresponding PLC code the performs the above sequence. Note the FLL instructions which zero out the BTR and BTW buffers before issuing the command.

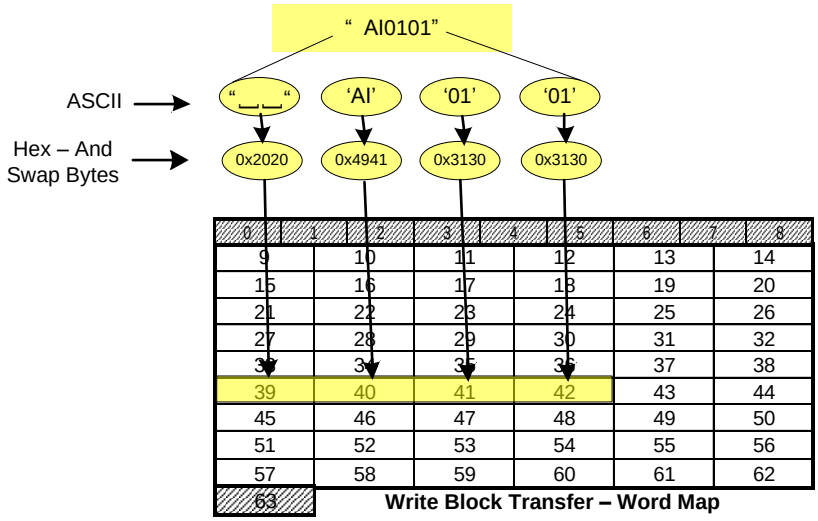
IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O



The EQU instructions wait for the Block Transfer Read to show that the IND780 is returning the same variable name that was requested. When the compares pass, the result is copied into the floating point variable, and the Read Request flag is cleared.

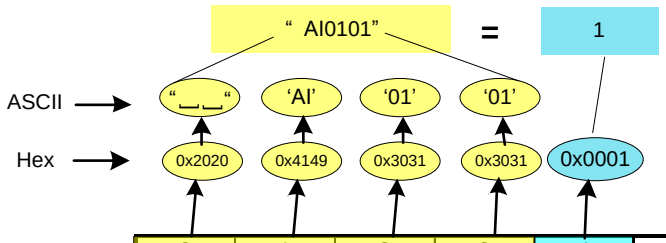
Read_Integer_SDV Routine

To Read an Integer variable, the PLC must first populate the BTW buffer starting at word **39** with the desired Shared Data Variable name (note the byte order). Again, since the Block Transfers are freewheeling, that is running constantly, nothing else needs to happen except to wait for the result from the IND780.



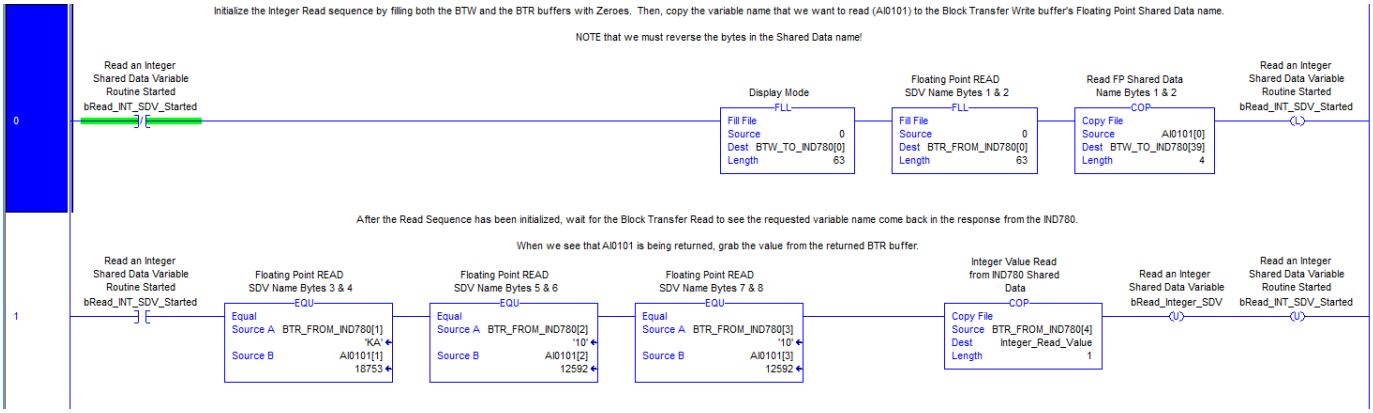
IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

When the PLC reads that the requested Shared Data Variable name (AI0101) is present in the Block Transfer Read buffer, it can now go get the resulting value.



0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	**Read Block Transfer – Word Map**					

Below is the corresponding PLC code the performs the above sequence. Note the FLL instructions which zero out the BTR and BTW buffers before issuing the command.

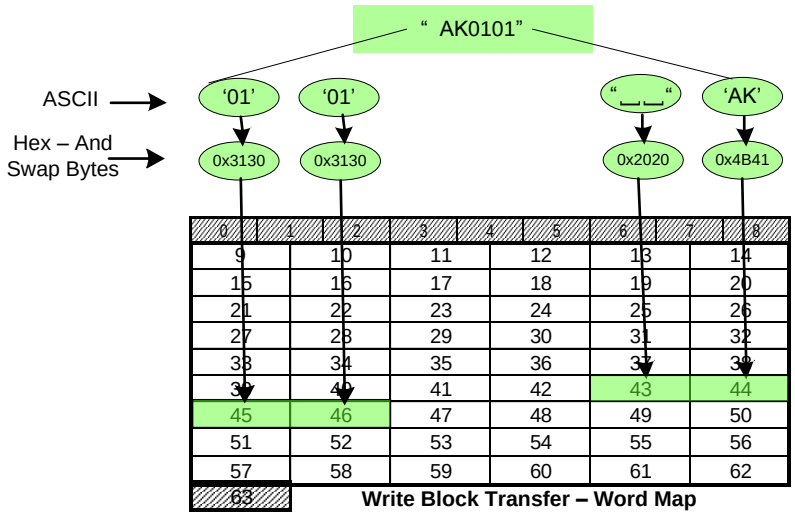


The EQU instructions wait for the Block Transfer Read to show that the IND780 is returning the same variable name that was requested. When the compares pass, the result is copied into the Integer Read variable, and the Read Request flag is cleared.

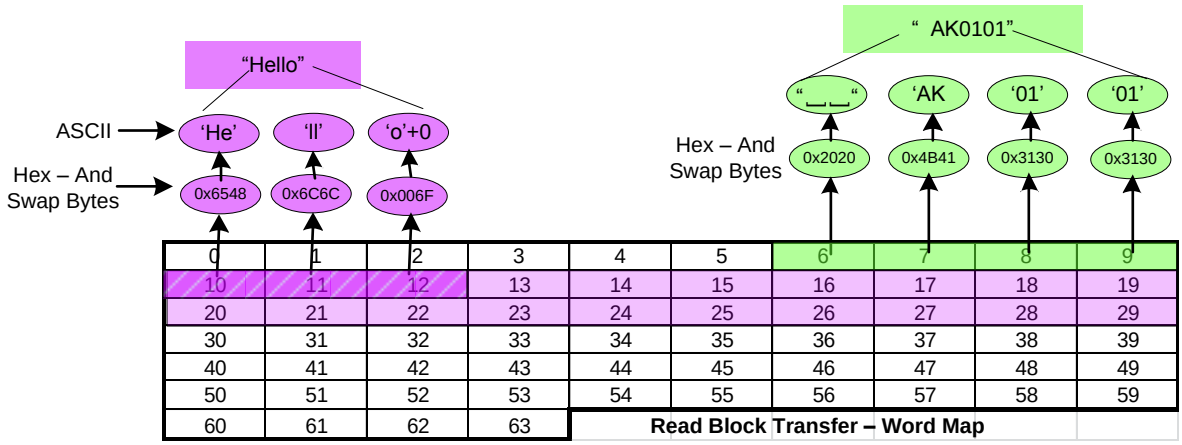
IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

Read_String_SDV Routine

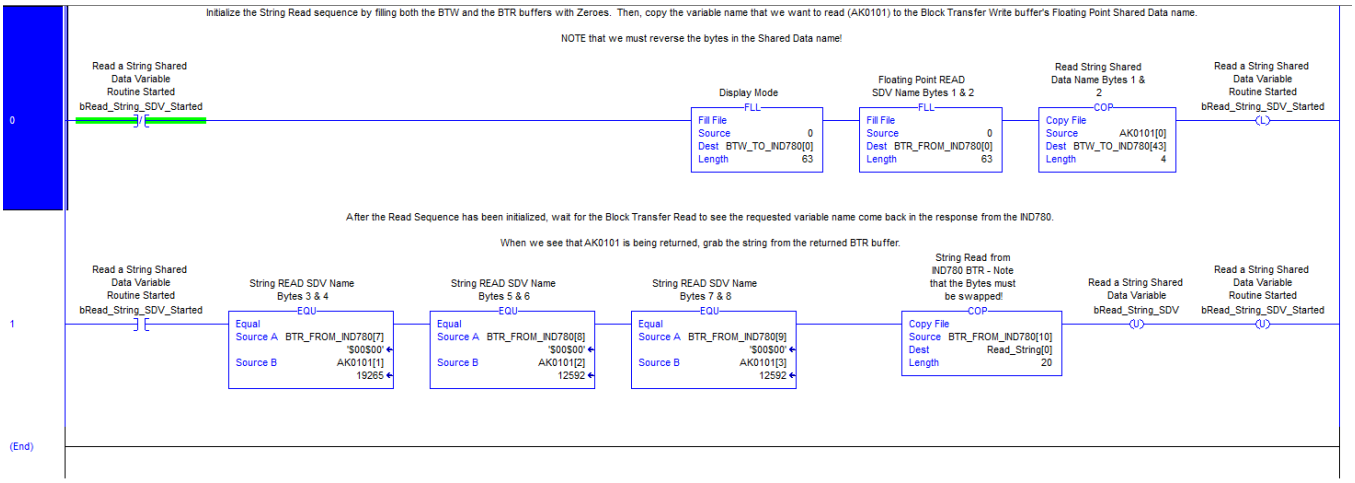
To Read a String variable, the PLC must first populate the BTW buffer starting at word **43** with the desired Shared Data Variable name (note the byte order). Again, since the Block Transfers are freewheeling, that is running constantly, nothing else needs to happen except to wait for the result from the IND780.



When the PLC reads that the requested Shared Data Variable name (AK0101) is present in the Block Transfer Read buffer (Starting at Word 6), it can now go get the resulting string.



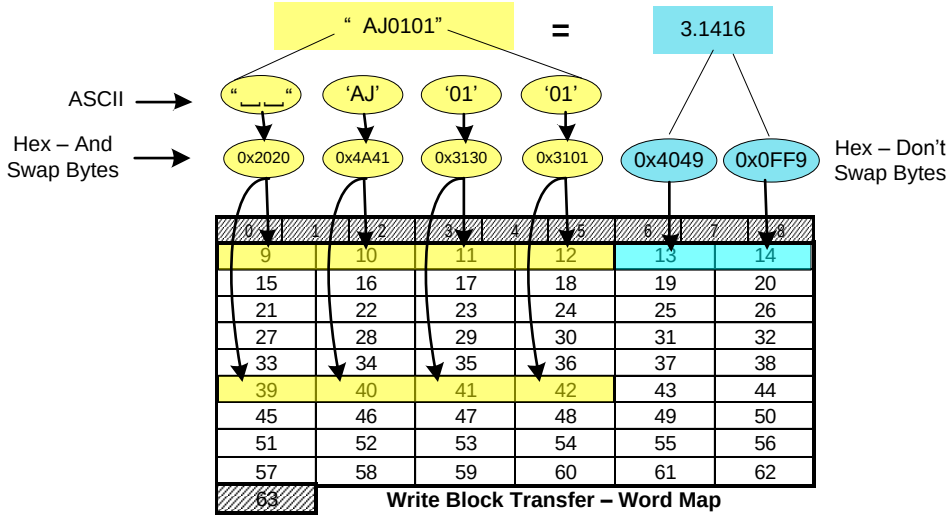
Below is the corresponding PLC code the performs the above sequence. Note the FLL instructions which zero out the BTR and BTW buffers before issuing the command.



The EQU instructions wait for the Block Transfer Read to show that the IND780 is returning the same variable name that was requested. When the compares pass, the result is copied into the String Read array, and the Read Request flag is cleared.

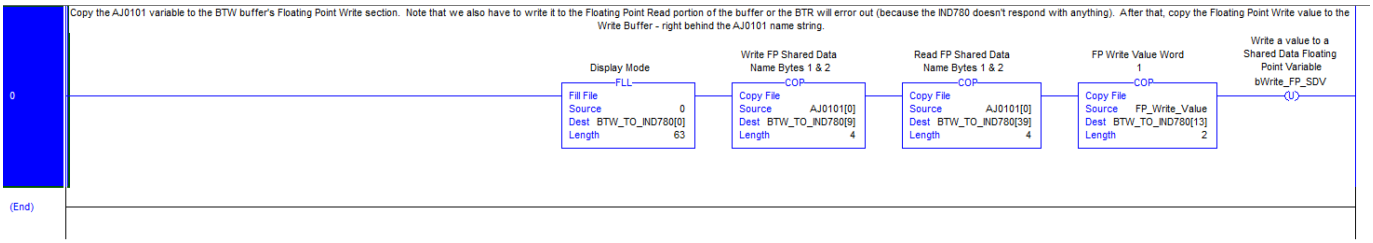
Write_Floating_Point_SDV Routine

The Write Floating Point sequence is more simple than the read because the only thing that needs to be done is to populate the Block Transfer Write Buffer with the Shared Data variable name and the Floating Point value desired as the diagram below shows. Note that the Shared Data Variable name is also copied to the Floating Point Read request at word 39.



IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

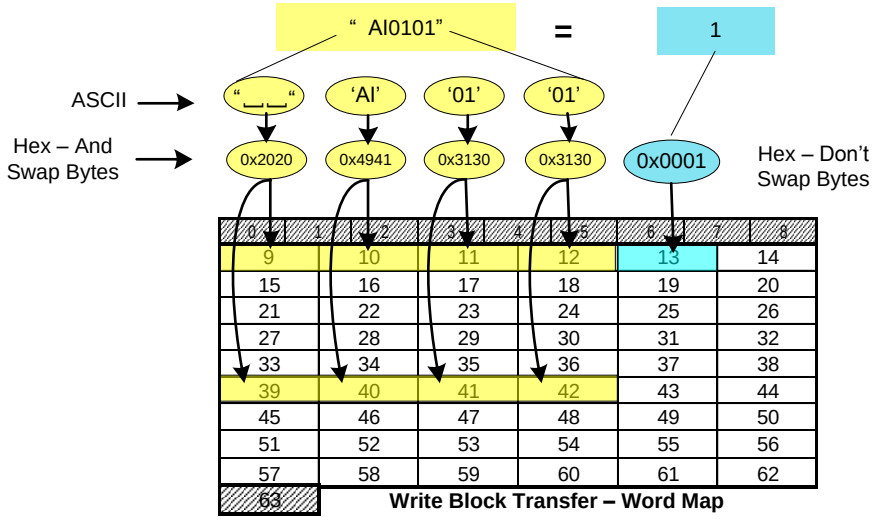
Below is the corresponding PLC code the performs the above sequence. Note the FLL instruction which zero out the BTW buffer before issuing the command. Also note that the Shared Data Variable name is copied to the Floating Point Read Request to prevent an Error on the subsequent Block Transfer Read.



Of course, it is probably advisable for the PLC program to read the Floating Point value back to ensure that the proper action has taken place. Since the Floating Point Shared Data variable has already been populated, this would be a fairly simple addition to the program (just read words 4 & 5 of the Block Transfer Read buffer).

Write_Integer_SDV Routine

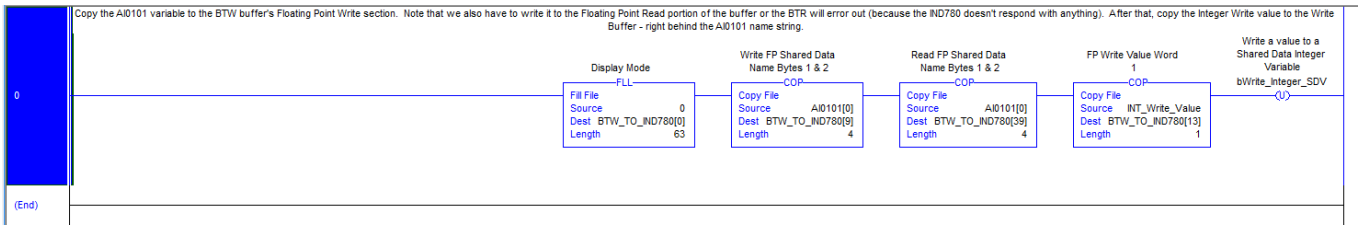
Again, the Write Integer sequence is simpler than the read because the only thing that needs to be done is to populate the Block Transfer Write Buffer with the Shared Data variable name and the Floating Point value desired as the diagram below shows. Note that the Shared Data Variable name is also copied to the Floating Point Read request at word 39.



Write Block Transfer – Word Map

IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

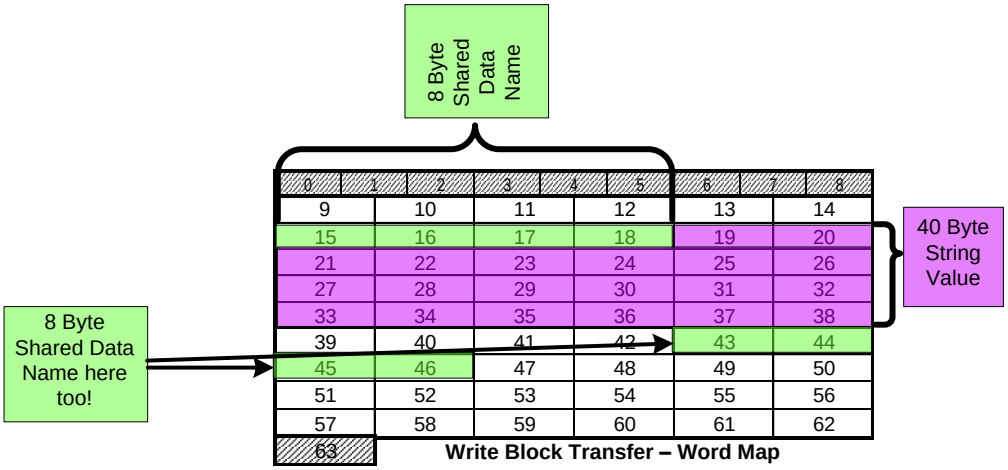
Below is the corresponding PLC code the performs the above sequence. Note the FLL instruction which zero out the BTW buffer before issuing the command. Also note that the Shared Data Variable name is copied to the Floating Point Read Request to prevent an Error on the subsequent Block Transfer Read.



Again, it is probably advisable for the PLC program to read the Integer value back to ensure that the proper action has taken place. Since the Floating Point Shared Data variable name has already been populated, this would be a fairly simple addition to the program (just read word 4 of the Block Transfer Read buffer).

Write_String_SDV Routine

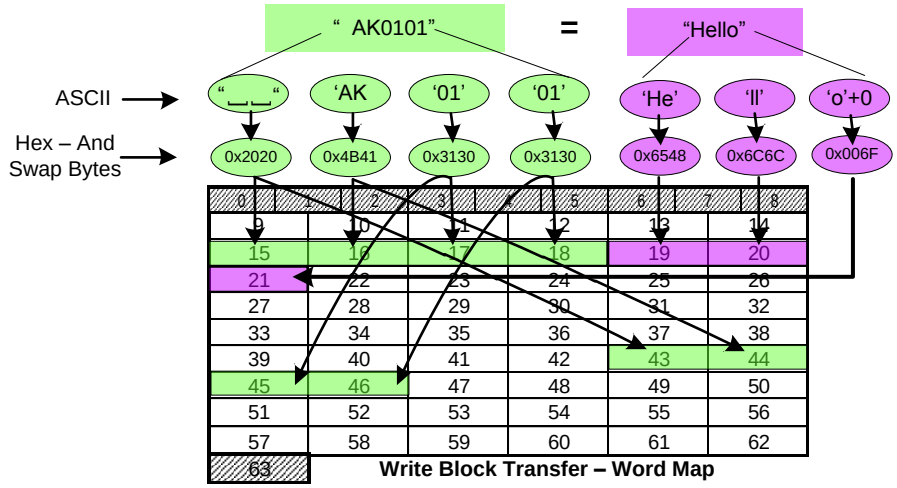
The General Form for the string Write looks like this.



The String Shared Data Variable name is put in starting at word 15. It also needs to be populated starting at word 43 so that the Block Transfer Read will not generate an Error. The string can be an array of up to 40 bytes starting at word 19 (note that all strings must have their bytes swapped within the word for the string to transmit correctly to the IND780).

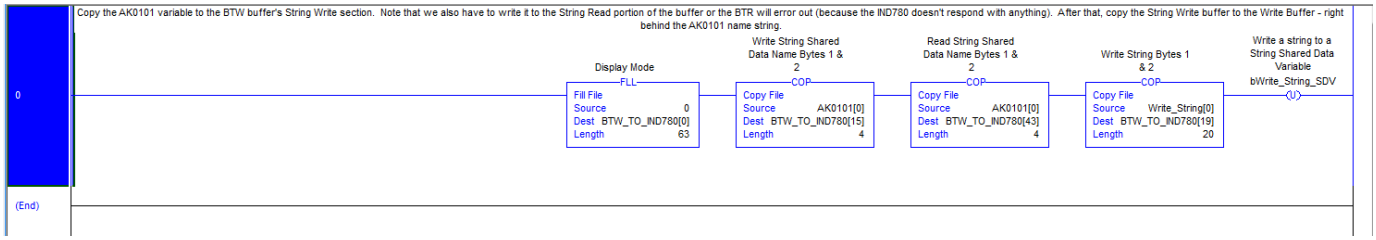
IND780 Shared Data Variable Access Using Allen-Bradley Remote I/O

Again, the Write String sequence is simpler than the read because the only thing that needs to be done is to populate the Block Transfer Write Buffer with the Shared Data variable name and the String Array desired as the diagram below shows.



Write Block Transfer – Word Map

Below is the corresponding PLC code to accomplish this.



Again, it is probably advisable for the PLC program to read the string back to ensure that the proper action has taken place. Since the String Shared Data variable name has already been populated, this data would already be available to the program in the Block Transfer Read Buffer starting at word 10.