RFID Function locks for Use with Omron PLCs

By Pepperl+Fuchs

Introduction:

These function blocks were designed for use with the Pepperl+Fuchs Ident*Control* RFID controllers. They can be used with all EtherNet/IP controllers. The function blocks can also be used on all EtherNet/IP-enabled Omron PLCs.

RFID Controller Compatibility:

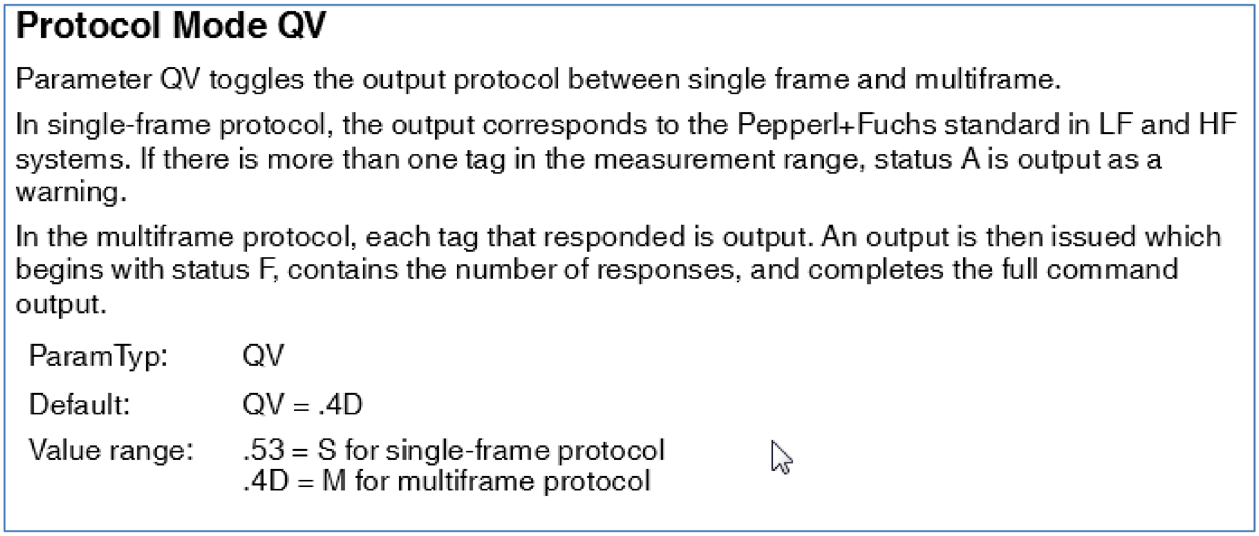
IC-KP-B17-AIDA1 – RFID controller for connection of up to 4 read/write heads

IC-KP2-2HB17-2V1D – RFID controller for connection of up to 2 read/write heads

IC-KP2-1HB17-2V1D – RFID controller for connection of up to 1 read/write head

Protocol:

These function blocks were written and tested with EtherNet/IP on an Omron PLC. They work with all IPH, low frequency, IQH, high frequency, and ISH read heads. They also work with IUH and UHF read heads as long as the single frame (SF) protocol is used. This protocol can be selected in the read heads parameter list by setting the parameter QV to “S”.



Protocol Mode QV

Parameter QV toggles the output protocol between single frame and multiframe.

In single-frame protocol, the output corresponds to the Pepperl+Fuchs standard in LF and HF systems. If there is more than one tag in the measurement range, status A is output as a warning.

In the multiframe protocol, each tag that responded is output. An output is then issued which begins with status F, contains the number of responses, and completes the full command output.

ParamTyp:	QV
Default:	QV = .4D
Value range:	.53 = S for single-frame protocol .4D = M for multiframe protocol

Figure 1: The setting of the protocol mode to “S” (single frame) in the IUH read head

Function Blocks:

Four function blocks have been written to minimize the ladder logic required to operate the Pepperl+Fuchs RFID system. Each one has a specific purpose. Use the PF_Set_Tag_Type function block on each read head any time a new controller is installed. Then pick the appropriate read or write function block that is required for your application. The function blocks available are:

PF_Set_Tag_Type – Sets the RFID chip type that will be used in the application

PF_Single_Read_Data – Reads the user-configurable data from a tag one time

PF_Single_Write_Date – Writes the user-configurable data to a tag one time

PF_Enhanced_Read_Data – Reads user-configurable data from a tag and remains running

Function Block General Operation:

All the function blocks operate similarly and there are certain parameters that are the same for all of them. The EN bit must remain on all the time. Put the trigger bit to low and wait until the DN and ER bits are also low. Now you are ready to start. Raise the trigger bit to activate the function block. When the DN bit is on, the function block has completed the operation. To reactivate the function block, make the trigger bit low and start the sequence again.

EN (Input, BOOL)

The enable bit must be on and remain on continuously.

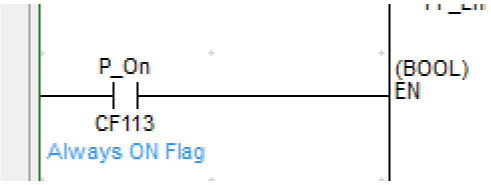


Figure 2: EN parameter must always be on

Assembly (Input, UINT)

The assembly instance input parameter must match exactly the assembly instance to which the RFID controller has been configured. This input parameter tells the function block how to lay out the memory map for the four heads. In Figure 4, assembly instance 107/157 has been configured. The assembly instance input parameter is the larger number of 157 as seen in Figure 3.

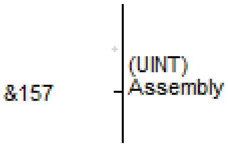
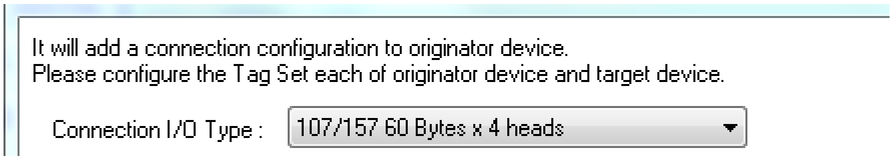


Figure 3: Assembly Instance Input Parameter

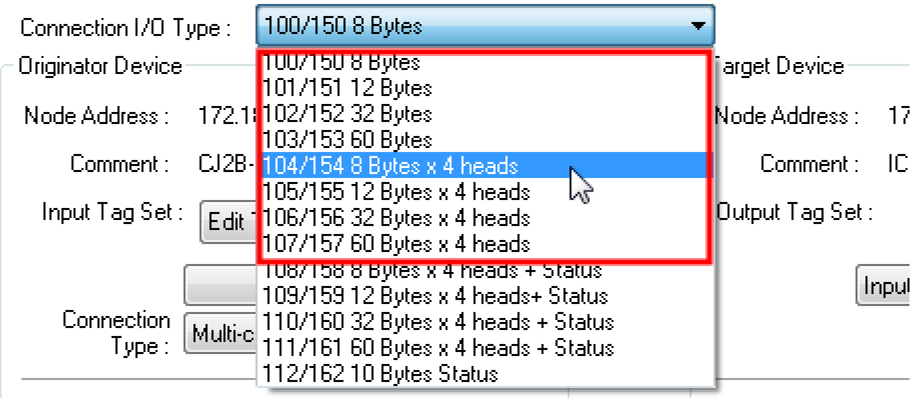


It will add a connection configuration to originator device.
Please configure the Tag Set each of originator device and target device.

Connection I/O Type : 107/157 60 Bytes x 4 heads

Figure 4: EtherNet/IP Connection IO Configuration of the RFID Controller

Only the assembly instances from 100/150 up to 107/157 are allowed to be used with the function blocks. We recommend that if 2- or 4-channel RFID controllers are used, then assembly instances 104/154 up to 107/157 should be used. These assembly instances map each read head in its own memory space. This is the quickest and easiest way to program multiple read heads at the same time.



Connection I/O Type : 100/150 8 Bytes

Originator Device
Node Address : 172.1
Comment : CJ2B-
Input Tag Set : Edit
Connection Type : Multi-c
100/150 8 Bytes
101/151 12 Bytes
102/152 32 Bytes
103/153 60 Bytes
104/154 8 Bytes x 4 heads
105/155 12 Bytes x 4 heads
106/156 32 Bytes x 4 heads
107/157 60 Bytes x 4 heads
108/158 8 Bytes x 4 heads + Status
109/159 12 Bytes x 4 heads+ Status
110/160 32 Bytes x 4 heads + Status
111/161 60 Bytes x 4 heads + Status
112/162 10 Bytes Status
arget Device
Node Address : 17
Comment : IC
Output Tag Set :
Inpu

Figure 5: Only 100/150 up to 107/157 can be used with these function blocks

Head_Number (Input, UINT)

The head number or channel number is the read head number that the function block acts on. Only head numbers 1, 2, 3, or 4 are allowed.

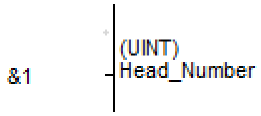


Figure 6: Read head that the function block acts on

Trigger (Input, BOOL)

The trigger bit controls the operation of a function block. Before execution, the trigger bit must be low. When you want to turn the function block on, turn this bit on. This bit must remain on for the entire time that the function block is executing. The DN bit comes on when the function block stops execution.

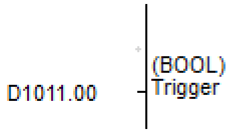


Figure 7: Trigger bit to enable and control the execution of the function block

DN (Output, BOOL)

This bit goes on when the function block completes its operation. This bit goes off when the trigger bit is turned off. When the function block is complete, success or failure, then this bit is turned on. To see if the function block completed without errors, monitor the ER bit.

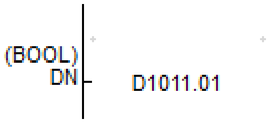


Figure 8: DN bit tells when the function block has completed

ER (Output, BOOL)

The ER bit is the error bit. It will come on at the same time as the DN bit if the function block completed with an error. For a more detailed explanation of the error, check the status.

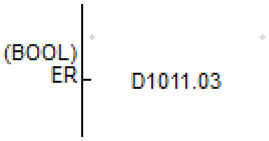


Figure 9: ER bit shows if the function block completed with errors

Status (Output, UINT)

The status gives you complete information about the running function block throughout the execution of the block. If the function block results in an error, this status will tell you the cause. The table below lists the most common errors. Some of the errors are generated by the function block and not the read head or controller. For a more complete list of errors, refer to the RFID controller manuals.

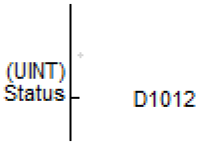


Figure 10: Status generated by the RFID system and the function block

Status	Description	Solution
0	Success	
4	Bad command	Check all the parameters of the FB
5	No tag present	Put tag in front of head and try again
6	No head	Check read head connection and tag type setting
255	Command pending	
254	Timeout	Controller may not be connected or the memory input parameters are wrong
253	Bad parameter	Check whether head number and assembly instance are in range

Output (IN/OUT, UINT address)

This is the memory location where the RFID controller's output data is mapped. It must match the memory mapping of the controller. Check the EtherNet/IP network configuration software to get the address.

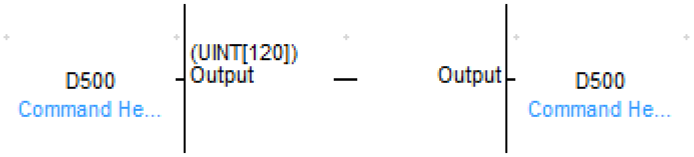
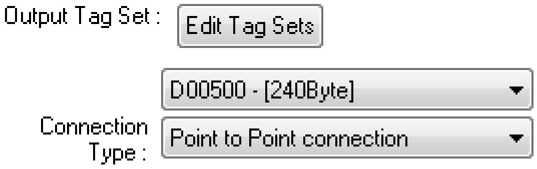


Figure 11: Output address/data where the RFID controller is mapped



Output Tag Set : D00500 - [240Byte]

Connection Type : Point to Point connection

Figure 12: Output data mapping of the RFID controller in the EtherNet/IP configuration software

Input (IN/OUT, UINT address)

This is the memory location where the RFID controller's input data is mapped. It must match the memory mapping of the controller. Check the EtherNet/IP network configuration software to get the address.

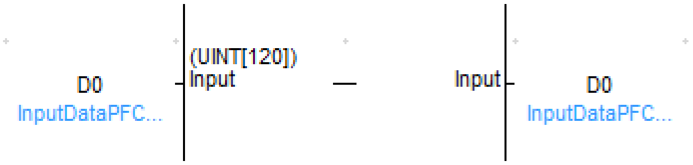
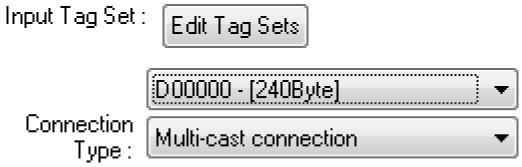


Figure 13: Input address/data where the RFID controller is mapped



Input Tag Set : D00000 - [240Byte]

Connection Type : Multi-cast connection

Figure 14: Input data mapping of the RFID controller in the EtherNet/IP configuration software

PF_Set_Tag_Type Specific Operation

This function block is designed to set the tag type of a read head. Setting the tag type lets the read head know what kind of RFID tag or chip will be read or written before the tag is put in front of the read head. With this knowledge, you can achieve the fastest communication method. In certain cases, the tag type must be configured before correct operation is possible. The specific port number of the controller you are communicating to is configured at the same time. This allows a read head to be swapped out and configured automatically. This command need only be sent once for each head of a new controller and then after that only when a controller is swapped out. For the sake of simplicity, many people just send this instruction when the PLC program starts up.

The EN bit must be enabled all the time. The trigger bit stays off before execution. The function block starts running as soon as the trigger bit transitions from off to on. The trigger bit must remain on for the entire time the function block is executing.

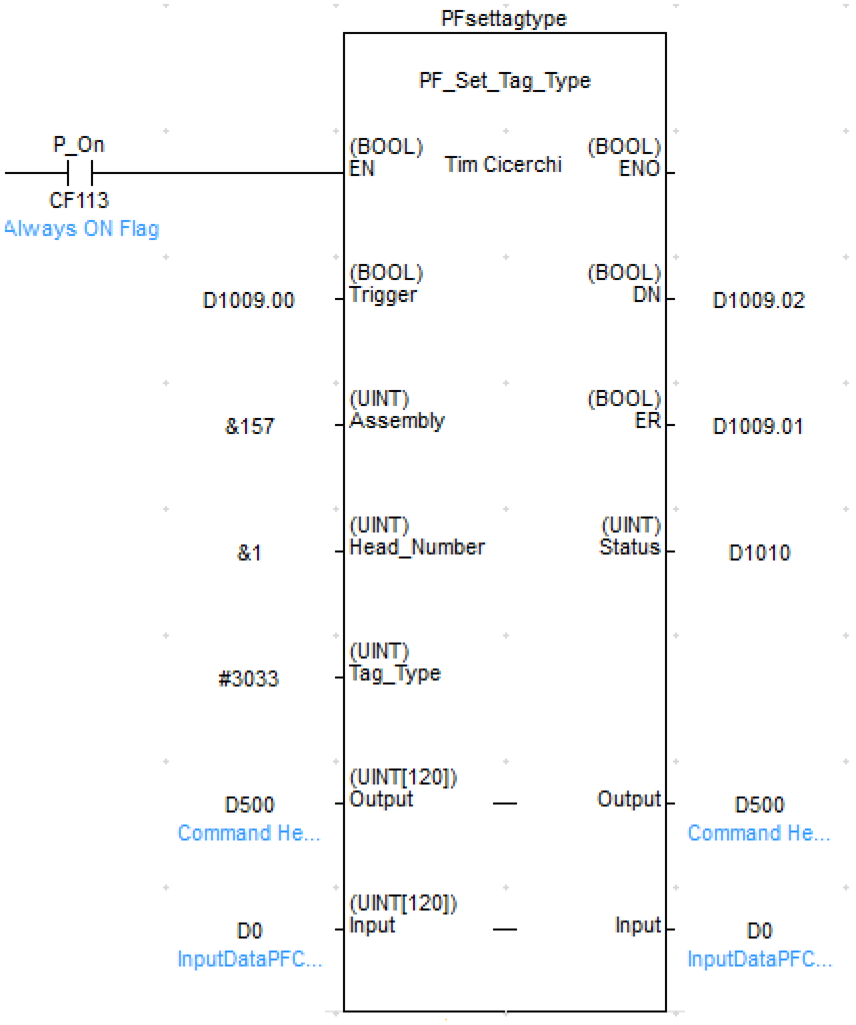


Figure 15: Set tag type function block

Tag_Type (Input, UINT)

The Set_Tag_Type specific parameter is most easily entered in HEX. The hex number is equivalent to the model number of the RFID tag in ASCII. Here is a list of the most common tag types.

RFID Tag Model Number	Tag_Type Parameter
IPC02	#3032
IPC03	#3033
IPC11	#3131
20	#3230

IQC21	#3231
IQC22	#3232
IQC24	#3234
IQC31	#3331
IQC33	#3333
IDC-1K	#3530
ICC	#3532
IUC	#3830

PF_Single_Read_Data Specific Operation

Once enabled, this function block attempts to read the data from the RFID tag one time. If the read is successful, no ER bit goes on, and the data is placed in the array called Data. The length of the data in the array depends on the length of the requested data. Two array elements of data are read for every 1 length parameter. This is because the length is a parameter of how many DINTs are read and the array data is an array of UINTs.

The EN bit must be enabled all the time. The trigger bit stays off before execution. The function block starts running as soon as the trigger bit transitions from off to on. The trigger bit must remain on for the entire time the function block is executing.

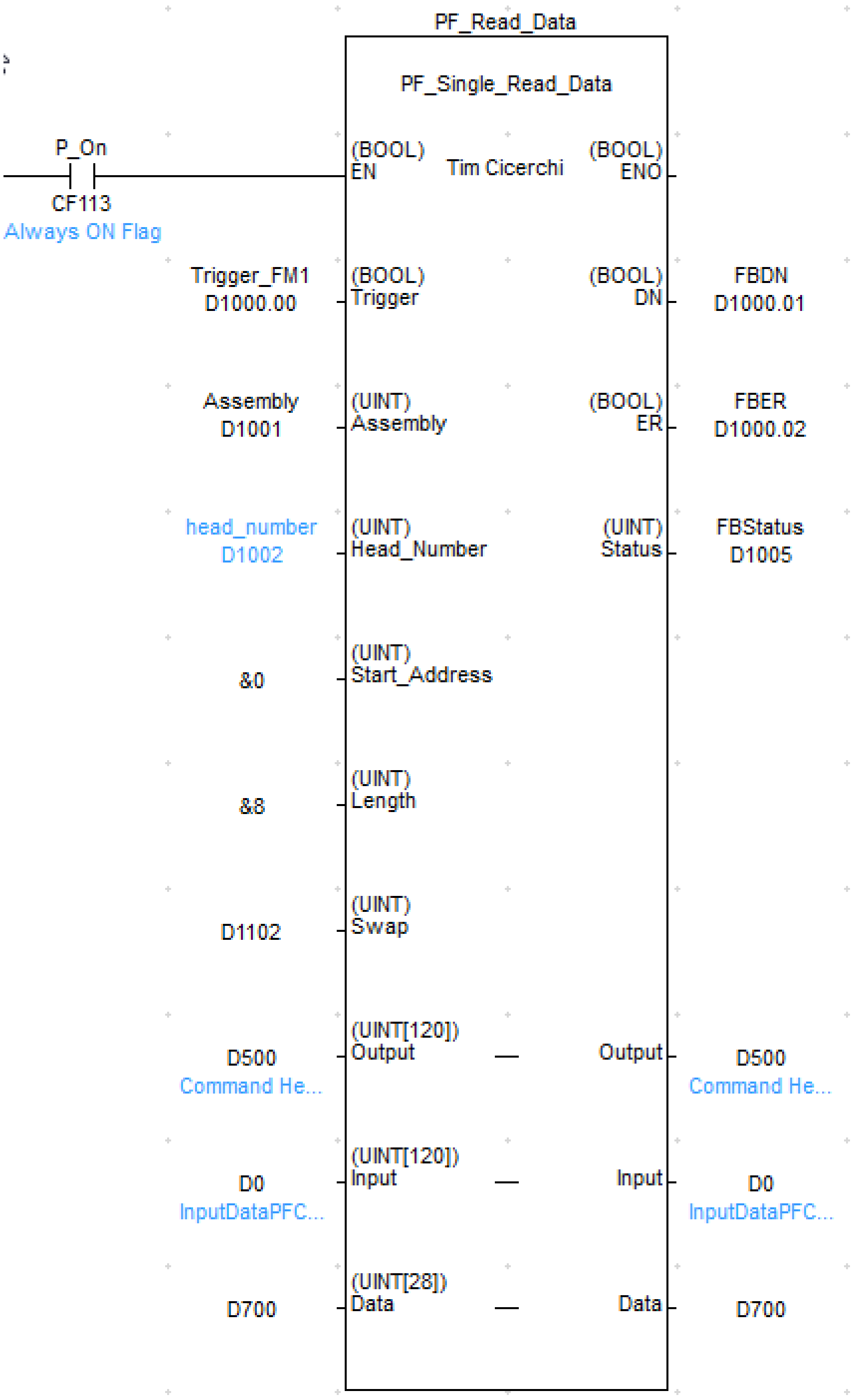


Figure 16: Single read function block

Start_Address (Input, UINT)

This is the block address that you want to start reading on the tag.

Length (Input, UINT)

The number of DINTs or 4-byte blocks that you want to read. The length must be less than or equal to the assembly instance you are using.

Swap (Input, UINT)

This parameter reads the data and swaps the bytes within the integer.

0=no swap, 1=swap

Assembly Instance	Lax Length (4-byte blocks)
100/150 or 104/154	1
101/151 or 105/155	2
102/152 or 106/156	7
103/153 or 107/157	14

Certain tags, because of their block size, are only allowed certain lengths. Tag type IQC33 is only allowed to have lengths of 2, 4, 6, 8, 10, 12, or 14, so in this case, we do not recommend that you use 100/150 or 104/154. Also, tag types with 32-byte blocks like the IQC37 can only have a length of 8 and only 103/153 or 107/157 can be used.

Data (In/OUT, UINT[28])

This is the array where the data is placed when the function block executes successfully. A length of 1 fills two UINTs, a length of 2 fills 4 UINTS, and so on. All unused array elements are filled with zeros.

PF_Single_Write_Data Specific Operation

This function block, once enabled, attempts to write the data to the RFID tag one time. If the write is successful, no ER bit goes on. The data is taken from the array called Data, and written to the tag. The length of the data written from the array depends on the length of the requested data. Two array elements of data are written for every 1 length parameter. This is because the length is a parameter of how many DINTs are read and the array called Data is an array of UINTs.

The EN bit must be enabled all the time. The trigger bit stays off before execution. The function block starts running as soon as the trigger bit transitions from off to on. <u>The trigger bit must remain on for the entire time the function block is executing</u>.

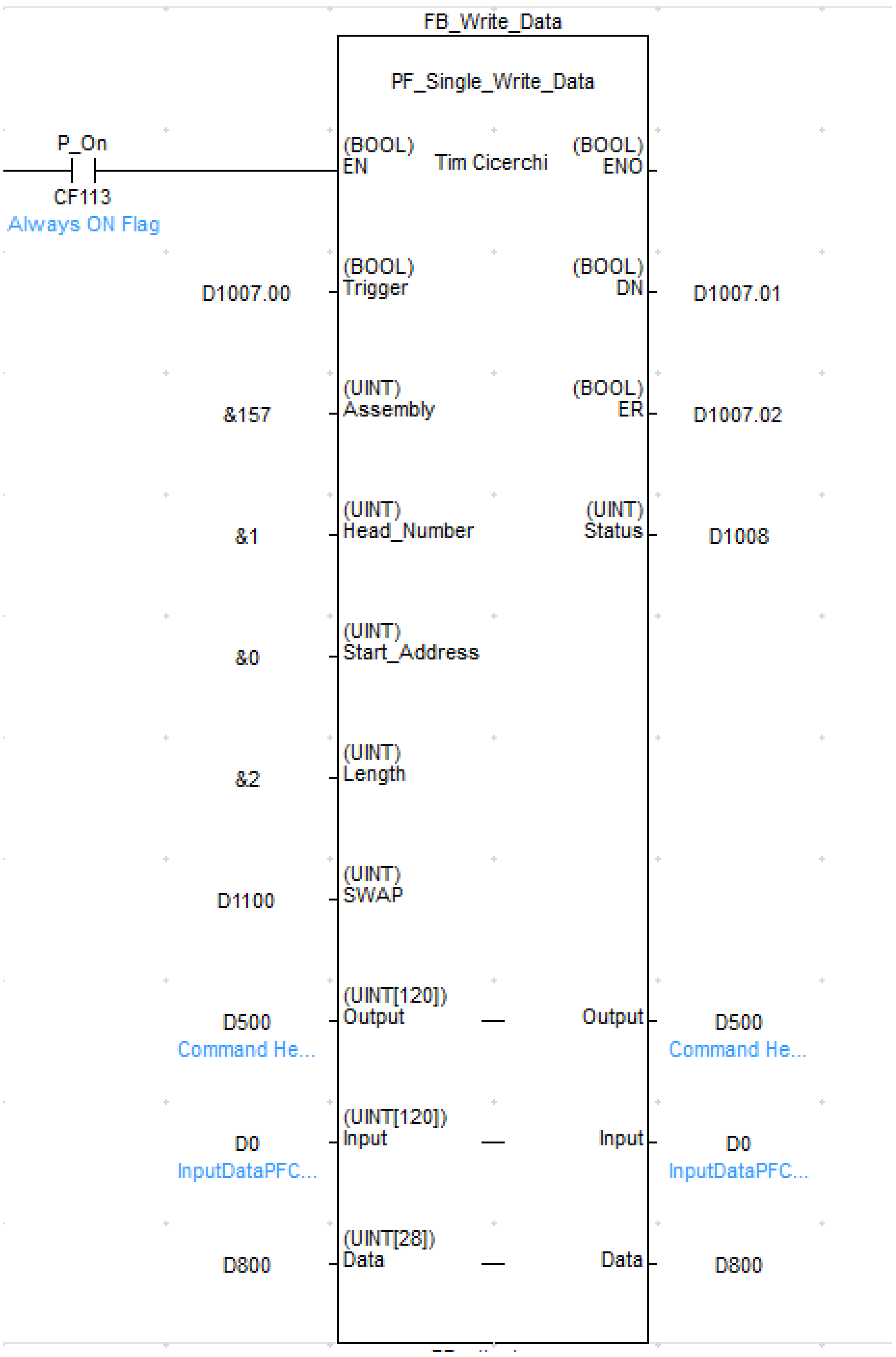


Figure 17: Function block to write data to a tag one time

Start_Address (Input, UINT)

This is the block address that you want to start writing to the tag.

Length (Input, UINT)

The number of DINTs or 4-byte blocks that you want to write. The length must be less than or equal to the assembly instance you are using.

Assembly Instance	Lax Length (4-byte blocks)
100/150 or 104/154	1
101/151 or 105/155	2
102/152 or 106/156	7
103/153 or 107/157	14

Certain tags, because of their block size, are only allowed certain lengths. Tag type IQC33 is only allowed to have lengths of 2, 4, 6, 8, 10, 12, or 14, so in this case, we do not recommend that you use 100/150 or 104/154. Also, tag types with 32-byte blocks like the IQC37 can only have a length of 8 and only 103/153 or 107/157 can be used.

Swap (Input, UINT)

This parameter reads the data and swaps the bytes within the integer.

0=no swap, 1=swap

Data (In/OUT, UINT[28])

This is the array where the data will be pulled from when the function block executes successfully. A length of 1 writes two UINTs, a length of 2 writes 4 UINTS, and so on.

PF_Enhanced_Read_Data Specific Operation

This function block is designed to read data and remain running continuously. It is often used to read tags that are passing the read head on the fly. As a new tag appears, the data is placed in the data array. When the tag leaves the read zone, a status 5 is sent and the data array is cleared. The data remains in the input image and in the data array for as long as specified in the DataRetentionTime setting. This EtherNet/IP parameter can be changed on the controller's web interface or on its graphical display. The EN bit and the trigger bit must both remain on continuously. As long as the DN and ER bits are off and the running bit is on, the function block is operating normally.

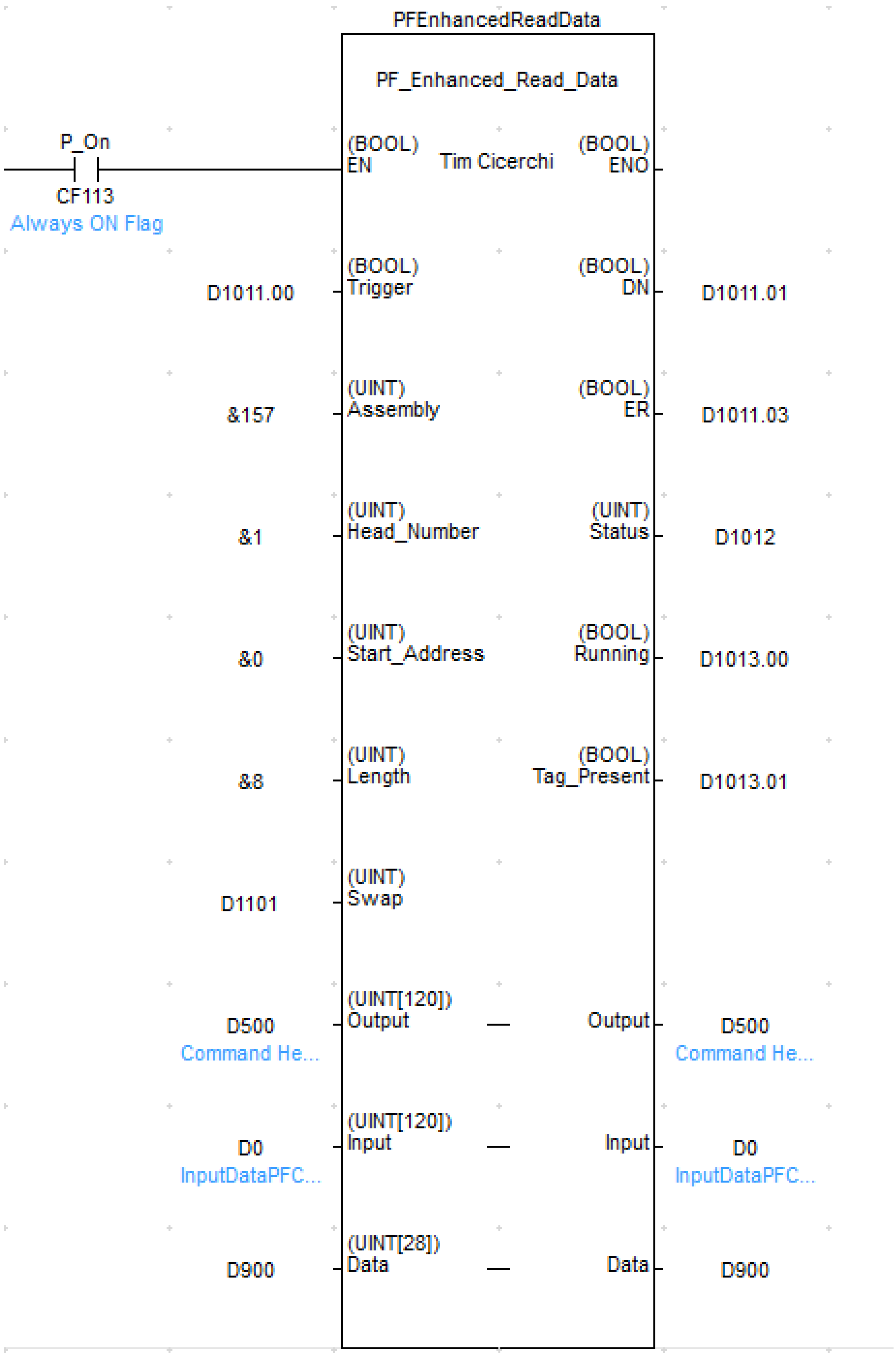


Figure 18: Enhanced read function block that can read tags on the fly and continuously

Start_Address (Input, UINT)

This is the block address that you want to start writing to the tag.

Length (Input, UINT)

The number of DINTs or 4-byte blocks that you want to write. The length must be less than or equal to the assembly instance you are using.

Assembly Instance	Lax Length (4-byte blocks)
100/150 or 104/154	1
101/151 or 105/155	2
102/152 or 106/156	7
103/153 or 107/157	14

Certain tags, because of their block size, are only allowed certain lengths. Tag type IQC33 is only allowed to have lengths of 2, 4, 6, 8, 10, 12, or 14, so in this case, we do not recommend that you use 100/150 or 104/154. Also, tag types with 32-byte blocks like the IQC37 can only have a length of 8 and only 103/153 or 107/157 can be used.

Swap (Input, UINT)

This parameter reads the data and swaps the bytes within the integer.

0=no swap, 1=swap

Data (In/OUT, UINT[28])

This is the array where the data is pulled from when the function block executes successfully. A length of 1 writes two UINTs, a length of 2 writes 4 UINTS, and so on.

Running (OUT, BOOL)

The input image is checked to see if an enhanced read command is running or not. The read will stop, for example, if a write command is sent after the enhanced read command is enabled. To reenable the enhanced read command, make the trigger low, wait for the DN and ER bits to go off, and raise the trigger bit again.

1=Command Running, 0=Command Not running

Tag_Present (OUT, BOOL)

Shows if the tag is in front of the read head or not.

0=tag not present, status is 5

1=tag present, status is 0