DATALOGIC™

DAD / DPD Driver

Reference Manual

Datalogic Automation Srl
Via S. Vitalino, 13
40012 - Lippo di Calderara di Reno
Bologna - Italy

DAD / DPD Driver Reference Manual

Ed.: 11/2009

23/11/09

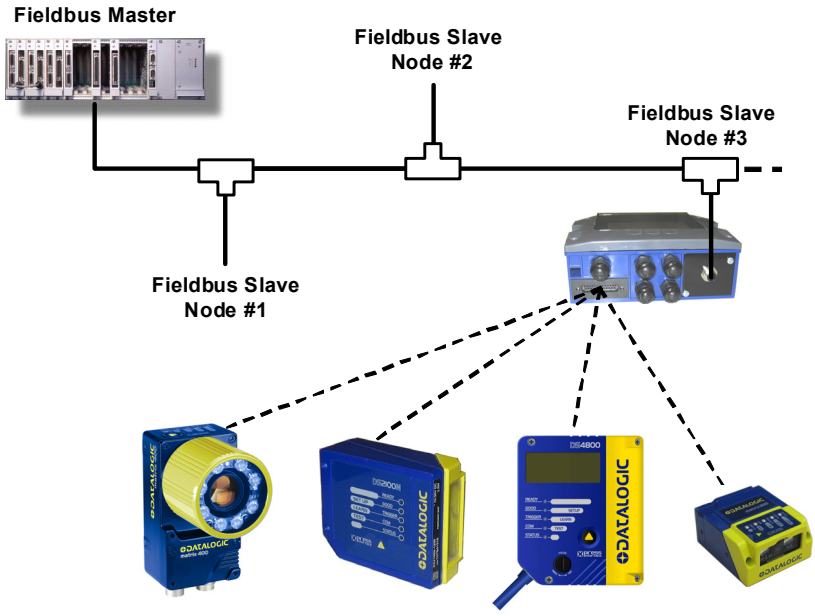
CONTENTS

1 FIELDBUS COMMUNICATION

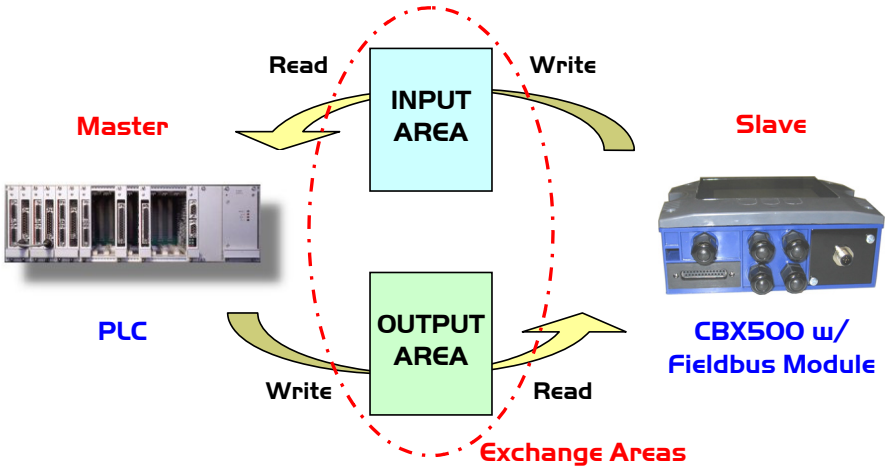
1.1 DATA EXCHANGE

The Fieldbus ***Master*** is usually a PLC (Siemens S7 or other). Sometimes it could be a PC-based device as well.

CBX500, CBX800 or SC4000 with a Fieldbus Module is always a ***Slave*** in the Fieldbus network.



Basically two shared memory areas (Exchange Areas) exist between SLAVE and MASTER so both devices provide information to each other. Exchange areas are physically placed in the Fieldbus Module inside the SLAVE.



Input and output areas always refer to the Master: this means that the slave writes to the Input buffer and the PLC writes to the Output buffer.

Dimensions of exchange areas can be set to different values by the PLC through the specific Fieldbus Configuration file (i.e. GSD file for Profibus) or by the slave's configuration program.

1.2 DATALOGIC FLOW CONTROL MODE (FCM)

The Datalogic Flow Control Mode is a powerful way to manage and optimise the communication with the Fieldbus Master. By enabling the FCM a few bytes of the exchange areas are reserved for driver operations and the rest are used by the application layer.

The reserved bytes are used to implement many different features such as:

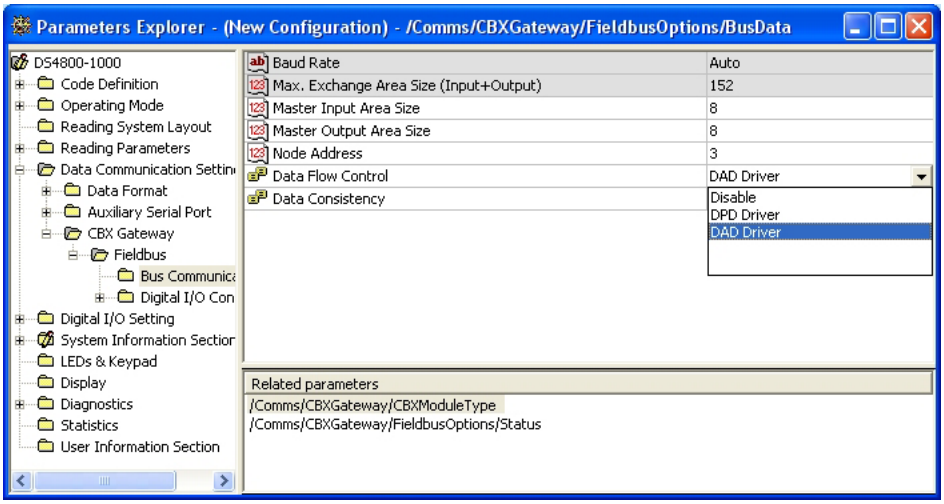
- Flow-control and corresponding buffering in both directions
- Fragmentation and reassembling of data longer than the exchange area sizes
- Synchronisation of flow control numbers
- Service Access Point oriented communication
- Length information

NOTE

If the Datalogic Flow Control is disabled, all the bytes of the exchange areas are used by the application layer. The input area is updated whenever a new reading event has to be transferred to the Master station.

- **In this situation, the Master must read the input area before it changes due to a new message, typically new barcode occurrence.**
- **Moreover, two occurrences of the same barcode cannot be understood, since the input area does not change.**
- **In addition, if application data is longer than input area sizes, data is automatically truncated.**

FCM can be selected by means of the Data Flow Control parameter through the reader configuration program (i.e. Genius™ or VisiSet™).



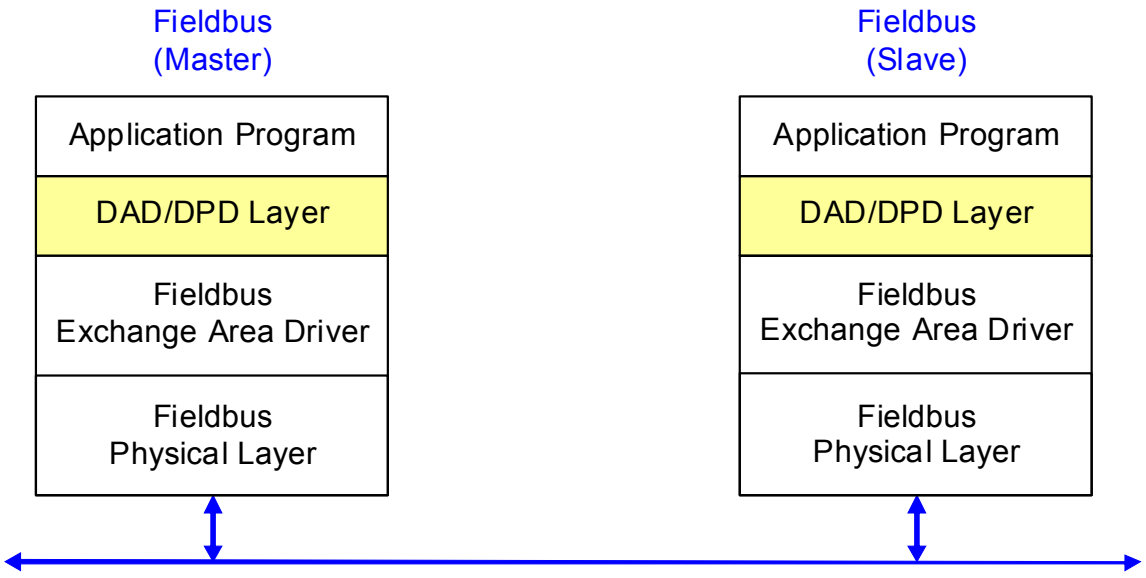
Basically three options are available:

- Disable → *No Datalogic Flow Control*
- DAD Driver → *Flow Control Enabled*
- DPD Driver → *Flow Control Enabled*

1.3 FLOW CONTROL DRIVERS

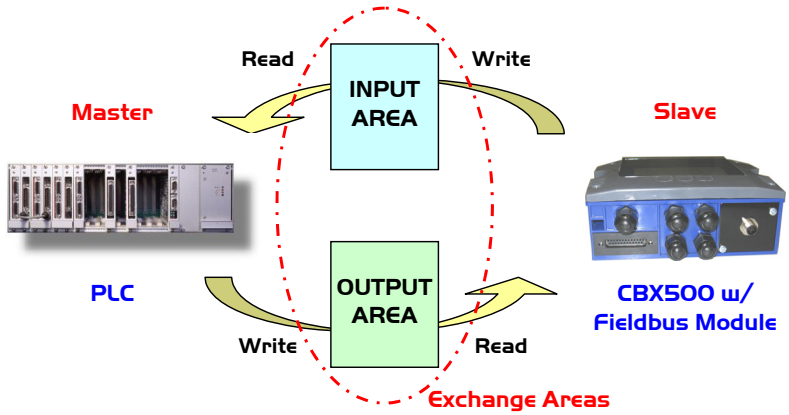
The Flow Control driver is a layer that is built upon the intrinsic data exchange mechanism. Basically such a layer is required because the intrinsic data exchange mechanism is not message oriented.

In the following figure the complete Stack is represented:



As described previously, it can be selected separately on the reader by means of the reader configuration tool. Obviously, the corresponding driver must be implemented on the Fieldbus Master (PLC) side.

2 FCM WITH DAD DRIVER



From now on we refer to the Input Area as a buffer made up of InputAreaSize bytes:

IN[0]	IN[1]	IN[2]	…	IN[InputAreaSize - 1]

and to the Output Area as a buffer made up of OutputAreaSize bytes:

OUT[0]	OUT[1]	OUT[2]	…	OUT[OutputAreaSize - 1]

Only the first **three** bytes are used by the DAD Driver layer in both buffers:

Control Field (byte 0) — used to issue and control the driver primitives such as flow-control, fragmentation and resynchronisation.

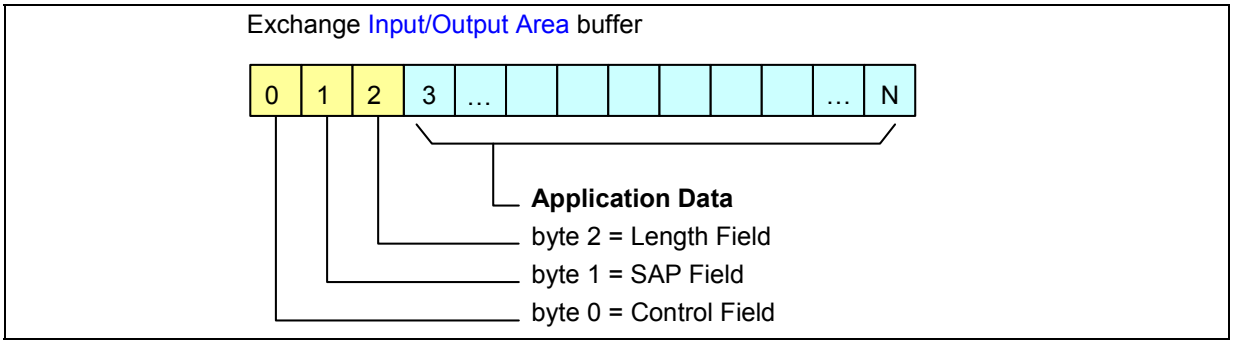
Service Access Point Field (byte 1) — used to distinguish among different services and to provide future expandability. (Since this SAP definition is introduced by the DAD Driver, it must not be confused with the SAP that is defined by the international standard).

Length Field (byte 2) — contains the number of bytes used by the application layer: *(see also note below)*
Length Field ≤ (InputAreaSize – 3) for the Input Area
Length Field ≤ (OutputAreaSize – 3) for the Output Area.

NOTE

If Data Consistency is enabled then the last byte is used for packet integrity and therefore the bytes available to the application are reduced by 1. See chapter 4 of this manual for details.

If Digital I/O Conditioning is used to echo reader Inputs to the Fieldbus Master or control reader Outputs by the Fieldbus Master, then Byte 0 is reserved for this purpose and therefore the DAD Driver occupies bytes 1 to 3. This also reduces the bytes available to the application by 1. See chapter 5 of this manual and Digital I/O Conditioning in the Configuration Parameters Help On Line.



The Application Data buffer holds useful information, typically the barcode messages, processed by the application program. IN[3] contains the first significant byte of the Application Data buffer (the same first byte you would see if the reader transmitted the barcode buffer onto a serial port instead of the Fieldbus interface).

The structure of the application buffer and its length strictly depend on the selected data format on the reader. Barcode messages longer than (InputAreaSize – 3) will be split in pieces through an automatic fragmentation process (see details in the "Fragmentation and Reassembling" paragraph).

2.1 CONTROL FIELD

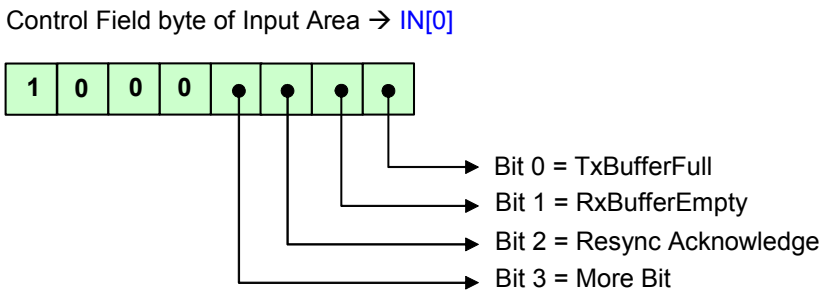
This is the core of the flow controlled communication.

The Input Area structure reserves bit 0 and bit 1 of IN[0] for handshake purposes while the Output Area structure, which is symmetrical, reserves bit 0 and 1 of OUT[0].

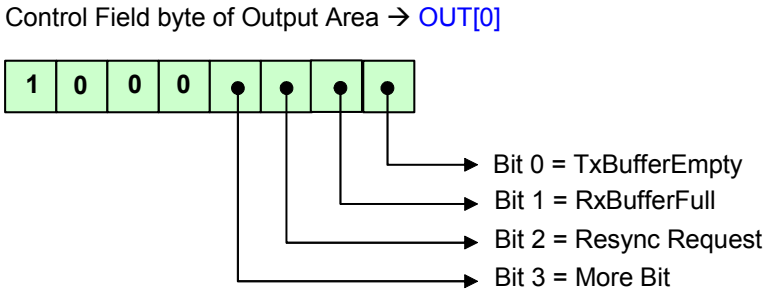
At any time the Master station can make a resynchronization request by means of bit 2 of the Output Area. This process, which resets the synchronization numbers (bit 0 and bit 1 of both Input and Output areas), has to be acknowledged by the Slave on bit 2 of the Input Area.

Bit 3 is used to control a fragmentation sequence in both directions.

Bit 7 is always set to 1.



IN[0].bit0	**TxBufferFull** toggles when Slave has made available new Input Area data
IN[0].bit1	**RxBufferEmpty** toggles when Output Area data has been read by Slave
IN[0].bit2	**Resync Acknowledge** set to 1 as an acknowledge to a resync request. With this bit, the master can detect a slave is on line.
IN[0].bit3	**More Bit** is 1 when this is not the last piece of a fragmentation sequence while it is 0 when this is the last piece
IN[0].bit4,5,6,7	Set to **0**, **0**, **0**, **1** when DAD messaging protocol is used



OUT[0].bit0	**TxBufferEmpty** must toggle when Input Area data has been read by Master
OUT[0].bit1	**RxBufferFull** must toggle when Master makes available new Output Area data
OUT[0].bit2	**Resync Request** set to 1 for one second to resynchronize the slave. After resynchronization, all 4 handshake bits are set to 0
OUT[0].bit3	**More Bit** must be 1 when this is not the last piece of a fragmentation sequence and it must be 0 when this is the last
OUT[0].bit4,5,6,7	Set to **0**, **0**, **0**, **1** when DAD messaging protocol is used

2.2 SAP FIELD

SAP (Service Access Point) is an identifier that is used to implement multiple services sharing the same communication channel between two remote stations.
The following values have been defined:

- SAP = 0 Used to transfer information messages between the reader and the PLC
- SAP = 255 Reserved for driver services (see details in paragraph 2.8)

All other SAP values are free and they could be used by dedicated application programs after agreement between the application programs themselves.

2.3 LENGTH FIELD

The Application layer uses all or a part of the remaining bytes of the Exchange Area buffers that are not used by the DAD Driver. The Length Field is introduced to keep the information of how many bytes are really used by the Application Layer.

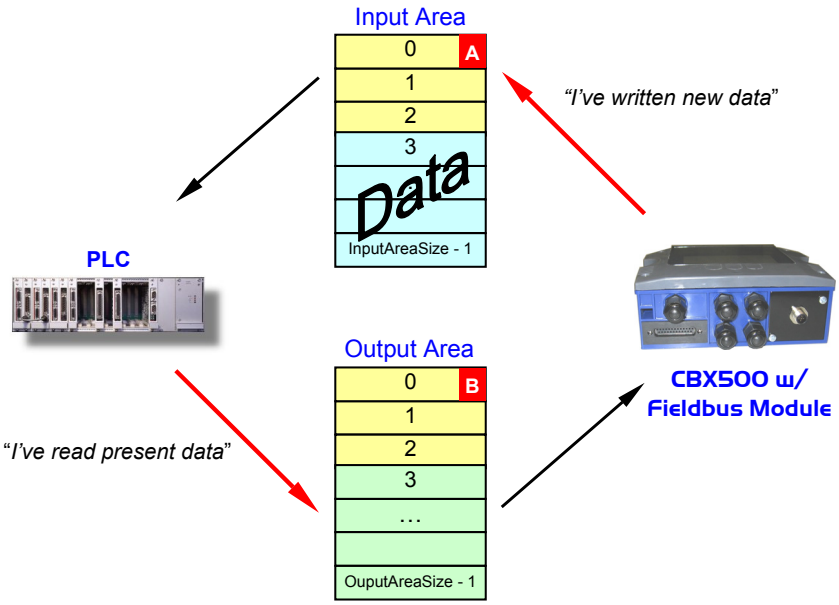
A fragment that is not the last one of a fragmentation sequence must fill this field with [InputAreaSize – 3] (or [OutputAreaSize - 3]), depending on whether it is an Input/Output fragment. Otherwise this field gets a value that is less than or equal to [InputAreaSize – 3].

2.4 DATA TRANSMISSION FROM THE READER TO PLC

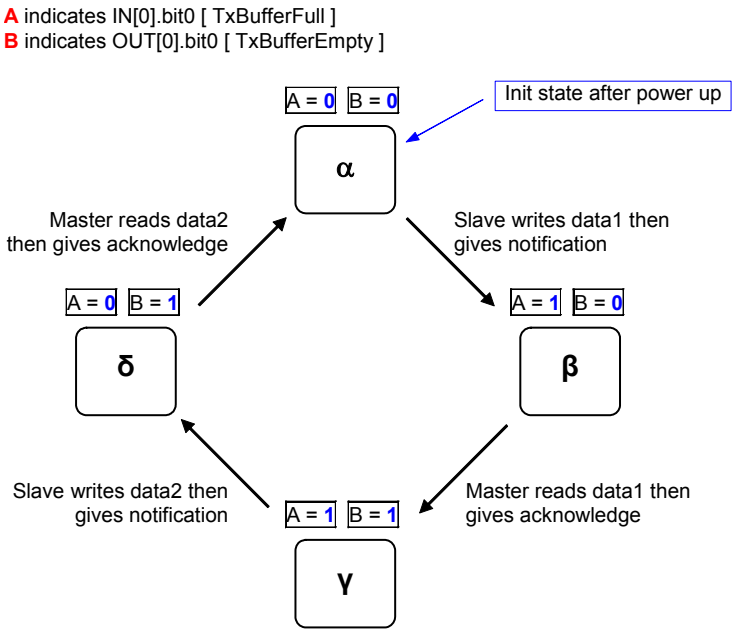
This paragraph describes how it is possible to exchange messages with flow control. The communication mechanism is simple:

- **IN[0].bit0** [**A**] is used by the reader to notify that "*Slave has written a new data so Master can read it*"
- **OUT[0].bit0** [**B**] must be used by PLC to notify that "*Master has read last data so Slave can send next message*"

This happens each time bit A (or B) changes its state (toggles). Bit level doesn't matter, only the transition has to be considered.



The following state machine shows data transmission from Slave to Master. Please note that each cycle transfers two data messages.



Let's analyse a typical data exchange based on the following settings:

- Flow Control = **DAD Driver**
- Input Area Size = **16**
- Output Area Size = **8**

1. After power up Input and Output areas are generally filled by zero. According to DAD driver implementation, Input area has Control Field = 80_{Hex} and SAP = 00_{Hex}.

Input Area	**80**$_{Hex}$	**00**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$							

2. Also PLC must set the Control Field of Output area properly, as long as DAD messaging protocol is utilised.

Input Area	80$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	**80**$_{Hex}$	**00**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$							

3. The reader reads a barcode "123456". Let's assume standard data formatting with <STX> as header and <CR><LF> as terminators. the reader toggles bit **A**.

Input Area	**81**$_{Hex}$	00$_{Hex}$	**09**$_{Hex}$	**02**$_{Hex}$	**31**$_{Hex}$	**32**$_{Hex}$	**33**$_{Hex}$	**34**$_{Hex}$	**35**$_{Hex}$	**36**$_{Hex}$	**0D**$_{Hex}$	**0A**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	80$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

4. PLC detects transition of bit **A** so now it can read incoming data (it copies 9 bytes in its memory from IN[3] on) then toggles bit **B** as acknowledge.
   ***Note**: before the acknowledge, all further barcodes read by the reader are buffered.*

Input Area	81$_{Hex}$	00$_{Hex}$	09$_{Hex}$	02$_{Hex}$	31$_{Hex}$	32$_{Hex}$	33$_{Hex}$	34$_{Hex}$	35$_{Hex}$	36$_{Hex}$	0D$_{Hex}$	0A$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	**81**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

5. The reader reads a barcode "10DL" and toggles bit **A**.

Input Area	**80**$_{Hex}$	00$_{Hex}$	**07**$_{Hex}$	**02**$_{Hex}$	**31**$_{Hex}$	**30**$_{Hex}$	**44**$_{Hex}$	**4C**$_{Hex}$	**0D**$_{Hex}$	**0A**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	81$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

6. PLC reads new data message (it copies 7 bytes in its memory from IN[3] on) then toggles bit **B** as acknowledge.

Input Area	80$_{Hex}$	00$_{Hex}$	07$_{Hex}$	02$_{Hex}$	31$_{Hex}$	30$_{Hex}$	44$_{Hex}$	4C$_{Hex}$	0D$_{Hex}$	0A$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	**80**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

7. The reader performs *No Read* and toggles bit **A**. Let's assume <CAN> as *Global No Read* character.

Input Area	**81**$_{Hex}$	00$_{Hex}$	**04**$_{Hex}$	**02**$_{Hex}$	**18**$_{Hex}$	**0D**$_{Hex}$	**0A**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	80$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

8. PLC reads new data message (it copies 4 bytes in its memory from IN[3] on) then toggles bit **B** as acknowledge.

Input Area	81$_{Hex}$	00$_{Hex}$	04$_{Hex}$	02$_{Hex}$	18$_{Hex}$	0D$_{Hex}$	0A$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	**81**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

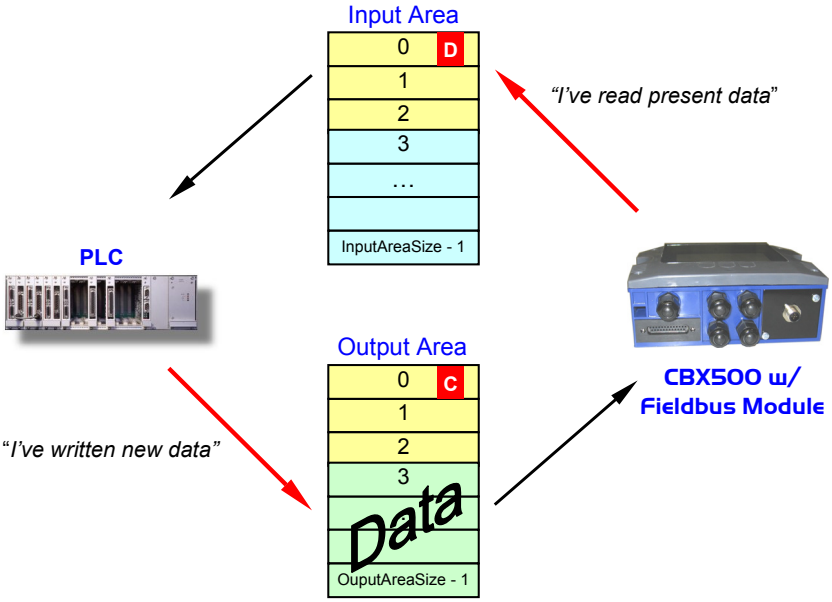
9. Data exchange continues…

2.5 DATA TRANSMISSION FROM PLC TO THE READER

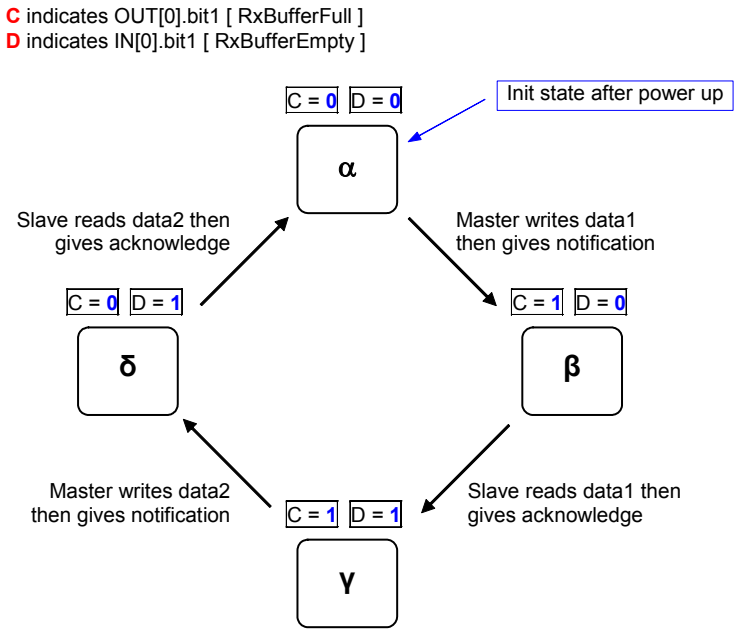
Analogous to the previous paragraph, flow control works even when data are coming from the Master towards the Slave. The communication mechanism is based on the same concepts:

- **OUT[0].bit1** [**C**] must be used by PLC to notify that "*Master has written new data so Slave can read it*"
- **IN[0].bit1** [**D**] is used by the reader to notify that "*Slave has read data so Master can send next message*"

This happens each time bit C (or D) changes its state (toggles). Bit level doesn't matter, only the transition has to be considered.



The following state machine shows data transmission from Master to Slave. Please note that each cycle transfers two data messages.

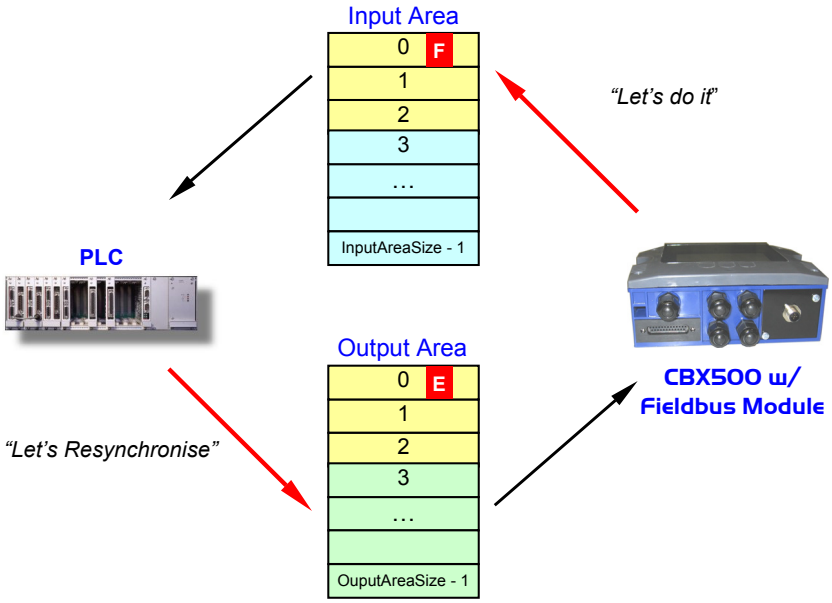


2.6 RESYNCHRONISATION

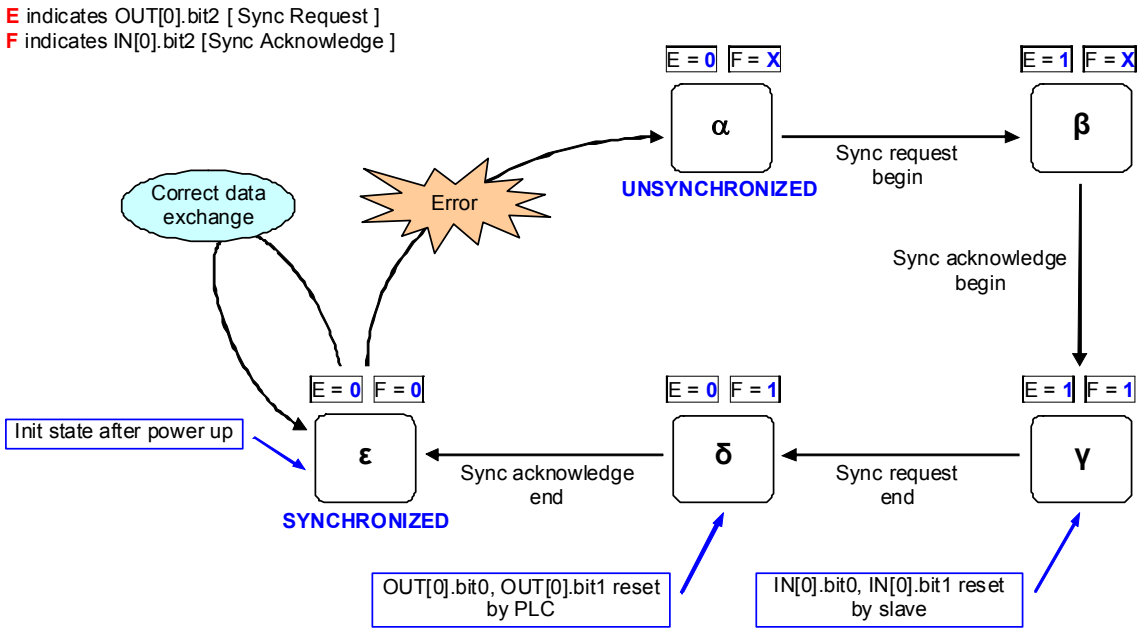
The resynchronisation process restarts the messaging protocol from a predefined state.
It may be used either at the Master startup to detect if the Slave is on line or during normal operations in case of errors requiring a protocol reset procedure.

The process is based on bit two:

- **OUT[0].bit2** [E] must be used by PLC to request the Resynchronisation
- **IN[0].bit2** [F] is used by the reader to acknowledge the request



The following state machine shows the resynchronisation cycle, requested by the PLC and performed together with the reader:



Let's analyse the resynchronisation process, starting from the previous data exchange discussed in "Data Transmission from the Reader to PLC"…

Input Area	81_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	81_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

1. PLC requests resynchonisation by setting bit **E** = 1.

Input Area	81_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	**85_{Hex}** 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

2. The reader detects the request and so it resets **IN[0].bit0** and **IN[0].bit1**. Then it gives an acknowledge back to the PLC by means of bit **F**.

Input Area	**84_{Hex}** 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	85_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

3. PLC has to reset **OUT[0].bit0** and **OUT[0].bit1** before completing its request with bit **E** = 0.

Input Area	84_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	**80_{Hex}** 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

4. The reader completes the acknowledge process by setting bit **F** = 0.

Input Area	**80_{Hex}** 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	80_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

5. Now Flow Control has been returned to a predefined state. All data exchange bits in the Control Field are surely zero and data transmission can proceed safely.

2.7 FRAGMENTATION AND REASSEMBLING

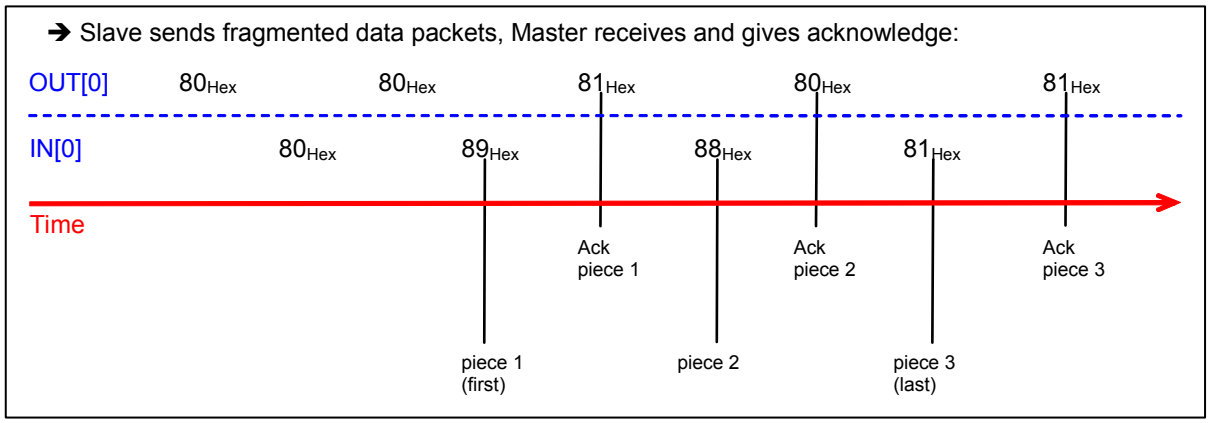
The fragmentation process is activated whenever Application Data cannot be contained in the related exchange area. Basically long messages are split into pieces which are transmitted separately. Reassembling allows the reconstruction of the whole messages.
The reader already implements these functions in the DAD layer, while the PLC needs a congruent management.

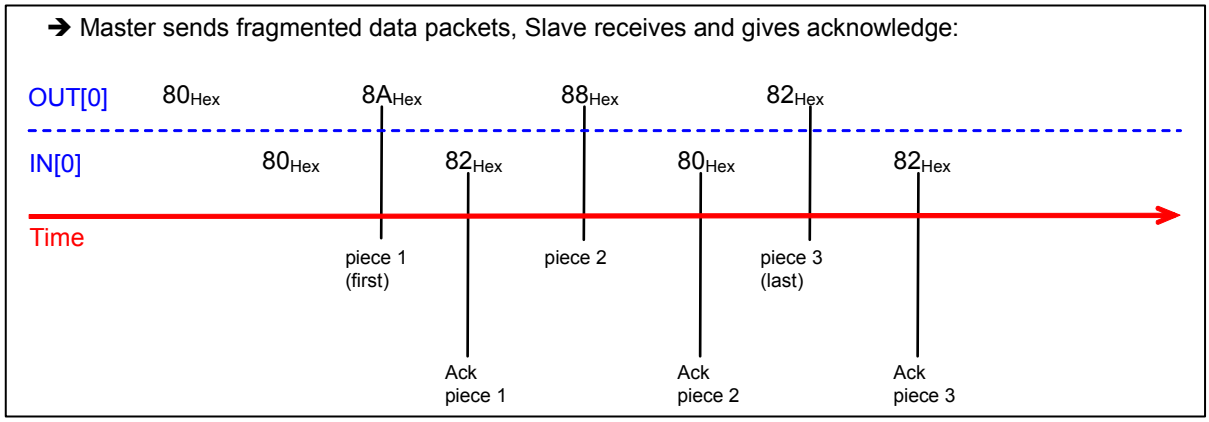
The fragmentation is based on the **More Bit** (bit 3) in the Control Field byte.
More Bit = 0 indicates that all the information is included within the current message. When Application Data is longer than (exchange area size – 3), the first partial message is transmitted having More Bit = 1. Following fragments keep More Bit = 1 and only the last piece will have More Bit = 0 again. Thanks to this mechanism, the receiver station may detect the last piece and so reassemble the entire information.

Some notes:

- Intermediate fragments have Length Field = (exchange area size – 3)
- Last fragment has Length Field ≤ (exchange area size – 3)
- Bit0 and Bit1 of both Input and Output areas are independently managed for any fragment

The following figures show how the Control Byte changes according to the fragmentation process. Both data flow directions are considered.





Let's analyse a fragmented data exchange based on the following settings:

- Flow Control = **DAD Driver**
- Input Area Size = **16**
- Output Area Size = **8**

1. After power up Input and Output areas are generally filled by zero. According to DAD driver implementation, Input area has Control Field = 80_{Hex} and SAP = 00_{Hex}.

Input Area	**80**$_{Hex}$	**00**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$							

2. Also PLC must set the Control Field of Output area properly, as long as DAD messaging protocol is utilised.

Input Area	80$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	**80**$_{Hex}$	**00**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$							

3. The reader reads a barcode with content "*1234567890abcde1234567890abcde*". Let's assume standard data formatting with <STX> as header and <CR><LF> as terminators. In this condition since the whole message cannot be included in Input Area, the reader transmits first fragment "<STX>*1234567890ab*" only (setting More Bit = 1) then it toggles bit **A**.

Input Area	**89**$_{Hex}$	00$_{Hex}$	**0D**$_{Hex}$	**02**$_{Hex}$	**31**$_{Hex}$	**32**$_{Hex}$	**33**$_{Hex}$	**34**$_{Hex}$	**35**$_{Hex}$	**36**$_{Hex}$	**37**$_{Hex}$	**38**$_{Hex}$	**39**$_{Hex}$	**30**$_{Hex}$	**61**$_{Hex}$	**62**$_{Hex}$
Output Area	80$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

4. PLC detects transition of bit **A** so now it can read first incoming fragment (it copies 13 bytes in its memory from IN[3] on) then toggles bit **B** as acknowledge.

Input Area	89$_{Hex}$	00$_{Hex}$	0D$_{Hex}$	02$_{Hex}$	31$_{Hex}$	32$_{Hex}$	33$_{Hex}$	34$_{Hex}$	35$_{Hex}$	36$_{Hex}$	37$_{Hex}$	38$_{Hex}$	39$_{Hex}$	30$_{Hex}$	61$_{Hex}$	62$_{Hex}$
Output Area	**81**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

5. The reader detects transition of bit **B** so it sends second fragment "*cde1234567890*" (still More Bit = 1) and toggles bit **A**.

Input Area	**88**$_{Hex}$	00$_{Hex}$	**0D**$_{Hex}$	**63**$_{Hex}$	**64**$_{Hex}$	**65**$_{Hex}$	**31**$_{Hex}$	**32**$_{Hex}$	**33**$_{Hex}$	**34**$_{Hex}$	**35**$_{Hex}$	**36**$_{Hex}$	**37**$_{Hex}$	**38**$_{Hex}$	**39**$_{Hex}$	**30**$_{Hex}$
Output Area	81$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

6. PLC reads second fragment (it copies 13 bytes in its memory from IN[3] on) then toggles bit **B** as acknowledge.

Input Area	88$_{Hex}$	00$_{Hex}$	0D$_{Hex}$	63$_{Hex}$	64$_{Hex}$	65$_{Hex}$	31$_{Hex}$	32$_{Hex}$	33$_{Hex}$	34$_{Hex}$	35$_{Hex}$	36$_{Hex}$	37$_{Hex}$	38$_{Hex}$	39$_{Hex}$	30$_{Hex}$
Output Area	**80**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

7. The reader sends third (last) fragment "*abcde<CR><LF>*" (finally More Bit = 0) and toggles bit **A**.

Input Area	**81**$_{Hex}$	00$_{Hex}$	**07**$_{Hex}$	**61**$_{Hex}$	**62**$_{Hex}$	**63**$_{Hex}$	**64**$_{Hex}$	**65**$_{Hex}$	**0D**$_{Hex}$	**0A**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	80$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

8. PLC reads last fragment (it copies 7 bytes in its memory from IN[3] on) and now the reassembling can be completed. Then it toggles bit **B** as acknowledge.

Input Area	81$_{Hex}$	00$_{Hex}$	07$_{Hex}$	61$_{Hex}$	62$_{Hex}$	63$_{Hex}$	64$_{Hex}$	65$_{Hex}$	0D$_{Hex}$	0A$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$
Output Area	**81**$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$	00$_{Hex}$								

9. Whole message has been completely transmitted.

2.8 SAP SERVICES

When SAP = 255, the FLUSH QUEUE is the unique driver service currently available. It performs flushing of the internal queues and may be issued at any time.

FLUSH QUEUE Service

Request: Flush data buffers (issued by the Master station to the reader)

Action: Flush all information from previous decoding phases

Response: Command accepted/rejected (generated by the reader toward the Master).

Application data areas must be formatted as follows:

Request Command	**byte 3**	**byte 4**
Flush data buffer	**'['** (5B Hex)	**'F'**(46 Hex)

Response Command	**byte 3**	**byte 4**
Command accepted	**'A'** (41 Hex)	**' '**(20 Hex)
Command rejected	**'C'** (43 Hex)	**' '**(20 Hex)

2.9 DAD INTERNAL QUEUES

The Fieldbus Module has two internal queues (one for each direction) to keep the application events: input queue and output queue.

The input queue is used when a new message (generally a barcode) has to be transmitted by the Fieldbus Module before the Master station has generated all the acknowledge handshakes for each previous transmission.

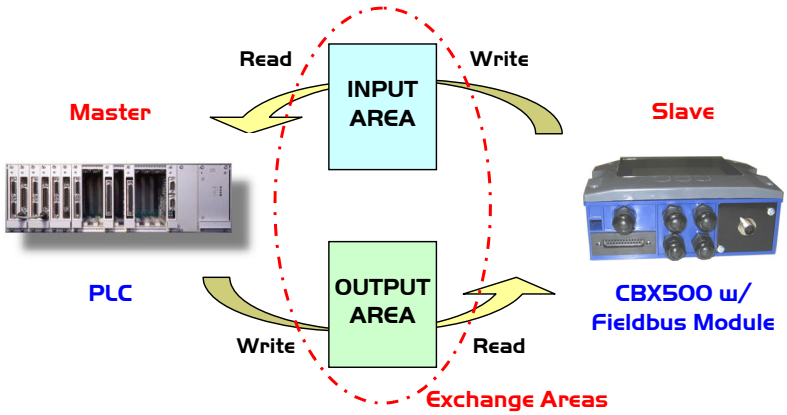
The output queue is rarely used at the moment.

The queues are sized in the following way:

- 50 elements for the input queue (number of messages buffered from Slave to Master)
- 26 elements for the output queue (number of messages buffered from Master to Slave)

The queues may be flushed by the Master station through the SAP=255 primitive. This is generally done at the Master startup if the Master station wants to cancel all the previous buffers that were generated before its startup. However, the Master station is free to decide not to cancel them.

3 FCM WITH DPD DRIVER



From now on we refer to the Input Area as a buffer made up of InputAreaSize bytes:

IN[0]	IN[1]	IN[2]	…	IN[InputAreaSize - 1]

and to the Output Area as a buffer made up of OutputAreaSize bytes:

OUT[0]	OUT[1]	OUT[2]	…	OUT[OutputAreaSize - 1]

The first **four** bytes are used by the DPD Driver layer in both buffers:

Control Field (byte 0) — used to issue and control the driver primitives such as flow-control, fragmentation and resynchronisation.

Fieldbus Station Address Field (byte 1) — contains the information of the selected device address of the Fieldbus network.

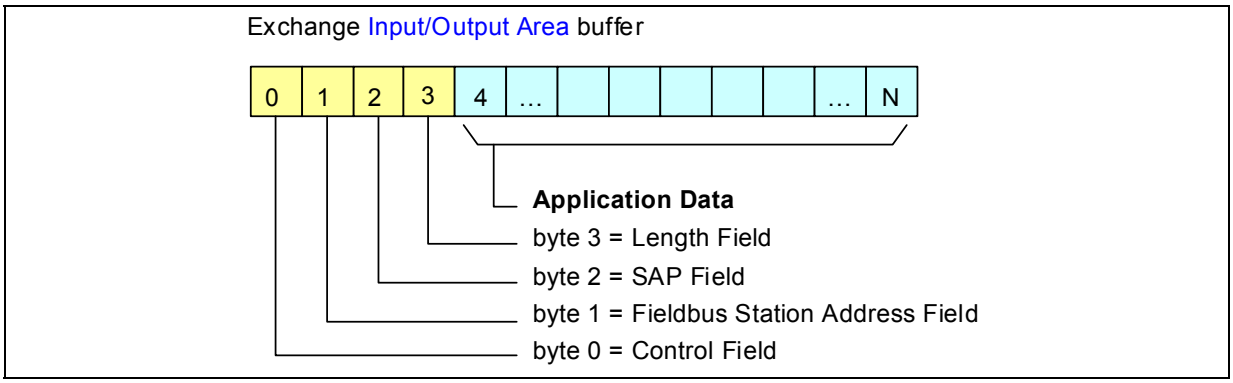
Service Access Point Field (byte 2) — used to distinguish among different services and to provide future expandability. (Since this SAP definition is introduced by the DPD Driver, it must not be confused with the SAP that is defined by the international standard).

Length Field (byte 3) — contains the number of bytes used by the application layer: *(see also note below)*
Length Field ≤ (InputAreaSize – 4) for the Input Area
Length Field ≤ (OutputAreaSize – 4) for the Output Area.

NOTE

If Data Consistency is enabled then the last byte is used for packet integrity and therefore the bytes available to the application are reduced by 1. See chapter 4 of this manual for details.

If Digital I/O Conditioning is used to echo reader Inputs to the Fieldbus Master or control reader Outputs by the Fieldbus Master, then Byte 0 is reserved for this purpose and therefore the DAD Driver occupies bytes 1 to 4. This also reduces the bytes available to the application by 1. See chapter 5 of this manual and Digital I/O Conditioning in the Configuration Parameters Help On Line.



The Application Data buffer holds useful information, typically the barcode messages, processed by the application program. IN[4] contains the first significant byte of the Application Data buffer (the same first byte you would see if the reader transmitted the barcode buffer onto a serial port instead of the Fieldbus interface).

The structure of the application buffer and its length strictly depend on the selected data format on the reader. Barcode messages longer than (InputAreaSize – 4) will be split in pieces through an automatic fragmentation process (see details in the "Fragmentation and Reassembling" paragraph).

3.1 CONTROL FIELD

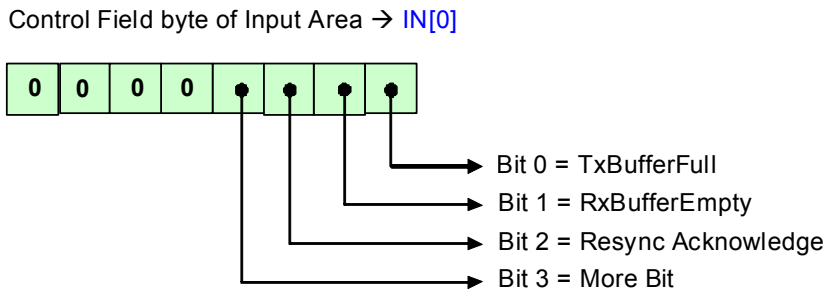
This is the core of the flow controlled communication.

The Input Area structure reserves bit 0 and bit 1 of IN[0] for handshake purposes while the Output Area structure, which is symmetrical, reserves bit 0 and 1 of OUT[0].

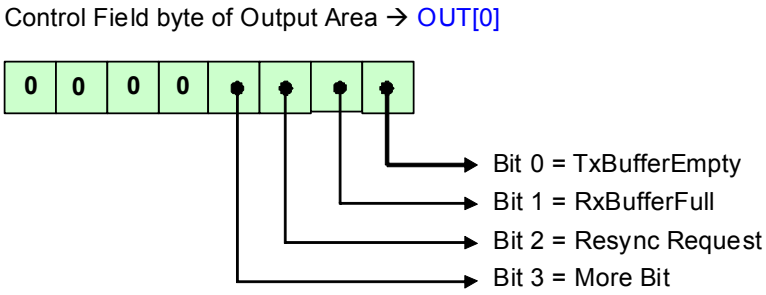
At any time the Master station can make a resynchronization request by means of bit 2 of the Output Area. This process, which resets the synchronization numbers (bit 0 and bit 1 of both Input and Output areas), has to be acknowledged by the Slave on bit 2 of the Input Area.

Bit 3 is used to control a fragmentation sequence in both directions.

Bit 7 is always set to 0.



IN[0].bit0	**TxBufferFull** toggles when Slave has made available new Input Area data.
IN[0].bit1	**RxBufferEmpty** toggles when Output Area data has been read by Slave.
IN[0].bit2	**Resync Acknowledge** set to 1 as an acknowledge to a resync request. With this bit, the master can detect a slave is on line.
IN[0].bit3	**More Bit** is 1 when this is not the last piece of a fragmentation sequence while it is 0 when this is the last piece.
IN[0].bit4,5,6,7	Set to **0**, **0**, **0**, **0** when DPD messaging protocol is used.



OUT[0].bit0	**TxBufferEmpty** must toggle when Input Area data has been read by Master.
OUT[0].bit1	**RxBufferFull** must toggle when Master makes available new Output Area data.
OUT[0].bit2	**Resync Request** set to 1 for one second to resynchronize the slave. After resynchronization, all 4 handshake bits are set to 0.
OUT[0].bit3	**More Bit** must be 1 when this is not the last piece of a fragmentation sequence and it must be 0 when this is the last.
OUT[0].bit4,5,6,7	Set to **0**, **0**, **0**, **0** when DPD messaging protocol is used.

3.2 STATION ADDRESS FIELD

The INPUT and OUTPUT buffers keep the information of the selected Fieldbus Address in this field.

3.3 SAP FIELD

SAP (Service Access Point) is an identifier that is used to implement multiple services sharing the same communication channel between two remote stations.
The following values have been defined:

- SAP = 0 Used to transfer information messages between the reader and the PLC
- SAP = 255 Reserved for driver services (see details in paragraph 3.9)

All other SAP values are free and they could be used by dedicated application programs after agreement between the application programs themselves.

3.4 LENGTH FIELD

The Application layer uses all or a part of the remaining bytes of the Exchange Area buffers that are not used by the DPD Driver. The Length Field is introduced to keep the information of how many bytes are really used by the Application Layer.

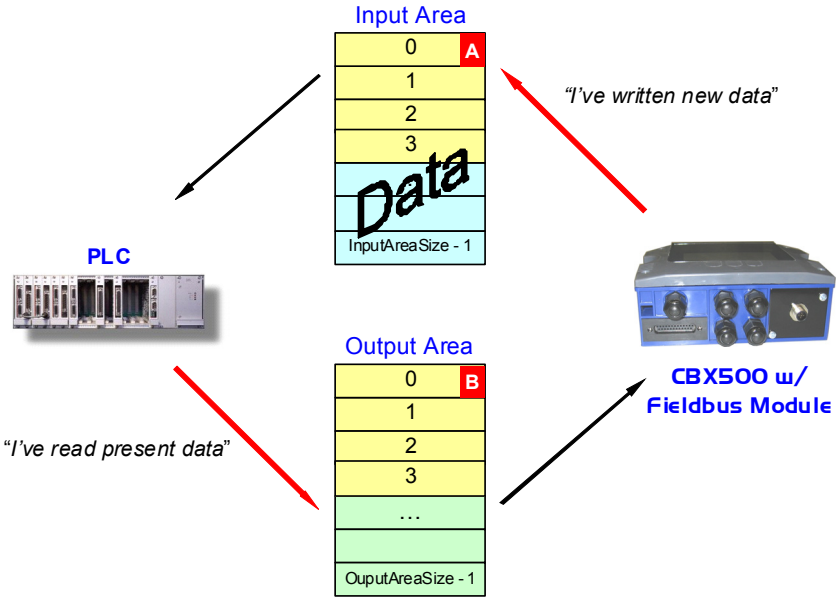
A fragment that is not the last one of a fragmentation sequence must fill this field with [InputAreaSize – 4] (or [OutputAreaSize – 4]), depending on whether it is an Input/Output fragment. Otherwise this field gets a value that is less than or equal to [InputAreaSize – 4].

3.5 DATA TRANSMISSION FROM THE READER TO PLC

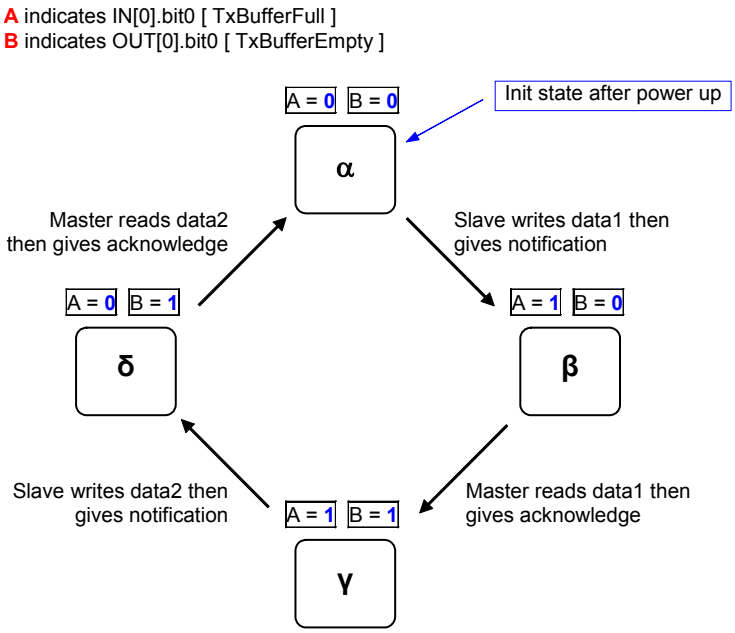
This paragraph describes how it is possible to exchange messages with flow control. The communication mechanism is simple:

- **IN[0].bit0** [**A**] is used by the reader to notify that "*Slave has written a new data so Master can read it*"
- **OUT[0].bit0** [**B**] must be used by PLC to notify that "*Master has read last data so Slave can send next message*"

This happens each time bit A (or B) changes its state (toggles). Bit level doesn't matter, only the transition has to be considered.



The following state machine shows data transmission from Slave to Master. Please note that each cycle transfers two data messages.



Let's analyse a typical data exchange based on the following settings:

- Flow Control = **DPD Driver**
- Fieldbus Address = **5**
- Input Area Size = **16**
- Output Area Size = **8**

1. After power up Input and Output areas are generally filled by zero. According to DPD driver implementation, Input area has Control Field = 00_{Hex} and SAP = 00_{Hex}

Input Area	**00**$_{Hex}$ **05**$_{Hex}$ **00**$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

2. Also PLC must set the Control Field of Output area properly, as long as DPD messaging protocol is utilised.

Input Area	00$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	**00**$_{Hex}$ **05**$_{Hex}$ **00**$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

3. The reader reads a barcode "123456". Let's assume standard data formatting with <STX> as header and <CR><LF> as terminators. the reader toggles bit A.

Input Area	**01**$_{Hex}$ **05**$_{Hex}$ **00**$_{Hex}$ **09**$_{Hex}$ **02**$_{Hex}$ **31**$_{Hex}$ **32**$_{Hex}$ **33**$_{Hex}$ **34**$_{Hex}$ **35**$_{Hex}$ **36**$_{Hex}$ **0D**$_{Hex}$ **0A**$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	00$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

4. PLC detects transition of bit A so now it can read incoming data (it copies 9 bytes in its memory from IN[4] on) then toggles bit B as acknowledge.
   ***Note**: before the acknowledge, all further barcodes read by the reader are buffered.*

Input Area	01$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 09$_{Hex}$ 02$_{Hex}$ 31$_{Hex}$ 32$_{Hex}$ 33$_{Hex}$ 34$_{Hex}$ 35$_{Hex}$ 36$_{Hex}$ 0D$_{Hex}$ 0A$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	**01**$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

5. The reader reads a barcode "10DL" and toggles bit A.

Input Area	**00**$_{Hex}$ **05**$_{Hex}$ **00**$_{Hex}$ **07**$_{Hex}$ **02**$_{Hex}$ **31**$_{Hex}$ **30**$_{Hex}$ **44**$_{Hex}$ **4C**$_{Hex}$ **0D**$_{Hex}$ **0A**$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	01$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

6. PLC reads new data message (it copies 7 bytes in its memory from IN[4] on) then toggles bit B as acknowledge.

Input Area	00$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 07$_{Hex}$ 02$_{Hex}$ 31$_{Hex}$ 30$_{Hex}$ 44$_{Hex}$ 4C$_{Hex}$ 0D$_{Hex}$ 0A$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	**00**$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

7. The reader performs *No Read* and toggles bit A. Let's assume <CAN> as *Global No Read* character.

Input Area	**01**$_{Hex}$ **05**$_{Hex}$ **00**$_{Hex}$ **04**$_{Hex}$ **02**$_{Hex}$ **18**$_{Hex}$ **0D**$_{Hex}$ **0A**$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	00$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

8. PLC reads new data message (it copies 4 bytes in its memory from IN[4] on) then toggles bit B as acknowledge.

Input Area	01$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 04$_{Hex}$ 02$_{Hex}$ 18$_{Hex}$ 0D$_{Hex}$ 0A$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$
Output Area	**01**$_{Hex}$ 05$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$ 00$_{Hex}$

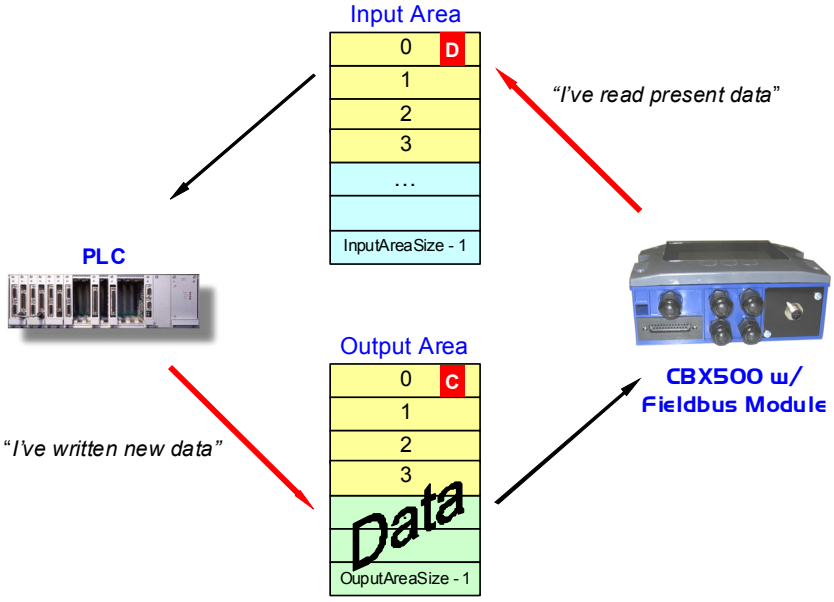
9. Data exchange continues…

3.6 DATA TRANSMISSION FROM PLC TO THE READER

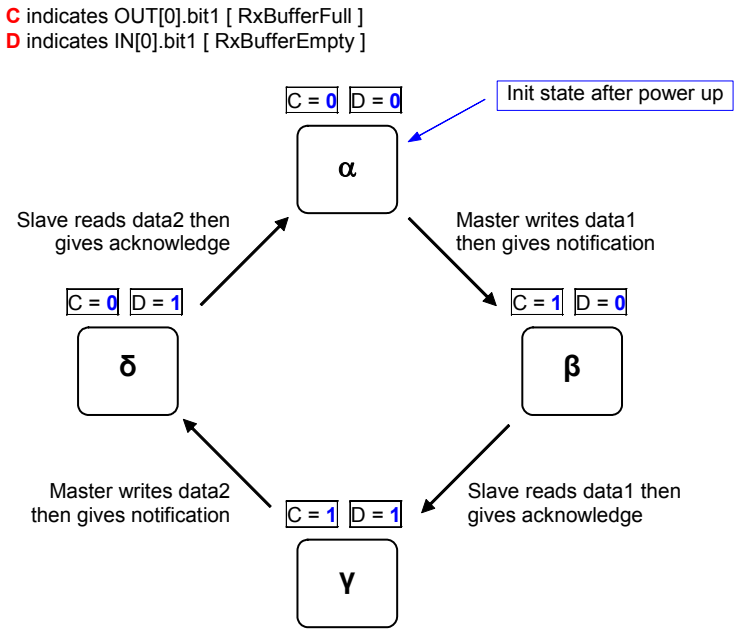
Analogous to the previous paragraph, flow control works even when data are coming from the Master towards the Slave. The communication mechanism is based on the same concepts:

- **OUT[0].bit1** [**C**] must be used by PLC to notify that "*Master has written new data so Slave can read it*"
- **IN[0].bit1** [**D**] is used by the reader to notify that "*Slave has read data so Master can send next message*"

This happens each time bit C (or D) changes its state (toggles). Bit level doesn't matter, only the transition has to be considered.



The following state machine shows data transmission from Master to Slave. Please note that each cycle transfers two data messages.

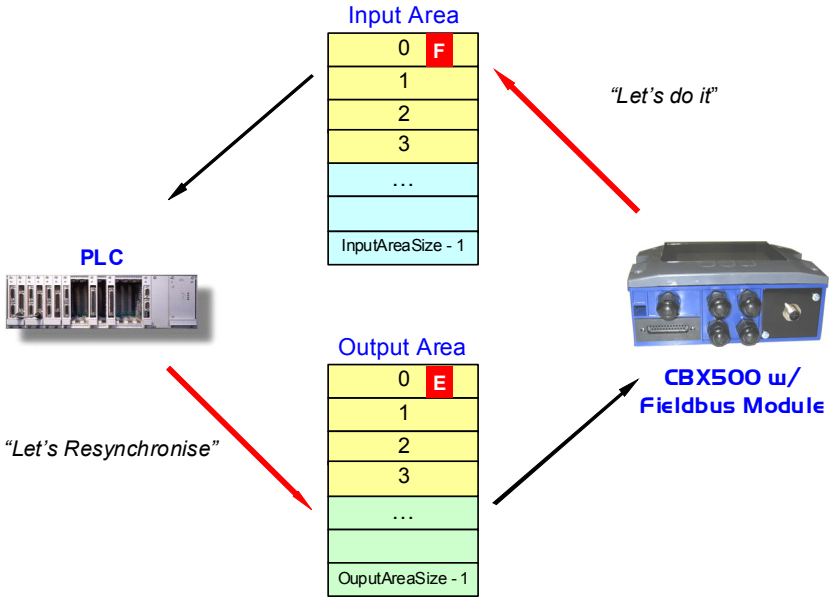


3.7 RESYNCHRONISATION

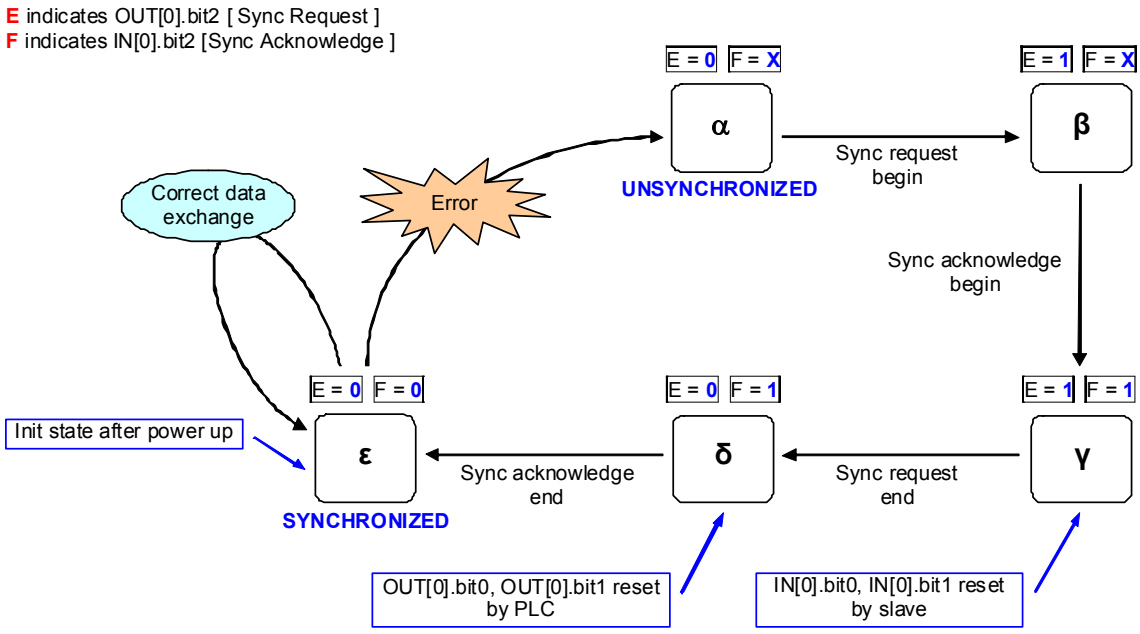
The resynchronisation process restarts the messaging protocol from a predefined state.
It may be used either at the Master startup to detect if the Slave is on line or during normal operations in case of errors requiring a protocol reset procedure.

The process is based on bit two:

- **OUT[0].bit2** [E] must be used by PLC to request the Resynchronisation
- **IN[0].bit2** [F] is used by the reader to acknowledge the request



The following state machine shows the resynchronisation cycle, requested by the PLC and performed together with the reader:



Let's analyse the resynchronisation process, starting from the previous data exchange discussed in "Data Transmission from the Reader to PLC"…

Input Area	01_{Hex} 05_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	01_{Hex} 05_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

1. PLC requests resynchonisation by setting bit E = 1.

Input Area	01_{Hex} 05_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	**05_{Hex}** 05_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

2. The reader detects the request and so it resets **IN[0].bit0** and **IN[0].bit1**. Then it gives an acknowledge back to the PLC by means of bit F.

Input Area	**04_{Hex}** 05_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	05_{Hex} 05_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

3. PLC has to reset **OUT[0].bit0** and **OUT[0].bit1** before completing its request with bit E = 0.

Input Area	04_{Hex} 05_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	**00_{Hex}** 05_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

4. The reader completes the acknowledge process by setting bit F = 0.

Input Area	**00_{Hex}** 05_{Hex} 00_{Hex} 04_{Hex} 02_{Hex} 18_{Hex} $0D_{Hex}$ $0A_{Hex}$ 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}
Output Area	00_{Hex} 05_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex} 00_{Hex}

5. Now Flow Control has been returned to a predefined state. All data exchange bits in the Control Field are surely zero and data transmission can proceed safely.

3.8 FRAGMENTATION AND REASSEMBLING

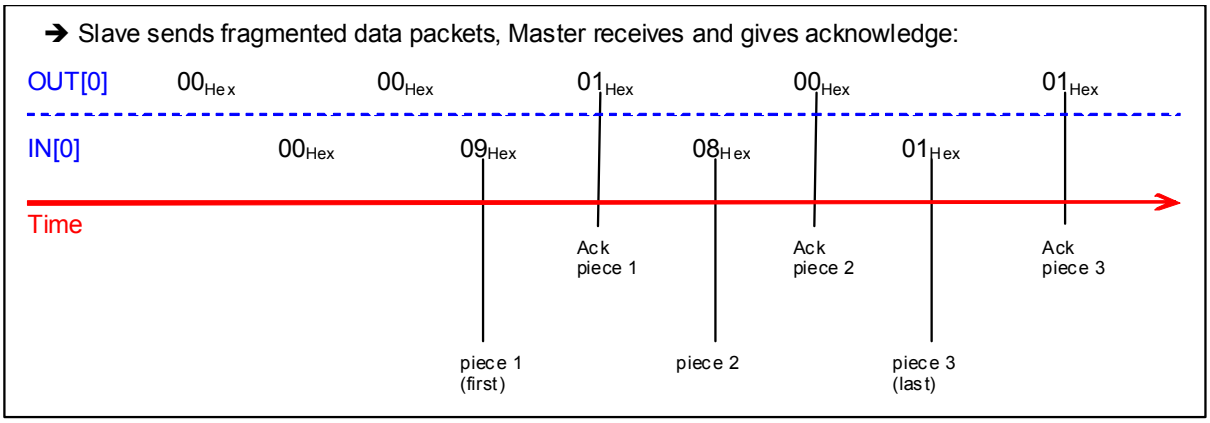
The fragmentation process is activated whenever Application Data cannot be contained in the related exchange area. Basically long messages are split into pieces which are transmitted separately. Reassembling allows the reconstruction of the whole messages.
The reader already implements these functions in the DPD layer, while the PLC needs a congruent management.

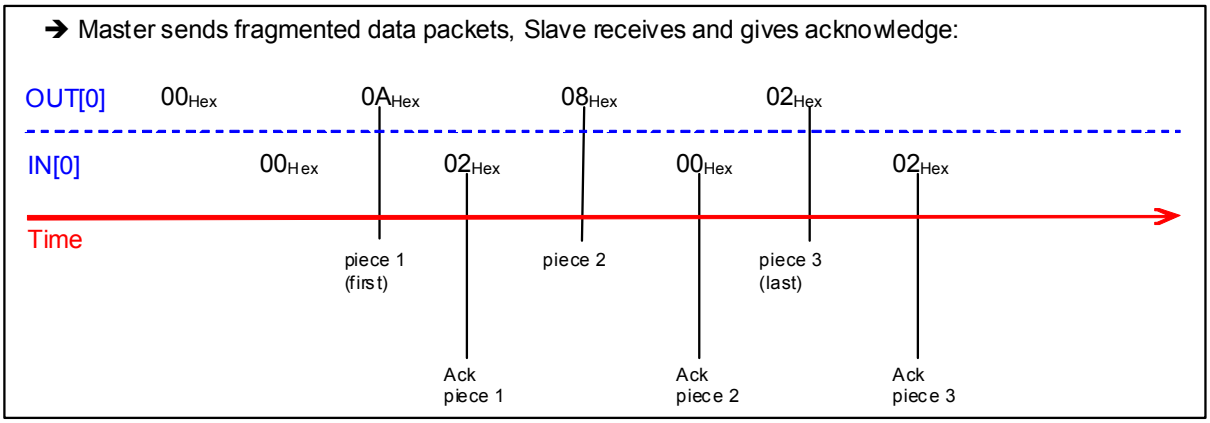
The fragmentation is based on the **More Bit** (bit 3) in the Control Field byte.
More Bit = 0 indicates that all the information is included within the current message. When Application Data is longer than (exchange area size – 4), the first partial message is transmitted having More Bit = 1. Following fragments keep More Bit = 1 and only the last piece will have More Bit = 0 again. Thanks to this mechanism, the receiver station may detect the last piece and so reassemble the entire information.

Some notes:

- Intermediate fragments have Length Field = (exchange area size – 4)
- Last fragment has Length Field ≤ (exchange area size – 4)
- Bit0 and Bit1 of both Input and Output areas are independently managed for any fragment

The following figures show how the Control Byte changes according to the fragmentation process. Both data flow directions are considered.





Let's analyse a fragmented data exchange based on the following settings:

- Flow Control = **DPD Driver**
- Fieldbus Address = **5**
- Input Area Size = **16**
- Output Area Size = **8**

1. After power up Input and Output areas are generally filled by zero. According to DPD driver implementation, Input area has Control Field = 00_{Hex} and SAP = 00_{Hex}

Input Area	**00**Hex	**05**Hex	**00**Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex
Output Area	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

2. Also PLC must set the Control Field of Output area properly, as long as DPD messaging protocol is utilised.

Input Area	00Hex	05Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex
Output Area	**00**Hex	**05**Hex	**00**Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

3. The reader reads a barcode with content "*1234567890abcde1234567890abcde*". Let's assume standard data formatting with <STX> as header and <CR><LF> as terminators. In this condition since the whole message cannot be included in Input Area, the reader transmits first fragment "<STX>*1234567890a*" only (setting More Bit = 1) then it toggles bit **A**.

Input Area	**09**Hex	05Hex	00Hex	**0C**Hex	**02**Hex	**31**Hex	**32**Hex	**33**Hex	**34**Hex	**35**Hex	**36**Hex	**37**Hex	**38**Hex	**39**Hex	**30**Hex	**61**Hex
Output Area	00Hex	05Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

4. PLC detects transition of bit **A** so now it can read first incoming fragment (it copies 12 bytes in its memory from IN[4] on) then toggles bit **B** as acknowledge.

Input Area	09Hex	05Hex	00Hex	0CHex	02Hex	31Hex	32Hex	33Hex	34Hex	35Hex	36Hex	37Hex	38Hex	39Hex	30Hex	61Hex
Output Area	**01**Hex	05Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

5. The reader detects transition of bit **B** so it sends second fragment "*bcde12345678*" (still More Bit = 1) and toggles bit **A**.

Input Area	**08**Hex	05Hex	00Hex	**0C**Hex	**62**Hex	**63**Hex	**64**Hex	**65**Hex	**31**Hex	**32**Hex	**33**Hex	**34**Hex	**35**Hex	**36**Hex	**37**Hex	**38**Hex
Output Area	01Hex	05Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

6. PLC reads second fragment (it copies 12 bytes in its memory from IN[4] on) then toggles bit **B** as acknowledge.

Input Area	08Hex	05Hex	00Hex	0CHex	62Hex	63Hex	64Hex	65Hex	31Hex	32Hex	33Hex	34Hex	35Hex	36Hex	37Hex	38Hex
Output Area	**00**Hex	05Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

7. The reader sends third (last) fragment "*90abcde<CR><LF>*" (finally More Bit = 0) and toggles bit **A**.

Input Area	**01**Hex	05Hex	00Hex	**09**Hex	**39**Hex	**30**Hex	**61**Hex	**62**Hex	**63**Hex	**64**Hex	**65**Hex	**0D**Hex	**0A**Hex	00Hex	00Hex	00Hex
Output Area	00Hex	05Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

8. PLC reads last fragment (it copies 9 bytes in its memory from IN[4] on) and now the reassembling can be completed. Then it toggles bit **B** as acknowledge.

Input Area	01Hex	05Hex	00Hex	09Hex	39Hex	30Hex	61Hex	62Hex	63Hex	64Hex	65Hex	0DHex	0AHex	00Hex	00Hex	00Hex
Output Area	**01**Hex	05Hex	00Hex	00Hex	00Hex	00Hex	00Hex	00Hex								

9. Whole message has been completely transmitted.

3.9 SAP SERVICES

When SAP = 255, the FLUSH QUEUE is the unique driver service currently available. It performs flushing of the internal queues and may be issued at any time.

FLUSH QUEUE Service

Request: Flush data buffers (issued by the Master station to the reader)

Action: Flush all information from previous decoding phases

Response: Command accepted/rejected (generated by the reader toward the Master).

Application data areas must be formatted as follows:

Request Command	**byte 3**	**byte 4**
Flush data buffer	**'['** (5B Hex)	**'F'**(46 Hex)

Response Command	**byte 3**	**byte 4**
Command accepted	**'A'** (41 Hex)	**' '**(20 Hex)
Command rejected	**'C'** (43 Hex)	**' '**(20 Hex)

3.10 DPD INTERNAL QUEUES

The Fieldbus Module has two internal queues (one for each direction) to keep the application events: input queue and output queue.

The input queue is used when a new message (generally a barcode) has to be transmitted by the Fieldbus Module before the Master station has generated all the acknowledge handshakes for each previous transmission.

The output queue is rarely used at the moment.

The queues are sized in the following way:

- 50 elements for the input queue (number of messages buffered from Slave to Master)
- 26 elements for the output queue (number of messages buffered from Master to Slave)

The queues may be flushed by the Master station through the SAP=255 primitive. This is generally done at the Master startup if the Master station wants to cancel all the previous buffers that were generated before its startup. However, the Master station is free to decide not to cancel them.

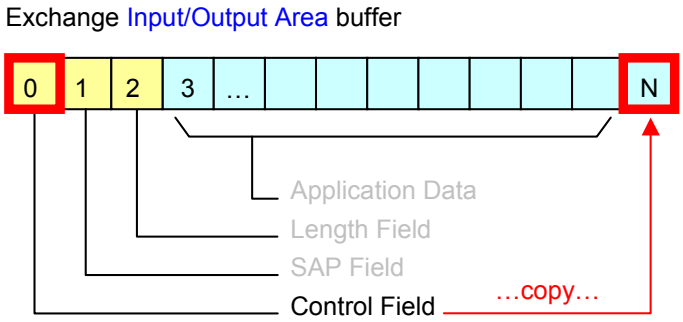
4 DATA CONSISTENCY

When Flow Control (either DAD or DPD Driver) is used, the Data Consistency option is available.

If enabled, the driver implements a specific mechanism to guarantee consistency of both transmitted and received data over the entire size of the exchange area.

4.1 DATA CONSISTENCY WITH DAD DRIVER

As shown in the following figure, the application layer copies the Control Field byte in the last position of the exchange area. In this way, as the Control Field changes from message to message, the whole message is consistent as long as the first and last bytes are matching.



From the practical point of view, a complete message in the Input area could be considered consistent as soon as the PLC verifies that IN[0] is equal to IN[InputAreaSize - 1].

The PLC should take care to double OUT[0] in OUT[OutputAreaSize - 1] to let the driver check the consistency.

Due to the new byte used by the DAD driver, barcode messages longer than (InputAreaSize – **4**) will be split into pieces through an automatic fragmentation process.

Let's analyse a fragmented and consistent data exchange based on:

- Flow Control = **DAD Driver**
- Data Consistency = **Enable**
- Input Area Size = **16**
- Output Area Size = **8**

1. After power up Input and Output areas are generally filled by zero. According to DAD driver implementation, Input area has Control Field = IN[0] = IN[15] = 80Hex and SAP = 00Hex.

Input Area	80Hex	00Hex	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	80Hex
Output Area	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**								

2. PLC must set Control Field = OUT[0] = OUT[7] = 80Hex according to DAD messaging protocol with Data Consistency.

Input Area	**80Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**80Hex**
Output Area	80Hex	00Hex	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	80Hex								

3. Slave reads a barcode with content "1234567890abcde1234567890abcde". Let's assume standard data formatting with <STX> as header and <CR><LF> as terminators. Slave transmits first fragment "<STX>1234567890a" only (setting More Bit = 1) then it toggles bit **A**.

Input Area	89Hex	**00Hex**	0CHex	02Hex	31Hex	32Hex	33Hex	34Hex	35Hex	36Hex	37Hex	38Hex	39Hex	30Hex	61Hex	89Hex
Output Area	**80Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**80Hex**								

4. PLC detects transition of bit **A** so now it can read first incoming fragment (it copies 12 bytes in its memory from IN[3] on) then toggles bit **B** as acknowledge.

Input Area	**89Hex**	**00Hex**	**0CHex**	**02Hex**	**31Hex**	**32Hex**	**33Hex**	**34Hex**	**35Hex**	**36Hex**	**37Hex**	**38Hex**	**39Hex**	**30Hex**	**61Hex**	**89Hex**
Output Area	81Hex	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	81Hex								

5. Slave detects transition of bit **B** so it sends second fragment "bcde12345678" (still More Bit = 1) and toggles bit **A**.

Input Area	88Hex	**00Hex**	0CHex	62Hex	63Hex	64Hex	65Hex	31Hex	32Hex	33Hex	34Hex	35Hex	36Hex	37Hex	38Hex	88Hex
Output Area	**81Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**81Hex**								

6. PLC reads second fragment (it copies 12 bytes in its memory from IN[3] on) then toggles bit **B** as acknowledge.

Input Area	**88Hex**	**00Hex**	**0CHex**	**62Hex**	**63Hex**	**64Hex**	**65Hex**	**31Hex**	**32Hex**	**33Hex**	**34Hex**	**35Hex**	**36Hex**	**37Hex**	**38Hex**	**88Hex**
Output Area	80Hex	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	80Hex								

7. Slave sends last fragment "90abcde<CR><LF>" (finally More Bit = 0) and toggles bit **A**.

Input Area	81Hex	**00Hex**	09Hex	39Hex	30Hex	61Hex	62Hex	63Hex	64Hex	65Hex	0DHex	0AHex	**00Hex**	**00Hex**	**00Hex**	81Hex
Output Area	**80Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**80Hex**								

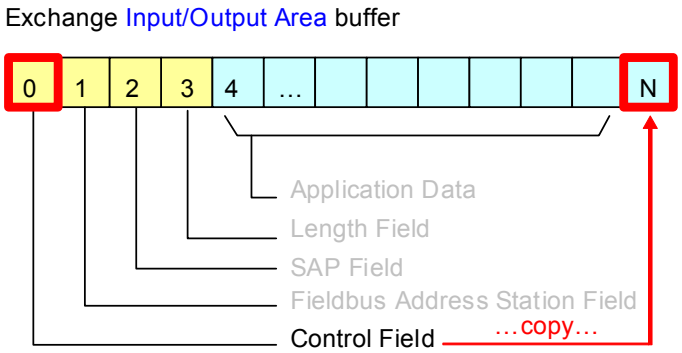
8. PLC reads last fragment (it copies 9 bytes in its memory from IN[3] on) and now the reassembling can be completed. Then it toggles bit **B** as acknowledge.

Input Area	**81Hex**	**00Hex**	**09Hex**	**39Hex**	**30Hex**	**61Hex**	**62Hex**	**63Hex**	**64Hex**	**65Hex**	**0DHex**	**0AHex**	**00Hex**	**00Hex**	**00Hex**	**81Hex**
Output Area	81Hex	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	**00Hex**	81Hex								

9. Whole message has been completely transmitted.

4.2 DATA CONSISTENCY WITH DPD DRIVER

Here the application layer still copies the Control Field byte in the last position of the exchange area. Data exchange structure appears slightly different, as shown in following picture:



If the DPD driver is used, barcode messages longer than (InputAreaSize – **5**) will be split into pieces through an automatic fragmentation process.

5 DIGITAL I/O CONDITIONING

This function allows using the reader I/Os as remote peripherals of the Host:

- Digital Input Conditioning allows the reader to notify (echo) the status of its physical inputs to the PLC
- Digital Output Conditioning allows the PLC to force the level of the physical reader outputs

Some of the benefits are:

- Fieldbus Master knows the status of each single reader input, (i.e. a parcel is coming, an oversized piece of baggage has been detected, a piece of baggage has reached the data Tx line, etc.)
- Fieldbus Master is able to force the reader in reading mode
- Fieldbus Master is able to modify the reader outputs status depending on the received barcode message (sorting applications)
- No dedicated software on the reader is necessary to manage its I/Os

NOTE

If any of the Digital I/O Conditioning parameters are enabled, the Flow Control Driver (DAD or DPD) starts at Byte 1 (second byte) of the Input/Output Areas, and Byte 0 of the Input/Output Areas is reserved for Digital I/O Conditioning parameters.

If none of the Digital I/O Conditioning parameters are enabled, the Flow Control Driver starts at Byte 0 (first byte) of the Input/Output Areas.

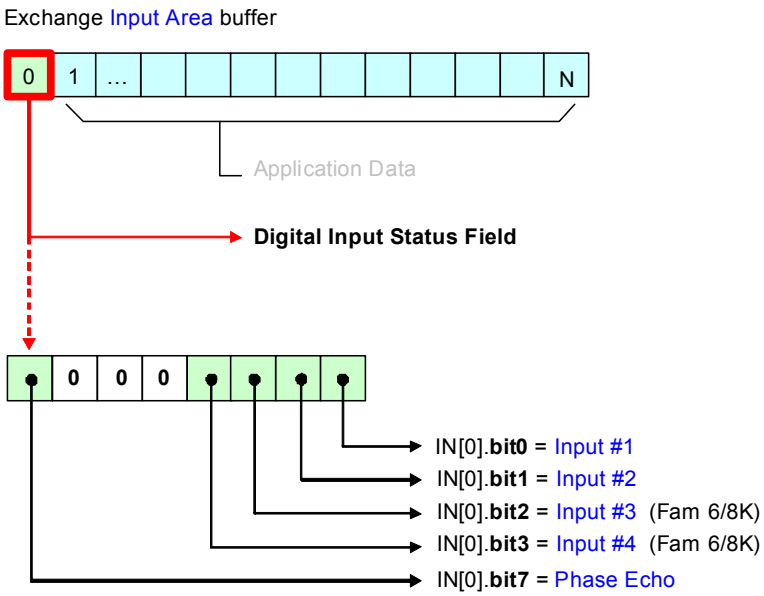
NOTE

The Digital I/O Conditioning functions work independently from the Flow Control options.

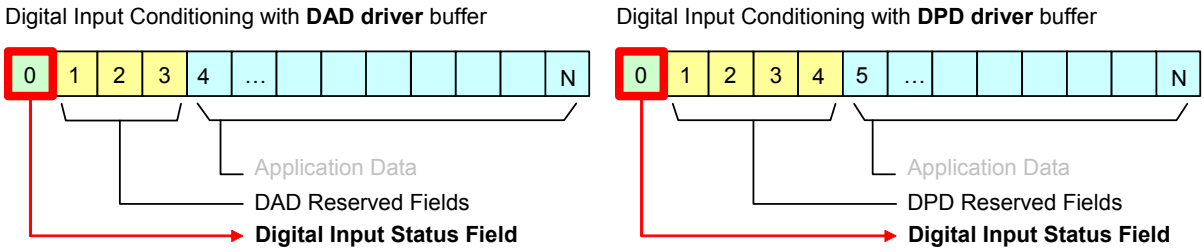
5.1 DIGITAL INPUT CONDITIONING

The status of a reader Digital Input can be echoed to the Fieldbus Master via the relative bit in Byte 0 (LSB) of the Input Area.

Without any Flow Control driver, the following Input mapping is available:



With Flow Control drivers, the Input area structure is as follows:



The Digital Input Status Field is continuously updated by the reader according to the status of the reader's physical inputs. **Only the selected inputs are echoed**.

5.1.1 Phase Echo

The Phase Echo bit echoes the status of the reading phase on the reader which was remotely activated from the Fieldbus Master, see par. 5.2.1 Phase Trigger.

For Scanners, the Phase Echo parameter can be enabled once the Start Input From Fieldbus parameter is selected.

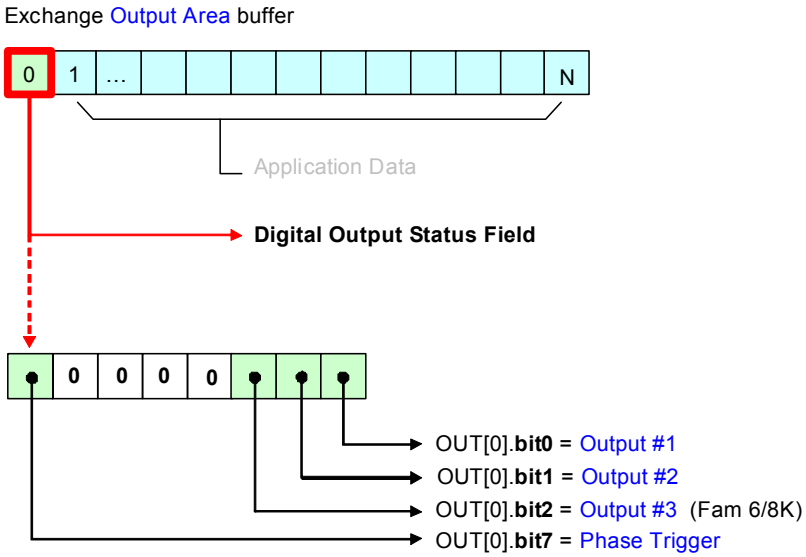
For Matrix readers Phase Echo parameter has meaning only when Operating Mode = Phase Mode and Reading Phase ON is driven by a Fieldbus Input Leading or Trailing Edge.

For configuration details see par 5.3 Reading Phase via Fieldbus.

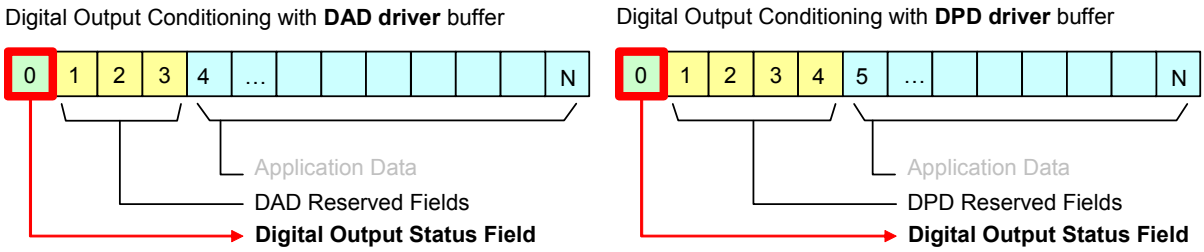
5.2 DIGITAL OUTPUT CONDITIONING

If a reader Digital Output is configured to Use External Fieldbus, then the output can be enabled to allow the Fieldbus Master to drive it via the relative bit in Byte 0 (LSB) of the Output Area.

Without any Flow Control driver, the following Output mapping is available:



With Flow Control drivers, the Output area structure is as follows:



The Digital Output Status Field can be set by the PLC at any time, then the reader's physical outputs will change accordingly. **Only the selected outputs are controlled**.

5.2.1 Phase Trigger

The Phase Trigger bit is used by the Fieldbus Master to drive the reading phase of the reader. Reading activation is provided through the Fieldbus network so there is no need of external photocells or I/0 signals.

For Scanners the Phase Trigger bit is used to activate the reading phase once the Start Input From Fieldbus parameter is selected.

For Matrix readers the Phase Trigger bit is used to activate the reading phase when Operating Mode = Phase Mode and Reading Phase ON is driven by a Fieldbus Input Leading or Trailing Edge.

For configuration details see par 5.3 Reading Phase via Fieldbus.

NOTE

An alternative way to control the reading phase via Fieldbus is:

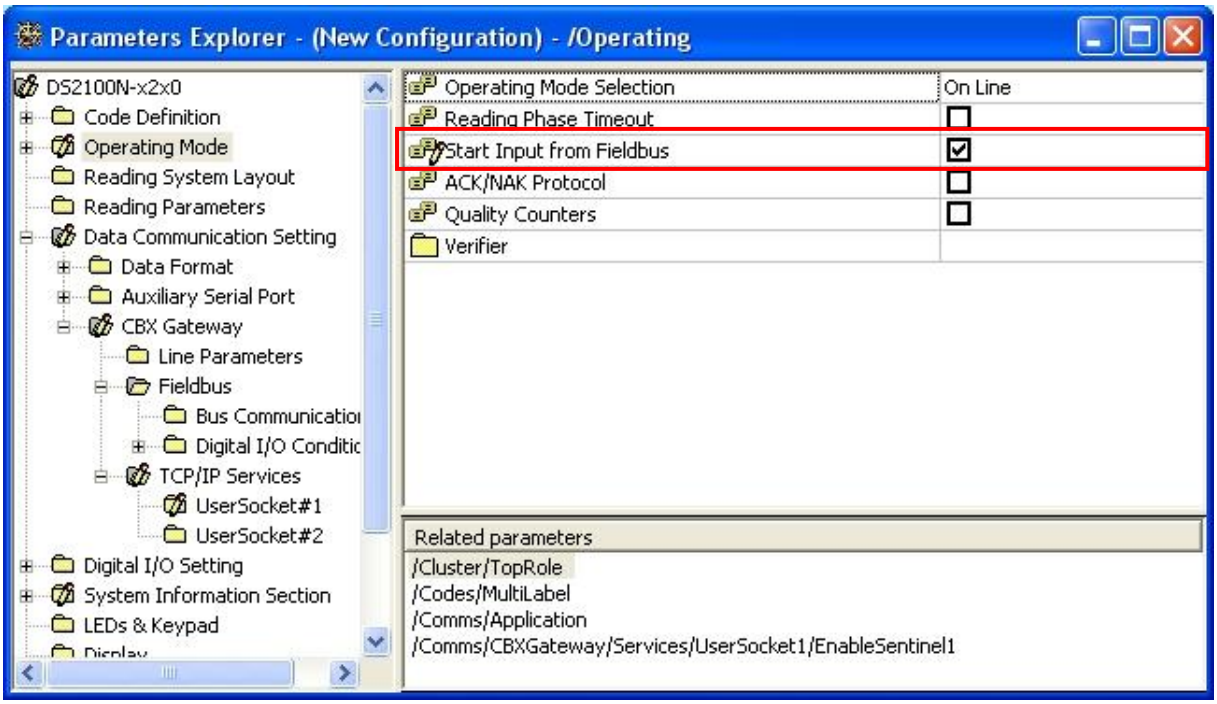
Scanners: setting the reader to operate in Serial On Line mode and send the user programmable Serial Start String and Serial Stop String through the PLC.

Matrix: set Operating Mode = Phase Mode and Reading Phase ON = Fieldbus String; set the Reading Phase ON String and Reading Phase OFF String. Send the user programmable strings through the PLC.

5.3 READING PHASE VIA FIELDBUS

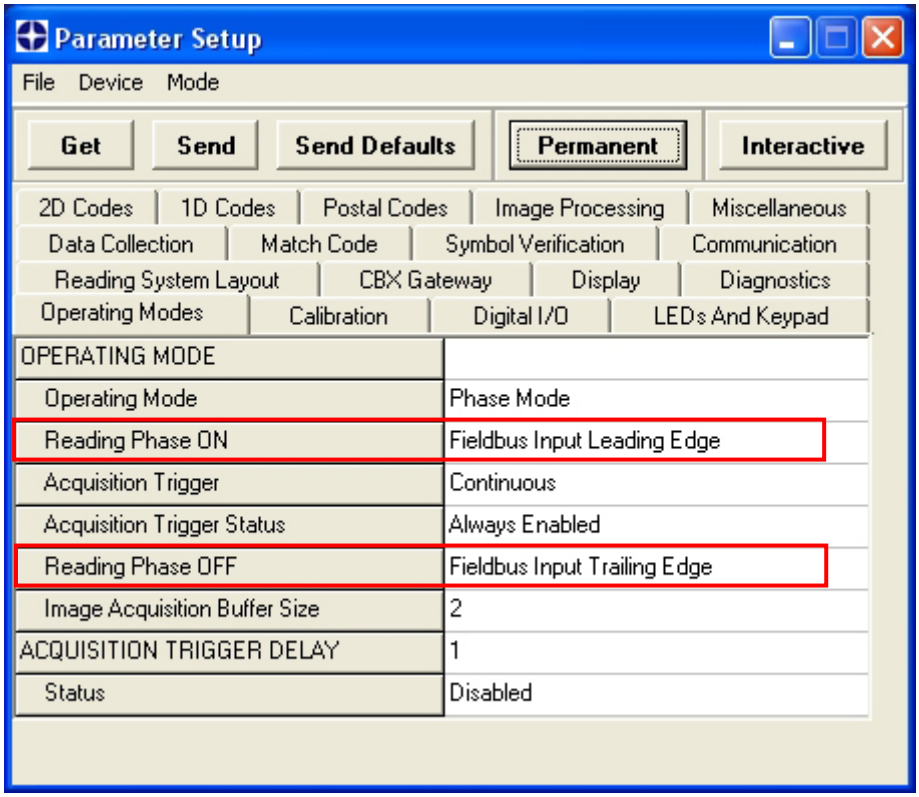
To configure reading phase control from the Fieldbus Master, follow the indications below using the reader configuration tool. More details are given in the specific product Help-On-Line.

For Scanners check the Start Input from Fieldbus parameter in the *Operating Modes* folder so the scanner understands that the phase trigger is going to be provided via Fieldbus.



Send the modified configuration to the scanner permanent memory.

For Matrix readers set the Reading Phase ON and Reading Phase OFF parameters to be driven by a Fieldbus Input Leading or Trailing Edge.



Send the modified configuration to the matrix permanent memory.

Now the reading phase is controlled as follows:

Reading phase starts as soon as the Fieldbus Master sets OUT[0].bit7

Reading phase stops as soon as the Fieldbus Master clears OUT[0].bit7

6 NETWORK CONFIGURATION

6.1 CONFIGURATION FILES FOR FIELDBUS

A Fieldbus configuration file is a readable ASCII text file that contains a complete description of the specific device. It basically includes both general info (i.e. vendor and device name, hw/sw releases) and device specific info (Input and Output area size).

Powerful configuration tools can be used to setup a Fieldbus network (i.e. Siemens SIMATIC Manager for Profibus). Based on the configuration files, these allow easy configuration of Fieldbus networks with devices from different manufacturers.

Note: A configuration file must first be installed into the PLC environment in order to let a new device be identified and to work on the Fieldbus network.

The CD-ROM is equipped with the following files:

- HMS_1811.GSD (Device description file for Profibus)
  for Standard Application Package 2K4K Rel 001 or Matrix SAP any version
- DLA_0BAC.GSD (Device description file for Profibus)
  for Standard Application Package 2K4K Rel 002 or later
- 324-9751-EDS_ABCC_DEV_V_2_2.EDS (Device description file for DeviceNet)
- 368-0164-EDS_ABCC_EIP_V_2_1.EDS (Device description file for Ethernet/IP)
- GSDML-V2.0-HMS-ABCC-PRT-20080602.XML (Device description file for Profinet)
- 334-2605-EDS_ABCC_COP.EDS (Device description file for CANopen)

The Fieldbus configuration file is a certified part of the device and must not be changed manually. This file is also not changed by the configuration tool.