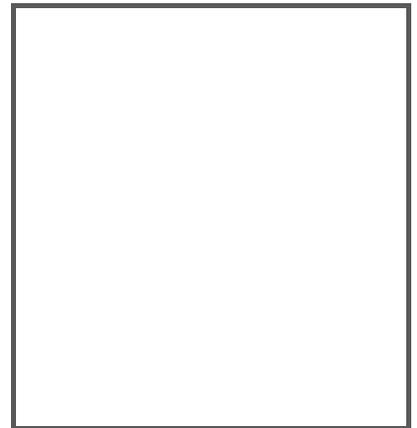


MANUAL
DAD DRIVER





With regard to the supply of products, the current issue of the following document is applicable: The General Terms of Delivery for Products and Services of the Electrical Industry, published by the Central Association of the Electrical Industry (Zentralverband Elektrotechnik und Elektroindustrie (ZVEI) e.V.) in its most recent version as well as the supplementary clause: "Expanded reservation of proprietorship"



CONTENTS

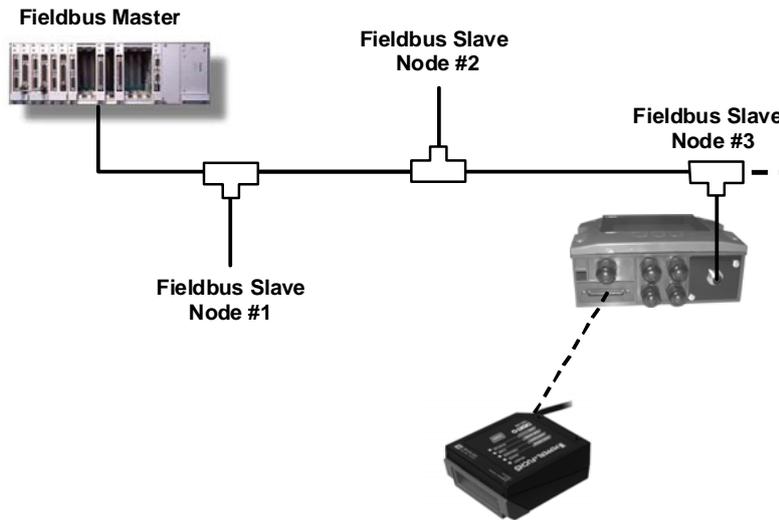
1	BUS COMMUNICATION	4
1.1	Data Exchange	4
1.2	Data Flow Control Mode (FCM).....	5
1.2.1	Flow Control drivers.....	6
1.2.2	FCM with DAD Driver	7
	Control Field	8
	SAP Field.....	9
	Length Field.....	9
	Data Transmission from the Reader to PLC.....	10
	Data Transmission from PLC to the Reader.....	12
	Resynchronisation	13
	Fragmentation and Reassembling.....	15
	SAP Services.....	17
	DAD internal queues.....	17
2	NETWORK CONFIGURATION	18
2.1	Configuration files for Fieldbus	18

1 BUS COMMUNICATION

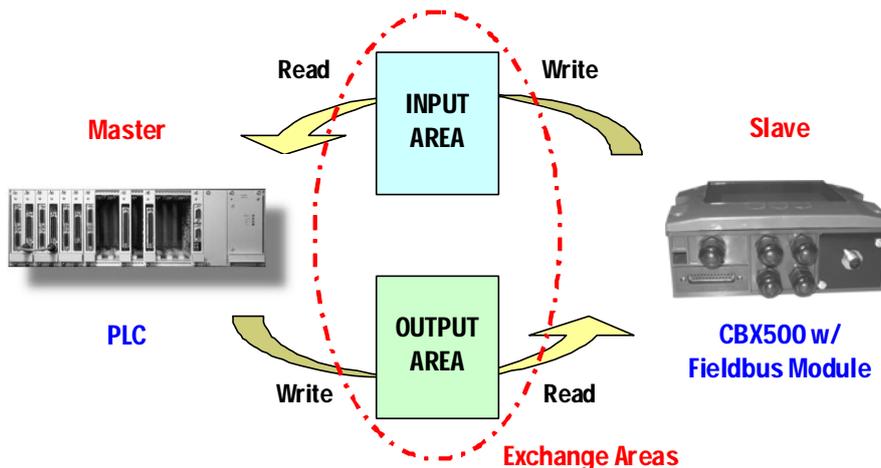
1.1 DATA EXCHANGE

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

CBX500 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 CBX500.



Input and output areas always refer to the Master: this means that the scanner 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 Genius™.

1.2 DATA FLOW CONTROL MODE (FCM)

The Data 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.

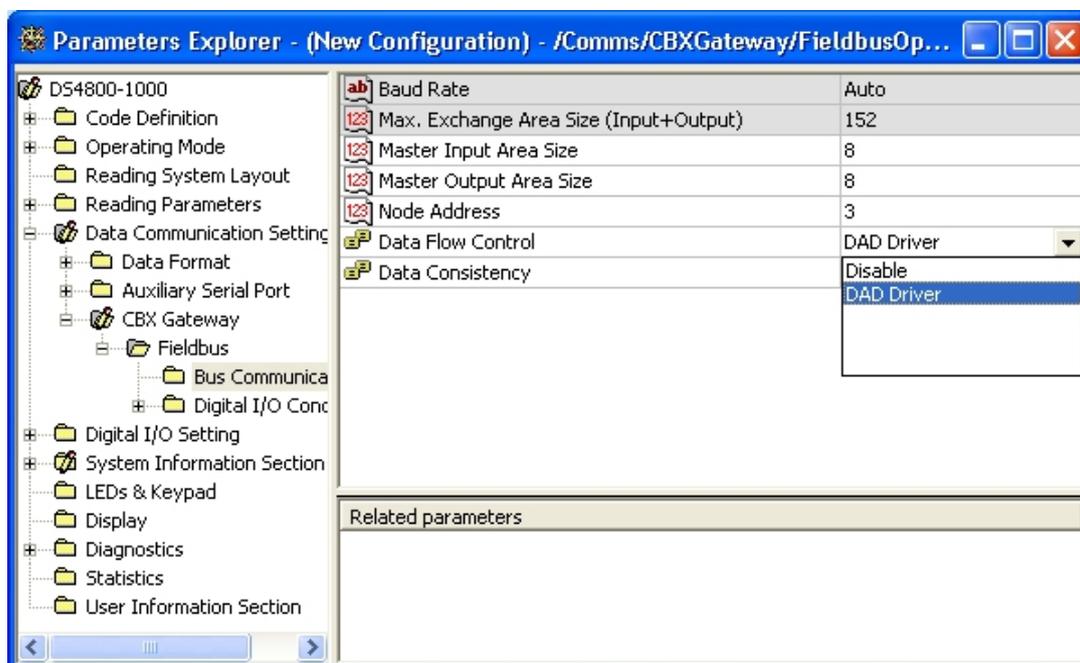
The reserved bytes are used to implement many different features such as:

- 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 Data 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 in Genius™.



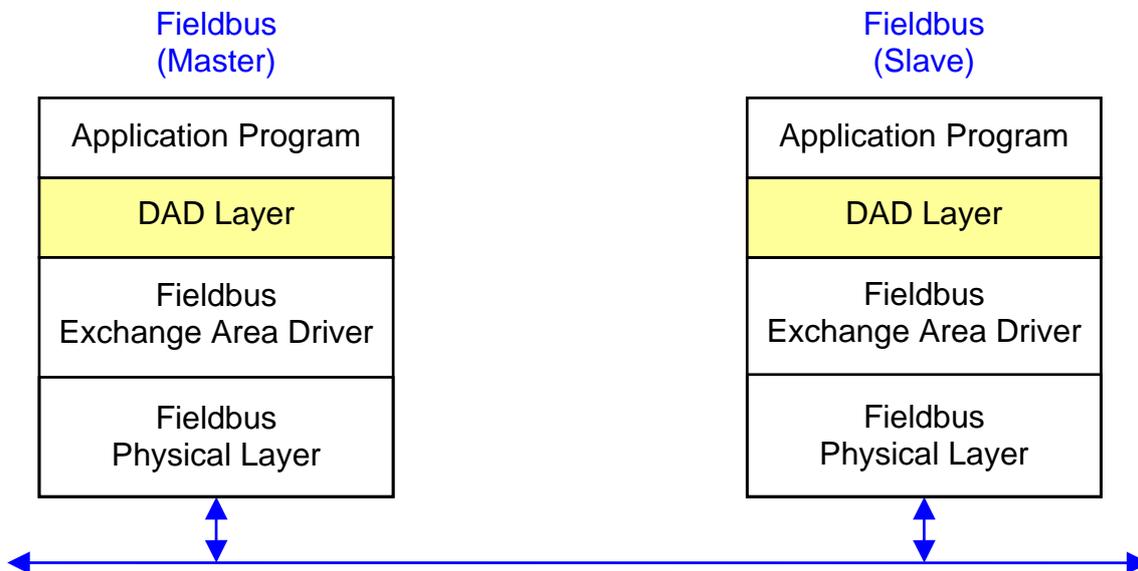
Basically two options are available:

- **Disable** → *No Data Flow Control*
- **DAD Driver** → *Data Flow Control Enabled*

1.2.1 Flow Control drivers

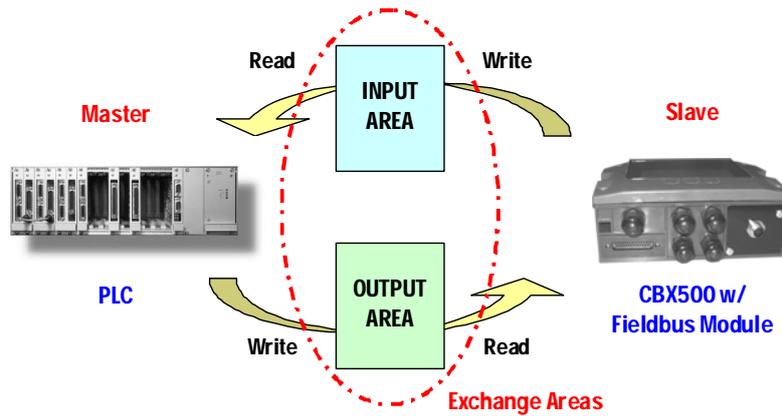
The Flow Control driver is a layer that is built upon the intrinsic data exchange mechanism. Basically such a layer is required because the intrinsic data exchange mechanism is not message oriented.

In the following figure the complete Stack is represented:



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

1.2.2 FCM with DAD Driver



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



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



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).

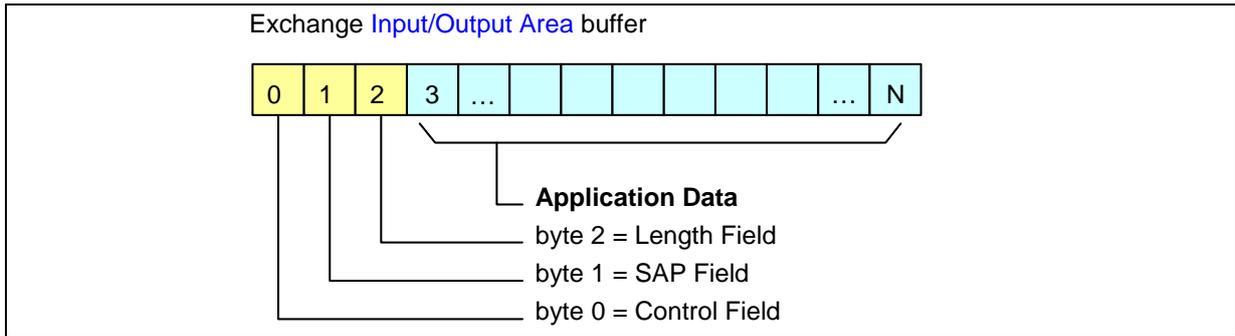
Length Field
(byte 2)

contains the number of bytes used by the application layer:
 $\text{Length Field} \leq (\text{InputAreaSize} - 3)$ for the Input Area
 $\text{Length Field} \leq (\text{OutputAreaSize} - 3)$ for the Output Area.



NOTE

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. See 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).

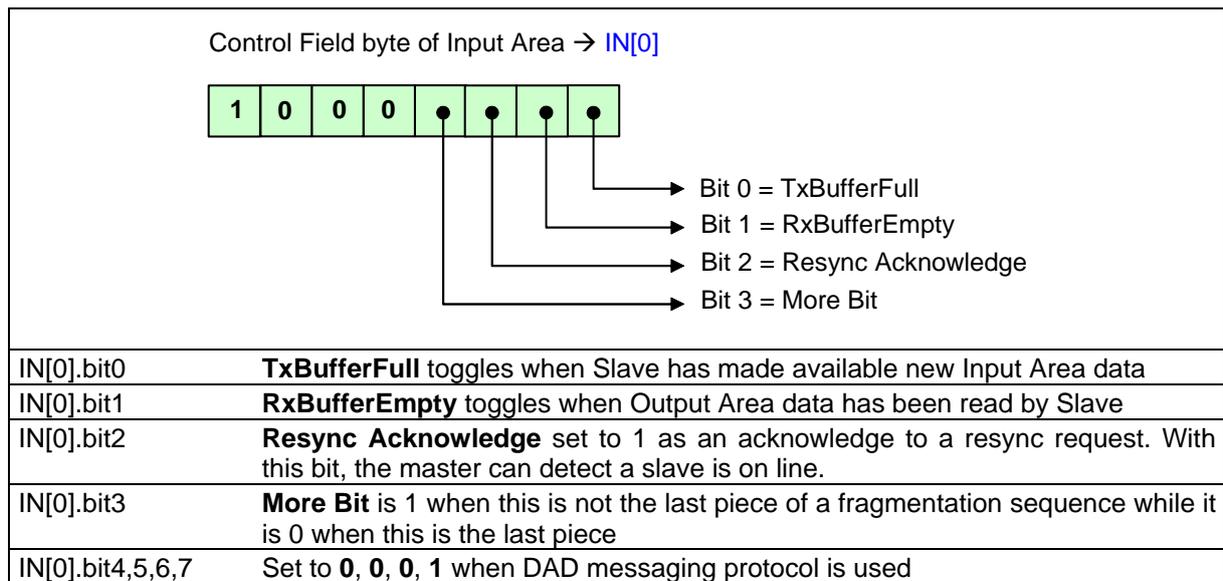
CONTROL FIELD

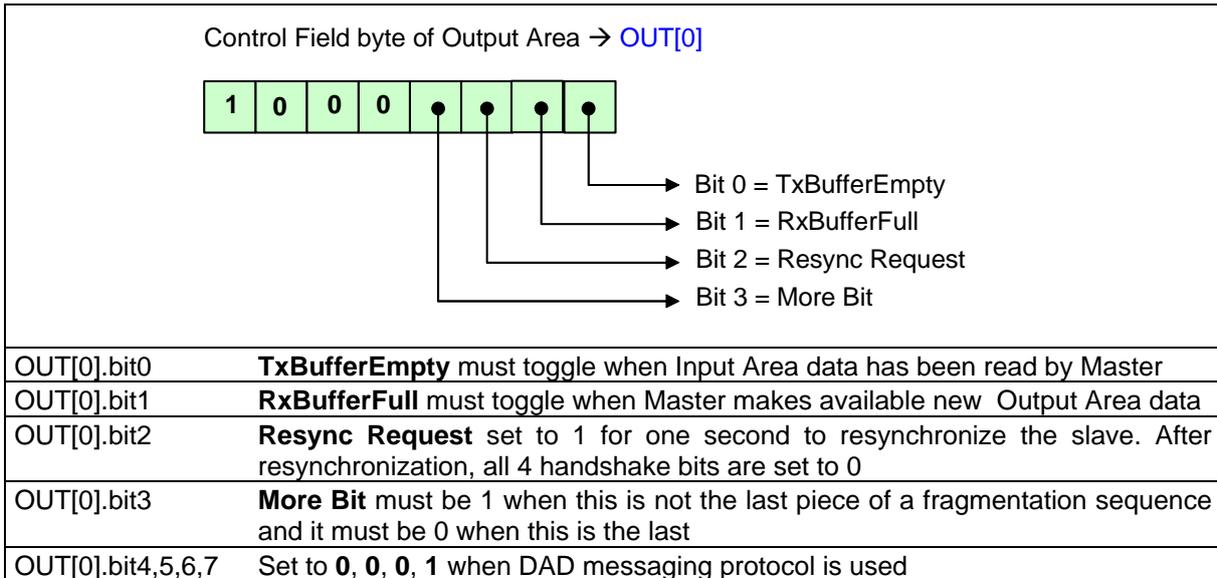
This is the core of the flow controlled communication.

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.





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 the "SAP Services" paragraph)

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

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.

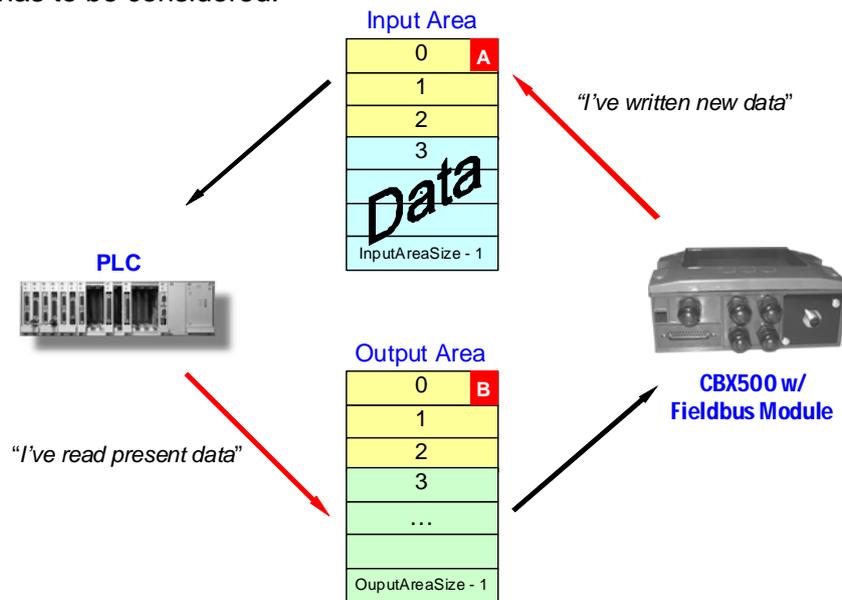
A fragment that is not the last one of a fragmentation sequence must fill this field with [InputAreaSize – 3] (or [OutputAreaSize - 3]), depending on whether it is an Input/Output fragment. Otherwise this field gets a value that is less than or equal to [InputAreaSize – 3].

DATA TRANSMISSION FROM THE READER TO PLC

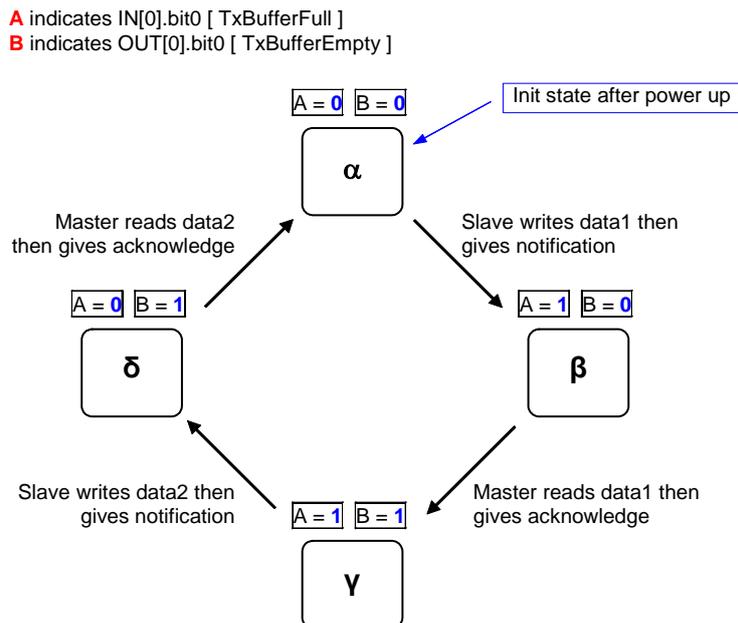
This paragraph describes how it is possible to exchange messages with flow control. The communication mechanism is simple:

- **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}														
Output Area	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}														
Output Area	80 _{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}														

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}														

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}					
Output Area	81 _{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}					
Output Area	80 _{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}								
Output Area	80 _{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}								
Output Area	81 _{Hex}	00 _{Hex}														

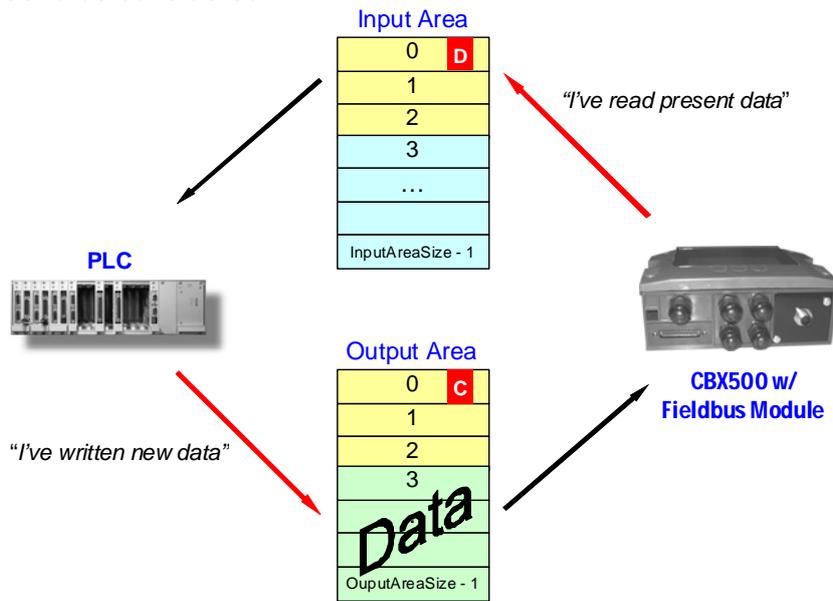
9. Data exchange continues...

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:

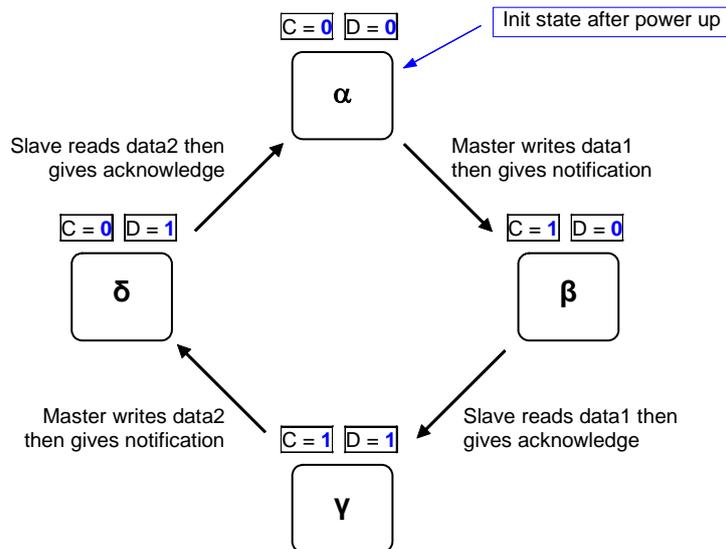
- ❑ **OUT[0].bit1 [C]** must be used by PLC to notify that “*Master has written new data so Slave can read it*”
- ❑ **IN[0].bit1 [D]** is used by the reader to notify that “*Slave has read data so Master can send next message*”

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.

C indicates OUT[0].bit1 [RxBufferFull]
D indicates IN[0].bit1 [RxBufferEmpty]

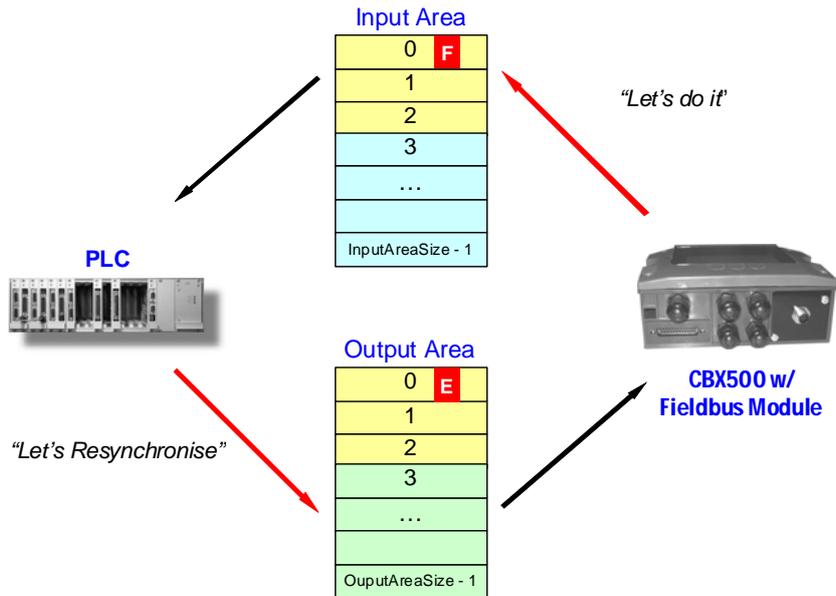


RESYNCHRONISATION

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.

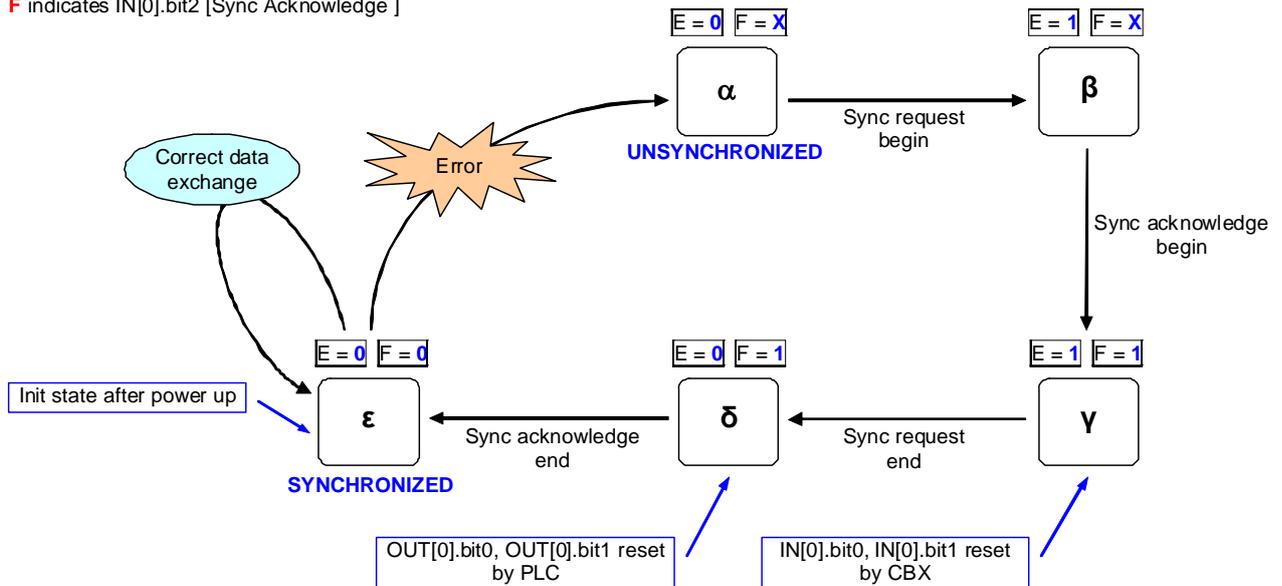
The process is based on bit two:

- **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:

E indicates OUT[0].bit2 [Sync Request]
F indicates IN[0].bit2 [Sync Acknowledge]



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}								
Output Area	81 _{Hex}	00 _{Hex}														

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

Input Area	81 _{Hex}	00 _{Hex}	04 _{Hex}	02 _{Hex}	18 _{Hex}	0D _{Hex}	0A _{Hex}	00 _{Hex}								
Output Area	85 _{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}								
Output Area	85 _{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}								
Output Area	80 _{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}								
Output Area	80 _{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.

FRAGMENTATION AND REASSEMBLING

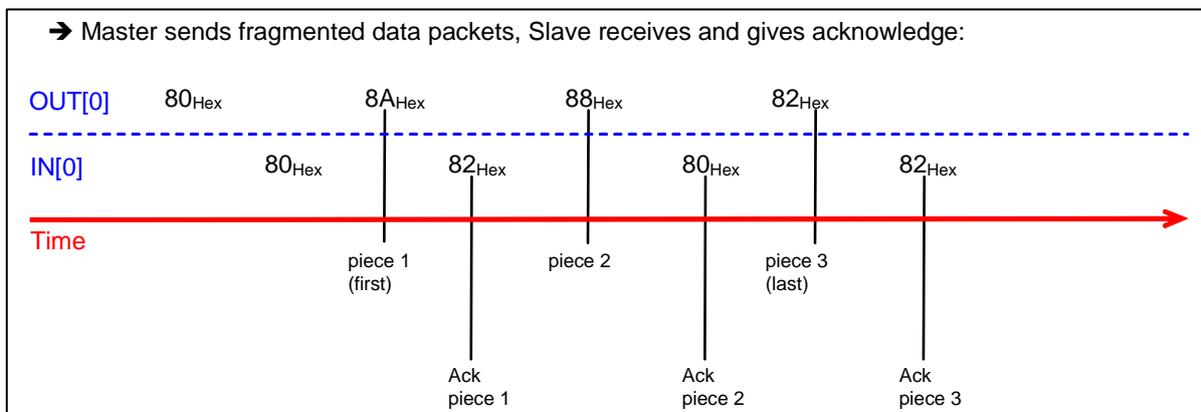
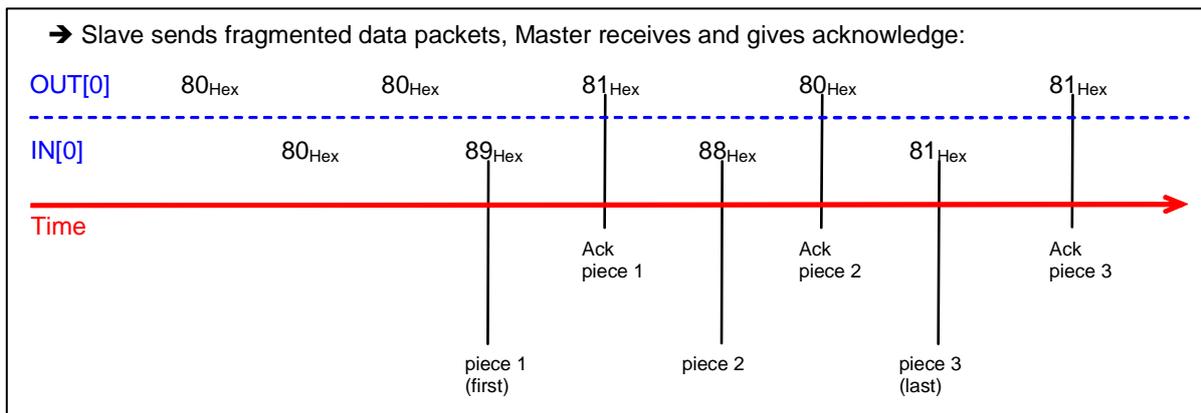
The fragmentation process is activated whenever Application Data cannot be contained in the related exchange area. Basically long messages are split into pieces which are transmitted separately. Reassembling allows the reconstruction of the whole messages. The reader already implements these functions 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}														
Output Area	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}														
Output Area	80 _{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}														

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}														

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}														

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}														

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}					
Output Area	80 _{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}					
Output Area	81 _{Hex}	00 _{Hex}														

9. Whole message has been completely transmitted.

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)

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.

2 NETWORK CONFIGURATION

2.1 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 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.



FACTORY AUTOMATION – SENSING YOUR NEEDS



Worldwide Headquarters

Pepperl+Fuchs GmbH
68307 Mannheim · Germany
Tel. +49 621 776-0
E-mail: info@de.pepperl-fuchs.com

USA Headquarters

Pepperl+Fuchs Inc.
Twinsburg, Ohio 44087 · USA
Tel. +1 330 4253555
E-mail: sales@us.pepperl-fuchs.com

Asia Pacific Headquarters

Pepperl+Fuchs Pte Ltd.
Company Registration No. 199003130E
Singapore 139942
Tel. +65 67799091
E-mail: sales@sg.pepperl-fuchs.com

www.pepperl-fuchs.com

 **PEPPERL+FUCHS**
SENSING YOUR NEEDS