W+D DIRECT MARKETING SOLUTIONS

KÖRBER PAPERLINK

IBC System Controller
Manual

Loehne, 2007-06-22

Created: **2006-11-07 HF**

Loehne, 2007-06-22

Last modifyed: 2008-02-22 HF

CONTENT

1. INTRODUCTION 5
2. SYSTEM CONTROL SETUP 6
2.1 Selection of Read Data for System Control 6
2.2 Global Preferences for System Control 7
2.3 Definition of Code Segments 9
2.4 Function assignment to selectable sub segments 12
2.5 Description of Available Functions 14
0 --- No function, delete function 14
1 - ID for collation 14
2 - Matchcode 14
5 - Sequence check sheets 14
6 - Sequence check in set 14
35 - Internal eject 15
39 - Size of set 15
40 - End of set 15
41 - Start of set 15
42 – Match code on first page 15
47 - Read verify: One bit set 15
48 - Read verify: Two gate marks 15
49 - Modulo 10 Checksum 16
50 - Modulo 10 Checksum weight 3-1-3-1 right -> left 16
51 - Parity check even 16
52 - Parity check odd 16
53 - DGR Final subset 16
54 - Verify job ID 17
55 - Selection for slave channel1 17
56 - Size of set Y, collating sheet X of Y 17
57 - Sheet No. X, collating sheet X (of Y) 17
58 – End of subset 17
59 – Match code on last page 17
60 - Sheet No. X, collating sheet X of Y descending 17
61 - Modulo 10 Checksum weight 3-1-3-1 right -> left 18
7 to 23 and 31 – Mueller Fct X 18
3 - Bundle change 18
4 - ZIP code / Bundle # 18
7 to 30 – Selection Feeder X 18
31 – Eject gate # 19
32 – Don't seal 19
33 – Postal meter # 19
34 – Tacker 19
36 – Change tractor 19
37 – Selection marker 1 19
38 - Selection marker 2 19
43 – Start of job 19
44 – End of job 20
45 - Stop machine 20
46 - Empty machine 20
3. TESTING AND DEBUGGING 21

3.1 Extended Display of Processed Data..... 21
3.2 Description of Error Flags 22

1. INTRODUCTION

This introduction is a supplement to the current manual of the **I**ntelligent **B**uhrs **C**ontroller **IBC.** It describes the special functionality for controlling Mueller system channels by reading codes.

Axode d:cod, Asentics CRxxx, Leuze BCL 22 or Betron Neurocheck camera systems can be used.

IBC controls the communication with the mail inserter, with the reading system and with the Mueller system channel. This enables a very flexible mail processing.

The IBC system control covers almost all common control codes. A change of the system channel control software is usually not necessary.

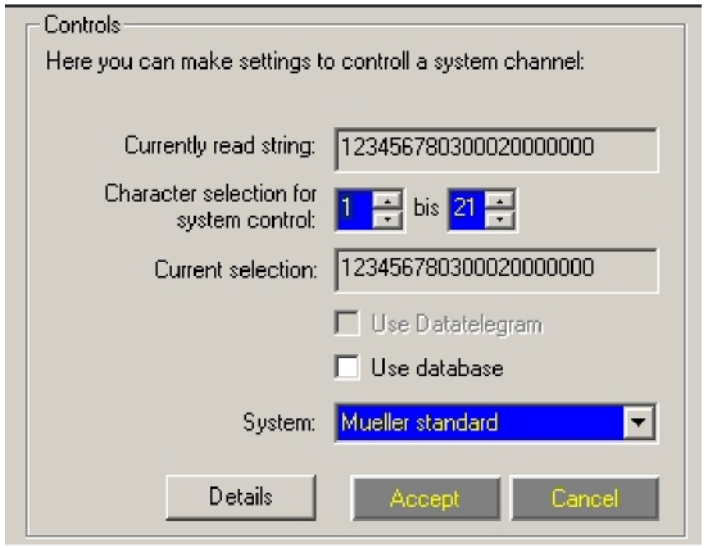
The IBC software is continuously enhanced and extended. New functions will be added if necessary. Downward compatibility is preserved in new versions. Upward compatibility is preserved as far as possible. In this case a program, which is saved by a new version can be loaded and used by an old version. New functions will cause errors and need to be deleted.

The Mueller system channels, to be controlled, must have the control software version CA1706BU or CA2006 installed, or a compatible one. The accumulator must be configured for collation with “start of set”. The corresponding H- bridge must be installed. Optionally the channel can be controlled using “end of collation”. This can have a few mechanical advantages. This mode of operation can be enabled by an entry in the INI file. In this case the H- bridge must be removed if EOC data is sent to the channel and it must be installed, if BOC data is sent. The current method of control can be determined observing the data in the second line of the output window. The 12th character will be “B” if BOC is sent, and it will be “E” if EOC is sent.

2. SYSTEM CONTROL SETUP

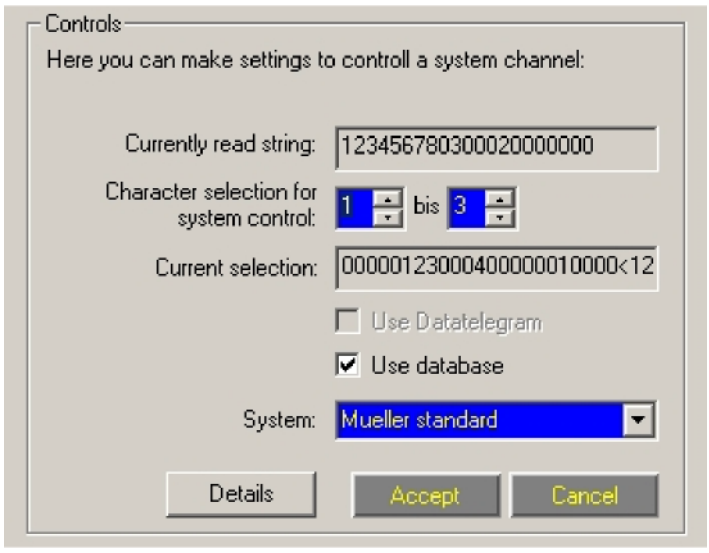
2.1 Selection of Read Data for System Control

The reading equipment is described in separate documents. To setup the reading please read the corresponding manuals and the IBC manual. After the reading is setup and a valid reading result is available in edit mode of the IBC, select the part of the read data, which will be used to control the Mueller channel:



Img. 1

The option Use Datatelegram should be used on Neurocheck based Betron-camera systems only and should else be disabled. Use database can be used to lookup a control code from a database. If activated, both, the data from the database and the originally read data can be used to control the system channel. The Current selection shows the data from database first and the original reading result ist appended using a '<' as a separation mark.

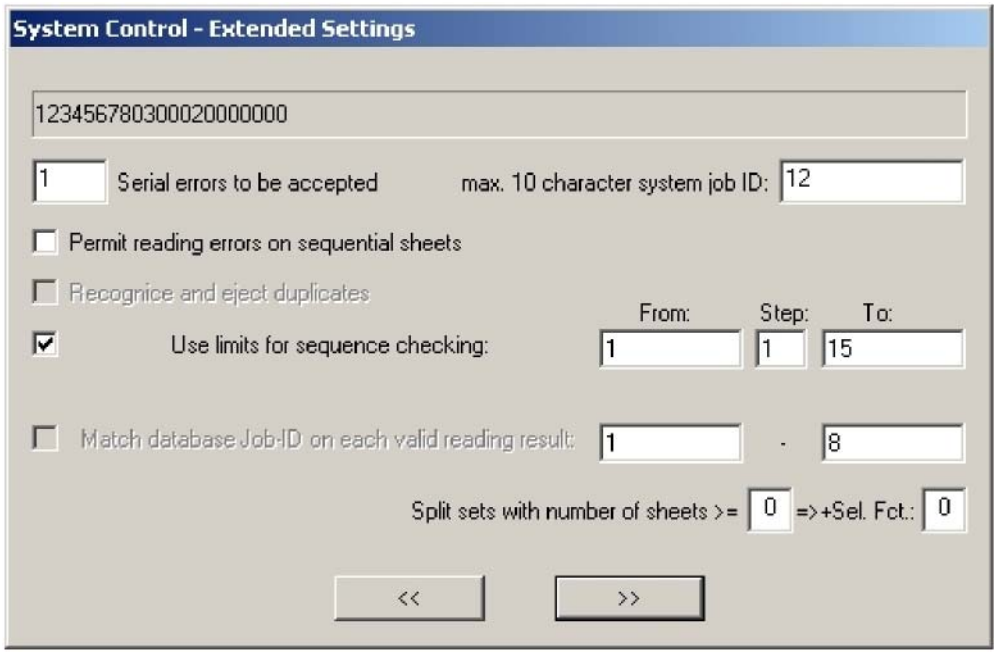


Img. 2

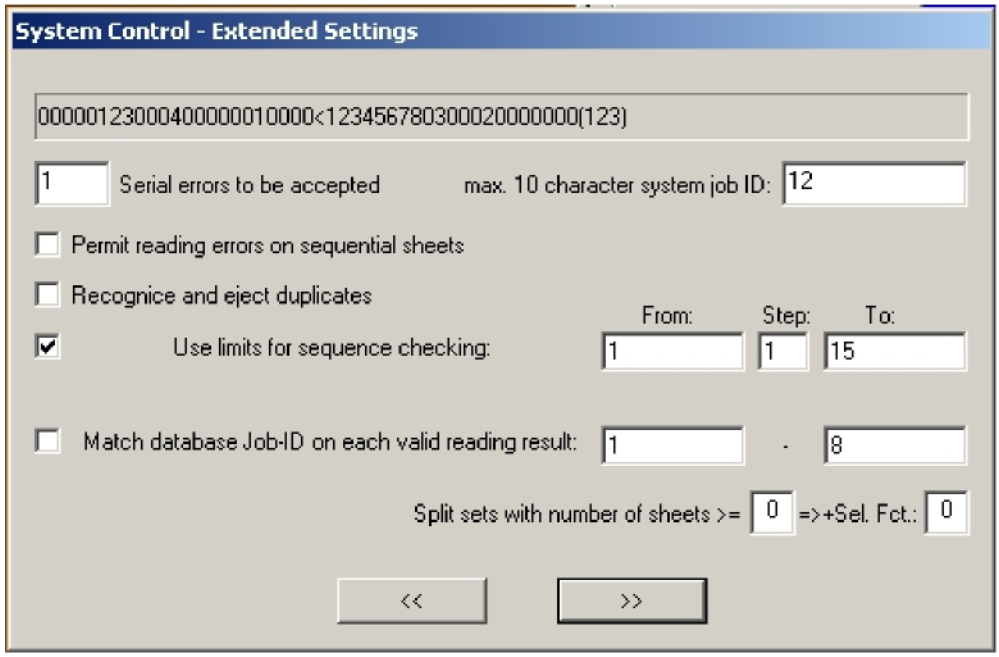
A variable number of setup dialogs can be entered using the Details button.

2.2 Global Preferences for System Control

After pushing the Details button, a dialog for global preferences is opened:



Img. 3 No use of database



Img. 4 Database used

"00000 123000400000010000" is retrieved from database, '<' is a separation mark followed by the originally read data "123456 780300020000000". At last the used index to retreive data from database is shown in braces "(123)". The data from database and the original reading result can both be used to control the system.

1) Serial errors to be accepted

A serial number of 1 causes the system channel to stop with every error that appears. A value of 3 will cause a stop after the third serial error. Nevertheless IBC will track the errors which did not stop the channel. This enables to continue production despite of sporadic reading errors. The faulty sets will be tracked and can be ejected.

2) Max. 10 character System Job ID

The entered job ID is assigned to each processed set in the channel. It can be used for form type matching in the inserter. Additionally a part of the reading result can be matched to this ID in the channel. Faulty sets can be ejected.

4) Permit reading errors on sequential sheets

On certain methods of collation like function 39 or 57, reading errors can be permitted on sequential sheets. This allows to process jobs where only the master page contains data. In this case only reading errors on the first page of a set will flag the set as bad.

6) Recognize and eject doublets

This option is only available, if IBC is used with the optional database. Doublets can appear after an error and restart on laser printers which print documents on a roll of paper. The double processed sets can be recognized and ejected.

8) Use limits for sequence checking

This is a global setting for the various sequence checking possibilities, which will be described later in this document. If the check box on the left is activated, the limits of the sequence code will be observed. Otherwise only the Step will be used.

10) Match database Job-ID on each valid reading result

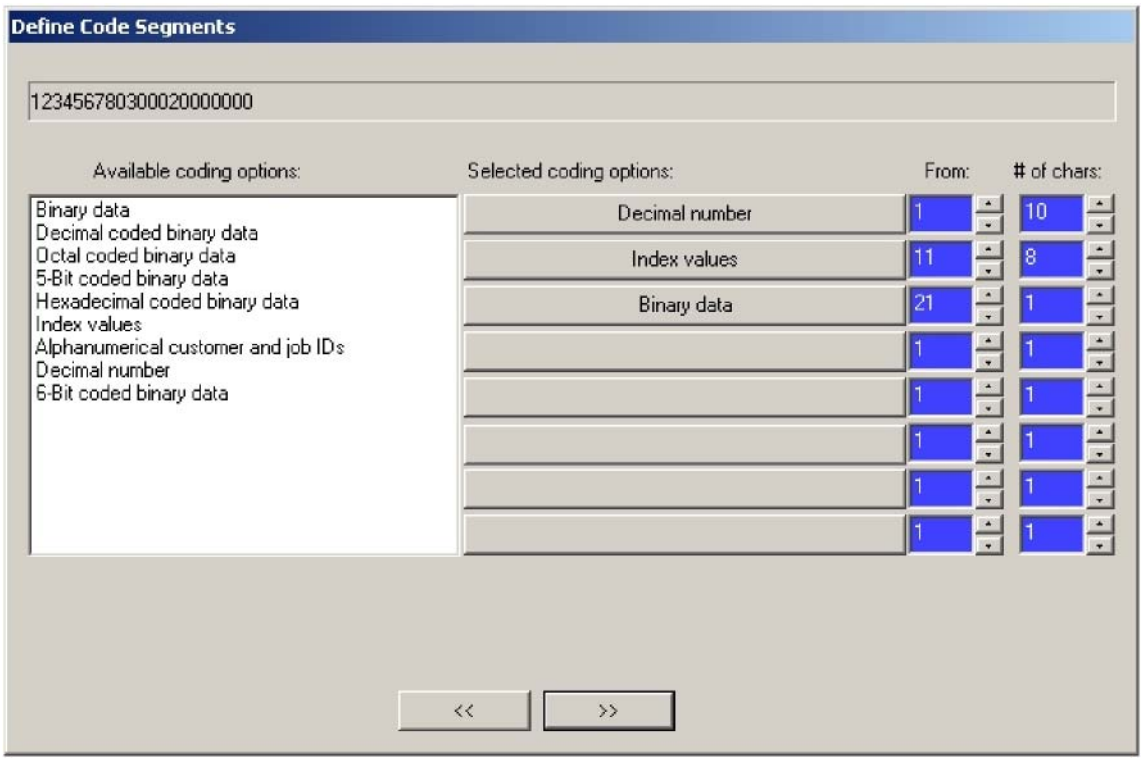
This option is only available, if IBC is used with the optional database. The read data must contain the name of the used database between the selected positions. Mismatches are treated like reading errors.

12) Split set with number of sheets >= … => + Sel.Fct.: …

Automatic set splitting after a certain number of sheets is enabled if the first value is >0. Optionally a selective function can be assigned to the split subsets.

Use the button << to get to the previous dialog or the button >> to get to the next dialog.

2.3 Definition of Code Segments



Img. 5

The read data or the data which are retrieved from the database must be assigned to segments of different coding. Currently eight types of segments can be used like shown in the list on the left. A double click with the left mouse button on one of the available coding options, adds the selected coding to the list of Selected coding options. A double click on one of the buttons with the selected coding deletes it from this list. From: defines the start position of the segment in the read data, # of chars: defines the segment size in characters.

1) **Binary data**

   This data format is often the result of OMR reading. Present marks are represented by ‘1’ missing marks are represented by ‘0’. Only these two digits are allowed in the selected segment. It is interpreted as a binary number. Other digits than ‘0’ or ‘1’ are handled like reading errors.

   Example for binary data: 100011101100111001101

2) **Decimal coded binary data**

   This segment is a decimal number which needs to be converted into a binary number to assign functions to certain bits of the result. The converted data can be used like described above. The table on the next page shows a few examples of decimal numbers and the corresponding binary numbers :

Decimal	Binary
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000
255	11111111
1027	10000000011

The conversion is always done using the maximum number of digits. This means for example a four digit decimal number is max. 9999 which is binary 10011100001111. A 0000 is converted to the binary value 00000000000000 with the same number of binary digits. Each bit or a block of bits of the binary result can later be assigned to different fu nctions.

3) Octal coded binary data

Octal coded data consists of digits between 0 and 7. Other digits lead into handling like reading errors. The least significant bit is on the right, the most significant on the left. Examples for octal numbers:

Decimal	Octal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	10
9	11
10	12
16	20
33	41

Each octal digit is converted into three digits in the binary number. An octal 1234 results in a binary 001 010 011 100. Each bit or a block of bits of the binary result can later be assigned to different functions.

4) 5-Bit coded binary data

This segment can contain the digits and characters 0-9 and A-V. Each character represents five bits. The value 10AV is converted into binary 00001 00000 01010 11111. Each bit or a block of bits of the binary result can later be assigned to different functions.

5) Hexadecimal coded binary data

This segment can contain the digits and characters 0-9 and A-F. Each character represents four bits. The value 10AF is converted into binary 0001 0000 1010 1111. Each bit or a block of bits of the binary result can later be assigned to different functions.

6) Index values

This segment can contain the digits and characters 0-9 and A-Z. Each character is an index to a bit in a 35 bit binary number to be set to '1'. The index 0 sets no bit! The value 16AZ results in the following binary data 10000000000000000000000001000100001. Each bit or a block of bits of the binary result can later be assigned to different functions.

This is an extension to the former feeder selection code logic used by BNC, which allowed only the digits 0-9 to be used.

7) Alphanumerical customer and job IDs

This kind of segment should only be used for match codes, customer IDs and job names.

8) Decimal number

Decimal numbers are typically the most often used codes Only the digits 0-9 are allowed.

9) 6-Bit coded binary data

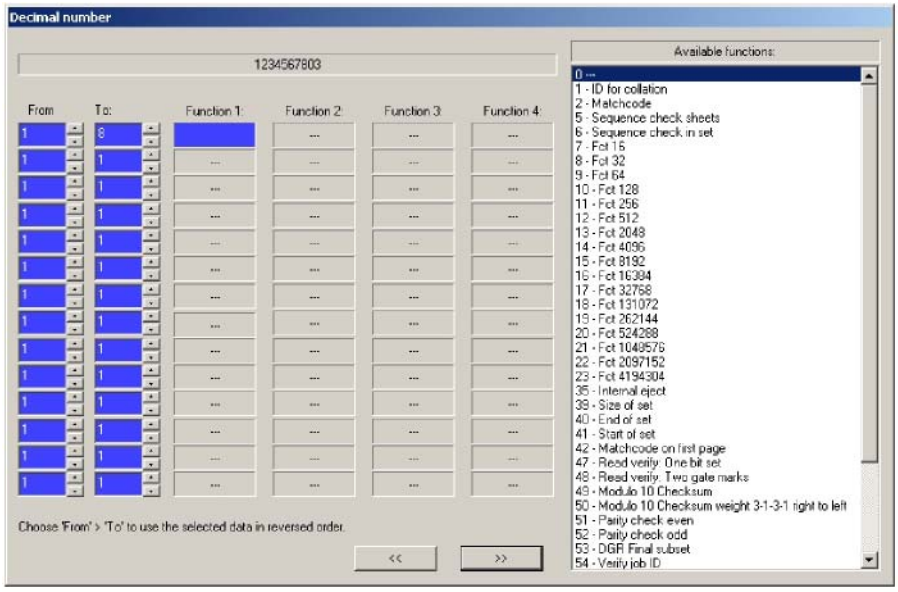
This segment can contain the digits and characters '0' – '9' (values 0-9), 'A' – 'X' (values 10 – 35), 'a' – 'x' (values 36 - 61), '$' (value 62) and '+' (value 63). Each character represents five bits. The value 10AV is converted into binary 000001 000000 001010 011111. Each bit or a block of bits of the binary result can later be assigned to different fu nctions.

Use the button << to get to the previous dialog or the button >> to get to the next dialog.

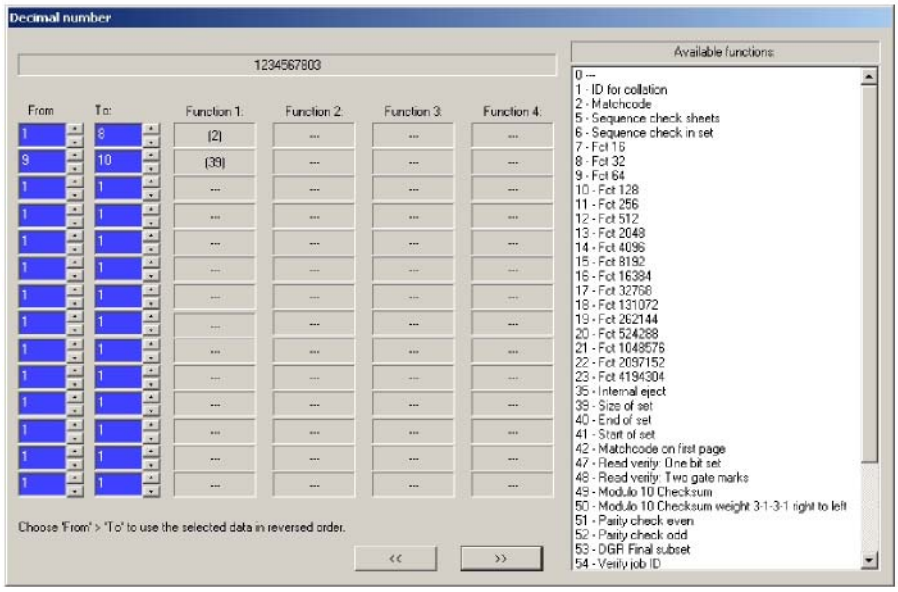
2.4 Function assignment to selectable sub segments

For each defined segment according section 2.3 of this document, a setup dialog is shown, where the corresponding segment can be assigned to a choice of functions using sub segments. The title bar of the dialog window shows the type of segment to be setup. Up to four functions can be assigned to each sub segment. The selected segment data is displayed in the field above. Function assignment starts at Function1 of sub segment 1. After that function 2 of the first sub segment or function 1 of the second sub segment can be assigned etc. A function is assigned by clicking on the function field using the left mouse button and then a double click on an item in the list of Available functions on the right.

Img. 6



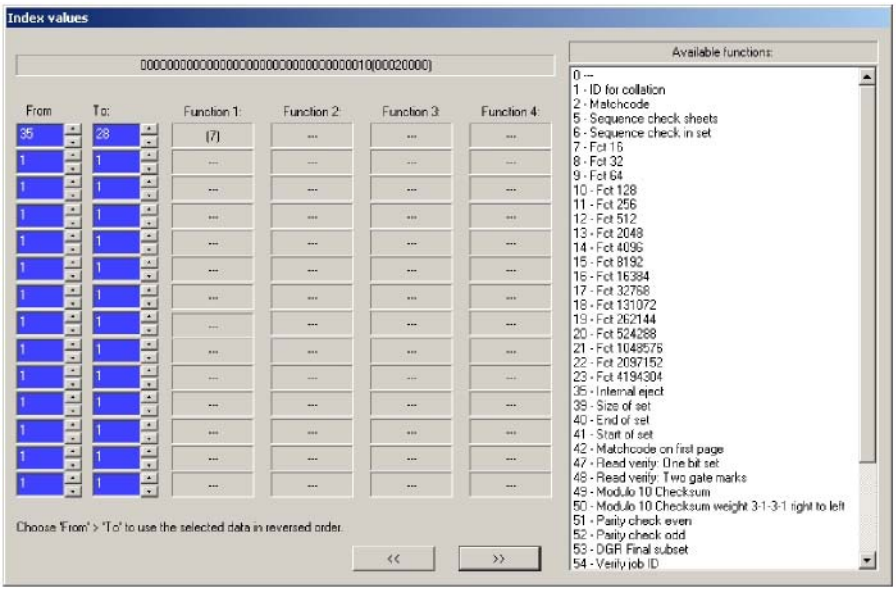
Img. 7



Function 0 --- is used to delete a function assignment.

Use the button << to get to the previous dialog or the button >> to get to the next dialog.

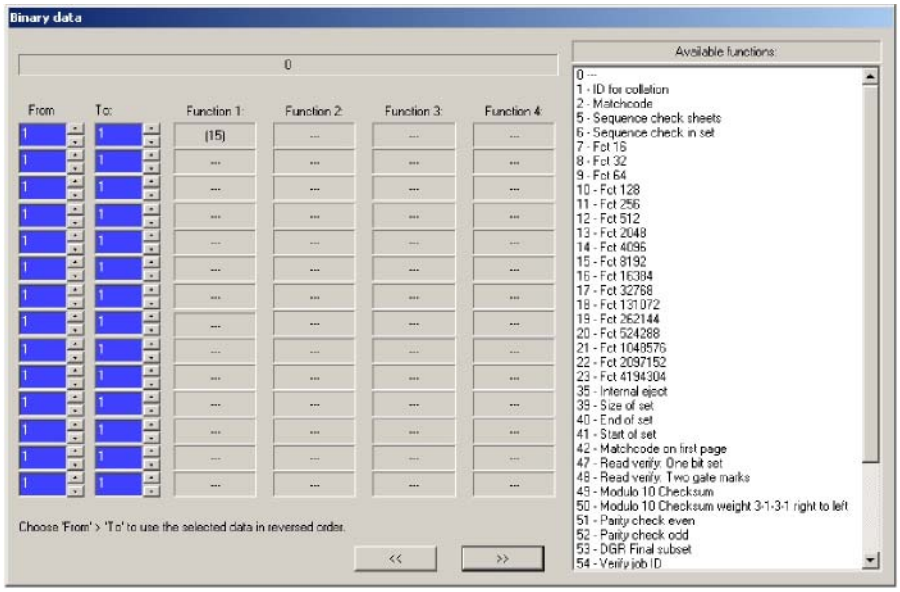
Img. 8



In this example the setup dialog for the segment “Index values“ comes up. The original data of the segment is shown in braces on the right. The 35- bit binary data on the left. The assignment „From: 35 To: 28“ to the function 7 reverses the bit order of the lowest eight bits and assigns them to the functions 7 to 14. The least significant bit is assigned to function 7, the most significant bit is assigned to function 14.

Use the button << to get to the previous dialog or the button >> to get to the next dialog.

Img. 9



In this sample finally the setup dialog for „Binary Data (ASCII)“ comes up. The segment contains only one digit, which will be used to control selective product eject.

Use the button << to get to the previous dialog or the button >> to get to the last dialog, as all segments are now assigned in this sample.

2.5 Description of Available Functions

By configuration currently two sets of functions are supported. Machines which communicate with the IBC using a serial interface RS232 support a reduced set of functions:

0 --- No function, delete function

By assigning this function to a sub segment a previously assigned function can be deleted.

1 - ID for collation

This function is used to collate a variable number of sheets per set, controlled by an ID, which is equal on all sheets of a set. If the ID changes, the collation of a new set is started. Usually the same sub segment is used as the match code in the next function.

2 - Matchcode

The corresponding sub segment contains an ID which is used for matching within a set or for matching with supplements from the feeders of the mail inserter to make sure that all personalized supplements belong to the addressee. This can be account statements, prepared business reply cards or invitations.

5 - Sequence check sheets

The sub segment is a sequence number which counts over all sheets of the job. Limits and step sizes are used as described in section 2.2 under 5). The sequence code can be a decimal or a binary number. If coded binary the sequence number may be up to 31 bits long. If it is a decimal number a maximum of nine digits are allowed.

6 - Sequence check in set

The sub segment is a sequence number which counts over all sheets of the set. Limits and step sizes are used as described in section 2.2 under 5). The sequence code can be a decimal or a binary number. If coded binary the sequence number may be up to 31 bits long. If it is a decimal number, a maximum of nine digits are allowed. The sequence count is reset to the first limit on every first page of a set.

35 - Internal eject

This function controls the internal eject gate of the channel. The corresponding sub segment contains only one digit which must be either ‘0’ or ‘1’. All other digits lead to a handling as reading error. If the digit is ‘1’, the set will be diverted at the eject gate. Else it will be processed as usual.

39 - Size of set

This function is used to collate a variable number of sheets per set, controlled by the sheet count of the set, which is printed on the first page of the set only. Sequential pages have the sheet count 0. The sheet count code can be a decimal or a binary number. If coded binary the sheet count number may be up to six bits long. If it is a decimal number, a maximum of two digits are allowed. The maximum sheet count is 99.

40 - End of set

This function is used to collate a variable number of sheets per set, controlled by an end of set flag which is ‘1’ on the last page of the set only. Leading pages of the set have the flag set to ‘0’. Other digits or characters lead to a handling as reading errors.

41 - Start of set

This function is used to collate a variable number of sheets per set, controlled by a start of set flag which is ‘1’ on the first page of the set only. Sequential pages of the set have the flag set to ‘0’. Other digits or characters lead to a handling as reading errors.

42 – Match code on first page

This function is used to assign a match code to a set of pages, where the match code is printed on the first page of the set only.

47 - Read verify: One bit set

This function performs a read verify. It checks, if all of the digits in the segment are ‘0’ except one of the digits must be ‘1’. Otherwise a reading error handling is launched.

48 - Read verify: Two gate marks

This function performs a read verify. It checks, if all the first and the last digits in the segment are ‘1’. Otherwise a reading error handling is launched.

49 - Modulo 10 Checksum

This function performs a read verify. It checks, if the sum of all single digits is a whole multiple of ten. The sub segment must have a size of at least two digits. Otherwise a reading error handling is launched.

50 - Modulo 10 Checksum weight 3-1-3-1 right -> left

This function performs a read verify. It checks, if the sum of all single digits is a whole multiple of ten. Before the sum is calculated, all digits at odd positions (count from the right) are multiplied by 3. The sub segment must have a size of at least two digits. Otherwise a reading error handling is launched.

51 - Parity check even

This function is a read verify for binary data. The segment has a minimum size of two digits which must be either ‘1’ or ‘0’. Other digits or characters lead to a handling as reading errors. The number of ‘1’s must be even within the sub segment. Otherwise a reading error handling is launched.

52 - Parity check odd

This function is a read verify for binary data. The segment has a minimum size of two digits which must be either ‘1’ or ‘0’. Other digits or characters lead to a handling as reading errors. The number of ‘1’s must be odd within the sub segment. Otherwise a reading error handling is launched.

53 - DGR Final subset

This function is used to collate a variable number of subsets per set in the chain of the inserter. The segment has a size of one digit which is either ‘1’ or ‘0’. Other digits or characters lead to a handling as reading errors. While it is ‘0’ the inserter continues waiting for further subsets. The final subset has this value set to ‘1’ and the inserter turns the chain for the next set. This function works with inverted logic referred to function 42.

54 - Verify job ID

This function matches the Max. 10 character System Job ID , look at Img. 3, to the selected sub segment. If not equal, the set is marked faulty.

55 - Selection for slave channel1

Allows to operate a slave channel in a dual channel system selectively by reading. The segment has a size of one digit which is either ‘1’ or ‘0’. Other digits or characters lead to a handling as reading errors. A ‘1’ selects the slave channel.

56 - Size of set Y, collating sheet X of Y

This function allowes to process the size of a set of pages together with function 57. The information may be decimal or binary.

57 - Sheet No. X, collating sheet X (of Y)

This function collates sets of a variable number of pages. The set starts with page 1. The page numbers must be in an ascending sequence. If used together with function 56 the total number of sheets in the set is observed for additional secu rity.

58 – End of subset

This function is used to collate a variable number of subsets per set in the chain of the inserter. The segment has a size of one digit which is either ‘1’ or ‘0’. Other digits or characters lead to a handling as reading errors. A ‘1’ terminates a subset. This allows for example separate folding of the letter, the corresponding account statements e.t.c.

59 – Match code on last page

This function is used to assign a match code to a set of pages, where the match code is printed on the last page of the set only.

60 - Sheet No. X, collating sheet X of Y descending

This function collates sets of a variable number of pages. The set starts with page X=Y. The page numbers must be in an descending sequence. This function can only be used together with function 56. The total number of sheets in the set is observed for security.

61 - Modulo 10 Checksum weight 3-1-3-1 right -> left

This function performs a read verify. It checks, if the sum of all single digits is a whole multiple of ten. Before the sum is calculated, all digits at odd positions (count from the left) are multiplied by 3. The sub segment must have a size of at least two digits. Otherwise a reading error handling is launched.

Systems which communicate using a serial RS232 interface offer the following functions:

7 to 23 and 31 – Mueller Fct X

Where X is a power of two between 16 and 4194304, without 65536. Each X can be assigned to a mail inserter function like selective eject, selective feeder control or similar.

If communication between the IBC and the inserter takes place using Ethernet, the following functions are available:

3 - Bundle change

The sub segment of this function has a size of one digit. The value can be ‘0’ or ‘1’. A ‘1’ signals a bundle change and increments an internal bundle number. This number has a range between “0000” and “9999” and is later handled similar to reading results of function 4: ZIP code/Bundle #.

4 - ZIP code / Bundle

The sub segment of this function has a maximum size of ten digits. It is used to bundle equal ZIP code. Envelopes with equal ZIP codes will be bundled together by special add on equipment.

7 to 30 – Selection Feeder X

Where X is a feeder number between 1 and 24. Each character of the selected sub segment contains the number of demanded supplements from the corresponding feeder. If the size of the sub segment is larger than one character, and if it is assigned to function “7 – Selection Feeder 1”, then the second character is assigned to function “8 – Selection Feeder 2”, the third is assigned to “9 – Selection Feeder 3” etc. If the assignment exceeds function “30 – Selection Feeder 24” a handling as reading error is launched. Valid data elements in the sub segment may be the digits ‚0’ to ‚9’ and the characters ‚A’ to ‚Z’ or ‚a’ to ‚z’ where ‚A’ and ‚a’ stand for 10, ‚B’ and ‚b’ stand for 11 etc.

31 – Eject gate #

This sub segment may contain a single digit between 0 and 9 or a four digit binary number. Values > 0 select an eject gate to eject the envelope. 0 stands for the conveyer as envelope destination.

32 – Don't seal

This function is used to leave selected envelopes unsealed. The segment has a size of one digit and contains either a ‘0’ for sealed or a ‘1’ for unsealed envelopes. Other digits or characters lead to a handling as reading errors.

33 – Postal meter #

This sub segment may contain a single digit between 0 and 9 or a four digit binary number. Values > 0 select a postal meter to frank the envelope. 0 stands for non franked envelopes.

34 – Tacker

This function is used to tack the sheets of a set. The segment has a size of one digit and contains either a ‘0’ for not tacked or a ‘1’ for tacked sets. Other digits or characters lead to a handling as reading errors. This function is not yet ready to be used.

36 – Change tractor

This function is used to change the tractor by reading. The segment has a size of one digit and contains either a ‘0’ for no change or a ‘1’ for a change demand. Other digits or characters lead to a handling as reading errors. This function is not yet ready to be used.

37 – Selection marker 1
38 - Selection marker 2

This function allows to control up to two markers selectively by reading. The segment has a size of up to two digits and contains either a ‘0’ for not marked or a ‘1’ for marked envelopes. Other digits or characters lead to a handling as reading errors. If the segment is two digits long, while the sub segment is assigned to function “37 – Selection marker 1”, the second digit is used to select marker 2.

43 – Start of job

This function is used to separate jobs on the mail inserter, controlled by a start of job flag which is ‘1’ on the first set of the job only. Sequential sets of the job have the flag set to ‘0’. Other digits or characters lead to a handling as reading errors. This information is processed customer specific on the inserter.

44 – End of job

This function is used to separate jobs on the mail inserter, controlled by an end of job flag which is ‘1’ on the last set of the job only. Leading sets of the job have the flag set to ‘0’. Other digits or characters lead to a handling as reading errors. This information is processed customer specific on the inserter.

45 - Stop machine

This function is used to stop the machine by a reading flag which is ‘1’ if the machine should stop and ‘0’ if it should continue running. Other digits or characters lead to a handling as reading errors.

46 - Empty machine

This function is used to empty the machine by reading a flag which is ‘1’ if the machine should be emptied and ‘0’ if it should continue running normal. Other digits or characters lead to a handling as reading errors.

3. TESTING AND DEBUGGING

3.1 Extended Display of Processed Data

The appearance of the output window of the system control can be setup. The section [Settings] of the INI- file contains two entries:

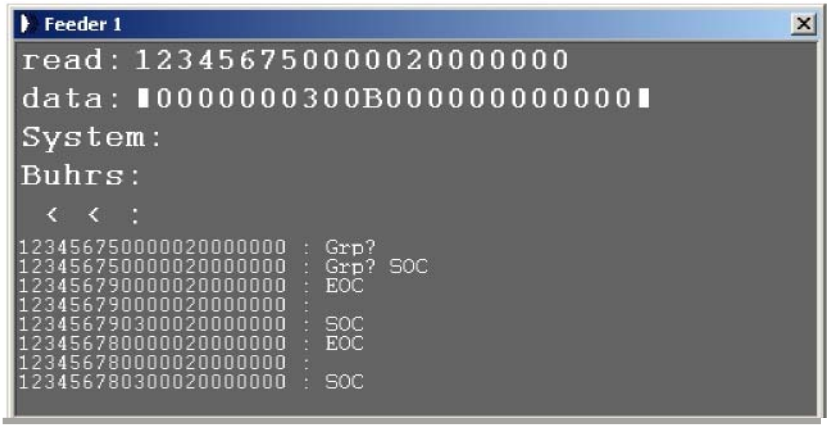
```
[Settings]
                                                    ... ...
LargeSys temWindow=TRUE
Detai ledSys temWindow=TRUE
...
```

If `LargeSystemWindow=TRUE` a double size font is used. If `DetailedSystemWindow=FALSE` the output window shows only the first two lines, on `DetailedSystemWindow=TRUE` it shows five lines of information.



Img. 10

The first line shows the read data or, if used the looked up data from database. The second line shows information about the current set. A set counter is shown and the begin of set flag „B1“ on the first page of a set or a „B0“ on sequential sheets. The third and the fourth line show information about the product in the handover unit of the system channel. The fifth line shows information about the product, which left the handover unit last. By moving and enlarging the window using the mouse, a history of the recently processed pages will be displayed, with processing information:



Img. 11

The first additional line shows the data related to the last page read. First the reading result is shown on the left followed by a ":" and flags and error flags. Error flags contain a '?' and inform about occurred errors. Flags give information about special control conditions.

Currently the following flags are used:

SOC **S**tart **o**f **C**ollation. The first page of a set has been read. EOC **E**nd **o**f **C**ollation. The last page of a set has been read.

3.2 Description of Error Flags

???	Reading errors or logically wrong data. It can be displayed on too short, too long, no data for a function
DB?	Error matching the Job -ID with the name of the database
SEQ?	Sequence error: 3 - Sequence check sheets 4 – Sequence check in set
MATCH? ID?	Matching error in the set: 2 – Match code Wrong match code: 1 – ID for collation 2 – Matchcode
BND? ZIP?	Wrong data for bundling: 3 – Bundle change Wrong data for ZIP sorting: 3 – Bundle change 4 – ZIP code / Bundle #
BIN?	Non binary data in a binary segment
DEC?	Non decimal data in a decimal segment
SEL?	Bad data for selection: Functions 7 - 30
Eject?	Bad data for eject gate selection: 31 – Eject gate #
Seal?	Bad data for selective not seal: 32 – Don't seal Bad
Meter?	data for selective franking: 33 – Postal meter#
Tacker?	Bad data for selective tacking: 34 – Tacker
IntX?	Bad data for internal eject: 35 – Internal eject
TrackX?	Bad data for tractor change: 36 – Change tractor
Marker?	Bad data for markers: 37 – Selection marker 1 38 – Selection marker 2
Grp?	Bad data for collation control: 39 - Size of set 40 - End of set 41 - Start of set
SubSet?	Bad data for subset collation in the chain: 42 – Subset follows 53 – DGR Final subset
Job?	Bad data for start/end of job: 43 – Start of job 44 – End of job
Stop?	Bad data for machine stop control: 45 – Stop machine Bad
Empty?	data for machine empty control: 46 – Empty machine Error
OneBit?	read verify: 47 - Read verify: One bit set
2Gate?	Error read verify: 48 - Read verify: Two gate marks
Mod10?	Error read verify: 49 - Modulo 10 Checksum
31Mod10?	Error read verify: 50, 61 - Modulo 10 Checksum weight 3-1-3-1...

Parity?	Error read verify parity check:	51 – Parity check even 52 – Parity check odd
JobID?	Bad data for Job-ID:	54 – Verify job ID
SelSlave?	Bad data for selective lave channel control:	55 – Selection for slave channel1
Function?	Unknown function: This error is shown, if a function is used in the program, which is not defined in the currently used software version. For example if a new program with new functions is loaded by an older IBC- software.	
	Bad binary data	
Bin Data?	Error converting decimal to binary	
DecBin?	Error converting octal to binary	
OctBin?	Error converting 5-Bit ASCII to binary	
5BitBin?	Error converting 6-Bit ASCII to binary	
6BitBin?	Error converting hexadecimal to binary	
Hex?	Error converting index to binary	
Index?	Bad data in a decimal segment	
Decimal?		