MAX interpreter

Programming manual

English Version / November.-99

Chapter 1 : General description

Interpreter of the MAX commands

The launching of the multilevel reading head allows to the user of AXIOME readers new possibilities for the management of the read marks. Indeed, the multilevel reading head (TMN) interprets each mark according to 15 levels of gray, the value of reading marks is going from 0 to E. In order to have this advantage without wasting any time during the transmission of data, AXIOME has launched the new interpreter of the MAX command.

In the case of the use of a level reading head (TAN), the value of the reading marks will be 0 or E.

Working principle (see **Figure 1**)

Once received a "RD" MAX reading command, the reader introduces the document and reads the whole amount of marks and external data (barcodes for example) on it. The value of each mark is in this way stored provisionally in the reader, until the ejection of the document.

Through the MAX "S1()" command, the application program calls the data of a rectangular zone from the upper side of the document. This command can be repeated for each zone from which the data must be captured. In this way, only the useful data will be transmitted.

The MAX "H1 or "H2" commands allow to eject the document in one of the stackers.

The MAX configuration file

The Max configuration file which can be compare to a *.ini file, allows to specify the action of each MAX command. This configuration file can be edited with any type of text editor.

The application program

The application created by the user allows to define and to transmit only the zones which are useful to the decoding. In this way, the transmission of data will be highly impressive.

The information are so managed by the computer which offers us its calculation power. Through the application program, the user can, in this way, make the difference between a mark badly erased (a gray mark) and one which is totally blackened (reader with TMN only).

The writing of the application which processes the received information from the reader will be affected by the programming language, for example Visual Basic (see Chapter 4 :).

Example : Organizational chart of an application program

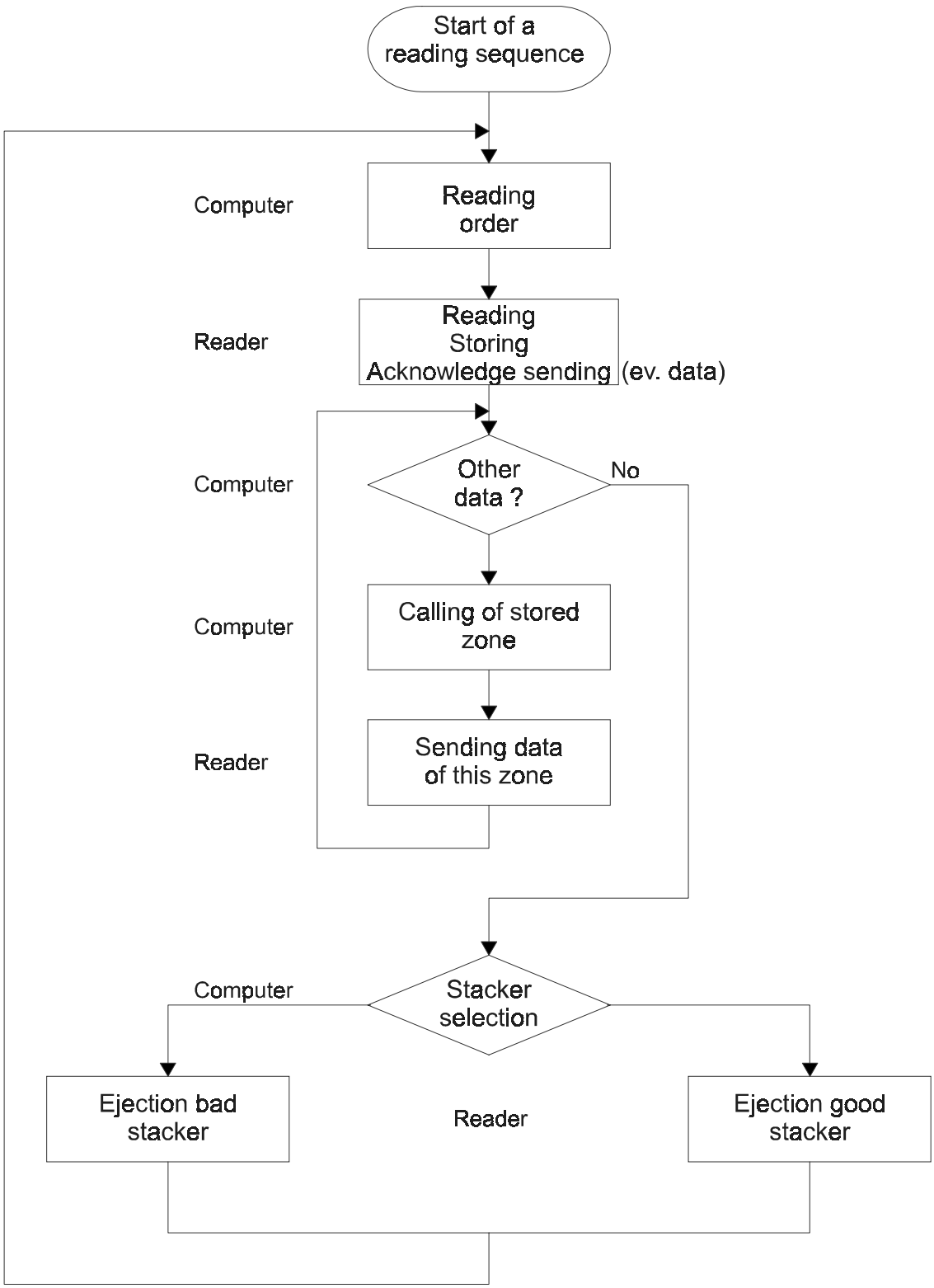


Figure 1

Chapter 2 : Commands

The commands are check orders for all the functions of the reader.

Command	Description
RD	Command for the reading of the next document
H1	Command for the ejection of the document in stacker 1 (Preferred stacker)
H2	Command for the ejection of the document in stacker 2 (Secondary stacker)
SL(K1)	Command to simulate the confirmation key
SL(DWN)	Command to move the lift down (if present)
VE	Command asking for the name and version of the configuration file
MD(CONFIG) *	Command to set the reader ready to receive the configuration file.
PR(....) *	Command for printing a text on the document (up to 200 characters)
DI(....) *	Command for displaying a text on the reader (up to 16 characters)
AL(…) *	Command for emitting an acoustic signal on the reader (alarm).
XD *	Command asking for external data (Barcode , OCR...)
S1(....) *	Command asking for the data of side 1 (upper side)
S2(....) *	Command asking for the data of side 2 (lower side)
B1(....) *	Command asking for the data of side 1 (upper side),binary mode (compressed)
B2(....) *	Command asking for the data of side 2 (lower side),binary mode (compressed)
C1 * and C2 *	Command asking for the amount of clocks read on sides 1 and 2
CN *	Command asking for the number of form read.
ST *	Command asking for the status of the reader

*** Commands which can be used as functions in the MAX configuration file**

RD

RD is the (ReaD) command which instructs the reader to call the next available document.
The RD string sent from a PC to the serial interface of the reader will set off the reading process and a reply from the reader on this same interface, this will depend on :

- **If the reading process succeeds** : the configuration parameter READACK, by default READACK=C1<0D>
  For example for a document with 12 lines (clock marks), the answer will be :
  012 followed by **CR**

- **If the reading process fails** : the configuration parameter SYSERROR, by default SYSERROR="E"$E<0D>
  For example, no document into the read (reader empty), the answer will be :
  E006 followed by **CR**

Also see READACK and SYSERROR on page 18

Note:

If the RD command is repeated, the same answer will be sent by the reader as so long the form was not ejected.

H1

The command H1 (Hopper1) ejects the document in stacker 1 (Preferred stack).
There is no reply from the reader

H2

The command H2 (Hopper2) ejects the document in stacker 2 (secondary stacker).
There is no reply from the reader

SL(K1)

The command SL(K1) (Select Key 1) simulate the pressure of the confirmation key. This command modifies the reader status (see page 16).
There is no reply from the reader

Note:

The axm915 does not have a confirmation key to validate an error on the reader, so it is necessary to use the command SL(K1) to validate.

SL(DWN)

The command SL(DWN) (Select Lift DoWN) permits moving the insertion lift down, when it contains documents. Can be used for example, in case of sequential reading.
There is no reply from the reader

Note:

Must be used only on full automatic reader containing a lift. There is no effect on the other reader.

VE

The command VE (VErsion) asks for the version number of the MAX program as well as for the name of the configuration file loaded into the reader. The format of the reply is 20 ASCII characters, always followed by <CR>

For example the answer will be: **MAX 120 <ProgName>** followed by **CR**

MD(CONFIG) *

The function MD(CONFIG) informs the reader that a configuration file will follow. This file must be sent with the same communication parameters as those in use. It's only after the file is accepted and the OK message is transmitted that the new baud rate (if different) will be validated.
See the configuration file on page 18.

Remark:

To simplify the sending of configuration file to the reader, it's also possible to include this command into the configuration file (See Example of MAX configuration file on page25)

PR(....) *

PR (for Print) is the function which allows printing a text on a document (If the reader is equipped with a printer). The number and type of printable characters depends on the reader being used (refer to the user manual).

Used as a command

It will inform the reader about the text to be printed. There is no reply from the reader and the printing will only be done after receipt of an ejection command.

For example, the **PR (My text)** command prints **My text** on the document.

Used as a function in a (READACK) macro:

For example, to systematically print the text HELLO as soon as the RD command is successful, the **READACK=PR(HELLO)** parameter would have to be defined. Therefor if the document is read (after a RD command), the text will be printed and no string will be sent to the PC.

In order to still have an answer on the **RD** command, we would have to define READACK as follows :

READACK=PR(HELLO) "OK"<0D>

In this case, the text **HELLO** is printed and the **OK** string followed by **CR** is transmitted to the PC.

DI(....) *

DI (for DIsplay) is the function which allows to display a text on the reader's display. The number and type of displayed characters is depends on the reader being used (refer to the user manual).

Used as a command

It will inform the reader about the text to be displayed. There is no reply from the reader and the execution is immediate.

For example the command **DI(DECODING)** displays **DECODING** on the display of the reader (Be careful about the type of characters accepted by the reader).

Used as a function in a (READACK) macro:

These functions can be used to display, for example, a message which warns the user that the reader is waiting for a Hn decision from the PC.

READACK=DI(SELECT?) C1<0D>

AL(…) *

AL (for ALarm) is the function which allows the reader to emit a beep sound.

Used as a command

It will inform the reader about the sound to emit. There is no reply from the reader and the execution is immediate.

For example the command **AL(2)** emits a brief sound.

Used as a function in a (READACK) macro:

These functions can be used to draw the operator's attention in case of an application which requests an intervention for each document.

READACK=AL(1) "OK"<0D>

List of alarms and their correspondences

AL(0) = no sound
AL(1) = very short sound
AL(2) = short sound
AL(3) = middle range sound
AL(5) = alarm
AL(6) = modulated sound

XD *

XD represents the external data (eXternal Data). The external data is all the data which does not come from the OMR reading heads, for example Barcode, OCR, etc...

Used as a command

It will set off a reply from the reader on this same interface and which will depend on :

- The configuration parameter DATAACK, by default DATAACK="#"$D<0D>
- The configuration parameter EXDATAMODE, by default EXDATAMODE=N,",$
- External data effectively read which will replace the parameter $D of the DATAACK.

For example with a Barcode symbolizing the value 123456 and the configuration parameters by default, the answer will be:

#"$123456 followed by CR(0DH)

Used as a function in a (DATAACK or READACK) macro:

The external data effectively read as well as the separators defined in the EXDATAMODE parameter will be sent back to the location of the XD function in the construction of the string which will be sent by the reader :

For example, suppose that it is useful to know the content of the Barcode as soon as the RD command is executed, we would have to define the READACK=XD<0D> parameter. In this way, if the document bearing the same Barcode as above is read (after a RD command), the following string will be sent to the reader : **"$123456** followed by **CR**

S1(....) *

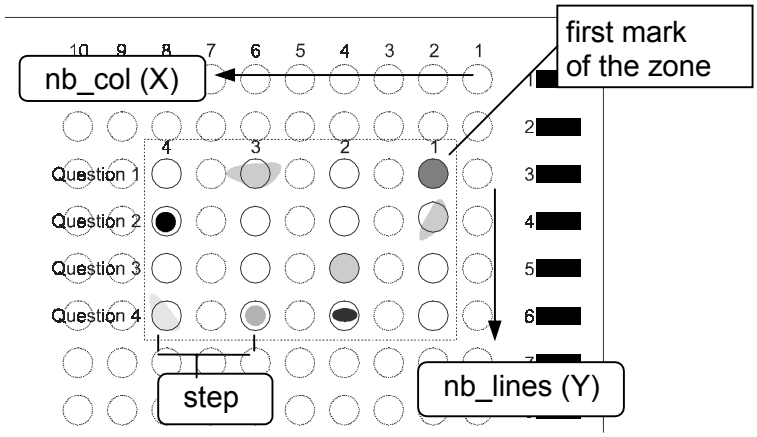
Command asking for the data read on side 1 (Side1) of the document. The S1 command must be followed by a definition of the reading zone. The definition of the horizontal zone starts from the value of the **col**umn of the mark the furthest to the right, followed by the value of the **n**um**b**er of transmitted **col**umns (don't count the columns left out if **step** >1). The definition of the vertical zone starts from the value of the **line** where the highest mark is, followed by the value of the **n**um**b**er of lines.

Each requested zone must be defined. If several zones are requested, you just have to send as many times the command including the definition of the corresponding zones. The number of requested characters corresponds to the **nb_col** times the **nb_lines.**

Used as a command :

Syntax of the command : **S1(col,nb_col[/step],line,nb_lines)**

Example of zone definition:



The zone definition command in the above is : **S1(2,4/2,3,4)**

The answer of the TMN reader with the "DEFAULT" macro (DATAACK="#"$D<0D>) is in the example :#**B050300D07000C83** followed by **CR**

The Size of the internal variable $D correspond to the number of characters **nb_col*nb_of_lines**

Note:

- If the reading head type is a TAN, the value could be only: 0 = not marked
  E = marked
- If the parameter nb_col is 0, the reader tacks the maximum number of column allowed by the reading head.
- If the parameter nb_lines is 0, the reader tack the maximum of lines (from the first to last clock of the document).
- If the asking zone is out of the document, the number of characters received will be correct but will have no signification.

Used as a function in a (READACK) macro:

The read data will be sent back to the location of the **Sn(...)** function in the construction of the string which will be sent by the reader :

For example, suppose that the columns 1 to 4 of line 1 represent the type of document. To identify it as soon as the RD command has succeeded, the READACK=S1(1,4,1,1)<0D> parameter will be defined. In this way, the 4 boxes read zone is transmitted to the PC :

For example with a TMN reading head, if the box 1,1 is ticked and the 3,1 is slightly marked the answer will be: **A020** followed by **CR**

S2(....) *

Command asking for the read data of side 2 (Side2) of the document. The zones are defined the same way as for side 1 (see S1).

See theory about the reversal of the form (Annex A : Theory about the reversal of the form)

B1(....) *

Command asking for the data read on side 1 of the document in a compressed Hexadecimal format. The B1 command must be followed by a definition of the reading zone. The definition of the horizontal zone starts from the value of the **col**umn of the mark the furthest to the right, followed by the value of the **n**um**b**er of transmitted **col**umns (don't count the columns left out if **step** >1). The definition of the vertical zone starts from the value of the **line** where the highest mark is, followed by the value of the **n**um**b**er of lines.

Each requested zone must be defined. If several zones are requested, you just have to send as many times the command including the definition of the corresponding zones. The number of requested characters corresponds to the **nb_col** times the **nb_lines** divided by two**.**

Note:

The difference between the Sn(…) and Bn(…) command is only in the reader's answer format:

- For the **Sn** command, each character is an **ASCII character corresponding to one single mark**. The value of this character can varying between 0 and E for a TMN reading head, while for a TAN reading head the character will be 0 or E.
- For the **Bn** command, each character is a **Hexadecimal character corresponding to two marks**. The value of this character can varying between <00> and <EE> for a TMN reading head, while for a TAN reading head the character will be <00>, <0E>, <E0> or <EE>.

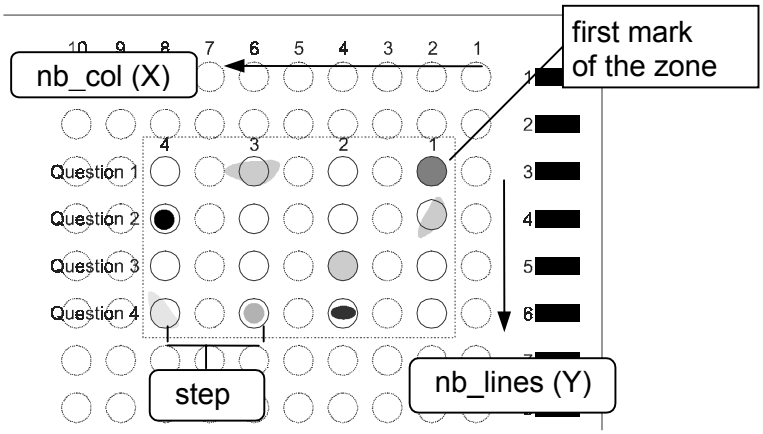
If we take as example the previous zone definition, for the command:

S1(2,4/2,3,4) the reader answer will be: B050300D07000C83
B1(2,4/2,3,4) the reader answer will be: <0B><05><03><D0><70><00><C0><38>

Used as a command :

Syntax of the command : **B1(col,nb_col[/step],line,nb_lines)**

Example of zone definition:



The zone definition command in the above is : **B1(2,4/2,3,4)**

The answer of the TMN reader with the macro DATAACK="#"$D<FF> is in the example : **#<0B><05><03><D0><70><00><C0><38>** followed by **<FF>**

The Size of the internal variable $D correspond to the number of characters **(nb_col*nb_of_lines) / 2** or **((nb_col * nb_lignes)+1) / 2** if the number of mark is odd.

Remarque:

- With the Bn(…) command, it's necessary to use a control character (see configuration file) defining the end of the string included between <F0> and <FF>, because all other combination is able to correspond to a possible mark. It's also possible, to define the end of the string, to count the number of characters, because it's corresponding to (nb_col*nd_lines)/2 or to ((nb_col*nd_lines)+1)/2 if the number of mark is odd.
- If the parameter nb_col is 0, the reader tacks the maximum number of column allowed by the reading head.
- If the parameter nb_lines is 0, the reader tack the maximum of lines (from the first to last clock of the document).
- If the asking zone is out of the document, the number of characters received will be correct but will have no signification.

Used as a function in a (READACK) macro:

The read data will be sent back to the location of the **Bn(...)** function in the construction of the string which will be sent by the reader :

For example, suppose that the columns 1 to 4 of line 1 represent the type of document. To identify it as soon as the RD command has succeeded, the READACK=S1(1,4,1,1)<FF> parameter will be defined. In this way, the 4 boxes read zone is transmitted to the PC :

For example with a TMN reading head, if the box 1,1 is ticked and the 3,1 is slightly marked the answer will be: **<0A><02>** followed by **<FF>**

B2(....) *

Command asking for the read data of side 2 of the document in a Hexadecimal format. The zones are defined the same way as for side 1 (see B1).

See theory about the reversal of the form (Annex A : Theory about the reversal of the form)

C1 * and C2 *

C1 represents the amount of lines (clocks) read on side 1 by the OMR reading head (in general the side read by the upper reading head of the reader).

C2 represents the other side of the document, taking into consideration that the reader is equipped with 2 OMR reading heads.

Used as a command

It will set off a reply of the reader on this same interface and which will depend on :

- The configuration parameter DATAACK, default DATAACK="#"$D<0D>
- The amount of read lines which will replace the $D of the DATAACK parameter (always 3 ASCII characters).

For example for a document with 65 lines then answer will be **#065** followed by **CR**.

Used as a function in a (DATAACK or READACK) macro :

The amount of read lines will be sent back to the location of the C1 function in the construction of the string which will be sent by the reader :

For example to know the amount of lines read as soon as the RD command has succeeded, the READACK=C1<0D> parameter has been defined by default, in this way, if the document with the same amount of lines as above is read (after a RD command), the following string will be sent by the reader :

065 followed by **CR.**

CN *

CN is a counter of read document, integrated into the reader. The counter start at zero after each powered up of the reader and is automatically incremented with each successfully RD command.

Used as a command

It will set off a reply of the reader on this same interface and which will depend on :

- The configuration parameter DATAACK, default DATAACK="#"$D<0D>
- The amount of read documents which will replace the $D of the DATAACK parameter (always 6 ASCII characters).

For example for the first document read then answer will be **#000001** followed by **CR**.

Used as a function in a (DATAACK or READACK) macro :

The amount of read documents sense the reader was powered up, will be sent back to the location of the CN function in the construction of the string which will be sent by the reader :

For example to know the amount of documents read as soon as the RD command has succeeded, the READACK=CN<0D> parameter can been defined, in this way, if a new document is read after the above one (after a RD command), the following string will be sent by the reader :

000002 followed by **CR.**

ST *

Command asking for the status of the reader. The status of the reader gives the state of the optical barriers and the switches.

Internal variable $D : 16 ASCII characters

Status :

1st char. :	actual state of the key	0 = not pressed 1 = pressed
2nd char. :	memorized state of the key	0 = was not pressed 1 = was pressed since the last status
3rd char. :	optical switch A	0 = no paper 1 = paper present under the optical switch
4th char. :	optical switch B	0 = no paper 1 = paper present under the optical switch
5th char. :	optical switch C	0 = no paper 1 = paper present under the optical switch
6th char. :	optical switch D	0 = no paper 1 = paper present under the optical switch
7th to 9th char. :		reserve for HEMERA
10th to 14th char. :		reserve
15th char. :	not corrected error	0 = the reader has no error 1 = the reader has an error
16th char. :	TAN reading head TMN reading head	selected sensibility "0 à 7" M

Example of configuration parameter DATAACK, by default DATAACK="#"$D<0D>, the answer will be: **#0010000000000000M** followed by **CR** (= *OSW A interrupted / TMN reading head)*

Note:

The pressing of the confirmation key is not reported into the status, if it was pressed to correct a reader error (see page 31 for a list of non-exhaustive errors).

Chapter 3 : MAX configuration file

Description

The MAX configuration file allows to personalize the behavior of the reader. It can be edited with any text editor and always starts at the first line with the @ character. In principle, no space is admitted at the beginning of the line before or after the = (equal) sign.

Note :

<function *>	is a function/command bearing a * at Chapter 2 :
<text>	is an ASCII text separated by " (inverted comma) or a control character such as <0A> for a line feed (LF). Example : "HELLO" <0D><0A>
[.....]	The parameters in brackets are optional and their order is free.
¦	Means OR

@<TITEL>	up to 8 characters
;<comment>	comments after ";"
Example: ;comments	
COM=[baud,parity,long,stop]	transmission parameters (no space is admitted).
Example: COM=19200,N,8,1	
EXDATAMODE=[P¦N,header_unit,header_label]	format of the reader's reply when a command for external data (XD) has succeed. *P* = Data parameter (depends on the used decoder type, refer to option manual). *N* = no position. The heading characters "*Header*" are not compulsory and have max. 8 characters.
Example 1: EXDATAMODE=N,",$	
Example 2: EXDATAMODE=P,,Barcode=	
AUTOCALL=[ON¦OFF]	*ON* = automatic call of the document *OFF* = waits for the read command.
Example : AUTOCALL=ON	

`AUTOEJECT=[H1¦H2¦OFF]`	*H1* = automatic ejection of the form into stacker 1. *H2* = automatic ejection of the form into stacker 2 *OFF* = waits for the ejection command.
Example : `AUTOEJECT=OFF`	
`READACK=[<text>][<function *>]`	format of the reader's reply when a reading command (RD) has succeeded.
Example : `READACK="#"C1":"S1(10,4,1,1) "/" XD<0D>`	
`DATAACK=[<text>][<function *>]$D`	format of the reader's reply when a data command (XD, S1, S2, C1, C2, ST) has succeeded. The internal variable $D will be replaced by the requested data.
Example : `DATAACK="#" $D<0D>`	
`SYSERROR=[<text>][<function*>][$E]`	format of the reader's replay when an error occurs. The internal variable $E will be replaced by the number of the error.
Example : `SYSERROR="ERREUR " $E AL(3) AL(6) <0D>`	
`READLEVEL=[<preset_read_level>]`	used to fix the sensitivity level of the reader (only for axm915, axm930, axm99x with TAN reading head). The default value is 4.
Example : `READLEVEL=5`	In the case of the axm930 and axm99x reader, the level can also be changed by pressing the reader switch. The reader using an adjustment selector does not care about this parameter, the selector primes.
`SHEETCTRL==[ON¦OFF¦LENGTH]`	*ON* = enables the length and thickness test of the form. *OFF* = disables the length and thickness test of the form. *LENGTH* = enables only the length test of the form.
Example : `SHEETCTRL=ON`	Only for reader equipped with detector.
`END`	end of the macro file.

Loading of the configuration file

The loading of the configuration file must be sent to the reader in a text file format. It stays stored until the next transfer of a new configuration file.

A simple COPY allows sending the configuration file to the reader.

Remark:

Before sending any configuration file, it's necessary to inform the reader. There is two way to do it:

1. Manual reset of the reader (refer to the reader user manual)
2. Use the **MD(CONFIG)** command (see page 9 or page 25)

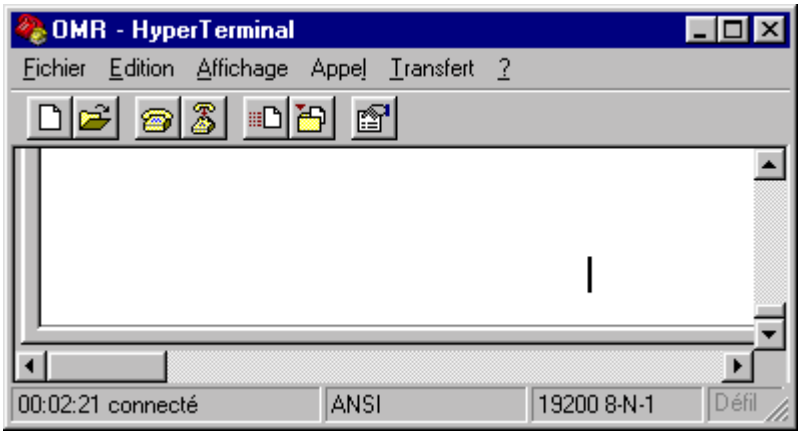
Communication :

The transfer will be effected with the parameters defined into the last configuration file down-loaded into the reader. The default configuration file (reset) communicates at 19200 baud, 8 data, No parity and 1 stop bit.

If the parameters are accepted, the reader returns **OK** followed by **CR**. In case of error of the configuration file, the number of the incorrect line and the line will be returned. The new parameters, notably the transmission speed, will take effect only after having sent the "OK" as reply.

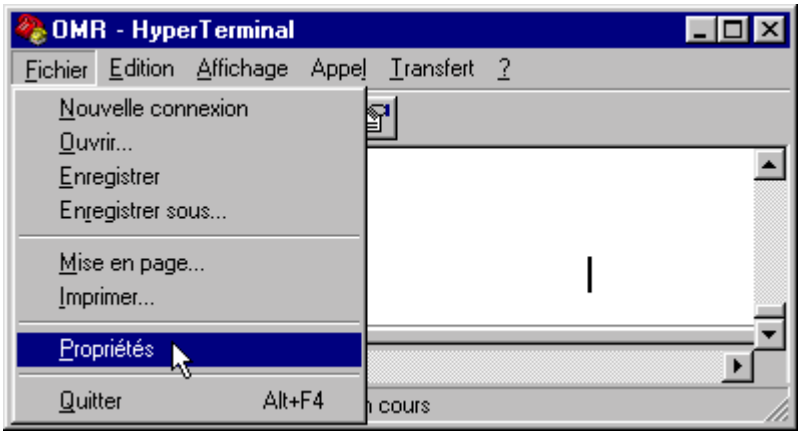
How to proceed

1. Run the Hyper Terminal communication program.



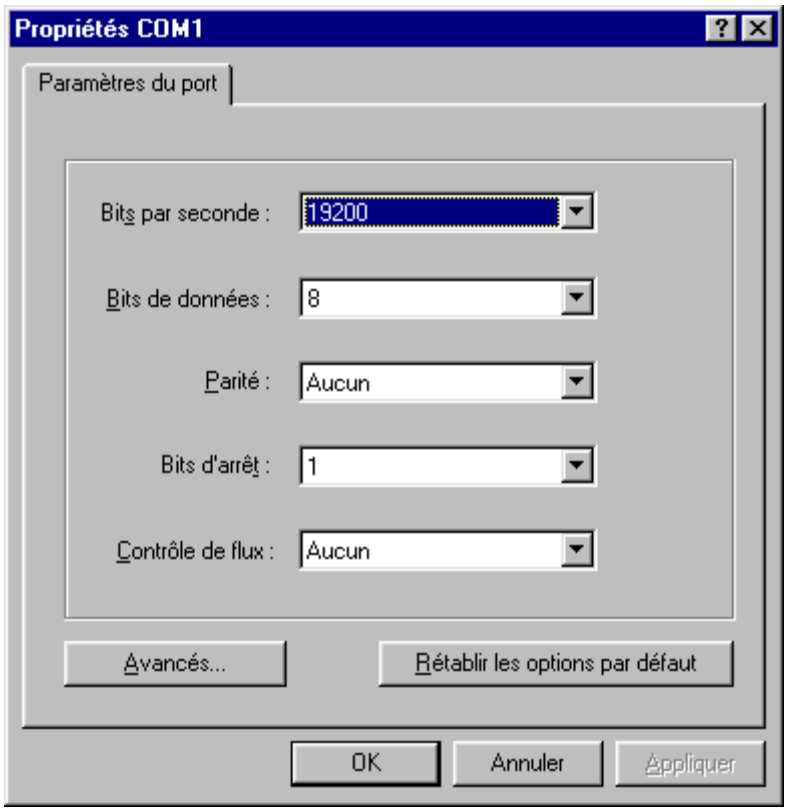
C:\Program Files\Accessoires\HyperTerminal\HYPERTRM.EXE

2. Select FILE, then PROPRIETE.

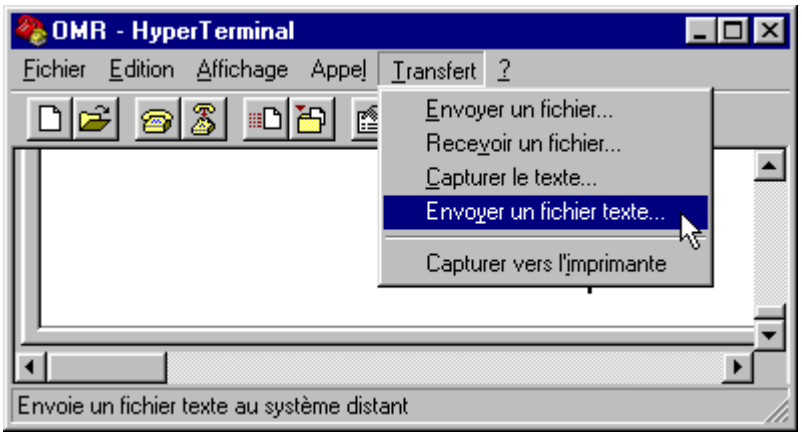


3. Select the communication port on which the reader is connected.

4. Configure the communication parameters in accordance to the old configuration file contains into the reader.

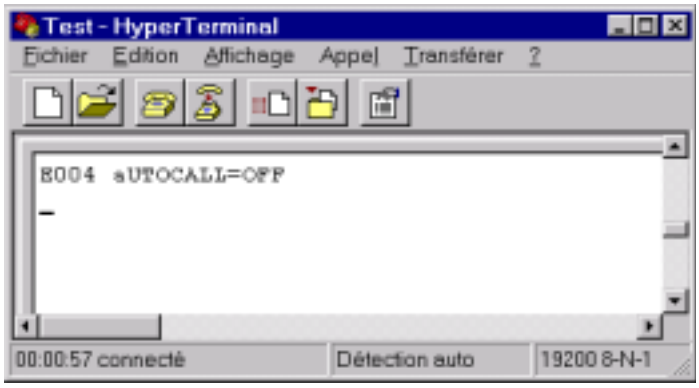


5. Select TRANSFER, then SEND A TEXT FILE.

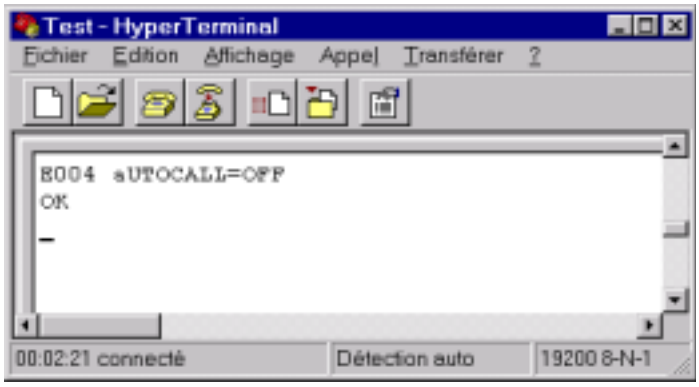


6. Select the configuration file you want to download.

If an error is contained into the configuration file, the reader will answer with **the incorrect line number and the line** followed by **CR**.



If the new configuration file is accepted, the reader answer with **OK** followed by **CR**.



Configuration values by default

The values by default are :

```
@DEFAULT
;setup
COM=19200,N,8,1
EXDATAMODE=N,",$
AUTOCALL=OFF
AUTOEJECT=OFF
SHEETCTRL=ON
;message from reader format (comment)
READACK=C1<0D>
DATAACK="#"$D<0D>
SYSERROR="E"$E<0D>
END
```

How to come back to the configuration by default

A) Reader equipped with a confirmation key and a display:
- Switch on the reader (switch ON) and press the **start button** as soon as the reader emits a sound signal. The start button must be kept pressed until the reader displays "READY TO RECEIVE".
- Repress and maintain the **start button** of the reader until it displays "DEFAULT VALID". The reader has come back to the default values.

B) Reader without display:
- Use the ERASE USER PARAMETER function contained into the maintenance menu of the reader (see user manual).

C) axm915:
- The parameters are lost every time when the reader is switched off.

Example of MAX configuration file

```
MD(CONFIG)
@MYPROG
READACK="#"C1":"S1(10,4,1,1) "/" XD<0D>
DATAACK="#" $D<0D>
SYSERROR="ERROR" $E<0D>
END
```

In this example, it will be not necessary to make a manual reset of the reader before sending the configuration file. The name of the program is MYPROG and the reader will replay on a successfully RD command with the number of clocks read follow by the character ":", a field of 5 characters representing eventually a form identification, a character "/" and the external data if present, the all followed by CR

The other parameters stay as default.

Note

The parameters are not saved on the axm915 and have to be reloaded every time by switching on the reader, when they are different from the default ones.

Chapter 4 : Example and demo

EXMAX is an application program example written in Visual Basic delivered on floppy disk. You will find the sources as well as the "EXMAX.EXE" executable file under directory VB application.

We recommend you to run this program for a better understanding about the functionality of the MAX interpreter.

EXMAX " example of application program in VB "

```
Option Explicit
Dim Timeout As Integer
Dim MsgReady As Integer
Dim AxmStr As String


Sub BtnBad_Click ()
Dim Ctrl As String

  'Send eject sheet on stacker 2 control no (response the reader)
  Ctrl = "H2"
  LblSent.Caption = Ctrl
  Comm1.Output = Ctrl
  LblRcvd.Caption = ""
  LblResult.Caption = ""
  BtnCall.Enabled = True
  BtnData.Enabled = False
End Sub


Sub BtnCall_Click ()
Dim Ctrl As String
Dim Dummy As Integer

  'Send call sheet control and prepare to receive the response
  Ctrl = "RD"
  LblSent.Caption = Ctrl
  MsgReady = False
  AxmStr = ""
  Comm1.Output = Ctrl
  LblRcvd.Caption = ""
  LblResult.Caption = ""
  Timeout = False
  Timer1.Enabled = True
  BtnCall.Enabled = False
  'wait for reader response
  While (Not MsgReady) And (Not Timeout)
    Dummy = DoEvents()
  Wend
```

```
  'test for timeout
  If Timeout Then
    LblResult.Caption = "TIMEOUT"
    BtnCall.Enabled = True
    Exit Sub
  End If
'process received message
  LblRcvd.Caption = AxmStr
  If Left$(AxmStr, 1) = "E" Then           'error message
    If Mid$(AxmStr, 2, 3) = "006" Then
      LblResult.Caption = "Reader EMPTY"
    Else
      LblResult.Caption = "Reader JAM"
      'more accurate message process must be put here...
    End If
    BtnCall.Enabled = True
  End If
  If Val(Left$(AxmStr, 3)) > 0 Then        'read sheet succesful
    LblResult.Caption = "Sheet with " + Left$(AxmStr, 3) + " clocks read OK
"
    BtnGood.Enabled = True
    Btnbad.Enabled = True
    BtnData.Enabled = True
  End If
End Sub


Sub BtnData_Click ()
Dim i As Integer
Dim Ylen, Xlen, Gap As Integer
Dim Ctrl As String
Dim OmrData As String
Dim Dummy As Integer

  'ask data from the reader

  Ctrl = "S1(" + EdtXpos.Text + "," + EdtXlen.Text + "/" + EdtStep.Text +
"," + EdtYpos.Text + "," + EdtYlen.Text + ")"
  LblSent.Caption = Ctrl
  MsgReady = False
  AxmStr = ""
  Comm1.Output = Ctrl
  LblRcvd.Caption = ""
  LblResult.Caption = ""
  Timeout = False
  Timer1.Enabled = True
  BtnData.Enabled = False
  'wait for response from the reader
  While (Not MsgReady) And (Not Timeout)
    Dummy = DoEvents()
  Wend
'test for timeout (communication line broken)
  If Timeout Then
    LblResult.Caption = "TIMEOUT"
    BtnCall.Enabled = True
    Exit Sub
```

```
  End If
  'process received data
  Gap = Val(EdtGap.Text)
  Ylen = Val(EdtYlen)
  Xlen = Val(EdtXlen)
  LblRcvd.Caption = AxmStr
  If Left$(AxmStr, 1) = "#" Then                          'DATAACK message
    For i = 1 To Ylen                                     'for each group
      OmrData = Mid$(AxmStr, 2 + ((i - 1) * Xlen), Xlen)  'extract the
group data
      LblResult.Caption = LblResult.Caption + Str$(ExtractBest(OmrData,
Gap))
      If i < Ylen Then LblResult.Caption = LblResult.Caption + ", " 'add
separator
    Next i
    BtnData.Enabled = True
  Else
    LblResult.Caption = "INVALIDE DATA"
  End If
End Sub


Sub BtnGood_Click ()
Dim Ctrl As String

  'Send eject sheet on stacker 1 control no (response the reader)
  Ctrl = "H1"
  LblSent.Caption = Ctrl
  Comm1.Output = Ctrl
  LblRcvd.Caption = ""
  LblResult.Caption = ""
  BtnCall.Enabled = True
  BtnData.Enabled = False
End Sub


Sub Comm1_OnComm ()
Dim c As String
  c = Comm1.Input
  If c = Chr$(13) Then
    MsgReady = True
    LblRcvd.Caption = AxmStr                  'display it
  Else
    AxmStr = AxmStr + c
  End If
End Sub
```

Function ExtractBest (Group As String, MinGap As Integer) As Integer

```
Static Ival(48) As Integer
Dim i As Integer
Dim Cell As Integer
Dim MaxVal As Integer
Dim MaxPos As Integer
Dim NextVal As Integer

  'convert ASCII representation to integer table with values from 0 to 14
  MaxVal = 0
  MaxPos = 1
  For i = 1 To Len(Group)
    Cell = Asc(Mid$(Group, i, 1))
    If Cell > Asc("9") Then
      Ival(i) = Cell - 55           'it was a letter A..E
    Else
      Ival(i) = Cell - 48           'it was a number 0..9
    End If
    If Ival(i) > MaxVal Then        'search the best value
      MaxVal = Ival(i)              'store it
      MaxPos = i                    'save it position
    End If
  Next i

  'search for the next best mark
  NextVal = 0
  Ival(MaxPos) = 0                  'erase the best mark already found
  For i = 1 To Len(Group)
    If Ival(i) > NextVal Then
      NextVal = Ival(i)
    End If
  Next i

'test if gap is satisfied
  If (MaxVal - NextVal) >= MinGap Then
    ExtractBest = MaxPos     'return the position of the best value
  Else
    if (MaxVal >= MinGap) And (NextVal >= MinGap) Then
      ExtractBest = -1       'more then 1 mark >= MinGap
    Else
      ExtractBest = 0        'MinGap not found
    End If
  End If
End Function
```

```
Sub Form_Load ()
  If Command$ = "/2" Then
    Comm1.CommPort = 2
  End If
  Comm1.PortOpen = True
  LblSent.Caption = ""
  LblRcvd.Caption = ""
  LblResult.Caption = ""
  EdtXpos.Text = 2
  EdtXlen.Text = 4
  EdtStep.Text = 2
  EdtYpos.Text = 3
  EdtYlen.Text = 4
  EdtGap.Text = 3
End Sub


Sub Timer1_Timer ()
  Timer1.Enabled = False
  Timeout = True
End Sub
```

Annex A : Theory about the reversal of the form

Coordinates of side 2 in relation to side 1 (OMR with double sided reading head)

In keeping the programming as simple as possible, the buffer of side 2 is organized as to be identical as the buffer of side1, so it's very easy to read a form in this way.

Note also the clock position on the right side as it is usual in Europe.

Simple sided form read on a reader equipped with 2 heads

A simple sided form can be read indiscriminately by one or the other reading head. The fields are accessible to the same coordinate and given in parameters to the Sn(…) or Bn(…) command. Only the n (1 or 2) must be defined into the program.

For example, for a field located at column 12 to column 14 and line 6 to line 10, the command to read on side 1 will be : **S1(12,3,6,5)** or **B1(12,3,6,5)**.

And now if feeding of the document is reversed, to be read by the other head, the command to read the same field will be : **S2(12,3,6,5)** or **B2(12,3,6,5)**.

Double sided form read on a reader equipped with 2 heads

The Figure 2 gives a graphical representation (half transparent) of a double sided form and the way how to return it.

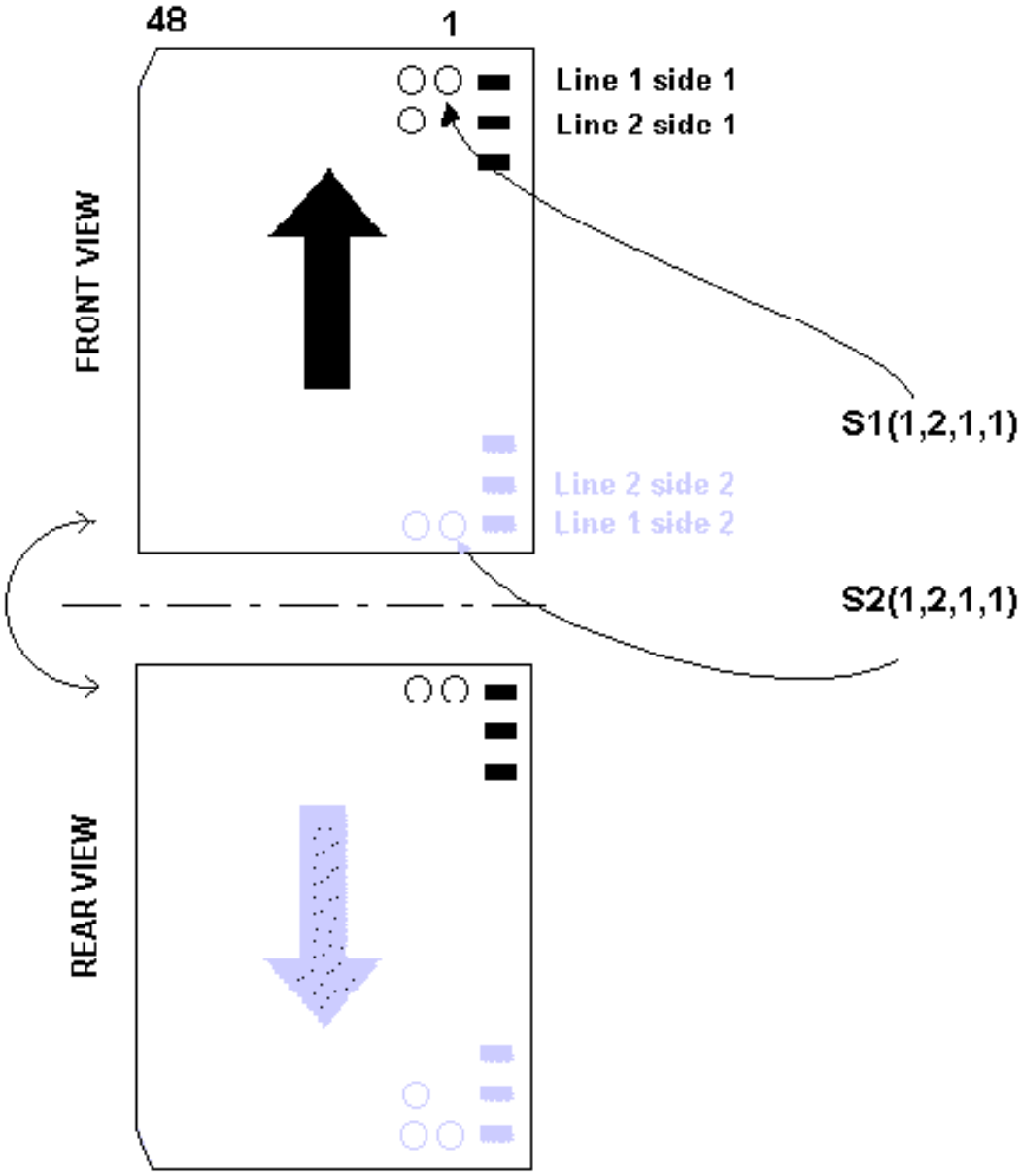


Figure 2

Annex B : Error messages

No	Description
000	DATA BUFFER EMPTY
001	BAD FEEDING
002	JAM BEFORE HEAD
003	JAM UNDER HEAD
004	JAM AFTER HEAD
005	JAM IN SORTING
006	NO SHEET ON LIFT
007	BAD TRAY FULL
008	GOOD TRAY FULL
009	SHEET TOO SHORT
010	SHEET TOO THIN
011	SHEET TOO THICK
012	SHEET TOO LONG
013	INCORRECT SHEET
014	-
015	-
016	-
017	-
018	-
019	NO SHEET TO SORT
020	PATH NOT FREE
021	HEAD INIT ERROR
022	NO DECODER
023	LIFT ERROR
024	FAILED Ch x
025	GOOD TRAY ERROR
026	SECURITY STOP
027	NO SHEET IN GOOD
028	NO SHEET IN BAD

Annex C : Hints and short-cuts for increased data processing

The transmitting:

It is preferable to use the maximum transmission speed of the reader, either 38'400 baud for the axm915, axm926, axm930, axm952/985, axm99x or 115'200 baud for the axm950. The amount of data can be important, if you take for example a complete A4 document.

Defining the zones:

It is sometimes more judicious to import a large data zone, even if some of the data is useless, rather than calling 3 smaller zones one after the other. The reaction time and the treatment of the information corresponds to the transmission of 10 ASCII characters at the baud rate of 38'400 bps. Therefore, if between two zones there is less than 10 empty cells, it is recommended to combine them into only one definition Sn(…) or Bn(…).

When using the command Bn(…) rather than the command Sn(…). And again, this depends on the amount of characters that have to be transmitted. Tests have shown that it would be better to use the command Bn(…), if the defined zone is larger than 80 cells. Below this, the gain in transmission time stays nominal in comparison to the reaction and the treatment of the of the information of the reader. On the opposite side, the gain in time is practically 50%.

The ST command:

The ST command should be used cautiously, as it can considerably show down the speed of the reader. Depending on the type of reader, the treatment time for the microprocessor to collect the information necessary for the status can be more or less long.

Treatment of the information in the PC:

The speed of the treatment of the information in the PC is directly tied to the PC. But we can say that with the actual PCs (Pentium II / III), the treatment of the information is insignificant in comparison with the speed of the OMR reader and the transmission time of the data and will very little influence the speed of the reader.

The barcode decoder:

The barcode decoder can significantly reduce the speed of the reader if no barcode is present on the documents. It is advised to disable by software (axm930, axm99x, axm950) or to manually disconnect the barcode decoder if it is not necessary.

Annex D : ASCII conversion table

DECIMAL	HEX	OCTAL	ASCII	ANSI	CTRL-CHAR
0	00	000	NUL		^@
1	01	001	SOH	□	^A
2	02	002	STX	□	^B
3	03	003	ETX	□	^C
4	04	004	EOT	□	^D
5	05	005	ENQ	□	^E
6	06	006	ACK	□	^F
7	07	007	BEL	□	^G
8	08	010	BS	(BackSpace)	^H
9	09	011	HT	(Tab)	^I
10	0A	012	LF	(LineFeed)	^J
11	0B	013	VT	(Vtab)	^K
12	0C	014	FF		^L
13	0D	015	CR	(Return)	^M
14	0E	016	SO		^N
15	0F	017	SI	□	^O
16	10	020	DLE	□	^P
17	11	021	DC1	□	^Q
18	12	022	DC2	□	^R
19	13	023	DC3	□	^S
20	14	024	DC4	□	^T
21	15	025	NAK	□	^U
22	16	026	SYN	□	^V
23	17	027	ETB	□	^W
24	18	030	CAN	□	^X
25	19	031	EM	□	^Y
26	1A	032	SUB	□	^Z
27	1B	033	ESC		^|
28	1C	034	FS	□	^\
29	1D	035	GS	□	^]
30	1E	036	RS	-	^^
31	1F	037	US		^'
32	20	040	SPACE	SPACE	SPACE
33	21	041	!	!	
34	22	042	"	"	
35	23	043	#	#	
36	24	044	$	$	
37	25	045	%	%	
38	26	046	&	&	
39	27	047	'	'	
40	28	050	(	(	
41	29	051	)	)	
42	2A	052	*	*	
43	2B	053	+	+	
44	2C	054	,	,	
45	2D	055	-	-	
46	2E	056	.	.	
47	2F	057	/	/	
48	30	060	0	0	
49	31	061	1	1	
50	32	062	2	2	

DECIMAL	HEX	OCTAL	ASCII	ANSI	CTRL-CHAR
51	33	063	3	3	
52	34	064	4	4	
53	35	065	5	5	
54	36	066	6	6	
55	37	067	7	7	
56	38	070	8	8	
57	39	071	9	9	
58	3A	072	:	:	
59	3B	073	;	;	
60	3C	074	<	<	
61	3D	075	=	=	
62	3E	076	>	>	
63	3F	077	?	?	
64	40	100	@	@	
65	41	101	A	A	
66	42	102	B	B	
67	43	103	C	C	
68	44	104	D	D	
69	45	105	E	E	
70	46	106	F	F	
71	47	107	G	G	
72	48	110	H	H	
73	49	111	I	I	
74	4A	112	J	J	
75	4B	113	K	K	
76	4C	114	L	L	
77	4D	115	M	M	
78	4E	116	N	N	
79	4F	117	O	O	
80	50	120	P	P	
81	51	121	Q	Q	
82	52	122	R	R	
83	53	123	S	S	
84	54	124	T	T	
85	55	125	U	U	
86	56	126	V	V	
87	57	127	W	W	
88	58	130	X	X	
89	59	131	Y	Y	
90	5A	132	Z	Z	
91	5B	133	[	[	
92	5C	134	\	\	
93	5D	135	]	]	
94	5E	136	^	^	
95	5F	137			
96	60	140	`	`	
97	61	141	a	a	
98	62	142	b	b	
99	63	143	c	c	
100	64	144	d	d	

DECIMAL	HEX	OCTAL	ASCII	ANSI	CTRL-CHAR
101	65	145	e	e	
102	66	146	f	f	
103	67	147	g	g	
104	68	150	h	h	
105	69	151	i	i	
106	6A	152	j	j	
107	6B	153	k	k	
108	6C	154	l	l	
109	6D	155	m	m	
110	6E	156	n	n	
111	6F	157	o	o	
112	70	160	p	p	
113	71	161	q	q	
114	72	162	r	r	
115	73	163	s	s	
116	74	164	t	t	
117	75	165	u	u	
118	76	166	v	v	
119	77	167	w	w	
120	78	170	x	x	
121	79	171	y	y	
122	7A	172	z	z	
123	7B	173	{	{	
124	7C	174	|	|	
125	7D	175	}	}	
126	7E	176	~	~	
127	7F	177	DEL	□	
128	80	200		□	
129	81	201		□	
130	82	202		‚	
131	83	203		*ƒ*	
132	84	204		„	
133	85	205		…	
134	86	206		†	
135	87	207		‡	
136	88	210		ˆ	
137	89	211		‰	
138	8A	212		Š	
139	8B	213		‹	
140	8C	214		Œ	
141	8D	215		□	
142	8E	216		**Ž**	
143	8F	217		□	
144	90	220		□	
145	91	221		‘	
146	92	222		’	
147	93	223		“	
148	94	224		”	
149	95	225		•	
150	96	226		–	

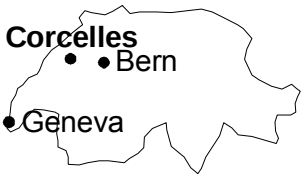
DECIMAL	HEX	OCTAL	ASCII	ANSI	CTRL-CHAR
151	97	227		—	
152	98	230		~	
153	99	231		™	
154	9A	232		š	
155	9B	233		›	
156	9C	234		œ	
157	9D	235		□	
158	9E	236		ž	
159	9F	237		Ÿ	
160	A0	240		SPACE	
161	A1	241		¡	
162	A2	242		¢	
163	A3	243		£	
164	A4	244		¤	
165	A5	245		¥	
166	A6	246		¦	
167	A7	247		§	
168	A8	250		¨	
169	A9	251		©	
170	AA	252		ª	
171	AB	253		«	
172	AC	254		¬	
173	AD	255		-	
174	AE	256		®	
175	AF	257		¯	
176	B0	260		°	
177	B1	261		±	
178	B2	262		²	
179	B3	263		³	
180	B4	264		´	
181	B5	265		µ	
182	B6	266		¶	
183	B7	267		·	
184	B8	270		¸	
185	B9	271		¹	
186	BA	272		º	
187	BB	273		»	
188	BC	274		¼	
189	BD	275		½	
190	BE	276		¾	
191	BF	277		¿	
192	C0	300		À	
193	C1	301		Á	
194	C2	302		Â	
195	C3	303		Ã	
196	C4	304		Ä	
197	C5	305		Å	
198	C6	306		Æ	
199	C7	307		Ç	
200	C8	310		È	

DECIMAL	HEX	OCTAL	ASCII	ANSI	CTRL-CHAR
201	C9	311		É	
202	CA	312		Ê	
203	CB	313		Ë	
204	CC	314		Ì	
205	CD	315		Í	
206	CE	316		Î	
207	CF	317		Ï	
208	D0	320		Ð	
209	D1	321		Ñ	
210	D2	322		Ò	
211	D3	323		Ó	
212	D4	324		Ô	
213	D5	325		Õ	
214	D6	326		Ö	
215	D7	327		×	
216	D8	330		Ø	
217	D9	331		Ù	
218	DA	332		Ú	
219	DB	333		Û	
220	DC	334		Ü	
221	DD	335		Ý	
222	DE	336		Þ	
223	DF	337		ß	
224	E0	340		à	
225	E1	341		á	
226	E2	342		â	
227	E3	343		ã	
228	E4	344		ä	
229	E5	345		å	
230	E6	346		æ	
231	E7	347		ç	
232	E8	350		è	
233	E9	351		é	
234	EA	352		ê	
235	EB	353		ë	
236	EC	354		ì	
237	ED	355		í	
238	EE	356		î	
239	EF	357		ï	
240	F0	360		ð	
241	F1	361		ñ	
242	F2	362		ò	
243	F3	363		ó	
244	F4	364		ô	
245	F5	365		õ	
246	F6	366		ö	
247	F7	367		÷	
248	F8	370		ø	
249	F9	371		ù	
250	FA	372		ú	
251	FB	373		û	
252	FC	374		ü	
253	FD	375		ý	
254	FE	376		þ	
255	FF	377		ÿ	

AXIOME Alpha SA
Rue de la Gare 5a
CH-2035 Corcelles
Switzerland

mailto:axiome@access.ch
http://www.axiome.ch/

Tel. ++41 (032) 730 46 20
Fax. ++41 (032) 731 64 93



AXIOME GmbH
Justus-v..-Liebig-Str.2
D-85435 Erding Germany

Tel. ++49 (08122) 88095-0
Fax. ++49 (08122) 88095-10

Printed in Switzerland by AXIOME SA