Table of Contents

Chapter 1: Form Parameters ................................................. 3
    Form Parameters ..................................................... 4

Chapter 2: Field Parameters .............................................. 27
    Field Parameters .................................................... 28
    DPI Reference ....................................................... 59

Chapter 3: Logic Blocks .................................................... 61
    LOGIC_BLOCKS ...................................................... 62
    IF Logic Using LOGIC_BLOCK .................................... 62
    LOGIC_BLOCK Structure .......................................... 63
    Using LOGIC_BLOCKS in an Application ...................... 64

Chapter 4: Field Persistence .............................................. 67
    Field Persistence ................................................... 68
    Persistence Saves Time .......................................... 68
    How Persistence Works .......................................... 69
FORM PARAMETERS

In this chapter we will explore the following topics:

- Form Parameters
Form Parameters

Form Parameters are modifiers which affect the entire Link File (.LNK). For example, general document settings such as paper size, type of auditing and the number of checks per page are indicated using Form Parameters. Form Parameters are located at the beginning of the Link File. They are listed as parameters or parameter abbreviations with the options listed in parentheses. The Form Parameter indicates what form to use and must be the first parameter listed, and all the others should follow.

The following Form Parameters are available for PayBase Applications:

- **AUDITPRINTER**
- **AUDITTYPE**
- **AUTOCODEHORIZONTALADJUSTMENT**
- **AUTOCODEREFERENCECNUMBERLENGTH**
- **AUTOCODEVERTICALADJUSTMENT**
- **AUTOMICRACCTNUMBERLOCATION**
- **AUTOMICRCHECKNUMBERLENGTH**
- **AUTOMICRCHEQUENUMBERLENGTH**
- **AUTOMICR HORIZONTALADJUSTMENT**
- **AUTOMICRONUSLOCATION**
- **AUTOMICRVERTICALADJUSTMENT**
- **CHECKDEPTH**
- **CHECKONTHE**
- **CHECKTYPE**
- **CHECKXADJUST**
- **CHECKYADJUST**
- **CHEQUEDEPTH**
- **CHEQUEONTHE**
- **CHEQUETYPE**
- **CHEQUEXADJUST**
- **CHEQUEYADJUST**
• CODEFONT
• CODEFONTSWITCH
• CODELAYOUT
• CURRENCY
• FIELDFILE
• FORMFIELDNAME
• FORMSPERPAGE
• FORMTYPE
• GROUPFIELDNAME
• INTERCHECKADJUSTMENT
• INTERCHEQUEADJUSTMENT
• LTRIM
• MICRFONT
• OUTTRAY
• PAPERDEPTH
• PAUSEMSG
• PAYMENTMETHOD
• PERSISTRESET
• PRINTERFIELDNAME
• REPEAT
• TRAY

AUDITPRINTER

This parameter allows addition auditing through PayBase.

Options: YES/NO
Default: No
Requirement: AUDIT field parameters
Prerequisite for: Job Summary Report Tray setting
32-bit platform
Description:
“Y” sends audit information to the Transaction History Log in addition to the PayBase audit file.
“N” sends audit information to the PayBase audit file only.
Example:
AUDITPRINTER(Y)

**AUDITTYPE**

This Form Parameter defines the extent of PayBase auditing

**Options:** BATCH/DETAIL

**Default:**
- BATCH when the Application uses a data file
- DETAIL when the Application uses manual entry (does not use a data file)

**Requirement:** AUDIT Field Parameters

**Description:**
"DETAIL" creates one audit record for each check printed.
"BATCH" creates one audit record containing a total for ALL checks printed.

**Example:**
AUDITTYPE(BATCH)
AT(DETAIL)

**AUTOCODEHORIZONTALADJUSTMENT**

This Form Parameter moves the entire CODE Line horizontally.

**Options:** n = a numeric value

**Default:** n = 0

**Requirement:**
REFERENCE(?AUTOMICR) Field Parameter
AUTOMICR field parameters

**Description:**
The numeric value assigned, n, will move the CODE Line n/300 inches horizontally. A positive value will move the CODE Line to the right; a negative value will move the CODE Line to the left. For example, AMHA(150) will move the CODE Line 150/300 or 1/2” to the right, and AMHA(-75) will move the CODE Line -75/300 or 1/4” to the left.

Example:

AUTOCODEHORIZONTALADJUSTMENT(-5)

### AUTOCODEREFERENCENUMBERLENGTH

This Form Parameter specifies the number of reference number digits to print on the CODE Line.

**Options:** any number between 3 and 14

**Requirement:**

REFERENCE(?AUTOMICR) Field Parameter

**Description:**

This parameter is used when the reference number is to be placed on the CODE Line. The assigned number specifies the maximum reference number length. Leading zeros are inserted if necessary.

Example:

AUTOCODEREFERENCENUMBERLENGTH(10)

### AUTOCODEVERTICALADJUSTMENT

This Form Parameter moves the entire CODE Line vertically.

**Options:** a numeric value

**Default:** 0

**Requirement:**

REFERENCE(?AUTOMICR) Field Parameter

**Description:**

The entered numeric value will move the CODE Line n/300 inches vertically. A positive value will move the CODE Line toward the bottom of the page, and a negative value will move the CODE Line toward the top of the page. For example, AMVA (150) will move the CODE Line 150/
300 or 1/2” down the page, and AMVA(-100) will move the CODE Line - 100/300 or 1/3” up the page.

Example:
AUTOCODEVERTICALADJUSTMENT(-150)

**AUTOMICRACCTNUMBERLOCATION**

This Form Parameter allows the position of the Account Number on the MICR Line to be specified by defining the position of the right Account Number symbol.

**Options:** n = any number between 32 and 14

**Default:** Number that places the left digit of the account number in position 31

**Requirement:**
- AUTOMICR(ACCTNUMBER) Field Parameter
- REFERENCE(?AUTOMICR) Field Parameter

**Description:**

The Account Number is the account number that is included on the MICR Line within positions 32 and 14. The length and position of the Account Number varies, it is located one space to the right of the Sort Code number, at position 31 and it always ends with a right Account Number symbol. By default, the position of the right Account Number symbol is flexible and depends on the length of the Account Number. AUTOMICRACCTNUMBERLOCATION is used when the left Account Number digit must be in a location other than position 31 by specifying the position of the right Account Number symbol, thus forcing it to remain in position.

For example, to place a 12 digit Account Number number on the MICR Line so that the left digit is in position 28, the right Account Number symbol would have to be in position 16. The AMOL parameter would be listed as AMOL(16).

If the Account Number will not be the same for every cheque, use the Field Parameter AUTOMICR(LOCATION) instead of AMOL.

**Example:**
AUTOMICRACCTNUMBERLOCATION(22)

**AUTOMICRCHECKNUMBERLENGTH**

This Form Parameter specifies the number of check number digits to print on the MICR Line.

**Options:** n = any number between 3 and 14

**Default:** n = 6
CHAPTER 1: FORM PARAMETERS

Form Parameters

Requirement:
- AUTOMICR(CHECKNUMBER) Field Parameter
- REFERENCE(??AUTOMICR) Field Parameter

Description:
This parameter is used when the check number is to be placed on the MICR Line. The assigned number specifies the maximum check number length. Leading zeros are inserted if necessary.

Example:
AUTOMICR(CHECKNUMBER)LENGTH(10)

AUTOMICR(CHEQUENUMBER)LENGTH

This Form Parameter specifies the number of cheque number digits to print on the MICR Line.

Options: n = any number between 3 and 14
Default: n = 6

Requirement:
- AUTOMICR(CHEQUENUMBER) Field Parameter
- REFERENCE(??AUTOMICR) Field Parameter

Description:
This parameter is used when the cheque number is to be placed on the MICR Line. The assigned number specifies the maximum cheque number length. Leading zeros are inserted if necessary.

Example:
AUTOMICR(CHEQUENUMBER)LENGTH(10)

AUTOMICR(HORIZONTALADJUSTMENT)

This Form Parameter moves the entire MICR Line horizontally.

Options: n = a numeric value
Default: n = 0

Requirement:
CHAPTER 1: FORM PARAMETERS

- Form Parameters

- REFERENCE(?AUTOMICR) Field Parameter
- AUTOMICR Field Parameters

**Description:**
The numeric value assigned, n, will move the MICR Line n/300 inches horizontally. A positive value will move the MICR Line to the right; a negative value will move the MICR Line to the left. For example, AMHA(150) will move the MICR Line 150/300 or 1/2" to the right, and AMHA(-75) will move the MICR Line -75/300 or 1/4" to the left.

**Example:**
AUTOMICR(HORIZONTALADJUSTMENT(-5))

AUTOMICRONUSLOCATION

This Form Parameter allows the position of the Onus number on the MICR Line to be specified by defining the position of the right Onus symbol.

**Options:** n = any number between 32 and 14
**Default:** Number that places the left digit of the onus number in position 31

**Requirement:**
- AUTOMICR(ONUS) Field Parameter
- REFERENCE(?AUTOMICR) Field Parameter

**Description:**
The Onus number is the account number that is included on the MICR Line within positions 32 and 14. The position of the right Onus symbol is flexible and depends on the length of the Onus number. AUTOMICRONUSLOCATION is used when the left Onus digit must be in a location other than position 31 by specifying the position of the right Onus symbol, thus forcing it to remain in position.

For example, to place a 12 digit onus number on the MICR Line so that the left digit is in position 28, the right onus symbol would have to be in position 16. The AMOL parameter would be listed as AMOL(16).

If the Onus number will not be the same for every check, use the Field Parameter AUTOMICR(LOCATION) instead of AMOL.

**Example:**
AUTOMICRONUSLOCATION(22)
AUTOMICRVERTICALADJUSTMENT

This Form Parameter moves the entire MICR Line vertically.

**Options:** n = a numeric value

**Default:** n = 0

**Requirement:**
- REFERENCE(AUTOMICR) Field Parameter
- AUTOMICR Field Parameters

**Description:**
The numeric value assigned, n, will move the MICR Line n/300 inches vertically. A positive value will move the MICR Line toward the bottom of the page, and a negative value will move the MICR Line toward the top of the page. For example, AMVA (150) will move the MICR Line 150/300 or 1/2" down the page, and AMVA(-100) will move the MICR Line -100/300 or 1/3" up the page.

**Example:**
AUTOMICRVERTICALADJUSTMENT(15)

CHECKDEPTH

This Form Parameter determines the position of the MICR Line on a form that is designed with the check above the stub on the page. It works by specifying the number of inches from the top of the page to the bottom of the first check.

**Options:** n = a numeric value

**Default:** n = 3.5

**Requirement:**
- A Form Design with the check above the stub.
- CHECKON(TOP) Form Parameter
- REFERENCE(AUTOMICR) Field Parameter

**Description:**
The numeric value, n specifies the distance in inches from the top of the page to the bottom of the first check. This measurement is used by PayBase to position the MICR Line.

Use the Field Parameter PAPERDEPTH instead of CHECKDEPTH when the Designer+ form is not designed with the check placed above the stub.
CHAPTER 1: FORM PARAMETERS

Form Parameters

Example:
CHECKDEPTH(5)

CHECKONTHE
This Form Parameter specifies the relative check and stub placement for MICR Line positioning.

Options: TOP/BOTTOM
Default: BOTTOM
Requirement:
• CHECKDEPTH Form Parameter if "TOP" is specified
• PAPERDEPTH Form Parameter if "BOTTOM" is specified
• REFERENCE(?AUTOMICR) Field Parameter
• AUTOMICR Field Parameters
Description:
"TOP" will correctly place the MICR Line when the Designer+ form includes the check above the stub on the page.
"BOTTOM" will correctly place the MICR Line when the Designer+ form includes the stub above the check on the page.
Example:
CHECKONTHE(T)

CHECKTYPE
Use this Form Parameter to specify the type of check to be printed if it is a personal check as opposed to a business check.

Options: BUSINESS/PERSONAL
Default: BUSINESS
Requirement: None
Description:
CHECKTYPE is used to correctly locate and format the MICR Line, because the MICR Line is different for business and personal check types.
For business checks, specify BUSINESS. The checks are approximately 3.5" from top to bottom and 8.25" from side to side. The business check MICR
CHAPTER 1: FORM PARAMETERS

Form Parameters

Line consists of, from left to right: the check number, the intransit number, the onus number, and the check amount.

For personal checks, specify PERSONAL. The checks are smaller, approximately 2.75" x 6". The personal check MICR Line consists of, from left to right, the intransit number, onus number, check number, and check amount.

Example:
CHECKTYPE(B)

CHECKYADJUST

Use this Form Parameter to adjust the vertical placement of all data fields in an application.

Options: n = a numeric value
Default: n = 0
Requirement: None
Description:

The numeric value, n, moves all of the data fields in an application n/300 inches vertically. A positive value will move the data fields toward the bottom of the page, and a negative value will move the data fields toward the top of the page. CHECKYADJUST does not reposition the Form, only the data fields.

This parameter may be included multiple times. The first CHECKYADJUST specifies the adjustment of the top check's data fields, the second CHECKYADJUST specifies the adjustment of the second check's data fields, etc.

Example:
CHECKYADJUST(5)

CHECKXADJUST

Use this Form Parameter to adjust the horizontal placement of all data fields in an application.

Options: n = a numeric value
Default: n = 0
Requirement: None
Description:
The numeric value, n, moves all of the data fields in an application n/300 inches horizontally. A positive value will move the data fields toward the bottom of the page, and a negative value will move the data fields toward the top of the page. CHECKXADJUST does not reposition the Form, only the data fields.

This parameter may be included multiple times. The first CHECKXADJUST specifies the adjustment of the top check's data fields, the second CHECKXADJUST specifies the adjustment of the second check's data fields, etc.

**Example:**
CHECKXADJUST(5)

**CHEQUEDEPTH**

This Form Parameter determines the position of the MICR Line on a form that is designed with the cheque above the remittance advice on the page. It works by specifying the number of inches from the top of the page to the bottom of the first cheque.

**Options:** n = a numeric value

**Default:** n = 4.0

**Requirement:**
- A Form Design with the cheque above the remittance advice.
- CHEQUEONTHE(TOP) Form Parameter
- REFERENCE(?AUTOMICR) Field Parameter

**Description:**
The numeric value, n specifies the distance in inches from the top of the page to the bottom of the first cheque. This measurement is used by i-Pay to position the MICR Line.

Use the Field Parameter PAPERDEPTH instead of CHEQUEDEPTH when the Designer+ form is not designed with the cheque placed above the remittance advice.

**Example:**
CHEQUEDEPTH(4.0)

**CHEQUEONTHE**

This Form Parameter specifies the relative cheque and remittance advice placement for MICR Line positioning.
CHAPTER 1: FORM PARAMETERS

Form Parameters

Options: BOTTOM
Default: BOTTOM
Requirement:
- PAPERDEPTH Form Parameter
- REFERENCE(AUTOMICR) Field Parameter
- AUTOMICR Field Parameters
Description:
CHEQUEONTHE will correctly place the MICR Line when the Designer+ form includes the remittance advice above the cheque on the page.
Example:
CHEQUEONTHE(B)

CHEQUETYPE
Use this Form Parameter to specify the type of cheque to be printed if it is a personal cheque as opposed to a business cheque.
Options: BUSINESS/PERSONAL
Default: BUSINESS
Requirement: None
Description:
CHEQUETYPE is used to correctly locate and format the MICR Line, because the MICR Line is different for business and personal cheque types.
For business cheques, specify BUSINESS. The cheques are approximately 3.5" from top to bottom and 8.25" from side to side. The business cheque MICR Line consists of, from left to right: the cheque number, the sort code number, the Account Number, and the cheque amount.
For personal cheques, specify PERSONAL. The cheques are smaller, approximately 2.75" x 6". The personal cheque MICR Line consists of, from left to right, the sort code number, account number, cheque number, and cheque amount.
Example:
CHEQUETYPE(B)

CHEQUEXADJUST
Use this Form Parameter to adjust the horizontal placement of all data fields in an application.
CHAPTER 1: FORM PARAMETERS

Form Parameters

Options: n = a numeric value
Default: n = 0
Requirement: None
Description:
The numeric value, n, moves all of the data fields in an application n/300 inches horizontally. A positive value will move the data fields toward the bottom of the page, and a negative value will move the data fields toward the top of the page. CHEQUEXADJUST does not reposition the Form, only the data fields.

This parameter may be included multiple times. The first CHEQUEXADJUST specifies the adjustment of the top cheque's data fields, the second CHEQUEXADJUST specifies the adjustment of the second cheque's data fields, etc.

Example:

CHEQUEXADJUST(5)

CHEQUEYADJUST

Use this Form Parameter to adjust the vertical placement of all data fields in an application.

Options: n = a numeric value
Default: n = 0
Requirement: None
Description:
The numeric value, n, moves all of the data fields in an application n/300 inches vertically. A positive value will move the data fields toward the bottom of the page, and a negative value will move the data fields toward the top of the page. CHEQUEYADJUST does not reposition the Form, only the data fields.

This parameter may be included multiple times. The first CHEQUEYADJUST specifies the adjustment of the data fields of the top cheque, the second CHEQUEYADJUST specifies the adjustment of the data fields of the second cheque, etc.

Example:

CHEQUEYADJUST(5)

CODEFONT

Use this Form Parameter to select either MICR or OCRB font for the CODE line.
CHAPTER 1: FORM PARAMETERS

Form Parameters

Options: MICR or OCRB
Default: MICR
Requirement: None
Description:
The CODE line is formatted using either MICR or OCRB for compatibility with standardized readers.
Example:
CODEFONT(MICR)

CODEFONTSWITCH

Use this Form Parameter to indicate whether a code font switch character ‘X’ is to be printed on the code line.

Options: No or Yes
Default: No
Requirement: None
Example:
CODEFONTSWITCH(N)

CODELAYOUT

Use this Form Parameter to select A, C, D, E1, E2 to determine the code layout for different forms of credits and cheques.

Options: A, C, D, E1, E2
Default: A
Requirement: None
Description:
Defines the layout of the CODE line for selection.
Example:
CODELAYOUT(A)
CHAPTER 1: FORM PARAMETERS

CURRENCY

Use this Form Parameter to select which currency is being used in displaying amounts. It is used to perform the Euro and US dollar conversions.

Options: UK, US, EU
Default: UK
Requirement: None
Example:
CURRENCY(UK)

FIELDFILE

Use this Form Parameter to create an ASCII test file to be used by a postprocessor.

Options: Yes/name of the file to be created
Default: No field file created
Requirement: None
Description:
FIELDFILE is used to create an ASCII text file containing all printed check data. The file can be used by a postprocessing program.
Selecting the “Yes” option creates a field file called $$BT$$$.DAT.
Selecting the “filename” created a fieldfile by the specified file name. For example, if the parameter is listed as FIELDFILE(DATA.DAT), PayBase creates a text file called DATA.DAT.
The format of the fieldfile is as follows:
^FORM FormName
^Field FieldName FieldData
^EJECT
where
“FormName” is the name of the form created in PayBase.
“FieldName” is the name of a Link File field.
“FieldData” is the value of that field.
^EJECT begins a new page.
Example:
FIELDFILE(CHECK.DAT)
FORMFIELDNAME

This parameter is used when two or more Form Designs are printed from the same data file. This procedure is also referred to as Form Switching.

**Options:** a .MOC filename

**Default:** the .CGL filename

**Requirement:** a .MOC file

**Description:**
The .MOC filename is the path and the filename of the Form to be printed when this Link File is included in a PayBase application. If no path is listed, the .MOC file is assumed to be in the current drive and directory. The Form Parameter FORM is the first parameter listed in a .CGL file, and all other Form Parameters should follow it.

**Example:**
FORM(Payroll)

FORMSPERPAGE

Use this parameter to specify the number of checks per page.

**Options:** 1 - 4

**Default:** 1

**Requirement:** A Form Design with the same number of checks per page.

**Description:**
The numeric value, n, specifies the number of checks to be printed on each page. The form must be designed with the same number of checks per page. PayBase begins a new page after “n” checks have been printed.

**Example:**
FORMSPERPAGE(4)

FORMTYPE

Use this Form Parameter to specify printing of multiple manual data entry documents using blank checks for starter kit applications.

**Options:** Single/Multiple

**Default:** Single

**Requirement:** Proper Form Design
CHAPTER 1: FORM PARAMETERS

Form Parameters

- Description:
The option “S” specifies printing of a single copy. The option “M” specifies printing of multiple copies, and PayBase prompts the user to enter the number of checks required.

FORMTYPE(M) prints the same check multiple times and should be used only when printing blank checks for starter kit applications or when printing non-MICR documents.

Example:
FORMTYPE(m)

GROUPFIELDNAME

This Form Parameter is used when a Group File is used in an Application. It identifies the Group File field name.

Options: x = a Link File field name
Default: None
Requirement: A Group (.CGI) file included in the Application

Description:
When a Group File is used, the Link File must contain a data field that is the name of a Group key.

The name of this data field is included as the option of the GROUPFIELDNAME parameter. A Non-Printing Field is often created as the field name in this instance.

Example:
GROUPFIELDNAME(NONPRINTING_key)

INTERCHECKADJUSTMENT

Use this Form Parameter to adjust data field placement on multiple-check documents.

Options: n = a numeric value
Default: None
Requirement: FORMSPERPAGE with a value of (2), (3), or (4)

Description:
This Form Parameter adjusts the vertical placement of data fields on every check EXCEPT the topmost check on a multiple-check document. It DOES NOT affect the MICR Line. The numeric value, n, moves the second check n/
300 inches, the third check $2n/300$ inches, and the fourth check $3n/300$ inches on the page. A positive value for $n$ moves the check toward the bottom of the page, and a negative value for $n$ moves the check toward the top of the page.

For example, when printing 4 checks per page, the Form Parameter INTERCHECKADJUSTMENT(6) would not move the first check; it would move the second check $6/300$ inches down the page; it would move the third check $2 \times 6/300$ or $12/300$ inches down the page; and it would move the fourth check $3 \times 6/300$ or $18/300$ inches down the page.

Example:

INTERCHECKADJUSTMENT(3)

**INTERCHEQUEADJUSTMENT**

Use this Form Parameter to adjust data field placement on multiple-cheque documents.

**Options:** $n$ = a numeric value  
**Default:** None  
**Requirement:** FORMSPERPAGE with a value of (2), (3), or (4)  
**Description:**

This Form Parameter adjusts the vertical placement of data fields on every cheque EXCEPT the topmost cheque on a multiple-cheque document. It DOES NOT affect the MICR Line. The numeric value, $n$, moves the second cheque $n/300$ inches, the third cheque $2n/300$ inches, and the fourth cheque $3n/300$ inches on the page. A positive value for $n$ moves the cheque toward the bottom of the page, and a negative value for $n$ moves the cheque toward the top of the page.

For example, when printing 4 cheques per page, the Form Parameter INTERCHEQUEADJUSTMENT(6) would not move the first cheque; it would move the second cheque $6/300$ inches down the page; it would move the third cheque $2 \times 6/300$ or $12/300$ inches down the page; and it would move the fourth cheque $3 \times 6/300$ or $18/300$ inches down the page.

Example:

INTERCHEQUEADJUSTMENT(3)

**LTRIM**

Use this parameter if leading spaces in the Data File are to be printed on the checks.

**Options:** Yes/No
Default: No  
Requirement: Application includes a Data File  
Description:  
This Form Parameter indicates whether or not leading spaces in the data file should be included on checks. "Y" indicates that all leading spaces SHOULD NOT be printed, "N" indicates that all leading spaces SHOULD be printed.  
To control the printing of leading spaces for individual fields rather than all fields on the whole form, the LRECL parameter can be used as a Field Parameter. See LTRIM Field Parameter.  
Example:  
LTRIM(N)  

MICRFONT  
Use this parameter to assign a MICR font code for a cheque design.  
Options: CMC7, MICR, OCRA, OCRB  
Default: CMC7  

OUTTRAY  
The OUTTRAY form parameter is designed to work with all supported BT printers. The following options are available for configuration:  
• Active (A) - Activates the printer and uses all default printer settings.  
• Left (L) - Selects the optional Left output bin.  
• Standard (S) - Specifies the standard default output bin.  
• Top (T) - Selects the standard Top output bin.  
• 1 - 7 - Specifies the desired tray number when printing.  
To limit which tray output goes, select OUTTRAY(nnn...) when 'n' is a numeric figure of the three possible options, 5, 6 or 7. Again the order cannot be changed; output first goes to Tray 7, then Tray 5 and finally Tray 6, but the user can limit the number. For example, OUTTRAY(75) will send output to Tray 7. When it is full, the printer will use Tray 5.  

Note  
MICR printing is not supported in manual feed Tray 1. Also, all printers are configured to check Tray 1 first.
Examples:
OUTTRAY(T)
OUTTRAY(A)
OUTTRAY(75)

PAPERDEPTH
Use this Form Parameter if the check stock length is not 11 inches, and the check is positioned below the stub on the page.

Options: 7, 11, 11.69, 14
Default: 11

Requirements:
- REFERENCE(?AUTOMICR) Field Parameter
- AUTOMICR Field Parameters

Description:
The number indicated in this Form Parameter indicates the distance in inches from the top of the page to the bottom of the page. This measurement is used to position the MICR Line.
If the check is located above the stub on the page, use the CHECKDEPTH Form Parameter.

Example:
PAPERDEPTH(14)

PAYMENTMETHOD
Use this Form Parameter to specify the types of payments used in transactions.

Options: Check(C), ACH(A), PPLX(P), Void(V), BACS(B)
Default: Check

Requirements: Check Fraud Avoidance (CFA) and/or BACS

Description: This parameter takes a character as arguments to designate a payment as either a check or a void record. The value ‘C’ designates the form as a Check. The ‘C’ value is for records that have been made and will be transmitted as a payment. The value ‘B’ designates this payment will process as a BACS payment. The value ‘V’ designates the form as voided. The ‘V’
value is for records that are voided checks that simply need to be put into the Check Fraud Avoidance database. This is any easy method of entering many void records at a time.

Example:
PAYMENTMETHOD(C)

PERSISTRESET
The PERSISTRESET form parameter is used to control the timing of the reset operation for persisted fields that have logical record scope. Once a logical record marker has been encountered PayBase must decide to either reset before processing the next record or processing and then resetting. The form parameter PERSISTRESET is used to facilitate this functionality. The two possible values for PERSISTRESET are PRE or POST. PRE indicates that PayBase should reset the persisted values and then process the current record. POST indicates that PayBase should process the current record and then reset. See the Field Persistence section for more information.

PRINTERFIELDNAME
Use this parameter if Multiple Printer Support is being used which directs specific check records to specific printers.

Options: FIELD0, FIELD1, FIELD2
Default: None
Requirement:
• PayBase with Multiple Printer Support
• PRINTERLIST Form Parameter
Description:
When an Application uses Multiple Printer Support and directs check to specific printers, the Link File must contain a field that is the name for the printer be used to print the current check record. The name of this Link File is included as the option of the PRINTERFIELDNAME parameter.

A Non-Printing Field is often created for the PRINTERFIELDNAME Form Parameter as PRINTERFIELDNAME(NONPRINTING_PRNT). This indicates that PayBase will use Directed Printing, and PayBase can find the name of the printer that will print each document in the field named NONPRINTING_PRNT. The NONPRINTING_PRNT field contains the name of one of the printers listed in the PRINTERLIST parameters. PayBase
reads this PRINTERLIST printer name in the NONPRINTING_PRNT field of each document and sends the document to that printer.

Example:

PRINTERFIELDNAME(NONPRINTING_PRINTER_NAME)

**REPEAT**

Use this parameter MDE Designs to display data from every field of the previous entry screen on the current entry screen.

Options: Yes/No

Default: No

Requirement: None

Description:

If the REPEAT parameter is set to “Y”, only the first data entry screen is blank, and all subsequent screens are displayed with the data from the previous screen still in place. Only fields to be repeated by using the Repeat function when defining fields.

Example:

REPEAT(Y)

**TRAY**

Use this Form Parameter to tell PayBase which paper tray to use to print the Application.


Default: x = U

Requirement: None

Description:

If the printer is to print the Application from any paper tray other than the upper tray, this Form Parameter must be included.

"U" indicates that the upper paper tray will be used.

"C" indicates the center tray will be used.

"L" indicates that the lower paper tray will be used.

"M" indicates that the manual paper tray will be used.
"F" signifies the front manual feed will be used.

"B" means the back manual feed will be used.

"H" indicates the high capacity tray will be used.

"A" signifies the auxiliary tray will be used.

Numbers "1-6" indicates that an assigned printer tray will be used. The user will assign the printer tray numbers 1-6 to the specific location of the tray, or the tray may already be assigned a number by the manufacturer. For instance, the Dataproducts Typhoon 30 and HP 5Si printers have trays that are numbered from 1 through 5. Simply choose the tray number from which stock will be taken.

For the Typhoon printer, please note in the example below that there cannot be commas in between the numbers and that grouping of trays must be the numbers, not letters.

**Example:**

TRAY(U)

Example (for the Typhoon 30 printer):

TRAY(AUTO)

TRAY(163)
FIELD PARAMETERS

In this chapter we will explore the following topics:

- Field Parameters
Field Parameters

Field Parameters are modifiers which affect only the field for which they are listed. For example, individual field settings such as the field length, justification, and format are specified using field parameters.

Field Parameters are located after the Form Parameters in the Link File. They are listed as the parameter or parameter abbreviation with an option listed in parentheses, for example VERTICALPOSITION(22).

The following Field Parameters are available:

- ACCUMULATOR
- AUDIT
- AUDITCODE
- AUTOMICR
- CALENDARPOPUP
- CODETYPE
- DECIMALPOINT
- EDITMASK
- FIELDREFERENCE
- FORMATCASE
- HORIZONTALPOSITION
- IF
- INPUTMASK
- JUSTIFY
- JUSTIFYLENGTH (Output Length)
- LENGTH (Input Length)
- LTRIM
- MICRTYPE
- MINLENGTH
- MODULUS
- ORIENTATION
• POSTFIX
• PPLX
• PREFIX
• REFERENCE
• REMOVECHARS
• REPLACE
• ROTATEFIELD
• SPLIT
• STARTINGVALUE
• TRACKING
• TYPE
• VALIDATE
• VERTICALPOSITION
• XADJUST
• YADJUST

**ACCUMULATOR**

This Field Parameter will add field data to or subtract the field data from a totals field.

**Options:** +/-, 1 - 9

**Default:** None

**Requirements:** REFERENCE(?n) Field Parameter, where n is the same number as the ACCUMULATOR value.

**Description:**

The values "n" identifies the REFERENCE number of the calculating field. "n" or "+n" values are added to the REFERENCE field, while "+n" values are subtracted from the REFERENCE field.

In a Link File, a total of nine addition or subtraction calculations can be performed. Each calculation has its own REFERENCE number. Each field with a calculated value is assigned a REFERENCE(?n) value, where n can be any number from 1 to 9.
The value of "n" assigned to the ACCUMULATOR parameter identifies the REFERENCE value that this field value is to be added to or subtracted from. A "n" or "+n" ACCUMULATOR value is added to the REFERENCE value, while a "-n" is subtracted from the REFERENCE value.

Example:
A net pay field is to be printed on the check, and the value of this field is NOT included in the data file. But the data file does include a gross pay field and a taxes field, and the net pay equals the gross pay minus taxes. REFERENCE and ACCUMULATOR Field Parameters can be used to calculate the net pay field.

1. Assign the Field Parameter ACCUMULATOR(+1) to the gross pay field. This means that the value of this field will be added to REFERENCE calculation #1.

2. Assign the Field Parameter ACCUMULATOR (-1) to the taxes field. This means that the value of this field will be subtracted from REFERENCE calculation #1.

3. Assign the Field Parameter REFERENCE(?1) to the net pay field. This means that this field will be the destination for any ACCUMULATOR Field Parameters assigned a 1 (either negative or positive). Remember, this field must be listed in the Link File AFTER any fields that it is referencing. In other words, all ACCUMULATOR fields must fall before the REFERENCE fields when calculating field values.

AUDIT
Use this Field Parameter to send the field value to the audit log when the check is printed.

Options: AMOUNT, CHECKNUMBER, PAYEE, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, +1, +2, +3, +4, +5, +6, +7, +8, +9, +10, +11, +12, +13, +14

Default: None

Requirement: AUDITTYPE(DETAIL) or AUDITTYPE(BATCH) Form Parameter

Description:
PayBase has the ability to audit the following items
Date
Time
User Name
CHAPTER 2: FIELD PARAMETERS

Function or feature performed

Check amount

Check number

Eight 100 character fields and seven 50 character fields defined by you.

Each item is listed in the audit log in a column. The Date, Time, User Name, and Function are automatically recorded in the audit file, but all other items must be assigned as AUDIT Field Parameters.

"x" is one of the following values

AMOUNT

AUDIT(AMOUNT) sends the field value to the Check amount column in the Audit File.

CHECKNUMBER

AUDIT(CHECKNUMBER) sends the field value Check number column in the Audit File.

PAYEE

AUDIT(PAYEE) sends the field value to the Payee column in the Audit File.

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

AUDIT(1), AUDIT(2), AUDIT(3), AUDIT(4), AUDIT(5), AUDIT(6), AUDIT(7), AUDIT(8) AUDIT(9), AUDIT(10), AUDIT(11), AUDIT(12), AUDIT(13) AUDIT(14), AUDIT(15). Each of these Field Parameters sends the value of the field to a user defined auditing column in the Audit File. These are optional audit fields that can be used to audit any data. The maximum field length is 100 characters for 1, 2, 3, 4, 5, 6, 7, 8 and 50 characters for 9, 10, 11, 12, 13, 14, 15.

+1, +2, +3, +4, +5, +6, +7, +8, +9, +10, +11, +12, +13, +14

AUDIT(+1), AUDIT(+2), AUDIT(+3), AUDIT(+4), AUDIT(+5), AUDIT(+6), AUDIT(+7), AUDIT(+8), AUDIT(+9), AUDIT(+10), AUDIT(+11), AUDIT(+12), AUDIT(+13), AUDIT(+14). Each of these Field Parameters sends the value of the field to an auditing column that uses TWO user-defined fields (as described above). AUDIT(+1) actually uses audit file fields AUDIT(1) and AUDIT(2), AUDIT(+3) uses AUDIT(3) and AUDIT(4), AUDIT(+5) uses AUDIT(5) and AUDIT(6) and AUDIT(+7) uses AUDIT(7) and AUDIT(8), AUDIT(+9) uses AUDIT(9) and AUDIT(10), AUDIT(+11) uses AUDIT(11) and AUDIT(12), AUDIT(+13) uses AUDIT(13) and AUDIT(14), AUDIT(+14) uses AUDIT(14) and AUDIT(15). The maximum field length of a double audit field is 200 characters.
A user defined field may be used as a single field or as part of a double field, but it may NOT be used as both. For example, if an Application uses the Field Parameter AUDIT(+1), it reserves fields AUDIT(1) and AUDIT(2). The next auditing field must be AUDIT(3) or AUDIT(+3) since AUDIT(1) and AUDIT(2) are reserved.

Example:
AUDIT(AMOUNT)
AUDIT(+1)

**AUDITCODE**

The following field values are available for this parameter:

**Options:** REFERENCENUMBER, SORTCODE, ACCTNUMBER, AMOUNT, TRANSACTIONCODE

**AUTOMICR**

Use this Field Parameter to automatically place the field value on the MICR Line. If the MICR button is selected from the Non-visible fields box, this parameter is automatically added to the field.

**Options:** INTRANSIT, ONUS, AMOUNT, EXTRA, CHECKNUMBER

**Default:** None

**Requirement:** REFERENCE(?AUTOMICR) Field Parameter

**Description:**
A MICRLine consists of the following five components

AUXILIARY
INTRANSIT
ONUS
EXTRA ONUS
AMOUNT

The AUTOMICR parameter defines a field value as one of these MICR Line components. AUTOMICR can also be used with the options PAYEE or LOCATION as described below.

"x" is one of the following values
CHAPTER 2: FIELD PARAMETERS

Field Parameters

INTRANSITAUTOMICR(INTRANSIT) automatically places the field value on the MICR Line in the intransit position.

ONUSAUTOMICR(ONUS) automatically places the field value on the MICR Line in the onus position.

AMOUNTAUTOMICR(AMOUNT) automatically places the field value on the MICR Line in the amount position.

EXTRA CHECKNUMBERAUTOMICR(CHECKNUMBER) automatically places the field value on the MICR Line in the auxiliary onus position.

Example:

AUTOMICR(INTRANSIT)

CALENDARPOPUP

Use this field parameter to have MDE date fields have a popup. The field parameter allows you to pick the date and not have to manually type it in.

CODETYPE

The following field codes are available for this parameter:

Options: REFERENCENUMBER, SORTCODE, ACCTNUMBER, AMOUNT, TRANSACTIONCODE

DECIMALPOINT

Use this Field Parameter to define the decimal point format for the field value. It will NOT affect field values that already contain decimal points.

Options: E/I, 1-6
Default: E
Requirement: TYPE(N) field setting
Description:

- The option "E" specifies an explicit decimal point: ".00" is added to the field value. For example, the field value "125" would be printed as "125.00".

- The option "I" specifies an implied decimal point: a decimal point is inserted before the last 2 digits of the field value. For example, the field value "125" would be printed as "1.25".
The following are examples of data values with various DECIMALPOINT options if the EDITMASK parameter for the field is listed as "##.##":

Field value DP(E) Printed result DP(I) Printed result

<table>
<thead>
<tr>
<th>Field value</th>
<th>DP(E)</th>
<th>Printed result</th>
<th>DP(I)</th>
<th>Printed result</th>
</tr>
</thead>
<tbody>
<tr>
<td>.3</td>
<td>0.30</td>
<td>0.30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.00</td>
<td>3.00</td>
<td>3.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>30.00</td>
<td>0.30</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Example:**

DECIMALPOINT(E)

**EDITMASK**

**Abbreviation:** E

Use this Field Parameter to format the value of a date field (see Date Edit Masks below) or a numeric field (see Numeric Edit Masks below). Edit Masks are built through the Edit Mask dialog box available when you click the Edit button in the Add/Change Field Parameter dialog box.

**Date Edit Masks**

**Options:** x = [B] [9/2] $DATEn

**Default:** None

**Requirements:** for Date Edit Masks, TYPE(D) Field Parameter

**Description:**

The data for the field MUST be in the format mm/dd/yy or mmddyy in order to be formatted using an EDITMASK parameter. For example, EDITMASK will format 01/28/98 or 012898, but it will not format January 31, 1998 or 310198. If you choose a Date Edit Mask with a Century option (eg. 1998) you must click either the “19yy” or “20yy” format. This will insert a “9” or “2” in the “Current Edit Mask” view. You also have the option to insert a blank field if no information is given.

**PayBase Date Edit Masks**

In addition to the previous Edit Masks, PayBase supports the following Edit Masks:

<table>
<thead>
<tr>
<th>$Date10</th>
<th>310198</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Date11</td>
<td>31011998</td>
</tr>
<tr>
<td>9$Date12</td>
<td>31-01-1998</td>
</tr>
</tbody>
</table>
Notes

- If “Blank Field if no information is found” is checked, a blank field is printed when no date information is found in the data source. If this option is not selected, and no date information is found in the data source, the current computer system date is printed.

- The previous century examples (9$DATE5January 31, 1998) are based on “19yy” format. If “20yy” chosen the previous example would read: 2$DATE5January 31, 2098.

Numeric Edit Masks

Options: x = [B] ** [+] [$/]$ #...## [.] [.] #...#

Default: None

Requirements: for Numeric Edit Masks, TYPE(N) Field Parameter

See DECIMALPOINT Field Parameter explanation below

Description: Numeric Edit Masks can be built a number of ways. Values are associated with each option selected. Observe the “Current Edit Mask” view as you build your numeric Edit Mask to view these values.

The following table lists some numeric EDITMASKS, the number to be formatted, and the printed result:
Numeric Edit Mask Examples

Table 2.2

<table>
<thead>
<tr>
<th>Value</th>
<th>Current Edit Mask</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>.###</td>
<td>.120</td>
</tr>
<tr>
<td>12</td>
<td>##.##</td>
<td>0.12</td>
</tr>
<tr>
<td>12</td>
<td>##</td>
<td>.12</td>
</tr>
<tr>
<td>1.2</td>
<td>##.##</td>
<td>1.20</td>
</tr>
<tr>
<td>12.345</td>
<td>###.##</td>
<td>12.35</td>
</tr>
<tr>
<td>12.34</td>
<td>+##.##</td>
<td>+12.34</td>
</tr>
<tr>
<td>-12.34</td>
<td>+##.##</td>
<td>-12.34</td>
</tr>
<tr>
<td>12.34-</td>
<td>+##.##</td>
<td>-12.34</td>
</tr>
<tr>
<td>12.34</td>
<td>+##.##</td>
<td>+12.34</td>
</tr>
<tr>
<td>-12.34</td>
<td>+##.##</td>
<td>-12.3</td>
</tr>
<tr>
<td>12.34</td>
<td>$$##.##</td>
<td>$12.34</td>
</tr>
<tr>
<td>12.34</td>
<td>**##.##</td>
<td>12.34</td>
</tr>
<tr>
<td>-0.1</td>
<td>**+##.#</td>
<td>-0.1</td>
</tr>
<tr>
<td>0.1</td>
<td>**##.#</td>
<td>0.1</td>
</tr>
<tr>
<td>1.23</td>
<td>**$##.##</td>
<td>*$1.23</td>
</tr>
<tr>
<td>1234.56</td>
<td>1.23</td>
<td>1,234.56</td>
</tr>
<tr>
<td>1234.56</td>
<td>B**$##.##</td>
<td>$1,234.56</td>
</tr>
<tr>
<td>123.45</td>
<td>**##.##</td>
<td>123.45</td>
</tr>
</tbody>
</table>

Notes

- **Numeric EDITMASK fields which print with a percent sign (%) indicate that the field value contains more digits than the EDITMASK parameter allows. Extra pound signs must be added to the EDITMASK which allow for all of the field value digits so that the field will print correctly.**

  For example, if the data value is 1234.56 with EDITMASK(##.##), the field will be printed as "%.34.56", indicating that there are additional field value digits which are not printed. Add two additional pound signs, making EDITMASK(####.##), and the number will be printed as "1234.56"

- **The “Use decimal point” option affects the way an EDITMASK formats a field value by specifying an explicit or implied decimal point. This option takes precedence over the EDITMASK parameter. An explicit DECIMALPOINT (DP(E)) adds “.00” to the field value, while an implied DECIMALPOINT (DP(I)) inserts a decimal point before the last two digits of the field value. The DECIMALPOINT parameter does not affect values that already include decimal points.**
FIELDREFERENCE

The FIELDREFERENCE field parameter takes three arguments, OPERATION, KEY, and SCOPE. The OPERATION arguments are READ, WRITE, or DELETE. In order to persist a field between forms the field key should have a FIELDREFERENCE (or FR) parameter and the operation should be WRITE. The KEY argument is the name of the persisted value assigned by the creator of the CGL. This name is used as the lookup for the value. The name should be unique per application. If two fields are named the same and both write data the second write operation will overwrite the first. The SCOPE argument indicates LOGICAL_RECORD or DATA_FILE. LOGICAL_RECORD scope indicates the field value should be erased when the logical record indicator, (LRM) as specified in the application, is encountered. DATA_FILE scope indicates the field value should be persisted for the entire data file without being erased by the process cycle between logical records.

The WRITE operation creates a new entry in the persisted fields list and copies the current value of the CGL field or overwrites the existing value in the list. Performing a read operation without having first written the field to the list results in a new entry in the list with an empty field value.

The READ operation uses the supplied KEY argument and looks it up in the list. If found the value is placed in the CGL field. If not found a new entry is created, a blank value is created, and the blank is copied to the field in the CGL.

The DELETE operation will remove the persisted field from the list. It destroys the field, not just erases it.

Some examples of FIELDREFERENCE.

In CGL1 we have a field we want to persist.
FIELD(MY_TEST_FIELD)
VERTICAL(2)
HORIZONTAL(20)
LENGTH(5)
FIELDREFERENCE(WRITE, CGL1.MY_TEST_FIELD, DATA_FILE)

In CGL2 we have a field we want to populate with data that was persisted from a previous form.
FIELD(RECEIVING_FIELD)
FR(READ, CGL1.MY_TEST_FIELD)

And then later on in the run we want to explicitly delete the field.
FIELD(DUMMY_FOR_DELETE)
FIELDREFERENCE(DELETE, CGL1.MY_TEST_FIELD)

For the most part, you will not want to explicitly delete a field during a run but the ability to do so seems natural given the fact that we are allowing them to create and modify fields. Also, it is possible to specify a delete operation in the same field as the read operation. PayBase will load a form's worth of data into the link file, resolve the group information, then perform the read operation before any other field parameter operation. After all the other field parameters are processed the write operation is performed. And last of all the delete operation is performed. See the Field Persistence section for more information.

FORMATCASE
Use this field parameter to specify the format of the field data.

Options: NONE, FIRST, UPPER, LOWER, SMART
Default: None
Requirement: None
Description: Available by right-clicking on a Field Branch and selecting Add/Change Field Parameters using FORMATCASE. Use the Options drop-down to select the appropriate case format. NONE will specify no field format. FIRST will capitalize only the first character of the field data. UPPER will specify all field characters to be capitalized. LOWER will specify all characters to be lower case. The SMART option will capitalize the first character of every word after a space in the field data.
Example:
FORMATCASE(FIRST)

HORIZONTALPOSITION
Use this parameter to specify horizontal field position. If the Application uses a data file, it specifies the horizontal position of the data field.

Options: any numeric value
Default: None
Requirement: None
Description: Available by right-clicking on a Field Branch and selecting Add/Change Field Parameters using HORIZONTALPOSITION. Use the Value field to type a numeric value.
Example:
HORIZONTALPOSITION(5)

**IF**

**Note**

*The IF field parameter has been expanded, allowing you to build more complicated expressions. However, you will need to use a LOGIC_BLOCK to build an expanded IF statement. See the LOGIC_BLOCKS section for more information.*

Use this parameter for conditional printing of a field

**Options:** see explanations below

**Requirement:** Link File fields referred to in an IF statement MUST be listed BEFORE the IF statement; when a Field and a number are compared, the Field must contain the TYPE(N) Field Parameter.

**Description:**

Use this parameter to list conditional situations that must be met in order for the field value to be printed. A maximum of TWO IF parameters may be assigned to each Link File field.

"x1" is a field name. It must be the name of a field that is listed in the Link File BEFORE the current field. To name the current field, list x1 as an asterisk (*).

"o" is one of the following comparisons:

- = value of x1 is the same as x2
- <> value of x1 is not the same as x2
- > value of x1 is greater than x2
- < value of x1 is less than x2
- >= value of x1 is greater than or equal to x2
- <= value of x1 is less than or equal to x2

"x2" is an alphanumeric value that specifies a field name, a text value, or a numeric value to be compared with the value of the field "x1". If "x2" is a text string, it must be surrounded by quotation marks ("First National Bank"). To list a blank for the value of "x2", list a blank space surrounded by quotation marks (" ").
If "x2" is a numeric value, a numeric comparison is performed with the value of "x1". If the value of "x2" is one or more words, a word comparison is performed.

When using IF statements for comparisons, it is important to know that all ASCII characters (punctuation marks, numbers, and letters) have a relative size. For example, the value "50" is smaller than the value "100", just as the value "BOTTOMLINE" is smaller than the value "TECHNOLOGIES". The following table is a list of most ASCII characters used in IF statements, from smallest to largest:

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>exclamation point</td>
</tr>
<tr>
<td>&quot;</td>
<td>quotation mark</td>
</tr>
<tr>
<td>#</td>
<td>pound sign</td>
</tr>
<tr>
<td>$</td>
<td>dollar sign</td>
</tr>
<tr>
<td>%</td>
<td>percent sign</td>
</tr>
<tr>
<td>&amp;</td>
<td>ampersand</td>
</tr>
<tr>
<td>'</td>
<td>single quotation mark</td>
</tr>
<tr>
<td>(</td>
<td>parentheses</td>
</tr>
<tr>
<td>*</td>
<td>asterisk</td>
</tr>
<tr>
<td>-n</td>
<td>negative numeric values</td>
</tr>
<tr>
<td>0</td>
<td>zero</td>
</tr>
<tr>
<td>+n</td>
<td>positive numeric values</td>
</tr>
<tr>
<td>?</td>
<td>question mark</td>
</tr>
<tr>
<td>A-Z</td>
<td>uppercase letters</td>
</tr>
<tr>
<td>a-z</td>
<td>lowercase letters</td>
</tr>
</tbody>
</table>

Note

Lowercase letters are considered to be larger than uppercase letters; for example the word “Animal” is smaller than the word “animal”. Numbers are always smaller than uppercase and lowercase letters; “500.00” is smaller than “ten dollars”.

Examples:

1 In a PayBase Application, a signature should be printed on checks which have amounts of $5000 or less, and the comment “Requires two signatures” should be printed on checks which have amount of more than $5000. The Link File should be organized as follows:
FIELD(AMOUNT)
  V(1)
  H(1)
  L(15)

FIELD(SIGNATURE)
  STARTINGVALUE(32001)
  TYPE(G)
  IF(AMOUNT,<=,5000)

FIELD(REQUIRE)
  STARTINGVALUE(Requires two signatures)
  IF(AMOUNT,>,5000)

PayBase will print the AMOUNT field on every check. PayBase will print the SIGNATURE field, which is a numbered graphics file, if the AMOUNT field value is less than or equal to 5000. PayBase will print the REQUIRE field starting value (Requires two signatures) if the AMOUNT field value is greater than 5000.

2 To print a bank name conditionally, depending upon the city in which the bank is located, the Field Parameters would be listed as follows:

FIELD(BANK_CITY)
  V(1)
  H(30)
  L(10)

FIELD(DESCRIPTION)
  S(First National Bank of Boston)
  IF(BANK_CITY,=,"BOSTON")

FIELD(DESCRIPTION)
  S(First National Bank of New York)
If(BANK_CITY,=,"NEW YORK")

In this example, the BANK_CITY value will be printed as listed in the data file or on the data entry screen. The DESCRIPTION field will only be printed conditionally: if the FIELD(BANK_CITY) has a value of "BOSTON", the DESCRIPTION field value will be printed as "First National Bank of Boston". The DESCRIPTION field is then repeated in the Link File with a new printing condition: when FIELD(BANK_CITY) has a value of "NEW YORK", the DESCRIPTION field value will be printed as "First National Bank of New York".

3. The next IF statement prints a second signature on any check which has an amount greater than $5,000.00.
FIELD(CHECK_AMT)
V(10)
H(10)
L(15)

FIELD(SIGNATURE1)
S(32001)
TYPE(G)

FIELD(SIGNATURE2)
S(32002)
TYPE(G)
IF(CHECK_AMT,>,5000.00)
The CHECK_AMT field and the SIGNATURE1 field will be printed on every check. The SIGNATURE2 field will only be printed if the CHECK_AMT field value is greater than 5000.00.

4. An IF statement can be used to replace a signature with a line of text. The following example prints "**VOID**" on any check which has a negative check amount.
FIELD(CHECK_AMT)
V(5)
H(10)
L(15)
In this example, the CHECK_AMT field will be printed on every check as listed in the data file or on the manual data entry screen. The SIGNATURE1 field value will be printed as graphic file 32001 only if the CHECK_AMT field value is greater than zero. The SIGNATURE1 field is listed again, this time with another condition: if the CHECK_AMT field is zero or less, the SIGNATURE1 field value will be printed as "**VOID**".

Precise alignment of multiple logo/signature fields can be done using IF statements. For example, a single field is often required to print two different logos. Logo 32001 prints when Company A checks are being printed; logo 32002 prints when company B checks are being printed. XADJUST and YDJUST parameters can be used in cooperation with IF statements to place each logo at the correct location on the check.
S(32002)

TYPE(G)

IF(COMPANY,=,"COMPANY B")

XADJUST(-75)

In this example, the COMPANY field value will be printed on every check as listed in the data file or on the manual data entry screen. The LOGO field value will be printed conditionally. If the COMPANY field value is “COMPANY A”, FIELD(LOGO) field value will be printed as graphic file 32001 and it will be adjusted 150/300 inches to the right. If the COMPANY field value is “COMPANY B”, FIELD(LOGO) field value will be printed as graphic file 32002 and it will be adjusted 75/300 inches to the left.

**INPUTMASK**

Use this field parameter to specify the input date format.

- **Options**: Several combinations of month, day and year and NONE.
- **Default**: None
- **Requirement**: The TYPE(D) field parameter must be included to specify the field as a date field.
- **Description**: Available by right-clicking on a Field Branch and selecting Add/Change Field Parameters using INPUTMASK. Click Edit and select the format allowed for the input date.
- **Example**:
  
  INPUTMASK(MMYYDD)

**JUSTIFY**

JUSTIFY is used to position character entries within a field in the check design.

- **Abbreviation**: J
- **Options**: L/C/R, where L=Left, C=Center and R=Right
- **Default**: L
- **Requirement**: JUSTIFYLENGTH field parameter is required when the length of the field in the PayBase Designer+ form design is not the same as the LENGTH field parameter, or when the LENGTH parameter is not included.
Example:
JUSTIFY(L)

Description:
Justification for each field to be printed must be specified in PayBase Designer+ and in the Link (.LNC) file.
PayBase determines the placement of a justified field as follows:
If the field has a numeric EDITMASK field parameter, placement is determined by the number of characters in the Edit Mask.
If the field does not have a numeric EDITMASK parameter, placement is determined by the JUSTIFYLENGTH parameter.
If the field does not have a JUSTIFYLENGTH parameter, placement is determined by the LENGTH parameter.

Note
Changing the number or character in the EDITMASK parameter, the value of the JUSTIFYLENGTH parameter, or the value of the LENGTH parameter will change the position of the field on the document.

JUSTIFYLENGTH (Output Length)
Use this Field Parameter to define the length (Output Length) of the field to be printed for justification positioning.

Abbreviation: JL
Options: n = a numeric value
Default: the value of the Field Parameter LENGTH (Input Length); or, if specified, the length of the EDITMASK Field Parameter. When neither LENGTH nor EDITMASK is used, the default n = 0.
Requirement: None
Description:
“n” is a numeric value which specifies the width of the field.
JUSTIFYLENGTH (Output Length) is used by PayBase to position a field correctly. JUSTIFYLENGTH (Output Length) needs to be included when the length of the field specified in the Form Design is not the same as the length of the field specified in the Link File.
JUSTIFYLENGTH (Output Length) must be included when the LENGTH parameter is not used, as when using a Group File to obtain data for the field.
Example:
**LENGTH (Input Length)**

Use this Field Parameter to specify the input length of the data field.

**Options:** $n$ = a numeric value, $n!$ = a numeric value followed by an exclamation point.

**Default:** $n = 0$

**Requirement:** JUSTIFYLENGTH (Output Length) Field Parameter when the value of the LENGTH (Input Length) Field Parameter is not the same as the length of the field in the Form Design. Type(N) field setting when "!" is used.

**Description:**

"$n$" is a numeric value which specifies the number of characters in the field. The value of "$n$" should be the same as the field length (Input Length) specified in the Form Design.

"!" requires the Field Parameter Type (N) and inserts leading zeros. For example, LENGTH(3!) would print a field value of "6" as "006".

**Example:**

LENGTH(6)

---

**LTRIM**

Use this Form Parameter if leading spaces in the Data file are to be printed on the checks.

**Options:** Y/N

**Default:** Y

**Requirement:** Application includes a data file

**Description:**

This Field Parameter indicates whether or not leading spaces in the data file should be included on checks. "Y" indicates that all leading spaces SHOULD NOT be printed, "N" indicates that all leading spaces SHOULD be printed.

To control the printing of leading spaces for individual fields rather than all fields on the whole form, the LRECL parameter can be used as a Field Parameter. See LTRIM Field Parameter.

**Example:**

LTRIM(N)
**MICRTYPE**

Use this Field Parameter in the rare instances when the Field Parameter AUTOMICR cannot be used. MICRTYPE manually places a field on the MICR line.

**Options:** INTRANSIT, ONUS, EXTRA, CHECKNUMBER

**Default:** None

**Requirement:** Form Design that includes MICR Line fields

**Description:**

This Field Parameter identifies the field as a manual MICR field, and it should be used only if the Field Parameter AUTOMICR cannot be used. "x" is one of the following values:

- **INTRANSIT** • MICRTYPE(INTRANSIT) places the value of the field on the MICR Line in the intransit position.
- **ONUS** • MICRTYPE(ONUS) places the value of the field on the MICR Line in the onus position.
- **EXTRA** • MICRTYPE(EXTRA) places the value of the field on the MICR Line in the extra onus position.
- **CHECKNUMBER** • MICRTYPE(CHECKNUMBER) places the value of the field on the MICR Line in the auxiliary onus position.

**Example:**

MICR(ONUS)

**MINLENGTH**

MinLength- allow users the ability to specify a minimum data length to allow passed in a given field.

1 argument - numeric minimum length of the data in question.

Format: MinLength(n)

Example: MinLength(6)
CHAPTER 2: FIELD PARAMETERS

Field Parameters

MODULUS

Use this Field Parameter to perform a modulus 9 calculation on Canadian payments and more.

**Abbreviation:** MOD

**Options:** x = 9

**Default:** None

**Requirement:** None

**Description:**

This feature will provide support for a Modulus 9 calculation to be performed on the payment number. The algorithm being used is consistent with the algorithm used in custom programs where Modulus 9 payment numbers are required specifically for Canadian payments.

MODULUS has a fixed value of 9. In the future, PayBase may support more modulus calculations which would add to the valid values for the field parameter.

The Modulus routine will use a straight Mod calculation. The Modulus calculation will be performed when the payment data is loaded and therefore will be used throughout processing (Check, CFA or ACH). The Modulus algorithm is protected information.

The CFA Manual Payments screen will include a check box for adding the Modulus 9 calculation to the check number entered.

Example:

MODULUS(9)

ORIENTATION

Use this Field Parameter when vertical or reversed printing is needed for the field.

**Options:** H/V/R, where H = horizontally, V= vertically, R = horizontally reversed

**Default:** H

**Requirement:** ORIENTATION(V) requires a font that has been rotated 270 degrees and the TYPE(U) or TYPE(N). ORIENTATION(R) requires a reverse font.

**Description:**

This parameter specifies whether the field is to be printed horizontally, vertically, or horizontally reversed.
"H" prints the field value normally, from left to right.
"V" prints the field vertically, from top to bottom.
"R" prints the field reversed, from right to left.

For Vertical Printing:
A vertical field should only be printed with fonts that have been rotated 270 degrees. In addition, the font must NOT have the right curly brace ({ } X'7D') defined. Data for vertical fields may NOT contain lower-case data. Use the Field Parameters TYPE(U) that forces all data to be in upper-case letters, or TYPE(N) that indicates that the field is a numeric field.

For Reverse Printing:
A reversed field can ONLY be printed with a reverse font.

Example:
ORIENTATION(V)

**POSTFIX**
Use this Field Parameter to print data to the immediate right of the field value.

**Options:** x = any alphanumeric value

**Default:** None

**Requirement:** None

**Description:**
"x" is an alphanumeric value that will be printed to the immediate right of the field value when the field is printed. For example, if the field value is "123.45" and POSTFIX(**) is listed, the field will be printed as "123.45**.

**Example:**
POSTFIX(DOLLARS)

**PPLX**
Use this field parameter to send the field value to the Check Fraud Avoidance (CFA) database when the check is printed. This parameter is only valid if CFA or CFAX is installed.

**Options:** PAYEE, DATE, AMOUNT, CHECKNUMBER, 1-15

**Default:** None.

**Requirement:** CFA Module
**Description:**

The PPLX parameter must be included in the following four fields because CFA must record the following information:

- Payee
- Date
- Check amount
- Check number

CFA may also record five additional fields defined by the user. "x" is one of the following values:

**PAYEE**

PPLX(PAYEE) sends the field value to the Payee column in the CFA log. A required field parameter.

**DATE**

PPLX(DATE) sends the field value to the Date column in the CFA log. A required field parameter.

**AMOUNT**

PPLX(AMOUNT) sends the field value to the Amount column in the CFA log. A required field parameter.

**CHECKNUMBER**

PPLX(CHECKNUMBER) sends the field value to the Payment Number (or check number) column in the CFA log. A required field parameter.

**1-15**

PPLX(1), PPLX(2), PPLX(3), PPLX(4), PPLX(5), PPLX(6), PPLX(7), PPLX(8), PPLX(9), PPLX(10), PPLX(11), PPLX(12), PPLX(13), PPLX(14), PPLX(15). Each of these field parameters sends the field value to a user defined column in the CFA log. These are optional field parameters.

**Example:**

PPLX(AMOUNT)

PPLX(3)

**PREFIX**

Use this Field Parameter to print data to the immediate left of the field value.

**Options:** x = any alphanumeric value
CHAPTER 2: FIELD PARAMETERS

Field Parameters

Default: None
Requirement: None
Description:
"x" is an alphanumeric value that will be printed to the immediate left of the field value when the field is printed. For example, if the field value is "123.45" and PREFIX(**) is listed, the field will be printed as "**123.45.

Example:
PREFIX($)

REFERENCE

Use this Field Parameter to obtain field data from a previous data field.

Options: see below
Default: None

Requirement: If REFERENCE(?AUTOMICR), the Field Parameters AUTOMICR(CHECKNUMBER), AUTOMICR(ONUS), and AUTOMICR(INTRANSIT) are required. A referenced field MUST be listed in the Link File BEFORE the field that is referring to it.

Description:
The REFERENCE parameter is used to obtain data from a source other than the primary data source, (the Data file or the MDE Design view), for the Link File.

"x" is an alphanumeric value (which usually identifies a Link File field name) which the current field will refer to in order to obtain its field value. For example, if the current field is FIELD(STUB_CHECKNO) and it includes the parameter REFERENCE(CHECKNO), PayBase will print the field value of FIELD(CHECKNO) as the field value of FIELD(STUB_CHECKNO).

"+x...+x" is a list of fields; the values of these fields can be added together (concatenated). All leading and trailing spaces are removed.

""s"" is an alphanumeric value, often a blank, which can be inserted between the concatenated fields. Note: this option does include one set of quotation marks when listed in the Field Parameter, as shown in the JOHN SMITH example below.

"?" indicates that the value “x” is NOT a field name. If? is included, x must be one of the following values: CHECKNUMBER, AUTOMICR, or n, as described below:
CHECKNUMBER
REFERENCE(?CHECKNUMBER) uses the current value of the automatic checknumber field in the Group file as a starting check number. If a Group file does not exist, or the group key cannot be found, or the CHECKNUMBER field is blank, PayBase will prompt you for the starting check number.

AUTOMICR
REFERENCE(?AUTOMICR) uses the values of the fields containing the Field Parameter AUTOMICR when printing the MICR Line.

?n
REFERENCE(?n) requires the Field Parameter ACCUMULATOR(n). "n" is a number between 1 and 9, and indicates any one of nine possible calculated values. REFERENCE(?n) is the total of all ACCUMULATOR(n) field values. For example, the value of a field which includes REFERENCE(?3) would be the total value of all of the fields which include the Field Parameter ACCUMULATOR(3). REFERENCE(?n) means exactly the same thing as the STARTINGVALUE(?n) Field Parameter.

Examples:
REFERENCE(?1)
    r(CITY+" "+STATE+" "+ZIPCODE)

Detailed Example:
The value of FIELD(FIRSTNAME) is “JOHN” and the value of FIELD(LASTNAME) is “SMITH”. The value of a field which includes the parameter REFERENCE(FIRSTNAME+LASTNAME) would be “JOHNSMITH”. The value of a field which includes the parameter REFERENCE(FIRSTNAME+" "+LASTNAME) would be “JOHN SMITH”.

REMOVECHARS
Use this Field Parameter to remove character(s) from a selected field. This is typically used when spaces between words need to be removed for security purposes.

Options: One or more characters
Default: None
Requirement: None
Description:
Available by right-clicking on a Field Branch and selecting Add/Change Field Parameters using REMOVECHARS. Type the replacement character(s).

**Example:**

REMOVECHARS("*"

---

**REPLACE**

Replace - allow users the ability to replace a string of characters from an incoming piece of data with another string of characters (1 or more characters).

3 arguments - string to be removed, string to be written in its stead, flag indicating case-sensitivity (defaulting to be Case-Sensitive).

**Format:** Replace(x,y,{CASE|NOCASE})

**Examples:**

- Replace('a','b') will replace all 'a' characters with 'b'
- Replace('a','b',CASE) will replace all 'a' characters with 'b'
- Replace('a','b',NOCASE) will replace all 'a' and 'A' characters with 'b'
- Replace('paybase','PayBase') will replace all instances of 'paybase' with 'PayBase'

---

**ROTATEFIELD**

Use this Field Parameter to select a counter-clockwise rotation angle of a design field.

**Options:** 90, 180 and 270 degrees counter-clockwise rotation

**Default:** None

**Requirement:** None

**Description:**

Available by right-clicking on a Field Branch and selecting Add/Change Field Parameters using ROTATEFIELD and also by selecting Rotate Field. Selection of rotation degrees will place entries into the selected Field as shown below.

**Example:**

PREFIX(&a180P) ***PREFIX rotates field 180 degrees***
POSTFIX(&aOP) ***POSTFIX rotates field back to previous setting.***

**SPLIT**

Use this Field Parameter to create a split using three components, Section Request, Section Length and Pad Character. This is typically used to fill empty space in a field with characters for security purposes. Section Request will reserve the number of lines in a field, section length will reserve the size of the field and pad character will fill the empty space with the entered character.

- **Options:** Section Request 1 - 100, Section Length 1 - 300, Pad Character optional
- **Default:** None
- **Description:**
  Available by right-clicking on a Field Branch and selecting Add/Change Field Parameters using SPLIT.
- **Example:**
  SPLIT(3,54,3)

**STARTINGVALUE**

Use this Field Parameter to list the data value for the field. This is often used for a signature or logo field to list the macro number of the graphic file.

- **Options:** x = up to 50 alphanumeric characters, see ? below
- **Default:** None
- **Requirement:** If option is ? CHECKNUMBER, CHECKNUMBER field. If option is ?n, ACCUMULATOR(n) Field Parameters.
- **Description:**
  "x" is an alphanumeric value for the field that will be constant for every document printed. The value of "x" can be uppercase or lowercase, and the maximum length is 50 characters.

A field with a STARTINGVALUE parameter that also includes the Field Parameters HORIZONTALPOSITION and VERTICALPOSITION uses STARTINGVALUE as a default value. When the MDE Design is used, unless the field is created as a Non-visible field, the STARTINGVALUE value is displayed for the field on the screen. The user may accept the STARTINGVALUE value or enter a new value.
"?" defines "x" as a special value. When "x" is preceded by a "?", the value of "x" is referenced from another source. The value of "?x" may be one of the following:

CHECKNUMBER

STARTINGVALUE(?CHECKNUMBER) prints the current check number and requires that the check number field have the name CHECKNUMBER.

?n

STARTINGVALUE(?n) requires the Field Parameter ACCUMULATOR(n). "n" is a number between 1 and 9, and indicates any one of nine possible calculated values. STARTINGVALUE(?n) is the total of all ACCUMULATOR(n) field values. For example, the value of a field which includes STARTINGVALUE(?3) would be the total value of all of the fields which include the Field Parameter ACCUMULATOR(3). STARTINGVALUE(?n) means exactly the same thing as the REFERENCE(?n) Field Parameter.

Example:

STARTINGVALUE(?CHECKNUMBER)

TRACKING

Use this parameter to administer a tracking key to track the starting audit tracking number. The audit tracking number appears on the remittance advice for use by a payer when referencing a cheque.

Options: Any alphanumeric character combination

Default: Current setting

Description: Available by right-clicking a Field Branch and selecting Add/Change Field Parameters using TRACKING.

Example:

TRACKING(45TTf44)

TYPE

Use this field parameter to configure the type of data allowed in a field, for example, whether it is a date field, a numeric field, a graphic field, etc.

Abbreviation - T

See explanations below.

**Default:** \( x = A \)

**Requirement:** None

**Description:**

"\( x \)" is one of the following values:

A

**TYPE(A)** specifies an alphanumeric field. Any upper-case or lower-case keyboard characters may be used.

B, 3of9, \([c]\)

**TYPE(B, 3of9,\([c]\))** specifies a BARCODE field. The value of "\( c \)" is an optional barcode check number. The only available options are **TYPE(B,3of9)** for the BARCODE or **TYPE(B,3of9,\([c]\))** for the BARCODE plus a sum of the barcode displayed as a check digit code after the BARCODE.

D

**TYPE(D)** specifies a date field. The input format MUST be mm/dd/yy or mm/dd/yy. **TYPE(D)** is required if a date EDITMASK parameter is used.

DD

**TYPE(DD)** specifies a decremental date field. Each check will be printed with 1 subtracted from the month value.

DI

**TYPE(DI)** specifies an incremental date field. Each check will be printed with 1 added to the month value.

G

**TYPE(G)** specifies a graphics field if the graphic images is prepared by Bottomline Technologies, Inc. The value of the field is the macro number of a graphics file prepared by Bottomline Technologies, Inc.

I

**TYPE(I)** specifies an incremental field. The value of an incremental field is increased by one for each form. The incrementing occurs immediately after the field has been printed. The initial value for an incremental field is usually defined by including a STARTINGVALUE field parameter.

"\( n \)" is a reset value which may be specified. When the value of the incremental field reaches "\( n \)", the field value is reset to its original value. For example, if the original incremental field value is 1, **TYPE(14)** would reset the field to 1 after every fourth document is printed.
CHAPTER 2: FIELD PARAMETERS

Note

A field which uses a REFERENCE field parameter to obtain data from an incremental field will have a value which is 1 greater than the incremental field.

N

TYPE(N) specifies a numeric field. Only numeric characters and one decimal point may be entered. TYPE(N) is required if a numeric EDITMASK parameter is used.

P

TYPE(P) specifies a PCX graphics field. The field data MUST be the name of a .PCX graphics file to be printed. Note that printing PCX graphic may take longer than printing graphic images prepared by Bottomline Technologies, Inc.

U

TYPE(U) is for Manual Data Entry Link files only, and it specifies an upper-case alphanumeric field. The field data may contain any upper-case or lower-case characters, but all data will be converted to upper-case characters.

V

TYPE(V) specifies a verbiage field. The field data MUST be numeric, and PayBase converts it into an alphanumeric phrase. For example, when used as a parameter for a legal line on a check, “6.24” would be converted to “Six and 24/100 Dollars”.

Z

TYPE(Z) specifies a POSTNET barcode field. The field data MUST contain a ZIPcode. If the field contains MORE than just a ZIPcode, (for example, City and State), any leading non-numeric characters will be ignored. This is only available if POSTNET XTended has been purchased.

Example:

TYPE(I)

VALIDATE

Use this parameter to require a field to be of a certain type. Only the format specified will be accepted when characters are entered into a field.

Options: Alpha, Alphanumeric, Date, Numeric, Required or Optional

Default: Current setting

Requirement: None
CHAPTER 2: FIELD PARAMETERS

Field Parameters

Description:
Available by right-clicking on a Field Branch and selecting Add/Change Field Parameters using VALIDATE. The first option is the field type and the second is the requirement.

Example:
VALIDATE(Alpha, Optional)

BACS Addition
This field parameter has been modified to allow the validation of BACS data criteria.

Adding 'BACS1' as a new 1st argument, where 'BACS1' means the data will contain at least 6 alphanumeric characters (i.e. A-Z, 0-9). Other allowed characters may be present (i.e. '&', '/', '.', '-' and blank space) but will not be included in the count of 6 alphanumeric. After stripping out non-alphanumeric characters, this data must not contain a string of all the same alphanumeric characters e.g. all zeros or all 'A's.. The following is a recap of all allowable characters:

i. A-Z (as specified for upper case alphabet)
ii. zero to 9
iii. blank space
iv. ampersand (&)
v. hyphen (-)
vi. full stop (.)
vii. solidus (/)

Adding 'Amount' as a new 1st argument, where 'Amount' is one of:

i. zero to 9
ii. full stop(.)

VERTICALPOSITION
Use this parameter to vertically position a field.

Options: any six alpha numeric characters
Default: None
Requirement: None
Description:
Any numeric value greater than 0
Example:
VERTICALPOSITION(234234)

**XADJUST**

Use this parameter to change the horizontal placement of a data field without making changes to the Form Design.

- **Options:** n = a numeric value
- **Default:** n = 0
- **Requirement:** None
- **Description:**
  "n" is a numeric value which moves the field n/300 inches horizontally. When "n" is a positive value, the field is moved to the right on the page, when "n" is a negative value, the field is moved to the left on the page.

  **Example:**
  XADJUST(10)

**YADJUST**

Use this parameter to change the vertical placement of a data field without making changes to the Form Design.

- **Options:** n = a numeric value
- **Default:** n = 0
- **Requirement:** None
- **Description:**
  "n" is a numeric value which moves the field n/300 inches vertically. When "n" is a positive value, the field is moved toward the bottom of the page, when "n" is a negative value, the field is moved toward the top of the page.

  **Example:**
  YADJUST(100)

**DPI Reference**

Inches to DPI Quick Reference
The figure below is to assist you with XADJUST and YADJUST parameters as well as AMVA and AMHA. All adjustments are based on 300 dots per inch (dpi). Select the Print button from the menu above to print this reference sheet for easy access.

The chart below will assist you when using the PayBase form parameters AMHA and AMVA. It can also be used with the field parameters XADJUST and YADJUST. Keep in mind that when you specify a negative number, your image will be moved either up or to the left. If you specify a positive number, your image will be moved either down or to the right. Please review the examples listed below:

AMVA(75) - will move the MICR line down 1/4”
AMHA(-38) - will move the MICR line 1/8” to the left

XADJUST(356) - will move the data/image 1 3/6” to the left
YADJUST(-188) - will move the data/image up 5/8”

<table>
<thead>
<tr>
<th>Inch</th>
<th>1/16</th>
<th>1/8</th>
<th>3/16</th>
<th>1/4</th>
<th>5/16</th>
<th>3/8</th>
<th>7/16</th>
<th>1/2</th>
<th>9/16</th>
<th>5/8</th>
<th>11/16</th>
<th>3/4</th>
<th>13/16</th>
<th>7/8</th>
<th>15/16</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPI</td>
<td>19</td>
<td>38</td>
<td>56</td>
<td>75</td>
<td>94</td>
<td>113</td>
<td>131</td>
<td>150</td>
<td>169</td>
<td>188</td>
<td>206</td>
<td>225</td>
<td>244</td>
<td>263</td>
<td>281</td>
<td>300</td>
</tr>
</tbody>
</table>
LOGIC BLOCKS

In this chapter we will explore the following topic:

• LOGIC_BLOCKS
• IF Logic Using LOGIC_BLOCK
• LOGIC_BLOCK Structure
• Using LOGIC_BLOCKS in an Application
PayBase, in conjunction with Designer+, is now equipped with more robust IF Logic. In addition to the current IF Logic functionality, you can use ELSE_IF and ELSE key words to construct an unlimited number of conditions. You can also use ALWAYS key words to reference any modified field values. All conditions are contained in LOGIC_BLOCKS explained below.

Note

See the IF logic section for more information.

IF Logic Using LOGIC_BLOCK

The new IF Logic allows multiple condition statements to be grouped together, therefore allowing multiple fields to be set based on those conditions. This new IF Logic is grouped using a LOGIC_BLOCK. The new LOGIC_BLOCK is similar to the Form and Field Parameters that appear in your Application tree view of your .CGL.

Once all field values are assigned and referenced, you can insert your LOGIC_BLOCK. LOGIC_BLOCKS are on the same level as the Field parameters. Like Field parameters, LOGIC_BLOCKS are new units of work and generally will be the last parameter section in the .CGL. Designer+ is equipped to put them at the end of the .CGL.

Notes

• In order to clearly state how the values are being set, it is recommended, that you insert comments for each field that is set in the LOGIC BLOCK.

• Any fields used in the LOGIC BLOCK must exist in the .CGL or you will receive an error.
LOGIC_BLOCK Structure

Each LOGIC_BLOCK will contain conditions and statements that follow it until a new LOGIC_BLOCK or Field parameter is found. LOGIC_BLOCKS can contain the following key words:

- ALWAYS
- IF
- ELSE_IF
- ELSE
- SET

Notes

- If necessary, the layout could allow for additional conditional logic to be added such as a SWITCH key word.
- Multiple SET key words can be used to assign values to previously assigned fields for all sections (ALWAYS, IF, ELSE_IF and ELSE).
- You will receive an error if a field has not been defined.

ALWAYS

The ALWAYS key word is used to set Fields that would currently reference any modified field values. This is necessary because the field level reference occurs before the LOGIC_BLOCK.

The ALWAYS key words can be before or after the logic conditions to perform SET operations. These SET key words occur not matter what course the logic follows.

IF and ELSE_IF

In addition to leveraging their current functionality, IF and ELSE_IF key words have been expanded to allow for multiple groupings of conditions. It will be set up in groups of three (field, operator, compare value) with the ability to group, AND and OR conditions. The conditions would still be defined using Field, Operator and Value. The AND, OR and grouping operators need to be allowed between condition sets.
Notes

- It is not recommended to use Field level condition logic and the LOGIC_BLOCK level condition logic.
- An IF key word must exist before an ELSE_IF or ELSE key word. This is checked when the .CGL is secured.

SET

When building your LOGIC_BLOCK, using your ALWAYS, IF, ELSE_IF and ELSE key words, you will use the SET key word to further define your fields and their values. A SET key word allows the following 2 arguments:

- The field name
- The value of the field

Notes

- The first argument is the actual field to assign the value to.
- The second argument works the same way the REFERENCE argument works now.
- All EDITMASKs for SET fields are reapplied after assigning new values.

Using LOGIC_BLOCKS in an Application

With more flexible and robust IF Logic, LOGIC_BLOCKS are useful in your Applications for printed values that could vary based on other values in your record. For example, if you have information such as different addresses in your Application, you could use a LOGIC_BLOCK to move address lines up if a certain ones are blank.

Field names that reference a Field set in the LOGIC_BLOCK now need to use a SET key word within the LOGIC_BLOCK to perform the same task. The following example shows how the HEADER field, which would normally reference the NAME value, now needs to be set in the LOGIC_BLOCK as well.
LOGIC_BLOCK Example

FIELD(FIRST)
  V(1)
  H(1)
  L(20)

FIELD(LAST)
  V(2)
  H(1)
  L(20)

FIELD(SUFFIX1)
  V(3)
  H(1)
  L(20)

FIELD(SUFFIX2)
  V(4)
  H(1)
  L(20)

FIELD(NAME)

FIELD(ACCOUNT)
  V(5)
  H(1)
  L(20)

FIELD(HEADER)
  *The Reference that would be done here now gets moved to the LOGIC_BLOCK
  *R(NAME+", "+ACCOUNT)

LOGIC_BLOCK (NAMEOFLB)
  IF(SUFFIX1,,", ",&&SUFFIX1,>>, ")
      SET(NAME,FIRST+" "+LAST+" "+SUFFIX1+SUFFIX2)
  ELSE_IF(SUFFIX1,,")
      SET(NAME,FIRST+" "+LAST+" "+SUFFIX2)
  ELSE
      SET(NAME,FIRST+" "+LAST)
  ALWAYS
      SET(HEADER,NAME+", "+ACCOUNT)
FIELD PERSISTENCE

In this chapter we will explore the following topics:

• Field Persistence
• Persistence Saves Time
• How Persistence Works
Field Persistence

Field Persistence allows for the use of global and local variables within your PayBase Application. This makes you Link files more robust in three ways:

1. Persistence enables you to reference data from one form to another.
2. With this information you can perform dynamic functions, such as sub total and totaling functions across forms.
3. In addition, you can reference fields that serve as tables, headers, etc.

Persistence Saves Time

Resolving data across forms, significantly reduces the need for custom PreProcessors to modify input Data files to provide static data for the process cycle.

Field Persistence Example

A logical record that contains a check, remittance and overflow must now be processed by a PreProcessor in order to supply sub totals, grand totals and copies of header data for each of the forms and the Top of Forms, form switch keys etc. Field Persistence dynamically resolves these data items within PayBase, thus eliminating the need for PreProcessors. It also stores these data items for future use in the Application.
How Persistence Works

Field Persistence requires a new Field Parameter and a new Form Parameter. Click on each below for a complete description.

- FIELDREFERENCE
- PERSISTRESET

A new library map is now part of PayBase. This map stores values identified in the .CGL as persisted fields which are later retrieved into receiving fields in another .CGL file.

Depending on the operation, the FIELDREFERENCE field parameter takes two or three arguments; OPERATION, KEY and SCOPE. This field parameter controls all reading and writing of persisted fields. When writing persisted fields all three arguments are used. However, when reading a persisted field only the first two (OPERATION and SCOPE) are used.

Additionally, the form parameter PERSISTRESET controls when the persisted field’s value is reset either beforehand or afterwards. When the process encounters the end of a record, it either resets the persisted fields with LOGICAL_Record scope before it loads and processes the next record, or after it loads and processes the next record.
**Example of Persisted Fields Across Form Boundaries**

The form parameter `PERSISTRESET` indicates that we reset our persisted fields with `LOGICAL_RECORD` scope before processing this form.

The fields that are to be persisted are declared with the `FIELDREFERENCE` parameter using the `WRITE` operator key word, a key to identify the value, and optionally a scope key word.

Note that the fields can be identified by any text string but a suggested approach is to use `CGL` name and field name in dot notation.

The persisted collection is stored in the PayBase Application Object.

The first thing to happen is the `SUB_TOTAL_BRIDGE` field looks up the persisted value from `chkForm1.SUB_TOTAL` then the referenced fields are picked up. Then the column is processed. Then the `SUB_TOTAL` field accumulates the column fields plus the `SUB_TOTAL` field. This value is then written back into the persisted field `chkForm1.SUB_TOTAL`.

On this last page we essentially do what we did on the previous page with the exception that we are going to persist our subtotal in a different persisted field with `DATA_FILE` scope for use as a running total for the entire data file.
INDEX:

A
ACCUMULATOR 29
ALWAYS 63
AUDIT 30
AUDITPRINTER 5
AUDITTYPE 6
AUTOCODEHORIZONTALADJUSTMENT 6
AUTOCODEREFERENCENUMBERLENGTH 7
AUTOCODEVERTICALADJUSTMENT 7
AUTOMICR 32
AUTOMICRACCTNUMBERLOCATION 8
AUTOMICRCHECKNUMBERLENGTH 8
AUTOMICRHORIZONTALADJUSTMENT 9
AUTOMICRONUSLOCATION 10
AUTOMICRVERTICALADJUSTMENT 11

C
CHECKDEPTH 11
CHECKONTHE 12
CHECKTYPE 12
CHECKXADJUST 13
CHECKYADJUST 13
CHEQUEDEPTH 14
CHEQUEONTHE 14
CHEQUETYPE 15
CHEQUEXADJUST 15
CHEQUEYADJUST 16
CODEFONT 16
CODEFONTSWITCH 17
CODELAYOUT 17
CURRENCY 18

D
Date Edit Masks 34
DECIMALPOINT 33
DPI Reference 59

E
EDITMASK 34
Example of Persisted Fields Across Form Boundries 70

F
Field Parameters 28
ACCUMULATOR 29
AUDIT 30
AUTOMICR 32
DECIMALPOINT 33
EDITMASK 34
FORMATCASE 38
HORIZONTALPOSITION 38
IF 39
INPUTMASK 44
JUSTIFY 44
JUSTIFYLENGTH 45
LENGTH 46
LTRIM 46
MICRTYPE 47
MINLENGTH 47
MODULUS 48
ORIENTATION 48
POSTFIX 49
PPLX 49
PREFIX 50
REFERENCE 51
REMOVCHEARS 52
REPLACE 53
ROTATEFIELD 53
SPLIT 54
STARTINGVALUE 54
TRACKING 55
TYPE 55
VALIDATE 57
VERTICALPOSITION 58
XADJUST 59
YADJUST 59
Field Persistence 68
Field Persistence Example 68, 70
FIELDFILE 18
Form Parameters 4
  AUDITPRINTER 5
  AUDITTYPE 6
  AUDITYPE 6
  AUTOCODEHORIZONTALADJUSTMENT 6
  AUTOCODEREFERENCECENUMBERLENGTH 7
  AUTOCODEVERTICALADJUSTMENT 7
  AUTOMICRACCTNUMBERLOCATION 8
  AUTOMICRCHECKNUMBERLENGTH 8
  AUTOMICRHORIZONTALADJUSTMENT 9
  AUTOMICRONUSLOCATION 10
  AUTOMICRVERTICALADJUSTMENT 11
  CHECKDEPTH 11
  CHECKONTHE 12
  CHECKTYPE 12
  CHECKXADJUST 13
  CHECKYADJUST 13
  CHEQUEDEPTH 14
  CHEQUEONTHE 14
  CHEQUETYPE 15
  CHEQUEXADJUST 15
  CHEQUEYADJUST 16
  CODEFONT 16
  CODEFONTSWITCH 17
  CODELAYOUT 17
  CURRENCY 18
  FIELDFILE 18
  FORMFIELDNAME 19
FORMSPERPAGE 19
FORMTYPE 19
GROUPFIELDNAME 20
INTERCHECKADJUSTMENT 20
INTERCHEQUEADJUSTMENT 21
LTRIM 21
MICRFONT 22
OUTTRAY 22
PAPERDEPTH 23
PAYMENTMETHOD 23
PRINTERFIELDNAME 24
REPEAT 25
SORT 25
TRAY 25
FORMATCASE 38
FORMFIELDNAME 19
FORMSPERPAGE 19
FORMTYPE 19

G
GROUPFIELDNAME 20

H
HORIZONTALPOSITION 38
How Persistence Works 69

I
IF and ELSE_IF 63
IF Logic Using LOGIC_BLOCK 62
Input Length 46
INPUTMASK 44
INTERCHECKADJUSTMENT 20
INTERCHEQUEADJUSTMENT 21

J
JUSTIFY 44
JUSTIFYLENGTH 45

L
LENGTH 46
LOGIC_BLOCK Example 65
LOGIC_BLOCK Structure 63
LOGIC_BLOCKS 62
LTRIM 21, 46

M
MICRFONT 22
MICRTYPE 47
MINLENGTH 47
MODULUS 48

N
Numeric Edit Masks 35

O
ORIENTATION 48
Output Length 45
OUTTRAY 22

P
PAPERDEPTH 23
PAYMENTMETHOD 23
Persistence Saves Time 68
POSTFIX 49
PPLX 49
PREFIX 50
PRINTERFIELDNAME 24

R
REFERENCE 51
REMOVECHARS 52
REPEAT 25
REPLACE 53
ROTATEFIELD 53

S
SET 64
SPLIT 54
STARTINGVALUE 54

T
TRACKING 55
TRAY 25
TYPE 55

U
Using LOGIC_BLOCKS in an Application 64

V
VALIDATE 57
VERTICALPOSITION 58

X
XADJUST 59

Y
YADJUST 59