DOS versus Windows printing

Enabling DOS applications to print to Windows printers isn't that straightforward:

- In the DOS days, printers were connected to a parallel (LPT) or serial (COM) port. Printers now connect to an USB port, a (wireless) network, or are even virtual (like PDF printers). DOS applications can only print to a parallel or serial port, while modern PC's don't have such ports anymore.
- Former DOS printers expected to receive text. An application adds printer control codes to end a line (line feed), eventually eject a page (form feed), or for instance print in bold (printer model specific control codes). Windows GDI printers often don't support text input and are not line orientated: They expect formatted, complete pages.
- DOS text is ASCII based: Only 128 (minus 32 control codes) characters are defined, basically those found on an US International keyboard. The original ASCII character set was extended to include an additional 128 characters. Due to the limitations (256-32 characters) of ASCII, a language specific DOS code page has to define what those additional 128 characters translate to in Windows text (Unicode with thousands of characters).

So some work is required to facilitate a DOS application to print to a Windows printer, mainly:

- The stream of text and printer control codes sent to a printer port has to be collected, at some moment interpreted and composed as Windows GDI pages.
- Those pages then have to be directed to a Windows printer (mostly not parallel/serial).

When a DOS application prints to a LPTx/COMx port, at finalizing the print job vDos will:

- Save the output to a #LPTx/COMx.asc file. Of no use to vDos itself, but to optional use an alternative external Windows program to convert and print that ASCII text.
- Convert the output to Windows Unicode text, and save it to a #LPTx/COMx.txt file. That file can be read into a Windows application, for instance to import (large amounts of) DOS data.
- By default convert the collected ASCII text and printer control codes to a format accepted by Windows printers.

Printing in vDos

If you don't tell vDos (in config.txt) what to do with data sent to a printer (port), it lets you select the Windows printer to print to. If the printer output is plain text, with line/form feeds, eventual printer codes inserted by the DOS application as if it was printing to an Epson, IBM Proprinter or PCL compatible printer, this should mostly work just fine.

You may however want to adjust the default settings how this data is handled by adding printer directives to config.txt.

LPTx/COMx = options (x = 1-9, though many DOS programs will at most support LPT1-3 and COM1-4).
Left to the equal sign defines what the application expects to print to. The options to the right what vDos is considered to do with that.

PRIVATE
If used, it has to be the very first option. The #LPTx/COMx.asc and .txt files won't be created. Saving some disk/network activity, or you don't want the (latest) printout visible to others with access to those files.

SEL: "Windows printer name"
Explicitly sets the printer to be used. SEL: "" will preselect Windows default printer.
Default: Display the Windows printer selection dialog.
Note, while this dialog is shown, the DOS application will continue to run. Eventual further printer output will be appended, but the .asc/.txt files won't be updated.

TIO: <seconds> | files | key | spool
Printer output is collected as print jobs (Windows requires this concept). By default considered to be finished as the application doesn't output further data for some time (timeouts). If the last print command was a form feed (eject page), wait for 1, else 5 seconds. If you get broken pages, or want the printer to respond quicker, you can use this option. Though you have to experiment what works (best or even at all) with your application.

- **TIO: <seconds>**. Specifies how many seconds (1 to 99) vDos will wait. Or disable the timeout functionality by TIO:0. DOS applications can address a printer in various ways. TIO:0 will only work for the 'nice', less obsolete method: Open, output data to, and finally close a DOS LPT/COM device. Mind, some applications don't do it this way: Nothing could get printed at all, or only as the application is ended.
- **TIO: files**. A number of files are opened as the application starts printing. Finishing, it is supposed to close one or more of those files. Triggering vDos to print the collected data.
- **TIO: key**. The application waits for a keypress when done printing. It could however first only check for (then get/wait for) a key being pressed, so you would then actually have to press one.
- **TIO: spool**. Print when Ctrl+S is pressed. The title bar of the vDos window will show spooling is in progress. A last resort to solve persistent timeout issues, or to accumulate DOS print jobs.

RAW
This option requires the printer to support DOS text input (PCL/PostScript printers do). The ASCII text is unmodified passed on to the printer. For instance to print to an actual PCL compatible printer if such a (mostly HP named) printer is set in your DOS application. Mind, the next options don't apply if RAW is used.

FONT: "Windows installed font"
Print text in this font, has to be a monospaced one: All characters having the same width.
Default: The built-in vDos font, as for displaying text.

NOLINES
Sequences of three or more "-" or "=" characters are replaced by line drawing characters. For instance "----" will be printed as "——". With NOLINES the original characters are printed.

HORZ: left [,scale]
left: The left margin in mm's (1 inch = 25.4mm).
scale: The percentage the width of the printout will be scaled to (50-200%).
Default: 10,95.

VERT: top [,scale [,bottom]]
top: The top margin in mm's.
scale: The percentage the height of the printout is scaled to (50-200%).
bottom: The bottom margin in mm's, a page is ejected at this position.
Default: 15,100,5.

Note: PCL supports margin setting. If used, those override the eventual left and top margins set here.

Example: LPT1 = font:"Courier New" tio:spool horz:20,90
Postpones printing (tio:spool), display a dialog to select the Windows printer (no SEL:), print in the Courier New font, lines start at 20mm from the left, shrink the width to 90%.

Other print options

DUMMY
Don't print, you just want to ignore the output to the printer. Or use an alternative external DOS-to-Windows print program, monitoring the vDos directory for new .asc files to print.
Example: LPT3 = DUMMY

CLIP
Don't print, copy the translated Unicode text to the Windows Clipboard (to paste into another program).
Example: COM4 = CLIP

"Windows command/program" [WAIT][HIDE] [program options] #LPTx/COMx.asc/txt
Open the .asc (ASCII) or .txt (Unicode) file by the specified Windows command/program.
Example: LPT2 = "%windir%\system32\notepad.exe" HIDE /p #lpt2.txt
(Let Notepad print (/p) the #lpt2.txt file, while its window is minimized (HIDE)).
Note: Eventual file paths here are those of Windows, not vDos/DOS.

"Windows device or file": ["Error caption" "Error message"]
Essentially not meant for printing: Communicate with the specified device/file interactively (the data is not collected as a print job). The colon at the end is mandatory.
Basic support for serial devices, such port has to be setup correctly before vDos is started. In the device manager, or by MODE COM… at the Windows(!) command prompt.
Example: COM1 = "COM1":

vDos will attempt to open the device or file exclusively, so this option can also be used for a simple conditional start of the DOS application.
Example: COM7 = "once.txt": "Attention" "Program X is already running"
once.txt is some existing file in the vDos directory.
Example: COM8 = "\\server\share\readonly\%username%": "Notice" "You are not allowed to start program X"
For legit users a file with that name is provided.

DOS application prints to PCL printer

If your DOS application can be set to print to a PCL printer: Most laser printers still support that text input (RAW option). If you don't have an actual PCL printer and the vDos print processor is too basic: vDos detects a PCL printer is addressed (the print data will contain PCL specific control sequences). If provided in the vDos directory, the PCL6 program is then started to convert the saved .asc file to a PDF document. That is then opened in the default Windows PDF viewer. PCL6 (GhostPCL) can be downloaded and is documented at: www.columbia.edu/~em36/ghostpcl.html.

Use a third-party program to print

The built-in support of Epson, IBM Proprinter and PCL printers especially lacks (bitmap) graphics/images. You could also be unhappy with the end result on paper, possibly with an application doing extensive formatting. Since vDos saves the ASCII output to an .asc file, you can use an external DOS-to-Windows print program to do the printing instead. Three of such programs, among others available online:

- **DOSPRN**
  Shareware, $14.95 for a single license, price drops with more licenses.
  DOSPRN's author added options to integrate it better with vDos. Besides Epson, DOSPRN also supports IBM Proprinter and limited PCL printing. You can also define alternative sets of printer codes.
  Example: LPT1 = "%ProgramFiles(x86)%\DOSPRN\DOSprn.exe" DOSPRN options #lpt1.asc
  Note: Omit (x86) with Windows 32-bit.
- **DOSPrinter** (Epson-only, feature rich, once the print processor used by vDos).
  Shareware, $40.00 for a single user license.
  Example: LPT2 = "DOSPrinter.exe" WAIT DOSPrinter options #lpt2.asc
  (Assuming the DOSPrinter program is located in the vDos directory).
- **WinPrint** (Epson-only)
  Freeware.
  WinPrint installs to the Windows system tray, monitoring a directory for new files to be printed.
  Example: LPT3 = dummy
  (vDos should only create the #lpt3.asc/.txt files, the first to be detected, read and printed by WinPrint).

In general, an external print program monitoring a directory for new files to print is a roundabout. It will have a timeout setting, how long to wait for no more data being added to the detected file. Set this value as low as possible. vDos creates the .asc file in one instant sweep, an actual timeout will only delay printing.

CopyClip - Copy to Clipboard enhancement

COPYCLIP = LPTx/COMx ["Copy" "Copy/Proceed" "Append" "Append/Proceed"]

With this directive, selecting text and releasing the mouse button, vDos will display a selection box of the defined actions (default descriptions shown above). An action can be omitted by using "".

Copy: Copy the selected text to Windows Clipboard.
Append: Add the selected text to that already gathered in vDos, then copy all to Windows Clipboard.
Proceed: Handle the gathered text as if it would be printed to LPTx/COMx, and set the Windows VDOS environment variable to the first/top line.

Example: COPYCLIP = LPT7 "Copy text" "" "Add text" ""
(With LPT7=DUMMY, just get the extra Append option to collect text).
Example: COPYCLIP = COM6 "" "Edit" "" ""
(With COM6="%windir%\system32\notepad.exe" #com6.txt, start Notepad with the selected text. No selection box is shown since only one action is defined).
Example: COPYCLIP = LPT5 "" "" "" ""
(Disable copy to Clipboard, no action defined).

CopyClip also enables a double click action. If used at a non-space, the characters to the left and right will be selected up to the first space. If the selected text is more than two characters, Copy/Proceed will be executed (even if it is disabled by "").