

Introduction

Printing a signature on a check or form is a simple process that begins with an image file that contains the signature. The first step is to get the signature itself into an image format, preferably a .BMP file. If you are creating the image, then have the signor write the signature using a pen with solid black ink. Scan the image and save it as a grayscale .BMP, with a minimum resolution of 300 dpi. A grayscale image can potentially give a “cleaner edge” to the printed signature than a straight black and white scan. Then use **MKPCL** to generate a PCL file from the image file, and later your application will use the PCL file to place the signature on the printed page. For **MKPCL** details, go to <http://www.spectracolorservices.com>.

It is assumed that the printing process is controlled entirely by the application that is printing the check or form, and it is not dependent on any kind of a printer driver to image the page. Further, the assumption is made that there is single check preprinted on a letter size sheet, perhaps with perforations dividing the sheet into three parts.

Often, however, you may not be in control of the making of the original signature. I was involved in a project a while back that entailed getting the signature of the appropriate county official for inclusion on birth certificates that were being standardized statewide. Some counties sent good looking image files and others just the signature on a scrap of paper that needed scanning. It was difficult to get a new (better) image from the county. Happily, **MKPCL** controls can make a substandard image look presentable. Consider...



When reduced to grayscale we would get something like the following...



These imperfections can be overcome by applying appropriate values to the highlight and shadow threshold flags (**-H ###**, **-S ###**) as **MKPCL** program options. Any of the three flawed originals can be made to match the “ideal” when **MKPCL** generates the PCL output. At this point we will assume we have a suitable image for printing on our output.

Creating the signature PCL

There are a number of attributes that can be embedded in the PCL file through the use of **MKPCL** command line flags. Refer to the **MKPCL** user manual for a more detailed explanation of the flags. For a baseline reference we will begin with a minimum **MKPCL** command.

mkpcl -c 3 signature.bmp signature.pcl

This will create a basic PCL file, with **-c 3** compression, of the signature, but our ultimate printing task is much easier to implement with some modifications. Always compress the PCL file as the file size savings are huge, maybe up to 80% or more.

PCL Macro

First off, we want the PCL file to be declared as a printer macro. This requires nothing more than adding “-m #####” to the MKPCL command line, where ##### is a unique macro ID number. I believe that values of 0-32767 are valid. The macro ID is then used later to “call” the macro to actually print the signature. In addition, we want to add the -p flag, without argument, to declare the macro as permanent. A permanent PCL macro will survive a printer “soft reset” so it could potentially be sent to the printer as a separate operation prior to the check printing job.

Our MKPCL command has now grown to ...

```
mkpcl -c 3 -m 1234 -p signature.bmp signature.pcl
```

Understand that the abilities of your underlying application to permit the timely insertion of data into the print stream will largely dictate the requirements of the PCL file.

Locating the Signature

Of course we want to position the signature at a very specific position on the PCL page. The PCL coordinate system has very precise addressing capabilities, by default 300 dpi on both the **x** and **y** axes. The origin **0,0** is at the intersection of the top and left margins. Through a process of trial and error, establish the necessary **x** and **y** values and we can add these values to the command line with the -x ##### and -y ##### flags. If you have downloaded one of the MKPCL demo packages, there is a file named “ruler.pcl” that you can print to assist in defining the desired coordinates.

Let’s assume that it works out to “-x 1555” and “-y 1755”. In fact those are the values we use on our checks to print the signature on the check in the center position on the sheet. It’s a place to start. For checks on top use “-y 705” and for checks on the bottom try “-y 2805”, leaving “-x 1555” unchanged. Now we have...

```
mkpcl -c 3 -m 1234 -p -x 1555 -y 1755 signature.bmp signature.pcl
```

The address specified defines the location of the top left of the signature image. We are almost done, but there is still a problem. We call the macro from within the application print stream, and through this we relocate the printing position for the signature which leaves the current print position (CAP) at the bottom of signature. All of this is unknown to the application, so we must return the CAP to its previous location. PCL has commands to first save and then later restore the CAP and if you add the -d flag to the command line the save / restore CAP commands will be included in the PCL file where needed. Our final command is then...

```
mkpcl -c 3 -m 1234 -p -x 1555 -y 1755 -d signature.bmp signature.pcl
```

The order of the flags on the MKPCL command line has no effect whatsoever on the output file.

Relative Positioning

On a check and many other business forms, the signature always needs to be printed at the same location on the page. There are occasions, however, where the signature needs to “float” perhaps to follow some random length text. What you can do is specify the -y value to be a relative rather than absolute position to the CAP. You do this simply by putting a + or - sign before the value. So “-y +50” will move the CAP 50 pixels down the page. The application will have to deal with the situation where there isn’t enough room below the text for the signature to print in its entirety.

Running the Operation

The first step is to direct our **signature.pcl** file to the printer to store the macro in printer memory. If your application provides no means of inserting a file into the print stream on demand, then you will have to “print” the file from your OS prompt.

From DOS do a binary copy...

```
copy /b signature.pcl \\servername\printersharename
```

From Unix, a raw print...

```
lp -d printername -o raw signature.pcl
```

This action will download the macro to the printer, but nothing will physically print. The macro will remain resident in printer memory until the printer is power cycled, the macro is specifically deleted, or the macro is overwritten by a new macro with an identical macro ID. If the macro download is performed by the application, have it sent during the initialization so it is only sent to the printer once, regardless of how many checks are to be printed in the current batch.

Calling the Macro

To image the signature you can call the macro on every page with a PCL command in the form **<esc>&f1234y2X** where **1234** is the macro ID associated with the signature in **MKPCL**. The macro will be run immediately upon receipt of the command. Note that **<esc>** represents the single escape character ascii decimal 27 or hex 0x1b common to all PCL commands. Depending on your application, perhaps this PCL code could be contained in a variable on the output format.

Alternatively, you can enable the macro for “automatic overlay” with **<esc>&f1234y4X**. This code only needs to be sent to the printer once at the beginning of the print job, and the printer will automatically run the macro on every subsequent page. The macro is run after the page has been imaged and just before the sheet is drawn though the printer. This code could appear somewhere in the initialization, or you could put it in a variable on the output format. If you choose the latter option, clear the variable after the first page has been printed.

If the macro is enabled for overlay, you needn’t be concerned about the disposition of the CAP after the macro has been run as the page is coming out of the printer immediately. This means that you could eliminate the **-d** flag from the **MKPCL** command line.

Security Concerns

If you are concerned that the signature macro is still resident in the printer once the checks have been printed, then either power cycle the printer or send the command **<esc>&f1234y8X** to delete the macro from the printer. Many applications provide for termination codes.

The other possibility is to have a “flash Memory” card installed in the printer where the signature macro could reside. Simply install the card before printing begins and remove it when you are done and store it in a secure location. An added benefit with the flash memory is that your application doesn’t need to download the macro as it is already in the card.

The drawback to this approach is that you might need to buy a card for multiple printers if there isn't a convenient storage location to share a common card.

Signature Fonts

There are a number of firms that offer to put a signature or logo etc. into a font that you can use in printing a check. While on the surface it might sound like a good approach, the font needs to be first downloaded to the printer and then later selected by the application to print the graphic. So, operationally, it is essentially the same process as making a PCL image from a scan. Once the font has been created, it is not very flexible to change by the user; whereas, it only takes a minute or two to rescan or resize an image with **MKPCL**.

Going Further

Once you are comfortable with overlays and macros, it is not much of a leap to ask yourself why can't we start with totally blank check stock and expand the macro to include the check form itself. You can and it works well.

This is of particular benefit to organizations that need to print checks drawn on multiple bank accounts. You can store macros of multiple signatures and check forms for that matter, each with a unique ID. Bundle them up into a single file for download, and you are always ready to print a check from any of your accounts. Then logic within your application would call up the appropriate overlay for the account at hand.

Check layouts are highly structured and defined by the financial industry, but you can simply clone one of your existing preprinted checks. You end up with a signature image, perhaps a company logo image, and some clusters of text scattered around the form. Blank check stock can be purchased in a variety of colors, if needed, to create visual identities for each account.

One thing unique to check printing in this context is the MICR font required to print the account details at the bottom of the check. The font is downloaded to the printer like a macro, and is then selected later by the application to print the MICR data. Further, bank specifications say that MICR data should be printed using magnetic toner. Many people do NOT use magnetic toner without incident. It is, though, something to discuss with the bank.

If you are printing three different checks on the same sheet, then you would NOT embed the **x,y values** in the signature file. Rather you would create a second macro by hand that would be used as the overlay. It would supply the appropriate **x,y** positioning codes before each of the three calls to the signature macro.

If you are interested in creating your own checks but don't feel comfortable enough to set it up by yourself, we do offer custom PCL programming at reasonable rates. Call Jim Asman at (604)584-0977 for details.