The product described in this book is a licensed product of NCR Corporation.

NCR is a registered trademark of NCR Corporation.

It is the policy of NCR Corporation (NCR) to improve products as new technology, components, software, and firmware become available. NCR, therefore, reserves the right to change specifications without prior notice.

All features, functions, and operations described herein may not be marketed by NCR in all parts of the world. In some instances, photographs are of equipment prototypes. Therefore, before using this document, consult with your NCR representative or NCR office for information that is applicable and current.

To maintain the quality of our publications, we need your comments on the accuracy, clarity, organization, and value of this book.

Address correspondence to:

Copyright © 2017
By NCR Corporation Duluth,
GA U.S.A.
All Rights Reserved
Preface

Audience

This guide is targeted to users who want to understand how NCR printer driver works in order to maximize the capabilities of the printer and print text/graphics documents successfully.

This guide is also geared toward software developers who need to create custom application to perform special printer functionalities using native printer command sequences such as barcode printing, bit-image download, bit-image printing, printer status notification, etc.

Notice: This document is NCR proprietary information and is not to be disclosed or reproduced without consent.
References

- NCR 7197 Thermal Receipt Printer Owner’s Manual (B005-000-1409)
- NCR 7167 Two Station-POS Printer Owner’s Manual (B005-000-1406)
- NCR K590 Self Service Printer Owner’s Guide (B005-0000-1346)
- NCR 7198 Thermal Receipt Printer Owner’s Guide (B005-0000-1736)
- NCR 7168 Thermal Receipt Printer Owner’s Guide (B005-0000-1713)
- NCR F306 FastLane Thermal Printer Owner’s Guide
- NCR 7152 Self Service Printer Owner’s Manual (B005-0000-1739)
- NCR 7346-F309 2ST FastLane Printer Owner’s Manual (B005-0000-1816)
# Table of Contents

Introduction .................................................................................................................................. 1  
Printer driver Architecture .................................................................................................. 2  
  Printing Mechanism in Windows .................................................................................. 2  
  Printer Driver Components ......................................................................................... 4  
  Minidriver .................................................................................................................. 5  
  Language Monitor ....................................................................................................... 5  
  NCR POS Application Programming Interface .................................................. 6  
SETTINGS ............................................................................................................................ 9  
  Resolution .................................................................................................................. 9  
  Paper Sizes ................................................................................................................. 10  
  Paper Sources ............................................................................................................. 11  
  Paper Orientation ........................................................................................................ 13  
  Two sided thermal Printing ....................................................................................... 14  
    Two Sided Thermal(2ST) Printing Modes ............................................................... 15  
    Print on both Sides ................................................................................................. 17  
  Knife Cut Feature ....................................................................................................... 19  
  Changing Default Settings ......................................................................................... 21  
Programming (NCR Language Monitor & NCRPOSAPI) ........................................... 22  
  Commands .................................................................................................................. 22  
    Application Information & Error Notification ...................................................... 22  
  How To Use NCRPOSAPI.dll ..................................................................................... 27  
Interfaces ........................................................................................................................... 31  
  NCRPOS API Exported Functions ............................................................................. 31  
  DLL-Related Functions ............................................................................................. 34  
  Job Control Functions ................................................................................................. 36
## Revision Record

<table>
<thead>
<tr>
<th>Issue</th>
<th>Date</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Nov 2004</td>
<td>First issue</td>
</tr>
<tr>
<td>B</td>
<td>Jun 2007</td>
<td>Driver 4.0 release; NCRPOSAPI 4.2 release; Windows standalone install documentation</td>
</tr>
<tr>
<td>C</td>
<td>Jan 2008</td>
<td>Addition of Knife Cut Feature</td>
</tr>
<tr>
<td>E</td>
<td>May 2008</td>
<td>Two Sided thermal printing feature</td>
</tr>
<tr>
<td>F</td>
<td>May 2008</td>
<td>Addition of 7198 Receipt Label printer. Removal of references for O/S prior to XP, and non-existant printer models</td>
</tr>
<tr>
<td>G</td>
<td>Apr 2017</td>
<td>Driver 6.x.x Release</td>
</tr>
</tbody>
</table>
Introduction

This manual discusses the windows printer driver technology in order to successfully print on the following NCR Retail raster printers:

- NCR 7167 Receipt Printer
- NCR 7167 Slip Printer
- NCR 7197 Receipt Printer
- NCR K590 Self-Service Printer
- NCR F306 Self-Service Printer
- NCR 7168 Receipt 2ST Printer
- NCR 7168 Slip Printer
- NCR 7198 Receipt 2ST Printer
- NCR 7198 Slip Printer
- NCR F307 Self-Service Printer
- NCR F309 Self-Service Printer
- NCR 7198xx4x Label Receipt Printer
- NCR 7199 Receipt Printer
Printer driver Architecture

Printing Mechanism in Windows

If an application uses a printer in Windows, it starts a complex interaction with the GDI16 / GDI32 library module, the printer device driver library module (which has a .DRV/.dll extension) and the Windows print spooler.
The following figure explains how the printing mechanism is being implemented in Windows platform.

![Diagram of Printing Mechanism in Windows](image)

**Figure 1. Printing Mechanism in Windows**
When the application wants printing to begin using a printer, it first obtains a handle to the printer device context using CreateDC or PrintDlg. This causes the printer device driver library module to be loaded into memory and to initialize itself. The program then calls the StartDoc function, which signals the beginning of a new document. The StartDoc function is handled by the GDI module. The GDI module calls the Control function in the printer device driver, telling the device driver to prepare for printing.

The call to StartDoc begins the process of printing a document; the process ends when the program calls EndDoc. These two calls act as bookends for the normal GDI functions that display text or graphics to the document pages. Each page is itself delimited by a call to StartPage to begin a page and EndPage to end the page. For example, if a program wants to draw an ellipse on the page, it first calls StartDoc to begin the print job, then StartPage to signal a new page. It then calls Ellipse, just as it does when drawing an ellipse on the screen. When the application program is finished, the program then calls EndPage.

**Printer Driver Components**

A Windows Raster printer driver consists of the following components:

- The *printer graphics driver DLL* implements the DDI to the degree necessary to support printer rendering features and capabilities that cannot be adequately handled by the graphics engine.

- The *printer interface driver DLL* creates an interface that allows users to set the physical properties of the printer and the logical properties of the document. This interface is also responsible for interacting with the Common Property Sheet User Interface to provide device- and document-specific property sheet page(s) in response to an application request.

- An optional print processor that helps the spooler process to despool print jobs.
• An optional language monitor to enable spooler to communicate with the bi-directional printer
• An optional port monitor that sends printer-ready data to hardware port drivers.

To help vendors provide printing solutions to customers, Microsoft provides a standard printer driver for non-Postscript printers called Universal Printer Driver (**Unidrv**). Instead of creating full-blown drivers, printer vendors only need to provide *minidrivers* to support new printers.

In Windows 2000/XP and later versions, system-supplied UNIDRV.DLL and UNIDRVUI.DLL are the printer graphics driver DLL and printer interface DLL respectively.

**Minidriver**

A printer minidriver is data file that provides device-driver support for a particular class of printers. A printer minidriver contains a collection of data structures referred to as the printer's data table. A data table contains data such as the printer's resolution, color capabilities, metrics for device fonts, and the strings that correspond to various printer commands. It also specifies printer model-specific aspects of how primitive bitmap and text operations are to be performed.

NCR provides minidrivers to allow Unidrv to support NCR POS printers.

**Language Monitor**

NCR printer drivers also include a language monitor based on **ESC/P POS** language to provide a full-duplex communication channel between the print spooler and the NCR printers.

This Language Monitor will be able to handle the following errors/status in the printer:

1. Printer is busy.
2. Receipt paper door or Slip Cassette door is open
3. Paper Feed button is pressed
4. Printing stopped due to paper condition
5. Slip motor jam
6. Knife error occurred
7. Thermal print head temperature and power supply voltage are out of range
8. Paper low (if paper low sensor enabled)
9. Receipt Paper exhausted
10. Unrecoverable error
11. Slip paper exhausted
12. Presenter Mechanism Error (Only for K590)
13. Paper jam status (Only for K590)
14. Printer door status (Only for K590)
15. Black Mark Detection Failure (Only for K590)
16. Printer Head Failure (Only for K590)
17. Slip Leading Edge Sensor Error
18. Slip Trailing Edge Sensor Error
19. Presenter Paper Detector Paper Status (Only for K590)
20. Printer Reset Status (Only for K590)

**NCR POS Application Programming Interface**

Aside from printing normal text/graphics documents, NCR printers are capable of reporting printer status and performing special printer functionalities using native printer command sequences such as barcode printing, Bitimage download and printing, etc. To take advantage of these functionalities, NCR provides NCRPOS API dll which application developers can exploit. This DLL will be part of the NCR POS printer driver installation package. It is copied during installation to the System directory of the OS.
NCR POS API DLL exports the following functions:

- NCRPOS_WritePrinter()
- NCRPOS_ReadPrinter()
- NCRPOS_PrinterStatus()

The application developers will load this DLL using `LoadLibrary` function call during run-time. After the DLL is loaded into the virtual memory address of the calling process, the application will call `GetProcAddress` function to get the addresses of the above mentioned exported. The application will then call these exported DLL functions using the function pointers returned by `GetProcAddress`. 
Figure 2. Architecture of Interfacing the DLL with an application module

**Note:** When using the NCRPOSAPI.DLL/NCRAPI.DLL from the application (provided with the package) for querying printer status, the user may get a time-out message. In such cases, the user is requested to wait for some time before querying the printer status again.
Settings

Resolution

This printer driver supports various print resolutions.

The following table describes the printer wise breakup of supported resolutions:

<table>
<thead>
<tr>
<th>Printer</th>
<th>Supported Resolution</th>
<th>Displayed Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCR K590 Self-Service</td>
<td>203 dpi x 68 dpi</td>
<td>203 x 68 dots per inch</td>
</tr>
<tr>
<td>NCR F306 Self-Service</td>
<td>203 dpi x 203 dpi</td>
<td>203 x 203 dots per inch</td>
</tr>
<tr>
<td>NCR F307 Self-Service</td>
<td>203 dpi x 203 dpi</td>
<td>203 x 203 dots per inch</td>
</tr>
<tr>
<td>NCR F309 Self-Service</td>
<td>203 dpi x 203 dpi</td>
<td>203 x 203 dots per inch</td>
</tr>
<tr>
<td>NCR 7197 Receipt</td>
<td>203 dpi x 68 dpi</td>
<td>203 x 68 dots per inch</td>
</tr>
<tr>
<td>NCR 7197 Receipt</td>
<td>203 dpi x 203 dpi</td>
<td>203 x 203 dots per inch</td>
</tr>
<tr>
<td>NCR 7167 Receipt</td>
<td>203 dpi x 68 dpi</td>
<td>203 x 68 dots per inch</td>
</tr>
<tr>
<td>NCR 7168 Receipt</td>
<td>203 dpi x 203 dpi</td>
<td>203 x 203 dots per inch</td>
</tr>
<tr>
<td>NCR 7198 Receipt</td>
<td>203 dpi x 68 dpi</td>
<td>203 x 68 dots per inch</td>
</tr>
<tr>
<td>NCR 7198xx4x Label Receipt</td>
<td>203 dpi x 203 dpi</td>
<td>203 x 203 dots per inch</td>
</tr>
<tr>
<td>NCR 7199 Receipt</td>
<td>70 dpi x 72 dpi</td>
<td>70 x 72 dots per inch</td>
</tr>
<tr>
<td>NCR 7167 Slip</td>
<td>140 dpi x 72 dpi</td>
<td>140 x 72 dots per inch</td>
</tr>
<tr>
<td>NCR 7168 Slip</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For K590, 7167, 7197, 7198 and 7199 Receipt printers, the default resolution is set to 203 x 203 dpi.
For 7167 and 7168 Slip printer, the default resolution is set to 140 x 72 dpi.

**Paper Sizes**

This printer driver supports various paper sizes. The following table describes the printer wise breakup of supported paper sizes:

<table>
<thead>
<tr>
<th>Printer</th>
<th>Supported Paper Sizes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCR K590 Self-Service</td>
<td>80 mm x 111 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>80 mm x 127 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>80 mm x 140 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>80 mm x 152 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>80 mm x 75 mm TO 80 mm x 600 mm</td>
</tr>
<tr>
<td></td>
<td>82.5 mm x 111 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>82.5 mm x 127 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>82.5 mm x 140 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>82.5 mm x 152 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>82.5 mm x 75 mm TO 82.5 mm x 600 mm</td>
</tr>
<tr>
<td></td>
<td>114 mm x 111 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>114 mm x 127 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>114 mm x 140 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>114 mm x 152 mm Black Mark</td>
</tr>
<tr>
<td></td>
<td>114 mm x 75 mm TO 114 mm x 600 mm</td>
</tr>
<tr>
<td>NCR 7197 Receipt</td>
<td>80 mm</td>
</tr>
<tr>
<td>NCR 7198xx4x Label Receipt</td>
<td>58 mm</td>
</tr>
</tbody>
</table>
For 7167, 7197, 7198 and 7199 Receipt printers, the default paper size is 80mm.

For K509 Self-Service printer, the default paper size is 80mm x 75mm TO 80mm x 600mm.

**Paper Sources**

This driver provides the following paper source for POS Printer.

<table>
<thead>
<tr>
<th>Printer</th>
<th>Supported Paper Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCR K590 Self-Service</td>
<td>Receipt</td>
</tr>
<tr>
<td>NCR 7197 Receipt</td>
<td>Receipt</td>
</tr>
<tr>
<td>NCR 7167 Receipt</td>
<td>Receipt</td>
</tr>
<tr>
<td>NCR 7167 Slip</td>
<td>Slip</td>
</tr>
<tr>
<td>NCR 7168 Receipt</td>
<td>Receipt</td>
</tr>
<tr>
<td>NCR 7168 Slip</td>
<td>Slip</td>
</tr>
<tr>
<td>NCR F306 Self-Service</td>
<td>Receipt</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------</td>
</tr>
<tr>
<td>NCR F307 Self-Service</td>
<td>Receipt</td>
</tr>
<tr>
<td>NCR F309 Self-Service</td>
<td>Receipt</td>
</tr>
<tr>
<td>NCR 7198 Receipt</td>
<td>Receipt</td>
</tr>
<tr>
<td>NCR 7198xx4x Label Receipt</td>
<td>Receipt</td>
</tr>
</tbody>
</table>
Paper Orientation

This printer driver supports the following paper orientation

- Portrait
- Landscape

The default paper orientation is set to *Portrait*. 
Two sided thermal Printing

The printer driver supports the printing on both sides of a sheet of paper for 7168, 7198 and F309 Printer. Two sided printing option is shown below.
Two Sided Thermal(2ST) Printing Modes

The printer driver supports five types of modes for printing on two sided thermal paper. Clicking on the Advanced button would show the window below for the 7168 printer. The Two-Sided Thermal printing modes are shown in a dropdown list.
**Single-sided Receipt:**
In this mode printer prints the data on front side of paper, as soon as it receives the data from application. This is the usual printing mode just like other printers.

**Single-sided Buffered Receipt:**
In this mode printer buffers the data until a paper cut command and then it prints the data at once on front side of the paper.

**Duplex**
In this mode the printer receives the data until paper cut command and then it prints the data on both sides of paper, where data on odd page number is printed on front side of paper and data on even page number is printed on back side of the paper.

**Duplex Split Receipt**
In this mode the printer receives single sided data until a paper cut command is received. The print data is then automatically divided into two parts where the first part is printed on the front side and the second part is printed on back of the receipt when knife cut command is send to the printer.

**Duplex Pre-Defined Image**
This option will allow the automatic printing of predefined data on back of the receipt. The pre-defined data is preloaded into the printer and when the receipt is printed the predefined data is automatically printed on the back of the receipt.
Print on both Sides

In order to select one sided or two sided printing, on the Layout tab, under Print on Both Sides, click one of the following options:

- None to print just on front side of paper.
- Flip on Long Edge to print on both sides of the paper with pages that are bound on the left edge.
- Flip on Short Edge to print on both sides of the paper with pages that are bound on the top edge.
In order to select any single sided options of thermal printing mode like Single-sided Receipt or Single-sided Buffered Receipt, the Print on Both Sides option must be selected as None in the window above.

In order to select any two sided options of thermal printing modes like Duplex/ Duplex Split Receipt/Duplex Pre-Defined Image, the Print on Both Sides options other than None must be selected in the window above.

**Note:** While printing the document with two sided thermal option as Duplex, the top margin on back side of paper is the default margin of physical printer device for back side printing and it is not adjustable.
Knife Cut Feature

This printer driver supports knife cut feature. With the knife cut feature, printer becomes capable of cutting the page after printing. The cutting of page takes place as per the selected options shown below.
**End page** option cuts the paper upon reaching the end of every page of the document.

**EndDoc** option cuts the paper upon reaching the end of the document.

**EndJob** option cuts the paper upon reaching the end of the job.

The default Knife Cut option is set to *End Page*.

Note: The K-590 printer configured with EndDoc or EndJob knife cut option, supports maximum document limit of 610 mm.
Changing Default Settings

To view and modify the default settings of your printer, perform the following steps:

1. Select an installed printer from *Printers* folder.
2. Right click on the printer icon and select ‘*Printing Preferences*’.
3. Select *Layout* tab to change paper orientation.
4. Select *Paper/Quality* tab to change paper source.
5. Press the *Advanced* button to change resolution and paper size settings.
Programming (NCR Language Monitor & NCRPOSAPI)

Commands

The different features and functions provided by the printer are controlled by sending commands from the host computer to the printer. The printer commands are made up of one or more bytes of data starting with a command control code followed by its supporting parameters. For in-depth information regarding the available commands supported by your printer, see references.

To be able to send commands to and read status from the printer, use the functions exported by NCRPOSAPI.dll.

Application Information & Error Notification

Language monitor generates an event to notify about the error message to user mode applications. This mechanism is explained below.

Language monitor registers a message called “NCRRSD_WIN32_GDI_ERROR” using RegisterWindowMessage() API. RegisterWindowMessage API generates a system wide unique message identifier for “NCRRSD_WIN32_GDI_ERROR” and guarantees that it does not conflict with any existing or future messages. An application that wants to receive this message has to call RegisterWindowMessage(“NCRRSD_WIN32_GDI_ERROR”) from within the application. Once an application calls RegisterWindowMessage() API, it can handle this message in its message handling routine.
Language monitor declares 32-bit error macros for NCR printer statuses to uniquely identify each printer status using this 32-bit macro. A list of these macros along with NCR printer statuses is shown in the table below.

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>NCR printer status</th>
<th>32-bit error code used by Language Monitor</th>
<th>32-bit value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Communication Interface busy</td>
<td>PTRSTAT_INTERFACE_BUSY</td>
<td>0x00002000</td>
</tr>
<tr>
<td>2.</td>
<td>Cover or Door open</td>
<td>PTRSTAT_COVER_OPEN</td>
<td>0x00000001</td>
</tr>
<tr>
<td>3.</td>
<td>Paper feed button is pressed</td>
<td>PTRSTAT_PAPERFEED_PRESSED</td>
<td>0x00000020</td>
</tr>
<tr>
<td>4.</td>
<td>Slip motor fault</td>
<td>PTRSTAT_MOTOR_FAULT</td>
<td>0x00002000</td>
</tr>
<tr>
<td>5.</td>
<td>Knife error</td>
<td>PTRSTAT_KNIFE_FAULT</td>
<td>0x00000400</td>
</tr>
<tr>
<td>6.</td>
<td>General fault</td>
<td>PTRSTAT_FAULT</td>
<td>0x00000800</td>
</tr>
<tr>
<td>7.</td>
<td>Unrecoverable error</td>
<td>PTRSTAT_FAILURE</td>
<td>0x00001000</td>
</tr>
<tr>
<td>8.</td>
<td>Printhead temperature or Voltage out of range</td>
<td>PTRSTAT_AUTORECOVER_FAULT</td>
<td>0x00000100</td>
</tr>
<tr>
<td>9.</td>
<td>Receipt low</td>
<td>PTRSTAT_REC_NEAREND</td>
<td>0x00000004</td>
</tr>
<tr>
<td>10.</td>
<td>Receipt empty</td>
<td>PTRSTAT_REC_EMPTY</td>
<td>0x00000002</td>
</tr>
<tr>
<td>11.</td>
<td>Paper condition error</td>
<td>PTRSTAT_PAPER_CONDITION</td>
<td>0x00000080</td>
</tr>
<tr>
<td>12.</td>
<td>Slip leading edge sensor no paper</td>
<td>PTRSTAT_SLP_LEAD_NO_PAPER</td>
<td>0x00000008</td>
</tr>
<tr>
<td>13.</td>
<td>Slip trailing edge sensor no paper</td>
<td>PTRSTAT_SLP_TRAIL_NO_PAPER</td>
<td>0x00000010</td>
</tr>
<tr>
<td>14.</td>
<td>Slip empty</td>
<td>PTRSTAT_SLP_EMPTY</td>
<td>0x00000040</td>
</tr>
<tr>
<td>15.</td>
<td>Presenter mechanism error</td>
<td>PTRSTAT_PRESENTER_ERR</td>
<td>0x00010000</td>
</tr>
<tr>
<td>16.</td>
<td>Presenter paper jam</td>
<td>PTRSTAT_PAPER_JAM</td>
<td>0x00020000</td>
</tr>
<tr>
<td>17.</td>
<td>Presenter door open</td>
<td>PTRSTAT_CDS_OPEN</td>
<td>0x00040000</td>
</tr>
<tr>
<td>Sr. No.</td>
<td>NCR printer status</td>
<td>32-bit error code used by Language Monitor</td>
<td>32-bit value</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------</td>
<td>----------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>18</td>
<td>Black mark detection failed</td>
<td>PTRSTAT_BMD_FAILURE</td>
<td>0x00080000</td>
</tr>
<tr>
<td>19</td>
<td>Printhead damaged</td>
<td>PTRSTAT_HEAD_FAILURE</td>
<td>0x00100000</td>
</tr>
<tr>
<td>20</td>
<td>Presenter Paper Detector Paper status</td>
<td>PTRSTAT_PRSNT_PDP_STATUS</td>
<td>0x00200000</td>
</tr>
<tr>
<td>21</td>
<td>Printer Reset status</td>
<td>PTRSTAT_PRINTER_RESET</td>
<td>0x00400000</td>
</tr>
<tr>
<td>22</td>
<td>Time Out</td>
<td>PTRSTAT_TIMEOUT</td>
<td>0x00800000</td>
</tr>
</tbody>
</table>

Below is a standard Win32 application code showing how to obtain NCR error message. An action of popping up a Message Box is shown in this code. Such an action is not the recommended way of handling errors. It is only shown for demonstration purposes. It is recommended that the application prompt the user to take the appropriate action to recover from the error, and then attempt to continue printing, or reprint depending on the situation.

/* main.c
*/

#include <windows.h>

/* define error constants */

#define PTRSTAT_COVER_OPEN   0x00000001 // Cover or door open.
#define PTRSTAT_REC_EMPTY    0x00000002 // Receipt out.
#define PTRSTAT_PAPERFEED_PRESSED 0x00000020 // paper feed button pressed

/* other error constants */

const char g_szClassName[] = "myWindowClass";

UINT NCRMsgID = 0;

/* The Window Procedure */
LRESULT CALLBACK WndProc(HWND hwnd, UINT msg, WPARAM wParam, LPARAM lParam)
{
if (msg == NCRMsgID)
{
    if (wParam & PTRSTAT_COVER_OPEN)
        MessageBox(hwnd, "Cover Open", "Error", MB_OK);

    if (wParam & PTRSTAT_REC_EMPTY)
        MessageBox(hwnd, "Out of Paper", "Error", MB_OK);

    if (wParam & PTRSTAT_PAPERFEED_PRESSED)
        MessageBox(hwnd, "Paper feed button is pressed.", "Error", MB_OK);

    /* process other errors here */
}

switch(msg)
{
    case WM_CLOSE:
        DestroyWindow(hwnd);
        break;
    case WM_DESTROY:
        PostQuitMessage(0);
        break;
    default:
        return DefWindowProc(hwnd, msg, wParam, lParam);
}
return 0;

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    WNDCLASSEX wc;
    HWND hwnd;
    MSG Msg;

    /* Registering the Window Class */
    wc.cbSize = sizeof(WNDCLASSEX);
    wc.style = 0;
    wc.lpfnWndProc = WndProc;
    wc.cbClsExtra = 0;
    wc.cbWndExtra = 0;
    wc.hInstance = hInstance;
wc.hIcon = LoadIcon(NULL, IDI_APPLICATION);
wC.hCursor = LoadCursor(NULL, IDC_ARROW);
wC.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
wC.lpszMenuName = NULL;
wC.lpszClassName = g_szClassName;
wC.hIconSm = LoadIcon(NULL, IDI_APPLICATION);

if(!RegisterClassEx(&wC))
{
    MessageBox(NULL, "Window Registration Failed!", "Error!", MB_ICONEXCLAMATION | MB_OK);
    return 0;
}

// Creating the Window
HWND = CreateWindowEx(WS_EX_CLIENTEDGE, g_szClassName,
"Test Application", WS_OVERLAPPEDWINDOW,
    CW_USEDEFAULT, CW_USEDEFAULT, 240, 120,
    NULL, NULL, hInstance, NULL);

if(hwnd == NULL)
{
    MessageBox(NULL, "Window Creation Failed!", "Error!", MB_ICONEXCLAMATION | MB_OK);
    return 0;
}

ShowWindow(hwnd, nCmdShow);
UpdateWindow(hwnd);

/* register so we can be notified about NCR error messages */
NCRMsgID = RegisterWindowMessage("NCRRSD_WIN32_GDI_ERROR");

/* The Message Loop */
while(GetMessage(&Msg, NULL, 0, 0) > 0)
{
    TranslateMessage(&Msg);
    DispatchMessage(&Msg);
}

return Msg.wParam;
How To Use NCRPOSAPI.dll

The pseudo-code to access the NCRPOS API DLL’s exported functions from an application using run-time dynamic linking is given below:

Library_handle = LoadLibrary ( DLL_Path \ DLL_name) // The LoadLibrary function maps the specified executable module into the address space of the calling process

If ( Library_handle == NULL )
{
    Message ( Failed to obtain DLL handle)
    Exit
}

Function_handle = GetProcAddress( Function_name ) // The GetProcAddress function retrieves the address of an exported function from the specified dynamic-link library (DLL)

NCRMessage = 0
If ( Function_handle == NULL)
    Message ( GetProcAddress Failed )
else
    Function_handle ( Printer_name, NCRMessage) // Call the function using the function pointer and providing pointer to message value to be returned.

FreeLibrary(Library_handle) // Loaded dll will be freed from the process address space
Below is a sample code to demonstrate how to communicate with a printer using NCRPOSAPI.dll. The code specifically sets up a print bar code command sequences and writes it to the printer.

/* main.c
 */

#include windows.h

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    HINSTANCE hLibrary;
    FARPROC hFunction = NULL;
    char PrinterName[] = "NCR 7167 Receipt"; /* or NCR K590 Self-Service or /
        /NCR 7197 Receipt or NCR 7167 Slip */

    HANDLE hPrinter;
    DOC_INFO_1 pDocInfo;
    DWORD pcWritten;
    char pBarCode[16];

    /***** Loads the NCRPOSAPI.DLL****/

    hLibrary = LoadLibrary("NCRPOSAPI.DLL");
    if(hLibrary == NULL)
    {
        MessageBox(NULL,"NCRAPI.DLL cannot be loaded", "Error", MB_ICONERROR);
        return 1;
    }

    /***** Retrieves the NCRPOS_WritePrinter() function handle ****/

    hFunction = GetProcAddress(hLibrary,"NCRPOS_WritePrinter");
    if(hFunction == NULL)
    {
        MessageBox(NULL, "NCRPOS_WritePrinter function failed", "Error", MB_ICONERROR);
        return 1;
    }

    /***** Retrieves the Printer handle ****/
if(!OpenPrinter(PrinterName, &hPrinter, NULL))
{
    MessageBox(NULL, "Error Opening Printer", "Error", MB_ICONERROR);
    return 1;
}

pDocInfo.pDocName = "Bar Code";
pDocInfo.pDatatype = NULL;
pDocInfo.pOutputFile = NULL;

StartDocPrinter(hPrinter, 1, (LPBYTE)&pDocInfo);
StartPagePrinter(hPrinter);

/***** Setup UPC-A bar code command ****/
pBarCode[0] = 0x1D;
pBarCode[1] = 0x6B;
pBarCode[2] = 0x00; /* UPC-A bar code type */

/***** numeric data ****/
pBarCode[3] = '0';
pBarCode[8] = '5';
pBarCode[10] = '7';
pBarCode[12] = '9';
pBarCode[13] = '0';
pBarCode[14] = '5'; /* check digit */
pBarCode[15] = 0; /* end of command */

/****** NCRPOS_WriterPrinter() function call *******
if(!hFunction( hPrinter, pBarCode, 16, &pcWritten))
{
    MessageBox(NULL, "Error in Writing to the Printer", "Error", MB_ICONERROR);
    return 1;
}
else
    MessageBox(NULL, "Command successfully Written to the Printer", "Success", MB_OK);

    EndPagePrinter(hPrinter);
    EndDocPrinter(hPrinter);
    ClosePrinter(hPrinter);
    FreeLibrary(hLibrary);

    return 0;
}
Interfaces

NCRPOS API Exported Functions

BOOL NCRPOS_WritePrinter(HANDLE hPrinter, LPVOID pBuf, DWORD cbBuf, LPDWORD pcWritten)
Notifies the print spooler that data should be written to the specified printer.

Parameters:

- **hPrinter** - [in] Handle to the printer. Use the **OpenPrinter** function to retrieve a printer handle.

- **pBuf** - [in] Pointer to an array of bytes that contains the data that should be written to the printer.

- **cbBuf** - [in] Specifies the size, in bytes, of the array.

- **pcWritten** - [out] Pointer to a value that receives the number of bytes of data that were written to the printer.

Returns:

- If the function succeeds the return value is nonzero.
- If the function fails, the return value is zero.

BOOL NCRPOS_ReadPrinter(HANDLE hPrinter, LPVOID pBuf, DWORD cbBuf, LPDWORD pNoBytesRead)
Retrieves data from the specified printer.
**Parameters:**

- `hPrinter` - [in] Handle to the printer for which to retrieve data. Use the **OpenPrinter** function to retrieve a printer handle.

- `pBuf` - [in] Pointer to a buffer that receives the printer data.

- `cbBuf` - [in] Specifies the size, in bytes, of the buffer to which `pBuf` points.

- `pNoBytesRead` - [out] Pointer to a value that receives the number of bytes of data copied into the array to which `pBuf` points.

**Returns:**

- If the function succeeds the return value is nonzero.
- If the function fails, the return value is zero.

---

```c
BOOL CALLBACK NCRPOS_PrinterStatus(LPSTR PrinterName, LPDWORD NCRPOSMsg)
```

Retrieves 32-bit printer status message identifier posted by the Language Monitor Module of the specified Printer Driver.

**Parameters:**

- `PrinterName` - [in] Pointer to a null-terminated string that specifies the printer object name.
- `NCRPOSMsg` - [out] Return value is 32-bit unique message identifier. See the table below for possible values.

**Returns:**

- If the function succeeds the return value is True.
- If the function fails, the return value is False.
<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>NCR printer status</th>
<th>32-bit error code used by Language Monitor</th>
<th>Macro</th>
<th>32-bit value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Communication interface busy</td>
<td>PTRSTAT_INTERFACE_EMPTY</td>
<td>0x00002000</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Cover or Door open</td>
<td>PTRSTAT_COVER_OPEN</td>
<td>0x00000001</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Paper feed button is pressed</td>
<td>PTRSTAT_PAPERFEED_PRESSED</td>
<td>0x00000200</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Slip motor fault</td>
<td>PTRSTAT_MOTOR_FAULT</td>
<td>0x00000200</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Knife error</td>
<td>PTRSTAT_KNIFE_FAULT</td>
<td>0x00000400</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>General fault</td>
<td>PTRSTAT_FAULT</td>
<td>0x00000800</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>Unrecoverable error</td>
<td>PTRSTAT_FAILURE</td>
<td>0x00010000</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>Printhead temp. or Voltage out of range</td>
<td>PTRSTAT_AUTORECOVER_FAULT</td>
<td>0x00001000</td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Receipt low</td>
<td>PTRSTAT_REC_NEAREND</td>
<td>0x00000004</td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>Receipt empty</td>
<td>PTRSTAT_REC_EMPTY</td>
<td>0x00000002</td>
<td></td>
</tr>
<tr>
<td>11.</td>
<td>Paper condition error</td>
<td>PTRSTAT_PAPER_CONDITION</td>
<td>0x00000080</td>
<td></td>
</tr>
<tr>
<td>12.</td>
<td>Slip leading edge sensor no paper</td>
<td>PTRSTAT_SLP_LEAD_NO_PAPER</td>
<td>0x00000008</td>
<td></td>
</tr>
<tr>
<td>13.</td>
<td>Slip trailing edge sensor no paper</td>
<td>PTRSTAT_SLP_TRAIL_NO_PAPER</td>
<td>0x00000010</td>
<td></td>
</tr>
<tr>
<td>14.</td>
<td>Slip empty</td>
<td>PTRSTAT_SLP_EMPTY</td>
<td>0x00000040</td>
<td></td>
</tr>
<tr>
<td>15.</td>
<td>Presenter mechanism error</td>
<td>PTRSTAT_PRESENTER_ERR</td>
<td>0x00010000</td>
<td></td>
</tr>
<tr>
<td>16.</td>
<td>Presenter paper jam</td>
<td>PTRSTAT_PAPER_JAM</td>
<td>0x00020000</td>
<td></td>
</tr>
<tr>
<td>17.</td>
<td>Presenter door open</td>
<td>PTRSTAT_CDS_OPEN</td>
<td>0x00040000</td>
<td></td>
</tr>
<tr>
<td>18.</td>
<td>Black mark detection failed</td>
<td>PTRSTAT_BMD_FAILURE</td>
<td>0x00080000</td>
<td></td>
</tr>
<tr>
<td>19.</td>
<td>Printhead damaged</td>
<td>PTRSTAT_HEAD_FAILURE</td>
<td>0x00100000</td>
<td></td>
</tr>
<tr>
<td>20.</td>
<td>Presenter Paper Detector Paper status</td>
<td>PTRSTAT_PRSNT_PDP_STATUS</td>
<td>0x00200000</td>
<td></td>
</tr>
<tr>
<td>21.</td>
<td>Printer Reset status</td>
<td>PTRSTAT_PRINTER_RESET</td>
<td>0x00400000</td>
<td></td>
</tr>
<tr>
<td>22.</td>
<td>Time Out</td>
<td>PTRSTAT_TIMEOUT</td>
<td>0x00800000</td>
<td></td>
</tr>
</tbody>
</table>
**DLL-Related Functions**

**HMODULE LoadLibrary(LPCTSTR lpFileName)**
Maps the specified executable module into the address space of the calling process.

**Parameter:**

*lpFileName* - [in] Pointer to a null-terminated string that names the executable module (either a .dll or .exe file). The name specified is the file name of the module and is not related to the name stored in the library module itself, as specified by the **LIBRARY** keyword in the module-definition (.def) file.

If the string specifies a path but the file does not exist in the specified directory, the function fails. When specifying a path, be sure to use backslashes (\), not forward slashes (/).

If the string does not specify a path, the function uses a standard search strategy to find the file.

**Returns:**

If the function succeeds, the return value is a handle to the module. If the function fails, the return value is NULL. To get extended error information, call **GetLastError**.

**FARPROC GetProcAddress(HMODULE hModule, LPCSTR lpProcName )**
Retrieves the address of an exported function or variable from the specified dynamic-link library (DLL).
Parameters:

*hModule* - [in] Handle to the DLL module that contains the function or variable. The **LoadLibrary** function returns this handle.

*lpProcName* - [in] Pointer to a null-terminated string containing the function or variable name, or the function's ordinal value. If this parameter is an ordinal value, it must be in the low-order word; the high-order word must be zero.

Returns:

If the function succeeds, the return value is the address of the exported function or variable.
If the function fails, the return value is NULL. To get extended error information, call **GetLastError**.

**BOOL FreeLibrary(HMODULE hModule)**
Decrement the reference count of the loaded dynamic-link library (DLL). When the reference count reaches zero, the module is unmapped from the address space of the calling process and the handle is no longer valid.

Parameters:

*hModule* - [in, out] Handle to the loaded DLL module. The **LoadLibrary** function returns this handle.

Returns:

If the function succeeds the return value is nonzero.
If the function fails, the return value is zero. To get extended error information, call **GetLastError**.
Job Control Functions

BOOL OpenPrinter( LPTSTR pPrinterName, LPHANDLE phPrinter,
LPPRINTER_DEFAULTS pDefault)
Retrieves a handle to the specified printer or print server.

Parameters:

pPrinterName - [in] Pointer to a null-terminated string that
specifies the name of the printer or print server.

phPrinter - [out] Pointer to a variable that receives a handle to the
open printer or print server object.

pDefault - [in] Pointer to a `PRINTER_DEFAULTS` structure. This
value can be NULL. Use the DesiredAccess member of the `PRINTER_DEFAULTS`
structure to specify the access rights that you need to the printer. The
access rights can be one of the following. (If `pDefault` is NULL, then the
access rights are `PRINTER_ACCESS_USE`.)

<table>
<thead>
<tr>
<th>Desired Access value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRINTER_ACCESS_ADMINISTER</td>
<td>To perform administrative tasks, such as those provided by SetPrinter.</td>
</tr>
<tr>
<td>PRINTER_ACCESS_USE</td>
<td>To perform basic printing operations.</td>
</tr>
</tbody>
</table>
| PRINTER_ALL_ACCESS                    | To perform all administrative tasks and basic printing
operations except for SYNCHRONIZE (see Standard Access Rights.) |
Returns:
If the function succeeds the return value is nonzero.
If the function fails, the return value is zero.

DWORD StartDocPrinter( HANDLE hPrinter, DWORD Level, LPBYTE pDocInfo)
Notifies the print spooler that a document is to be spooled for printing.

Parameters:
hPrinter - [in] Handle to the printer.
Level - [in] Specifies the version of the structure to which pDocInfo points.
pDocInfo - [in] Pointer to a structure that describes the document to print.

Returns:
If the function succeeds, the return value identifies the print job.
If the function fails, the return value is zero.

BOOL StartPagePrinter( HANDLE hPrinter )
Notifies the spooler that a page is about to be printed on the specified printer.

Parameters:
hPrinter - [in] Handle to the printer.

Returns:
If the function succeeds, the return value is nonzero.
If the function fails, the return value is zero.
BOOL EndPagePrinter( HANDLE hPrinter )
Notifies the print spooler that the application is at the end of a page in a print job.

Parameters:
    hPrinter - [in] Handle to the printer for which the page will be concluded.

Returns:
    If the function succeeds, the return value is nonzero.
    If the function fails, the return value is zero.

BOOL EndDocPrinter( HANDLE hPrinter )
Ends a print job for the specified printer.

Parameters:
    hPrinter - [in] Handle to a printer for which the print job should be ended.

Returns:
    If the function succeeds, the return value is nonzero.
    If the function fails, the return value is zero.

BOOL ClosePrinter( HANDLE hPrinter )
Closes the specified printer object.

Parameters:
    hPrinter - [in] Handle to the printer object to be closed.

Returns:
    If the function succeeds, the return value is nonzero.
    If the function fails, the return value is zero.