Zebra® Kiosk Driver

Reference Guide

P1006873-003RC

Table of Contents

Zebra Kiosk Printer Windows driver installation 5
Driver Installation 5
Zebra Kiosk Printer Windows driver documentation 8
Printing Preferences 9
Paper Size 10
Scaling 10
Document Options 10
Printer Properties 11
Sharing 11
Advanced 11
Color Management 12
Security 12
Device Settings 12
Printer Function 19
About 22
Ports 23
The Language Monitor 24
Windows APIs for Communication with the Printer 24
Event Notification 25
Status update in Windows “Printers and Faxes” or “Devices and Printer” 25
Appendix A 25
Windows Compatible Status 25
Statuses Defined in winspool.h 25
Windows Incompatible Status 27
Appendix B 28
Appendix C 29
Programming example 29
Background 29
Implementation in calling application 29
Implementation in monitor thread 30
WMI script to get basic status 32

Appendix D 35
Setup Printer Forms in Windows XP and Vista 35
Setup Printer Forms in Windows 7 36
References 39
Release Notes 40
Known issues 40

Zebra Kiosk Printer Windows driver installation

Note: If you have an older Zebra Kiosk Printer driver installed on your system please download the Un-Install Utility and remove the old driver. You can find the Utility by following the below suggestions.

On Zebra.com, follow these selections to get to the page where the Windows Driver Uninstaller software is available for download:

Zebra.com website

Select Drivers & Downloads from row highlighted in yellow

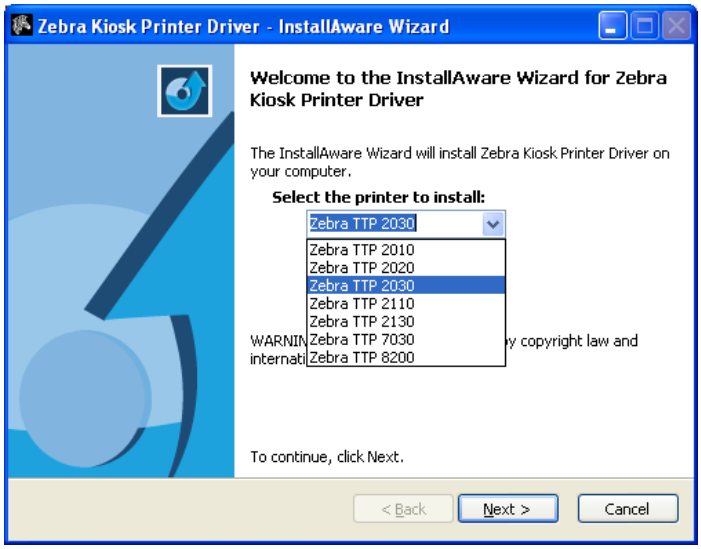
Select Utilities

Select Embedded/Kiosk Utilities

Driver Installation

The Zebra Kiosk Printer Windows driver will be deployed with an Installer application that installs the driver files on the Hard disk and pre-installs the driver for the selected printer.

After the start of the Installer application you have to select your respective printer



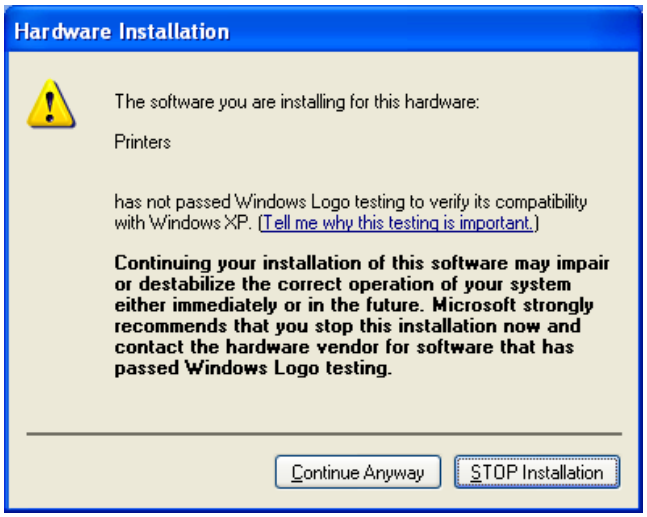
Continue with Next.

Accept the end user agreement and continue with Next.

Read the release notes and continue with Next.

The application will then start copying the driver files to the default driver directory C:\Zebra\Kiosk\WindowsDriver and start the pre-installation of the driver.

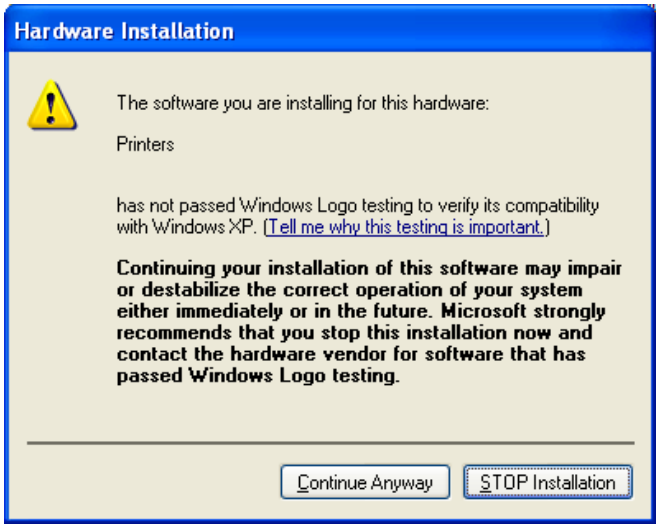
Due to the nature of the driver being not signed it will cause the Hardware Installation warning to display.



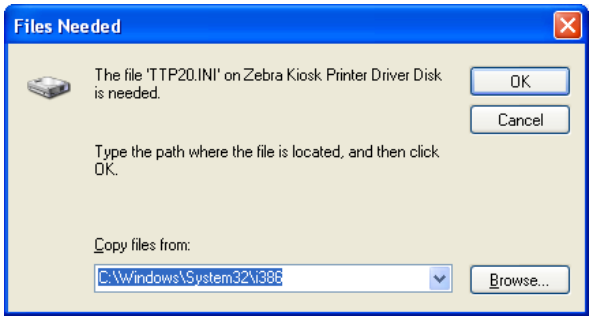
Continue the installation process with Continue Anyway.

Once the installation is completed the Installer application will ask to plug in the printer.

If you now plug your printer into the USB port the Hardware Installation Wizard will start with the unsigned driver warning.



Continue with Continue Anyway.



A Files Needed dialog will appear and you have to direct the dialog with the Browse button to the driver installation directory C:\Zebra\Kiosk\WindowsDriver and continue with Open than OK.

You will receive another Files Needed dialog and this time you need to browse to the respective OS version x86 or AMD64 and continue as before.

The Installation will finish.

Note:

If you are intending to install a serial or parallel printer on a 64 bit OS you will not be able to receive status at current release.

Zebra Kiosk Printer Windows driver documentation

The Zebra Kiosk Printer Windows driver is based on the Microsoft Unidriver architecture for raster based printers. Zebra provides two OEM libraries (UI and Rendering) to enable specific printer functionalities

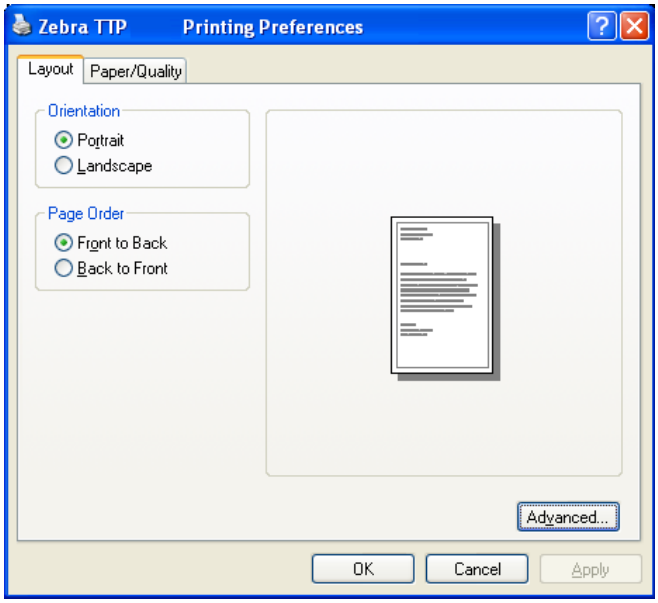
within the driver. In addition to the standard Microsoft driver, Zebra provides a bi-directional interface through a Language Monitor DLL.

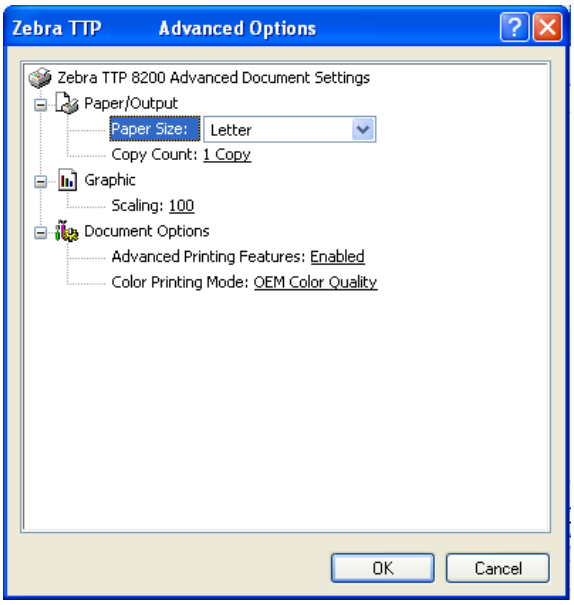
Due to the function compatibility of the four different printer families (TTP 2000, TTP 2100, TTP 7030 and TTP 8200) the drivers will share many functions in the UI and Rendering DLL as well as the Language Monitor.

All of the OEM features will be described below.

Printing Preferences

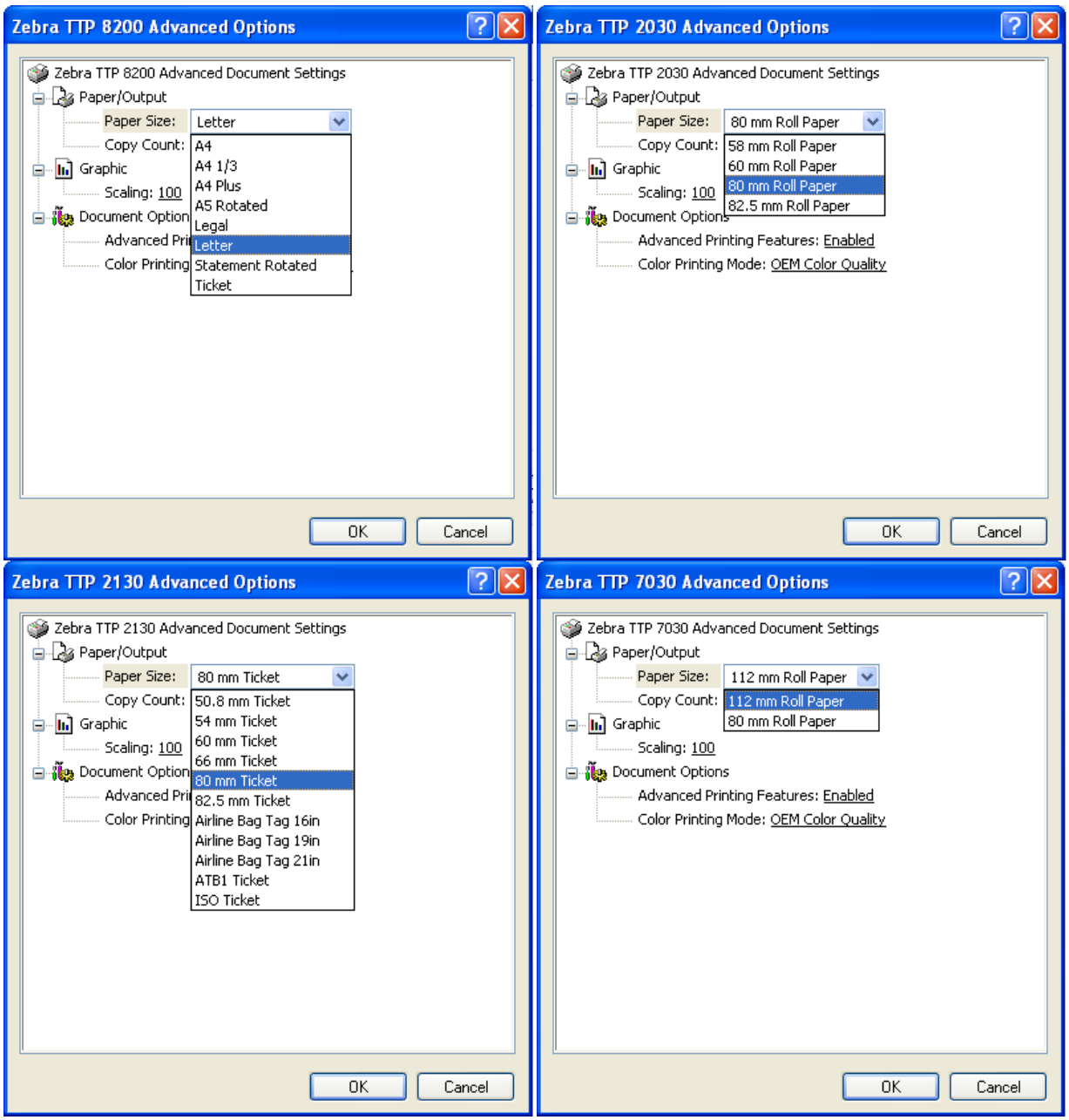
All the printer families share the same basic dialogs.





Paper Size

Variation can only be found in the available paper sizes.



Scaling

With the Scaling option you can change the size of your printable area and print larger pages on small paper if you scale down.

Document Options

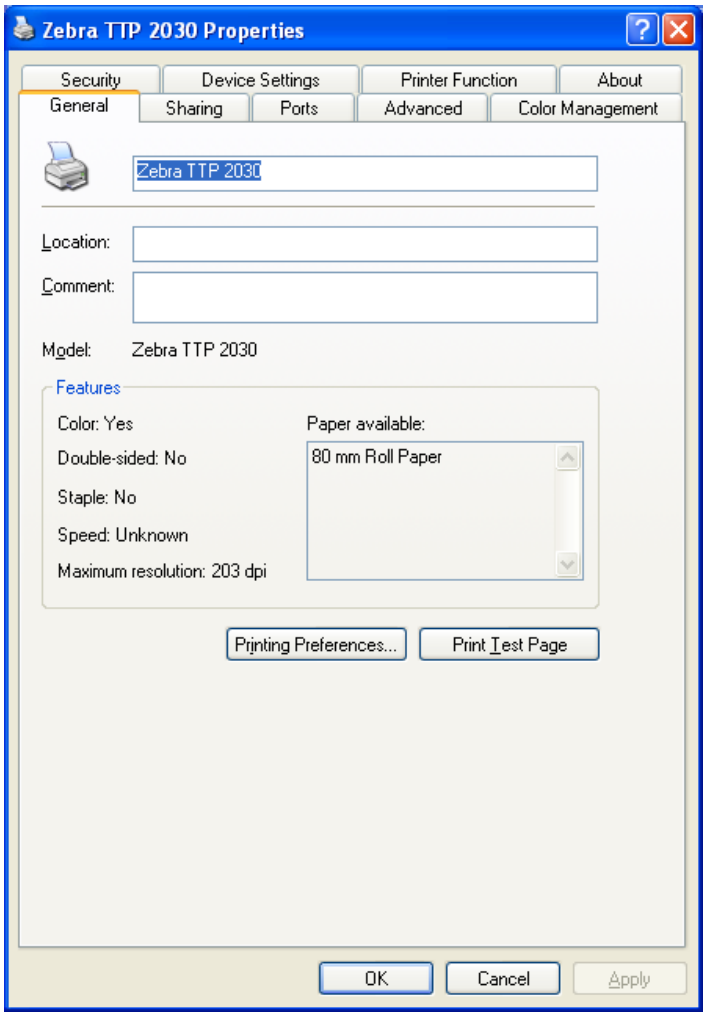
Advanced Printing Features

This is a Microsoft Unidriver setting and should always be set to Enabled.

Color Printing Mode

This OEM setting allows the user to select one of two currently available dithering modes.
OEM Color Quality mode is the default and will do a dithering similar to a Riemersma dither algorithm with a gray scaling effect.
B/W Quality mode will use a Threshold dithering algorithm that only displays black and white areas.

Printer Properties



Sharing

Use the Sharing tab if you want to share your printer with other computers on a Network.

Advanced

All the Advanced settings should stay default and are Microsoft Unidriver specifics.

Color Management

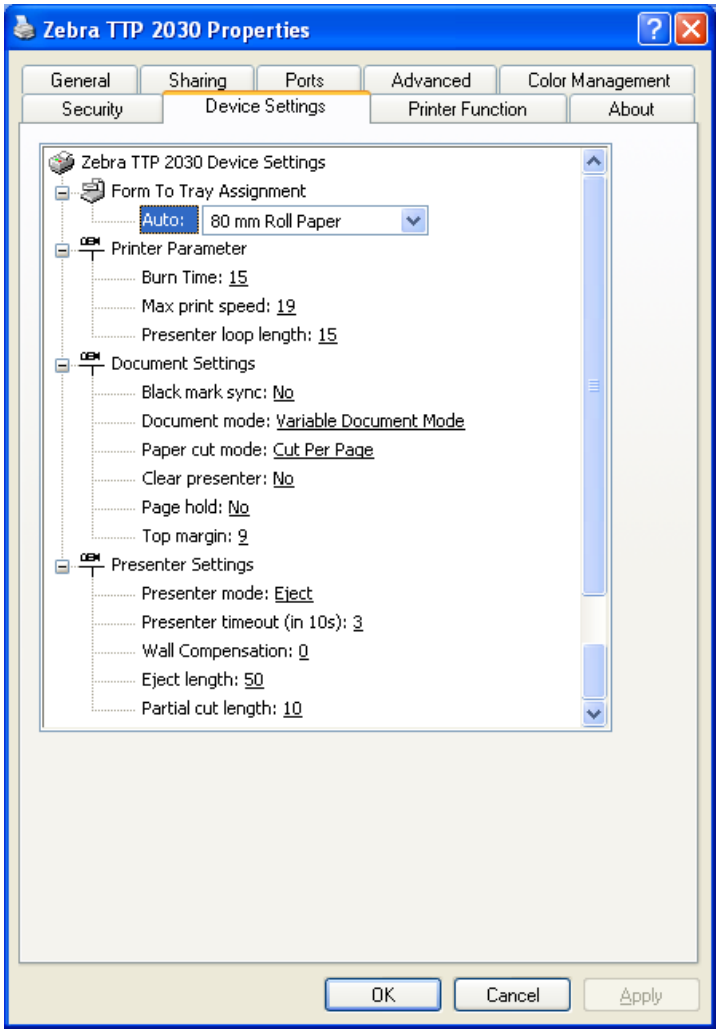
Even though we have two color options in the Printing Preferences, this feature is not supported because the Kiosk printers are not color printers. The color option in Printing Preferences is designed to allow the driver to render color images and create gray scale images to be printed on our Black and White printer.

Security

In the Security tab you can set the access control of specific system users for your printer. In some cases where you need to lock down your user account (e.g. in Kiosk applications) you need to grand the Kiosk user full administrator access to the printer. Typically a “normal” user has only Print rights but in order to get status from the printer the user also needs Manage Printer and Manage Documents permissions.

Device Settings

The Device Settings tab is the first of three OEM settings. All printers share most of the settings with some variances that are described below.



Default settings for the Device Settings by printer.

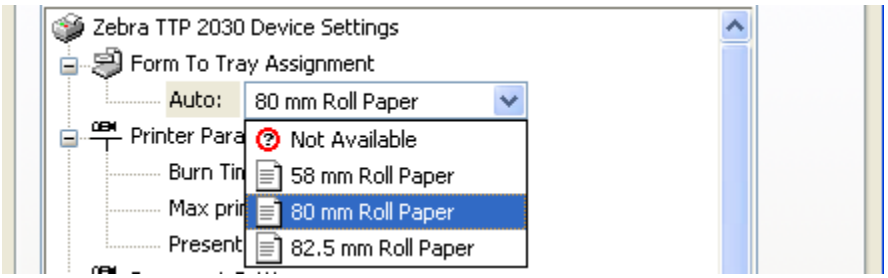
Printer / Setting	TTP 2000	TTP 2100	TTP 7030	TTP 8200
Form to Tray Ass.	80 mm Roll Paper	80 mm Ticket	112 mm Roll Paper	Letter
Burn Time	15	15	15	15
Max print speed	19	17	17	19
Presenter loop length	15	N/A	15	15
Black mark sync	No	Yes	No	No
Document mode	Variable Document Mode	Fixed Document Mode	Variable Document Mode	Variable Document Mode
Paper cut mode	Cut Per Page	Cut Per Page	Cut Per Page	Cut Per Page
Clear Presenter	No	No	No	No
Page hold	No	No	No	No
Top margin	9	9	14	19
Presenter mode	Eject	N/A	Eject	Eject
Presenter timeout	0	N/A	0	0
Wall Compensation	0	0	0	0
Partial cut length	0	0	N/A	N/A

We will now discuss all properties in detail and highlight specific printer differences as they apply.

Form To Tray Assignment

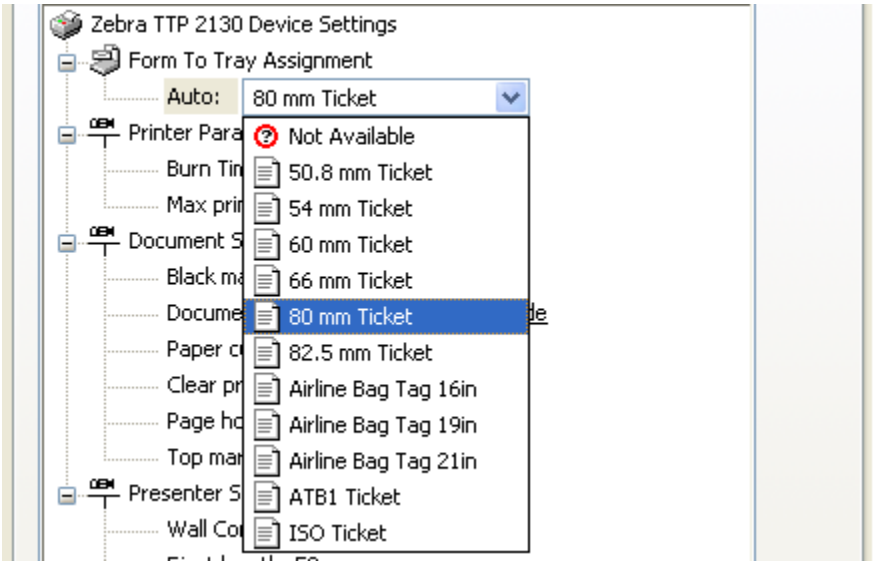
This selection shows the currently selected paper form. You can select from a variety of paper forms and custom forms generated in Server Properties (See Appendix D for more information). This form should be set to the same form as in the Printing Preferences.

The following forms are available for the TTP 2000:



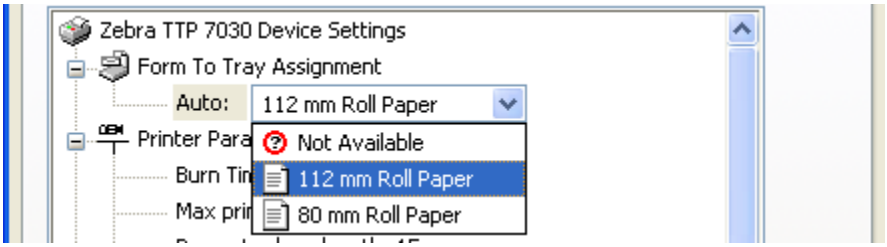
The pre-defined forms have a length of 40.64 cm or 16 inch.

The following forms are available for the TTP 2100:



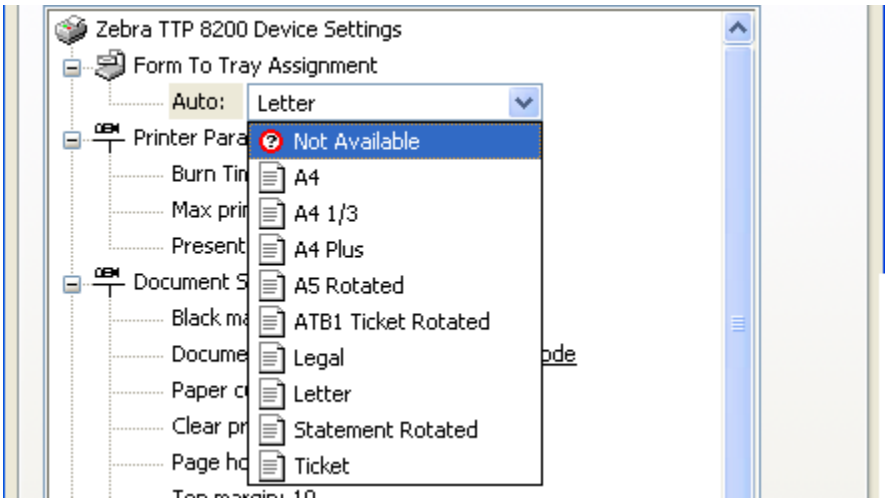
The pre-defined forms have a length of 15 cm or 5.91 inch or the specific form length of the bag tag or ticket.

The following forms are available for the TTP 7030 :



The pre-defined forms have a length of 40.6 cm or 16 inch.

The following forms are available for the TTP 8200 :



The pre-defined forms have a length of the specific form length

Printer Parameter

Burn Time

The Burn Time setting affects the Printer Parameter 7 each time a print is issued. The possible values are between 1 and 15.

Note: See the Technical Manual for more information on this setting.

Max print speed

The Max print speed setting affects the Printer Parameter 8 each time a print is issued. The possible values are 1 to 19 for a TTP 2000, 1 to 17 for a TTP 2100, 1 to 17 for a TTP 7030 and 1 to 19 for a TTP 8200. In many cases it is not suitable to set the print speed below a value of 10 due to resonance issues of the printer in the Kiosk environment.

Note: See the Technical Manual for more information on this setting.

Presenter loop length

The Presenter loop length setting affects the Printer Parameter 9 each time a print is issued. The possible values are 3 to 255. Each step represents a 3.2 cm increment. A value of 0 will disable the presenter loop.

Note: See the Technical Manual for more information on this setting.

Document Settings

Black mark sync

The Black mark sync setting affects the Printer Parameter 35 each time a print is issued. The possible values are 0 = no synchronization and 1 = cut synchronization with TOF marks. This setting is used for synchronizing the printer with TOF marks on the media. For all printers but the TTP 2100 this parameter is set to “No” as a default.

Note: See the Technical Manual for more information on this setting.

Note: Before using a printer with Black Mark media you need to calibrate the printer with the respective media. Please see the Technical Manual for calibration instructions.

Note: The TTP 8200 has no parameter 35 so this setting will affect parameter 36 instead.

Document mode

The Document mode setting is a driver only setting and does not affect the Printer Parameter 36. The possible values are 0 for fixed document mode and 1 for variable document mode. In fixed document mode the driver will print a full page according to the page length definition. In variable mode the driver will omit sending any white space at the end of a document and will issue a cut as soon the print image has ended. If you want to print a specific form you need to synchronize the print with a TOF mark.

Paper cut mode

The Paper cut mode setting is a driver only setting and does not affect any Printer Parameters. The possible values are “No Cut”, “Cut Per Page” and “Cut Per Document”. If the setting is set to “No Cut”, the driver will not issue any cut commands and paper will be fed through the presenter until a cut is issued. (E.g. pressing the Feed button.) If set to “Cut Per Page” the driver will issue a cut command after every page of a document. If set to “Cut Per Document” the driver will only issue a cut at the end of a document.

Note: If you are printing a multipage document with the setting “Cut Per Document” you will get one long printout without a separation between each page.

Note: If you are printing a multipage document with setting “Cut Per Page” each page will be ejected with an ENQ (Clear Presenter) command after a cut if ClearPresenter is set to “Yes” or eject / retract before the driver sends a next page to the printer depending on the selected Presenter mode setting.

Note: Use the “Cut Per Document” in connection with the “Partial Cut” setting to enable a document to be cut partially between pages and full at the end of a document. (This feature is only available in the TTP 2000 and TTP 2100 family of printers.)

Clear presenter

The Clear presenter setting has two possible values and will issue an ENQ (Clear Presenter) command if it is set to “Yes” or do nothing if it is set to “No” after the printer has cut and ejected a page. You can use this feature to fully eject a page from the presenter after it is printed.

Page hold

The Page hold setting has two possible values and the driver will hold a page in the presenter when printing a multipage document if the setting is set to “Yes”. Pages will not be held if the setting is set to “No”.

Note: This feature only works if the “Enable bidirectional support” box is checked in the Ports tab and the Language Monitor is running.

Note: This feature works together with the Presenter mode and timeout setting. If you don’t allow retraction by setting the Presenter timeout value to 0 the print process will hang until the current page is taken out of the presenter because the driver will not send any new pages until the presenter has been cleared. If you allow retraction and the current page retracts due to the timeout period expired while in hold mode the driver will terminate the current print and no further pages print.

Note: Page Hold will not work when printing parameter sheets from the Printer Function tab.

Note: Print jobs are held when the "Delete Print Job on Error" box is unchecked.
When printing one document multiple times it will still be looked at as one document in the print queue and deleted on error.

Example of Page Hold: You want to print a multipage document and have Presenter mode set to “Eject”, Presenter timeout to 30 seconds (“3”) and the Page hold option to “Yes”.

When printing your document and taking every page out of the presenter before the timeout period expires, the driver will send each following page until the document is fully printed.

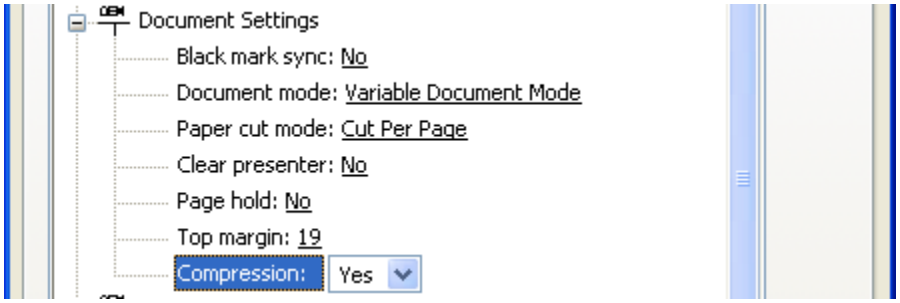
If printing your document and not taking a page and the timeout period expires the printer will retract this page and clear the presenter and also send an error code to the driver indicating that a “retract” has occurred. The driver will then stop printing and delete the current print job.

Top Margin

The Top margin setting is affecting the physical distance between the top of the paper and the cutter. Due to the mechanical design, the printer will always have a top margin depending on the printer family. This distance between the cutter and the print head can be reduced by reversing the paper. The value entered in this setting will determine the amount the printer has to reverse paper. (See the Technical Manual description of the ESC j command.)

Note: The physical distance for a TTP 2000 is 9 mm, for a TTP 2100 is 9 mm, for a TTP 7030 is 14 mm and for a TTP 8200 is 19 mm.

Compression



This setting is a TTP 8200 only setting and affects the way the driver is sending data to the printer. The TTP 8200 has the capability to receive compressed data lines. For more information about this feature please see the Technical Manual.

Presenter Settings

Presenter mode

The Presenter mode setting along with the Presenter timeout controls the Printer Parameter 45. The two possible values are “Eject” (as default) and “Retract”. This takes effect when a new page is printed. (See Technical Manual for further information on the Parameter 45.)

Note: This setting is not present for a TTP 2100 and can only be used with a special retract option added to a TTP 7030 printer. (See the Technical Manual for more information about retract options for the TTP 7030.)

Presenter timeout

The Presenter timeout setting along with the Presenter mode setting controls the Printer Parameter 45. The values entered range from 0 to 30 and represent timeout delays in 10 second steps. (E.g. a value of 3 is a 30 second timeout before the page will be retracted into the Waste bin.)

Note: Setting this value to 0 will keep the receipt in the presenter until the kiosk user takes the receipt.

Wall Compensation

The Wall Compensation setting affects the Printer Parameter 47 each time a print is issued. The possible values are between 0 and 255 and represent the amount the paper is fed in addition to the Eject length setting. (See the Technical Manual description of the Wall Compensation parameter.)

Eject length

The Eject length setting is affecting the physical length a ticket or receipt is ejected out of the presenter after a cut. The possible values are between 1 and 255 and represent the amount of media ejected in mm. (See the Technical Manual for more information on the ESC FF command.)

Partial cut length

The partial cut length setting affects the Printer Parameter 60 each time a print is issued. The possible values are between 1 and 40. You need to set this value according to your print width of your printer. (See the Technical Manual for more information on the Parameter 60.)

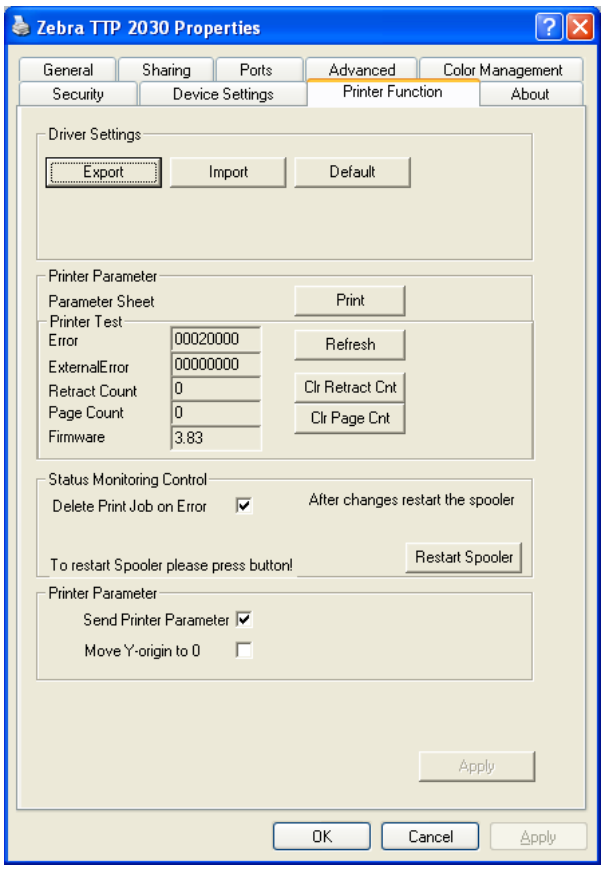
Note: This setting is only available on TTP 2000 and TTP 2100 printer families.

Note: Use the “Cut Per Document” in connection with the “Partial Cut” setting to enable a document to be cut partially between pages and full at the end of a document.

Note: “Partial Cut” can’t be used together with the “ClearPresenter” property.

Example: You have a two page receipt that should be cut partially between the first and the second page. You are using an 80 mm paper and decide to cut 10 mm each side into the paper. You need to set the Paper cut mode to “Cut Per Document” to indicate to the driver that you want only one full cut at the end of the document. Then you select a Partial cut length of 10 allowing the printer to cut 10 mm into each side of the paper. When you print your document the printer will print the first page do a partial cut and print the second page and will do another partial cut followed by a full cut. This is an expected behavior since neither the driver nor the printer knows the end of the document.

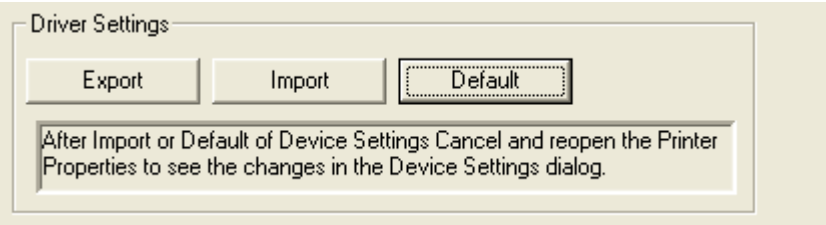
Printer Function



Device Settings

In the Device Settings section you can set all your Device Settings parameter to default by clicking on the "Default" button.

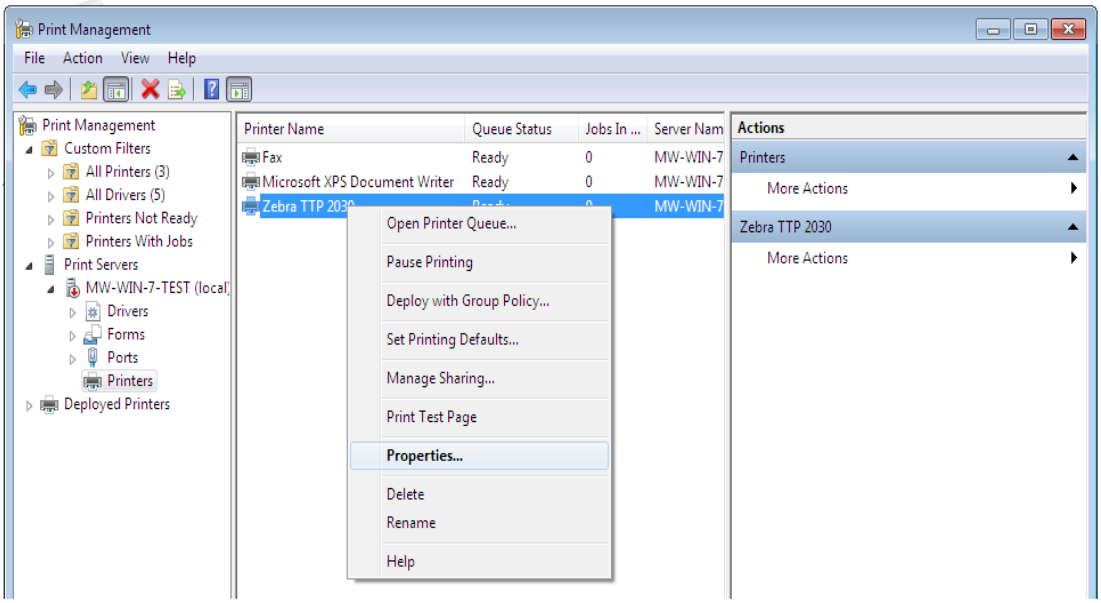
Note: If you default the Device Settings you need to exit the Printer Properties dialog with "Cancel" and reopen it to see the changes in the Device Settings tab. The values will not automatically refreshed when you switch into the Device Setting tab.



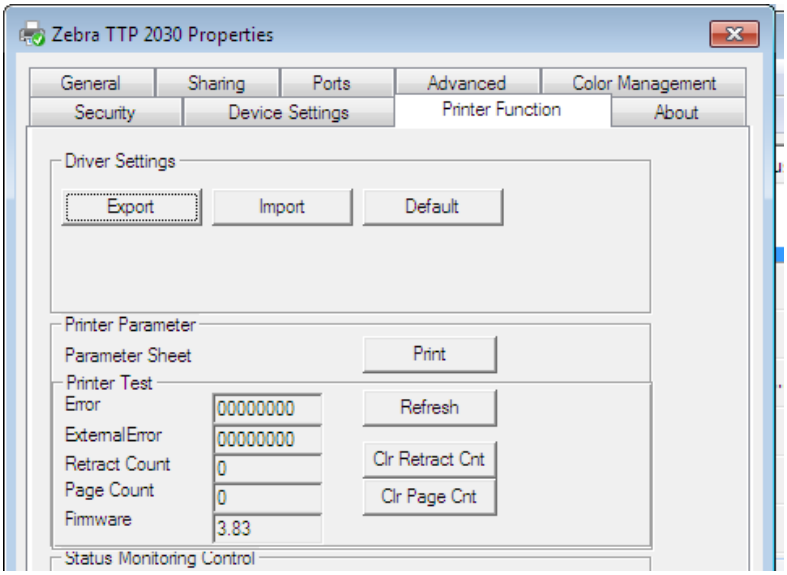
With the "Export" and "Import" buttons you can save and reload the current Device Settings into a XML file that will be saved in the "C:\Zebra" directory.

Note: If you Import the Device Settings you need to exit the Printer Property dialog with "Cancel" and reopen it to see the changes in the Device Settings tab. The values will not automatically refresh when you switch into the Device Setting tab.

Note: In Windows 7 you need to start the Print Management Console as an Administrator and select the printer you want to export / import or default the Device Properties from.



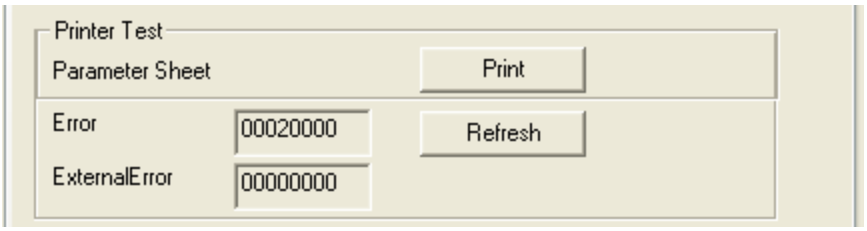
Select Properties and Export the settings. You will find an XML file in the directory c:\zebra with the printer model as part of the name. (E.g. TTP2000Properties.XML)



Printer Test

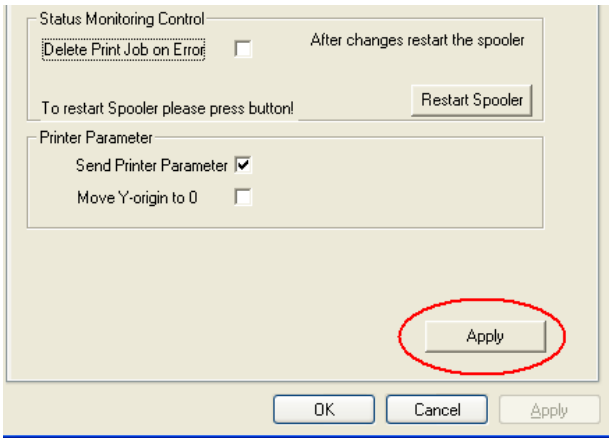
You can print a Printer Parameter Sheet by clicking on the “Print” button.

When clicking on the “Refresh” button the Error values in the Error and External Error field will be updated with current status values. See Appendix A for a reference of the displayed values.



Status Monitoring Control

The driver will delete the current and all new print jobs when the printer is in an error state if the box "Delete Print Job on Error" is checked. If the checkbox is unchecked all print jobs should be held and after the printer returns to normal state reprinted. If you make a change here you need to restart the spooler by clicking the "Restart Spooler" button.Note: Click the "Apply" button for the changes to take effect.



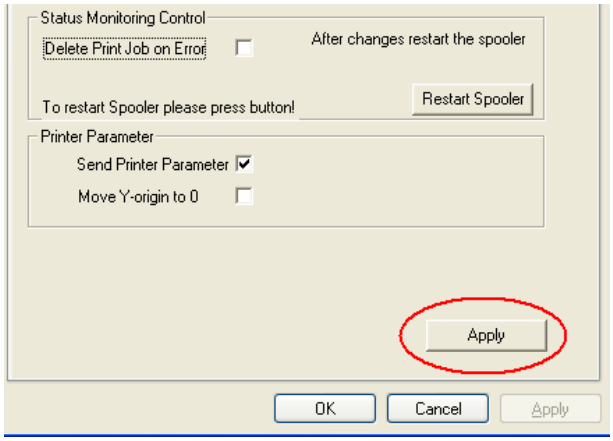
Printer Parameter

If the checkbox "Send Printer Parameter" is checked the driver will send the settings from the Device Settings dialog to the printer. If you want to set your Printer Parameters manually without the driver overwriting the parameters you can un-check this checkbox.

If you check the "Move Y-origin to 0" checkbox the driver will move the top margin of the printed image to the current print line location so the printer is able to print the full image without cutting off the top portion of the image.

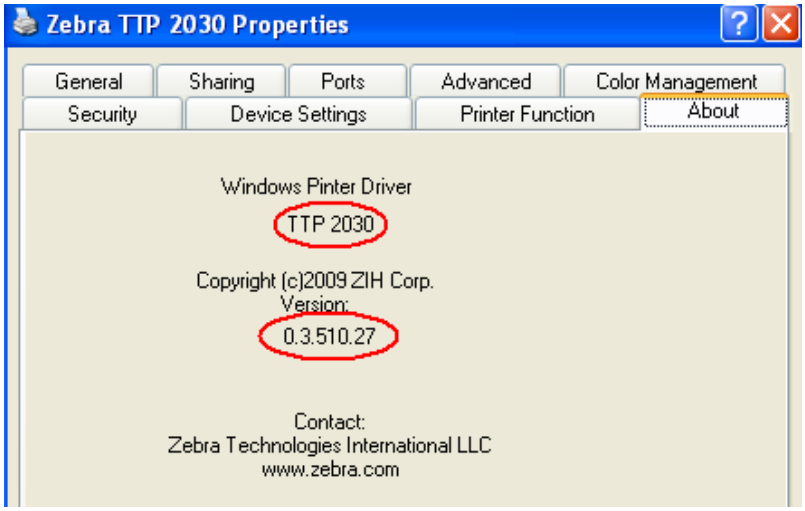
Note: This will not affect the mechanical top margin of the printer, only the Top margin setting in the Device Settings dialog will reverse the media physically.

Note: Click the “Apply” button for the changes to take effect.



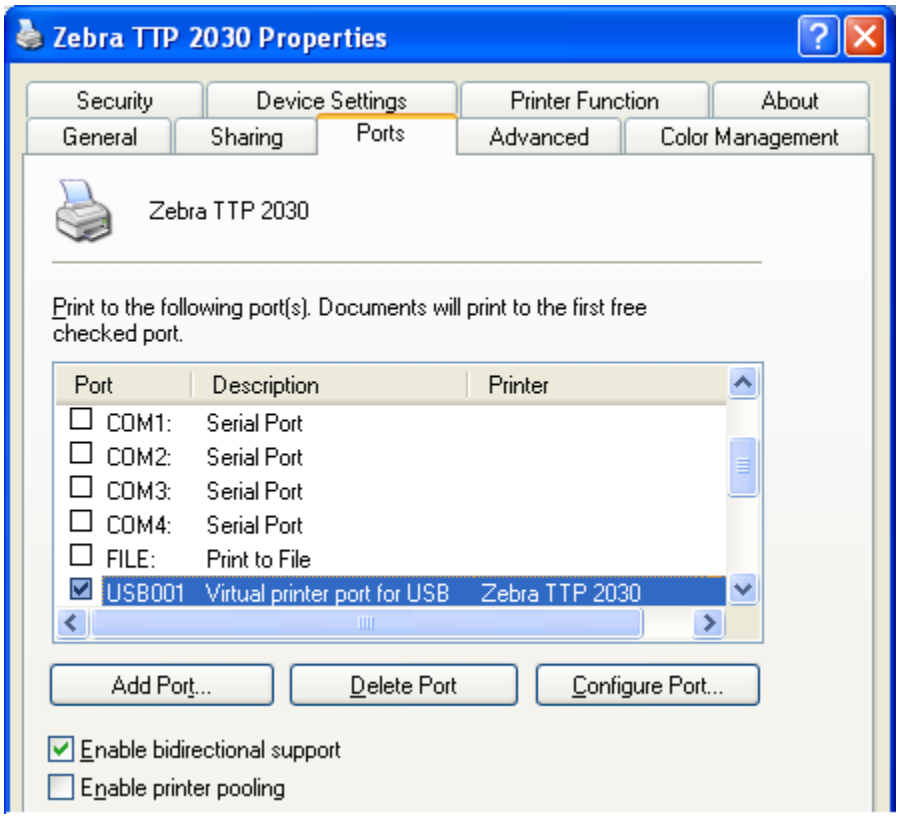
About

The About dialog shows the printer model and the driver version.



Ports

The Ports dialog shows the selected port the printer is connected to. This dialog is the same for all printer families.



The checkbox “Enable bidirectional support” controls the functionality of the Language Monitor.

If you don’t want status responses read from the printer you can disable this function by un-checking this checkbox.

Note: If you are uploading firmware via the Zebra Toolbox program you need to ensure that the “Enable bidirectional support” is unchecked and the spooler is restarted. If you don’t restart the spooler the change will not take effect!

The Language Monitor

The Language Monitor is part of the Windows driver and is located between the Driver UI and the Port Monitor, which takes care of the direct communication with the selected port.
The Zebra Language Monitor has a Windows API interface through the GetPrinterData and GetPrinter functions. It also offers event notification when status of the printer changes.
All the default Windows status responses can also be scripted with WMI scripts.

Following we will call the Language Monitor just LM.

See description and programming example in Appendix C.

Windows APIs for Communication with the Printer

In order to make bi-directional communication easier and also compatible to more than one printer of the same kind on a specific PC, we implemented the LM GetPrinterData function. This is a Windows API described in the Windows documentation. To retrieve immediate printer status from the Spooler you can also use the GetPrinter function. The GetPrinterData function is preferred over GetPrinter due to the fact that with GetPrinterData, all statuses and errors display, while with GetPrinter, only printer errors display.

GetPrinterData

The GetPrinterData function retrieves configuration data for the specified printer or print server. See the Microsoft documentation (http://msdn.microsoft.com/en-us/library/dd144912(VS.85).aspx) for more information on how to use this function.

Note: See Appendix B for available key words.

Note: You can set any of the key values with new entries using the SetPrinterData function. See the Microsoft documentation (http://msdn.microsoft.com/en-us/library/dd145083(VS.85).aspx) for more information on how to use this function.

GetPrinter

The GetPrinter function retrieves information about the specified printer. See the Microsoft documentation (http://msdn.microsoft.com/en-us/library/dd144911(VS.85).aspx) for more information on how to use this function.

Note: Zebra Printer status: It is recommended to use the PRINTER_INFO_3 structure to inquire for the printer status presented by the LM.

Note: The spooler status is changed by SetPort. When using SetPort with custom messages, you can't set these to be displayed or used by the spooler. This is a known bug; "SetPort doesn't work with custom status messages." (Microsoft)

Therefore, all custom messages will be declared as PRINTER_STATUS_ERROR and a KPL value is placed in the ExternalError key. The custom messages are only accessible through the GetPrinterData function.

Event Notification

In order to eliminate the need for a program to continuously poll the printer for status, Zebra implemented an Event notification in the LM.
This notification, used together with the WaitForMultipleObjects Windows API function, enables applications to react on status changes rather than looking for status periodically.
When the internal polling thread recognizes a status change or error then it will fire an event, either an error or a status event.
The Application can open an event object to the LM events and initialize the "Wait for event" function.
The necessary event names can be extracted from the registry with the GetPrinterData API function.
When an event occurs, call the GetPrinterData function and you get the error or status condition returned.
See Appendix B for the keywords.
See the Appendix C for a programming sample.

Status update in Windows "Printers and Faxes" or "Devices and Printer"

In the case that the printer is not printing the status will be checked every 10 seconds (depending on the setting of the READ_THREAD_IDLE_SLEEP key in the LM registry setting). During printing and on error the status will be checked more frequently.

The same status that can be gathered with the GetPrinterData or GetPrinter API will be displayed in the Printer folder.

Note: In some cases it may be possible that the PnP ping is not properly executed on the system and therefore the idle thread of the LM is not activated after a power off situation of the printer. In this case the LM will be reactivated the next time a print job is executed.

Appendix A

Windows Compatible Status

These statuses will also be stored in the printer ERROR key in the registry and can be extracted with GetPrinterData.

Statuses Defined in winspool.h

Table 4 • Windows Status
Windows Status Compares to Zebra Status

PRINTER_STATUS_PAPER_JAM	Paper jam (ESC ENQ 1 = NAK 1)
PRINTER_STATUS_USER_INTERVENTION	Cutter not home (ESC ENQ 1 = NAK 2)
PRINTER_STATUS_PAPER_OUT	Out of paper (ESC ENQ 1 = NAK 3)
PRINTER_STATUS_DOOR_OPEN	Print head lifted (ESC ENQ 1 = NAK 4)
PRINTER_STATUS_PAPER_PROBLEM	Paper feed problem (ESC ENQ 1 = NAK 5)
PRINTER_STATUS_NOT_AVAILABLE	Temperature error (ESC ENQ 1 = NAK 6)
PRINTER_STATUS_ERROR	Presenter jam (ESC ENQ 1 =NAK 7), check ExternalError

PRINTER_STATUS_NOT_AVAILABLE	Retract jam (ESC ENQ 1 = NAK 8), check ExternalError
PRINTER_STATUS_NOT_AVAILABLE	Black mark not found (ESC ENQ 1 = NAK 10), check ExternalError
PRINTER_STATUS_NOT_AVAILABLE	Black mark calibration error (ESC ENQ 1 = NAK 11), check ExternalError
PRINTER_STATUS_NOT_AVAILABLE	Index error (ESC ENQ 1 = NAK 12), check ExternalError
PRINTER_STATUS_NOT_AVAILABLE	Checksum error (ESC ENQ 1 = NAK 13), check ExternalError
PRINTER_STATUS_NOT_AVAILABLE	Wrong firmware (ESC ENQ 1 = NAK 14), check ExternalError
PRINTER_STATUS_NOT_AVAILABLE	Retract occurred (ESC ENQ 1 = NAK 16), check ExternalError
PRINTER_STATUS_NOT_AVAILABLE	Paused (ESC ENQ 1 = NAK 17), check ExternalError
PRINTER_STATUS_TONER_LOW	Paper near end (ESC ENQ 6)
PRINTER_STATUS_NO_TONER	Weekend paper status (ESC ENQ 6) (only for TTP 7030 and TTP 8200 with special hardware)
PRINTER_STATUS_OUTPUT_BIN_FULL	Paper in presenter (ESC ENQ 6)

Table 5 • Status definition in Winspool.h

#define PRINTER_STATUS_ERROR	0x00000002
#define PRINTER_STATUS_PAPER_JAM	0x00000008
#define PRINTER_STATUS_PAPER_OUT	0x00000010
#define PRINTER_STATUS_PAPER_PROBLEM	0x00000040
#define PRINTER_STATUS_OFFLINE	0x00000080
#define PRINTER_STATUS_OUTPUT_BIN_FULL	0x00000800
#define PRINTER_STATUS_NOT_AVAILABLE	0x00001000
#define PRINTER_STATUS_TONER_LOW	0x00020000
#define PRINTER_STATUS_NO_TONER	0x00040000
#define PRINTER_STATUS_USER_INTERVENTION	0x00100000
#define PRINTER_STATUS_DOOR_OPEN	0x00400000

Note: In order to indicate the Kiosk printer status of Paper-near-end or Weekend-paper-status Zebra is using two for thermal Kiosk printer's unused Microsoft status codes. The codes used are PRINTER_STATUS_TONER_LOW for Paper-near-end and PRINTER_STATUS_NO_TONER for Weekend-paper-status. The Weekend-paper-status is only present with printers that have the option of two sensors on their roll holder. (See the Technical Manual for your printer for more information on the available options.)

Windows Incompatible Status

If a printer status doesn't have a corresponding Windows status the Error key will have PRINTER_STATUS_NOT_AVAILABLE set and you need to evaluate the ExternalError key.

Statuses that have a representation within the Windows status may also have an ESC ENQ 1 NAK value (see Table 4) and will be stored in the printer ExternalError key in the registry and can be extracted with GetPrinterData using the ExternalError key.
For the meanings of these NAK responses see the appropriate Technical Manual for your printer, under the ESC ENQ 1 section.

Note: Any other Windows status may be used in the future, so mask away undefined bits in your application!

Appendix B

Table 8 • GetPrinterData Key Values

Printer	DsMonitor Key Explanation	Type
DeviceID	Printer's device ID string	REG_BINARY
Error	Printer Error or Status in Windows 16-bit format	REG_DWORD
ErrorEvent	Error event name for error event trigger	REG_SZ
ExternalError	Extended status according to Appendix B	REG_DWORD
Firmware	Firmware version	REG_BINARY
PageCount	Page counter for cut pages	REG_DWORD
PCB_REV	Printers PCB revision number	REG_BINARY
PCB_SN	Printers PCB serial number	REG_BINARY
StatusEvent	Status event name for status event trigger	REG_SZ
RetractCount	Retract counter for retracted pages	REG_DWORD
DeleteJob	Flag to delete print jobs on error	REG_DWORD
Head_Temp	Head temperature (ESC ENQ B)	REG_DWORD

Note:

Some of these values are not implemented at time of this release.

Appendix C

Programming example

Background

In order to incorporate the new way of status monitoring you need to get a little bit of background on what happens in a Kiosk when you print and when you should monitor your status.
Status monitoring can be handled in two different ways.
Monitor in the printing application
Monitor in a separate application

When you monitor in your printing application you would commonly look at the printer before sending a print job to see if the printer is OK and then send your print job. After the print job is signaled as being printed you would check status again to see if the printer has any errors or if the paper has been taken, etc.

Monitoring in a separate application usually doesn't allow direct interaction with the printed job so you are trying to poll the printer as often as you can to get most accurate information on what the printer is doing. This is usually a very time consuming task and you have to care for synchronizing with a current print job.

Since the latter example is most commonly used for status monitoring, we have incorporated an event notification into the LM to allow a monitoring application to do other tasks and have a separate thread listening for the printer status or error event change. When this occurs the thread is simply getting the status and reporting this back to the main program or doing any other kind of reporting.

To accommodate this notification for all error and status changes we incorporated two mechanisms in the LM.

1. Monitoring while printing

We implemented status monitoring in the internal printing structure of the LM. When you open a Document, print it and close the Document again the LM will check the printer status before and after printing and will also react to write errors if such occur. Then it will set the printer status and raise the error event.

2. Monitoring while idle

We implemented an internal status thread which polls the printer when it is idle in a predefined cycle and provides changed status information in the same manner. It will set the status and raise an error or status event. Therefore, it is not necessary to implement your own monitoring loop. You can simply wait for an event in your application's idle loop.

Implementation in calling application

1. **Step: Open the Printer**

The first step of your implementation is to open the printer you want to monitor and get the Error event and Status even name.

```
bRet = OpenPrinter(m_csPrinter.GetBuffer(1), &hPrinter, &pd);
…
```

```
if ((dRet = GetPrinterData(hPrinter, "ErrorEventName", &dType, (LPBYTE)cTmp, 100,
&dNeeded))!=ERROR_SUCCESS)
…
if ((dRet = GetPrinterData(hPrinter, "StatusEventName", &dType, (LPBYTE)cTmp, 100,
&dNeeded))!=ERROR_SUCCESS)
…
```

2. **Step: Open the Event Handles**

Then you open the two event handles and fill these handles into a structure you will pass on to the new thread.

```
typedef struct _CStatusThreadInfo
{
        HWND  myHwnd;
        DWORD         dSleepTime;
        HANDLE        hPrinter;
        HANDLE        hError;
        HANDLE        hStatus;
        BOOL    m_hStatusEventKillThread;
} CStatusThreadInfo;
…
if ((cTi.hError = OpenEvent(SYNCHRONIZE, TRUE, m_csErrorEvent))==NULL)
…
if ((cTi.hStatus = OpenEvent(SYNCHRONIZE, TRUE, m_csStatusEvent))==NULL)
```

3. **Step: Start Monitoring**

When all this is done you can start your monitoring thread.

```
m_StatusThread = AfxBeginThread( StatusThreadProc, &cTi, THREAD_PRIORITY_NORMAL,0,0,NULL);
```

Implementation in monitor thread

4. **Step: Fill Event Arrays**

In the monitoring thread you create and fill and array of handles with the error and status event handle.

```
        myHandle[0] = pInfo->hError;
        myHandle[1] = pInfo->hStatus;
```

5. **Step: Start the Waiting Loop**

Then you are ready to start the waiting loop.

```
for ( ; ; )
{
    if (pInfo->m_hStatusEventKillThread)
    {
        OutputDebugStringA("### [Thread msg.] Kill thread...\n");
        pInfo->m_hStatusEventKillThread = FALSE;
        AfxEndThread( 1 );
        return 1;
    }
    if ((dwRet = WaitForMultipleObjects(2, myHandle, FALSE, pInfo->dSleepTime))!=WAIT_FAILED)
    {
        if (dwRet==WAIT_OBJECT_0 || dwRet==WAIT_OBJECT_0+1)
        {
            if ((dwRet = GetPrinterData(hPrinter, "Error", &dType, (LPBYTE)&dwResult, sizeof(dwResult),
&dNeeded))!=ERROR_SUCCESS)
            {
                    sprintf( str, "### [Status Thread error %d] read [%08X]\n", dwRet, dwResult);
                    OutputDebugStringA(str);
            }
            sprintf( str, "### [Status Thread] read [%08X]\n", dwResult);
            OutputDebugStringA(str);
            SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)(str));
            if (dwResult & 0x00000000)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_OK"));
            if (dwResult & PRINTER_STATUS_ERROR)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_ERROR"));
            if (dwResult & PRINTER_STATUS_PENDING_DELETION)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_PENDING_DELETION"));
            if (dwResult & PRINTER_STATUS_PAPER_JAM)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_PAPER_JAM"));
            if (dwResult & PRINTER_STATUS_PAPER_OUT)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_PAPER_OUT"));
            if (dwResult & PRINTER_STATUS_PAPER_PROBLEM)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_PAPER_PROBLEM"));
            if (dwResult & PRINTER_STATUS_OFFLINE)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_OFFLINE"));
            if (dwResult & PRINTER_STATUS_IO_ACTIVE)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_IO_ACTIVE"));
            if (dwResult & PRINTER_STATUS_BUSY)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_BUSY"));
            if (dwResult & PRINTER_STATUS_PRINTING)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_PRINTING"));
            if (dwResult & PRINTER_STATUS_OUTPUT_BIN_FULL)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_OUTPUT_BIN_FULL"));
            if (dwResult & PRINTER_STATUS_PROCESSING)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_PROCESSING"));
            if (dwResult & PRINTER_STATUS_USER_INTERVENTION)
               SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
          (LPARAM)("PRINTER_STATUS_USER_INTERVENTION"));
            if (dwResult & PRINTER_STATUS_DOOR_OPEN)
```

```
            SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
        (LPARAM)("PRINTER_STATUS_DOOR_OPEN"));

          if (dwResult & PRINTER_STATUS_TONER_LOW)
            SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
        (LPARAM)("PRINTER_STATUS_PAPER_NEAR_END"));
          if (dwResult & PRINTER_STATUS_NO_TONER)
            SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
        (LPARAM)("PRINTER_STATUS_PAPER_WEEKEND"));
          if (dwResult & PRINTER_STATUS_NOT_AVAILABLE)
          {
            SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
        (LPARAM)("PRINTER_STATUS_EXTERNAL_ERROR"));
            if ((dwRet = GetPrinterData(hPrinter, "ExternalError", &dType, (LPBYTE)dwResult, sizeof(dwResult),
        &dNeeded))!=ERROR_SUCCESS)
                {
                        sprintf( str, "### [Status Thread error %d] read [%08X]\n", dwRet, dwResult);
                        OutputDebugStringA(str);
                }
                sprintf( str, "### [Status Thread External Error] read [%08X]\n", dwResult);
                OutputDebugStringA(str);
                SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
(LPARAM)(str));
            }
        }
        else
            SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0,
(LPARAM)("Timeout"));
    }
    else
    {
        dwRet = GetLastError();
        sprintf( str, "### Wait function failed! [%d]\n", dwRet);
        OutputDebugStringA(str);
    }
}
```

When an event occurs you need to get the status with GetPrinterData using the “Error” key and decode the result according to the sample or any way you feel necessary. In any case you can send a message or do any form of status reporting you want to do.

WMI script to get basic status

```
' VBScript source code
ttpname=""
strComputer = "."
Set objWMIService = GetObject("winmgmts:" _
   & "{impersonationLevel=impersonate}!\\" & strComputer & "\root\cimv2")
Set wbemObjectSet = objWMIService.ExecQuery("SELECT * FROM Win32_Printer")
For Each wbemObject In wbemObjectSet
   if wbemObject.Default = TRUE then
      ttpname = wbemObject.Caption
          Wscript.Echo  "Printer " & ttpname
          Select Case wbemObject.PrinterStatus
          Case 1
                  strPrinterStatus = "Other"
            strExtendedPrinterStatus  = wbemObject.ExtendedPrinterStatus
              Case 2
            strPrinterStatus = "Unknown"
              Case 3
            strPrinterStatus = "Idle"
              Case 4
```

```
            strPrinterStatus = "Printing"
               Case 5
            strPrinterStatus = "Warmup"
               Case 6
            strPrinterStatus = "Stopped printing"
               Case 7
            strPrinterStatus = "Offline"
      End Select
          Wscript.Echo  "Printer Status: " & strPrinterStatus
      Select Case wbemObject.DetectedErrorState
          Case 0
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Unknown"
          case 1
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Other"
          case 2
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " No Error"
          case 3
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Low Paper"
          case 4
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " No Paper"
          case 5
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Low Toner"
          case 6
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " No Toner"
          case 7
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Door Open"
          case 8
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Jammed"
          case 9
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Offline "
          case 10
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Service Requested"
          case 11
             Wscript.Echo  "DetectedErrorState: " & wbemObject.DetectedErrorState & " Output Bin Full"
      End Select

     Select Case wbemObject.ExtendedDetectedErrorState
          Case 0
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Unknown"
          case 1
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Other"
          case 2
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " No Error"
          case 3
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Low
Paper"
          case 4
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " No Paper"
          case 5
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Low
Toner"
          case 6
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " No Toner"
          case 7
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Door
Open"
          case 8
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Jammed"
          case 9
             Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Service
Requested"
          case 10
```

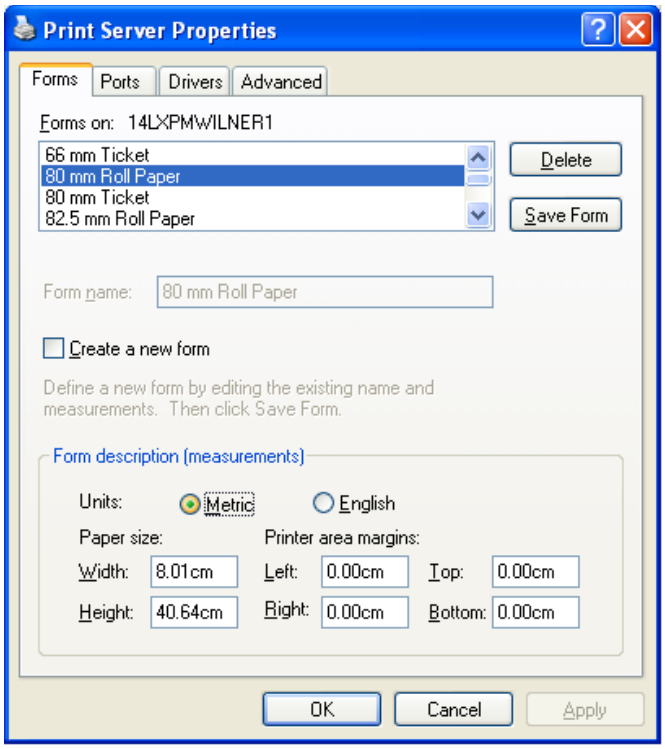
```
            Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Output
Bin Full"
         case 11
            Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Paper
Problem"
         case 12
            Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Cannot
Print Page"
         case 13
            Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " User
Intervantion Required"
         case 14
            Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Out of
Memory"
         case 15
            Wscript.Echo  "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Server
Unknown"
      End Select

   end if
Next
Wscript.Echo  "Printer " & ttpname
```

Appendix D

Setup Printer Forms in Windows XP and Vista

Windows XP and Vista allows you to control global settings for print servers by using the Print Server Properties dialog box. You can access this dialog box by doing the following:

1. Double-click on the printer's icon in the Control Panel or select Settings in the Start menu and then choose the Printers option.
2. In the Printers window, select Server Properties from the File menu.



Use the Forms tab of the Print Server Properties dialog box to view printer forms.

Viewing and Creating Printer Forms

Forms are used by the print server to define the standard sizes for paper, envelopes, and transparencies. To view the current settings for a printer form, follow these steps:

1. Open the Print Server Properties dialog box and then click on the Forms tab as shown above.
2. Use the Forms On list box to select the form you want to view.
3. The form settings are shown in the Measurements area. You can't change or delete the default system forms.

To create a new form, follow these steps:

1. Access the Forms tab of the Print Server Properties dialog box.

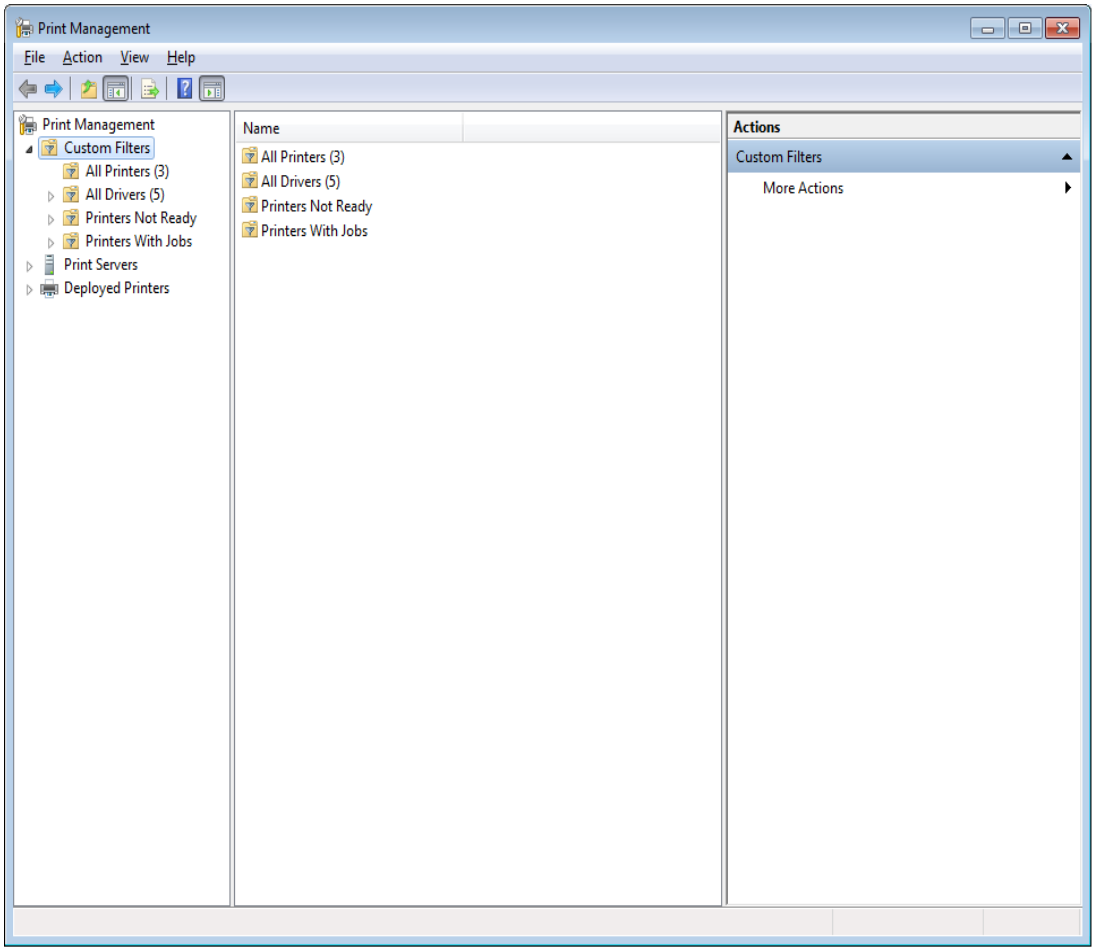
2. Use the Forms On list box to select the existing form on which you want to base the new form.
3. Select the Create A New Form check box.
4. Enter a new name for the Form in the Form Description For field.
5. Use the fields in the Measurements area to set the paper size and margins.
6. Choose the Save Form button to save the form.

Setup Printer Forms in Windows 7

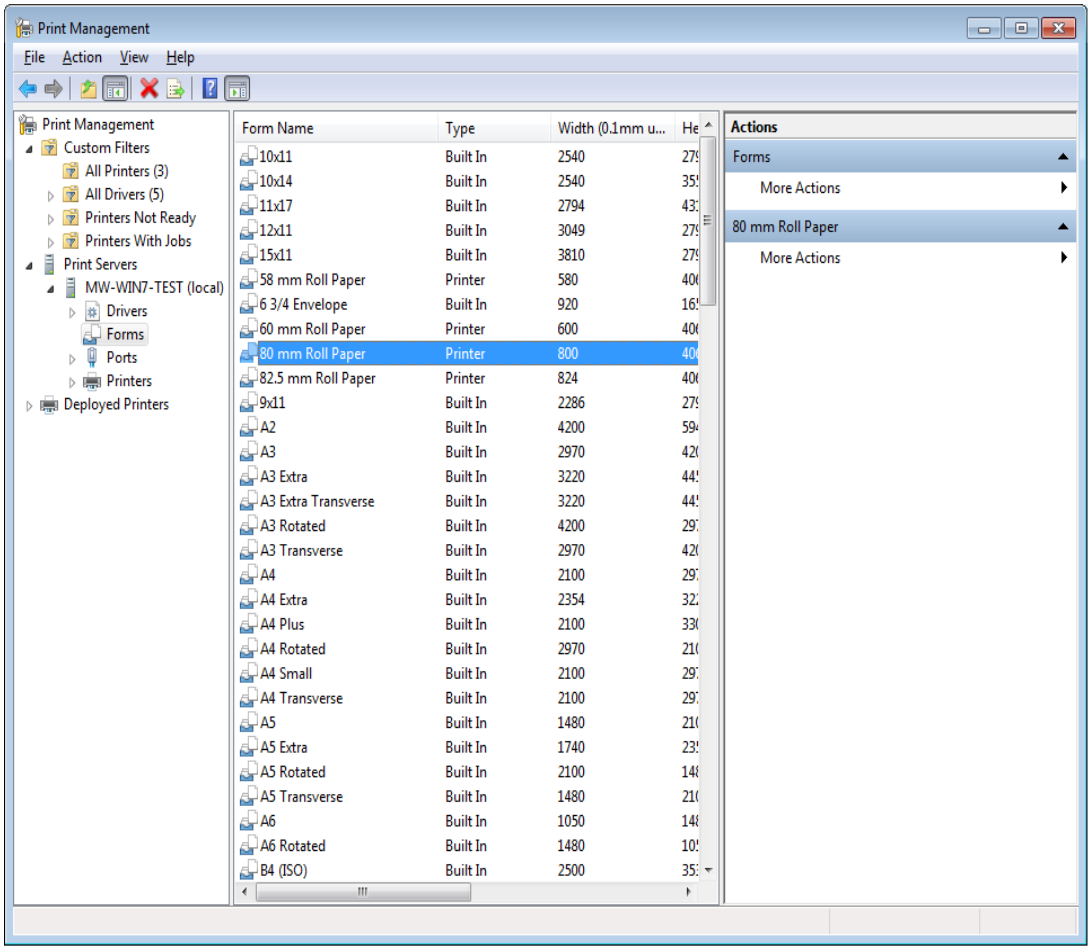
You can use Print Management to manage printer forms.

To manage forms

1. Open Print Management.



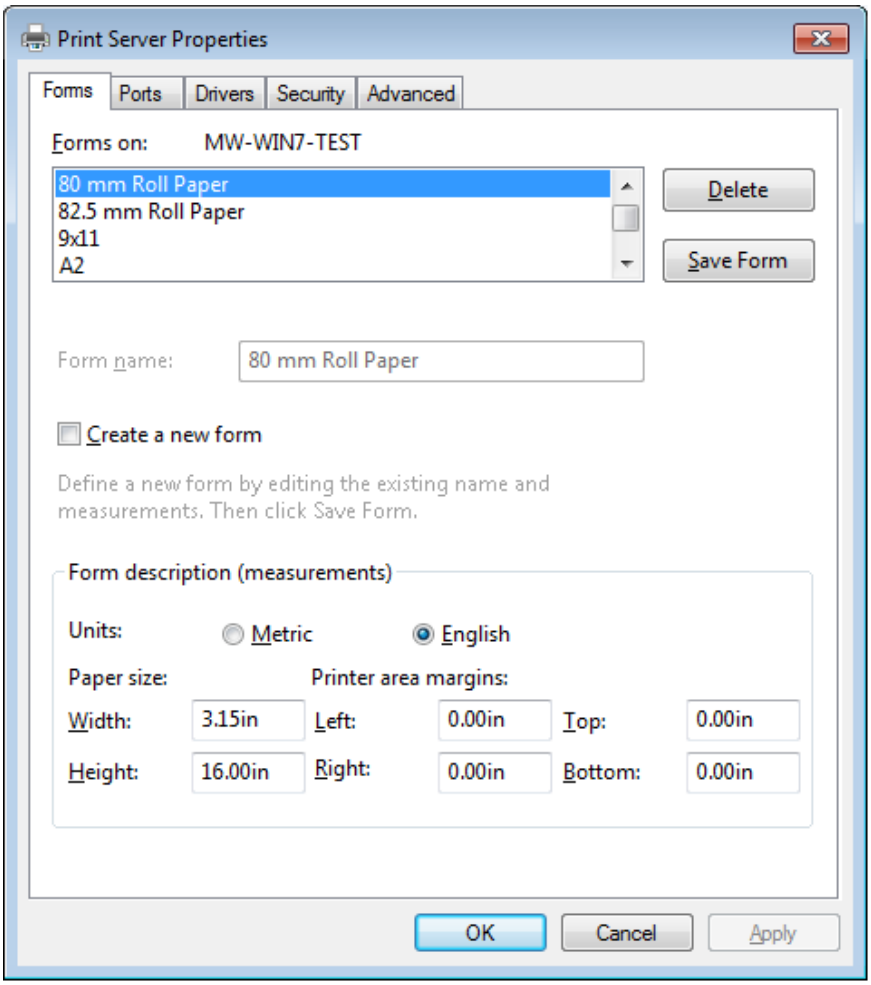
2. In the left pane, click **Print Servers**, click the applicable print server, right-click **Forms**, and then click **Manage Forms**.



3. In the **Printer Server Properties** dialog box, do the following:

    1. To create a new form, select an existing form, select the **Create a new form** check box, change the printer measurement units, paper size, and printer area margins as needed,

click **Save Form**, and then click **OK**.



2. To delete a form, select the form, click **Delete**, and then click **OK**.

Additional considerations

- To open Print Management, click **Start**, point to **Administrative Tools**, and then click **Print Management**.

- You must have Administrative permissions to perform this task.

References

How to find PaperSize for custom print sizes under Windows NT and later versions by using Windows API functions
http://support.microsoft.com/kb/304639

Article ID: 304639 - Last Review: February 2, 2005 - Revision: 4.4

Manage Forms in Windows 7 and Server 2008 R2
http://technet.microsoft.com/en-us/library/dd759110.aspx

Configuring Print Server Properties in Windows XP and Vista
http://technet.microsoft.com/en-us/library/cc722527.aspx

Release Notes

DriverVer=03/30/2010,0.3.510.36

Driver file version=0.3.0.1716

This is the second revision of the new Kiosk Windows driver.

Changes since 0.3.510.34

Fix of bug 395 Printer never comes online after set offline

Fix of bug 393 Eject length doesn't eject enough, also added max eject length to 600 mm

Changes since 0.3.510.33

Fix of bug 393 Eject lengths doesn't eject enough

Fix of bug 391 Spooler crash on delete printer

Known issues

1. Extra page feed when using partial cut
   When printing a 2 page document with partial cut on Black Mark media you will get a partial cut line and cut 9 mm behind the expected cut at the end of a page and one extra page at the end of the document.
2. Found new hardware wizard will stop and ask for a file
   Due to the driver missing a Microsoft signature it may happen after preinstallation of the driver files that during printer installation the hardware wizard stops and asks for a file location. This is the default driver directory where the installer deployed the driver files. (C:\zebra\kiosk\WindowsDriver)