

PrimeZone™

Access Guide
Version 3.60

Image Enhancement/Automatic Zoning Engine

Prime Recognition
PHONE: 650-631-9800
FAX: 650-631-5686
EMAIL: support@primerecognition.com
WEB: www.primerecognition.com
FTP: [ftp.primerecognition.com](ftp://ftp.primerecognition.com)

Copyright © 1996-1999 Prime Recognition. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photo-copying, recording, or otherwise without the prior written permission of the publishers.

Printed in the United States of America.

Statement of Limited Warranty

Prime Recognition warrants to the original licensee of this program that it conforms to Prime Recognition's specifications. Should this program, in Prime Recognition's option, malfunction due to non-conformity with Prime Recognition's specifications, Prime Recognition will, at its option, repair, replace, or update the program at no charge, provided that the program has not been subjected to misuse, abuse, accident, disaster or non-Prime Recognition authorized alterations, modification, and/or repairs. In no event shall Prime Recognition be liable for incidental or consequential damages in connection with or arising out of the furnishing, performance, or use of any of these programs. Prime Recognition reserves the right to modify or revise all or part of this document without notice and shall not be responsible for any loss, cost, or damage, including consequential damage caused by reliance on these materials.

US Government Restricted Rights

The documentation and software are provided with restricted rights. Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (C)(1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013.

Trademarks

PrimeZone is a trademark of Prime Recognition.

Other trademarks appearing in this manual are trademarks of their respective companies.

Table of Contents

Preface

Scope of Developer's Guide.....	vii
Contents of Developer's Guide	viii
Other Publications	ix
Documentation Conventions	ix

Overview

Product Architecture.....	1-1
Inputs to PrimeZone	1-2
File based Images	1-2
Image Resolution	1-2
Image Size	1-2
Image Enhancement.....	1-2
Application Specific Data	1-2
Text Orientation	1-2
Outputs From PrimeZone.....	1-3
New Images	1-3
Zones	1-3
Zone limits	1-3

System Requirements

Personal Computer	2-1
Software	2-1

Installation

Software Installation.....	3-1
PATH.....	3-1
SRCHFILE.EXE.....	3-2
Hardware Key Installation	3-2
Testing Installation Process	3-3
PZSERVER.INI Configuration	3-3
PRPZDLL.INI Configuration	3-3
PRIMAGE.INI Configuration	3-4
Testing the Development Environment	3-4
Distributing the PrimeZone Engine.....	3-6

PrimeZone Server

Overview.....	4-1
Configuration	4-2
[PrimeZone Server]	4-2
Version=2.70.....	4-2
LogFile=0	4-2
[Target Files].....	4-2
ImagePath=e:\images*.tif+	4-2
OverwriteExistingOutput=0.....	4-2
[Image Enhancement1].....	4-2
Deskew=0.....	4-2
RegisterHorizontal=0.....	4-2
RegisterVertical=0	4-2
LineRemovalHorizontal=0.....	4-2
LineRemovalVertical=0.....	4-2
InverseType=0	4-2
DotShading=0	4-2
Despeckle=0	4-2
SubImage=0	4-2
Rotate=0.....	4-2
PeriodRemoval=0.....	4-2
Smooth=0.....	4-2
AutoRotate=0.....	4-2
Crop=0.....	4-2
DialationErosion=0.....	4-3
[Zoning]	4-3
ApplicationType=PHONE_BOOK	4-3
OutputExtension=.ptm	4-3
[Image Enhancement2].....	4-3
Deskew=0.....	4-3
RegisterHorizontal=0.....	4-3
RegisterVertical=0	4-3
LineRemovalHorizontal=0.....	4-3
LineRemovalVertical=0.....	4-3
InverseType=0	4-3
DotShading=0	4-3
Despeckle=0	4-3
SubImage=0	4-3
Rotate=0.....	4-3
PeriodRemoval=1	4-3
Smooth=0.....	4-3
AutoRotate=0.....	4-3
Crop=0.....	4-3

DialationErosion=0.....	4-3
[Save Image].....	4-4
SaveImage=1.....	4-4
ImageExtension=.FIX.....	4-4
Starting Server	4-4
Stopping Server.....	4-4
Error Reporting.....	4-4

PrimeZone Applications

Phone Books.....	5-1
Green Bar (Computer Reports/Tables).....	5-3
Lines	5-6
Columns.....	5-9

Troubleshooting

API Error Reporting.....	6-1
Activity Log.....	6-1
Troubleshooting Suggestions	6-1
API Error Numbers.....	6-2

Contacting Prime Recognition & Warranty/Support

Warranty/Support	7-1
------------------------	-----

Programming to PrimeZone API

Include Files.....	8-1
Libraries.....	8-1
Programming Languages	8-1
API Call Functionality Groups.....	8-2
API Call Sequence.....	8-3
API Call Functionality.....	8-4
Error reporting	8-4
API Calls.....	8-5
pz_open()	8-6
pz_close().....	8-7
pz_select_application()	8-8
pz_read_image()	8-9
pz_enhance_image()	8-11
pz_save_image()	8-13
pz_process_image()	8-14
pz_convert_output_to_file().....	8-15

pz_free_memory().....	8-16
-----------------------	------

Example Applications

Functionality	9-1
Files	9-2
Example Source Code.....	9-2
Visual Basic Source Code Sample.....	9-2

Preface

Scope of Access Guide

The PrimeZone Access Guide provides information on the installation and use of the PrimeZone Server and the installation and use of the Developer's Toolkit and the Application Program Interface (API) to the PrimeZone. It is intended for users of the PrimeZone Server and/or developers who will write programs that interface to, and control, the PrimeZone engine.

In preparing this Guide, we assume that you have some experience with:

- basic OCR terminology

and for developers:

- interfacing to Windows based DLL (Dynamic Link Libraries)

- basic Windows environment programming

Prime Recognition also offers other products, such as PrimeOCR, and PrimeProof. PrimeOCR is the leading high accuracy OCR engine. PrimeProof is a series of Windows based tools which "front end" and "back end" the PrimeOCR engine. PrimeProof functionality includes image display, OCR zoning, post OCR data verification, and OCR "job" creation. This guide does not document PrimeOCR, or PrimeProof.

Contents of Access Guide

- **Chapter 1, Overview**
This chapter, gives a brief overview of the PrimeZone engine, describing its form, its features, and the facilities available to the user/developer.
- **Chapter 2, System Requirements**
This chapter provides detailed descriptions of the hardware and software required to support the PrimeZone engine.
- **Chapter 3, Installation**
This chapter provides instructions and procedures for installing PrimeZone including the Prime Recognition hardware key.
- **Chapter 4, PrimeZone Server**
This chapter describes the functionality, configuration and operation issues of a prewritten application which allows end users to use the PrimeZone engine without programming.
- **Chapter 5, Supported Applications**
This chapter provides detailed descriptions of the specific applications supported by PrimeZone, including any INI files or other data requirements of application.
- **Chapter 6, Troubleshooting**
This chapter provides suggestions and points that might be helpful in troubleshooting PrimeZone, including all error messages.
- **Chapter 7, Contacting Prime Recognition**
This chapter describes how to contact Prime Recognition for sales or technical support.
- **Chapter 8, Programming to PrimeZone API**
This chapter provides detailed descriptions of the calls that interface to, and control, PrimeZone.
- **Chapter 9, Example Application**
This chapter includes an example application with source code, written in C/C++.


Other Publications

Other publications that may be of interest include:

- *PrimeOCR Access Guide*
- *PrimeOCR data sheet*
- *PrimeProof data sheet*
- *Prime Recognition High Accuracy OCR Cost Justification (White Paper)*
- *Prime Recognition High Accuracy OCR "Cleaner Data" Justification (White Paper)*

Documentation Conventions

The following conventions are used in this *Access Guide*:

Convention	Usage
bold	Bold text indicates emphasis
<i>italic</i>	<p>Italics indicate variable information in command lines and in dialog box entries. Variable information is the appropriate data that you enter <i>in place</i> of the italic entry in the example or instructions.</p> <p>Italics in text also indicate titles of manuals, chapters, or headings.</p>
Courier	Courier is used when presenting messages from the program to the user.
[]	Square brackets indicate optional entries. If an optional entry is desired, enter the information between the brackets. Brackets are also used to denote keystrokes in text, e.g., [F1].
	This icon points out important information.

Chapter 1

Overview

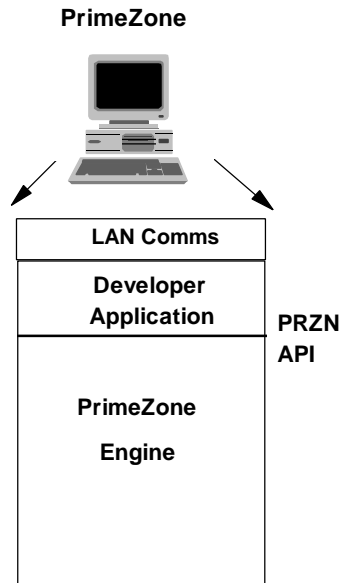
Prime Recognition's "PrimeZone" is a software program that runs on one Win95/Win98 or NT based personal computer. It can modify images, identify "zones" from bit map images created from scanning, FAX, or other electronic means, and produce PDF Image Only output. Unlike conventional zoning programs PrimeZone is typically customized to particular vertical applications such as greenbar reports, phone books, and so on. PrimeZone can act as a "front end" to the PrimeOCR Job Server. It creates ".PTM" files, consisting of engine and zone configuration data, which are read directly by the PrimeOCR Job Server. PrimeZone can also serve as an image enhancement prior to OCR or simply used to convert images into PDF Image Only output.

Product Architecture

PrimeZone is designed as a DLL (Dynamic Link Library) that implements the PrimeZone API. The PrimeZone DLL will load and unload a series of DLLs, as required, that implement Prime Recognition's technology. All required technology is contained within the licensed PrimeZone product and is managed by the PrimeZone engine. No other products must be licensed, integrated, or managed by the developer and his/her code.

However, the PrimeZone engine is not a stand-alone application. A program must be written that interfaces to, and controls, PrimeZone through PrimeZone API calls, which are defined in Chapter 8. A simple example program, written by Prime Recognition, is included in the Toolkit in source code form and is described in Chapter 9. Prime Recognition has also written a more sophisticated application, called PrimeZone Server, which allows you to use the PrimeZone engine with little or no development. If you want to begin using the PrimeZone engine right away, or you do not want to ever program your own PrimeZone interface program then you should evaluate the PrimeZone Server program.

If PrimeZone is used as a server in a LAN environment (a typical production imaging scenario), then the developer's application is responsible for all external communications to the imaging system, including all LAN traffic, except that PrimeZone will read image files from a file server if the image path includes a DOS file mapping to the file server.



Inputs to PrimeZone

File based Images

Image files must be supplied with a DOS filename and path. The PrimeZone engine will automatically find and read the image file header, automatically identify the file type (e.g., TIFF vs. PCX), and read the image into RAM for processing.

Supported file types are:

- TIFF Version 5.0 and above
 - Uncompressed, Modified G3 (FAX), G3, G4, Multipage
- PCX

Image Resolution

Image resolutions of 200, 240, 300, 400, and 600 dpi (dots per inch) are supported. Both height and width resolution must be the same. Resolution is sensed automatically.

FAX Resolutions

Fine resolution FAX images (204 X 196 dpi) can be submitted. FAX images are treated as 200 X 200 images, which leads to a 2% image location error.

Image Size

The overall size of the image is not limited by PrimeZone.

PDF Image Only output is constrained by the PDF Reader, 45 inches by 45 inches.

Image Enhancement

PrimeZone's autozoning functionality is relatively sensitive to image quality, in particular skew. We very strongly recommend that images have skew of less than .5% ("200" as reported by `pz_enhance_image()` call). Use the included deskew and image enhancement features of PrimeZone to clean up the image as much as possible. The PrimeZone engine allows you to save the image after any enhancement for later processing by the PrimeOCR engine.

Application Specific Data

Often applications will have unique data requirements. Most often this data is supplied by an INI file. See *Chapter 5* for a discussion of available application types and their respective INI file/data requirements.

Text Orientation

Text should be in an upright orientation. This is not a requirement of PrimeZone but is a requirement of PrimeOCR.

Outputs From PrimeZone

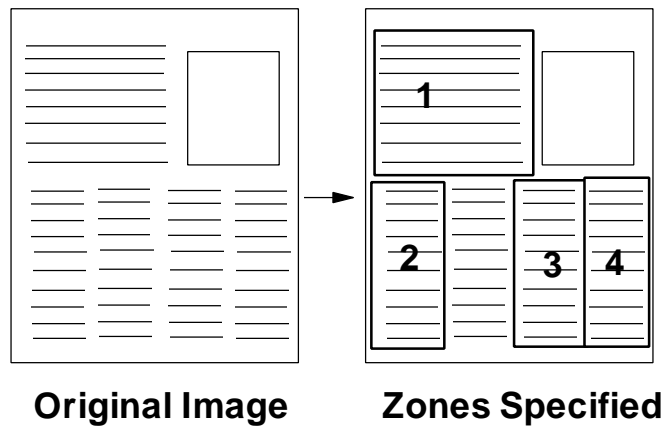
New Images

If desired by the user/developer, PrimeZone can save the enhanced image with the same extension as the original file, overwriting it, or with a new extension (traditionally “.FIX”).

PrimeZone can also generate PDF Image Only files.

Zones

If so selected, the output from PrimeZone can also include the location and size of zones or regions on the image. Zones are rectangles of area on the original image which, ideally, do not overlap. The zones are reported in a "template" file, a format defined by Prime Recognition. A template file includes a number of configuration data besides region location and size. This data is typically hardcoded into an application specific section of the engine or provided via an INI file. It is not automatically determined by PrimeZone.



You may find out more about the "template" file format by reviewing *PrimeOCR Access Guide*.

Zone limits

999 zones per image*

*This can be increased by Prime Recognition on a custom basis. Please call for details.

Chapter 2

System Requirements

The PrimeZone Engine runs on IBM or compatible personal computers. For it to work properly and efficiently, it needs the following hardware and software:

Personal Computer

- IBM PC or 100% compatible computer with 486/Pentium/Pentium II or 100% compatible CPU.
- CD-ROM drive to install software.
- A hard disk with 10 megabytes of available disk space for software installation.
- A Prime Recognition hardware key installed on the PC's printer port.
- 32 megabytes of RAM

Software

- NT 3.51/4.0 or Win95/Win98

Chapter 3

Installation

The installation of the PrimeZone software is composed of two simple steps:

- Installation of the software
- Installation of the hardware key

Software Installation

Run SETUP.EXE from the main directory of the CD, select to install PrimeZone, and follow the directions presented.

The installation program will handle most installation issues automatically, including decompressing all files into a new or existing directory selected by the installer.

The directory will end up with the following subdirectories:

BIN	executables required by PrimeZone engine
DOC	electronic copy of this manual
LIB	libraries required to link to PrimeZone engine
INCLUDE	*.h files for developer application development
SAMPLES	example source files required for developer application development

PATH

If you locate your application in the \BIN directory no PATH modifications are required. If you wish to move your application to another directory:

- The \BIN directory must be added to the PATH statement. Call Prime Recognition if you need help in changing the PATH operating system settings.



Warning

During the use of a development tool such as Visual C++, set the output file name (for the application) to the \BIN directory. Otherwise, if you let the output executable default to \WINDEBUG, for example, then the engine will fail to initialize since the executable is not in the \BIN directory.

SRCHFILE.EXE

A program called SRCHFILE.EXE is also installed to the \BIN directory. If you run into any problems with PrimeZone you should run this program. It searches for file conflicts with previously installed software from Prime Recognition or other vendors.

SRCHFILE.EXE must be run from the same directory as the controlling application to accurately find problem files. It does not search the whole drive, it instead searches for files in the same order that an executable file would try to load files.

If you later load other software onto this PC and PrimeZone begins to experience problems we recommend that you run SRCHFILE.EXE again to see if the new software has caused any file problems.

Hardware Key Installation

The hardware key supplied with the product must be installed on the LPT1 printer port for the PrimeZone engine to initialize and run. Please use the supplied thumb screws to securely attach the key to the port. You may attach a printer cable to the hardware key as all normal LPT1 traffic is passed through.

If you have multiple hardware keys from different vendors (probably on your development system, not your production system) you may have to experiment with the stacking order of the keys, a key usually performs best if it is the first key, i.e., closest to the PC.

The engine will go out periodically to read the hardware key so the key must be present during all engine processing.

If you are receiving hardware key errors from API calls:

- Make sure the key is securely attached to the LPT1 port.
- Remove the printer cable and any other hardware keys, if they are attached to the key.
- Note that only the hardware key supplied with the product will pass the engine's tests.

The installation program will automatically install the correct hardware key software driver. If for some reason you need to install the hardware key driver yourself, run SETUP.EXE from the \RAINB directory on the installation CD. The program will run in the background without any user dialog. Reboot the computer once the drivers have been installed.

Testing Installation Process

After the software and the hardware key has been installed, and the computer has been re-booted, try running the PrimeZone Server application, PRIMEZONE.EXE. After starting the program, verify program settings in the PZSERVER.INI configuration file by selecting “Configure PRZone Server” under the “Log/Configuration Files” menu.. The PZSERVER.INI file defines what images to process, how to process the images and how the output should be saved. If any errors occur then something is probably wrong with:

- The installation.
- The hardware key. See *Hardware Key Installation* section.
- Your environment. Read *Chapter 2* for the minimum system requirements of the PrimeZone engine, and remove all other loaded programs. Run SRCHFILE.EXE to test for software file conflicts.
- Settings in PZSERVER.INI. Ensure that the directory being processed contains images, and other settings are toggled on to generate desired output.

PZSERVER.INI Configuration

A file called “PZSERVER.INI” is located in the \BIN directory. This file controls the configuration of selected items of the PrimeZone server. This file defines what files to process, where they are located, what image enhancements need to be done, if any, what autozoning needs to be completed, if any, and what type of file format should be saved.

This file can be accessed by selecting “Configure PRZone Server” under the “Log/Configuration Files” menu.

To change the setting of an item merely edit the number after the “=”. See *Chapter 4* for a discussion of PZSERVER.INI.

PRPZDLL.INI Configuration

A file called “PRPZDLL.INI” is located in the \BIN directory. This file controls the configuration of selected items of the PrimeZone engine. Many of the configuration items are used by Prime Recognition for remote technical support purposes, these items are not of direct interest to developers.

This file can be accessed by selecting “Configure PRZone Engine” under the “Log/Configuration Files” menu.

To change the setting of an item merely edit the number after the “=”. A summary of the data below is included in the PRZNDLL.INI file for quick reference.

ShowProgressWindow=2

0=NO, 1=Independent Window, 2=Window within App window[default]

Change this setting to 0 to disable the display of the progress update window during PrimeZone operation. This saves about 2% of CPU time. Setting "2" places the progress window in a fixed position 330 units "Left" and 0 units "Top". If the parent window is not at least 330 units wide no part of the progress window will be visible. There is no way to change the position of the window except by Prime Recognition customization.

PrintIntermediateResults=0

0=NO, 1=YES - PR use only

Debug=0

0=NO, 1=YES - PR use only

PTMVersion=300

Version of template file format. Legal values are 210, 250, 270,300. We recommend using the latest valid version where possible.

LogFile=0

0=NO, 1=YES

Change this setting to 1 to enable the generation of a text file that logs all major PrimeZone engine activity, including calls made to PrimeZone engine. This file is often helpful to Prime Recognition for remote technical support. The LOG file's default path and filename is "\BIN\PRZONE.LOG".

WhiteSpaceIs0InImage=1

0=NO, 1=YES [Most common]

In most image types "0" means white (and black is "1"). If instead white is "1" in your image types then change this item to 0.

MostSignificantBit=0

0 = PC, 2= UNIX

"0" is the correct setting for PC generated files. If your images came from a UNIX system you may need to change this setting to "2".

[PrimeZoneDLL-Column]**[PrimeZoneDLL-Lines]****[PrimeZoneDLL-PhoneBook]****[PrimeZoneDLL-GREENBAR]**

See *Chapter 5* for the meaning of these settings.

A copy of PRPZDLL.INI file is not required for distribution. Items will default to those settings shown above if PRPZDLL.INI is not present.

PRIMAGE.INI Configuration

A file called "PRIMAGE.INI" is located in the \BIN directory. This file controls the configuration of the image enhancement options of PrimeZone.

This file can be accessed by selecting “Configure Image Processing” under the “Log/Configuration Files” menu.

To change the setting of an item merely edit the number after the “=”.

The settings for these features are exactly those of the ScanFix (TM) image processing engine which is embedded within the PrimeZone engine. If you need the documentation for these settings please review *Chapter 3* of the *PrimeOCR Access Guide* or contact Prime Recognition technical support.

You must restart the engine for any changes in this file to take effect.

Testing the Development Environment

Once the supplied example application, PZEXAM.EXE runs smoothly, then try to compile it from the supplied source code and make files. Some compilation issues are covered in *Chapter 8*. If you are not using Visual C++, you will need to consult the documentation for your programming tools. Errors at this point will most likely be a mismatch between the supplied PrimeZone source code and your development environment.

Again, we very strongly recommend that you do not proceed any further until you are able to get the PrimeZone example code working in your environment. Once a working example of the code is available then problems with your application can be much more easily isolated to the differences between your code and the example application.

We highly recommend that you use our code as a starting point for your development, if possible, and incrementally modify the code, testing at each significant modification. So long as the PrimeZone example code is used in conjunction with our engine, you can modify it as much or as little as you like with no royalties or any other legal hindrance.

Distributing the PrimeZone Engine

Once you have finished developing the application that interfaces to and controls the PrimeZone engine you will want to distribute the PrimeZone engine and your program to your production environment. A license is required for each PC in your production environment.

An installation of a run time PrimeZone engine requires the following setup:

- A \BIN directory with all files that are in the toolkit \BIN directory

- If application is not in \BIN directory then FF20M32L.DLL and FFWIN32.LOK must be in the same directory as the application and the PATH statement must point to \BIN directory.

- A hardware key software driver loaded into the operating system. One way to load the proper driver is to run the SETUP.EXE program found in the \RAINB directory of the installation CD on the target PC.

You may use the standard Prime Recognition installation program to install a run time license. It may generate subdirectories and files that are not required for a run time installation. You may erase these extra files and directories at your option.

Chapter 4

PrimeZone Server

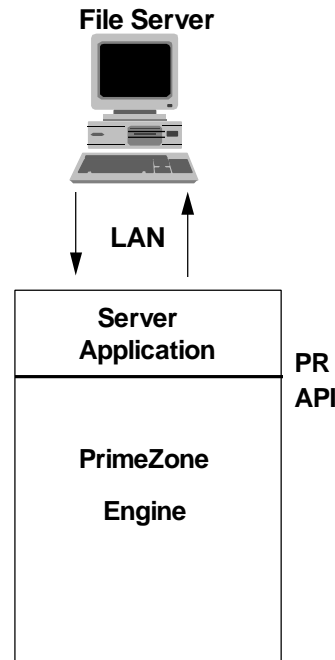
Overview

The PrimeZone Server is an application, written by Prime Recognition, that controls the PrimeZone engine. The PrimeZone Server is designed to read one or more images from a user specified directory, configure and start the PrimeZone engine processing, and then put the PrimeZone results in the same directory. If it encounters a recoverable error it writes the error to an

error file and continues processing.

The input to the PrimeZone Server is image files, its output can include enhanced image files and/or template files, and/or PDF Image Only files. The function of the PrimeZone Server is controlled primarily by a PZSERVER.INI file. The PrimeZone Server reads the PZSERVER.INI file to identify which images to process and how to process those images. The format and contents of the PZSERVER.INI file are described later in this chapter.

Although the images to be processed can be configured in the INI file, this information can also be sent to the engine via a command line. This is an important function as it allows the target images to be configured at run time without modifying an INI file.



Key Features

- Can be used to enhance images prior to zoning, these enhanced images may be saved for later OCR processing.
- Can automatically find zones on images, and generate a template file, that may be used later by PrimeOCR engine.
- Can convert image files to PDF Image Only files.
- Since all input and output is file based the PrimeZone Server can be located anywhere on a LAN/WAN network where it has file access. Since the server's architecture is file based it is not dependent on any specific network protocols.
- Can process all images in a directory by specifying "*.tif" in the image input path.
- Can process all images in subdirectories of a directory by specifying "*.tif+" in the image input path.
- Multiple instances can be run at one time
 - Simplifies management of processing of multiple directories (create an instance per directory)
 - Utilizes multiple processors if any exist

Configuration

All configuration is accomplished through the PZSERVER.INI file, a simple ASCII based text file that may be edited with any ASCII only text editor such as notepad. This file can be accessed by selecting “Configure PRZone Server” under the “Log/Configuration Files” menu.

The contents of this file are as follows:

[PrimeZone Server]

Version=3.60

Identifies the version of INI file. Must match or be compatible with engine version number.

LogFile=0

If a non zero number is placed after the equals sign a file called “PRZONE.LOG” will be created in the directory in which the application is placed. This log file will show the major steps performed by PrimeZone.

ExitOnComplete=0

If non zero then server unloads from memory once batch is complete.

[Target Files]

ImagePath=e:\images*.tif+

Full path to image(s) to be processed. Wildcards in filename and extension are allowed. A “+” appended to the string will cause any subdirectories to also be processed.

OverwriteExistingOutput=0

If not “0”, then if an output file already exists with the OutputExtension noted above (or the ImageExtension if a image processing step only application), this file will be skipped during processing.

[Image Enhancement1]

Place a “1” after “=” to turn on an image enhancement step. The configuration of these steps are in PRIMAGE.INI (except FileCrop, MarginClean, and DeleteZones).

Deskew=0

RegisterHorizontal=0

RegisterVertical=0

LineRemovalHorizontal=0

LineRemovalVertical=0

InverseType=0

DotShading=0

Despeckle=0

SubImage=0

Rotate=0

PeriodRemoval=0

Smooth=0

AutoRotate=0

Crop=0

DialationErosion=0

FilmCrop=0

Assumes an image with 4 solid black borders. Removes black borders and corrects image for any skew present. Developed for a microfiche application but may be used for any image that has same characteristics.

MarginClean=0

Removes long/thin black objects in outeredges of image that are substantially bigger than possible text objects.

DeleteZones=0

If a template exists in the same directory as target image, it will read template and delete areas on image that are reported as zones in template. (The zone type reported by the template is ignored, all zone types will cause erasure).

[Zoning]

ApplicationType=PHONE_BOOK

Valid application types are:

PHONE_BOOK
GREEN_BAR
LINES
COLUMN

Make sure you type in this value precisely with no trailing spaces or tabs. For detailed information on each application type see *Chapter 5*.

OutputExtension=ptm

If the application creates a non image output file it will have this extension. These files are used by PrimeOCR for configuration of zones on an image. Long extensions, i.e. greater than three characters in length, are supported (but can not be easily read by PrimeView).

[Image Enhancement2]

This value is the same as ImageEnhance except processing occurs after any zoning step. For example, in the PHONE_BOOK application PERIOD_REMOVAL should not occur before zoning, it should occur after zoning.

If, for example, no zoning step occurs, you may still use these settings to perform a second iteration of image enhancement on the target image. This enhancement will occur on the image that was already enhanced by the first image enhancement settings (if any).

Deskew=0
RegisterHorizontal=0
RegisterVertical=0
LineRemovalHorizontal=0
LineRemovalVertical=0
InverseType=0
DotShading=0
Despeckle=0
SubImage=0
Rotate=0
PeriodRemoval=1
Smooth=0
AutoRotate=0
Crop=0
DialationErosion=0

[Image Enhancement3]

This value is the same as ImageEnhance except processing occurs after [Image Enhancement3].

Deskew=0

[Save Image]**SaveImage=1**

"0" means do not save the processed image. A value of "1" saves the processed image. A value of "2" saves the image in a PDF Image Only format. This save occurs after any deskew or image enhancement. Deskew and many image enhancement steps physically alter the image layout. Therefore it is highly recommended that the image be saved after any deskew and/or image enhancement steps occur so that the new altered image is available to the PrimeOCR engine in exactly the same form as was used to find the zone data.

ImageExtension=FIX

If SaveImage is not "0", then the processed image is saved with the original file name and path plus this extension. Long extensions, i.e. greater than three characters in length, are supported.

Desample

1-600, dpi of images that will be saved. If blank or 0 then leave dpi of image equal to original. 200 is a common setting if desampling is desired. Lines of pixels will be removed/added to image to achieve the desired DPI. For example, a 400 DPI image, desampled to 200 DPI, will have every other line of pixels removed.

ChangeDPI

1-600, reported dpi of images that will be saved. If blank or 0 then leave dpi of image equal to original. In this setting, unlike Desample, no changes are made to the actual image, only the reported DPI is changed. This setting is useful to change the DPI of image to a value that can be used by DPI sensitive products, such as PrimeOCR.

Starting Server

Clicking on the PrimeZone Server icon initiates the server interface. To start processing, press the "Start" button. To start the server programmatically (no manual intervention) execute PRIMEZONE.EXE. If you supply an image path, (wildcards are allowed in path) then the server will process that image path instead of the image path found in the INI file.

For example,

```
WinExec("primezone.exe START", SW_SHOW);
```

Or from the Run Command:

```
primezone.exe START c:\images\*.tif
```

Note: The "START" variable is case sensitive.

Stopping Server

To stop the server in the middle of a series of images, press the "Stop" button in the server window. If you restart the engine after stopping it in the middle of series it will start processing at the beginning of the image range, overwriting any previous output depending on the "OverwriteExistingOutput" setting in the PZSERVER.INI configuration file.

Progress Reporting

During operation, the server will update the progress of the images being processed. The right window displays the current status of each image being processed. The left side of the updates several statistics during processing. The stats are tabulated by the total files/directories found in the batch, and how many image files have completed processing.

DIRS:	Complete	(number of directories completed in the batch)
	Total	(indicates total number of directories found in the batch)
FILES:	Complete	(indicates total number of files completed processing in all directories – includes files that generated an error)
	Total	(indicates total number of image files found in all directories)
FILES (in Dir):	Complete	(indicates total number of files completed processing in this directory only – includes files that generated an error)
	Total (in DIR)	(indicates total number of image files found in current directory)
ERRORS:	Total	(indicates total number of errors encountered while processing all directories)
TIME:	Elapsed	(indicates total time elapsed since starting batch processing of all files)
	Remaining	(is an estimate of how much time remains to process all files in the batch)
All time is shown in seconds.		

Error Reporting

If an recoverable error is sensed, it will be reported to a file called "PRZONE.LOG". This file details the date, time, image file that caused error, and error number. The engine will then continue processing.

Unrecoverable errors will cause a message box to appear with the error number. These messages are also logged to the error file. Clearing this message box will cause the Server to terminate.

Chapter 5

PrimeZone Applications

PrimeZone is not a generic zoning tool. It must be customized for particular types of applications. This chapter documents the available application types along with any special setup or data required by each application. If you want an application type added please contact Prime Recognition. Not all application types available will be listed in this directory. Proprietary, custom, or application types in development will not show up in this listing.

Phone Books

Application Name:
PHONE_BOOK

Functionality:

This application will define the columns of a phone book image as zones. It will try to cut off text at the top of the columns which is header information, however, note that this feature is very difficult to accomplish and hence is subject to inaccuracies. Bias is given to including header information rather than cutting out, incorrectly, valid data. Footer data is not excluded from zone definition.

Data Assumptions:

Long columns of data with minimal or no header information at top of page.

Columns are of equal width.

Spacing between columns is constant.

Columns have more than 3-4 lines of text.

(We strongly recommend you use the PERIOD_REMOVAL feature of image enhancement, but only after zoning is performed, in other words in [Image Enhancement2])

Data Requirements:

Edit configuration items present in PRPZDLL.INI in \BIN directory. The contents of this file should be as follows:

```
" [PrimeZoneDLL-PhoneBook ]"  
"Width=10"  
"Indent1=50"  
"Indent2=50"  
"Indent3=50"  
"Indent4=50"
```

The meaning of these parameters is as follows:

Width

Approximate minimum width in pixels between columns of text on page across target sample of images. (10 is a good number to start with. Try larger or smaller numbers if program gives bad results and page is out of the ordinary in width.)

IndentX

Distance in pixels on each edge of image that program can ignore in its calculations. This parameter allows algorithms to ignore noise that is often at edge of images causing faster speed and fewer errors. Make sure that parameters never touch valid data, therefore, we recommend these settings be kept very conservative. 1=Left edge, 2= Top edge, 3=Right edge, 4=Bottom edge.

Output:

A file in PTM (template) format. All document, page, and zone information except for zone bounding box is hard coded into the application as follows:

Document Configuration Settings

Output Format	"3"	Prime OCR format (".pro") format
# Pages	"1"	

Page Configuration Settings

Page Number	"1"	First page
Document Source	"1"	"1" means that this page is from the same document as the last page processed by the engine.
Image Quality	"0"	Default quality setting
Print Type	"0"	Machine Print
Language	"10"	U.S. English
Deskew	"0"	Do not perform (perform during PrimeZone if necessary)
Image PreProcess	"0"	Do not perform (perform during PrimeZone if necessary)
AutoZone	"0"	Do not Autozone
Number of Zones	"Y"	Determined by PrimeZone

Zone Description

Zone Number	"X"
-------------	-----

Zone Restrictions	"0"	No restrictions
Lexical Check	"1"	Perform lexical check on output
Accuracy Level	"99999"	Run all five engines
Left Dimension	"X"	These parameters determined by PrimeZone
Top Dimension	"X"	
Right Dimension	"X"	
Bottom Dimension	"X"	

Refer to *PrimeOCR Access Guide* for further information on these settings. Contact Prime Recognition if you would like any of these settings changed or if you want Prime Recognition to add the ability to change these settings through a INI file setting.

Green Bar (Computer Reports/Tables)

Application Name:
GREEN_BAR

Functionality:

This application will find each field of data in a column based computer report or other type of table based image. It can also, at the user's discretion, find fixed fields on the image (fields that are not in columns, such as the title of report, or a date of report field).

Data Assumptions:

Green bar images for the purposes of this function are defined as images with tables of data such as "green bar" computer reports. However, many other types of images fit this definition.

(a)The columns cannot change size or location moving vertically down the page. This means that abnormal lines, for example, a "Totals" line with a different format, will not be recognized correctly. (Contact Prime Recognition for a custom version in this case).

(b.)A unique anchor object can be defined that is always the same distance from the beginning of the columnar data.

(c.)Data in columns is separated from other columns by some space.

(d.) Lines of data are separated from other lines columns by some space (characters from different lines never overlap or touch).

(e.) If blank lines are expected in the column based output the "AllowGaps" setting must be set to 1. Blank lines are exactly the same size as non blank lines.

Data Requirements:

Edit configuration items present in PRPZDLL.INI in \BIN directory. The contents of this file should be as follows:

```
" [PrimeZoneDLL-GREENBAR] "  
"Width=15"  
"Height=1"  
"Indent1=50"  
"Indent2=50"  
"Indent3=50"  
"Indent4=50"  
"MasterTemplate=c:\images\template\avv.ptm"  
"LineHeight=30"  
"LineJump=37"  
"FixedZones=0"  
"Columns=19"  
"AllowGaps=0"
```

The meaning of these parameters is as follows:

Width

Approximate width in pixels of a space character in image divided by 2. (15 is a good number to start with. Try larger or smaller numbers if program gives bad results and page is out of the ordinary in width.)

Height

Approximate width in pixels between lines of text divided by 5. (1 is a good number to start with. Try larger or smaller numbers if program gives bad results and page is out of the ordinary in width.)

IndentX

Distance in pixels on each edge of image that program can ignore in its calculations. This parameter allows algorithms to ignore noise that is often at edge of images causing faster speed and fewer errors. Make sure that parameters never touch valid data, therefore, we recommend these settings be kept very conservative. 1=Left edge, 2= Top edge, 3=Right edge, 4=Bottom edge.

MasterTemplate

Full Path to a template which contains all relevant configuration data. The output template created by PrimeZone will use the configuration data defined in the MasterTemplate. To create a MasterTemplate:

1. Run the PRZONE.EXE program from PrimeProof.

2. Open a typical image (make sure it is typical).
3. Define the anchor zone (it is one of the choices in the Current Zone/Content Restriction box). This should be an object on the image that will always be in the same position relative to the columnar data and will always be the same size. A word, or a picture is OK. You must pick an object that is a series of connected lines or lines that are very close together (for example a word is composed of characters that are very close together.) Do not pick multiple words as your object. Draw an imaginary rectangle around the anchor. You must pick an object, whose enclosing rectangle has a unique height and width, there should be no other objects near it that have the same shape.
4. Define the anchor region. This is a large rectangle that surrounds the anchor. It is where PrimeZone will search for the anchor. It allows differences in the physical placement of the anchor in the image (due, for example, to scanning misalignment or skew that was corrected) to occur and yet PrimeZone will still find the anchor. Set the anchor region as wide as you can without including any other objects that are of the same shape as the anchor.
5. Define any fixed zones. Fixed zones are regions that do not repeat in columns. They typically are header information such as report title, or page number, or date, etc. However, they can be at any location on the page. These regions must always be in the same place relative to the anchor. These zones must also be defined before the column zones (they must have zone numbers less than columnar zones). Make sure you accurately count these fixed zones and set the "FixedZones" attribute below.
6. Define one row of columnar data. Green Bar assumes that variable data is found in rows. Each row has multiple fields that are lined up going down the page in columns. Define the first row of fields. You should not pick an example image to create this master template in which the first row is blank. Pick an image that has data starting at the first expected row. Define the size of each region to be as large as possible without overlapping with another zone. For fields which are close together, the defined zones should extend halfway into the blank space between zones in both vertical and horizontal dimensions of zones that are placed close together. For fields that are far apart extend the zone definition about two characters width beyond the expected end of a zone.

Warning: Your image may contain fields that do not contain the maximum number of characters for a particular field. Make sure that you extend zones to their fullest designed length.

If for some reason you do not want to recognize data from a field it is OK to skip this field, i.e., do not define a zone for it.

If PrimeZone finds an object in the designated area it will report the zone coordinates of the object in the output template. If PrimeZone can not find an object in the designated zone it will report back the zone coordinates that were defined in the MasterTemplate. Sometimes PrimeZone can not find an object because there is no object, the field is blank, other times PrimeZone can not

find the object due to operational problems. However, in either case the zone is sent to the OCR engine. If the zone was truly blank then the OCR engine will report no output, if data was there then the OCR engine will read and report the data.

7. For both fixed and columnar zones make sure that each zone's Content Restriction/Lexical Check/Accuracy Level is set correctly. Content Restriction and Lexical Check are a very powerful way to increase OCR accuracy. Data in Green Bar reports can often be constrained. A common Content Restriction setting, for example, is ALPHAUPPERCASE for part numbers, or NUMERIC for a date field such as "01/12/96".

8. Set document and image configuration data at this time. These settings will be used by the PrimeOCR engine. For example, most users should select "PrimeOCR Output" under "Output Format". Image Processing is a special case. We would recommend that any image processing be done by PrimeZone (with `pz_save_image()` to save processed image for PrimeOCR), or a process before PrimeZone. Be very careful if you select any image processing to take place within PrimeOCR that it does not change the image location, otherwise all zone location data determined by PrimeZone will be wrong.

9. Save the template to a known file name and put this file name and path in the ini file as the MasterTemplate.

Warning: Save the image used to create this template to a safe, known location. You will probably want to refine the MasterTemplate at some point and if you have not saved the original image you'll probably end up starting over with a new image.

LineHeight

Height in pixels of uppercase letters.

LineJump

Pixels from bottom of one line to bottom of next line.

FixedZones

Number of fixed zones defined in MasterTemplate. Zero is OK.

Columns

Number of zones defined in one row of MasterTemplate. Zero is OK but implies that you are just trying to find FixedZones.

AllowGaps

If zero any line gap in output will cause PrimeZone to stop looking for rows of data. If set to one PrimeZone will continue looking for lines until it hits the end of the page.

Warning: Any non trivial noise or unwanted data in the zone regions below the current working line will cause PrimeZone to report the zones on that lower line, even if AllowGaps=0, and even if all the other zone regions are blank (except for the noise/data region).

Output:

A file in PTM (template) format. All document, page, and zone configuration data in the output file is the exact data found in the MasterTemplate.

Lines**Application Name:**

LINES

Functionality:

This application will find lines of text in a document or any other image objects which have the same shape as a line of text.

Data Assumptions:

Note that roughly 3 or more space characters in the line of text will cause the line to be broken into multiple zones, i.e. lines.

Data Requirements:

Edit configuration items present in PRPZDLL.INI in \BIN directory. The contents of this file should be as follows:

```
" [PrimeZoneDLL-Lines]"  
"Width=30"  
"Indent1=50"  
"Indent2=50"  
"Indent3=50"  
"Indent4=50"
```

The meaning of these parameters is as follows:

Width

Approximate width in pixels of a space character in image. (30 is a good number to start with. Try larger or smaller numbers if program gives bad results and page is out of the ordinary in width.)

IndentX

Distance in pixels on each edge of image that program can ignore in its calculations. This parameter allows algorithms to ignore noise that is often at edge of images causing faster speed and fewer errors. Make sure that parameters never touch valid data, therefore, we recommend these settings be kept very conservative. 1=Left edge, 2= Top edge, 3=Right edge, 4=Bottom edge.

Output:

All document, page, and zone information except for zone bounding box is hard coded into the application as follows:

Document Configuration Settings

Output Format	"3"	Prime OCR format (".pro") format
# Pages	"1"	

Page Configuration Settings

Page Number	"1"	First page
Document Source	"0"	"0" means that this page is not from the same document as the last page processed by the engine.
Image Quality	"0"	Default quality setting
Print Type	"0"	Machine Print
Language	"10"	U.S. English
Deskew	"0"	Do not perform (perform during PrimeZone if necessary)
Image PreProcess	"0"	Do not perform (perform during PrimeZone if necessary)
AutoZone	"0"	Do not Autozone
Number of Zones	"Y"	Determined by PrimeZone

Zone Description

Zone Number	"X"	
Zone Restrictions	"0"	No restrictions
Lexical Check	"1"	Perform lexical check on output
Accuracy Level	"99999"	Run all five engines
Left Dimension	"X"	These parameters determined by PrimeZone
Top Dimension	"X"	

Right "X"
Dimension

Bottom "X"
Dimension

Refer to *PrimeOCR Access Guide* for further information on these settings.
Contact Prime Recognition if you would like any of these settings changed or if you want Prime Recognition to add the ability to change these settings through a INI file setting.

Columns

Application Name:
COLUMN

Functionality:

This application is intended to find full page columns plus a header area. It can identify 1 or multiple columns per page. It is not suitable for columns that are not rectangular in shape. Any footer data is thrown into the column above it, in other words footers are not identified. Please contact Prime Recognition if you need this function customized to recognize footers or some other requirement.

Data Assumptions:

Columns are separated from each other by space greater than the a single space character. Columns are rectangular in shape. Text exists in the header area that is significantly larger than the text in the columns.

Data Requirements:

Edit configuration items present in PRPZDLL.INI in \BIN directory. The contents of this file should be as follows:

```
" [PrimeZoneDLL-Column]"  
"Width=10"  
"Height=10"  
"Indent1=50"  
"Indent2=50"  
"Indent3=50"  
"Indent4=50"
```

The meaning of these parameters is as follows:

Width

Approximate width in pixels of a space character in image. (10 is a good number to start with. Try larger or smaller numbers if program gives bad results and page is out of the ordinary in width.)

Height

Approximate distance in pixels of white space vertically between lines of text.(10 is a good number to start with. Try larger or smaller numbers if program gives bad results and page is out of the ordinary in width.)

IndentX

Distance in pixels on each edge of image that program can ignore in its calculations. This parameter allows algorithms to ignore noise that is often at edge of images causing faster speed and fewer errors. Make sure that parameters never touch valid data, therefore, we recommend these settings be kept very conservative. 1=Left edge, 2= Top edge, 3=Right edge, 4=Bottom edge.

Output:

All document, page, and zone information except for zone bounding box is hard coded into the application as follows:

Document Configuration Settings

Output Format	"3"	Prime OCR format (".pro") format
# Pages	"1"	

Page Configuration Settings

Page Number	"1"	First page
Document Source	"0"	"0" means that this page is not from the same document as the last page processed by the engine.
Image Quality	"0"	Default quality setting
Print Type	"0"	Machine Print
Language	"10"	U.S. English
Deskew	"0"	Do not perform (perform during PrimeZone if necessary)
Image PreProcess	"0"	Do not perform (perform during PrimeZone if necessary)
AutoZone	"0"	Do not Autozone
Number of Zones	"Y"	Determined by PrimeZone

Zone Description

Zone Number	"X"	
Zone Restrictions	"0"	No restrictions
Lexical Check	"1"	Perform lexical check on output
Accuracy Level	"99999"	Run all five engines
Left Dimension	"X"	These parameters determined by PrimeZone
Top Dimension	"X"	
Right Dimension	"X"	
Bottom Dimension	"X"	

Refer to *PrimeOCR Access Guide* for further information on these settings.
Contact Prime Recognition if you would like any of these settings changed or if you want Prime Recognition to add the ability to change these settings through a INI file setting.

Chapter 6

Troubleshooting

The first step during development should be to try to compile and run the C/C++ source code sample program as is. If the sample program does not compile or run correctly with the included example images then there is a problem with your environment.

API Error Reporting

The best way to determine the cause of errors is to always check the value of the "status" variable returned by each PrimeZone API call. A value of 0 indicates success, any negative value indicates an error.

All API calls perform as much input validation as possible, such as receiving a NULL pointer, or input data out of range. Bad inputs will cause immediate function termination (no intended processing occurs) and reporting of bad input.

If an error is reported that is not listed in the documentation please call Prime Recognition technical support.

Activity Log

A text file log of all major activities of the engine can be generated for debugging purposes. Turn on the log file by setting the value of "LogFile=" in the PRPZDLL.ini file to 1.

The name of the log file is PRZONE.LOG, and the size is 1000K bytes. Once the log file exceeds this size it is renamed as PRZONE.001 and a new PRZONE.LOG file is started.

The log file includes a description of all API calls made, along with the supplied variable values. It also includes major internal actives. We recommend you call Prime Recognition for assistance in reading the log file.

Troubleshooting Suggestions

If you have installed imaging products from TMSSequoia Data or other imaging vendors:

Run the "SRCHFILE.EXE" program (in \BIN directory) to search for file conflicts on your system.

There could be a conflict with the DLLs from these vendors. Temporarily remove the code from these other vendors to see if any problems persist.

If the program runs much of time but once in a while does not run correctly:

Restart the operating system and in some cases, reboot the system. The majority of these problems are caused by a lack of RAM.

Make sure you are calling `pz_free_results()` after you are done with the RAM based output, otherwise the operating system will eventually run out of memory which will cause PrimeZone operating errors.

Make sure that enough disk space exists to write PrimeZone results.

API Error Numbers

- 100 Invalid hInstance
- 101 Invalid hWnd
- 113 Could not find a file called "PRPZ.DLL" in \BIN directory pointed to by the PATH variable. Check to see that \BIN is in the PATH.
- 115 Called `pz_open()` when engine was already open. (call `pz_close()`).
- 116 Called `pz_close()` when engine was not open.
- 320 Could not open output_file file
- 321 Template version was incorrect (set in PRPZDLL.INI)
- 401 Unknown application type
- 887 Could not find the hardware key, or hardware key contents were not read correctly.
- 860<->900 Assorted hardware key errors.
- 1003 Not licensed for deskew option.
- 1017 Could find or open PRIMAGE.INI file. (It should be in \BIN directory).
- 1018 Error trying to enhance image.
- 1022 Pointer to "skew" not provided in `pz_enhance_image()` call.
- 10031 Not licensed for image enhancement option.

Any error not reported here implies a serious internal error which can not be solved by end user intervention. Please contact Prime Recognition.

Chapter 7

Contacting Prime Recognition & Warranty/Support

We encourage you to call Prime Recognition with any technical or sales questions or comments. We are very interested in feedback on our products. If you need any functionality which is not in the existing product please let us know.

If you have a significant problem with our product or support, please contact us. We will do our best to set it right.

Prime Recognition can be contacted by phone, FAX, mail, Email, Web, or FTP as noted below.

Prime Recognition
Four Buttercup Street
San Carlos, CA 94070-1528
Phone: 650-631-9800
FAX: 650-631-5686
EMAIL: support@primerecognition.com
FTP: <ftp.primerecognition.com>
WEB: www.primerecognition.com

Warranty/Support

Prime Recognition offers an annual warranty and support program. The program offers free unlimited technical support via phone, FAX, FTP, mail, or Email at the above numbers, plus free maintenance upgrades and 50% discount on major upgrades.

On-site Technical support is available for a daily charge plus travel expenses. Please call to verify the current daily rate and availability.

Chapter 8

Programming to PrimeZone API

The PrimeZone interface is a DLL (Dynamic Link Library) with an API. The DLLs were developed in Microsoft Visual C++ , however, the API to the developer was implemented with a "C" style interface which can be accessed by C, C++, Visual Basic, Delphi, and other tools/languages.

SPECIAL NOTE: AS OF MARCH, 1998, THE PRIMEZONE DLL IS NOT SHIPPED WITH THE STANDARD PRODUCT. IF YOU ARE INTERESTED IN PROGRAMMING TO THE DLL YOU MUST CONTACT PRIME RECOGNITION THE DEVELOPER'S VERSION.

THE FOLLOWING DATA IS ACCURATE FOR THE DLL PRIOR TO MARCH, 1998 AND IS INDICATIVE OF THE FUNCTIONALITY AND STYLE OF THE PRIMEZONE DLL, HOWEVER, IT IS NOT CURRENT FOR THE PRIMEZONE DLL AFTER MARCH, 1998.

Include Files

Any call to the PrimeZone API requires these include files:

```
#include <prpzapi.h>
```

This file details relevant defined constants and data structures and the API call formats.

Libraries

The PRPZDLL.LIB library must be linked into the controlling application.

Programming Languages

Developers using other languages should refer to their language documentation for instructions on how to call a DLL with a C style function call interface.

API Call Functionality Groups

The API calls can be grouped by functionality. These groups are:

Open/Configure/Close Engine

- pz_open()
- pz_close()
- pz_configure_application()

Input/Process Image

- pz_read_image()
- pz_enhance_image()
- pz_process_image()

Convert Output to a File

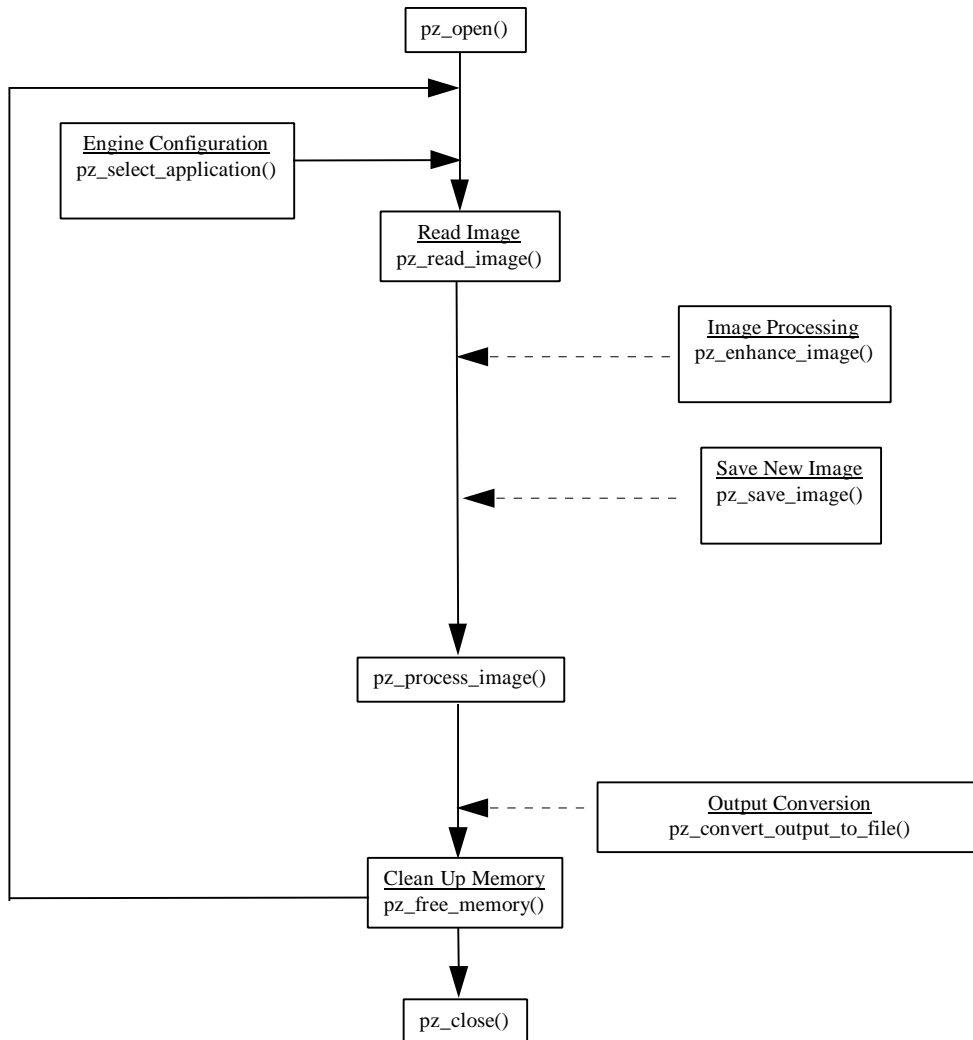
- pz_convert_output_to_file()
- pz_pdf_image_out ()

Free Memory

- pz_free_memory()

API Call Sequence

Each API call described later in this chapter includes a discussion of when it can be called in relation to other calls. A figure below gives a rough description of the flow, however, please refer to the actual API descriptions for a more precise description. The line going down the middle of the diagram represents the minimum process for one page. Solid horizontal lines indicate mandatory calls, although the calls may only need to be made once at engine startup. Dotted horizontal lines indicate optional calls.



API Call Functionality

Error reporting

Each function returns an integer or "status" variable indicating 0 for success and -X for errors.

API Calls

Programming to API	1
pz_open()	6
pz_close().....	7
pz_select_application().....	8
pz_read_image().....	9
pz_enhance_image().....	11
pz_save_image()	13
pz_process_image().....	14
pz_desample_image().....	15
pz_convert_output_to_file()	16
pz_pdf_image_out ()	17
pz_free_memory()	18

pz_open()

short pz_open(HANDLE hInstance, HWND hWnd);

Function:

Initializes PrimeZone Engine.

Required?:

Required.

Timing:

Should be called only once and before any other call.

Input Variables:

HINSTANCE hInstance

The instance of the user program.

HWND hWnd

The handle to the user program window.

Typical Return Values:

A positive return value indicates the version number of the software expressed as an integer. For example, "250" indicates release 2.50.

A negative return value indicates an error.

-100 Invalid hInstance

-101 Invalid hWnd

-113 Could not find a file called "PRPZ.DLL" in PRDEV\BIN directory pointed to by the DOS PATH variable. Check to see that PRDEV\BIN is in the PATH.

-115 Called pz_open() when engine was already open. (call pz_close()).

-887 Could not find the hardware key, or hardware key contents were not read correctly.

-860<->900 Assorted hardware key errors.

pz_close()

short pz_close();

Function:

Cleans up PrimeZone engine resources right before exit.

Required?:

Required.

Timing:

Must be the last call made to engine.

Input Variables:

None.

Typical Return Values:

-116 Called pz_close() when engine was not open.

Notes:

Any calls made to the engine after pz_close() are not valid.

pz_select_application()

short pz_select_application(short app_type);

Function:

Configures engine for a specific application type.

Required?:

Required.

Timing:

Must be called at least once after pz_open() and before any other call.

Input Variables:

short app_type

Value of application type. Valid values are:

PHONE_BOOK (equivalent to "0")

GREEN_BAR (equivalent to "1")

LINES (equivalent to "2")

COLUMN (equivalent to "3")

Typical Return Values:

-401 Invalid app_type

Notes:

pz_read_image()

short pz_read_image(char * filename);

Function:

Reads an image file into engine. See Notes below for supported file formats.

Required?:

Required.

Timing:

Can be made at any time after pz_select_application and before pz_process_image() call.

Must be called only once per image or memory leaks will occur.

Input Variables:

char * filename

Pointer to a string which includes the full DOS path and filename for the image file, e.g. "d:\images\4931.tif".

Return Variables:

Typical Return Error Values:

Any error usually indicates that it could not find image, or image was not of a supported file format.

Notes:

Supported file formats include:

TIFF (Version 5.0 and above)

-uncompressed

-Group 3 (fax)

-Group 3

-Group 4

-MultiPage

PCX

File resolutions are read by the engine and do not need be input by the developer.

Supported file resolutions (dpi) are:

600 X 600
400 X 400
300 X 300
240 X 240
200 X 200
204 X 196 (Fine mode FAX)

PrimeZone requires the following image attributes:

- The most significant bit is to the left of the least significant bit in each image byte
- A "0" bit is the equivalent of white on a image (and "1" is black)

pz_enhance_image()

short pz_enhance_image(short deskew, short enhance_flag, short * return_skew);

Function:

Enhances image based on deskew and/or enhance_flag and PRIMAGE.INI settings. The flags in the call turn on (or off) deskew/enhancement. The settings in the PRIMAGE.INI file configure the actual parameters for processing.

Returns measured skew if deskew is turned on.

Required?:

Optional.

Timing:

Call can be made anytime after pz_read_image() and before pz_process_image().

Input Variables:

short deskew

Allowed values are:

"0" - Do not perform deskew.

"1" - Perform deskew.

short enhance_flag

OR'd together values for following parameters:

HORIZONTAL_REGISTER	1
VERTICAL_REGISTER	2
HORIZONTAL_LINE_REMOVAL	4
VERTICAL_LINE_REMOVAL	8
INVERSE_TEXT	16
DESHADE	32
DESPECK	64
SUBIMAGE	128
ROTATE	256
PERIOD_REMOVAL	512
SMOOTH	1024
AUTOROTATE	2048
CROP	4096
DIALATION_EROSION	8192

For example, to perform ROTATE and DESPECK enter 320 (256 +64).

Return Values:

short * return_skew

If deskew is turned on then this value represents the amount of skew present in the document. Skew is reported as 1 pixel vertical displacement per X pixels of horizontal displacement. For example, a value of "35" means that for every 35 pixels of horizontal displacement the image was skewed one pixel in the clockwise direction. A negative number means skew displacement in the counterclock wise direction. A large number means smaller skew and vice

versa. However, "0" skew means that no skew was detected. Skew smaller than 250 is generally reported as 0 skew.

Typical Error Return Values:

- 1017 Could not find or open PRIMAGE.INI file. (It should be in \PRDEV\BIN directory).
- 1018 Error trying to enhance image.
- 1022 Pointer to skew not provided in call.

Notes:

The settings in the PRIMAGE.INI file are exactly those settings used in the ScanFix product from TMSSequoia. For documentation on these settings see *Chapter 3* of the *PrimeOCR Access Guide* or contact Prime Recognition.

pz_save_image()

short pz_save_image(char * filename);

Function:

Save image currently in memory to supplied path/filename. This is used most frequently to save image after enhancement so that it is available to the OCR engine in the exact same state as was used for zoning.

Required?:

Optional.

Timing:

Call can be made anytime after pz_read_image() and before pz_free_memory().

Input Variables:

char * filename

Full path and file name of desired output file. Format will be same as the input file. Most often this filename will be the input file name, causing an overwrite (replacement) of the existing image file.

Typical Return Values:

Notes:

pz_process_image()

short pz_process_image(void);

Function:

Performs zoning on current image in memory.

Required?:

Required.

Timing:

Can be made at any time after pz_read_image() and pz_free_memory().

Input Variables:**Typical Return Values:****Notes:**

If an application requires a INI file then this call causes the reading of the INI file on the first usage of this call after the pz_select_application() call. Therefore, if you want to change the ini file settings for an application without opening/closing engine you should:

- Change contents of INI file
- Call pz_select_application() (selecting same application will still cause INI file read below).
- Call pz_process_image() (after appropriate pz_read_image(), etc.)

pz_desample_image()

short pz_desample_image(short desample_dpi);

Function:

Downsamples images to the defined dpi.

Required?:

Optional.

Timing:

Can be made at any time after pz_process_image() and before pz_free_memory(), although it makes sense to desample the image before saving output.

Input Variables:

short desample_dpi

The resulting dpi of the image after desampling.

The dpi must be positive and less than 600.

Typical values are 200 and 300.

Typical Return Values:

Notes:

Desampling an image, while decreasing image resolution, saves on storage requirements of the resulting output. Use this function to minimize the size of resultant image files while maintaining image definition.

An image should not be desampled to below 200 dpi for consistent OCR results.

pz_convert_output_to_file()

short pz_convert_output_to_file(char * output_file);

Function:

Writes zoning results to a PrimeOCR Job Server format template file.

Required?:

Optional. (Although unclear why you would not call this to get results.)

Timing:

Can be made at any time after pz_process_image() and before pz_free_memory().

Input Variables:

char * output_file

Full path and file name for output file. We recommend but it is not a requirement to use ".PTM" as the extension to template files.

Typical Return Values:

-320 Could not open output_file file

-321 Template version was incorrect (set in PRPZDLL.INI)

Notes:

See *PrimeOCR Access Guide* for a discussion of the PrimeOCR template format.

pz_pdf_image_out ()

```
short pz_pdf_image_out(char * from_file, char * image_out, short current_page,  
                      short num_tiff_pages, short append);
```

Function:

Save image currently in memory to supplied path/filename in PDF Image Only format.

Required?:

Optional.

Timing:

Call can be made anytime after pz_read_image() and before pz_free_memory().

Input Variables:

char * from_file

Full path and file name of input file.

char * image_out

Full path and file name of desired output file. Format will be PDF Image Only format, version 2.1.

short append

Defines if current image is being appended to an existing file.

Typical Return Values:

-744 Could not create the export file

Notes:

pz_free_memory()

short pz_free_memory(void);

Function:

Frees memory allocated during the processing of an image.

Required?:

Required.

Timing:

Can be made at any time after pz_read_image(). Once called any calls normally made after pz_read_image() are invalid.

Input Variables:

Typical Return Values:

Notes:

Chapter 9

Example Applications

One sample application is included with the Developer's Toolkit:

-C/C++ Source Code Sample PZEXAM.EXE

This is a simple application that interfaces to the PrimeZone API. The sample was written in C++ (but in simple "C" style) using Microsoft Visual C++ Version 2.0. We recommend that you look at the sample application early in your reading, as it will illustrate quickly how to interface, and control the PrimeZone engine.

SPECIAL NOTE: AS OF MARCH, 1998, THE PRIMEZONE EXAMPLE APPLICATION FILES ARE NOT SHIPPED WITH THE STANDARD PRODUCT. IF YOU ARE INTERESTED IN THESE FILES YOU MUST CONTACT PRIME RECOGNITION TO RECEIVE A DEVELOPER VERSION.

Functionality

The sample application processes a single phone book page as follows:

- Initializes PrimeZone engine
- Loads supplied test image
- Configures PrimeZone engine with phone book application
- Processes image
- Converts output to template file
- Releases PrimeZone RAM memory
- Closes PrimeZone engine

Files

This application includes:

PZEXAM.EXE	compiled application
PZEXAM.CPP	C++ source code
PZEXAM.DEF, PZEXAM.RC	Resource & other files
PZEXAM.MAK	Make file
PHBK1.TIF	Test image

Example Source Code

The example application has virtually no user interface and is not "event" driven as a "good" Windows program should be, it is intentionally simplistic to focus attention on the PrimeZone code. All PrimeZone code and processing occurs within the lines noted below in the WinMain function.

```
//*****Prime Recognition Example Code Start*****
```

```
PrimeZone Code...
```

```
//*****Prime Recognition Example Code End*****
```

Except for the required PrimeZone header files, all other code outside of these lines is used to create a minimal Windows program to "host" the PrimeZone process.

The example program reports errors, if any, through a message box and terminates.

Feel free to modify and use the sample application code to create an interface to the PrimeZone engine.

Visual Basic Source Code Sample

If you are interested in a VB based source code sample please contact Prime Recognition.