# M500 User Manual and Programmer SDK

**Revision 1P**
**Doc No 560802-1A**
**May 19, 2017**

For the latest driver and manual visit: http://www.e-seek.com

E-SEEK Inc.
Website: www.e-seek.com
Patented Product

R & D Center
9471 Ridge haven Ct. #E
San Diego, CA 92123
Tel: (858) 495-1900
Fax: (858) 495-1901

Sales & Marketing
245 Fischer Ave #D5
Costa Mesa, CA 92626
Tel: (714) 545-3316
Fax: (714) 545-3595

## TABLE OF CONTENTS

# 1. Introduction

Thank you for choosing this device.

This User's Guide provides descriptions of the operating procedures and programming APIs for E-seek Model M500 Authentication Unit. Carefully read this User's Guide before using this device.

The actual screens that appear may be slightly different from the screen images used in this User's Guide. Model M500 Authentication Unit (ID Card Type) is hereafter referred to as "this device"

**Manual Convention**

- Caution: This warns of a possibility of damage to this device.
- Important: This indicates instructions that should be followed to ensure correct functionality and efficiency of this device.
- Note: This indicates an item of general importance.
- Reminder: This indicates an item of relative importance.
- Detail: This indicates an item of specific importance.

**Restrictions**

- Unauthorized use or reproduction of this User's Guide, whether in its entirety or in part, is strictly prohibited.
- The information contained in this User's Guide is subject to change without notice.

## 1.1 Product Feature

This device is an ID card authentication system that performs user authentication by ID cards and supports most ID cards.

## 1.2 Prerequisite

- ID card compatible with this device must be prepared.

# 2. Device Description

The E-Seek Model M500 Forensic Reader introduces a new performance standard for ID card reading and authentication. It extracts hidden security features of a drivers' license, Military ID or any identity card for authentication purposes in addition to reading and decoding the electronic information contained in the barcode or magnetic stripe. Both sides of a scanned card are captured at 600 dpi resolution with a single card insertion.

The Model M500 also incorporates a high performance PDF417 decoder and 3 Track magnetic readers. It connects to a PC using a high speed USB 2.0 connection.

Ideal for Law Enforcement, Bars, Nightclubs, Casinos, and Access Control. The M500 can be used anywhere your application requires reading and/or authenticating an ID or Drivers' License.

## 2.1 Overview of Model M500

Figures, 1 and 2 illustrate the major modules and components of the M500.

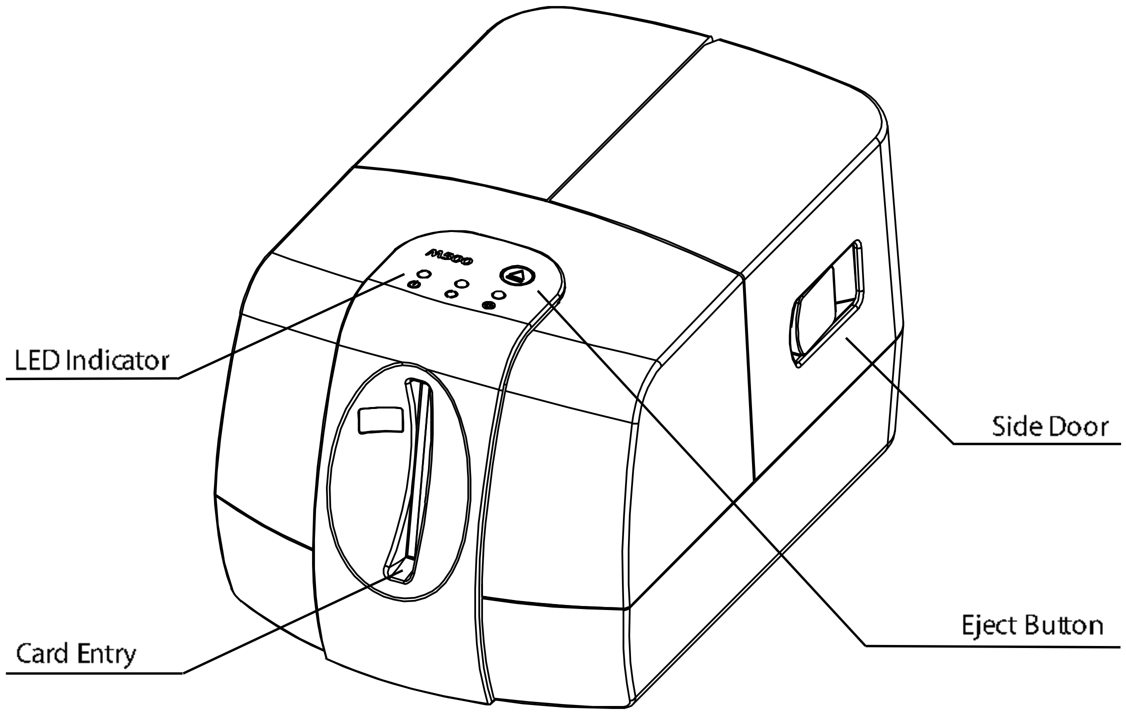


**Figure 1**

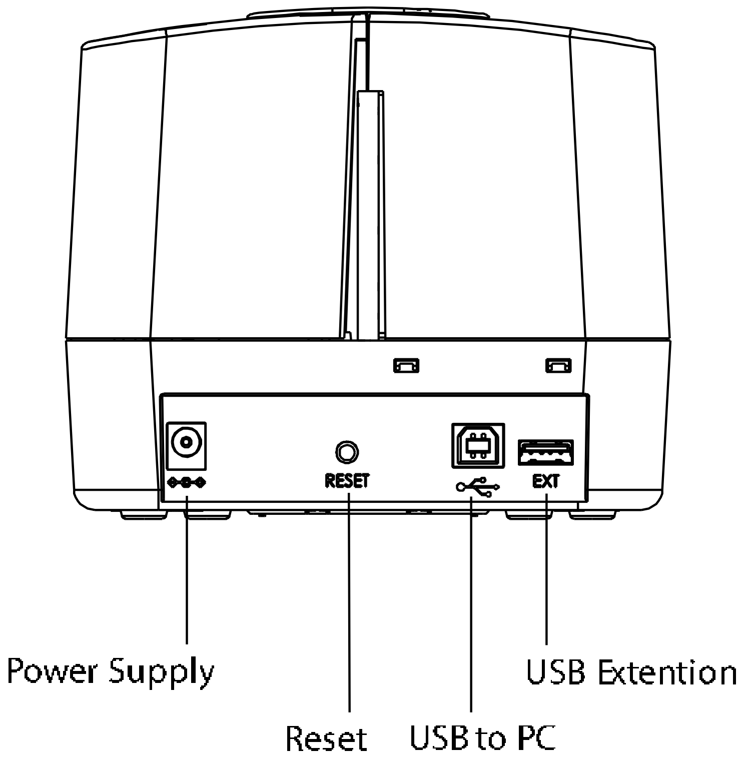


**Figure 2**

## 2.2 PRODUCT SPECIFICATION

| Part Name | Description |
|---|---|
| Processor | Freescale ARM i.MX27 |
| Memory | 128 MB DDR SDRAM<br>1 MB Flash Memory |
| Scanning Method | CCD 24bit RGB Duplex 600 DPI with 1200 DPI option. |
| Barcode Reader | 2D: PDF417 Linear: CODE 39 & CODE 128 |
| Magnetic Reader | 3 Track Magnetic Reader (ANSI, ISO, CDL, AAMVA) |
| External Interfaces | Power Jack (15 V DC 2.4A)<br>USB 2.0 Device Port<br>USB 2.0 extension Port |
| Indicator(s) | Speaker<br>Green LED for Power status<br>Blue and White center mounted LEDs under user control<br>Red LED for Error status |
| Communication | High Speed USB 2.0 |
| Weight | M500 device: 4.25 lb (Without Power Supply) |
| Dimensions | 8.5" Length x 5.7" Width x 4.8" Height |
| Regulatory | Complies with FCC & CE rules |
| Power Supply | Output 15V DC 2.4 A / Input 100-240V 50-60 Hz |

## 2.3 TECHNICAL DETAILS

- Dual Side Images in a Single Pass
- UV, IR, RGB Images at same time
- 600DPI and with 1200 DPI option
- 24 bit Color, 8 bits IR & 24 bit UV image
- Capture ROI Image (Region Of Interest)
- CR80 Size (Driver License, ID-1)
- Decode 2D (PDF417) & 1D
- 3 Tracks Magnetic Stripe Reader (ANSI, ISO, CDL, AAMVA)
- USB 2.0 High speed
- Easy Operation
- Jam Free
- Patented

## 2.4 Unpacking Device

The M500 package includes:

- M500 Device
- Power Supply
- USB Cable
- Cleaning Card
- Calibration Card
- Power Supply
- The AC/DC power supply
- The power cord plugs into the power jack

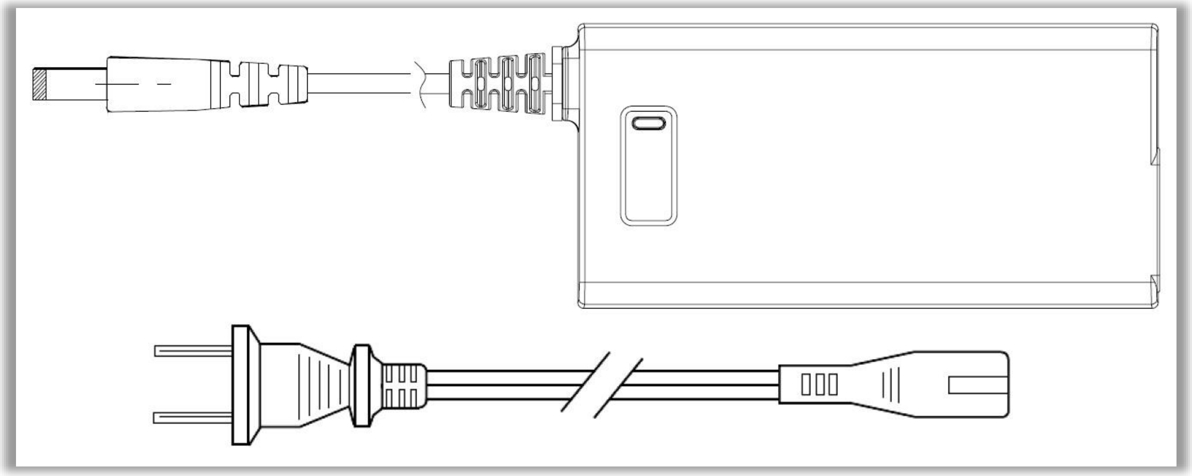


**Figure 3**

### 2.4.1 USB Cable

The M500 is provided with a USB interface cable. This cable allows the M500 to interface with standard USB 2.0 or higher port on your computer.

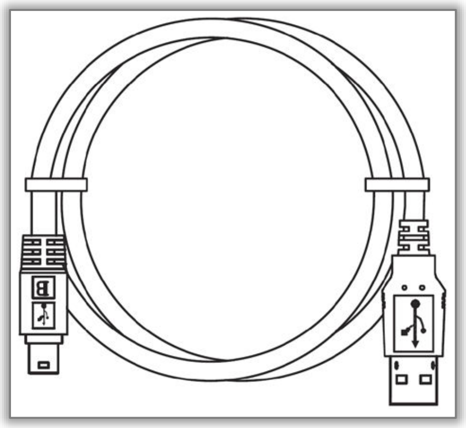


**Figure 4**

### 2.4.2 White Balance Calibration Card

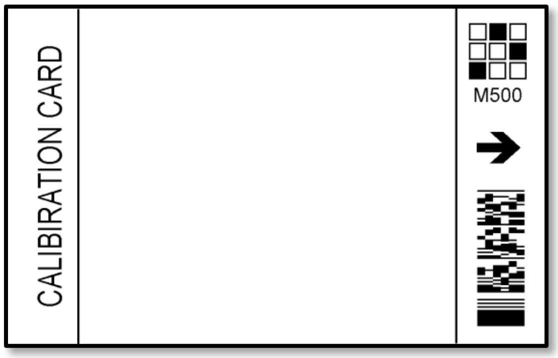
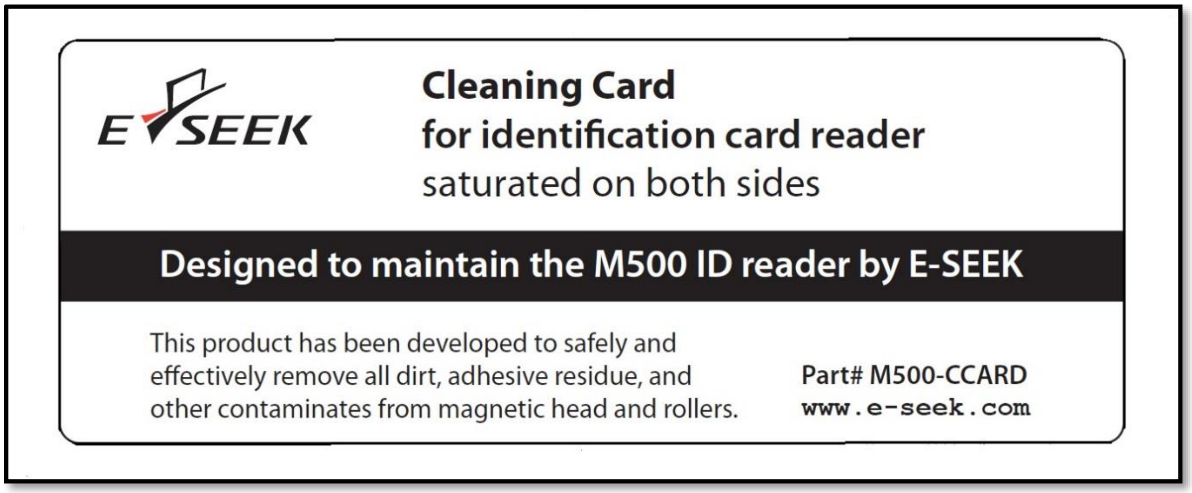


**Figure 5**

The calibration card is used to calibrate the white balance. Calibration could be required after shipping or prolonged use. To perform the white balance calibration simply insert the card with the arrow side first.

After prolonged use or if the card becomes scratched it should be discarded.

### 2.4.3 Cleaning Card



Cleaning Card
for identification card reader
saturated on both sides

Designed to maintain the M500 ID reader by E-SEEK

This product has been developed to safely and effectively remove all dirt, adhesive residue, and other contaminates from magnetic head and rollers.

Part# M500-CCARD
www.e-seek.com

**Figure 6**

The cleaning card cleans the rollers and magnetic head. The need for cleaning varies but the general guideline is once a month for high usage and once every three months for low usage. High usage is defined as more than 3,000 scans per month.

## 3. Getting Started

1. Pre-Installing Driver
   Download the driver from [http://e-seek.com/products/m-500/](http://e-seek.com/products/m-500/)

2. Log in to your computer as an administrator

   Installing the 32 bit Driver (for Windows 32 bit XP, Vista, 7 and 8)
   Run M500_USB_DRIVER-32.exe

   Installing the 64 bit driver (for Windows 64 bit Vista, 7 and 8)
   Run M500_USB_DRIVER-64.exe

WinDriver should appear under Jungo in Device Manager Window.

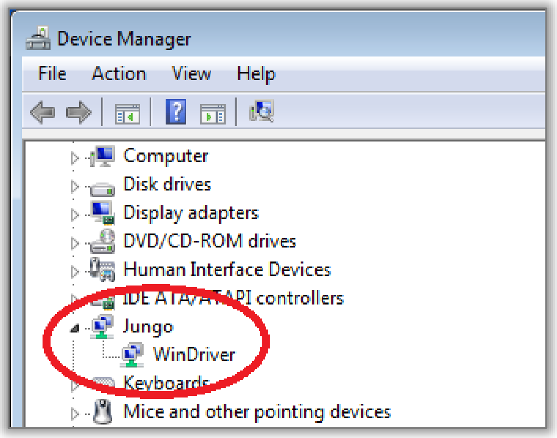


**Figure 7**

Next, connect M500 Power cable and then USB cable to your host PC.
Now the M500 should appears under Jungo in Device Manager Window.

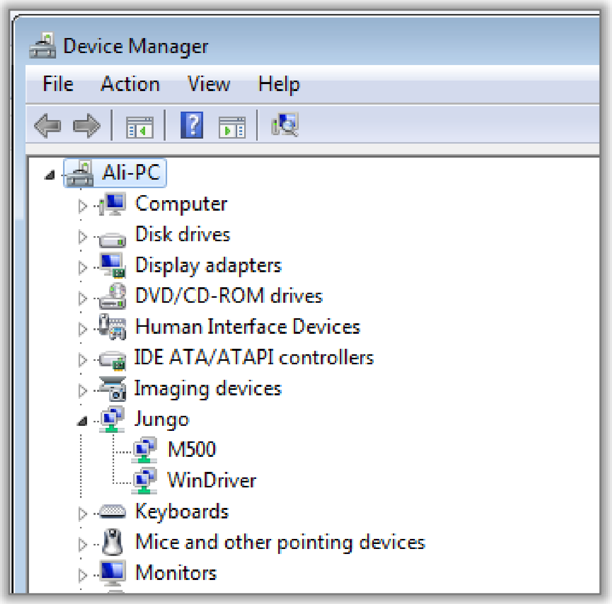


**Figure 8**

At this point check the M500 top LED status, and make sure the Green light is solid ON.

If the RED light blinks it indicates that the scanner encounter a fatal error. Check the error type by opening the "M500dll.log" file.

**NOTE:** The most common error is the mismatch versions between the M500 firmware and the M500dll.dll. The M500dll.dll contains a copy of the firmware it was designed to work with embedded inside the DLL file.

There are three ways of upgrading the firmware:

## 3.1 Upgrading Firmware

### 3.1.1 Upgrade firmware from Demo application

The Demo application checks the M500 firmware version when it starts or the M500 boots. If the firmware in the M500 is not the correct version for this DLL then the demo software will pop up a dialog box with the option of upgrading the firmware.

### 3.1.2 Upgrade by restarting the scanner

This method will force the scanner to synchronize (upgrade or downgrade) the firmware in M500dll.dll.

Step 1: Connect the M500 to PC and run your application (which loads M500dll.dll)
Step 2: Unplug the power to the M500.
Step 3: Press and hold the Eject button (located behind the red LED).
Step 4: While holding the Eject bottom connect the power to M500.
Step 5: Hold the Eject button for until the Green LED starts flashing (about five seconds).
Step 6: The M500 will automatically restart followed by a beep.
Step 7: The scanner firmware is now upgraded using the M500dll's embedded firmware image.

### 3.1.3 Upgrade firmware from User application

The software developer can check the firmware version using the **GetVer** () API and upgrade the firmware using the **ProgramFlash**() API if the firmware version is not compatible with the DLL.. Refer to **Section 9.1** and the DEMO sample application

**NOTE:** The firmware needs to be compatible with the DLL that the user application is using. The DLL and firmware have a three digit major, minor, and build version number. However the build number of released firmware is always zero. For example DLL version 1.8.0, 1.8.1, and 1.8.2 would all be compatible with firmware 1.8.

## 4. Running Demo Application

Download The M500 Demo Application from http://e-seek.com/products/m-500/
Unzip the files to your PC, and run M500.exe (either 64 bit or 32 bit)

If the red light is flashing, and the Demo program will pop up a dialog box with the option to upgrade the firmware.

For other error types see the log file “M500dll.log”.

## 5. Scope

The PC software consists of an application exe, a C# API assembly, and a C/C++ DLL that communicates with the M500 over USB. The PC software also includes a third party USB driver from Jungo. This document covers the M500 C# sample application and the C# API that gives a C# developer a simple interface to the unmanaged C/C++ M500 DLL.

## 6. Operation

When a card is inserted the M500 firmware will:

- Read the magnetic stripe
- Scan using the White LEDs
- Decode the PDF417 barcode
- Scan using the IR LEDs
- Scan using the UV LEDs
- Eject the card

Optionally, for faster scans, it is possible to skip the IR and UV scans when only the RGB image, magnetic stripe data, and PDF417 barcode data are needed.

### 6.1 Indicator LEDs

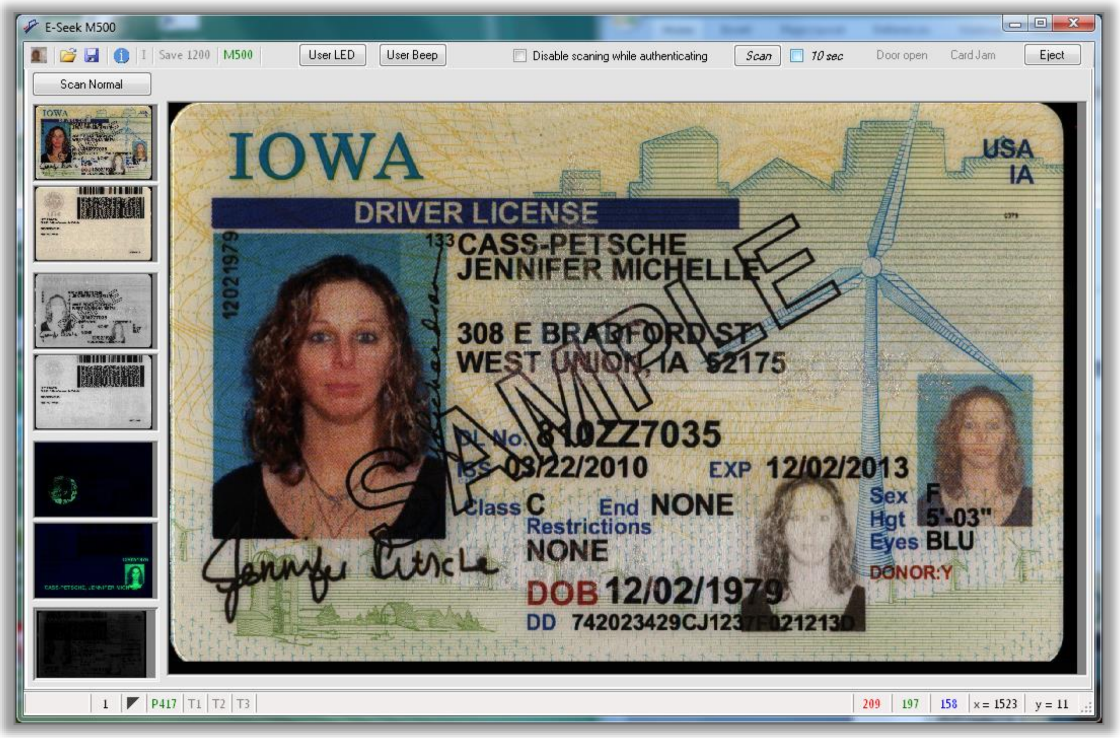
M500 LED status table is as follow:

| | | |
|---|---|---|
| 🟢 | Solid | Scanner Ready |
| | Blink | Scanning |
| 🔴 | Solid | Door Open |
| | Blink | Fatal Error |
| 🔵 | Application Defined | |

## 6.2 Eject Button

This button, which is located on the top of the units, is used to try to eject a jammed card.

## 6.3 GUI



IOWA
DRIVER LICENSE
USA IA
CASS-PETSCHE
JENNIFER MICHELLE
308 E BRADFORD ST
WEST UNION, IA 52175
DL No. 810ZZ7035
ISS 03/22/2010 EXP 12/02/2013
Class C End NONE
Restrictions NONE
Sex F
Hgt 5'-03"
Eyes BLU
DONOR:Y
DOB 12/02/1979
DD 742023429CJ1237F021213D
SAMPLE

Figure 9

The GUI has a toolbar and controls on the top, small preview images on the left, a large selectable main image, and a status bar on the bottom.

### 6.3.1 Small Image Preview Pane

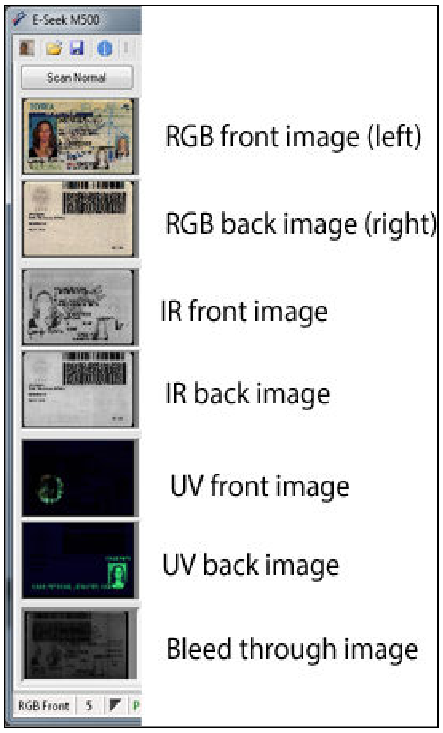


**Figure 10**

There are seven small panes which display the scanned card using different lighting.

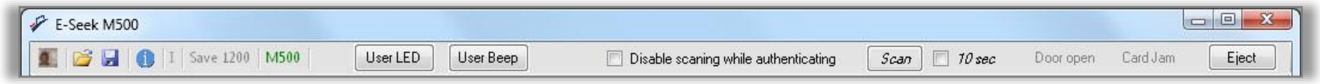
- The first two images are the front (right) and back (left) images using white light.
- The next two images were captured using IR light.
- The next two images after that were captured using UV light.

Finally, the last image is a low resolution image with the right side illuminate using IR light and detected on the left side which is not illuminated. The intensity of the “bleed through” image is an indication of the card’s opacity.

The big image displays the RGB front (right) image by default but the user can select the image to display by clicking on one of the seven small preview panes on the left.

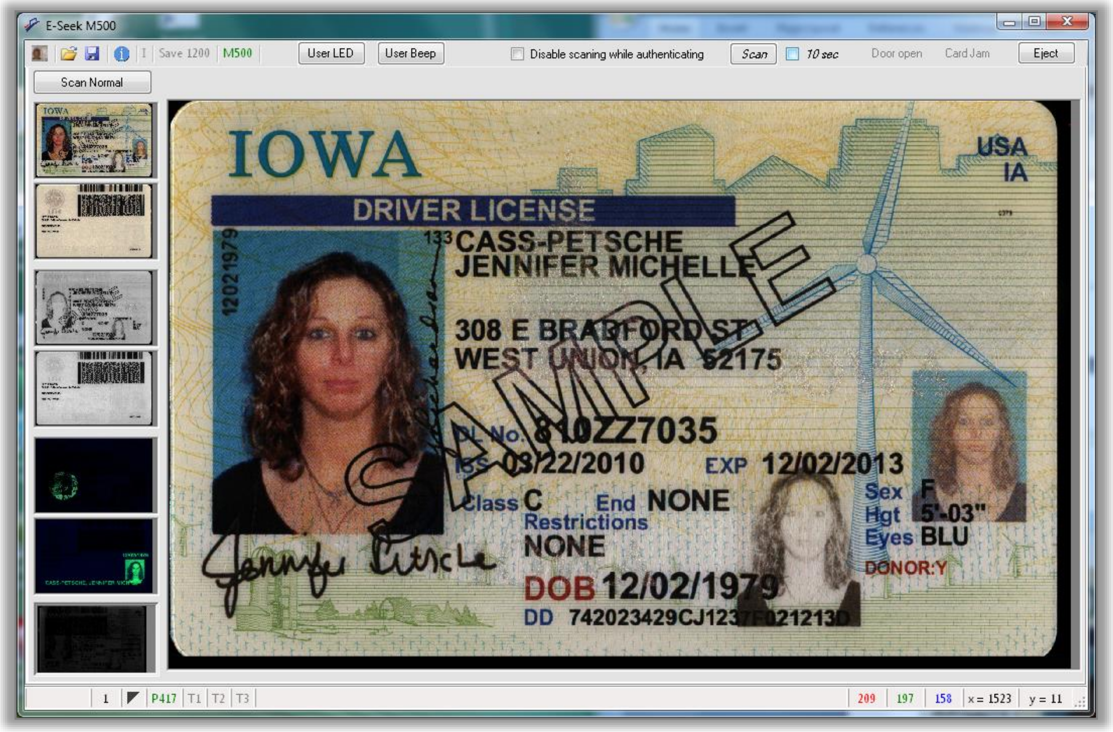
Double clicking on a small preview image will create bitmap file and then open it using the program that is set by windows to view .bmp files. (Note: for this to work the demo application should be run from a directory that has write privileges)

### 6.3.2 TOOLBAR AND CONTROLS



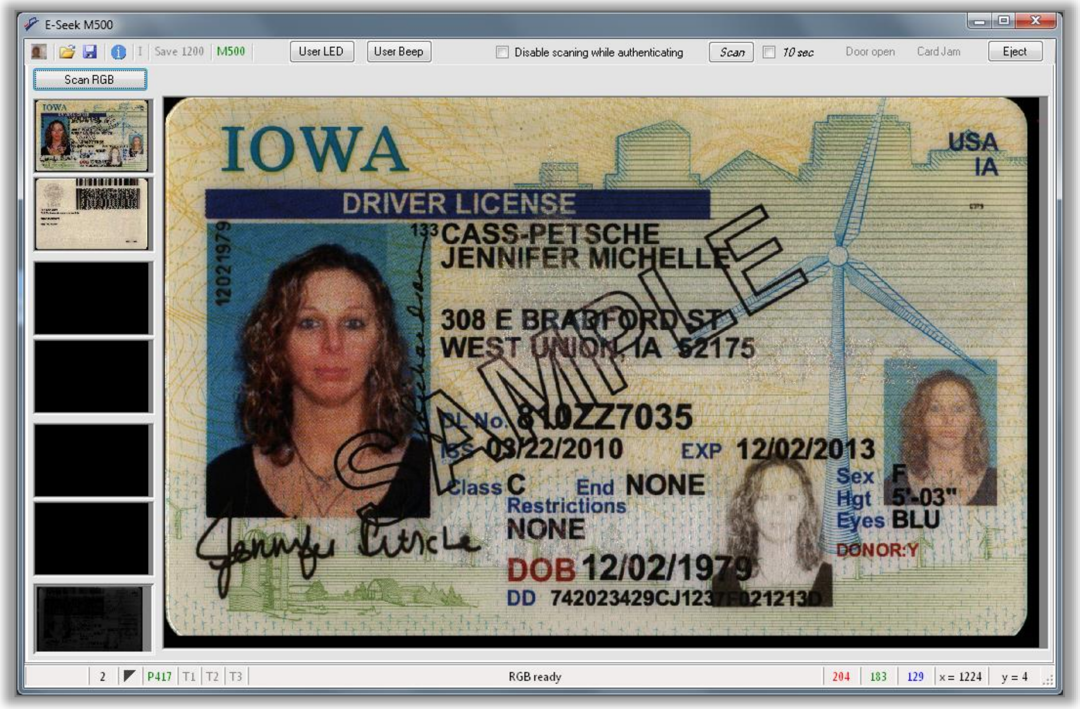
- The leftmost toolbar button re-acquires the image from the scanner for testing.
- The open toolbar button opens an achieved tiff file.
- The save toolbar button saves the scan as an archive tiff file.
- The blue “i” toolbar button opens a modal dialog box with the current version numbers.
- The “M500” text is green when the USB connection is good and red when the USB connection fails.
- The “User LED” button toggles the middle user LEDs on the top of the unit. There are a blue LED and a white LED mounted next to each other. Clicking on this button cycles thru blue LED on, white LED on, blue LED blinking, white LED blinking, and both LEDs off. Note that early production units do not have a white LED.
- The “User Beep” button toggles thru three beep sounds.
- The “Disable scanning while authenticating” check box is unchecked for normal operation and checked to demonstrate how the application software can disable (and re-enable) scanning.
- “Scan” was used to during EMI testing and is beyond the scope of this document.
- “10 sec” is used to perform aging testing and is beyond the scope of this document.
- The “Door open” text turns red when the door is open.
- The “Card Jam” text turns red when a card jams.
- The “Eject” button starts the card eject procedure.

### 6.3.3 Scan Modes



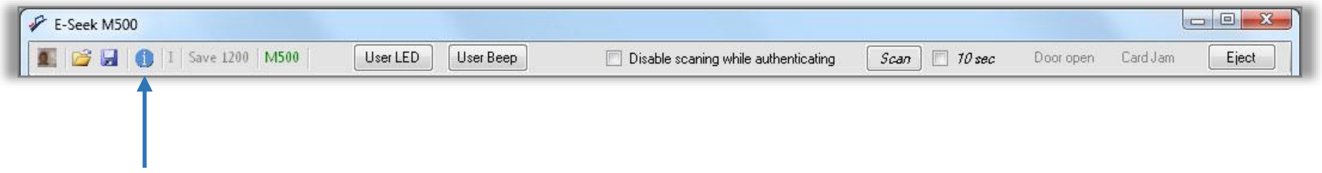
By default the normal scan mode is RGB-IR-UV.

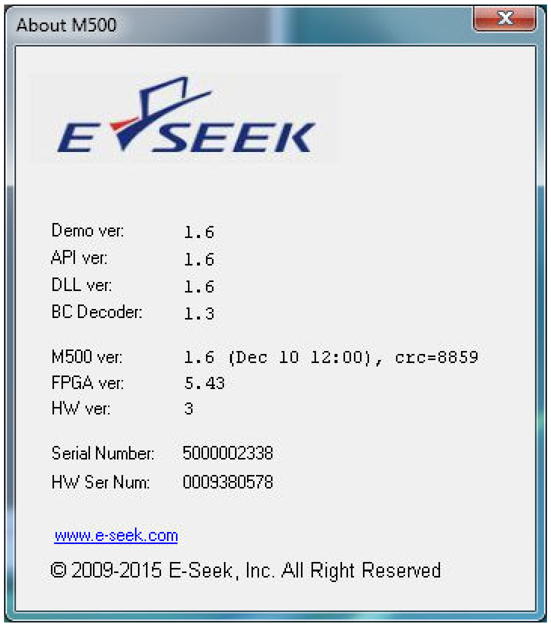
Click on the "Scan Normal" button until it changes to "Scan RGB" for fast RGB only scanning.



### 6.3.4 About Dialog Toolbar Button

Clicking on the blue "i" brings up the about dialog box.




About M500

| | |
|---|---|
| Demo ver: | 1.6 |
| API ver: | 1.6 |
| DLL ver: | 1.6 |
| BC Decoder: | 1.3 |
| M500 ver: | 1.6 (Dec 10 12:00), crc=8859 |
| FPGA ver: | 5.43 |
| HW ver: | 3 |
| Serial Number: | 5000002338 |
| HW Ser Num: | 0009380578 |

www.e-seek.com

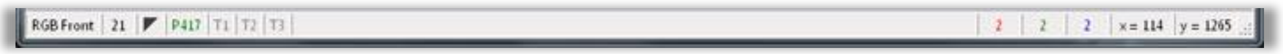
© 2009-2015 E-Seek, Inc. All Right Reserved

Note that the API, DLL, and M500 (firmware) versions are always the same (In this case 1.6).

The DLL checks for the expected M500 (firmware) and barcode decoder versions and brings up a dialog box if the versions do not match.

The T1, T2, and T3 likewise will turn green when data is found on the magnetic stripe for tracks 1/2/3.

### 6.3.5 Status Bar



The triangle indicates the PDF417's orientation if it exists.

The five possibilities are front/back, normal/upside down, and not found.

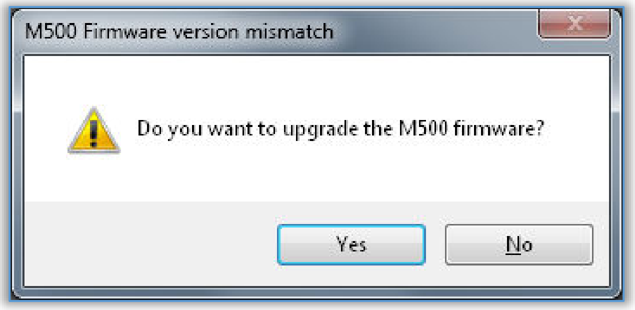
The PDF417 text will turn green when a PDF417 barcode is found.

The T1, T2, and T3 likewise will turn green when data is found on the magnetic stripe for tracks 1/2/3.

### 6.3.6 Errors

The demo GUI checks the flash version and will display a dialog box allowing the user to upgrade the firmware from the DLL's internal firmware image.

Whenever the DLL detectcs a firmware version mismatch or other serious error it will sends an error event (In this case `EV_ERR_FW_VER)` and send a command to the firmware to flash its red LED. On receiving this event the demo application will pop up the following dialog allowing the user to upgrade the M500's firmware.

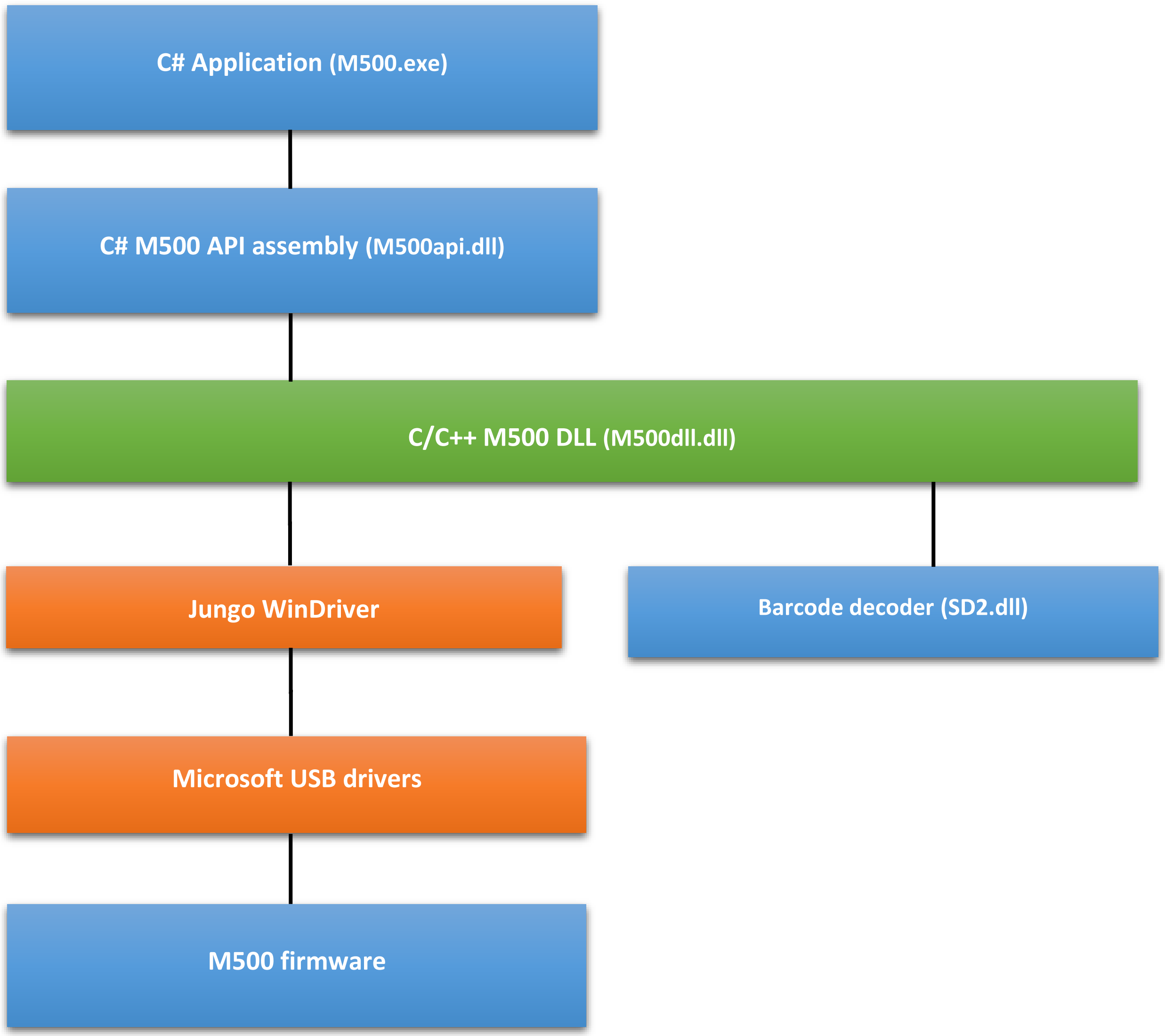

M500 Firmware version mismatch

Do you want to upgrade the M500 firmware?

Starting with DLL version 1.9 the older M500 firmware will be automatically upgraded unless the user application overwrite this default feature. (see section 9.1 AutoFwUpgrade API)

## 7. Architecture

The main purpose of the C# demo application is to provide an example of how to write an application that interfaces with the M500 using a C# API.


C# Application (M500.exe)
C# M500 API assembly (M500api.dll)
C/C++ M500 DLL (M500dll.dll)
Jungo WinDriver
Barcode decoder (SD2.dll)
Microsoft USB drivers
M500 firmware

The application (M500.exe or user application), M500api.dll, M500dll.dll, and SD2.dll must be in the same directory. The DLL will create a log file (M500dll.log) in the directory in which it is running by default but it can be disabled if needed.

# 8. M500 Demo App

The C# M500APP project contains the Main app and GUI. It creates the "M500.exe" executable.
The modules in this project are:

- FormM500demo.cs
- FormGUI.cs
- FormUpdate.cs
- Settings.cs
- MagWnd.cs
- UsrImage.cs
- AboutForm.cs

## 8.1 FormM500demo.cs

This is the main form and contains the code that interfaces with the M500 C# API. It calls the Init() function which initializes the M500DLL which then creates its own thread to communicate with the M500 and automatically transfer images without using the applications UI thread.

## 8.2 FormGUI.cs

This module contains GUI code that was separated from the FormM500demo.cs module to simplify it. It contains code to Open and Save a TIFF file and update the magnify window. It does not interface with the M500 so one does not need to understand this code to interface with the M500.

## 8.3 FormUpdate.cs

This module contains subroutines that update the GUI.

## 8.4 MagWnd.cs

This module manages the floating magnifier window and is not needed by the developer to interface with the M500.

## 8.5 UsrImage.cs

This module contains code to support re-sizing the bitmaps windows as the size of the main window changes and is not needed by the developer to interface with the M500.

## 8.6 AboutForm.cs

This module is used to display the version of application, C# API assembly, M500 DLL, barcode decoder, and M500 firmware.

# 9. C# API

The C# API provides a simple interface to the M500. The C# developer should be able to use this interface to quickly interface with the M500 without needing to interface with the M500 DLL unmanaged code directly.

The DLL creates its own thread to poll the M500. The application should register for call back events at initialization. The DLL will then call back the application on the DLLs thread when an event occurs. The application should then synchronize the call back to its thread using the Invoke method in FormM500demo.cs.

## 9.1 API Functions

`void` **`SetLogDir`**`(LOG_DIR)  [Optional]`
Call this function before Init() to override the default log directory. By default if this function is not called the M500DLL will create the M500DLL.LOG file in the same directory it is running in. Pass this function the string of the desired log directory. To disable logging pass the string "null".

`void` **`AutoFwUpgrade`**`(bool bAuto)`
Call this function with false before Init() to override the default behavior (as of version 1.9) of automaticaly updating the M500's firmware if it is out of date. You do not have to call this function if you want the default behavior of automatic firmware upgrade.

`void` **`Init`**`()`
Call this function at initialization such as during form load.

`void` **`RegCB`**`(OnNewEvent)`
Register event call back.

`void` **`LastScan`**`(iLastScan)  [Optional]`
By default, cards are scanned using white, IR, and UV light.
To override the last scan pass iLastScan with the folling values after the RGB image is ready:

```
0 =   RGB-IR-UV (normal)
3 =   RGB only
```

`void` **`Close`**`()`
Call this function before closing the application such as during form closed.

`void` **`ClearData`**`(byte byGrey)`
Call this function to clear the bitmap images, barcode, and magnetic stripe data. There is no need to do this under normal circumstances but an application may wish to do this to after the scanned data has been processed for maximum privacy.

The byte passed to ClearData() will be copied to every 24bit RGB pixels.
Passing 0 will results in black bitmaps (until the next scan)
Or passing 0x80 will result in medium gray.

`bool LogIn(bool bLogin)`

When true the unit will scan when a card is inserted (normal operation).
When false the unit will not scan when a card is inserted.

`void UserLED(E_LED eLED)`

Controls the center LEDs. First generation units only have a blue LED. Second generation units will also have a white LED. In addition to on and off one can also blink the user LEDs. The parameter E_LED is an enumeration value:

```
LED_USR_OFF,        // Both blue and white LEDs OFF
LED_USR1_ON,        // Blue LED on.    (white OFF)
LED_USR2_ON,        // White LED on.   (blue  OFF)
LED_USR1_BLINK,     // Blink blue LED  (white OFF)
LED_USR2_BLINK,     // Blink white LED (blue  OFF)
```

`void UserBeep(E_BEEP eBeep)`

Creates a beep sound. The E_BEEP enumeration has three values:

```
BEEP_1,
BEEP_2,
BEEP_3,
```

`bool EjectCard()`

Starts the card eject procedure as if one had pressed the eject button.
This allows the application to try to eject a stuck card without the user having to press the eject button.

`void GetVer(out M500_VER ver)`

Gets the E-Seek serial number (`EsSerNum`), Silicon serial number (`DsSerNum`), DLL version, Barcode decoder version, firmware version, and hardware version as defined by the `M500_VER` structure.

The members of the M500_VER structure that may be of intrest to the deverloper are:

```
ulong  EsSerNum;        // E-Seek serial number
//
byte   DllMajor;        // DLL version number
byte   DllMinor;
byte   DllBuild;
byte   FwMajor;         // Firmware version number
byte   FwMinor;
byte   FwBuild;         // Always zero
```

`void ProgFlash(NULL)`

For manufacting a string with a filename of the flash image is passed.
For developers pass NULL instead of a string to update the firmware from the embedded image in the DLL.

`int  SaveAs(string strFilename)`

Saves the last scan as a tif file.

`int  Open(string strFilename)`

Refreshes the APIs bitmaps, barcode, and magstripe data from an M500 tif file.

`bool WrUserData (byte[] aryData)`

Writes a user data byte array to flash (128 byte limit).
Flash should not be used store frequetly changing data as it is limited to 10,000 reliable writes.

`bool RdUserData(byte[] aryData)`

Reads a user data byte array from flash (128 byte limit).

Events:

The M500 DLL sends event call backs to the application on a thread the M500 DLL creates.

| | |
|---|---|
| `EV_MID_SEN` | Card is at the middle sensor and a scan is about to begin |
| `EV_MS_RDY` | Mag stripe data ready |
| `EV_RGB_RDY` | RGB image ready |
| `EV_IR_RDY` | IR image ready |
| `EV_UV_RDY` | UV image ready |
| `EV_USB_ON` | USB communication with M500 |
| `EV_USB_OFF` | USB cannot communicate with M500 |
| `EV_CARD_IN` | A card has been inserted (first sensor) |
| `EV_CARD_OUT` | A card has been ejected |
| `EV_CARD_JAM` | A card jam was detected |
| `EV_CANCEL` | Scan aborted |
| `EV_DOOR_OPEN` | Door opened |
| `EV_DOOR_CLOSE` | Door closed |
| `EV_CLEAN_CARD` | Cleaning card |
| `EV_WHITE_CAL` | White calibration card |
| `EV_WHITE_CALV` | White card verification (in reverse for dust detection) |
| `EV_ERR_FW_VER` | Error – firmware is not the correct version |
| `EV_ERR_BC_VER` | Error – SD2.DLL barcode deocoder is not the correct version |
| `EV_ERR_SER_NUM` | Error – Serial Number is invalid, possible flash corruption |
| `EV_ERR_AFE_REG` | Error – AFE registers are invalid, images may be saturated |
| `EV_ERR_FW_UPGR` | Error – firmware was not the correct version but was automaticaly updated |

The application must use **MethodInvoker** beform performing UI (user interface) operations.
The callback uses the thread that the M500 DLL created and can not perform UI

## 9.2 API STRUCTURES

The C# API M500_IMG class has a bitmap for each of  the seven images:

```
class M500_IMG
{
…
  Bitmap m_RgbFrBmp = new Bitmap(XPIXELS, YPIXELS, PixelFormat.Format24bppRgb);
  Bitmap m_RgbBkBmp = new Bitmap(XPIXELS, YPIXELS, PixelFormat.Format24bppRgb);
  Bitmap m_IrFrBmp  = new Bitmap(XPIXELS, YPIXELS, PixelFormat.Format8bppIndexed);
  Bitmap m_IrBkBmp  = new Bitmap(XPIXELS, YPIXELS, PixelFormat.Format8bppIndexed);
  Bitmap m_UvFrBmp  = new Bitmap(XPIX_UV, YPIX_UV, PixelFormat.Format24bppRgb);
  Bitmap m_UvBkBmp  = new Bitmap(XPIX_UV, YPIX_UV, PixelFormat.Format24bppRgb);
  Bitmap m_MsBxBmp  = new Bitmap(XPIX_MS, YPIX_MS, PixelFormat.Format8bppIndexed);
…
}
```

The first two images are the RGB front and back.
The next two images are the IR front and back.
The last two main images are the UV front and back.
The last image is a low resolution IR bleed image captured mag stripe scan.

Assuming the API assembly is instantiated in the application as:

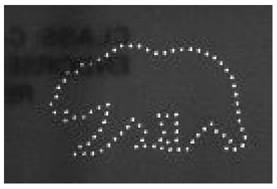
```
public static CM500api m_M500 = new CM500api();
```

The RGB front bitmap would be accesed by the application as:

```
m_M500.m_Images.m_RgbFrBmp
```

The last image there is a low resolution image taken during the magnetic strip capture (m_MsBxBmp). This image is taken by illuminating the right side of the card with IR light and then getting an image of the light that goes thrugh the card.  A few states have laser holes that create light dots where the light goes through the card.

The image below is the outline of a bear from light going through the laser holes on a California driver's license.



The C# API `M500_BC` structure contains a 2D data structure.

```
unsafe public struct M500_BC
{
    public BC_1D d1;         // 1D (linear)
    public BC_2D d2;         // 2D (PDF417)
    public int   iMajor;
    public int   iMinor;
    public int   iOrient;    // Orientation
    public int   iTry;
    public fixed byte cModel[8];
}
```

If the PDF417 barcode is found the iOrient element will
The `iOrient` element has four enumerated values of the card orinetation and zero for unknown.

0 = Unknown orientation
1 = Normal orientation (Front of card is on the right).
2 = Front on the right but upside down.
3 = Front is on the left.
4 = Front is on the left and upside down.

The main element of the 2D barcode structure is the length (zero if not decoded).

```
unsafe public struct BC_2D
{
    public short nLen;       // Length
    …
}
```

After getting the barcode call back check the length to determine if the data is valid. For example, to check if the PDF417 data was decoded check the length in "`nLen`".

`m_BC.d2.nLen` // Length of valid data in aryP417

The C# API `M500_ DATA` structure has byte arrays to access the PDF417 and mag stripe data.

Check "`nLen`" to determine the number of valid bytes.

```
public struct M500_DATA
{
    public  byte[] aryP417;    // Byte array to PDF417
    public  byte[] aryT1;      // Byte array to magnetic track 1
    public  byte[] aryT2;      // Byte array to magnetic track 2
    public  byte[] aryT3;      // Byte array to magnetic track 3
}
```

## 4.3 Example pseudo code

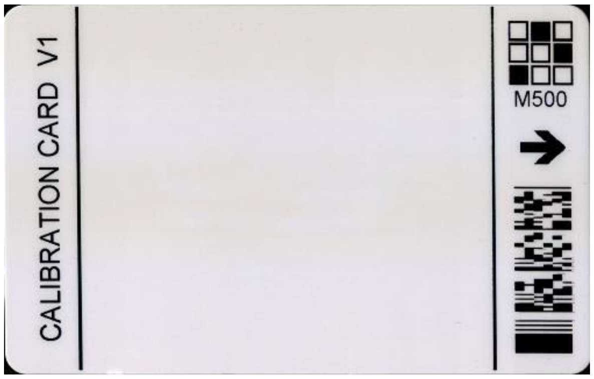
```
CM500api m_M500 = new CM500api(); // C# API object

m_M500.Init();
m_M500.RegCB(OnNewEvent);  // Register event callback
…


// Event callback in DLLs thread
public void OnNewEvent(int iEvent)
{
   switch (iEvent)
      {
      case EV_MS_RDY:
        Invoke(new MethodInvoker(ScanMsRdy));      // Calls ScanMsRdy on applications thread
        break;
      case EV_RGB_RDY:
        Invoke(new MethodInvoker(ScanRgbRdy));     // RGB image ready
        break;
      case EV_BC_RDY:
         Invoke(new MethodInvoker(EvScanBcRdy));   // Barcode ready
         break;
      case EV_IR_RDY:
        Invoke(new MethodInvoker(ScanIrRdy));      // IR image ready
        break;
      case EV_UV_RDY:
        Invoke(new MethodInvoker(ScanUvRdy));      // UV image ready
        Break
       …
      }
}

// RGB image is ready, process on apps thread
private void ScanRgbRdy()
{
   m_M500.UpdateRgb();    // Update the RGB bitmap with the last scan
   fmUpdateRgb();         // Update the GUI and process the image
}

…

m_M500.Close()
```

# 10. MAINTENANCE

## 10.1 CALIBRATION CARD

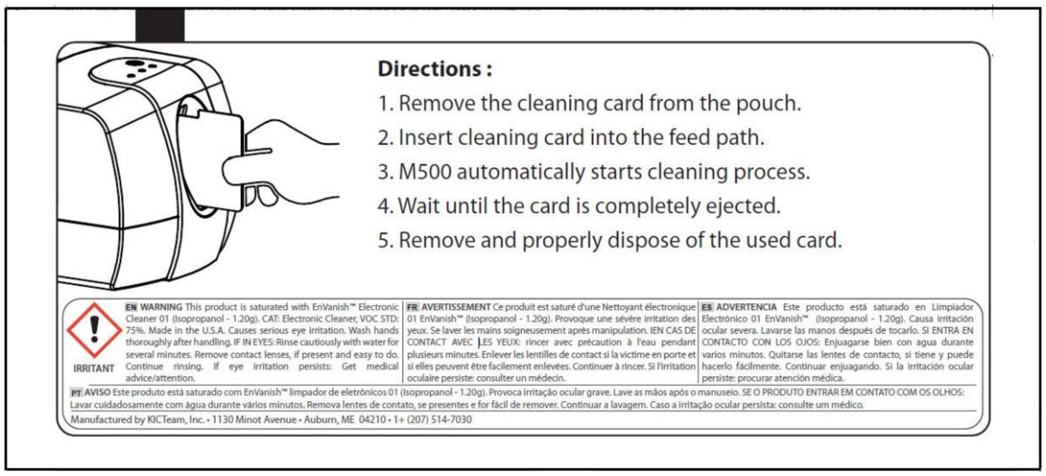
The main Purpose of the White Calibration Card is to perform a white color balance calibration. This calibration occurs when the Calibration Card is inserted into the M500 with the ARROW FACING IN towards the M500. (See the Calibration Card illustration below). When this Calibration process is performed, an **EV_WHITE_CAL** event will occur, which will cause certain settings to be calculated and stored within the M500. During this process, the data displayed through the application software will be the previous calibration settings. The new settings will not be displayed until the next scan. These settings are used for color balance calculations within the M500 during normal scanning operation.

The secondary purpose of the White Calibration Card feedback in the determination of whether or not dust or debris may have accumulated on the M500's mirrors. The Verification Process occurs when the Calibration Card is inserted into the M500 with the ARROW FACING OUTWARD. When this is performed, an **EV_WHITE_CALV** will occur. At this point, the application software should display RGB images to the user to facilitate the determination of whether or not there is dust or debris on the mirrors. Note. The Verification process does NOT re-calibrate the M500.



*Calibiration Card*

The cleaning card is designed to clean the internal rollers and the Magnetic Read Head inside the M500. The need for cleaning varies, but is primarily dependent on the usage or the number of card scans. The general guideline is for cleaning once a month for high usage and once every three months for low usage environments. This process is very simple to perform as it is an automatic process once the cleaning card is inserted into the M500.



**Directions :**

1. Remove the cleaning card from the pouch.
2. Insert cleaning card into the feed path.
3. M500 automatically starts cleaning process.
4. Wait until the card is completely ejected.
5. Remove and properly dispose of the used card.

IRRITANT

**EN WARNING** This product is saturated with EnVanish™ Electronic Cleaner 01 (Isopropanol - 1.20g). CAT: Electronic Cleaner, VOC STD: 75%. Made in the U.S.A. Causes serious eye irritation. Wash hands thoroughly after handling. IF IN EYES: Rinse cautiously with water for several minutes. Remove contact lenses, if present and easy to do. Continue rinsing. If eye irritation persists: Get medical advice/attention.

**FR AVERTISSEMENT** Ce produit est saturé d'une Nettoyant électronique 01 EnVanish™ (Isopropanol - 1.20g). Provoque une sévère irritation des yeux. Se laver les mains soigneusement après manipulation. IEN CAS DE CONTACT AVEC LES YEUX: rincer avec précaution à l'eau pendant plusieurs minutes. Enlever les lentilles de contact si la victime en porte et si elles peuvent être facilement enlevées. Continuer à rincer. Si l'irritation oculaire persiste: consulter un médecin.

**ES ADVERTENCIA** Este producto está saturado en Limpiador Electrónico 01 EnVanish™ (Isopropanol - 1.20g). Causa irritación ocular severa. Lavarse las manos después de tocarlo. SI ENTRA EN CONTACTO CON LOS OJOS: Enjuagarse bien con agua durante varios minutos. Quitarse las lentes de contacto, si tiene y puede hacerlo fácilmente. Continuar enjuagando. Si la irritación ocular persiste: procurar atención médica.

**PT AVISO** Este produto está saturado com EnVanish™ limpador de eletrônicos 01 (Isopropanol - 1.20g). Provoca irritação ocular grave. Lave as mãos após o manuseio. SE O PRODUTO ENTRAR EM CONTATO COM OS OLHOS: Lavar cuidadosamente com água durante vários minutos. Remova lentes de contato, se presentes e for fácil de remover. Continuar a lavagem. Caso a irritação ocular persista: consulte um médico.

Manufactured by KICTeam, Inc. • 1130 Minot Avenue • Auburn, ME 04210 • 1+ (207) 514-7030

## 10.2 Air Cleaning

Even though the M500 is basically a sealed unit, dust and debris can, over time, accumulate inside the scan path, and in particular on the mirrors inside the unit. The detection process should be performed in a periodic basis to assess if dust has degraded the image quality.
The Detection and Cleaning process for the M500 mirrors requires the following items:

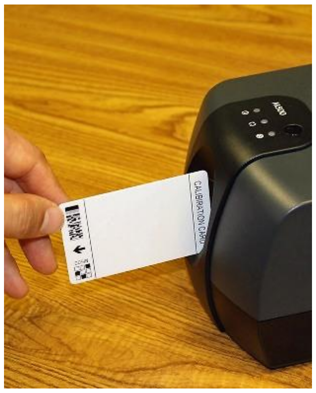
1. Calibration Card, this card is designed and made by E-Seek, Item # M500-CALIB.
2. M500 path Air Divider, a plastic part made to block the M500 scan path and divide the air flow, Item # M500-AIRDV
3. Air Duster, aerosol can of compressed air. The preferred air duster is designed to leave no residue with a high filtration. The non-flammable model of the product is preferred. The following Brand has been used and tested by E-Seek. Chemtronics Ultrajet 70.

https://www.chemtronics.com/p-844-ultrajet-70.aspx

There are three parts to maintain the M500:

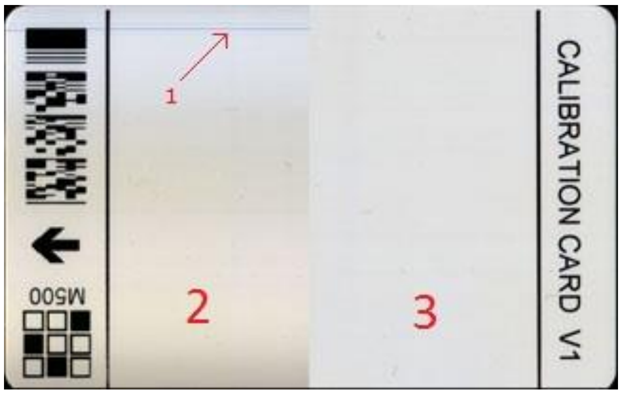
| | |
|---|---|
| Verification | (Step 1-2) |
| Cleaning | (Step 3-5) |
| Calibration | (Step 6-7) |

**Step 1:** Insert the Calibration Card with the arrow facing outward (away from the M500).



*Calibration card inserted with arrow **facing outward***

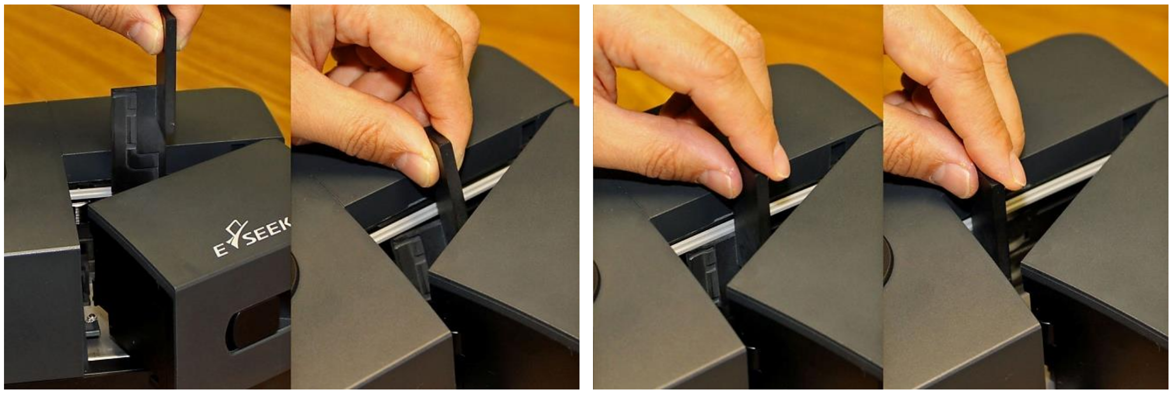
**Step 2:** Verifying that the dust/debris exists on the mirrors.



*Calibration card Image from application*

If there is dust on the mirrors, you will see a dust line (as illustrated by the number 1 on the figure below) on the left section (number2) of the image. The Left section (number 2) is the uncorrected side of the image and the Right Section (number 3) is the corrected side of the image. This method helps to see the raw uncorrected image which allows the dust line to be visible. If such a line exists there is dust on the mirrors, and you should proceed to the Cleaning process, Step 3.

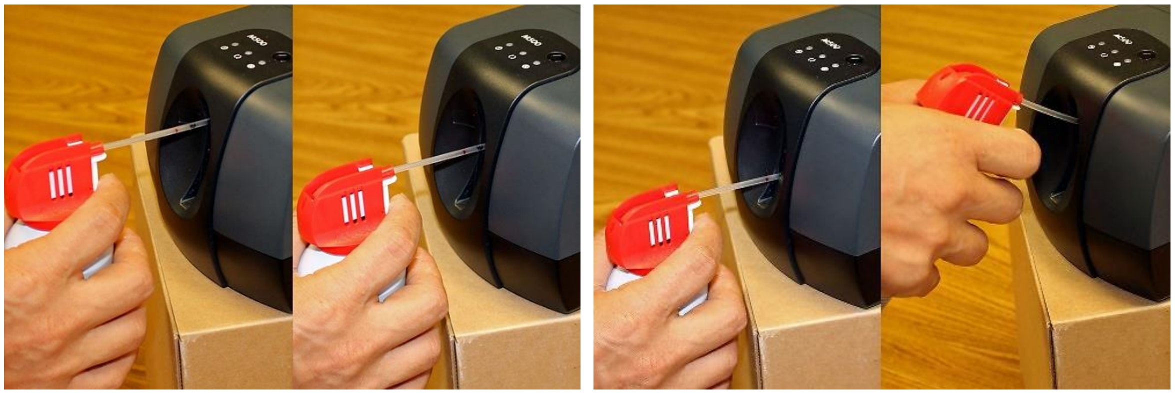
**Step 3:** Cleaning Process. Open the M500 side door and insert the Plastic Air Divider.



*Air Divider Plastic insert*

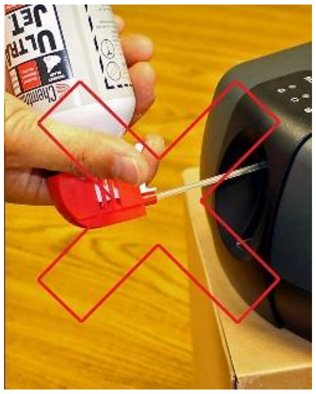
1. Insert the Air Divider down inside the M500 into the opening.
2. Position the Air Divider under the top spring loaded line (Grey Plastic).
3. Align the Air Divider to the M500 side wall.
4. Push the Air Divider completely into the M500 Scan path.

**Step 4:** Use of the Air Duster. Insert the pressurized Air Duster nozzle into the M500 scan opening.



*Air Duster operation*

Follow the operational directions on the Air Duster for use. Typically, you want to press the Air Release Valve in each of the 4 positions shown below for about 2 seconds each.  Do not over use the duster in any position, if the duster gets too cold, wait for about 10 seconds and complete the cleaning operation.

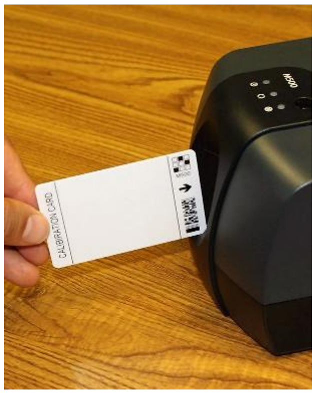


*DO NOT hold the Duster upside down.*

**Step 5:** Remove the Plastic Air Divider and close the side door of the M500. Next REPEAT only Steps 1 & 2. If you see the Dust Line disappear in the uncorrected (Left Section of the Image), then proceed to Step 6. If the Dust Line is still present or has moved to a new location (only on the Left uncorrected image side), then go to Step 3.

Note: Once the dust is removed from the mirrors, you may see a new line on the Corrected section (Right Side) of the image. At this time ignore the Right side image.

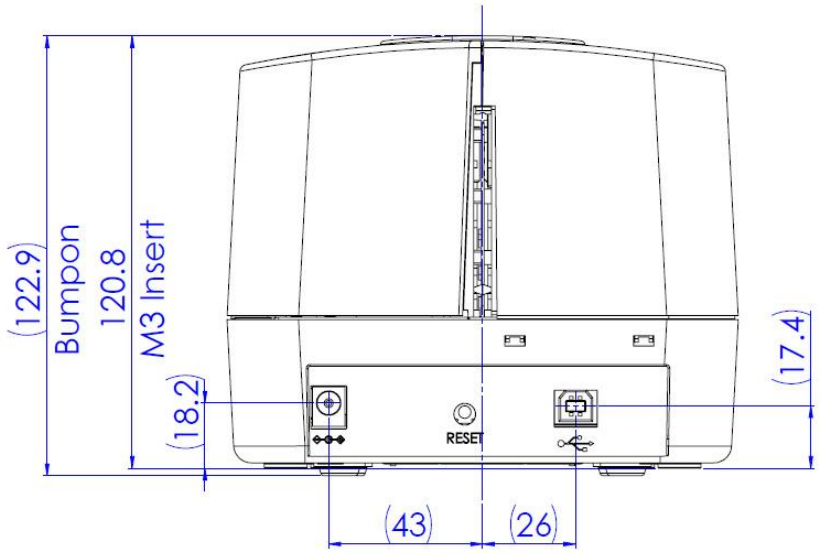
**Step 6:** Calibration. At this point you need to calibrate the M500 without any dust present on the mirrors. This is a simple and mostly automatic process. Simply insert the Calibration Card into the M500 with the arrow FACING INWARD. The M500 will accept the card into the unit and goes through the calibration process automatically. Once the calibration is complete the Calibration Card will be ejected. This operation will store the Calibration Data within the M500.

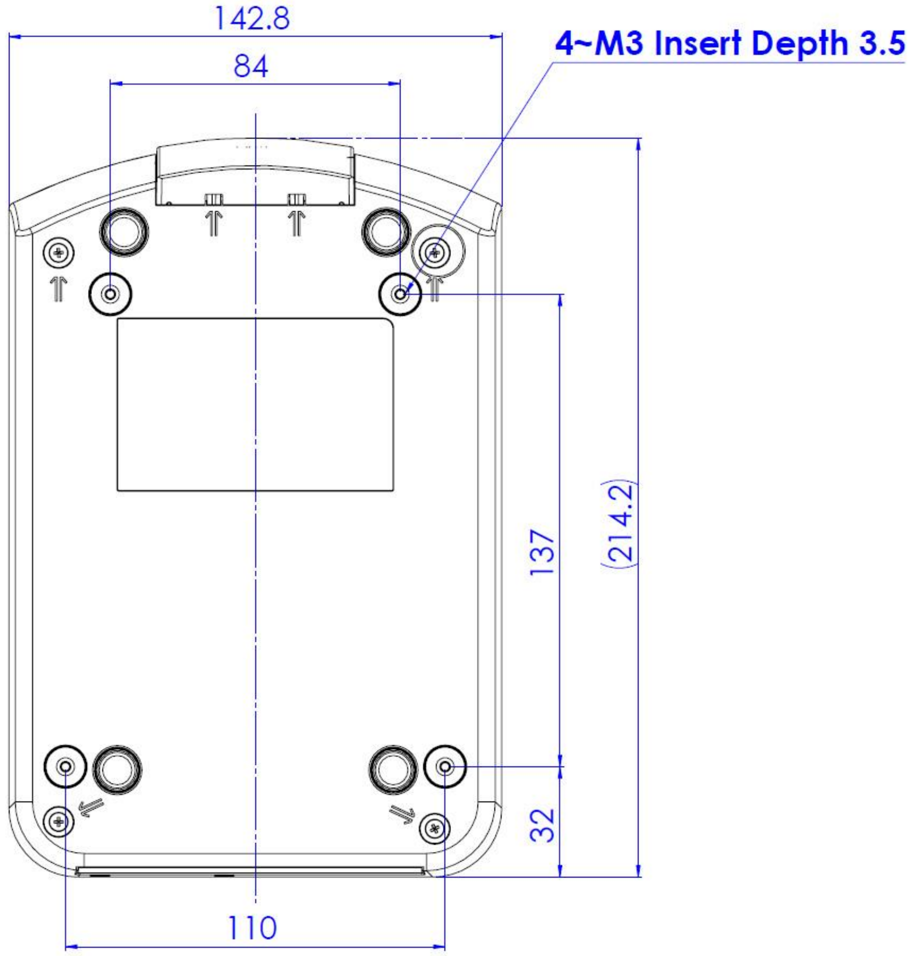


*Calibration card inserted with arrow **facing inward***

**Step 7:** Completion. At this time you can repeat only Steps 1 & 2 to verify your cleaning results. You should see the Left image with no dust and the Right image with a clean White even background.

## 10.3 Mechanical Drawing


(122.9)
Bumpon
120.8
M3 Insert
(18.2)
RESET
(17.4)
(43)
(26)


142.8
84
4~M3 Insert Depth 3.5
137
(214.2)
32
110