ThinClient Development Kit for Windows XPe

Windows XP Embedded Development Guide for the eBOX-III 38XX Series (TcDK)

By: Sean D. Liming and John R. Malin

SJJ Embedded Micro Solutions

First Printing: January 2007

Published in the United States by

SJJ Embedded Micro Solutions, LLC.
11921 Tivoli Park Row #5
San Diego, CA 92128 USA

www.sjjmicro.com

Table of Contents

1 Introduction

Welcome to the Windows XP Embedded Development Guide for the eBOX-III. Windows XP Embedded is used in a variety of applications from point-of-service, kiosk, ATM machines, gaming machines, industrial controls, robotics, medical test equipment, office automation, set-top boxes, multimedia devices, etc. OEMs take advantage of the XPe's componentization and the Embedded Enabling Features (EEF) to build these embedded devices.

The eBOX-III is an industrial grade PC that provides the core PC technologies that support applications like thin-clients, point-of-sale solutions, multi-media terminals, mini-web-servers, print servers, etc. Best of all, the eBOX-III is enclosed in an industrial grade case that is ready for your application.

This guide will discuss how to create, build, and deploy XP Embedded images for the eBOX-III. Platform and device driver components are included in the kit so you will be able to get to work on creating custom images right away. The different exercises build images from the basics to the more complex. Along the way, different techniques and features for custom shells, branding, EWF, and other features used in XPe will be introduced. Techniques for deploying the image will also be discussed.

1.1 Kit Contents

The kit comes with everything to get started working with this guide. Your kit should contain the following:

Software Only Package:

- CD containing: XP Embedded components for the platform and device drivers, and sample configurations for the different demos.
- Device drivers CD for the eBOX-III 38XX series
- Development Guide

Complete Kit:

- Above software package
- ICOP eBox-III, containing a fanless VIA EDEN-N processor, with 256MB system memory, integrated Audio, Ethernet, Parallel, Serial, VGA, USB I/O ports and 512MB IDE bootable flash storage.
- 1GB USB Flash disk used for OS transfers.
- Optional: 802.11 Wireless
- 120 days fully functional evaluation version of Windows XP Embedded SP2.

You will need to download Windows XPe Service Pack 2 Feature Pack 2007 from MSDN: http://msdn.microsoft.com/embedded/windowsxpembedded/default.aspx

Check the download site:
http://msdn.microsoft.com/embedded/downloads/xp/default.aspx

The target system requires the following items to complete the exercises:

- PS/2 Keyboard
- PS/2 Mouse
- VGA Monitor

1.2 Further Reading, Information and Training

Windows XP Pro and Windows XP Embedded are very broad subjects to write about, and the details are outside the scope of a simple getting started guide. This guide is intended as getting started guide for implementing XPe on the eBOX-III. Here are some suggested references for more information on Windows XP Embedded:

- **Books:**

  Windows XP Embedded Advanced, Sean D. Liming, RTC Books, 2003, ISBN:0-929392-77-9.

  Windows XP Embedded Supplemental Toolkit, Sean D. Liming, Cedar Hill Publishing, 2004, ISBN: 1-932373-96-9. *Available at www.sjjmicro.com*

- **Community Sites:**

  XPE Center @ seanliming.com – www.seanliming.com
  XPe Files.com – www.xpefiles.com

- **Training:**

  Windows XP Embedded Training Curriculum - Standard 3 Day course covers Windows XPe Service Pack 2 Feature Pack 2007. A development and target PC is required for each student. The course has been designed to support several optional deployment labs. Please see www.sjjmicro.com for more information.

1.3 What's Next?

This guide was designed to be read in sequential order. Each section builds on the previous sections; although the later exercises are independent of each other. If you are more experienced with XPe, you can skip to the sections that are of more importance.

Section 2 covers the development system setup. It will cover installing the XPe software and importing the XPe component for the eBOX-III.

Section 3 will talk about the eBOX-III platform components, so you know what they are and how the XPe components were developed.

Section 4 discusses the OS deployment method provided with this kit. Taking advantage of the new USB Boot 2.0 solution, a USB flash disk with an XPe image will be used to deploy images to the internal flash drive.

Sections 5-7 are the main Exercise sections that will cover building different custom images. Each exercise creates a different image to demonstrate the different projects that one can create. All of the exercises use the hardware platform component, which is the cornerstone for development.

2 Development Setup

Please follow the instructions for setting up the eBOX-III hardware before you perform any of the software setup or exercises.

2.1 Development System Requirements

The development machine requires the following

- Pentium III 500 MHz or equivalent is the minimum, but a Pentium 4 2 GHz or higher is highly recommended.
- 1GB of RAM minimum - 2GB RAM recommended.
- 1GB HDD space available.
- 10/100Base-T Ethernet card (Optional) – for downloading images over an Ethernet connection.
- Windows XP Professional SP1 minimum.
- Windows XPe SP2 with Feature Pack 2007.

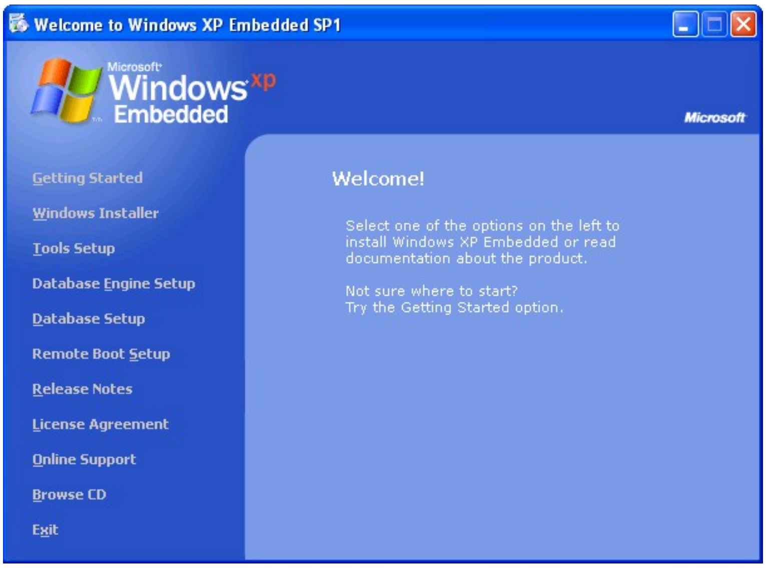
Note: You can install the XPe Database on a separate SQL Server for team development. This guide assumes that development tools and database are installed on the same system (Standalone). Please review the on-line help for more information about SQL installation and usage.

2.2 Installing the Tools

The XPe SP2 demo kit and OEM kit come with several CDs. XPe has a sequential installation processes. The installation is organized so you install XPe SP1 first, then install XPe SP2 on top of the XPe SP1, and finally install XPe Feature Pack 2007 on top of XPe SP2. You will need to download Feature Pack 2007 from the MSDN website. A database is at the center of the development process. You must decide if the database will reside on your local system or on an SQL server. This guide assumes that you are setting up the database on your development system in a standalone configuration.

2.2.1 Part 1: XP Embedded SP1 Setup

Insert CD1 into your development system. The opening installation screen provides the different installation steps.



Welcome!

Select one of the options on the left to install Windows XP Embedded or read documentation about the product.

Not sure where to start? Try the Getting Started option.

Step1: Windows Installer for Windows 2000 (only). If your host is a Windows 2000 Professional system, you may need to update Windows Installer to v2.0. During the following steps, you may need to re-boot your Windows 2000 system to complete the installation process.

Step2: Tools setup on the host systems.

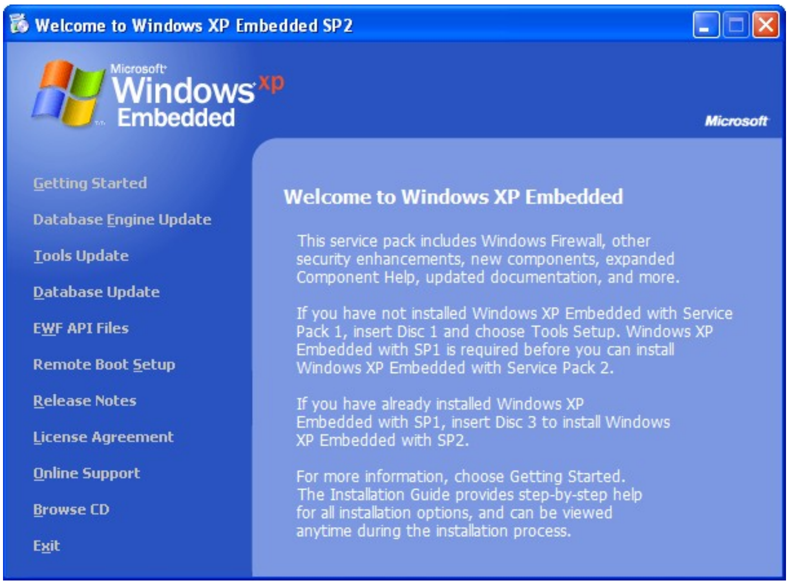
Step3: Database Engine Setup: Installs the Microsoft Data Engine on your development system. You will need to reboot the system in order to install the XPe SP1 database in Step 4.

Step4: Database setup: Installs the XPe SP1 database

Step5 (Optional): Remote Boot Setup: If your network environment supports remote boot or PXE server, you can download XPe images to a target with that has a PXE client built into the network card. TFTP and Remote Boot manager services will be installed. These services are intended to be installed on Windows Server 2000 with DHCP server configured and enabled.

2.2.2 Part 2: XP Embedded SP2 Setup

Now that XPe SP1 is installed, you will need to run the setup CD containing the updates for SP2.



Welcome to Windows XP Embedded

This service pack includes Windows Firewall, other security enhancements, new components, expanded Component Help, updated documentation, and more.

If you have not installed Windows XP Embedded with Service Pack 1, insert Disc 1 and choose Tools Setup. Windows XP Embedded with SP1 is required before you can install Windows XP Embedded with Service Pack 2.

If you have already installed Windows XP Embedded with SP1, insert Disc 3 to install Windows XP Embedded with SP2.

For more information, choose Getting Started. The Installation Guide provides step-by-step help for all installation options, and can be viewed anytime during the installation process.

Step1: Database Engine Update – This update addressed the vulnerabilities of MSDE.

Step2: Tools Update – Updates several of the development tools. You should reboot the system once you have completed Steps 1 and 2

Step3: Database Update – Updates the database and repositories to include the binaries for XP SP2.

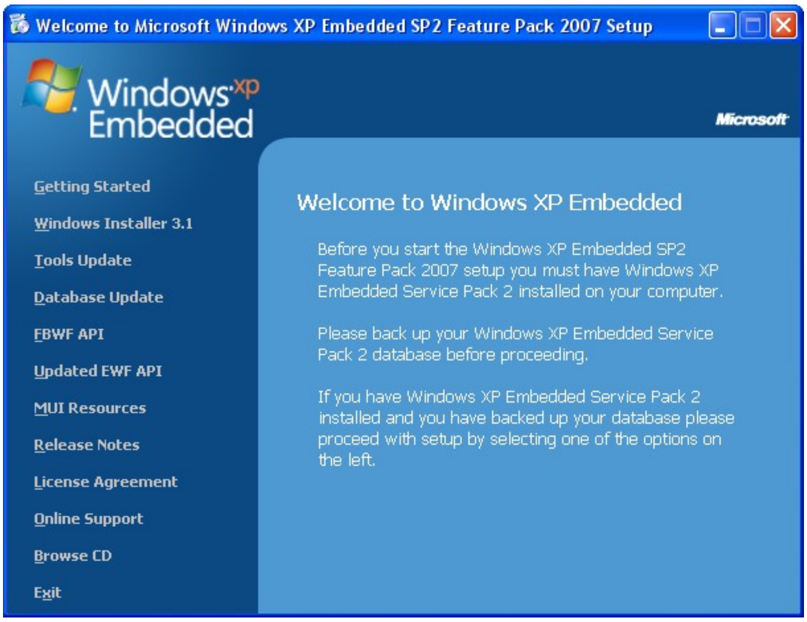
Step4: (Optional): Remote Boot Setup: This version supports Windows Server 2003. If your network environment supports remote boot or PXE server, you can download XPe images to a target which has a PXE client built into the network card. TFTP and Remote Boot manager services will be installed. These services are intended to be installed on Windows Server 2003 with DHCP server configured and enabled.

2.2.3 Part 3: Install Feature Pack 2007

To install Feature Pack 2007 you will need to download the trial version from the MSDN website: http://msdn.microsoft.com/embedded/windowsxpembedded/default.aspx

Check the download site:
http://msdn.microsoft.com/embedded/downloads/xp/default.aspx

The download contains an update to Windows XPe Service Pack 2. You will just need to run the installer to install on top of the installation that you have completed so far.



Welcome to Windows XP Embedded

Before you start the Windows XP Embedded SP2 Feature Pack 2007 setup you must have Windows XP Embedded Service Pack 2 installed on your computer.

Please back up your Windows XP Embedded Service Pack 2 database before proceeding.

If you have Windows XP Embedded Service Pack 2 installed and you have backed up your database please proceed with setup by selecting one of the options on the left.

Step1: Tools Update – Updates several of the development tools.

Step2: Database Update – Updates the database and repositories to include the binaries for Feature Pack 2007.

Option: You can also install or extract the API sets fro EWF and FBWF if you are going to write applications that control EWF FBWF.

2.3 *Importing Components*

2.3.1 Part 1: Extract the Kit Files

The XPe Development CD contains a self extracting zip file with the materials needed to build the configurations.

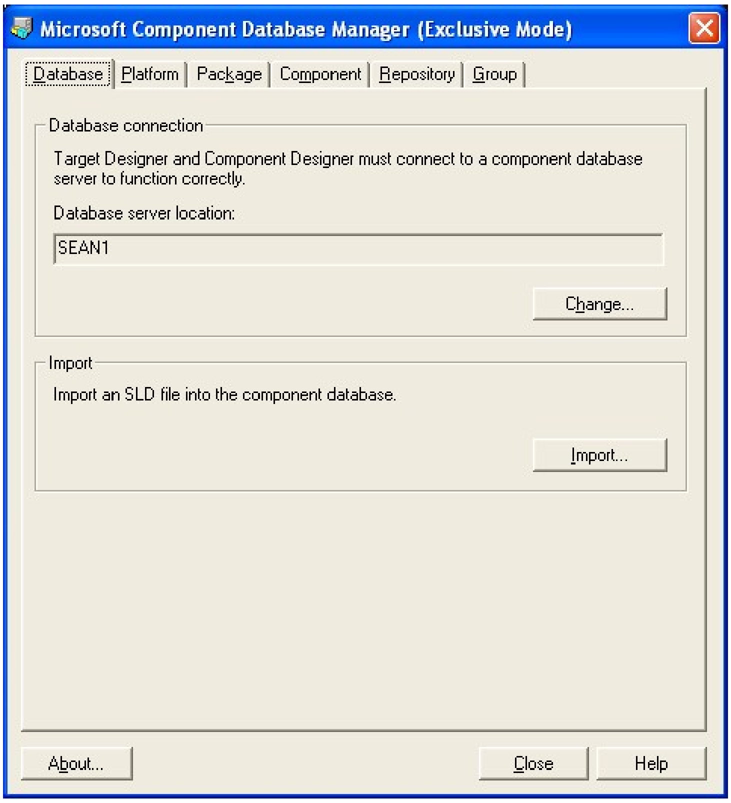
1. **Insert** the CD into the development machine.
2. **Open** File Explorer
3. Right click on the **SJJ601.zip** and select “Extra All” from the context menu to extract the contents to the development machine’s hard drive. Take note where you install the files since we will be using them throughout the text.

2.3.2 Part 2: Import the SLD files

The next step is to import the eBOX-III XPe components into the database. To make this step simple, a single SLD file (SJJ601_TDK.SLD) has been provided that contains all the components needed for the hardware and exercises.

> ***Note:*** Make sure that you import the components directly from their folder locations. The various subfolders are linked to the SLD file that is to be imported. These subfolders will be copied into the XPe repository, so Target Designer knows where to get the files when building an image.

1. Open **Component Database Manager**. CDM opens to the Database tab.



Database connection

Target Designer and Component Designer must connect to a component database server to function correctly.

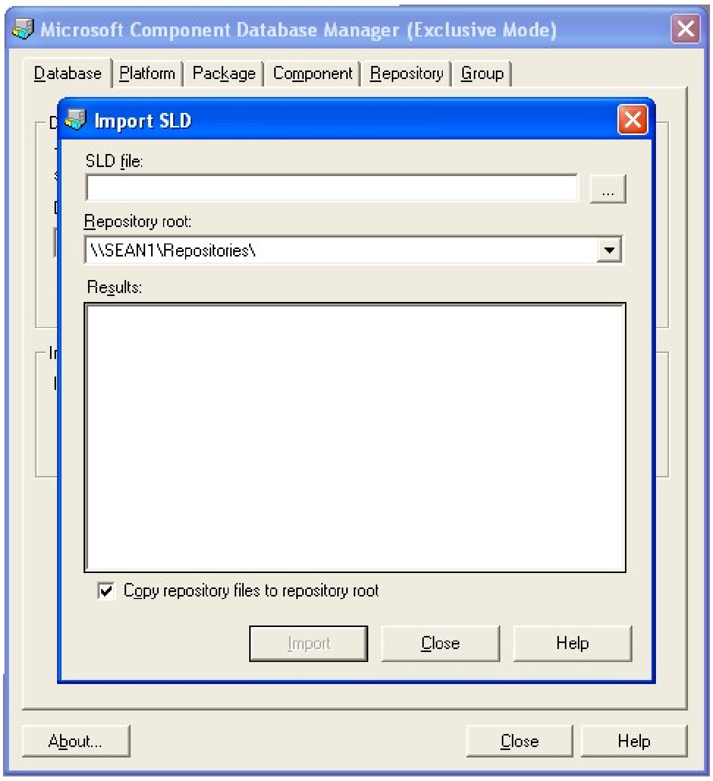
Database server location:

SEAN1

Import

Import an SLD file into the component database.

2. Click on the **Import** button. This will open the Import SLD dialog.

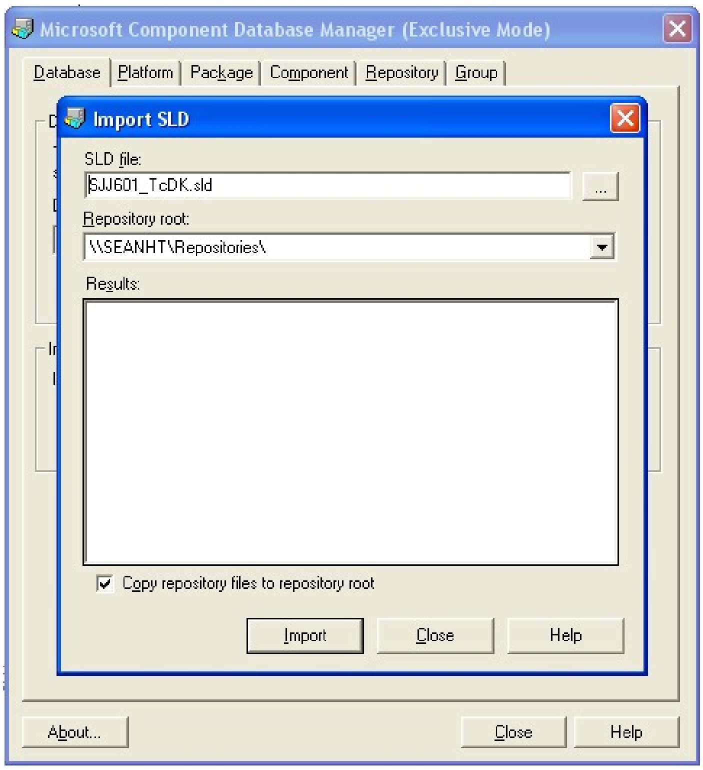


SLD file:

Repository root: \\SEAN1\Repositories\

☑ Copy repository files to repository root

3. Click on the (**…**) button, and locate the SJJ601_TcDK.sld file on the CD.
4. Click Import.



SLD file: SJJ601_TcDK.sld

Repository root: \\SEANHT\Repositories\

☑ Copy repository files to repository root

3 Background on the Components

In the last section, you imported the SJJ601_TcDK SLD into the database. The pre-built components were created to speed up the development process. This section discusses some details on the components and how they were created. You can skip this section and move to section 4, if you want to get going and build an image.

The SJJ601_TcDK.SLD contains the following components:

Component	Description	Notes
Vinyl AC'97 Codec Combo Driver (WDM)	Vinyl AC97Audio driver components	
VIA Rhine II Fast Ethernet Adapter	Ethernet adapter components	
VIA/S3G UniChrome IGP	Video driver component	
ICOP_eBox III 38XX Series	Platform component for the eBox III 38XX platforms	Platform component can be re-used for different projects.
SJJ601_Transfer	USB boot image	To be used to help deploy the images for the different exercises
SJJ601_Example1, 2, & 3	Three exercise components	
MediaPlayer Shell	Shell for Exercie 4	
RDP ShortCuts	Shortcuts for Exercise 3	

3.1 Where to begin: Target Analyzer

There are over 10,000 components that make up the XPe database, and a majority of those components are related to hardware. The starting point for any new XPe project is to run Target Analyzer on the hardware to capture information about the target hardware configuration and pull in the core components that are required to boot the OS on the hardware. There are two different versions of Target Analyzer: TA.EXE a 16bit MS-DOS application and TAP.EXE a Win32 application. TA.EXE gathers a very small set of information based on sniffing the PCI bus. In contrast, TAP.EXE running in Windows XP Pro, looks inside the registry and finds all the enumerated drivers that are in the system, thus provided the most detailed and complete information on the platform. Both utilities generate a PMQ file that contains the information that the import processor will use to compare against the database.

Windows XP Pro with SP2 and the eBOX-III device drivers were installed on the eBOX-III. TAP.EXE was run on the system to capture all the enumerated hardware details including multimedia streaming support. The resulting component is the ICOP_eBox III 38XX Series.

> ***Note:*** If you don't have the audio driver installed, TAP.EXE will find the audio codec device but not the Microsoft streaming support required for audio. As a result you may have an audio driver that loads correctly in the XPe image but has no audio available because of the missing streaming components.

The platform macro component has been created to save you time, so you do not have to install XP Pro and run TAP.EXE yourself. Any devices that you want to attached to the USB, Parallel, and serial ports will require a component, if there isn't one in the database.

3.2 Missing Driver Components and Support Applications

When a PMQ file is imported into Component Designer, it is a good idea to create a log file. The log file will provide information on what was found in the database, and most importantly, what wasn't found. In the case of the eBOX-III, there were several drivers that were not in the database: audio, video, and Ethernet.

Every XP driver must (or should) have an INF file that is used for setup. Component Designer can import an INF file to generate a driver component. The INF files for each of the missing device drivers were imported to create the various SLD files. In some cases the INF file was part of a setup utility that must be installed in order to obtain the INF file. Component Helper from the *Windows XP Embedded Supplemental Toolkit* was used to capture the INF file, driver files, and any application files during installation. For example, the Vinyl AC97 Audio components were generated using Component Helper. The driver component was separated from the extra audio applications that come with the system. The video adapter and Ethernet components were simply generated with the INF files.

3.3 Splash Screen SJJ ICOP– Device Branding

One of the most commonly requested features is to remove all Windows splash screens during booting, and in some cases, OEMs want their own splash screen to appear for branding purposes. This can be accomplished with the /bootlogo switch in the boot.ini file and a custom boot.bmp file in the \Windows directory.

There is no option to set this boot logo switch from within target designer. The Splash Screen SJJ ICOP component provides an example on how to add your own splash screen without a lot of POST-build / POST-FBA work. The solution adds a new boot_new.ini file to the root of the image. During FBA, the PostFBA.EXE exchanges the boot.ini file created with the Target Designer with the new boot_new.ini file. The boot.bmp file in the \Windows directory will be displayed on the screen during a normal XPe boot.

> ***Note***: If your system requires further boot switch settings, add them to the boot_new.ini and re-import the SLD file.

If you want to add your own splash screen, just edit the boot.bmp file with a graphics editor.

3.4 Custom Components: SJJ601_Examples

The exercise macro components are used with each of the exercises. These components contain dependencies on the software layer components

There are also two components that are used in some of the exercises: RDP ShortCuts and MediaPlayer Shell. The RDP ShortCuts component contains two shortcut resources that place shortcut links on the desktop and start bar. The RDP ShortCuts component will be use in Exercise 3. The MediaPlayer shell is an example of a custom shell component that will be used in Exercise 4. Any application can be used as a shell in XPe, including Media Player, so you can use this component as a basis for making your own application as a shell.

4 Image Deployment Setup for USB Boot

Before we get into the exercises and build images, we need to discuss how to get an XPe image loaded onto the eBOX-III's internal flash drive. Transferring the image from the development system to the target system is truly left up to the OEM. Every target system is different. There are a couple of issues that must be addressed when discussing deployment:

1. XPe images are big, > 35 MB, so a practical deployment method must be used to transfer the OS to the target that can transfer a large image quickly.
2. Long file and folder names MUST be preserved. Using XCOPY under DOS will not work. A 32-bit environment or alternative solution must be used to transfer the OS, so that all long filenames and long folder names will be preserved in the copied image.

The eBOX-III doesn't have built in CD ROM or floppy drives, so that rules out certain transfer methods. With Feature Pack 2007, we can take advantage of the XPe USB Boot 2.0 solution and the eBox III 38XX's USB flash boot BIOS support. The kit comes with the SJJ601_Transfer macro component that builds the custom XPe image. The resulting XPe image allows deploying an XPe image from a directory on the USB disk or network share.

Here are the steps to create the bootable USB flash disk that will be use to deploy the exercise images in the next section:

1. Make sure that you have imported the SJJ601_TcDK into the XPe database.
2. **Open** Target Designer.
3. Create a new configuration called **XPe Transfer**.
4. Add the **SJJ601_Transfer** component to the configuration. The component can be found in the "SJJ Embedded Micro Solutions TcDK SJJ601" category.
5. The next step is to run a **dependency check**. You can run Check Dependencies several ways: from the menu bar: Configuration->Check Dependencies, click on the Check Dependencies on the tool bar, or hitting F5. Check Dependencies looks at the configuration to make sure that all the components in the configuration have their dependent component or component group relationships satisfied. The Check Dependencies should finish with no errors or warnings. Make sure *Auto-resolve dependencies* is selected in Target Designer options.
6. The final step is to **build the image**. Like Check Dependencies, building the image can be initiated several different ways: from the menu: Configuration->Build Target Image..., click on the hammer in the tool bar, or hit F7.
7. A build dialog will appear. The destination should be set to the "\Windows Embedded Images" folder. Keep all the defaults and click on the **Build** button.
8. If another dialog appears asking to delete the contents in the "\Windows Embedded Images" folder, click the Yes button. The image should build with 1 warning. The warning is to indicate that the image is a test image and is only good for approximately 120 days.
9. **Save** the configuration.

The next step is to prepare the flash disk using the new UFDprep.exe utility.

10. Insert the flash disk in to the development PC.
11. Open a command window
12. Run the **UFDprep.exe** utility, which can be found under "\Program Files\Windows Embedded\Utilities":

    c:\>ufdprep <drive>, where <drive> is the drive letter for the USB flash disk you want to format

13. Once the format has completed, **copy** the XPe image to the USB flash disk.
14. Undock the USB flash disk.
15. Insert the USB flash disk into the target.

16. Boot the target and hit the DEL key to enter the BIOS setup.
17. In the BIOS Advandced Settings, set the USB ARMD Emulated as Hard Disk. Also set only IDE-0 as the first boot device.
18. Save the settings, and exit the BIOS setup.
19. Let the system boot from the USB flash disk. The system will run through FBA and the XPe build will boot.
20. Create a subdirectory on the flash disk called XPe Image. This subdirectory will be used to transfer the local image. The USB flash disk is now ready for use.
21. Power down the target system and remove the USB flash disk.

5 Exercise 1: Creating a Basic Image

The first exercise is to build a very simple and very basic XPe image, so you can get familiar with building and deploying the OS image. The shell will be the Command Shell, and we will use the ICOP_eBOX III 38XX Series as the platform component. You should have completed the importing of the SJJ601_TcDK.SLD file in section 2 and created the transfer image for the USB drive in section 4 before starting this exercise.

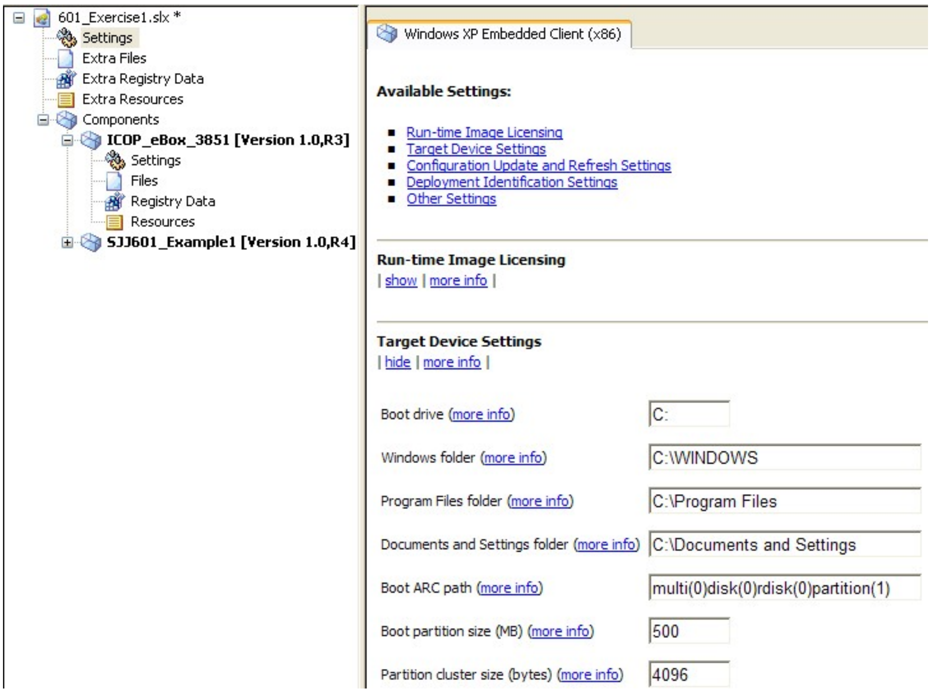
5.1 Build the Configuration Yourself

Target Designer is used to create and build the custom XPe images. In this section we first create a new configuration, add components to the configuration, make some setting changes, resolve all component dependencies, and then build the image.

1. Open Target Designer.
2. From the menu, select **File->New.**
3. The New configuration dialog appears. Type in the new configuration name: **601_Exercise1.**
4. Using the Component Browser on the left-hand side, add the following components:

   - **ICOP_eBox III 38XX Series** (Located under: Software->Test & Development->Hardware->Platforms).
   - **SJJ601_Example1** (Located under SJJ Embedded Micro Solutions TcDK SJJ601).

5. Now we need to edit some of the settings for the components we just added. At the very top of the configuration, under 601_Exercise1, click on **Settings**.
6. The details pane on the far right should change. Click on the “**Show**” under Target Device Settings.
7. Change the Boot partition size (MB) to **500.**



Boot drive (more info) C:

Windows folder (more info) C:\WINDOWS

Program Files folder (more info) C:\Program Files

Documents and Settings folder (more info) C:\Documents and Settings

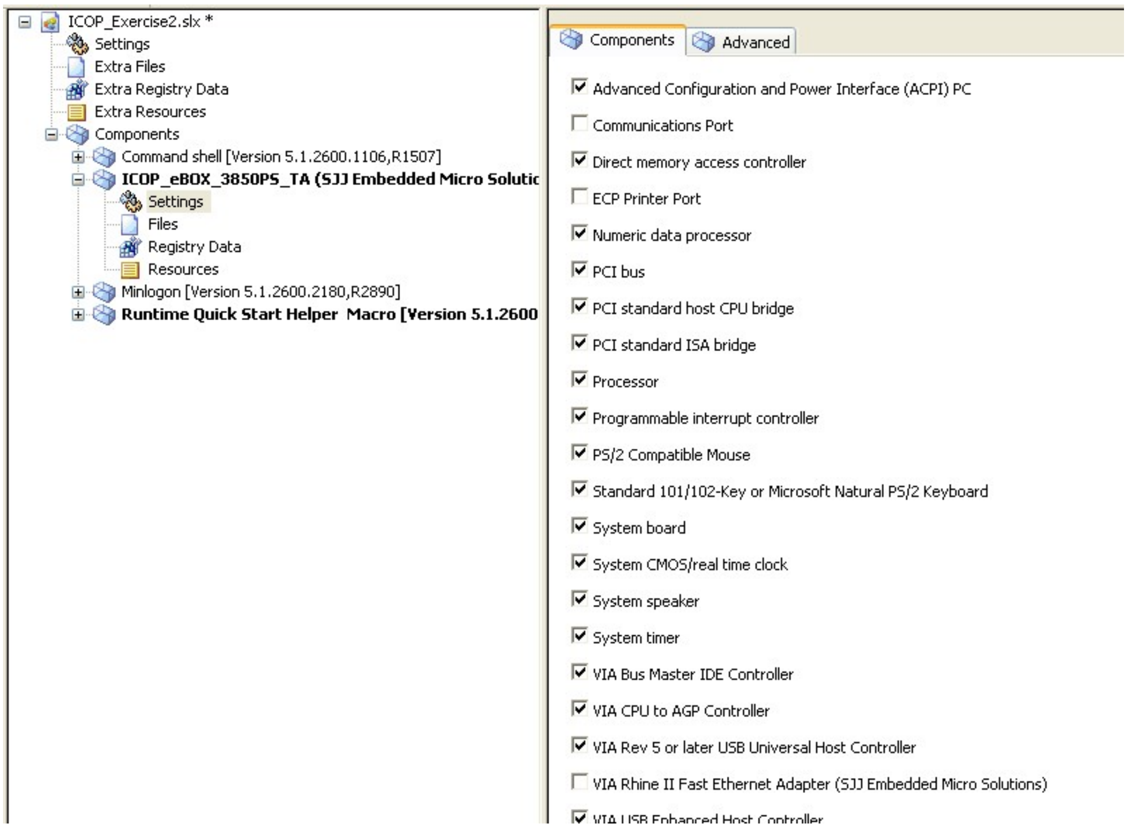
Boot ARC path (more info) multi(0)disk(0)rdisk(0)partition(1)

Boot partition size (MB) (more info) 500

Partition cluster size (bytes) (more info) 4096

8. Now select **Settings** under the ICOP_eBOX III 38XX Series component
9. To create smaller images you will want to remove items that are not needed in an image. In this case, we will remove some hardware components before we perform a Check Dependencies. The details pane on the right will change to the settings for the ICOP_eBOX III 38XX Series component. Uncheck the following items:

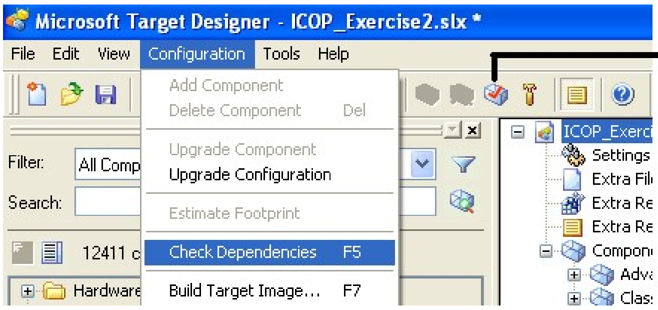
    - Communications Port
    - Direct Parallel
    - ECP Printer Port
    - VIA Rhine II Fast Ethernet Adapter
    - Vinyl AC'97 Codec Combo Driver
    - RAS Async Adapter
    - Terminal Server Device Redirector
    - Terminal Server Keyboard Driver
    - Terminal Server Mouse Driver
    - WAN Miniport (IP)
    - WAN Miniport (L2TP)
    - WAN Miniport (PPPOE)
    - WAN Miniport (PPTP)


ICOP_Exercise2.slx *
Settings
Extra Files
Extra Registry Data
Extra Resources
Components
Command shell [Version 5.1.2600.1106,R1507]
ICOP_eBOX_3850PS_TA (SJJ Embedded Micro Solutic
Settings
Files
Registry Data
Resources
Minlogon [Version 5.1.2600.2180,R2890]
Runtime Quick Start Helper Macro [Version 5.1.2600
Components
Advanced

☑ Advanced Configuration and Power Interface (ACPI) PC
☐ Communications Port
☑ Direct memory access controller
☐ ECP Printer Port
☑ Numeric data processor
☑ PCI bus
☑ PCI standard host CPU bridge
☑ PCI standard ISA bridge
☑ Processor
☑ Programmable interrupt controller
☑ PS/2 Compatible Mouse
☑ Standard 101/102-Key or Microsoft Natural PS/2 Keyboard
☑ System board
☑ System CMOS/real time clock
☑ System speaker
☑ System timer
☑ VIA Bus Master IDE Controller
☑ VIA CPU to AGP Controller
☑ VIA Rev 5 or later USB Universal Host Controller
☐ VIA Rhine II Fast Ethernet Adapter (SJJ Embedded Micro Solutions)
☑ VIA USB Enhanced Host Controller

10. Now that the changes have been made, the next step is to run a dependency check. You can run Check Dependencies several ways: from the menu bar: **Configuration->Check Dependencies,** click on the Check Dependencies on the tool bar, or hit **F5**. Check Dependencies looks at the configuration to make sure that all the components in the configuration have their dependent component or component group relationships satisfied. The Check Dependencies should finish with no errors or warnings. Make sure

Auto-resolve dependencies is selected in Target Designer options before running Check Dependencies.



Tool Bar Button: Run Dependency Check

11. The final step is to build the image. Like Check Dependencies, building the image can be initiated several different ways: from the menu**: Configuration->Build Target Image...**, click on the hammer in the tool bar, or hit **F7**.
12. A build dialog will appear. The destination should be set to “\Windows Embedded Images” folder. Keep all the defaults and click on the **Build** button.
13. If another dialog appears asking to delete the contents in the “\Windows Embedded Images” folder, click the **Yes** button. The image should build with 1 warning. The warning is to indicate that the image is a test image and is only good for approximately 120 days. The image should be around 110MB in size.

5.2 Deploying the Operating System

The USB flash disk with XPe image created in section 4 will be used to deploy the OS image to the target system.

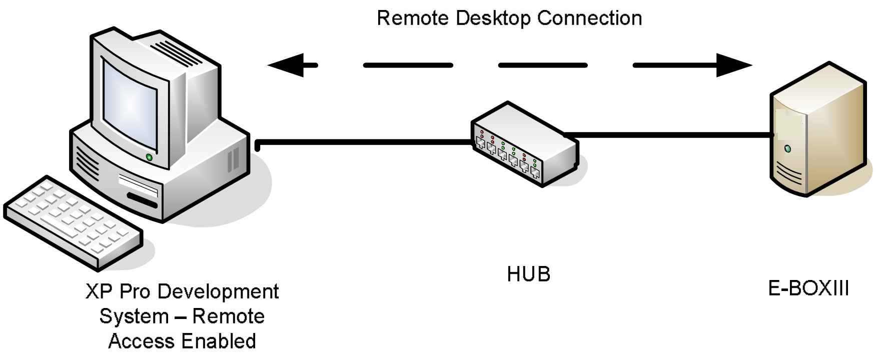
1. Insert the USB flash disk that has the XPe transfer image into the development system.
2. **Copy** the 601_Exercise1 XPe image found in Windows Embedded Images to the XPe Image folder on the USB flash disk. Copy the full OS image directory tree.
3. **Unmount** the USB flash disk.
4. Insert the USB flash disk into the eBox 3851.
5. **Turn on** the power to the eBox 3851.
6. Once the system boots to the XPe image, go to the Control Panel->Administrative Tools-Computer Management->**Disk Management.**
7. **Format** the D drive with FAT or NTFS.
8. **Copy** the image in the XPe image folder to the D drive. Copy the full OS image directory tree.
9. When the copy is completed, **power down** the system.
10. **Remove** the USB flash disk.
11. **Turn on** the power to the eBox 3851 and let the system boot the internal flash disk. The system will boot the XPe image and start FBA.

The system boots the Command Shell. There is not much that the image can do, but we successfully demonstrated how to build an image and download the image to the target system.

6 Exercise 2: Thin Client Image with EWF

The eBOX-III is an ideal platform for industrial thin client use. Now that we are familiar with building and deploying images to the eBOX-III, let's build a basic thin client image that contains the Remote Desktop Connection application, which is one of the most popular thin client applications on the market. The RDC will connect to your XP Pro development workstation that has Remote Desktop access enabled. Explorer is going to be the shell. A special component called the RDP ShortCuts provides the shortcuts that will appear on the desktop and Start bar. The two shortcut resources are good examples of how to create shortcuts in Component Designer.

To complete the exercise, you will need to have an Administrator or Power user account with a password on the development station. You will also need to know the TCP/IP address of the XP Pro system. This lab assumes that the network supports DHCP.


Remote Desktop Connection
XP Pro Development System – Remote Access Enabled
HUB
E-BOXIII

The image will also include the Enhanced Write Filter (EWF) setup with a RAM overlay. EWF protects a partition from write access, but maintains the writes in an overlay so the partition still looks read/writable. Since the overlay is RAM REG, EWF will be protecting the boot partition, thus protecting the OS from corruption. A command line utility, EWFMGR.EXE, can be used to control the state of EWF: enable, disable, and commit changes.

6.1 EWF Manager Console Application

XPe includes the Enhanced Write Filter console application command-line tool, Ewfmgr.exe, and can be used to issue a set of commands to the Enhanced Write Filter (EWF) driver, report the status of each protected volume overlay and report the status of the overall EWF configurations. Just add the EWF Manager Console Application component to your configuration to add the Ewfmgr.exe application. In addition to the EWFMGR.EXE application, there is also an EWF API set that allow you to build the same EWF commands into your custom application.

Usage: **EWFMGR <*drive-letter*>(optional) [*options*]**

Note *Because most EWF manager commands are executed on the next boot, you must reboot the system for the command to take effect.*

Parameters:

drive-letter

Specifies the volume path. Optional parameter to specify the protected volume. To view the status of the protected volume, specify the drive letter for the protected volume, for example, ewfmgr c:.

options

Specifies the EWF volume boot options.

The following table lists the EWF manager boot commands.

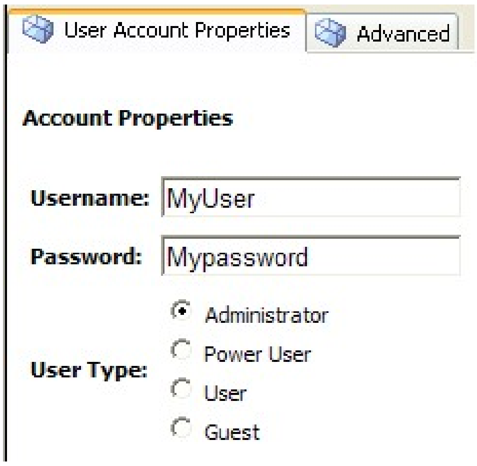
Boot command	Description
All	Displays information about all protected volumes and performs a command, such as **disable**, **enable**, **commit**, **checkpoint**, and **restore**, on each volume if specified.
Checkpoint	Starts a new overlay level. Same as **SetLevel**= [Current Overlay Level + 1].
Commit	Commits all current level data in the overlay to the protected volume, and resets the current overlay level to 1. **Commit** can be combined with the **Disable** command to commit and then disable.
CommitandDisable	Commits all current level data in the overlay to the protected volume and disables the overlay. The overlay is written to the protected volume on the next system boot. Committing the overlay can impact the speed of the boot process. You can use the -live command for EWF RAM Mode to immediately commit the overlay to the protected volume and disable the overlay without having to reboot the system. For example, ewfmgr c: -commitanddisable -live The -live command is supported on EWF RAM mode only.
Description	Allows the user to associate an ASCII string with an overlay level. This command can be combined with the **SetLevel** command.
Disable	Disables the overlay on the specified protected volume.
Enable	Enables the write filter so that data written to the protected media is cached in the overlays. The current overlay level becomes 1 as soon as EWF is started, and a new overlay is created at level 1.
Gauge	Displays the percentage to which the EWF volume has been filled.
NoCmd	Clears the current pending command.
Persist	Specifies a 64-byte field that persists throughout all overlays for a specific protected volume.
Restore	Restores to the prior overlay. Same as **SetLevel**=[Current Overlay Level – 1]

Boot command	Description
SetLevel	Sets the current overlay level to the specified level. The current overlay level is obtained from Ewfmgr.exe. Valid values for levels are: [Current overlay level +1]. Starts a new overlay level. [0 - Current overlay level]. Sets the level, discarding all data above the specified level. [- Level]. Deletes all the data in the specified level and beyond.

6.2 Build the Configuration Yourself

Example 1 was a simple image, and this exercise will be a little more involved. In this example we will build a Thin Client platform that will connect to the desktop system via RDP. The image will boot to the Explorer shell and shortcuts for Remote Desktop Connection will be available to connect to the remote PC. EWF will be setup to protect the life of the flash disk and will be controlled using EWFMGR.EXE.

1. Open **Target Designer.**
2. From the menu, select **File->New.**
3. The New configuration dialog appears. Type in the new configuration name: **601_Exercise2.**
4. Using the Component Browser on the left hand side, add the following components:
    - **SJJ601_Exercise2** (Located under SJJ Embedded Micro Solutions TcDK SJJ601).
5. Now we need to edit some of the settings for the components we just added. At the very top of the configuration, under 601_Exercise2, click on **Settings**.
6. The details pane on the far right should change. Click on the “**Show**” under Target Device Settings.
7. Change the Boot partition size (MB) to **500**.
8. In the configuration, locate and expand the **User Account** component and click on **Settings**.
9. Edit the following property settings with account information that matches the user account on the desktop system that the target will connect to. The picture below is just an example. Make sure to use the appropriate Username and Password for your system.



Account Properties

Username: MyUser

Password: Mypassword

User Type:
● Administrator
○ Power User
○ User
○ Guest

- UserName: <Enter the account name>
- Password: <Enter the accounts password>
- User Type: Administrator

10. In the configuration, locate the Enhanced Write Filter and click on **Settings**.
11. Set the Overlay Type to **RAM (Reg)**.
12. Now that the changes have been made, the next step is to run a dependency check. You can run Check Dependencies several ways: from the menu bar: **Configuration->Check Dependencies,** click on the Check Dependencies on the tool bar, or hitting **F5**. Check Dependencies looks at the configuration to make sure that all the components in the configuration have their dependent component or component group relationships satisfied. The Check Dependencies should finish with no errors or warnings. Make sure *Auto-resolve dependencies* is selected in Target Designer options before running Check Dependencies.
13. The final step is to build the image. Like Check Dependencies, building the image can be initiated several different ways: from the menu**: Configuration->Build Target Image...**, click on the hammer in the tool bar, or hit **F7**.
14. A build dialog will appear. The destination should be set to “\Windows Embedded Images” folder. Keep all the defaults and click on the **Build** button.
15. If another dialog appears asking to delete the contents in the “\Windows Embedded Images” folder, click the **Yes** button. The image should build with 1 warning. The warning is to indicate that the image is a test image and is only good for approximately 120 days. The image should be around 155MB in size.

6.3 Deploying the Operating System

The USB flash disk with XPe image created in section 4 will be used to deploy the OS to the target system.

6.3.1 Part 1: Deploying the OS

1. Insert the USB flash disk that has the XPe transfer image into the development system.
2. On the USB flash disk, delete any previous image found in the XPe Image folder. Delete the full OS image directory tree.
3. **Copy** the 601_Exercise2 XPe image found in Windows Embedded Images to the XPe Image folder on the USB flash disk. Copy the full OS image directory tree.
4. **Unmount** the USB flash disk.
5. Insert the USB flash disk into the eBox 3851.
6. **Turn on** the power to the eBox 3851.

7. Once the system boots to the XPe image, go to the Control Panel->Administrative Tools-Computer Management->**Disk Management.**
8. **Format** the D drive with FAT or NTFS.
9. **Copy** the image in the XPe image folder to the D drive. Copy the full OS image directory tree.
10. When the copy is completed, **power down** the system.
11. **Remove** the USB flash disk.
12. **Turn on** the power to the eBox 3851 and let the system boot the internal flash disk. The system will boot the XPe image and start FBA.

6.3.2 Part 2: Check EWF

You should be able to run CMD.EXE from start->run and then run EWFMGR.EXE to check the status of EWF. The status for drive C should show the Type is *RAM REG* and State should be *Enabled*:

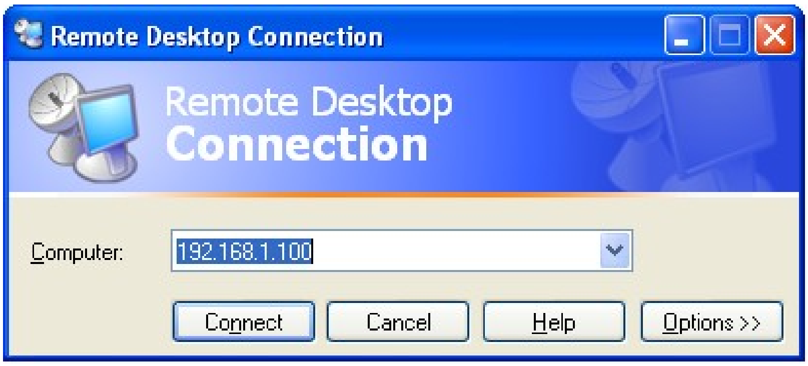
```
C:\Windows\System32>ewfmgr

RAM (REG) Configuration
    Device Name         "\Device\HarddiskVolume1" [C:]
    HORM                Not Supported

C:\Windows\system32>ewfmgr c:

    Protected Volume Configuration
     Type           RAM (REG)
     State          ENABLED
     Boot Command    NO_CMD
      Param1        0
      Param2        0
     Persistent Data ""
     Volume ID       7A 36 2F E4 00 7E 00 00 00 00 00 00 00 00 00 00
     Device Name     "\Device\HarddiskVolume1" [C:]
     Max Levels      1
     Clump Size      512
     Current Level   1

     Memory used for data 7936512 bytes
     Memory used for mapping 4096 bytes
```

Try creating a file or folder on the system, and then rebooting the target to see if the file or folder remains. The file or folder should be lost since it was stored in the RAM overlay and lost during the reboot. You can temporarily disable EWF using EWFMGR.EXE in order to add programs or make changes to the system.

6.3.3 Part 3: Run Remote Desktop Connection

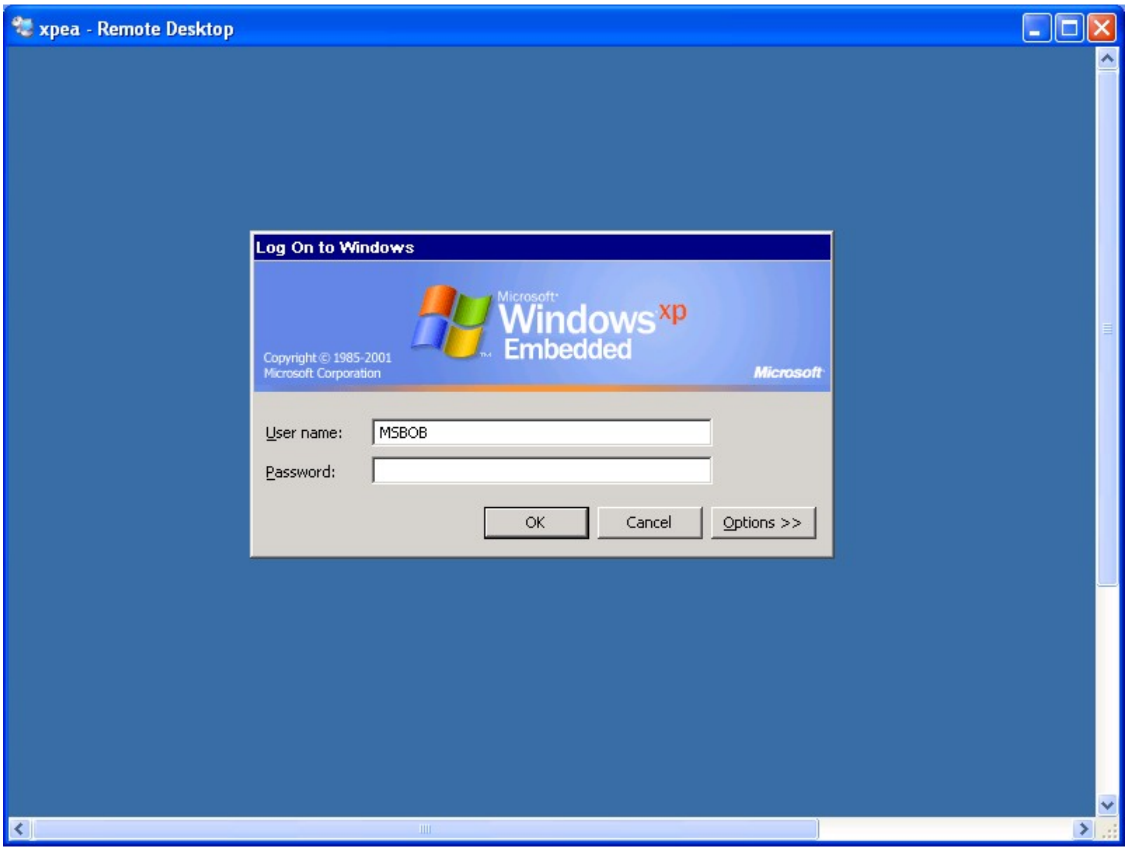
Now let's try the Remote Desktop Connection.

1. On the development system, use network settings or IPCONFIG /all to get the TCP/IP address of the development system.
2. On the eBOX-III, click on the Remote Desktop link found on the desktop.
3. A dialog appears asking for the computer name. Type the TCP/IP address of the development system and click the **Connect** button.



Computer: 192.168.1.100

4. The remote connection is established, and the remote window appears. A Windows Logon dialog box appears in the remote window. Type in the user account and password to access the system.


xpea - Remote Desktop
Log On to Windows
Microsoft Windows xp Embedded
Copyright © 1985-2001 Microsoft Corporation
Microsoft
OK
Cancel
Options >>

User name: MSBOB

Password:

5. The Remote Desktop logs in, and you will notice that the logon screen appears on the target system. XP only supports one connection to a machine at a time. If you logon to the target locally, the remote desktop connection would be terminated.

7 Exercise 3: Multimedia Station with Splash Screen

The eBOX-III has an integrated AC97 for sound. The size of the eBOX-III makes the platform ideal for a small multimedia solution. We will now use the full platform macro component in this exercise to playback music from MediaPlayer. To make this image a little different, we will use a custom shell: MediaPlayer Shell. There is also an example of a Splash screen to replace the standard Windows splash screen. All you have to do is add the SJJ_SPLASH screen component; the FBA resource will replace the BOOT.INI with one that supports the splash screen.

You will need to connect speakers or headphones to the audio out port for this exercise.

7.1 Build the Configuration Yourself

The following are the steps to re-create the configuration.

1. Open **Target Designer.**
2. From the menu, select **File->New.**
3. The New configuration dialog appears. Type in the new configuration name: **601_Exercise3.**
4. Using the Component Browser on the left hand side, add the following components:

    - **SJJ601_Exercise3** (Located under SJJ Embedded Micro Solutions TcDK SJJ601).

5. Now we need to edit some of the settings for the components we just added. At the very top of the configuration, under 601_Exercise3, click on **Settings**.
6. The details pane on the far right should change. Click on the "**Show**" under Target Device Settings.
7. Change the Boot partition size (MB) to **500**.
8. Now that the changes have been made, the next step is to run a dependency check. You can run Check Dependencies several ways: from the menu bar: **Configuration->Check Dependencies,** click on the Check Dependencies on the tool bar, or hitting **F5**. Check Dependencies looks at the configuration to make sure that all the components in the configuration have their dependent component or component group relationships satisfied. The Check Dependencies should finish with no errors or warnings. Make sure *Auto-resolve dependencies* is selected in Target Designer options before running Check Dependencies.
9. The final step is to build the image. Like Check Dependencies, building the image can be initiated several different ways: from the menu**: Configuration->Build Target Image...**, click on the hammer in the tool bar, or hit **F7**.
10. A build dialog will appear. The destination should be set to "\Windows Embedded Images" folder. Keep all the defaults and click on the **Build** button.
11. If another dialog appears asking to delete the contents in the "\Windows Embedded Images" folder, click the **Yes** button. The image should build with 1 warning. The warning is to indicate that the image is a test image and is only good for approximately 120 days. The image should be around 210MB in size.

7.2 Deploying the Operating System

Once the image has been built you might want to include some media files for testing the image playback.

7.2.1 Part 1: Deploying the OS

1. Insert the USB flash disk that has the XPe transfer image into the development system.
2. On the USB flash disk, delete any previous image found in the XPe Image folder. Delete the full OS image directory tree.
3. **Copy** the 601_Exercise3 XPe image found in Windows Embedded Images to the XPe Image folder on the USB flash disk.
4. **Unmount** the USB flash disk.
5. Insert the USB flash disk into the eBox 3851.
6. **Turn on** the power to the eBox 3851.
7. Once the system boots to the XPe image, go to the Control Panel->Administrative Tools-Computer Management->**Disk Management.**
8. **Format** the D drive with FAT or NTFS.
9. **Copy** the image in the XPe image folder to the D drive. Copy the full OS image directory tree.
10. When the copy is completed, **power down** the system.
11. **Remove** the USB flash disk.
12. **Turn on** the power to the eBox 3851 and let the system boot the internal flash disk. The system will boot the XPe image and start FBA.

7.2.2 Part 2: Running Media Player

1. Once FBA has completed, the image should boot to the MediaPlayer.
2. You may be asked a few setup options for MediaPlayer. Follow the Wizard's instructions and choose the defaults.
3. With the speakers or headphones attached to the eBOX-III, select a song to play.

Note: Closing media player will not force Windows to re-launch media player. You will have to reboot the system to bring back Media Player.

Note: Use the Power switch to turn on and off the system.

8 Summary

This guide provided the basic setup, building steps, and deployment methods to get XPe running on the eBOX III. The kit itself with pre-built components that address the core device drivers of the eBOX III allows you to get a jump start into XPe development. With the base components already completed, you only need to create components for the peripheral devices that you want to attach to the eBOX III to create these unique devices.

The exercises in this guide are a few brief examples of what can be done with the eBOX III and XPe. There are many other concepts and features to explore with the eBOX III and XPe, such as remote management, headless operation, mini-webserver, print servers, wireless connectivity, just to name a few.

A Bibliography

A.1 Books

Windows NT Embedded Step-By-Step, Sean D. Liming, Annabooks, 2000, ISBN: 0-929392-68-X

Windows XP Embedded Advanced, Sean D. Liming, RTC Books, 2003, ISBN:0-929392-77-9

Windows XP Embedded Supplemental Toolkit, Sean D. Liming, Cedar Hill Publishing, 2004, ISBN: 1-932373-96-9.

B SJJ Embedded Micro Solutions

SJJ Embedded Micro Solutions is dedicated to making learning embedded tools and solutions easier. Our books, toolkits, and training courses provide a hands-on approach to learning embedded systems. Our consulting services complement out products by offering support for all aspects of your embedded design hardware, software, manufacturing, and life-cycle of the product.

Developing an embedded system is more than just picking a few components and building an image. Developers have to take into account how the system is used, manufactured, and supported in the field. We take an architectural approach to deliver the right solution for your product. We have experience in a variety of markets; our previous XP Embedded projects have included:

- Thin clients
- Gaming Consoles
- Industrial Controls
- Voice over IP Systems
- Test Equipment
- Consumer Electronics
- Automotive

If you are looking for support on your Windows CE or XP Embedded project, we can support your project in different areas:

- Hardware design review and consultation
- OAL and Bootloader Development
- Develop CE or XP device drivers
- Create and integrate components for applications and device drivers
- Architect configurations for customer's hardware
- Implement security policies
- Enterprise connectivity and integration
- Implementation of XPe embedded features such as EWF, DUA, Cloning, Remote boot, CD-Boot, CF, SMS, and SUS, and USB flash boot.
- Application development including .NET, EWF APIs, and XPe Power Management APIs
- Develop architecture and build documentation
- Training for engineering staff

Please contact us for more information on consultation and availability.

Web: www.sjjmicro.com
E-mail: sales@sjjmicro.com

C Support

Please contact SJJ Embedded Micro Solutions (support@sjjmicro.com) for Windows XP Embedded training, educational material, or support.

Please contact ICOP for any hardware questions or technical assistance. http://www.embeddedpc.net/