Overview of System Development for Windows CE.NET
Overview

- Selecting a Windows Embedded Operating System
- The Windows CE Platform Development Cycle
- The Application Development Options
Selecting a Windows Embedded Operating System

- Battery Powered?
  - No
  - CPU Architecture?
    - X86 PC
    - OS RAM Footprint?
      - < 16MB
      - > 16MB
      - Use Existing Applications?
        - YES: Windows XP Embedded
        - No: Windows CE
        - Any CE Supported
          - YES
          - No: Windows CE
Selecting a Windows Embedded Operating System (con’t)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Windows CE</th>
<th>Windows XPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk Space</td>
<td>5MB ~ 56MB</td>
<td>500 k ~ 16 MB</td>
</tr>
<tr>
<td>Image size</td>
<td>200KB ~ 14MB+</td>
<td>5 MB ~ 35 MB+</td>
</tr>
<tr>
<td>Configurability</td>
<td>Extremely modular</td>
<td>Modular</td>
</tr>
<tr>
<td>API Support</td>
<td>Best of Win32 plus Windows CE–specific enhancements</td>
<td>Complete Win32, same as XP</td>
</tr>
<tr>
<td>CPU Support</td>
<td>X86, MIPS, SHx, ARM</td>
<td>Pentium class x86</td>
</tr>
<tr>
<td>Device Driver</td>
<td>Fine–tuned for size</td>
<td>Same as XP</td>
</tr>
<tr>
<td>Security</td>
<td>Improved over 3.0</td>
<td>Same as XP</td>
</tr>
</tbody>
</table>
The Windows CE Platform Development Cycle

- Getting Started Quickly With Platform Builder
- Taking Advantage of Parallel Development
- Typical Project Timeline
- Moving form SDB to Custom Hardware
- The Iterative Development Process
Getting Started Quickly with Platform Builder

- New Platform Wizard: provides a start
  - Select a base configuration for your platform
  - Click initial features
  - Click preconfigured BSP

- Basic configurations
  - Starting point for the creation of an operating system
  - Select from New Platform Wizard
Getting Started Quickly with Platform Builder (con’t)

- CEPC Emulator
  - A virtual target hardware platform on the workstation
  - Allows system development to begin without any target hardware attached, including applications and debugging
- SDBs and BSPs in Platform Builder
  - Choose the board closest to your own architecture
  - Sample BSPs (board support packages) are provided
  - Develop for board until your hardware is ready
Taking Advantage of parallel Development

Linear Development

Port H/W \(\rightarrow\) Customize OS \(\rightarrow\) Develop Apps

Parallel Development Model

Port H/W
Customize OS
Develop Apps
Moving from SDB to Custom Hardware

- On a Standard Development Board (SDB) and/or Emulation
  - Configure the operating system
  - Build the operating system image
  - Debug the operating system image
  - Generate a Platform SDK
  - Develop your applications
  - Integrate and test

- On your custom hardware when it is available
  - Create the Board Support Package (BSP)
    - Bootloader
    - OAL
    - Device Drivers
  - Rebuild the system using your new BSP
  - Debug and test on hardware
  - Produce and deploy
The Iterative Development Process

START

Configure Platform

Customize platform

Develop device drivers

Create or add custom components

Modify source code Configuration files

Download to target device

Build OS Image

Custom target device ?

Yes

Develop OAL Board support package And bootloader

Develop custom Application using Exported SDK

No

Platform complete ?

Yes

Export SDK

Custom apps complete ?

No

Debug Platform

Continue modifications

FINISH

Final Testing And verification

Yes

Platform complete ?

No

Custom apps complete ?

Windows CE.NET
The Application Development Options

Native Application Model

- eMbedded Visual Tools
- C or C++
- Win32 Applications

Managed Application Model

- Visual Studio.Net With Device extensions
- C# or VB.NET
- Applications for .NET Compact Framework

Compiled

Runtime

Windows CE.NET Operating System
The Embedded Visual Tools (Native) Application Model

- Based on Win32 API
  - Windows CE subset familiar to Win32 programmers
  - Some APIs specific to windows CE: Database and Thread handling
- Visible and non-visible windows with message processors
  - WndProc
  - WndMain
- When developing a program for Windows CE, you must first determine the hardware platform and processor on which your program is going to run
What is eMbedded Visual C/C++?

- Similar to Microsoft Visual C++
- Special features specific to Windows CE
  - Built in computers and linkers for supported processors
  - Debug tools, remote tools, and platform manager
  - Emulation environment to develop without device
- API differences between CE and Desktop
  - All redundant WIN32 API Functions are removed
  - If the function is part of an API that is not supported by Windows CE, you will have to find another solution
  - If Windows CE supports the feature, you must find functions in Windows CE that can be combined to support this feature
The eVC Application Development Environment

- C Run-time Library
  - Retail and Debug version
  - Subset of the full C library

- Active Template Library
  - Provides a range of ActiveX controls and COM features
  - Componentized to be smaller than MFC

- Component Object Model
  - Is a fundamental object model on which ActiveX controls and OLE are built

- ActiveX
  - Is a dynamic-link library (DLL) or executable (.exe) that contains one or more COM components

- Microsoft Foundation Class Library (MFC)
  - Complete object-oriented application framework
The Managed Application Model

- Smart Device Extensions for Visual Studio .NET (SDE)
  - Is a Visual Studio integration package which plugs into Visual Studio .NET
  - Includes a set of pre-built device profiles
    - Allows you to create applications for Windows CE devices using WinForms, ADO .NET, and XML Web services
- Languages all Compile to Intermediate Language format
  - C# application development
  - Visual Basic .NET application development
The Managed Application Model (con’t)

- .NET Compact Framework (CF)
  - Device-side runtime support package for .NET application

- Common Language Runtime (CLR)
  - Execution engine to manage .NET applications
  - Just-In-Time compiler for intermediate language format

- Class Library
  - Form-related classes, Data and XML classes, and GDI support
  - True subset of .NET Framework classes
Comparing Native and Managed Applications

- **Native Applications**
  - Must be rebuilt for each new CPU or Platform
  - Developer manages system resources
  - Can run without extra support files
  - Can access all operating system services and APIs
  - Must be ported to run on desktop systems
  - Support COM, ActiveX programming

- **Managed Applications**
  - Built once for all devices
  - Runtime engine manages system resources
  - Require runtime support files (.NET CF)
  - Applications access only the services exposed by the CF
  - Run directly on desktop CF without porting
  - Legacy interoperability not supported