Part Number: 30629-000-002

PTC-1800 Software Development Kit Programming Guide

PTC-1800

Software Development Kit Programming Guide

Part Number: 30629-000-002

Release Date: November 2000

The information in this manual is subject to change without notice.

Symbol shall not be liable for technical or editorial omissions or mistakes in this manual nor shall it be liable for incidental or consequential damages resulting from your use of the information contained in this manual.

Table of Contents

About This Manual

This manual was written by the Symbol Technical Publications Group. This group provides technical documentation that supports the Symbol PTC-1800 product line. This product uses the Windows CE Operating System. Every effort has been made to provide accurate and concise information to you, our customer.

Contacting Symbol's Support Center

Symbol's Support Center may be contacted to obtain help in resolving any PTC-1800 based system problem that you may experience.

The Product Support Center’s toll-free number is 1-800-653-5350. The center is open from 8 a.m. to 6 p.m. (EST), Monday through Friday.

For further information on Symbol's products and services, please visit our website at **www. symbol.com.**

This Manual’s Purpose and Scope

This Software Development Kit Programming Guide provides information pertaining to PTC-1800 units that operate in a Windows CE 2.12 based software environment. This manual is meant to provide information on the various software components of this powerful product.

Introduction

The PTC-1800 is a Portable Teletransaction Computer that converges advanced 32-bit transaction processing with voice support from the Symbol wireless local area network (LAN).

The PTC-1800 uses an Intel StrongArm RISC™ processor operating at 190 MHz and Microsoft Windows CE™ to deliver Voice-over-IP (VoIP) telephony, messaging, and transaction-processing capabilities. Users can leverage their wireless LAN infrastructure with the PTC-1800, thus providing access to wireless telephone, paging, and data collection capabilities.

By using the Windows CE operating system, application developers can easily create value-added applications based upon the Win32 API interface. The PTC-1800 also supports PenRight!'s MobileBuilder™ cross-platform Rapid Application Development tool for ease of application development. Using the Internet browser option, the PTC-1800 can access e-mail and web pages.

The PTC-1800 unit is small, lightweight, hand held, and ruggedly constructed. It is designed for convenient one-handed keypad operation to enter information or select options via a backlit transflective monochrome 1/8 VGA graphic display.

The PTC-1800 also supports a laser scanner to enter barcode information easily and incorporates radio communications to transmit or receive rapidly needed information to and from a host processor.

User Features

Standard features of the PTC-1800 include a simplified keypad, easy-to-view display, integrated auto-ID, an infrared data acquisition (IrDA) port for wireless printing, and a high-capacity lithium-ion battery pack. Advanced features include the ability to toggle among multiple applications and telephone calls without disconnecting the caller or losing data.

Scope

This document covers specific information needed for programming the PTC-1800 unit for Windows CE. The target audience is assumed to be familiar with programming for Windows and Windows CE. For specific details on Windows CE programming, see Microsoft's documentation.

Related Manuals

PTC-1800 Administrator's Guide
(Part no. 30585-000-001).

PTC-1800 Product Reference Guide
(Part no. 30635-000-001).

PTC-1800 Quick Start Guide
(Part no. 24831-000-002)**.**

For additional information on the PTC-1800, visit Symbol's website at **www.symbol.com** .

Getting Help

If you experience a problem with your PTC-1800, contact your Symbol service representative or the Symbol Product Center at 1-800-653-5350.

Getting Ready

Visual Studio

Microsoft Visual Studio is the standard integrated development environment for Windows CE. This section will indicate what is needed to use Visual Studio with the PTC-1800.

C/C++ Programming

Setting up for Programming C/C++

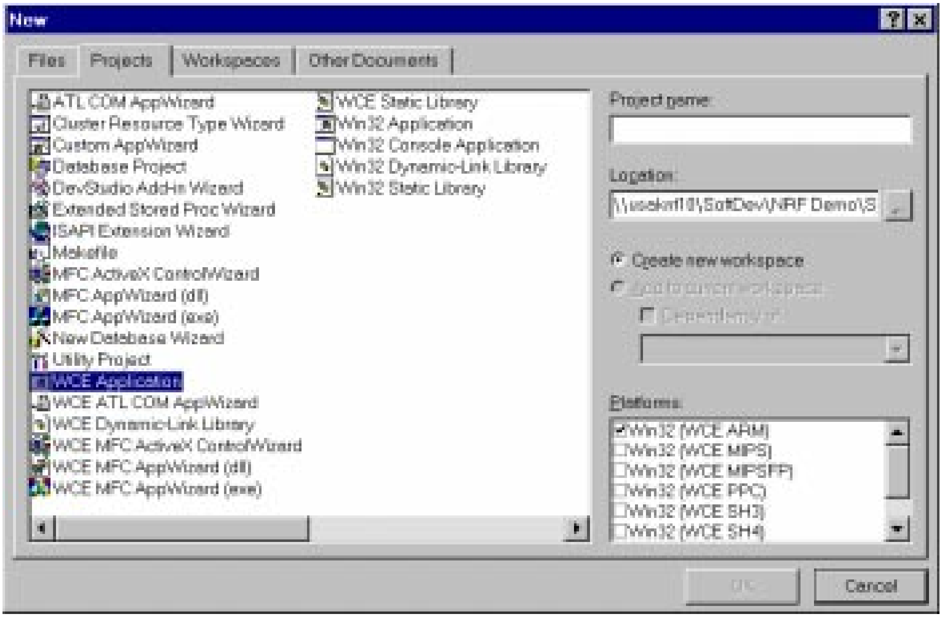
Setup/Install in the following order/sequence:

- Use the latest version of Visual Studio (6.0 or later). Make sure that the Visual C/C++ modules are installed.
- Install the Windows CE C++ Toolkit. This is a purchase item from Microsoft.
- Install Windows CE H/PC Pro SDK for C++. The H/PC Pro SDK may be downloaded from Microsoft's web site (**www.microsoft.com**).
- Install the PTC-1800 SDK for C/C++.
- Read Appendix A, Known Microsoft Problems (as of March 2000).

Visual C++

With the proper toolkits and platforms installed, you can build and debug projects for the PTC-1800. Visual Studio will take care of downloading the necessary files to your device, including any necessary DLLs.

When creating a new project, the illustration below shows how to select a Windows CE application and the ARM processor.



Location: \\usaknf1\SoftDev\NRF Demo\S

● Create new workspace

☑ Win32 (WCE ARM)
□ Win32 (WCE MIPS)
□ Win32 (WCE MIPSFP)
□ Win32 (WCE PPC)
□ Win32 (WCE SH3)
□ Win32 (WCE SH4)

Microsoft Foundation Class (MFC) Programming

With the proper toolkits and platforms installed, you can build and debug MFC projects for the PTC-1800.

The PTC-1800 does not include the MFC DLLs in its Windows CE 2.12 image. A programmer who wishes to use MFC must include the necessary DLLs (available from Microsoft) in the program image.

Visual Basic Programming

Setting up for Programming Visual Basic

- Use the latest version of Visual Studio (6.0 or later). Be sure that the Visual Basic modules are installed.
- Install the Windows CE Visual Basic Toolkit. This is a purchase item from Microsoft.
- Install Windows CE H/PC Pro SDK for Visual Basic. The H/PC Pro SDK can be downloaded from the Microsoft web site (**www.microsoft.com**).
- Install the PTC-1800 SDK for Visual Basic. Symbol will provide this program. Visual Basic

With the proper toolkits and platforms installed, you may build and debug Visual Basic projects for the PTC-1800. Visual Basic will even build the installation package for you.

The PTC-1800 does not include the Visual Basic runtime or supported controls in its Windows CE 2.12 image. A programmer who wishes to use Visual Basic must include the necessary DLLs (available from Microsoft) in the program image.

The PTC-1800 Windows CE image is setup to recognize the Visual Basic DLLs in the "**\harddisk**" directory instead of the standard "**\windows**" directory.

Active Sync

Microsoft ActiveSync is the main communication vehicle between a desktop PC and a Windows CE device, both for the development and exchange of data. Use Version 3.0 or later for all development. The use of prior versions is not recommended.

ActiveSync has replaced the previous Microsoft product named “CE Services.” ActiveSync may be downloaded from Microsoft’s web site.

Examples of ActiveSync services are

- An “Explorer-like” window on your desktop PC that allows you to view folders on the Windows CE device and transfer files.
- A communications link to your Windows CE device for Visual Studio and associated tools, including the Visual Studio debugger, Remote Registry Editor, and others.

Active Sync Connections

For these services to work, a “connection” is required between your desktop PC and the CE device. An Active Sync connection is either a partnership or a guest account. A partnership allows the “connection” to operate over a LAN.

A connection is a “desktop-PC-to-CE-device” entity that is indifferent to the communications media. Connections may be created via:

- Serial port to serial port via cable.
- Serial port to serial port via IrDA.
- Microsoft Networking via RF and wireless LAN (requires partnership).

The number of connections that a Windows CE device is able to support is limited. Only one partnership on a desktop PC and one Windows CE device can be active at one time.

Using ActiveSync with the PTC-1800

The PTC-1800 lacks a serial port so the IrDA port must be used for the initial partnership with the device. Once the IrDA partnership is established, the communications link can be switched from IrDA to LAN (RF + wired).

The following IrDA dongle has been used with success attached to desktop PCs and communicating with PTC-1800 (Extended Systems Jeteye PN ESI9680B).

This dongle comes with QuickBeam Suite software. Install this software on your desktop PC. Symbol recommends the use of the default port COM4.

Initial Partnership

Once the IrDA is installed, follow these steps to create an initial partnership with the PTC-1800:

- Select **File|Get Connected** on ActiveSync. ActiveSync will start checking COM ports for the presence of a Windows CE device. You need to execute the next step quickly before ActiveSync gives up on COM4.
- On the PTC-1800, Start "**repllog.exe**" (System Menu|Programs|PC Link). Place the PTC-1800's IrDA port in line with the IrDA device.
- If you are successful, ActiveSync will indicate that it has found a new device. Follow the prompts to specify that you want to create a partnership. Specify that you DO want a partnership. You can deselect all of the sync options for the partnership.
- If you get a prompt asking whether you want partnerships on multiple computers, select **Yes**. PTC-1800 will support a maximum of two partnerships.
- The partnership should be established. If you have not been successful, try these steps again.

Once an initial partnership is established, you may continue to use IrDA as your communications link or you may switch to RF and wireless LAN, which is a much faster link.

Guest Connection

Once the IrDA is installed, follow these steps to create an initial guest connection with the PTC-1800:

- Select **File|Get Connected** on ActiveSync. ActiveSync will start checking COM ports for the presence of a Windows CE device. You need to execute the next step quickly before ActiveSync gives up on COM4.

- On the PTC-1800, Start “**repllog.exe**” (System Menu|Programs|PC Link). Place the PTC-1800’s IrDA port in line with the IrDA device.

- If you are successful, ActiveSync will indicate that it has found a new device. Follow the prompts to specify that you want to create a partnership. Specify that you DO want a guest account.

- The guest account should be established. If you have not been successful, try these steps again.

Partnership Via Wireless LAN

Once you have an initial partnership set up between a desktop PC and a CE device, you can switch to wireless LAN as your communication link for faster communications. Here are the steps to achieve this link:

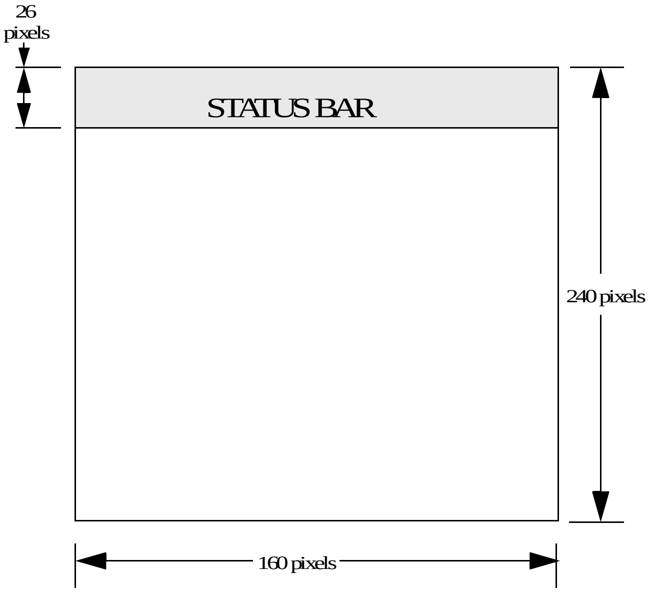
- Make sure that the radio in the PTC-1800 is registered to an access point on the LAN to which your desktop PC is connected.
- Make sure that WINS in the PTC-1800 is set up for your WINS name server. The PTC-1800 will look, by name, for the same PC with which it had established a partnership via IrDA. It uses WINS to do this.
- Break the existing IrDA partnership with your PC, if any.
- Start "**repllog.exe /remote**" on the PTC-1800 (System Menu|Programs|Remote Link). Select **Network** in the list box.
- The partnership will be established.

PTC-1800 Programming Overview

This section will define how to program different devices on the PTC-1800.

Display

The display on the PTC-1800 is 160 x 240 (1/ 8 VGA). The Status Bar will take up 26 pixels on the top of the screen leaving a screen size of 160 x 214 pixels.



Use the Windows CE call **SystemParametersInfo** to get the screen size and use **SetWindowPos** to set your window size to the full work area on window creation.

Contrast

The contrast control of the display can be programmed. For details, see section on “PTC-1800 Programming Details.”

Keyboard

The PTC-1800 is basically a Windows device. Although it does not have a mouse, all selects and entries are done with the keyboard. See the ***PTC-1800 Product Reference Guide*** document which defines the necessary key sequences needed to accomplish this.

In general, you use the **Softkey 1** key (tab key) to move to the control that you wish to activate. In most cases, the control is activated but not always. In those cases, the best you can do is to test to see whether the control you want has focus (the **focus rectangle** may be visible).

Navigating on property sheets is a bit more difficult. The property sheet tabs themselves are one control that is included in the tab order. Tab order is important. The focus rectangle is not visible when selected. After selecting the tabs, you can move between them with the cursor using the right/left keys (arrow keys).

Hot Keys

The **Phone**, **Msg**, and **Talk** keys are defined as application hot keys and are reserved for defined applications. The **Data** key may be used by a programmer.

A programmer can set the WM_HOTKEY message to be sent to a given window by using the **RegisterHotKey** function call. The **fsModifier** flag should be set to MOD_WIN and the **vk** parameter set to 0xC2 for the **Data** key.

Typically, this is used to bring the application into the foreground.

Serial Ports

The PTC-1800 Windows CE image supports three communication ports: COM1:, COM4:, and COM5:. When opening a driver in Windows CE be sure that the "**:**" is included in the driver name.

COM1:

COM1: communicates with the scanner decode chip.

COM4:

COM4: is the ***raw infrared*** (raw IR) port. If IrDA is not needed, then a programmer may use this port. See Microsoft's Windows CE documentation about programming a raw IR port.

COM5:

COM5: is the ***IrCOMM*** port. See Microsoft's Windows CE documentation about programming an ***IrCOMM*** port.

IrDA

Windows CE-based devices have three options for implementing infrared communications: ***raw infrared*** (raw IR), ***IrCOMM***, and ***Infrared Sockets*** (IrSock).

See the section above for ***raw IR*** and ***IrCOMM*** information. For more information about ***IrSock***, see Microsoft's CE documentation about programming Windows Sockets.

Scanner

A programmer may either use COM1: to program the scanner directly or use the scanner keyboard wedge program to implement scanner input as keystroke data.

Scanner Keyboard Wedge

The Scanner Keyboard Wedge program will always send an **F9** key down and up events before the beginning of scanner data. It will always send an **F10** key down and up events at the end of scanner data. This allows a programmer to take advantage of the scanner data for any processing needed.

Backlight

The backlight control of the display may be programmed to be turned on and off. For details, see section on “PTC-1800 Programming Details.”

Vibrator (Motor)

The vibrator (motor) can be programmed to turn on and off. For details, see section on “PTC-1800 Programming Details.”

Buzzer

The buzzer may be programmed to be on or off. Turning the buzzer on allows audio streams to be played on the system buzzer along with the current system receiver (either internal or as headset). For details, see section on “PTC-1800 Programming Details.”

Persistent Registry

Windows CE allows several methods to be used for saving the registry to persistent (nonvolatile) storage. This allows the registry to be restored if the unit needs to be cold booted.

The PTC-1800 supports the **RegCopyFile** function call to save the registry contents to the file **\harddisk\saved.reg**. It will restore this file to the registry on a cold boot.

RegCopyFile

This function saves a copy of the current Windows CE RAM-based registry to a specified file, defined in **PTC1800.h** header file included in PTC1800 C/C++ Software Development Kit.

Syntax

BOOL RegCopyFile (LPCWSTR *lpCopyFileName*).

Parameters

LpCopyFileName specifies the name of the file to which the registry is saved. For the PTC-1800, it should be **\harddisk\saved.reg.**

Return Values

TRUE indicates success. FALSE indicates failure. To obtain extended error information, call **GetLastError**.

Remarks

The saved registry (.reg) file can exist on any mounted file system that is accessible by the operating system. **RegCopyFile** always attempts to create the specified file and open it exclusively for read/write access.

PTC-1800 Programming Details

The PTC-1800 uses the **KernelIoControl** function calls to allow programmability of certain nonstandard hardware devices.

KernelIoControl

The KernelIoControl function provides the kernel with a generic Io control code (IOCTL)

Syntax

BOOL KernelIoControl (**DWORD** ***dwIoControlCode***, **LPVOID** ***lpInBuf***, **DWORD** ***nInBufSize***, **LPVOID** ***lpOutBuf***, **DWORD** ***nOutBufSize***, **LPDWORD** ***lpBytesReturned***).

Parameters

dwIoControlCode—IO control code.

lpInBuf —Pointer to the input buffer.

NInBufsize —Size in bytes of ***lpInBuf***.

lpOutBuf —Pointer to the output buffer.

NOutBufSize —Maximum number of bytes that can be returned in ***lpOutBuf***.

LpBytesReturned —Actual number of bytes returned in ***lpOutBuf***.

Return Values

TRUE indicates success. FALSE indicates failure.

Remarks

The kernel calls the function **OEMIoControl** when a device driver or routine calls the kernel function **KernelIoControl** and passes an I/O control code (IOCTL). The system is fully preemptible when this function is called. The kernel does no processing, but passes all parameters directly to the function supplied by Symbol. This function is provided solely to allow your application to communicate with the kernel mode code of the OEM adaptation layer.

For **KernelIoControl** calls specific to the PTC-1800, the following parameter definition will always be true.

> ***dwIoControlCode*** — IO control code.
>
> ***lpInBuf*** — Pointer to an ULONG.
>
> ***NInBufsize*** — Size of an ULONG.
>
> ***lpOutBuf*** — Pointer to an ULONG.
>
> ***NOutBufSize*** — Size of a ULONG.
>
> ***LpBytesReturned*** — Size of a ULONG when a value is returned in ***lpOutBuf***.

Sample

All Symbol-supplied **KernelIoControl** function calls follow the same basic format as the code snippet that is supplied below.

```
    ULONG ulIoControlCode;
    ULONG Param;
    ULONG Result;
    ULONG ResultSize;

    Param = IOCTL_CONTRAST_UP;
    KernelIoControl

(IOCTL_HAL_CONTRAST,

        (PVOID)&Param,
        sizeof(ULONG),
        &Result,
        sizeof(ULONG),
        &ResultSize
        );
```

Scanner

The control lines to the scanner decode module may be manipulated through **KernelIoControl** function calls.

dwIoControlCode: IOCTL_HAL_SCANNER

lpInBuf	lpOutBuf	Description
IOCTL_SCANNER_INIT	N/A	Initializes the scanner decode module settings.
IOCTL_SCANNER_ON	N/A	Turns scanner decode module power on.
IOCTL_SCANNER_OFF	N/A	Turns scanner decode module power off.
IOCTL_SCANNER_STATE	1 if the scanner decode module is powered on, and 0 if the scanner decode module is powered off.	Returns the state of the scanner decode module power.
IOCTL_SCANNER_WAKE	N/A	Wakes up the scanner decode module.
IOCTL_SCANNER_SLEEP	N/A	Allows the scanner decode module to sleep.

lpInBuf	lpOutBuf	Description
IOCTL_SCANNER_RESET_ON	N/A	Places the scanner decode module into the reset state.
IOCTL_SCANNER_RESET_OFF	N/A	Places the scanner decode module into the run state.
IOCTL_SCANNER_RESET_STATE	1 if the scanner decode module is in reset, and 0 if the scanner decode module is in run state.	Returns the state of the scanner decode module reset signal.

Backlight

The backlight may be manipulated through **Kernel IoControl** function calls.

dwIoControlCode:
IOCTL_HAL_BACKLIGHT

lpInBuf	lpOutBuf	Description
IOCTL_BACKLIGHT_INIT	N/A	Initializes the backlight settings.
IOCTL_BACKLIGHT_ON	N/A	Turns the backlight on.
IOCTL_BACKLIGHT_OFF	N/A	Turns the backlight off.
IOCTL_BACKLIGHT_STATE	1 if backlight is on, and 0 if backlight is off.	Returns the state of the backlight.

Vibrator (Motor)

The vibrator (motor) may be manipulated through **KernelIoControl** function calls.

dwIoControlCode: IOCTL_HAL_MOTOR

lpInBuf	lpOutBuf	Description
IOCTL_MOTOR_INIT	N/A	Initializes the vibrator settings.
IOCTL_MOTOR_ON	N/A	Turns the vibrator on.
IOCTL_MOTOR_OFF	N/A	Turns the vibrator off.
IOCTL_MOTOR_STATE	1 if the vibrator is on, and 0 if the vibrator is off.	Returns the state of the vibrator.

Buzzer

The buzzer may be manipulated through **KernelIoControl** function calls.

dwIoControlCode: IOCTL_HAL_BUZZER

lpInBuf	lpOutBuf	Description
IOCTL_BUZZER_INIT	N/A	Initializes the buzzer settings.
IOCTL_BUZZER_ON	N/A	Turns the buzzer on.
IOCTL_BUZZER_OFF	N/A	Turns the buzzer off.
IOCTL_BUZZER_STATE	1 if the buzzer is on, and 0 if the buzzer is off.	Returns the state of the buzzer.

Contrast

The display contrast may be manipulated through **KernelIoControl** function calls. The contrast can have a value of 0–32.

dwIoControlCode: IOCTL_HAL_CONTRAST

lpInBuf	lpOutBuf	Description
IOCTL_CONTRAST_INIT	N/A	Initializes the contrast settings.
IOCTL_CONTRAST_UP	N/A	Moves the contrast up one setting.
IOCTL_CONTRAST_DOWN	N/A	Moves the contrast down one setting.

dwIoControlCode
IOCTL_HAL_CONTRAST_GET

lpInBuf	lpOutBuf	Description
N/A	A value from 0–32 to which the contrast is currently set.	Gets gross contrast values.

dwIoControlCode: IOCTL_HAL_CONTRAST_SET

lpInBuf	lpOutBuf	Description
Contrast value from 0–32 to which the contrast will be set.	N/A	Sets gross contrast values.

Appendix A

Microsoft Errata For Windows CE 2.12

The following is from Microsoft's web site (http://support.microsoft.com/support/kb/articles/Q247/3/31.asp) dealing with using a Windows CE 2.12 Software Development Kit with Visual Studio (6.0 or later).

Error Spawning Compiler for Windows CE 2.12 Platforms

The information in this article applies to

- Microsoft Windows CE Toolkit for Visual C++ 6.0.
- Microsoft Windows CE Platform Builder, version 2.12.

Symptoms

With a successful install of the Microsoft Windows CE Toolkit for Microsoft Visual C++ 6.0 and a software development kit (SDK) exported from Microsoft Windows CE Platform Builder 2.12, attempts to build projects targeting the 2.12 platform fail with an error similar to the following:

> Error spawning **clmips.exe**.

Cause

The Microsoft Windows CE Toolkit for Visual C++ 6.0 was shipped prior to the release of Windows CE, Version 2.12 and does not include the compilers needed to build applications for this operating system.

Resolution

By using the compilers provided for Windows CE 2.11, you will be able to build and debug applications targeting Windows CE 2.12:

1. Start Microsoft Visual C++ 6.0.
2. From the **Tools** select **Options**, and then select the **Directories** tab.
3. Select the 2.12 platform in the Windows CE configuration drop-down list.
4. Select **Executable Files** in the Show Directories for drop-down list.
5. Change the WCE212\BIN entry to WCE211\BIN.

6. Select **Library Files** in the Show Directories for drop-down list.

7. Change occurrences of “Mfc” to “MFC” (capitalize all the letters).

8. Change occurrences of **Atl** to **ATL** (capitalize all the letters).

Note: *The preceding changes will not allow applications to be built for the MIPS16 or Thumb processors, which are supported by Windows CE 2.12. Check the Microsoft Windows CE products web site at* ***www.microsoft.com/windowsce*** *for information on the availability of updated development tools.*

Status

This behavior is by design.

More Information

Steps to Reproduce Behavior

1. Install an SDK exported from Platform Builder 2.12 into the Windows CE Toolkit for Visual C++ 6.0.
2. Create or load a Windows CE application and attempt to build for the platform installed in Step 1 using a CPU configuration other than x86. An error occurs while spawning the Windows CE compiler for the requested processor.

Additional Query Words

Toolkit Windows CE Platform Builder SDK kbWindows CE 2.12.

Keywords

kbVC600 kbDSupport kbGrpVCCompiler.

Version

N\A:2.12; WINDOWS: 6.0.

Platform

N\A WINDOWS.

Issue type

kbprb.

Index

Symbol Technologies, Inc.

One Symbol Plaza

Holtsville, NY 11742-1300

Document No: 30629-000-002

Printed in U.S.A.

3 0 6 2 9 - 0 0 0 - 0 0 2