7 Programming

The following programming information pertains to the 700 Series Color Mobile Computer:

- Creating CAB Files *(page 224)*
- Customization and Lockdown *(page 241)*
- FTP Server *(page 242)*
- Kernel I/O Control Functions *(page 250)*
- Network Selection APIs *(page 266)*
- Notifications *(page 289)*
- Reboot Functions *(page 291)*
- Remapping the Keypad *(page 292)*

Note: “700 Color” pertains to 740, 741, 750, 751, 760, and 761 Computers unless otherwise noted.

Creating CAB Files

The Windows CE operating system uses a .CAB file to install an application on a Windows CE-based device. A .CAB file is composed of multiple files that are compressed into one file. Compressing multiple files into one file provides the following benefits:

- All application files are present.
- A partial installation is prevented.
- The application can be installed from several sources, such as a desktop computer or a Web site.

Use the CAB Wizard application (CABWIZ.EXE) to generate a .CAB file for your application.

Creating Device-Specific CAB Files

Do the following to create a device-specific .CAB file for an application, *in the order provided*:

1. Create an .INF file with Windows CE-specific modifications *(page 224)*.
2. *Optional* Create a SETUP.DLL file to provide custom control of the installation process *(page 236)*.
3. Use the CAB Wizard to create the .CAB file, using the .INF file, the optional SETUP.DLL file, and the device-specific application files as parameters *(page 239)*.

Creating an .INF File

An .INF file specifies information about an application for the CAB Wizard. Below are the sections of an .INF file:

[Version]

This specifies the creator of the file, version, and other relevant information.

Required? Yes

- **Signature**: “*signature_name*”
  “$Windows NT$”
- **Provider**: “*INF_creator*”
  The company name of the application, such as “Microsoft.”
- **CESignature**
  “$Windows CE$”

Example

```
[Version]
Signature = “$Windows NT$”
Provider = “Intermec”
CESignature = “$Windows CE$”
```

[CEStrings]

This specifies string substitutions for the application name and the default installation directory.

Required? Yes

- **AppName**: *app_name*
  Name of the application. Other instances of %AppName% in the .INF file are replaced with this string value, such as RP32.
- **InstallDir**: *default_install_dir*
  Default installation directory on the device. Other instances of %InstallDir% in the .INF file are replaced with this string value. Example: \SDMMC_Disk\%AppName%

Example

```
[CEStrings]
AppName=“Game Pack”
InstallDir=%CE1%\%AppName%
```

[Strings]

This section is optional and defines one or more string keys. A string key represents a string of printable characters.

Required? No

- **string_key**: *value*
  String consisting of letters, digits, or other printable characters. Enclose *value* in double quotation marks ““”” if the corresponding string key is used in an item that requires double quotation marks. No string_keys is okay.

Example

```
[Strings]
reg_path = Software\Intermec\My Test App
```

[CEDevice]

Describes the platform for the targeted application. All keys in this section are optional. If a key is nonexistent or has no data, Windows CE does not perform any checking with the exception being *UnsupportedPlatforms*. If the *UnsupportedPlatforms* key exists but no data, the previous value is not overridden.

Required? Yes

- **ProcessorType** : *processor_type*
  The value that is returned by **SYSTEMINFO**.dwProcessorType.For example, the value for the ARM CPU is 2577
- **UnsupportedPlatforms**: *platform_family_name*
  This lists known unsupported platform family names. If the name specified in the [**CEDevice.xxx**] section is different from that in the [**CEDevice**] section, both *platform_family_name* values are unsupported for the microprocessor specified by xxx. That is, the list of unsupported platform family names is appended to the previous list of unsupported names. Application Manager will not display the application for an unsupported platform. Also, a user will be warned during the setup process if the .CAB file is copied to an unsupported device.

Example

```
[CEDevice]
UnsupportedPlatforms = pltfrm1 ; pltfrm1 is unsupported
[CEDevice.SH3]
UnsupportedPlatforms = ; pltfrm1 is still unsupported
```

- **VersionMin**: *minor_version*
  Numeric value returned by **OSVERSIONINFO**.dwVersionMinor. The .CAB file is valid for the currently connected device if the version of this device is greater than or equal to **VersionMin**.
- **VersionMax**: *major_version*
  Numeric value returned by **OSVERSIONINFO**.dwVersionMajor. The .CAB file is valid for the currently connected device if the version of this device is less than or equal to **VersionMax**.
- **BuildMin**: *build_number*
  Numeric value returned by **OSVERSIONINFO**.dwBuildNumber. The .CAB file is valid for the currently connected device if the version of this device is greater than or equal to **BuildMin**.
- **BuildMax**: *build_number*
  Numeric value returned by **OSVERSIONINFO**.dwBuildNumber. The .CAB file is valid for the currently connected device if the version of this device is less than or equal to **BuildMax**.

Example

The following code example shows three [**CEDevice**] sections: one that gives basic information for any CPU and two that are specific to the SH3 and the MIPS microprocessors.

```
[CEDevice]                       ; A “template” for all platforms
UnsupportedPlatforms = pltfrm1   ; Does not support pltfrm1

; The following specifies version 1.0 devices only.
VersionMin = 1.0
VersionMax = 1.0

[CEDevice.ARM]                ; Inherits all [CEDevice] settings
; This will create a .CAB file specific to ARM devices.
ProcessorType = 2577          ; ARM .cab file is valid for ARM microprocessors.
UnsupportedPlatforms =        ; pltfrm1 is still unsupported

; The following overrides the version settings so that no version checking is
performed.
VersionMin =
VersionMax =

[CEDevice.MIPS]                  ; Inherits all [CEDevice] settings
; This will create a .CAB file specific to “MIPS” devices.
ProcessorType = 4000          ; MIPS .CAB file is valid for MIPS
microprocessor.
UnsupportedPlatforms =pltfrm2 ; pltfrm1, pltfrm2 unsupported for MIPs .CAB
file.
```

Note: To create the two CPU-specific .CAB files for the SETUP.INF file in the previous example, run the CAB Wizard with the “/cpu arm mips” parameter.

[DefaultInstall]

This describes the default installation of your application. Note that under this section, you will list items expanded upon later in this description.

Required? Yes

- **Copyfiles**: *copyfile_list_section*
  Maps to files defined later in the .INF file, such as Files.App, Files.Font, and Files.Bitmaps.
- **AddReg**: *add_registry_section*
  Example: RegSettings.All
- **CEShortcuts**: *shortcut_list_section*
  String that identifies one more section that defines shortcuts to a file, as defined in the [**CEShortcuts**] section.
- **CESetupDLL**: *setup_DLL*
  Optimal string that specifies a SETUP.DLL file. It is written by the Independent Software Vendor (ISV) and contains customized functions for operations during installation and removal of the application. The file must be specified in the [**SourceDisksFiles**] section.
- **CESelfRegister**: *self_reg_DLL_filename*
  String that identifies files that self-register by exporting the **DllRegisterServer** and **DllUnregisterServer** Component Object Model (COM) functions. Specify these files in the [**SourceDiskFiles**] section. During installation, if installation on the device fails to call the file's exported **DllRegisterServer** function, the file's exported **DllUnregisterServer** function will not be called during removal.

Example

```
[DefaultInstall]
AddReg = RegSettings.All
CEShortcuts = Shortcuts.All
```

[SourceDiskNames]

This section describes the name and path of the disk on which your application resides.

Required? Yes

- **disk_ordinal**: *disk_label,,path*
  1=,"App files" , C:\Appsoft\RP32\...
  2=,"Font files",,C:\RpTools\...
  3=,"CE Tools" ,,C:\windows ce tools...
- **CESignature**: "$Windows CE$"

Example

```
[SourceDisksNames]                          ; Required section
1 = ,"Common files",,C:\app\common          ; Using an absolute path
[SourceDisksNames.SH3]
2 = ,"SH3 files",,sh3                       ; Using a relative path
[SourceDisksNames.MIPS]
2 = ,"MIPS files",,mips                     ; Using a relative path
```

[SourceDiskFiles]

This describes the name and path of the files in which your application resides.

Required? Yes

- **filename**: *disk_number[,subdir]*
  RPM.EXE = 1,c:\appsoft\...
  WCESTART.INI = 1
  RPMCE212.INI = 1
  TAHOMA.TTF = 2

Note: [,**subdir**] is relative to the location of the INF file.

Example

```
[SourceDisksFiles]       ; Required section
begin.wav = 1
end.wav = 1
sample.hlp = 1
[SourceDisksFiles.SH3]
sample.exe = 2        ; Uses the SourceDisksNames.SH3 identification of 2.
[SourceDisksFiles.MIPS]
sample.exe = 2        ; Uses the SourceDisksNames.MIPS identification of 2.
```

[DestinationDirs]

This describes the names and paths of the destination directories for the application on the target device. *Note Windows CE does not support directory identifiers.*

Required? Yes

- **file_list_section**: *0,subdir*
  String that identifies the destination directory. The following list shows the string substitutions supported by Windows CE. Use these only for the beginning of the path. \

%CE1%	\Program Files
%CE2%	\Windows
%CE3%	\My Documents
%CE4%	\Windows\Startup
%CE5%	\My Documents
%CE6%	\Program Files\Accessories
%CE7%	\Program Files\Communication
%CE8%	\Program Files\Games
%CE9%	\Program Files\Pocket Outlook
%CE10%	\Program Files\Office
%CE11%	\Windows\Start Menu\Programs
%CE12%	\Windows\Start Menu\Programs\Accessories
%CE13%	\Windows\Start Menu\Programs\Communications
%CE14%	\Windows\Start Menu\Programs\Games
%CE15%	\Windows\Fonts
%CE16%	\Windows\Recent
%CE17%	\Windows\Start Menu

  %InstallDir%
  Contains the path to the target directory selected during installation. It is declared in the [**CEStrings**] section
  %AppName%
  Contains the application name defined in the [**CEStrings**] section.

Example

```
[DestinationDirs]
Files.Common  = 0,%CE1%\My Subdir      ; \Program Files\My Subdir
Files.Shared  = 0,%CE2%                ; \Windows
```

[CopyFiles]

This section, under the [**DefaultInstall**] section, describes the default files to copy to the target device. Within the [**DefaultInstall**] section, files were listed that must be defined elsewhere in the INF file. This section identifies that mapping and may contain flags.

Required? Yes

- **copyfile_list_section**: *destination_filename,[source_filename]*
  The *source_filename* parameter is optional if it is the same as *destination_filename*.
- **copyfile_list_section**: *flags*
  The numeric value that specifies an action to be done while copying files. The following table shows values supported by Windows CE.

Flag	Value	Description
COPYFLG_WARN_IF_SKIP	0x00000001	Warn user if skipping a file is attempted after error.
COPYFLG_NOSKIP	0x00000002	Do not allow a user to skip copying a file.
COPYFLG_NO_OVERWRITE	0x00000010	Do not overwrite files in destination directory.
COPYFLG_REPLACEONLY	0x00000400	Copy the source file to the destination directory only if the file is already in the destination directory.
CE_COPYFLG_NO_DATE_DIALOG	0x20000000	Do not copy files if the target file is newer.
CE_COPYFLG_NODATECHECK	0x40000000	Ignore date while overwriting the target file.
CE_COPYFLG_SHARED	0x80000000	Create a reference when a shared DLL is counted.

Example

```
[DefaultInstall.SH3]
CopyFiles = Files.Common, Files.SH3
[DefaultInstall.MIPS]
CopyFiles = Files.Common, Files.MIPS
```

[AddReg]

This section, under the [**DefaultInstall**] section, is optional and describes the keys and values that the .CAB file adds to the device registry. Within the [**DefaultInstall**] section, a reference may have been made to this section, such as “AddReg=RegSettings.All”. This section defines the options for that setting.

Required? No

- **add_registry_section**: *registry_root_string*
  String that specifies the registry root location. The following list shows the values supported by Windows CE.
  - HKCR Same as HKEY_CLASSES_ROOT
  - HKCU Same as HKEY_CURRENT_USER
  - HKLM Same as HKEY_LOCAL_MACHINE
- **add_registry_section**: *value_name*
  Registry value name. If empty, the “default” registry value name is used.
- **add_registry_section**: *flags*
  Numeric value that specifies information about the registry key. The following table shows the values that are supported by Window CE.

Flag	Value	Description
FLG_ADDREG_NOCLOBBER	0x00000002	If the registry key exists, do not overwrite it. Can be used with any of the other flags in this table.
FLG_ADDREG_TYPE_SZ	0x00000000	REG_SZ registry data type.
FLG_ADDREG_TYPE_MULTI_SZ	0x00010000	REG_MULTI_SZ registry data type. Value field that follows can be a list of strings separated by commas.
FLG_ADDREG_TYPE_BINARY	0x00000001	REG_BINARY registry data type. Value field that follows must be a list of numeric values separated by commas, one byte per field, and must not use the 0x hexadecimal prefix.
FLG_ADDREG_TYPE_DWORD	0x00010001	REG_DWORD data type. The noncompatible format in the Win32 Setup .INF documentation is supported.

Example

```
AddReg = RegSettings.All

[RegSettings.All]
HKLM,%reg_path%,,0x00000000,alpha            ; <default> = “alpha”
HKLM,%reg_path%,test,0x00010001,3            ; Test = 3
HKLM,%reg_path%\new,another,0x00010001,6     ; New\another = 6
```

[CEShortCuts]

This section, a Windows CE-specific section under the [**DefaultInstall**] section, is optional and describes the shortcuts that the installation application creates on the device. Within the [**DefaultInstall**] section, a reference may have been made to this section, such as “ShortCuts.All”. This section defines the options for that setting.

Required? No

- **shortcut_list_section**: *shortcut_filename*
  String that identifies the shortcut name. It does not require the .LNK extension.
- **shortcut_list_section**: *shortcut_type_flag*
  Numeric value. Zero or empty represents a shortcut to a file; any non-zero numeric value represents a shortcut to a folder.
- **shortcut_list_section**: *target_file_path*
  String value that specifies the destination location. Use the target file name for a file, such as MyApp.exe, that must be defined in a file copy list. For a path, use a *file_list_section* name defined in the [**DestinationDirs**] section, such as *DefaultDestDir*, or the *%InstallDir%* string.
- **shortcut_list_section**: *standard_destination_path*
  Optional string value. A standard *%CEx%* path or *%InstallDir%*. If no value is specified, the *shortcut_list_section* name of the current section or the *DefaultDestDir* value from the [**DestinationDirs**] section is used.

Example

```
CEShortcuts = Shortcuts.All
[Shortcuts.All]
Sample App,0,sample.exe         ; Uses the path in DestinationDirs. Sample
App,0,sample.exe,%InstallDir%    ; The path is explicitly specified.
```

Sample .INF File

```
[Version]           ; Required section
Signature = “$Windows NT$”
Provider = “Intermec Technologies Corporation”
CESignature = “$Windows CE$”

;[CEDevice]
;ProcessorType =

[DefaultInstall]    ; Required section
CopyFiles = Files.App, Files.Fonts, Files.BitMaps, Files.Intl,
Files.TelecomNcsCE, Files.Windows, Files.Import, Files.Export, Files.Work,
Files.Database, Files.WinCE AddReg = RegSettings.All ;CEShortcuts =
Shortcuts.All

[SourceDisksNames]  ; Required section
1 = ,“App files” ,,c:\appsoft\...
2 = ,”Font files” ,,c:\WinNT\Fonts
3 = ,”CE Tools”  ,,c:\windows ce tools\wce400\700ie\mfc\lib\x86

[SourceDisksFiles]  ; Required section
rpm.exe   = 1,C:\Appsoft\program\wce400\WCEX86Rel700
wcestart.ini = 1
```

```
rpmce212.ini = 1
intermec.bmp = 1
rpmlogo.bmp = 1
rpmname.bmp = 1
import.bmp  = 1
export.bmp  = 1
clock.bmp  = 1
printer.bmp = 1
filecopy.bmp = 1
readme.txt  = 1
lang_eng.bin = 1
rpmdata.dbd = 1,database\wce1
tahoma.ttf  = 2
mfcce212.dll = 3
olece212.dll = 3
olece211.dll = 1,c:\windows ce tools\wce400\NMSD61102.11\mfc\lib\x86
rdm45wce.dll = 1,c:\rptools\rdm45wce\4_50\lib\wce400\wcex86rel
picfmt.dll  = 1,c:\rptools\picfmt\1_00\wce400\wcex86rel6110
fmtctrl.dll = 1,c:\rptools\fmtctrl\1_00\wce400\wcex86rel6110
ugrid.dll  = 1,c:\rptools\ugrid\1_00\wce400\wcex86rel6110
simple.dll  = 1,c:\rptools\pspbm0c\1_00\wce400\wcex86rel
psink.dll = 1,c:\rptools\psink\1_00\wce400\WCEX86RelMinDependency
pslpwce.dll =1,c:\rptools\pslpm0c\1_00\wce400\WCEX86RelMinDependency
npcpport.dll = 1,c:\rptools\cedk\212_03\installable drivers\printer\npcp
;dexcom.dll  = 1,c:\rptools\psdxm0c\1_00\x86
ncsce.exe  = 1,c:\rptools\ncsce\1_04
nrinet.dll  = 1,c:\rptools\ncsce\1_04

[DestinationDirs] ; Required section
;Shortcuts.All = 0,%CE3%  ; \Windows\Desktop
Files.App         = 0,%InstallDir%
Files.DataBase      = 0,%InstallDir%\DataBase
Files.BitMaps     = 0,%InstallDir%\Bitmaps
Files.Fonts       = 0,%InstallDir%\Fonts
Files.Intl        = 0,%InstallDir%\Intl
Files.TelecomNcsCE  = 0,%InstallDir%\Telecom\NcsCE
Files.Windows     = 0,%InstallDir%\Windows
Files.Import      = 0,%InstallDir%\Import
Files.Export      = 0,%InstallDir%\Export
Files.Work        = 0,%InstallDir%\Work
Files.WinCE       = 0,\storage_card\wince

[CEStrings]       ; Required section
AppName = Rp32
InstallDir = \storage_card\%AppName%

[Strings]         ; Optional section
;[Shortcuts.All]
;Sample App,0,sample.exe               ; Uses the path in DestinationDirs.
;Sample App,0,sample.exe,%InstallDir%     ; The path is explicitly specified.

[Files.App]
rpm.exe,,,0
rpm.ini,rpmce212.ini,,0
mfcce212.dll,,,0
olece212.dll,,,0
olece211.dll,,,0
rdm45wce.dll,,,0
picfmt.dll,,,0
```

```
fmtctrl.dll,,,0
ugrid.dll,,,0
simple.dll,,,0
psink.dll,,,0
pslpwce.dll,,,0
npcpport.dll,,,0
;dexcom.dll,,,0

[Files.DataBase]
rpmdata.dbd,,,0

[Files.Fonts]
tahoma.ttf,,,0

[Files.BitMaps]
intermec.bmp,,,0
rpmlogo.bmp,,,0
rpmname.bmp,,,0
import.bmp,,,0
export.bmp,,,0
clock.bmp,,,0
printer.bmp,,,0
filecopy.bmp,,,0

[Files.Intl]
lang_eng.bin,,,0

[Files.TelecomNcsCE]
ncsce.exe,,,0
nrinet.dll,,,0

[Files.Windows]
readme.txt,,,0

[Files.Import]
readme.txt,,,0

[Files.Export]
readme.txt,,,0

[Files.Work]
readme.txt,,,0

[Files.WinCE]
wcestart.ini,,,0

[RegSettings.All]
HKLM,”SOFTWARE\Microsoft\Shell\AutoHide”,,0x00010001,1
; Autohide the taskbar HKLM,”SOFTWARE\Microsoft\Shell\OnTop”,,0x00010001,0
  ; Shell is not on top
HKLM,”SOFTWARE\Microsoft\Clock”,SHOW_CLOCK,0x00010001,0
  ; Clock is not on taskbar
```

Using Installation Functions in SETUP.DLL

SETUP.DLL is an optional file that enables you to perform custom operations during installation and removal of your application. The following list shows the functions that are exported by SETUP.DLL.

Install_Init	Called before installation begins. Use this function to check the application version when reinstalling an application and to determine if a dependent application is present.
Install_Exit	Called after installation is complete. Use this function to handle errors that occur during application installation.
Uninstall_Init	Called before the removal process begins. Use this function to close the application, if the application is running.
Uninstall_Exit	Called after the removal process is complete. Use this function to save database information to a file and delete the database and to tell the user where the user data files are stored and how to reinstall the application.

Note; Use [**DefaultInstall**] > **CESelfRegister** (page 228) in the .INF file to point to SETUP.DLL.

After the CAB File Extraction

Cab files that need to cause a warm reset after cab extraction will need to create the __RESETMEPLEASE__.TXT file in the "\Windows" directory. The preferred method to create this file is within the DllMain portion of the SETUP.DLL file. It looks like this:

```
#include <windows.h>
#include <Tlhelp32.h>
#include <winioctl.h>
#include <ce_setup.h>   // in the public SDK dir

#define IOCTL_TERMINAL_RESET CTL_CODE (FILE_DEVICE_UNKNOWN,FILE_ANY_ACCESS,
2050, METHOD_NEITHER)

BOOL APIENTRY DllMain( HANDLE h, DWORD reason, LPVOID lpReserved )
{
  return TRUE;
} // DllMain


//************************************************************************
// $DOCBEGIN$
// BOOL IsProcessRunning( TCHAR * pname );
//
// Description: Get process table snapshot, look for pname running.
//
// Arguments: pname - pointer to name of program to look for.
// for example, app.exe.
//
// Returns: TRUE - process is running.
//          FALSE - process is not running.
// $DOCEND$
//************************************************************************
BOOL IsProcessRunning( TCHAR * pname )
{
  HANDLE hProcList;
  PROCESSENTRY32 peProcess;
  DWORD thDeviceProcessID;
  TCHAR lpname[MAX_PATH];

  if ( !pname || !*pname ) return FALSE;

  _tcscpy( lpname, pname );
  _tcslwr( lpname );

  hProcList = CreateToolhelp32Snapshot( TH32CS_SNAPPROCESS, 0 );

  if ( hProcList == INVALID_HANDLE_VALUE ) {
    return FALSE;
  } // end if

  memset( &peProcess, 0, sizeof(peProcess) );
  peProcess.dwSize = sizeof(peProcess);

  if ( !Process32First( hProcList, &peProcess ) ) {
    CloseToolhelp32Snapshot( hProcList );
    return FALSE;
  } // end if

  thDeviceProcessID = 0;

  do {
    _tcslwr( peProcess.szExeFile );

    if ( _tcsstr( peProcess.szExeFile, lpname ) ) {
      thDeviceProcessID = peProcess.th32ProcessID;
      break;
    } // end if
  } while ( Process32Next( hProcList, &peProcess ) );

  if ( ( GetLastError() == ERROR_NO_MORE_FILES ) && ( thDeviceProcessID == 0
) ) {
    CloseToolhelp32Snapshot( hProcList );
    return FALSE;
  } // end if

  CloseToolhelp32Snapshot( hProcList );
  return TRUE;
} // IsProcessRunning

codeINSTALL_INIT Install_Init(
  HWND hwndParent,
  BOOL fFirstCall,
  BOOL fPreviouslyInstalled,
  LPCTSTR pszInstallDir )
{
  return codeINSTALL_INIT_CONTINUE;
}

codeINSTALL_EXIT Install_Exit (
  HWND hwndParent,
  LPCTSTR pszInstallDir,
  WORD cFailedDirs,
  WORD cFailedFiles,
  WORD cFailedRegKeys,
  WORD cFailedRegVals,
  WORD cFailedShortcuts )
{
  HANDLE h;
  TCHAR srcfile[MAX_PATH];
  TCHAR dstfile[MAX_PATH];

  if (cFailedDirs || cFailedFiles || cFailedRegKeys ||
    cFailedRegVals || cFailedShortcuts)
    return codeINSTALL_EXIT_UNINSTALL;

  if ( IsProcessRunning( L”autocab.exe” ) )
  {
    h = CreateFile( L”\\Windows\\__resetmeplease__.txt”,
      (GENERIC_READ | GENERIC_WRITE), 0, NULL, CREATE_ALWAYS,
      FILE_ATTRIBUTE_HIDDEN, NULL );

    if ( h != INVALID_HANDLE_VALUE )
      CloseHandle( h );
    else
    {
      // Couldn’t create the file.  If it failed because the file already
exists, it is not fatal.
      // Otherwise, notify user of the inability to reset the device and they
will have to
      // perform it manually after all of the installations are complete.
    } // end if
  }
  else
  {
    DWORD dret;

    h = CreateFile( L”SYI1:”,
      (GENERIC_WRITE | GENERIC_READ), 0, NULL, OPEN_EXISTING,
      FILE_ATTRIBUTE_NORMAL, NULL );

    // Force a warm start NOW.
    if ( h != INVALID_HANDLE_VALUE )
    {
      DeviceIoControl( h, IOCTL_TERMINAL_RESET, NULL, 0, NULL, 0, &dret,
NULL);

      // Won’t return, but we’ll show clean up anyway
      CloseHandle( h );
    }
    else
    {
      // Couldn’t access SYSIO. Notify user.
    } // end if
  } // end if

  return codeINSTALL_EXIT_DONE;
}

codeUNINSTALL_INIT
Uninstall_Init(
  HWND hwndParent,
  LPCTSTR pszInstallDir ) {
```

```
  // TODO: Perform the reverse of INSTALL_INIT here
  return codeUNINSTALL_INIT_CONTINUE;
}

codeUNINSTALL_EXIT
Uninstall_Exit(HWND hwndParent) {

  // TODO: Perform the reverse of INSTALL_EXIT here
  return codeUNINSTALL_EXIT_DONE;
}
```

The system software looks for the following directory structure and files on the installed media card whether it be an SD card or CF card or embedded flash file system. No other folders need exist.

```
\2577\autorun.exe
\2577\autorun.dat
\2577\autocab.exe
\2577\autocab.dat
\cabfiles\*.cab
```

Creating CAB Files with CAB Wizard

After you create the .INF file and the optional SETUP.DLL file, use the CAB Wizard to create the .CAB file. The command-line syntax for the CAB Wizard is as follows:

```
cabwiz.exe “inf_file” [/dest dest_directory] [/err error_file] [/cpu cpu_type
[cpu_type]]
```

A batch file, located in <program> directory, with the following commands, works well:

```
cabwiz.exe c:\appsoft\<program>\<inf_file_name>
cd \appsoft\<program>
```

“inf_file”	The SETUP.INF file path.
dest_directory	The destination directory for the .CAB files. If no directory is specified, the .CAB files are created in the “inf_file” directory.
error_file	The file name for a log file that contains all warnings and errors that are encountered when the .CAB files are compiled. If no file name is specified, errors are displayed in message boxes. If a file name is used, the CAB Wizard runs without the user interface (UI); this is useful for automated builds.
cpu_type	Creates a .CAB file for each specified microprocessor tag, which is a label used in the Win32 SETUP.INF file to differentiate between different microprocessor types. The */cpu* parameter, followed by multiple *cpu_type* values, must be the last qualifier in the command line.

Example

This example creates .CAB files for the ARM and MIPS microprocessors, assuming the Win32 SETUP.INF file contains the ARM and MIPS tags:

```
cabwiz.exe “c:\myfile.inf” /err myfile.err /cpu arm mips
```

Note: CABWIZ.EXE, MAKECAB.EXE, and CABWIZ.DDF (Windows CE files available on the Windows CE Toolkit) must be installed in the same directory on the desktop computer. Call CABWIZ.EXE using its full path for the CAB Wizard application to run correctly.

Troubleshooting the CAB Wizard

To identify and avoid problems that might occur when using the CAB Wizard, follow these guidelines:

- Use %% for a percent sign (%) character when using this character in an .INF file string, as specified in Win32 documentation. This will not work under the [**Strings**] section.
- Do not use .INF or .CAB files created for Windows CE to install applications on Windows-based desktop platforms.
- Ensure the MAKECAB.EXE and CABWIZ.DDF files, included with Windows CE, are in the same directory as CABWIZ.EXE.
- Use the full path to call CABWIZ.EXE.
- Do not create a .CAB file with the MAKECAB.EXE file included with Windows CE. You must use CABWIZ.EXE, which uses MAKECAB.EXE to generate the .CAB files for Windows CE.
- Do *not* set the read-only attribute for .CAB files.

Customization and Lockdown

Pocket PC (Windows Mobile) is a hardware specification created by Microsoft Corporation. Devices that wish to carry the Pocket PC logo must meet the minimum hardware requirements set in the Pocket PC specification. Manufacturers are free to add extra hardware functionality.

Pocket PC devices also use a specialized version of the CE operating system. This operating system is built from Windows CE 4.2 but contains customizations, most notably the lack of a desktop and the addition of the Today Screen.

To carry the Pocket PC logo, all devices must be tested at an Independent Test Laboratory. The ITL testing is done based on Microsoft requirements. The test lab then reports the findings back to Microsoft Corporation and Intermec Technologies. If the 700 Color Computer passed all tests, Intermec is allowed to ship the device with the Pocket PC logo. Each time the operating system is modified, Intermec must resubmit to ITL testing.

This means we cannot change the operating system much and still be a Pocket PC device. For example, if we remove Word from the Start menu, the device would fail ITL testing and we would not be able to ship devices with the Pocket PC logo.

Although many customers want a Pocket PC device, some customers would prefer that their users not have access to all of the Pocket PC features. Intermec cannot customize the operating system in any way but a custom application can:

Delete items from the Start menu, and Programs folder. These items are just shortcuts in the file system so the application is not really being deleted. Cold booting the device will bring these items back so the application will need to be run on every cold boot.
Use the RegFlushKey() API to save a copy of the registry to a storage device. See the 700 Color Management Tools portion for more information on how to do this. Saving a copy of the registry restores most system settings in a cold boot situation.
Use the SHFullScreen() API in conjunction with other APIs to make the application take up the entire display and prevent the start menu from being available.
Remap keys and disable keys on the keypad.
Create a custom SIP.
Make changes to the registry to configure the device.

Should you want your 700 Color Computer to display a full screen, keep in mind that your computer is Pocket-PC certified by Microsoft Corporation. Check out resources on programming for the Pocket PC, using the following links. These give full instructions on how to display full screen.

- Instructions on how to create a full screen application for eVC++ applications using an SHFullScreen() API:
  http://support.microsoft.com/support/kb/articles/Q266/2/44.ASP
- Instructions on how to create a full screen application for eVB applications also using the SHFullScreen() API:
  http://support.microsoft.com/support/kb/articles/Q265/4/51.ASP

FTP Server

FTP support is provided through the FTP Server application FTPDCE.EXE (MS Windows CE Versions) which is provided as part the base system.

FTPDCE is the Internet File Transfer Protocol (FTP) server process. The server can be invoked from an application or command line. Besides servicing FTP client requests the FTP Server also send a “network announcement” to notify prospective clients of server availability.

Note: You should consult the RFC959 specification for proper use of some of these commands at the following URL:

- http://www.ietf.org/rfc/rfc959.txt for the text version, or
- http://www.w3.org/Protocols/rfc959/ for an html version

Do the following to send commands:

1. Start an FTP client and connect to the device FTP server.
2. Log in with “intermec” as the user name and “cr52401” for the password.
3. From the FTP client, send the command.
4. Wait for a response.

Synopsis

ftpdce [*options*]

Options

–Aaddr	(where *addr* is in the form of a.b.c.d) Sets the single target address to which to send the network announcement. *Default is broadcast.*
–Bbyte	Sets the FTP data block size. Smaller sizes may be useful over slower links. *Default is 65536.*
–Cname	Sets the device name. Used by Intermec management software.
–Fvalue	Disables the default Intermec account. A value of “0” disables the account. *Default is “1”.* Note that disabling the default account without providing a working access control list on the server will result in a device that will not accept any FTP connections.
–Hsec	Sets the interval between network announcements in seconds.A value of “0” turns the network announcement off. *Default is 30 seconds.*
–Iaddr	(where *addr* is in the form of a.b.c.d) Sets the preferred 6920 Communications Server *(optional).*
–Llog	(where *log* is either “0” or “1”) Sets the state of logging. *Default is 0 (disabled).*
–Nsec	Specifies the number of seconds to wait before initially starting FTP server services.
–Pport	Sets the UDP port on which the network announcement will be sent. *Default port is 52401.*
–Qport	Sets the port on which the FTP Server will listen for connections. *Default port is 21.*
–Rdir	Sets the FTP mount point to this directory. Default is the root folder of the object store.
–Tscrip	Sets the script name for the 6920 Communications Server to process.
–Uurl	Sets the default URL for this device.
–Z“parms”	Sets extended parameters to be included in the network announcement.

Configurable Parameters Via the Registry Editor

The following parameters receive default values during the installation of the Intermec FTP Server components. A few of the parameters are visible in the registry by default, but most must be created in order to modify the default behavior of the FTP server.

BlockSize

Setting this parameter configures the Intermec FTP Server to transmit and receive Ethernet packets using the specified data block size. By default, the FTP server transmits and receives data using a 64K data block size. Adjusting this value may be useful in certain wireless TCP/IP installations.

Key	`HKLM\Software\Intermec\IFTP`
Value Type	REG_DWORD - data block size, in bytes.
Valid Range	0x100-0x10000 (256-65536 decimal).
Default	65536

DeviceName

This parameter configures the Intermec FTP Server to include the specified device name in the Intermec Device Network Announcement (IDNA). Adjusting this value may be useful in assigning a symbolic name to this device for asset tracking.

Key	`HKLM\Software\Intermec\IFTP`
Value Type	REG_SZ
Valid Range	None.
Default	None.

DeviceURL

This parameter configures the Intermec FTP Server to transmit the specified URL in the IDNA. This can be used by Intermec management software for asset management.

Key	`HKLM\Software\Intermec\IFTP`
Value Type	REG_SZ
Valid Range	None.
Default	None.

IDNATarget

This parameter configures the Intermec FTP Server to transmit the IDNA to a specific destination instead of a general UDP broadcast. This parameter is useful on networks that do not allow UDP broadcasts to be routed between subnets. The use of this parameter restricts the reception of the IDNA to the target destination only.

Key	`HKLM\Software\Intermec\IFTP`
Value Type	REG_SZ
Valid Range	None.
Default	None.

ManifestName

This parameter configures the Intermec FTP Server to transmit the specified manifest name in the IDNA. This parameter is used by the Intermec 6920 Communications Server for communication transactions. See the 6920 Communications Server documentation for proper use of this parameter.

Key	`HKLM\Software\Intermec\IFTP`
Value Type	REG_SZ
Valid Range	None.
Default	iftp.ini

PauseAtStartup

This parameter configures the Intermec FTP Server to sleep for the specified number of seconds before making the FTP service available on the device.

Key	`HKLM\Software\Intermec\IFTP`
Value Type	REG_DWORD - stored in seconds.
Valid Range	None.
Default	0

Root

This parameter configures the Intermec FTP Server to set the root of the FTP mount point to the specified value. *Note that this must map to an existing directory or you will not be able to log into the FTP Server.*

Key	`HKLM\Software\Intermec\IFTP`
Value Type	REG_SZ
Valid Range	None.
Default	\

Transferring Files Over TCP/IP Networks

The File Transfer Protocol (FTP) server transfers files over TCP/IP networks. The FTPDCE.EXE program is a version that does not display a window, but can run in the background.

FTPDCE is the Internet File Transfer Protocol (FTP) server process. The server can be invoked from an application or command line. Besides servicing FTP client requests, the FTP Server also sends a “network announcement” to notify prospective clients of server availability.

Remarks

The FTP Server currently supports the following FTP requests:

CDUP	Changes to the parent directory of the current working directory.
CWD	Changes working directory.
DELE	Deletes a file.
HELP	Gives help information.
LIST	(This FTP request is the same as the ls -lgA command). Gives list files in a directory.
MKD	Makes a directory.
MODE	*(Always Uses Binary).* Specifies data transfer mode.
NLST	*(Not supported)* Gives a name list of files in directory (this FTP request is the same as the *ls* command).
NOOP	Does nothing.
PASS	Specifies a password.
PWD	Prints the current working directory.
QUIT	Terminates session.
RETR	Retrieves a file.
RMD	Removes a directory.
RNFR	Specifies rename-from file name.
RNTO	Specifies rename-to file name.
STOR	Stores a file.
SYST	Shows the operating system type of server system.
TYPE	*(Binary transfers only.)* Specifies the data transfer type with the Type parameter.
USER	Specifies user name.
XCUP	*(Not Normally Used)* Changes the parent directory of the current working directory.
XCWD	*(Not Normally Used)* Changes the current directory.
XMKD	*(Not Normally Used)* Creates a directory.
XPWD	*(Not Normally Used)* Prints the current working directory.
XRMD	*(Not Normally Used)* Removes a directory.

SITE	The following extended OEM commands are supported by the SITE request. For Microsoft FTP clients, you can send site commands by preceding the command with “quote” such as “quote site status.”	
	ATTRIB	Gets or sets the attributes of a given file. (SITE ATTRIB)
		Usage **QUOTE SITE ATTRIB** [+*R* | -*R*] [+*A* | -*A*] [+*S* | -*S*] [+*H* | -*H*] [[*path*] *filename*] + Sets an attribute. – Clears an attribute. *R* Read-only file attribute. *A* Archive file attribute. *S* System file attribute. *H* Hidden file attribute. To retrieve the attributes of a file, only specify the file. The server response will be: *200-AD SHRCEIX filename* If the flag exists in its position shown above, it is set. Also, in addition to the values defined above, there is also defined: *C* Compressed file attribute. *E* Encrypted file attribute. *I* INROM file attribute. *X* XIP file attribute (execute in ROM, not shadowed in RAM).
	BOOT	Reboots the server OS. This will cause the system on which the server is executing to reboot. The FTP Server will shut down cleanly before reboot. All client connections will be terminated. Cold boot is default except for the PocketPC build in which the default is warm boot. (SITE BOOT) Usage: **QUOTE SITE BOOT** [*WARM* | *COLD*]
	COPY	Copies a file from one location to another. (SITE COPY) Usage: **QUOTE SITE COPY** [*source*] [*destination*] Example: `QUOTE SITE COPY ‘\Storage Card\one.dat’ ‘\Storage Card\two.dat’`
	EXIT	Exits the FTP Server. This command will shut down the FTP Server thus terminating all client connections. (SITE EXIT) Usage: **QUOTE SITE EXIT**
	HELP	Gives site command help information. (SITE HELP) Usage: **QUOTE SITE HELP** [*command*]
	KILL	Terminates a running program. (SITE KILL) Usage: **QUOTE SITE KILL** [*program* | *pid*]
	LOG	Opens or closes the program log. (SITE LOG) Usage: **QUOTE SITE LOG** [*open* [*filename*]| *close*]
	PLIST	Lists the running processes (SITE PLIST) Usage: **QUOTE SITE PLIST**
	RUN	Starts a program running. If the program to run has spaces in path or filename, wrapping the name with single quotes is required. Usage: **QUOTE SITE RUN** [*program*] Example: `QUOTE SITE RUN ‘\Storage Card\app.exe’`

STATUS	Returns the current settings of the FTP Server. MAC, serial number, model, IP address, network announcement information as well as OS memory usage are returned. (SITE STATUS) Usage: **QUOTE SITE STATUS**
TIMEOUT	Toggles idle timeout between 120 to 1200 seconds (2 to 20 minutes). If this timer expires with no activity between the client and the server, the client connection will be disconnected. If the optional seconds argument is supplied, the server will set the connection timeout to the number of seconds specified. *Default is 120 seconds or 2 minutes.* (SITE TIMEOUT) Usage: **QUOTE SITE TIMEOUT** [*seconds*]
EKEY	Gives site command electronic key information. (SITE HELP) Usage: **QUOTE SITE EKEY** [*command*]
EVAL	Gives site command electronic value information. (SITE HELP) Usage: **QUOTE SITE EVAL** [*command*]
GVAL	Gives site command general value information. (SITE HELP) Usage: **QUOTE SITE GVAL** [*command*]
PVAL	Gives site command value information. (SITE HELP) Usage: **QUOTE SITE PVAL** [*command*]

The remaining FTP requests specified in RFC 959 are recognized, but not implemented.

The banner returned in the parenthetical portion of its greeting shows the version number of the FTP Server as well as the MAC address, serial number and operating system of the machine hosting the server.

The FTP Server supports browsing from the latest Netscape and Microsoft web browsers. Drag-and-drop capability is available using this environment.

The FTPDCMDS subdirectory contains commands to use from the web browser.

- Click EXITME.BIN to execute a SITE EXIT command.
- Click REBOOTME.BIN to execute SITE BOOT command.
- Use the GET command on these files to have the FTP Server execute these commands.
  - **Security**:
    A customer configurable access control list may be installed on the 700 Color Computer. This list will allow customers to restrict access via the FTP Server to users they wish and is in addition to default Intermec accounts that are disabled using the *-F0* option at runtime.

    The access control list is named FTPDCE.TXT and is placed in the same directory on the 700 Color Computer as the FTPDCE.EXE server. The FTP Server encrypts this file to keep the information safe from unauthorized users. This file is encrypted when the FTP Server is started so a file that is placed onto the 700 Color Computer after the FTP Server starts will require a restart of the FTP Server to take effect.

The format of the FTPDCE.TXT is as follows:

```
FTPDCE:user1!passwd1<cr><lf>user2!passwd2<cr><lf>user3
!passwd3<cr><lf>...
```

Note: The user accounts and passwords are case sensitive.
Once the access control list is encrypted on the 700 Color Computer, the FTP Server hides this file from users. Once an access control list is installed on the 700 Color Computer, a new one is not accepted by the FTP Server until the previous one is removed. Encrypted access control lists are not portable between 700 Color Computers.

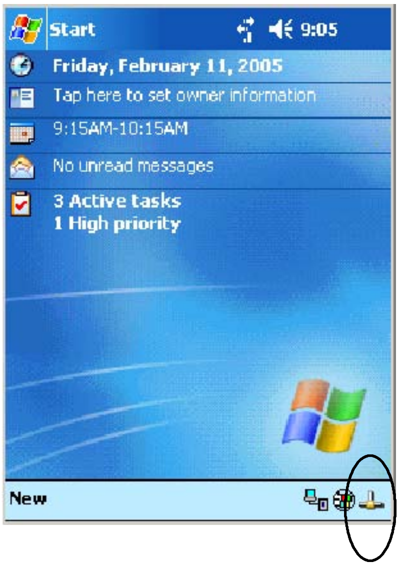
Stopping the FTP Server from Your Application

To allow application programmers the ability to programmatically shut down the FTP Server, the FTP Server periodically tests to see if a named event is signaled. The name for this event is “ITC_IFTP_STOP” (no quotes).

For examples on how to use events, consult the Microsoft Developer Network Library at www.msdn.com. The MSDN Library is an essential resource for developers using Microsoft tools, products, and technologies. It contains a bounty of technical programming information, including sample code, documentation, technical articles, and reference guides.

Autostart FTP

This automatically starts the FTP Server (FTPDCE.EXE) when the 700 Color Computer is powered on. This is provided with the NDISTRAY program (the Network Driver Interface Specification tray application), which displays the popup menu that currently allows you to load and unload the network drivers. Tap the antenna icon in the System Tray of the Today screen *(a sample antenna icon is circled below)* for this pop-up menu.



Friday, February 11, 2005

Tap here to set owner information

9:15AM-10:15AM

No unread messages

3 Active tasks
1 High priority

The default is to start the FTP Server at boot time, unless the following registry entry is defined and set to “0” which disables AutoFTP. “1” enables the AutoFTP. The entry can be set from the NDISTRAY pop-up menu by selecting either **AutoFTP On** or **AutoFTP Off**.

```
HKEY_LOCAL_MACHINE\Software\Intermec\Ndistray\StartupIFTP
```

These new entries are located below the selections to load the network drivers. If the StartupIFTP registry key is not defined, the FTP Server is loaded by default, to provide “out-of-the-box” capability for customers who want to begin loading files to the 700 Color Computer without any prior configuration.

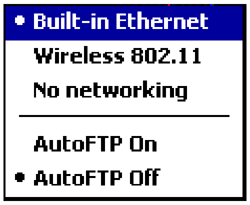
Note: If a network driver is unloaded using the NDISTRAY popup menu, and the FTP Server is running, the FTP Server is stopped.

On a resume, if AutoFTP is enabled and the FTP Server is running, it is stopped and restarted. NDISTRAY uses a helper application named RESETIFTP to implement the restart on resume feature.

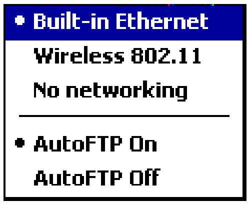
To do an AutoFTP Installation Check:

1 Ensure the FTP Server is running “out-of-the-box” the first time.

2 Tap **Start** > **Today** to access the Today screen, then tap the antenna icon in the System Tray to bring up the NDISTRAY pop-up menu. Select **AutoFTP Off** to disable AutoFTP. Do a warm boot and confirm the FTP Server is not running.



3 Tap **Start** > **Today** to access the Today screen, then tap the antenna icon in the System Tray to bring up the NDISTRAY pop-up menu. Select **AutoFTP On** to enable AutoFTP, reboot, confirm it is running.



4 Unload the network driver when the FTP Server is running and confirm that it is not running any more.

5 Load the FTP Server, establish a connection, then suspend and resume. The server should still run, but the FTP connection to the client should be dropped.

Kernel I/O Controls

This describes the KernelIoControl() functions available to application programmers. Most C++ applications will need to prototype the function as the following to avoid link and compile errors.

```
extern “C” BOOL KernelIoControl(DWORD dwIoControlCode, LPVOID lpInBuf, DWORD
nInBufSize, LPVOID lpOutBuf, DWORD nOutBufSize, LPDWORD lpBytesReturned);
```

IOCTL_HAL_GET_DEVICE_INFO

This IOCTL returns either the platform type or the OEMPLATFORM name based on an input value.

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_GET_DEVICE_INFO, LPVOID
lpInBuf, DWORD nInBufSize, LPVOID lpOutBuf, DWORD
nOutBufSize, LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Points to a DWORD containing either the SPI_GETPLATFORMTYPE or SPI_GETOEMINFO value.
lpInBufSize	Must be set to sizeof(DWORD).
lpOutBuf	Must point to a buffer large enough to hold the return data of the function. If SPI_GETPLATFORMTYPE is specified in *lpInBuf*, then the “PocketPC\0” Unicode string is returned. If SPI_GETOEMINFO is specified in *lpInBuf*, then the “Intermec 700\0” Unicode string is returned.
nOutBufSize	The size of *lpOutBuf* in bytes. Must be large enough to hold the string returned.
lpBytesReturned	The actual number of bytes returned by the function for the data requested.

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

IOCTL_HAL_ITC_READ_PARM

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_ITC_READ_PARM,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Points to this structure. See “*ID Field Values*” below.
	`struct PARMS {`
	` BYTE id;`
	` BYTE ClassId;`
	`};`

nInBufSize	Must be set to the size of the PARMS structure.
lpOutBuf	Must point to a buffer large enough to hold the return data of the function. If this field is set to NULL and *nOutBufSize* is set to zero when the function is called the function will return the number bytes required by the buffer.
nOutBufSize	The size of *lpOutBuf* in bytes.
lpBytesReturned	Number of bytes returned by the function for the data requested.

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the error value. Either ERROR_INVALID_PARAMETER or ERROR_INSUFFICIENT_BUFFER may be returned when this function is used to get the error.

ID Field Values

The *id* field of the PARMS structure may be one of the following values:

ID Field Values
ITC_NVPARM_ETHERNET_ID This IOCTL returns the Ethernet 802.11b or 802.11b/g MAC Address. Six bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_SERIAL_NUM This IOCTL returns the serial number of the device in BCD format. Six bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_MANF_DATE This IOCTL returns the device date of manufacture in the BCD YYYY/MM/DD format. Four bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_SERVICE_DATE This IOCTL returns the device’s date of last service in BCD YYYY/MM/DD format. Four bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.

ID Field Values (continued)
ITC_NVPARM_DISPLAY_TYPE This IOCTL returns the device's display type. One byte is returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_EDG_IP This IOCTL returns the device Ethernet debug IP address. Four bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_EDBG_SUBNET This IOCTL returns the device Ethernet debug subnet mask. Four bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_ECN This IOCTL returns ECNs applied to the device in a bit array format. Four bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_CONTRAST This IOCTL returns the device default contrast setting. Two bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_MCODE This IOCTL returns the manufacturing configuration code for the device. Sixteen bytes are returned in the buffer pointed to by the *lpOutBuffer* parameter.
ITC_NVPARM_VERSION_NUMBER This IOCTL returns the firmware version for various system components. These values for the *ClassId* field of the PARMS structure are allowed when ITC_NVPARM_VERSION_NUMBER is used in the *id* field: • **VN_CLASS_KBD** Returns a five-byte string, including null terminator, that contains an ASCII value which represents the keypad microprocessor version in the system. The format of the string is *x.xx* with a terminating null character. • **VN_CLASS_ASIC** Returns a five-byte string, including null terminator, that contains an ASCII value which represents the version of the FPGA firmware in the system. The format of the string is *x.xx* with a terminating null character. • **VN_CLASS_BOOTSTRAP** Returns a five-byte string, including null terminator, that contains an ASCII value which represents the version of the Bootstrap Loader firmware in the system. The format of the string is *x.xx* with a terminating null character.
ITC_NVPARM_INTERMEC_SOFTWARE_CONTENT This IOCTL reads the manufacturing flag bits from the non-volatile data store that dictates certain software parameters. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer* that indicates if Intermec Content is enabled in the XIP regions. TRUE indicates that it is enabled. FALSE indicates that it is not enabled.
ITC_NVPARM_ANTENNA_DIVERSITY This IOCTL reads the state of the antenna diversity flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer* that indicates if there is a diversity antenna installed. TRUE indicates that it is installed. FALSE indicates that it is not installed.
ITC_NVPARM_WAN_RI This IOCTL reads the state of the WAN ring indicator flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer* that indicates the polarity of the WAN RI signal. TRUE indicates active high. FALSE indicates active low.

ID Field Values (continued)
ITC_NVPARM_RTC_RESTORE This IOCTL reads the state of the real-time clock restore flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the RTC is restored upon a cold boot. FALSE indicates that the RTC is not restored.
ITC_NVPARM_INTERMEC_DATACOLLECTION_SW This IOCTL reads the state of the data collection software enabled flag. A BOOLEAN DWORD is returned in the buffer pointer to by *lpOutBuffer* that indicates the data collection software is to install at boot time. FALSE indicates the data collection software should not install.
ITC_NVPARM_INTERMEC_DATACOLLECTION_HW This IOCTL reads the data collection hardware flags. A BYTE is returned in the buffer pointer to by *lpOutBuffer* that indicates the type of data collection hardware installed. The maximum possible value returned is ITC_DEVID_SCANHW_MAX. • **ITC_DEVID_SCANHW_NONE** No scanner hardware is installed. • **ITC_DEVID_OEM2D_IMAGER** OEM 2D imager is installed. • **ITC_DEVID_INTERMEC2D_IMAGER** Intermec 2D imager is installed. • **ITC_DEVID_SE900_LASER** SE900 laser is installed. • **ITC_DEVID_SE900HS_LASER** SE900HS laser is installed. • **ITC_DEVID_INTERMEC_EVIO** EVIO linear imager is installed. The high bit indicates whether the S6 scanning engine is installed. The bit mask for this is ITC_DEVID_S6ENGINE_MASK. A non-zero value indicates that the S6 scanning engine is installed.
ITC_NVPARM_WAN_INSTALLED This IOCTL reads the state of the WAN radio installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the WAN radio is installed. FALSE indicates that no WAN radio is installed.
ITC_NVPARM_WAN_FREQUENCY This IOCTL reads the state of the WAN radio frequency flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the WAN radio frequency is United States. FALSE indicates that the WAN radio frequency is European.
ITC_NVPARM_WAN_RADIOTYPE This IOCTL reads the WAN radio ID installed by manufacturing. A BYTE is returned in the buffer pointer to by *lpOutBuffer* which indicates the type of WAN radio hardware installed. The maximum possible value returned is ITC_DEVID_WANRADIO_MAX. The current definitions are: • **ITC_DEVID_WANRADIO_NONE** No WAN radio installed. • **ITC_DEVID_WANRADIO_SIERRA_SB555** CDMA Sierra Wireless radio. • **ITC_DEVID_WANRADIO_XIRCOM_GEM3503** GSM/GPRS Intel (Xircom) radio. • **ITC_DEVID_WANRADIO_SIEMENS_MC45** GSM/GPRS Siemens radio. • **ITC_DEVID_WANRADIO_SIEMENS_MC46** GSM/GPRS Siemens radio.
ITC_NVPARM_80211_INSTALLED This IOCTL reads the state of the 802.11b or 802.11b/g radio installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the 802.11b or 802.11b/g radio is installed. FALSE indicates that no 802.11b or 802.11b/g radio is installed.
ITC_NVPARM_80211_RADIOTYPE This IOCTL reads the 802.11b or 802.11b/g radio ID installed by manufacturing. A BYTE is returned in the buffer pointer to by *lpOutBuffer* that indicates the type of 802.11b or 802.11b/g radio hardware installed. The maximum possible value returned is ITC_DEVID_80211RADIO_MAX. The current definitions are: • **ITC_DEVID_80211RADIO_NONE** No 802.11b or 802.11b/g radio installed. • **ITC_DEVID_80211RADIO_INTEL_2011B** Intel 2011B radio installed.

ID Field Values (continued)
ITC_NVPARM_BLUETOOTH_INSTALLED This IOCTL reads the state of the Bluetooth radio installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the Bluetooth radio is installed. FALSE indicates that no Bluetooth radio is installed.
ITC_NVPARM_SERIAL2_INSTALLED This IOCTL reads the state of the serial 2 (COM2) device installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the serial 2 device is installed. FALSE indicates that no serial 2 device is installed.
ITC_NVPARM_VIBRATE_INSTALLED This IOCTL reads the state of the vibrate device installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the vibrate device is installed. FALSE indicates that no vibrate device is installed.
ITC_NVPARM_LAN9000_INSTALLED This IOCTL reads the state of the Ethernet device installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the Ethernet device is installed. FALSE indicates that no Ethernet device is installed.
ITC_NVPARM_SIM_PROTECT_HW_INSTALLED This IOCTL reads the state of the SIM card protection hardware installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the SIM card protection hardware is installed. FALSE indicates that no SIM card protection hardware is installed.
ITC_NVPARM_SIM_PROTECT_SW_INSTALLED This IOCTL reads the state of the SIM card protection software installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the SIM card protection software is installed. FALSE indicates that no SIM card protection software is installed.
ITC_NVPARM_SIM_PROTECT_SW_INSTALLED This IOCTL reads the state of the SIM card protection software installed flag. A BOOLEAN DWORD is returned in the buffer pointed to by *lpOutBuffer*. TRUE indicates that the SIM card protection software is installed. FALSE indicates that no SIM card protection software is installed.

IOCTL_HAL_ITC_WRITE_SYSPARM

Describes and enables the registry save location.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_ITC_WRITE_SYSPARM,LPVOID
lpInBuf,DWORD nInBufSize, LPVOID lpOutBuf, DWORD
nOutBufSize, LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	A single byte that may be one of the *id* values. See “*ID Field Values*” on the next page.
nInBufSize	Must be set to the size of the *lpInBuf* in bytes.
lpOutBuf	Must point to a buffer large enough to hold the data to be written to the non-volatile data store.
nOutBufSize	The size of *lpOutBuf* in bytes.
lpBytesReturned	The number of bytes returned by the function.

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the error value. Either ERROR_INVALID_PARAMETER or ERROR_INSUFFICIENT_BUFFER may be returned when this function is used to get the error.

ID Field Values

The *id* field of *lpInBuf* may be one of the following values:

ID Field Values

ITC_REGISTRY_SAVE_ENABLE
This function enables or disables the save registry to non-volatile media feature of the RegFlushKey() function. *lpOutBuf* must be set to zero (FALSE) if the feature is to be disabled or one (TRUE) if the feature is to be enabled.

ITC_ DOCK_SWITCH
This IOCTL sets a position of the dock switch. The dock switch may be set to either “modem” or “serial” positions. *lpOutBuf* must point to a buffer that contains a byte value of either DOCK_MODEM or DOCK_SERIAL as defined in OEMIOCTL.H; the value specifies the position the switch is to be set. The call appears as follows:

```
// port = DOCK_MODEM or DOCK_SERIAL as defined in oemioctl.h
BOOL SetDockSwitch( BYTE port)
{
 DWORD cmd = ITC_DOCK_SWITCH;
 DWORD cbRet;

 return KernelIoControl(IOCTL_HAL_ITC_WRITE_SYSPARM,&cmd, sizeof(cmd),
&port,sizeof(port),&cbRet)
}
```

ITC_ WAKEUP_MASK
This IOCTL sets a bit mask that represents the mask for the five programmable wakeup keys. The I/O key is not a programmable wakeup key. By default it is always the system resume key and all other keys are set to disable key wakeup. A zero in a bit position masks the wakeup for that key. A one in a bit position enables wakeup for that key. *lpOutBuf* must point to a buffer that contains a byte value of a wakeup mask consisting of the OR’ed constants as defined in OEMIOCTL.H. Only the following keys are programmable as wakeup events.

```
#define  SCANNER_TRIGGER 1
#define  SCANNER_LEFT    2
#define  SCANNER_RIGHT   4
#define  GOLD_A1         8
#define  GOLD_A2         0x10
```

ITC_AMBIENT_KEYBOARD *(does not apply to the 730 Computer)*
This IOCTL sets the threshold for the keypad ambient sensor. This can be a value from 0 (always off) to 255 (always on). *lpOutBuf* must point to a buffer that contains a byte value of the desired setting.

ITC_AMBIENT_FRONTLIGHT *(does not apply to the 730 Computer)*
This IOCTL sets the threshold for the frontlight ambient sensor. This can be a value from 0 (always off) to 255. *lpOutBuf* must point to a buffer that contains a byte value of the desired setting.

IOCTL_HAL_GET_DEVICEID

This IOCTL returns the device ID. There are two types of device IDs supported, which are differentiated based on the size of the *output* buffer. The UUID is returned if the buffer size is set to *sizeof(UNIQUE_DEVICEID)*, otherwise the oldstyle device ID is returned.

Usage

#include “pkfuncs.h”
#include “deviceid.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_GET_DEVICEID,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL. STRICT_ID settings are not supported.
lpInBufSize	Should be set to zero.
lpOutBuf	Must point to a UNIQUE_DEVICEID structure as defined by DEVICEID.H if the UUID is to be returned
nOutBufSize	The size of the UNIQUE_DEVICEID in bytes if the UUID is to be returned. A DEVICE_ID as defined by PKFUNCS.H is returned if the size in bytes is greater than or equal to *sizeof(DEVICE_ID)*.
lpBytesReturned	The number of bytes returned by the function.

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

IOCTL_HAL_GET_OAL_VERINFO

Returns the HAL version information of the Pocket PC image.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_GET_OAL_VERINFO,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL.
lpInBufSize	Should be set to zero.
lpOutBuf	Must point to a VERSIONINFO structure as defined by OEMIOCTL.H. The fields should have these values: • `cboemverinfo` sizeof (tagOemVerInfo); • `verinfover` 1 • `sig;` “ITC\0” • `id;` ‘N’ • `tgtcustomer` “” • `tgtplat` SeaRay • `tgtplatversion` Current build version number • `tgtcputype[8];` “Intel\0” • `tgtcpu` “PXA255\0”; • `tgtcoreversion` “” • `date` Build time • `time` Build date
nOutBufSize	The size of VERSIONINFO in bytes.
lpBytesReturned	Returns *sizeof(PVERSIONINFO).*

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

IOCTL_HAL_GET_BOOTLOADER_VERINFO

Returns the HAL version information of the Pocket PC image.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_GET_OAL_VERINFO,LPVOID
lpInBuf, DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL.
nInBufSize	Should be set to zero.
lpOutBuf	Must point to a VERSIONINFO structure as defined by OEMIOCTL.H. The fields should have these values: • `cboemverinfo` Sizeof (tagOemVerInfo); • `verinfover` 1 • `sig;` “ITC\0” • `id;` ‘B’ • `tgtcustomer` “” • `tgtplat` SeaRay • `tgtplatversion` Current build version number of the bootstrap loader • `tgtcputype[8];` “Intel\0”; • `tgtcpu` “PXA255\0” • `tgtcoreversion` “” • `date` Build time • `time` Build date
nOutBufSize	The size of VERSIONINFO in bytes.
lpBytesReturned	The number of bytes returned to *lpOutBuf*.

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

IOCTL_HAL_WARMBOOT

Causes the system to perform a warm-boot. The object store is retained.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_WARMBOOT,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL.
lpInBufSize	Should be set to zero.
lpOutBuf	Should be NULL.
nOutBufSize	Should be zero.

Return Values

None.

IOCTL_HAL_COLDBOOT

Causes the system to perform a cold-boot. The object store is cleared.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_COLDBOOT,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL.
lpInBufSize	Should be set to zero.
lpOutBuf	Should be NULL.
nOutBufSize	Should be zero.

Return Values

None.

IOCTL_HAL_GET_RESET_INFO

This IOCTL code allows software to check the type of the most recent reset.

Usage

#include “oemioctl.h”

Syntax

BOOL **KernelIoControl(** IOCTL_HAL_GET_RESET_INFO,LPVOID *lpInBuf*,DWORD *nInBufSize*,LPVOID *lpOutBuf*,DWORD *nOutBufSize*,LPDWORD *lpBytesReturned* **);**

Parameters

lpInBuf	Should be set to NULL.
lpInBufSize	Should be set to zero.
lpOutBuf	Must point to a HAL_RESET_INFO structure. See sample below.
nOutBufSize	The size of HAL_RESET_INFO in bytes.
lpBytesReturned	The number of bytes returned by the function.

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

Sample

```
typedef struct {
  DWORD ResetReason;                         // most recent reset type
  DWORD ObjectStoreState;                    // state of object store
} HAL_RESET_INFO, * PHAL_RESET_INFO;

// Reset reason types
#define HAL_RESET_TYPE_UNKNOWN          0
#define HAL_RESET_REASON_HARDWARE       1  // cold
#define HAL_RESET_REASON_SOFTWARE       2  // suspend
#define HAL_RESET_REASON_WATCHDOG       4
#define HAL_RESET_BATT_FAULT            8  // power fail
#define HAL_RESET_VDD_FAULT             16 // warm boot

// Object store state flags
#define HAL_OBJECT_STORE_STATE_UNKNOWN    0
#define HAL_OBJECT_STORE_STATE_CLEAR      1
```

IOCTL_HAL_GET_BOOT_DEVICE

This IOCTL code allows software to check which device CE booted from.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_GET_BOOT_DEVICE,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL.
lpInBufSize	Should be set to zero.
lpOutBuf	Must point to a buffer large enough to hold a DWORD (4 bytes) that contains the boot device. The following boot devices are supported: #define HAL_BOOT_DEVICE_UNKNOWN 0 #define HAL_BOOT_DEVICE_ROM_XIP 1 #define HAL_BOOT_DEVICE_ROM 2 #define HAL_BOOT_DEVICE_PCMCIA_ATA 3 #define HAL_BOOT_DEVICE_PCMCIA_LINEAR 4 #define HAL_BOOT_DEVICE_IDE_ATA 5 #define HAL_BOOT_DEVICE_IDE_ATAPI 6
nOutBufSize	The size of *lpOutBuf* in bytes (4).
lpBytesReturned	The number of bytes returned by the function.

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

IOCTL_HAL_REBOOT

Causes the system to perform a warm-boot. The object store is retained.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_HAL_REBOOT,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL.
lpInBufSize	Should be set to zero.
lpOutBuf	Should be NULL.
nOutBufSize	Should be zero.

Return Values

None.

IOCTL_PROCESSOR_INFORMATION

Returns processor information.

Usage

#include “pkfuncs.h”

Syntax

```
BOOL KernelIoControl( IOCTL_PROCESSOR_INFORMATION,LPVOID
lpInBuf,DWORD nInBufSize,LPVOID lpOutBuf,DWORD
nOutBufSize,LPDWORD lpBytesReturned );
```

Parameters

lpInBuf	Should be set to NULL.
nInBufSize	Should be set to zero.
lpOutBuf	Should be a pointer to the PROCESSOR_INFO structure. The PROCESSOR_INFO structure stores information that describes the CPU more descriptively.

```
typedef __PROCESSOR_INFO {
WORD   wVersion;            // Set to value 1
WCHAR  szProcessorCore[40]; // “ARM\0”
WORD   wCoreRevision;       // 4
WCHAR  szProcessorName[40]; // “PXA255\0”
WORD   wProcessorRevision;  // 0
WCAHR  szCatalogNumber[100];// 0
WCHAR  szVendor[100];       // “Intel Corporation\0”
DWORD  dwInstructionSet;    // 0
DWORD  dwClockSpeed;        // 400
}
```

nOutBufSize	Should be set to sizeof(PROCESSOR_INFO) in bytes.
lpBytesReturned	Returns sizeof(PROCESSOR_INFO);

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

IOCTL_GET_CPU_ID

Returns Xscale processor ID.

Usage

#include “oemioctl.h”

Syntax

```
BOOL KernelIoControl( IOCTL_GET_CPU_ID,LPVOID lpInBuf,
DWORD nInBufSize,LPVOID lpOutBuf,DWORD nOutBufSize,LPDWORD
lpBytesReturned );
```

Parameters

lpInBuf	Should point to a CPUIdInfo structure defined in OEMIOCTL.H.
lpInBufSize	Should be *sizeof(CPUIdInfo)*.
lpOutBuf	Should be NULL.
nOutBufSize	Should be set to 0.
lpBytesReturned	Returns *sizeof(PROCESSOR_INFO);*

Return Values

Returns TRUE if function succeeds. Returns FALSE if the function fails. GetLastError() may be used to get the extended error value.

Network Selection APIs

The Network Selection APIs change the network adapter configuration programmatically. Both drivers support the same IOCTL function numbers for loading and unloading the drivers.

Loading and unloading of the 802.11b or 802.11b/g driver is performed by the FWL1: device in the system by performing DeviceIOControl() calls to the driver.

Loading and unloading of the driver for the built-in Ethernet adapter is performed by the SYI1: device in the system by performing DeviceIOControl() calls to the driver.

- For loading an NDIS driver associated with an adapter, the IOCTL is IOCTL_LOAD_NDIS_MINIPORT.
- For unloading NDIS drivers associated with an adapter the IOCTL is IOCTL_UNLOAD_NDIS_MINIPORT.

Example

```
#include <winioctl.h>
#include “sysio.h”
void DoLoad(int nDevice) {
   LPTSTR devs[] = { _T(“SYI1:”), _T(“FWL1:”) };
   HANDLE hLoaderDev;
   DWORD bytesReturned;
   hLoaderDev = CreateFile(devs[nDevice], GENERIC_READ|GENERIC_WRITE, 0,
   NULL, OPEN_EXISTING, 0, NULL);
   if (hLoaderDev != INVALID_HANDLE_VALUE) {
      if (!DeviceIoControl( hLoaderDev, IOCTL_LOAD_NDIS_MINIPORT, NULL, -1, NULL, 0,
      &bytesReturned, NULL)){
         MessageBox(NULL, TEXT(“SYSIO IoControl Failed”), TEXT(“Network
         loader”),MB_ICONHAND);
         if (hLoaderDev!=INVALID_HANDLE_VALUE) CloseHandle(hLoaderDev);
         hLoaderDev = INVALID_HANDLE_VALUE; // bad handle
      }else {
         CloseHandle(hLoaderDev);
      }
   }
}
void DoUnload(int nDevice) {
   LPTSTR devs[] = { _T(“SYI1:”), _T(“FWL1:”) };
   HANDLE hLoaderDev;
   DWORD bytesReturned;
   hLoaderDev = CreateFile(devs[nDevice], GENERIC_READ|GENERIC_WRITE, 0,
   NULL, OPEN_EXISTING, 0, NULL);
   if (hLoaderDev != INVALID_HANDLE_VALUE) {
      if (!DeviceIoControl( hLoaderDev, IOCTL_UNLOAD_NDIS_MINIPORT, NULL, -1, NULL, 0,
      &bytesReturned, NULL)){
         MessageBox(NULL, TEXT(“SYSIO IoControl Failed”),TEXT(“Network
         loader”),MB_ICONHAND);
         if (hLoaderDev!=INVALID_HANDLE_VALUE) CloseHandle(hLoaderDev);
         hLoaderDev = INVALID_HANDLE_VALUE; // bad handle
      }else {
         CloseHandle(hLoaderDev);
      }
   }
}
```

The API provided by Intermec Technologies exposes a limited set of routines that allows a programmer to access and affect the 802.11b or 802.11b/g network interface card from within their application. The routines provided also reads/writes values to the CE registry that pertain to the 802.11b or 802.11b/g radio driver. By using the provided functions, a programmer can alter the 802.11b or 802.11b/g parameters of Network Name (SSID), WEP keys, infrastructure modes, radio channel, and power management modes. A programmer can also retrieve network connect status and signal strength indication from the RF network card.

The API is contained within the 80211API.DLL file that should be present in any load with the 802.11b or 802.11b/g networking installed.

NETWLAN.DLL PRISMNDS.DLL	This file is the 802.11b or 802.11b/g driver. It is present in all 700 CE loads that use the 802.11b or 802.11b/g network interface card.
80211API.DLL	This file is an Intermec authored file that provides the programmer with a set of API calls to configure or monitor status of the 802.11b or 802.11b/g network.
MOD80211.DLL	The CORE module for the 802.11 NIC. It provides the 802.11b or 802.11b/g status information displayed when the CORE application is running.
80211CONF.EXE	This is the “Control Panel” for configuring the 802.11b or 802.11b/g network parameters. Note that it is an EXE file and is actually called by CPL802.CPL (see below). It is also called by the CORE application when the “Configuration” button is pressed.
CPL802.CPL	A control panel application that does nothing but call 80211CONF.EXE.
80211SCAN.EXE	Internally manages the Scan List activity.
802PM.DLL	This handles profile management for radio configurable values.
URODDSVC.EXE	This handles radio configuration and security authentication based on a selected profile.

The Profile Manager supports up to four radio configuration profiles. These profiles are the same as those set by the Wireless Network control panel applet that runs on the Windows CE unit. You can configure different 802.11b or 802.11b/g profiles and switch between them using the 802.11 API. See the ConfigureProfile() function on page 283 for more information.

Basic Connect/Disconnect Functions

Below are functions available for the 700 Color Computer when enabled with the 802.11b or 802.11b/g radio module.

RadioConnect()

Connects to the available radio. Use this function if you plan on using a lot of API calls that talk directly to the radio. Note that the 802.11b or 802.11b/g radio must be enabled via NDISTRAY before you can connect to it.

Syntax	`UINT **RadioConnect( );**`
Parameters	None.
Return Values	ERROR_SUCCESS when successful, otherwise ERR_CONNECT_FAILED
Remarks	Call this function before you call any other function found within this API. It hunts out and connects to the 802.11b or 802.11b/g radio available on the system. Check extended error codes if it returns anything else for information.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_RadioConnect)();`
	`#else`
	`UINT RadioConnect();`
	`#endif`

RadioDisconnect()

Call this function when done using the 802.11 API to clean up a connection from a previous RadioConnect() call. If you do not call this function, you may leave memory allocated.

Syntax	`UINT **RadioDisconnect( );**`
Parameters	None.
Return Values	ERROR_SUCCESS when successful, otherwise ERR_CONNECT_FAILED.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_RadioDisconnect)();`
	`#else`
	`UINT RadioDisconnect();`
	`#endif`

RadioDisassociate()

Call this function to have the 802.11b or 802.11b/g radio disassociate from the current service set. The radio then enters an “off” mode until it is woken again by setting the Service Set Identifier (SSID). Also, the NDIS driver generates an NDIS media disconnect event.

Syntax	`UINT` **`RadioDisassociate( );`**
Parameters	None.
Return Values	ERROR_SUCCESS when successful, otherwise ERR_CONNECT_FAILED.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_RadioDisassociate)();`
	`#else`
	`UINT RadioDisassociate();`
	`#endif`

Query Information Functions

GetAssociationStatus()

Call this function to obtain the radio’s current association status with a service set.

Syntax	`UINT` **`GetAssociationStatus(`** `ULONG` *`&`* **`);`**	
Parameters	NDIS_RADIO_ASSOCIATED	Indicates the radio is associated with an access point
	NDIS_RADIO_SCANNING	Indicates the radio is looking for an access point with which to associate
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	Data is only valid if the function returns ERROR_SUCCESS. Also, if ERROR_SUCCESS is returned, your ULONG reference is populated by one of the parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetAssociationStatus)(ULONG &);`
	`#else`
	`UINT GetAssociationStatus(ULONG &);`
	`#endif`	

GetAuthenticationMode()

Call this function to obtain the radio's current authentication mode.

Syntax	`UINT` **`GetAuthenticationMode(`** `ULONG` *`&`* **`);`**	
Parameters	NDIS_RADIO_AUTH_MODE_OPEN	802.11 Open Authentication. Indicates that the radio is using an open system.
	NDIS_RADIO_AUTH_MODE_SHARED	802.11 Shared Authentication. Indicates that the radio is using a shared key.
	NDIS_RADIO_AUTH_MODE_AUTO	Auto switch between Open/Shared. Indicates automatic detection is used when available.
	NDIS_RADIO_AUTH_MODE_ERROR	Defined as error value. Indicates the authentication mode was not determined at this time or is unknown.
	NDIS_RADIO_AUTH_MODE_WPA	WPA Authentication
	NDIS_RADIO_AUTH_MODE_WPA_PSK	WPA Preshared Key Authentication
	NDIS_RADIO_AUTH_MODE_WPA_NONE	WPA None
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	Data is only valid if ERROR_SUCCESS is returned. Also, if ERROR_SUCCESS is returned, your USHORT reference is populated with one of the parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetAuthenticationMode)(ULONG &);`
	`#else`
	`UINT GetAuthenticationMode(ULONG &);`
	`#endif`	

GetBSSID()

Call this function to get the current MAC address (BSSID) of the service set. In ESS mode, this is the MAC address of the access point the radio is associated with. In IBSS mode, this is a randomly generated MAC address, and serves as the ID for the IBSS.

Syntax	`UINT` **`GetBSSID(`** `TCHAR *` **`);`**
Parameters	Pointer to a character array, which is populated with the current BSSID after a successful call.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	If ERROR_SUCCESS is returned, your TCHAR array is populated with the BSSID of the current service set: `xx-xx-xx-xx-xx-xx`

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetBSSID)(TCHAR *);`
	`#else`
	`UINT GetBSSID(TCHAR *);`
	`#endif`

GetDiversity()

Call this function to get the current diversity setting of your 802.11b or 802.11b/g radio. This uses an optional NDIS5.1 OID to query the radio, of which a large number of 802.11b or 802.11b/g devices do not support. This may be inaccurate.

Syntax	`UINT GetDiversity(USHORT *);`	
Parameters	ANT_PRIMARY	The primary antenna is selected.
	ANT_SECONDARY	The secondary antenna is selected.
	ANT_DIVERSITY	The radio is in diversity mode, and uses both antennas
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	If ERROR_SUCCESS is returned, your USHORT reference is populated with one of the parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetDiversity)(USHORT *);`
	`#else`
	`UINT GetDiversity(USHORT *);`
	`#endif`	

GetLinkSpeed()

Call this function to get the current link speed of the 802.11b or 802.11b/g radio.

Syntax	`UINT GetLinkSpeed( int & );`
Parameters	This function accepts an int reference, and your int is populated with the current link speed, in Mbps, rounded to the nearest whole integer, for example: 1, 2, 5, 11, etc.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	Data returned is valid if ERROR_SUCCESS is returned.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetLinkSpeed)(int &);`
	`#else`
	`UINT GetLinkSpeed(int &);`
	`#endif`

GetMac()

Call this function to get the MAC address of the 802.11b or 802.11b/g radio.

Syntax	`UINT GetMac( TCHAR * );`
Parameters	Pointer to a character array, which is populated with the MAC address after a successful call.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	If ERROR_SUCCESS is returned, your TCHAR array is populated with the formatted MAC address of the adapter, as follows: `xx-xx-xx-xx-xx-xx`

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetMac)(TCHAR *);`
	`#else`
	`UINT GetMac(TCHAR *);`
	`#endif`

Note: Be sure to call RadioConnect() *before* calling this function for this function to work properly.

GetNetworkMode()

Call this function to get the current Network Mode (SSID) for the 802.11b or 802.11b/g radio.

Syntax	`UINT GetNetworkMode( ULONG & );`	
Parameters	NDIS_NET_MODE_IBSS	802.11 Ad-Hoc Mode.
	NDIS_NET_MODE_ESS	802.11 Infrastructure Mode.
	NDIS_NET_MODE_UNKNOWN	Anything Else/Unknown Error
	NDIS_NET_AUTO_UNKNOWN	Automatic Selection. *Use of this option is not supported or recommended.*
	NDIS_NET_TYPE_OFDM_5G	5 Gigahertz 54 Mbps
	NDIS_NET_TYPE_OFDM_2_4G	802.11 2.4 Gigahertz
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	If ERROR_SUCCESS is returned, your ULONG reference is populated with one of the parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetNetworkMode)(ULONG &);`
	`#else`
	`UINT GetNetworkMode(ULONG &);`
	`#endif`	

GetNetworkType()

Call this function to get the current network type of the radio. Do not confuse this with GetNetworkMode().

Syntax	`UINT GetNetworkType( ULONG & );`	
Parameters	NDIS_NET_TYPE_FH	Indicates this is a frequency hopping radio.
	NDIS_NET_TYPE_DS	Indicates that this is a direct sequence radio.
	NDIS_NET_TYPE_UNDEFINED	Indicates this radio type is unknown or undefined.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	If ERROR_SUCCESS is returned, your ULONG reference is populated with one of the parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetNetworkType)(ULONG &);`
	`#else`
	`UINT GetNetworkType(ULONG &);`
	`#endif`	

GetSSID()

Call this function to get the desired SSID of the 802.11b or 802.11b/g radio.

Syntax	`UINT GetSSID( TCHAR * );`
Parameters	Pointer to a character array, which is populated with the current SSID when successful.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	If ERROR_SUCCESS is returned, your TCHAR array is populated with the desired SSID.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetSSID)(TCHAR *);`
	`#else`
	`UINT GetSSID(TCHAR *);`
	`#endif`

Note: Call RadioConnect() *before* this function for this function to work properly.

GetPowerMode()

Call this function to get the current power savings mode of the radio.

Syntax	`UINT **GetPowerMode(** ULONG *&* **);**`	
Parameters	NDIS_RADIO_POWER_MODE_CAM	Continuous Access Mode *(ie: always on)*.
	NDIS_RADIO_POWER_MODE_PSP	Power Saving Mode.
	NDIS_RADIO_POWER_UNKNOWN	Unknown power mode.
	NDIS_RADIO_POWER_AUTO	Auto. *(Available for 730 Mobile Computers)*
	NDIS_RADIO_POWER_MODE_FAST_PSP	Fast PSP, good savings, fast
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	If ERROR_SUCCESS is returned, your ULONG reference is populated with one of the parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetPowerMode)(ULONG &);`
	`#else`
	`UINT GetPowerMode(ULONG &);`
	`#endif`	

Note: Do not use Automatic Switching mode at this time.

GetRSSI()

Call this function to get the current RSSI (Radio Signal Strength Indicator), in Dbm.

Syntax	`UINT **GetRSSI(** ULONG & **);**`
Parameters	References a ULONG that is populated with the current RSSI after a successful call.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	If ERROR_SUCCESS is returned, your ULONG reference contains the RSSI. Valid RSSI range is from –100 Dbm to –30 Dbm.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetRSSI)(ULONG &);`
	`#else`
	`UINT GetRSSI(ULONG &);`
	`#endif`

GetTXPower()

Call this function to get the current transmit power of the radio.

Syntax	`UINT GetTXPower( ULONG & );`	
Parameters	NDIS_POWER_LEVEL_63	63 mW
	NDIS_POWER_LEVEL_30	30 mW
	NDIS_POWER_LEVEL_15	15 mW
	NDIS_POWER_LEVEL_5	5 mW
	NDIS_POWER_LEVEL_1	1 mW
	NDIS_POWER_LEVEL_UNKNOWN	Unknown Value or Error.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	If ERROR_SUCCESS is returned, your ULONG reference is populated with the TX power in milliwatts (mW). Valid ranges are from 5 mW to 100 mW.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetTXPower)(ULONG &);`
	`#else`
	`UINT GetTXPower(ULONG &);`
	`#endif`	

GetWepStatus()

Call this function to get the current state of the radio's WEP and encryption levels.

Syntax	`UINT GetWepStatus( ULONG & );`	
Parameters	NDIS_ENCRYPTION_1_ENABLED	WEP is enabled; TKIP and AES are not enabled, and a transmit key may or may not be available. *(same as NDIS_RADIO_WEP_ENABLED)*
	NDIS_ENCRYPTION_DISABLED	Indicates that AES, TKIP, and WEP are disabled, and a transmit key is available. *(Same as NDIS_RADIO_WEP_DISABLED)*
	NDIS_ENCRYPTION_NOT_SUPPORTED	Indicates encryption (WEP, TKIP, AES) is not supported. *(Same as NDIS_RADIO_WEP_NOT_SUPPORTED)*
	NDIS_ENCRYPTION_1_KEY_ABSENT	Indicates that AES, TKIP, and WEP are disabled, and a transmit key is not available. *(Same as NDIS_RADIO_WEP_ABSENT)*
	NDIS_ENCRYPTION_2_ENABLED	Indicates that TKIP and WEP are enabled; AES is not enabled, and a transmit key is available.
	NDIS_ENCRYPTION_2_KEY_ABSENT	Indicates that there are no transmit keys available for use by TKIP or WEP, TKIP and WEP are enabled; and AES is not enabled.
	NDIS_ENCRYPTION_3_ENABLED	Indicates that AES, TKIP, and WEP are enabled, and a transmit key is available.
	NDIS_ENCRYPTION_3_KEY_ABSENT	Indicates that there are no transmit keys available for use by AES, TKIP, or WEP, and AES, TKIP, and WEP are enabled.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	If ERROR_SUCCESS is returned, your ULONG reference is populated with one of the parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetWepStatus)(ULONG &);`
	`#else`
	`UINT GetWepStatus(ULONG &);`
	`#endif`	

GetRadioIpAddress()

Call this function to obtain a formatted string indicating whether DHCP is enabled, and what is the current adapters IP address.

Syntax	`UINT GetRadioIpAddress( TCHAR * );`
Parameters	Pointer to a character array that contains the formatted string of the IP address and static/DHCP information.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	If ERROR_SUCCESS is returned, your TCHAR array contains a string formatted as follows: `IP: DHCP Enabled\nxxx.xxx.xxx.xxx\n` or `IP: DHCP Disabled\nxxx.xxx.xxx.xxx\n`

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetRadioIpAddress)(TCHAR *);`
	`#else`
	`UINT GetRadioIpAddress(TCHAR *);`
	`#endif`

GetCCXStatus()

Call this function to get information about the current CCX status of the adapter.

Syntax	`UINT GetCCXStatus( ULONG & );`	
Parameters	NDIS_NETWORK_EAP_MODE_OFF	Disable EAP mode.
	NDIS_NETWORK_EAP_MODE_ON	Enable EAP mode.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	If ERROR_SUCCESS is returned, your ULONG reference is populated with one of parameters listed above.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetCCXStatus)(ULONG &);`
	`#else`
	`UINT GetCCXStatus(ULONG &);`
	`#endif`	

Set Information Functions

AddWep()

Call this function to add a WEP key to the radio. Call this function multiple times when adding more than one WEP key. Save the "default" key for last. For example, when adding four keys, and the second key is the default transmit key, add keys 1, 3 and 4 *before* you add key 2.

Note: Add the default transmit key *last*.

Syntax	`UINT` **`AddWep(`** `ULONG, BOOL, TCHAR *` **`);`**	
Parameters	ULONG	Specifies the key index to be set. Valid values are 0–3.
	BOOL	When set to TRUE, specifies that this key is the default transmit key.
	TCHAR	Pointer to a character array that specifies the key data in either HEX (length of 10 or 26) or ASCII (length of 5 or 13). This string must be null-terminated.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	When adding WEP keys to the radio, turn off encryption before you add the keys, then turn encryption back on afterwards. Also, be sure to add the TRANSMIT KEY last.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_AddWep)(ULONG, BOOL, TCHAR *);`
	`#else`
	`UINT AddWep(ULONG, BOOL, TCHAR *);`
	`#endif`	

EnableWep()

Enables or disables WEP encryption on the radio (TRUE/FALSE).

Syntax	`UINT` **`EnableWep(`** `BOOL` **`);`**
Parameters	Set BOOL to TRUE to enable WEP encryption, or FALSE to disable WEP encryption.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.

Remarks	Call this function with TRUE as the parameter to enable WEP encryption. Call this function with the FALSE parameter to disable WEP encryption. This call is an alias for EncryptionStatus(). See the following:
	`EnableWEP(TRUE) = EncryptionStatus(NDIS_ENCRYPTION_1_ENABLED)`
	`EnableWEP(FALSE) = EncryptionStatus(NDIS_ENCRYPTION_DISABLED)`

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_EnableWep)(BOOL);`
	`#else`
	`UINT EnableWep(BOOL);`
	`#endif`

EncryptionStatus()

Call this function to set the desired encryption status.

Syntax	`UINT EncryptionStatus( UINT mode );`	
Parameters	NDIS_ENCRYPTION_1_ENABLED	WEP is enabled; TKIP and AES are not enabled, and a transmit key may or may not be available. *(same as NDIS_RADIO_WEP_ENABLED)*
	NDIS_ENCRYPTION_DISABLED	Indicates that AES, TKIP, and WEP are disabled, and a transmit key is available. *(Same as NDIS_RADIO_WEP_DISABLED)*
	NDIS_ENCRYPTION_NOT_SUPPORTED	Indicates that encryption (WEP, TKIP, and AES) is not supported. *(Same as NDIS_RADIO_WEP_NOT_SUPPORTED)*
	NDIS_ENCRYPTION_1_KEY_ABSENT	Indicates that AES, TKIP, and WEP are disabled, and a transmit key is not available. *(Same as NDIS_RADIO_WEP_ABSENT)*
	NDIS_ENCRYPTION_2_ENABLED	Indicates that TKIP and WEP are enabled; AES is not enabled, and a transmit key is available.
	NDIS_ENCRYPTION_2_KEY_ABSENT	Indicates that there are no transmit keys available for use by TKIP or WEP, TKIP and WEP are enabled; and AES is not enabled.
	NDIS_ENCRYPTION_3_ENABLED	Indicates that AES, TKIP, and WEP are enabled, and a transmit key is available.
	NDIS_ENCRYPTION_3_KEY_ABSENT	Indicates that there are no transmit keys available for use by AES, TKIP, or WEP, and AES, TKIP, and WEP are enabled.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	None.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_EncryptionStatus)(UINT mode);`
	`#else`
	`UINT EncryptionStatus(UINT mode);`
	`#endif`	

SetAuthenticationMode()

Call this function to set the desired authentication mode.

Syntax	`UINT` **`SetAuthenticationMode(`** `ULONG` **`);`**	
Parameters	NDIS_RADIO_AUTH_MODE_OPEN	802.11 Open Authentication. Indicates that the radio is using an open system.
	NDIS_RADIO_AUTH_MODE_SHARED	802.11 Shared Authentication. Indicates that the radio is using a shared key.
	NDIS_RADIO_AUTH_MODE_AUTO	Auto switch between Open/Shared. Indicates automatic detection is used when available.
	NDIS_RADIO_AUTH_MODE_ERROR	Defined as error value. Indicates the authentication mode was not determined at this time or is unknown.
	NDIS_RADIO_AUTH_MODE_WPA	WPA Authentication
	NDIS_RADIO_AUTH_MODE_WPA_PSK	WPA Preshared Key Authentication
	NDIS_RADIO_AUTH_MODE_WPA_NONE	WPA None
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	None.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_SetAuthenticationMode)(ULONG);`
	`#else`
	`UINT SetAuthenticationMode(ULONG);`
	`#endif`	

SetChannel()

This function is currently not implemented. Ad-hoc networks automatically select a channel or use the already existing channel.

Syntax	`UINT` **`SetChannel(`** `USHORT` **`);`**
Parameters	USHORT value that should populate with the desired channel (1–14).
Return Values	None.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_SetChannel)(USHORT);`
	`#else`
	`UINT SetChannel(USHORT);`
	`#endif`

SetNetworkMode()

Call this function to set the desired Network Mode.

Syntax	`UINT SetNetworkMode( ULONG );`	
Parameters	NDIS_NET_MODE_IBSS	802.11 Ad-Hoc Mode.
	NDIS_NET_MODE_ESS	802.11 Infrastructure Mode.
	NDIS_NET_MODE_UNKNOWN	Anything Else/Unknown Error
	NDIS_NET_AUTO_UNKNOWN	Automatic Selection. *Use of this option is not supported or recommended.*
	NDIS_NET_TYPE_OFDM_5G	5 Gigahertz 54 Mbps
	NDIS_NET_TYPE_OFDM_2_4G	802.11 2.4 Gigahertz
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	None.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_SetNetworkMode)(ULONG);`
	`#else`
	`UINT SetNetworkMode(ULONG);`
	`#endif`	

SetPowerMode()

Call this function to set the desired power mode.

Syntax	`UINT SetPowerMode( ULONG mode );`	
Parameters	NDIS_RADIO_POWER_MODE_CAM	Continuous Access Mode *(ie: always on)*.
	NDIS_RADIO_POWER_MODE_PSP	Power Saving Mode.
	NDIS_RADIO_POWER_UNKNOWN	Unknown power mode.
	NDIS_RADIO_POWER_AUTO	Auto. *(Available for 730 Computers)*
	NDIS_RADIO_POWER_MODE_FAST_PSP	Fast PSP, good savings, fast
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	None.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_SetPowerMode)(ULONG mode);`
	`#else`
	`UINT SetPowerMode(ULONG mode);`
	`#endif`	

SetSSID()

Call this function with a pointer to a null-terminated TCHAR array containing the desired SSID to set the desired SSID of the adapter.

Syntax	`UINT SetSSID( TCHAR * );`
Parameters	Pointer to a character array that contains the desired SSID. This should be null-terminated.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	If an "ANY" network is desired, pass in _T("ANY").

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_SetSSID)(TCHAR *);`
	`#else`
	`UINT SetSSID(TCHAR *);`
	`#endif`

SetCCXStatus()

Call this function to set the desired CCX / Network EAP status.

Syntax	`UINT SetCCXStatus( ULONG );`	
Parameters	NDIS_NETWORK_EAP_MODE_OFF	Disable Network EAP / CCX
	NDIS_NETWORK_EAP_MODE_ON	Enable Network EAP / CCX
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	None.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_SetCCXStatus)(ULONG);`
	`#else`
	`UINT SetCCXStatus(ULONG);`
	`#endif`	

SetMixedCellMode()

Call this function to set the desired mixed cell mode.

Syntax	`UINT SetMixedCellMode( ULONG );`	
Parameters	NDIS_MIXED_CELL_OFF	Disable Mixed Cell
	NDIS_MIXED_CELL_ON	Enable Mixed Cell
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.	
Remarks	None.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_SetMixedCellMode)(ULONG);`
	`#else`
	`UINT SetMixedCellMode(ULONG);`
	`#endif`	

RemoveWep()

Call this function with a key index of 0–3 to remove the WEP key at that index.

Syntax	`UINT RemoveWep( ULONG );`
Parameters	ULONG value that specifies the key index to set. Valid values are 0–3.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	On disassociation with all BSSIDs of the current service set, WEP key is removed by the adapter.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_RemoveWEP)(ULONG);`
	`#else`
	`UINT RemoveWEP(ULONG);`
	`#endif`

Helper Functions

ConfigureProfile()

If using the Intermec 802.11 Profile Management system, you can program the API to configure the radio to a specific profile by passing the profile name.

Syntax	`UINT ConfigureProfile( TCHAR * );`
Parameters	Pointer to a character array that contains the profile name. This should be null-terminated.
Return Values	ERROR_SUCCESS when successful, ERR_QUERY_FAILED when the query failed, or ERR_CONNECT_FAILED if a connection with the radio failed.
Remarks	Call this function with a pointer to a null-terminated TCHAR array that contains the name of the profile you wish to configure. This function reads profile data from the profile manager, sets that profile as the default active profile, and configures the radio appropriately. If needed, the supplicant and any other related services are automatically started and stopped.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_ConfigureProfile)(TCHAR *);`
	`#else`
	`UINT ConfigureProfile(TCHAR *);`
	`#endif`

EnableZeroConfig()

This enables or disables the Wireless Zero Configuration Wizard from Microsoft. After calling this function, a warm-boot is required for the change to take effect.

Note: Enabling this function effectively disables all the SET commands in this API.

Syntax	`UINT` **`EnableZeroConfig(`** `USHORT` **`);`**	
Parameters	TRUE	Enable Wireless Zero Config
	FALSE	Disable Wireless Zero Config
Return Values	ERROR_SUCCESS when successful, ERR_ZERO_CONFIG_CHANGE_FAILED when the query failed.	
Remarks	Call this function to set the desired Zero Config status.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_EnableZeroConfig)(USHORT);`
	`#else`
	`UINT EnableZeroConfig(USHORT);`
	`#endif`	

isZeroConfigEnabled()

Call this function to determine whether Zero Config is currently enabled.

Syntax	`UINT` **`isZeroConfigEnabled( );`**
Parameters	None.
Return Values	TRUE if ZeroConfig is enabled, and FALSE if it is disabled.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_isZeroConfigEnabled)();`
	`#else`
	`UINT isZeroConfigEnabled();`
	`#endif`

isOrinoco()

Call this function to determine whether the current radio is an ORiNOCO, Lucent, or WaveLAN radio.

Syntax	`UINT` **`isOrinoco( );`**
Parameters	None.
Return Values	TRUE if this is an ORiNOCO radio, and FALSE if it is not.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_isOrinoco)();`
	`#else`
	`UINT isOrinoco();`
	`#endif`

isSupplicantRunning()

Call this function to determine whether the security supplicant is running.

Syntax	`UINT` **`isSupplicantRunning( );`**
Parameters	None.
Return Values	TRUE if the security supplicant is running, FALSE if it is not running.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_isSupplicantRunning)();`
	`#else`
	`UINT isSupplicantRunning();`
	`#endif`

StartScanList()

If a scan list is configured on the system, this causes the API to begin the process of scanning for an available network. This call can take quite a while to process *(depending upon the length of the scan list and how long it takes to find a valid network)*, you may wish to call it from a separate thread.

Syntax	`UINT` **`StartScanList( );`**
Parameters	None.
Return Values	ERROR_SUCCESS when successful.
Remarks	Call this function to start the scan list functionality of the system.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_StartScanList)();`
	`#else`
	`UINT StartScanList();`
	`#endif`

StartSupplicant()

Call this function to start the supplicant service if it is installed on the system.

Syntax	`UINT` **`StartSupplicant( );`**
Parameters	None.
Return Values	ERROR_SUCCESS when successful.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_StartSupplicant)();`
	`#else`
	`UINT StartSupplicant();`
	`#endif`

StopSupplicant()

Call this function to stop the supplicant service.

Syntax	`UINT` **`StopSupplicant( );`**
Parameters	None.
Return Values	ERROR_SUCCESS when successful.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_StopSupplicant)();`
	`#else`
	`UINT StopSupplicant();`
	`#endif`

isDHCPEnabled()

Call this function to determine whether DHCP is enabled on the current adapter.

Syntax	`UINT` **`isDHCPEnabled( );`**
Parameters	None.
Return Values	TRUE if DHCP is enabled, FALSE if it is not.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_isDHCPEnabled)();`
	`#else`
	`UINT isDHCPEnabled();`
	`#endif`

RenewDHCP()

Call this function to force a DHCP renewal on the current network adapter.

Syntax	`UINT` **`RenewDHCP( );`**
Parameters	None.
Return Values	ERROR_SUCCESS when successful.
Remarks	You should not have to call this function on Microsoft PocketPC 2003 or Microsoft Windows CE 4.2 .NET and later devices.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_RenewDHCP)();`
	`#else`
	`UINT RenewDHCP();`
	`#endif`

GetCurrentDriverName()

Call this function to populate the TCHAR array with the driver name.

Syntax	`UINT` **`GetCurrentDriverName( TCHAR * );`**
Parameters	Pointer to a TCHAR array which contains the name of the driver when successful.
Return Values	ERROR_SUCCESS when successful.
Remarks	This function is called with a pointer to a TCHAR array that is large enough to hold the name of the driver PLUS the null terminator.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_GetCurrentDriverName)(TCHAR *);`
	`#else`
	`UINT GetCurrentDriverName(TCHAR *);`
	`#endif`

ResetRadioToSystemSave()

Call this function to force the radio to reset to the last desired active profile.

Syntax	`UINT` **`ResetRadioToSystemSave( );`**
Parameters	None.
Return Values	ERROR_SUCCESS when successful.
Remarks	None.

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_ResetRadioToSystemSave)();`
	`#else`
	`UINT ResetRadioToSystemSave();`
	`#endif`

EnableSuppLogging()

Call this function to set the desired supplicant logging mode.

Syntax	`UINT` **`EnableSuppLogging( ULONG );`**	
Parameters	NDIS_SUPP_LOGGING_ON	Supplicant Logging Enabled
	NDIS_SUPP_LOGGING_OFF	Supplicant Logging Disabled
Return Values	ERROR_SUCCESS when successful.	
Remarks	None.	

Definitions	`#ifdef DYNAMIC_LOADING`
	`typedef UINT (*PFN_EnableSuppLogging)(ULONG);`
	`#else`
	`UINT EnableSuppLogging(ULONG);`
	`#endif`	

SwitchPacketDriver()

Call this function to switch between available packet drivers on the system.

Syntax	UINT **SwitchPacketDriver(USHORT);**	
Parameters	INTERMEC_PACKET_DRIVER	Intermec Packet Driver (ZNICZIO)
	NDISUIO_PACKET_DRIVER	Microsoft Packet Driver (NDISUIO)
Return Values	ERROR_SUCCESS when successful.	
Remarks	After switching to a new packet driver, perform a warm boot for changes to take effect.	
Definitions	#ifdef DYNAMIC_LOADING typedef UINT (*PFN_SwitchPacketDriver)(USHORT); #else UINT SwitchPacketDriver(USHORT); #endif	

Deprecated Functions

The following functions are deprecated. While these are not removed from the API, these are no longer supported. Their parameters are no longer applicable and the return value for all of these functions is: ERR_FUNCTION_DEPRECATED

Function	Syntax
GetRTSThreshold(Deprecated)	UINT **GetRTSThreshold(** USHORT *&* **);**
GetMedia(Deprecated)	UINT **GetMedia(** ULONG *&* **);**
GetMedium(Deprecated)	UINT **GetMedium(** ULONG *&* **);**
GetNicStats(Deprecated)	UINT **GetNicStats(** NDIS_802_11_STATISTICS *&* **);**
SetRTSThreshold(Deprecated)	UINT **SetRTSThreshold(** USHORT *&* **);**
SetTXRate(Deprecated)	UINT **SetTXRate(** UCHAR **);**
EncryptWepKeyForRegistry(Deprecated)	UINT **EncryptWepKeyForRegistry(** TCHAR * *szDest*, TCHAR * *szSource* **);**
SetDiversity(Deprecicated)	UINT **SetDiversity(** USHORT **);**

Notifications

Use the following information to programmatically control the vibrator, to write an application to turn on the vibrator when a message is received via the WLAN radio link, and turn it off when the user hits a key.

Vibrator support is implemented in the NLED driver as a false LED. The vibrator is LED 5 and is identified with an CycleAdjust of –1. The vibrate option is only available in the notifications panel when the vibrator is present in the system.

Regarding an applications interface to NLED.DLL, LEDs must be available for use by applications. This is possible via two functions exported by the COREDLL.DLL file. To use the LED functions, declare these as extern ”C” as follows:

```
extern ”C” BOOL WINAPI NLEDGetDeviceInfo(UINT nInfoId,
void *pOutput);
extern ”C” BOOL WINAPI NLEDSetDevice( UINT nDeviceId, void
*pInput);
```

The LEDs are enumerated for access through the data structures associated with these APIs:

• Notification LED	0
• Radio On LED	1 *(does not apply to the 730 Computer)*
• Alpha Lock LED	2
• Scanner LED	3
• Low Battery	4
• Vibrator	5 *(does not apply to the 730 Computer)*

NLEDGetDeviceInfo

Usage

#include “nled.h”

Syntax

```
BOOL NLEDGetDeviceInfo ( UINT nInfoId, void *pOutput );
```

Parameters

nInfoId	Integer specifying the information to return. These values are defined:	
	NLED_COUNT_INFO	Indicates the *pOutput* buffer specifies the number of LEDs on the device.
	NLED_SUPPORTS_INFO_ID	Indicates the *pOutput* buffer specifies information about the capabilities supported by the LED.
	NLED_SETTINGS_INFO_ID	Indicates the *pOutput* buffer contains information about the LED current settings.
pOutput	Pointer to the buffer to which the information is returned. The buffer points to various structure types defined in “nled.h”, depending on the value of *nId*, as detailed in the following table:	
	Value of *nID*	**Structure in *pOutput***
	LED_COUNT_INFO	NLED_COUNT_INFO
	NLED_SUPPORTS_INFO	NLED_SUPPORTS_INFO
	NLED_SETTINGS_INFO	NLED_SETTINGS_INFO

NLEDSetDevice

Usage

#include “nled.h”

Syntax

```
BOOL NLEDSetDevice ( UINT nDeviceId, void *pInput );
```

Parameters

nDeviceId	Integer specifying the device identification. The following is defined:	
	NLED_SETTINGS_INFO_ID	Contains information about the desired LED settings.
pInput	Pointer to the buffer that contains the NLED_SETTINGS_INFO structure.	

Reboot Functions

There are several methods, via Kernel I/O Control functions, that an application program can use to force the 700 Color Computer to reboot.

IOCTL_HAL_REBOOT

IOCTL_HAL_REBOOT performs a warm-boot. See page 263.

IOCTL_HAL_COLDBOOT

Invoking the KernelIOControl function with IOCTL_HAL_COLDBOOT forces a cold reboot. This resets the 700 Color Computer and reloads Windows CE as if a power-up was performed. The contents of the Windows CE RAM-based object store are discarded. See page 260.

IOCTL_HAL_WARMBOOT

This function is supported on 700 Color Computers. It performs a warm boot of the system, preserving the object store. See page 260.

Remapping the Keypad

Note: Use caution when remapping the keypad. Improper remapping may render the keypad unusable. Data within the 700 Color Computer could also be lost, should any problems occur.

Applications have the ability to remap keys on the 700 Color Numeric Keypad and 700 Color Alphanumeric Keypad. This will allow applications to enable keys that would otherwise not be available, such as the [F1] function key. Also, to disable keys that should not be available, such as the alpha key because no alpha entry is required. Care should be exercised when attempting to remap the keypad because improper remapping may cause the keypad to become unusable. This can be corrected by cold booting the device which will cause the default keymap to be loaded again.

Note that remapping the keys in this way affects the key mapping for the entire system, not just for the application that does the remapping.

There are three "planes" supported for the 700 Color Numeric Keypad and Alphanumeric Keypad. Keys that are to be used in more than one shift plane must be described in each plane.

Unshifted Plane

The unshifted plane contains values from the keypad when not pressed with other keys, such as the following:

Press the Keys		
Numeric Keypad	**Alphanumeric Keypad**	**To Enter This**
1 Caps Send	M1	1
5 JKL A3	T5	5
9 WXYZ	Y9	9

Gold Plane

The gold plane contains values from the keypad when a key is simultaneously pressed with the [**Gold**] ○ key on the numeric keypad or the [**Gold/White**] ● key on the alphanumeric keypad, such as the following:

Press the Keys		
Numeric Keypad	**Alphanumeric Keypad**	**To Enter This**
[Gold] ○ 1 Caps Send	[Gold/White] ● Esc Send	Send
[Gold] ○ 5 JKL A3	[Gold/White] ● C A3	A3
[Gold] ○ 9 WXYZ	[Gold/White] ● P PgDn	PgDn

Alpha (Blue) Plane

The alpha plane contains values from the keypad when the keypad has been placed in alpha mode by pressing the blue alpha key, such as the following:

Press the Keys		
Numeric Keypad	**Alphanumeric Keypad**	**To Enter This**
[Alpha] (Alpha) (1 Caps Send)	[Alpha] (Alpha) (Cap Lock)	Caps
[Alpha] (Alpha) (5 JKL A3)	[Alpha] (Alpha) (J PgUp)	j
[Alpha] (Alpha) (9 WXYZ)	[Alpha] (Alpha) (W7)	w

Key Values

Key values for each plane are stored in the registry. All units ship with a default key mapping already loaded in the registry. Applications that wish to change the default mapping need to read the appropriate key from the registry into an array of Words, modify the values required and then write the updated values back into the registry. The registry access can be done with standard Microsoft API calls, such as RegOpenKeyEx(), RegQueryValueEx(), and RegSetValueEx().

Numeric Keypad

For the 700 Color Numeric Keypad, the following registry keys contain the plane mappings:

- The unshifted plane mapping can be found in the registry at:
  `HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\KEYBD\Vkey`
- The gold plane mapping can be found in the registry at:
  `HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\KEYBD\VkeyGold`
- The alpha plane mapping can be found in the registry at:
  `HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\KEYBD\VkeyAlpha`

Alphanumeric Keypad

For the 700 Color Alphanumeric Keypad, the following registry keys contain the plane mappings:

- The unshifted plane mapping can be found in the registry at:
  `HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\KEYBD\ALPHA\Vkey`
- The gold plane mapping can be found in the registry at:
  `HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\KEYBD\ALPHA\VkeyGold`
- The alpha plane mapping can be found in the registry at:
  `HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\KEYBD\ALPHA\VkeyAlpha`

How Key Values Are Stored in Registry

To know which fields to update in the registry, you must know what Scan Codes are assigned to each physical key (see the *"Keypad Scan Codes and Meanings"* table on the next page). The Scan Code is used at the lowest level of the system to let the keypad driver know which physical key has been pressed. The keypad driver takes that scan code and looks it up in a table (a copy of the one stored in the registry) to determine which values to pass on to the operating system.

Each registry key is just an array that describes to the keypad driver what value needs to be passed for each physical key. The key values are indexed by the scan code, this is a zero-based index. For example in the unshifted plane, the [4] key has a scan code of 0x06. This means that the seventh word under the "Vkey" registry key will have the value for the [4] key. Taking a sample of the "Vkey" registry key shows the following values:

```
00,00,0B,05,02,03,C1,07,04,03,BE,00,34,00,00,00,. . .
```

The value is 34,00. The values are in reverse byte order because that is the way the processor handles data. When writing an application, nothing needs to be done to swap the bytes, as this will happen automatically when the data is read into a byte value. This is something you just need to be aware of when looking at the registry. Knowing this, we can see that the value that the keypad driver will pass to the system is a hex 34. Looking that up on an UNICODE character chart, we see that it maps to a "4". If you wanted the key, labeled "4", to output the letter "A" instead, you would need to change the seventh word to "41" (the hexadecimal representation of "A" from the UNICODE chart), then put the key back into the registry.

Note: Do not remap scan codes 0x01, 0x41, 0x42, 0x43, 0x44. Remapping these scan codes could render your 700 Color Computer unusable until a cold-boot is performed.

If you wish to disable a certain key, remap its scan code to 0x00.

Change Notification

Just changing the registry keys will not immediately change the key mappings. To notify the keypad driver that the registry has been updated, signal the "ITC_KEYBOARD_CHANGE" named event using the CreateEvent() API.

Advanced Keypad Remapping

It is also possible to map multiple key presses to one button and to map named system events to a button. The multiple key press option could be useful to cut down on the number of keys needed to press in a given situation or to remap which key behaves like the action key. Mapping events to a button could be useful to change which buttons will fire the scanner, control volume, and allow for suspending and resuming the device. If you need help performing one of these advanced topics please contact Intermec Technical Support.

Scan Codes

At the lowest driver level, the 700 Color Numeric Keypad and the 700 Color Alphanumeric Keypad identifies keys as scan codes. These scan codes are sent via the keypad microcontroller, and cannot be changed without modifying the keypad firmware.

Numeric Keypad

The following scan codes pertain to the 700 Color Numeric keypad:

Numeric Keypad Scan Codes and Meanings

Press this Key	Meaning	ScanCode
	Reserved	0x00
I/O	I/O button	0x01
	Scanner Handle Trigger	0x02
	Scanner Left	0x03
	Scanner Right	0x04
4 GHI A2	4/GHI/A2	0x06
	None	0x07
◁	Left arrow/Back Tab	0x08
	None	0x09
BkSp /	BkSp// (forward slash)	0x0A
○	[Gold] key	0x0B
	None	0x0C
Esc −	Esc/– (minus sign)	0x0D
▼	Down arrow/Volume decrease	0x0E
1 Caps Send	1/Caps/Send	0x0F
7 PQRS PgUp	7/PQRS/PgUp	0x10
Alpha	[Alpha] key	0x11
	None	0x12
▲	Up arrow/Volume increase	0x13
▷	Right arrow/Tab	0x14
2 ABC End	2/ABC/End	0x15
8 TUV	8/TUV/* (asterisk)	0x16
0 Win	0/Win	0x17
5 JKL A3	5/JKL/A3	0x18
	None	0x19
Action +	Action/+ (plus symbol)	0x1A
3 DEF ☀	3/DEF/backlight	0x1B
9 WXYZ	9/WXYZ/PgDn	0x1C

Numeric Keypad Scan Codes and Meanings (continued)

Press this Key	Meaning	ScanCode
Enter @	Enter/@ (at symbol)	0x1D
6 MNO A4	6/MNO/A4	0x1E
	None	0x1F–0x40
◎▯	Charge Detect	0x41
✓◎	LCD frontlight	0x42
◎	Ambient light	0x42
	Threshold crossed	0x42
	Headset detected	0x43
	Keypad Backlight	0x44
◎	Ambient Light	0x44
	Threshold Crossed	0x44

Alphanumeric Keypad

The following scan codes pertain to the 700 Color Alphanumeric keypad:

Alphanumeric Keypad Scan Codes and Meanings

Press this Key	Meaning	ScanCode
	Reserved	0x00
I/O	I/O button	0x01
	Scanner Handle Trigger	0x02
	Scanner Left	0x03
	Scanner Right	0x04
A A1	A/A1 key	0x05
B A2	B/A2 key	0x06
Esc Send	Escape/Send	0x07
◀	Left arrow/Back Tab	0x08
▲ ♪+	Up arrow/Volume increase	0x09
♪- ▼	Down arrow/Volume decrease	0x0A
▶	Right arrow/Tab	0x0B
Action End	Action/End	0x0C
E Win	E/Win	0x0D
F =	F/= (equal sign)	0x0E
G ★	G/* (asterisk)	0x0F
C A3	C/A3	0x10
H /	H// (forward slash)	0x11
D A4	D/A4	0x12

Alphanumeric Keypad Scan Codes and Meanings (continued)

Press this Key	Meaning	ScanCode
J PgUp	J/PgUp	0x13
K @	K/@ (as symbol)	0x14
L -	L/– (minus sign)	0x15
M 1	M/1	0x16
N 2	N/2	0x17
I ☼	I/backlight	0x18
P PgDn	P/PgDn	0x19
Q ,	Q/, (comma)	0x1A
R +	R/+ (plus sign)	0x1B
S 4	S/4	0x1C
T 5	T/5	0x1D
O 3	O/3	0x1E
Cap Lock	Caps/Lock	0x1F
BkSp ←	BkSp	0x20
V •	V/. (period)	0x21
W 7	W/7	0x22
X 8	X/8	0x23
U 6	U/6	0x24
⊘	Gold/White	0x25
	NumLock	0x26
Space	Space	0x27
Z 0	Z/0	0x28
Enter	Enter	0x29
Y 9	Y/9	0x2A
	None	0x2B–0x40
◎ ▯	Charge Detect	0x41
✓◎	LCD frontlight	0x42
◎	Ambient light	0x42
	Threshold crossed	0x42
	Headset detected	0x43
	Keypad Backlight	0x44
◎	Ambient Light	0x44
	Threshold Crossed	0x44

Sample View of Registry Keys

The following is a sample view of the current default key mapping for the 700 Color Numeric Keypad. See the registry on your device for the latest key mappings.

```
[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\KEYBD]
”ResumeMask”=dword:7
”Vkey”=hex:00,00,0B,05,02,03,C1,07,04,03,BE,00,34,00,00,00,\
  25,00,00,00,08,00,03,02,00,00,1B,00,28,00,31,00,\
  37,00,01,02,00,00,26,00,27,00,32,00,38,00,30,00,\
  35,00,00,00,01,03,33,00,39,00,0D,00,36,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,07,05,01,05,03,05,02,05

”VkeyGold”=hex: 00,00,0B,05,02,03,C1,07,04,03,BE,00,34,00,00,00,\
  09,01,00,00,BF,00,03,02,00,00,BD,00,75,00,72,00,\
  21,00,01,02,00,00,76,00,09,00,73,00,38,01,5B,00,\
  35,00,00,00,BB,01,09,05,22,00,32,01,36,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,07,05,01,05,03,05,02,05

”VkeyAlpha”=hex:  00,00,0B,05,02,03,C1,07,04,03,BE,00,47,00,00,00,\
  25,00,00,00,08,00,03,02,00,00,1B,00,28,00,02,02,\
  50,00,01,02,00,00,26,00,27,00,41,00,54,00,20,00,\
  4A,00,00,00,01,03,44,00,57,00,0D,00,4D,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
  00,00,07,05,01,05,03,05,02,05
```

A Configurable Settings

This appendix contains information about the Data Collection, Intermec Settings, SNMP, Unit Information, Utilities, and Wireless Network control panel applets that may be on the 700 Series Color Mobile Computer.

Note: “700 Color” pertains to 740, 741, 750, 751, 760, and 761 Computers unless otherwise noted.

SNMP, Intermec Settings, and Data Collection settings that can appear under **Settings** are dependent on what hardware configuration is done for each 700 Color Computer at the time of shipment. These settings currently only appear if a scanner or an imager option is present.

Likewise, other control panel applets that are specifically related to the 802.11b or 802.11b/g radio module will appear when a 802.11b or 802.11b/g radio module is installed in a 700 Color Computer. Control panel applets that are specific for Wireless Printing, CDMA/1xRTT, and GSM/GPRS radio modules will only appear when each respective hardware configuration is done on the 700 Color Computer. *See Chapter 4, “Network Support,” for more information about the radio modules or the wireless printing.*

Information about using reader commands and configuration bar codes to configure some of your settings is also in this appendix.

Note: Information about the settings you can configure with the Intermec Settings control panel applet is described in the *Intermec Computer Command Reference Manual* (P/N: 073529). The online manual is available from the Intermec web site at www.intermec.com.

Configuration Parameters

A configuration parameter changes the way the 700 Color Computer operates, such as configuring a parameter to have the 700 Color Computer emit a very loud beep in a noisy environment. Use any of the following methods to execute configuration parameters:

- Change Data Collection and SNMP parameters via control panel applets later in this appendix.
- Send parameters from an SNMP management station. See "*SNMP Configuration on the 700 Color Computer*" starting on page 195.
- Scan EasySet bar codes. You can use the EasySet bar code creation software from Intermec Technologies Corporation to print configuration labels. Scan the labels to change the scanner configuration and data transfer settings.

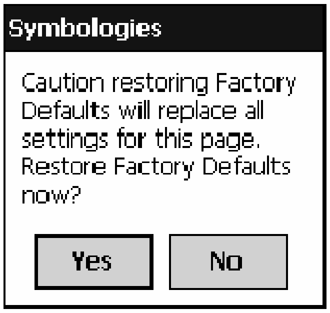
  Use the Intermec EasySet software to print configuration labels you can scan to change your configuration settings. For more information, see the EasySet online help. EasySet is available from the Intermec Data Capture web site.

Changing a Parameter Setting

Menus of available parameters for each group are listed. Use the scroll bars to go through the list. Expand each menu (+) to view its parameter settings. Tap a parameter to select, or expand a parameter (+) to view its subparameters.

Note that each parameter or subparameter is shown with its default setting or current setting in (< >) brackets. Tap a parameter or subparameter to select that parameter, then do any of the following to change its setting: Tap **Apply** to apply any changes. *Note that these illustrations are from a Symbologies parameter.*

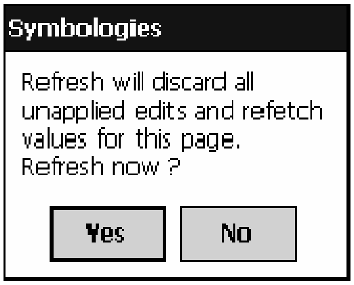
- Typing a new value in an entry field.
- Choosing a new value from the drop-down list.
- Selecting a different option. The selected option contains a bullet.
- Tap **Defaults**, then **Apply** to restore factory-default settings. Tap **Yes** when you are prompted to verify this action.



Symbologies

Caution restoring Factory Defaults will replace all settings for this page. Restore Factory Defaults now?

- Tap **Refresh** to discard changes and start again. Tap **Yes** when you are prompted to verify this action.



Symbologies

Refresh will discard all unapplied edits and refetch values for this page. Refresh now ?

About Configuration Parameters

You can find this information about each configuration parameter:

- **Name and Purpose**:
  Describes the parameter and its function.
- **Action**:
  Describes what to do with a parameter once that parameter is selected.
- **SNMP OID**:
  Lists the SNMP OID for the parameter.
- **Syntax** or **Options**:
  Syntax lists the two-character code for the parameter, if the parameter is configurable by scanning a bar code or by sending parameters through a network. Both **Syntax** and **Options** list acceptable values for the parameter.

Data Collection Control Panel Applet

Note: This applet is not available in units with PSM Build 3.00 or newer. To determine your PSM Build version, tap **Start** > **Programs** > **File Explorer** > the **PSMinfo** text file.

Intermec Settings

If your unit has PSM Build 3.00 or newer, then you may have the Intermec Settings control panel applet in place of the Data Collection applet. Information about the settings you can configure with the Intermec Settings applet is described in the *Intermec Computer Command Reference Manual*. The online manual is available from the Intermec web site at www.intermec.com.

See the Data Collection Resource Kit in the Intermec Developer Library (IDL) for information about data collection functions. Contact your Intermec representative for more information.

Data Collection

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Data Collection** to access its control panel applet.

Use the left and right arrows to scroll through the tabs along the bottom of the control panel applet, then tap a tab to access its menus. These tabs represent the following groups of settings or parameters:

- **Symbologies** *(starting on page 303)*
- **Symbology Options** *(starting on page 324)*
- **Beeper/LED** *(starting on page 332)*
- **Imager** *(starting on page 338)*
- **Virtual Wedge** *(starting on page 343)*

Symbologies

You can change bar code symbology parameter settings in your 700 Color Computer via the **Data Collection** control panel applet. The following parameters are for bar code symbologies. Additional information about the more common bar code symbologies are in Appendix B, "*Bar Codes*." *Note that these parameters are listed in the order of their appearance within this tab.*

Most of these symbologies apply to both the imager and the laser scanner tools. However, when using an imager, the Macro PDF *(page 314)*, Micro PDF417 *(page 316)*, Matrix 2 of 5 *(page 318)*, Telepen *(page 319)*, and Code 11 *(page 320)* symbologies are not supported. Likewise, when using a laser scanner, the QR Code *(page 321)*, Data Matrix *(page 322)*, and MaxiCode *(page 323)* symbologies are not supported.

Note: The 730 Computer uses the EV10 APS linear imager which supports 1D symbologies.

The following table shows which bar code symbologies are supported by an imager, a laser scanner, or the EV10 APS Linear Imager

Bar Code Symbology	Imager	Laser Scanner	EV10 APS Linear Imager
Code 39	X	X	X
Interleaved 2 of 5	X	X	X
Standard 2 of 5	X	X	X
Matrix 2 of 5		X	X
Code 128	X	X	X
Code 93	X	X	X
Codabar	X	X	X
MSI		X	X
Plessey		X	X
UPC	X	X	X
EAN/EAN 128	X	X	X
Code 11		X	X
PDF417	X	X	X
Micro PDF417		X	X
Telepen		X	X
Data Matrix	X		
QR Code	X		
MaxiCode	X		

Code 39

Code 39 is a discrete, self-checking, variable length symbology. The character set is uppercase A–Z, 0–9, dollar sign ($), period (.), slash (/), percent (%), space ( ), plus (+), and minus (-).

Action

Tap (+) to expand the **Code 39** parameter, select the setting to be changed, then tap an option to change this setting or select an option from the drop-down list.

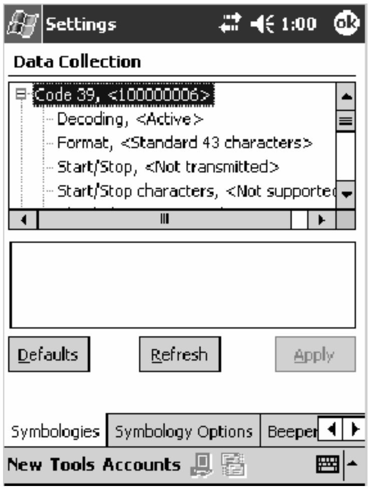
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.3.1

Options

Decoding	0 1	Not active Active (default)
Format	0 1	Standard 43 characters (default) Full ASCII
Start/Stop	0 1	Not transmitted (default) Transmitted
Start/Stop characters *(Not supported when using an imager)*	0 1 2	$ (dollar sign) only * (asterisk) only (default) $ and * (dollar sign and asterisk)
Check digit	0 1 2 3 4 5 6	Not used (default) Mod 43 transmitted Mod 43 not transmitted French CIP transmitted French CIP not transmitted Italian CPI transmitted Italian CPI not transmitted
Bar code length	0 1	Any length (default) Minimum length
Minimum length	001–254	Minimum length 1–254 (default is 6)

Note: If **Bar code length** = “1” then **Minimum length** is entered.



Data Collection

Code 39, <100000006>
Decoding, <Active>
Format, <Standard 43 characters>
Start/Stop, <Not transmitted>
Start/Stop characters, <Not supporte

Standard 2 of 5

Standard 2 of 5 is a discrete and self-checking symbology that uses the bars to encode information and the spaces to separate the individual bars.

Action

Tap (+) to expand the **Standard 2 of 5** parameter, select the setting to be changed, then tap an option to change this setting or select an option from the drop-down list.

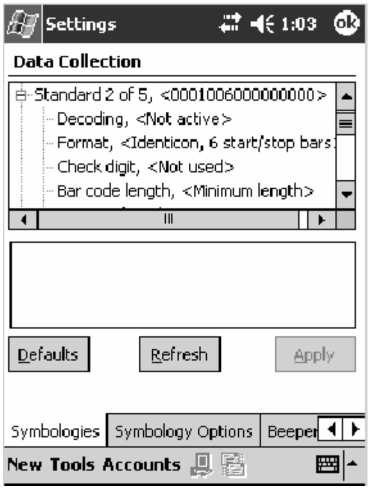
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.4.1

Options

Decoding	0 1	Not active (default) Active
Format	0 1	Identicon, 6 start/stop bars (default) Computer Identics, 4 start/stop bars
Check digit	0 1 2	Not used (default) Mod 10 transmitted Mod 10 not transmitted
Bar code length	0 1 2	Any length Minimum length (default) Fixed lengths
Minimum length	001–254	Minimum length 1–254 (default is 6)
Fixed length 1	000–254	Fixed bar code length 0–254 (default is 0)
Fixed length 2	000–254	Fixed bar code length 0–254 (default is 0)
Fixed length 3	000–254	Fixed bar code length 0–254 (default is 0)

Note: If **Bar code length** = "1" then **Minimum length** is entered. If **Bar code length** = "2" then **Fixed length 1**, **Fixed length 2**, or **Fixed length 3** is entered.



Data Collection

Standard 2 of 5, <0001006000000000>
Decoding, <Not active>
Format, <Identicon, 6 start/stop bars
Check digit, <Not used>
Bar code length, <Minimum length>

Codabar

Codabar is a self-checking, discrete symbology.

Action

Tap (+) to expand the **Codabar** parameter, select a setting to be changed, then select an option from the drop-down list to change this setting.

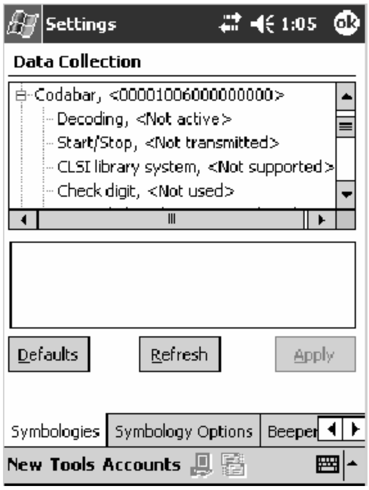
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.5.1

Options

Decoding	0 1	Not active (default) Active
Start/Stop	0 1 2 3 4	Not transmitted (default) abcd transmitted ABCD transmitted abcd/tn*e transmitted DC1–DC4 transmitted
CLSI library system *(Not supported when using an imager)*	0 1	Not active (default) Active
Check digit	0 1 2	Not used (default) Transmitted Not transmitted
Bar code length	0 1 2	Any length Minimum length (default) Fixed lengths
Minimum length	003–254	Minimum length 3–254 (default is 6)
Fixed length 1	000–254	Fixed bar code length 0–254 (default is 0)
Fixed length 2	000–254	Fixed bar code length 0–254 (default is 0)
Fixed length 3	000–254	Fixed bar code length 0–254 (default is 0)

Note: If **Bar code length** = "1" then **Minimum length** is entered. If **Bar code length** = "2" then **Fixed length 1**, **Fixed length 2**, or **Fixed length 3** is entered.



Data Collection

Codabar, <00001006000000000>
Decoding, <Not active>
Start/Stop, <Not transmitted>
CLSI library system, <Not supported>
Check digit, <Not used>

UPC/EAN

UPC/EAN are fixed-length, numeric, continuous symbologies that use four element widths.

Action

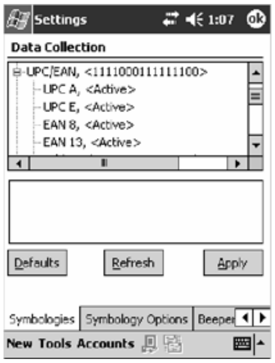
Tap (+) to expand the **UPC/EAN** parameter, select the setting to be changed, then select an option to change this setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.6.1

Options

UPC A	0 1	Not Active Active (default)
UPC E	0 1	Not Active Active (default)
EAN 8	0 1	Not Active Active (default)
EAN 13	0 1	Not Active Active (default)
Add-on digits	0 1	Not required (default) Required
Add-on 2 digits	0 1	Not active (default) Active
Add-on 5 digits *(Not supported when using an imager)*	0 1	Not active (default) Active
UPC A check digit	0 1	Not transmitted Transmitted (default)
UPC E check digit	0 1	Not transmitted Transmitted (default)
EAN 8 check digit	0 1	Not transmitted Transmitted (default)
EAN 13 check digit	0 1	Not transmitted Transmitted (default)
UPC A number system	0 1	Not transmitted Transmitted (default)
UPC E number system	0 1	Not transmitted Transmitted (default)
UPC A re-encoding	0 1	UPC A transmitted as UPC A UPC A transmitted as EAN 13 (default)
UPC E re-encoding	0 1	UPC E transmitted as UPC E (default) UPC E transmitted as UPC A
EAN 8 re-encoding	0 1	EAN 8 transmitted as EAN 8 (default) EAN 8 transmitted as EAN 13



UPC/EAN, <1111000111111100>
UPC A, <Active>
UPC E, <Active>
EAN 8, <Active>
EAN 13, <Active>

Code 93

Code 93 is a variable length, continuous symbology that uses four element widths.

Action

Tap the **Code 93** parameter, then select an option to change this parameter setting. Tap (+) to access the **Code 93 Lengths** parameter.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.7.1

Options

0	Not active (default)
1	Active

Code 93 Length

Sets the Code 93 bar code length.

Action

Tap (+) to expand the **Code 93** parameter, then tap (+) to expand the **Code 93 Lengths** parameter. Tap the setting to be changed, then tap an option to change this setting.

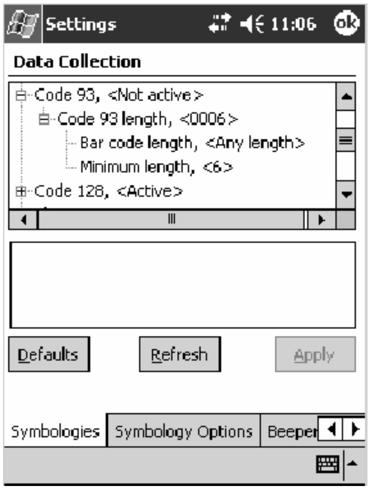
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.19.1

Options

Bar code length	0 1	Any length Minimum length (default)
Minimum length	001–254	Minimum length 1–254 (default is 6)

Note: If **Bar code length** = “1” then **Minimum length** is entered.



Code 93, <Not active>
Code 93 length, <0006>
Bar code length, <Any length>
Minimum length, <6>
Code 128, <Active>

Code 128

Code 128 is a variable-length, continuous, high-density, alphanumeric symbology that uses multiple element widths and supports the extended ASCII character set.

Action

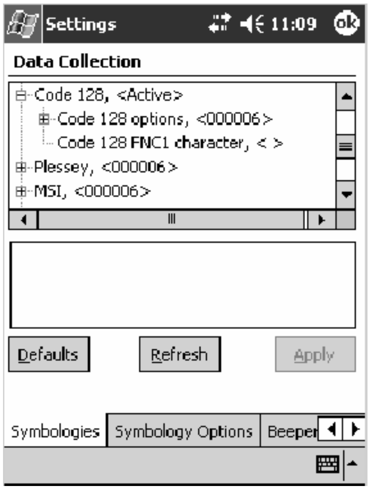
Tap the **Code 128** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.9.1

Options

0 Not active (default)
1 Active



Data Collection

Code 128, <Active>
Code 128 options, <000006>
Code 128 FNC1 character, < >
Plessey, <000006>
MSI, <000006>

This illustration is from a 700 Color Computer using a laser scanner.

Code 128 Options

Set the following for the Code 128 parameter. *Note that the **EAN 128]C1** and **CIP 128 French Pharmaceutical** options are not available when you use an imager with your 700 Color Computer.*

Action

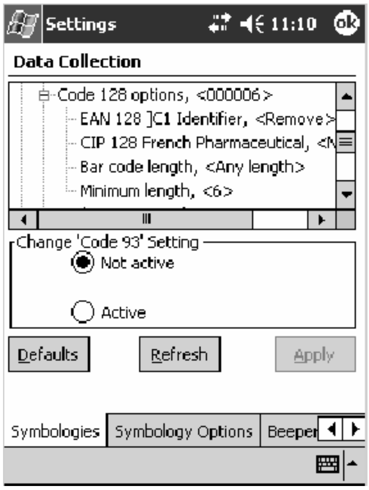
Tap (+) to expand the **Code 128 Options** parameter, select a setting, then select an option to change this setting.

SNMP OID

None.

Options

EAN 128]C1 Identifier *(Not supported when using an imager)*	0 1	Remove (default) Include
CIP 128 French Pharmaceutical *(Not supported when using an imager)*	0 1	Not active (default) Active
Bar code length	0 1	Any length (default) Minimum length
Minimum length	001–254	Minimum length 1–254 (default is 6)



Data Collection

Code 128 options, <000006>
EAN 128]C1 Identifier, <Remove>
CIP 128 French Pharmaceutical, <N
Bar code length, <Any length>
Minimum length, <6>

Change 'Code 93' Setting
● Not active
○ Active

This illustration is from a 700 Color Computer using a laser scanner.

Code 128 FNC1 Character

The Code 128 FNC1 character (EAN 128 norms) can be any ASCII character and is used as a separator when multiple identifiers and their fields are concatenated. *Note that this is not available when you use an imager with your 700 Color Computer.*

Non-printable ASCII characters can be entered using the following syntax where *HH* is the hexadecimal value of the character.

```
\xHH
```

For example, the GS character, whose hexadecimal value is 1D, would be entered as `\x1D`. In addition,the following characters have their own identifiers:

• BEL	\a
• BS	\b
• FF	\f
• LF	\n
• CR	\r
• HT	\t
• VT	\v

Action

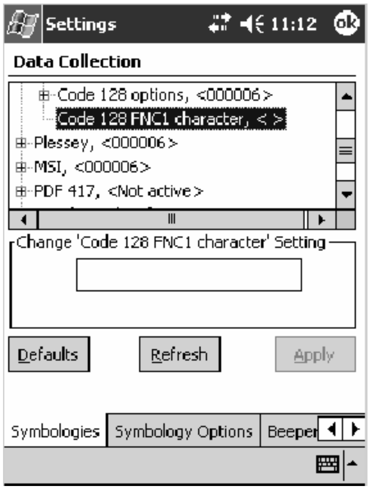
Tap (+) to expand the **Code 128** parameter, then type the ASCII characters to be set for the **Code 128 FNC1 character** parameter.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.21.1

Options

Any ASCII character (default is the GS function character — ID hex)



⊞ Code 128 options, <000006>
Code 128 FNC1 character, < >
⊞ Plessey, <000006>
⊞ MSI, <000006>
⊞ PDF 417, <Not active>

Change 'Code 128 FNC1 character' Setting

Plessey

Plessey is a pulse-width modulated symbology like most other bar codes. It includes a start character, data characters, an eight-bit cyclic check digit, and a termination bar. The code is continuous and not self-checking. You need to configure two parameters for Plessey code: Start Code and Check Digit. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

Tap (+) to expand the **Plessey** parameter, select the setting to be changed, then select an option to change this setting or select an option from the drop-down list.

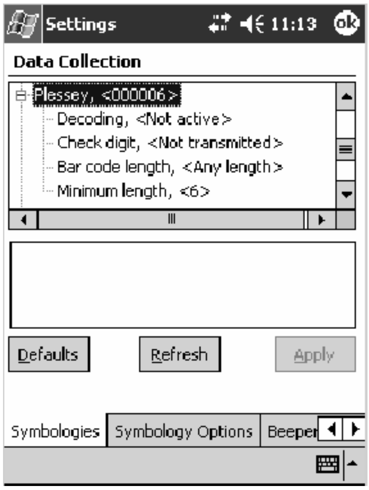
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.10.1

Options

Decoding	0 1	Not active (default) Active
Check digit	0 1	Not transmitted (default) Transmitted
Bar code length	0 1	Any length Minimum length (default)
Minimum length	001–254	Minimum length 1–254 (default is 6)

Note: If **Bar code length** = “1” then **Minimum length** is entered.



Plessey, <000006>
Decoding, <Not active>
Check digit, <Not transmitted>
Bar code length, <Any length>
Minimum length, <6>

MSI

MSI is a symbology similar to Plessey code (page 312) that includes a start pattern, data characters, one or two check digits, and a stop pattern. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

Tap (+) to expand the **MSI** parameter, select the setting to be changed, then select an option to change this setting or select an option from the drop-down list.

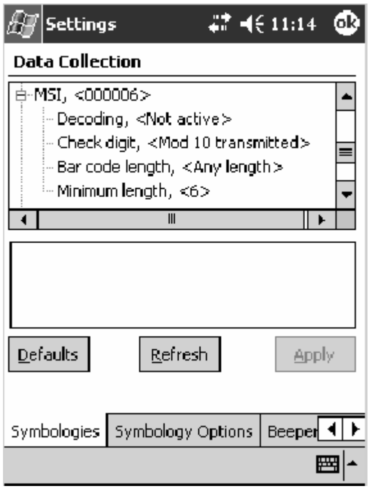
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.15.1

Options

Decoding	0 1	Not active (default) Active
Check digit	0 1 2 3	Mod 10 transmitted (default) Mod 10 not transmitted Double Mod 10 transmitted Double Mod 10 not transmitted
Bar code length	0 1	Any length Minimum length (default)
Minimum length	001–254	Minimum length 1–254 (default is 6)

Note: If **Bar code length** = “1” then **Minimum length** is entered.



Data Collection

MSI, <000006>
Decoding, <Not active>
Check digit, <Mod 10 transmitted>
Bar code length, <Any length>
Minimum length, <6>

PDF417

PDF417 is a stacked two-dimensional symbology that provides the ability to scan across rows of code. Each row consists of start/stop characters, row identifiers, and symbol characters, which consist of four bars and four spaces each and contain the actual data. This symbology uses error correction symbol characters appended at the end to recover loss of data.

Because the virtual wedge translates incoming data into keypad input, the size of the keypad buffer limits the effective length of the label to 128 characters. Longer labels may be truncated. For PDF417 labels of more than 128 characters, you can develop an application that bypasses the keypad buffer.

Action

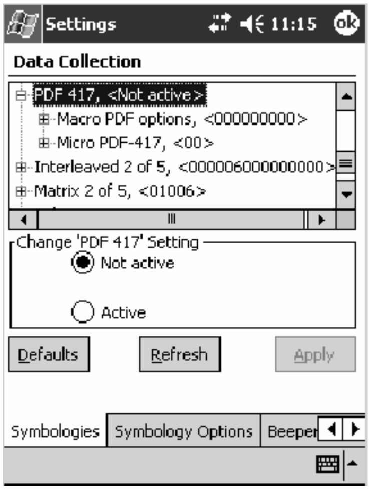
Tap the **PDF417** parameter, then select an option to change this parameter setting. Tap (+) to access either the **Macro PDF options** parameter or the **Micro PDF417** parameter.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.17.1

Options

0	Not active
1	Active (default)



PDF 417, <Not active>
Macro PDF options, <000000000>
Micro PDF-417, <00>
Interleaved 2 of 5, <000006000000000>
Matrix 2 of 5, <01006>

Change 'PDF 417' Setting
● Not active
○ Active

This illustration is from a 700 Color Computer using a laser scanner.

Macro PDF options

Macro PDF is used when a long message requires more than one PDF417 label. *Note that this is not available when you use an imager with your 700 Color Computer.*

- Select **Buffered** to store a multi-label PDF417 message in the Sabre buffer, thus transmitting the entire message when all labels are read.

- Select **Unbuffered** for multi-label PDF417 messages that are too long for the Sabre buffer (memory overflow). Each part of the PDF417 label is transmitted separately, and the host application must then assemble the message using the macro PDF control header transmitted with each label. *Control Header is only present in macro PDF codes and is always transmitted with unbuffered option.*

Action

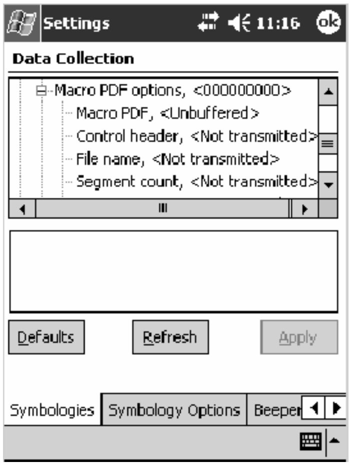
Tap (+) to expand the **PDF417** parameter, tap (+) to expand the **Macro PDF** parameter, select a setting to be changed, then select an option to change this setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.22.1

Options

Macro PDF	0 1	Unbuffered Buffered (default)
Control header	0 1	Not transmitted (default) Transmitted
File name	0 1	Not transmitted (default) Transmitted
Segment count	0 1	Not transmitted (default) Transmitted
Time stamp	0 1	Not transmitted (default) Transmitted
Sender	0 1	Not transmitted (default) Transmitted
Addressee	0 1	Not transmitted (default) Transmitted
File size	0 1	Not transmitted (default) Transmitted
Checksum	0 1	Not transmitted (default) Transmitted



Macro PDF options, <000000000>
Macro PDF, <Unbuffered>
Control header, <Not transmitted>
File name, <Not transmitted>
Segment count, <Not transmitted>

Micro PDF417

Micro PDF417 is a multi-row symbology derived from and closely based on PDF417 *(page 314)*. A limited set of symbology sizes is available, together with a fixed level of error correction for each symbology size. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

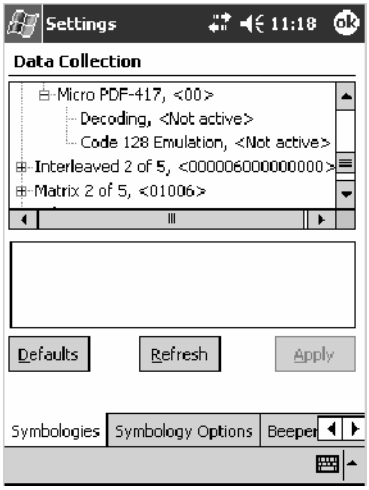
Tap (+) to expand the **PDF417** parameter, tap (+) to expand the **Micro PDF417** parameter, select a setting to be changed, then select an option to change this setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.27.1

Options

Decoding	0 1	Not active (default) Active
Code 128 Emulation	0 1	Not active (default) Active



⊟-Micro PDF-417, <00>
Decoding, <Not active>
Code 128 Emulation, <Not active>
⊞-Interleaved 2 of 5, <000006000000000>
⊞-Matrix 2 of 5, <01006>

Interleaved 2 of 5

Interleaved 2 of 5 (I 2 of 5) is a high-density, self-checking, continuous, numeric symbology used mainly in inventory distribution and the automobile industry.

Note: An Interleaved 2 of 5 bar code label must be at least three characters long for the 700 Color Computer to scan and decode correctly.

Action

Tap (+) to expand the **Interleaved 2 of 5** parameter, select the setting to be changed, then tap an option to change this setting or select an option from the drop-down list.

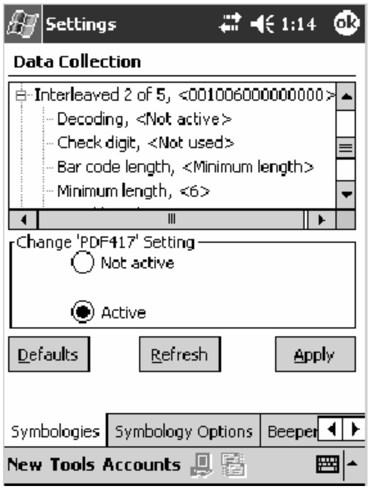
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.23.1

Options

Decoding	0 1	Not active (default) Active
Check digit	0 1 2 3 4	Not used (default) Mod 10 transmitted Mod 10 not transmitted French CIP transmitted French CIP not transmitted
Bar code length	0 1 2	Any length Minimum length (default) Fixed lengths
Minimum length	003–254	Minimum length 3–254 (default is 6)
Fixed length 1	003–254	Fixed bar code length 3–254 (default is 3)
Fixed length 2	003–254	Fixed bar code length 3–254 (default is 3)
Fixed length 3	003–254	Fixed bar code length 3–254 (default is 3)

Note: If **Bar code length** = "1" then **Minimum length** is entered. If **Bar code length** ="2" then **Fixed length 1**, **Fixed length 2**, or **Fixed length 3** is entered.



Data Collection

Interleaved 2 of 5, <001006000000000>
Decoding, <Not active>
Check digit, <Not used>
Bar code length, <Minimum length>
Minimum length, <6>

Change 'PDF417' Setting
○ Not active
● Active

Matrix 2 of 5

Matrix 2 of 5 is a numerical symbology. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

Tap (+) to expand the **Matrix 2 of 5** parameter, select the setting to be changed, then tap an option to change this setting or select an option from the drop-down list.

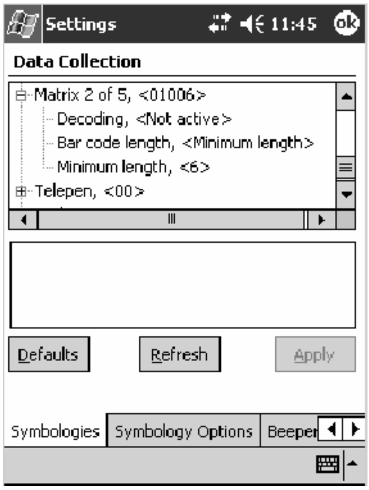
SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.24.1

Options

Decoding	0 1	Not active (default) Active
Bar code length	0 1	Any length Minimum length (default)
Minimum length	001–254	Minimum length 1–254 (default is 6)

Note: If **Bar code length** = "1" then **Minimum length** is entered.



Settings 11:45

Data Collection

Matrix 2 of 5, <01006>
Decoding, <Not active>
Bar code length, <Minimum length>
Minimum length, <6>
Telepen, <00>

Telepen

Telepen is an alphanumeric, case-sensitive, full ASCII symbology. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

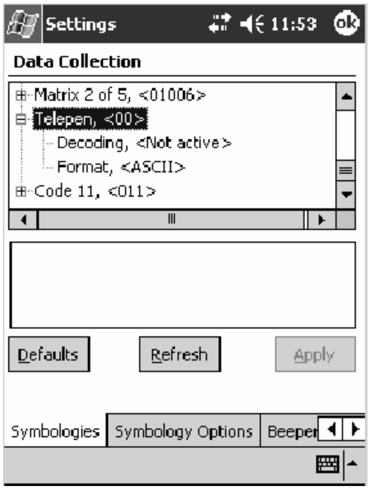
Tap (+) to expand the **Telepen** parameter, select the setting to be changed, then tap an option to change this setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.25.1

Options

Decoding	0 1	Not active (default) Active
Format	0 1	ASCII (default) Numeric



⊞ Matrix 2 of 5, <01006>
⊟ Telepen, <00>
Decoding, <Not active>
Format, <ASCII>
⊞ Code 11, <011>

Code 11

Code 11 is a high density, discrete numeric symbology that is extensively used in labeling telecommunications components and equipment. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

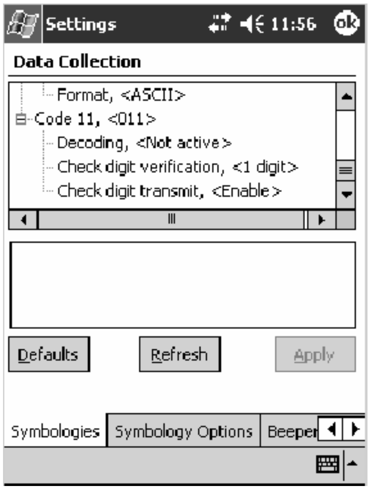
Tap (+) to expand the **Code 11** parameter, select the setting to be changed, then tap an option to change this setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.26.1

Options

Decoding	0 1	Not active (default) Active
Check digit verification	1 2	1 digit (default) 2 digits
Check digit transmit	0 1	Disable Enable (default)



Data Collection

Format, <ASCII>
Code 11, <011>
Decoding, <Not active>
Check digit verification, <1 digit>
Check digit transmit, <Enable>

QR Code

QR Code (Quick Response Code) is a two-dimensional matrix symbology containing dark and light square data modules. It has position detection patterns on three of its four corners and features direct encodation of the Japanese Kana-Kanji character set. It can encode up to 2509 numeric or 1520 alphanumeric characters and offers three levels of error detection. *Note that this is not available when you use a laser scanner with your 700 Color Computer or if you are using a 730 Computer.*

Action

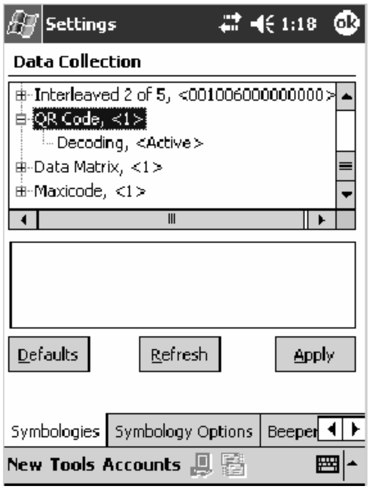
Tap (+) to expand the **QR Code** parameter, select the setting to be changed, then tap an option to change this setting or select an option from the drop-down list.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.35.1

Options

Decoding	0	Not active
	1	Active (default)



Data Collection

⊞ Interleaved 2 of 5, <001006000000000>
⊟ QR Code, <1>
 Decoding, <Active>
⊞ Data Matrix, <1>
⊞ Maxicode, <1>

Data Matrix

A two-dimensional matrix symbology, which is made of square modules arranged within a perimeter finder pattern. The symbology utilizes Error Checking and Correcting (ECC) algorithm with selectable levels for data error recovery and Cyclic Redundancy Check algorithm to validate the data. The character set includes either 128 characters conforming to ISO 646 (ANSI X3.4 - 1986) or 256 extended character set. Maximum capacity of a symbol is 2335 alphanumeric characters, 1556 8-bit byte characters or 3116 numeric digits. *Note that this is not available when you use a laser scanner with your 700 Color Computer or if you are using a 730 Computer.*

Action

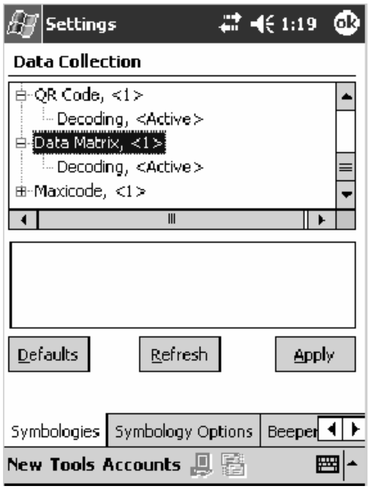
Tap (+) to expand the **Data Matrix** parameter, select the setting to be changed, then tap an option to change this setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.34.1

Options

Decoding	0	Not active
	1	Active (default)



Data Collection

QR Code, <1>
Decoding, <Active>
Data Matrix, <1>
Decoding, <Active>
Maxicode, <1>

MaxiCode

MaxiCode is a fixed-size 2-D matrix symbology which is made up of offset rows of hexagonal elements arranged around a unique circular finder pattern. ASCII data is encoded in six-bit symbol characters. The symbol contains 33 rows which are alternately 30 and 29 elements wide. There are five different code sets. A single MaxiCode symbol can encode up to 93 characters of data. *Note that this is not available when you use a laser scanner with your 700 Color Computer or if you are using a 730 Computer.*

Action

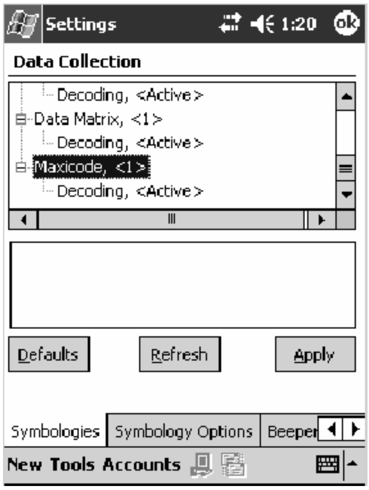
Tap (+) to expand the **MaxiCode** parameter, select the setting to be changed, then tap an option to change this setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.1.1.33.1

Options

Decoding	0	Not active
	1	Active (default)



Data Collection

- Decoding, <Active>
- Data Matrix, <1>
  - Decoding, <Active>
- Maxicode, <1>
  - Decoding, <Active>

Symbology Options

Data Collection

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > the **Data Collection** icon to access its control panel applet.

Use the right and left arrows to scroll to the **Symbology Options** tab, then tap this tab to access its parameters. The following are parameters for bar code symbology options. *Note that these are listed in the order of their appearance within the Symbology Options tab.*

Symbology ID

Identifies the bar code symbology in which data is encoded by prepending a user-specified symbology identifier to the data. You can prepend one of these types of character strings to identify the symbology:

- **User-defined ASCII Character** *(Option 1)*:
  A user-defined symbology identifier is a single ASCII character. You can assign a custom identifier character to each bar code symbology. *Note that this is not available when you use an imager with your 700 Color Computer.*
- **AIM ISO/IEC Standard** *(Option 2 — Required to define symbology IDs)*:
  The AIM Standard has a three-character structure which indicates the symbology and optional features. See the *AIM ISO/IEC Standard* for information.

Action

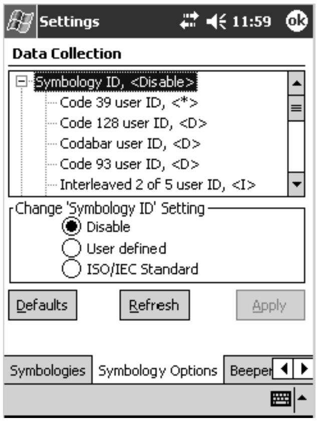
Select **Symbology ID**, then select an option to change this parameter setting. Tap (+) to expand the **Symbology ID** parameter, then select any of the user ID parameters listed. *See the top of the next page for a sample screen of the Code 39 user ID.*

SNMP OID

1.3.6.1.4.1.1963.15.3.3.4.1.22.1

Options

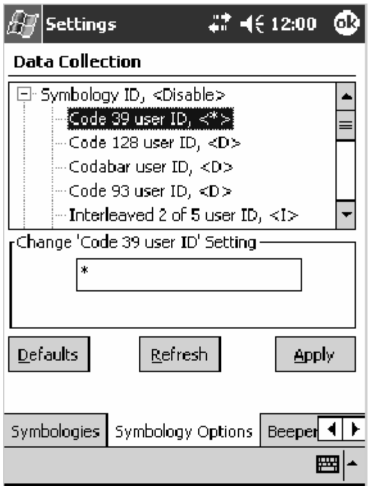
0	Disable (default)
1	User defined *(disabled when using an imager)*
2	ISO/IEC Standard



Data Collection

⊟ Symbology ID, <Disable>
- Code 39 user ID, <*>
- Code 128 user ID, <D>
- Codabar user ID, <D>
- Code 93 user ID, <D>
- Interleaved 2 of 5 user ID, <I>

Change 'Symbology ID' Setting
● Disable
○ User defined
○ ISO/IEC Standard


Settings

Data Collection

Symbology ID, <Disable>
Code 39 user ID, <*>
Code 128 user ID, <D>
Codabar user ID, <D>
Code 93 user ID, <D>
Interleaved 2 of 5 user ID, <I>

Change 'Code 39 user ID' Setting

*

Code 39 User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Code 39 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Code 39 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.3.1

Options: *x* where *x* is a single ASCII character. Default is asterisk (*).

Code 128 User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Code 128 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Code 128 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.5.1

Options: *x* where *x* is a single ASCII character. Default is asterisk (*).

Codabar User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Codabar bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Codabar user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.2.1

Options: *x* where *x* is a single ASCII character. Default is D.

Code 93 User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Code 93 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Code 93 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.4.1

Options: *x* where *x* is a single ASCII character. Default is asterisk (*).

Interleaved 2 of 5 User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Interleaved 2 of 5 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Interleaved 2 of 5 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.10.1

Options: *x* where *x* is a single ASCII character. Default is I (not lowercase L).

PDF417 User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify PDF417 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **PDF417 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.12.1

Options: *x* where *x* is a single ASCII character. Default is an asterisk (*).

MSI User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify MSI bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **MSI user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.11.1

Options: *x* where *x* is a single ASCII character. Default is D.

Plessey User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Plessey bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Plessey user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.13.1

Options: *x* where *x* is a single ASCII character. Default is D.

Standard 2 of 5 User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Standard 2 of 5 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Standard 2 of 5 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.23.1

Options: *x* where *x* is a single ASCII character. Default is D.

UPC A User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify UPC-A (Universal Product Code) bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **UPC A user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.6.1

Options: *x* where *x* is a single ASCII character. Default is A.

UPC E User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify UPC-E bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **UPC E user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.7.1

Options: *x* where *x* is a single ASCII character. Default is E.

EAN 8 User ID

If "1" was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify EAN-8 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **EAN 8 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.8.1

Options: *x* where *x* is a single ASCII character. Default is \xFF.

EAN 13 User ID

If "1" was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify EAN-13 (European Article Numbering) bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **EAN 13 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.9.1

Options: *x* where *x* is a single ASCII character. Default is F.

Matrix 2 of 5 User ID

If "1" was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Matrix 2 of 5 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Matrix 2 of 5 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.24.1

Options: *x* where *x* is a single ASCII character. *Default is D.*

Telepen User ID

If "1" was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Telepen bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Telepen user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.25.1

Options: *x* where *x* is a single ASCII character. Default is an asterisk (*).

Code 11 User ID

If “1” was selected in the **Symbology ID** parameter, you can set your own ASCII character to identify Code 11 bar code data. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action: Tap (+) to expand the **Symbology ID** parameter, select the **Code 11 user ID** parameter, then enter a user ID value to change this parameter setting.

SNMP OID: 1.3.6.1.4.1.1963.15.3.3.4.1.16.1

Options: *x* where *x* is a single ASCII character. Default is asterisk (*).

Prefix

Prepends a string of up to 20 ASCII characters to all scanned data.

Action

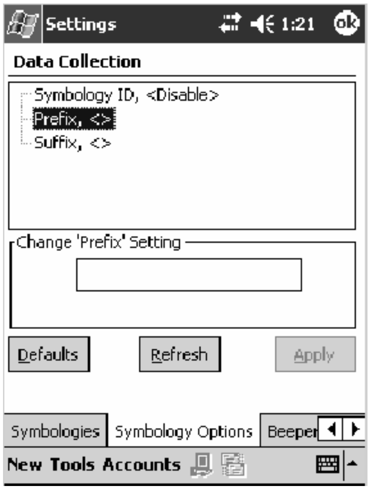
Tap the **Prefix** parameter, then enter a prefix value to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.4.1.29.1

Options

Acceptable values are up to 20 ASCII characters. Embedded null (<NUL >) characters are not allowed. Default is no characters (disabled).



Data Collection

- Symbology ID, <Disable>
- Prefix, <>
- Suffix, <>

Change 'Prefix' Setting

Suffix

Appends a string of up to 20 ASCII characters to all scanned data.

Action

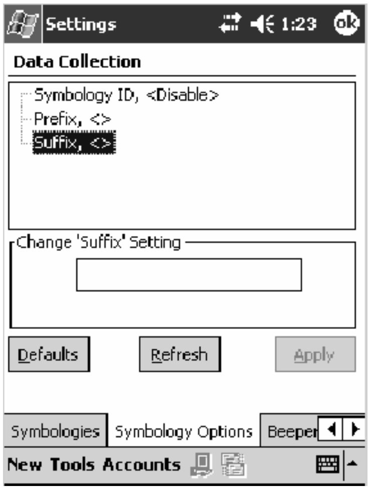
Tap the **Suffix** parameter, then enter a suffix value to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.4.1.30.1

Options

Acceptable values are up to 20 ASCII characters. Embedded null (<NUL >) characters are not allowed. Default is no characters (disabled).



Data Collection

Symbology ID, <Disable>
Prefix, <>
Suffix, <>

Change 'Suffix' Setting

Beeper/LED

Data Collection

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > the **Data Collection** icon to access its control panel applet.

Use the right and left arrows to scroll to the **Beeper/LED** tab, then tap this tab to access its parameters.

Most of these functions are not available when using an imager. The following table shows which functions are supported either by an imager or by a laser scanner.

Beeper Function	Imager	Laser Scanner
Beeper	X	
Beeper Volume		X
Beeper Frequency		X
Good Read Beeps		X
Good Read Beep Duration		X

The following are parameters for features on the 700 Color Computer. *Note that these are listed in the order of their appearance.*

Beeper

Sets the volume for the good read beep. *Note that this is not available when you use a laser scanner with your 700 Color Computer.*

Action

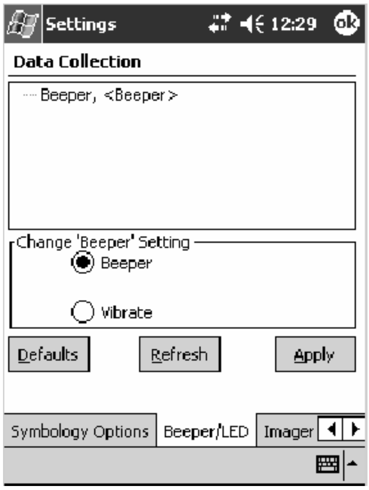
Tap the **Beeper** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.1.4.1.6.1

Options

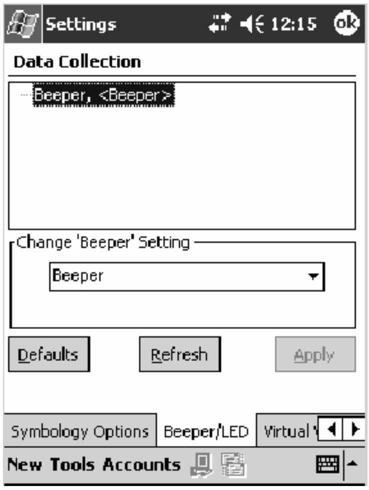
1 Beeper (default)
4 Vibrate *(not supported on 730 Computers)*



Beeper, <Beeper>

Change 'Beeper' Setting
● Beeper
○ Vibrate

700 Color with Imager Screen



Beeper, <Beeper>

Change 'Beeper' Setting
Beeper

730 Screen

Beeper Volume

Sets the volume for the good read beep. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

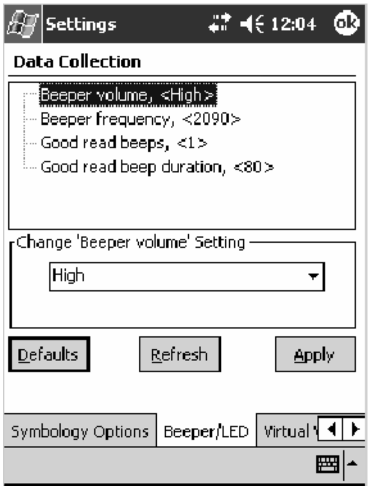
Tap the **Beeper volume** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.1.4.1.6.1

Options

0	Low
1	High (default)
2	Medium
3	Off
4	Vibrate



Settings 12:04

Data Collection

- Beeper volume, <High>
- Beeper frequency, <2090>
- Good read beeps, <1>
- Good read beep duration, <80>

Change 'Beeper volume' Setting: High

Disabling the Volume

Sounds & Notifications

To disable the beeper, tap **Start** > **Settings** > the **Personal** tab > **Sounds & Notifications** > the **Volume** tab, drag the **System volume** slider bar to the left "Silent" position, then tap **ok** to exit this applet. See Chapter 1, "*Introduction*" for more information.

Beeper Frequency

Sets the frequency for the good read beep. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

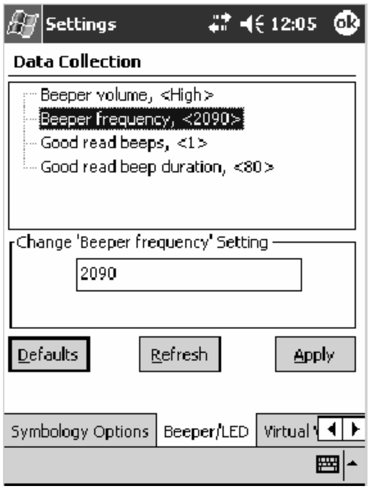
Tap the **Beeper frequency** parameter, then enter a frequency value to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.1.4.1.7.1

Options

1000–4095 (default is 2090)



Settings 12:05

Data Collection

- Beeper volume, <High>
- Beeper frequency, <2090>
- Good read beeps, <1>
- Good read beep duration, <80>

Change 'Beeper frequency' Setting

2090

Good Read Beeps

Sets the number of good read beeps. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

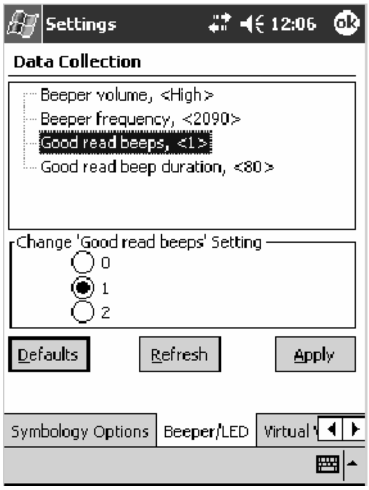
Tap the **Good read beeps** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.1.4.1.8.1

Options

0	No beeps
1	One beep (default)
2	Two beeps



Settings 12:06

Data Collection

- Beeper volume, <High>
- Beeper frequency, <2090>
- Good read beeps, <1>
- Good read beep duration, <80>

Change 'Good read beeps' Setting

○ 0
● 1
○ 2

Good Read Beep Duration

Sets the duration of the good read beep. *Note that this is not available when you use an imager with your 700 Color Computer.*

Action

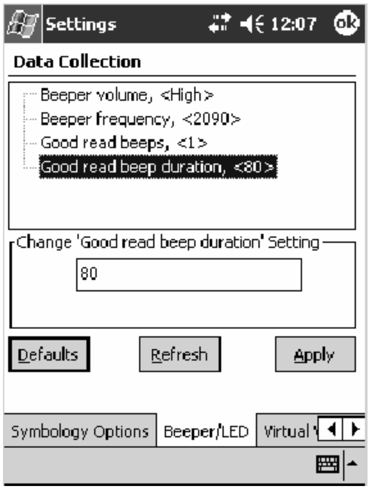
Tap the **Good read beep duration** parameter, then enter a duration value to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.1.4.1.9.1

Options

0–2550	Beep duration in milliseconds. (default is 80)



Data Collection

Beeper volume, <High>
Beeper frequency, <2090>
Good read beeps, <1>
Good read beep duration, <80>

Change 'Good read beep duration' Setting

80

Imager

Note: These instructions do *not* apply to the 730 Computer.

Data Collection

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Data Collection** to access its control panel applet.

Use the right and left arrows to scroll to the **Imager** tab, then tap this tab to access its parameters.

The following are parameters for the imager. Note that these are listed in the order of their appearance within the Imager tab.

Aimer LED Duration

The Aimer LED Duration controls the time the Aimer LED is turned on when the scan button is pressed. After this time, images are captured for decoding. The purpose is to position the Aimer LED on the bar code symbol before attempting to decode the bar code. *Note that this is not available when you use a laser scanner with your 700 Color Computer.*

Action

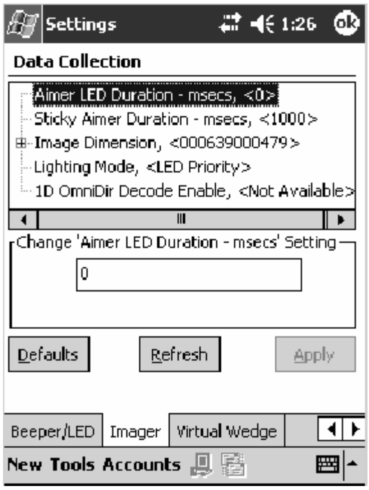
Tap the **Aimer LED Duration** parameter, then enter a value to change this setting. Note that values must be in 50 ms increments, such as 500, 650, or 32500. Values not entered in 50 ms increments are rounded down. For example, 2489 ms is rounded down to 2450 ms, 149 ms is rounded down to 100 ms, etc..

SNMP OID

1.3.6.1.4.1.1963.15.3.3.3.1.1.21.1

Options

0–65500 ms (Default is 0)



Data Collection

Aimer LED Duration - msecs, <0>
Sticky Aimer Duration - msecs, <1000>
Image Dimension, <000639000479>
Lighting Mode, <LED Priority>
1D OmniDir Decode Enable, <Not Available>

Change 'Aimer LED Duration - msecs' Setting

0

Sticky Aimer Duration

The Sticky Aimer Duration controls the time the Aimer LED stays on after the a bar code read completes or after the trigger button is released. *Note that this is not available when you use a laser scanner with your 700 Color Computer.*

Action

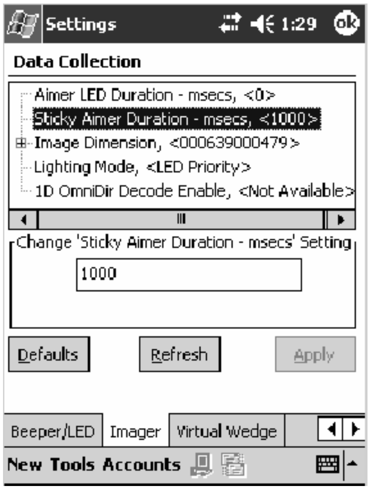
Tap the **Sticky Aimer Duration** parameter, then enter a value to change this setting. Note that values must be in 50 ms increments, such as 500, 650, or 32500. Values not entered in 50 ms increments are rounded down. For example, 2489 ms is rounded down to 2450 ms, 149 ms is rounded down to 100 ms, etc..

SNMP OID

1.3.6.1.4.1.1963.15.3.3.3.1.1.24.1

Options

0–65535 ms (Default is 1000)



Data Collection

Aimer LED Duration - msecs, <0>
Sticky Aimer Duration - msecs, <1000>
Image Dimension, <000639000479>
Lighting Mode, <LED Priority>
1D OmniDir Decode Enable, <Not Available>

Change 'Sticky Aimer Duration - msecs' Setting

1000

Image Dimension

The image dimensions control the vertical size of the image for decoding. This can restrict the image to one bar code when otherwise, there might be more than one bar code in the image to be decoded. *Note that this is not available when you use a laser scanner with your 700 Color Computer.*

Action

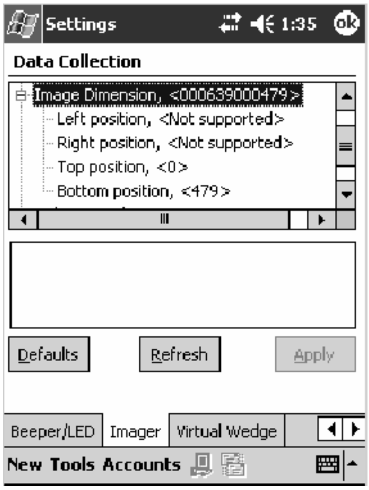
Tap the **Image dimension** parameter, select the position to be changed, then tap an option or enter a value to change this position.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.3.1.1.22.1

Options

Left position	0	Not supported
Right position	0	Not supported
Top position	0–478	Position in pixels (Default is 0)
Bottom position	0–479	Position in pixels (Default is 479)



Data Collection

Image Dimension, <000639000479>
Left position, <Not supported>
Right position, <Not supported>
Top position, <0>
Bottom position, <479>

Lighting Mode

The Lighting Mode sets the lighting mode of the imager. When set to “LED Priority,” the imager depends more on ambient lighting to illuminate the bar code for reading. When set to “Aperture Priority,” the imager uses its built-in LED to illuminate the bar code for reading. *Note that this is not available when you use a laser scanner with your 700 Color Computer.*

Action

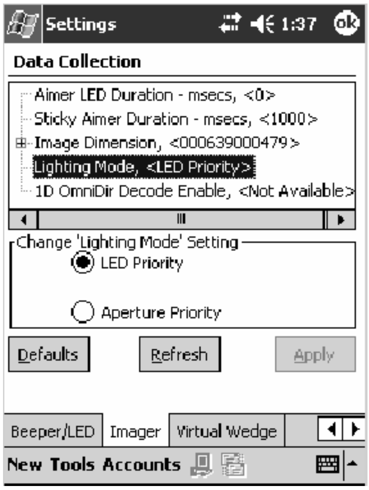
Tap the **Lighting Mode** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.3.1.1.23.1

Options

0	LED Priority (default)
1	Aperture Priority


Settings 1:37

Data Collection

Aimer LED Duration - msecs, <0>
Sticky Aimer Duration - msecs, <1000>
⊞ Image Dimension, <000639000479>
Lighting Mode, <LED Priority>
1D OmniDir Decode Enable, <Not Available>

Change 'Lighting Mode' Setting
● LED Priority
○ Aperture Priority

1D OmniDir Decode Enable

The 1D OmniDir Decode Enable affects the scanning abilities of the IT4000 Imager. With 1D omni directional enabled, the imager is able to decode images and bar code labels regardless of the orientation of the label. With 1D omni directional disabled, the imager only decodes labels in the direction of the aimer LED. *Note that this is not available when you use a laser scanner with your 700 Color Computer.*

Action

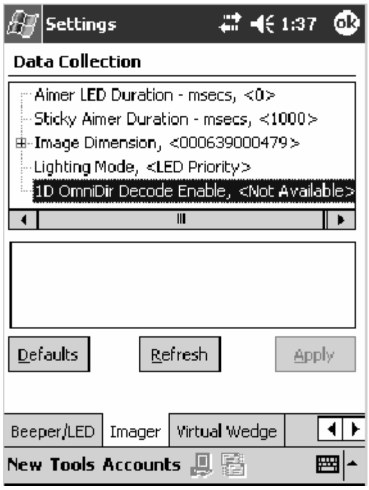
Tap the **1D OmniDir Decode Enable** parameter,then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.3.3.1.1.25.1

Options

0	Disabled
1	Enabled (default)



Data Collection

- Aimer LED Duration - msecs, <0>
- Sticky Aimer Duration - msecs, <1000>
- Image Dimension, <000639000479>
- Lighting Mode, <LED Priority>
- 1D OmniDir Decode Enable, <Not Available>

Virtual Wedge

Data Collection

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Data Collection** to access its control panel applet.

Use the right and left arrows to scroll to the **Virtual Wedge** tab, then tap this tab to access its parameters.

The following are parameters for the virtual wedge scanner. *Note that these are listed in the order of their appearance within the Virtual Wedge tab.*

Virtual Wedge

Enables or disables the virtual wedge for the internal scanner. The virtual wedge retrieves scanned Automatic Data Collection (ADC) data and sends it to the keypad driver so that the 700 Color Computer can receive and interpret the data as keypad input.

Because the virtual wedge translates incoming data into keypad input, the size of the keypad buffer limits the effective length of the label to 128 characters. Longer labels may be truncated. For labels of more than 128 characters, you need to develop an application that bypasses the keypad buffer.

Action

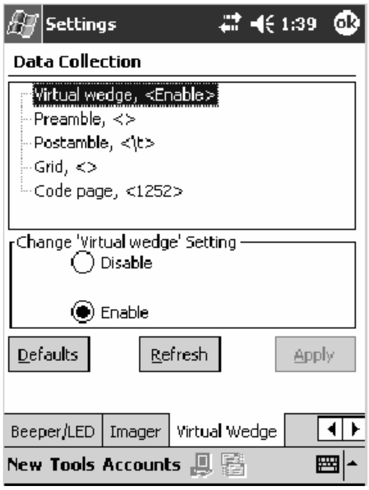
Tap the **Virtual Wedge** parameter, then tap an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.2.1.1.2.1

Options

0	Disable
1	Enable (default)



Data Collection

Virtual wedge, <Enable>
Preamble, <>
Postamble, <\t>
Grid, <>
Code page, <1252>

Change 'Virtual wedge' Setting
○ Disable
● Enable

Preamble

Sets the preamble that precedes any data you scan with the 700 Color Computer. Common preambles include a data location number or an operator number.

Action

Tap the **Preamble** parameter, then enter a preamble value to change this parameter setting.

SNMP OID

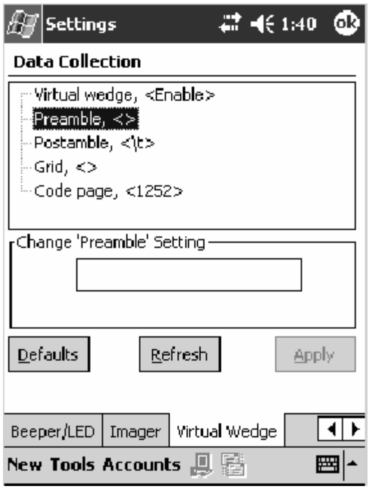
1.3.6.1.4.1.1963.15.3.2.1.1.3.1

Syntax

```
ADdata
```

where *data* is any acceptable values up to 31 ASCII characters. Embedded null (<NUL >) characters are not allowed. Below are the non-printing characters you can use for Virtual Wedge Preambles. *Default is blank.*

\a	Alert (bell)
\b	Backspace
\f	Form Feed
\n	New line/line feed
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\xnnnn	*nnnn* is up to four HEX digits. Use leading zeros to fill out to four digits to ensure proper conversion. For example, to prepend the character M to scanned data, set the Preamble to either 1) M, or 2) x004D, where 4D is the HEX equivalent for an uppercase M.



Data Collection

- Virtual wedge, <Enable>
- Preamble, <>
- Postamble, <\t>
- Grid, <>
- Code page, <1252>

Change 'Preamble' Setting

Note: When you enter the AD command without data, the preamble is disabled. If you want to use quotation marks or the following combinations of characters as part of the appended data, separate those characters from the AD command with quotes. If you do not use quotes as described here, the 700 Color Computer interprets the characters as another configuration command:
AD
AE
AF
KC
BV
EX
DF

Example

To use the two-character string BV as a preamble, scan this command (as a Code 39 label) or send this command through the network: `$+AD“BV”`

Postamble

Sets the postamble that is appended to any data you scan with the 700 Color Computer. Common postambles include cursor controls, such as tabs or carriage return line feeds.

Action

Tap the **Postamble** parameter, then enter a postamble value to change this parameter setting.

SNMP OID

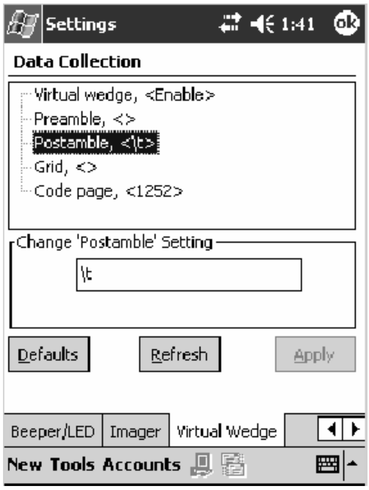
1.3.6.1.4.1.1963.15.3.2.1.1.4.1

Syntax

```
AEdata
```

where *data* is any acceptable values up to 31 ASCII characters. Embedded null (<NUL >) characters are not allowed. Below are the non-printing characters you can use for Virtual Wedge Postambles:

\a	Alert (bell)
\b	Backspace
\f	Form Feed
\n	New line/line feed
\r	Carriage return
\t	Horizontal tab (default)
\v	Vertical tab
\xnnnn	*nnnn* is up to four HEX digits. Use leading zeros to fill out to four digits to ensure proper conversion. For example, to prepend the character M to scanned data, set the Preamble to either 1) M, or 2) x004D, where 4D is the HEX equivalent for an uppercase M.



Virtual wedge, <Enable>
Preamble, <>
Postamble, <\t>
Grid, <>
Code page, <1252>

Change 'Postamble' Setting: \t

Note: When you enter the AE command without data, the postamble is disabled. If you want to use quotation marks or the following combinations of characters as part of the appended data, separate those characters from the AE command with quotes. If you do not use quotes as described here, the 700 Color Computer interprets the characters as another configuration command.
AD
AE
AF
KC
BV
EX
DF

Example

To use the two-character string BV as a postamble, scan this command (as a Code 39 label) or send this command through the network: `$+AE“BV”`

Grid

Sets the virtual wedge grid, which filters the data coming from this 700 Color Computer. The data server supports data filtering, which allows you to selectively send scanned data. The virtual wedge grid is similar to the "format" argument of the C Runtime Library scan function.

Action

Tap the **Grid** parameter, then enter a grid value to change this parameter setting.

SNMP OID

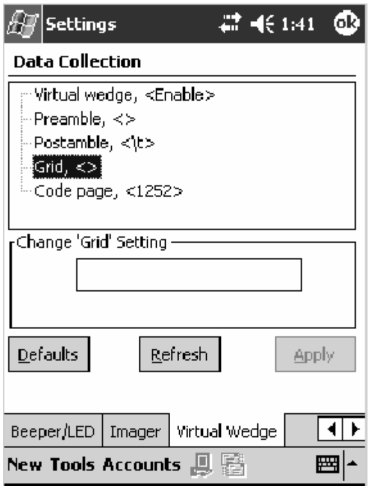
1.3.6.1.4.1.1963.15.3.2.1.1.5.1

Syntax

```
AF<symID> filter-expression= > editing-expression
```

where:

- *<symID>*
  The AIM symbology ID *(optional)*.
- *filter-expression*
  Any character string that includes valid filter expression values. See the user manual provided with your resouce kit via the IDL for a list of valid filter expression values.
- *editing-expression*
  Any character string that includes valid editing expression values. See the user manual provided with your resouce kit via the IDL for a list of valid editing expression values.



Data Collection

Virtual wedge, <Enable>
Preamble, <>
Postamble, <\t>
Grid, <>
Code page, <1252>

Change 'Grid' Setting

Code Page

Sets the virtual wedge code page. The code page controls the translation from the character set of the raw collected data to Unicode, which is the character set expected by Windows CE applications. The default code page is 1252, which is the Windows Latin 1 (ANSI) character set.

Action

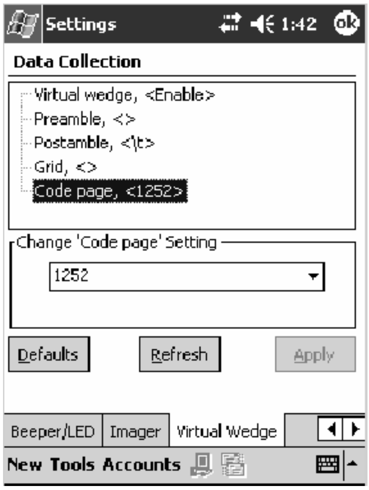
Tap the **Code Page** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.15.3.2.1.1.6.1

Options

The only acceptable value for the code page parameter is "*1252*," which is the default.



Data Collection

- Virtual wedge, <Enable>
- Preamble, <>
- Postamble, <\t>
- Grid, <>
- Code page, <1252>

Change 'Code page' Setting: 1252

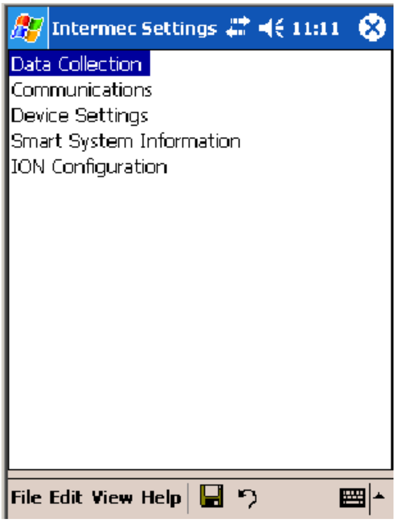
Intermec Settings Control Panel Applet

You may have the Intermec Settings control panel applet. Information about the settings you can configure with this applet is described in the *Intermec Computer Command Reference Manual*. The online manual is available from the Intermec web site at www.intermec.com.

See the Data Collection Resource Kit in the IDL for information about data collection functions. The IDL is available as a download from the Intermec web site at www.intermec.com. Contact your Intermec representative for more information.

Intermec Settings

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Intermec Settings** to access its control panel applet.



SNMP Control Panel Applet

Note: This applet is not available in units with PSM Build 3.00 or newer. To determine your PSM build version, tap **Start** > **Programs** > **File Explorer** > the **PSMinfo** text file.

Intermec Settings

If your unit has PSM Build 3.00 or newer, then you may have the Intermec Settings control panel applet in place of the SNMP applet. Information about the settings you can configure with the Intermec Settings applet is described in the *Intermec Computer Command Reference Manual*. The online manual is available from the Intermec web site at www.intermec.com.

Simple Network Management Protocol (SNMP) parameters include identification information, security encryption, security community strings, and traps.

SNMP

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **SNMP** to access its control panel applet.

Tap a tab to access its menus. These tabs represent three groups of settings or parameters:

- **Security** *(starting on the next page)*
- **Traps** *(starting on page 357)*
- **Identification** *(starting on page 359)*

Security

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **SNMP** > the **Security** tab to access its parameters.

The following are parameters that affect encryption and community strings. *Note that these are listed in the order of their appearance within the Security tab.*

Read Only Community

Sets the read-only community string for this 700 Color Computer, which is required for processing of SNMP get and get next requests.

Action

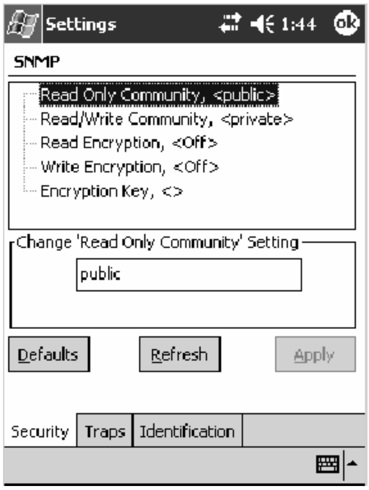
Tap the **Read Only Community** parameter, then enter a community string to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.10.5.1.2.0

Options

The read-only community string can be up to 128 ASCII characters. Default is Public.



Settings 1:44

SNMP

Read Only Community, <public>
Read/Write Community, <private>
Read Encryption, <Off>
Write Encryption, <Off>
Encryption Key, <>

Change 'Read Only Community' Setting
public

Read/Write Community

Sets the read/write community string, which is required for processing of SNMP set requests by this 700 Color Computer. An SNMP packet with this name as the community string will also process SNMP get and next requests.

Action

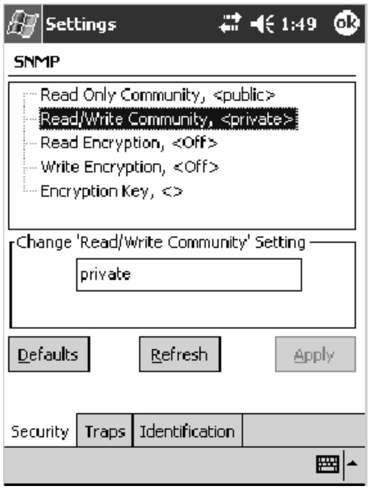
Tap the **Read/Write Community** parameter, then enter a community string to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.10.5.1.3.0

Options

The read/write community string can be up to 128 ASCII characters. Default is Private.



SNMP

Read Only Community, <public>
Read/Write Community, <private>
Read Encryption, <Off>
Write Encryption, <Off>
Encryption Key, <>

Change 'Read/Write Community' Setting

private

Read Encryption

Sets the packet-level mode of security for SNMP read-only requests. If you enable read encryption, all received SNMP get and get next packets have to be encrypted or the packet will not be authorized. If encryption is enabled, you can only use software provided by Intermec Technologies.

Note: To enable security encryption, you also need to set the Security Encryption Key (page 356).

Action

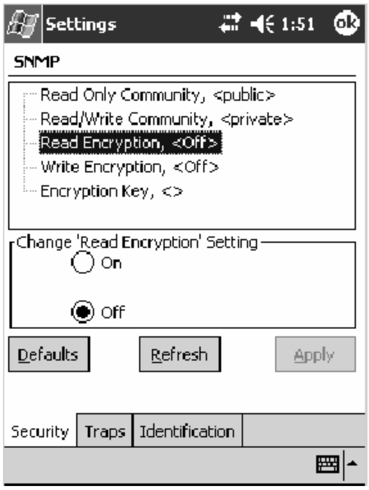
Tap the **Read Encryption** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.10.5.1.4.0

Options

1	On	SNMP get and get next packets must be encrypted
2	Off	SNMP packets do not have to be encrypted (default)



Read Only Community, <public>
Read/Write Community, <private>
Read Encryption, <Off>
Write Encryption, <Off>
Encryption Key, <>

Change 'Read Encryption' Setting
○ On
● Off

Write Encryption

Sets the packet-level mode of security for SNMP read/write requests. If you enable write encryption, all SNMP packets that are received with the read/write community string have to be encrypted or the packet will not be authorized. You need to use software from Intermec Technologies that supports encryption.

Note: To enable security encryption, you also need to set the Security Encryption Key (page 356).

Action

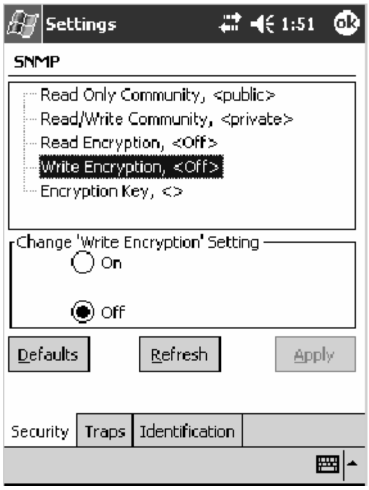
Tap the **Write Encryption** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.10.5.1.5.0

Options

1	On	SNMP packets must be encrypted
2	Off	SNMP packets do not have to be encrypted (default)



Settings 1:51

SNMP

- Read Only Community, <public>
- Read/Write Community, <private>
- Read Encryption, <Off>
- Write Encryption, <Off>
- Encryption Key, <>

Change 'Write Encryption' Setting

○ On

● Off

Encryption Key

Identifies the key that this 700 Color Computer uses to encrypt or decipher SNMP packets. Encryption is used only by software provided by Intermec Technologies. If encryption is enabled, SNMP management platforms will not be able to communicate with the 700 Color Computer. The encryption key is returned encrypted.

Action

Tap the **Encryption Key** parameter, then enter a security encryption key value to change this parameter setting.

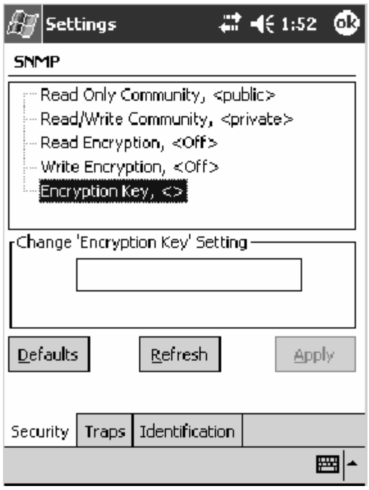
Note: You also need to set either **Read Encryption** (page 354) or **Write Encryption** (page 355) or both.

SNMP OID

1.3.6.1.4.1.1963.10.5.1.6.0

Options

The encryption key can be from 4 to 20 ASCII characters. Default is NULL.



SNMP

- Read Only Community, <public>
- Read/Write Community, <private>
- Read Encryption, <Off>
- Write Encryption, <Off>
- Encryption Key, <>

Change 'Encryption Key' Setting

Traps

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **SNMP** > the **Traps** tab to access its parameters.

The following are authentication and threshold parameters for traps. *Note that these are listed in the order of their appearance within the Traps tab.*

Authentication

Determines whether to send authentication traps. When trap authentication is enabled, an authentication trap is sent if an SNMP packet is received by the master agent with an invalid community string.

Action

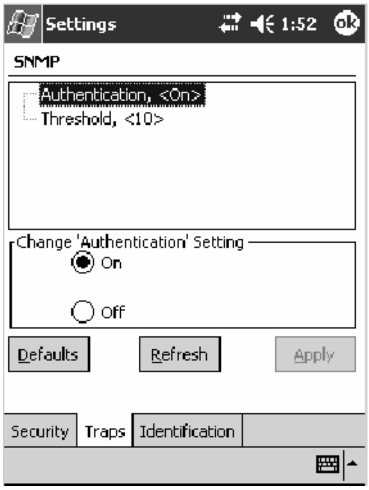
Tap the **Authentication** parameter, then select an option to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.10.5.2.2.0

Options

1	On (default)
2	Off



SNMP

Authentication, <On>
Threshold, <10>

Change 'Authentication' Setting

● On

○ Off

Threshold

Determines the maximum number of traps per second that the master agent generates. If the threshold is reached, the trap will not be sent.

Action

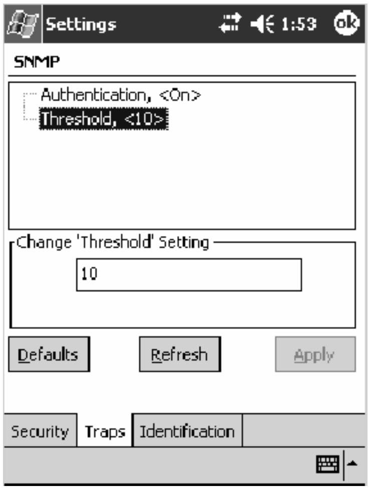
Tap the **Threshold** parameter, then enter a threshold value to change this parameter setting.

SNMP OID

1.3.6.1.4.1.1963.10.5.2.3.0

Options

Any positive integer value. Default is 10.



SNMP

Authentication, <On>

Threshold, <10>

Change 'Threshold' Setting

10

Identification

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **SNMP** > the **Identification** tab to access its parameters.

The following are parameters for contact, location, and name information for support purposes. *Note that these are listed in the order of their appearance within the Identification tab.*

Contact

Sets the contact information for the person responsible for this 700 Color Computer.

Action

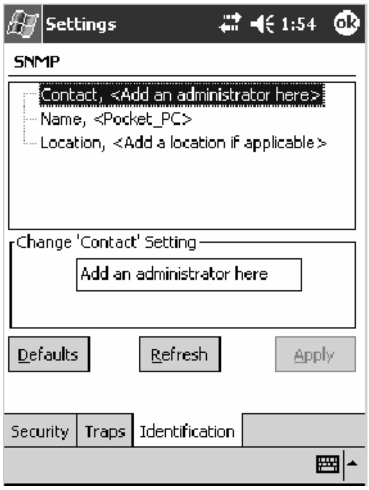
Tap the **Contact** parameter, then enter the name of your contact representative to change this parameter setting.

SNMP OID

1.3.6.1.2.1.1.4.0

Options

The identification contact may be up to 255 ASCII characters. Default is no characters or blank.



SNMP

Contact, <Add an administrator here>
Name, <Pocket_PC>
Location, <Add a location if applicable>

Change 'Contact' Setting

Add an administrator here

Name

Sets the assigned name for this 700 Color Computer.

Action

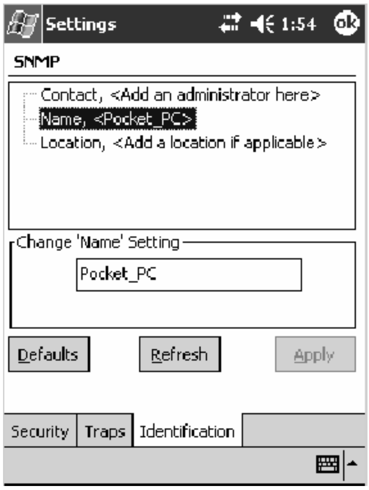
Tap the **Name** parameter, then enter the name of your 700 Color Computer to change this parameter setting.

SNMP OID

1.3.6.1.2.1.1.5.0

Options

The identification name may be up to 255 ASCII characters. Default is no characters or blank.



SNMP

Contact, <Add an administrator here>
Name, <Pocket_PC>
Location, <Add a location if applicable>

Change 'Name' Setting: Pocket_PC

Location

Sets the identification location for this 700 Color Computer, such as “Shipping.”

Action

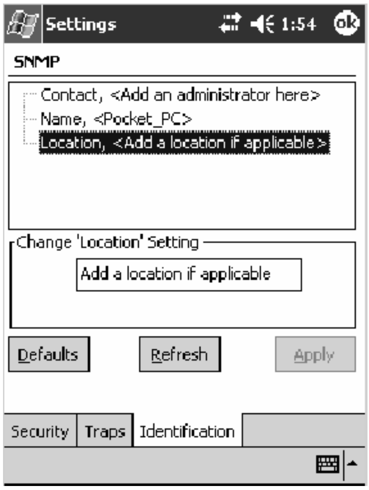
Tap the **Location** parameter, then enter the location of where your 700 Color Computer to change this parameter setting.

SNMP OID

1.3.6.1.2.1.1.6.0

Options

The identification location may be up to 255 ASCII characters. Default is no characters or blank.



SNMP

Contact, <Add an administrator here>
Name, <Pocket_PC>
Location, <Add a location if applicable>

Change 'Location' Setting

Add a location if applicable

Unit Information Control Panel Applet

Note: This applet is not available in units with PSM Build 3.00 or newer. To determine your PSM build version, tap **Start** > **Programs** > **File Explorer** > the **PSMinfo** text file.

Intermec Settings

If your unit has PSM Build 3.00 or newer, then you may have the Intermec Settings control panel applet in place of the Unit Information applet. Information about the settings you can configure with the Intermec Settings applet is described in the *Intermec Computer Command Reference Manual*. The online manual is available from the Intermec web site at www.intermec.com.

Unit Information is a read-only control panel applet that provides information about your 700 Color Computer, such as software version builds, available CAB files, and the internal battery status.

Unit Information

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Unit Information** to access its control panel applet.

Tap a tab to access its menus. These tabs represent three groups of settings or parameters:

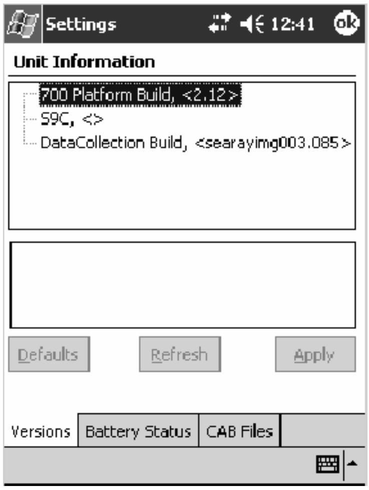
- **Versions** *(next paragraph)*
- **Battery Status** *(starting on page 363)*
- **CAB Files** *(starting on page 364)*

Versions

You can view the latest software build version on your 700 Color Computer by accessing the **Unit Information** control panel applet.

Unit Information

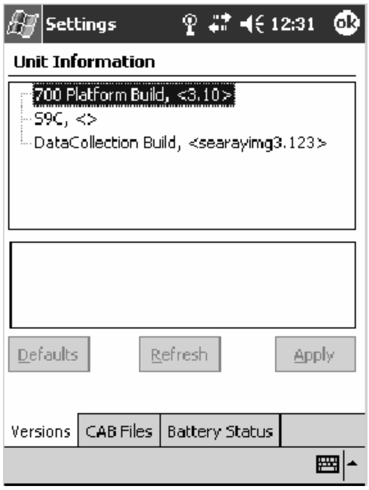
To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Unit Information** > the **Versions** tab to view the latest software build version. Tap **ok** to exit this information.



Unit Information

700 Platform Build, <2.12>
S9C, <>
DataCollection Build, <searayimg003.085>

700 Color Screen



Unit Information

700 Platform Build, <3.10>
S9C, <>
DataCollection Build, <searayimg3.123>

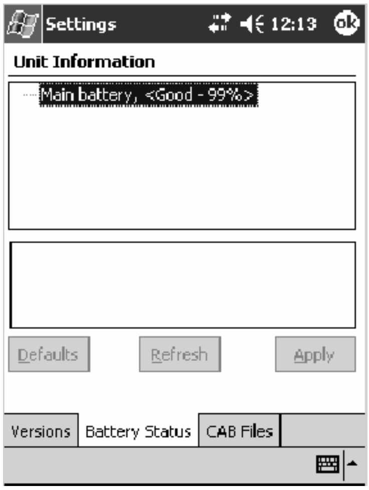
730 Screen

Battery Status

You can view the battery status for your 700 Color Computer by accessing the **Unit Information** control panel applet.

Unit Information

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Unit Information** > the **Battery Status** tab to view the current status. Tap **ok** to exit this information.



Unit Information

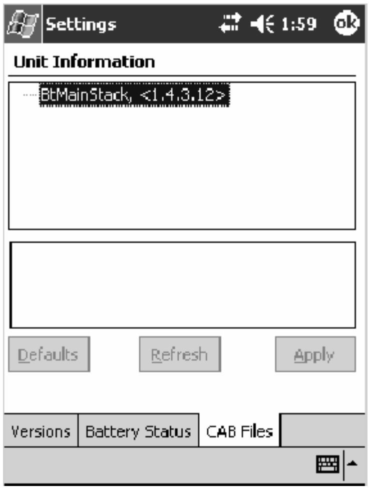
Main battery, <Good - 99%>

CAB Files

You can view the latest developer or released version of each CAB file from Intermec Technologies Corporation that are installed in your 700 Color Computer via the **Unit Information** control panel applet. *Custom CAB files are not displayed in this applet.* See the *Software Tools User's Manual* for more information about these files.

Unit Information

To access the information from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Unit Information** > the **CAB Files** tab to view the current CAB file versions. Tap **ok** to exit this information.



Unit Information

BtMainStack, <1.4.3.12>

When a CAB file is built, a registry entry is created with a build number for that file. This CAB Files control panel applet looks for a registry key for each CAB file installed. When the registry entry is found, the CAB file name and version number information are displayed. If a CAB file has not been installed, then its information is not displayed.

Below is a list of CAB files from Intermec Technologies that are available for your 700 Color Computer with their latest developer or released version of the software build. Should you need to add any of these to your 700 Color Computer, contact an Intermec representative.

- **BtMainStack**:
  Installation of the Main Bluetooth Stack is handled automatically as part of the operating system boot-up procedure. *See Chapter 4, "Network Support," for more information about Bluetooth wireless printing.*
- **Comm Port Wedge**:
  The software build for the Comm Port Wedge. *Note that the Comm Port Wedge CAB file is available.*
- **NPCPTest**:
  This installs a Norand® Portable Communications Protocol (NPCP) Printing test application which will print to an Intermec® 4815, 4820, or 6820 Printer. *See Chapter 5, "Printer Support," for more information.*

- **S9C Upgrade**:
  Installs the files needed to upgrade the S9C scanner firmware. *See the 700 Color Management Tools for information on upgrading the firmware.*
- **WinCfg**:
  Configures the NRINET.INI file, launches the NRINet client, and loads and unloads the LAN and WLAN device drivers.
- **Wireless Printing Sample**:
  Installs a sample application that developers can use for reference when they are developing their own Wireless Printing applications. The source code for this application is included as part of the Bluetooth Resource Kit in the IDL, which is available as a download from the Intermec web site at www.intermec.com. Contact your Intermec representative for more information.
- **ActiveX Control Tools**:
  This lists some of the CAB files that may be available with which to install ActiveX Control Tools. *See the online help for more information.*
  - **AXCommunication**:
    Communication controls that transmit or receive messages from input connections.
  - **AXFileTransfer**:
    File transfer controls that transmit and receive files using the Trivial File Transfer Protocol (TFTP).
  - **AXReaderCommand**:
    Reader command functions that modify and retrieve configuration information from your 700 Color Computer.
  - **AXVWedge**:
    The virtual wedge control that retrieves scanned ADC data and sends it to the keyboard driver to interpret data as keyboard input.

Utilities Control Panel Applet

The Utilities control panel applet examines and modifies settings and operational modes of specific hardware and software on the 700 Color Computer, including the dock switch, registry storage, wakeup mask, and application launch keys.

To access the settings from the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Utilities** to access its control panel applet.

Use the left and right arrows to scroll through the tabs along the bottom of the control panel applet, then tap a tab to access its menus. These tabs represent the following groups of settings or parameters:

- **Dock Switch** *(next paragraph)*
- **Registry Save** *(page 367)*
- **Wakeup Mask** *(page 368)*
- **App Launch** *(page 369)*

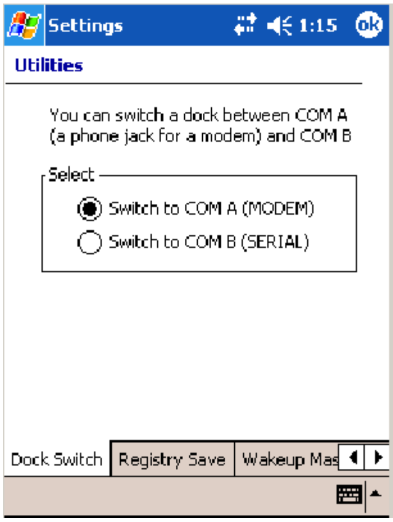
Dock Switch

From the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Utilities** > the **Dock Switch** tab to access the Dock Switch control panel applet.

Use this applet to control the position of the dock switch. This can be set either to a COM A (phone jack for a modem) position or to a COM B (serial) position.

If switched to COM B and suspended, the terminal has the following behavior:

- If the 700 Color Computer is on charge, the dock switch remains switched to COM B.
- If the 700 Color Computer is off charge, the dock switch switches to COM A and remain in this position until the 700 Color Computer resumes charge.



You can switch a dock between COM A (a phone jack for a modem) and COM B

Select

● Switch to COM A (MODEM)

○ Switch to COM B (SERIAL)

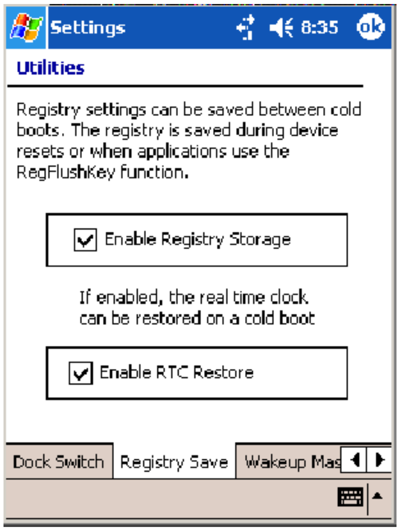
Registry Save

Utilities

From the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Utilities** > the **Registry Save** tab to access the Registry Save control panel applet.

For Windows Mobile 2003, the only medium available for saving the registry is the Flash File System (PSM). Registry data is stored in the “\Flash_File_Store\Registry” path. Check **Enable Registry Storage** to enable this function.

To ensure that the 700 Color Computer restores the real-time clock after a cold-boot, check the **Enable RTC Restore** option. *Note that this does not apply to the 730 Computer.*



Utilities

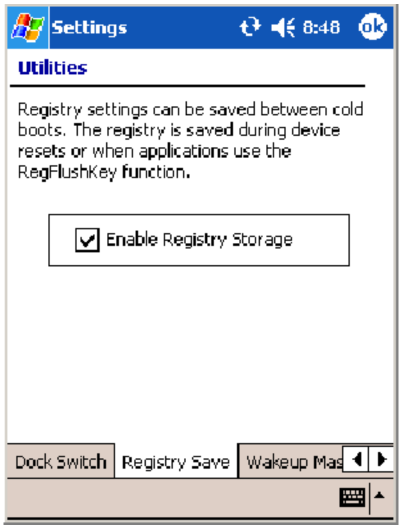
Registry settings can be saved between cold boots. The registry is saved during device resets or when applications use the RegFlushKey function.

☑ Enable Registry Storage

If enabled, the real time clock can be restored on a cold boot

☑ Enable RTC Restore

700 Color Screen



Utilities

Registry settings can be saved between cold boots. The registry is saved during device resets or when applications use the RegFlushKey function.

☑ Enable Registry Storage

730 Screen

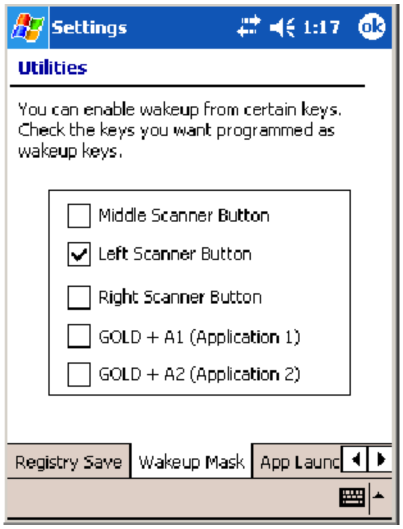
Wakeup Mask

Utilities

From the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Utilities** > the **Wakeup Mask** tab to access the Wakeup Mask control panel applet.

This applet programs three scanner buttons and the A1 and A2 application keys to be "wakeup" or resume keys. That is, to prompt the 700 Color Computer to "wake up" or resume activity after going to "sleep" as a result of being inactive after a length of time. This information will remain between warm and cold boots.

Check the appropriate box, then tap **ok** to apply your settings.



Utilities

You can enable wakeup from certain keys. Check the keys you want programmed as wakeup keys.

□ Middle Scanner Button
☑ Left Scanner Button
□ Right Scanner Button
□ GOLD + A1 (Application 1)
□ GOLD + A2 (Application 2)

Based on your setting, do the following to "wake up" the 700 Color Computer.

If you select:	Then do this on Numeric Keyboard	Then do this on Alphanumeric Keyboard
Middle Scanner Button	Squeeze the button on the Scan Handle	Squeeze the button on the Scan Handle
Left Scanner Button	Squeeze the left scanner button	Squeeze the left scanner button
Right Scanner Button	Squeeze the right scanner button	Squeeze the right scanner button
GOLD + A1 (Application 1)	Press [**Gold**] ◎ (. A1)	Press [**Gold/White**] ⊘ (A A1)
GOLD + A2 (Application 2)	Press [**Gold**] ◎ (4 GHI A2)	Press [**Gold/White**] ⊘ (B A2)

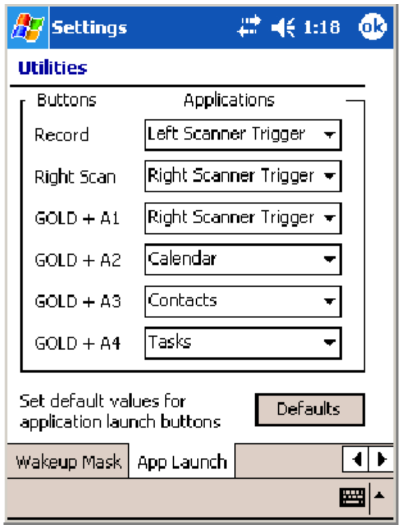
App Launch

Utilities

From the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Utilities** , then scroll to the right to tap the **App Launch** tab to access the Application Launch control panel applet.

This applet programs or maps two scanner buttons and four application keys to start up to six applications. *Note that the left scanner button also acts as the record button.*

For 700 Color Computers with either a laser scanner or an imager, default mappings are shown in the following illustration.



Buttons	Applications
Record	Left Scanner Trigger
Right Scan	Right Scanner Trigger
GOLD + A1	Right Scanner Trigger
GOLD + A2	Calendar
GOLD + A3	Contacts
GOLD + A4	Tasks

Set default values for application launch buttons

For 700 Color Computers without either a laser scanner or an imager, the default maps the Record, Calendar, Contacts, and Tasks applications the top four and the A3 and A4 buttons are ”unassigned” or available for two more applications.

Note: Record, Calendar, Contacts, and Tasks are Pocket PC applications. See Chapter 2, “*Windows Mobile 2003*,” for more information about these applications.

- To assign an application to a button, select an application from the applicable drop-down list box.
- To assign a new application, select the “Add new application” option, which brings up an Open File dialog and browse SD or CF storage cards for new applications.
- To disable or unmap a currently mapped application from a corresponding button, select “unassigned” from the applicable drop-down list.
- To restore these buttons to their defaults, tap **Defaults** in the lower right corner.

Note; You cannot map an application to more than one button. Should you assign the same application to two buttons, a verification prompt will appear after the second button to confirm whether you want to remap the application. If you tap **Yes**, the applet changes the first button to "unassigned" and map the application to the second button.

Note: All changes are activated immediately upon selection.

Wireless Network Control Panel Applet

Note: See Chapter 4, "*Network Support*," for information about the 802.11b or 802.11b/g radio module.

About the Wireless Network

Your wireless adapter (network interface card) connects to wireless networks of two types: infrastructure networks and ad-hoc networks.

- Infrastructure networks get you onto your corporate network and the internet. Your 700 Color Computer establishes a wireless connection to an access point, which links you to the rest of the network. When you connect to a network via an access point, you are using the 802.11b or 802.11b/g infrastructure mode.
- Ad-hoc networks are private networks shared between two or more clients, even with no access point.

Each wireless network is assigned a name (or Service Set Identifier — SSID) to allow multiple networks to coexist in the same area without infringement.

Intermec Technologies recommends using security measures with wireless networks to prevent unauthorized access to your network and to ensure your privacy of transmitted data. The following are required elements for secure networks:

- Authentication by both the network and the user
- Authentication is cryptographically protected
- Transmitted data

There are many schemes available for implementing these features.

Terminology

Below are terms you may encounter as you configure your wireless network:

- **CKIP** (Cisco Key Integrity Protocol)
  This is Cisco's version of the TKIP protocol, compatible with Cisco Airnet products.
- **EAP** (Extensible Authentication Protocol)
  802.11b or 802.11b/g uses this protocol to perform authentication. This is not necessarily an authentication mechanism, but is a common framework for transporting actual authentication protocols. Intermec Technologies provides a number of EAP protocols for you to choose the best for your network.

- **TKIP** (Temporal Key Integrity Protocol)
  This protocol is part of the IEEE 802.11i encryption standard for wireless LANs., which provides per-packet key mixing, a message integrity check and a re-keying mechanism, thus overcoming most of the weak points of WEP. This encryption is more difficult to crack than the standard WEP. Weak points of WEP include:

  - No Installation Vector (IV) reuse protection
  - Weak keys
  - No protection against message replay
  - No detection of message tampering
  - No key updates

- **WEP** (Wired Equivalent Privacy) **encryption**
  With preconfigured WEP, both the client 700 Color Computer and access point are assigned the same key, which can encrypt all data between the two devices. WEP keys also authenticate the 700 Color Computer to the access point — unless the 700 Color Computer can prove it knows the WEP key, it is not allowed onto the network.

  WEP keys are only needed if they are expected by your clients. There are two types available: 64-bit (5-character strings, 12345) (default) and 128-bit (13-character strings, 1234567890123). Enter these as either ASCII (12345) or Hex (0x3132333435).

- **WPA** (Wi-Fi Protected Access)
  This is an enhanced version of WEP that does not rely on a static, shared key. It encompasses a number of security enhancements over WEP, including improved data encryption via TKIP and 802.11b or 802.11b/g authentication with EAP.

  WiFi Alliance security standard is designed to work with existing 802.11 products and to offer forward compatibility with 802.11i.

Configuring Your Wireless Network

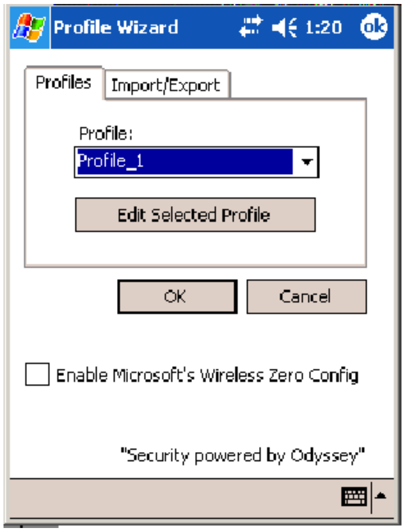
Wireless Network

To start 802.11b or 802.11b/g communications on the 700 Color Computer, tap **Start** > **Settings** > the **System** tab > **Wireless Network** to access the Profile Wizard for the 802.11b or 802.11b/g radio module.

A profile contains all the information necessary to authenticate you to the network, such as login name, password or certificate, and protocols by which you are authenticated.

You can have up to four profiles for different networks. For example, you may have different login names or passwords on different networks, or you may use a password on one network, and a certificate on another.

Use the Profiles page to select and configure between the networking environments assigned to this 802.11b or 802.11b/g radio.



Profile: Profile_1

□ Enable Microsoft's Wireless Zero Config

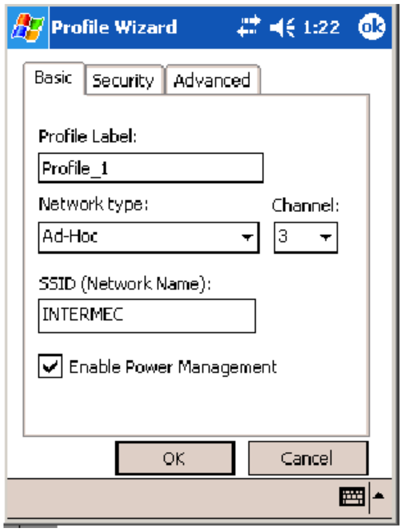
"Security powered by Odyssey"

- **Profile**:
  Tap the drop-down list to choose between four different profiles assigned to this unit, then tap **Edit Select Profile**, make the changes needed for this profile *(starting on the next page)*, then tap **OK** to return to the Profiles page.
- **Enable Microsoft's Wireless Zero Config**
  Check this box to enable Microsoft's Wireless Zero Config application. This effectively disables the Intermec software solution for 802.11b or 802.11b/g, including configuration via the CORE application and the Wireless Network control panel applet.

Basic

Use the Basic page to set the network type, name, and manage battery power for this profile. Tap **ok** or **OK** to return to the Profiles page.

- **Profile Label**:
  Enter a unique name for your profile.
- **Network type**:
  Tap the drop-down list to select either “Infrastructure” if your network uses access points to provide connectivity to the corporate network or internet; or “Ad-Hoc” to set up a private network with one or more participants.
- **Channel**:
  If you selected “Ad-Hoc” for the network type, select the channel on which you are communicating with others in your network. There are up to 11 channels available.
- **SSID (Network Name)**:
  This assumes the profile name *unless another name is entered in this field*. If you want to connect to the next available network or are not familiar with the network name, enter “ANY” in this field. Consult your LAN administrator for network names.
- **Enable Power Management**:
  Check this box to conserve battery power (default), or clear this box to disable this feature.



Profile Label: Profile_1

Network type: Ad-Hoc

Channel: 3

SSID (Network Name): INTERMEC

☑ Enable Power Management

Security

The following are available from the **8021x Security** drop-down list. *Note that the last four methods are available if you have purchased the security package. Contact your Intermec representative for information.*

- None *(below)*
- PEAP *(page 378)*
- TLS *(page 382)*
- TTLS *(page 385)*
- LEAP *(page 389)*

None

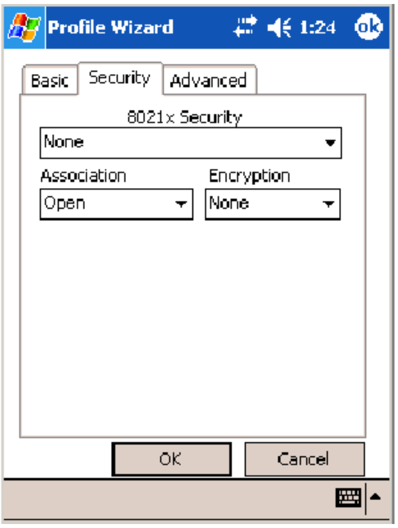
Use "None" to disable 802.11b or 802.11b/g Security and enable either WEP or WPA-PSK encryption.

To Disable 802.1x Security

1 Set **8021x Security** as "None."

2 Set **Association** to "Open."

3 Set **Encryption** to "None."



8021x Security: None

Association: Open

Encryption: None

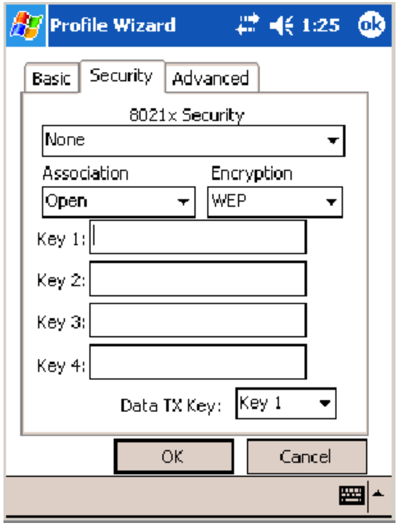
To Enable WEP Encryption

1 Set **8021x Security** as “None.”

2 Set **Association** to either “Open” if WEP keys are not required; or “Shared” when WEP keys are required for association.

3 Set **Encryption** to “WEP.” See page 372 for information about WEP encryption.

4 If you had set **Association** to “Shared,” then select a data transmission key from the **Data TX Key** drop-down list near the bottom of this screen, then enter the encryption key for that data transmission in the appropriate **Key #** field.



8021x Security: None

Association: Open

Encryption: WEP

Key 1:

Key 2:

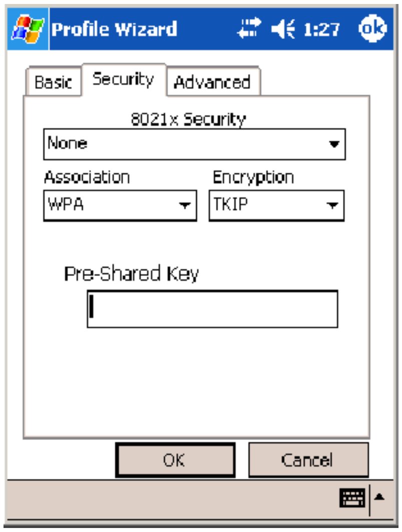
Key 3:

Key 4:

Data TX Key: Key 1

To Enable WPA Encryption Using a Preshared Key

1. Set **8021x Security** as “None.”
2. Set **Association** to “WPA.” See page 372 for information about WPA encryption.
3. Skip **Encryption** as it is automatically set to “TKIP.” See page 372 for more information about TKIP.
4. Enter the temporal key as ASCII (12345) in the **Pre-Shared Key** field.



8021x Security: None

Association: WPA

Encryption: TKIP

Pre-Shared Key

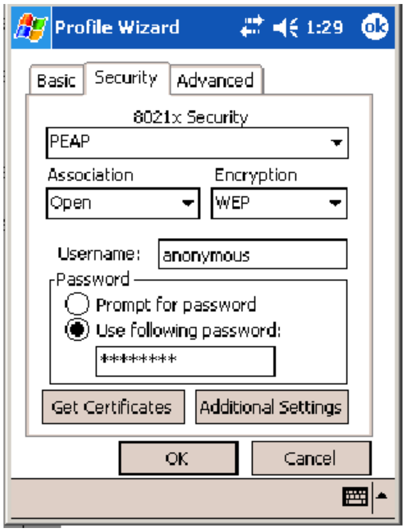
PEAP (Protected EAP)

This protocol is suitable for performing secure authentication against Windows domains and directory services. It is comparable to EAP-TTLS (see page 385), both in its method of operation and its security, though not as flexible. This does not support the range of inside-the-tunnel authentication methods supported by EAP-TTLS. Microsoft and Cisco both support this protocol.

Use “PEAP” to configure the use of PEAP as an authentication protocol and to select “Open,” “WPA,” or “Network EAP” as an association mode.

To Enable PEAP with an Open Association

1. Set **8021x Security** as “PEAP.”
2. Set **Association** to “Open.”
3. Skip **Encryption** as it is automatically set to “WEP.” See page 372 for information about WEP encryption.
4. Enter your unique user name and password to use this protocol. Select **Prompt for password** to have the user enter this password each time to access the protocol; or leave **Use following password** as selected to automatically use the protocol without entering a password.
5. Tap **Get Certificates** to obtain or import server certificates. See page 388 for more information.
6. Tap **Additional Settings** to assign an inner PEAP authentication and set options for server certificate validation and trust. See page 381 for more information.



8021x Security: PEAP

Association: Open

Encryption: WEP

Username: anonymous

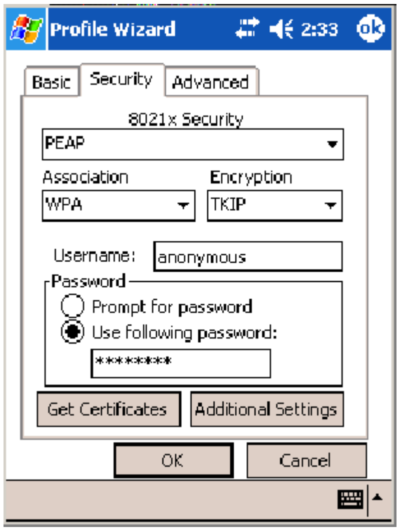
Password

○ Prompt for password

● Use following password: ********

To Enable PEAP with WPA Encryption

1. Set **8021x Security** as "PEAP."
2. Set **Association** to "WPA." See page 372 for information about WPA encryption.
3. Skip **Encryption** as it is automatically set to "TKIP." See page 372 for more information about TKIP.
4. Enter your unique user name and password to use this protocol. Select **Prompt for password** to have the user enter this password each time to access the protocol, or leave **Use following password** as selected to automatically use the protocol without entering a password.
5. Tap **Get Certificates** to obtain or import server certificates. See page 388 for more information.
6. Tap **Additional Settings** to assign an inner PEAP authentication and set options for server certificate validation and trust. See page 381 for more information.



8021x Security: PEAP

Association: WPA

Encryption: TKIP

Username: anonymous

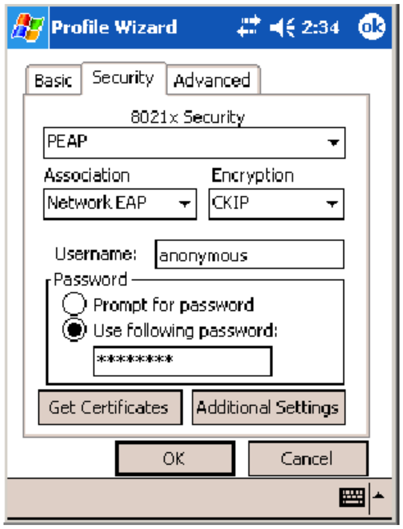
Password

○ Prompt for password

● Use following password: ********

To Enable PEAP with Network EAP

1. Set **8021x Security** as "PEAP."
2. Set **Association** to "Network EAP." See page 371 for information about EAP.
3. Set **Encryption** to either "WEP" or "CKIP." See page 371 for information about CKIP and page 372 for information about WEP encryption.
4. Enter your unique user name and password to use this protocol. Select **Prompt for password** to have the user enter this password each time to access the protocol, or leave **Use following password** as selected to automatically use the protocol without entering a password.
5. Tap **Get Certificates** to obtain or import server certificates. See page 388 for more information.
6. Tap **Additional Settings** to assign an inner PEAP authentication and set options for server certificate validation and trust. See page 381 for more information.



8021x Security: PEAP

Association: Network EAP

Encryption: CKIP

Username: anonymous

Password
○ Prompt for password
● Use following password:
****

Additional Settings

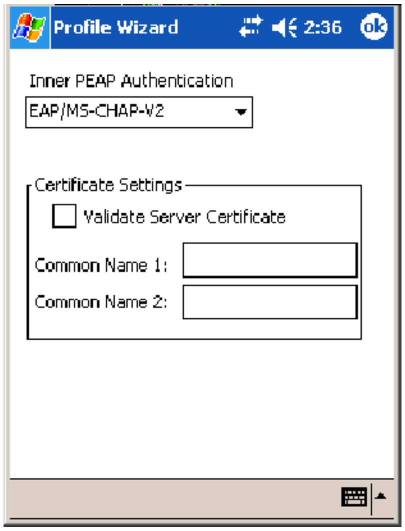
1 Select an authentication method from the **Inner PEAP Authentication** drop-down list.

EAP/MS-CHAP-V2	Authenticates against a Windows Domain Controller and other non-Windows user databases. This is Microsoft's implementation of PEAP.
EAP/Token Card	Use with token cards. The password value entered is never cached. This is Cisco's implementation of PEAP.
EAP/MD5-Challenge	Message Digest 5. A secure hashing authentication algorithm.

2 Check **Validate Server Certificate** to verify the identity of the authentication server based on its certificate when using TTLS or PEAP.

3 Enter the **Common Names** of trusted servers. *Note that if these fields are left blank, the server certificate trust validation is not performed or required.*

4 Click **ok** to return to the Security page.



Inner PEAP Authentication
EAP/MS-CHAP-V2

Certificate Settings
□ Validate Server Certificate
Common Name 1:
Common Name 2:

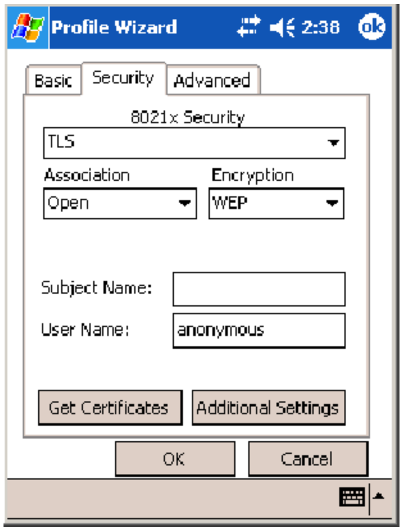
TLS (EAP-TLS)

EAP-TLS is a protocol that is based on the TLS (Transport Layer Security) protocol widely used to secure web sites. This requires both the user and authentication server have certificates for mutual authentication. While cryptically strong, this requires corporations that deploy this to maintain a certificate infrastructure for all their users.

Use “TLS” to configure the use of EAP-TLS as an authentication protocol, and select either “Open” or “WPA” as an association mode.

To Enable TLS with an Open Association

1. Set **8021x Security** as “TLS.”
2. Set **Association** to “Open.”
3. Skip **Encryption** as it is automatically set to “WEP.” See page 372 for information about WEP encryption.
4. Enter your unique **Subject Name** and **User Name** to use this protocol.
5. Tap **Get Certificates** to obtain or import server certificates. See page 388 for more information.
6. Tap **Additional Settings** to set options for server certificate validation and trust. See page 384 for more information.



8021x Security: TLS

Association: Open

Encryption: WEP

Subject Name:

User Name: anonymous

To Enable TLS with WPA Encryption

1 Set **8021x Security** as "TLS."

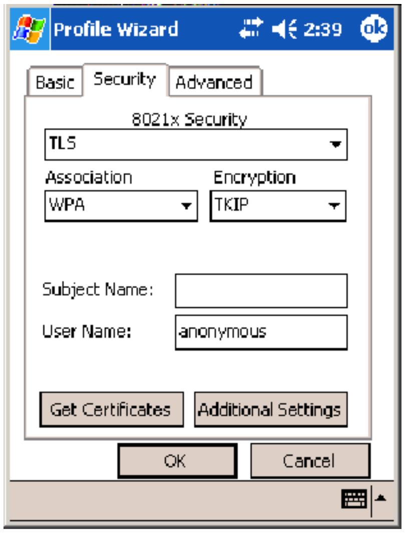
2 Set **Association** to "WPA." See page 372 for information about WPA encryption.

3 Skip **Encryption** as it is automatically set to "TKIP."See page 372 for more information about TKIP.

4 Enter your unique **Subject Name** and **User Name** as credentials for this profile.

5 Tap **Get Certificates** to obtain or import server certificates. See page 388 for more information.

6 Tap **Additional Settings** to set options for server certificate validation and trust. See page 384 for more information.



8021x Security: TLS

Association: WPA

Encryption: TKIP

Subject Name:

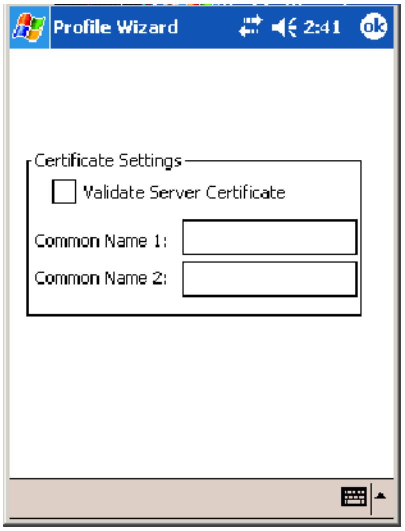
User Name: anonymous

Additional Settings

1 Check **Validate Server Certificate** to verify the identity of the authentication server based on its certificate when using PEAP or TLS.

2 Enter the **Common Names** of trusted servers. *Note that if these fields are left blank, the server certificate trust validation is not performed or required.*

3 Click **ok** to return to the Security page.



Certificate Settings

□ Validate Server Certificate

Common Name 1:

Common Name 2:

TTLS (EAP-Tunneled TLS)

This protocol provides authentication like EAP-TLS (see page 382) but does not require certificates for every user. Instead, authentication servers are issued certificates. User authentication is done using a password or other credentials that are transported in a securely encrypted "tunnel" established using server certificates.

EAP-TTLS works by creating a secure, encrypted tunnel through which you present your credentials to the authentication server. Thus, inside EAP-TTLS there is another *inner authentication protocol* that you must configure via Additional Settings.

Use "TTLS" to configure the use of EAP-TTLS as an authentication protocol, and select either "Open" or "WPA" as an association mode.

To Enable TTLS with an Open Association (default configuration)

1 Set **8021x Security** as "TTLS."

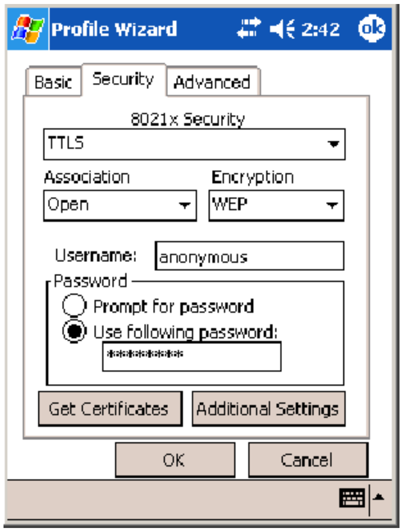
2 Set **Association** to "Open."

3 Skip **Encryption** as it is automatically set to "WEP." See page 372 for information about WEP encryption.

4 Enter your unique user name and password to use this protocol. Select **Prompt for password** to have the user enter this password each time to access the protocol, or leave **Use following password** as selected to automatically use the protocol without entering a password.

5 Tap **Get Certificates** to obtain or import server certificates. See page 388 for more information.

6 Tap **Additional Settings** to assign an inner TTLS authentication and an inner EAP, and set options for server certificate validation and trust. See page 387 for more information.



8021x Security: TTLS

Association: Open

Encryption: WEP

Username: anonymous

Password
○ Prompt for password
● Use following password: *********

To Enable TTLS with WPA Encryption

1 Set **8021x Security** as "TTLS."

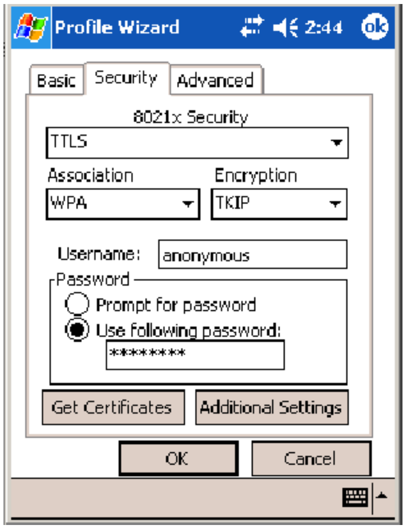
2 Set **Association** to "WPA." See page 372 for information about WPA encryption.

3 Skip **Encryption** as it is automatically set to "TKIP." See page 372 for more information about TKIP.

4 Enter your unique user name and password to use this protocol. Select **Prompt for password** to have the user enter this password each time to access the protocol, or leave **Use following password** as selected to automatically use the protocol without entering a password.

5 Tap **Get Certificates** to obtain or import server certificates. See page 388 for more information.

6 Tap **Additional Settings** to assign an inner TTLS authentication and an inner EAP, and set options for server certificate validation and trust. See page 387 for more information.



8021x Security: TTLS

Association: WPA

Encryption: TKIP

Username: anonymous

Password

○ Prompt for password

● Use following password: *********

Additional Settings

1 Select an authentication protocol from the **Inner TTLS Authentication** drop-down list:

PAP	Password Authentication Protocol. A simple authentication protocol that sends security information in the clear.
CHAP	Challenge Handshake Authentication Protocol. Use of Radius to authenticate a terminal without sending security data in the clear. Authenticates against non-Windows user databases. *You cannot use this if authenticating against a Windows NT Domain or Active Directory.*
MS-CHAP; MS-CHAP-V2	Authenticates against a Windows Domain Controller and other non-Windows user databases.
PAP/Token Card	Use with token cards. The password value entered is never cached.
EAP	Extensible Authentication Protocol. See page 371 for information about EAP.

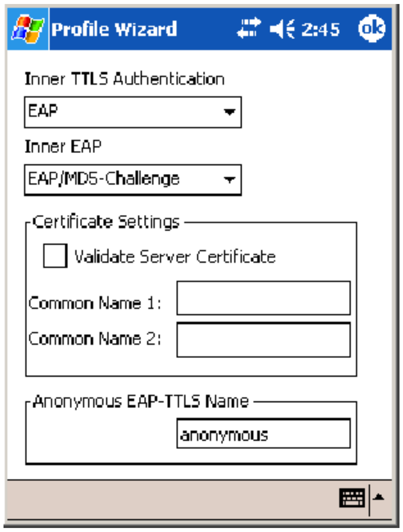
2 If you select "EAP" for the inner authentication protocol, then select an inner EAP protocol from the **Inner EAP** drop-down list.

3 Enter the **Common Names** of trusted servers. *Note that if these fields are left blank, the server certificate trust validation is not performed or required.*

4 Check **Validate Server Certificate** to verify the identity of the authentication server based on its certificate when using TTLS, PEAP, and TLS.

5 Enter the **Anonymous EAP-TTLS Name** as assigned for public usage. Use of this outer identity protects your login name or identity.

6 Click **ok** to return to the Security page.



Profile Wizard 2:45 ok

Inner TTLS Authentication: EAP

Inner EAP: EAP/MD5-Challenge

Certificate Settings

□ Validate Server Certificate

Common Name 1:

Common Name 2:

Anonymous EAP-TTLS Name: anonymous

To Get Certificates

Certificates are pieces of cryptographic data that guarantee a public key is associated with a private key. They contain a public key and the entity name that owns the key. Each certificate is issued by a certificate authority.

Use this page to import a certificate onto the 700 Color Computer.

Root Certificates

1 Tap the <<< button next to the **Import Root Certificate** field to select the root certificate (DER-encoded .CER file) to import.

2 Click **Import Root Cert** to install the selected certificate.

User Certificate

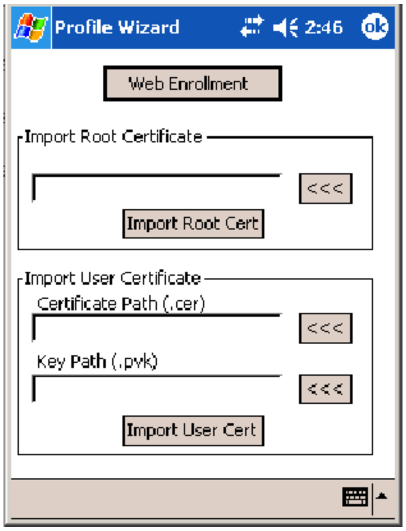
1 Tap the <<< button next to the **Certificate Path** field to select the user certificate (DER-encoded .CER file without the private key) to import.

2 Tap the <<< button next to the **Key Path** field to select the private key (.PVK file) which corresponds to the user certificate chosen in step 1.

3 Tap **Import User Cert** to install the selected certificate.

Web Enrollment

Tap **Web Enrollment** to obtain a user certificate over the network from an IAS Server. Tap **ok** to return to the Security page.



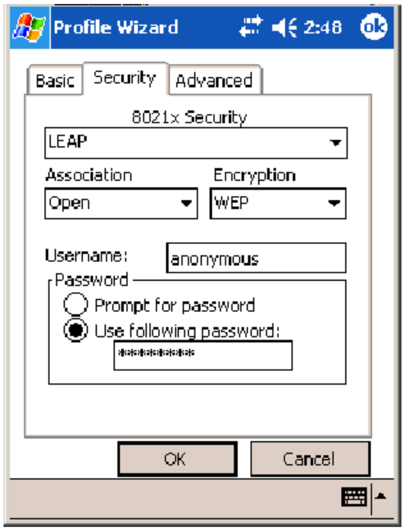
LEAP (Cisco Lightweight EAP)

LEAP is the Cisco Lightweight version of EAP. See page 371 for information about EAP.

Use “LEAP” to configure the use of LEAP as an authentication protocol, select “Open,” “WPA,” or “Network EAP” as an association mode, or assign Network EAP. *Note that this defaults to the Network EAP.*

To Enable LEAP with an Open Association

1. Set **8021x Security** as “LEAP.”
2. Set **Association** to “Open.”
3. Skip **Encryption** as it is automatically set to “WEP.” See page 372 for information about WEP encryption.
4. Enter your unique **User Name** to use this protocol.
5. Select **Prompt for password** to have the user enter this password each time to access the protocol, or leave **Use following password** as selected to automatically use the protocol without entering a password.



8021x Security: LEAP

Association: Open

Encryption: WEP

Username: anonymous

Password

○ Prompt for password

● Use following password: *********

To Enable LEAP with WPA Encryption

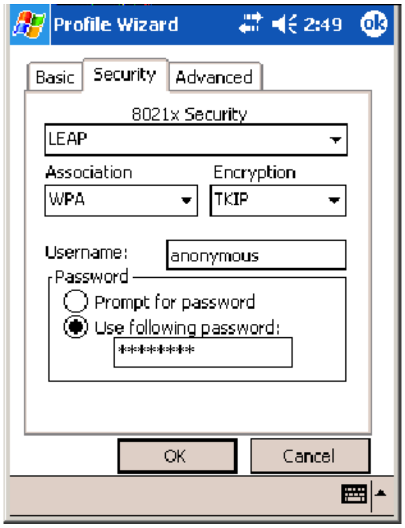
1 Set **8021x Security** as “LEAP.”

2 Set **Association** to “WPA.” See page 372 for information about WPA encryption.

3 Skip **Encryption** as it is automatically set to “TKIP.” See page 372 for more information about TKIP.

4 Enter your unique **User Name** to use this protocol.

5 Select **Prompt for password** to have the user enter this password each time to access the protocol, or leave **Use following password** as selected to automatically use the protocol without entering a password.



8021x Security: LEAP

Association: WPA

Encryption: TKIP

Username: anonymous

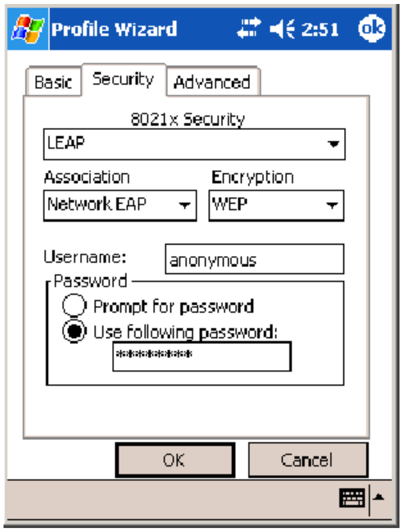
Password

○ Prompt for password

● Use following password: ********

To Enable LEAP with Network EAP

1. Set **8021x Security** as “LEAP.”
2. Set **Association** to “Network EAP,” an EAP protocol for the network. See page 371 for information about EAP.
3. Set **Encryption** to either “WEP” or “CKIP.” See page 371 for information about CKIP and page 372 for information about WEP encryption.
4. Enter your unique **User Name** to use this protocol.
5. Select **Prompt for password** to have the user enter this password each time to access the protocol, or leave **Use following password** as selected to automatically use the protocol without entering a password.



8021x Security: LEAP

Association: Network EAP

Encryption: WEP

Username: anonymous

Password

○ Prompt for password

● Use following password: *********

Advanced

Use this page to configure additional settings for this profile. Tap **ok** or **OK** to return to the Profiles page.

- **Detect Rogue APs**:
  Wireless NICs and APs associate based on the SSID configured for the NIC. Given an SSID, the BSSID with the strongest signal is often chosen for association. After association, 802.1x authentication may occur and during authentication credentials to uniquely identify a user — these are passed between the NIC and the AP.

  The base 802.1x technology does not protect the network from "rogue APs." These can mimic a legitimate AP to authentication protocols and user credentials. This provides illegal users ways to mimic legitimate users and steal network resources and compromise security.

  Check this box to detect and report client behavior suspected of being rogue APs. Once a rouge AP is detected, your 700 Series Computer no longer associates with that AP until you perform a warm boot.

  Clear this box to solve AP connection problems that result when an AP gets put on the rogue AP list due to inadvertant failed authentications and *not* because it is a *real* rouge.

- **Enable mixed cell**:
  Mixed cell is a profile-dependent setting. If enabled, you can connect to mixed cell without using WEP, then you can query the cell to determine whether you can use encryption.

- **Enable Logging**:
  Check this box to log what activity incurs for this profile.

Other Configurable Parameters

The following parameters can be configured by sending reader commands through the network or from an application. See "*Using Reader Commands*" on page 394 for more information.

Reader Command	Description	Options
Audio Volume	Changes the volume of all audio signals.	0 Off 1 Very quiet 2 Quiet 3 Normal (default) 4 Loud 5 Very loud
Automatic Shutoff	Sets the length of time the 700 Color Computer remains on when there is no activity. When you turn on the 700 Color Computer, it either resumes exactly where it was when you turned it off or boots and restarts your application.	1 1 minute 2 2 minutes 3 3 minutes (default) 4 4 minutes 5 5 minutes
Backlight Timeout	Sets the length of time that the display backlight remains on. If you set a longer timeout value, you use the battery power at a faster rate.	10 10 seconds 30 30 seconds 60 1 minute (default) 120 2 minutes 180 3 minutes 240 4 minutes 300 5 minutes
Date/Time	Sets the current date and time.	Date Year 0000–9999 *(1999)* Month 1–12 *(6)* Day 1–31 *(1)* Time Hour 0–23 *(0)* Minute 0–59 *(00)* Second 0–59 *(00)*
Key Clicks	Enables or disables the keypad clicks. The 700 Color Computer emits a click each time you press a key or decode a row of a two-dimensional symbology.	0 Disable clicks 1 Enable soft key clicks 2 Enable loud key clicks (default)

Using Reader Commands

After the 700 Color Computer is connected to your network, you can send the 700 Color Computer a reader command from an application to perform a task, such as changing the time and date. Some reader commands temporarily override the configuration settings and some change the configuration settings.

Change Configuration

The Change Configuration command must precede any configuration command. If you enter a valid string, the 700 Color Computer configuration is modified and the computer emits a high beep. To send the Change Configuration command through the network, use the `$+ [`*`command`*`]` syntax where *command* is the two-letter command syntax for the configuration command followed by the value to be set for that command.

You can also make changes to several different commands by using the `$+ [command]...[command n]` syntax. There are seven configuration command settings that can be changed in this way. *See each command for information on respective acceptable "data" values.*

Command	Syntax
Audio Volume	BV*data*
Automatic Shutoff	EZ*data*
Backlight Timeout	DF*data*
Key Clicks	KC*data*
Virtual Wedge Grid	AF*data*
Virtual Wedge Postamble	AE*data*
Virtual Wedge Preamble	AD*data*

Note: See pages 344 and 346 for more information about the **Virtual Wedge Postamble** and **Virtual Wedge Preamble** commands.

Example 1

To change the Beep Volume to Off, you can send this string to the 700 Color Computer through the network: `$+BV0`
where:
`$+` Indicates Change Configuration.
`BV` Specifies the Audio Volume parameter.
`0` Specifies a value of Off.

Example 2

To change the Beep Volume to Very Quiet and the Virtual Wedge Grid to 123: `$+BV1AF123`
where:
`$+` Indicates Change Configuration
`BV1` Specifies Audio Volume, set to Very Quiet (1)
`AF123` Specifies Virtual Wedge Grid, set to a value of 123.

Set Time and Date

This command sets the date and time on the 700 Color Computer. The default date and time is *June 1, 1999 at 12:00 AM*.

From the network, send the following:

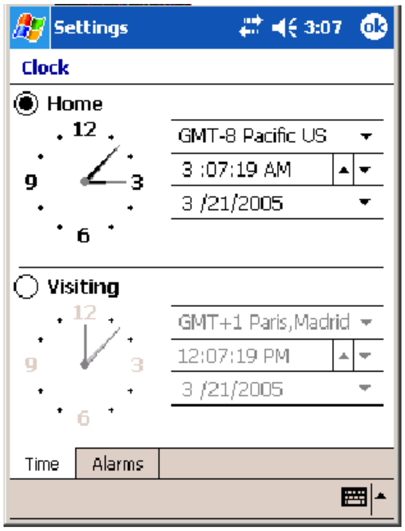
```
/+ yyyymmddhhmmss
```

where acceptable values for the date are:

`yyyy`	0000–9999	Year
`mm`	01–12	Month of the year
`dd`	01–31	Day of the month
`hh`	00–23	Hour
`mm`	00–59	Minutes
`ss`	00–59	Seconds

Clock

You can also set the time and date by using Configuration Management in Unit Manager, or by using the Clock control panel applet in the Settings menu. To access this control panel applet, tap **Start** > **Settings** > the **System** tab > the **Clock** icon to access its control panel applet.



Clock

● Home

GMT-8 Pacific US

3 :07:19 AM

3 /21/2005

○ Visiting

GMT+1 Paris,Madrid

12:07:19 PM

3 /21/2005

Configuration Bar Codes

You can change some settings on your 700 Color Computer by scanning the following Code 39 bar code labels.

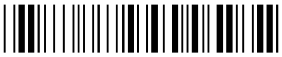
- You can use the Data Collection control panel to set the three Virtual Wedge parameters *(starting on page 343).*

Note: When you use a bar code creation utility to make a scannable bar code label, the utility probably adds opening and closing asterisks automatically. Asterisks are included here for translation purposes.

Audio Volume

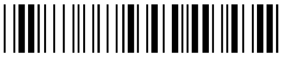
Note: The Audio Volume parameter information is on page 393.

Turn Audio Off



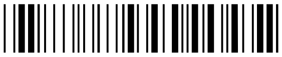
$+BV0

Set Audio Volume to very quiet



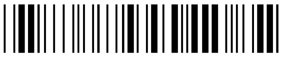
$+VB1

Set Audio Volume to quiet



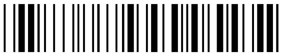
$+BV2

Set Audio Volume to normal *(default)*



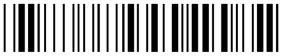
$+BV3

Set Audio Volume to loud



$+BV4

Set Audio Volume to very loud



$+BV5

Automatic Shutoff

Note: The Automatic Shutoff parameter information is on page 393.

Set Automatic Shutoff to 1 minute

$+EZ1

Set Automatic Shutoff to 2 minutes

$+EZ2

Set Automatic Shutoff to 3 minutes *(default)*

$+EZ3

Set Automatic Shutoff to 4 minutes

$+EZ4

Set Automatic Shutoff to 5 minutes

$+EZ5

Backlight Timeout

Note: The Backlight Timeout parameter information is on page 393.

Backlight Timeout 10 seconds

$+DF10

Backlight Timeout 30 seconds

$+DF30

Backlight Timeout 1 minute *(default)*

$+DF60

Backlight Timeout 2 minutes

$+DF120

Backlight Timeout 3 minutes

$+DF180

Backlight Timeout 4 minutes

$+DF240

Backlight Timeout 5 minutes

$+DF300

Key Clicks

Note: The Key Clicks parameter information is on page 393.

Disable key clicks

$+KC0

Enable soft key clicks

$+KC1

Enable loud key clicks *(default)*

$+KC2

Virtual Wedge Grid, Preamble, Postamble

The following parameters are user-configurable strings. Refer to a full ASCII chart for more information.

Grid

For Virtual Wedge Grid, the first part of the bar code would be the following, which can include a string of up to 240 characters. *Parameter information starts on page 348.*

*$+AF

Preamble

For Virtual Wedge Preamble, the first part of the bar code would be below, followed by a string of up to 31 characters *(no <NUL>)* and an asterisk. *Default is no characters. Parameter information is on page 344.*

*$+AD

Postamble

For Virtual Wedge Postamble, the first part of the bar code would be below, followed by a string of up to 31 characters *(no <NUL>)* and an asterisk. *Default is no characters. Parameter information is on page 346.*

*$+AE

B Troubleshooting

This appendix contains a brief explanation of what you can do to troubleshoot your 700 Color Mobile Computer.

The 700 Color Computer does not respond when connected to a power supply.

Make sure your ac adapter or dock is plugged in and is making good contact with your mobile computer.

The 700 Color Computer will not turn on when the I/O key is pressed.

The battery may be low and need recharging.

The Power status LED starts blinking.

The Power status LED provides the status of your battery pack when it is connected to a charging device. See Chapter 1, "Introduction," for LED descriptions.

The 700 Color Computer will not turn on and the screen is blank.

The battery may be *critically low*. Put your mobile computer in a charger for at least five minutes, remove it from the dock, then press the I/O button. The screen should turn on.

Put your mobile computer back in the charger and continue to charge your mobile computer for at least four hours to make sure the battery is fully charged.

The 700 Color Computer will not turn on when placed in a dock.

Make sure the dock is plugged in and your mobile computer is securely seated in the dock.

The 700 Color Computer shuts down during operation.

You may have a very low battery. Put the battery in a charger.

The 700 Color Computer does not turn off.

The 700 Color Computer does not turn off while it is processing data. If this situation continues for a long period of time, it does run down the battery unless you have it connected to an external power, or in a charging dock.

Should the 700 Color Computer lock up, perform a warm-boot. If the mobile computer stays locked up, then perform a cold-boot. See Chapter 1, "Introduction," for information on performing warm-boots and cold-boots.

The 700 Color Computer takes a long time to boot up after a warm- or cold-boot is performed.

Normal recovery time is 30 to 45 seconds. If the recovery takes longer, then contact support personnel.

I Index

The Classes and Functions Index covers classes and functions for the 700 Series Color Mobile Computer.

The General Index covers all topics. Those in italics are figures, those in bold are tables.

The Files Index is to assist you in locating descriptions for device drivers, applications, utilities, batch files, or other files within this publication.

Classes and Functions

General Index

Files Index

Intermec

Corporate Headquarters
6001 36th Avenue West
Everett, Washington 98203
U.S.A.

tel 425.348.2600

fax 425.355.9551

www.intermec.com