1. NDIS Network Drivers

Windows CE Platform Builder version 2.0 and later supports the Network Driver Interface Specification (NDIS). NDIS refers to the interface by which one or more local area network (LAN) and wide area network (WAN) adapter drivers communicate with one or more underlying network adapters, with one or more overlying protocol drivers, and with the operating system. The Windows CE communications architecture provides support for NDIS version 4.0. Windows CE version 3.0 supports an array of new features that include intermediate drivers, NDISWAN, Toen Ring, an improved binding of adapters, a layered miniport structure, and an enhanced testing application (NDISTest).

The network driver interface specification (NDIS) is the mechanism by which the Windows CE operating system (OS) supports network connectivity. NDIS provides a pair of abstraction layers that are used to connect networking drivers to protocol stacks, such as TCP/IP and Infrared Data Association (IrDA), and to network adapters, such as Ethernet cards. NDIS presents two sets of application programming interfaces (APIs) for writers of network drivers: one set interfaces to the networking protocol stacks and one set interfaces to network interface cards (NICs).

Windows CE versions 2.0 and later implement a subset of the NDIS 4.0 model that is used by Microsoft Windows NT, enabling OEMs and independent hardware vendors (IHVs) to port existing Windows NT networking drivers to Windows CE. The full NDIS supports several types of network drivers, but Windows CE versions 2.0 and later support only miniport drivers and intermediate drivers, but not monolithic or full NIC drivers.

For miniport drivers, Windows CE is largely source-code-compatible with Windows NT. This means that, with a few exceptions, Windows CE and Windows NT support identical NDIS APIs. Consult the Microsoft Windows NT Device Driver Kit for extended information on how to write a miniport driver. Because full documentation is available in the Microsoft Windows NT Device Driver Kit, this documentation does not discuss at length the process of writing miniport drivers. Miniport drivers are complex pieces of software, and for this reason Microsoft recommends that you adapt one of the sample miniport drivers or port an existing miniport driver from another OS, such as Windows NT, rather than writing one from scratch.

For a complete list of the NDIS APIs that are supported on Windows CE, including information on the minor differences between the Windows CE API and its Windows NT counterpart, consult the Microsoft Windows CE API Reference.

2. Introduction to NDIS

The Network Driver Interface Specification (NDIS) describes the interface by which one or more network adapter drivers communicate with one or more underlying network adapters, with one or more overlying protocol drivers, with one or more miniport drivers, and with the operating system. NDIS provides a fully abstracted interface for network adapter-driver development.
NDIS provides a pair of abstraction layers that are used to connect network drivers to an overlying protocol stack, such as Transmission Control Protocol/Internet Protocol (TCP/IP) or Infrared Data Association (IrDA) and an underlying network adapter. NDIS performs a set of external functions for network adapter drivers, such as registering and intercepting hardware interrupts and communicating with underlying network adapters.

The Windows library (Ndis.dll) provides a fully abstracted interface to which you can write a customized network adapter driver for Windows CE. The library exports all of the Windows CE Kernel-mode functions that are required for driver development. The Ndis.dll file also maintains binding and state information about all of the underlying network adapter drivers. NDIS supports the following network components:

- A network adapter driver that receives a network packet from an upper-layer driver for transmission on the network through an underlying network adapter
- A network adapter driver that accepts a network packet from an underlying network adapter and passes the packet up to an upper-layer driver
- An upper-layer driver that sets specific configuration parameters for a network adapter or a network adapter driver
- An upper-layer driver that queries a network adapter driver for specific configuration data from an underlying network adapter or from the network adapter driver
- A network adapter driver that informs an overlying driver asynchronously of changes in the status of the network or of the network adapter

The following diagram shows the general NDIS architecture that is implemented in Windows-based platforms.

The NDIS interface is located between an upper-level protocol driver, such as the TCP/IP protocol driver on the top of the communications architecture, the intermediate and miniport drivers in the middle of the communications architecture, and a network adapter at the bottom of the communications architecture. Because NDIS provides an interface to the upper and lower edges of a miniport driver, the NDIS interface often is referred to as the NDIS Wrapper. The NDIS Wrapper provides the operating environment for drivers that use NDIS; its components are located in the Ndis.dll file.

Microsoft Windows CE 3.0

**3. Windows CE Communications Architecture**

One of the key features of Windows CE-based devices is the ability to communicate with other devices. Windows CE supports two basic types of communication: serial communication and communication over a network. Most devices feature built-in communications hardware, such as a serial port or an IR transceiver. The NDIS implementation on Windows CE supports the following communications media: Ethernet (802.3), Token Ring (802.5), IrDA, and WAN. The following diagram outlines the communications architecture of the Windows CE operating system.
In the Windows CE communications architecture, the NDIS interface is located below the IrDA, TCP/IP, and Point-to-Point Protocol (PPP) protocol drivers. The NDIS Wrapper presents an interface to the upper and lower edges of a miniport driver. To an upper-level driver, such as the TCP/IP protocol driver, the NDIS interface looks like a miniport driver. To the miniport, the NDIS interface looks like an upper-level protocol driver. On the bottom of the communications architecture, the NDIS interface functions as a network adapter driver that interfaces directly with the network adapter at the lower edge. At the upper edge, the network adapter driver presents an interface to allow upper layers to send packets on the network, handle interrupts, reset or halt the network adapter, and query or set the operational characteristics of the driver.

4. Windows CE Implementation of NDIS

Windows CE 3.0 supports NDIS 4.0, which also is implemented in Windows NT 4.0. The NDIS implementation on Windows CE and Windows NT 4.0, with some minor exceptions, is source-code compatible. NDIS miniport drivers are fully portable, and you should be able to port an existing miniport driver from Windows NT to a Windows CE-based device. You also can create a customized miniport driver for a Windows CE-based device by following the steps that are provided in the Windows CE driver development kit (DDK) on how to write a miniport driver. Because Windows CE does not require backward compatibility with legacy drivers, it does not support full NIC drivers or monolithic network drivers.

The following list shows the key features of NDIS support in Windows CE 3.0:

- Portability of drivers between platforms that support NDIS
5. NDIS Differences Between Windows CE and Windows NT

Although the NDIS implementation on Windows CE and Windows NT is source-code compatible, and although both operating systems, with a few exceptions, support identical APIs, there are some differences in the implementation of NDIS between Windows CE and Windows NT:

- Windows NT and Windows CE share the same source code for miniport drivers, but each uses a different compilation method. In Windows NT, miniport drivers are compiled as system (.sys) files. Because Windows CE does not support system files, you must compile a miniport driver as a dynamic-link library (DLL). To compile a miniport driver as a DLL, the driver must export the `DriverEntry` function in its .def file. `DriverEntry` initializes the driver and then calls other functions in the DLL to perform platform-specific initializations.

- Windows CE does not support .inf files for dynamic installation and configuration of NDIS miniport drivers.

- Windows CE does not provide a built-in support for Direct Memory Access (DMA) functionality. However, developers can implement DMA code in a miniport driver for a specific combination of a Windows CE-based platform and a network adapter. Windows CE does not support contiguous physical memory allocations. This memory must be allocated in a platform-specific way.

- The NDIS wrapper interface (Ndis.dll) that is supported in Windows CE does not make any calls to the multipacket send handler in the miniport driver. However, a miniport driver's regular send handler should return `RESET_IN_PROGRESS` if the send handler is invoked while the adapter is being reset.

- The I/O port address that is returned from the `NdisMRegisterIoPortRange` function represents a 32-bit virtual address in Windows CE and should be treated as a 32-bit value. Because Windows CE does not use the lower 64-KB region of the I/O address space, you should not use 16-bit values for I/O port addresses. Many Windows CE devices emulate I/O ports on top of mapped memory.

- The Windows CE implementation of NDIS does not provide access to the PCMCIA attribute memory. A driver instead must use PC Card Services functions to read the card tuple information.

- The IrDA protocol driver supports only a single statically bound adapter.

- The `NdisSendPackets` function is not supported in Windows CE.
• Windows CE does not prevent a miniport driver's **MiniportHandleInterrupt** function from being called before the miniport driver's **PacketIndicateHandler** function is registered.

Microsoft Windows CE 3.0

### 6. Windows CE Miniport-Driver Architecture

Miniport drivers directly manage network adapters that are installed on a device. At the lower edge, miniport drivers use NDIS to "talk" to the adapter hardware. At the upper edge, miniport drivers present an interface to allow protocol drivers to configure the adapter and to send and receive packets over the network.

Windows CE 3.0 supports WAN miniport drivers. A WAN miniport provides protocol drivers (for example, PPP) access to a WAN interface.

Windows CE currently supports miniport drivers for the following medium types:

- Ethernet (802.3)
- Token Ring (802.5)
- IrDA
- WAN

### 7. Writing a Miniport Driver

When you write an NDIS miniport driver, you should make sure that the driver is easily portable across all of the other platforms that support the NDIS 4.0 interface. Ideally, porting from one hardware platform to another will require only recompilation with a system-compatible compiler.

To guarantee cross-platform portability, you should avoid calling operating system-specific functions and instead use the NDIS equivalents of such calls. NDIS exports a wide array of functions for writing drivers, so that it is not necessary to make calls to the operating system directly.

You should write your miniport driver in the C language by using the ANSI C standard. Make sure not to include any calls to C run-time library functions in your driver code, instead of NDIS-provided functions. Kernel mode on Windows NT does not allow the use of floating-point operations; so, if you use floating-point calls in your driver, the driver might work only on Windows CE. If you must include platform-specific features in your driver, make sure to encapsulate the code between the **#ifdef** and **#endif** statements.

Windows CE 3.0 ships with several driver code samples. Driver developers should consult this sample code to gain a better understanding of the concept of writing an NDIS miniport driver for Windows CE. The driver sample code that is shipped with Windows CE 3.0 include the following:

- NE2000 ISA, PCI, and PC Card drivers
- Proxim Wireless Ethernet PC Card driver
FAST IR driver that uses National Semiconductor chipset

### 8. NDIS Driver Registration and Initialization

All NDIS miniport and intermediate drivers must provide a `DriverEntry` function. When a miniport driver is loaded, NDIS calls `DriverEntry`, which creates an association between the miniport driver and the NDIS library, and registers the miniport with NDIS.

The `DriverEntry` function requires the following syntax:

```c
NTSTATUS DriverEntry(
    IN PDRIVER_OBJECT pDriverObject,
    IN PUNICODE_STRING pRegistryPath);
```

The following parameters are passed into the `DriverEntry` function:

- A pointer to the driver object, which was created by the I/O system
- A pointer to the registry path, which specifies where driver-specific parameters are stored

The `DriverEntry` function must call the `NdisMInitializeWrapper` function and then the `NdisMRegisterMiniport` function. `DriverEntry` first calls `NdisMRegisterMiniport` with the parameters that are passed to `DriverEntry`, which returns a wrapper handle. `DriverEntry` then passes the handle to `NdisMRegisterMiniport`.

Windows CE does not support the common network .inf file format for dynamic installation and registry configuration of miniport drivers that are used in Windows NT. Instead, Windows CE uses the registry for dynamic installation and binding of device drivers. Therefore, your driver installation program must make sure to create the appropriate registry settings for your miniport driver in the Windows CE registry.

The following diagram shows the process for registering a miniport driver and initializing the NDIS library.