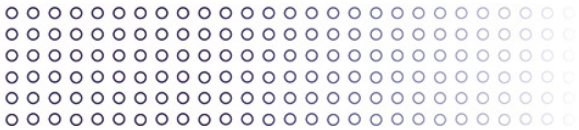

OPTION
WIRELESS TECHNOLOGY

Technical Specification

NDIS Driver Interface Description Document

Option Confidential

Option nv, Gaston Geenslaan 14, 3001 Leuven - Belgium - Tel +32 16 317 411 - Fax +32 16 207 164

About this document

Overview and Purpose

This document describers the interface of the NDIS driver.
The driver has two interfaces accessible to the user. An application interface, available through the CreateFile() Win API and an interface accessible through the registry that can modify data the driver uses to make connections.

This document forms, along with sample source code for the GtmNicApp.cpl, an application that uses the first interface and the defdata.reg file constitute a kit designed to help the creation of an application that uses the Ndis driver.

This document relates to Ndis drivers for all our cards HSDPA, 3.6, 7.2 and HSUPA.
Where there are differences, these will be stated, but there are two variants, 7.2 and pre 7.2.

The Ndis driver also has a different name for each, so pre 7.2 drivers have 'Irp; in the file name (eg Gtm51Irp.sys) while 7.2 drivers have 'Ip' in the name (eg Gt51Ip.sys).

The 'm' in the first name implies it is a USB based card of 3.6 or lower speed. An 'n; signifies a PC based card of 1.8 mbps speed.

7.2 cards don't use this small letter after the 'Gt' so are just Gt51Ip.sys for example.

The numeric values '50' and '51' relate to the Ndis version number (same as the OS version number). So Windows XP, is 5.1, so Gt151Ip.sys will be loaded on XP and later. Gt50Ip.sys will be loaded on Windows 2000 machines.

Currently, 08/2007, there is no specific Ndis 6.0 driver for Vista, we use the XP driver as Vista is backwards compatible with XP in this respect.

Confidentiality

Version History

Date	Version	Author(s)	Remarks
Oct 03, 2007	001ext	M. Sykes	Initial version
Oct 22, 2007	002ext	M. Sykes	Updated TCP configuration
Dec 5, 2007	003ext	M. Sykes	AT_OWANCALL unsolicited event support added
Feb 21, 2008	004ext	M. Sykes	IOCTL_GT_NDIS_GPRS_SET_TCPWINDOWSIZE added
Aug 6, 2009	005ext	J. Schrijvers	IOCTL_GT_NDIS_GPRS_READ_DNSADDRESS_FORCEUSE and IOCTL_GT_NDIS_GPRS_WRITE_DNSADDRESS_FORCEUSE added

Table of contents

1 INTRODUCTION

The Ndis driver is a network driver fully compliant with the Ndis 5.1 specification. As such it uses standard system functionality and will appear to the system as a network device supporting the 802.3 (Ethernet) type. The driver internally implements a DHCP server to transfer addresses obtained form the network to the client PC and handles ARP requests too.

The assignment of IP address for the network connection should therefore be left as the Windows default which is 'Assigned by DHCP'.

The interface the driver exposes to applications can be accessed via the standard CreateFile, CloseHandle and DeviceIoControl functions and consists of a number of IOCTLS controlling and querying various aspects of the Ndis drivers functionality.

The data used by the driver to make a connection, phone number, at commands, username etc can be set during installation. When the driver first runs, but not after, it copies data from a specific area to the area given it by the system in the registry.

An example of this data and where to put it given in defdata.reg. All that is needed is to set the data in the file and run it at installation time, but before the ndis driver loads.

It has to be noted that there are two different areas in the registry where this data is read from dependant on the technology of the card. For any of the 7.2 devices running in 7.2 mode then the data is read from HKLM\System\CurrentControlSet\Services\ {GT07DOT2-11ED-4329-B92E-3ADA2FCFCDD0}\Profiles\Default. For 3.6 and slower cards it is read from ... {GT5E3DA4-11ED-4329-B92E-3ADA2FCFCDD0}\profiles\Default.

When the system then loads the Ndis driver it will copy this new data to the area given it in HKLM\System\CurrentControlSet\Control\Class\ {4D36E972-E325-11CE-BFC1-08002BE10318}\<num> ('num' is defined by the system).

The Ndis driver will there after use this data to make connections. This data can also be overridden from an application using the IOCTL interface.

2 REFERENCES

Ref	Document	
1	GtNDISDeviceIO.h	
2	Deneric_DRV_TS_PnP Compliant Applications.doc	

3 NDIS DRIVER CONNECTION BEHAVIOUR

3.1 Connection via the Ndis driver IOCTL CONNECT

When the device is inserted, the driver is loaded by the PnP manager. The driver reads its configuration data from the registry at HKLM\System\CurrentControlSet\ontrol\Class\ {4D36E972-E325-11CE-BFC1-08002BE10318}\<num>

This data can also be seen on the Advanced tab of the device properties in Device Manager.

This data is composed of information used by the driver to make a connection. So, for example, user name, password (if used by the network), APN, and so on.

Of special note are the AT connection and configuration strings, seven of each. These are special AT commands that the driver runs in order to configure the card, or, to wait for a certain state, or to enter a PIN for example.

The format of the AT connection strings is always <command>^response1~response2~ etc.

What this means is that the text up to the '^' is treated as an AT command, and sent to the device. It is repeatedly sent until the response from the card contains any of the text between the '^' and the first '~', and between subsequent '~'s.

*7.2 difference:

For the pre 7.2 drivers the default strings are:

AT^OK~
AT+CREG?^0,1~0,5~

For 7.2 devices the default strings are:

AT+CGREG?^0,1~0,5~1,1~1,5~
at_owancall=1,1,1^OK~
at_owancall?^,1~, 1~
at_owandata?^owandata~

So, the AT+CREG? example above will be sent, repeatedly, until the response form the card is +CREG=0,1 or +CREG=0,5.

PIN commands are treated somewhat differently. Because of the chance of SIM lock, if the PIN is set in correctly 3 times, any AT command containing the text PIN is sent just once, and regardless of the response from the card, the next step is proceeded to.

The final AT command for the non 7.2 drivers is the dial command ATDT*99#. If a CONNECT_xxx is returned by the hardware, the driver starts the LCP negotiation which results in an IP and DNS address being received from the network, and then given up to the operating system.

This process can be observed if the control panel applet is used to view trace information at a depth of 14.

While the connection strings for the pre version 7.2 drivers can be changed, or even deleted, those for the 7.2 drivers must not be unless they are sent by an application. If they are, the driver will fail to connect.

If the application needs to send the commands, then all but the last string can be deleted, and the application can set up the call, finally calling the CONNECT IOCTL.

The 7.2 driver uses the last string, _OWANDATA?, to get the IP addresses from the card.

The DNS addresses supplied by the network can be overridden by the addresses specified through IOCTL_GT_NDIS_GPRS_WRITE_CONFIGDATA. To enable overruling, IOCTL_GT_NDIS_GPRS_WRITE_DNSADDRESS_FORCEUSE should hold a value 1 in its in buffer. To disable overruling,
IOCTL_GT_NDIS_GPRS_WRITE_DNSADDRESS_FORCEUSE should hold a value 0 in its in buffer.
The DNS overruling control setting can be checked through
IOCTL_GT_NDIS_GPRS_READ_DNSADDRESS_FORCEUSE.
The DNS address overruling control setting will be applied in the subsequent IOCTL_GT_NDIS_GPRS_ CONNECT call(s).

After successful negotiation of the IP addresses by either LCP on pre 7.2 drivers, or via _OWANDATA on 7.2 drivers, the driver tells the OS it is connected, and the OS sends DHCP requests to the driver. The driver replies with the IP addresses it has, and the system is up and running, and the NDIS driver can now ship IP packets onto the device and to the network.

There are also 7 config strings, which are identical to the connect strings in format, and can also be used to pre set card information, such as frequency selection, or whatever AT command is required.

3.2 Automatic connection on card insertion

The Ndis driver can connect automatically to the network when the driver starts. This behaviour is controlled by the AutoConnect flag.

When this flag is set to on, or true, the Ndis driver automatically sends the AT connection strings in the order they are listed resulting in an open connection to the network.

If this flag is set to off or false, the Ndis driver does not connect automatically. Instead, an application can cause the connection via the CONNECT IOCTL detailed later. When the Ndis driver receives this IOCTL it runs all the AT connection strings as detailed above.

3.3 Connection initiated via the Application channel, only on pre 7.2 devices

Another flag of note is the WaitForDCDGoing1 flag. If it is set TRUE, the driver does not send any AT commands to the hardware, what it does is wait for the DCD line to go high on the device. When this happens, the driver starts the LCP negotiation.

This is used if another application wants to send AT commands through the Application channel on the card. When the application has sent all the AT commands it needs to configure the device, it dials, and when DCD goes high, the NDIS driver takes over the rest of the connection process by doing the LCP negotiation.

3.4 Interrelation of the flags and how they affect which connection strings are run

If AutoConnect is off, then only the config strings are run when the driver starts.

If AutoConnect is on, all the strings are run automatically when the driver starts.

If WaitForDCDGoing1 is on, none of the strings are processed.

3.5 Auto reconnection

Another flag that controls the connection behaviour is AutoReconnection. With the inbuilt modules, there is often a switch on the laptop that controls the radio. When this is turned off the Ndis driver sees a DCD drop, and disconnects. With AutoReconnect set true, it then tries to reconnect, polling the state of the card to see if it is registered. It does this by rerunning the connection strings. AT+CREG? will only return 0,1 or 0,5 etc if the radio is on, and the card has an active context. Of course, on a non module card without the radio switch this flag just reconnects immediately to the network so it should only be set true for the module with radio switches on the laptop.

3.6 Overriding the default connection data

The data used by the Ndis driver, the AT connection strings, user name and password and so on can be overridden by using the registry. Create a file in the form of defaultdata.reg with the required data and run it to add this data to the registry at installation time. The Ndis driver will read this data once, when first run, and copy it to its local area whereupon it will be used every time it connects.

Note that as stated on the 7.2 drivers, the _owandata? string must not be deleted or overwritten, and the other strings on the 7.2 driver can be removed if the application takes over responsibility for sending them.

3.7 Disconnection behaviour

For Nozomi drivers after 2.1.1.56, Cobra drivers after 3.1.1.60 and all 7.2 drivers the Ndis driver has an accompanying service, gtdetectsc.exe, which causes the card to disconnect from the network on standby, hibernate, and power down.

This allows the network to free those resources committed to that device.

This service needs to be installed in the usual way using the service api or via the command line, 'gtdetectsc -register' and then started via 'net start gtdetectsc'.

3.8 TCP window size

The Ndis driver can also set the TcpWindowSize which is useful for maximising throughput on fast, high latency links. The Tcp window size controls the size of data sent by a sender before it waits for an acknowledge. On fast high RTT links, this limits throughput, and bigger window sizes help a lot. There are four possibilities for setting this:

1) TcpWindowSize, GlobalMaxTcpWindowSize (read from driver area/inf file), Tcp3123Opts (value of 3), and MaxConnectRetransmissions (value of 5) are set in TcpIp\Paramaters key.

2) TcpWindow size is set in the TcpIp\Paramaters\Interfaces\<net iface GUID> key.

3) No settings are made.

4) The filter driver (GtTdiFltr.sys) is accessed. The Tcp Window size is set through IOCTL_GT_NDIS_GPRS_SET_TCPWINDOWSIZE.

5) TcpWindowSize and Tcp3123Opts are set hard coded in TcpIp\Paramaters key as 96,360 and 1 respectively

3.9 NDIS Driver IOCTL Interface Description

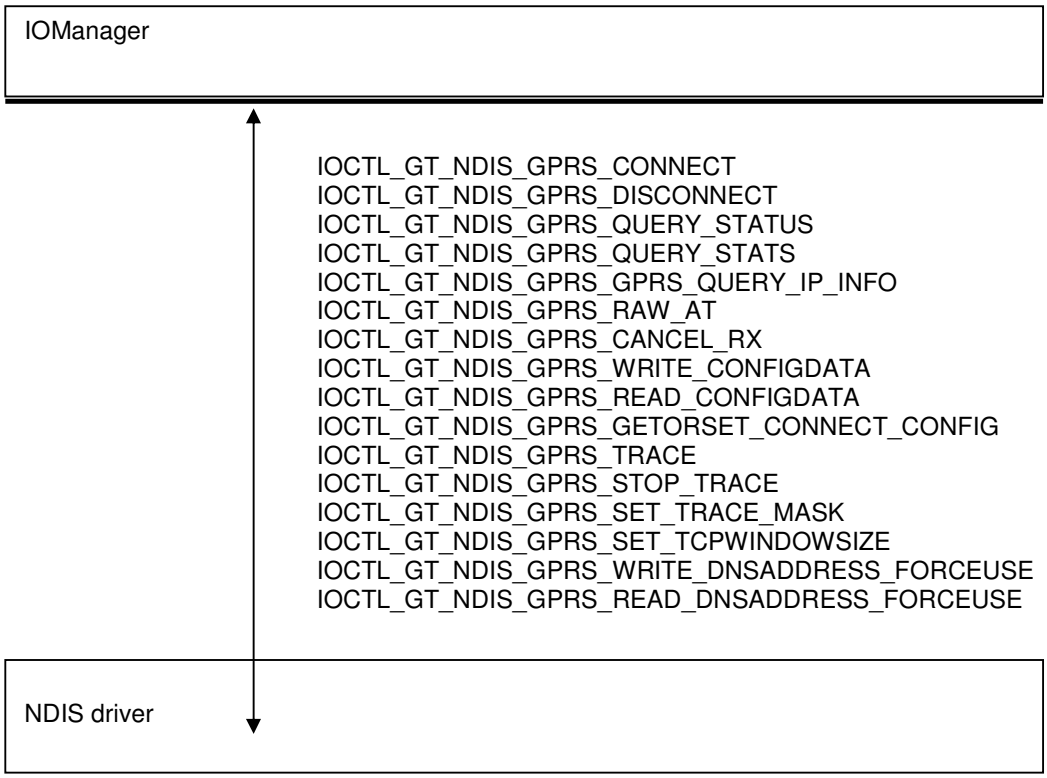

Figure 2 shows the upper interface to the NDIS driver.

Interface, opening:
The upper interface is composed of IOCTLs, IRP_MJ_CREATE, IRP_MJ_CLOSE. and IRP_MJ_CLEANUP.
To access the driver, an application calls CreateFile() with the file name "\\\\.\\GTNDIS*n*" where *n* is a character representation of a value from 0 to 10. This value indicates the number of cards on the system this driver supports. So, with one card inserted and the driver running, this value will be '0'.
One thing to note is that the NDIS NIC driver (this driver) cannot be unloaded if an application has opened the device with a call to CreateFile(). The application needs to close the handle first. Therefore it must register for device event notification, specifying GUID_NDIS_LAN_CLASS as the interface class GUID. On receiving a DBT_DEVICEQUERYREMOVE device event for the device object, the application must close the handle. See Reference 2 for sample code to do this and PnPCompliantApplications.doc.

3.9.1.1 IOCTL_GT_NDIS_GPRS_CONNECT

In buffer is optionally a GT_NDISGPRSConnectParam struct.
No out buffer.
This ctl code causes a reconnect, using the data if supplied, if not, with the data read from the registry. The data supplied is not written to the registry for permanent use by the driver.

A successful registration with the network puts the card in to data mode.

The version 4 drivers ignore a user name and password passed in this structure. To set username and password on version 4 drivers the relevant AT command needs to be sent via the application channel.

3.9.1.2 IOCTL_GT_NDIS_GPRS_DISCONNECT

No in or out buffer.
This ctl code just disconnects the current session and puts the card into a state where it
The card is put into AT mode.

3.9.1.3 IOCTL_GT_NDIS_GPRS_QUERY_STATUS

In buffer. a 4 byte value. If 1, the status is returned the next time it changes, If 0 the current status is rerturned.
Out buffer is a GT_NDISGPRSStatus struct.

3.9.1.4 IOCTL_GT_NDIS_GPRS_QUERY_STATS

No in buffer.
Out buffer is a GT_NDISGPRSStats struct.

3.9.1.5 IOCTL_GT_NDIS_GPRS_GPRS_QUERY_IP_INFO

No in buffer.
Out buffer is a GT_NDISGPRSIpAddr struct.

3.9.1.6 IOCTL_GT_NDIS_GPRS_RAW_AT

In buffer is a CHAR buffer containing the AT command, (CR and NULL terminated)
The start of the buffer can, optionally, be a 4 byte int value to be used as the timeout in seconds for the AT command. The driver has a default 3 second timeout.
Out buffer is the response, including carriage returns.

3.9.1.7 IOCTL_GT_NDIS_GPRS_CANCEL_RX

No in buffer.
No out buffer.
This ctl cancels and resubmits a read request.

3.9.1.8 IOCTL_GT_NDIS_GPRS_WRITE_ CONFIGDATA

In buffer is a ConfigData struct.
No out buffer.
This ctl sets new configuration data in the driver, and causes the driver to store the data in the registry so it will be used next time the driver loads.

3.9.1.9 IOCTL_GT_NDIS_GPRS_READ_ CONFIGDATA

No in buffer.
Out buffer is a ConfigData struct.
Gets the configuration data used to make a connection form the driver.

3.9.1.10 IOCTL_GT_NDIS_GPRS_GETORSET_CONNECT_CONFIG

In buffer is three four byte integers. The first controls the value of autoconnect, the second, the value of WaitForDCDgoing1, the third, AutoReConnect. These values become the values in use by the driver, and, are written to the registry.
Out buffer is three four byte integers.

This ctl sets and retrieves the values for Autoconect and Waitfor DCDgoing1. These values affect the way the driver behaves. If Autroconect is TRUE, the driver connects the PC automatically to the network. Any AT commands set as configuration data are run, until the expected response is received.
If WaitForDCDGoing1 is set TRUE, the driver waits for the DCD line going high event before connecting the PC to the network. Any AT commands set as configuration data are not run.
If AutoReConnect is TRUE, the driver tries to reconnect immediately after a disconnect, either form app or DCD going 1.

3.9.1.11 IOCTL_GT_NDIS_GPRS_TRACE

No in buffer
Out buffer big enough to contain character trace data, 1024 bytes is good.
This ctl also enables tracing in the driver.

3.9.1.12 IOCTL_GT_NDIS_GPRS_STOP_TRACE

No In or out buffer
This ctl stops the driver from generating trace data.

3.9.1.13 IOCTL_GT_NDIS_GPRS_SET_TRACE_MASK

In buffer is 4 byte int whose value is a bit wise combination of 1,2,4,8,16,32
This sets the level of tracing in the driver. 16 is the default value, and will give the progress of the IP negotiation with the network.
14 is a useful value as it shows all communication and negotiation.
15 shows 14 plus, all the function calls, but is usually too detailed.

3.9.1.14 IOCTL_GT_NDIS_GPRS_GET_NET_CFG_ID

Out buffer 2 * 48 bytes (the returned data is in wide char format, Unicode)
In buffer is null.
This IOCTL returns the GUID the system associates with this network device. This can also be found in HKLM\System\CurentControlSet\Control\Class\<net GUID>\<numeric id>\NetCfgInstanceId

3.9.1.15 IOCTL_GT_NDIS_GPRS_SET_TCPWINDOWSIZE

In buffer is a 4 byte integer value that is the window size to be used
No out buffer
This IOCTL passes a Tcp Window size value to the Ndis driver. The Ndis driver then passes this value and its IP address to the filter driver (GtTdiFltr.sys). When the filter driver sees a Tcp Session init on that IP address it changes the window size to the one given. Other IP address sessions are not affected

3.9.1.16 IOCTL_GT_NDIS_GPRS_WRITE_DNSADDRESS_FORCEUSE

In buffer is a 1 byte char whose value should be 0 or 1.
No out buffer.
This IOCTL controls overruling of the DNS addresses provided by the network.
When the in buffer holds 0, the DNS addresses provided by the network are applied and no overruling is done. The DNS addresses provided in the ConfigData struct through IOCTL_GT_NDIS_GPRS_WRITE_CONFIGDATA are not applied.
When the in buffer holds 1, the DNS addresses provided by the network are overruled by the ones provided in the ConfigData struct through IOCTL_GT_NDIS_GPRS_WRITE_CONFIGDATA.

The DNS address overruling control setting will be applied in the subsequent IOCTL_GT_NDIS_GPRS_CONNECT call(s).

3.9.1.17 IOCTL_GT_NDIS_GPRS_READ_DNSADDRESS_FORCEUSE

No in buffer.
Out buffer is a 1 byte char whose value should be 0 or 1.
This IOCTL reads out the DNS addresses overruling control as set by IOCTL_GT_NDIS_GPRS_WRITE_DNSADDRESS_FORCEUSE.
When the in buffer holds 0, the DNS addresses provided by the network are applied and no overruling is done.
When the in buffer holds 1, the DNS addresses provided by the network are overruled by the ones provided in the ConfigData struct through IOCTL_GT_NDIS_GPRS_WRITE_CONFIGDATA (and can be read out by IOCTL_GT_NDIS_GPRS_READ_CONFIGDATA).