

A world map showing the continents, with a horizontal bar across the middle. The bar is divided into five colored segments: orange, red, dark brown, blue, and green.

# Gobi™ Connection Manager

80-VR668-1 C

## Qualcomm Confidential and Proprietary

**Restricted Distribution.** Not to be distributed to anyone who is not an employee of either Qualcomm or a subsidiary of Qualcomm without the express approval of Qualcomm's Configuration Management.

Not to be used, copied, reproduced in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm.

Qualcomm reserves the right to make changes to the product(s) or information contained herein without notice. No liability is assumed for any damages arising directly or indirectly by their use or application. The information provided in this document is provided on an "as is" basis.

This document contains Qualcomm confidential and proprietary information and must be shredded when discarded.

QUALCOMM is a registered trademark of QUALCOMM Incorporated in the United States and may be registered in other countries. Other product and brand names may be trademarks or registered trademarks of their respective owners. CDMA2000 is a registered certification mark of the Telecommunications Industry Association, used under license. ARM is a registered trademark of ARM Limited. QDSP is a registered trademark of QUALCOMM Incorporated in the United States and other countries.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

QUALCOMM Incorporated  
5775 Morehouse Drive  
San Diego, CA 92121-1714  
U.S.A.

Copyright © 2009 QUALCOMM Incorporated.  
All rights reserved.

# Revision History

Version	Date	Description
A	Jun 2009	Initial release
B	Jul 2009	Removed numerous references
C	Oct 2009	Added slides 21-23, 37-38, 45, 52-57; updated slides 5, 7-9, 32, 42, 50-51

# Contents

- Operating System Support
- SDK Functionality and Usage
- Operator Image Management
- Notebook Integration
- Operator Customizations
- Application Feature Support
- References

# Overview

- Operating system support
  - This section describes topics associated with operating systems that are supported for Gobi™
    - Windows
    - Linux

# Overview (cont.)

## ■ SDK functionality and usage

- This section describes topics associated with the functionality and usage of the software development kit for the Gobi API
  - Gobi1000™ API capabilities
  - Gobi2000™ API capabilities
  - Maintenance
  - Backward compatibility
  - Sample Connection Manager (CM)
  - API call frequency
  - Device presence management

# Overview (cont.)

- Operator image management
  - This section describes topics associated with the capability of Gobi to select operator-specific firmware
    - QDL service
    - Operator image selection
    - Operator image confirmation

# Overview (cont.)

## ■ Notebook integration

- This section describes topics associated to when applications are integrated onto specific OEM notebooks
  - Multiple device support
  - Simultaneous control and data
  - Selective suspend concerns/best practice
  - Wireless disable behavior
  - Notebook OEM support
  - USB drivers
  - Power management

# Overview (cont.)

- Operator customizations
  - This section describes topics associated with customizations for specific operators
    - Operator plug-ins
    - Operator settings
    - Operator certification milestones
    - IMSI blocking for Japan
    - PRL updates



# Overview (cont.)

- Application feature support
  - This section describes the functionality provided by Gobi for additional user features
    - EAP-SIM integration
    - Reading the SIM while in CDMA
    - GPS/aGPS configuration and usage
    - Auto-connect feature
    - CM coexistence/concurrency
    - Windows 7 Native UI/MBB API configuration
    - Remote wake-up

# Operating System Support

# Windows

- XP/Vista/Windows 7 (32/64)
  - Microsoft WHQL-certified drivers
  - Windows logo compliant
  - All Gobi modules are supported
- Available interfaces
  - Microsoft Windows 7 mobile broadband WWAN API
  - Gobi CM APIs
  - AT port APIs
  - Diagnostic port APIs
  - Access to GPS data
    - Windows location/sensor APIs (Gobi2000)
    - Gobi CM APIs
    - NMEA port
- For detailed documentation, see [R1] and [R2]

- Gobi1000
  - AT port APIs
  - Ubuntu and Linpus distributions (provided upon request)
- Gobi2000
  - AT port APIs
  - Gobi CM APIs
  - Access to GPS data
    - Gobi CM APIs
  - Ubuntu, Linpus, SUSE, Fedora, and Vanilla distributions (provided upon request)

# SDK Functionality and Usage

# Gobi1000 API Capabilities

- APIs in Gobi1000 SDK
  - Device connectivity (connect/disconnect, etc.)
  - Wireless data service (session state/start/stop/autostart, etc.)
  - Network access service (serving/home network, preferences, etc.)
  - Device management functions (capabilities, IDs, etc.)
  - SMS service (save, send, delete, etc.)
  - Firmware management service (get ID, upgrade, etc.)
  - Position determination service (configure, start/stop, etc.)
  - Callback functions – Asynchronous notifications (session state, data bearer, SMS, etc.)

# Gobi2000 API Capabilities

- APIs in Gobi2000 SDK
  - All Gobi1000 APIs
  - Device connectivity
    - QCWWAN2kEnumerateDevices
    - QCWWAN2kConnect
    - QCWWAN2kGetConnectedDeviceID
  - Wireless data service
    - GetIPAddress
  - Device management service
    - GetFirmwareRevisions
    - GetIMSI
    - GetOfflineReason
    - UIMGetICCID
  - Firmware management service
    - UpgradeFirmware2k

# Gobi2000 API Capabilities (cont.)

- APIs in Gobi2000 SDK (cont.)
  - Position determination service
    - PDSInjectTimeReference
    - Get/SetPDSDefaults
    - Xtra Functions
    - Get/SetAGPSConfig
  - Card Application Toolkit (CAT) service
  - Remote management service
  - OMA-DM service
  - Callback functions – Asynchronous notifications
    - SetActivationStatusCallback
    - SetPowerCallback
    - SetLURejectCallback
    - SetCATEventCallback



# Maintenance

## ■ SDK package

- CM is responsible for integrating the most current SDK version, in association with notebook OEMs and operators
- Regular integration of the most current version is advised
- Releases can occur for several reasons
  - CM API new features
  - Bug fixes

## ■ Drivers package

- Notebook OEM is responsible for integrating the driver version appropriate for a specific platform requirement
- Integration of the most current version is advised
- Releases can occur for several reasons
  - New drivers for new OEM notebooks (VID/PIDs)
  - New features
  - Bug fixes

# Backward Compatibility

- Backward compatibility
  - A newer product is backward compatible when it is able to take the place of an older product by interoperating with an application that was designed for the older product
  - A product consists of the following:
    - Gobi device
    - PC software SDK
    - Drivers package

# Backward Compatibility (cont.)

- CM support for multiple Gobi versions
  - CMs are integrated with specific versions of Gobi
  - There are different libraries (DLLs and LIBs) for each Gobi version
  - Subsequent libraries are backward compatible with all API library functions (provided in earlier Gobi versions)
  - CM compatibility requires loading the correct library for the specific Gobi version
  - CM is responsible for determining which Gobi library to use that matches the installed Gobi hardware – See example provided by the sample CM (included with the SDK)

# SDK Sample CM

- Sample CM
  - SDK is distributed with a sample CM in source code form
  - Required development environment
    - Gobi1000 – Visual Studio 2005 Visual C++
    - Gobi2000 – Visual Studio 2008 Visual C++
  - This code can be used by your CM in any way that abides by the SDK license agreement
  - Sample CM provides basic functionality; however, it is not intended to be used commercially

# API Call Frequency

- CM usage of the Gobi API function calls
  - Polling
    - Do not poll more frequently than 5 sec
    - USB Selective Suspend behavior will be affected by more frequent polling
  - Timeout best practices
    - A timeout can occur from an API function call when the underlying mechanism does not respond within the expected time frame
    - The call should be retried a minimum of two times before declaring a failure to the application
    - Refer to the timeout recommended per API function

# Device Presence Management

- Device presence management
  - CMs must monitor the device to be able to take the appropriate action if the device appears or disappears
    - When the user changes the Airplane mode switch
    - When the user affects the module through other applications (e.g., Windows Device Manager)
- Device removal detection
  - CMs typically poll the device to verify that it has not disappeared
    - Many API functions could be used for presence verification
    - A polling interval of less than 5 sec could affect the notebook power management of the module
      - » USB Selective Suspend behavior could be affected
  - Call QCWWANGetConnectedDeviceID for Gobi1000 and QCWWAN2kGetConnectedDeviceID for Gobi2000 every 10 sec
    - If Error 6 (Unable to detect WWAN device) is returned
      - » Call QCWWANDisconnect to release the current handle

# Device Presence Management (cont.)

- Device arrival detection
  - Use Windows to determine when to start the reconnection procedure to the device
    - Wait for a WM\_DEVICECHANGE event to detect a new device
    - Prevents unnecessary cycles through the polling process below
    - Optional, as the process described below would work without this step
  - Use the CM API to connect/reconnect to the device
    - Detecting and reconnecting is a combined procedure when the API is used
      - » For Gobi1000: Call QCWWANConnect to establish communications with the module
      - » For Gobi2000: Call QCWWAN2kEnumerateDevices to detect the new Gobi device, then call QCWWAN2kConnect to establish communications with the module
    - Pause for 5 sec
    - Repeat the above actions, as desired

# Operator Image Management



# Qualcomm Downloader (QDL) Service

- QDL functionality
  - Gobi hardware does not maintain operational firmware on the module
  - Firmware is downloaded to the module by QDL functionality
  - When the module first boots, it enumerates a single USB port for QDL purposes
  - QDL application downloads currently selected firmware to the module when QDL port presence is detected; this process takes approximately 5 sec
  - After firmware is successfully loaded, QDL port is no longer present

# Operator Image Selection

- Methods and best practices
  - Gobi is a multimode device
  - Firmware exists on the PC hard drive
  - Firmware is downloaded to the module at device initialization
  - CDMA (1xRTT/EVDO) and UMTS (WCDMA/HSPA) are separate firmware images
  - When an operator image is selected, it is critical to use the Gobi CM API
    - There is no guarantee that shortcuts will be functional in future Gobi releases
      - » Images may not always be stored at the same location
      - » Newer modules retain default operator information for multiple device support
    - Important steps (i.e., device reset) are performed automatically by the API
    - Using the API ensures forward compatibility in subsequent Gobi releases

# Operator Image Confirmation

- Confirming the correct firmware is selected
  - Operator firmware can be selected by different applications
    - If different operator firmware is selected outside of your application, there is no guarantee that the module will revert to your firmware when it returns to your application
    - Your application should confirm that correct firmware is always selected
    - Appropriate functionality must be supported by your CM to guarantee that your firmware is selected
  - Refer to example code in the sample CM provided with the SDK
  - Recommended steps
    - Call GetFirmwareInfo() to verify the correct operator image is running
    - If the correct operator image is not running, call GetImageStore()
    - For each image folder, call GetImageInfo()
    - When the correct operator image is found, call UpgradeFirmware()

# Notebook Integration

# Multiple Gobi Device Support

- Gobi1000 does not support multiple Gobi devices
  - Only the last operator selected is utilized for new Gobi device enumeration
  - Use Device Manager to disable all but one Gobi device
  - It is critical to select appropriate firmware through the CM API, prior to utilizing a different Gobi device
- Gobi2000 provides multiple device support
  - Multiple Gobi2000 devices can be active simultaneously
  - Each Gobi device can run different Gobi firmware simultaneously, using the CM API
  - Gobi remembers the last operator image it was running on the device
  - Each Gobi device communicates the last operator to the QDL firmware downloader
  - Multiple device support requires use of the CM API

# Simultaneous Control and Data

- NDIS and RAS data connection capabilities exist on Gobi
  - Only a single RAS port is supplied
  - If RAS port is used for data connectivity, it cannot be used for other functions while a data connection is established
  - NDIS connectivity is controlled through the CM API
  - NDIS and RAS ports function simultaneously
  - AT commands can be issued while an NDIS data connection is active
- Minimized AT migration for simultaneous control and data
  - For CMs that currently use two AT ports (one for control, one for data)
    - Utilize the CM API to control an NDIS data session
      - » StartDataSession/StopDataSession
    - Use the CM API for all functionality related to the data session
    - The AT port can be used for functionality not related to the data session
      - » Microsoft RAS functions that return information, based on AT port usage (bytes transmitted/received, data rates, etc.), do not function properly in relation to an NDIS data session

# USB Selective Suspend Concerns/Best Practices

- Selective suspend may be enabled or disabled for each platform, per OEM requirement
- Device polling period
  - Period at which a device is polled should not be less than the selective suspend timeout value
  - If this rule is not obeyed, device never enters the suspended state
  - Timer is currently hard-coded at 5 sec

# Wireless Disable Behavior

- Notebook wireless disable switch operation
  - Gobi1000 – Powers on/off the module
  - Gobi2000 – Enables/disables Low Power Mode (LPM) of the module
    - Enables/disables WWAN radio; standalone GPS is not affected
- CMs should work with OEM to manage the device accordingly



# Windows USB Drivers

- Composite driver
  - Serial driver that supports QDL (for firmware download), NMEA, diagnostics, and modem devices
- RmNet network driver
  - Qualcomm proprietary network driver (control and data path)
- Mobile broadband driver
  - Mobile broadband driver for Microsoft Windows 7 operating system
- Filter driver
  - Driver that sits below the composite driver to eliminate redundant beeps on device enumeration
- Windows 7 location sensor driver
  - Gobi2000 only

# Linux USB Drivers

- Composite driver
  - Serial driver that supports QDL (for firmware download) and modem devices
- RmNet network driver
  - Qualcomm proprietary network driver (control and data path)

# OEM Device Support

- OEM Gobi devices
  - Vendor IDs (VIDs) and Product IDs (PIDs) are uniquely assigned per OEM Gobi device
  - Gobi OEM hardware is manufactured for specific VID/PIDs
  - Gobi OEM firmware is produced for specific OEMs
- Gobi USB drivers are produced to uniquely support OEM devices
  - New OEM customers can be added in any release
  - To determine if a specific OEM VID/PID is supported, examine the release notes or ReadMe to verify that OEM VID/PID is present

# OEM Device Support (cont.)

- Gobi1000 SDK has a dependency on the OEM VID/PIDs supported
  - Upgrade to the most current Gobi1000 SDK to ensure support for all OEM Gobi devices
  - Gobi1000 SDK ReadMe lists the OEM Gobi device VID/PIDs supported
- Gobi2000 SDK works independently on any OEM Gobi2000 device

# Power Management

- Power management for the embedded module can take many forms
  - Hardware control from the notebook
  - Software control from the notebook
  - Software control from the operating system
  - Software control from an application
- CM control of module power can be through multiple mechanisms
  - Notebook OEM-supplied API
  - Operating system API
  - Device API
- Potential synchronization issues exist
  - Multiple APIs
  - Multiple applications

# Power Management (cont.)

- The module supports the following power states:
  - Powered-on/off – Controlled by the notebook
  - LPM – Controlled through the notebook, OS, or device API
    - Enables/disables WWAN radio; standalone GPS is available
  - Online/Offline – Controlled through the device API
    - RF enabled/disabled
- CM power management
  - LPM can be managed through a notebook, OS, or device API
  - Offline can be managed through the device API
- CM should display the current module power state

# Operator Customizations

# Operator Plug-Ins

- Plug-ins are generally required for account activation and/or module provisioning
- Activation
  - Some operators require integration of their proprietary plug-ins in a CM
  - Direct relationship must be established with those operators to get access to those plug-ins
  - Applicable only if your CM needs to directly support those operators
  - In some cases, generic mechanisms can support module activation and provisioning
    - OTAPA/OTASP can be performed to provision the module
    - Approach may have limitations; generic mechanisms may be limited to module provisioning, as account activation must be performed manually



# Operator-Specific Settings

- Operators have specific settings to support connectivity with their network
- Generally associated with SMS settings, APNs, usernames, or passwords
- Some of these may be defaulted in the operator-specific firmware
- In many cases, these values must be chosen by the CM
- Mechanisms are provided to support the CM manipulation of these fields
  - CM API, AT commands, or both
  - In limited cases, an operator plug-in may provide these capabilities
- CM cannot assume that correct values are set in all cases, so best practices include setting appropriate values when initially connecting to the device or prior to every data connection
- To get a comprehensive list of fields that need to be set, it is best to work directly with the appropriate operator

# Operator Certification Milestones

- Milestone 1 – Module acceptance
  - Module acceptance by an operator includes firmware and USB drivers
- Milestone 2 – Operator CM acceptance
  - Operator CM is usually integrated with the module in a standalone configuration
  - Approved firmware and drivers are used by the CM during integration
  - CM acceptance includes module firmware, USB drivers, and SDK
  - CM acceptance may require changes to SDK

# Operator Certification Milestones (cont.)

- Milestone 3 – Notebook acceptance
  - Notebook acceptance is predicated upon module acceptance
  - Operators expect the notebook OEM to deliver the same firmware and USB drivers that were approved during module acceptance
  - Operator CM expects that the notebook will also include the same SDK that was integrated with the operator CM
  - Only the module firmware is guaranteed not to change between module and CM acceptance and notebook submission
  - Differences in the USB drivers and SDK versions must be managed between the notebook OEM and the operator

# Blocking the IMSI Display for Japan

## ■ Japan Regulatory Requirement

- The Japanese regulatory agency, Telec, has declared that the IMSI shall not be easily accessible by the user
- AT access has been disabled for the NTT DoCoMo build
  - AT+CIMI has been disabled
  - AT+CNUM has been modified to return only the MSISDN
- CM API access to the IMSI has been disabled
- QMI access has been disabled
- Microsoft WWAN API provides a method to suppress the display of the IMSI

# Preferred Roaming List Updates

- Preferred Roaming List (PRL) is a CDMA operator set of preferred networks
  - These are updated by the operators on a sporadic basis
- The available PRL is provided in the firmware release for an operator
  - The current firmware may not always contain the latest PRL
- PRL updates can occur with updated firmware, OTA, through an API function or an AT command
- API functions or AT commands exist to support different PRL update procedures
  - Manual PRL update through the CM API
    - ActivateManual
    - PRL file must be stored locally
  - Automatic PRL update from the Operator
    - ActivateAutomatic/StartOMADMSession
    - PRL file is sent by the network
  - Gobi versions and operator images may have restrictions for this support

# Application Feature Support

# EAP-SIM/EAP-AKA Integration

- AT+CSIM has been enabled in order to support existing EAP-SIM/EAP-AKA clients that are implemented at the CM level

# Reading the SIM while in CDMA

- SIM access in CDMA mode
  - Gobi1000 does not support this functionality
  - CM API function GetIMSI has been added for the Gobi2000 SDK
    - This function is supported for both CDMA and UMTS modes
  - AT commands to access the SIM are not available while in CDMA mode



# GPS Capabilities, Configuration, and Usage

- Gobi1000 supports standalone GPS only
  - Configuration of a GPS session is only supported through the CM API
  - Starting/stopping a GPS session is only supported through the CM API
  - Access to GPS data is supported through the NMEA port or the CM API
  - A CM must be running to support GPS

# GPS Capabilities, Configuration, and Usage (cont.)

- Gobi2000 supports standalone GPS, XTRA, and assisted GPS
  - Standalone and XTRA are supported on all Gobi2000 firmware
  - aGPS is supported only for specific operators
  - CM configuration of a GPS session is supported only through the CM API
  - The default connection profile must be set to enable XTRA and aGPS
    - This is performed using the SetDefaultProfile CM API function
    - XTRA will establish an NDIS session to download data from a server
    - DUN calls will be blocked until the XTRA session is complete (a silent DUN retry mechanism could hide this delay from the user)
  - Starting/stopping of a GPS session can be done through the CM API or by opening the NMEA port
    - NMEA port autostart is supported on a per OEM basis
    - To utilize the autostart feature in the application, coordinate with the OEM to determine whether autostart mode is supported for the NMEA port
  - Access to GPS data is supported through the NMEA port, CM API, or Windows 7 location/sensor API

# Autoconnect Feature

- Gobi supports an automatic data session through the autoconnect feature
- The autoconnect feature is enabled/disabled through the CM API
- When autoconnect is enabled
  - The default connection profile must be set by using SetDefaultProfile
  - The module assumes responsibility for the data session
  - The module automatically connects to the network when in an active state
  - The module automatically starts a data session when it goes to an active state
    - At power-on
    - At resumption from selective suspend
    - When LPM is disabled
    - Through AT Commands (i.e., AT+CFUN)
    - Through CM API functions (i.e., SetPower)
- When autoconnect is disabled
  - The module automatically releases an active data session
  - The module assumes no control over a data session (Manual mode)

# CM Coexistence/Concurrency

- There are no restrictions by Gobi on the existence of multiple CMs
- Multiple interfaces exist for CMs to control data sessions
  - AT commands
  - Gobi CM API
  - Windows 7 MBB API
- Multiple clients can execute simultaneously
  - Only a single client can control an AT port connection
  - Multiple clients are supported through the Gobi CM API
    - Gobi1000 supports up to two clients simultaneously
    - Gobi2000 supports up to six clients simultaneously
  - Multiple clients are supported through the Windows 7 MBB API
- Potential issues exist when multiple applications operate simultaneously
- Applications should manage the potential issues to avoid user confusion

# CM Coexistence/Concurrency (cont.)

- Starting/stopping a data connection is based on session ownership
  - When starting a data connection through DUN
    - The client that opened the AT port handle owns the data session
    - Only that client can terminate the data session
  - When starting a data connection through the Gobi API (StartDataSession)
    - The data connection is initiated when the first client requests a data session
    - Each client that requests a data session will own a virtual session
      - » The connection is shared by all clients that request a data session
    - A client can only terminate its own virtual session
    - The connection is closed when all clients have terminated their own virtual sessions
  - When starting a data connection through the use of the Auto-Connection feature
    - The connection is owned by the module as the module started the session automatically
    - Only the module can end the data session
    - The module will end the data session when the Auto-Connection feature is disabled
  - In Windows 7, when starting a data connection through the MBB API
    - The operating system owns the data session
    - The MBB API allows any application to terminate the existing data session

# CM Coexistence/Concurrency (cont.)

- Connection Manager control and display
  - Conflicts may occur when starting/stopping a data connection across different interfaces
    - A connection owned by a DUN client can only be stopped by that client
    - A connection owned by a Gobi API client can only be stopped by that client
    - A connection owned by the module (Auto-Connect is enabled) can only be stopped by disabling the Auto-Connect feature (from any client)
    - In Windows 7, a connection owned by the OS can only be stopped by a MBB API client
  - Current data connection status can be displayed, regardless of session ownership
    - For DUN connections, RAS functions can be used to determine the data connection status, regardless of who owns the data session
    - For NDIS connections, Gobi API functions or callbacks can be used to determine the data connection status, regardless of who owns the data session
    - In Windows 7, for NDIS connections, MBB API functions or callbacks can be used to determine the data connection status, regardless of who owns the data session

# CM Coexistence/Concurrency (cont.)

- The CM should display the current status to the user
  - Should be performed at execution to present the user with a valid current state
  - Can be maintained by registering for asynchronous events or polling
  - Could apply to connection state, data bearer/rate, roaming, messages, etc.
- A Gobi workaround is included for Windows 7 for status display between MBB and DUN connections
  - When a DUN session is active, the MBB driver will be hidden automatically in the native UI
  - Qualcomm recommends that, when an MBB session is active, the CM removes user access to the DUN connection
  - The supported functionality does not change; the information displayed to the user is altered to avoid confusion, as only a single data session can exist



# Windows 7 Native UI/MBB API Configuration

- APN, username, and password
  - Managed through the Windows 7 native UI
    - When first defining a profile in the Windows 7 native UI , the driver is queried for the APN, username, and password through the `OID_GET_PROVISIONED_CONTEXTS`
      - » This is only done once, at profile creation
    - The module returns the information from the current default context
      - » The default context is NULL, unless initialized by an application through the API function `SetDefaultProfile`
    - The user can modify the information stored in the OS profile through the native UI GUI
    - That information is not stored back in the driver by the native UI, as it does not call the `OID_SET_PROVISIONED_CONTEXTS`
  - Managed through a CM
    - CMs typically map the MCC/MNC from the IMSI to a preferred APN/UN/PW
      - » Ambiguity can be resolved through user interaction with the CM GUI
    - The CM can set the default profile information
      - » Using the `SetDefaultProfile` Gobi API function
      - » Using the `OID_SET_PROVISIONED_CONTEXTS` MBB API function
    - The CM can set the information for the native UI (through the MS XML interface)



# Remote Wake-Up

- Remote management via SMS is supported in Gobi2000
  - SMS messages can remotely wake up the module
  - API functions are available to manage this capability
    - SetSMSWake/GetSMSWake
    - A mask is set to determine which SMS messages wake up the module
  - The module can wake up the notebook
    - Special notebook functionality is required
    - Refer to the notebook OEM to determine if it is supported on the target platform

# References

Ref.	Document	
<b>Resources</b>		
R1	<i>Microsoft Mobile Broadband Driver Development</i>	<a href="http://msdn.microsoft.com/en-us/library/dd323269(VS.85).aspx">http://msdn.microsoft.com/en-us/library/dd323269(VS.85).aspx</a>
R2	<i>Microsoft Sensor API</i>	<a href="http://msdn.microsoft.com/en-us/library/dd318953(VS.85).aspx">http://msdn.microsoft.com/en-us/library/dd318953(VS.85).aspx</a>

# Thank You

