Bluetopia® Revision Information

Bluetopia Version 2.1.3 Changes (from Version 2.0)

- General
  - ❑ Updated L2CAP to support enhanced retransmission mode (ERTM) and streaming modes, in addition to Basic Mode.
  - ❑ Updated all Copyright Headers on all source files
  - ❑ Changed OBEX_ Function names to GOEP_ function names in GOEP (also added underscores for consistency).
  - ❑ Clean up API Definitions (linkage conventions)
  - ❑ Change constants used for maximum supported stack MTU
  - ❑ Added support for new Extended Inquiry Record Data Types
  - ❑ Added support for Secure Simple Pairing (in GAP and L2CAP)
  - ❑ Add function in HCI to be able to see if a command is supported (without fetching it from the chip everytime). The function is called HCI_Command_Supported() and even works on pre Bluetooth 1.2 devices.
  - ❑ Fixed potential thread shutdown issues in BTPSTMR, HCIDRV (framework), and HCI.
  - ❑ Updated newer GAP Inquiry Results (2.0/2.1) to be more consistent with the old naming convention (<= 1.2). This involved renaming the actual event names themselves to have Inquiry_Result in the event name.
  - ❑ Removed ACL Queue Threshold constant from L2CAPAPI.h and added ability to set/query this threshold via the L2CA_Channel_Queue_Threshold_t structure.
  - ❑ Add GAP Function to set/query the Inquiry Mode (that way the caller doen’st have to use HCI directly)
  - ❑ Broke out Vendor Specific API functions into seperate header (BTPSVEND.c/h) to allow external linking/building by customer (for chip initialization, patch-RAM, etc).

- Generic Stack configuration file that contains defaults (file is called BTPSCFG.h) for various thresholds/constants for the built stack. Some of these values are:
    - Maximum MTU for stack
    - Maximum HCI Packet Size (largest ACL Packet)
    - Maximum RFCOMM Frame Size
    - Maximum SPP Frame Size
    - Default SPP Frame Size
    - Default SPP Buffer Sizes
    - Thread Stack Sizes
    - Maximum number of Timers
    - Maximum number of Mailbox Entries
    - L2CAP ACL Queueing Parameters
    - Default SPP Server Mode
    - Default SDP Disconnect Mode
    - L2CAP Timer Values
    - RFCOMM Timer Values
    - SDP Timeout values
- Broke out GOEP Constants into new OBXTypes.h. Added code to GOEP to actually map between OBEX_Header_ID_t's and the real constants (as opposed to using them directly in the real, raw, OBEX stream).
- Updated SDP to support 16 bit integers (required changing some int's to long's).
- Updated SPP to support 16 bit integers when adding SPP SDP records.
- Updated GAP so that Remote Feature, Remote Name, and Remote Version requests can all be outstanding at the same time to the same BD_ADDR. Prior to this, it was not possible since they all used the same list.
- Fixed bugs in GAP that did not dispatch Remote Name or Remote Feature callbacks when the Device was reset.
- Updated GOEP to support either client or server side authentication by removing the OBEX_GenerateDefaultAuthenticateResponse() function and replacing it with a new function to simply perform the MD5 Hash called GOEP_Generate_Digest_Nonce().

- Updated GAP_Query_Remote_Features() and GAP_Query_Remote_Version_Information() functions to utilize BD_ADDR_t instead of Connection_Handle_t to be more consistent with GAP API usage.

- Fixed API Function prototype typedef's to make sure all were consistent (some were missing the "PFN_" prefix and some were missing the "_t" suffix). Also some were cut and paste errors (i.e. stipulated the wrong/duplicate function).

- Updated all L2CAP based profiles to correctly flag In/Out MTU based on largest supported MTU size of the stack. Added Configuration constants to BTPSCFG.h and updated profiles to utilize these constants.

- Added code to BTSER/BTCOMM to change SPP Transmit/Receive Buffer (to default constants defined in BTPSCFG.h)

- Updated OTP to support Extended Directory Entry Names (for Windows/CE backwards compatibility)

- Updated PBAP to be more consistent with rest of stack. Fixed enumerated types mapping and consistency. Also fixed data types of some parameters so that the actual lengths were capped by the data type itself (i.e. a Byte_t instead of an unsigned int).

- Updated Object Push Profile to maintain open files while putting/getting files.

- Updated Profiles to support smaller stack usage (unions for event data structures).

- Updated sample applications to support Mode menu options (connectability/discoverabilty/etc).

- Updated sample applications to support master/slave role switch.

- Updated sample applications so that the Inquiry now returns the maximum supported results.

- Updated PBAP Sample application to support Client and Server in same application.

- Updated HCIUSBT.h (Types) to allow a mechanism to specify new USB Driver Information parameters (Driver Type and Delay).

- Added Common Versioning Schema to core.

- Updated GAVD to correctly handle determining the next unused LSEID (instead of simply picking the next number in a list it now picks the first unused LSEID number).

- Updated GAVD to now return any Data Events if the Endpoint is not in streaming mode.

- Updated GAVD to correctly handle Start/Suspend Stream responses (with multiple SEIDs).

- Updated the following Profiles:

  - HCR
  - PAN
  - GAVD
  - AVCTP
  - A2DP
  - AVRCP

  to have an extra types file that is included when built. This will allow the option for specifying special #pragma's (for compilers that require it) for packing without altering the source code.

- Updated HCI Driver to correctly handle the USB SCO Configuration setting.

- Updated SCO module to correctly calculate the USB SCO Configuration setting dynamically (based on the number of active SCO connections AND the current Voice Settings.

- Fixed bugs in SCO module relating to SCO/eSCO handling with supported features and/or HCI Versions.

- Fixed Hands Free profile such that it correctly handles S2, S1, S0 eSCO configuration based on chipset version and EDR support (as per specification).

- Split PAN into two portions, the BNEP portion (BNEPType.h) and the PAN portion. Also updated BNEP portion to have NO reliance on SS1VNDIS constants/types.

- Updated AVCTP, GAVD, DUN, FAX, Headset, Hands Free, LAP, and SAP to support Service Level Security. We loosely call this "Level 2 Security". It is "Level 2 security" for AVCTP and GAVD, but the other ones, it's more Service Level Security.

- Added a new Debug Library (SS1BTDBG). This debug library will hook the HCI Debug Callback and write ASCII logs (BTExplorer style) to either a file or a Debug console. It will also allow the writing of the FTS BTSnoop binary file format.

- Updated all sample applications to now support the new Debug library.

- Updated AVCTP to return Profile ID in Events since this value is known. Allows only a single Callback to be used (like the rest of the stack).

- Updated all samples to support UART/RS-232 and USB transports.

- Created sample applications for all profiles.

- Added SDP UUID constants to A2DP Header files (previously the Profile and Service Class UUID's were undefined)

- Updated all profiles to make sure that SDP Profile/Protocol Version constants were utilizing a symbolic constant (now defined in header files) instead of hardcoded version numbers.

- Updated GAVD/AVDTP Protocol version constant to reflect latest specification version (1.2).

- Updated A2DP Profile version constant to reflect latest version (1.2).

- Updated AVRCP to include latest known IEEE 1394 A/V Pass-through command constants.

- Updated AVRCP to include AVRCP 1.4 Types.

- Updated AVRCP to be a library of helper functions to build/decode AVRCP Messages.

- Fixed/Added QWord_t Handling MACROs in BaseTypes.h.

- Fixed GAVD Profile such that Reconfigure Endpoint now correctly works, as well as fixing Query Endpoint Configuration such that it now behaves correctly.

- Update SBC Libary to support interleaving for both the Encoder and Decoder. This allows Interleaved Sample Data (left channel/right channel (or vice-versa)) to be passed instead of requiring that the data be in two contiguous buffers, one for each channel.

- Updated GAP and L2CAP to attempt Page Scan Mode R2 first. If it fails, then try Mode R1.

- Updated HID and HCR Profiles so that they fully support Security Level 2 (the ability to reject an incoming connection).

- Updated HCITypes.h and BTBTypes.h to support latest Bluetooth 3.0 + HS specification.

- Updated HCI.c and HCIAPI.h to support latest Bluetooth 3.0 + HS specification.

- Updated AVRCP to include building and parsing of ALL AVRCP 1.4 messages.

- Updated AVRCP sample applications to utilize new AVRCP Profile API.

- Added Ability to send arbitrary AT commands/responses over Hands Free Profile.

- Added functionality to HCIDRV to support passing through vendor specific commands/events in a more extensible manner.

- Windows

  - Updated sample Applications to show HCI Version of Local Chip.

  - Updated sample Application for Message Access Profile (MAP).

  - Updated sample Application for SYNC Profile (SYNC).

  - ❑ Updated all sample Applications (except StackHCI and StackSCO) to support Secure Simple Pairing as well as legacy pairing.

  - ❑ Updated all Windows Sample applications to center all dialogs that the applications display.

  - ❑ Fixed HCIUSB Library for Windows stack overflow in Receive Thread (exhibited itself as Stack Hanging when utilizing SCO Data over USB).

  - ❑ Fixed Windows USB Driver Isochronous streaming to actually specify the USB Frame Number for each individual Isochronous Packet. This fixes the problems with the XP stack regarding specifying the ASAP flag for Isochronous data. This also hase the benefit that audio playback is much, much, smoother under high load situations.

  - ❑ Updated all Windows samples to not use HCI Write Link Key. This entailed caching Link Keys and responding with the correct Link when requested. This will allow devices with no non-volatile storage to maintain pairing information (as the keys will be managed by the application). This has the benefit of also showing customers the accepted way of handling link key information.

  - ❑ Updated Windows USB Driver to support SCO Audio (Isochronous Transfers).

  - ❑ Updated Windows USB Driver to correctly support suprise removal (wasn't handled correctly before).

  - ❑ Updated Windows USB Driver to correctly handle Power Management (Suspend/Resume). It should now pass Plug and Play and WQHL tests. Also did all development under Driver Verifier with all tests on to find/fix more issues.

  - ❑ Updated Windows USB Driver to support 64 bit Vista. This also entailed creating an installation file (.inf) to support 64 bit Vista.

  - ❑ Miscellaneous Windows USB driver updates to improve stability when driver removed/dongle removed/device closed/etc.

  - ❑ Removed unused button "Connection State" from Windows StackAVC sample because it was unused. Replaced it with "Send Message" button and added new edit box for Class of Device and Local Name.

  - ❑ Updated Windows StackAVR sample to actually utilize the Category 1 4 check boxes when adding the SDP record to the SDP database. Prior to this, the value used was hard-coded.

  - ❑ Updated Windows SCO Audio Samples (StackSCO, StackHDS, and StackHFR) to allow more options for SCO Audio processing. The options available are SCO Transport (CODEC/HCI), PCM Format (8 bit/16 bit), and Test Mode (None/1KHz Tone/Loopback). In addition to testing, this allows customers to see how to utilize SCO audio in an application.

  - ❑ Updated Windows sample application to link with new AVRCP 1.4 library (prior to this only AVCTP was linked as there was no AVRCP library).

- Windows CE

  - Updated to completely support EVERYTHING that Windows and Linux versions contain, EXCEPT for SCO over HCI support in the CE USB Driver. All other features are the same.

- Linux

  - Updated to completely support EVERYTHING that Windows and Windows CE versions contain.

  - Updated all Linux Sample Applications to support SS1BTDBG (ability to output debug logs).

  - Updated Linux SCO Audio Samples (LinuxSCO, LinuxHDS, and LinuxHFR) to allow more options for SCO Audio processing. The options available are SCO Transport (CODEC/HCI), PCM Format (8 bit/16 bit), and Test Mode (None/1KHz Tone/Loopback). In addition to testing, this allows customers to see how to utilize SCO audio in an application.

  - Updated sample Applications to show HCI Version of Local Chip.

  - Updated sample Application for Message Access Profile (MAP).

  - Updated sample Application for SYNC Profile (SYNC).

  - Updated all sample Applications (except LinuxHCI and LinuxSCO) to support Secure Simple Pairing as well as legacy pairing.Updated all Linux sample applications to support new commands to be able to Set the Discoverability Mode, Connectability Mode, and Pairability Mode.

  - Updated all Linux sample applications to support new commands/functionality:

    - Get/Set Local Name

    - Get/Set Class of Device

    - Get Remote Name

    - Ability for Dedicated/General Bonding

    - Add startup code from other sample applications (version, master/slave r/s etc)

    - Ability to specify all RFCOMM Ports (instead of hardcoded port number)

    - More Inquiry Results

    - Level 4 Security Secure Simple Pairing

    - Settable PIN codes/PassKeys (prior versions were hardcoded)

    - Display the Inquiry Result List

    - No longer use HCI_Write_Link_Key() to store Link Keys into the chip locally. Now the Link Keys are stored in the application and the correct Link Key is supplied during Link Key Requests. This allows all Linux samples to have the same functionality that is available in the Windows samples.

- ❑ Tested Linux Virtual Serial Port Driver on 2.6 and updated installer.

- ❑ Updated SS1SER for Linux to use the defined Serial Port Prefix (defined in BTPSCFG.h) instead of hardcoding the name to ttyS (which is the default Linux name).

- ❑ Updated LinuxSCO and StackSCO applications to only print out received SCO Data Indications every second to avoid overwhelming the user with displayed messages. Also added commented out code (and comments) to show how to either loop back the SCO data or how to insert live data into the system.

- ❑ Updated LinuxHDS and StackHDS applications to only print out received Audio Data Indications every second to avoid overwhelming the user with displayed messages. Also added commented out code (and comments) to show how to either loop back the Audio data or how to insert live data into the system.

- ❑ Updated LinuxHFR and StackHFR applications to only print out received Audio Data Indications every second to avoid overwhelming the user with displayed messages. Also added commented out code (and comments) to show how to either loop back the Audio data or how to insert live data into the system.

- ❑ Updated all Linux Makefiles that use the archiver to use $(AR) instead (and define it if not defined) so that cross compilations can change the archiver (if required)

- ❑ Updated Linux BaseTypes.h to not assume Host Little Endian MACRO's.

Bluetopia Version 2.0 Changes (from Version 1.3.2)

- General
  - ❑ Updated API documentation.
  - ❑ Fix for BLU-59 – 'HFRE 1.5 profile stops reporting signal strength indicator.'
  - ❑ Fix for BLU-55 – ' HFRE 1,5 Profile Sample application does not demonstrate proper handling of event etHFRE_Response_Hold_Status_Indication'
  - ❑ Fix for BLU-54 – 'PBAP_PullPhoneBook_Request PBAP Client function fails with some PBAP Server implementations'
  - ❑ Fix for BLU-52 – 'HFRE 1.5 Profile does not correctly support simultaneous AG and HF audio connections'
  - ❑ Fix for BLU-52 – 'L2CAP Get info doesn't return all information correctly'. GES stack variants nly.

Bluetopia Version 1.3.3 Changes (from Version 1.3.2)

- General
  - ❑ Updated API documentation.
  - ❑ Added HCI_LMP_CLASS_OF_DEVICE_MAJOR_DEVICE_CLASS_TOY device class constants.
  - ❑ Added HCI_LMP_CLASS_OF_DEVICE_MINOR_DEVICE_CLASS_TOY… device class constants.
  - ❑ Added more HCI_LMP_COMID_MANUFACTURER_NAME_... constants to reflect most current Manufacturer List that have been added to the Bluetooth Assigned Numbers Document.
  - ❑ Added new SDP_ATTRIBUTE_ID_… constants to reflect most current Attribute ID constants that have been added to the Bluetooth Assigned Numbers Document.
  - ❑ Added Basic Imaging Profile (BIP) and sample applications to distribution.
  - ❑ Added Basic Printing Profile (BPP) and sample applications to distribution.
  - ❑ Added SDP_ATTRIBUTE_ID_DID_... SDP Attribute ID constants and the PNP Information UUID to the SDP constants to support Device Identification (DID).
  - ❑ Fixed calculation of the number of packets when a multi-segment signaling packet is required in GAVD. This problem would manifest itself by incorrectly calculating the number of signaling packets that would be needed for segmentation when the segmentation would result in more than one packet.
  - ❑ Added Bluetooth 2.0 packet type bit constants (these are different in that these bit values specify to NOT use the specified packet type). All other packet type bit constants specify the packet types to use.
  - ❑ Fixed IS_WHITE_SPACE() MACRO in the Hands Free Profile as well as internally in the stack to correctly take into account the range from the TAB character to the carriage return character. Prior to this, the range was incorrectly the TAB character to the new line character.
  - ❑ Fixed incorrect parenthesis in the OBEX_APPLICATION_PARAMETERS_SIZE() MACRO in GOEPAPI.h.
  - ❑ Updated HID sample applications to utilize best effort delivery service types. This allows the sample applications to work (with no changes) with some HID devices on the market (for instance the Inside/Out Keyboard).
  - ❑ Updated GAVD Profile to support reporting of channel empty indications on the channels it supports. This allows for more manageable flow control to be utilized on devices that do not support any other mechanism for flow control. This fix works in conjunction with the L2CAP Queuing mechanism. Note that this feature is not enabled by default in our sample applications and/or profiles.

- ❑ Updated the HID Profile, ProcessHIDTransaction() function to set the L2CAP Connection State to Ready before making the event callback for HID_TRANSACTION_HEADER_TRANSACTION_TYPE_HANDSHAKE and HID_TRANSACTION_HEADER_TRANSACTION_TYPE_DATA transactions. Prior to this, this information was updated after the callback which could cause problems when certain functions were called within the callback.

- ❑ Added ability to flush SPP transmit buffers (and be notified via Callback when buffer is empy complete). This functionality is utilized by calling the SPP_Purge_Buffer() function.

- ❑ Modified to allow use of Bluetooth 2.1 devices with stack.

- ❑ Implemememented Host to Host Control packet flow control. Note that this now requires applications riding directly on top of L2CAP and RFCOMM to monitor for scenarios where data is not completely written to the HCI device and respond appropriately. Applications based on the SPP profile should not require changes, as they will inherit the handling implemented in the SPP layer.

- ❑ Implemented profile cleanup registration functionality.

- ❑ Revised Licensing validation mechanism.

- ❑ Corrected potential overrun issue in GOEP layer

- ❑ Incorporated capability of handling multiple application TLV's in GOEP layers,

- ❑ Added File I/O Abstraction layer to isolate changes required to support OBEX and FTP profiles in embedded applications.

- ❑ Incorporated defined call outs during stack initialization to support vendor and device specific initialization requirements.

- ❑ Fixed bug in HID profile handling of ptBoot/btReport mode setting.

- ❑ Incorporated Hands Free 1.5 profile support.

- ❑ Incorporated AVRCP and associated profile support.

- ❑ Incorporated A2DP profile support.

- Windows

  - ❑ Updated Virtual Serial Port Device Driver for plug and play support (bug fixes)

  - ❑ Updated Virtual NDIS driver to support NDIS 5.0 deserialized Miniport.

  - ❑ Added Virtual NDIS driver installation instructions and utility.

  - ❑ Updated FTP profile such that when it returns the time and date for a directory entry (Server only) that it now correctly specifies that UTC time is used (in the callback). Prior to this, UTC time was used, but this flag was not set when the building the entry.

  - ❑ Added BTCOMM_Purge_Buffer() function to the BTCOMM Profile.

- Windows CE
    - ❑ Fixed a bug in the FTP Profile regarding the calculation of the required size of the FTP Directory Entries structure in the etOTP_Get_Directory_Response event. It now correctly calculates the size required for the allocation. Previously, the incorrect calculation could result in memory corruption in certain circumstances.
    - ❑ Updated FTP profile such that when it returns the time and date for a directory entry (Server only) that it now correctly specifies that UTC time is used (in the callback). Prior to this, UTC time was used, but this flag was not set when the building the entry.
    - ❑ Updated VCOM Driver with miscellaneous bug fixes.
    - ❑ Added BTCOMM_Purge_Buffer() function to the BTCOMM Profile.
- Linux
    - ❑ Added BTSER_Purge_Buffer() function to the BTSER Profile.

Bluetopia Version 1.3.2 Changes (from Version 1.2.7)

- General
    - ❑ Updated API documentation.
    - ❑ Added support for Bluetooth Version 2.0 + EDR.
    - ❑ Added more HCI_LMP_COMID_MANUFACTURER_NAME_... constants to reflect most current Manufacturer List that have been added to the Bluetooth Assigned Numbers Document.
    - ❑ Added HCI_LMP_CLASS_OF_DEVICE_MAJOR_DEVICE_CLASS_WEARABLE device class constants.
    - ❑ Fixed bug with the Headset profile in the processing of unknown (Vendor Specific) Headset profile Commands/Responses. This bug when manifest itself by being unable to receive any data after the reception of an unknown (Vendor Specific) Command/Response.
    - ❑ Updated Headset profile to allow Ring Indication Events to be allowed even when In-band Ringing was specified.
    - ❑ Fixed OTP_Open_Server_Port() function to return the OTPID. Previously it was returning the Identifier of the lower layer function that it used. An issue with this would only manifest itself if both the OTP layer and the GOEP layer Open Server Port function are being used.
    - ❑ Fixed an issue with the Generic Access Profile (GAP) that could result in outstanding remote name requests or authentication requests not being cleaned up in the event of a Bluetooth Device Reset.

- ❑ Updated OTP layer to allow for long names in the OTP_ObjectInfo_t structure. Previously a limit of 64 characters existed on the name member in this structure.

- ❑ Updated File Transfer Profile to handle file names longer then 64 character return from the OTP layer via the OTP_ObjectInfo_t structure.

- ❑ Added functionality to SCO to allow for the ability to change the packet types. This is particularly useful for Bluetooth Devices that report incorrect values for their SCO Buffer capabilities.

- ❑ Fixed a bug that would cause the Link Supervision Timeout to not be set under certain circumstances.

- ❑ Fixed a potential Memory Leak in the L2CAP layer that could manifest itself under certain circumstances when a remote device disconnects.

- ❑ Fixed SBC compatibility issue allowing the library to run on processor that report the sizeof(int) equal two.

- ❑ Change the error handling in the Serial Port Profile (SPP) pertaining to the way that it handles a failure to send a Modem Status during the creation of a connection.

- ❑ Fixed a bug in the OTP layer that could cause memory to be accessed after it had been freed under very specific circumstances.

- ❑ Improved BCSP handling of devices that do not support the Garrulous state.

- ❑ Fixed memory leak with the SIM Access Profile (SAP) that could occur if a connection to a remote server failed.

- ❑ Fixed bug with StackPAN and PocketPAN sample applications that would result in the PAN SDP record not being un-registered if PAN Cleanup was called which a PAN server was open.

- ❑ Improved SCO Layer compatibility with Bluetooth device vendors that no longer support the HCI_Add_SCO HCI Command.

- Windows

  - ❑ Fixed Resource leak of Thread Handle.

  - ❑ Fixed bug with the Bluetooth USB Device Driver that would cause some systems to not shutdown if a Bluetooth Device was inserted.

  - ❑ Added functionality to the Virtual COM Port Driver to allow its properties to be changed from the Windows Device Manager.

- Windows CE

  - ❑ Fixed Resource leak of Thread Handle.

  - ❑ Fixed bug with the PocketPAN sample applications that resulted in the Open Server/Close Server button not being correctly updated.

Bluetopia Version 1.2.7 Changes (from Version 1.2.6)

- General

  - ❑ Updated API documentation.

  - ❑ Added more HCI_LMP_COMID_MANUFACTURER_NAME_... constants to reflect most current Manufacturer List that have been added to the Bluetooth Assigned Numbers Document.

  - ❑ Updated HCI_LMP_COMID_MANUFACTURER_NAME_... constants to reflect Manufacturer List name changes that have occurred (Mitel Semiconductor, Telencomm, Inc, Gennum Corportation, Research in Motion, and the reserved LMP Testing value).

  - ❑ Updated SDP_OS_ID_STRING_… constants to reflect most current OS Identification string list that have been added to the Bluetooth Assigned Numbers Document.

  - ❑ Added SDP_LINUX_DISTRIBUTION_ID_STRING_LINUXWARE constant to the SDP_LINUX_DISTRIBUTION_ID_STRING list constants.

  - ❑ Updated L2CAP sample applications (StackChat, PocketChat, LinuxL2CAP, and QNXL2CAP) to support the case in the etConnect_Confirmation event when the remote device returns the status of PENDING. Prior to this update, the PENDING status was incorrectly parsed as a connection error and the connection was incorrectly flagged as not present.

  - ❑ Fixed bug in Hardcopy Cable Replacement Profile regarding the HCR_Data_Write() function. This function incorrectly calculated the amount of data to send on each pass through the loop. This would result in sending extraneous data under certain circumstances. This has been fixed to now send only the data that is passed to the function.

  - ❑ Updated BTCOM/BTSER Profile to allow the options of mapping DSR to DCD on the outgoing SPP Port Status. This allows for emulation of a true NULL Modem cable where these lines are physically tied together in hardware.

  - ❑ Updated the LAN Access Profile to force the mapping DSR to DCD on the outgoing SPP Port Status. This allows for emulation of a true NULL Modem cable where these lines are physically tied together in hardware.

  - ❑ Fixed typographical error in the GAP API Header file (GAPAPI.h) regarding the definition of the GAP_AUTHENTICATION_INFORMATION_SIZE constant. Prior to this update the word Authentication was not spelled correctly.

  - ❑ Added capability to OBEX FTP Profile to allow the programmatic disconnection of a currently connected client to an OBEX FTP Server. Prior to this, no function existed to force the disconnection of the connected OBEX FTP Client. The FTP_Close_Server_Connection() function was added to implement this functionality.

  - ❑ Added capability to OBEX Object Push Profile to allow the programmatic disconnection of a currently connected client to an OBEX Object Push Server.

Prior to this, no function existed to force the disconnection of the connected OBEX Object Push Client. The OBJP_Close_Server_Connection() function was added to implement this functionality.

- Fixed problem in OTP_Close_Port() function that incorrectly set the OBEX Server type to Unknown when this function was called on a currently connected OBEX Server Port. This bug when manifest itself, by not accepting any further connections on the Port and responding to a connection request with the OBEX_BAD_REQUEST_RESPONSE error code. This previously was not a problem as none of the OBEX Profiles would allow the disconnection of a currently connected client to an OBEX Server. The newest OBEX Profiles allow this functionality, which exposed this problem (see above).

- Changed all UUID's used by all of the SDP functions that insert SDP records into the SDP Database to use 16-bit UUIDs instead of 128-bit UUIDs. This was done because some implementations do not correctly support 128-bit UUIDs, even though 128-bit UUIDs are mandatory. This does not mean that 128-bit UUIDs are not supported (they can still be added), it only means that the utility functions provided in all the profiles will add 16-bit UUIDs to the database instead of 128-bit UUIDs. This change involved all of the Profiles and also the SDP Database record itself (that is maintained by the SDP Database server). As stated previously, the SDP Client and Server still support 128-bit UUIDs the same as they did previously.

- Fixed problem in the Headset profile that would manifest itself by not issuing a callback when the SCO connection was disconnected. This problem was caused in the handling of the etSCO_Disconnect event where the SCO Connection ID was incorrectly set to zero (to signify no connection) before the Audo Disconnect Event callback was issued. The resetting of this ID was moved further down in the code and now the correct callback is issued.

- Changed initialization of the default outgoing MTU to be L2CAP_DEFAULT_CONNECTION_MTU_SIZE instead of zero in the Generic Audio/Video Distribution (GAVD) Profile. Prior to this change, problems were encountered when the remote side of the the GAVD connection did not negotiate the MTU and relied on the L2CAP default. This caused problems because the profile incorrectly thought the outgoing MTU was zero instead of the default. This has been fixed by initializing the default outgoing MTU to the L2CAP default (which is correct).

- Fixed problem in Generic Audio/Video Distribution (GAVD) Profile which would not allow a reconfiguration on a previously configured stream endpoint to occur.

- Updated Generic Audio/Video Distribution (GAVD) Profile sample applications such that the Codec Specific Information that is written to the Capabilities and Configuration is more inline with the actual values used by the Advanced Audio Distribution Profile (A2DP) SBC Codec.

- Added event handler for the etGAVD_Reconfigure_Indication Event in the Generic Audio/Video Distribution (GAVD) Profile sample applications. This handler simply accepts any reconfiguration request that is received for an Endpoint by issuing a successful reconfiguration response.

- Updated Hands Free Profile so that the state information for the current connection is updated before dispatching the etHFRE_Open_Service_Level_Connection_Indication callback. This change allows the application to issue other Hands Free function calls from within the callback.

- Updated the Hands Free Profile such that when parsing Call Waiting Notification, Caller Identification Notification, and Attach Phone Number data streams to parse phone numbers that are not specified in quotation marks. It was found through testing that one manufacturer (incorrectly) did not place the phone number within quotation marks. Prior to this fix, the phone number was not parsed as it was looking for quotation marks to enclose the number (as per the specification). The profile has been changed such that it will also parse a phone number that is not enclosed in quotation marks.

- Fixed HID profile such that it now correctly initializes the default outgoing MTU to L2CAP_DEFAULT_CONNECTION_MTU_SIZE. This would only cause problem if the remote connection did not specify an MTU during the L2CAP configuration process. Now the default L2CAP connection MTU size is used in this case (as per the specification).

- Updated the serial based profiles (BTCOMM/BTSER/DUN/FAX/LAP) to use larger receive buffers internally. This will improve bandwidth because the largest chunk of data that will be read will now be the same size as the RFCOMM Frame size that is being used for the channel. Prior to this, the receive buffer size being used was 512 bytes, which wasn't very efficient on the Bluetooth side (this doesn't map well to the defined Bluetooth Packet sizes).

- Added new event to Generic Access Profile (GAP) such that Inquiry results are returned while they are in the process of being discovered. The event that was added is called etInquiry_Entry_Result. This event contains the newly found Bluetooth device information. Note that all the results are still returned when the GAP Inquiry is complete (this functionality has not changed). This feature was added so that graphical user interface applications could display the devices as soon as they were found, as opposed to waiting until the Inquiry procedure was complete and displaying the devices all at once. Also note, that this event doesn't impact the existing Inquiry process schema in any way, so backwards compatibility is fully maintained (in other words the etInquiry_Result event still functions in the same capacity as it always has).

- Fixed RFCOMM issue regarding the setting of the RFCOMM_CR_BIT when a server connection was accepted from a remote peer when a remote connection was made to the remote peer (i.e. the local device initiated the RFCOMM connection, however, the remote peer connected back to the local RFCOMM entity while the connection was still active). This bit was incorrectly handled in this case. This bit was only incorrectly handled in the scenario above, and was correctly handled in all other cases.

- Added ability to support Security Mode 2 (Service Level Enforced Security) in the Serial Port Profile (SPP). Prior to this, RFCOMM and L2CAP supported this functionality, however SPP did not easily support this scenario. The SPP_Set_Server_Connection_Mode() and SPP_Get_Server_Connection_Mode() functions were added to enable/query the parameters for this functionality. The etPort_Open_Request_Indication event was added, as well as the SPP_Open_Port_Request_Response() function to respond to a connection request. These new functions/events allow individual SPP Server Ports to be configured to inform via the registered callback when a connection has been requested, and allow the application the opportunity to perform whatever authorization/authentication it deems necessary. When the application decides that the connection is valid/invalid, it will then call the SPP_Open_Port_Request_Response() function to inform the remote side of the result.

- Added ability to support Security Mode 2 (Service Level Enforced Security) in the GOEP and OTP layers. This is simply an extension of the Serial Port Profile (SPP). See notes above for a description on the changes that were made to SPP. GOEP and OTP use the exact same mechanism, except that the names of the events and the API functions are named differently For example the SPP_Open_Port_Request_Response() function is called GOEP_Open_Port_Request_Response() in the GOEP API and OTP_Open_Port_Request_Response() in OTP API. The remaining events and functions have been renamed similarly (see GOEPAPI.h and OTPAPI.h and/or consult the documentation for more information).

- Added ability to support Security Mode 2 (Service Level Enforced Security) in the Serial based profiles (BTCOMM and BTSER). Prior to this, RFCOMM and L2CAP supported this functionality, however SPP and the profiles that existed above SPP did not easily support this scenario. The BTCOMM_Set_Server_Connection_Mode () , BTSER_Get_Server_Connection_Mode(), BTCOMM_Get_Server_Connection_Mode () and BTSER_Set_Server_Connection_Mode functions were added to enable/query the parameters for this functionality. The etBTCOMM_Open_Port_Request_Indication and etBTSER_Open_Port_Request_Indication events were added, as well as the BTCOMM_Open_Port_Request_Response () and BTSER_Open_Port_Request_Response() functions to respond to the connection request. These new functions/events allow individual COM/Serial Server Ports to be configured to inform via the registered callback when a connection has been requested, and allow the application the opportunity to perform whatever authorization/authentication it deems necessary. When the application decides that the connection is valid/invalid, it will then call the BTCOMM_Open_Port_Request_Response () or BTSER_Open_Port_Request_Response functions to inform the remote side of the result.

- Added ability to support Security Mode 2 (Service Level Enforced Security) in the OBEX File Transfer Profile (FTP). Prior to this, RFCOMM and L2CAP supported this functionality, however SPP and the profiles above it did not easily support this scenario. The FTP_Set_Server_Mode () and FTP_Get_Server_Mode () functions were added to enable/query the parameters for this functionality. The etFTP_Server_Connect_Request_Indication event was added, as well as the FTP_Server_Connect_Request_Response () function to respond to a connection request. These new functions/events allow individual FTP Servers to be configured to inform via the registered callback when a connection has been requested, and allow the application the opportunity to perform whatever authorization/authentication it deems necessary. When the application decides that the connection is valid/invalid, it will then call the FTP_Server_Connect_Request_Response () function to inform the remote side of the result.

- Added ability to support Security Mode 2 (Service Level Enforced Security) in the OBEX Object Push Profile (OBJP). Prior to this, RFCOMM and L2CAP supported this functionality, however SPP and the profiles above it did not easily support this scenario. The OBJP_Set_Server_Mode () and

OBJP_Get_Server_Mode () functions were added to enable/query the parameters for this functionality. The etOBJP_Server_Connect_Request_Indication event was added, as well as the OBJP_Server_Connect_Request_Response () function to respond to a connection request. These new functions/events allow individual Object Push Servers to be configured to inform via the registered callback when a connection has been requested, and allow the application the opportunity to perform whatever authorization/authentication it deems necessary. When the application decides that the connection is valid/invalid, it will then call the OBJP_Server_Connect_Request_Response () function to inform the remote side of the result.

- Fixed problem in OBEX Object Push Profile when the server received an Abort Request while the client was currently putting an Object. Prior to this fix, the ObjectFileName parameter of the OBJP_Server_Object_Put_Indication_Data member in the OBJP_Server_Event_Data of the etOBJP_Server_Object_Put_Indication Event was in correct. The name that was returned was that of the Default Object instead of the Object that was currently being put. This has been fixed, so that now, in the case of an Abort during a Put operation, the server will issue the correct callback with the correct ObjectFileName.

- Changed HCI_Create_Connection() call in the GAP_Initiate_Bonding() function so that it allows the remote device to perform Master/Slave Role Switch when the connection is accepted. Prior to this change, the HCI_Create_Connection() call in the GAP_Initiate_Bonding() function did not allow the remote side to become the Master and the bonding would fail when the remote side forced a Role switch when accepting the ACL connection. This change has little impact, as the connection is being made for Bonding/Pairing purposes.

- Fixed problem in OBEX Object Push Profile that prevented Objects of unknown types from being pushed (obtUnknownObject). Prior to this fix, when this type was specified as the type in the OBJP_Put_Object() function, the function would return an error. This has now been fixed, so that this function now accepts the unknown type. This allows other objects (such as pictures) to be pushed through the Object Push Profile (unknown objects were always able to be received).

- Added a new function to the Generic Access Profile (GAP) layer to allow the user the ability to determine the HCI ACL Connection Handle that is currently being used for an active connection to a remote device. The name of the function that was added is called GAP_Query_Connection_Handle() and can be found in the GAPAPI.h header file.

- Added a new error to the BTERRORS.h header file for the GAP_Query_Connection_Handle() function that was added to GAP. This new error that was added is named BTPS_ERROR_DEVICE_NOT_CONNECTED.

- Fixed RFCOMM such that it now only disconnects the errant channel during connection establishment (if it can be determined) instead of disconnecting all channels.

- Added code to RFCOMM such that whenever information is received that is destined for a channel that is not currently in use, a Disconnect Mode (DM) is issued on that channel.

- Fixed problem in SBC Decoder which could cause spikes in the decoded data. This problem occurred only in the decoder and was caused by an overflow problem when decoding the data to a 16 bit signed value. In some cases the decoded sample value would not fall within the range of a signed 16 bit integer. This overflow condition has now been fixed.

- Updated the serial based profiles (BTCOMM and BTSER) to allow the ability for the Bluetooth application to insert data into the actual serial stream (the stream that was being routed between the Bluetooth Stack and the Virtual COM/Serial Port. The BTCOMM_Data_Write() function was added to the BTCOMM profile and the BTSER_Data_Write() function was added to the BTSER profile.

- Fixed problem in Serial Port Profile (SPP) when processing the etRemote_Port_Negotiation_Indication event from RFCOMM. The buffer that was declared to hold the Port Negotiation parameters was not large enough to hold the entire event that was dispatched from SPP. The size of this buffer has now been increased such that the entire event can be built successfully and not overrun the memory buffer.

- Fixed problem in Generic Access Profile (GAP) regarding General Bonding. Prior to this fix, it was possible for an Authentication Status Event to be dispatched when the ACL Link was terminated (even though the Authentication Status Event had already been dispatched). This has been fixed, so that now the Authentication Status Event is only dispatched at most one time during General Bonding. This problem did not occur during Dedicated Bonding.

- Added support for Bluetooth Version 1.2 HCI Commands/Events.

- Added new types to support new types defined in Bluetooth Version 1.2 HCI. This primarily involved adding the Supported Commands type (Supported_Commands_t) and the AFH Channel Map type (AFH_Channel_Map_t). These types and utility MACROs can be found in the BTTypes.h file.

- Improved handling of Continuation Bit in L2CAP Configuration Requests/Responses. This functionality was updated because in Bluetooth Version 1.2, it is possible that this feature would have to be used (if the configuration information would not fit within the signaling MTU). This is now handled by a new flag (L2CA_CONFIG_OPTION_FLAG_CONTINUATION) that was added to the Option_Flags member of the L2CA_Config_Request_t, L2CA_Config_Confirmation_t, L2CA_Config_Indication_t, and L2CA_Config_Response_t structures.

- Added new Bluetooth Version 1.2 L2CAP Information type support for Extended Features. This entailed adding support within L2CAP and adding the constant L2CAP_INFORMATION_REQUEST_INFOTYPE_EXTENDED_FEATURE_MASK. This also entailed adding the L2CAP_Extended_Feature_Mask_t type definition (and support MACROs) to the BTTypes.h file.

- Updated code in GAP when bonding to correctly work on Bluetooth Devices that will not let Authentication occur on an already encrypted link (as the Link Key must already have been generated). This change was made because some Bluetooth Devices still exhibit this behavior (although only two have been found), even though there is an Errata that has been accepted to address this issue which says that Authentication can occur on an encrypted link (because there is no physical or technical reason why this could not occur).

- Added the RFCOMM_Get_Channel_Status() function which allows the programmer a mechanism to query the current state of an individual RFCOMM Channel for a specific Bluetooth Device Address. This function also allows the programmer the ability to determine the current Control Channel State for a specific Bluetooth Device Address. This functionality was added to allow programmers a means to determine when an RFCOMM Channel has been completely disconnected, as well as to determine when there is an outstanding message on a specific Channel (to aid with new connections).

- Added the SPP_Get_Port_Connection_State() function which allows the programmer a mechanism to query the current state of an individual SPP/RFCOMM Channel for a specific Bluetooth Device Address. This function also allows the programmer the ability to determine the current Control Channel State for a specific Bluetooth Device Address. This functionality was added to allow programmers a means to determine when an SPP Port/RFCOMM Channel has been completely disconnected, as well as to determine when there is an outstanding message on a specific Channel (to aid with new connections).

- Fixed problem in OTP regarding the handling of OBEX Put Requests with headers split over multiple OBEX packets before the reception of the body. Prior to this fix the remote device would be disconnected if headers were received in this manner.

- Fixed problem in Generic Object Exchange Profile (GOEP) when processing the etPort_Close_Port_Indication event from SPP. Prior to this fix, if a remote OBEX client disconnected from a local OBEX server while building an OBEX packet split over multiple SPP reads, i.e. OBEX Put Request, the next connection attempt would fail.

- Fixed L2CA_Get_Timer_Values() function to return a value to zero upon successful execution. Prior to this fix, the return value was not being set as documented.

- Fixed OTP_Delete_Sync_Object_Response() and OTP_Put_Sync_Object_Response() functions to set the return value in a case where a memory allocation fails. Prior to this fix, the return value was not being set correctly.

- Fixed problem in File Transfer Profile (FTP) and Object Transfer Profile (OBP) when handling Abort Requests. Prior to this fix, if an Abort Request were immediately followed by another Command before the Abort Response was received the Abort Response would be improperly matched as the Command Response.

• Windows

  - Added __DLLMODEF__ definition to the L2CA_Flush_Channel_Data() function declaration in L2CAPAPI.h. The function was exported correctly because it was correctly specified in the DEF file for export, however, callers of the function could have had potential problems calling it because it was not declared as an imported function.

  - Updated command parser in the Console SPP (CONSSPP) sample application to be more in line with the command parser used in the Linux and QNX sample applications.

- ❑ Fixed spelling error in the OBEX Object Push Profile SDP constants SDP_OBJECT_SUPPORTED_FEATURE_VCALENDAR1_0 and SDP_OBJECT_SUPPORTED_FEATURE_ICALENDAR1_0. Prior to this fix, the spelling of 'CALENDAR' was incorrect ('CALANDER' was used for these two constants). This spelling error was only for the above two constants, all other spellings of 'CALENDAR' were correct. This entailed changing the constants in the Header File (OBJPAPI.h) and the locations they were used in the source file (OBJP.c).

- ❑ Fixed bug in the OBEX FTP Profile (server only) that would incorrectly return "." and ".." as directories on Windows XP. Also, changed the comparison code such that any file or directory that begins with a "." as the first character will now be returned. Prior to this fix, any file or directory that contained "." or ".." as the first character(s) was/were not returned.

- ❑ Updated HCI COM Driver (HCICOMM.DLL) such that it now supports COM Port numbers greater than 9. Prior to this, COM Port numbers greater than 9 were supported, however, only if a DOS Device was installed for the COM Port. Now, the full Windows WIN32 device name is used to allow the COM Port to be opened.

- ❑ Updated COM Port Library (SS1COM.DLL) such that it now supports COM Port numbers greater than 9. Prior to this, COM Port numbers greater than 9 were supported, however, only if a DOS Device was installed for the COM Port. Now, the full Windows WIN32 device name is used to allow the COM Port to be opened.

- ❑ Updated L2CAP so that it will respond to the Information Request for the Extended Features Mask. This request was added in the Bluetooth 1.2 specification so that clients can determine if the remote device's Bluetooth Protocol Stack supports enhanced features of Bluetooth 1.2.

- ❑ Updated BTTypes.h such that it defines the L2CAP Extended Features Mask data type.

- ❑ Updated L2CAPTyp.h such that it defines the Bluetooth 1.2 Information Request for Extended Features mask. This entailed defining the Information Request type constant as well as defining the currently defined Bluetooth 1.2 Extended Features Mask bit values.

- ❑ Updated USB Driver such that the National Semiconductor Bluetooth device now opens correctly. This entailed removing the USB Port Reset URB that was sent when the device was initialized the very first time.

- ❑ Fixed problem in Virtual COM Port Device Driver that caused some Windows XP machines to freeze when the actual COM port was opened by an application.

- ❑ Added Personal Area Network Profile (PAN). This also entails sample applications and Virtual NDIS Driver. Note that the driver is only supported under Windows 2000/XP.

- Windows CE

  - ❑ Updated HCI COMM Driver to support COM0: as a valid device. Windows CE supports devices numbered 0 through 9.

1.

- ❑ Updated all sample applications to support COM0: as a valid COM Port Device.

- ❑ Added support for HP iPAQ 5450 and HP iPAQ 5555 internal Bluetooth Module. This entailed changing the existing Compaq iPAQ 3870 and Compaq iPAQ 3970 Bluetooth Driver.

- ❑ Added BTCOMM, Dial-up Networking, FAX, and LAN Access Profiles. This also entails sample applications as well as Virtual COM Port Installation/Configuration for the Virtual COM Port Driver.

- ❑ Fixed bug in OBEX File Transfer Profile that would return an OBEX error when the contents of an empty directory were listed. No the server correctly returns success with no files present.

- ❑ Added __DLLMODEF__ definition to the L2CA_Flush_Channel_Data() function declaration in L2CAPAPI.h. The function was exported correctly because it was correctly specified in the DEF file for export, however, callers of the function could have had potential problems calling it because it was not declared as an imported function.

- ❑ Implemented power management detection in the HCI Driver. When the HCI Driver determines that the device has been suspended (only when a port is open), it will re-open the port when power is restored, and dispatch an Event through the stack to allow the application programmer to know that the power state changed. The event that is dispatched is the etDevice_Power_Event. This event was added to the HCI layer (HCIAPI.h), so this event will only be received if Asynchronous HCI Events have been registered for. To utilize the power management facility, a Power Management Device must be installed into the system. The HCI Driver will use the services of this device if it is installed into the system, and ignore the Power Management detection if the device is not found in the system.

- ❑ Fixed bug in the OBEX FTP Profile (server only) that could incorrectly return "." and ".." as directories. Also, changed the comparison code such that any file or directory that begins with a "." as the first character will now be returned. Prior to this fix, any file or directory that contained "." or ".." as the first character(s) was/were not returned.

- ❑ Fixed problem in the OBEX Object Push profile in the OBJP_Register_Server_SDP_Record() function that incorrectly wrote to memory that was not allocated (in some circumstances) when building the Supported Formats list. The fix simply entailed breaking out the loop when all supported features had been built into the list (instead of continuing through the loop).

- ❑ Added Personal Area Network Profile (PAN). This also entails sample applications, Virtual NDIS Driver Install/Remove application and Virtual NDIS Driver.

• Linux

- ❑ Improved Lock File handling in HCI Drivers to be more robust. This primarily fixed a problem that would cause the Lock File to be deleted when the Lock belonged to a Process that no longer existed. This problem would manifest itself if the application was closed and the stack was not closed, leaving the Lock file intact. Then, when the stack was opened, it would find the Lock file and determine that the process that owned the Lock was no longer in existence (which was correct), however the Lock file was incorrectly deleted and not restored in this case.

- ❑ Updated Hands Free sample application such that the commands to open a Hands Free Client or Audio Gateway Client now accept a parameter to specify which RFCOMM Port to connect with. Prior to this update, this sample application used a constant value for the RFCOMM Port to connect to.

- ❑ Updated Hands Free sample application such that the event handlers for the etHFRE_Control_Indicator_Status_Indication and etHFRE_Control_Indicator_Status_Confirmation events now print out a carriage return/line feed when displaying the data. This improves the readability and aesthetics when these events are received.

- ❑ Updated sample applications with fixes to the command parser. This involved making sure that overruns did not occur when parsing input data and also adding code to exit the application if fgets() returned an error. It was found that in some situations, fgets() would fail if the window was closed (the 'X' button was pressed), however, it would treat the failure as no input, and loop back around to accept more input. The problem, however, was that the application would not shutdown because it was still running in this loop. By exiting the application when an error occurs with fgets() this problem is no longer encountered.

- ❑ Updated the HCICOMM Driver to include new Baud Rate mappings for higher speed UARTs.

- ❑ Fixed spelling error in the OBEX Object Push Profile SDP constants SDP_OBJECT_SUPPORTED_FEATURE_VCALENDAR1_0 and SDP_OBJECT_SUPPORTED_FEATURE_ICALENDAR1_0. Prior to this fix, the spelling of 'CALENDAR' was incorrect ('CALANDER' was used for these two constants). This spelling error was only for the above two constants, all other spellings of 'CALENDAR' were correct. This entailed changing the constants in the Header File (OBJPAPI.h) and the locations they were used in the source file (OBJP.c).

- ❑ Fixed bug in the OBEX FTP Profile (server only) that would prevent files or directories that began with "." or "..". The comparison code was changed such that any file or directory that begins with a "." or ".." as the first character will now be returned. Prior to this fix, any file or directory that contained "." or ".." as the first character(s) was/were not returned.

- ❑ Updated Hands Free Sample application to clarify the client connection nomenclature and comments. Also updated the GAP_Event_Callback() function such that it no longer prints out that it received a GAP Event that it is not interested in.

• QNX

- ❑ Added new fstat() mode comparison test of S_ISNAM() to test for the QNX specific special Named File type in the OBEX Object Push and OBEX File Transfer Profiles. The QNX SH4 Biscayne platform uses this type of flag for files that are created in the /tmp directory (which is actually a RAM Disk).

- ❑ Improved Lock File handling in HCI Drivers to be more robust. Improved Lock File handling in HCI Drivers to be more robust. This primarily fixed a problem that would cause the Lock File to be deleted when the Lock belonged to a Process that no longer existed. This problem would manifest itself if the application was closed and the stack was not closed, leaving the Lock file intact. Then, when the stack was opened, it would find the Lock file and determine that the process that owned the Lock was no longer in existence (which was correct), however the Lock file was incorrectly deleted and not restored in this case.

- ❑ Updated Hands Free sample application such that the commands to open a Hands Free Client or Audio Gateway Client now accept a parameter to specify which RFCOMM Port to connect with. Prior to this update, this sample application used a constant value for the RFCOMM Port to connect to.

- ❑ Updated Hands Free sample application such that the event handlers for the etHFRE_Control_Indicator_Status_Indication and etHFRE_Control_Indicator_Status_Confirmation events now print out a carriage return/line feed when displaying the data. This improves the readability and aesthetics when these events are received.

- ❑ Updated sample applications with fixes to the command parser. This involved making sure that overruns did not occur when parsing input data and also adding code to exit the application if fgets() returned an error. It was found that in some situations, fgets() would fail if the window was closed (the ‘X’ button was pressed), however, it would treat the failure as no input, and loop back around to accept more input. The problem, however, was that the application would not shutdown because it was still running in this loop. By exiting the application when an error occurs with fgets() this problem is no longer encountered.

- ❑ Updated HCICOMM Driver to pass unsupported Baud Rates (Baud Rates that do not have constants defined in the QNX header files) directly to the Serial Driver. This should allow higher speed UART drivers to function correctly, as prior to this, the driver would not even be attempted to be opened.

- ❑ Fixed spelling error in the OBEX Object Push Profile SDP constants SDP_OBJECT_SUPPORTED_FEATURE_VCALENDAR1_0 and SDP_OBJECT_SUPPORTED_FEATURE_ICALENDAR1_0. Prior to this fix, the spelling of ‘CALENDAR’ was incorrect (‘CALANDER’ was used for these two constants). This spelling error was only for the above two constants, all other spellings of ‘CALENDAR’ were correct. This entailed changing the constants in the Header File (OBJPAPI.h) and the locations they were used in the source file (OBJP.c).

- ❑ Fixed bug in the OBEX FTP Profile (server only) that would prevent files or directories that began with “.” or “..”. The comparison code was changed such that any file or directory that begins with a “.” or “..” as the first character will now be returned. Prior to this fix, any file or directory that contained “.” or “..” as the first character(s) was/were not returned.

- ❑ Updated Hands Free Sample application to clarify the client connection nomenclature and comments. Also updated the GAP_Event_Callback() function such that it no longer prints out that it received a GAP Event that it is not interested in.

Bluetopia Version 1.2.6 Changes (from Version 1.2.5)

- General
  - ❑ Updated API documentation.
  - ❑ Added code in Generic COM/Serial Port, Dial-up Networking, FAX, and LAN Access Profiles such that when a server channel is closed (but not un-registered) the Modem Status Signals are all reported as non-asserted.
  - ❑ Changed code in Headset Profile such that if a buffer overrun is detected when building a Command an Error Response is returned. Previously, no response was returned and no more commands could ever be received unless the remote side disconnected and reconnected.
  - ❑ Added code in L2CAP to send a L2CAP Connection Response Refusal (No Resources) when a Client attempts a connection and the Connection could not be added to the system because of lack of resources. Previous to this fix, this case (although extremely rare) did not send a Connection Response back to the Connection originator.
  - ❑ Implemented Erratum E1275 which involved the HCI_Read_Voice_Settings() and the HCI_Write_Voice_Settings() functions. This entailed adding HCI_VOICE_SETTING_INPUT_DATA_FORMAT_UNSIGNED and HCI_VOICE_SETTING_AIR_CODING_FORMAT_NONE constants for use with the Voice Settings Parameter of the HCI_Write_Voice_Setting() and the HCI_Read_Voice_Setting() functions.
  - ❑ Updated SCO Module to support Erratum E1275. This involved adding efUnsigned to the SCO_Data_Encoding_Format_t enumerated type and aeNone to the SCO_Air_Encoding_Type_t enumerated type. These values are used with the SCO_Query_Data_Format() and the SCO_Change_Data_Format() functions.
  - ❑ Added more HCI_LMP_COMID_MANUFACTURER_NAME_... constants to reflect most current Manufacturer List that have been added to the Bluetooth Assigned Numbers Document.
  - ❑ Added new SDP_ATTRIBUTE_ID_… constants to reflect most current Attribute ID constants that have been added to the Bluetooth Assigned Numbers Document.
  - ❑ Added more SDP Attribute ID Tags constants to the SDPTypes file to reflect some of the more recent profiles. Most notably, the HID and Hardcopy Cable Replacement Profiles.
  - ❑ Added Human Interface Device (HID) Profile and sample applications to distribution.
  - ❑ Added Hardcopy Cable Replacement Profile (HCRP) and sample applications to distribution.
  - ❑ Added SIM Access Profile (SAP) and sample applications to distribution.

- Added Generic Audio/Video Distribution Profile (GAVD) and sample applications to distribution.

- Added Advanced Audio Distribution Profile (A2DP), Subband Codec (SBC), and sample applications to distribution.

- Exposed subset of the Bluetopia asynchronous timers so that profiles are able to implement timing in a consistent manner. All timer functionality is defined in the BSC Module. The functions that were added are the BSC_StartTimer() and BSC_StopTimer() functions.

- Changed case-sensitive comparisons to case-insensitive comparisons in OBEX Object Push Profile when trying to determine the type of Object being pushed/pulled.

- Fixed bug in Audio Gateway portion of the Headset Profile where the Microphone and Speaker Gain were sent to the remote Headset incorrectly. Audio Gateways are supposed to send these commands as unsolicited, which means that the terminating characters need to be a carriage return/line feed, instead of simply a carriage return. These changes were in the HDSET_Set_Speaker_Gain() and HDSET_Set_Microphone_Gain() functions.

- Added code in OBEX Object Push Profile to make sure that only one etOBJP_Server_Disconnect_Indication Event is dispatched when the remote client Disconnects from the OBEX Object Push Server, and then Closes the OBEX Port (individually). Prior to this fix, two etOBJP_Server_Disconnect_Indication Events would be dispatched when this circumstance occurred. Now, a single etOBJP_Server_Disconnect_Indication is dispatched in this circumstance.

- Added code in OBEX FTP Profile to make sure that only one etFTP_Server_Disconnect_Indication Event is dispatched when the remote client Disconnects from the OBEX FTP Server, and then Closes the OBEX Port (individually). Prior to this fix, two etFTP_Server_Disconnect_Indication Events would be dispatched when this circumstance occurred. Now, a single etFTP_Server_Disconnect_Indication is dispatched in this circumstance.

- Fixed bug in Hands Free Profile such that the Hands Free module no longer responds with “ERROR” responses for unsolicited responses. In fact, the Hands Free Module never will respond to unsolicited responses from the Audio Gateway. An example of this would be if the unsolicited response “NO CARRIER” was received from the Audio Gateway. Prior to this fix, the Hands Free module would have responded with “ERROR” because it did not know how to process this. Now, no response is sent, per the specification.

- Added Command Confirmation event to the Hands Free Profile such that it is now possible to view any responses to any commands that are issued. This allows the application the ability to see if a command was responded to either successfully or with an error.

- Fixed bug in Headset Profile such that the Headset module no longer responds with “ERROR” responses for unsolicited responses. In fact, the Headset Module never will respond to unsolicited responses from the Audio Gateway. An example of this would be if the unsolicited response “NO CARRIER” was received from the Audio Gateway. Prior to this fix, the Headset module would have responded with “ERROR” because it did not know how to process this. Now, no response is sent, per the specification.

- ❑ Fixed bug in RFCOMM where an open confirmation or disconnect event would be dispatched under certain circumstances when the port was being closed (RFCOMM_Release_Request() function). Prior to this fix, it was possible for an event to be dispatched while the port was closing. Now, no event is dispatched when the port is closed by the local client.

- ❑ Changed code in L2CAP such that it is now possible to issue another L2CAP Connect Request to the same Board Address and PSM in the L2CAP Disconnect Event. Prior to this fix, attempting to reconnect to the same Board Address and PSM did not work correctly.

- ❑ Added more HCI_LMP_CLASS_OF_DEVICE_MINOR_DEVICE_CLASS... constants to reflect Digitizer Tablet and Card Reader that have been added to the Bluetooth Assigned Numbers Document.

- ❑ Fixed bug in L2CAP where etDisconnect_Confirmation event was being called without the Result field being initialized. The code has been changed now so that the Result field indicates success (L2CAP_DISCONNECT_RESPONSE_RESULT_SUCCESS).

- ❑ Added RFCOMM_Send_Data_With_Credits() function to RFCOMM to expose internal functionality that existed to allow RFCOMM Credits (when using Credit based flow control) to be sent in the header of an RFCOMM Data packet.

- ❑ Added SPP_Get_Configuration_Parameters() and SPP_Set_Configuration_Parameters() functions to SPP to allow a programmatic method of changing the negotiated RFCOMM Frame Size that SPP will use when negotiating RFCOMM/SPP connections. This function also allows the default Transmit and Receive Buffer sizes to be set (as opposed to always using the default). The ability to change the RFCOMM Frame size that SPP will use allows optimizations (memory and/or throughput) to be made (when required).

- ❑ Improved Credit handling in SPP such that SPP now attempts to send any credits that need to be sent with Data that is scheduled to be transmitted. This allows much higher throughput to be achieved because Data packets are not scheduled with Credit packets. SPP takes advantage of the newly added RFCOMM_Send_Data_With_Credits() function to achieve this functionality.

- ❑ Added support in the OTP layer to allow OBEX Synchronization. This entailed adding the OTP_Client_Put_Sync_Object_Request() , OTP_Put_Sync_Object_Response(), OTP_Client_Delete_Sync_Object_Request(), and OTP_Delete_Sync_Object_Response() functions. The etOTP_Delete_Sync_Object_Request, etOTP_Delete_Sync_Object_Response, etOTP_Put_Sync_Object_Request, and etOTP_Put_Sync_Object_Response Events to support the OBEX Synchronization functionality.

- ❑ Changed the OBEX Packet size used for the OBEX File Transfer Profile and the OBEX Object Push Profile to improve performance (sizes were chosen to maximize throughput using DH5 baseband packets). The previously chosen values were not optimized for DH5 baseband packets (which equates to the maximum throughput).

- Windows

    - ❑ Updated USB Driver to better handle Windows Power Management.

    - ❑ Updated BTCOM/DUN/FAX/LAP Profiles such that they now correctly report the state of DTR/DSR signals to the remote device. Previous versions of these profiles always passed the state of these signals as asserted. Now the state passed to the remote device depends on the actual signal state.

    - ❑ Updated BTCOM/DUN/FAX/LAP Profile Definitions File (DEF) such that they now correctly alias the Open and SDP Registration function names. This was a problem when using these profiles with the Visual C++ Compiler.

    - ❑ Updated BTCOM/DUN/FAX/LAP Profiles such that the first parameter of the Event Callback is the Bluetooth Stack ID (as documented). The value of this parameter was the actual Profile ID, not the Bluetooth Stack ID. This has been changed to match all the documentation (it now returns the Bluetooth Stack ID as the first parameter in all of the Event Callbacks).

    - ❑ Updated BCSP Driver that was present in the HCI Driver to improve performance.

- Windows CE

    - ❑ Fixed bug in OBEX File Transfer Profile that would not allow an empty directory to be changed into.

    - ❑ Updated BTCOM/DUN/FAX/LAP Profile Definitions File (DEF) such that they now correctly alias the Open and SDP Registration function names. While technically not a problem (as there is only a single Windows CE Compiler generally used), this change was made to make the Windows CE Definitions File (DEF) look closer to the Windows Versions.

    - ❑ Updated BTCOM/DUN/FAX/LAP Profiles such that the first parameter of the Event Callback is the Bluetooth Stack ID (as documented). The value of this parameter was the actual Profile ID, not the Bluetooth Stack ID. This has been changed to match all the documentation (it now returns the Bluetooth Stack ID as the first parameter in all of the Event Callbacks).

    - ❑ Updated BCSP Driver that was present in the HCI Driver to improve performance.

- Linux

    - ❑ Updated BTSER/DUN/FAX/LAP Profiles such that they now correctly report the state of the DTR/DSR signals to the remote device. Previous versions of these profiles always passed the state of these signals as asserted. Now the state passed to the remote device depends on the actual signal state.

    - ❑ Updated BTSER/DUN/FAX/LAP Profiles such that the first parameter of the Event Callback is the Bluetooth Stack ID (as documented). The value of this parameter was the actual Profile ID, not the Bluetooth Stack ID. This has been changed to match all the documentation (it now returns the Bluetooth Stack ID as the first parameter in all of the Event Callbacks).

    - ❑ Updated BCSP Driver that was present in the HCI Driver to improve performance.

- QNX

- ❑ Added USB Driver (requires QNX 6.2.2) and updated all sample applications with the option of using USB as the HCI Transport.
- ❑ Updated BCSP Driver that was present in the HCI Driver to improve performance.

Bluetopia Version 1.2.5 Changes (from Version 1.2.3)

- General
  - ❑ Updated API documentation.
  - ❑ Added API documentation for the following Profiles: Headset, Hands Free, OBEX File Transfer, OBEX Object Push, Dial Up Networking, FAX, LAN Access, and generic Bluetooth Serial Port Profile.
  - ❑ Added more HCI_LMP_COMID_MANUFACTURER_NAME_... constants to reflect most current Manufacturer List that have been added to the Bluetooth Assigned Numbers Document.
  - ❑ Updated the list of SDP Service Attributes ID to reflect some of the newer Attribute ID values that have been added to the Bluetooth Assigned Numbers Document.
  - ❑ Changed the Development Versions of Bluetopia to support the transfer of 2 Megabytes of data. Previous Versions of Bluetopia were limited to 32 Kilobytes.
  - ❑ Added support for CSR chips running BCSP. This is only applicable to COM Ports and is specified in the HCI Comm Information structure when the stack is opened.
  - ❑ Updated all supplied sample applications to support BCSP.
  - ❑ Updated Hands Free Profile to most current version. Previously released versions of this Profile were based on 0.96 Draft Standards. This release releases the version based on the 1.0 Voting Draft Specification.
  - ❑ Added new fields to HCI_COMMDriverInformation_t structure to support Delay when opening COM Ports. This was added because some devices (notably Compact Flash cards) require a delay before the chip is ready to accept data).
  - ❑ Fixed bugs in L2CA_Ping() and L2CA_Get_Info() functions that would not allow a callback to installed that had zero as the value for the callback parameter.
  - ❑ Fixed comments for L2CA_Un_Register_PSM() and L2CA_Group_Data_Write() to correctly identify the correct return values for these functions. Also fixed numerous typographical mistakes in the comments.
  - ❑ Removed superfluous comma that is present in the L2CA_Link_Connect_Request_Config_t enumerated type. The last element of this enumeration had a comma that followed it. This comma caused problems with some compilers (shouldn't be there anyway) so it has been removed.
  - ❑ Added support for Total Number of Credits (that are available for sending) that can be larger than a single byte. Each credit of Credits can be maximum of a

Byte, however the total can indeed be greater than the size of a Byte (for example two Credit packets of 255 each could be sent yielding 510 Total Credits). There is an Errata that has been submitted to limit the total to 255. With this change, even if this Errata isn't made mandatory, no problems should occur with devices that send a total Credit Count greater than 255.

- Fixed bug in RFCOMM_Open_Request() that incorrectly typecast the TEI to a Byte_t (8 bit) which would truncate the Word_t (16 bit) value. This caused problems when more than 255 connections had been attempted (or made).

- Memory Leak fixed in RFCOMM when Page Timeout occurs on connection request.

- RFCOMM now allows Connections to be initiated from RFCOMM Callbacks (without causing Infinite Loops).

- RFCOMM now allows ONLY a single Open Request to be outstanding (prior to this multiple RFCOMM Open Requests could be submitted and were not dispatched correctly). This limitation of only one outstanding RFCOMM Open Request is an RFCOMM Specification limitation, and has not been introduced because of Bluetopia's implementation. RFCOMM should never have allowed more than one simultaneous Open Request to be outstanding in the first place. With this fix, if a user attempts to open two requests simulataneously the second call to RFCOMM_Open_Request() fails. This does not mean that there cannot be more than one simultaneous RFCOMM Channel open, it only means that a single RFCOMM Channel establishment procedure can be active at any given time (once the channel is fully established another Open Request can be issued).

- Fixed RFCOMM Module such that it now correctly frees the memory associated with the lower level L2CAP Channel when disconnecting an RFCOMM Client. Previously, under some circumstances, the Channel Information was cleaned up, but the memory was not freed, resulting in a Memory Leak (under certain disconnection circumstances).

- Fixed bug in RFCOMM Module where L2CAP Server Channel would not be deleted correctly when the connecting Device would not complete the L2CAP Connection/Configuration properly. This occurred most prominently when an L2CAP Connection Request sent and NOTHING else was ever sent. Future attempts from ANY Remote Device to that RFCOMM Server Port would result in an L2CAP Connection Response with a Refused NO Resources.

- Fixed bug in SPP_Open_Remote_Port() that would not release memory when an error occurred submitting the RFCOMM Open Request. Prior to this, the Transmit and Receive buffers were not freed when this occurred.

- Added code in the SPP_Change_Buffer_Size() function and the SPP_Purge_Buffer() such that Credits are not reported to RFCOMM until the connection is fully in an open state. Prior to this, it was possible for these functions to send RFCOMM Credits to the remote side before the initial credits were sent (on connection establishment). This would only occur in a very specific scenario.

- Fixed code in Headset Profile Client functions that set the Bluetooth Class of Device bits wrong (cut and paste error).

- ❑ Updated all routines that are used by GAP and L2CAP when establishing remote connections to use Page Scan Repetition Mode R1 instead of Page Scan Mode R0. This was done because Page Scan Repetition Mode R1 matches the default Page Scan Period on all Bluetooth Devices.

- ❑ Updated GAP so that when a Connection is created for Bonding all supported Packet Types are enabled for the connection request. Prior to this, only DM1/DH1 Packets were enabled for the Bonding Connection Request.

- Windows

  - ❑ Updated all Windows Sample Applications to support COM Ports 1-9, and Baud Rates 9600, 19200, 38400, 57600, 115200, 230400, 460800, and 921600.

  - ❑ BCSP Support Added.

  - ❑ Updated USB Driver to support new chipsets that are not fully compliant with the Bluetooth Specification. The USB Driver has been tested with CSR, Ericsson, Philips, Silicon Wave, Texas Instruments, Transilica, and Zeevo. Devices that support the Generic Bluetooth Wireless Device Audio Class and conform to the Bluetooth USB Specification should work with no problems.

  - ❑ Changed incorrect statement in description of the OBJP_Open_Server() in OBEX Object Push Profile which incorrectly said the InBoxDirectory parameter was a pointer to a NULL terminated Wide Character String. This is incorrect, it is a pointer to a NULL terminated ASCII string. This comment has been updated.

- Windows CE

  - ❑ Updated all Windows CE Sample Applications to support COM Ports 1-9, and Baud Rates 9600, 19200, 38400, 57600, 115200, 230400, 460800, and 921600.

  - ❑ Updated all Windows CE Pocket PC Sample Applications to use a smaller font in the main display windows to allow more information to be displayed to the user.

  - ❑ BCSP Support Added.

  - ❑ Added Driver to allow Bluetopia to open internal Bluetooth Module on Compaq iPAQ Model 3870 and Compaq iPAQ Model 3970.

- Linux

  - ❑ Linux Profiles added: OBEX File Transfer, OBEX Object Push, Dial-up Networking, FAX, LAN Access, and generic Bluetooth Serial Port Profile. Bluetopia for Linux now has the following Profiles: Headset, Hands Free, OBEX File Transfer, OBEX Object Push, Dial-up Networking, FAX, LAN Access, and Generic Bluetooth Serial Port Profile.

  - ❑ Linux sample applications added: HCI, L2CAP, SDP, DUN, FAX, LAP, Headset, Hands Free, OBEX Object Push, OBEX File Transfer. Bluetopia for Linux now has the following sample applications: HCI, SCO, L2CAP, SDP, SPP, Headset, Hands Free, OBEX File Transfer, OBEX Object Push, Dial-up Networking, FAX, LAN Access, and Generic Bluetooth Serial Port Profile.

  - ❑ Linux Virtual Serial Port's added to support Dial-up Networking, FAX, LAN Access, and Generic Bluetooth Serial Port Profiles.

    - ❑ BCSP Support Added.

    - ❑ Added Linux Locking Mechanism so that two different stack applications couldn't open the same Serial Port simultaneously. Linux allows a Serial Port to be opened by multiple applications simultaneously. To avoid two Bluetopia Applications from sharing a Serial Port locking was added.

- QNX

    - ❑ QNX Profiles added: OBEX File Transfer, OBEX Object Push. Bluetopia for QNX now has the following Profiles: Headset, Hands Free, OBEX File Transfer, and OBEX Object Push.

    - ❑ QNX sample applications added: HCI, L2CAP, SDP, Headset, Hands Free, OBEX Object Push, OBEX File Transfer. Bluetopia for QNX now has the following sample applications: HCI, SCO, L2CAP, SDP, SPP, Headset, Hands Free, OBEX File Transfer, and OBEX Object Push.

    - ❑ BCSP Support Added.

    - ❑ Added QNX Locking Mechanism so that two different stack applications couldn't open the same Serial Port simultaneously. QNX allows a Serial Port to be opened by multiple applications simultaneously. To avoid two Bluetopia Applications from sharing a Serial Port locking was added.