
Triangle MicroWorks, Inc.
DNP3 Slave Source Code Library

What's New?

a description of the changes included in all versions of the library

Version 3.09.00
March 15, 2010

Property of Triangle MicroWorks, Inc.



This Source Code and the associated Documentation contain proprietary information of Triangle MicroWorks, Inc. and may not be copied or distributed in any form without the written permission of Triangle MicroWorks, Inc.

Copies of the source code may be made only for backup purposes.









© 1994-2010 Triangle MicroWorks, Inc. All rights reserved.


This document describes features or corrections that have been added to the DNP3 Slave Source Code Library.


The symbols to the left of each revision are used to help define the following kinds of revisions:


- dnp3** Additional or enhanced DNP3 features.
-  Additional or enhanced source-code features, allowing implementers to more efficiently install the Source Code Library.
-  Corrections to problems, with indications for when the problems were introduced.


Version 3.09.00 (March 15, 2010)


-  Improved the Link Reset interface to support an “Implementation Dependent” event that requires a Link Reset.
-  Modified Device Profile generation routines to process one Data Set Element at a time in order to reduce memory requirements.
-  Modified WinIOTarg to prefer an exact address match when multiple channels are open and disconnectOnNewSyn==true.
-  Added localIpAddress to WinIoTarg to specify the source address to use when Client (or UDP sender) is sending IP messages. This, in conjunction with the Destination IP address and Routing Tables allows control over which Ethernet card is used to establish a connection. (At Windows command prompt, enter "route print" "route add xxx" can be used to add a route to cause a particular NIC to be used.)
-  Enhanced WinIOTarg to support a list of addresses from which TCP/IP connections will be accepted.
-  Created function to save the DNP3 Outstation Device Profile (current values section).
-  Added user data pointers to TMWCHNL, TMWSCCTR, and TMWSESN.
-  *Description:* When multiple application contexts are used, closing one deleted the memory pool lock, which could cause problems if another application context was open
Introduced: v3.00
Resolution: Fixed in v3.09


- 


Description: WinIOTarg could fail to process all available characters, causing timeouts when running at high baud rates.
Introduced: v3.00
Resolution: Fixed in v3.09
- 

Description: When configured to use system time, WinIOTarg System time not setting the dstlnEffect flag correctly.
Introduced: v3.00
Resolution: Fixed in v3.09
- 

Description: The Source Code Library could stop responding if a Link Status request was received while waiting for a Link Layer response.
Introduced: v3.00
Resolution: Fixed in v3.09
- 


Description: LinIOTarg could cause high levels of processor utilization when multiple channels were opened.
Introduced: v3.07
Resolution: Fixed in v3.09
- 


Description: When generating a device profile using the current values, while determining if a point was reported in a Class 0 response, a wrong index could be used, which could return incorrect results or cause a bad reference.
Introduced: v3.07
Resolution: Fixed in v3.09
- 


Description: The Source Code Library would incorrectly call the database functions if the Master resent the same block during asynchronous file transfer.
Introduced: v3.00.38
Resolution: Fixed in v3.09
- 

Description: User Managed Events did not support Analog Output Events (Object Group 42) and Analog Command Events (Object Group 43).
Introduced: v3.00
Resolution: Fixed in v3.09









Version 3.08.00 (September 25, 2009)




- 

Made improvements to Current Values XML Device Profile generation.
- 










Modified online status detection for serial lines to require reception of data.
- 


Description: When the Source Code Library had Secure Authentication enabled against a Master with Secure Authentication disabled, looking up user number 0 used in a challenge resulted in a Null pointer.
Introduced: v3.07.00
Resolution: Fixed in v3.08

-  *Description:* In the simulated database, a Data Set descriptor name is of data type VSTR instead of None.
Introduced: v3.00
Resolution: Fixed in v3.08
-  *Description:* If a Select/Operate response will not fit in the Transmit fragment, it can overwrite end of buffer. (Note that this can only happen if the request is within one byte of filling the fragment, so that there is not room for the IIN bits in the response.)
Introduced: v3.00
Resolution: Fixed in v3.08
-  *Description:* User managed events can use the wrong default variation when reporting events.
Introduced: v3.00.40
Resolution: Fixed in v3.08
-  *Description:* After receiving a Disable Unsolicited Response Request, the Source Code Library incorrectly canceled the Confirm Timer.
Introduced: v3.00
Resolution: Fixed in v3.08
-  *Description:* In dnp\dnplink.c, _findSession did not return a handle to a newly created session when the pAutoOpenCallback was configured and called.
Introduced: v3.01.01
Resolution: Fixed in v3.08
-  *Description:* Sessions were not marked offline if the initial Null unsolicited response timed out.
Introduced: v3.00
Resolution: Fixed in v3.08
-  *Description:* The response to a Direct Operate request that exactly filled a fragment would contain extra characters.
Introduced: v3.00
Resolution: Fixed in v3.08
-  *Description:* Adding a new event while waiting for the confirmation of an unsolicited response could incorrectly restart the delay timer.
Introduced: v3.00
Resolution: Fixed in v3.08

-  *Description:* When writing a data set, a negative integer with a length of either 1 or 2 it incorrectly becomes a positive number.
Introduced: v3.00.43
Resolution: Fixed in v3.08
-  *Description:* Object 2 and Object 4 Variation 3 generated eventConfirm instead of eventSent.
Introduced: v3.00
Resolution: Fixed in v3.08
-  *Description:* The rs232Reader thread in WinIOTarg was missing a stop event.
Introduced: v3.00
Resolution: Fixed in v3.08


Version 3.07.00 (July 17, 2009)

-  Added LinIOTarg (low-level target for Linux) and new command line-based examples (that can be used with Linux).
-  Integrated WinIOTarg (low-level target for Windows) into the standard release package.
-  Updated Device Profile to an XML Device Profile document. Also included a link to the Triangle MicroWorks [ICD Editor](#), which can be used to edit the XML Device Profile document.
-  Improved counting of transmitted and received challenge messages for DNP3 Secure Authentication.
-  Added FormatMessage Win32 API function to WinIoTarg to provide more descriptive errors instead of just numeric values.
-  Added ability to write v2.06 DNP3 XML Schema.
-  Added additional confirmation of Select cancellation: when a Select with a different sequence number is received after the initial Select; or when the Operate following a Select does not match the Select.
-  Added file read retries to the DNP3 Slave Source Code Library.
-  Modified Link State behavior and Link Layer error responses to improve compliance.

 *Description:* Calling dnplink_linkReset() retry did not handle retries properly. (Note: normally, applications should not call this function, but should let the Source Code Library handle Link Resets.


Introduced: v3.00

Resolution: Fixed in v3.07

 *Description:* The Source Code Library did not properly handle Control Code = 0 for Object 12 Variation 2.


Introduced: v3.00

Resolution: Fixed in v3.07

 *Description:* When a packet was received with a duplicate Transport function sequence number, but with the FIN bit not set, the packet should be discarded. The SDNP SCL discarded the entire transport layer segment-series when it receives a duplicate sequence number.

Introduced: v3.00

Resolution: Fixed in v3.07

 *Description:* When using SDNPDATA_KEEP_LAST_RESPONSE, . The outstation SCL could respond to a request with a different SEQ number.

Introduced: v3.00


Resolution: Fixed in v3.07

 *Description:* When using WinIOTarg for the low-level target interface, the UDP address was not validated properly.

Introduced: v3.00

Resolution: Fixed in v3.07.00


Version 3.06.00 (April 14, 2009)

 Improved documentation of wrap/unwrap functions for Secure Authentication. Provided tolerance for padding being removed.

 *Description:* Secure Authentication session keys larger than 16 bytes did not work.


Introduced: v3.00

Resolution: Fixed in v3.06


 *Description:* The Source Code Library incorrectly returned both Object Error and Parameter Error in response to a read of unsupported event objects.

Introduced: v3.00


Resolution: Fixed in v3.06

- 


Description: The Source Code Library did not properly differentiate Analog Output Assign Class commands for outputs.
Introduced: v3.00
Resolution: Fixed in v3.06

- 

Description: sdnpsim_binOutOpPatternMask ignored the control field, always setting the specified points to “on.”
Introduced: v3.00
Resolution: Fixed in v3.06


- 

Description: Pattern masks did not work properly for quantities larger than 254.
Introduced: v3.00
Resolution: Fixed in v3.06


- 

Description: With asynchronous file reads, if the read with data responded immediately, the sdnpdata_confirmFileRead function was not called properly.
Introduced: v3.00
Resolution: Fixed in v3.06


Version 3.05.01 (March 5, 2009)

- 


Added support for DNP3 Secure Authentication updates and updated documentation due to updates from DNP3 Technical Committee. Updates included allowing aggressive mode response when a pre-challenge is sent with an Application Confirm and no previous challenge response has been received.

- 


Added ability to exclude points from Class 0, but still read them with specific object group reads.

- 


Added TMWCNFG_MAX_APPLRCVS to “break out” of loop in tmwapp!_checkForInput() after the specified number of iterations.

- 


Added ability to configure SOE vs. Last Reported on a per-point basis.

- 


Added support for returning current value in event responses.

- 


Added new #define statements (SDNPDATA_SUPPORT_EVENT_SCAN , SDNPDATA_SUPPORT_EVENT_VAR, and SDNPDATA_SUPPORT_EVENT_MODE) to remove some optional code (event scanning, per-point default class, and per-point event mode) to save code space if these options are not used.


- 


Added retries for DNP Link Layer *Test Link* and *Request Status of Link* messages.

- 


Modified timer values in tmwlink_channelCallback to speed up retries.

 *Description:* When SDNPCNFG_SUPPORT_XML_STATIC is TMWDEFS_TRUE, Device Attribute and Data Set responses could fail.
Introduced: v3.00
Resolution: Fixed in v3.05.01

 *Description:* If a Select with the same sequence number but different data was received, the new Select was not discarded.
Introduced: v3.00
Resolution: Fixed in v3.05.01


 *Description:* Receiving a frame with more than 250 bytes of data (the maximum frame size defined in the specification) could cause the Source Code Library to write past the end of a buffer.
Introduced: v3.00
Resolution: Fixed in v3.05.01


Version 3.05.00 (December 4, 2008)

 Additional enhancement to support for DNP Authentication, including:


- Support for session key lengths other than 16 bytes
- Support for Secure Authentication Aggressive Mode and Prechallenge in the same message


 Improved comments on compiler options such as TMWCONFIG_USE_MANAGED_SCL.

 *Description:* When the Master challenged an unsolicited response, it used the wrong Application Sequence number.
Introduced: v3.04
Resolution: Fixed in v3.05


 *Description:* A Secure Authentication aggressive mode object could extend past the end of the transmit buffer.
Introduced: v3.04
Resolution: Fixed in v3.05

Version 3.04.01 (September 30, 2008)


 *Description:* The aggressive mode object (g120v9) incorrectly included New CSQ in the HMAC calculation, as it did in Secure Auth Version 1 aggressive mode object (g120v3).
Introduced: v3.04.00
Resolution: Fixed in v3.04.01


 *Description:* The authentication statistics incorrectly labeled “sent’ as “received.”
Introduced: v3.00
Resolution: Fixed in v3.04


Version 3.04 (September 12, 2008)


 Added support for v2.0 DNP3 Secure Authentication


 Added enhancements to DNP3 File Transfer.


 Added ability Activate Config to specify more than one file object using a single object header.

 Added checking for ST, RD bits for Data Set (Objec87) Reads, and for WR bits for Data Sets (Object 87) writes. Added checking for EV bit for Data Set Events (Object 88).

 Added protection against processing a Write Last Time request if the time was not read first.


 *Description:* The afterTxCallback could process a request a second time.
Introduced: v3.0
Resolution: Fixed in v3.04


 *Description:* tmwdb_lockQueue and tmwdb_unlockQueue were described in the header comments for tmwdb_storeEntry in tmwdb.h, but they were not defined in tmwdb.c.
Introduced: v3.00
Resolution: Fixed in v3.04


 *Description:* IIN1.4 (Need Time) would be set regarding of the setting of respondNeedTime.
Introduced: v3.00
Resolution: Fixed in v3.04

Version 3.03.00 (April 1, 2008)








 Added ability to include session indicator in channel statistics.


 Modified handling of event queues to that double-bit and single-bit binary data appears as a single chronological sequence, based on discussions by the DNP Technical Committee.


 *Description:* File open retry did not use the permissions from the open request.
Introduced: v3.00
Resolution: Fixed in v3.03

-  *Description:* The Source Code Library did not start the scan timer if scanPeriod was changed from 0 to a non-zero value.
Introduced: v3.00
Resolution: Fixed in v3.03.





Version 3.02.00 (December 7, 2007)

-  Added ability to send events using a qualifier specifying a one-byte index.
-  Added support for writing to IIN 1.4-Need Time bit (in order to clear it).
-  Implemented security changes recommended at the 2007 DNP Technical Committee Face-to-Face meeting. (Note that additional changes are likely to be recommended by the Technical Committee once further issues have been resolved.)
-  *Description:* The 49-day timer roll-over could potentially leave a timer at the end of the queue. This condition could only occur around the time the 32 bit millisecond value returned from tmwtarg_getMSTime() wrapped back to zero. It was possible for a timer that should expire shortly before the wrap to zero to get left after a timer that would not expire until after the wrap to zero. This meant that this timer would not expire for more than 49 days. If the timer was cancelled and restarted, it would function properly. Most timers are used for checking for a confirm or a response, etc. and will be cancelled when the response is received and will not be affected by this issue. However, some timers such as those used for event scanning would not be cancelled and would not expire in a timely fashion.
Note: any application using timers for event scanning or other uses in which the timer is not canceled after a short period of time should upgrade to this version (or later).
Introduced: v3.00
Resolution: Fixed in v3.02.00
-  *Description:* Users could not set the OVER_RANGE flag for analog inputs.
Introduced: v3.00
Resolution: v3.02.00
-  *Description:* dnpdefs.h did not include all valid status code responses.
Introduced: v3.00
Resolution: Fixed in v3.02.00
-  *Description:* The Source Code Library did not correctly implement RFC3394 AES Encryption Algorithm
Introduced: v3.0.01.00
Resolution: Fixed in v3.02.00






 *Description:* DNP Aggressive Mode was not generated or verified correctly.
Introduced: v3.01.00
Resolution: Fixed in v3.02.00


 *Description:* The simulated database no longer initialized to the default configuration.
Introduced: v3.01.00
Resolution: Fixed in v3.02.00


Version 3.01.01 (August 8, 2007)


-  Added ability to keep time on a per-session basis. Added pSession to TMWDTIME structure; target layer can use this parameter to return a time specific to the session.
-  Added ability to specify an event's default event variation at the time the event is added.
-  Added ability to specify (on a per-Counter point basis) whether frozen, running, or frozen and running values are reported.
-  Added support for keeping the last sent and last received messages as described in the DNP Application Layer specification. If a duplicate request is received, it will not be processed, but the same response will be sent. This feature can be disabled (to avoid the extra memory and overhead required to support it) by setting SDNPDATA_KEEP_LAST_RESPONSE to TMWDEFS_FALSE.


Version 3.01.00 (May 23, 2007)

-  Added support for DNP3 Security.
-  Added Activate Configuration support to Function Code 31.
-  Modified online notification algorithm to facilitate notification of sessions that never go online.
-  *Description:* When reading Object 0 Variation 254 8bitstartstop, the Source Code Library looked at an uninitialized part of the response message.
Introduced: v3.00
Resolution: Fixed in v3.01.00
-  *Description:* A read response using 16 bit start/stop indices could run pas the end of a buffer if the last point in the buffer required a change of variation because its flags were not nominal.
Introduced: v3.00.29
Resolution: Fixed in v3.01.00

 *Description:* The SDNP File Read Directory response did not limit the response data to the maximum block size specified in the Open request.
Introduced: v3.00
Resolution: v3.01.00

 *Description:* The Source Code Library would always send the configured Master address in a Reset Link response, rather than using the source address from the Reset Link message.
Introduced: v3.00.29
Resolution: Fixed in v3.01.00


 *Description:* The Source Code Library could use excessive stack space due to recursion when configured to not use link confirms.
Introduced: V3.00
Resolution: Fixed in v3.01.00


 *Description:* The Source Code Library did not build with TMWCNFG_SUPPORT_FLOAT True and TMWCNFG_SUPPORT_DOUBLE False
Introduced: v3.00
Resolution: Fixed in v3.01.00


Version 3.00.49 (November 1, 2006)


dnp3 Updated Data Sets support (defined in TB2004-004) to include support for Controls, as described in TB2004-004 Rev e


 Added Data Set Prototype names


 Modified Object 42 (Analog Output Status Events) default variation to be consistent with Object 40 (Analog Output Status).

 Renamed TMWCHNL_STAT_CALLBACK and TMWSESN_STAT_CALLBACK macros to TMWCHNL_STAT_CALLBACK_FUNC and TMWSESN_STAT_CALLBACK_FUNC. This change avoids having a macro and a typedef with the same name.


 *Description:* The Source Code Library could get stuck in an infinite loop when switching to a variation with flags due to an overflow condition.
Introduced: v3.00.29
Resolution: Fixed in v3.00.49


 *Description:* The Source Code Library would not compile if SDNPDATA_SUPPORT_OBJ10_WRITE was turned off but SDNPDATA_SUPPORT_OBJ10_WRITE was left on.
Introduced: v3.00.45
Resolution: Fixed in v3.00.49

 *Description:* IIN Buffer Overflow could fail to be reported when events were discarded.
Introduced: v3.00
Resolution: Fixed in v3.00.49


 *Description:* Adding a very large Data Set could result in writing past the end of the buffer.
Introduced: v3.00.43
Resolution: Fixed in v3.00.49


Version 3.00.46 (April 27, 2006)


 Modified release to include WinIoTarg.dll built with Visual Studio 6. This DLL is used when using Visual Studio to build the sample applications.


 Updated sample Makefiles to correctly build Source Code Library.

Version 3.00.45 (March 17, 2006)

 Improved documentation in protocol-specific section of the manual. Provided additional text, reorganized layers, and included all header files that comprise the API.

 Added support for Object 10, Variation 1 (Binary Outputs). (This variation is beyond Subset Level 3).


 *Description:* During file transfer, the simulated database did not flag the end of file properly if the file size was a multiple of the block size.
Introduced: v3.00.38
Resolution: Fixed in v3.00.45








 *Description:* The Source Code Library would not generate unsolicited responses for events that belonged in more than one class.
Introduced: v3.00
Resolution: Fixed in v3.00.45

Version 3.00.44 (February 2, 2006)






dnp3 Updated Data Sets support (defined in TB2004-004) to include support for Rev. D. TYPE now allows optional name following UUID.


dnp3 Added support for Output Event objects, and updated Configuration/Interoperability (C-I) guide to show supported functions and variations.

 Added a sample application to provide an example of a multi-threaded, event-driven application.







-  Set receive indication in diagnostic information to indicate the direction of the message.
-  Added confirm timeout and request status timeout to link layer statistics.
-  Enhanced statistics to include additional values such as CRC errors, number of invalid start bytes, frame length errors, etc.
-  Split into tmwtarg.h/c into tmwtarg.h/c and tmwtargp.c/c. This change moves all Triangle MicroWorks, Inc. specific code into a separate file. Customers only need to modify tmwtarg.c.
-  Set Object 50, Variation 1 as the default variation returned by a read of Object 50, Variation 0. This change provides compatibility with devices that incorrectly issue Object 50, Variation 0 read requests (this request is beyond Subset Level 3).
-  Modified Slave behavior to send Application Confirm when requested by the Master. This behavior is specifically not required by TB2004-001, but was provided to support interoperability with non-compliant Master devices.
-  *Description:* Some sdnpstring assign class functions called functions for incorrect data types.
Introduced: v3.00.40
Resolution: Fixed in v3.00.44

Version 3.00.43 (December 2, 2005)



- dnp3** Added support for Data Sets (defined in TB2004-004).
- dnp3** Added support for Object 0 (Device Description), as defined in TB2003-001.
-  Added tmwtarg_getSFloat and tmwtarg_storeSFloat functions to retrieve/store 32-bit single precision floating point values, compensating for byte order (and floating point format if the native format is not IEEE-754).
-  Added support for Qualifier 0x27 (2 octet index, 8-bit quantity) in received requests.
-  Improved efficiency of adding events to event queue.
-  Added sdnpdata_setTime function. This function allows the target code to determine whether it should set the time based on time syncs sent to this session.
-  Renamed variable *data* to *dataBuf* in mdnppmem.c, dnpmem.c, and tmwmem.c because *data* is a reserved word in some compilers.

-  *Description:* The Source Code Library would mark a session offline when it canceled a request because another request was received. The session would immediately go online when the new request was processed.
Introduced: v3.00
Resolution: Fixed in v3.00.43.












Version 3.00.42 (September 9, 2005)

-  Improved documentation by updating *DNP3 Slave.doc* to improve readability and include all user-modifiable files.
-  Added subversion to SCL filename (e.g., sdnpv30042.exe). This allows the version number to be determined from the file name. The version number continues to be defined in *tmwvrsn.h*.
-  The Source Code Library will now set the confirm request bit if the target application sets the event buffer overflow IIN bit, even if no events are available.
-  *Description:* Unsolicited responses were not sent promptly when a link was restored.
Introduced: v3.00
Resolution: Fixed in v3.00.42
-  *Description:* The Source Code Library could potentially reference a Null pointer in *tmwdlist_removeEntry*
Introduced: v3.00
Resolution: Fixed in v3.00.42
-  *Description:* An improper loop index type was used in *sdnpo0034.c* (three places), *sdnpo110.c* (1 place) and *sdnpo112.c* (1 place).
Introduced: v3.00
Resolution: Fixed in v3.00.42

Version 3.00.41 (July 20, 2005)

-  Improved documentation (in *sdnpdata.h*) of counter objects to facilitate compliance with TB2002-001.
-  *Description:* Fixed frames were sent using TCP, even when configured for UDP only.
Introduced: v3.00.39
Resolution: Fixed in v3.00.41

Version 3.00.40 (June 3, 2005)

-  Added interface functions to sdnpdata.h/c to allow target application to manage the event queues (e.g., to store event queues in nonvolatile RAM). The target application needs to configure:
`#define SDNPCNFG_USER_MANAGED_EVENTS TMWDEFS_TRUE`
and implement the sdnpdata_umEventxxx() functions.
-  Added processing in case the Restart IIN bit is set in sdnpdata_getIIN(). The bit will now be latched until a write object 80 v1 (Clear Restart) request is received.
-  Added sdnpdata_eventAndStaticRead() , which is called at beginning and end of a read that includes both events and static data. This function can be used to implement a database lock to ensure that the static data matches the last event data sent.
-  Added support for Object 80 Variation 1 Read (Read IIN).
-  Added DNPSlave sample application executables and required DLLs to allow running this application without having to build it.
-  Improved documentation of memory type configuration.
-  Fixed compiler warning for TMWTYPES_BOOL type.
-  *Description:* Reading events could event result in writing 1 byte past end of a buffer. This problem occurred only if the first event of an objectGroup would overflow the buffer. The second and subsequent events worked correctly.
Introduced: v3.00
Resolution: Fixed in v3.00.40
-  *Description:* The source code library did not support Assign Class for String objects.
Introduced: v3.00
Resolution: Fixed in v3.00.40
-  *Description:* The simulated database returned the incorrect file size for a file open on a directory.
Introduced: v3.00
Resolution: Fixed in v3.00.40
-  *Description:* TMWMEM_FREE_FUNC was doubly defined.
Introduced: v3.00.39
Resolution: Fixed in v3.00.40

Version 3.00.39 (March 29, 2005)



Added `tmwtarg_transmitUDP()` function and `networkType` field to `dnplink_config` structure to facilitate implementation of the DNP IP Networking Specification and Dual End Point Configurations described in this specification.



Improved memory management within Source Code Library. Added new `#define` (`TMWCNFG_ALLOC_ONLY_AT_STARTUP`) to `tmwcnfg.h`. Also added new functions that the target to set the maximum number of each buffer in each pool at runtime. Both “only at startup” and static allocations use a linked list of free and allocated buffers instead of just an array that was accessed sequentially.



Added `dnpdata_unsolEventMask()` function to notify the target when a DNP3 Master enables or disables unsolicited messages.



Added two new statistics events to indicate when events are sent and acknowledged. Also added `#define` statements to `dnpcnfg.h` to allow individual DNP3 statistics to be compiled in or out.



Added new structure `TMWTYPES_SCALED_FLOAT` in `tmwtypes.h` to allow `sdnpdata_anlInRead()` to return BOTH `sfloat` and long types, in order to support reads of long and `sfloat` variations.



Description: The Source Code Library did not handle read responses that span fragments correctly.

Introduced: v3.00

Resolution: Fixed in v3.00.39



Description: The Source Code Library did not delay between unsolicited responses that timed out if link confirm mode is set to always.

Introduced: v3.00

Resolution: Fixed in v3.00.39



Description: The Source Code Library responded incorrectly to Freeze commands if frozen objects were not supported.

Introduced: v3.00


Resolution: Fixed in v3.00.39



Description: The Source Code Library sent an unsolicited null response whenever TCP/IP connections reconnected.

Introduced: v3.00

Resolution: Fixed in v3.00.39

-  *Description:* When reading double bits, if a the version switched from v1 to v2 (because the flags were not nominal) occurred with 4 (or 12 or 20 etc) points in v1 portion, an extra byte would be sent in the response.
Introduced: v3.00.38
Resolution: Fixed in v3.00.39

Version 3.00.38 (January 18, 2005)

dnp3 Added support for new double bit and self address discovery mechanism.

dnp3 Added support for double bit input object groups 3 and 4.



Modified the API for the channel, session, sector modify functions to initialize a structure with the current settings. Any of these settings may then be modified by the target code.



Improved API to simplify modification of the frame and fragment sizes.



Improved handling of events to streamline memory usage when an event buffer overflows. The buffer for the removed event is now reused for the new event.



Added support for the address discovery mechanism described in the DNP SelfAddress document.



Description: The Source Code Library could return the wrong sequence number if it received a request with an unsupported Function Code.
Introduced: v3.00
Resolution: Fixed in v3.00.38



Description: A read request could fail if both an initial unsolicited response and a multifragment read were waiting for application confirmation.
Introduced: v3.00
Resolution: Fixed in v3.00.38



Description: The Source Code Library did not properly switch between synchronized and unsynchronized Common Time of Occurrence (CTO) variations when events switched between valid and invalid time stamps.
Introduced: v3.00
Resolution: Fixed in v3.00.38

Version 3.00.37 (December 15, 2004)



Simplified simulated database (sdnpsim.c/.h). The simulated database now always uses a linked list for all data types.



Description: The Source Code Library would return the wrong status code for an Operate that exactly matched the Select after receiving a failed Operate for that

Select. This fix is required to pass the DNP3 Slave Conformance Test sections 8.2.1.2.15 and 8.4.1.2.12.

Introduced v3.00.28

Resolution: Fixed in v3.00.37

Version 3.00.36 (November 2, 2004)



Added ability to perform time synchronization as described in Section 3.3.1 of Layer Independent Topics Draft J. The Source Code Library will adjust the time as shown in Figure 3-1 It will add the time between time F and time G (time first bit received and time when clock is set) before calling `tmwtarg_setDateTime()`.



Added support for file transfer event mode. This feature also brings the Slave into compliance with Section 4.17.1.1 of the Application Layer specification. This feature added a new function (`sdnpdata_FileEventClass()`), as well as an additional parameter (*handle*) to `sdnpdata_readFileInfo()`.

Version 3.00.35 (September 22, 2004)



Added `#define TMWCNFG_MEMORY_ALIGN_NEEDED` to `tmwcnfg.h` to support processors requiring long word (4 byte) alignment and compilers that create unpacked structures.



The Source Code Library now sorts the Binary Event queues by time stamp.



Added configuration option to determine which event to discard (oldest or most recent) when the event queues overflow.



Reviewed and enhanced the Device Profile document to include all configuration parameters require for conformance tests.



Added processing in case the buffer overflow IIN bit is set in `sdnpdata_getIIN()`. The bit will now be latched until an application confirm is received and there is room in all of the event queues.



Added `tmwapp_startTimer` and `tmwapp_cancelTimer` to `tmwtarg.c`.
NOTE: These functions must be defined in the target in order for the Source Code Library to link.



Description: The Source Code Library did not set the IIN2.2 (Parameter Error) bit if an Assign Class request failed for some (but not all) specified points.

Introduced v3.00


Resolution: Fixed in v3.00.35





Description: The Source Code Library incorrectly set the IIN2.2 (Parameter Error) bit for `DNPDEFS_CROB_ST_FORMAT_ERROR`.


Introduced: v3.00


Resolution: Fixed in v3.00.35


 *Description:* The Source Code Library assumed that any write to Object 80, Variation 1 was a Clear Restart.
Introduced: v3.00
Resolution: Fixed in v3.00.35

 *Description:* The link status request configuration could cause a failure to send a response.
Introduced: v3.00
Resolution: Fixed in v3.00.35


 *Description:* `_checkAddressMatchCallback` passed the wrong parameters to `_findSession`.
Introduced: v3.00.22
Resolution: Fixed in v3.00.35


 *Description:* The parameter list was inconsistent in `TMWLINK_OPEN_SESSION_FUNC` and `_openSession` in `dnplink.c`.
Introduced: v3.00
Resolution: Fixed in v3.00.35


 *Description:* The Source Code Library could stop calling `tmwtarg_transmit()` if `tmwtarg_transmit()` returned `TMWDEFS_FALSE`.
Introduced: v3.00
Resolution: Fixed in v3.00.35


 *Description:* Error indication IIN bits set in response to broadcast or `directNoAck` commands were still set in the next response.
Introduced: v3.00
Resolution: Fixed in v3.00.35


Version 3.00.34 (July 30, 2004)

 Added ability to configure default variation per point, instead of just per object group.

 *Description:* The Source Code Library could call `tmwtarg_startTimer` with a negative value. For some operating systems, this causes timers to stop running.
Introduced: v3.00.32
Resolution: Fixed in v3.00.34

 *Description:* The algorithm for sorting timers did not work properly if the two times being compared were more than 17 days apart.
Introduced: v3.00
Resolution: Fixed in v3.00.34


 *Description:* CRC calculations and checks did not work properly with 32-bit data types.
Introduced: v3.00
Resolution: Fixed in v3.00.34

 *Description:* An event that did not belong in any valid event class would remain in the event queue.
Introduced: v3.00
Resolution: Fixed in v3.00.34

Version 3.00.33 (June 17, 2004)


 Added performance enhancements to event processing.


Version 3.00.32 (June 4, 2004)


 Enhanced support for multiple threads. Added support for one timer queue per channel. Also improved locking/unlocking to prevent threads from being interrupted or deadlocked. This enhancement required two changes to the API:


- Added new `#define(TMWCNFG_MULTIPLE_TIMER_QS)` to allow support of multiple timer queues (default is `TMWDEFS_FALSE`)
- Added new parameter to `tmwdb_storeEntry()` to indicate whether the queue should be locked (default is not to lock the queue).


Any implementation using multiple threads should upgrade to this release


 Improved generation of link status messages. The Link Status request will now be sent only if the session is idle for the configured period of time.


 Modified setting of OVER-RANGE bit for analog values. The bit will now be set if the value exceeds the value that can be reported by the requested variation.


 *Description:* The `_timerCallback` function in `tmwtimer.c` did not always lock the proper channel.
Introduced: v3.00
Resolution: Fixed in v3.00.32


 *Description:* While responding to a read request for Object 2 Variation 3, the Source Code Library could write past the end of a buffer.
Introduced: v3.00
Resolution: Fixed in v3.00.32

 *Description:* For some unsupported qualifiers, the Source Code Library would return a Null Response without the Parameter Error IIN bit set.
Introduced: v3.00
Resolution: Fixed in v3.00.32


 *Description:* The Source Code Library would not compile if OBJ50V1 was defined as TMWDEFS_FALSE *and* OBJ50V3 was defined as TMWDEFS_TRUE.
Introduced: v3.00
Resolution: Fixed in v3.00.32


 *Description:* tmwappl_initApplication() could not be called more than once.
Introduced: v3.00.29
Resolution: Fixed in v3.00.32


 *Description:* The Source Code Library did not properly report all events if they did not fit into a single fragment.
Introduced: v3.00
Resolution: Fixed in v3.00.32


 *Description:* A 31-day timer would expire immediately.
Introduced: v3.00
Resolution: Fixed in v3.00.32


Version 3.00.29 (March 31, 2004)


 Updated Configuration/Interoperability Guide to more clearly show Function Code and qualifier support for each object and variation.


 Modified default variations. The protocol requires changing variations when flags are not nominal, allowing the default variation to be without flags.

 *Description:* The Source Code Library did not calculate times correctly
Introduced: v3.00.27
Resolution: Fixed in v3.00.29

 *Description:* When configured for Static memory support, the Source Code Library could fail to allocate buffers if they contained non-zero data on startup.
Introduced: v3.00.23
Resolution: Fixed in v3.00.29


 *Description:* The Source Code Library incorrectly indicated an error condition (Operate did not follow select).
Introduced: v3.00.28
Resolution: Fixed in v3.00.29

 *Description:* The Source Code Library would sometimes respond incorrectly (rather than reporting an error) when it received requests for unsupported qualifiers.
Introduced: v3.00
Resolution: Fixed in v3.00.29

 *Description:* The Source Code Library did not properly check for TMWDNFG_SIMULATED_DB when configured for static memory, which could result in accesses to unallocated memory when testing with the simulated database.

Introduced: v3.00.20

Resolution: Fixed in v3.00.29

 *Description:* Virtual Terminal event scanning incorrectly used *stringScanPeriod* instead of *virtualTerminalScanPeriod*.

Introduced: v3.00

Resolution: v3.00.29

Version 3.00.28 (March 2, 2004)

dnp3 Added support for duplicate Select and Operate requests. This change was required due to a revision in the DNP3 Slave Conformance Test Procedure document to bring it in line with TB2000-002.



Added support for callback function to be called when a fragment is given to the transport layer for transmission.



Removed Incremental Timeout parameter, which does not apply for DNP3 sessions, and added Application Confirmation timeout. Previously, the Source Code Library used the Incremental Timeout as an Application Confirmation timeout timer.



Added enhancements to support simultaneous Master and Slave sessions on the same channel.



Continued improvements to documentation. Provided more descriptive comments in session header files.



Description: Memory could be leaked when processing Diagnostic code on inactive sessions.

Introduced: v3.00

Resolution: Fixed in v3.00.28



Description: A Slave device would respond to a Reset Link command using the configured destination address, rather than the address of the Master that issued the command.

Introduced: v3.00

Resolution: Fixed in v3.00.28




Description: The Source Code Library could reference a Null pointer if it was using the Simulated Database and was configured not to use dynamic memory.



Introduced: v3.00.14

Resolution: Fixed in v3.00.28




Version 3.00.27 (February 3, 2004)

-  *Description:* The link layer did not always set the DIR bit correctly.
Introduced: v3.00
Resolution: Fixed in v3.00.27


Version 3.00.26 (January 15, 2004)

-  Modified the default application layer confirm timeout to 10 seconds to prevent timeouts on slow baud rates.
-  *Description:* The unsolicited retry counter would roll over from 65535 to 0 (which restarted the online retry rate).
Introduced: v3.00
Resolution: v3.00.26









Version 3.00.25 (December 12, 2003)

- dnp3** Added support for TB2003-002. When reading a point, if the flags are not nominal, the variation is switched to one that includes flags.
-  Improved efficiency of static data read requests. The Source Code Library can now skip over objects during the SDNPSESN_QUAL_PARSE_ONLY pass.
-  *Description:* The Source Code Library would report an “Invalid message length error” if the SDNPDATA_XXXQUANTITY function returned TMWDEFS_NULL.
Introduced: v3.00
Resolution: Fixed in v3.00.25
-  *Description:* Some CROB-related functions returned TMWDEFS_FALSE instead of type DNPDEFS_CROB_ST. This could cause the Source Code Library to incorrectly interpret the command as succeeding.
Introduced: v3.00
Resolution: Fixed in v3.00.25



Version 3.00.24 (November 10, 2003)

-  *Description:* The Source Code Library would send object 34 before object 30, even if Object 30 was requested first.
Introduced: v3.00
Resolution: Fixed in v3.00.24

Version 3.00.23 (October 31, 2003)

-  Added locks to tmwmem.c to add support for multiple threads to memory management routines.
-  Provided optimizations when diagnostics are not compiled in.
-  Added support for automatic installation of sessions. Added support for a user callback function that will be called when a message is received on an open channel for an address that is not configured. After calling this callback function, the Source Code Library will again check to see if a session is open for this address. If so, the message will be processed.
-  Improved operation of unsolicited delay timer. When events are read, the Source Code Library checks to see if any the event queue for any class is empty. If so, the unsolicited delay timer for that class is canceled.
-  *Description:* The Source Code Library could reference a Null pointer when adding events if the device was out of memory and there were no events on the current event queue.
Introduced: v3.00
Resolution: v3.00.23
-  *Description:* Link Status messages could be sent to all sessions on a channel without waiting for the response.
Introduced: v3.00
Resolution: v3.00.23
-  *Description:* The idle callback function could be called even though the Slave was still waiting for an application layer confirmation for an unsolicited message.
Introduced: v3.00.20
Resolution: Fixed in v3.00.23
-  *Description:* The diagnostic routines would sometimes not display DNP Analog floating point values correctly.
Introduced: v2.00
Resolution: Fixed in v2.00.23

Version 3.00.22 (October 8, 2003)

-  Cleaned up compiler warnings and issues identified by *lint*.
-  Changed the interface to sdnp_xxxChanged() functions to remove the time stamp, since the Source Code Library sets the time stamp. If the application needs to

specify the time stamp, it should call `sdnpxxx_addEvent()` instead of using scanning.



Added `pCheckAddrCallbackFunc` function. This function facilitates the use of modem pools with the Source Code Library.



Description: Attempting to send or receive large (more than 250 character) DNP Virtual Terminal messages could cause a buffer overflow condition in the DNP diagnostics routine.

Introduced: v3.00

Resolution: Fixed in v3.00.22



Description: Bytes updated just before transmitting were not checked for proper ENDIAN before storing in the transmit buffer.

Introduced: v3.00

Resolution: Fixed in v3.00.22



Description: The Source Code Library sent one less link retry than was configured.

Introduced: v3.00

Resolution: Fixed in v3.00.22



Description: The Source Code Library used an incorrect pointer in `sdnp070_readObj70`, resulting in invalid values for file transfer objects

Introduced: v3.00

Resolution: Fixed in v3.00.22



Description: The Source Code Library truncated Object 40 Variation 4 data to 32 bits instead of using double precision or long floating point.

Introduced: v3.00

Resolution: Fixed in v3.00.22



Description: An unsolicited vterm message during a vterm write caused the Source Code Library to incorrectly parse the write request.

Introduced: v3.00


Resolution: Fixed in v3.00.22

Version 3.00.20 (September 2, 2003)





Added callback function to let target application know when a channel is idle and can be disconnected. This change resulted in a change to the API: the number of parameters passed to `dnpchnl_openChannel()` was reduced by two: “`TMWCHNL_STAT_CALLBACK pCallback`” and “`void *pCallbackParam`” were moved to structure `DNPCHNL_CONFIG`, which is also an argument to the `dnpchnl_openChannel` function.


 *Description:* If TMWCNFG_SUPPORT_DOUBLE wasn't supported, compilation issues would occur.
Introduced: v3.00
Resolution: Fixed in v3.00.20


 *Description:* The Source Code Library did not handle Object 50 Variation 1 properly if Qualifer was other than 7
Introduced: v3.00
Resolution: Fixed in v3.00.20


Version 3.00.19 (August 8, 2003)


 Added configuration parameter to allow disabling of multifragment response messages.


 *Description:* The Source Code Library could write past the end of a buffer when generating a large number of events.
Introduced: v3.00
Resolution: Fixed in v3.00.19

 *Description:* The Source Code Library could call tmwtarg_startTimer even if the timer was already running. This issue only affected event driven implementations (i.e., implementations that did not use the polled timer implementation).
Introduced: v3.00
Resolution: Fixed in v3.00.19


 *Description:* Frozen Counters were not included in the response to a Class 0 poll
Introduced: v3.00
Resolution: Fixed in v3.00.19

 *Description:* A DNP Slave session would not cancel the selectTimer, which could cause a crash.
Introduced: v3.00
Resolution: Fixed in v3.00.19


 *Description:* A read specifying a limited quantity would return the specified number of each object type. This fix is required to pass the DNP3 2002 Conformance Test Procedures.
Introduced: v3.00
Resolution: Fixed in v3.00.19

- 

Description: If a Select Request message larger than 256 bytes was received, the Slave could crash.
Introduced: v3.00
Resolution: Fixed in v3.00.19

- 

Description: The DNP Link layer checked the FCB before checking the data block CRCs.
Introduced: v3.00
Resolution: Fixed in v3.00.19


- 

Description: Configuring allowMultiCROBRequests to FALSE (in the SDNPSESN_CONFIG structure) would cause Conformance Test 8.2.4.2 Multi Object Binary Output to fail for that session.
Introduced: v3.00
Resolution: Fixed in v3.00.19


Version 3.00.18 (July 22, 2003)

No updates to DNP Slave Source Code Library in this release.


Version 3.00.17 (July 10, 2003)

- 


Improved message displayed when a frame is received for an address that is not configured. This message now also includes the link address.

- 


Modified Session statistics callback function to provide event buffer overflow indication.

- 


An evaluation version of a working sample of a Windows application is now shipped with the Source Code Library. The evaluation expires after running for two hours.

- 


Improved implementation of internal data passed to callback functions to return a well defined structure and status codes.

- 


Added TMWTARG_CONFIG structure to allow specification of parameters that are common to multiple protocols.


- 


Description: Analog values did not set the over-range bit in the flags.
Introduced: v3.00
Resolution: Fixed in v3.00.17


- 

Description: On a multi-fragment response, the Source Code Library would continue to send fragments if it did not receive confirmation of the first fragment. This fix is required to pass the DNP3 2002 Conformance Test Procedures.
Introduced: v3.00
Resolution: Fixed in v3.00.17


-  **Description:** The Application layer First Fragment bit could still be set after the first fragment of a multifragment response. This fix is required to pass the DNP3 2002 Conformance Test Procedures.
Introduced: v3.00
Resolution: Fixed in v3.00.17


-  **Description:** Returning TMWDEFS_FALSE from sdnpdata_xxxInRead could result in invalid data being reported.
Introduced: v3.00
Resolution: Fixed in v3.00.17. Modified sdnpdata_xxxRead functions to return void. The target must fill in appropriate flags information to indicate a read failure.


-  **Description:** The Source Code Library did not indicate an error if the target assign class and assign deadband functions failed.
Introduced: v3.00
Resolution: Fixed in v3.00.17. The Source Code Library now checks for an error response from assignClass functions and from sdnpdata_anlgnDBandWrite. If one of these functions fails and returns FALSE, the IIN2.2 bit will be set.


-  **Description:** The Source Code Library improperly used Signed values for Analog Deadbands (Object 35 variation 1)
Introduced: v3.00
Resolution: Fixed in v3.00.17


Version 3.00.16 (June 19, 2003)


-  Added limit checks for sprintf statements. This prevents buffer corruption caused by writing past the end of the current buffer.


-  Added conditional compile statements to avoid unnecessary string copies when diagnostics are not enabled.


-  Cleaned up compiler warnings generated by GNU 68K compiler


-  Modified Source Code Library to avoid string copies for diagnostics when the diagnostics aren't enabled.


-  Added current version number to *tmwvrsn.c/h* in the *utils* directory to simplify determining which version of the Source Code Library is in use.


-  Cleaned up compiler warnings and issues identified by *lint*.


-  *Description:* A read of Object 60 Variation 2 that required multiple fragments would result in an “Error Unsupported” Qualifier on the Master
Introduced: v3.00
Resolution: Fixed in v3.00.16


-  *Description:* Object 60 Variation x Qualifier 7 or 8 did not read up to the specified number if multiple object types existed
Introduced: v3.00
Resolution: Fixed in v3.00.16


-  *Description:* Conformance test 8.11.2.5.8 failed if configured to use static memory with only 1 ASDU (txData) buffer. The Slave failed to send the required unsolicited message.
Introduced: v3.00
Resolution: Fixed in v3.00.16


-  *Description:* An *sprintf* statement in *dnpchnl.c* had an incorrect number of arguments, potentially causing crashes.
Introduced: v3.00.15
Resolution: Fixed in v3.00.16

-  *Description:* A Counter Freeze operation would cause a crash in the Source Code Library if *dnpdata_binCntrGetPoint()* returned Null to indicate that the point is currently disabled.
Introduced: v3.00
Resolution: Fixed in v3.00.16

-  *Description:* Invalid data would be reported by the Source Code Library if an *sdnpdata.c* read function returned *TMWDEFS_FALSE*, indicating that the read was not successful.
Introduced: v3.00
Resolution: Read functions in v3.00.16 will no longer have a return value. The target application must set one of the corresponding flags if there is a problem reading the point.

-  *Description:* Once the *Local_IIN* bit was set, it would remain set, even after the IED switched to remote mode.
Introduced: v3.00
Resolution: Fixed in v3.00.16


-  *Description:* The Forced flag was incorrectly reset for Binary Inputs on an Integrity Poll.
Introduced: v3.00
Resolution: Fixed in v3.00.16


 *Description:* If the DNP Link Layer discarded a frame (for example, due to an invalid start character or an invalid CRC), it might also discard the start character of the next frame.

Introduced: v3.00

Resolution: Fixed in v3.00.16


Version 3.00.15 (May 27, 2003)

 Add unsolicited response parameters to `sdnpseasn_modifySession()`. When the retry period is changed, the Source Code Library does not cancel an existing timer and then restart it. Instead, the next time a timer is started, it will use the new value. If it max retries is lowered below the current retry count, the new value will take effect the next time the current retry count is incremented.

 *Description:* The Source Code Library would crash when performing a Direct Operate No Ack followed by a Read


Introduced: v3.00.14

Resolution: Fixed in v3.00.15

 *Description:* The Source Code Library would not compile when configured for Static memory.

Introduced: v3.00.12


Resolution: Fixed in v3.00.15

 *Description:* The DNP Transport Function Sequence Number would rollover from 62 to 0, instead of from 63 to 0.

Introduced: v3.00


Resolution: Fixed in v3.00.15

Version 3.00.14 (May 13, 2003)

 Added support for DNP3 Time Synchronization over a Local Area Network (LAN)

 Added support for Function Code 24 (Record Current Time)

 Added support for Object 50, Variation 3 (Time and Date Last Recorded)

 The DNP3 link layer can now request and respond to requests for link status (link function code 9). In addition there is a configuration parameter to support issuing a request for link status periodically. There is also a new function in *dnplink.c* that will allow the user to issue a request for link status.



Improved detection of sessions going offline. Any application layer error will now take a session offline. Receiving any valid fragment from that session will take it back online.



Improved memory management. Implemented a new memory management scheme that supports static memory as well as providing limits on the number of each type of structure that can be allocated. It also provides details on how many structures of each type have been allocated and keeps track of the maximum number that was allocated. This provides feedback that can be used in sizing applications and monitoring allocated memory. Cleaned several areas in which memory was not being freed properly.



Added configuration option to SDNP SCL to use received destination address (for compatibility w/ v2.x SCL). Added a new configuration parameter, `validateSourceAddress`, which enables or disables the validation of the source address in received frames. In addition DNP3 responses are now always replied to the source address of the received request as opposed to the destination address of the session. If validation is enabled these two addresses will always be the same. If validation is disabled they might be different. In fact, if validation is disabled a DNP3 will properly process requests from multiple master sessions. Unsolicited responses are always sent to the destination address configured in the slave session.



Description: If a DNP3 slave device received any request, other than an unsolicited confirmation, while it was waiting for an unsolicited confirmation it would inadvertently cancel the unsolicited retry timer and hence stop retrying unsolicited responses.

Introduced: Version 3.00

Resolution: Fixed in Version 3.00.14



Description: DNP Fragment Size and frame size configuration did not work properly.

Introduced: Version 3.00

Resolution: Fixed in Version 3.00.14. There are now channel level configuration parameters for transmit/receive fragment and frame size. Frame size is specified in bytes on the wire, fragment size is specified as application layer data bytes in a fragment.










Description: DIR bit was not set on application confirmations to events.




Introduced: Version 3.00


Resolution: Fixed in Version 3.00.14

Version 3.00.12 (April 28, 2003):

-  Improved description of formal parameters and functions in header files.
-  Added Incremental Application Layer Response Timeout configuration parameter.
-  Unsolicited Responses now retry using the offline retry period, even if new events are detected.
-  The Incremental Timeout timer is now disabled if the timeout value is set to zero
-  *Description:* The Source Code Library could reference unallocated memory and cause a crash if Link Layer Confirms were enabled and the Application Layer timeout was configured to be less than ((Link Layer timeout) * (number of Link Layer retries+1)).
Introduced: Version 3.00
Resolution: Fixed in version 3.00.12. When the Application Layer times out, it now tells the Transport Layer, which in turn tells the Link Layer.
-  *Description:* If the analog deadband value exceeded the range of its storage container (e.g., if it is defined as TMWDEFS_SHORT and exceeded 0xFFFF), the Source Code Library truncated the value instead of using a full-scale value.
Introduced: Version 3.00
Resolution: Fixed in version 3.00.12
-  *Description:* Analog values were stored as unsigned (instead of signed) values
Introduced: Version 3.00
Resolution: Fixed in version 3.00.12. Modified DNPUTIL_ANALOG_VALUE structure and related code to use signed values.

Version 3.00.11 (April 24, 2003):

-  If a connection is broken or the remote device restarts, the Source Code Library will attempt to reopen the connection.
-  The maximum number of events is now configurable. See *tmwcnfg.h*
-  Source Code Library now passes Triangle MicroWorks, Inc. DNP Conformance Test Scripts.

-  *Description:* The Source Code Library could send unsolicited responses even when `unsolicited` was configured as `TMWDEFS_FALSE`.
- Introduced:* Version 3.00
- Resolution:* Fixed in version 3.00.11

Version 3.00 (November 20, 2002):



The Triangle MicroWorks, Inc. Source Code architecture was redesigned and reimplemented to follow modern software engineering practices and leverage new techniques in software design. We have also incorporated suggestions from our existing customer base. The main advantages to this redesign include:

- 1) Common source code architecture across all TMW libraries - The use of a common architecture across all libraries significantly reduces time required to port additional protocols to a target device. In addition, using common software across all products results in better code, since common functions are utilized much more frequently than protocol specific functions.
- 2) Source Code Library calling routines are now compatible with a wider variety of event driven techniques, allowing the target application to achieve higher performance.
- 3) Most configuration parameters are now passed as arguments in Source Code Library function calls instead of macros. This allows for data hiding and protects configuration parameters from accidentally being changed while the Source Code Library is running. In addition, macro calls to target hardware and the database interface routines were replaced with function calls.
- 4) Significant reduction in the time to implement new features - In addition to being more flexible, the use of modern software design practices significantly reduces the time required for Triangle MicroWorks to support new features. As the existing protocols are constantly being improved by the associated Technical Committees and Working Groups, it is essential that the Source Code Libraries are able to remain up to date with the latest standards.
- 5) Added the ability to communicate with multiple masters. The Source Code Library can track separate sequence numbers, databases, and event buffers so that each remote master is totally independent.



We have updated the new DNP3 Slave Source Code Library to be configurable for either a single-port or multi-port implementation.



Warm and Cold restart would not work correctly if certain Internal Indication (IIN) bits were also set in the response message.

Version 2.32 (July 10, 2002):

dnp3 Added support for Object 12 Variations 2 and 3, Pattern Control Blocks and Pattern Masks. These variations support the simultaneous control of multiple binary outputs.

dnp3 Added support for Object 70 Variation 2, File Authentication. The File Authentication request is now supported and the authentication key is included in file open and file delete requests.



Description: Binary commands to uninstalled points were returning a null response with the OUT OF RANGE IIN bit set. The specification requires the response to echo the request with a status of NOT SUPPORTED(4) in addition to setting the IIN bit.

Resolution: The SCL now responds as per the specification.



Description: Link layer confirms were being returned for broadcast messages. The specification states that the slave device should not respond to broadcast messages.

Resolution: The SCL no longer responds to broadcast messages.

Version 2.31 (November 14, 2001):

dnp3 Now fulfills all requirements for compliance with DNP3-2001. Most of these requirements were already covered by the Source Code Library, but the following changes were made in this version:

- The optional limitation on the number of attempts to regenerate an unsolicited message was removed. The regeneration attempts must continue indefinitely but can occur at a less frequent interval. The configuration macro **DNP_CFG_UN SOL_MAX_RETRIES()** now defines the number of retry attempts allowed before the retry interval is controlled with the new configuration macro **DNP_CFG_UN SOL_OFFLINE_INTERVAL()** instead of **DNP_CFG_UN SOL_RETRY_DELAY()**.
- Three new Control Relay Output Block status codes were added. They are 8 (**TOO_MANY_OPS** – request not accepted because too many operations requested), 9 (**NOT_AUTHORIZED** – request not accepted because of insufficient authorization) and 126 (**UNDEFINED** – request not accepted because of some other undefined reason).

dnp3 Added support for object 110 (static string object) and object 111 (event string object). Read, write, assign class and response function codes are included in the support for object 110. Read, response, and unsolicited function codes are included in the support for object 111. The following macros were added in order to facilitate support for object 110 and 111:

- **DNP_DATA_STRING_QTY()** – returns the total number of string objects.
- **DNP_DATA_STRING_READ()** – returns the value of a string object point.

- **DNPDATA_STRING_CHANGED()** – returns true if a string object has changed, else returns false.
- **DNPDATA_STRING_WRITE()** – writes a new value to a string object point.
- **DNPDATA_STRING_EVENT_CLASS()** – returns the event class of a string event.
- **DNPDATA_STRING_ASSIGN_CLASS()** – assigns a new class to a string event.
- **DNPDATA_STRING_EVENT_BUFFER_SIZE()** – sets the array size of the string event buffer.
- **DNPCONFIG_RBE_STRING_SCAN_PRD()** – the interval, in milliseconds between scans for string events.



Corrected a bug which was not allowing full sized frames to be received. The CRC's and the two sync characters and the link layer length byte were being subtracted from the allowed length of a frame but this did not need to be done since the lookup table which was being used to check the length of the frame already accounted for these bytes in the frame length. Introduced in version 2.29.



Corrected several bugs in sequential file transfer. The file transfer event buffer was not cleared after a file read request was received because a confirm request was not sent in the response. A file read response would not increment the file pointer if the response data was being sent in unsolicited responses. After an assign class request was received, the next class poll request would not parse correctly. After a read request, an unsolicited response would be sent immediately instead of waiting for an unsolicited reply timeout. Corrected a bug where a file close would not work correctly if a file was just created due to a file open for write command. Fixed an initialization problem with the status of a file delete command. These bugs were all introduced in version 2.27.

Version 2.30 (August 10, 2001):



Corrected a misspelling of **DNPDBAS_SUPPORT_FILE_TFER** which had three P's instead of two. It is used to remove code during compilation if file transfer support is turned off in the file **dnpdata.h**. Introduced in version 2.27 when sequential file transfer was added.

Version 2.29 (July 19, 2001):



Corrected a bug which did not allow events other than sequential file events to be reported correctly. The pointers for the response object header and response object data were not incremented correctly while building a response to a read of a specific object type or a class poll. Introduced in version 2.27.



Corrected a bug which caused compiler warnings if sequential file transfer support was turned off. Now, the SCL compiles correctly whether or not sequential file transfer support (Object 70, variations 3-7) is supported. Introduced in version 2.27.



Corrected bug in **dnpphys.c** where a buffer overflow could occur if the number of bytes received exceeded the maximum receive frame buffer size. The length byte embedded in the message is checked against the maximum receive frame buffer

size. If the length is too large, the frame is not reparsed and the parsing routine starts looking for a new sync character starting with the byte directly following the length byte. Introduced in version 2.00.



Corrected bug which could allow an initial unsolicited response to occur even if the restart IIN flag is not set. Now the restart IIN flag being set is a condition of the initial null unsolicited response being sent. Introduced in version 2.02.



Fixed bug when a select request had a sequence number of 15 and the corresponding operate request had a sequence number of 0 when the sequence number loops back to zero. This would cause an error for no matching select. This is now a valid case and does not cause a no matching select error. Introduced in version 2.28.

Version 2.28 (June 26, 2001):



Corrected two problems with select/operate sequences that were introduced in V2.21. The first bug allowed multiple operate requests to be performed on the same select request. Now only one operate request, who's sequence number is one greater than the select request, is allowed to operate on that select request. The second bug allowed an operate request to be performed on a select request even if an invalid operate request had already been received. Now if an invalid operate request is received, the select request is also invalidated.



Corrected a problem that occurred with write requests that was introduced in V2.27. Before version 2.27 which added sequential file transfer, all write requests were for static data. But in version 2.27, sequential file transfer write requests use event data. A bug was introduced that tried to process all write requests as event data, not static data. This bug was fixed so file write requests use event data and all other write requests use static data.

Version 2.27 (February 15, 2001):

dnp3 Added support for sequential file transfer as described in the DNP Technical Bulletin 2000-001, Sequential File Transfer Objects. The following table details the new objects included in this version:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes (dec)	Qual Codes (hex)
70	3	File Command Object	25,27	0x5B		
70	4	File Command Status Object	1,22,26,30	0x06,0x07, 0x08,0x5B	129,130	0x5B
70	5	File Transport Object	1,2,22	0x06,0x07, 0x08,0x5B	129,130	0x5B
70	6	File Transport Status Object	1,22	0x06,0x07, 0x08	129,130	0x5B
70	7	File Descriptor Object	1,22,28	0x06,0x07, 0x08,0x5B	129,130	0x5B

Authentication, (Obj 70, Var 2) was not included. This implementation limits the number of open files to one. The Operational Mode field in the File Command Object supports the APPEND mode. The ABORT function code used to stop file transfers is supported. Custom status code strings are not supported so for the File Command Status Object and File Transport Status Object, no optional ASCII strings to describe additional error types are allowed. All responses to requests are stored as events. Object 70, variations 4 through 7 can be polled for specifically through read requests or can be retrieved through a class poll. No events are sent immediately if the results are known. Object 70, variations 4 through 7 are always the same class. Assigning a class to one variation will assign the same class to all variations. When a file is opened for read, the data from the file is not stored in the event at the time of the read request. The file is read at the time of the event poll. Therefore if there are any File Transport Object events still in the event buffer, it is important to retrieve this data before closing the file otherwise the data will become unavailable.



Corrected a problem that occurred during the installation procedure of the DNP Slave SCL. For the simulation functions (**binInputChanged()**, **anlgInputChanged()**, **cntrChanged()** and **fcntrChanged()**) in both dnpsim1.c and dnpsim2.c, if a point number supplied was greater than the maximum quantity defined in **BIN_INPUT_QTY()**, **CNTR_QTY()**, and **ANLG_INPUT_QTY()** the SCL would crash. A check has been placed inside each of the above functions so that if the point number is out of range then the function returns a failure but does not crash. Introduced in version 2.06.

Version 2.26 (January 3, 2001):

dnp3 Now, if not confirmed, the initial null unsolicited response will continue forever, regardless of the setting of **DNPCONFG_UN SOL_MAX_RETRIES()**. This conforms to the DNP3-2000 Certification Procedure.

dnp3 Now, the count of unsolicited retries is reset upon the successful reception of any correctly addressed application layer message from the DNP3 Master station. This means that if unsolicited responses have stopped because the number of consecutive unconfirmed unsolicited responses has exceeded **DNPCONFG_UN SOL_MAX_RETRIES()**, then unsolicited responses will start again after the successful reception of any correctly addressed application layer message from the DNP3 Master station. Previously, the count of unsolicited retries was reset only when an “enable unsolicited” function (code 20) request was received.

dnp3 Now, link layer confirmation timers are started only after successful transmission of link layer frames containing requests for confirmation has been indicated. (Previously link layer timers were started after link layer frames were generated, but before they were transmitted.)

dnp3 Now, if an application layer response containing a request for application confirmation is transmitted and if link layer confirmations are also being requested, the application confirmation timer will not be started until after the link layer confirmation has been received. This means that link layer confirmation times do not need to be considered when setting the application layer confirmation timeout value.



Removed **maxInterTime** as a parameter to the **DNPTARG_COMM_RECEIVE()**. Target Application software should use their own timeout value to determine if an inter-character time has occurred. This timeout may be set much tighter, since it may be used by very low-level communications routines, than the **DNPCONFG_PHYS_CHAR_TIMEOUT()**, which is used by the Source Code Library to perform similar, but not as precise inter-character timeout tests.



Enhanced protocol analyzer and command line interface to include the following:

- New script and delay commands to allow simple scripting, or pre-built command sequences.
- Display of expected and received CRCs when CRC mismatches occur.
- Extended help for individual commands that is available by entering “help *command*.”
- Corrected problem with show command that did not include all possible parameters in its display.
- When displaying values with either the set or show command, no longer remove final 's' (plural) from some units names (e.g., “ms”) when the corresponding value was 1 (singular).



Continued to enhance comments, and to remove warnings produced by specific compilers.



Corrected a problem that stopped unsolicited messaging if link layer confirmations were enabled and a link reset failed while attempting to transmit the unsolicited message. Unsolicited messaging would have restarted after a successful link reset caused by a polled response. Now, unsolicited messaging and link resets will continue normally, based on unsolicited configuration parameters such as of **DNP_CFG_UNSol_MAX_RETRIES()** and **DNP_CFG_UNSol_RETRY_DELAY()**. Introduced in version. Introduced in version 2.00.



Corrected a problem that occurred with class 0 polls (object 60, variation 1) for simple implementations that disable support for all event object variations (object 2, 22, 23, and 32). The problem resulted in no static data being responded to the class 0 poll. This problem was introduced in V2.20.



Corrected a problem with the example, simulated database contained in **DNPsim1.c**. This problem did not affect the Source Code Library. The problem was that the last reported value used as a comparison to detect running counter change events was the last reported *frozen* counter value, not the last reported *running* counter value. This caused endless reports of both running and frozen counter events whenever a **DNPsim1** counter was incremented. Introduced in version 2.19

Version 2.25 (November 1, 2000):

dnp3 No longer allow **DNPDEFS_IIN_BUFFER_OVFL** to be set by **DNPTARG_GET_IIN()**. The DNP Technical Committee has ruled that this bit can only reflect the state of the event buffers, which are internally managed by the Source Code Library. Previously, multi-fragment requests were not processed, and the **DNPDEFS_IIN_BUFFER_OVFL** was asserted; now fragments are processed regardless of the setting of their first and final bits. Also, if a response to a operate or select command cannot fit in the response buffer, the **DNPDEFS_IIN_BUFFER_OVFL** is no longer asserted.

dnp3 Added support for reading object 80, variation 0.



Corrected a problem related to the disabling of support for specific change event types: If support for a specific binary change variation, analog change event variation, or running counter change event variation is disabled by setting any of **DNPDATA_SUPPORT_OBJ_2_Vxx**, **DNPDATA_SUPPORT_OBJ_32_Vxx**, or **DNPDATA_SUPPORT_OBJ_22_Vxx** to **TMWDEFS_FALSE**, then polls for these variations would have caused the Source Code Library to stop responding. This problem was introduced in Version 2.18 in an attempt to fix a problem introduced in Version 2.16.



Corrected a problem related to assign class if either running counter change events or frozen counter change events are disabled: If all **DNPDATA_SUPPORT_OBJ_22_Vxx** are set to **TMWDEFS_FALSE**, or if all **DNPDATA_SUPPORT_OBJ_23_Vxx** are set to **TMWDEFS_FALSE**, and if function 22 (assign class) is attempted to either object 20 or object 21, a function call through a

null function pointer would have resulted. This problem was introduced in Version 2.19.

Version 2.24 (October 24, 2000):

dnp3 Added support for reading and writing user-defined indications (object 80). More specifically, added the following symbols:

DNPDATA_SUPPORT_OBJ_80_READ,
DNPDATA_USER_INDICATIONS_QTY(),
DNPDATA_USER_INDICATIONS_READ(), and
DNPDATA_USER_INDICATIONS_WRITE().

dnp3 Now, DNP3 Masters may only clear the Restart IIN (IIN1-7). If a DNP3 Master attempts to set IIN1-7, the bit will not be set and IIN2-2 (Parameter Error) will be asserted instead.



Changed the return type of the following macros from the **DNPSCL_CTRLSTAT** type that was defined in Version 2.23, to **DNPDEFS_CROB_ST** in order to allow the macros to return any of the defined DNP3 response status codes:

DNPDATA_BIN_OUTPUT_SELECT(),
DNPDATA_BIN_OUTPUT_CONTROL(),
DNPDATA_ANLG_OUTPUT_SELECT(),
DNPDATA_ANLG_OUTPUT_CONTROL().



Corrected a problem introduced in Version 2.20 that resulted in the application layer confirmation timer, unsolicited confirmation timer, and unsolicited retry timer being restarted when *any* message was transmitted, even if it wasn't an unsolicited message (e.g., a response to an control would delay unsolicited responses because the retry timer would be restarted).



Corrected a problem introduced in Version 2.21 that caused the restart IIN (IIN1-7) to clear automatically after an initial message is transmitted. The correct operation requires the DNP3Master to explicitly clear the IIN1-7.

Version 2.23 (October 19, 2000):



Changed the return type of the following macros from **TMWDEFS_BOOL** to a newly created **DNPSCL_CTRLSTAT** in order to allow the macros to indicate a control operation other than simple success or failure:

DNPDATA_BIN_OUTPUT_SELECT(),
DNPDATA_BIN_OUTPUT_CONTROL(),
DNPDATA_ANLG_OUTPUT_SELECT(),
DNPDATA_ANLG_OUTPUT_CONTROL().



Removed commas from the last element of enum definitions. Some compilers report these as errors.



Now, when an illegal function request from a DNP Master Station is detected, and the "bad function" IIN is asserted, a diagnostic message is displayed through **DNPDIAG_PUTS()**, and the **funcUnsupported** statistical counter is incremented

through **DNPPHYS_COUNT_STATISTIC()**. Also, when a illegal qualifier is detected, changed from incrementing **funcUnsupported** to **qualUnsupported**.



Within DNPlink.c, corrected a type-cast problem with CRC calculations that prevented normal communications. This problem was introduced in Version 2.22.



Corrected a problem with **dnpdbas_storePointNum()** when the number of points within an object type is exactly 256. This problem caused invalid data in response messages only when the index qualifier (0x17) was used (as when reporting change events). No problem occurred if the number of points was less than 256 or greater than 256. This problem was introduced in Version 2.00.

Version 2.22 (October 18, 2000):



Changed use of **DNPCONFIG_UNSQL_MAX_RETRIES()** to limit the number of "retries" not "attempts"; i.e., if **DNPCONFIG_UNSQL_MAX_RETRIES()** is 0, a single unsolicited message will be transmitted, but if not confirmed, no retries will be attempted. In Version 2.21, if **DNPCONFIG_UNSQL_MAX_RETRIES()** was 0, no unsolicited messages would have been transmitted.



Revised comments in DNPconfig.h that now states that link addresses FFF0 to FFFF (65519 to 65535) are reserved for broadcast addresses.



Within DNPdata.h, corrected comments describing valid return values for **DNPDATA_DFLT_OBJ_xx_VAR()** macros.



Added type-casts, added example #pragma's, changed variable types, etc., to reduce and/or remove warnings produced by strict compilers. More specifically, all files compile without warnings using warning level 4 of Microsoft Visual C++ V5.0.



Within DNPlink.c, corrected a problem that prevented correct operation of data link confirmations when **DNPCONFIG_LINK_CNFM_MODE()** was set to **TMWDEFS_LINKCNFM_SOMETIMES** or **TMWDEFS_LINKCNFM_ALWAYS**. This problem was introduced in Version 2.20.



Within DNPdbas.c, removed more potential compilation errors from the **DNPDBAS_STORE_POINT_NUM()** macro when **DNPDATA_SUPPORT_OVER_256_ANYOBJ()** is **TMWDEFS_FALSE**. This problem was originally introduced in Version 2.18; a correction was attempted in Version 2.21.

Version 2.21 (October 6, 2000):

dnp3 Now fulfills all requirements for compliance with DNP3-2000. The specific revisions are described below.

dnp3 Added support for a configurable number of unsolicited retries when no confirmation has been received via **DNPCONFIG_UNSQL_MAX_RETRIES()**. This value can be set from 0 to 255 with 255 being equivalent to infinite retries. Once the maximum number of retries has been reached, no more unsolicited messages will be sent unless a confirmation is received or the master sends an enable unsolicited message. This fulfills requirements set forth by DNP Technical Bulletin 9912-002.

dnp3 Added support for broadcast messages to include addresses 0xFFFFD and 0xFFFFE while still including the original broadcast address 0xFFFF. 0xFFFFE is handled the same as 0xFFFF in that if a broadcast message is received on either of these two addresses, the All Stations flag, IIN1-0 will be set and remain asserted until a confirmation is received from the Master. When a broadcast message is received on address 0xFFFFD, the All Stations flag, IIN1-0, is set in the next message sent by the slave and is immediately cleared afterwards. No confirmation is requested from the Master. This fulfills requirements set forth by DNP Technical Bulletin 9912-003.

dnp3 Added support for select and operate commands to allow multiple select commands to be received with the same sequence number and data without restarting the select timer. In addition, if a select command is received with a different sequence number or data has changed it is considered a new select command and the select timer is restarted. Finally, the operate command must have a sequence number one greater than the corresponding select command. If the sequence number is not correct the operate command will not be executed. If multiple operate commands are received with the same sequence number and data, the slave will repeat the previous response but will not execute the operate command again. This fulfills requirements set forth by DNP Technical Bulletin 2000-002.

dnp3 Now supports `DNPDEFS_DBAS_FLAG_REFERENCE_CHK` when the `DNPDATA_ANLG_INPUT_READ()` and `DNPDATA_ANLG_INPUT_CHANGED()` macros are called.



Within `DNPdbas.c`, removed a potential compilation error from the `DNPDBAS_STORE_POINT_NUM()` macro when `DNPDATA_SUPPORT_OVER_256_ANYOBJ()` is `TMWDEFS_FALSE`. This problem was introduced in Version 2.18.

Version 2.20 (September 7, 2000):

dnp3 Added support for floating point analog input change event objects, and added support for analog input reporting deadband objects. Specifically, support for the following object variations was added:

obj 32, variation 5	(short floating point analog input change event without time)
obj 32, variation 7	(short floating point analog input change event with time)
obj 34, variation 1	(16-bit analog input reporting deadband)
obj 34, variation 2	(32-bit analog input reporting deadband)
obj 34, variation 3	(floating point analog input reporting deadband)



Now uses a newly defined type `DNPSCS_ANLG_UNION` for analog values passed through the database interface. Previously, only signed 32-bit integers were used.

The new **DNP_SCL_ANALG_UNION** includes integers and floating point formats. This allows Target Application analog types to be transmitted and received in their native format. Conversions from native format to request object variation format is automatically supplied by the Source Code Library.



Created new **DNP_scl.h** to consolidate literal constant and data type definitions that are specific to the Source Code Library. **DNPdefs.h**, on the other hand, contains literal constant and data type definitions that are specific to the DNP3 protocol.



Enhanced the example database interface implementations in **DNPsim1.*** and **DNPsim2.***, including examples of using the newly defined **DNP_SCL_ANALG_UNION**, and examples of the newly supported analog input deadband objects.



Added **DNP_CONFIG_SET_UN_SOL_INIT_ENABLED()** to allow modification of the **DNP_CONFIG_UN_SOL_INIT_ENABLED()** parameter through optional remote terminal interface commands.



Added more type-casts to prevent more warnings from strict compilers.



Corrected a problem with cold and warm restarts: Previously, if the cold or warm restart request required an application confirmation, the restart would have been initiated after the confirmation was sent and before any response to the response could be sent. Now the restart is not initiated until after the response to the request has been sent. This problem was introduced in version 2.00.



Corrected a problem with unsolicited reporting: Previously, when at least one event class was disabled for unsolicited reporting, it was possible for empty unsolicited messages to be reported when other unsolicited conditions for the disabled class were met. Now, only conditions for enabled unsolicited classes are considered when determining if an unsolicited message can be sent. This problem was introduced in version 2.12.



Corrected a problem with unsolicited reporting relating to the buffer overflow internal indication (IIN): Previously only one unsolicited response message would be sent when a buffer overflow condition existed. Now, unsolicited response messages will be repeatedly sent even if their only content is to indicate the buffer overflow condition. They will be sent until an application layer confirmation is received to indicate the remote Master has received the overflow indication. This problem was introduced in version 2.15.



Corrected typographical problems in **DNPdpa.c** (replaced references of **PDPDATA_SUPPORT_OBJ_110_111** with **DNPDATA_SUPPORT_OBJ_110_111**), in **DNPdiag.*** (replaced references of **DNPDATA_SDNPLIB_xxxx** with **DNPDATA_SCL_xxxx**), and in **DNPcmd.c** (replaced missing “Unsupported” printf parameter that only caused errors when **DNP_CONFIG_UN_SOL_SUPPORTED()** was **TMWDEFS_FALSE**). These problems were introduced in versions 2.16, 2.19, and 2.19, respectively.

Version 2.19 (March 5, 2000):

dnp3 Added support for counter change events and frozen counter events. Specifically, support for the following object variations was added:

obj 22, variation 1	(32-bit counter change events without time)
obj 22, variation 2	(16-bit counter change events without time)
obj 22, variation 5	(32-bit counter change events with time)
obj 22, variation 6	(16-bit counter change events with time)
obj 23, variation 1	(32-bit frozen counter events without time)
obj 23, variation 2	(16-bit frozen counter events without time)
obj 23, variation 5	(32-bit frozen counter events with time)
obj 23, variation 6	(16-bit frozen counter events with time)

dnp3 Application confirmation timers and unsolicited response timers are now started only after a message requiring a confirmation is transmitted. Previously, these timers were started when a message was generated, thereby counting any time between generation and transmission against the expected confirmation time from a remote DNP Master.

dnp3 Change events for binary inputs, analog inputs, counter change events, and frozen counter events are no longer buffered if the corresponding point has not been assigned to an event class (1, 2, or 3). Previously, only scanning the points was prevented; now both are prevented.

dnp3 Improved support for explicit polls for change events that are not assigned to an event class. While this is no longer possible for binary inputs, analog inputs, counter change events, and frozen counter events (see above), this change is particularly important for virtual terminal and string events which are not normally assigned to an event class.

dnp3 When `TMWDEFS_EVENT_MODE_CURRENT` (only one change event is reported per point, no matter how many times a point is detected to have changed between reports) is used for analog inputs, and multiple change events for the same point are detected between reports, changes in event class between event detections are now handled. Previously, the class was assumed not to have changed, and the count of events in the original class and the count of events in the new class were not altered. Now, the count of events in the original class is decremented, and the count of events in the new class is incremented.

dnp3 Previously, after a confirmation for a earlier message containing events was not received, and the generation of a new message was beginning, only the events that were previously transmitted, together with any new events, were marked ready for transmission. Now, all events are marked ready for transmission. This change mainly resulted in code-space savings, with no real functional differences.



To support running counter change events and frozen counter events, added the following:

DNPRBE_COUNTER_EVENT	type definition
For running counter change events:	
DNPCONFIG_RBE_CNTR_SCAN_PRD()	configuration parameter
DNPDATA_DFLT_OBJ_22_VAR()	database interface parameter
DNPDATA_SUPPORT_OBJ_22_Vx()	database interface parameter
DNPDATA_CNTR_CHANGED()	database interface function
DNPDATA_CNTR_EVENT_MODE()	database interface parameter
DNPDATA_CNTR_EVENT_BUFFER_SIZE()	database interface parameter
DNPDATA_CNTR_EVENT_CLASS()	database interface parameter
DNPDATA_CNTR_ASSIGN_CLASS()	database interface function
dnprbe_cntrStoreEvent()	report-by-exception alternative entry point
dnprbe_cntrScan()	report-by-exception alternative entry point
RbeCntrScanPrd	Terminal interface configuration parameter.
Dfltobj22Var	Terminal interface configuration parameter.
For frozen counter change events:	
DNPCONFIG_RBE_FCTR_SCAN_PRD()	configuration parameter
DNPDATA_DFLT_OBJ_23_VAR()	database interface parameter
DNPDATA_SUPPORT_OBJ_23_Vx()	database interface parameter
DNPDATA_FCTR_CHANGED()	database interface function
DNPDATA_FCTR_EVENT_MODE()	database interface parameter
DNPDATA_FCTR_EVENT_BUFFER_SIZE()	database interface parameter
DNPDATA_FCTR_EVENT_CLASS()	database interface parameter
DNPDATA_FCTR_ASSIGN_CLASS()	database interface function
dnprbe_fctrStoreEvent()	report-by-exception alternative entry point
dnprbe_fctrScan()	report-by-exception alternative entry point
RbeFctrScanPrd	Terminal interface configuration parameter.
Dfltobj23Var	Terminal interface configuration parameter.



For consistency with the additions to the interface for counter change events and frozen counter events, also renamed (and/or moved) the following:

Old	New
DNPCONFIG_RBE_BIN_BUFFER_SIZE()	DNPDATA_BIN_EVENT_BUFFER_SIZE()
DNPCONFIG_RBE_ANLG_BUFFER_SIZE()	DNPDATA_ANLG_EVENT_BUFFER_SIZE()
DNPDATA_CNTR_FROZEN_READ()	DNPDATA_FCTR_READ()
dnprbe_scano1()	dnprbe_binScan()
dnprbe_scano32()	dnprbe_anlgScan()



Added new timers for independently scanning counter for change events and frozen counters for events.



Added example implementations of a database interface to running counter change events and frozen counter events to DNPsim1 and DNPsim2.



Added new **DNPTARG_NUM_BITS_IN_A_BYTE** configuration parameter which specifies how many bits are contained in an instance of a **TMWDEFS_UCHAR** data type. This parameter is used by the **DNPPkbin** module, which previously assumed bytes contained 8 bits. Some compilers, particularly some of those used for Digital Signal Processors, use 32-bit entities for all data types, including bytes.



Reorganized the data structures and renamed variables and functions within **DNPrbe.c** to be more cohesive, to encapsulate more data and functions within the **DNPrbe** module, to improve DNP response time, and to reduce code space requirements.



The unsolicited notification timers and retry timers have been moved completely inside **DNPrbe.c**. They are no longer started when an unsolicited response is transmitted; instead they are conditionally re-started when the confirmation is received or a timeout occurs. Also, they are restarted from the timestamp of the oldest event still queued but not yet transmitted, thus providing a slightly more accurate notification delay.



To **DNPdbas.c**, renamed and added local functions, and added macros (e.g., **DNPDBAS_xxxx_PROTOTYPE()**) to enhance readability. These changes alone resulted in no functional differences.



Removed uses of **DNPDBAS_xxxx_FUNC** function pointer type-definitions since some elementary compilers cannot handle them. These were added in Version 2.16. Also added a **#pragma** that is necessary for a specific compiler, but placed this **#pragma** under conditional **#ifdef** control.



Renamed and reorganized some members of **DNPDPA_TYPE**, which is used internally by the Source Code Library. These changes merely enhance readability and enhance data and functional encapsulation.



Removed duplicate text in protocol analyzer legend.



Updated **SDNPwin C-I Guide.rtf**, the example implementation of a Configuration and Interoperability Guide, with more Microsoft Word “comments” concerning conditional support of unsolicited reporting and assign class functionality.



Fixed problem with calling **dnprbe_binStoreEvent()** from within **DNPPkbin.c**. In versions prior to V2.16, **dnprbe_binStoreEvent()** was passed a **TMWDEFS_UCHAR** to indicate the binary point's Boolean status (on or off). In V2.16, a change was made to pass a **DNPDEFS_DBAS_FLAG** value that indicates other database flags (e.g. on-line/off-line) in addition to its on or off status. However, the call in **DNPPkbin.c** was not changed for Version 2.16, and a Boolean status was incorrectly passed instead of a **DNPDEFS_DBAS_FLAG** value. That call now correctly uses a **DNPDEFS_DBAS_FLAG** value.



To **DNPvterm.c** and **DNPvterm.h**, added **#include "dnpconfig.h"** to resolve conditional compilation directives.

Version 2.18 (January 28, 2000):

dnp3 Changed conditional support in the **dnpdbas_objTable[]** to exclude static objects and controls completely (rather than make them parse-only) when support is disabled using **DNPDATA_SUPPORT_OBJ_nn_Vxx**. Event objects, on the other hand, are left as parse-only when support is disabled.



To reduce code-space requirements, now conditionally excludes select, operate, and direct-operate functionality if support for binary and analog output controls is disabled using **DNPDATA_SUPPORT_OBJ_12_V1** and **DNPDATA_SUPPORT_OBJ_41_Vx**.



Added comment to **DNPDATA_SUPPORT_FREEZE** that states that freeze operations are required if binary counters are supported. Also modified comments in **DNPsim1.c** and **DNPsim2.c** preceding inclusion of both running and frozen counters in the class 0 response (in **dnp3sim1_staticPollObjHdrs[]** or **dnp3sim2_buildClass0Headers()**).



In **dnp3sim2.c**, addition conditional compilation around **sim2ReadPackedBit()**: excluding it, and the reference to the **dnp3pkbin** module, if fast packed binary scanning is disabled (**DNPSIM2_PACKED_BIN_NUM_PACKS** is zero).



Corrected a problem when support for object variations is disabled. Specifically, if support for binary or analog output controls was disabled by setting **DNPDATA_SUPPORT_OBJ_12_V1** or **DNPDATA_SUPPORT_OBJ_41_Vx** to **TMWDEFS_FALSE**, and if an operate (or select) function were attempted, a call through a null function pointer would have been executed. Similarly, when support for static or event object variations was disabled, the variations were left as parse-only; if a read function were attempted, a call through a null function pointer would have been executed. Now, some of the object variations are completely excluded (see the “**dnp3**” item above). For all other objects, function pointers are tested to determine if support is disabled. This problem was introduced in Version 2.16.

Version 2.17 (January 21, 2000):



Corrected a problem in **dnp3rbe_availableClassIINs()** that may have prevented simultaneous indications of class 1, 2, and 3 data through Internal Indications (IINs). This problem was introduced in Version 2.12.

Version 2.16 (January 16, 2000):

dnp3 Added support for the following object variations:

obj 1, variation 2	(binary inputs with status)
obj 20, variation 1	(32-bit running counters with status)
obj 20, variation 2	(16-bit running counters with status)
obj 20, variation 5	(32-bit running counters without status)
obj 21, variation 1	(32-bit frozen counters with status)
obj 21, variation 2	(16-bit frozen counters with status)
obj 21, variation 9	(32-bit frozen counters without status)

Support for object 20, variation 6 and object 21, variation 10 (both 16-bit without status) was in-place previously.

- dnp3** Added **DNP_CFG_UNSOLED_INIT_ENABLED()** to allow unsolicited messaging to be enabled at startup without being explicitly enabled by a remote DNP master. This configuration parameter should be left at its default value of 0, which **does not** enable unsolicited messaging at startup without being explicitly enabled by a remote DNP master. Any other value other than 0 will cause a violation of “The DNP V3.00 Intelligent Electronic Device (IEC) Certification Procedures, V1.00.” This parameter was added to allow interoperability with DNP Masters that do not yet meet the requirements necessary to interoperate with DNP3-1999 certified slaves.
- dnp3** To allow a more complete interface to database status flags for individual data points, added **pFlags** parameter to the following macros:

```
DNPDATA_BIN_INPUT_READ()
DNPDATA_BIN_INPUT_CHANGED()
DNPDATA_BIN_OUTPUT_READ()
DNPDATA_CNTR_READ() (also added p16BitRoller)
DNPDATA_FROZEN_CNTR_READ() (also added p16BitRoller)
DNPDATA_ANLG_INPUT_READ()
DNPDATA_ANLG_INPUT_CHANGED()
DNPDATA_ANLG_OUTPUT_READ()
```

Added examples of reporting database status conditions to the database simulation in **DNPsim2.c**. To illustrate that the flags are an optional part of the database interface, the simulated database in **DNPsim1.c** ignores **pFlags** parameters completely.

Also, database status flags are more completely supported in binary input change event and analog input change event processing.

- dnp3** Freeze, write, and assign-class operations which fail no longer assert IIN2-0 (bad function) and cease processing the request; now failed freeze, write, and assign-class operations assert IIN2-2 and continue processing the rest of the request.



Added the following database macros:

DNPDATA_SUPPORT_OVER_256_ANYOBJ	To include/exclude support for more than 256 points of any single object type
DNPDATA_SUPPORT_OBJ_1_Vx	To include/exclude support for binary input (static) variations
DNPDATA_SUPPORT_OBJ_2_Vx	To include/exclude support for binary input change event variations
DNPDATA_SUPPORT_OBJ_10_V2	To include/exclude support for binary output statuses (reads of binary outputs)

DNPDATA_SUPPORT_OBJ_12_V1	To include/exclude support for include/exclude relay output blocks (binary outputs)
DNPDATA_SUPPORT_OBJ_20_Vx	To include/exclude support for running counter variations
DNPDATA_SUPPORT_OBJ_21_Vx	To include/exclude support for frozen counter variations
DNPDATA_SUPPORT_OBJ_32_Vx	To include/exclude support for analog input change event variations
DNPDATA_SUPPORT_FREEZE	To include/exclude support for freezing counters
DNPDATA_SUPPORT_FREEZE_CLEAR	To include/exclude support for freezing and clearing counters
DNPDATA_CNTR_TYPE	To control the native counter type. While 32-bit counters are now supported, setting the native counter type to TMWDEFS_USHORT (16-bit) and excluding support for 32-bit counters may save code space under some compilers.
DNPDATA_BIN_OUPUT_SELECT()	To allow application-specific action upon select (not execute) operation for two-pass (SBO) controls.
DNPDATA_ANLG_OUTPUT_SELECT()	To allow application-specific action upon select (not execute) operation for two-pass (SBO) controls.
DNPDATA_DFLT_OBJ_01_VAR()	To configure the binary input (static) variation reported when variation 0 is requested
DNPDATA_DFLT_OBJ_20_VAR()	To configure the running counter variation reported when variation 0 is requested
DNPDATA_DFLT_OBJ_21_VAR()	To configure the frozen counter variation reported when variation 0 is requested



If no support is included for binary or analog input change events, the **DNPPrbe** module may now be totally excluded from the compiled image.



Changed or removed the return type from many **DNPDATA** macros -- many now return **TMWDEFS_BOOL** to indicate success or failure. Other macros, which previously indicated status conditions through the return value, now use the **pFlags** parameter to provide status information and now do not have a return type (they are void). In summary, the **TMWDEFS_DBSTAT** type is no longer used by this module, and, in fact, is no longer defined.



Renamed the symbols listed in the following table. Most of these symbols were not normally used by application-specific software.

Old	New
DNPDEFS_OBJ_STATUS_XXX	DNPDEFS_DBAS_FLAG_XXX and added new values.
DNPDEFS_OBJGRP_XXXX	DNPDEFS_OBJ_nn_XXXX where nn represents the object group number.
DNPDEFS_OBJGRP_CNTR_CHNG_EVENTS	DNPDEFS_OBJ_22_CNTR_EVENTS.
DNPDEFS_OBJGRP_VTERM_EVENT	DNPDEFS_OBJ_23_VTERM_EVENTS
DNPDEFS_OBJGRP_STRING_EVENT	DNPDEFS_OBJ_111_STRING_EVENTS
DNPDEFS_OBJGRP_STRING_OUTPUT	DNPDEFS_OBJ_110_STRING_DATA
DNPDATA_VTERM_PROCESS_DATA()	DNPDATA_VTERM_PROCESS_RCVD()
TMWDEFS_DBSTAT_XXXX	DNPDBAS_STATUS_XXXX
dnprbe_o2StoreChangeEvent()	dnprbe_binStoreEvent()
dnprbe_o32StoreChangeEvent()	dnprbe_anlgStoreEvent().
dnprbe_getEvent01()	dnprbe_binGetEvent()
dnprbe_getEvent032()	dnprbe_anlgGetEvent()



Reduced code space requirements by adding conditional compilation, rearranging code, and removing unnecessary pointer de-references. Even without removing support for DNP features, these modifications can result in savings of several kilobytes under some compilers.



Prevented some warnings from some compilers by adding conditional compilation, breaking-up complicated expressions, adding temporary variables, rearranging code, etc.



To the example database simulations in **DNPsim1.c** and **DNPsim2.c**, added remote terminal interface commands (part of the optional diagnostic interface) for configuring the new default variations for object 1 (binary inputs), 20 (running counters), and 21 (frozen counters) because now more than one variation of each type is supported.



Made the command line parameter to the remote terminal interface commands (part of the optional diagnostic interface) a pointer to a const string (to prevent coding unintentional writes).



Now conditionally calls **dnpvterm_init()** early within **dnpdvrs_initDNP()**. This removed the need for an initialized static variable which is a problem for some compiler environments. (Initialized static variables still exist in the optional **DNPpkbin.c** and **DNPinln.c** modules.)



Revised and enhanced comments throughout.



Corrected text label displayed by the **dnpcmd_displayEventStatus()** (part of the optional diagnostic remote terminal interface).



In **DNPsim1.c**, corrected a problem with simulation of pulse-on binary output controls. The problem was that pulse-on's were treated as latch-off's. This problem was introduced in V2.14.



In **DNPprbe.c**, corrected problem with returning the correct amount of time until the next unsolicited message is ready. Some implementations may have used this time (because it may have eventually been passed to **DNPTARG_COMM_RECEIVE()**) to put the DNP task to "sleep" in a real-time operating system environment. If so, because the time was reported incorrectly, unsolicited messages may not have been transmitted until some other DNP action (such as scanning for changes) would have caused the DNP task to be resumed. This problem was introduced on Version 2.13.



In **DNPpkbin.c** (optionally used for fast scanning of binary inputs), corrected a problem that could have caused **DNPDATA_BIN_INPUT_EVENT_CLASS(pointNum)** to be called with an invalid point number (**0xff** or **0xffff**). Also, **DNPpkbin.c** now supports point numbers higher than 255.

Version 2.15 (August 16, 1999):

dnp3 Event buffer overflow was added to the conditions that can cause an unsolicited response. Also, when a buffer overflow occurs, the class count for the oldest event, which is thrown away, is now decremented.



Within the optional **dnpcmd.c** module, added **const** to **linkCnfmStrings** to prevent it from being compiled as an initialized data variable (it should go in **const/ROM** sections instead).



Within **DNPsim2.c**, added more conditional compile directives to withdraw virtual terminal diagnostics if virtual terminal objects are not supported.



Fixed problem that cleared restart IIN upon application layer confirmation of a message that had the bit asserted. The restart bit must be cleared explicitly by the master. The problem was introduced in version 2.13.



Rearranged first three lines of **dnpdbe_unsolReady()**; in pure 'C', of course, all variables must be declared at the top of each block. The problem was introduced in version 2.13.

Version 2.14 (August 10, 1999):

dnp3 Now ensures that whenever a class indicates an unsolicited message should be sent, that at least that class is sent. In other words, for each class index of **DNPCONFG_UNSol_SEND_MASK()**, we now internally force the bit for the corresponding class to be on.

dnp3 Added software to ensure that in order to indicate that an unsolicited response should be sent for a particular class of event data, the number of events in that class must be more than zero. This means that a value of 0 for **DNPCONFG_UNSol_MIN_EVENTS()** is effectively the same as a value of 1.



Modified **SDNPwin C-I Guide.rtf** and added many embedded comments to aide customization of the document for implementations of the DNP3 Source Code Library 97.



Moved `DNPCONFG_APPL_NEED_TIME_DLY` to `DNPtarg.c`, and renamed it `DNPTARG_APPL_NEED_TIME_DLY`. This was done because this is a target hardware specific configuration parameter, and is very closely related to `DNPTARG_SET_SYSTEM_DATE_TIME()`.



Changed default value for `DNPCONFG_APPL_REQ_BUFFER_SIZE` to 249, which should save over 1.5k of RAM for memory-constrained devices, and should be adequate for normal DNP messaging.



Simplified the `DNPsim2` simulated database and modified the `DNPsim1` simulated database to make the documentation less complicated.



Made minor enhancements to diagnostic displays.



Enhanced the functionality of the optional terminal interface commands for configuration parameters. Default values are now displayed with each parameter. Display order is now more easily controlled. Combined some `TMWPARAM` macros to make `TMWparam.h` less complicated. Added new `displaySize` remote terminal interface command.



Added `DNPTARG_PARAM_DESCRIPTOR()` to allow target-hardware-specific configuration parameters to be viewed or set through the optional remote terminal interface.



Corrected problem introduced in Version 2.13 that cause communications to stop when a link-layer `NACK` with `DFC=0` was received from the DNP Master station.



Corrected problem introduced in Version 2.13 with the assign class function: Now, to remove a object type from all event classes, the static object type header must be preceded by object 60, variation 1 (it was previously 0).



Corrected problem introduced in Version 2.06 related to responding to master requests that request application confirmation. The problem was that for the application confirmation being returned, the length of the data link frame was 1 byte too short, causing a CRC error. It is rare and usually a mistake if master requests also contain requests for application confirmations; this is because the response to the request is basically a confirmation. With a request for confirmation, the slave will have to respond with two messages: the confirmation and the response to the request.

Version 2.13 (July 23, 1999):

dnp3 Conforming to the DNP3-1999 specification, changed data link behavior regarding link function codes and the FCV bit: Now, received data link frames using function codes 0 (reset link), 4 (unconfirmed user data), and 9 (link status request) are rejected if the FCV bit is set (function code 1 (reset user process) is not supported by the Source Code Library). Also, received data link frames using function codes 2 (test link) and 3 (confirmed user data) are rejected if the FCV bit is clear. When rejecting a frame, the Source Code Library now responds with a data link frame

using function code 15 (link service not implemented). This meets the "****" requirements in the DNP3 Slave Test Procedures, Version 1.00, section 6.6.2.5.

dnp3 When a link layer frame is received that contains a request for a link layer confirmation, and if the link has not previously been reset, the Source Code Library will now respond by sending **NACK** with **DFC=0**. This is also conformant to the DNP3-1999 specification, and pertains to the DNP3 Slave Test Procedures, Version 1.00, section 6.1.2.

dnp3 When a data link frame is transmitted with request for confirmation, and if, in response, a **NACK** with **DFC=0** is received, the Source Code Library now requests a link reset. Then, upon receiving an **ACK** to the link reset, the Source Code Library now rebuilds and resends the original frame.

dnp3 Corrected status flags for analog inputs and analog output statuses, as recommended by the DNP Technical Committee: Now, if an analog input or output status is over-range, it must not clear the on-line flag.

dnp3 Conforming to the DNP3-1999 specification, response message fragments which are not large enough to contain all requested events no longer set buffer overflow IIN. The class IINs indicate the presence of additional event data.

dnp3 Conforming to the DNP3-1999 specification, added support for persistent “buffer overflow” and “all stations” Internal Indication bits (IINs). Now, when either the buffer overflow or all-stations IIN is contained in a message, the message will also contain a request for confirmation. These IINs will persist in all subsequent messages until the confirmation is received.

dnp3 Added support for **DNP_CFG_UNSOL_RETRY_DELAY()**, an unsolicited message retry timer used to limit when unsolicited responses can be transmitted. This parameter is used in conjunction with the application layer confirmation timeout: While waiting for confirmation of an unsolicited message, the Source Code Library has always buffered (not responded to) read function requests from the DNP Master. However, once the unsolicited message was confirmed, or once the confirmation timeout had expired, and once any buffered read request was responded to, it was possible for another unsolicited message to be immediately transmitted. Now, if the unsolicited message is not confirmed, it is not possible for another unsolicited message to be transmitted until the time specified by **DNP_CFG_UNSOL_RETRY_DELAY()** expires.



Replaced **DNP_CFG_DIAG_STRINGS_ENABLED()** and **DNP_CFG_DIAG_COMMANDS_ENABLED()** with **DNP_CFG_DIAG_FEATURE_MASK()**, a single configuration parameter used to control conditional compilation of multiple diagnostic features. Added **TMWDEFS_DIAG_FEATURE_STATISTICS**, (one of the bits that can be included in **DNP_CFG_DIAG_FEATURE_MASK()**) to control conditional compilation of diagnostic statistic counters.



Renamed `dnpphys_errors` to `dnpphys_statistics`, and added many new diagnostic counters, including successful frame transmissions and receptions, and successful fragment transmissions and receptions.



Changed the data type for diagnostic statistic counters to be `DNPCONFG_DIAG_STATISTIC_COUNTER`. This allows the implementer to tailor the diagnostic statistic counting to the target application. Possible values for this type are `unsigned char`, `unsigned short`, or `unsigned long`.

Version 2.12 (July 21, 1999):

dnp3 Added support for assign class function (a Subset Level 3 feature).

dnp3 Added support for several new configuration parameters which control the unsolicited messaging behavior. Now, there is a separate notification timer (`DNPCONFG_UNSol_NOTIFY_DELAY(classIndex)`) for each class of events. Also when conditions for a specific class indicate that an unsolicited message should be transmitted, a parameter (`DNPCONFG_UNSol_SEND_MASK(classIndex)`) has been added to configure which classes of events should be included in the unsolicited message. To be consistent with this parameter scheme, the `DNPCONFG_UNSol_MIN_CLASS_n()` configuration parameter has been changed to `DNPCONFG_UNSol_MIN_EVENTS(classIndex)`.

dnp3 Unsolicited notification delay timers continue to run even if unsolicited responses have been disabled for a particular class of events.

dnp3 Conforming to the DNP3-1999 specification, read function requests are now allowed while waiting for confirmation of the initial null unsolicited response (the initial null unsolicited response is transmitted upon power-up and contains no data; its purpose is merely to inform the DNP Master that DNP communications have been initialized). By allowing read responses, the device will respond to normal data polls even if the destination address of the unsolicited response is mis-configured.

dnp3 Conforming to the DNP3-1999 specification, if `DNPCONFG_UNSol_PERMITTED()` is `TMWDEFS_FALSE`, the enable and disable application function codes are rejected; the Source Code Library responds with the “bad function” internal indication bit asserted.



Changed name of `DNPCONFG_UNSol_ENABLED()` to `DNPCONFG_UNSol_PERMITTED()` to avoid confusion with the application layer unsolicited enable/disable function codes. Now `DNPCONFG_UNSol_SUPPORTED()` controls compile-time support for unsolicited messaging, and `DNPCONFG_UNSol_PERMITTED()` controls run-time support for unsolicited messaging.



Support for unsolicited response messaging can now be conditionally excluded from compilation using the `DNPCONFG_UNSol_SUPPORTED()` configuration parameter.

By setting this parameter to **TMWDEFS_FALSE**, the code-space requirements should be reduced by approximately 1.5 kbytes.



No longer use structure tags that conflict in name with their corresponding typedef names. Some compilers do not separate these two name-spaces.



Added two new parameter types to **TMWparam.c**: **TMWPARAM_TYPE_ARRAY_USHORT** and **TMWPARAM_TYPE_ARRAY_ULONG**. These are used to create (optional) remote terminal interface structures for user input of configuration parameters that are arrays; e.g., unsolicited notification delay is an array of 3 – one index per event class.



Minor cosmetic changes and code space savings.



Moved many symbol definitions from individual header files to **TMWDEFS.h**. This was done to consolidate all symbols which are part of the Target Application Interface, and to prevent any circular include problems. Now implementers only need to **#include “tmwdefs.h”** to resolve these symbols.

The following is a list of new symbols and symbol name changes.

Old	New
(didn't exist previously)	DNPONCFG_UNSQL_SUPPORTED
DNPONCFG_UNSQL_ENABLED	DNPONCFG_UNSQL_PERMITTED
DNPONCFG_UNSQL_MIN_CLASS_1	DNPONCFG_UNSQL_MIN_EVENTS(0)
DNPONCFG_UNSQL_MIN_CLASS_2	DNPONCFG_UNSQL_MIN_EVENTS(1)
DNPONCFG_UNSQL_MIN_CLASS_3	DNPONCFG_UNSQL_MIN_EVENTS(2)
DNPONCFG_UNSQL_NOTIFY_DELAY	DNPONCFG_UNSQL_NOTIFY_DELAY(classIndex)
(didn't exist previously)	DNPONCFG_UNSQL_SEND_MASK(classIndex)
DNPVRS_COI	TMWDEFS_COI
DNPVRS_COI_COLD_RESTART	TMWDEFS_COI_POWER_ON
DNPVRS_COI_WARM_RESTART	TMWDEFS_COI_WARM_RESTART
DNPLINK_CNFM	TMWDEFS_LINKCNFM
DNPLINK_CNFM_NEVER	TMWDEFS_LINKCNFM_NEVER
DNPLINK_CNFM_SOMETIMES	TMWDEFS_LINKCNFM_SOMETIMES
DNPLINK_CNFM_ALWAYS	TMWDEFS_LINKCNFM_ALWAYS
DNPDBAS_STATUS	TMWDEFS_DBSTAT
DNPDBAS_STATUS_OFF_LINE	TMWDEFS_DBSTAT_OFF_LINE_NO_DATA
DNPDBAS_STATUS_NO_EVENTS	TMWDEFS_DBSTAT_OFF_LINE_NO_DATA
DNPDBAS_STATUS_OK	TMWDEFS_DBSTAT_ON_LINE_SUCCESS
DNPDBAS_STATUS_VALUE_OVERRANGE	TMWDEFS_DBSTAT_VALUE_OVERRANGE
DNPDBAS_STATUS_TOO_MANY_EVENTS	TMWDEFS_DBSTAT_MORE_DATA_EXISTS
DNPDBAS_STATUS_REQUEST_ERROR	TMWDEFS_DBSTAT_REQUEST_ERROR
DNPDBAS_STATUS_FILE_ID_OVFL	TMWDEFS_DBSTAT_MORE_DATA_EXISTS
DNPDBAS_EVENT_MODE_SOE	TMWDEFS_EVENT_MODE_SOE
DNPDBAS_EVENT_MODE_CURRENT	TMWDEFS_EVENT_MODE_CURRENT
DNPDBAS_EVENT_MODE	TMWDEFS_EVENT_MODE
DNPDBAS_CLASS_MASK	TMWDEFS_CLASS_MASK
DNPDBAS_CLASS_NONE	TMWDEFS_CLASS_MASK_NONE
DNPDBAS_CLASS_0	(no longer used)
DNPDBAS_CLASS_1	TMWDEFS_CLASS_MASK_1
DNPDBAS_CLASS_2	TMWDEFS_CLASS_MASK_2
DNPDBAS_CLASS_3	TMWDEFS_CLASS_MASK_3
DNPDBAS_CLASS_ALL_EVENTS	(no longer used)
DNPDBAS_OPERATION	TMWDEFS_BNCTL_MASK
DNPDBAS_OPERATION_NONE	TMWDEFS_BNCTL_MASK_NONE
DNPDBAS_OPERATION_PULSE_ON	TMWDEFS_BNCTL_MASK_PULSE_ON
DNPDBAS_OPERATION_PULSE_OFF	TMWDEFS_BNCTL_MASK_PULSE_OFF
DNPDBAS_OPERATION_LATCH_ON	TMWDEFS_BNCTL_MASK_LATCH_ON
DNPDBAS_OPERATION_LATCH_OFF	TMWDEFS_BNCTL_MASK_LATCH_OFF
DNPDBAS_OPERATION_REPEATED	TMWDEFS_BNCTL_MASK_REPEATED
DNPDBAS_OPERATION_UNPAIRED	TMWDEFS_BNCTL_MASK_UNPAIRED
DNPDBAS_OPERATION_PAIRED_TRIP	TMWDEFS_BNCTL_MASK_PAIRED_TRIP
DNPDBAS_OPERATION_PAIRED_CLOSE	TMWDEFS_BNCTL_MASK_PAIRED_CLOSE
DNPDBAS_BINARY_CONTROL	TMWDEFS_BNCTL_DESCRIPTOR
croboOperation	bnctlOperation
croboCount	bnctlCount
croboOnTimeMs	bnctlOnTimeMs
croboOffTimeMs	bnctlOffTimeMs
DNPDTIME_MS_SINCE_70	TMWDEFS_MS_SINCE_70
TMWSFTMR_MILLISECONDS	TMWDEFS_MILLISECONDS
TMWSFTMR_SECONDS	TMWDEFS_SECONDS
TMWSFTMR_MINUTES	TMWDEFS_MINUTES
TMWSFTMR_HOURS	TMWDEFS_HOURS
TMWSFTMR_DAYS	TMWDEFS_DAYS
DNPDIAG_ID	TMWDEFS_DIAGID

Old	New
DNPDIAG_ID_LINK	TMWDEFS_DIAGID_LINK
DNPDIAG_ID_TPORT	TMWDEFS_DIAGID_TPORT
DNPDIAG_ID_APPL	TMWDEFS_DIAGID_APPL
DNPDIAG_ID_USER	TMWDEFS_DIAGID_USER
DNPDIAG_ID_MMI	TMWDEFS_DIAGID_MMI
DNPDIAG_ID_STATIC_DATA	TMWDEFS_DIAGID_STATIC_DATA
DNPDIAG_ID_STATIC_HDRS	TMWDEFS_DIAGID_STATIC_HDRS
DNPDIAG_ID_COS_DATA	TMWDEFS_DIAGID_COS_DATA
DNPDIAG_ID_COS_HDRS	TMWDEFS_DIAGID_COS_HDRS
DNPDIAG_ID_SOE_DATA	TMWDEFS_DIAGID_SOE_DATA
DNPDIAG_ID_SOE_HDRS	TMWDEFS_DIAGID_SOE_HDRS
DNPDIAG_ID_TX	TMWDEFS_DIAGID_TX
DNPDIAG_ID_RX	TMWDEFS_DIAGID_RX
DNPDIAG_ID_ERROR	TMWDEFS_DIAGID_ERROR
DNPDIAG_ID_EXCLUDE_LAYERS	(no longer used)
<code>dnprbe_availableClasses()</code>	<code>dnprbe_availableClassIINS()</code>
(didn't exist previously)	DNPDATA_SUPPORT_ASSIGN_CLASS
(didn't exist previously)	DNPDATA_BIN_INPUT_ASSIGN_CLASS()
(didn't exist previously)	DNPDATA_ANLG_INPUT_ASSIGN_CLASS()

Version 2.11 (July 9, 1999):

dnp3 Added `DNPDATA_DFLT_OBJ_40_VAR()` to allow configuration of default variation for object 40 (analog output statuses).

dnp3 Added `pNativeTime` and `pTimeSet` to the `DNPSIM1_BIN_INPUT_CHANGED()` and `DNPDATA_ANLG_INPUT_CHANGED()` macros to allow the Target Application to transfer knowledge of when changes occur.

dnp3 Added conditional compile parameters to control support for variations of analog inputs and outputs. This allows code space requirements to be reduced when some or all variations of analog inputs and outputs are not implemented.



Significantly revised the structure and content of comments surrounding macros in `DNPconfig.h` and `DNPdata.h`.



Enhanced conditional compile statements for the features listed below. For each feature, header files are no longer `#include`'d if support is not configured. This allows header files and c-files for non-required features to be completely excluded from the Target Application compile/link environment. See the Optional Utility Files section of the Implementers Guide for more information.

- Support for virtual terminal. (If `DNPDATA_SUPPORT_OBJ_112_113` is `TMWDEFS_FALSE`, then the `DNPvterm` module can be excluded.)
- Support for fast packed binary input scanning. (If `DNPDATA_PACKED_BIN_NUM_PACKS` is zero, then the `DNPpkbin` module can be excluded.)
- Support for diagnostic strings and commands. (If `DNPCONFIG_DIAG_STRINGS_ENABLED()` and `DNPCONFIG_DIAG_COMMANDS_ENABLED()` are `TMWDEFS_FALSE`, then many modules can be excluded; e.g., `TMWprntf` and/or `DNPcmd`.)

- Support for the in-line diagnostic terminal. (If `DNP_CFG_DIAG_INLN_BUFFER_SIZE()` is zero, then the `DNP_in1n` module can be excluded.
- Divided the `DNPdiag` module into separately optional `DNP_in1n` and `DNPcmd` modules for clarity, and so that they can be independently excluded if necessary.
 - Changed default value for `DNP_CFG_DIAG_STRINGS_ENABLED()` to `TMWDEFS_FALSE` because the most typical scenario is to save code-space by removing diagnostics and using a test Master station to perform protocol diagnostics
 - Changed the as-shipped values for `DNP_CFG_RBE_BIN_BUFFER_SIZE()` and `DNP_CFG_RBE_ANALG_BUFFER_SIZE()` to hold 10 binary events and 10 analog events, respectively. This makes it easier to test buffer overflow conditions. However, these sizes are typically too small for normal implementations.
 - Changed the default value for `DNP_CFG_SBO_BUFFER_SIZE()` after re-computing the largest possible select object header.
 - Renamed (and moved) `DNP_CFG_STORExx()`, `DNP_CFG_GETxx()`, and `DNP_CFG_UNUSED_PARAMETER()` to `DNPTARG_*` because they are target (hardware and compiler) dependent.
 - Changed the name of `DNPTARG_COMM_INIT()` to `DNPTARG_COMM_OPEN()` to be consistent with other TMW products.
 - Changed `DNPTARG_COMM_OPEN()`, `DNPDATA_INIT()`, and `dnpdvrs_initDNP()` to return a Boolean to indicate initialization success.
 - Created `TMWdtime.c` and `TMWdtime.h`. Replaced the `DNPDTIME_DATE_TIME` type with `TMWDTIME`, a more general date/time structure that is shared with other TMW products. More specifically, the `seconds` and `milliseconds` fields were combined into a single `msecsAndSecs` field. This allowed a more efficient modification to be made to the conversion to/from DNP-time (6 byte time) within `DNPdtime.c`.
 - Moved non-DNP-specific date/time functions from the `DNPdtime` module to the `TMWdtime` module so that other TMW products can share them.
 - In `TMWdefs.h`, added `TMWDEFS_SFLOAT`, and changed all the “float” type definitions within `DNPdbas.c` to use `TMWDEFS_SFLOAT`.
 - Because of an internal change within the optional `TMWparam` module, which is used to present configuration parameters as run-time changeable parameters through the optional remote terminal command interface, many `DNP_CFG_xxx` and `DNPDATA_xxx` configuration parameter macros were changed to add parenthesis.
 - Revised many diagnostic display strings to increase understanding. Added `#define DNPDIAG_SDNPLIB_xxxx` constants for the product name and version, and added `#define TMWDEFS_COMPANY_x` constants for company information. These constants may be used by the Target Application.



Changed variable names and enhanced comments within the **DNPpkbin** module to help the understanding of this module. Also corrected some minor problems in this module (listed below).



To save code space, removed **tmwintel_store24()**, **tmwintel_get24()**, **tmwmtr1a_store24()**, and **tmwmtr1a_get24()** from the **tmwintel** and **tmwmtr1a** modules. These functions are used by other TMW products, such as the IEC 870-5 Source Code Libraries, but are not used by DNP Source Code Libraries.



Added type-casts and variable qualifiers such as **static** or **const** to help prevent compiler warning for some embedded cross-compilers. For the same reasons, removed use of keywords such as **"data"** as macro parameter names.



Continued revision of source code comments to enhance or more accurately document source code files.



Fixed a problem that caused the Source Code Library to “hang” if **DNPDATA_GET_CLASS_0_HEADERS()** was implemented with a true semaphore (as exemplified in **DNPsim2**), and if a class 0 request was received. The problem was that **DNPDATA_GET_CLASS_0_HEADERS()** was called twice without an intervening **DNPDATA_UNLOCK_CLASS_0_HEADERS()**. The problem was fixed by adding **pMultiFragLastByte** to the **DNPDATA_TYPE** structure, and preserving the end of request messages through multi-fragment response calls of **dnpdpa_run()**; this relieves having to call **DNPDATA_GET_CLASS_0_HEADERS()** twice.



In **DNPSIM2.h**, added **DNPSIM2_SUPPORT_OBJ_xx()** macros that previously existed only in **DNPSim1.h** and in **DNPdata.h**, but not yet in this example implementation.



In the example database implementations provided in **DNPSim1.c** and **DNPSim2.c**, corrected references to the number of analog inputs and binary inputs. In one place within **DNPSim1.c**, the number of binary inputs was referenced instead of the number of analog inputs. In one place within **DNPSim2.c**, the number of analog inputs was referenced instead of the number of binary inputs.



In **DNPpkbin.c**, an optional module that provides support for fast scanning of packed binary inputs, the initial values of packed binary inputs are set by calling **DNPDATA_PACKED_BIN_PACK_READ()** rather than by clearing them to zero. In addition, change events are reported for the first snap shot processed. Previously, because initial values were set to zero, and because no change events were reported from the first snap shot, it was possible that near initialization time, consecutive static data polls of packed binary inputs could have reported two different values with no intervening change events.



In **DNPprbe.c**, **dnprbe_scan01()** is now prevented from double-scanning binary inputs that are members of packed binary “packages” and would normally be scanned by the **DNPpkbin** module.



In **TMWprntf.c**, an optional module used only to display diagnostic strings, corrected count of consumed **printf** buffer space used to prevent illegal memory overwrites. Previously, non-formatting characters were not counted, which means that it would

have been possible that some memory over-writes, which normally should not occur, would not have been detected through this means.

Version 2.10 (December 16, 1998):

dnp3 Changed inherent type for analog outputs from shorts to longs. This allows support for more analog output variations.



Added **DNPDATA_CLOSE()** and **dnpdvrs_closeDNP()**. This macro and function allow for such things as freeing dynamic memory allocated within **DNPDATA_INIT()** (the library itself never allocates dynamic memory). For similar reasons, a new parameter was added to initialization macros and entry points to provide the reason for initialization. The reason may be “warm restart” in which case dynamic memory would not need to be re-allocated.



Fixed problem that caused multi-fragment responses that contained static data only (no events) to incorrectly not request application layer confirmations. This problem was introduced in Version 2.00.

Version 2.08 (November 20 1998):



Enhanced the friendliness of the “set” and “show” remote diagnostic terminal commands. These commands are provided as part of the optional diagnostic remote terminal interface to allow displaying and/or changing configuration parameters.



Added additional source-code documentation (comments) to several **TMWxxxx** modules.



Fixed problem for response objects transmitted with qualifiers 17 and 28, and also for incorrect toggling of FCB bit in link layer. Both of these problems were introduced in Version 2.06.

Version 2.06 (November 4, 1998):



Added full support for three-sided "Triangle" interface between the Source Code Library and the Target Application. Now all calls from the Source Code Library to the Target Application are made through C-Macros defined in **DNPtarg.h**, **DNPcfg.h**, and **DNPdata.h**. The efficiencies resulting from this design are:

- conserves code space
- executes faster
- easier to understand
- quicker to install



Renamed **DNPbnchg** module to **DNPpkbin** to reflect the fact that it is an optional module, and is useful only when binary inputs are organized and accessible as packed bits within 8-bit, 16-bit, or 32-bit words. The application interface to this module is now entirely contained within **DNPdata.h**; therefore, this module should not need to be modified in order to be used.

Version 2.04 (September 14, 1998):

dnp3 Added floating point support – Object 30 (analog inputs) Variation 5 (short float) and Object 41 (analog output block) Variation 3 (short float).



Removed **DNP_CFG_APPL_CONFIRM_MODE** as a configuration parameter because the DNP3 Technical Committee recommended that all multifragment response messages request application confirmation (even if they contain only static data).

Version 2.02 (April 1, 1998):



Now uses new **DNPDEFS_CROB_CODE_XXXX** definitions instead of explicit literal values for object 12 – control relay output blocks (binary outputs). This aids understanding of the confusing specification for that object.



Fixed a problem that had trip and close values swapped for control relay output blocks. This problem was introduced in version 2.00.

Version 2.00 (January 10, 1998):

dnp3 Now uses method published in Technical Bulletin 9701-003 for responding to cold and warm restarts. Replaced **dpacoldRestart()** and **dpawarmRestart()** with a single **dpaRestart()** function which returns **DNPDPA_PARSE_STAT**. Created new **DNPDPA_RESTART_TYPE** to pass to the **dnpRestart()** function. Removed **coldRestartSeq** and **warmRestartSeq** from **DNPDPA_TYPE** structure definition.

dnp3 Now supports multi-fragment responses to file-id write function codes.

dnp3 Added a new configuration constant, **DNP_CFG_APPL_CONFIRM_MODE**, that allows application layer confirms to be requested only for message fragments containing event data, or also for non-final multi-fragment responses.

dnp3 Created **dnpdpa_buildClass0scan()** to allow the content of class 0 responses to be configured during run-time. This function should be called whenever class assignments are changed.

dnp3 Now starts notification delay timer for initial unsolicited message so that it does not transmit immediately at startup.



Created **DNPconfig.h** to consolidate all configuration parameters into a single file. Implementers merely need to change their chosen configuration parameters with implementation-specific values.



Created **DNPdata** module to remove point functions from the **DNPdbas** module, and to define a separation between protocol-specific and platform-specific database functionality. **DNPdbas** now contains only protocol-specific functions, and **DNPdata** contains platform-specific functions.



Created **DNPsim** module as an example implementation of the **DNPdata** module, containing only simulated database values.



Revised the **DNPdvr**s module to handle event-driven applications more smoothly. Created new **dnpdvrs_processDNP()** which should be called repeatedly as the main entry point into the DNP library. This function now handles receiving messages and transmitting messages, preventing implementers from also having to call **dnpdvrs_receivedChars()**, which no longer exists. Because of this new software architecture, the **dnpdvrs_signalDNP()** function has been removed.



Created **target.h**, which contains prototypes for platform-specific functions that must be implemented to install the Source Code Library.



dnpdvrs_startSerialTransmit() has been moved to **dnptask**, and **dnpdvrs_sendFirstChar()** which is called by **dnptask_startSerialTransmit()** has been renamed **dnpdvrs_transmitFrame()**.



dndnpdvrs_transmitFinished() no longer exists, but the functionality to determine when a message has been completely transmitted is still needed before a warm or cold restart can begin. This functionality is encapsulated in the application-specific **target_waitForTxFinished()** function.



dnpdvrs_doColdRestart() no longer exists, and is replaced by application-specific **target_coldRestart()** function, prototyped in **target.h**.



Made major modifications to the **DNP**rbe module:

Packed event structures (to conserve memory).

Moved some configuration parameters to the newly created **DNP**confg.h.

Added **dnprbe_periodicScanForChanges()** that replaces application-specific software for calling change event scanning routines. Instead, now their calling period is configured in **DNP**confg.h, and they are called automatically from within the **dnprbe_periodicScanForChanges()**, which is called from **dnpdvrs_processDNP()**.

Created **dnprbe_o2StoreChangeEvent()** and **dnprbe_o32StoreChangeEvent()** to allow external (Target Application) software to store change events detected through external means in the Source Code Library change event queues

Created **dnprbe_set030ChangeThresholds()** because thresholds for analog change events should be reset after a analog input point is reported in any way - either through a change event or as static data. This function is called by variation functions.

Changed the way analog input thresholds are computed and stored. More memory per input is required, but the execution time required to determine if an analog input has changed should be drastically decreased.

Changed the way report-only-most-recent mode of analog change reporting is handled. Now, in this mode, the value of the analog input is read when the event is

reported, not when it is detected. (Similarly, the timestamp, if reported, is when the value is read (at report time), not at detection time.) Consequently, there is no reason to call `dnprbe_set030ChangeThresholds()` at detection time, only at report time.



Created **DNPbnchg** module to provide fast binary scanning to detect binary change events with high time resolution. Because of its low-level nature, this module must be modified by implementers to gain access to application-specific data. `dnprbe_scan01()` detects if binary input points are being scanned for changes with the **DNPbnchg** module and therefore will not "double" scan them.



Created **DNPdtime** module to provide general date/time utility and conversion routines. Created `dnpdvrs_getDNPTIME()` and `dnpdvrs_setDNPTIME()` as the Target Application interface to date/time in DNP format. This functionality replaced `dnpdbas_highTimeIED` and `dnpdbas_lowTimeIED`.



Created **DNPdefs.h** to consolidate all protocol definitions in one file. Implementers should not edit this file.



Created **TMwsftmr** module to manage software countdown timers. Changed all software count down timers to use the type defined by this module, and moved their management to this module. A free-running 32-bit millisecond timer is assumed to be available and is read through `dnpdvrs_getMsTime()`. This timer could be based on a timer interrupt, or even on a free-running hardware timer. This functionality replaces `dnpdvrs_readMSecTimer()`, `dnpdvrs_writeMSecTimer()`, and `dnpdvrs_writeMSecTimer32()`. `dnpdvrs_timerISR()` no longer exists, and there is no longer a need to call any DNP library function with the same regularity (i.e., from a timer ISR), except that `dnpdvrs_processDNP()` should be called as often as possible.



Created **TMwbcd** module to be used as a general utility for implementers who need to do BCD conversion.



Added `dnpphys_getNeededChars()` function; it now returns the number of characters needed by the parse routines to complete the current frame, as well as the first character needed if starting a new frame. This can be used by powerful Target Application software such as real-time operating systems that may block the DNP process until the character(s) it needs are received. Also revised `dnpphys_parseDNPFrame()` to include inter-character delay timeout processing, and to allow processing multiple characters at once. Also, `dnpphys_parseDNPFrame()` now strips CRC bytes before providing the frame to upper layers of the protocol.



Create `dnpphys_restartFrame()` which basically replaced `PHYS_NEXT_FRAME()` macro. This function was made public so that the frame could be reset from the outside (e.g., when inter-character time gap is too long.)



Modified the **DNPlink** module to clean up memory usage, and to make some variable names more informative. No longer allow error counters to overflow; they stop counting at newly created `DNPPHYS_MAX_ERROR_COUNT`. Removed

LinkCheckAllCrcs() as that functionality is now performed in **dnpphys_parseDNPframe()**.



Added **DNPDPA_TX_FLAGS** type and definitions so that the type of content of messages can be transferred down to lower layers. (For application layer restart messages, the link layer needs to know to not request confirmations, and the physical layer needs to know to begin a cold or warm restart after the message has completely been transmitted.)



Changed **dpaIniToObjInfo()** to use new **DNPCONFIG_DFLT_VARIATION_OBJXX** configuration parameters to find variation to use when variation 0 is requested (instead of first variation in object definition table). Entries in the object definition table no longer need to be ordered with the default variation first



Created new **DNPDBAS_STATUS** that is now returned from all database functions. This replaced the functionality previously provided by **DNPDBAS_EVENT_STATUS** and **dnpdbas_getEventStatus()** which were removed.



Created new **DNPDBAS_CLASS_MASK** type to help ensure type safety throughout the Source Code Library.



The **numPoints** member of the **DNPDBAS_OBJ_ENTRY** structure was changed to **pNumPoints** and now points to the number of configured points per object. This allows the definition of points in a module (e.g., **dnpdata.c**, **dnpsim.c**) other than **dnpdbas.c**.



Added **pMaxResponse** as a parameter to the write variation functions in order to relieve the need for a **pMaxRespFrag** configuration parameter.



Removed **DNPDBAS_TABLE_TYPE** structure type and replaced it with types for individual object types in **dnpdata.h**.



Changed the name of any instance of "**data**" as a variable name or parameter name because the word is a keyword under some compilers.



Renamed **dnptask_timers()** to **dnptask_checkTimers()**, made it static (local to the **dnptask** module), called it from **dnptask_main()**, and modified it to check the new soft timers. Created **dnptask_getNextTimeout()** to return the remaining time on the soft timer that is nearest elapsing.



Fixed problem with not being able to clear Restart IIN bit that was introduced in 1.73.



Now reports an "object unknown" Internal Indication (IIN) error when variation 0 is requested for an object or function that doesn't support it.









Version 1.76 (June 11, 1997):



Fixed problem with remembering application sequence number used in unsolicited responses: Confirmation of unsolicited responses was near impossible because


sequence number was remembered incorrectly. This problem was introduced in V1.73

Version 1.74 (April 24, 1997):

-  Continued to revise all files in order to bring them up to date with corporate coding standard. This mainly consisted of revision comments and removing "prefixes" off of locally (static) defined functions and variables.
-  Renamed **DNPd1** module to **DNP1ink** module to be consistent with other TMW products.
-  Replaced **dnpdvrs_coldRestartObj []**, and **dnpdvrs_warmRestartObj []** (simulated messages) with **DNPDVRS_COLD_RESTART_DELAY** and **DNPDVRS_WARM_RESTART_DELAY**. (The messages are now created during run-time.)
-  Replaced examples of serial receive and transmit interrupt service routines with more general examples.
-  In **DNPDBAS_OBJ_ENTRY** table, now uses standard qualifier (**DNPDBAS_QUAL_STD_***) codes for all readable objects (e.g., inputs), writable objects (e.g., outputs), and event objects.
-  Restored FILE ID definitions to the **DNPdbas** header file.
-  Added some type casting to relieve some minor type casting warnings.
-  Corrected data link sequence numbering during multi-fragment responses (and during multiple single-fragment-multi-frame responses), by removing all instances of setting the **d1t.tPortSeq** equal to the **req.tPortSeq**. This effectively de-synchronizes the receive and transmit data link sequence number, but assures that transmitted data link sequence numbers are always sequential. Changed type of **multi fragNum** to **TMWDEFS_UCHAR** from **TMWDEFS_SHORT** in order to relieve some minor type casting warnings.

Version 1.73 (April 4, 1997):

dnp3 Now supports multifragment responses.

-  Revised all files in order to bring them up to date with corporate coding standard. In particular, a "module name prefix" has been pre-pended to all public symbols names, and all filenames have been changed to remove underscores. This standard was implemented in order to 1) increase readability and maintainability, 2) eliminate conflicts with other user-defined public symbols, and 3) provide a consistent look and feel for all software produced by the Triangle MicroWorks, Inc. development teams

Version 1.72 (February 10, 1997):

dnp3 Changed point number index variables for report-by-exception from **unsigned char** to **unsigned short** in order to accommodate more than 256 points.



Added conditional code for most-significant-byte-order target devices (e.g., Motorola 68000) to the following functions: **DNPdbas_store32()**, **DNPdbas_store16()**, **DNPdbas_get32()**, **DNPdbas_get16()**.

Version 1.70 (October 27, 1996):



Change **DNPDBAS_QUAL_EVENT_DATA** define from **0x0010** to **0x0040**.



Fixed problem with freeze function code which prevented processing request messages with multiple object headers.



Fixed problem with marking events in previously transmitted unsolicited responses as needing to be resent if a new request is received before the confirmation of the unsolicited response. Now, the events are only marked for resending if the new request uses the read function code.

Version 1.68 (October 23, 1996):



Fixed problem with the “disable unsolicited responses” function code request. If change events existed when a “disable unsolicited responses” request message was received, empty unsolicited response messages would be generated until an application layer confirm message was received. Fixed this by clearing notification delay when disable unsolicited response message received and making all unsolicited response messages require an application layer confirmation.

Version 1.66 (August 8, 1996):



As a diagnostic aide, added simulated counters to database. These counters increment for each change-of-state detected on corresponding binary inputs points.

Version 1.62 (February 15, 1996):



For efficiency, replaced reference to **pPhysRec** pointer with **newByte**.



Added typecast to constants for long variables to avoid warnings on some compilers.



Fixed parenthesis on select and operate store/compare loops.



Added sample routines for reading/writing analog output points.