TELETASK

MICROS+

DOMOTICS SYSTEM

MS WINDOWS
SOFTWARE LIBRARY

TDS LIBRARY V3.00.doc - 17/05/10

1 Table of contents

2 Introduction

2.1 TELETASK DOMOTIC SYSTEMS WINDOWS LIBRARY V3.1

Platform: To be used on all IBM compatible personal computers with Microsoft WINDOWS XP (sp 2), WINDOWS Server 2003 or WINDOWS VISTA operating system.

It is the purpose to generate a start-up environment for software developers, who are interested in generating their own PC-based control environment for the TELETASK domotics systems. In this way the developer can create his own user interface and connected solutions and services using a standard personal computer.

This situation should stimulate creative minds in working with their own control environment as an exclusive add-on to a TELETASK system. In this way the developer can generate solutions which are not available at TELETASK but only on his own to be commercialized platform.

This Library is to be used with the TELETASK DoIP central units.

2.2 LICENCE AGREEMENT.

The "TELETASK WINDOWS LIBRARY V3.1" consists of three parts:

- A 'dynamic link library' (DLL) which implements all necessary functions to communicate in an easy way with the "TELETASK central unit"
- Some examples on the use of the DLL
- The setup of the protocol to communicate with the "TELETASK central unit" without the use of the DLL.

The Licence agreement refers to all three parts of the "TELETASK WINDOWS LIBRARY V3.1"

To use the TDS LIBRARY V3.1 the TELETASK DoIP Open Protocol license (TDS15132) is needed.

TELETASK has the right to make any changes to this library at any point without any prior notice.

Due to the unlimited freedom of the developer, it is impossible for TELETASK to give free support or to carry any responsibility on the use and the results of the use of this libraries in any way.
Claims in any way for any reason are not taken into any consideration.
By using this libraries, the developer and the user agree with this statement.

3 DLL Functions & Events

3.1 Constructor

To create a new Instance of “Tds_Port”, two constructors are available

- `public Tds_Port(string SrvName, int SrvPort, int Tag)`
- `public Tds_Port(IPAddress Address, int SrvPort, int Tag)`

The two constructors can be used to create a “Tds_Port” for opening an Ethernet connection. As first parameter you can either pass a string (e.g. a domain) or an IP-address. The “SrvPort” parameter is the port number to which you want to connect. The port number used on the DoIP central unit for communication with the TELETASK Library is **55957**.
The “Tag” parameter is a general purpose integer you can use for a variety of reasons (e.g. To identify the connection if you build an application with multiple connections)

3.2 Opening and closing a connection

- `public bool Connect()`

  The Connect method is used to open the “Tds_Port”, if the connection was successfully established, or the connection was already opened “*true*” is returned, if connecting failed the return value is “*false*”.

- `public void Close()`

  The “Close” method closes the connection, it is advisable to call this method before you exit your program or if you do not intend to send or receive data for a longer period.

- `public void KeepAlive()`

  This function should be called regularly to keep the socket open when no other data is being sent. If no data is sent for 5 minutes the central unit will close the connection.

3.3 Setting a function

-
  ```
  public int FunctionSet(byte Cen, int Fnc, int Number, byte
                         Setting)
  ```

-
  ```
  public int FunctionSet(byte Cen, int Fnc, int Number, byte
                         Setting,   byte[] Data)
  ```

  Whit this functions you set the indicated function to a state that corresponds the setting and the data (if needed)

INPUT:

- Cen = the number (from 1 - 10) of the central unit of the output for: FNC_REL, FNC_DIM, FNC_ MOTORFNC, FNC_SENSOR, FNC_AUDIO. For all other functions the Cen parameter must be 0.
- Fnc: One of the constants starting with 'FNC_'
- Number: The number of the outputs.
- For most functions the "Data" parameters are not used. The setting parameter can be SET_ON, SET_OFF or SET_TOGGLE.
- For dimmers the setting parameter needs to be 0-100 indicating the level to which you want to set the dimmer.

- For function Regime Number the Regime you want to activate (Auto, Workday, Weekend, Simulation, None, Custom). All regimes, except "Custom" need to have the Setting SET_ON, "Custom" can also have the setting SET_OFF.
- For the motors there are no "Data" parameters, The setting can be: "SET_MTRUP", "SET_MTRDOWN", "SET_MTRSTOP", "SET_MTRSTARTSTOP", SET_MTRUPSTOP", "SET_MTRDOWNSTOP" and "SET_MTRUPDOWN.
- For FNC_AUDIO there are no "Data" parameters, The setting can be: SET_AUDIOUP, ..., SET_AUDIOKEY9 (see Constants for the full list of audio actions).
- For FNC_TPKEY number is the input number, Setting can be SET_PULSE, SET_CLOSED or SET_OPENED.
- For the FNC_SENSOR the number of "Data" parameters depends on the setting, the possibilities are:

Setting	Data Parameters
SET_ TEMPUP	/
SET_ TEMPDOWN	/
SET_ TEMPFROST	/
SET_ TEMPNIGHT	/
SET_ TEMPSTANDBY	/
SET_ TEMPDAY	/
SET_ TEMPSETNIGHT	1: The new night preset
SET_ TEMPSETSTANDBY	1: The new standby preset
SET_ TEMPSETDAY	1: The new day preset
SET_ TEMPMODE	/ (this is used to scroll through the "modes")
SET_ TEMPAUTO	/
SET_ TEMPHEAT	/
SET_ TEMPCOOL	/
SET_ TEMPVENT	/ (only for Airzone sensors)
SET_ TEMPSTOP	/ (only for Airzone sensors)
SET_ TEMPHEATP	/ (only for Airzone sensors)
SET_ TEMPSPEED	/ (this is used to scroll through the speeds)
SET_ TEMPSPLOW	/
SET_ TEMPSPMED	/
SET_ TEMPSPHIGH	/
SET_ TEMPSPAUTO	/
SET_TEMPON	/
SET_TEMPOFF	/
SET_TEMPONOFF	/

Note that the parameters for night, standby an day need to be recalculated to a "TDS" value, see the conversion functions.

- Example: If you set cen = 1, Fnc=FNC_RELAY, Number=3, State=255 the central unit will set the Relay with number 3, on Central Unit 1, to ON.

OUTPUT: (Return value)

- 0 = Message successfully transmitted
- -1 = communication port not opened
- -2 = No Answer

3.4 Getting a function

- `public int FunctionGet(int Cen, int Fnc, int Number)`

This function asks for the state of the indicated function. As a result of this function an Event will occur (regardless if the LOG channel for the function is open or not).

INPUT:
- Cen: see FunctionSet
- Fnc: see FunctionSet (+ FNC_COND, except FNC_TPKEY)
- Number: see FunctionSet.

OUTPUT: see FunctionSet.

3.5 Logging a function

- `public int FunctionLog(int Fnc, int Setting)`

This function will open/close a channel for the function
Example: If you call this function with the parameter Fnc=FNC_RELAY and State= SET_ON, all changes on relays will occur as 'event'! In case you set State=SET_OFF no more events will occur from relays.
After a PROSOFT download, all channels will be closed.

INPUT:
- Fnc: see FunctionSet (+ FNC_COND, except FNC_TPKEY)
- Number: see FunctionSet.

OUTPUT: see FunctionSet.

3.6 Dynamic messages

- `public int WriteDisplayMessage(int Cen, int[] BusNr, int[] Address, bool Message, bool Ascii, String Line1, String Line2, int Beeps)`

This function can be used to create "*Dynamic messages*".
Example: If you call this functions with the parameters Cen = 1 BusNr = {1, 2}, Adres = {11, 7 }, Message = false, Ascii = true , two lines of text and Beeps = 10. An ALARM will appear on the interfaces of central unit 1 with addresses BusNr = 1, Adres = 11 & BusNr = 2, Adres = 7, displaying the two lines of text, and the interfaces will beep 10 times.
To create a Message/Alarm on all interfaces: set BusNr = {0}, Adres = {0}

INPUT:
- BusNr, Adres: Two arrays of the same size that contain the bus number(s) and interface address(es) of the interface(s) on which to display the Message/Alarm
- Message: true for a "message", false for an "alarm".
- Ascii: true for a text containing only ascii characters, false for text containing non ascii characters. (text with non ascii characters needs to be in the unicode "UTF-16" format. Unicode text can only have 8 characters)
- Line1, Line2 The two lines of text for the message limited to 16 ascii characters or 8 unicode characters for each line.
- Beeps, the number of times the interface(s), on which the message is displayed, should beep.

OUTPUT: see FunctionSet.

3.7 Sensor conversion functions

- `public double ConvertSensValToTemp(int val)`

  This function converts an integer value (derived from the event 'Report' with Fnc= FNC_SENSOR) to degrees Celsius. This is needed for Day an Night values, for standby values, you only have to divide by 10 (because standby is expressed as a difference from the Day/Night value)

- `public int ConvertSensValToHum(int val)`

  This function converts an integer value (derived from the event 'Report' with met Fnc= FNC_SENSOR) to % humidity.

- `public double ConvertSensValToLux(int val)`

  This function converts an integer value (derived from the event 'Report' with Fnc= FNC_SENSOR to a lux value.

- `public int ConvertTempToSensVal(double val)`

  This function converts degrees Celsius to an integer. Required for the Method 'FunctionSet' with Fnc= FNC_SENSOR and Action = SET_TEMPSETDAY or SET_TEMPSETNIGHT. For FunctionSet with Fnc = FNC_SENSOR and Action = SET_TEMPSETSTANDBY the temperature value only needs to be multiplied by 10).

- `public int ConvertHumToSensVal(double val)`

  This function converts % humidity to an integer. Required for the Method 'FunctionSet' with Fnc= FNC_SENSOR and Action = SET_TEMPSETDAY or SET_TEMPSETNIGHT.

- `public int ConvertLuxToSensVal(double val)`

  This function converts a lux value to an integer. Required for the Method 'FunctionSet' with Fnc= FNC_SENSOR and Action = SET_TEMPSETDAY or SET_TEMPSETNIGHT.

3.8 Miscellaneous

- `public String GetPortName()`

  This function will return the name of the "Tds_Port" (the name or the IP-address)

- `public int GetTag()`

  This will return the "Tag" associated with the "Tds_Port".

3.9 Events

-
  ```
  public delegate void ReportEventHandler(Tds_Port sender, int Cen,
  int Fnc, int Number, int[] Parameters);
  ```

  This event occurs when something changes in the central unit (on condition that the log channel is open), or as result of a "FunctionGet".

- For most functions there is only one extra parameter, this parameter indicates the state of the function. This can be SET_ON, SET_OFF, a percentage from 0 to 100% (for dimmers), or the active source for an audio system (or SET_AUDIOOFF if the audiozone is OFF).
- For a motor there are two bytes to indicate the state, the first one indicates the direction, the second one the power.
- For the sensors the parameters are (note that sensor values need to be recalculated to °C, % hum or Lux value. Also note that the Value, Target, Day and Night are 2 bytes long, a value >= SENS_MAX indicate an error)

Value	The value measured by the sensor (2 bytes)
Target	The Target for the sensorzone (2bytes)
Day	The value of the Day preset (2bytes)
Night	The value of the Night preset (2bytes)
Standby	The standby value (zero for light and humidity sensor)
Active Preset	SET_TEMPDAY, SET_TEMPNIGHT, SET_TEMPSTANDBY or SET_OFF (if there is no active preset)
Mode	The mode for the sensor (zero for light and humidity sensor)
SpeedMode	The speedmode for the sensor (zero for light and humidity sensor)
Power	The power
Frost Protection	The frost protection (zero for light and humidity sensor)
NOT USED	NOT USED
Diretcion	Direction (currently only used for daikin).

4 BUILD-UP OF THE EXAMPLE PROGRAMS

The example programs are based on the following parts:

- The Windows Custom Control TELETASK.DLL wherein the control functions for the TELETASK systems are included.(TDSLIBRARY.DLL is a 32-bit Dynamic link library.)
- A graphical user interface which uses the available control functions on a C# 2.0 environment.

These parts are usable on personal computers which runs on Microsoft WINDOWS XP (sp 2), WINDOWS Server 2003 or WINDOWS Vista operating system.

The included examples are build on the Microsoft .Net framework 2.0. This framework is included on your TELETASK V3.00 CD-ROM and if necessary it will be installed when installing PROSOFT.

The program sources are included on your CD-ROM and have to be opened in Visual Studio 2005.

4.1 Basic example

This example allows to connect to one TELETASK central unit. With this program you can SET and GET a function and open and close a LOG channel.
The received data from the TELETASK central unit is displayed as a series of bytes.

4.2 Dynamic message Example

This demo allows you to connect up to two MICROS+ central units to activate a dynamic message on one or two central units.

The Socket information (IP-address and port number) for each of the Ethernet connections can be set in the top part of the Form.
In the Centre of the Form you can enter the AUTOBUS address of the interface on which you want to display the message and the two lines of text for the message.
At the bottom there are four buttons, the first sends the message to MICROS+ A, the third button does the same but to MICROS+ B and the middle button will simultaneously handle the writing of the message on both the MICROS+ central unit (the two central units need to have display interface on the same address).

5 Constants

The functions in the DLL use a parameter *“Fnc”* and can have following values

FNC_RELAY = 1	(control or get the status of a relay)
FNC_DIMMER = 2	(control or get the status of a dimmer)
FNC_MOTORFNC = 6	(control or get the status of a Motor)
FNC_LOCMOOD = 8	(control or get the status of a Local Mood)
FNC_TIMEDMOOD = 9	(control or get the status of a Timed Local Mood)
FNC_GENMOOD = 10	(control or get the status of a General Mood)
FNC_FLAG = 15	(control or get the status of a Flag)
FNC_SENSOR = 20	(control or get the status of a Sensor zone)
FNC_AUDIO = 31	(control or get the status of a Audio zone)
FNC_PROCES = 3	(control or get the status of a Process function)
FNC_REGIME = 14	(control or get the status of a Regime function)
FNC_SERVICE = 53	(control or get the status of a Service function)
FNC_MESSAGE = 54	(control or get the status of a Messages or Alarms)
FNC_COND = 60	(get the status of a Condition)
FNC_TPKEY = 52	(simulate a key press on an interface)

The Setting can have the following values

SET_ON = 255
SET_TOGGLE = 103
SET_OFF = 0

SET_DIM = 0 - 100	(The level to which you want to set the dimmer)
SET_MTRUP = 1	(To start a motor in the UP direction)
SET_MTRDOWN = 2	(To start a motor in the DOWN direction)
SET_MTRSTOP = 3	(To stop a motor)
SET_MTRSTARTSTOP = 6	(= PROSOFT motor start stop function)
SET_MTRUPSTOP = 7	(= PROSOFT motor up stop function)
SET_MTRDOWNSTOP = 8	(= PROSOFT motor start down function)
SET_MTRUPDOWN = 55	(= PROSOFT motor up down function)
SET_AUDIOUP = 32	(Control Audio Function: Vol. Up)
SET_AUDIODOWN = 33	(Control Audio Function: Vol. Down)
SET_AUDIOON = 36	(Control Audio Function: On)
SET_AUDIOOFF = 37	(Control Audio Function: Off)
SET_AUDIOFM = 38	(Control Audio Function: Tuner)
SET_AUDIOFM2 = 47	(Control Audio Function: Tuner Extra)
SET_AUDIOCD = 39	(Control Audio Function: CD)
SET_AUDIOCD2 = 48	(Control Audio Function: CD Extra)
SET_AUDIOTAPE = 40	(Control Audio Function: Tape)
SET_AUDIOTAPE2 = 49	(Control Audio Function: Tape Extra)
SET_AUDIOVIDEO = 41	(Control Audio Function: Video)
SET_AUDIOVIDEO2 = 43	(Control Audio Function: Video Extra)
SET_AUDIOAUX = 42	(Control Audio Function: Aux)
SET_AUDIOAUX2 = 35	(Control Audio Function: Aux Extra)
SET_AUDIOSRC6 =71	
SET_AUDIOSRC7 = 72	
SET_AUDIOSRC8 = 73	
SET_AUDIOSRC6_2 = 74	
SET_AUDIOSRC7_2 =75	
SET_AUDIOSRC8_2 = 76	
SET_AUDIOMUTE = 77	

SET_AUDIOKEY0 = 78	(Control Audio Function: Key)
SET_AUDIOKEY1 = 79	(Control Audio Function: Key)
SET_AUDIOKEY2 = 80	(Control Audio Function: Key)
SET_AUDIOKEY3 = 81	(Control Audio Function: Key)
SET_AUDIOKEY4 = 82	(Control Audio Function: Key)
SET_AUDIOKEY5 = 83	(Control Audio Function: Key)
SET_AUDIOKEY6 = 84	(Control Audio Function: Key)
SET_AUDIOKEY7 = 85	(Control Audio Function: Key)
SET_AUDIOKEY8 = 86	(Control Audio Function: Key)
SET_AUDIOKEY9 = 87	(Control Audio Function: Key)

SET_TEMPUP = 21	(Increase the Temperature target with 0.5°C)
SET_TEMPDOWN = 22	(Decrease the Temperature target with 0.5°C)

SET_TEMPFROST = 24	(To set the system to frost protection)
SET_TEMPDAY = 26	(Set the preset to DAY)
SET_TEMPNIGHT = 25	(Set the preset to NIGHT)
SET_TEMPSTANDBY = 93	(Set the preset to STANDBY)

SET_TEMPSETDAY = 29	(To set the Day preset temperature)
SET_TEMPSETSTANDBY = 88	(To set the Standby preset temperature)
SET_TEMPSETNIGHT = 27	(To set the Night preset temperature)

SET_TEMPSPEED = 31	(To scroll through the different speeds)
SET_TEMPSPLOW = 97	(Set speed to Low)
SET_TEMPSPMED = 98	(Set speed to Medium)
SET_TEMPSPHIGH = 99	(Set speed to High)
SET_TEMPSPAUTO = 89	(Set speed to Auto)

SET_TEMPMODE = 30	(To scroll through the different modes)
SET_TEMPAUTO = 94	(Set mode to Auto)
SET_TEMPHEAT = 95	(Set mode to Heat)
SET_TEMPCOOL = 96	(Set mode to Cool)
SET_TEMPVENT = 105	(Set mode to Vent, Airzone only)
SET_TEMPSTOP = 106	(Set mode to Stop, Airzone only)
SET_TEMPHEATP = 107	(Set mode to Heat+, Airzone only)

SET_TEMPONOFF = 104	(To toggle between ON and OFF)
SET_TEMPON = 108	
SET_TEMPOFF = 109	

SET_REGAUTO =0	(set Regime to AUTO)
SET_REGWORKD = 1	(set Regime to Workday)
SET_REGWKEND = 2	(set Regime to Weekend day)
SET_REGSIM = 3	(set Regime to Simulation)
SET_REGNONE =4	(set Regime to None)
SET_REGCUST = 5	(set custom list on or off)

SET_PULSE = 1	(For a "Short" press)
SET_CLOSED = 2	(For a long press)
SET_OPENED = 3	(For the end of a long press)

SENS_MAX = 0x3F00	(the maximum possible value for a sensor)

6 Communication

This section describes what you have to know if you write your own interface but you won’t use the DLL. If you use the DLL you don’t have to read this section.

The TDS CENTRAL unit is connected to a CCT (custom command terminal), which has to be operated by the user and can send commands to the TDS CENTRAL. The communication between the CCT and the TDS CENTRAL unit is a standard Ethernet socket on **port 55957**.

All commands and messages in both directions will use the same frame format:

STX (02h)	Length	Command Number	Parameter 1		Parameter n	ChkSm

The length does not include the ChkSm-byte.
The ChkSm is calculated on Command Number + Command Parameters + Length + STX
After the ChkSm the central unit sends an acknowledge byte 0A (hex). If no acknowledge byte is not send (from TDS to CCT) the command is not handled.

If there are parameter that can be higher than 255, two bytes are needed for this parameters. In this case the higher byte is sent first.

To get or set an output there is a central unit parameter to define to which central unit an output belongs (from 1 to 10). This parameter is only needed for the functions: FNC_RELAY, FNC_DIMMER FNC_MOTOR, FNC_SENSOR, FNC_AUDIO. For all other functions the central unit parameter needs to be 0

6.1 Function Set

- Description: This command allows the CCT to set individual functions.
- Direction: From TDS to CCT.
- Length: ? (depending on the number of parameters)
- Command number: 07h
- Parameter 1 = Central Unit
- Parameter 2 = Fnc
- Parameter 3 & 4 = Number (2bytes)
- Parameter 5 = Setting
- Parameter 6 - ... = Data (optional)

Details: see DLL functions

6.2 Function Get

- Description: When the TDS receives this command it reports the state of the specified device.
- Direction: From CCT to TDS
- Length: 7
- Command number: 06h
- Parameter 1 = Central unit
- Parameter 2 = Fnc
- Parameter 3 & 4 = Number (2bytes)

6.3 Function Group Get

- Description: An extension of the Function Get. With this command it is possible to ask for the status of multiple devices of the same type at once
- Direction: From CCT to TDS
- Lenght: ? (depending on the number of outputs)
- Command number: 09h
- Parameter 1 = Central unit
- Parameter 2 = Fnc
- Parameter 3 & 4, ... = Numbers (2bytes)

6.4 Function Log On/Off

- Description: When the TDS receives this command it (de-)activates it’s channel for reporting the function! Remark: If you have multiple connections to the TDS, the TDS keeps a list of functions which need to be reported for each connection.
- Direction: From CCT to TDS
- Length: 5
- Command number: 03h
- Direction: From CCT to TDS
- Parameter 1 = Fnc
- Parameter 2 = state

6.5 Event Report

- Description: TDS sends this command it the level of the specified load has changed (if the Function Log for this function on this connection has been set) , or as an answer to a Function Get (always)
- Direction: From TDS to CCT
- Length: ?
- Command number: 10h
- Parameter 1 = Central Unit
- Parameter 2 = Fnc
- Parameter 3 & 4 = Number (2bytes)
- Parameter 5 = ErrorState (Not Used)
- Parameter 6 - ... = State(s)

Details: See DLL functions

6.6 Write Display Message

- Description: This Command can be used to create a "Dynamic Message"
- Direction: From CCT to TDS
- Length: ?
- Command number: 04h
- Parameter 1 = Central Unit
- Parameter 2 to n = Bus numbers
- Parameter n+1 to 2*n -1= Address numbers
- Parameter 2*n = Message (1 for message, 0 for alarm)
- Parameter 2*n + 1 = Ascii (1for ascii, 0 for Unicode)
- Parameter 2*n+2 to 2*n +17 = Line 1
- Parameter 2*n+18 to 2*n + 33 = Line 2
- Parameter 2*n + 34 = Beeps

6.7 Send keep alive

- Description: This command can be send to keep the socket open.
- Direction: From CCT to TDS
- Length = 3
- Command number: 0xBh

6.8 Sensor values

All sensor values communicated with the central unit are in a two byte(short) format, to convert between the actual states and this two byte format, use the following conversions (Also note that values >= SENS_MAX indicate an error with the sensor.

6.8.1 Temperatures

To change from short to °C: short/10 -273
To change from °C to short: °C+273 *10

Note: this formula applies for actual states, for temperature differences (the STANDBY value) you only need to multiply or divide by 10 (e.g. ± 3.5°C = 35)

6.8.2 Humidity

Byte = % humidity

6.8.3 Lux values

To change from byte to lux = $(10^{(\text{byte} / 40)}) - 1$
To change from lux to byte = Log10(lux + 1) * 40

6.8.4 General Analog Sensors

With Smax en Smin being the minimum and maximum values for “value”, the value can be calculated from the short as:

- 4-20mA:
  Value = (((Smax – Smin) / 720) x (short - 180)) + Smin
- 0-20mA:
  Value = (((Smax – Smin) / 900) x value) + Smin
- 0-10V and 5-10V
  Value = (((Smax - Smin) / 1023) x value) + Smin