ioBoard C# Dll

Generated by Doxygen 1.7.4

Sun Nov 13 2011 08:47:51

Contents

1 Namespace Documentation

1.1 Package myIoBoard3

The DLL is an C# ioBoard interface (Controller).

Classes

- struct Control_params

  PID Regulator parameters used in boards.
- struct MyControlParams

  PID Regulator parameters used in application program.
- struct ControllerState

  PID regulator state.
- struct RcvStatus

  Board status.
- struct Hardware

  Board hardware features.
- struct EventData

  Event data used for analog input (boards Ai4Ao1, Ai4Ao2)
- class ioBoard

  The class defines a set of functions that allows a program (acting as a tcp/ip client) to communicate with the ioBoard.net boards. At the moment (October 2011) the DLL has been tested with the following boards: BasEth, BasEth2, Ai4Ao1, Ai4Ao2, Di6, Di4Do2.

Enumerations

- enum ioBoardType {

  Unknown, Ai4Ao1, Ai4Ao2, Di6,

  Di4Do2 }

  Represents possible board types.
- enum ioBoardResult {

  NotOK, OK, Pending, ValueTooLarge,

  ValueTooSmall, BoardNumberTooLarge, BoardTypeWrong, InputNumberTooLarge,

  OutputNumberTooLarge, UnknownState }

  Represents the request result.

1.1.1 Detailed Description

The DLL is an C# ioBoard interface (Controller).

1.1.2 Enumeration Type Documentation

1.1.2.1 enum myIoBoard3::ioBoardResult

Represents the request result.

Enumerator:

NotOK request failed

OK request succeeded

Pending still data to receive

ValueTooLarge the required value is too large: it has been set to a default max value 20mA

ValueTooSmall the required value is too small: it has been set to a default min value 4mA

BoardNumberTooLarge the board number is too large

BoardTypeWrong unknow board type

InputNumberTooLarge the input number is too large

OutputNumberTooLarge the output number is too large

UnknownState unknown Controller state

1.1.2.2 enum myIoBoard3::ioBoardType

Represents possible board types.

Enumerator:

Unknown unknown type

Ai4Ao1 board response: "AI4 AO1 V01"

Ai4Ao2 board response: "AI4 AO2 V01"

Di6 board response: "DI6 V01"

Di4Do2 board response: "DI4DI2 V01"

2 Class Documentation

2.1 myIoBoard3.Control_params Struct Reference

PID Regulator parameters used in boards.

Properties

- ushort Reference `[get, set]`

  Desired (set) input value.

- ushort Kp [get, set]
  Proportional gain.
- ushort Ki [get, set]
  Integral gain.
- ushort Kd [get, set]
  Derivative gain.
- ushort InZero [get, set]
  Input offset; typically=0x4800 (12mA)

2.1.1 Detailed Description

PID Regulator parameters used in boards.

2.1.2 Property Documentation

2.1.2.1 ushort myIoBoard3.Control_params.InZero [get, set]

Input offset; typically=0x4800 (12mA)

2.1.2.2 ushort myIoBoard3.Control_params.Kd [get, set]

Derivative gain.

2.1.2.3 ushort myIoBoard3.Control_params.Ki [get, set]

Integral gain.

2.1.2.4 ushort myIoBoard3.Control_params.Kp [get, set]

Proportional gain.

2.1.2.5 ushort myIoBoard3.Control_params.Reference [get, set]

Desired (set) input value.

2.2 myIoBoard3.ControllerState Struct Reference

PID regulator state.

Properties

- byte OutCh [get, set]
  The number of the used output.
- byte State [get, set]
  Regulator state: get: Disabled(0), Enabled(1) set: Disable(0), Enable(1), Reset(2)
- byte InCh [get, set]
  The number of the used input.

2.2.1 Detailed Description

PID regulator state.

2.2.2 Property Documentation

2.2.2.1 byte myIoBoard3.ControllerState.InCh [get, set]

The number of the used input.

2.2.2.2 byte myIoBoard3.ControllerState.OutCh [get, set]

The number of the used output.

2.2.2.3 byte myIoBoard3.ControllerState.State [get, set]

Regulator state: get: Disabled(0), Enabled(1) set: Disable(0), Enable(1), Reset(2)

2.3 myIoBoard3.EventData Struct Reference

Event data used for analog input (boards Ai4Ao1, Ai4Ao2)

Properties

- byte Num [get, set]
  Channel number: 0, 1, 2, 3.
- double Val [get, set]
  Input value.

2.3.1 Detailed Description

Event data used for analog input (boards Ai4Ao1, Ai4Ao2)

2.3.2 Property Documentation

2.3.2.1 byte myIoBoard3.EventData.Num [get, set]

Channel number: 0, 1, 2, 3.

2.3.2.2 double myIoBoard3.EventData.Val [get, set]

Input value.

2.4 myIoBoard3.Hardware Struct Reference

Board hardware features.

Properties

- byte HwReserved [get, set]
  reserved
- ushort HwTemp [get, set]
  Board temperature (not implemented yet)
- ushort HwVcc [get, set]
  Power supply 3.3V (not implemented yet)
- ushort HwTemp30 [get, set]
  Board ??? (not implemented yet)
- ushort HwTemp85 [get, set]
  Board ??? (not implemented yet)
- ushort HwTick [get, set]
  Count of the board tick (50Hz) modulo 65536.

2.4.1 Detailed Description

Board hardware features.

2.4.2 Property Documentation

2.4.2.1 byte myIoBoard3.Hardware.HwReserved [get, set]

reserved

2.4.2.2 ushort myIoBoard3.Hardware.HwTemp [get, set]

Board temperature (not implemented yet)

2.4.2.3 ushort myIoBoard3.Hardware.HwTemp30 [get, set]

Board ??? (not implemented yet)

2.4.2.4 ushort myIoBoard3.Hardware.HwTemp85 [get, set]

Board ??? (not implemented yet)

2.4.2.5 ushort myIoBoard3.Hardware.HwTick [get, set]

Count of the board tick (50Hz) modulo 65536.

2.4.2.6 ushort myIoBoard3.Hardware.HwVcc [get, set]

Power supply 3.3V (not implemented yet)

2.5 myIoBoard3.ioBoard Class Reference

The class defines a set of functions that allows a program (acting as a tcp/ip client) to communicate with the ioBoard.net boards. At the moment (October 2011) the DLL has been tested with the following boards: BasEth, BasEth2, Ai4Ao1, Ai4Ao2, Di6, Di4Do2.

Public Member Functions

- bool Connect (string ip)

  Connects the program to the boards using two tcp/ip channels: Basic and Control. Both channels use the same IP address but different ports. Ports has the values: 58001 (Basic) and 58003 (Control) and cannot be changed. The Basic channel is for communication with the Ethernet interface (Basic board) The Control channel is for communication with I/O interfaces (Control boards).
- void Disconnect ()

  Disconnects the program from the boards: both (Basic and Control) channels are disconnected.
- bool GetMACAddress (ref string rcv)

  Reads the MAC address of the Basic interface board.
- bool GetVersion (ref string rcv)

  Reads the version of the Basic interface board.
- bool SetIPAddress (string snd)

  Sets a new IP Address. The new Address is valid after restart of the board.
- bool SetMACAddress (string snd)

  Only for internal manufacturer's use.
- string EnableLog (bool enable)

  If enable is true creates a log file with the name logRandomNumer.txt where the RandomNumber is between 0 and 1000, otherwise it closes the log file if it exists. Enables / disables logging of all messages passed between a program and boards.
- int Init ()

  Initializes the I/O boards.
- ioBoardType GetBoardType (byte boardId)

  Gets a board type.
- ioBoardResult GetStatus (byte boardId, ref string st, ref RcvStatus status)

  Gets a board status.
- ioBoardResult GetHardware (byte boardId, ref string st, ref Hardware hardware)

  Gets a board hardware data.
- ioBoardResult GetBoardVersion (byte boardId, ref string st)

  Gets the board version.
- ioBoardResult GetInputs (byte boardId, ref double[] anaVal, ref bool[] digVal)

  Gets board inputs, used for both (analog and digital) boards. Depending on the board type called, the input values are either as anaVal or digVal.
- ioBoardResult SetDigitalOutput (byte boardId, bool[] outMask, bool[] outValue)

  Sets digital outputs.

- ioBoardResult SetAnalogCurrentOutput (byte boardId, byte outNum, double anaValue)

  Gets analog (current) outputs.
- ioBoardResult GetAnalogCurrentOutput (byte boardId, byte outNum, ref double anaValue)

  Gets analog (current) output value.
- ioBoardResult SetEventMask (byte boardId, ioBoardType boardType, bool[] digInput)

  Sets inputs for event generation.
- ioBoardResult GetEventMask (byte boardId, ref bool[] digInput)

  Gets inputs generating events.
- ioBoardResult SetControllerState (byte boardId, byte inNum, byte outNum, byte state)

  Sets PID regulator state.
- ioBoardResult GetControllerState (byte boardId, byte outNum, ref string st, ref ControllerState state)

  Gets PID regulator state.
- ioBoardResult SetControllerParameters (byte boardId, byte outNum, double reference, ushort kp, ushort ki, ushort kd, double in_zero)

  Sets the PID regulator parameters.
- ioBoardResult GetControllerParameters (byte boardId, byte outNum, ref string st, ref MyControlParams par)

  Sends a get PID regulator parameters request.
- ioBoardResult GetEvents (ref byte board, ref EventData[] anaInput, ref bool[] digInput)

  Reads events from all I/O boards (broadcast) The events are organized in groups of bytes: the first byte in the group is the eventId, the following bytes are the events (their number depends on eventId). For analog boards Ai4Ao1 and Ai4Ao2 the eventId = 0, 1, 2, 3 are analog inputs with a 3-byte event. For digital boards Di6 and Di4Do2 the eventId = 0 is a byte with all digital inputs (the MSB is Di0).

Properties

- ioBoardType[] BoardType `[get, set]`

  Represents the board type for a given board number.
- bool Connected `[get]`

  Represents the connection status of the Ethernet interface board returning true if the board is connected to the computer, otherwise false.
- bool Initialized `[get]`

  Represents the initialisation status of the I/O boards returning true if the boards are initialized, otherwise false.
- int LastId `[get]`

  Represents the number of the last board.
- bool Log `[get]`

  Represents the status of logging returning true if log is enabled, otherwise false.

2.5.1 Detailed Description

The class defines a set of functions that allows a program (acting as a tcp/ip client) to communicate with the ioBoard.net boards. At the moment (October 2011) the DLL has been tested with the following boards: BasEth, BasEth2, Ai4Ao1, Ai4Ao2, Di6, Di4Do2.

2.5.2 Member Function Documentation

2.5.2.1 bool myIoBoard3.ioBoard.Connect (string *ip*)

Connects the program to the boards using two tcp/ip channels: Basic and Control. Both channels use the same IP address but different ports. Ports has the values: 58001 (Basic) and 58003 (Control) and cannot be changed. The Basic channel is for communication with the Ethernet interface (Basic board) The Control channel is for communication with I/O interfaces (Control boards).

Parameters

ip	The default IP address of the board is 169.254.19.63. The IP address can be changed using the function SetIPAddress.

Returns

true if connected successfully to both channels, otherwise false.

2.5.2.2 void myIoBoard3.ioBoard.Disconnect ( )

Disconnects the program from the boards: both (Basic and Control) channels are disconnected.

2.5.2.3 string myIoBoard3.ioBoard.EnableLog (bool *enable*)

If enable is true creates a log file with the name logRandomNumer.txt where the RandomNumber is between 0 and 1000, otherwise it closes the log file if it exists. Enables / disables logging of all messages passed between a program and boards.

Parameters

enable	true value enables logging, false disables

Returns

the log file name if file operation has been succesfull, empty string otherwise

2.5.2.4 ioBoardResult myIoBoard3.ioBoard.GetAnalogCurrentOutput (byte *boardId,* byte *outNum,* ref double *anaValue*)

Gets analog (current) output value.

Parameters

boardId	number 0..lastId
outNum	0 (for Ai4Ao1), 0..1 (for Ai4Ao2)
anaValue	current value (double) in mA

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.5 ioBoardType myIoBoard3.ioBoard.GetBoardType (byte *boardId*)

Gets a board type.

Parameters

boardId	number 0..lastId

Returns

result as ioBoardType value

2.5.2.6 ioBoardResult myIoBoard3.ioBoard.GetBoardVersion (byte *boardId,* ref string *st*)

Gets the board version.

Parameters

boardId	number 0..lastId
st	board version as a string

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.7 ioBoardResult myIoBoard3.ioBoard.GetControllerParameters (byte *boardId,* byte *outNum,* ref string *st,* ref MyControlParams *par*)

Sends a get PID regulator parameters request.

Parameters

boardId	number
outNum	regulator output number
st	regulator parameters as a string
par	regulator parameters as a MyControllerParams structure

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.8 ioBoardResult myIoBoard3.ioBoard.GetControllerState (byte *boardId,* byte *outNum,* ref string *st,* ref ControllerState *state*)

Gets PID regulator state.

Parameters

boardId	number 0..lastId
outNum	regulator output number: 0..3
st	regulator state as a string
state	regulator state as a structure ControllerState

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.9 ioBoardResult myIoBoard3.ioBoard.GetEventMask (byte *boardId,* ref bool[ ] *digInput*)

Gets inputs generating events.

Parameters

boardId	number 0..lastId
digInput	array of input values (bool)

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.10 ioBoardResult myIoBoard3.ioBoard.GetEvents (ref byte *board,* ref EventData[ ] *anaInput,* ref bool[ ] *digInput*)

Reads events from all I/O boards (broadcast) The events are organized in groups of bytes: the first byte in the group is the eventId, the following bytes are the events (their number depends on eventId). For analog boards Ai4Ao1 and Ai4Ao2 the eventId = 0, 1, 2, 3 are analog inputs with a 3-byte event. For digital boards Di6 and Di4Do2 the eventId = 0 is a byte with all digital inputs (the MSB is Di0).

Parameters

board	number 0..lastId
anaInput	array of input data (EventData)
digInput	array of input values (bool)

Returns

number of bytes sent

2.5.2.11 ioBoardResult myIoBoard3.ioBoard.GetHardware (byte *boardId,* ref string *st,* ref Hardware *hardware*)

Gets a board hardware data.

Parameters

boardId	number 0..lastId
st	board hardware info as a string
hardware	board hardware info as a Hardware structure

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.12 ioBoardResult myIoBoard3.ioBoard.GetInputs (byte *boardId,* ref double[ ] *anaVal,* ref bool[ ] *digVal*)

Gets board inputs, used for both (analog and digital) boards. Depending on the board type called, the input values are either as anaVal or digVal.

Parameters

boardId	number 0..lastId
anaVal	array of input values (double)
digVal	array of input values (bool)

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.13 bool myIoBoard3.ioBoard.GetMACAddress (ref string *rcv*)

Reads the MAC address of the Basic interface board.

Parameters

rcv	MAC address

Returns

true if the MAC address has been read, otherwise false

checks whether the boards accepted the command

2.5.2.14 ioBoardResult myIoBoard3.ioBoard.GetStatus (byte *boardId,* ref string *st,* ref RcvStatus *status*)

Gets a board status.

Parameters

boardId	number 0..lastId

st	board status as a string
status	board status as a RcvStatus structure

Returns

result as ioBoardResult value; if the result is different than OK the request failed

2.5.2.15 bool myIoBoard3.ioBoard.GetVersion (ref string *rcv*)

Reads the version of the Basic interface board.

Parameters

rcv	version

Returns

true if the version has been read, otherwise false

checks whether the boards accepted the command

2.5.2.16 int myIoBoard3.ioBoard.Init ( )

Initializes the I/O boards.

Returns

last board id (the boards are counted sequentially from 0). If initialisation fails a negative value is returned.

2.5.2.17 ioBoardResult myIoBoard3.ioBoard.SetAnalogCurrentOutput (byte *boardId,* byte *outNum,* double *anaValue*)

Gets analog (current) outputs.

Parameters

boardId	number 0..lastId
outNum	0 (Ai4Ao1 board) or 0, 1 (Ai4Ao2 board)
anaValue	current value (double) in mA

Returns

result as ioBoardResult value; if the result is different than OK the output has been not set

checks the low (4mA) and high (20mA) limits

2.5.2.18 ioBoardResult myIoBoard3.ioBoard.SetControllerParameters (byte *boardId,* byte *outNum,* double *reference,* ushort *kp,* ushort *ki,* ushort *kd,* double *in_zero*)

Sets the PID regulator parameters.

Parameters

boardId	number 0..lastId
outNum	regulator output number: 0 (for Ai4Ao1), 0..1 (for Ai4Ao2)
reference	regulator set value: current 4..20mA
kp	regulator Kp value (unsigned integer)
ki	regulator Ki value (unsigned integer)
kd	regulator Kd value (unsigned integer)
in_zero	regulator input zero: 4..20mA (typical values: 0 and 12mA)

Returns

result as ioBoardResult value; if the result is different than OK the regulator parameters have been not set

2.5.2.19 ioBoardResult myIoBoard3.ioBoard.SetControllerState (byte *boardId,* byte *inNum,* byte *outNum,* byte *state*)

Sets PID regulator state.

Parameters

boardId	number 0..lastId
inNum	regulator input number: 0..3
outNum	regulator output number: 0 (for Ai4Ao1), 0..1 (for Ai4Ao2)
state	0 - Disable, 1 - Enable, 2 - Reset

Returns

result as ioBoardResult value; if the result is different than OK the regulator state has been not set

2.5.2.20 ioBoardResult myIoBoard3.ioBoard.SetDigitalOutput (byte *boardId,* bool[ ] *outMask,* bool[ ] *outValue*)

Sets digital outputs.

Parameters

boardId	number 0..lastId
outMask	an array of boolean values masking the digital outputs: if mask=true the digital output will be set, otherwise not
outValue	an array of digital output values (boolean)

Returns

result as ioBoardResult value; if the result is different than OK the output has been

not set

2.5.2.21 ioBoardResult myIoBoard3.ioBoard.SetEventMask (byte *boardId,* ioBoardType *boardType,* bool[] *digInput*)

Sets inputs for event generation.

Parameters

boardId	number 0..lastId
boardType	board type
digInput	an array of boolean values defining the inputs: if a bit=true the corresponding input will generate event

Returns

result as ioBoardResult value; if the result is different than OK the mask has been not set

2.5.2.22 bool myIoBoard3.ioBoard.SetIPAddress (string *snd*)

Sets a new IP Address. The new Address is valid after restart of the board.

Parameters

snd	new IP Address

Returns

true if IP Address has been accepted by the board, otherwise false

checks whether the address has 3 dots

checks whether the boards accepted the command

2.5.2.23 bool myIoBoard3.ioBoard.SetMACAddress (string *snd*)

Only for internal manufacturer's use.

Parameters

snd	new MAC address

Returns

true if MAC Address has been accepted by the board, otherwise false

checks whether the boards accepted the command

2.5.3 Property Documentation

2.5.3.1 ioBoardType [] myIoBoard3.ioBoard.BoardType [get, set]

Represents the board type for a given board number.

2.5.3.2 bool myIoBoard3.ioBoard.Connected [get]

Represents the connection status of the Ethernet interface board returning true if the board is connected to the computer, otherwise false.

2.5.3.3 bool myIoBoard3.ioBoard.Initialized [get]

Represents the initialisation status of the I/O boards returning true if the boards are initialized, otherwise false.

2.5.3.4 int myIoBoard3.ioBoard.LastId [get]

Represents the number of the last board.

2.5.3.5 bool myIoBoard3.ioBoard.Log [get]

Represents the status of logging returning true if log is enabled, otherwise false.

2.6 myIoBoard3.MyControlParams Struct Reference

PID Regulator parameters used in application program.

Properties

- byte OutNum [get, set]
  Number of the used output.
- double Reference [get, set]
  Desired (set) input value.
- ushort Kp [get, set]
  Proportional gain.
- ushort Ki [get, set]
  Integral gain.
- ushort Kd [get, set]
  derivative gain
- double InZero [get, set]
  Input offset; typically=0x4800 (12mA)

2.6.1 Detailed Description

PID Regulator parameters used in application program.

2.6.2 Property Documentation

2.6.2.1 double myIoBoard3.MyControlParams.InZero [get, set]

Input offset; typically=0x4800 (12mA)

2.6.2.2 ushort myIoBoard3.MyControlParams.Kd [get, set]

derivative gain

2.6.2.3 ushort myIoBoard3.MyControlParams.Ki [get, set]

Integral gain.

2.6.2.4 ushort myIoBoard3.MyControlParams.Kp [get, set]

Proportional gain.

2.6.2.5 byte myIoBoard3.MyControlParams.OutNum [get, set]

Number of the used output.

2.6.2.6 double myIoBoard3.MyControlParams.Reference [get, set]

Desired (set) input value.

2.7 myIoBoard3.RcvStatus Struct Reference

Board status.

Properties

- byte Flags [get, set]
  For the last board flags = 1, otherwise flag = 0.
- ushort OverFlow [get, set]
  Count of overflows.
- ushort ChkErr [get, set]
  Count of check sum errors.
- ushort StartErr [get, set]
  Count of start error (first byte wrong)
- ushort RecFrames [get, set]
  Count of received frames.
- ushort Chksum [get, set]
  Ckeck sum.

2.7.1 Detailed Description

Board status.

2.7.2 Property Documentation

2.7.2.1 ushort myIoBoard3.RcvStatus.ChkErr [get, set]

Count of check sum errors.

2.7.2.2 ushort myIoBoard3.RcvStatus.Chksum [get, set]

Ckeck sum.

2.7.2.3 byte myIoBoard3.RcvStatus.Flags [get, set]

For the last board flags = 1, otherwise flag = 0.

2.7.2.4 ushort myIoBoard3.RcvStatus.OverFlow [get, set]

Count of overflows.

2.7.2.5 ushort myIoBoard3.RcvStatus.RecFrames [get, set]

Count of received frames.

2.7.2.6 ushort myIoBoard3.RcvStatus.StartErr [get, set]

Count of start error (first byte wrong)

Index